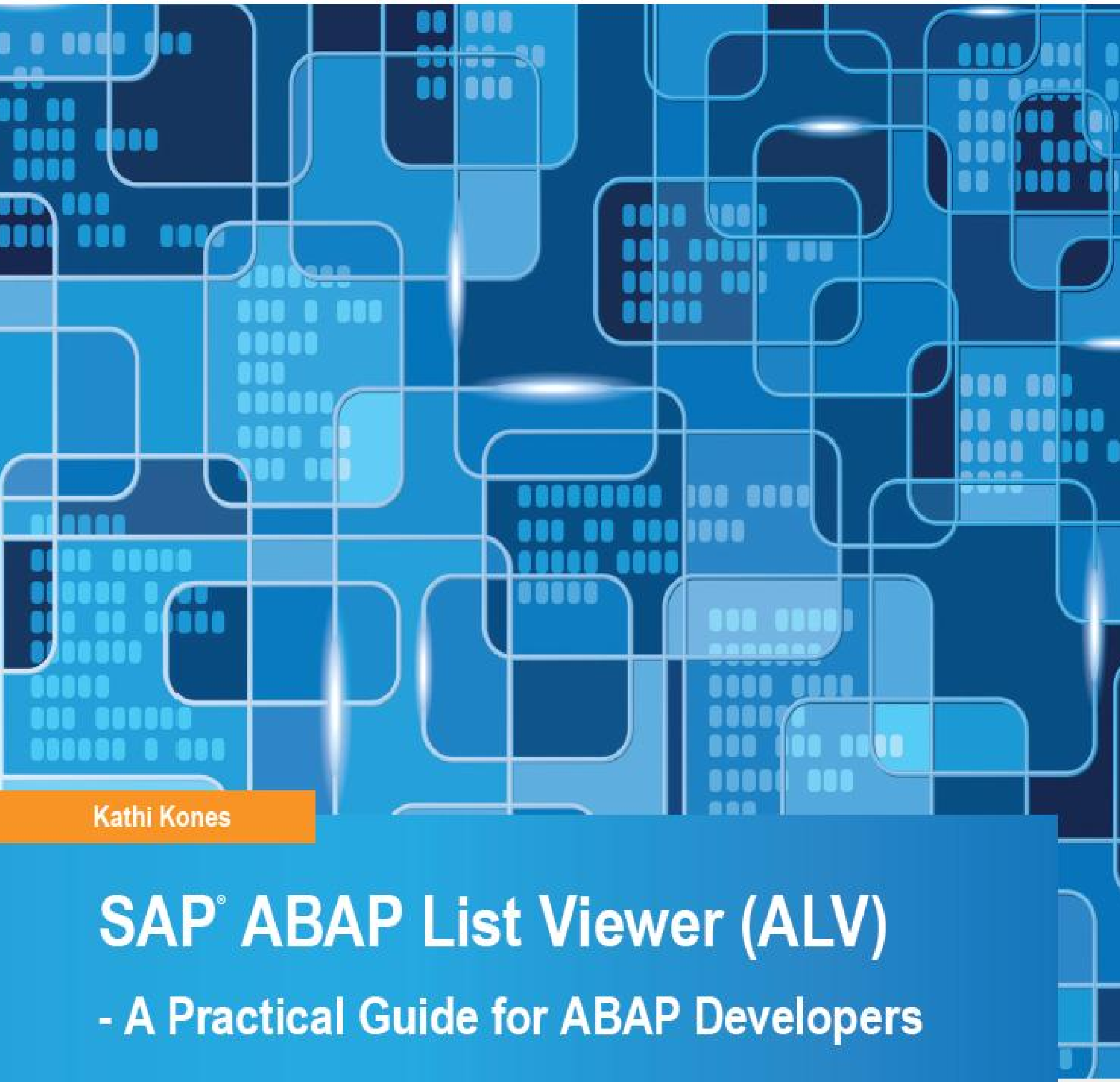
espresso tutorials



Kathi Kones

SAP® ABAP List Viewer (ALV) - A Practical Guide for ABAP Developers

- Learn how to write a basic SAP ALV program
- Get tips on adding sorting and grouping features
- Walk through the control framework and function modules
- Dive into how to add editable fields, events, and layout variants

Table of Contents

Preface 9

1 SAP List Viewer (ALV) types 13

1.1 ALV predecessors 13

1.1.1 Standard lists 13
1.1.2 Dialog-oriented programs 14

1.2 Function module techniques 15

1.2.1 ALV list display function modules 15
1.2.2 ALV grid display function modules 15

1.3 Object-oriented techniques 16

1.3.1 ALV control framework 16
1.3.2 ALV object model 16
1.3.3 ALV with integrated data access 17

1.4 Web Dynpro 17

1.5 Summary 18

2 Writing an ALV program using function modules 19

2.1 Create the ABAP program 19

2.2 Data declarations 20

2.3 Select-Options 22

2.4 Selection of data for ALV output 23

2.5 Main logic section 25

2.6 Building the field catalog table 26

2.7 Calling the ALV function module 29

2.8 Summary 30

3 Writing an ALV program using the ALV control framework 31

3.1 Create the ABAP program 31

3.2 Data declarations 33

3.3 Select-Options 35

3.4 Selection of data for ALV output 36

3.5 Main logic section 38

3.6 Building the field catalog table 38

3.7 Screen call 42

3.8 Process before output (PBO) and process after input (PAI) module logic 45

3.9 PF-status for screen 49

3.10 Custom control on screen 51

3.11 Enabling background execution 56

3.12 Summary 57

4 Adding layout features to an ALV program 59

4.1 Training scenario 59

4.2 Layout features 60

4.2.1 Function module 61

4.2.2 ALV control framework 62

4.3 Alternating shaded and non-shaded lines 64

4.3.1 Function module 64

4.3.2 ALV control framework 65

4.4 Optimizing column widths 66

4.4.1 Function module 66

4.4.2 ALV control framework 67

4.5 Displaying totals at the top 67

4.5.1 Function module 68

4.5.2 ALV control framework 68

4.6 Displaying a title at the top 69

4.6.1 Function module 70

4.6.2 ALV control framework 72

4.7 Previewing layout features 74

4.8 Summary 75

5 Adding sort features to an ALV program 77

5.1 Training scenario 77

5.2 Sort features 78

5.2.1 Function module 79

5.2.2 ALV control framework 81

5.3 Configuring a sort group 82

5.3.1 Function module 82

5.3.2 ALV control framework 84

5.4 Changing the sort field in a sort group 86

5.4.1 Function module 86

5.4.2 ALV control framework 88

5.5 Changing column order to reflect sort order 90

5.5.1 Function module 90

5.5.2 ALV control framework 92

5.6 Configuring a two-level sort 93

5.6.1 Function module 93

5.6.2 ALV control framework 95

5.7 Populating the sort table from the selection screen 97

5.7.1 Function module 98

5.7.2 ALV control framework 101

5.8 Summary 103

6 Adding more features to an ALV program 105

6.1 Passing hidden columns of data 105

6.1.1 Function module 106

6.1.2 ALV control framework 107

6.2 Displaying totals and subtotals immediately 107

6.2.1 Function module 109

6.2.2 ALV control framework 111

6.3 Adding record counts 113

6.3.1 Function module 114

6.3.2 ALV control framework 120

6.4 Handling ALV report layout variants 126

6.4.1 Function module 135

6.4.2 ALV control framework 136

6.5 Adding a top_of_page event and a logo 137

6.5.1 Function module 137

6.5.2 ALV control framework 144

6.6 Adding hotspot logic 154

6.6.1 Function module 154

6.6.2 ALV control framework 162

6.7 Excluding buttons from the ALV application toolbar 170

6.7.1 Function module 170

6.7.2 ALV control framework 173

6.8 Adding buttons to the ALV application toolbar 176

6.8.1 Function module 177

6.8.2 ALV control framework 190

6.9 Summary 193

7 Solving challenges with handy features applicable to many program types 195

7.1 Retrieving the variant name during transaction code se38 background execution 195

7.2 Modifying the selection screen for different user groups 198

7.3 Converting all currency values to a user-specified “report currency” 200

7.4 Summary 207

8 Adding edit capability to an ALV program 209

8.1 Training scenario 209

8.2 Enabling edit based on a selection screen checkbox 211

8.2.1 Function module 213
8.2.2 ALV control framework 222

8.3 Enabling edit using toolbar button 230

8.3.1 Function module 231
8.3.2 ALV control framework 244

8.4 Summary 256

9 Conclusion 258

Appendix 259

Comparison of some report types 259

Resources 259

A The Author 262

B Index 263

C Disclaimer 268

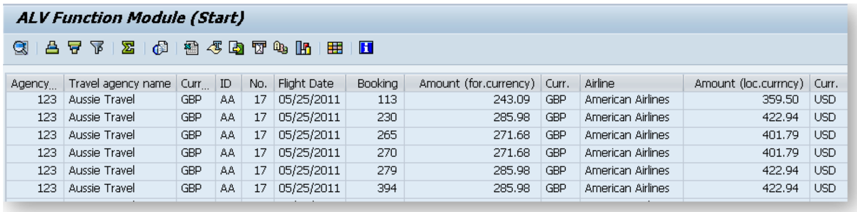
D Credits 269

2 Writing an ALV program using function modules

In this chapter, you'll learn how to write a report using an ALV function module technique, specifically, the REUSE_ALV_GRID_DISPLAY function module. For the training scenario, you'll retrieve data from the SAP Flight Application tables in order to evaluate the amount of income that various travel agencies have generated booking airline flights. The retrieved data will include two currency amounts and three currency keys.

2.1 Create the ABAP program

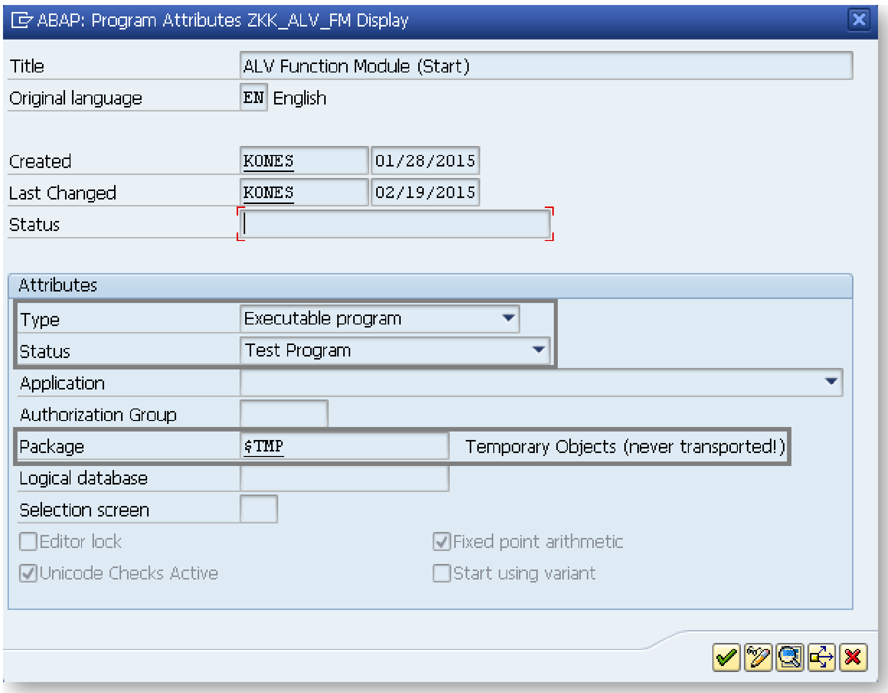
A preview of the ALV output from this initial program is shown in Figure 2.1.


ALV Function Module (Start)

Agency...	Travel agency name	Curr...	ID	No.	Flight Date	Booking	Amount (for.currency)	Curr.	Airline	Amount (loc.currncy)	Curr.
123	Aussie Travel	GBP	AA	17	05/25/2011	113	243.09	GBP	American Airlines	359.50	USD
123	Aussie Travel	GBP	AA	17	05/25/2011	230	285.98	GBP	American Airlines	422.94	USD
123	Aussie Travel	GBP	AA	17	05/25/2011	265	271.68	GBP	American Airlines	401.79	USD
123	Aussie Travel	GBP	AA	17	05/25/2011	270	271.68	GBP	American Airlines	401.79	USD
123	Aussie Travel	GBP	AA	17	05/25/2011	279	285.98	GBP	American Airlines	422.94	USD
123	Aussie Travel	GBP	AA	17	05/25/2011	394	285.98	GBP	American Airlines	422.94	USD

Figure 2.1: Preview (function module – FM)

Using transaction code se38 (or se80, if you prefer), type a name for the new program, then click on the CREATE button. (I have used the name ZKK_ALV_FM for this initial program.) Complete the TYPE and STATUS fields (Figure 2.2), then click on the SAVE button. When prompted for the Package, click on the LOCAL OBJECT button. This fills the Package field with $TMP and positions your cursor in the new program.



ABAP: Program Attributes ZKK_ALV_FM Display

Title	ALV Function Module (Start)	
Original language	EN	English
Created	KONES	01/28/2015
Last Changed	KONES	02/19/2015
Status		

Attributes

Type	Executable program	
Status	Test Program	
Application		
Authorization Group		
Package	$TMP	Temporary Objects (never transported!)
Logical database		
Selection screen		

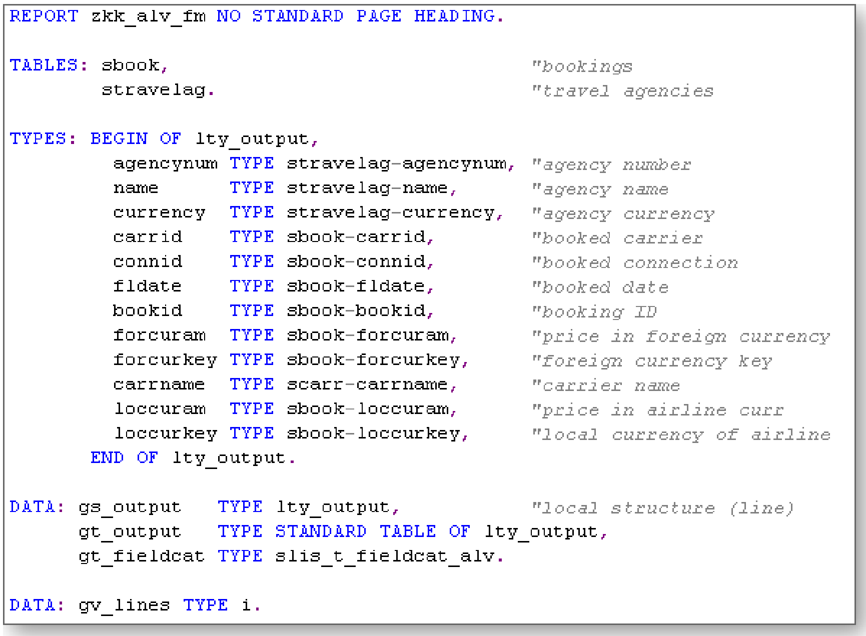
☐ Editor lock ✓ Fixed point arithmetic
✓ Unicode Checks Active ☐ Start using variant

Figure 2.2: Program attributes (FM)

2.2 Data declarations

As shown in Figure 2.3, begin the data declarations section of the program by listing the database tables used in the SELECT-OPTIONS statement: SBOOK and STRAVELAG (Figure 2.4). This will prevent a syntax error.

A local *TYPE* called LTY_OUTPUT lists the fields to be displayed in this ALV. A single-line structure and matching internal tables (GS_OUTPUT and GT_OUTPUT) are declared next, based on the local TYPE LTY_OUTPUT.



```
REPORT zkk_alv_fm NO STANDARD PAGE HEADING.

TABLES: sbook,                                      "bookings
        stravelag.                                  "travel agencies

TYPES: BEGIN OF lty_output,
         agencynum TYPE stravelag-agencynum,  "agency number
         name      TYPE stravelag-name,       "agency name
         currency  TYPE stravelag-currency,   "agency currency
         carrid    TYPE sbook-carrid,         "booked carrier
         connid    TYPE sbook-connid,         "booked connection
         fldate    TYPE sbook-fldate,         "booked date
         bookid    TYPE sbook-bookid,         "booking ID
         forcuram  TYPE sbook-forcuram,       "price in foreign currency
         forcurkey TYPE sbook-forcurkey,      "foreign currency key
         carrname  TYPE scarr-carrname,       "carrier name
         loccuram  TYPE sbook-loccuram,       "price in airline curr
         loccurkey TYPE sbook-loccurkey,      "local currency of airline
       END OF lty_output.

DATA: gs_output   TYPE lty_output,            "local structure (line)
      gt_output   TYPE STANDARD TABLE OF lty_output,
      gt_fieldcat TYPE slis_t_fieldcat_alv.

DATA: gv_lines TYPE i.
```

Figure 2.3: Data declarations (FM)

Local TYPE vs. data dictionary structure

Instead of defining your output structure as a local TYPE in your program, you can define it as a structure in the data dictionary. The technique you use may depend upon your employer's or client's standards and practices, the number of changes you expect to make over time to the output structure, and the ease of making those changes.

Currency keys

Some types of data require a “partner” field for clarity—for instance, currency amounts require currency keys, count and weight amounts require units of measure, and texts that can be stored in multiple languages require language keys. To facilitate troubleshooting and flexibility, we will provide all of the applicable currency keys in the ALV interface. In Chapter 6.1, you will see how you can hide fields on initial display of the ALV.

Referring again to Figure 2.3, you’ll see a global table called GT_FIELDCAT. The *field catalog table* is used to pass information (such as output length or data type) about the fields included in the output structure.

Field catalog table (SLIS_T_FIELDCAT_ALV)

The field catalog table contains information about each of the fields (or columns) in the ALV output. If your structure is not already defined in the data dictionary, you will need to populate this information into the field catalog table yourself. You will see later in this chapter, though, that you can refer to metadata in the data dictionary when populating your field catalog table.

The final data item declared in this simple program is a global variable GV_LINES that will be used to verify that records were found for display using the ALV interface.

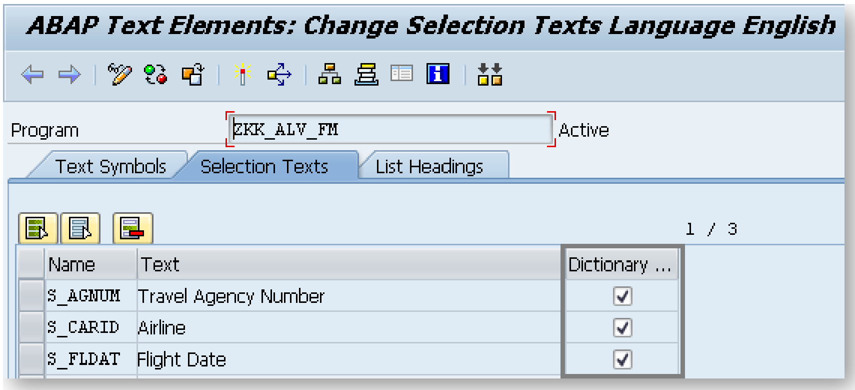
2.3 Select-Options

After the data declarations, type three SELECT-OPTIONS as shown in Figure 2.4. Save, check, and activate your program.

```
SELECT-OPTIONS: s_agnum FOR stravelag-agencynum DEFAULT '123',
                s_carid FOR sbook-carrid,
                s_fldat FOR sbook-fldate.
```

Figure 2.4: Selection-options declaration (FM)

Change the SELECT-OPTIONS labels that will be displayed to the user from question marks to the texts stored in the data dictionary by using the menu path GOTO • TEXT ELEMENTS • SELECTION TEXTS. Check the checkboxes (Figure 2.5). Activate the selection texts, then go back to your program source code.

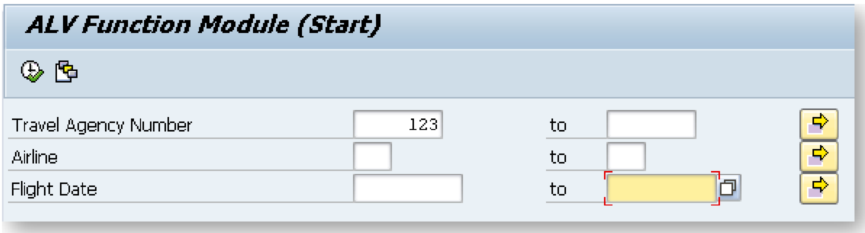

ABAP Text Elements: Change Selection Texts Language English

Program ZKK_ALV_FM Active

Name	Text	Dictionary ...
S_AGNUM	Travel Agency Number	✓
S_CARID	Airline	✓
S_FLDAT	Flight Date	✓

Figure 2.5: Copying selection texts from the data dictionary (FM)

The selection screen should look like Figure 2.6 when done.


ALV Function Module (Start)

Travel Agency Number	123	to	
Airline		to	
Flight Date		to	

Figure 2.6: Selection screen (FM)

2.4 Selection of data for ALV output

We will type placeholders for the INITIALIZATION and the AT SELECTION-SCREEN events, but will leave them empty for this initial program (Figure 2.7).

B Index

- **Bold page numbers** for ALV control framework.
- *Italicized page numbers* for function module.
- Normal page numbers for general references.

&

&data_save *213*, *220*, *241*

A

ABAP 17, 260
ABAP List Viewer 9
add_protocol_entry **226**
add_text **151**
alternate row shading *64*, **65**
ALV 13, 260
ALV control framework **16**, **18**, **31**, **38**, **42**, **64**, **165**, **176**, **230**, **249**, **255**, **258**
ALV object model 16
ALV with IDA 17
ALV wrapper 16
amount field *28*, **41**
application toolbar 127, 170, 176, *182*, **192**, **228**, 230, **244**, **248**, **250**, **253**
assign *219*, *236*, **245**
at selection-screen 23, 36, 197, 198
attributes tab **42**
authority-check 135, 154, 198

B

background 56, 137, 144, 196
butn_type **191**

C

call method **47**
call screen **42**
cell-level editing *213*
CF **64**, 260
cfieldname *28*, **41**, 78, 104, 204
change layout button 74, 105, *116*, *119*, **123**, **126**
checkbox **191**, 198, 211
cl_alv_changed_data_protocol *219*, *236*
cl_dd_area **149**
cl_dd_document **148**
cl_gui_alv_grid **164**, **168**, **174**, **175**
col_pos *91*, **93**
column optimization 66
column order 78
colwidth_optimize *66*
constant 138
convert_to_local_currency 204, 206
count (field catalog) 114, *117*, **123**
count (layout structure) 113, *114*, **121**
countfname *115*, *117*, **122**, **123**
ctabname *28*
currency exchange rate 202, 206
currency key 22, 29, 34, 41, 68, 77, 78, 105, 204

custom container **35**, **47**, **146**
custom control **52**
cwidth_opt **67**

D

data dictionary structure *26*, **39**
data element 210
data_changed *213*, *217*, *218*, **225**, **226**, **229**, *234*, *235*, **244**, **245**, **254**
debugger 45, 171, 174, 178, 249, 250
dialog-oriented program 14
display_document **153**
display_protocol **226**
disvariant 129
do_sum 78, *101*, **103**, 104, 108, *110*, **112**, *117*, **123**, *239*, **251**
docking container **56**
dynamic document **144**, **146**, **149**, **152**

E

e_column_id **164**, **166**, **168**
e_row_id **164**, **168**
e_row_id-index **165**
e_row_id-rowtype **165**, **170**
edit *213*, *216*, **225**, *233*, *239*, **251**, **252**
editable ALV 209
edt_cll_cb *214*, *218*, *233*, *235*
element list tab **35**, **44**, **54**
er_data_changed **225**
event handler **145**, **164**, **225**, **229**
events 137
events table *139*, *156*, 164, *217*, *234*
expa *84*, **86**, 104

F

fcode *171*, *172*
field catalog table *22*, *25*, **34**, **38**
field-symbol *219*, *236*, **245**
flow logic tab **45**
FM *62*, 260
for all entries in *214*, **223**
frame title 201
function key **49**
FUNCTION TEXT *183*

G

get_container **146**
get_frontend_fieldcatalog **250**
get_selected_cells **250**
grid control **16**
grid_title **72**
grid-level editing *213*
group 77, 78, 82, *84*, **86**, 103, 104
GUI 260
GUI status **49**, 176, *177*

H

help (F1) 130
hidden column 105
hidden column (no_out) 105
hotspot 154, *161*, **168**
hotspot_click **164**, **165**, **176**
HTML **144**, **146**, 260

I

i_callback_pf_status_set *157*, *186*, *188*
i_callback_top_of_page *157*
i_callback_user_command *157*
i_grid_settings *213*, *218*, *231*, *233*, *235*
i_list_type *140*
i_structure_name *26*

icon function code 174
ICON NAME *183*
icon text *184*
IDA 17, 260
IDES 260
INFO. TEXT *183*
initialization 23, 36
initialize_document **148**
in-memory 13, 17
input help (F4) 130, 132, 202
integrated data access 17

J

Java 17

K

key points 18, 30, 57, 75, 104, 193, 207, 256
key_hotspot *162*

L

layout feature 60
layout icon **51**
layout variant 60
local file format 60
local type 20, 33
loop at screen 199
lvc_fieldcatalog_merge **39**
lvc_s_cell **252**
lvc_s_col **252**
lvc_s_fcat **38**, **203**
lvc_s_glay *214*, *233*
lvc_s_layo **62**
lvc_s_row **252**
lvc_s_scrl-s_col_info-fieldname *240*
lvc_s_scrl-s_row_info-index *240*
lvc_s_stbl **252**
lvc_t_fcat **34**
lvc_t_sort **81**
lvc_variant_existence_check 134
lvc_variant_f4 134

M

master data 77
mc_evt_enter **229**, **254**
mc_evt_modified **229**, **254**
mc_fc **174**
merge_document **153**
method 16, 25, 38, 165
Microsoft Excel 60, 127, *144*
modif id 198
module pool 14

N

new_line **152**
new_row **152**
no_display 196
no_out 78, 104, *106*, **107**, 203
no_sum 108

O

object navigator 46, 55
object-oriented programming 9, 16, 258
ok_code **35**, **44**, **49**, **51**
OM 260
outputlen *216*, **225**, *233*, *239*, **251**

P

parameter 128, 130, 196, 198, 211
performance trace 24, 37
pf-status **47**, **49**
popup_to_confirm *220*, *241*
popup_to_display_text 209
process after input **45**
process before output **45**

Q

qfieldname 104

R

R/2 platform 13
R/3 platform 13
radio button 98, 101, **191**
ready_for_input **209**
record order 78
release status *15*, *18*, 134
report currency 110, 113, 200
report layout variant 127, 135
repository browser 46
reuse_alv_commentary_write *138*
reuse_alv_events_get *139*, *217*, *234*
reuse_alv_fieldcatalog_merge *26*
reuse_alv_grid_display *15*, *19*, *26*, *29*, *62*, *157*, *239*
reuse_alv_grid_display_lvc *209*
reuse_alv_grid_layout_info_get *238*
reuse_alv_grid_layout_info_get/set *239*
reuse_alv_grid_layout_info_set *238*, *241*
reuse_alv_list_display *15*
reuse_alv_list_layout_info_get/set *239*
reuse_alv_popup_to_select 158, 161, 166
reuse_alv_variant_existence 132, 134
reuse_alv_variant_f4 132, 134
rs_abap_source_scan 47, 260
runtime analysis 24, 37

S

salv 16
SAP 260
SAP Community Network 18, 260
SAP Help Portal 260
SAP List Viewer 13
SAP NetWeaver 13
SAP S/4HANA 13, 17
screen painter **42**, **51**, **52**
screen-active 199
screen-input 199
select single 158, 167
SELECT… UP TO 1 ROWS. ENDSELECT. 158
selection screen block 128, 201
selection text 23, 35
select-options 22, 35, 130
set handler **166**, **192**, **254**
set pf-status *186*, *187*
set_column_style **151**
set_frontend_fieldcatalog **251**
set_row_height **146**
set_scroll_info_via_id **252**
set_table_for_first_display **39**, **47**, **147**, **193**
set_toolbar_interactive **192**, **248**
showicon 176, *183*, 260
single-level sort *93*, **95**
slis_extab *171*
slis_fieldcat_alv *26*, *203*
slis_layout_alv *61*
slis_selfield *158*, *162*, *164*, *220*, *239*
slis_selfield-col_stable *241*
slis_selfield-refresh *241*
slis_selfield-row_stable *241*
slis_selfield-tabindex *158*, *160*, *239*
slis_selfield-value *165*, *242*
slis_t_extab *171*
slis_t_fieldcat_alv *22*
slis_t_sortinfo_alv *79*

sort direction 79
sort feature 78
splitter container **145**, **146**, **193**
spos *83*, **85**
standard list 13
start-of-selection 24, 36
subroutine 25, 38
subtot *84*, **86**, 104
subtotals 68, 77, 78, 93, 103, 108
sy-pfkey *178*
sy-slset 197

T

table control 14
table type 202, *213*, **223**, *231*, **244**
tcurc-waers 202
text symbol 71, 73
tilde ~ 24, 37
toolbar **190**
top_of_list 137
top_of_page 137, **165**, 196
top_of_page text table *140*, **150**
totals 67, 78
totals_bef **68**
totals_before_items *68*
training scenario 59, 69, 77, 78, 93, 105, 108, 127, 154, *170*, **173**, 176, 200, 209, *220*, 230, *241*
transactional data 78
transporting...where *239*
two-level sort *93*, **95**
typ *140*

U

ui_func **174**
ui_functions **174**
UL (underline) 78, 104
user command 154
user_command **190**, *217*

V

validation 130, 131
variant alias 196

W

Web Dynpro 17
window_titlebar *70*
with small heading **74**

Z

zebra *64*, **65**
zebra stripe 64