

```
*&-----*
*& Report   ZALV_EDITF4DISPLAY   *
*&                                               *
*&-----*
*&                                               *
*&                                               *
*&-----*
```

REPORT zalv_editf4display.

*Type pools for alv
TYPE-POOLS : slis.

*structure for t582a tbale
TYPES : BEGIN OF ty_table,
 infty TYPE infty,
 pnnnn TYPE pnnnn_d,
 zrmkz TYPE dzrmkz,
 zeitb TYPE dzeitb,
 dname TYPE dianm,
 davoe TYPE davoe,
 END OF ty_table.

*Structure for infotype text
TYPES : BEGIN OF ty_itext,
 infty TYPE infty,
 itext TYPE intxt,
 sprsl TYPE sprsl,
 END OF ty_itext.

*Structure for output display
TYPES : BEGIN OF ty_output,
 infty TYPE infty,
 itext TYPE intxt,
 pnnnn TYPE pnnnn_d,
 zrmkz TYPE dzrmkz,
 zeitb TYPE dzeitb,
 dname TYPE dianm,
 davoe TYPE davoe,
 END OF ty_output.

*internal table and work area declarations
DATA : it_table TYPE STANDARD TABLE OF ty_table INITIAL SIZE 0,
 it_output TYPE STANDARD TABLE OF ty_output INITIAL SIZE 0,
 it_pbo TYPE STANDARD TABLE OF ty_output INITIAL SIZE 0,
 it_itttext TYPE STANDARD TABLE OF ty_itext INITIAL SIZE 0,
 wa_table TYPE ty_table,
 wa_output TYPE ty_output,
 wa_itttext TYPE ty_itext.

*Data declarations for dropdown lists for f4
DATA: it_dropdown TYPE lvc_t_drop,
 ty_dropdown TYPE lvc_s_drop,
*data declaration for refreshing of alv
 stable TYPE lvc_s_stbl.

*Global variable declaration
DATA: gstring TYPE c.

```

*Data declarations for ALV
DATA: c_ccont TYPE REF TO cl_gui_custom_container,      "Custom container
object
      c_alvgd      TYPE REF TO cl_gui_alv_grid,         "ALV grid object
      it_fcat      TYPE lvc_t_fcat,                    "Field catalogue
      it_layout    TYPE lvc_s_layo.                    "Layout

*ok code declaration
DATA:
      ok_code      TYPE ui_func.

*initialization event

INITIALIZATION.

*start of selection event
START-OF-SELECTION.

*select the infotypes maintained
SELECT infty
      pnnnn
      zrmkz
      zeitb
      dname
      davo
      davoe
      FROM t582a UP TO 10 ROWS
      INTO CORRESPONDING FIELDS OF TABLE it_table.

* *Select the infotype texts
IF it_table[] IS NOT INITIAL.
  SELECT itext
        infty
        sprsl
        FROM t582s
        INTO CORRESPONDING FIELDS OF TABLE it_ittext
        FOR ALL ENTRIES IN it_table
        WHERE infty = it_table-infty
        AND sprsl = 'E'.
ENDIF.

*Appending the data to the internal table of ALV output
LOOP AT it_table INTO wa_table.

      wa_output-infty = wa_table-infty.
      wa_output-pnnnn = wa_table-pnnnn.
      wa_output-zrmkz = wa_table-zrmkz.
      wa_output-zeitb = wa_table-zeitb.
      wa_output-dname = wa_table-dname.
      wa_output-davo = wa_table-davo.
      wa_output-davoe = wa_table-davoe.

* For texts

      READ TABLE it_ittext INTO wa_ittext WITH KEY infty = wa_table-infty.
      wa_output-itext = wa_ittext-itext.

      APPEND wa_output TO it_output.
      CLEAR wa_output.

```

```

ENDLOOP.

* Calling the ALV screen with custom container

CALL SCREEN 0600.

*On this statement double click it takes you to the screen painter SE51.
*Enter the attributes
*Create a Custom container and name it CCONT and OK code as OK_CODE.
*Save check and Activate the screen painter.
*Now a normal screen with number 600 is created which holds the ALV grid.
* PBO of the actual screen ,
* Here we can give a title and customized menus
*create 2 buttons with function code 'SAVE' and 'EXIT'.
* GIVE A SUITABLE TITLE

*&-----*
*&      Module  STATUS_0600  OUTPUT
*&-----*
*      text
*-----*
MODULE status_0600 OUTPUT.
  SET PF-STATUS 'DISP'.
  SET TITLEBAR 'ALVF4'.
ENDMODULE.          " STATUS_0600  OUTPUT

* calling the PBO module ALV_GRID.
*&-----*
*&      Module  PBO  OUTPUT
*&-----*
*      text
*-----*
MODULE pbo OUTPUT.

*Creating objects of the container
CREATE OBJECT c_ccont
EXPORTING
  container_name = 'CCONT'.

* create object for alv grid
create object c_alvgd
exporting
i_parent = c_ccont.

* SET field for ALV
PERFORM alv_build_fieldcat.

* Set ALV attributes FOR LAYOUT
PERFORM alv_report_layout.

CHECK NOT c_alvgd IS INITIAL.

* Call ALV GRID

CALL METHOD c_alvgd->set_table_for_first_display
EXPORTING
  is_layout          = it_layout
  i_save             = 'A'
CHANGING
  it_outtab          = it_output
  it_fieldcatalog    = it_fcata

```

```

EXCEPTIONS
  invalid_parameter_combination = 1
  program_error                 = 2
  too_many_lines                = 3
  OTHERS                        = 4.
IF sy-subrc <> 0.
  MESSAGE ID sy-msgid TYPE sy-msgty NUMBER sy-msgno
           WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
ENDIF.

```

```

ENDMODULE.          " PBO OUTPUT

```

```

*&-----*
*&      Form alv_build_fieldcat
*&-----*
*      text
*-----*
*      <--P_IT_FCAT text
*-----*
*subroutine to build fieldcat

```

```

FORM alv_build_fieldcat.

```

```

  DATA lv_fldcat TYPE lvc_s_fcat.
  CLEAR lv_fldcat.

```

```

  lv_fldcat-row_pos   = '1'.
  lv_fldcat-col_pos   = '1'.
  lv_fldcat-fieldname = 'INFTY'.
  lv_fldcat-tabname   = 'IT_OUTPUT'.
  lv_fldcat-outputlen = 8.
  lv_fldcat-scrtext_m = 'Infotype'.
  lv_fldcat-icon      = 'X'.
  APPEND lv_fldcat TO it_fcat.
  CLEAR lv_fldcat.

```

```

  lv_fldcat-row_pos   = '1'.
  lv_fldcat-col_pos   = '2'.
  lv_fldcat-fieldname = 'PNNNN'.
  lv_fldcat-tabname   = 'IT_OUTPUT'.
  lv_fldcat-outputlen = 15.
  lv_fldcat-scrtext_m = 'Structure'.
  lv_fldcat-icon      = ''.
  APPEND lv_fldcat TO it_fcat.
  CLEAR lv_fldcat.

```

```

  lv_fldcat-row_pos   = '1'.
  lv_fldcat-col_pos   = '3'.
  lv_fldcat-fieldname = 'ITEXT'.
  lv_fldcat-tabname   = 'IT_OUTPUT'.
  lv_fldcat-outputlen = 60.
  lv_fldcat-scrtext_m = 'Description'.
  lv_fldcat-icon      = ''.
  APPEND lv_fldcat TO it_fcat.
  CLEAR lv_fldcat.

```

```

  lv_fldcat-row_pos   = '1'.
  lv_fldcat-col_pos   = '5'.
  lv_fldcat-fieldname = 'ZRMKZ'.

```

```

lv_fldcat-tabname    = 'IT_OUTPUT'.
lv_fldcat-outputlen = 1.
lv_fldcat-scrtext_m = 'PERIOD'.
lv_fldcat-icon      = ''.
APPEND lv_fldcat TO it_fcat.
CLEAR lv_fldcat.

lv_fldcat-row_pos   = '1'.
lv_fldcat-col_pos   = '6'.
lv_fldcat-fieldname = 'ZEITB'.
lv_fldcat-tabname   = 'IT_OUTPUT'.
lv_fldcat-outputlen = 5.
lv_fldcat-scrtext_m = 'Time constraint'.
lv_fldcat-edit      = 'X'.
*To avail the existing F4 help these are to
*be given in the field catalogue
lv_fldcat-f4availabl = 'X'.
lv_fldcat-ref_table  = 'T582A'.
lv_fldcat-ref_field  = 'ZEITB'.

APPEND lv_fldcat TO it_fcat.
CLEAR lv_fldcat.

lv_fldcat-row_pos   = '1'.
lv_fldcat-col_pos   = '7'.
lv_fldcat-fieldname = 'DNAME'.
lv_fldcat-tabname   = 'IT_OUTPUT'.
lv_fldcat-outputlen = 15.
lv_fldcat-scrtext_m = 'Dialogmodule'.
lv_fldcat-icon      = ''.
APPEND lv_fldcat TO it_fcat.
CLEAR lv_fldcat.

lv_fldcat-row_pos   = '1'.
lv_fldcat-col_pos   = '8'.
lv_fldcat-fieldname = 'DAVO'.
lv_fldcat-tabname   = 'IT_OUTPUT'.
lv_fldcat-outputlen = 15.
lv_fldcat-scrtext_m = 'Start'.
lv_fldcat-edit      = 'X'.
APPEND lv_fldcat TO it_fcat.
CLEAR lv_fldcat.

lv_fldcat-row_pos   = '1'.
lv_fldcat-col_pos   = '9'.
lv_fldcat-fieldname = 'DAVOE'.
lv_fldcat-tabname   = 'IT_OUTPUT'.
lv_fldcat-outputlen = 15.
lv_fldcat-scrtext_m = 'End'.
lv_fldcat-icon      = ''.
APPEND lv_fldcat TO it_fcat.
CLEAR lv_fldcat.

*To create drop down for the field 'DAVO'
* with our own f4 help
ty_dropdown-handle = '1'.
ty_dropdown-value  = ' '.
APPEND ty_dropdown TO it_dropdown.
ty_dropdown-handle = '1'.
ty_dropdown-value  = '1'.
APPEND ty_dropdown TO it_dropdown.
ty_dropdown-handle = '1'.

```

```

ty_dropdown-value = '2'.
APPEND ty_dropdown TO it_dropdown.
ty_dropdown-handle = '1'.
ty_dropdown-value = '3'.
APPEND ty_dropdown TO it_dropdown.

CALL METHOD c_alvkd->set_drop_down_table
EXPORTING
    it_drop_down = it_dropdown.

LOOP AT it_fcat INTO lv fldcat.
    CASE lv fldcat-fieldname.
** To assign dropdown in the fieldcatalogue
    WHEN 'DAVO'.
        lv fldcat-drdrn_hndl = '1'.
        lv fldcat-outputlen = 15.
        MODIFY it_fcat FROM lv fldcat.
    ENDCASE.
ENDLOOP.

ENDFORM.                " alv_build_fieldcat

*&-----*
*&      Form  alv_report_layout
*&-----*
*      text
*-----*
*      <--P_IT_LAYOUT  text
*-----*
*Subroutine for setting alv layout
FORM alv_report_layout.
    it_layout-cwidth_opt = 'X'.
    it_layout-col_opt = 'X'.
    it_layout-zebra = 'X'.

ENDFORM.                " alv_report_layout

* PAI module of the screen created. In case we use an interactive ALV or
*for additional functionalities we can create OK codes
*and based on the user command we can do the coding.
*&-----*
*&      Module  PAI  INPUT
*&-----*
*      text
*-----*
MODULE pai INPUT.

*To change the existing values and refresh the grid
*And only values in the dropdown or in the default
*F4 can be given , else no action takes place for the dropdown
*and error is thrown for the default F4 help and font changes to red
*and on still saving, value is not changed

    c_alvkd->check_changed_data( ).

*Based on the user input
*When user clicks 'SAVE;
    CASE ok_code.

        WHEN 'SAVE'.

```

```

*A pop up is called to confirm the saving of changed data
  CALL FUNCTION 'POPUP_TO_CONFIRM'
    EXPORTING
      titlebar          = 'SAVING DATA'
      text_question     = 'Continue?'
      icon_button_1     = 'icon_booking_ok'
    IMPORTING
      answer            = gstring
    EXCEPTIONS
      text_not_found   = 1
      OTHERS           = 2.
  IF sy-subrc NE 0.
*   MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
*   WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
  ENDIF.

*When the User clicks 'YES'
  IF ( gstring = '1' ).
    MESSAGE 'Saved' TYPE 'S'.
*Now the changed data is stored in the it_pbo internal table
  it_pbo = it_output.
*Subroutine to display the ALV with changed data.
  PERFORM redisplay.
  ELSE.
*When user clicks NO or Cancel
  MESSAGE 'Not Saved' TYPE 'S'.
  ENDIF.
**When the user clicks the 'EXIT; he is out
  WHEN 'EXIT'.
    LEAVE PROGRAM.
  ENDCASE.

  CLEAR: ok_code.

ENDMODULE.          " PAI INPUT
*&-----*
*&      Form REDISPLAY
*&-----*
*      text
*-----*
* --> p1      text
* <-- p2      text
*-----*
FORM redisplay .

*Cells of the alv are made non editable after entering OK to save

  CALL METHOD c_alvkd->set_ready_for_input
    EXPORTING
      i_ready_for_input = 0.

*Row and column of the alv are refreshed after changing values

  stable-row = 'X'.
  stable-col = 'X'.

*Refreshed ALV display with the changed values
*This ALV is non editable and contains new values
  CALL METHOD c_alvkd->refresh_table_display
    EXPORTING
      is_stable = stable

```

```

EXCEPTIONS
  finished = 1
  OTHERS = 2.
IF sy-subrc <> 0.
* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
* WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
ENDIF.

ENDFORM.          " REDISPLAY

```

Initial Output.

ALV with editable F4 ,new values saved and display refreshed

Infotype	Structure	Description	PERIOD	TC	Dialogmodule	Start	End
0000	P0000	0000 Actions	I	1	RP_0000	1	1
0001	P0001	0001 Organizational Assignment	I	1	RP_0001	1	1
0002	P0002	0002 Personal Data	I	1	RP_0002	3	1
0003	P0003	0003 Payroll Status	I	A	RP_0003		
0004	P0004	0004 Challenge	I	2	RP_0004	1	1
0005	P0005	0005 Leave Entitlement	I	2	RP_0005		
0006	P0006	0006 Addresses	I	T	RP_0006	1	1
0007	P0007	0007 Planned Working Time	I	1	RP_0007	1	2
0008	P0008	0008 Basic Pay	I	T	RP_0008	1	1
0009	P0009	0009 Bank Details	I	T	RP_0009	1	1

Default F4 HELP for the field TC

Infotype	Structure	Description	PERIOD	TC	Dialogmodule	Start	End
0000	P0000	0000 Actions	I	1	RP_0000	1	1
0001	P0001	0001 Organizational Assignment	I				
0002	P0002	0002 Personal Data	I				
0003	P0003	0003 Payroll Status	I				
0004	P0004	0004 Challenge	I				
0005	P0005	0005 Leave Entitlement	I				
0006	P0006	0006 Addresses	I				
0007	P0007	0007 Planned Working Time	I				
0008	P0008	0008 Basic Pay	I				
0009	P0009	0009 Bank Details	I				

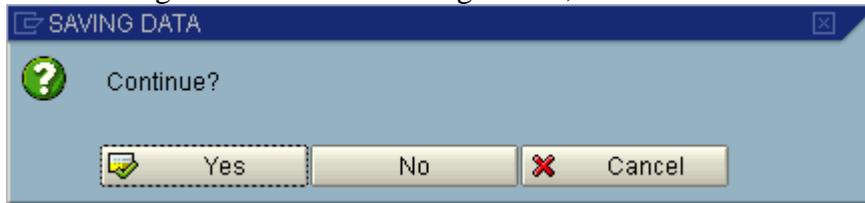
TC Short text

- 1 Record must have no gaps, no overlappings
- 2 Record may include gaps, no overlappings
- 3 Record may include gaps, can exist more than once
- A Infotype exists just once from Jan.1 1800 to Dec.12 9999
- B IT exists for maximum of once from Jan.1 1800 to Dec.12 9999
- T Time constraint is based on subtype or subtype table
- Z Time constraint for time management infotypes -> T554Y

F4 help in the dropdown

fotype	Structure	Description	PERIOD	TC	Dialogmodule	Start	End
0000	P0000	0000 Actions	I	1	RP_0000	1	1
0001	P0001	0001 Organizational Assignment	I	1	RP_0001	1	1
0002	P0002	0002 Personal Data	I	1	RP_0002	1	1
0003	P0003	0003 Payroll Status	I	A	RP_0003	2	3

After editing the values and clicking SAVE,



When YES, Changed Data gets SAVED and ALV is refreshed.

System Help

ALV with editable F4 ,new values saved and display refreshed

SAVE EXIT

Infotype	Structure	Description	PERIOD	TC	Dialogmodule	Start	End
0000	P0000	0000 Actions	I	3	RP_0000	2	1
0001	P0001	0001 Organizational Assignment	I	1	RP_0001	1	1
0002	P0002	0002 Personal Data	I	1	RP_0002	3	1
0003	P0003	0003 Payroll Status	I	A	RP_0003	1	1
0004	P0004	0004 Challenge	I	2	RP_0004	1	1
0005	P0005	0005 Leave Entitlement	I	2	RP_0005	3	1
0006	P0006	0006 Addresses	I	T	RP_0006	1	1
0007	P0007	0007 Planned Working Time	I	1	RP_0007	1	2
0008	P0008	0008 Basic Pay	I	T	RP_0008	1	1
0009	P0009	0009 Bank Details	I	T	RP_0009	1	1

Saved