

# Simple Example for Using ALV in Web Dynpro for ABAP



**Release SAP NetWeaver 2004s**



## 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.
<b>Example text</b>	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.
<b>Example text</b>	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.

## Table of Contents

Copyright.....	2
Icons in Body Text .....	3
Typographic Conventions.....	3
Table of Contents .....	4
Task .....	5
Objectives.....	5
 <b>Creating a New Web Dynpro Component and Building Up the Component</b>	
<b>Controller Context.....</b>	<b>6</b>
Procedure.....	6
 <b>Search View .....</b>	<b>8</b>
Procedure.....	8
 <b>Layout View.....</b>	<b>9</b>
Procedure.....	10
 <b>Using the Component ALV Table in the Component Controller .....</b>	<b>11</b>
Procedure.....	11
 <b>Create Web Dynpro Application and Test .....</b>	<b>13</b>
Procedure.....	13
<b><i>Author Bio .....</i></b>	<b><i>14</i></b>



## Using an ALV Table in Web Dynpro

In this tutorial an ALV table is used to display the content of a context node.

### Task

The task of this simple tutorial is to use an ALV table inside a window to display the content of context node *NODE\_FLIGHTTAB*.

### Objectives

By the end of this tutorial, you will be able to:

- ✓ Use an ALV inside a window

### Knowledge

- Knowledge of ABAP OO programming language
- Basic knowledge of programming Web Dynpro applications
- Basic knowledge of ABAP workbench



## Creating a New Web Dynpro Component and Building up the Component Controller Context

This tutorial starts with the creation of a new Web Dynpro component called **Z00\_WDT\_FLIGHTLIST\_SIMPLE**.

Afterwards the component controller context is built up.

### Procedure

#### Create a new Web Dynpro component.

Start the ABAP Workbench (*se80*) and create the new Web Dynpro component **Z00\_WDT\_FLIGHTLIST\_SIMPLE**. Use *MAIN* as window name. And delete the view name.

#### Create context element in component controller for storing the search criteria.

Create node **NODE\_FLIGHT** based on dictionary structure **SFLIGHT** in the context of the component controller. Click on button *Add Attribute from Structure* and choose fields **CARRID** and **CONNID**.

The screenshot shows the 'Create Nodes' dialog box with the following fields:

- Node Name: node\_flight
- Interface Node: No
- Input Element (Ext.): No
- Dictionary structure: sflight
- Cardinality: 1..1
- Selection: 0..1
- Init. Lead Selection: Yes
- Singleton: Yes
- Supply Function: (empty)

The 'Add Attribute from Structure' button is highlighted with a green checkmark.

The screenshot shows the 'Select Components of Structure SFLIGHT' dialog box with the following table:

Component	Key	RT	Component Type	Data Type	Length	Deci.	Sh
MANDI	<input checked="" type="checkbox"/>	<input type="checkbox"/>	S_MANDI	CLNT	3	0	0 Cli
CARRID	<input checked="" type="checkbox"/>	<input type="checkbox"/>	S_CARR_ID	CHAR	3	0	0 Air
CONNID	<input checked="" type="checkbox"/>	<input type="checkbox"/>	S_CONN_ID	NUMC	4	0	0 Fil
FLDATE	<input checked="" type="checkbox"/>	<input type="checkbox"/>	S_DATE	DATS	8	0	0 Fil
PRICE	<input type="checkbox"/>	<input type="checkbox"/>	S_PRICE	CURR	15	2	2 Air

#### Create a context element in component controller for storing the flight list.

Create node **NODE\_FLIGHTTAB** based on dictionary structure **SFLIGHT** in the context of the component controller. Set the cardinality to 0..N. Click on button *Add Attribute from Structure* and choose fields **CARRID**, **CONNID**, **FLDATE**, **PRICE**, **CURRENCY**, **PLANETYPE**, **SEATSMAX**, **SEATSOCC** and **PAYMENTSUM**.

Component	Key RT	Component Type	Data Type	Length	Deci...	Sh
MANDI	<input checked="" type="checkbox"/>	S_MANDI	CLNT	3	0	Cl
CARRID	<input checked="" type="checkbox"/>	S_CARR_ID	CHAR	3	0	Air
CONNID	<input checked="" type="checkbox"/>	S_CONN_ID	NUMC	4	0	Fi
FLDATE	<input checked="" type="checkbox"/>	S_DATE	DATS	8	0	Fi
PRICE	<input type="checkbox"/>	S_PRICE	CURR	15	2	Air
CURRENCY	<input type="checkbox"/>	S_CURRCODE	CUKY	5	0	Lo
PLANETYPE	<input type="checkbox"/>	S_PLANETYE	CHAR	10	0	Air
SEATSMAX	<input type="checkbox"/>	S_SEATSMAX	INT4	10	0	Me
SEATSOCC	<input type="checkbox"/>	S_SEATSOCC	INT4	10	0	Oc
PAYMENTSUM	<input type="checkbox"/>	S_SUM	CURR	17	2	To
SEATSMAX_B	<input type="checkbox"/>	S_SMAX_B	INT4	10	0	Me
SEATSOCC_B	<input type="checkbox"/>	S_SOCC_B	INT4	10	0	Oc

An ALV table always displays all fields that are available in the dictionary, if the context node has a dictionary reference. To only display the selected fields inside the ALV table clear the dictionary reference in property *DICTIONARY STRUCTURE* in node *NODE\_FLIGHTTAB*.

Property	Value
<b>Nodes</b>	
Node Name	NODE_FLIGHTTAB
Interface Node	<input type="checkbox"/>
Input Element (Ext.)	<input type="checkbox"/>
Dictionary structure	
Cardinality	0..n
Selection	0..1
Initialization Lead Selection	<input checked="" type="checkbox"/>
Singleton	<input checked="" type="checkbox"/>

### Create a method for filling the flight list.

Create method `FILL_FLIGHTTAB` in the component controller to fill node `NODE_FLIGHTTAB`. (This method will be called later from out of the `SEARCHVIEW`.)



It is important that the table data of context node `NODE_FLIGHTTAB` is not copied. In order to ensure this, the variable holding the table (`lt_flights`) that is handed over to the context node via `BIND_TABLE` must be a local variable.

```
FILL_FLIGHTTAB ( )
```

```

method FILL_FLIGHTTAB .
* fill context node "node_flighttab"

DATA:
  node_node_flight      TYPE REF TO if_wd_context_node,
  node_node_flighttab   TYPE REF TO if_wd_context_node,
  elem_node_flight      TYPE REF TO if_wd_context_element,
  stru_node_flight      TYPE if_componentcontroller=>element_node_flight,

  ls_where(72)         TYPE c,
  lt_where              LIKE TABLE OF ls_where,

  lt_flights           TYPE TABLE OF sflight.

* navigate from <CONTEXT> to <NODE_FLIGHT> via lead selection
node_node_flight = wd_context->get_child_node( name = `NODE_FLIGHT` ).

* get element via lead selection
elem_node_flight = node_node_flight->get_element( ).

* get all declared attributes
elem_node_flight->get_static_attributes(
  IMPORTING
    static_attributes = stru_node_flight ).

* create where condition
IF NOT stru_node_flight-carrid EQ ''.
  CONCATENATE 'CARRID = '' stru_node_flight-carrid '' INTO ls_where.
  APPEND ls_where TO lt_where.
ENDIF.

IF NOT stru_node_flight-connid EQ '0000'.
  CONCATENATE 'CONNID = '' stru_node_flight-connid '' INTO ls_where.
  IF stru_node_flight-carrid NE ''.
    CONCATENATE 'AND' ls_where INTO ls_where separated by space.
  ENDIF.
  APPEND ls_where TO lt_where.
ENDIF.

* read data
select * from sflight into table lt_flights
  WHERE (lt_where).

* navigate from <CONTEXT> to <NODE_FLIGHTTAB> via lead selection
node_node_flighttab = wd_context-
>get_child_node( name = `NODE_FLIGHTTAB` ).

* fill context node
node_node_flighttab->bind_table( lt_flights ).

endmethod.

```



## Search View

The view *SEARCHVIEW* will be used to enter the search criteria for the database selection that fills context node *NODE\_FLIGHTTAB*. The search criteria will be stored inside context node *NODE\_FLIGHT*.

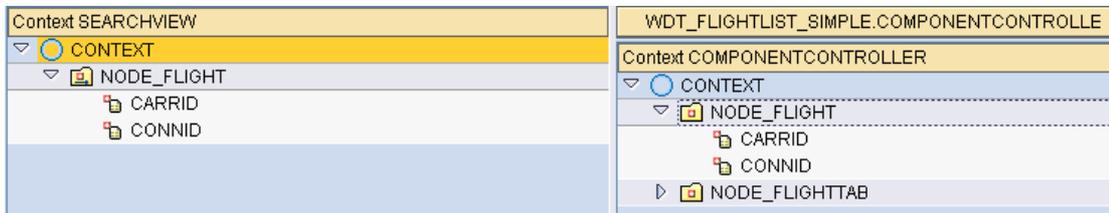
## Procedure

### Create view for searching.

Create view *SEARCHVIEW*.

## Map context element for storing search criteria.

Copy and map context node *NODE\_FLIGHT* from the component controller's context to the context of view *SEARCHVIEW*.

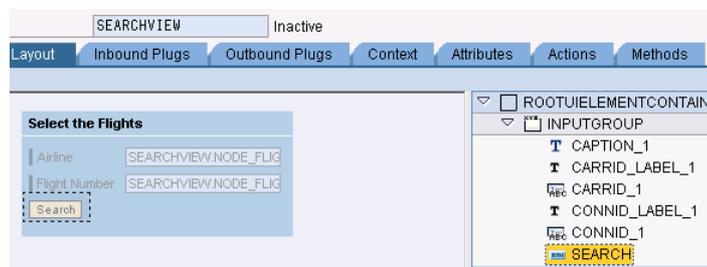


## Create the layout of the search view

In the layout of view *SEARCHVIEW* create a group called *INPUTGROUP*. Enter "Select the Flights" in the *TEXT* property of *CAPTION\_1* of group *INPUTGROUP*.

Inside *INPUTGROUP* create labels and input fields for the two attributes *CARRID* and *CONNID* of context node *NODE\_FLIGHT*. Hint: Use function *Create Container Form* out of the context menu of *INPUTGROUP*.

Create button *SEARCH* inside group *INPUTGROUP*. Enter "Search" as text. Create action *SEARCH* and invoke component controller method *FILL\_FLIGHTTAB()* in the corresponding method *ONACTIONSEARCH*.



### ONACTIONSEARCH ( )

```
method ONACTIONSEARCH .
    wd_Comp_Controller->Fill_Flighttab( ).
endmethod.
```



## Layout View

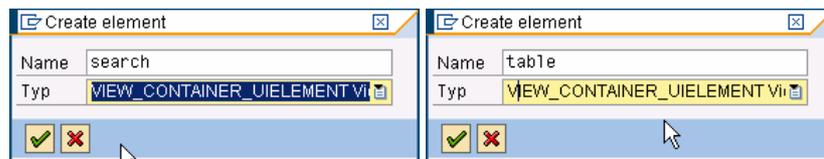
To display more than one view at the same time inside a window, we need a view for creating the layout.

## Procedure

### Create Layout View.

Create view LAYOUTVIEW and navigate to the layout tab. Change the *Layout* property of the ROOTUIELEMENTCONTAINER to MatrixLayout.

Create two new elements, VIEW\_CONTAINER\_UIELEMENT, each as a child of the ROOTUIELEMENTCONTAINER. Name them SEARCH and TABLE.

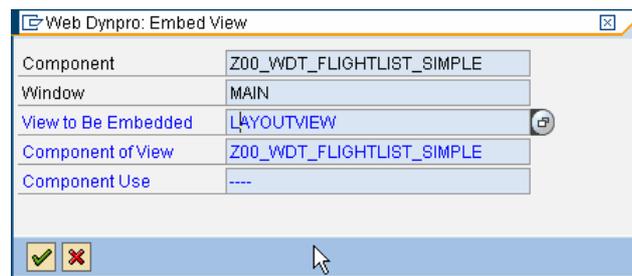


Change the *Layout Data* property of both view container UI elements to *MatrixHeadData*.

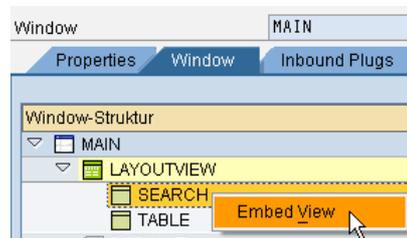
### Assign LAYOUTVIEW and SEARCHVIEW to window.

It's not possible to display a view directly inside a browser, but a window. Therefore the views just created have to be assigned to the window *MAIN*.

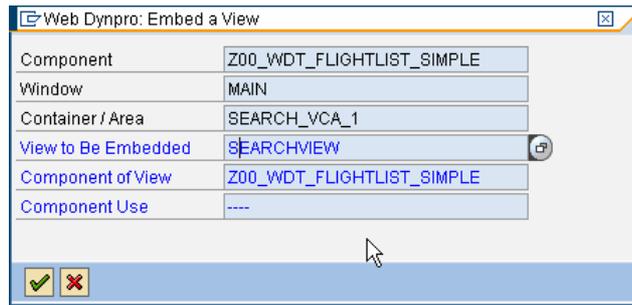
Open the window *MAIN* and navigate to the window tab. Open the context menu of the root element *MAIN* and choose *Embed View*. First embed view *LAYOUTVIEW*:



After view *LAYOUTVIEW* has been embedded into window *MAIN*, open the context menu of the view container *SEARCH* and choose *Embed View*.



Embed view *SEARCHVIEW*:



## Using the Component ALV Table in the Component Controller

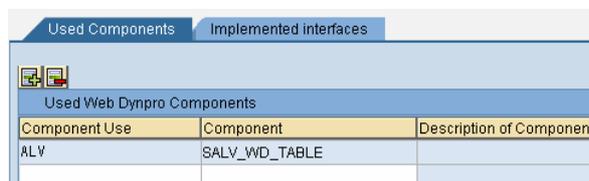
When you search for special flights by entering a carrier and a connection, you want to see the result in an ALV table.

### Procedure

#### Define component usage.

If you want to see the data within a Web Dynpro ALV, you have to define the Web Dynpro component for ALV **SALV\_WD\_TABLE** as a usage component of your Web Dynpro component.

Double-click on your component and define the component usage of SALV\_WD\_TABLE as follows:

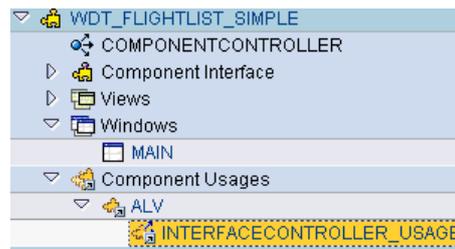


#### Set data to ALV for display (via reverse context mapping).

The selected flights are inside context node *NODE\_FLIGHTTAB*. To display them in the ALV it is necessary to map the context node *NODE\_FLIGHTTAB* to the context node *DATA* of the ALV interface controller.

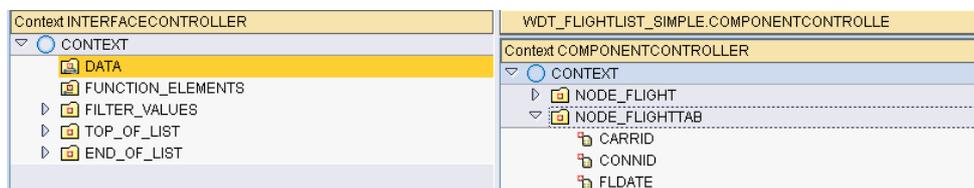
Open your Web Dynpro component's node *Component Usages* -> *ALV* -> *INTERFACECONTROLLER\_USAGE*.

Hint: If the node *Component Usage* is not displayed in the object tree, save your component and refresh the tree display.



Click on the Controller Usage button. The component controller of your Web Dynpro component appears on the right side of the screen.

Map context node `NODE_FLIGHTTAB` of your Web Dynpro component to context `DATA` of the interface controller of the ALV component.

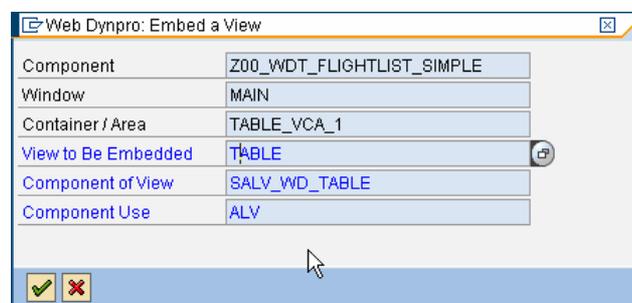


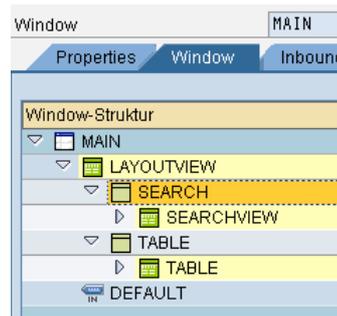
After you have done the mapping, the context node `DATA` has a double-arrow inside the folder icon.



## Embed view `TABLE` of component `SALV_WD_TABLE` into `LAYOUTVIEW`.

To embed the ALV table into the `LAYOUTVIEW`, go to the window `MAIN` and switch to tab page `window`. Embed the ALV view `TABLE` by choosing the context menu entry `embed view`. A popup appears. Use the F4 help and select the following entry.





Activate your Web Dynpro Component.

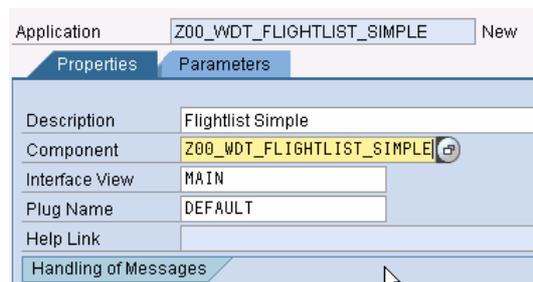


## Create and Test Web Dynpro Application

Each Web Dynpro component needs a Web Dynpro application to be executed.

### Procedure

Create a Web Dynpro application for your Web Dynpro component:



Test your Web Dynpro application. The result will look like the following:

**Select the Flights**

Fluggesellschaft:

Flugnummer:

Sicht    Filter Einstellungen

Airline	Flug-Nr.	Datum	Preis	Währung	Fl.-Typ	Kapazität	Belegt	Summe
LH	0400	01.01.2005	666,00	EUR	A310-300	280	270	210.595,86
LH	0400	29.01.2005	666,00	EUR	A310-300	280	271	213.179,94
LH	0400	26.02.2005	666,00	EUR	A310-300	280	271	213.379,74
LH	0400	26.03.2005	666,00	EUR	A310-300	280	269	209.576,88
LH	0400	23.04.2005	666,00	EUR	A310-300	280	271	210.569,22
LH	0400	21.05.2005	666,00	EUR	A310-300	280	271	211.561,56
LH	0400	28.05.2005	666,00	EUR	A310-300	280	271	211.854,60
LH	0400	18.06.2005	666,00	EUR	A310-300	280	271	209.163,96
LH	0400	28.06.2005	666,00	EUR	A310-300	280	270	211.281,84
LH	0400	16.07.2005	666,00	EUR	A310-300	280	271	208.997,46

Zeile 1 von 84

## Author Bio



Claudia Dangers is a senior development consultant in SAP's Software Technology and Development department. Since she joined SAP in 1999 she has worked on numerous projects and gained practical experience in ABAP and BSP development, in the creation of concepts, in coaching and code reviews, and as a sub-project lead and training instructor. Claudia is very interested in new technologies. Currently she is dealing with Web Dynpro ABAP, kernel-based BADI's and the Switch and Enhancement Framework.