LAURIX CONSULTING

SALV: ALV List Quickstart

Snippets to get your ALV Grid
up and running in no time 🚀
laurix.com/post/abap/alv/salv-quickstart

Basics

```
cl_salv_table=>factory(
  IMPORTING
    r_salv_table = DATA(lo_alv)
  CHANGING
    t_table      = mt_data ).
"Do stuff…
lo_alv->display( ).
```

Column settings

```
"Change header text
DATA(lo_cols) = lo_alv->get_columns( ).
DATA(lo_col) = lo_cols->get_column( 'COLNAME' ).
lo_col->set_long_text( 'LONG_HEADER' ).
lo_col->set_medium_text( 'MEDIUM_HEADER' ).
lo_col->set_short_text( 'SHORT_HEADER' ).

"Optimize column width
DATA(lo_cols) = lo_alv->get_columns( ).
lo_cols->set_optimize( abap_true ).

"Sort a column up
lo_alv->get_sorts( )->add_sort( 'COLUMNNAME' ).
```

Hotspots

```
"Register column
DATA(lo_column_tab) = CAST cl_salv_column_table(
 lo_alv->get_columns( )->get_column(
   'COLUMNNAME') ).
lo_column_tab->set_cell_type(
  if_salv_c_cell_type=>hotspot ).

"Register click handler
DATA(lo_events) = lo_alv->get_event( ).
SET HANDLER lcl_application->on_link_click
  FOR lo_events.

"Handler implementation
CLASS lcl_application DEFINITION.
  PRIVATE SECTION.
    METHODS on_link_click
     FOR EVENT link_click OF cl_salv_events_table
      IMPORTING row column.
ENDCLASS.
CLASS lcl_application IMPLEMENTATION.
  METHOD on_link_click.
    "Handle hotspot click…
  ENDMETHOD.
ENDCLASS.
```

Line selection mode

```
lo_alv->get_selections( )->set_selection_mode(
  if_salv_c_selection_mode=>single ).
```

Variants

```
DATA(lo_layout) = lo_alv->get_layout( ).
"Set layout key to identify the particular ALV
lo_layout->set_key(
  VALUE #( report = sy-repid ) ).
"Remove restriction on saving layouts
lo_layout->set_save_restriction(
  if_salv_c_layout=>restrict_none ).
"Set initial layout
lo_layout set_initial_layout( 'VARIANT_NAME' ).
"Allow setting layouts as default layouts
lo_layout->set_default( abap_true ).
```

Aggregations

```
"Show total for column 'COLUMNNAME'
DATA(lo_aggrs) = lo_alv->get_aggregations( ).
lo_aggrs->add_aggregation(
  columnname  = 'COLUMNNAME'
  aggregation = if_salv_c_aggregation=>total ).
```

Handling toolbar functions

```
"Enable default ALV toolbar functions
lo_alv->get_functions( )->set_default(
  abap_true ).
```

Custom functions require a custom GUI status. Copy status *SALV_STANDARD* of program *SALV_DEMO_** to get up and running.

```
"Register a custom GUI status for an ALV
lo_alv->set_screen_status(
  pfstatus      = 'SALV_STANDARD'
  report        = sy-repid
  set_functions = lo_alv->c_functions_all ).

"Handler method for custom functions
METHODS on_user_command
  FOR EVENT added_function OF cl_salv_events
    IMPORTING !e_salv_function.

METHOD on_user_command.
  CASE e_salv_function.
    WHEN 'MYFUNCTION'.
      "Handle custom function
  ENDCASE.
ENDMETHOD.
```

Coloring list data

```
"Colored cell/row
"Add color column to data structure
TYPES:
  BEGIN OF ty_s_data,
    carrid   TYPE s_carr_id,
    seatsocc TYPE s_seatsocc,
    t_color  TYPE lvc_t_scol,   "Color column
  END OF ty_s_data.

DATA mt_data TYPE TABLE OF ty_s_data.

"Color a particular cell based on a condition
LOOP AT mt_data
  ASSIGNING FIELD-SYMBOL(<s_data>)
  WHERE seatsocc > 300.

  APPEND VALUE #(
    fname = 'SEATSOCC'
    color-col = col_negative
    color-int = 0  "'Intensified' off
    color-inv = 0  "'Inverse' off
  ) TO <s_data>-t_color.
ENDLOOP.

"Color an entire row based on a condition
LOOP AT mt_data ASSIGNING <s_data>
  WHERE seatsocc < 100.
  "Don't specify a field name to color whole row
  APPEND VALUE #( color = col_positive )
    TO <s_data>-t_color.
ENDLOOP.

"Register the color column
lo_alv->get_columns( )->set_color_column(
  'T_COLOR' ).

"Colored column
DATA(lo_col_tab) = CAST cl_salv_column_table(
  lo_alv->get_columns( )->get_column(
    'COLUMNNAME' ) ).

"It's impossible to change color for key fields,
"so unregister as key field when needed
lo_col_tab->set_key( abap_false ).
lo_col_tab->set_color(
  VALUE #( col = col_positive ) ).
```