**ALV**

# SAP ABAP Tutorial - ALV Grid Example with cl\_gui\_alv\_grid and Screen Painter

# *from* SAP ABAP Programming and HANA Database Tutorials

<https://www.kodyaz.com/articles/sap-abap-tutorial-alv-grid-cl_gui_alv_grid-screen-painter.aspx>

# An Easy Reference for ALV GRID CONTROL

# <https://abap-blog.ru/wp-content/uploads/2020/04/ALV.pdf>

# The ALV Grid control

# *from* ABAP Tips /Compiled by Henrik Frank/ - Last updated 28.06.2008

# <https://henrikfrank.dk/abaptips/abapindex.htm?page=abapobjects%2falvgrid_control.htm>

## [**alv\_call\_stack**](https://abap-blog.ru/osnovy-abap/dinamicheskie-zhurnalnye-tochki/attachment/alv_call_stack/)

# <https://abap-blog.ru/?s=alv>

# The wrapper class implemented to encapsulate ALV Grid functionality is CL\_GUI\_ALV\_GRID.

# List data - the data in an internal table to be listed.

# Field Catalog - an internal table to define specifications on how the fields of our list will be displayed.

# There are three procedures to generate the field catalog as

# *Automatic* generation,

# *Semi-automatic* generation,

# *Manual* generation.

# The internal table for the field catalog must be referenced to the dictionary type LVC\_T\_FCAT.

# Layout Structure - we fill a structure to specify - layout options, grid customizing, totals options, color adjustments - for the grid. The layout structure must be of type LVC\_S\_LAYO.

# Grid container

# ALV Grid instance requires a container to be linked to the screen. An instance of one of the following classes can be used as container

# *cl\_gui\_custom\_container*,

# cl\_gui\_docking\_container,

# cl\_gui\_dialogbox\_container.

**Step 1.**

Add a custom control on the screen which will be related to the   
custom container. Let’s give it the name *C\_ALV*.

**Step 2.**

Declare global variables to be used for ALV Grid.

DATA gr\_alvgrid TYPE REF TO cl\_gui\_alv\_grid.

*\* Name of the custom control added on the screen*

DATA gc\_custom\_control\_name TYPE scrfname VALUE ‘CC\_ALV’.

DATA gr\_ccontainer TYPE REF TO cl\_gui\_custom\_container.

DATA gt\_fieldcat TYPE lvc\_t\_fcat .

DATA gs\_layout TYPE lvc\_s\_layo.

**Step 3.**

Declare your internal table /*gt\_list*/ which is supposed to hold the list data.

DATA BEGIN OF gt\_list OCCURS 0.

INCLUDE STRUCTURE SFLIGHT .

*\* In further sections, some additional fields will added here for some functionality*

DATA END OF gt\_list.

**Step 4.**

Fill your list data as you want.

**Step 5.**

Call the screen which comprises the ALV Grid control. At PBO of   
this screen we will deal with creating the ALV Grid instance.

*\* PBO*

PROCESS BEFORE OUTPUT.

...

MODULE display\_alv.

...

MODULE display\_alv OUTPUT.

PERFORM display\_alv.

ENDMODULE.

**Step 6.**

Check whether an instance of the container /or ALV Grid/ exists. If it exists, refreshing it, and if not, creating and setting ALV for the first display.

After creating the ALV Grid instance we call ***cl\_gui\_alv\_grid->set\_table\_for\_first\_display*** method to make our list displayed.

**Building Field Catalog**

The simplest way /*Automatic* generation/ applies if our list structure is similar to a dictionary table. To do this, we simply eliminate the form call and *pass the name of dictionary structure* /in our   
example, SFLIGHT/ to the parameter**I\_STRUCTURE\_NAME**/.

*\* Semi-automatically field-catalog preparement*

FORM prepare\_field\_catalog CHANGING pt\_fieldcat TYPE lvc\_t\_fcat.

DATA ls\_fcat type lvc\_s\_fcat.

CALL FUNCTION 'LVC\_FIELDCATALOG\_MERGE'

EXPORTING i\_structure\_name = 'SFLIGHT'

CHANGING ct\_fieldcat = pt\_fieldcat[]

EXCEPTIONS

inconsistent\_interface = 1

program\_error = 2

OTHERS = 3.

IF sy-subrc <> 0.

*\* Exception handling*

ENDIF.

LOOP AT pt\_fieldcat INTO ls\_fcat.

CASE ls\_fcat-fieldname.

WHEN 'CARRID'.

ls\_fcat-outputlen = '10'.

ls\_fcat-coltext = 'Airline Carrier ID'.

MODIFY pt\_fieldcat FROM ls\_fcat.

WHEN 'PAYMENTSUM'.

ls\_fcat-no\_out = 'X'.

MODIFY pt\_fieldcat FROM ls\_fcat .

ENDCASE .

ENDLOOP .

ENDFORM .

**Layout Adjustments**

To define general appearance of our ALV Grid we fill a structure of type **LVC\_S\_LAYO**.

FORM ***prepare\_layout*** CHANGING ps\_layout TYPE lvc\_s\_layo.

ps\_layout-zebra = 'X'.

ps\_layout-grid\_title = 'Flights'.

ps\_layout-smalltitle = 'X'.

ENDFORM.

**Printing Adjustments**

We handle printing adjustments via a structure to be passed to the parameter   
**is\_print** of the method ***set\_table\_for\_first\_display***.

The print output of the field *PRNTLSTINF* is not visible in the print preview   
of the ALV Grid. If you create a spool request first, you can check the final list layout   
in transaction SP01.

*Rem*

*PRNTLSTINF* - Prints list information. If this field is set, information on sorting, subtotals and filters   
defined as well as data statistics are printed at the beginning of the list.

**Excluding Unwanted Standard Function Buttons**

In your list, you may want to exclude some of the standard function buttons   
since they are not useful for your list. To exclude those buttons, you *fill a table of type* **UI\_FUNCTIONS** *and pass it to the parameter* **IT\_TOOLBAR\_EXCLUDING** of the method ***set\_table\_for\_first\_display***. *The function codes for the buttons* may be acquired by inspecting the constant attributes of the class *cl\_gui\_alv\_grid* or putting a break point into a method, like the event-handling method of the *event after\_user\_command*, which deals with the ALV command. *To hide the entire toolbar*, you can set the field *NO\_TOOLBAR* of the layout structure to ‘X’.

FORM ***exclude\_tb\_functions*** CHANGING pt\_exclude TYPE ui\_functions.

DATA ls\_exclude TYPE ui\_func.

ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_maximum.

APPEND ls\_exclude TO pt\_exclude.

ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_minimum.

APPEND ls\_exclude TO pt\_exclude.

ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_subtot.

APPEND ls\_exclude TO pt\_exclude.

ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_sum.

APPEND ls\_exclude TO pt\_exclude.

ls\_exclude = cl\_gui\_alv\_grid=>mc\_fc\_average.

APPEND ls\_exclude TO pt\_exclude.

ls\_exclude = cl\_gui\_alv\_grid=>mc\_mb\_sum .

APPEND ls\_exclude TO pt\_exclude.

ls\_exclude = cl\_gui\_alv\_grid=>mc\_mb\_subtot .

ENDFORM .

Here, names beginning with **MC\_FC\_** are names *for functions directly* and the names beginning with **MC\_MB\_** are *for the function menus* including some subfunctions as menu entries. By excluding one from the latter type, you exclude all of the functions under it.