

# Programming the ALV Configuration Model 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>Copying an Existing Web Dynpro Component .....</b>	<b>6</b>
Procedure.....	6
 <b>Create View for Displaying ALV Table .....</b>	<b>6</b>
Procedure.....	6
 <b>Create and Test Web Dynpro Application .....</b>	<b>12</b>
Procedure.....	12
<b><i>Author Bio .....</i></b>	<b><i>13</i></b>



## Handling ALV Tables in Web Dynpro

This tutorial shows you the usage of the ALV configuration model for doing the following tasks:

- Set the row count of the ALV table
- Fade out columns
- Sort the table
- Display icons instead of text

### Task

The starting point of this tutorial is the solution application of the tutorial “Simple Example for Using the ALV Inside Web Dynpro for ABAP,” where you can search for special flights and show the details of the selected flight in an ALV table.

The task of this tutorial is to get familiar with the ALV configuration model. You will learn how to use it to adjust the display of the ALV table to your needs.

### Objectives

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

- ✓ Configure the ALV

### Knowledge

- Knowledge of ABAP OO programming language
- Basic knowledge of programming Web Dynpro applications
- Basic knowledge of ABAP workbench
- Familiar with the tutorial, “Simple Example for Using the ALV Inside Web Dynpro for ABAP”



## Copying an Existing Web Dynpro Component

In the system there is a master copy of a Web Dynpro component called *WDT\_FLIGHTLIST\_SIMPLE*. You can copy this component as described below.

### Procedure

#### Copying the Web Dynpro Component

1. Start the ABAP Workbench (se80) and select the Web Dynpro component *WDT\_FLIGHTLIST\_SIMPLE*.
2. Open the context menu of *WDT\_FLIGHTLIST\_SIMPLE* and copy the Web Dynpro component to *Z00\_WDT\_FLIGHTLIST\_CONFIG*.
3. Open the context menu of the new component *Z00\_WDT\_FLIGHTLIST\_CONFIG* and create a Web Dynpro application *Z00\_WDT\_FLIGHTLIST\_CONFIG*.
4. Select the interface view by using F4 help. Choose *MAIN*.
5. Select a plug name by using F4 help and choose *default*.
6. Activate the new Web Dynpro component.



## Create View for Displaying ALV Table

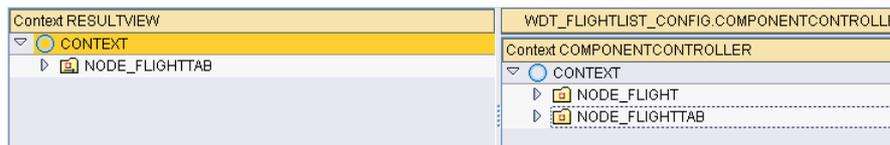
In the last simple tutorial, we embedded the ALV table directly into the window. This is the easiest way of displaying an ALV table. In this tutorial we want to use a separate view to display the ALV table. This gives us the possibility to use the standard hook methods of the view to configure the ALV, instead of using the standard hook methods of the component controller.

### Procedure

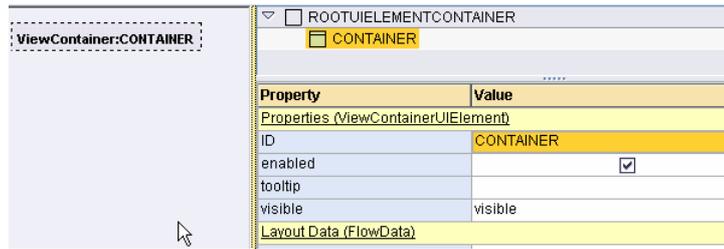
#### Create view *RESULTVIEW*.

Create view *RESULTVIEW*.

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

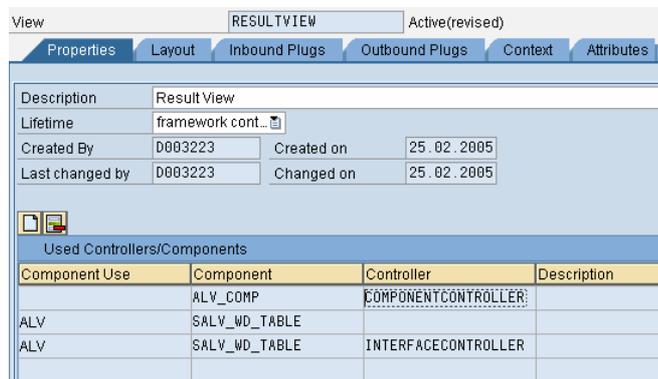
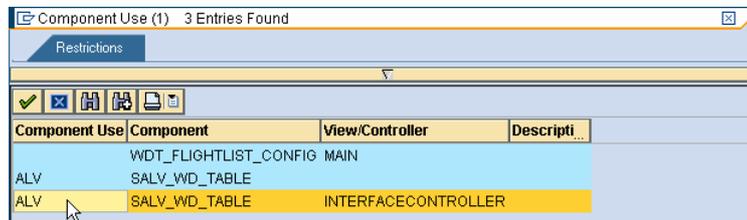


In the layout of view *RESULTVIEW* create a *ViewContainerUIElement* called *CONTAINER*.



**Define component usage SALV\_WD\_TABLE in ResultView.**

To be able to use the ALV component model inside view RESULTVIEW it is necessary to define the component usage of SALV\_WD\_TABLE in the view. Navigate to the properties of view RESULTVIEW and press button  (*Create Controller Usage*) and choose the following entry from the list on the popup:



**Configuring ALV in ResultView**

You want to do the following changes to the standard ALV layout:

- Display 5 lines in a page instead of the default value of 10 lines.
- Display a traffic light icon instead the occupied seats.
- Sort the table descending by the occupied seats.
- Delete the column with the status.

**Procedure**

**Instantiate ALV component.**

Implement the standard hook method *WDDOINIT* of the view *RESULTVIEW*:

The first step - if you want to program the ALV component model - is to create an instance of the ALV component. This can easily be done with the help of the Web Dynpro code

wizard.

Instantiate Used Component

Component Use

```

WDDOINIT()

METHOD wddoinit .

* Create component usage for alv component
DATA: l_ref_cmp_usage TYPE REF TO if_wd_component_usage.

l_ref_cmp_usage = wd_this->wd_cpuse_alv( ).
IF l_ref_cmp_usage->has_active_component( ) IS INITIAL.
    l_ref_cmp_usage->create_component( ).
ENDIF.

[...]
```

### Call interface method GET\_MODEL().

The next step is to get the ALV configuration model. The Web Dynpro code wizard also supports this step:

Method Call in Used Controller

Component Name

Component Use

Controller Name

Method Name

```

WDDOINIT()

[...]
```

```

* Get config model
DATA: l_ref_interfacecontroller TYPE REF TO iwci_salv_wd_table .
l_ref_interfacecontroller = wd_this->wd_cpifc_alv( ).

DATA: l_value TYPE REF TO c1_salv_wd_config_table.

l_value = l_ref_interfacecontroller->get_model( ).

[...]
```

### Configure ALV.

Now we can start with configuring the ALV table. First we want to change the visible row count to 5.

```

WDDOINIT()

[...]
```

```

* set visible row count
```

```
l_value->if_salv_wd_table_settings~set_visible_row_count( '5' ).
[...]
```

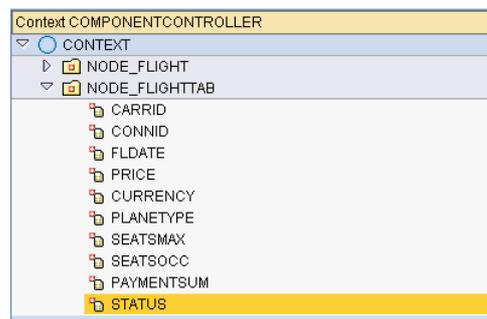
To sort the table descending by the occupied seats, the following coding has to be included:

#### WDDOINIT ( )

```
[...]
* Sort rows by seatsocc descending
DATA: lr_field TYPE REF TO cl_salv_wd_field.

lr_field =
  l_value->if_salv_wd_field_settings~get_field( 'SEATSOCC' ).
lr_field->if_salv_wd_sort~create_sort_rule( sort_order =
  if_salv_wd_c_sort=>sort_order_descending ).
[...]
```

For displaying the traffic light in column SEATSOCC, we need an additional column which holds the name of the icon to display. Therefore the first step is to add a new field to node *NODE\_FLIGHTTAB* of the component controller. The name of the field is *STATUS* and the type is *STRING*.



Thereafter the coding of method *FILL\_FLIGHTTAB* needs to be enhanced to fill the new field *STATUS* in the internal table *lt\_flights*. If no seat is free on a flight the red traffic light should be displayed. If 1 to 50 seats are free the yellow traffic light should be displayed and for the rest the green one.

#### FILL\_FLIGHTTAB ( )

```
[...]
Data:
  lt_flights TYPE if_componentcontroller=>Elements_Node_Flighttab,
  ls_flights type if_componentcontroller=>Element_Node_Flighttab,
  lv_seatsfree type i.

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

* fill column STATUS
LOOP AT lt_flights INTO ls_flights.
```

```

lv_seatsfree = ls_flights-seatsmax - ls_flights-seatsocc.
IF lv_seatsfree = 0.
  ls_flights-status = 'ICON_RED_LIGHT'.
ELSEIF lv_seatsfree <= 50.
  ls_flights-status = 'ICON_YELLOW_LIGHT'.
ELSE.
  ls_flights-status = 'ICON_GREEN_LIGHT'.
ENDIF.
modify lt_flights from ls_flights transporting status.
ENDLOOP.

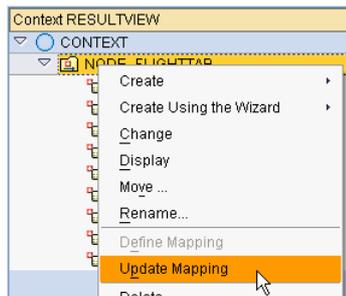
* navigate from <CONTEXT> to <NODE_FLIGHT> via lead selection
node_node_flighttab = wd_context->get_child_node( name =
`NODE_FLIGHTTAB` ).

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

endmethod.

```

The next step is to navigate to the context tab of the view *RESULTVIEW* and update the mapping.



Now the configuration for displaying an image (instead of the number of occupied seats) needs to be done in method *WDDOINIT* of view *RESULTVIEW*:

```

WDDOINIT ( )

[...]

* Display icon in column seatsocc
DATA: lr_column TYPE REF TO c1_salv_wd_column,
      lr_image   TYPE REF TO c1_salv_wd_uie_image,
      lv_icon    TYPE string.

  lr_column = l_value->if_salv_wd_column_settings~get_column(
'SEATSOCC' ).
  CREATE OBJECT lr_image.
  lr_image->SET_SOURCE_FIELDNAME( 'STATUS' ).
  lr_column->set_cell_editor( lr_image ).           "Display
traffic light images in column 'SEATSOCC'

[...]

```

The column *STATUS* is only internal and should not be displayed in the alv table. To delete the column, the following coding has to be inserted:

```
WDDOINIT ( )
```

```
[...]
```

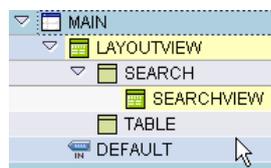
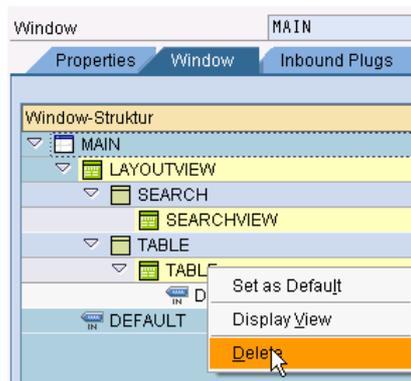
```
* delete column STATUS
```

```
  l_value->if_salv_wd_column_settings~delete_column( 'STATUS' ).
```

```
ENDMETHOD.
```

### Embed view **TABLE** of component **SALV\_WD\_TABLE** into window **MAIN**.

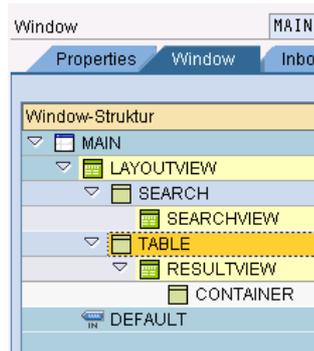
Go to the window *MAIN* and switch to tab page *window*. Delete the embedded view *TABLE*. Hint: This only deletes the embedding relationship, not the view at all!



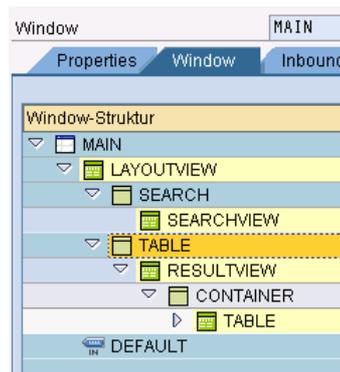
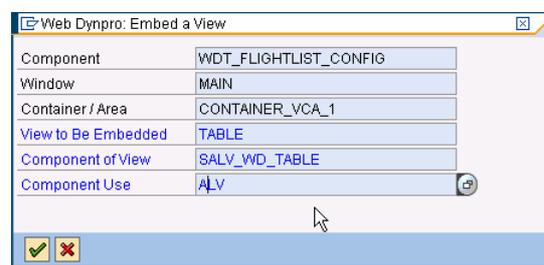
Embed the view *RESULTVIEW* to view container *TABLE* using the context menu *embed view*. A popup appears. Use the F4 help and select the following entry:

Web Dynpro: Embed a View	
Component	WDT_FLIGHTLIST_CONFIG
Window	MAIN
Container / Area	TABLE_VCA_1
View to Be Embedded	RESULTVIEW
Component of View	WDT_FLIGHTLIST_CONFIG
Component Use	----

At the bottom of the dialog, there are two icons: a green checkmark and a red 'X'.



Embed the ALV view *TABLE* to the container inside the *RESULTVIEW* using the context menu *embed view*. A popup appears. Use the F4 help and select the following entry:



## 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	28.06.2005	666,00	EUR	A310-300	280	⊙△⊙	211.281,84
LH	0400	26.03.2005	666,00	EUR	A310-300	280	⊙△⊙	209.576,88
LH	0400	13.08.2005	666,00	EUR	A310-300	280	⊙△⊙	207.212,58
LH	0400	05.11.2005	666,00	EUR	A310-300	280	⊙⊙⊙	90.849,06
LH	0400	08.10.2005	666,00	EUR	A310-300	280	⊙⊙⊙	85.847,40

Seite 9 von 17

## 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.