Example of a Simple ALV Grid Report

```
REPORT  ZTUFI091                                  .
*&---------------------------------------------------------------------*
*& Report  ZDEMO_ALVGRID                                              *
*&                                                                     *
*&---------------------------------------------------------------------*
*&                                                                     *
*& Example of a simple ALV Grid Report                                 *
*& ...................................                                *
*&                                                                     *
*& The basic requirement for this demo is to display a number of       *
*& fields from the EKKO table.                                         *
*&---------------------------------------------------------------------*
*REPORT  zdemo_alvgrid                 .

TABLES:     ekko.

type-pools: slis.                                   "ALV Declarations
*Data Declaration
*----------------
TYPES: BEGIN OF t_ekko,
  ebeln TYPE ekpo-ebeln,
  ebelp TYPE ekpo-ebelp,
  statu TYPE ekpo-statu,
  aedat TYPE ekpo-aedat,
  matnr TYPE ekpo-matnr,
  menge TYPE ekpo-menge,
  meins TYPE ekpo-meins,
  netpr TYPE ekpo-netpr,
  peinh TYPE ekpo-peinh,
 END OF t_ekko.

DATA: it_ekko TYPE STANDARD TABLE OF t_ekko INITIAL SIZE 0,
      wa_ekko TYPE t_ekko.
```

```
*ALV data declarations
data: fieldcatalog type slis_t_fieldcat_alv with header line,
      gd_tab_group type slis_t_sp_group_alv,
      gd_layout    type slis_layout_alv,
      gd_repid     like sy-repid,
      gt_events     type slis_t_event,
      gd_prntparams type slis_print_alv.
**********************************************************************
*Start-of-selection.
START-OF-SELECTION.

perform data_retrieval.
perform build_fieldcatalog.
perform build_layout.
perform build_events.
perform build_print_params.
perform display_alv_report.
*&---------------------------------------------------------------------*
*&      Form  BUILD_FIELDCATALOG
*&---------------------------------------------------------------------*
*       Build Fieldcatalog for ALV Report
*----------------------------------------------------------------------*
form build_fieldcatalog.

* There are a number of ways to create a fieldcat.
* For the purpose of this example i will build the fieldcatalog manualy
* by populating the internal table fields individually and then
* appending the rows. This method can be the most time consuming but can
* also allow you  more control of the final product.

* Beware though, you need to ensure that all fields required are
* populated. When using some of functionality available via ALV, such as
* total. You may need to provide more information than if you were
* simply displaying the result
*               I.e. Field type may be required in-order for
*                    the 'TOTAL' function to work.
```

```
  fieldcatalog-fieldname   = 'EBELN'.
  fieldcatalog-seltext_m   = 'Purchase Order'.
  fieldcatalog-col_pos     = 0.
  fieldcatalog-outputlen   = 10.
  fieldcatalog-emphasize   = 'X'.
  fieldcatalog-key         = 'X'.
*  fieldcatalog-do_sum      = 'X'.
*  fieldcatalog-no_zero     = 'X'.
  append fieldcatalog to fieldcatalog.
  clear  fieldcatalog.

  fieldcatalog-fieldname   = 'EBELP'.
  fieldcatalog-seltext_m   = 'PO Item'.
  fieldcatalog-col_pos     = 1.
  append fieldcatalog to fieldcatalog.
  clear  fieldcatalog.

  fieldcatalog-fieldname   = 'STATU'.
  fieldcatalog-seltext_m   = 'Status'.
  fieldcatalog-col_pos     = 2.
  append fieldcatalog to fieldcatalog.
  clear  fieldcatalog.

  fieldcatalog-fieldname   = 'AEDAT'.
  fieldcatalog-seltext_m   = 'Item change date'.
  fieldcatalog-col_pos     = 3.
  append fieldcatalog to fieldcatalog.
  clear  fieldcatalog.

  fieldcatalog-fieldname   = 'MATNR'.
  fieldcatalog-seltext_m   = 'Material Number'.
  fieldcatalog-col_pos     = 4.
  append fieldcatalog to fieldcatalog.
  clear  fieldcatalog.
```

```
  fieldcatalog-fieldname   = 'MENGE'.
  fieldcatalog-seltext_m   = 'PO quantity'.
  fieldcatalog-col_pos     = 5.
  append fieldcatalog to fieldcatalog.
  clear  fieldcatalog.

  fieldcatalog-fieldname   = 'MEINS'.
  fieldcatalog-seltext_m   = 'Order Unit'.
  fieldcatalog-col_pos     = 6.
  append fieldcatalog to fieldcatalog.
  clear  fieldcatalog.

  fieldcatalog-fieldname   = 'NETPR'.
  fieldcatalog-seltext_m   = 'Net Price'.
  fieldcatalog-col_pos     = 7.
  fieldcatalog-outputlen   = 15.
  fieldcatalog-datatype     = 'CURR'.
  append fieldcatalog to fieldcatalog.
  clear  fieldcatalog.

  fieldcatalog-fieldname   = 'PEINH'.
  fieldcatalog-seltext_m   = 'Price Unit'.
  fieldcatalog-col_pos     = 8.
  append fieldcatalog to fieldcatalog.
  clear  fieldcatalog.
endform.                      " BUILD_FIELDCATALOG


*&---------------------------------------------------------------------*
*&      Form  BUILD_LAYOUT
*&---------------------------------------------------------------------*
*       Build layout for ALV grid report
*----------------------------------------------------------------------*
form build_layout.
  gd_layout-no_input          = 'X'.
  gd_layout-colwidth_optimize = 'X'.
```

```
  gd_layout-totals_text       = 'Totals'(201).
*  gd_layout-totals_only        = 'X'.
*  gd_layout-f2code            = 'DISP'.  "Sets fcode for when double
*                                         "click(press f2)
*  gd_layout-zebra             = 'X'.
*  gd_layout-group_change_edit = 'X'.
*  gd_layout-header_text       = 'hellllo'.
endform.                      " BUILD_LAYOUT



*&---------------------------------------------------------------------*
*&      Form  DISPLAY_ALV_REPORT
*&---------------------------------------------------------------------*
*       Display report using ALV grid
*----------------------------------------------------------------------*
form display_alv_report.
  gd_repid = sy-repid.
  call function 'REUSE_ALV_GRID_DISPLAY'
       exporting
            i_callback_program      = gd_repid
            i_callback_top_of_page   = 'TOP-OF-PAGE'  "see FORM
            i_callback_user_command = 'USER_COMMAND'
*            i_grid_title           = outtext
            is_layout               = gd_layout
            it_fieldcat             = fieldcatalog[]
*            it_special_groups       = gd_tabgroup
            it_events               = gt_events
            is_print                = gd_prntparams
            i_save                  = 'X'
*            is_variant              = z_template
       tables
            t_outtab                = it_ekko
       exceptions
            program_error           = 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.                      " DISPLAY_ALV_REPORT


*&---------------------------------------------------------------------*
*&      Form  DATA_RETRIEVAL
*&---------------------------------------------------------------------*
*       Retrieve data form EKPO table and populate itab it_ekko
*----------------------------------------------------------------------*
form data_retrieval.

select ebeln ebelp statu aedat matnr menge meins netpr peinh
 up to 10 rows
  from ekpo
  into table it_ekko.
endform.                      " DATA_RETRIEVAL


*---------------------------------------------------------------------*
* Form  TOP-OF-PAGE                                                   *
*---------------------------------------------------------------------*
* ALV Report Header                                                   *
*---------------------------------------------------------------------*
Form top-of-page.
*ALV Header declarations
data: t_header type slis_t_listheader,
      wa_header type slis_listheader,
      t_line like wa_header-info,
      ld_lines type i,
      ld_linesc(10) type c.

* Title
  wa_header-typ  = 'H'.
  wa_header-info = 'EKKO Table Report'.
```

```
  append wa_header to t_header.
  clear wa_header.

* Date
  wa_header-typ  = 'S'.
  wa_header-key = 'Date: '.
  CONCATENATE  sy-datum+6(2) '.'
               sy-datum+4(2) '.'
               sy-datum(4) INTO wa_header-info.   "todays date
  append wa_header to t_header.
  clear: wa_header.

* Total No. of Records Selected
  describe table it_ekko lines ld_lines.
  ld_linesc = ld_lines.
  concatenate 'Total No. of Records Selected: ' ld_linesc
                   into t_line separated by space.
  wa_header-typ  = 'A'.
  wa_header-info = t_line.
  append wa_header to t_header.
  clear: wa_header, t_line.

  call function 'REUSE_ALV_COMMENTARY_WRITE'
       exporting
            it_list_commentary = t_header.
*            i_logo             = 'Z_LOGO'.
endform.


*---------------------------------------------------------------------*
*       FORM USER_COMMAND                                             *
*---------------------------------------------------------------------*
*       --> R_UCOMM                                                   *
*       --> RS_SELFIELD                                               *
*---------------------------------------------------------------------*
FORM user_command USING r_ucomm LIKE sy-ucomm
```

```
                    rs_selfield TYPE slis_selfield.

* Check function code
  CASE r_ucomm.
    WHEN '&IC1'.
*   Check field clicked on within ALVgrid report
    IF rs_selfield-fieldname = 'EBELN'.
*     Read data table, using index of row user clicked on
      READ TABLE it_ekko INTO wa_ekko INDEX rs_selfield-tabindex.
*     Set parameter ID for transaction screen field
      SET PARAMETER ID 'BES' FIELD wa_ekko-ebeln.
*     Sxecute transaction ME23N, and skip initial data entry screen
      CALL TRANSACTION 'ME23N' AND SKIP FIRST SCREEN.
    ENDIF.
  ENDCASE.
ENDFORM.



*&---------------------------------------------------------------------*
*&      Form  BUILD_EVENTS
*&---------------------------------------------------------------------*
*       Build events table
*----------------------------------------------------------------------*
form build_events.
  data: ls_event type slis_alv_event.

  call function 'REUSE_ALV_EVENTS_GET'
       exporting
            i_list_type = 0
       importing
            et_events   = gt_events[].
  read table gt_events with key name =  slis_ev_end_of_page
                            into ls_event.
  if sy-subrc = 0.
    move 'END_OF_PAGE' to ls_event-form.
    append ls_event to gt_events.
```

```
  endif.

    read table gt_events with key name =  slis_ev_end_of_list
                            into ls_event.
  if sy-subrc = 0.
    move 'END_OF_LIST' to ls_event-form.
    append ls_event to gt_events.
  endif.
endform.                      " BUILD_EVENTS


*&---------------------------------------------------------------------*
*&      Form  BUILD_PRINT_PARAMS
*&---------------------------------------------------------------------*
*       Setup print parameters
*----------------------------------------------------------------------*
form build_print_params.
  gd_prntparams-reserve_lines = '3'.   "Lines reserved for footer
  gd_prntparams-no_coverpage = 'X'.
endform.                      " BUILD_PRINT_PARAMS


*&---------------------------------------------------------------------*
*&      Form  END_OF_PAGE
*&---------------------------------------------------------------------*
form END_OF_PAGE.
  data: listwidth type i,
        ld_pagepos(10) type c,
        ld_page(10)    type c.

  write: sy-uline(50).
  skip.
  write:/40 'Page:', sy-pagno .
endform.
```

```
*&---------------------------------------------------------------------*
*&      Form  END_OF_LIST
*&---------------------------------------------------------------------*
form END_OF_LIST.
  data: listwidth type i,
        ld_pagepos(10) type c,
        ld_page(10)    type c.

  skip.
  write:/40 'Page:', sy-pagno .
endform .
```

ABAP Example Program ALV Grid Control

ABAP Example Program ALV Grid Control

You need to create a screen 100 for calling it, and in the Element list of the sceen supply OK_CODE of type OK & in the layout, place a Custom - control with name DILEEP_TEST1.

Then activate modules STATUS_0100 and USER_COMMAND_0100 in the flow logic .

```
REPORT ZTEST_DIL4 .
TABLES ZMSTKSUM.
DATA : OK_CODE LIKE SY-UCOMM,
       TAB_DISPLAY TYPE TABLE OF ZMSTKSUM,
       C_CONTAINER TYPE SCRFNAME VALUE 'DILEEP_TEST1',
       ALV_GRID TYPE REF TO CL_GUI_ALV_GRID,
       C_CUSTOM_CONTAINER TYPE REF TO CL_GUI_CUSTOM_CONTAINER.


SELECTION-SCREEN BEGIN OF BLOCK B1 WITH FRAME TITLE TEXT-001.

SELECT-OPTIONS : S_WERKS FOR ZMSTKSUM-WERKS.

SELECTION-SCREEN END OF BLOCK B1.

SELECT * FROM ZMSTKSUM INTO TABLE TAB_DISPLAY WHERE WERKS IN S_WERKS.
```

```
CALL SCREEN 100.
*&---------------------------------------------------------------------

*&      Module  STATUS_0100  OUTPUT
*&---------------------------------------------------------------------

*       text
*---------------------------------------------------------------------

MODULE STATUS_0100 OUTPUT.
  SET PF-STATUS 'MAIN'.
*  SET TITLEBAR 'xxx'.
  IF C_CUSTOM_CONTAINER IS INITIAL.
CREATE OBJECT C_CUSTOM_CONTAINER EXPORTING CONTAINER_NAME = C_CONTAINER
.
    CREATE OBJECT ALV_GRID EXPORTING I_PARENT = C_CUSTOM_CONTAINER.

    CALL METHOD ALV_GRID->SET_TABLE_FOR_FIRST_DISPLAY
      EXPORTING
*    I_BUFFER_ACTIVE               =
*    I_BYPASSING_BUFFER            =
        I_STRUCTURE_NAME              = 'ZMSTKSUM'
*    IS_VARIANT                    =
*    I_SAVE                        =
*    I_DEFAULT                     = 'X'
*    IS_LAYOUT                     =
*    IS_PRINT                      =
*    IT_SPECIAL_GROUPS             =
*    IT_TOOLBAR_EXCLUDING          =
      CHANGING
        IT_OUTTAB                     = TAB_DISPLAY
*    IT_FIELDCATALOG               =
*    IT_SORT                       =
*    IT_FILTER                     =
```

```
*  EXCEPTIONS
*    INVALID_PARAMETER_COMBINATION = 1
*    PROGRAM_ERROR                 = 2
*    others                        = 3
        .
    IF SY-SUBRC <> 0.
* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
*            WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
    ENDIF.

  ENDIF.
ENDMODULE.                    " STATUS_0100  OUTPUT

*&---------------------------------------------------------------------

*&      Module  USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------

*       text
*----------------------------------------------------------------------

MODULE USER_COMMAND_0100 INPUT.
  CASE OK_CODE.
    WHEN 'EXIT'.
      LEAVE PROGRAM.

  ENDCASE.
ENDMODULE.                    " USER_COMMAND_0100  INPUT
```

An Interactive ALV Report

An Interactive ALV Report

```
*&----------------------------------------------------------------------*
*& Report  ZZ_22038_22098_002                                           *
*&                                                                      *
*&----------------------------------------------------------------------*
*& This is an Interactive ALV report, where on line slection we can see
```

```
*&  the secondry list
*&
*&                                                                    *
*&--------------------------------------------------------------------*

REPORT  ZZ_22038_22098_002 NO STANDARD PAGE HEADING LINE-SIZE 650
MESSAGE-ID ZZ_9838                      .

TYPE-POOLS: SLIS.
*type declaration for values from ekko
TYPES: BEGIN OF I_EKKO,
       EBELN LIKE EKKO-EBELN,
       AEDAT LIKE EKKO-AEDAT,
       BUKRS LIKE EKKO-BUKRS,
       BSART LIKE EKKO-BSART,
       LIFNR LIKE EKKO-LIFNR,
       END OF I_EKKO.

DATA: IT_EKKO TYPE STANDARD TABLE OF I_EKKO INITIAL SIZE 0,
      WA_EKKO TYPE I_EKKO.

*type declaration for values from ekpo
TYPES: BEGIN OF I_EKPO,
       EBELN LIKE EKPO-EBELN,
       EBELP LIKE EKPO-EBELP,
       MATNR LIKE EKPO-MATNR,
       MENGE LIKE EKPO-MENGE,
       MEINS LIKE EKPO-MEINS,
       NETPR LIKE EKPO-NETPR,
       END OF I_EKPO.

DATA: IT_EKPO TYPE STANDARD TABLE OF I_EKPO INITIAL SIZE 0,
      WA_EKPO TYPE I_EKPO .

*variable for Report ID
DATA: V_REPID LIKE SY-REPID .
```

```
*declaration for fieldcatalog
DATA: I_FIELDCAT TYPE SLIS_T_FIELDCAT_ALV,
      WA_FIELDCAT TYPE SLIS_FIELDCAT_ALV.



DATA: IT_LISTHEADER TYPE SLIS_T_LISTHEADER.

* declaration for events table where user comand or set PF status will
* be defined
DATA: V_EVENTS TYPE SLIS_T_EVENT,
      WA_EVENT TYPE SLIS_ALV_EVENT.

* declartion for layout
DATA: ALV_LAYOUT TYPE SLIS_LAYOUT_ALV.

* declaration for variant(type of display we want)
DATA: I_VARIANT TYPE DISVARIANT,
      I_VARIANT1 TYPE DISVARIANT,
      I_SAVE(1) TYPE C.

*PARAMETERS : p_var TYPE disvariant-variant.

*Title displayed when the alv list is displayed
DATA:  I_TITLE_EKKO TYPE LVC_TITLE VALUE 'FIRST LIST DISPLAYED'.
DATA:  I_TITLE_EKPO TYPE LVC_TITLE VALUE 'SECONDRY LIST DISPLAYED'.

INITIALIZATION.
  V_REPID = SY-REPID.
  PERFORM BUILD_FIELDCATLOG.
  PERFORM EVENT_CALL.
  PERFORM POPULATE_EVENT.

START-OF-SELECTION.
  PERFORM DATA_RETRIEVAL.
  PERFORM BUILD_LISTHEADER USING IT_LISTHEADER.
```

```
  PERFORM DISPLAY_ALV_REPORT.

*&---------------------------------------------------------------------*
*&      Form  BUILD_FIELDCATLOG
*&---------------------------------------------------------------------*
*       Fieldcatalog has all the field details from ekko
*----------------------------------------------------------------------*
FORM BUILD_FIELDCATLOG.
  WA_FIELDCAT-TABNAME = 'IT_EKKO'.
  WA_FIELDCAT-FIELDNAME = 'EBELN'.
  WA_FIELDCAT-SELTEXT_M = 'PO NO.'.
  APPEND WA_FIELDCAT TO I_FIELDCAT.
  CLEAR WA_FIELDCAT.

  WA_FIELDCAT-TABNAME = 'IT_EKKO'.
  WA_FIELDCAT-FIELDNAME = 'AEDAT'.
  WA_FIELDCAT-SELTEXT_M = 'DATE.'.
  APPEND WA_FIELDCAT TO I_FIELDCAT.
  CLEAR WA_FIELDCAT.

  WA_FIELDCAT-TABNAME = 'IT_EKKO'.
  WA_FIELDCAT-FIELDNAME = 'BUKRS'.
  WA_FIELDCAT-SELTEXT_M = 'COMPANY CODE'.
  APPEND WA_FIELDCAT TO I_FIELDCAT.
  CLEAR WA_FIELDCAT.

WA_FIELDCAT-TABNAME = 'IT_EKKO'.
  WA_FIELDCAT-FIELDNAME = 'BUKRS'.
  WA_FIELDCAT-SELTEXT_M = 'DOCMENT TYPE'.
  APPEND WA_FIELDCAT TO I_FIELDCAT.
  CLEAR WA_FIELDCAT.

WA_FIELDCAT-TABNAME = 'IT_EKKO'.
  WA_FIELDCAT-FIELDNAME = 'LIFNR'.
  WA_FIELDCAT-NO_OUT    = 'X'.
  WA_FIELDCAT-SELTEXT_M = 'VENDOR CODE'.
```

```
  APPEND WA_FIELDCAT TO I_FIELDCAT.
  CLEAR WA_FIELDCAT.


ENDFORM.                      "BUILD_FIELDCATLOG

*&---------------------------------------------------------------------*
*&      Form  EVENT_CALL
*&---------------------------------------------------------------------*
*   we get all events - TOP OF PAGE or USER COMMAND in table v_events
*----------------------------------------------------------------------*
FORM EVENT_CALL.
  CALL FUNCTION 'REUSE_ALV_EVENTS_GET'
   EXPORTING
     I_LIST_TYPE           = 0
   IMPORTING
     ET_EVENTS             = V_EVENTS
*  EXCEPTIONS
*    LIST_TYPE_WRONG       = 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.                      "EVENT_CALL

*&---------------------------------------------------------------------*
*&      Form  POPULATE_EVENT
*&---------------------------------------------------------------------*
*      Events populated for TOP OF PAGE & USER COMAND
*----------------------------------------------------------------------*
FORM POPULATE_EVENT.
  READ TABLE V_EVENTS INTO WA_EVENT WITH KEY NAME = 'TOP_OF_PAGE'.
  IF SY-SUBRC EQ 0.
    WA_EVENT-FORM = 'TOP_OF_PAGE'.
```

```
    MODIFY V_EVENTS FROM WA_EVENT TRANSPORTING FORM WHERE NAME =
WA_EVENT-FORM.
  ENDIF.

  READ TABLE V_EVENTS INTO WA_EVENT WITH KEY NAME = 'USER_COMMAND'.
  IF SY-SUBRC EQ 0.
    WA_EVENT-FORM = 'USER_COMMAND'.
    MODIFY V_EVENTS FROM WA_EVENT TRANSPORTING FORM WHERE NAME =
WA_EVENT-NAME.
  ENDIF.
ENDFORM.                      "POPULATE_EVENT



*&---------------------------------------------------------------------*
*&      Form  data_retrieval
*&---------------------------------------------------------------------*
*   retreiving values from the database table ekko
*----------------------------------------------------------------------*
FORM DATA_RETRIEVAL.
  SELECT EBELN AEDAT BUKRS BSART LIFNR FROM EKKO INTO TABLE IT_EKKO.

ENDFORM.                      "data_retrieval
*&---------------------------------------------------------------------*
*&      Form  bUild_listheader
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->I_LISTHEADEtext
*----------------------------------------------------------------------*
FORM BUILD_LISTHEADER USING I_LISTHEADER TYPE SLIS_T_LISTHEADER.
  DATA HLINE TYPE SLIS_LISTHEADER.
  HLINE-INFO = 'this is my first alv pgm'.
  HLINE-TYP = 'H'.
ENDFORM.                      "build_listheader

*&---------------------------------------------------------------------*
```

```
*&      Form  display_alv_report
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM DISPLAY_ALV_REPORT.
  V_REPID = SY-REPID.
  CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
   EXPORTING
     I_CALLBACK_PROGRAM                = V_REPID
*   I_CALLBACK_PF_STATUS_SET          = ' '
     I_CALLBACK_USER_COMMAND           = 'USER_COMMAND'
     I_CALLBACK_TOP_OF_PAGE            = 'TOP_OF_PAGE'
     I_GRID_TITLE                      = I_TITLE_EKKO
*   I_GRID_SETTINGS                   =
*   IS_LAYOUT                         = ALV_LAYOUT
     IT_FIELDCAT                       = I_FIELDCAT[]
*   IT_EXCLUDING                      =
*   IT_SPECIAL_GROUPS                 =
*   IT_SORT                           =
*   IT_FILTER                         =
*   IS_SEL_HIDE                       =
*     i_default                         = 'ZLAY1'
     I_SAVE                            = 'A'
*     is_variant                        = i_variant
     IT_EVENTS                         = V_EVENTS
    TABLES
      T_OUTTAB                         = IT_EKKO
* EXCEPTIONS
*   PROGRAM_ERROR                     = 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.                    "display_alv_report
```

```
*&---------------------------------------------------------------------*
*&      Form  TOP_OF_PAGE
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM TOP_OF_PAGE.
  CALL FUNCTION 'REUSE_ALV_COMMENTARY_WRITE'
    EXPORTING
      IT_LIST_COMMENTARY       = IT_LISTHEADER
*    i_logo                   =
*    I_END_OF_LIST_GRID       =
            .

ENDFORM.                     "TOP_OF_PAGE

*&---------------------------------------------------------------------*
*&      Form  USER_COMMAND
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->R_UCOMM    text
*      -->,          text
*      -->RS_SLEFIELDtext
*----------------------------------------------------------------------*
FORM USER_COMMAND USING R_UCOMM LIKE SY-UCOMM
RS_SELFIELD TYPE SLIS_SELFIELD.
  CASE R_UCOMM.
    WHEN '&IC1'.
      READ TABLE IT_EKKO INTO WA_EKKO INDEX RS_SELFIELD-TABINDEX.
      PERFORM BUILD_FIELDCATLOG_EKPO.
      PERFORM EVENT_CALL_EKPO.
      PERFORM POPULATE_EVENT_EKPO.
```

```
      PERFORM DATA_RETRIEVAL_EKPO.
      PERFORM BUILD_LISTHEADER_EKPO USING IT_LISTHEADER.
      PERFORM DISPLAY_ALV_EKPO.
  ENDCASE.
ENDFORM.                    "user_command
*&---------------------------------------------------------------------*
*&      Form  BUILD_FIELDCATLOG_EKPO
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM BUILD_FIELDCATLOG_EKPO.

  WA_FIELDCAT-TABNAME = 'IT_EKPO'.
  WA_FIELDCAT-FIELDNAME = 'EBELN'.
  WA_FIELDCAT-SELTEXT_M = 'PO NO.'.
  APPEND WA_FIELDCAT TO I_FIELDCAT.
  CLEAR WA_FIELDCAT.
  WA_FIELDCAT-TABNAME = 'IT_EKPO'.
  WA_FIELDCAT-FIELDNAME = 'EBELP'.
  WA_FIELDCAT-SELTEXT_M = 'LINE NO'.
  APPEND WA_FIELDCAT TO I_FIELDCAT.
  CLEAR WA_FIELDCAT.
  WA_FIELDCAT-TABNAME = 'I_EKPO'.
  WA_FIELDCAT-FIELDNAME = 'MATNR'.
  WA_FIELDCAT-SELTEXT_M = 'MATERIAL NO.'.
  APPEND WA_FIELDCAT TO I_FIELDCAT.
  CLEAR WA_FIELDCAT.
WA_FIELDCAT-TABNAME = 'I_EKPO'.
  WA_FIELDCAT-FIELDNAME = 'MENGE'.
  WA_FIELDCAT-SELTEXT_M = 'QUANTITY'.
  APPEND WA_FIELDCAT TO I_FIELDCAT.
  CLEAR WA_FIELDCAT.

WA_FIELDCAT-TABNAME = 'I_EKPO'.
  WA_FIELDCAT-FIELDNAME = 'MEINS'.
  WA_FIELDCAT-SELTEXT_M = 'UOM'.
```

```
  APPEND WA_FIELDCAT TO I_FIELDCAT.
  CLEAR WA_FIELDCAT.

WA_FIELDCAT-TABNAME = 'I_EKPO'.
  WA_FIELDCAT-FIELDNAME = 'NETPR'.
  WA_FIELDCAT-SELTEXT_M = 'PRICE'.
  APPEND WA_FIELDCAT TO I_FIELDCAT.
  CLEAR WA_FIELDCAT.


ENDFORM.                      "BUILD_FIELDCATLOG_EKPO

*&---------------------------------------------------------------------*
*&      Form  event_call_ekpo
*&---------------------------------------------------------------------*
*   we get all events - TOP OF PAGE or USER COMMAND in table v_events
*----------------------------------------------------------------------*
FORM EVENT_CALL_EKPO.
  CALL FUNCTION 'REUSE_ALV_EVENTS_GET'
   EXPORTING
     I_LIST_TYPE           = 0
   IMPORTING
     ET_EVENTS             = V_EVENTS
* EXCEPTIONS
*   LIST_TYPE_WRONG       = 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.                      "event_call_ekpo


*&---------------------------------------------------------------------*
*&      Form  POPULATE_EVENT
```

```
*&---------------------------------------------------------------------*
*        Events populated for TOP OF PAGE & USER COMAND
*---------------------------------------------------------------------*
FORM POPULATE_EVENT_EKPO.
  READ TABLE V_EVENTS INTO WA_EVENT WITH KEY NAME = 'TOP_OF_PAGE'.
  IF SY-SUBRC EQ 0.
    WA_EVENT-FORM = 'TOP_OF_PAGE'.
    MODIFY V_EVENTS FROM WA_EVENT TRANSPORTING FORM WHERE NAME =
WA_EVENT-FORM.
  ENDIF.

  ENDFORM.                      "POPULATE_EVENT

*&---------------------------------------------------------------------*
*&      Form  TOP_OF_PAGE
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
FORM F_TOP_OF_PAGE.
  CALL FUNCTION 'REUSE_ALV_COMMENTARY_WRITE'
    EXPORTING
      IT_LIST_COMMENTARY       = IT_LISTHEADER
*    i_logo                   =
*    I_END_OF_LIST_GRID       =
             .

ENDFORM.                      "TOP_OF_PAGE

*&---------------------------------------------------------------------*
*&      Form  USER_COMMAND
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
*      -->R_UCOMM    text
*      -->,          text
*      -->RS_SLEFIELDtext
```

```
*---------------------------------------------------------------------*

*retreiving values from the database table ekko
FORM DATA_RETRIEVAL_EKPO.
SELECT EBELN EBELP MATNR MENGE MEINS NETPR FROM EKPO INTO TABLE IT_EKPO.
ENDFORM.

FORM BUILD_LISTHEADER_EKPO USING I_LISTHEADER TYPE SLIS_T_LISTHEADER.
DATA: HLINE1 TYPE SLIS_LISTHEADER.
HLINE1-TYP = 'H'.
HLINE1-INFO = 'CHECKING PGM'.
ENDFORM.



FORM DISPLAY_ALV_EKPO.
CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
 EXPORTING
*   I_INTERFACE_CHECK                 = ' '
*   I_BYPASSING_BUFFER                = ' '
*   I_BUFFER_ACTIVE                   = ' '
   I_CALLBACK_PROGRAM                = V_REPID
*   I_CALLBACK_PF_STATUS_SET          = ' '
*   I_CALLBACK_USER_COMMAND           = 'F_USER_COMMAND'
   I_CALLBACK_TOP_OF_PAGE            = 'TOP_OF_PAGE'
*   I_CALLBACK_HTML_TOP_OF_PAGE       = ' '
*   I_CALLBACK_HTML_END_OF_LIST       = ' '
*   I_STRUCTURE_NAME                  =
*   I_BACKGROUND_ID                   = ' '
   I_GRID_TITLE                      = I_TITLE_EKPO
*   I_GRID_SETTINGS                   =
*   IS_LAYOUT                         =
   IT_FIELDCAT                       = I_FIELDCAT[]
*   IT_EXCLUDING                      =
*   IT_SPECIAL_GROUPS                 =
*   IT_SORT                           =
*   IT_FILTER                         =
```

```
*   IS_SEL_HIDE                          =
*   I_DEFAULT                            =
   I_SAVE                               = 'A'
*   IS_VARIANT                           =
   IT_EVENTS                            = V_EVENTS
  TABLES
    T_OUTTAB                            = IT_EKPO
 EXCEPTIONS
   PROGRAM_ERROR                        = 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 .
```

ALV Reporting - REUSE_ALV_BLOCK_LIST_DISPLAY

ALV Reporting - REUSE_ALV_BLOCK_LIST_DISPLAY

```
REPORT z_alv_list_block.
TYPE-POOLS: slis. " ALV Global types
SELECTION-SCREEN :
SKIP,
BEGIN OF LINE,
COMMENT 5(27) v_1 FOR FIELD p_max. "#EC NEEDED
PARAMETERS p_max(2) TYPE n DEFAULT '02' OBLIGATORY.
SELECTION-SCREEN END OF LINE.
DATA:
* 1st Table
BEGIN OF gt_kna1 OCCURS 0, " Data displayed
kunnr LIKE kna1-kunnr, " Customer number
ernam LIKE kna1-ernam, " Name of Person who Created
erdat LIKE kna1-erdat, " Creation date
name1 LIKE kna1-name1, " Name 1
END OF gt_kna1,
* 2nd Table
```

```
BEGIN OF gt_mara OCCURS 0,
ernam LIKE mara-ernam, " Name of Person who Created
matnr LIKE mara-matnr, " Material number
ersda LIKE mara-e rsda, " Creation date
brgew LIKE mara-brgew, " Gross weight
END OF gt_mara,
* 3rd Table
BEGIN OF gt_vbak OCCURS 0,
vkorg LIKE vbak-vkorg, " Sales organization
kunnr LIKE vbak-kunnr, " Sold-to party
vbeln LIKE vbak-vbeln, " Sales document
netwr LIKE vbak-netwr, " Net Value of the Sales Order
waerk LIKE vbak-waerk, " SD document currency
END OF gt_vbak.
*---------------------------------------------------------------------*
INITIALIZATION.
v_1 = 'Maximum of records to read'.
*---------------------------------------------------------------------*
START-OF-SELECTION.
* Read data
SELECT * FROM kna1
UP TO p_max ROWS
INTO CORRESPONDING FIELDS OF TABLE gt_kna1.
SELECT * FROM mara
UP TO p_max ROWS
INTO CORRESPONDING FIELDS OF TABLE gt_mara.
SELECT * FROM vbak
UP TO p_max ROWS
INTO CORRESPONDING FIELDS OF TABLE gt_vbak.
CALL FUNCTION 'REUSE_ALV_BLOCK_LIST_INIT'
EXPORTING
i_callback_program = sy-cprog
i_callback_user_command = 'USER_COMMAND'.
PERFORM list_append TABLES gt_kna1
USING '1'
'GT_KNA1'.
```

```
PERFORM list_append TABLES gt_mara
USING '2'
'GT_MARA'.
PERFORM list_append TABLES gt_vbak
USING '3'
'GT_VBAK'.
PERFORM f_list_display.
*---------------------------------------------------------------------*
* FORM USER_COMMAND *
*---------------------------------------------------------------------*
FORM user_command USING i_ucomm LIKE sy-ucomm
is_selfield TYPE slis_selfield. "#EC CALLED
CASE i_ucomm.
WHEN '&IC1'. " Pick
CASE is_selfield-tabname.
WHEN 'GT_MARA'.
WHEN 'GT_KNA1'.
WHEN 'GT_VBAK'.
READ TABLE gt_vbak INDEX is_selfield-tabindex.
IF sy-subrc EQ 0.
* Sales order number
SET PARAMETER ID 'AUN' FIELD gt_vbak-vbeln.
* Display Sales Order
CALL TRANSACTION 'VA03' AND SKIP FIRST SCREEN.
ENDIF.
ENDCASE.
ENDCASE.
ENDFORM. " USER_COMMAND
*---------------------------------------------------------------------*< /div>
* Form list_append
*---------------------------------------------------------------------*
FORM list_append TABLES ut_table
USING u_no TYPE char1
u_tabname TYPE slis_tabname.
* Macro definition
DEFINE m_fieldcat.
```

```
ls_fieldcat-fieldname = &1.
ls_fieldcat-ref_tabname = &2.
append ls_fieldcat to lt_fieldcat.
END-OF-DEFINITION.
DEFINE m_sort.
ls_sort-fieldname = &1.
ls_sort-up = 'X'.
append ls_sort to lt_sort.
END-OF-DEFINITION.
DATA :
ls_fieldcat TYPE slis_fieldcat_alv,
lt_fieldcat TYPE slis_t_fieldcat_alv, " Field catalog
ls_sort TYPE slis_sortinfo_alv,
lt_sort TYPE slis_t_sortinfo_alv. " Sort table
DATA:
lt_events TYPE slis_t_event,
ls_event TYPE slis_alv_event,
ls_layout TYPE slis_layout_alv.
ls_layout-group_change_edit = 'X'.
ls_layout-colwidth_optimize = 'X'.
ls_layout-zebra = 'X'.
ls_layout-detail_popup = 'X'.
ls_layout-get_selinfos = 'X'.
ls_layout-max_linesize = '200'.
CASE u_no.
WHEN '1'.
* Build field catalog and sort table
m_fieldcat 'KUNNR' 'KNA1'.
m_fieldcat 'ERNAM' 'KNA1'.
m_fieldcat 'ERDAT' 'KNA1'.
m_fieldcat 'NAME1' 'KNA1'.
m_sort 'KUNNR'.
WHEN '2'.
m_fieldcat 'MATNR' 'MARA'.
m_fieldcat 'ERNAM' 'MARA'.
m_fieldcat 'ERSDA' 'MARA'.
```

```
m_fieldcat 'BRGEW' 'MARA'.
m_sort 'MATNR'.
WHEN '3'.
m_fieldcat 'VBELN' 'VBAK'.
m_fieldcat 'VKORG' 'VBAK'.
m_fieldcat 'KUNNR' 'VBAK'.
m_fieldcat 'NETWR' 'VBAK'.
m_fieldcat 'WAERK' 'VBAK'.
m_sort 'VBELN'.
ENDCASE.
IF u_no CA '13'.
MOVE 'TOP_OF_PAGE' TO ls_event-name.
CONCATENATE 'TOP_OF_PAGE' u_no INTO ls_event-form.
APPEND ls_event TO lt_events.
ELSE.
MOVE 'TOP_OF_LIST' TO ls_event-name.
CONCATENATE 'TOP_OF_LIST' u_no INTO ls_event-form.
APPEND ls_event TO lt_events.
ENDIF.
CALL FUNCTION 'REUSE_ALV_BLOCK_LIST_APPEND'
EXPORTING
it_fieldcat = lt_fieldcat
is_layout = ls_layout
i_tabname = u_tabname
it_events = lt_events
it_sort = lt_sort
* i_text =
TABLES
t_outtab = ut_table
EXCEPTIONS
program_error = 1
maximum_of_appends_reached = 2
OTHERS = 3.
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. " LIST_APPEND
*---------------------------------------------------------------------*
* Form f_list_display
*---------------------------------------------------------------------*
FORM f_list_display.
DATA ls_print TYPE slis_print_alv.
ls_print-no_print_selinfos = 'X'. " Display no selection infos
ls_print-no_print_listinfos = 'X'. " Display no listinfos
ls_print-reserve_lines = 2. " Lines reserved for end of page
CALL FUNCTION 'REUSE_ALV_BLOCK_LIST_DISPLAY'
EXPORTING
i_interface_check = ' '
is_print = ls_print
EXCEPTIONS
program_error = 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. " F_LIST_DISPLAY
*---------------------------------------------------------------------*
* FORM top_of_page1 *
*---------------------------------------------------------------------*
FORM top_of_page1. "#EC CALLED
PERFORM top_of_page.
ENDFORM.
*---------------------------------------------------------------------*
* FORM top_of_page3 *
*---------------------------------------------------------------------*
FORM top_of_page3. "#EC CALLED
PERFORM top_of_page.
ENDFORM.
*---------------------------------------------------------------------*
* FORM top_of_page *
```

```
*---------------------------------------------------------------------*
FORM top_of_page.
ULINE.
WRITE : sy-uname, sy-title(56) CENTERED, sy-datum.
ULINE.
ENDFORM.
*---------------------------------------------------------------------*
* FORM top_of_list2 *
*---------------------------------------------------------------------*
FORM top_of_list2. "#EC CALLED
WRITE 'TOP OF LIST2'.
ENDFORM.
************** END OF PROGRAM Z_ALV_LIST_BLOCK **********************
```

Reincarnation of REUSE_ALV_FIELDCATALOG_MERGE

These FORMs are for people accustomed to
  REUSE_ALV_FIELDCATALOG_MERGE (despite the 72 ch source line limit)
  but not happy with LVC_FIELDCATALOG_MERGE which has NO Internal
table option

We do not want to crowd DDIC with too many structures!

The routines handle any internal table using field name as title
if not a DDIC data element.

Create a Include ZJNCINCLUDE with the 2 FORMs

There are two FORMs: ZJNC_DUMP_LIST will be very useful as it is
simple and needs no screen and can be called any number of times.

Should be very useful also for debugging esp. where Excel is not
available as you can dump any internal table anytime and inspect contents.

I wrote these routine mainly for debugging and the problem created by
half-line long comments in internal tables. With RTTI there is no source

code dependency.

ZJNC_DUMP_GRID is for OO-GRID loving people who can adapt that routine
for one off reports.
You can call this ONLY ONCE as it is best used using a dummy selection
screen - tip from SDN ABAP FAQ.

As FORMs use RTTI there is no special case for Structures!

```
-----------------------------------------------------------------------------------------
*&---------------------------------------------------------------------*
*&  Include           ZJNCINCLUDE
*
*&---------------------------------------------------------------------*

*&---------------------------------------------------------------------*
*&  Reincarnations of REUSE_ALV_FIELDCATALOG_MERGE
*&---------------------------------------------------------------------*

* Author Jayanta Narayan Choudhuri
*         Flat 302
*         395 Jodhpur Park
*         Kolkata 700 068
*       Email sss@cal.vsnl.net.in
*       URL:  http://www.geocities.com/ojnc

*---------------------------------------------------------------------
* These FORMs are for people accustomed to
*   REUSE_ALV_FIELDCATALOG_MERGE (despite the 72 ch source line limit)
*   but not happy with  LVC_FIELDCATALOG_MERGE
* We do not want to crowd DDIC with too many structures!
*---------------------------------------------------------------------

*&---------------------------------------------------------------------*
*&      Form  ZJNC_DUMP_LIST  Our Good Old ALV list -  RECOMMENDED!
*&---------------------------------------------------------------------*
```

```
FORM zjnc_dump_list  USING value(p_it_name) TYPE c
                           value(p_wa_name) TYPE c
                           value(p_heading) TYPE c.

  TYPE-POOLS: slis.

  DATA:
    stru_ref    TYPE REF TO cl_abap_structdescr,
    comp_tab    TYPE abap_compdescr_tab,
    one_comp    TYPE abap_compdescr,
    one_name    TYPE string,
    type_ref    TYPE REF TO cl_abap_typedescr,
    is_ddic     TYPE abap_bool,
    lt_ddic     TYPE dd_x031l_table,
    wa_ddic     TYPE x031l,
    lt_fcat     TYPE slis_t_fieldcat_alv,
    wa_fcat     TYPE slis_fieldcat_alv,
    ls_layo     TYPE slis_layout_alv,
    l_alv       TYPE REF TO cl_gui_alv_grid.

  FIELD-SYMBOLS: <fs_type>  TYPE ANY,
                 <fs_table> TYPE STANDARD TABLE,
                 <fs_line>  TYPE ANY.

  ASSIGN (p_it_name) TO <fs_table>.

  ASSIGN (p_wa_name) TO <fs_line>.

  ls_layo-colwidth_optimize = 'X'.
  ls_layo-zebra = 'X'.
  ls_layo-window_titlebar = p_heading.
  ls_layo-box_tabname   = p_it_name.

  stru_ref ?= cl_abap_structdescr=>describe_by_data( <fs_line> ).

  comp_tab = stru_ref->components.
```

```
  LOOP AT comp_tab INTO one_comp.
    CLEAR wa_fcat.
    wa_fcat-tabname   = p_it_name.
    wa_fcat-fieldname = one_comp-name.

    CONCATENATE p_wa_name '-' one_comp-name INTO one_name.

    ASSIGN (one_name) TO <fs_type>.

    type_ref ?= cl_abap_typedescr=>describe_by_data( <fs_type> ).

    is_ddic = type_ref->is_ddic_type( ).

    IF is_ddic = abap_true.
      lt_ddic = type_ref->get_ddic_object( ).

      LOOP AT lt_ddic INTO wa_ddic.
        CLEAR wa_ddic-tabname.
        SELECT SINGLE
               dd03l~tabname
          INTO wa_ddic-tabname
          FROM dd03l
         WHERE dd03l~fieldname = wa_ddic-fieldname
           AND dd03l~tabname NOT LIKE '/%'.          " I live in normal
namespace


        wa_fcat-ref_tabname    = wa_ddic-tabname.
        wa_fcat-ref_fieldname = wa_ddic-fieldname.

        SELECT SINGLE
               dd04t~scrtext_s
               dd04t~scrtext_m
               dd04t~scrtext_l
          INTO (wa_fcat-seltext_s, wa_fcat-seltext_m,
```

```
wa_fcat-seltext_l)
          FROM dd04t
         WHERE dd04t~rollname   = wa_ddic-fieldname
           AND dd04t~ddlanguage = sy-langu.

      ENDLOOP.
    ELSE.
      MOVE one_comp-name TO: wa_fcat-seltext_s, wa_fcat-seltext_m,
wa_fcat-seltext_l.
    ENDIF.

    APPEND wa_fcat TO lt_fcat.

  ENDLOOP.

  CALL FUNCTION 'REUSE_ALV_LIST_DISPLAY'
    EXPORTING
      is_layout   = ls_layo
      it_fieldcat = lt_fcat
    TABLES
      t_outtab    = <fs_table>.

ENDFORM.                      "ZJNC_DUMP_LIST

*&---------------------------------------------------------------------*
*&      Form  ZJNC_DUMP_GRID  Object Oriented
*&---------------------------------------------------------------------*
FORM zjnc_dump_grid  USING value(p_it_name) TYPE c
                           value(p_wa_name) TYPE c
                           value(p_screen)  TYPE n
                           value(p_heading) TYPE c.

  DATA:
    stru_ref    TYPE REF TO cl_abap_structdescr,
    comp_tab    TYPE abap_compdescr_tab,
    one_comp    TYPE abap_compdescr,
```

```
  one_name    TYPE string,
  type_ref    TYPE REF TO cl_abap_typedescr,
  is_ddic     TYPE abap_bool,
  lt_ddic     TYPE dd_x031l_table,
  wa_ddic     TYPE x031l,
  lt_fcat     TYPE lvc_t_fcat,
  wa_fcat     TYPE lvc_s_fcat,
  ls_layo     TYPE lvc_s_layo,
  l_alv       TYPE REF TO cl_gui_alv_grid.

FIELD-SYMBOLS: <fs_type>  TYPE ANY,
               <fs_table> TYPE ANY TABLE,
               <fs_line>  TYPE ANY.

ASSIGN (p_it_name) TO <fs_table>.

ASSIGN (p_wa_name) TO <fs_line>.

ls_layo-cwidth_opt = 'X'.
ls_layo-zebra = 'X'.
ls_layo-grid_title = p_heading.
ls_layo-box_fname = p_it_name.

stru_ref ?= cl_abap_structdescr=>describe_by_data( <fs_line> ).

comp_tab = stru_ref->components.

LOOP AT comp_tab INTO one_comp.
  CLEAR wa_fcat.
  wa_fcat-tabname   = p_it_name.
  wa_fcat-fieldname = one_comp-name.

  CONCATENATE p_wa_name '-' one_comp-name INTO one_name.

  ASSIGN (one_name) TO <fs_type>.
```

```
    type_ref ?= cl_abap_typedescr=>describe_by_data( <fs_type> ).

    is_ddic = type_ref->is_ddic_type( ).

    IF is_ddic = abap_true.
      lt_ddic = type_ref->get_ddic_object( ).

      LOOP AT lt_ddic INTO wa_ddic.
        CLEAR wa_ddic-tabname.
        SELECT SINGLE
               dd03l~tabname
          INTO wa_ddic-tabname
          FROM dd03l
         WHERE dd03l~fieldname = wa_ddic-fieldname
           AND dd03l~tabname NOT LIKE '/%'.          " I live in normal
namespace



        wa_fcat-ref_table = wa_ddic-tabname.
        wa_fcat-ref_field = wa_ddic-fieldname.

        SELECT SINGLE
               dd04t~scrtext_s
               dd04t~scrtext_m
               dd04t~scrtext_l
          INTO (wa_fcat-scrtext_s, wa_fcat-scrtext_m,
wa_fcat-scrtext_l)
          FROM dd04t
         WHERE dd04t~rollname   = wa_ddic-fieldname
           AND dd04t~ddlanguage = sy-langu.

      ENDLOOP.

    ELSE.
      MOVE one_comp-name TO: wa_fcat-scrtext_s, wa_fcat-scrtext_m,
wa_fcat-scrtext_l.
```

```
    ENDIF.

    APPEND wa_fcat TO lt_fcat.

  ENDLOOP.

  CREATE OBJECT l_alv EXPORTING i_parent = cl_gui_container=>screen0.

  CALL METHOD l_alv->set_table_for_first_display
    EXPORTING
      is_layout       = ls_layo
    CHANGING
      it_outtab       = <fs_table>
      it_fieldcatalog = lt_fcat.

  CALL SELECTION-SCREEN p_screen.

ENDFORM.                        "ZJNC_DUMP_GRID

-------------------------------------------------------------------------------
* SAMPLE TEST PROGRAM

REPORT yjnc1.

INCLUDE zjncinclude.

TYPES : BEGIN OF ty_pur,
          ebeln1  TYPE ekko-ebeln,
          aedat   TYPE ekko-aedat,
          ebelp   TYPE ekpo-ebelp,
          matnr   TYPE ekpo-matnr,
          txz01   TYPE ekpo-txz01,
          menge1  TYPE ekpo-menge,
          belnr1  TYPE ekbe-belnr,
          budat1  TYPE ekbe-budat,
          bwart   TYPE ekbe-bwart,
```

```
          menge2  TYPE ekbe-menge,
          belnr2  TYPE ekbz-belnr,
          budat2  TYPE ekbz-budat,
          menge3  TYPE ekbz-menge,
          dmbtr   TYPE ekbz-dmbtr,
          buzei   TYPE ekbz-buzei,
          jnc1    TYPE sy-datum,
          jnc2    TYPE i,
          jnc3(4) TYPE c,
        END OF ty_pur.

DATA: it_pur TYPE STANDARD TABLE OF ty_pur,
      wa_pur TYPE ty_pur,
      rows   TYPE i.

* Not Recommended Style but still used!
DATA: BEGIN OF it_mat OCCURS 0,
        jnc1    TYPE sy-datum,
        matnr   LIKE makt-matnr,
        jnc2    TYPE i,
        maktx   LIKE makt-maktx,
        mtart   TYPE mara-mtart,
        jnc3(4) TYPE c,
     END OF it_mat.

DATA: BEGIN OF it_str OCCURS 0.
        INCLUDE STRUCTURE makt.
DATA: END OF it_str.

SELECTION-SCREEN BEGIN OF SCREEN 1001.
SELECTION-SCREEN END   OF SCREEN 1001.

START-OF-SELECTION.

  MOVE     100 TO rows.
```

```
SELECT ekko~ebeln
       ekko~aedat
       ekpo~ebelp
       ekpo~matnr
       ekpo~txz01
       ekpo~menge
       ekbe~belnr
       ekbe~budat
       ekbe~bwart
       ekbe~menge
       ekbz~belnr
       ekbz~budat
       ekbz~menge
       ekbz~dmbtr
       ekbz~buzei
  INTO TABLE it_pur
  UP TO rows ROWS
  FROM ekko
       INNER JOIN ekpo
       ON  ekko~ebeln = ekpo~ebeln
       AND ekpo~loekz = ' '
          INNER JOIN ekbe
          ON  ekpo~ebeln = ekbe~ebeln
          AND ekpo~ebelp = ekbe~ebelp
          AND ekbe~bewtp = 'E'
       LEFT OUTER JOIN ekbz
       ON  ekpo~ebeln = ekbz~ebeln
       AND ekpo~ebelp = ekbz~ebelp
       AND ekbz~bewtp = 'M'
 WHERE ekko~bstyp = 'F'
   AND ekko~loekz = ' '
 ORDER BY ekko~ebeln
          ekpo~ebelp.

LOOP AT it_pur INTO wa_pur.
  COMPUTE wa_pur-jnc1 =  sy-datum - sy-tabix.
```

```
    MOVE sy-tabix TO wa_pur-jnc2.
    MOVE 'Helo'   TO wa_pur-jnc3.
    MODIFY it_pur FROM wa_pur.
  ENDLOOP.

* PERFORM zjnc_dump_grid  USING 'IT_PUR' 'WA_PUR' 1001 'Purchase
Report'.

* IT_PUR is as per recommended OO style
  PERFORM zjnc_dump_list  USING 'IT_PUR' 'WA_PUR' 'Purchase Dump'.

  SELECT makt~matnr
         makt~maktx
         mara~mtart
    INTO CORRESPONDING FIELDS OF TABLE it_mat
    UP TO rows ROWS
    FROM mara INNER JOIN makt
         ON makt~matnr = mara~matnr
         AND makt~spras = sy-langu.

  LOOP AT it_mat.
    COMPUTE it_mat-jnc1 =  sy-datum - sy-tabix.
    MOVE sy-tabix TO it_mat-jnc2.
    MOVE 'Helo'   TO it_mat-jnc3.
    MODIFY it_mat.
  ENDLOOP.


* PERFORM zjnc_dump_grid  USING 'IT_MAT[]' 'IT_MAT' 1001 'Material
Dump'.

* Note that IT_MAT with Header line is 2 in 1
* IT_MAT[] is the table object and IT_MAT is the header work area

  PERFORM zjnc_dump_list  USING 'IT_MAT[]' 'IT_MAT' 'Material Dump'.
```

```
  SELECT *
    INTO TABLE it_str
    UP TO rows ROWS
    FROM makt
   WHERE makt~spras = sy-langu.

* Note that IT_STR with Header line is 2 in 1
* IT_STR[] is the table object and IT_STR is the header work area
* As FORM zjnc_dump_list uses RTTI there is no special case for
Structures

  PERFORM zjnc_dump_list  USING 'IT_STR[]' 'IT_STR' 'Structure Dump'.
```


ALV Reporting - Z_LIST_MATERIALS

```
REPORT Z_LIST_MATERIALS.

TYPE-POOLS: SLIS.
TABLES:MARC, MARD, VBAP, LIPS, EKPO, VBFA, EKBE, MARM, VBBE, MARA, MBEW.

SELECTION-SCREEN BEGIN OF BLOCK SEL WITH FRAME TITLE TEXT-001.
SELECT-OPTIONS: S_WERKS FOR MARC-WERKS, " Plant
                S_MATNR FOR MARC-MATNR, " Material
                S_MTART FOR MARA-MTART. " Material Type SELECTION-SCREEN END OF BLOCK SEL.
PARAMETERS: P_VARI LIKE DISVARIANT-VARIANT. " ALV Variant

CONSTANTS: FORMNAME_TOP_OF_PAGE TYPE SLIS_FORMNAME VALUE 'TOP_OF_PAGE'.

DATA: BEGIN OF INV OCCURS 100,
       WERKS LIKE MARD-WERKS, " Plant
       MATNR LIKE MARD-MATNR, " Material
       MTART LIKE MARA-MTART, " Material Type
       STPRS LIKE MBEW-STPRS, " Standard Price
       AVAIL LIKE MARD-LABST, " Available
       LABST LIKE MARD-LABST, " Unrestricted use
       INSME LIKE MARD-INSME, " Quality Inspection
```

```
        RETME LIKE MARD-RETME, " Returns
        TRANS LIKE MARC-UMLMC, " Stock in transit (calculated)
        UMLMC LIKE MARC-UMLMC, " Stock Transfer (plant)
        UMLME LIKE MARD-UMLME, " Transfer (SLoc)
        WESBS LIKE EKBE-WESBS, " GR Blocked Stock
        TRAME LIKE MARC-TRAME, " Stock in transit
        SPEME LIKE MARD-SPEME, " Blocked
        KWMENG LIKE VBAP-KWMENG, " Sales orders
        LFIMG LIKE LIPS-LFIMG, " Scheduled for Delivery
        MENGE LIKE EKPO-MENGE, " Open Purch. Orders
        VALUE LIKE MBEW-SALK3, " Stock Value (Calculated)
        MEINS LIKE MARA-MEINS, " Unit of measure
      END OF INV.

DATA: FIELDTAB TYPE SLIS_T_FIELDCAT_ALV,
      HEADING  TYPE SLIS_T_LISTHEADER,
      LAYOUT   TYPE SLIS_LAYOUT_ALV,
      EVENTS   TYPE SLIS_T_EVENT,
      REPNAME  LIKE SY-REPID,
      F2CODE   LIKE SY-UCOMM VALUE  '&ETA',
      G_SAVE(1) TYPE C,
      G_EXIT(1) TYPE C,
      G_VARIANT LIKE DISVARIANT,
      GX_VARIANT LIKE DISVARIANT.

INITIALIZATION.
 REPNAME = SY-REPID.
 PERFORM INITIALIZE_FIELDCAT USING FIELDTAB[].
 PERFORM BUILD_EVENTTAB USING EVENTS[].
 PERFORM BUILD_COMMENT USING HEADING[].
 PERFORM INITIALIZE_VARIANT.

AT SELECTION-SCREEN ON VALUE-REQUEST FOR P_VARI.
 PERFORM F4_FOR_VARIANT.

AT SELECTION-SCREEN.
```

```
 PERFORM PAI_OF_SELECTION_SCREEN.

START-OF-SELECTION.
 PERFORM GET_MARD.
 PERFORM GET_UNIT_OF_MEASURE.
 PERFORM GET_MARC.
 PERFORM GET_EKPO.
 PERFORM GET_LIPS.
 PERFORM GET_VBAP.
 PERFORM GET_OPEN.
 PERFORM GET_PRICE.

END-OF-SELECTION.

 PERFORM BUILD_LAYOUT USING LAYOUT.
 PERFORM WRITE_OUTPUT.



*&---------------------------------------------------------------------*
*&      Form  INITIALIZE_FIELDCAT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->P_FIELDTAB[]  text                                           *
*----------------------------------------------------------------------*
FORM INITIALIZE_FIELDCAT USING P_FIELDTAB TYPE SLIS_T_FIELDCAT_ALV.
 DATA: L_FIELDCAT TYPE SLIS_FIELDCAT_ALV.
* fixed columns (obligatory)
 CLEAR L_FIELDCAT.
 L_FIELDCAT-TABNAME    = 'INV'.
 L_FIELDCAT-FIX_COLUMN = 'X'.
 L_FIELDCAT-NO_OUT     = 'O'.
 L_FIELDCAT-FIELDNAME  = 'WERKS'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME  = 'MATNR'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
```

```
* totalized columns
 CLEAR L_FIELDCAT.
 L_FIELDCAT-TABNAME   = 'INV'.
 L_FIELDCAT-SP_GROUP  = 'A'.
 L_FIELDCAT-DO_SUM    = 'X'.
 L_FIELDCAT-FIELDNAME = 'LABST'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME = 'INSME'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME = 'RETME'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME = 'UMLME'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME = 'WESBS'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME = 'SPEME'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
* columns with different description
 L_FIELDCAT-FIELDNAME = 'KWMENG'.
 L_FIELDCAT-SELTEXT_M = 'Sales Orders'.
 L_FIELDCAT-SELTEXT_S = 'Sales Or'.
 L_FIELDCAT-SELTEXT_L = 'Sales Orders Qty'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME = 'LFIMG'.
 L_FIELDCAT-SELTEXT_M = 'Sched. Delivery'.
 L_FIELDCAT-SELTEXT_S = 'Schd. Del'.
 L_FIELDCAT-SELTEXT_L = 'Scheduled for Delivery'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME = 'TRANS'.
 L_FIELDCAT-SELTEXT_M = 'Stk. in transit'.
 L_FIELDCAT-SELTEXT_S = 'Stk. trns'.
 L_FIELDCAT-SELTEXT_L = 'Stock in transit (sum)'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME = 'AVAIL'.
 L_FIELDCAT-SELTEXT_M = 'Available'.
```

```
 L_FIELDCAT-SELTEXT_S = 'Avail.'.
 L_FIELDCAT-SELTEXT_L = 'Stock Available'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME = 'MENGE'.
 L_FIELDCAT-SELTEXT_M = 'Open Orders'.
 L_FIELDCAT-SELTEXT_S = 'Open Ord'.
 L_FIELDCAT-SELTEXT_L = 'Open Purchase Orders'.
 APPEND L_FIELDCAT TO P_FIELDTAB.

* columns not displayed
 CLEAR L_FIELDCAT.
 L_FIELDCAT-TABNAME   = 'INV'.
 L_FIELDCAT-SP_GROUP  = 'A'.
 L_FIELDCAT-NO_OUT    = 'X'.
 L_FIELDCAT-FIELDNAME = 'MEINS'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME = 'UMLMC'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME = 'TRAME'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME = 'STPRS'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
 L_FIELDCAT-FIELDNAME = 'VALUE'.
 APPEND L_FIELDCAT TO P_FIELDTAB.
ENDFORM.                              " INITIALIZE_FIELDCAT

*&---------------------------------------------------------------------*
*&      Form  BUILD_EVENTTAB
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->P_EVENTS[]  text                                             *
*----------------------------------------------------------------------*
FORM BUILD_EVENTTAB USING P_EVENTS TYPE SLIS_T_EVENT.
 DATA: LS_EVENT TYPE SLIS_ALV_EVENT.
 CALL FUNCTION 'REUSE_ALV_EVENTS_GET'
```

```
      EXPORTING
           I_LIST_TYPE = 0
      IMPORTING
           ET_EVENTS   = P_EVENTS.
 READ TABLE P_EVENTS WITH KEY NAME = SLIS_EV_TOP_OF_PAGE
                           INTO LS_EVENT.
 IF SY-SUBRC = 0.
   MOVE FORMNAME_TOP_OF_PAGE TO LS_EVENT-FORM.
   APPEND LS_EVENT TO P_EVENTS.
 ENDIF.
ENDFORM.                                  " BUILD_EVENTTAB

*&---------------------------------------------------------------------*
*&      Form  BUILD_COMMENT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->P_HEADING[]  text                                            *
*----------------------------------------------------------------------*
FORM BUILD_COMMENT USING P_HEADING TYPE SLIS_T_LISTHEADER.
 DATA: HLINE TYPE SLIS_LISTHEADER,
       TEXT(60) TYPE C,
       SEP(20) TYPE C.
 CLEAR: HLINE, TEXT.
 HLINE-TYP  = 'H'.
 WRITE: TEXT-101 TO TEXT+23.
 HLINE-INFO = TEXT.
 APPEND HLINE TO P_HEADING.
 CLEAR TEXT.
 WRITE: 'User: ' TO TEXT,
        SY-UNAME TO TEXT+6,
        'Date: ' TO TEXT+25,
        SY-DATUM TO TEXT+31,
        'Page: ' TO TEXT+50,
        SY-PAGNO TO TEXT+56.
 HLINE-INFO = TEXT.
```

```
 APPEND HLINE TO P_HEADING.



ENDFORM.                      " BUILD_COMMENT

*---------------------------------------------------------------------*
*       FORM TOP_OF_PAGE                                              *
*---------------------------------------------------------------------*
*       ........                                                      *
*---------------------------------------------------------------------*
FORM TOP_OF_PAGE.
 CALL FUNCTION 'REUSE_ALV_COMMENTARY_WRITE'
      EXPORTING
           IT_LIST_COMMENTARY = HEADING.
ENDFORM.
*&---------------------------------------------------------------------*
*&      Form  INITIALIZE_VARIANT
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
FORM INITIALIZE_VARIANT.
 G_SAVE = 'A'.
 CLEAR G_VARIANT.
 G_VARIANT-REPORT = REPNAME.
 GX_VARIANT = G_VARIANT.
 CALL FUNCTION 'REUSE_ALV_VARIANT_DEFAULT_GET'
      EXPORTING
           I_SAVE     = G_SAVE
      CHANGING
           CS_VARIANT = GX_VARIANT
      EXCEPTIONS
           NOT_FOUND  = 2.
 IF SY-SUBRC = 0.
   P_VARI = GX_VARIANT-VARIANT.
 ENDIF.
```

```
ENDFORM.                              " INITIALIZE_VARIANT

*&---------------------------------------------------------------------*
*&      Form  F4_FOR_VARIANT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM F4_FOR_VARIANT.
 CALL FUNCTION 'REUSE_ALV_VARIANT_F4'
      EXPORTING
           IS_VARIANT = G_VARIANT
           I_SAVE     = G_SAVE
      IMPORTING
           E_EXIT     = G_EXIT
           ES_VARIANT = GX_VARIANT
      EXCEPTIONS
           NOT_FOUND  = 2.
 IF SY-SUBRC = 2.
   MESSAGE ID SY-MSGID TYPE 'S'      NUMBER SY-MSGNO
           WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
 ELSE.
   IF G_EXIT = SPACE.
     P_VARI = GX_VARIANT-VARIANT.
   ENDIF.
 ENDIF.
ENDFORM.                              " F4_FOR_VARIANT

*&---------------------------------------------------------------------*
*&      Form  PAI_OF_SELECTION_SCREEN
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM PAI_OF_SELECTION_SCREEN.
*
 IF NOT P_VARI IS INITIAL.
   MOVE G_VARIANT TO GX_VARIANT.
```

```
   MOVE P_VARI TO GX_VARIANT-VARIANT.
   CALL FUNCTION 'REUSE_ALV_VARIANT_EXISTENCE'
        EXPORTING
             I_SAVE     = G_SAVE
        CHANGING
             CS_VARIANT = GX_VARIANT.
   G_VARIANT = GX_VARIANT.
 ELSE.
   PERFORM INITIALIZE_VARIANT.
 ENDIF.
ENDFORM.                               " PAI_OF_SELECTION_SCREEN


*&---------------------------------------------------------------------*
*&      Form  GET_MARD
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM GET_MARD.
 SELECT MATNR WERKS LABST INSME RETME UMLME SPEME
        FROM MARD
        INTO CORRESPONDING FIELDS OF INV
        WHERE MATNR IN S_MATNR
        AND   WERKS IN S_WERKS.
   COLLECT INV.
 ENDSELECT.
 PERFORM FILTER_BY_MATERIAL_TYPE.
ENDFORM.                               " GET_MARD

*&---------------------------------------------------------------------*
*&      Form  FILTER_BY_MATERIAL_TYPE
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM FILTER_BY_MATERIAL_TYPE.
 LOOP AT INV.
```

```
   CLEAR INV-MTART.
   SELECT SINGLE MTART
          INTO INV-MTART
          FROM MARA
          WHERE MATNR EQ INV-MATNR
          AND   MTART IN S_MTART.
   IF SY-SUBRC EQ 0.
     MODIFY INV.
   ELSE.
     DELETE INV.
   ENDIF.
 ENDLOOP.
ENDFORM.                      " FILTER_BY_MATERIAL_TYPE

*&---------------------------------------------------------------------*
*&      Form  GET_MARC
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM GET_MARC.
 LOOP AT INV.
   SELECT SINGLE UMLMC TRAME
          FROM MARC
          INTO CORRESPONDING FIELDS OF INV
          WHERE MATNR EQ INV-MATNR
          AND   WERKS EQ INV-WERKS.
   IF SY-SUBRC EQ 0.
     INV-TRANS = INV-UMLMC + INV-TRAME.
     MODIFY INV.
   ENDIF.
 ENDLOOP.
ENDFORM.                                " GET_MARC
*&---------------------------------------------------------------------*
*&      Form  GET_EKPO
*&---------------------------------------------------------------------*
*       text
```

```
*---------------------------------------------------------------------*
FORM GET_EKPO.
 DATA: WESBS LIKE INV-WESBS,
       SHKZG LIKE EKBE-SHKZG,
       MEINS LIKE EKPO-MEINS,
       LMEIN LIKE EKPO-LMEIN.
 LOOP AT INV.
   CLEAR: WESBS, SHKZG, MEINS, LMEIN.
   SELECT YWESBS YSHKZG XMEINS XLMEIN
          INTO (WESBS, SHKZG, MEINS, LMEIN)
          FROM EKPO AS X JOIN EKBE AS Y
          ON   XEBELN = YEBELN
          AND  XEBELP = YEBELP
          WHERE XMATNR EQ INV-MATNR
          AND   XWERKS EQ INV-WERKS
          AND   XLOEKZ NE 'L'.
     IF SHKZG EQ 'H'.
       MULTIPLY WESBS BY -1.
     ENDIF.
     IF MEINS NE LMEIN.
       PERFORM CONVERT_UNIT_OF_MEASURE CHANGING INV-MATNR MEINS WESBS.
     ENDIF.
     ADD WESBS TO INV-WESBS.
   ENDSELECT.
   MODIFY INV.
 ENDLOOP.
ENDFORM.                              " GET_EKPO
*&---------------------------------------------------------------------*
*&      Form  GET_LIPS
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
FORM GET_LIPS.
 DATA: LFIMG LIKE INV-LFIMG.
 LOOP AT INV.
   CLEAR: LFIMG, INV-LFIMG.
```

```
   SELECT OMENG
          INTO LFIMG
          FROM VBBE
          WHERE MATNR EQ INV-MATNR
          AND   WERKS EQ INV-WERKS
          AND   VBTYP EQ 'J'.
     ADD LFIMG TO INV-LFIMG.
   ENDSELECT.
   MODIFY INV.
 ENDLOOP.
ENDFORM.                              " GET_LIPS
*&---------------------------------------------------------------------*
*&      Form  GET_VBAP
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM GET_VBAP.
 DATA: KWMENG LIKE INV-KWMENG.
 LOOP AT INV.
   CLEAR: KWMENG, INV-KWMENG.
   SELECT OMENG
          INTO KWMENG
          FROM VBBE
          WHERE MATNR EQ INV-MATNR
          AND   WERKS EQ INV-WERKS
          AND   VBTYP EQ 'C'.
     ADD KWMENG TO INV-KWMENG.
   ENDSELECT.
   INV-AVAIL = INV-LABST - INV-INSME - INV-KWMENG - INV-LFIMG.
   MODIFY INV.
 ENDLOOP.
ENDFORM.                              " GET_VBAP

*&---------------------------------------------------------------------*
*&      Form  GET_UNIT_OF_MEASURE
*&---------------------------------------------------------------------*
```

```
*       text
*---------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*---------------------------------------------------------------------*
FORM GET_UNIT_OF_MEASURE.
 LOOP AT INV.
   SELECT SINGLE MEINS
          FROM MARA
          INTO INV-MEINS
          WHERE MATNR EQ INV-MATNR.
   MODIFY INV.
 ENDLOOP.
ENDFORM.                                  " GET_UNIT_OF_MEASURE

*&---------------------------------------------------------------------*
*&      Form  GET_OPEN
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
FORM GET_OPEN.
 DATA: BEGIN OF XTAB OCCURS 10,        " Open orders table
          WERKS LIKE EKPO-WERKS,
          LGORT LIKE EKPO-LGORT,
          MATNR LIKE EKPO-MATNR,
          MENGE LIKE EKPO-MENGE,
          MENGK LIKE EKPO-MENGE,
        END OF XTAB.
 RANGES: L_WERKS FOR MARD-WERKS.

 LOOP AT INV.
   REFRESH XTAB.
   CLEAR: XTAB, L_WERKS.
   MOVE INV-WERKS TO L_WERKS-LOW.
   CALL FUNCTION 'MB_ADD_PURCHASE_ORDER_QUANTITY'
        EXPORTING
```

```
            X_MATNR = INV-MATNR
            X_MEINS = INV-MEINS
            X_ELIKZ = SPACE
            X_LOEKZ = SPACE
        TABLES
            XTAB    = XTAB
            XWERKS  = L_WERKS.
   MOVE XTAB-MENGE TO INV-MENGE.
   MODIFY INV.
 ENDLOOP.
ENDFORM.                                  " GET_OPEN

*&---------------------------------------------------------------------*
*&      Form  GET_PRICE
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM GET_PRICE.
 LOOP AT INV.
   SELECT SINGLE STPRS
          FROM MBEW
          INTO INV-STPRS
          WHERE MATNR EQ INV-MATNR
          AND   BWKEY EQ INV-WERKS
          AND   BWTAR EQ SPACE.
     IF SY-SUBRC EQ 0.
       INV-VALUE = INV-STPRS *
                 ( INV-LABST + INV-INSME + INV-TRANS + INV-SPEME ).
       MODIFY INV.
     ENDIF.
 ENDLOOP.
ENDFORM.                      " GET_PRICE

*----------------------------------------------------------------------*
*       FORM CONVERT_UNIT_OF_MEASURE                                   *
*----------------------------------------------------------------------*
```

```
*       text                                                           *
*----------------------------------------------------------------------*
*  -->  P_MATNR                                                        *
*  -->  P_VRKME                                                        *
*  -->  P_QUANT                                                        *
*----------------------------------------------------------------------*
FORM CONVERT_UNIT_OF_MEASURE USING P_MATNR P_VRKME P_QUANT.
 DATA: UMREZ LIKE MARM-UMREZ,
       UMREN LIKE MARM-UMREN.
 SELECT SINGLE UMREZ UMREN
        INTO (UMREZ, UMREN)
        FROM MARM
        WHERE MATNR EQ P_MATNR
        AND   MEINH EQ P_VRKME.
 IF SY-SUBRC EQ 0.
   COMPUTE P_QUANT = ( P_QUANT * UMREZ ) / UMREN.
 ENDIF.
ENDFORM.

*&----------------------------------------------------------------------*
*&      Form  BUILD_LAYOUT
*&----------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->P_LAYOUT  text                                               *
*----------------------------------------------------------------------*
FORM BUILD_LAYOUT USING P_LAYOUT TYPE SLIS_LAYOUT_ALV.
 P_LAYOUT-F2CODE       = F2CODE.
 P_LAYOUT-ZEBRA        = 'X'.
 P_LAYOUT-DETAIL_POPUP = 'X'.
ENDFORM.                                  " BUILD_LAYOUT

*&----------------------------------------------------------------------*
*&      Form  WRITE_OUTPUT
*&----------------------------------------------------------------------*
*       text
```

```
*---------------------------------------------------------------------*
FORM WRITE_OUTPUT.
 SORT INV BY WERKS MATNR.
 CALL FUNCTION 'REUSE_ALV_FIELDCATALOG_MERGE'
      EXPORTING
           I_PROGRAM_NAME     = REPNAME
           I_INTERNAL_TABNAME = 'INV'
           I_INCLNAME         = REPNAME
      CHANGING
           CT_FIELDCAT        = FIELDTAB.
 IF SY-SUBRC <> 0.
   WRITE: 'SY-SUBRC: ', SY-SUBRC, 'REUSE_ALV_FIELDCATALOG_MERGE'.  ENDIF.  CALL FUNCTION 'RE
USE_ALV_LIST_DISPLAY'
      EXPORTING
           I_CALLBACK_PROGRAM = REPNAME
           I_STRUCTURE_NAME   = 'INV'
           IS_LAYOUT          = LAYOUT
           IT_FIELDCAT        = FIELDTAB
           I_DEFAULT          = 'A'
           I_SAVE             = G_SAVE
           IS_VARIANT         = G_VARIANT
           IT_EVENTS          = EVENTS[]
      TABLES
           T_OUTTAB           = INV.
 IF SY-SUBRC <> 0.
   WRITE: 'SY-SUBRC: ', SY-SUBRC, 'REUSE_ALV_LIST_DISPLAY'.  ENDIF.
ENDFORM.                              " WRITE_OUTPUT
```


Use Simple ALV Functions to Make Reporting Easy

```
*&---------------------------------------------------------------------*
*& Report  ZBC_ALV_EXAMPLE                                            *
*&                                                                     *
*&---------------------------------------------------------------------*
* This program explains how we can use simple ALV functions to make    *
* reporting easy and looks pretty
```

```
************************************************************************
* Programmer        : Venkat Reddy           ETA                      *
* Date              : 10/02/04                                        *
************************************************************************
* Maintenance Log                                                     *
*---------------------------------------------------------------------*
* Changed By      Date         Transport#   Description               *
*------------ -------------------------------------------------------*
* Venkat Reddy    10/02/04     EGD913575    Changed program to avoid  *
*---------------------------------------------------------------------*
REPORT  ZBC_ALV_EXAMPLE.

************************************************************************
*                 D-A-T-A  D-E-C-L-A-R-A-T-I-O-N-S                     *
************************************************************************
tables:  sflight.

**-- TYPE-POOLS Definition
**Includes the types and constants of a type group. Since the types and
*constants specified in a type group have global validity, you cannot
*use the statement within a FORM or FUNCTION.
type-pools: slis.
PARAMETERS: P_VARI LIKE DISVARIANT-VARIANT.

**-- ALV variables
*****- Field Catalog structure
data: ls_fieldcat     type slis_fieldcat_alv,    "Field Catalog list

**--- Field Catalog table
      gt_fieldcat     type slis_t_fieldcat_alv,  "Field Catalog

**--- Layout ( How you would like to see the output )
      gs_layout       type slis_layout_alv,       "List Layout

**--  Report name
      g_repid         like sy-repid,
```

```
      g_save(1)           type c,
     g_exit(1)           type c,
     g_variant           like disvariant,
     gx_variant          like disvariant.


**-- Flight Info Internal table
data: lt_sflight like sflight occurs 0 with header line.

**********************************************************************
*                 C-O-N-S-T-A-N-T-S                                  *
**********************************************************************

**********************************************************************
*                 S-E-L-E-C-T-I-O-N  S-C-R-E-E-N                     *
**********************************************************************

selection-screen begin of block a with frame title text-100.
select-options: s_carrid  for  sflight-carrid,
                s_connid  for  sflight-connid,
                s_fldate  for  sflight-fldate default sy-datum.
selection-screen end of block a .


**********************************************************************
*                 I-N-I-T-I-A-L-I-Z-A-T-I-O-N                        *
**********************************************************************
initialization.

  g_repid = sy-repid.

**-- Fill ALV field catalog
  perform initialize_fieldcat using gt_fieldcat[].

***-- Build Events
*  perform build_eventtab using gt_events[].
```

```
*
**-- Read the default variant
  perform initialize_variant.

***********************************************************************
*                  A-T  S-E-L-E-C-T-I-O-N  S-C-R-E-E-N                *
***********************************************************************
at selection-screen on value-request for p_vari.
**-- Display all existing variants
  call function 'REUSE_ALV_VARIANT_F4'
       exporting
            is_variant = g_variant
            i_save     = g_save
       importing
            e_exit     = g_exit
            es_variant = gx_variant
       exceptions
            not_found  = 2.
  if sy-subrc = 2.
    message id sy-msgid type 'S'      number sy-msgno
            with sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
  else.
    if g_exit = space.
      p_vari = gx_variant-variant.
    endif.
  endif.


***********************************************************************
*                  S-T-A-R-T  O-F  S-E-L-E-C-T-I-O-N                  *
***********************************************************************
start-of-selection.

**-- Read Flight information.
perform read_flight_info.
```

```
**-- Fill ALV field catalog and display report.
  if not lt_sflight[] is initial.
    perform dislay_alv_report.
  endif.

*=====================================================================*
*                 FORMS / SUB ROUTINES                                *
*=====================================================================

*&---------------------------------------------------------------------*
*&      Form  initialize_fieldcat
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->P_GT_FIELDCAT[]  text
*----------------------------------------------------------------------*
form initialize_fieldcat using l_fieldcat type slis_t_fieldcat_alv.

  clear ls_fieldcat.

* Air line
  ls_fieldcat-fieldname   = 'CARRID'.
  ls_fieldcat-key         = 'X'.
  ls_fieldcat-col_pos     = 1.
  ls_fieldcat-seltext_s   = 'Airline'.
  ls_fieldcat-seltext_l   = 'Airline'.
  append ls_fieldcat to l_fieldcat.
  clear ls_fieldcat.

* Flight Number
  ls_fieldcat-fieldname   = 'CONNID'.
  ls_fieldcat-key         = 'X'.
  ls_fieldcat-col_pos     = 2.
  ls_fieldcat-seltext_s   = 'Flight Number'.
  ls_fieldcat-seltext_l   = 'Flight Number'.
  append ls_fieldcat to l_fieldcat.
```

```
  clear ls_fieldcat.

* Flight date
  ls_fieldcat-fieldname   = 'FLDATE'.
  ls_fieldcat-key         = 'X'.
  ls_fieldcat-col_pos     = 3.
  ls_fieldcat-seltext_s   = 'Flight date'.
  ls_fieldcat-seltext_l   = 'Flight date'.
  append ls_fieldcat to l_fieldcat.
  clear ls_fieldcat.

* Airfare
  ls_fieldcat-fieldname   = 'PRICE'.
  ls_fieldcat-col_pos     = 4.
  ls_fieldcat-do_sum      = 'X'.
  ls_fieldcat-seltext_s   = 'Airfare'.
  ls_fieldcat-seltext_l   = 'Airfare'.
  append ls_fieldcat to l_fieldcat.
  clear ls_fieldcat.

* Local Currency
  ls_fieldcat-fieldname   = 'CURRENCY'.
  ls_fieldcat-col_pos     = 5.
  ls_fieldcat-seltext_s   = 'Local Currency'.
  ls_fieldcat-seltext_l   = 'Local Currency'.
  append ls_fieldcat to l_fieldcat.
  clear ls_fieldcat.

* Plane Type
  ls_fieldcat-fieldname   = 'PLANETYPE'.
  ls_fieldcat-col_pos     = 6.
  ls_fieldcat-seltext_s   = 'Plane type'.
  ls_fieldcat-seltext_l   = 'Plane type'.
  append ls_fieldcat to l_fieldcat.
  clear ls_fieldcat.
```

```
* Maximum capacity
  ls_fieldcat-fieldname   = 'SEATSMAX'.
  ls_fieldcat-col_pos     = 7.
  ls_fieldcat-seltext_s   = 'Max. seats'.
  ls_fieldcat-seltext_l   = 'Max. seats'.
  append ls_fieldcat to l_fieldcat.
  clear ls_fieldcat.

* Occupied seats
  ls_fieldcat-fieldname   = 'SEATSOCC'.
  ls_fieldcat-col_pos     = 8.
  ls_fieldcat-seltext_s   = 'Seats occupied'.
  ls_fieldcat-seltext_l   = 'Seats occupied'.
  append ls_fieldcat to l_fieldcat.
  clear ls_fieldcat.

* Total
  ls_fieldcat-fieldname   = 'PAYMENTSUM'.
  ls_fieldcat-col_pos     = 9.
  ls_fieldcat-do_sum     = 'X'.
  ls_fieldcat-seltext_s   = 'Total amount'.
  ls_fieldcat-seltext_l   = 'Total amount'.
  append ls_fieldcat to l_fieldcat.
  clear ls_fieldcat.

* Total
  ls_fieldcat-fieldname   = 'PAYMENTSUM'.
  ls_fieldcat-col_pos     = 9.
  ls_fieldcat-do_sum     = 'X'.
  ls_fieldcat-seltext_s   = 'Total amount'.
  ls_fieldcat-seltext_l   = 'Total amount'.
  append ls_fieldcat to l_fieldcat.
  clear ls_fieldcat.


* Max. Capacity, Buss. Class
```

```
  ls_fieldcat-fieldname   = 'SEATSMAX_B'.
  ls_fieldcat-col_pos     = 10.
  ls_fieldcat-seltext_s   = 'Max.Buss.class cap.'.
  ls_fieldcat-seltext_l   = 'Max.Buss.class cap.'.
  append ls_fieldcat to l_fieldcat.
  clear ls_fieldcat.

* Max. occupancy, Buss. Class
  ls_fieldcat-fieldname   = 'SEATSOCC_B'.
  ls_fieldcat-col_pos     = 11.
  ls_fieldcat-seltext_s   = 'Max.Bus.CL.occupied'.
  ls_fieldcat-seltext_l   = 'Max.Bus.CL.occupied'.
  append ls_fieldcat to l_fieldcat.
  clear ls_fieldcat.

* Max. Capacity, First. Class
  ls_fieldcat-fieldname   = 'SEATSMAX_F'.
  ls_fieldcat-col_pos     = 12.
  ls_fieldcat-seltext_s   = 'Max.Buss.class cap.'.
  ls_fieldcat-seltext_l   = 'Max.Buss.class cap.'.
  append ls_fieldcat to l_fieldcat.
  clear ls_fieldcat.

* Max. occupancy, First. Class
  ls_fieldcat-fieldname   = 'SEATSOCC_F'.
  ls_fieldcat-col_pos     = 13.
  ls_fieldcat-seltext_s   = 'Max.Bus.CL.occupied'.
  ls_fieldcat-seltext_l   = 'Max.Bus.CL.occupied'.
  append ls_fieldcat to l_fieldcat.
  clear ls_fieldcat.


ENDFORM.                    " initialize_fieldcat
*&---------------------------------------------------------------------*
*&      Form  read_flight_info
*&---------------------------------------------------------------------*
```

```
*       text
*---------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*---------------------------------------------------------------------*
FORM read_flight_info .

refresh lt_sflight.
clear   lt_sflight.

**-- Read data from SFLIGHT table
  select *
    from SFLIGHT
    into table lt_sflight
    where carrid in s_carrid
     and  connid in s_connid
     and  fldate in s_fldate.
 if sy-subrc <> 0.
   message e208(00) with text-101.
 endif.

ENDFORM.                    " read_flight_info
*&---------------------------------------------------------------------*
*&      Form  dislay_alv_report
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*---------------------------------------------------------------------*
FORM dislay_alv_report .

CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
 EXPORTING
*   I_INTERFACE_CHECK              = ' '
*   I_BYPASSING_BUFFER             =
```

```
*   I_BUFFER_ACTIVE                  = ' '
   I_CALLBACK_PROGRAM                =  g_repid
*   I_CALLBACK_PF_STATUS_SET         = ' '
*   I_CALLBACK_USER_COMMAND          = ' '
   I_STRUCTURE_NAME                  =  'sflight'
*   IS_LAYOUT                        =
   IT_FIELDCAT                       =  gt_fieldcat
*   IT_EXCLUDING                     =
*   IT_SPECIAL_GROUPS                =
*   IT_SORT                          =
*   IT_FILTER                        =
*   IS_SEL_HIDE                      =
   I_DEFAULT                        = 'X'
   I_SAVE                           = 'A'
  IS_VARIANT                       = GX_VARIANT
*   IT_EVENTS                        =
*   IT_EVENT_EXIT                    =
*   IS_PRINT                         =
*   IS_REPREP_ID                     =
*   I_SCREEN_START_COLUMN            = 0
*   I_SCREEN_START_LINE              = 0
*   I_SCREEN_END_COLUMN              = 0
*   I_SCREEN_END_LINE                = 0
* IMPORTING
*   E_EXIT_CAUSED_BY_CALLER          =
*   ES_EXIT_CAUSED_BY_USER           =
  TABLES
    T_OUTTAB                         = lt_sflight
 EXCEPTIONS
   PROGRAM_ERROR                    = 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.                    " dislay_alv_report
*&---------------------------------------------------------------------*
*&      Form  initialize_variant
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
FORM initialize_variant .
 g_save = 'A'.
  clear g_variant.
  g_variant-report = g_repid.
  gx_variant = g_variant.
  call function 'REUSE_ALV_VARIANT_DEFAULT_GET'
       exporting
            i_save     = g_save
       changing
            cs_variant = gx_variant
       exceptions
            not_found  = 2.
  if sy-subrc = 0.
    p_vari = gx_variant-variant.
  endif.

ENDFORM.                    " initialize_variant
```

Test ALV Display With Header & Footer

[日期：2006-10-18]　　来源：sap-img　作者：sapsky　　[字体：大 中 小]

Test ALV Display With Header & Footer

```
*Program to Test ALV Display With Header & Footer.
*&---------------------------------------------------------------------*
*& Report  ZRJR02                                                     *
*&---------------------------------------------------------------------*
REPORT  ZRJR02                                    .
```

```
*Table declaration.
TABLES:ZEMP_MST,ZDEPT_MST,ZDESG_MST,ZSL_TXN.

*Varriable declaration.
TYPE-POOLS SLIS.
DATA : POS TYPE I.
DATA  REPID  LIKE SY-REPID.

DATA : F1 TYPE SLIS_T_FIELDCAT_ALV,
       F2 TYPE SLIS_FIELDCAT_ALV,
       L_LAYOUT TYPE SLIS_LAYOUT_ALV.
DATA  L_POS TYPE I VALUE 1.                "position of the column
DATA  GT_FIELDCAT TYPE SLIS_T_FIELDCAT_ALV.
*DATA  GT_SORT TYPE SLIS_T_SORTINFO_ALV.

data:  GT_EVENTS    TYPE SLIS_T_EVENT,
       FS_EVENTCAT LIKE LINE OF GT_EVENTs.

*Internal table declaration.
*DATA  BEGIN OF IT_SORT OCCURS 5.
*      INCLUDE TYPE SLIS_SORTINFO_ALV.
*DATA  END OF IT_SORT.

DATA:BEGIN OF ITAB OCCURS 0,
      ZEMPNO    LIKE ZEMP_MST-ZEMPNO,
      ZEMPNAME  LIKE ZEMP_MST-ZEMPNAME,
      ZDEPTCD   LIKE ZEMP_MST-ZDEPTCD,
      ZDEPTNAME LIKE ZDEPT_MST-ZDEPTNAME,
      ZDESGCD   LIKE ZEMP_MST-ZDESGCD,
      ZDESGNAME LIKE ZDESG_MST-ZDESGNAME,
     END OF ITAB.
REFRESH ITAB.CLEAR ITAB.

START-OF-SELECTION.
```

```
SELECT A~ZEMPNO A~ZEMPNAME A~ZDEPTCD B~ZDEPTNAME A~ZDESGCD C~ZDESGNAME
       FROM ZEMP_MST AS A
         INNER JOIN ZDEPT_MST AS B
           ON A~ZDEPTCD EQ B~ZDEPTCD
         INNER JOIN ZDESG_MST AS C
           ON A~ZDESGCD EQ C~ZDESGCD
       INTO CORRESPONDING FIELDS OF TABLE ITAB.
IF SY-SUBRC <> 0.
   MESSAGE E899(M3) WITH 'No records'.
ENDIF.

perform f_build_eventcat.
PERFORM LAYOUT.

END-OF-SELECTION.

*&---------------------------------------------------------------------*
*&      Form  LAYOUT
*&---------------------------------------------------------------------*
FORM LAYOUT .
  PERFORM FCAT USING 'ZEMPNO'    'ITAB' '' 'Emp.No.'    'ZEMPNO'    'ZEMP_MST'  ''.
  PERFORM FCAT USING 'ZEMPNAME'  'ITAB' '' 'Emp. Name' 'ZEMPNAME'  'ZEMP_MST'  ''.
  PERFORM FCAT USING 'ZDEPTCD'   'ITAB' '' 'Dept.Code' 'ZDEPTCD'   'ZEMP_MST'  ''.
  PERFORM FCAT USING 'ZDEPTNAME' 'ITAB' '' 'Dept.Name' 'ZDEPTNAME' 'ZDEPT_MST' ''.
  PERFORM FCAT USING 'ZDESGCD'   'ITAB' '' 'Desg.Code' 'ZDESGCD'   'ZEMP_MST'  ''.
  PERFORM FCAT USING 'ZDESGNAME' 'ITAB' '' 'Desg.Name' 'ZDESGNAME' 'ZDESG_MST' ''.

*  PERFORM LSORT USING  'ZEMPNO' 'IDATA' ''.
*  PERFORM LSORT USING  'ZEMPNAME' 'IDATA' ''.

*  MOVE IT_SORT[] TO GT_SORT[].

  REPID = SY-REPID.
  CALL FUNCTION 'REUSE_ALV_LIST_DISPLAY'
    EXPORTING
         I_CALLBACK_PROGRAM       = REPID
```

```
        IT_FIELDCAT              = F1
*        IT_SORT                  = GT_SORT
        I_SAVE                   = 'X'
        IT_EVENTS                = GT_EVENTS[]

     TABLES
         T_OUTTAB                 = ITAB.
  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.                      " LAYOUT


*&---------------------------------------------------------------------*
*&      Form  FCAT
*&---------------------------------------------------------------------*
FORM FCAT USING P_FIELD P_TABLE P_SUM P_TEXT P_RFIELD P_RTABLE P_DISP.
  ADD 1 TO POS.
  F2-COL_POS       = POS.
  F2-FIELDNAME     = P_FIELD.
  F2-TABNAME       = P_TABLE.
  F2-SELTEXT_L     = P_TEXT.
  F2-REF_FIELDNAME = P_RFIELD.
  F2-REF_TABNAME   = P_RTABLE.
  F2-DO_SUM        = P_SUM.
  F2-NO_OUT        = P_DISP.
  APPEND F2 TO F1.
  CLEAR F2.
ENDFORM.                  " FCAT

*&---------------------------------------------------------------------*
*&      Form  LSORT
*&---------------------------------------------------------------------*
*FORM LSORT USING P_FIELD P_TABLE P_UP.
*  ADD 1 TO L_POS.
```

```
*  IT_SORT-SPOS      = L_POS.
*  IT_SORT-FIELDNAME = P_FIELD.
*  IT_SORT-TABNAME   = P_TABLE.
*  IT_SORT-UP        = P_UP.
*  APPEND IT_SORT.
*ENDFORM.                    " LSORT
*-------------
FORM F_BUILD_EVENTCAT .
 CLEAR: GT_EVENTS.  REFRESH: GT_EVENTS.

  CLEAR: FS_EVENTCAT.
  FS_EVENTCAT-NAME = 'TOP_OF_PAGE'.
  FS_EVENTCAT-FORM = 'F_REPORT_HEADER_ALV'.
  APPEND FS_EVENTCAT TO GT_EVENTS.



  CLEAR: FS_EVENTCAT.
  FS_EVENTCAT-NAME = 'END_OF_LIST'.
  FS_EVENTCAT-FORM = 'F_WRITE_SUMMARY'.
  APPEND FS_EVENTCAT TO GT_EVENTS.

ENDFORM.                    " F_BUILD_EVENTCAT

FORM F_REPORT_HEADER_ALV.
CALL FUNCTION 'Z_YHEAD_PRINT'
 EXPORTING
   TITLE1        = 'XYZ Limited'
   TITLE2        = 'Employee Master'
   TITLE3        = 'Created on '
   COLOR         = 'X'
          .
ENDFORM.

*&---------------------------------------------------------------------*
*&      Form  F_WRITE_SUMMARY
*&---------------------------------------------------------------------*
```

```
*       Write summary before exit
*-----------------------------------------------------------------------*
FORM F_WRITE_SUMMARY .

write:/ 'Welcome to XYZ Limited'.
write:/ 'This is a test program to display Report in ALV Format'.

ENDFORM.
```

How to make ALV header like this?

How to make ALV header like this?

Header long text 1 Header long text 2 Header long text 3

Col_1 Col_2 Col_3 Col_4 Col_5 Col_6 Col_7 Col_8 Col_9

---------- Cell conents ----------

---------- Cell conents ----------

---------- Cell conents ----------

---------- Cell conents ----------

---------- Cell conents ----------

You could try:

```
data: gt_list_top_of_page type slis_t_listheader. " Top of page text.

Initialization.
perform comment_build using gt_list_top_of_page[].

form top_of_page.
* Note to self: the gif must be loaded into transaction OAOR with
* classname 'PICTURES' AND TYPE 'OT' to work with ALV GRID Functions.
* I Loaded NOVALOGO2 into system.
 call function 'REUSE_ALV_COMMENTARY_WRITE'
    exporting
* I_LOGO = 'NOVALOGO2'
* i_logo = 'ENJOYSAP_LOGO'
       it_list_commentary = gt_list_top_of_page.

endform. " TOP_OF_PAGE
```

```
form comment_build using e04_lt_top_of_page type slis_t_listheader.
data: ls_line type slis_listheader.
      clear ls_line.
      ls_line-typ = 'A'.
      ls_line-info = 'Special'(001).
      fgrant = xgrant.
      concatenate ls_line-info fgrant
      'Stock Option Report to the board'(002)
            into ls_line-info separated by space.
                  condense ls_line-info.
      append ls_line to e04_lt_top_of_page.

endform. " COMMENT_BUILD
```

How to implement a footer in alv grid programming? What is the procedure and the code regarding to create a footer?

Use following syntex for footer print in alv:

```
* For End of Page

form END_OF_PAGE.

  data: listwidth type i,

        ld_pagepos(10) type c,

        ld_page(10)    type c.

  write: sy-uline(50).
  skip.
  write:/40 'Page:', sy-pagno .

endform.

*  For End of Report

form END_OF_LIST.

  data: listwidth type i,
```

```
    ld_pagepos(10) type c,
    ld_page(10)    type c.
  skip.
  write:/40 'Page:', sy-pagno .

endform.
```

Line Color in ALV Example

An example of using linecolor (ALV).

Here you have a good example of coloring rows, columns and specific cells in alvs. It comes in an example of how to use hashed tables.

```
For coloured rows and cols check gp_bymat
for coloured specific rows uncheck gp_bymat.
HTH.
Horacio
ps: code:
report zuseofhashedtables.
************************************************************************
** Program: ZUseOfHashedTables                                        **
************************************************************************
** Author: Horacio Zapettini                                          **
**                                                                    **
** Versions: 4.6b - 4.6c                                              **
************************************************************************
** Notes:                                                             **
**     this program shows how we can use hashed tables to improve     **
**     the responce time.                                             **
**     It shows,                                                      **
**        1. how to declare hashed tables                             **
**        2. a cache-like technique to improve access to master data  **
**        3. how to collect data using hashed tables                  **
**        4. how to avoid deletions of unwanted data                  **
************************************************************************
** Results: the test we run read about 31000 rows from mkpf, 150000   **
**          rows from mseg, 500 rows from makt and 400 from lfa1.      **
```

```
**          it filled ht_lst with 24500 rows and displayed them in    **
**          alv grid format.                                          **
**                                                                    **
**          It took about 65 secodns to perform this task (first time **
**          we run it when all the db buffers are empty.              **
**                                                                    **
**          The same program with standard tables needed 140 seconds  **
**          to run with the same recordset and with buffers filled in **
**                                                                    **
**          A simmilar test over more than a million rows
**********************************************************************
** Objetive: show a list that consists of  all the material movements **
**          '101' - '901' for a certain range of dates in mkpf-budat. **
** the columns to be displayed are:                                   **
**          mkpf-budat,                                               **
**          mkpf-mblnr,                                               **
**          mseg-lifnr,                                               **
**          lfa1-name1,                                               **
**          mkpf-xblnr,                                               **
**          mseg-zeile                                                **
**          mseg-charg,                                               **
**          mseg-matnr,                                               **
**          makt-maktx,                                               **
**          mseg-erfmg,                                               **
**          mseg-erfme.                                               **
** or show a sumary list by matnr - menge                             **
**                                                                    **
** You'll have to create a pf-status called vista -                   **
** See form set_pf_status for details                                 **
**********************************************************************

** tables used -
tables: mkpf,
        mseg,
        lfa1,
        makt.
```

```
** global hashed tables used
*
data: begin of wa_mkpf, "header
      mblnr like mkpf-mblnr,
      mjahr like mkpf-mjahr,
      budat like mkpf-budat,
      xblnr like mkpf-xblnr,
      end of wa_mkpf.
data: ht_mkpf like hashed table of wa_mkpf
      with unique key mblnr mjahr
      with header line.
data: st_mkpf like standard table of wa_mkpf
      with header line.

*
data: begin of wa_mseg, " line items
      mblnr like mseg-mblnr,
      mjahr like mseg-mjahr,
      zeile like mseg-zeile,
      bwart like mseg-bwart,
      charg like mseg-charg,
      matnr like mseg-matnr,
      lifnr like mseg-lifnr,
      erfmg like mseg-erfmg,
      erfme like mseg-erfme,
      end of wa_mseg.
data ht_mseg like hashed table of wa_mseg
      with unique key mblnr mjahr zeile
      with header line.
data st_mseg like standard table of wa_mseg
      with header line.


** cache structure for lfa1 records
data: begin of wa_lfa1,
```

```
      lifnr like lfa1-lifnr,
      name1 like lfa1-name1,
      end of wa_lfa1.
data ht_lfa1 like hashed table of wa_lfa1
      with unique key lifnr
      with header line.

** cache structure for material related data
data: begin of wa_material,
      matnr like makt-matnr,
      maktx like makt-maktx,
      end of wa_material.
data: ht_material like hashed table of wa_material
        with unique key matnr
        with header line.

** result table
data: begin of wa_lst, "
      budat like mkpf-budat,
      mblnr like mseg-mblnr,
      lifnr like mseg-lifnr,
      name1 like lfa1-name1,
      xblnr like mkpf-xblnr,
      zeile like mseg-zeile,
      charg like mseg-charg,
      matnr like mseg-matnr,
      maktx like makt-maktx,
      erfmg like mseg-erfmg,
      erfme like mseg-erfme,
      mjahr like mseg-mjahr,
      end of wa_lst.

data: ht_lst like hashed table of wa_lst
        with unique key mblnr mjahr zeile
        with header line.
```

```
data: begin of wa_lst1, " sumary by material
      matnr like mseg-matnr,
      maktx like makt-maktx,
      erfmg like mseg-erfmg,
      erfme like mseg-erfme,
      color_line(4) TYPE c,           " Line color
      color_cell    TYPE lvc_t_scol,  " Cell color
      celltab type LVC_T_STYL,
      end of wa_lst1.

data: ht_lst1 like hashed table of wa_lst1
        with unique key matnr
        with header line.

** structures for alv grid display.
** itabs
type-pools: slis.

data: it_lst            like standard table of wa_lst with header line,
      it_fieldcat_lst   type slis_t_fieldcat_alv with header line,
      it_sort_lst       type slis_t_sortinfo_alv,
      it_lst1           like standard table of wa_lst1 with header line,
      it_fieldcat_lst1  type slis_t_fieldcat_alv with header line,
      it_sort_lst1      type slis_t_sortinfo_alv.
** structures
data: wa_sort         type slis_sortinfo_alv,
      ls_layout       type slis_layout_alv.
** color management.
DATA  : wa_color    TYPE lvc_s_scol.
* Internal table for color management.
DATA : it_color    TYPE TABLE          OF lvc_s_scol.

* itab for input enabling.
DATA: lt_celltab TYPE lvc_t_styl. "
```

```
** global varialbes

data: g_lines type i.

data: g_repid like sy-repid,
      ok_code       like sy-ucomm.

** selection-screen

"text: Dates:
select-options: so_budat for mkpf-budat default sy-datum.
"text: Material numbers.
select-options: so_matnr for mseg-matnr.
selection-screen uline.
selection-screen skip 1.
"Text: show summary by material.
parameters: gp_bymat as checkbox default ''.
parameters: gp_hier  as checkbox default 'X'.
start-of-selection.
  perform get_data.
  perform show_data.

end-of-selection.



*---------------------------------------------------------------------*
*       FORM get_data                                                 *
*---------------------------------------------------------------------*
*       ........                                                      *
*---------------------------------------------------------------------*
form get_data.
        select mblnr mjahr budat xblnr
            into table ht_mkpf
           from mkpf
          where budat in so_budat. " make use of std index.
** have we retrieved data from mkpf?
```

```
  describe table ht_mkpf lines g_lines.
  if g_lines > 0.
** if true then retrieve all related records from mseg.
** Doing this way we make sure that the access is by primary key
** of mseg.
** The reason is that is faster to filter them in memory
** than to allow the db server to do it.
    select mblnr mjahr zeile bwart charg
             matnr lifnr erfmg erfme
      into table ht_mseg
      from mseg
        for all entries in ht_mkpf
     where mblnr = ht_mkpf-mblnr
       and mjahr = ht_mkpf-mjahr.
  endif.

** fill t_lst or t_lst1 according to user's choice.
  if gp_bymat = ' '.
    perform fill_ht_lst.
  else.
    perform fill_ht_lst1.
  endif.
endform.

form fill_ht_lst.
  refresh ht_lst.
** Example: how to discard unwanted data in an efficient way.
  loop at ht_mseg.
*   filter unwanted data
    check ht_mseg-bwart = '101' or ht_mseg-bwart = '901'.
    check ht_mseg-matnr in so_matnr.
*   read header line.
    read table ht_mkpf with table key mblnr = ht_mseg-mblnr
    mjahr = ht_mseg-mjahr.
    clear ht_lst.
*  * note : this may be faster if you specify field by field.
```

```
    move-corresponding ht_mkpf to ht_lst.
    move-corresponding ht_mseg to ht_lst.

    perform read_lfa1 using ht_mseg-lifnr changing ht_lst-name1.
    perform read_material using ht_mseg-matnr changing ht_lst-maktx.
    insert table ht_lst.
  endloop.
endform.

form fill_ht_lst1.
data: colorear.
  refresh ht_lst1.
** Example: how to discard unwanted data in an efficient way.
**          hot to simulate a collect in a faster way
  loop at ht_mseg.
*   filter unwanted data
    check ht_mseg-bwart = '101' or ht_mseg-bwart = '901'.
    check ht_mseg-matnr in so_matnr.
*  * note : this may be faster if you specify field by field.

    read table ht_lst1 with table key matnr = ht_mseg-matnr
    transporting erfmg.
    if sy-subrc <> 0. " if matnr doesn't exist in sumary table
    " insert a new record
      clear ht_lst1.
      ht_lst1-matnr = ht_mseg-matnr.
      perform read_material using ht_mseg-matnr changing ht_lst1-maktx.
      ht_lst1-erfmg = ht_mseg-erfmg.
      ht_lst1-erfme = ht_mseg-erfme.
      if colorear = ''.
        colorear = 'X'.
        refresh it_color.
        ht_lst1-color_cell[] = it_color[].
        MOVE 'C410' TO ht_lst1-color_line.
      else.
        colorear = ' '.
```

```
        refresh it_color. clear it_color.
        MOVE 'MATNR' TO wa_color-fname.
        MOVE '6'          TO wa_color-color-col.
        MOVE '1'          TO wa_color-color-int.
        MOVE '1'          TO wa_color-color-inv.
        APPEND wa_color TO it_color.
        MOVE 'MAKTX' TO wa_color-fname.
        MOVE '3'          TO wa_color-color-col.
        MOVE '1'          TO wa_color-color-int.
        MOVE '1'          TO wa_color-color-inv.
        APPEND wa_color TO it_color.

        MOVE 'ERFMG' TO wa_color-fname.
        MOVE '5'          TO wa_color-color-col.
        MOVE '1'          TO wa_color-color-int.
        MOVE '1'          TO wa_color-color-inv.

        APPEND wa_color TO it_color.
        ht_lst1-color_cell[] = it_color[].

        clear ht_lst1-color_line.

      endif.

      insert table ht_lst1.
    else." a record was found.
    " collect erfmg.  To do so, fill in the unique key and add
    " the numeric fields.
      ht_lst1-matnr = ht_mseg-matnr.
      add ht_mseg-erfmg to ht_lst1-erfmg.
      modify table ht_lst1 transporting erfmg.
    endif.
  endloop.
endform.

** implementation of cache for lfa1.
```

```
form read_lfa1 using p_lifnr changing p_name1.
        read table ht_lfa1 with table key lifnr = p_lifnr
        transporting name1.
  if sy-subrc <> 0.
    clear ht_lfa1.
    ht_lfa1-lifnr = p_lifnr.
    select single name1
       into ht_lfa1-name1
      from lfa1
    where lifnr = p_lifnr.
    if sy-subrc <> 0. ht_lfa1-name1 = 'n/a in lfa1'. endif.
    insert table ht_lfa1.
  endif.
  p_name1 = ht_lfa1-name1.
endform.

** implementation of cache for material data
form read_material using p_matnr changing p_maktx.
  read table ht_material with table key matnr = p_matnr
  transporting maktx.
  if sy-subrc <> 0.
    ht_material-matnr = p_matnr.
    select single maktx into  ht_material-maktx
      from makt
     where spras = sy-langu
       and matnr = p_matnr.
    if sy-subrc <> 0. ht_material-maktx = 'n/a in makt'. endif.
    insert table ht_material.
  endif.
  p_maktx = ht_material-maktx.
endform.
form show_data.
  if gp_hier = 'X'. "no anda.
*    perform show_hierarchicalALV.
  else.
    if gp_bymat = ' '.
```

```
      perform show_ht_lst.
    else.
      perform show_ht_lst1.
    endif.
  endif.
endform.
form show_hierarchicalALV.
st_mkpf[] = ht_mkpf[].
st_mseg[] = ht_mseg[].
call function 'REUSE_ALV_HIERSEQ_LIST_DISPLAY'
*  exporting
*   I_INTERFACE_CHECK              = ' '
*   I_CALLBACK_PROGRAM             =
*   I_CALLBACK_PF_STATUS_SET       = ' '
*   I_CALLBACK_USER_COMMAND        = ' '
*   IS_LAYOUT                      =
*   IT_FIELDCAT                    =
*   IT_EXCLUDING                   =
*   IT_SPECIAL_GROUPS              =
*   IT_SORT                        =
*   IT_FILTER                      =
*   IS_SEL_HIDE                    =
*   I_SCREEN_START_COLUMN          = 0
*   I_SCREEN_START_LINE            = 0
*   I_SCREEN_END_COLUMN            = 0
*   I_SCREEN_END_LINE              = 0
*   I_DEFAULT                      = 'X'
*   I_SAVE                         = ' '
*   IS_VARIANT                     =
*   IT_EVENTS                      =
*   IT_EVENT_EXIT                  =
*    i_tabname_header               =
*    i_tabname_item                 =
*   I_STRUCTURE_NAME_HEADER        =
*   I_STRUCTURE_NAME_ITEM          =
*    is_keyinfo                     =
```

```
*   IS_PRINT                        =
*   IS_REPREP_ID                    =
*   I_BUFFER_ACTIVE                 =
*   I_BYPASSING_BUFFER              =
* IMPORTING
*   E_EXIT_CAUSED_BY_CALLER         =
*   ES_EXIT_CAUSED_BY_USER          =
  tables
    t_outtab_header                 = st_mkpf
    t_outtab_item                   = st_mseg
* EXCEPTIONS
*   PROGRAM_ERROR                   = 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.
form show_ht_lst.
  "needed because the FM can't use a hashed table.
  it_lst[] = ht_lst[].

  perform fill_layout using 'full display'
                        changing ls_layout.

  perform fill_columns_lst.
*  perform sort_lst.
  g_repid = sy-repid.
  call function 'REUSE_ALV_GRID_DISPLAY'
       exporting
            i_callback_program       = g_repid
            i_callback_pf_status_set = 'SET_PF_STATUS'
            is_layout                = ls_layout
            it_fieldcat              = it_fieldcat_lst[]
```

```
*            it_sort                   = it_sort_lst
       tables
            t_outtab                 = it_lst
       exceptions
            program_error            = 1
            others                   = 2.

endform.
form show_ht_lst1.
  "needed because the FM can't use a hashed table.
  it_lst1[] = ht_lst1[].

  perform fill_layout using 'Sumary by matnr'
                      changing ls_layout.

  perform fill_columns_lst1.
*  perform sort_lst.
  g_repid = sy-repid.
  call function 'REUSE_ALV_GRID_DISPLAY'
       exporting
            i_callback_program       = g_repid
            i_callback_pf_status_set = 'SET_PF_STATUS'
            is_layout                = ls_layout
            it_fieldcat              = it_fieldcat_lst1[]
*            it_sort                   = it_sort_lst
       tables
            t_outtab                 = it_lst1
       exceptions
            program_error            = 1
            others                   = 2.

endform.

form fill_layout using p_window_titlebar
               changing cs_layo type slis_layout_alv.
  clear cs_layo.
```

```
  cs_layo-window_titlebar        = p_window_titlebar.
  cs_layo-edit                   = 'X'.
  cs_layo-edit_mode              = space.
  MOVE 'COLOR_LINE' TO cs_layo-info_fieldname.

* Field that identify cell color in inetrnal table
  MOVE 'COLOR_CELL' TO cs_layo-coltab_fieldname.
*  move 'CELLTAB' TO cs_layo-stylefname.



endform.                      " armar_layout_stock

form set_pf_status using rt_extab type slis_t_extab.
** create a new status
** and then select extras -> adjust template -> listviewer
  set pf-status 'VISTA'.
endform.         "set_pf_status
define add_lst.
  clear it_fieldcat_lst.
  it_fieldcat_lst-fieldname     = &1.
  it_fieldcat_lst-outputlen     = &2.
  it_fieldcat_lst-ddictxt       = 'L'.
  it_fieldcat_lst-seltext_l       = &1.
  it_fieldcat_lst-seltext_m       = &1.
  it_fieldcat_lst-seltext_m       = &1.
  if &1 = 'MATNR'.
    it_fieldcat_lst-emphasize = 'C111'.
  endif.
  append it_fieldcat_lst.
end-of-definition.
define add_lst1.
  clear it_fieldcat_lst.
  it_fieldcat_lst1-fieldname     = &1.
  it_fieldcat_lst1-outputlen     = &2.
  it_fieldcat_lst1-ddictxt       = 'L'.
  it_fieldcat_lst1-seltext_l       = &1.
```

```
  it_fieldcat_lst1-seltext_m       = &1.
  it_fieldcat_lst1-seltext_m       = &1.
  append it_fieldcat_lst1.
end-of-definition.

form fill_columns_lst.
* set columns for output.

  refresh it_fieldcat_lst.
*
  add_lst 'BUDAT' 10.
  add_lst   'MBLNR' 10.
  add_lst  'LIFNR' 10.
  add_lst  'NAME1' 35.
  add_lst  'XBLNR' 15.
  add_lst    'ZEILE' 5.
  add_lst    'CHARG' 10.
  add_lst   'MATNR' 18.
  add_lst   'MAKTX' 30.
  add_lst   'ERFMG' 17.
  add_lst   'ERFME' 5.
  add_lst   'MJAHR' 4.
endform.
form fill_columns_lst1.
* set columns for output.

  refresh it_fieldcat_lst1.

  add_lst1 'MATNR' 18.
  add_lst1 'MAKTX' 30.
  add_lst1 'ERFMG' 17.
  add_lst1 'ERFME' 5..
endform.
```

Horacio Zapettini

Program to Calculate FI Opening Balance

How to find the Opening balance for a given period in FI Module for a Particular GL A/c.

I was calculated opening balance, code is below maybe it will be helpful.

```
*find period.
  CALL FUNCTION 'DATE_TO_PERIOD_CONVERT'
    EXPORTING
      i_date         = s_budat-low
      i_periv        = i_tab-periv                          "'K4'
    IMPORTING
      e_buper        = v_donem
      e_gjahr        = v_gjahr
    EXCEPTIONS
      input_false    = 1
      t009_notfound  = 2
      t009b_notfound = 3
      OTHERS         = 4.
*calc opening balance hesabý
  SELECT * FROM knc1 WHERE kunnr = i_tab-kunnr
                     AND bukrs = i_tab-bukrs " s_bukrs
                     AND gjahr EQ v_gjahr.

    v_dnm = v_donem.
* opening balance first calc > old year ,
    WHILE v_dnm > 1.
      v_dnm = v_dnm - 1.
      CONCATENATE 'knc1-um' v_dnm 's' INTO v_field_name_borc.
      CONCATENATE 'knc1-um' v_dnm 'h' INTO v_field_name_alacak.
      ASSIGN (v_field_name_borc) TO <fs1> .
      ASSIGN (v_field_name_alacak) TO <fs2> .
      i_tab-dmbtr_s = i_tab-dmbtr_s + ( <fs1>  ). " borc
      i_tab-dmbtr_h = i_tab-dmbtr_h + ( <fs2>  ). " borc
```

```
    ENDWHILE.

*opening balance last calc> old
* add days which is from selected date-low month
    IF v_donem > 1.
      v_dnm = v_donem - 1.
    ELSE.
      v_dnm = v_donem.
    ENDIF.

    SELECT SINGLE * FROM t009b WHERE periv = i_tab-periv    "'K4'
                                AND bdatj = s_budat-low+0(4)
                                AND poper = v_dnm.

    t009b-butag = t009b-butag + 1.
    IF s_budat-low+6(2) NE t009b-butag.
      v_date_high = s_budat-low - 1.
      IF v_donem = 1.
        v_date_low = s_budat-low.
        v_date_low+4(4)  = '0101'.
      ELSE.
        CONCATENATE t009b-bdatj t009b-bumon t009b-butag INTO
        v_date_low.
      ENDIF.
      SELECT *  FROM bsad WHERE bukrs EQ i_tab-bukrs "IN s_bukrs
                            AND kunnr = i_tab-kunnr
                            AND budat BETWEEN v_date_low AND
                            v_date_high
                            AND umskz = space
                            AND blart IN s_blart.

        IF bsad-shkzg = 'S'.
          i_tab-dmbtr_s = i_tab-dmbtr_s + ( bsad-dmbtr ).
        ELSEIF bsad-shkzg = 'H'.
          i_tab-dmbtr_h = i_tab-dmbtr_h + ( bsad-dmbtr   ).
        ENDIF.
```

```
      ENDSELECT.
      SELECT *  FROM bsid WHERE bukrs EQ i_tab-bukrs "IN s_bukrs
                          AND kunnr = i_tab-kunnr
                          AND budat BETWEEN v_date_low AND
                          v_date_high
                          AND umskz = space
                          AND blart IN s_blart.
*                             AND gsber IN gsber.
        IF bsid-shkzg = 'S'.
          i_tab-dmbtr_s = i_tab-dmbtr_s + ( bsid-dmbtr   ).
        ELSEIF bsid-shkzg = 'H'.
          i_tab-dmbtr_h = i_tab-dmbtr_h + ( bsid-dmbtr   ).
        ENDIF.
      ENDSELECT.
    ENDIF.
    "opening balance ( þirket bazlý )z1 degeri
   i_tab-z1 = i_tab-z1 + ( knc1-umsav + i_tab-dmbtr_s - i_tab-dmbtr_h ).
* for israel
    i_tab-dmbtril_s = i_tab-dmbtr_s .
    i_tab-dmbtril_h = i_tab-dmbtr_h .
  ENDSELECT.
```

Download a report to excel with format (border, color cell, etc)

Try this program...it may help you to change the font ..etc.

Code:

```
REPORT ZSIRI NO STANDARD PAGE HEADING.
* this report demonstrates how to send some ABAP data to an
* EXCEL sheet using OLE automation.
INCLUDE OLE2INCL.
* handles for OLE objects
DATA: H_EXCEL TYPE OLE2_OBJECT,         " Excel object
      H_MAPL TYPE OLE2_OBJECT,          " list of workbooks
      H_MAP TYPE OLE2_OBJECT,           " workbook
```

```
      H_ZL TYPE OLE2_OBJECT,            " cell
      H_F TYPE OLE2_OBJECT.             " font
TABLES: SPFLI.
DATA  H TYPE I.
* table of flights
DATA: IT_SPFLI LIKE SPFLI OCCURS 10 WITH HEADER LINE.




*&---------------------------------------------------------------------*
*&   Event START-OF-SELECTION
*&---------------------------------------------------------------------*
START-OF-SELECTION.
* read flights
  SELECT * FROM SPFLI INTO TABLE IT_SPFLI UP TO 10 ROWS.
* display header
  ULINE (61).
  WRITE: /     SY-VLINE NO-GAP,
          (3)  'Flg'(001) COLOR COL_HEADING NO-GAP, SY-VLINE NO-GAP,
          (4)  'Nr'(002) COLOR COL_HEADING NO-GAP, SY-VLINE NO-GAP,
          (20) 'Von'(003) COLOR COL_HEADING NO-GAP, SY-VLINE NO-GAP,
          (20) 'Nach'(004) COLOR COL_HEADING NO-GAP, SY-VLINE NO-GAP,
          (8)  'Zeit'(005) COLOR COL_HEADING NO-GAP, SY-VLINE NO-GAP.
  ULINE /(61).
* display flights
  LOOP AT IT_SPFLI.
  WRITE: / SY-VLINE NO-GAP,
           IT_SPFLI-CARRID COLOR COL_KEY NO-GAP, SY-VLINE NO-GAP,
           IT_SPFLI-CONNID COLOR COL_NORMAL NO-GAP, SY-VLINE NO-GAP,
           IT_SPFLI-CITYFROM COLOR COL_NORMAL NO-GAP, SY-VLINE NO-GAP,
           IT_SPFLI-CITYTO COLOR COL_NORMAL NO-GAP, SY-VLINE NO-GAP,
           IT_SPFLI-DEPTIME COLOR COL_NORMAL NO-GAP, SY-VLINE NO-GAP.
  ENDLOOP.
  ULINE /(61).
* tell user what is going on
```

```
  CALL FUNCTION 'SAPGUI_PROGRESS_INDICATOR'
     EXPORTING
*           PERCENTAGE = 0
            TEXT       = TEXT-007
       EXCEPTIONS
            OTHERS     = 1.
* start Excel
  CREATE OBJECT H_EXCEL 'EXCEL.APPLICATION'.
*  PERFORM ERR_HDL.

  SET PROPERTY OF H_EXCEL  'Visible' = 1.
*  CALL METHOD OF H_EXCEL 'FILESAVEAS' EXPORTING #1 = 'c:\kis_excel.xls'
.

*  PERFORM ERR_HDL.
* tell user what is going on
  CALL FUNCTION 'SAPGUI_PROGRESS_INDICATOR'
     EXPORTING
*           PERCENTAGE = 0
            TEXT       = TEXT-008
       EXCEPTIONS
            OTHERS     = 1.
* get list of workbooks, initially empty
  CALL METHOD OF H_EXCEL 'Workbooks' = H_MAPL.
  PERFORM ERR_HDL.
* add a new workbook
  CALL METHOD OF H_MAPL 'Add' = H_MAP.
  PERFORM ERR_HDL.
* tell user what is going on
  CALL FUNCTION 'SAPGUI_PROGRESS_INDICATOR'
     EXPORTING
*           PERCENTAGE = 0
            TEXT       = TEXT-009
       EXCEPTIONS
            OTHERS     = 1.
* output column headings to active Excel sheet
```

```
  PERFORM FILL_CELL USING 1 1 1 'Flug'(001).
  PERFORM FILL_CELL USING 1 2 0 'Nr'(002).
  PERFORM FILL_CELL USING 1 3 1 'Von'(003).
  PERFORM FILL_CELL USING 1 4 1 'Nach'(004).
  PERFORM FILL_CELL USING 1 5 1 'Zeit'(005).
  LOOP AT IT_SPFLI.
* copy flights to active EXCEL sheet
    H = SY-TABIX + 1.
    PERFORM FILL_CELL USING H 1 0 IT_SPFLI-CARRID.
    PERFORM FILL_CELL USING H 2 0 IT_SPFLI-CONNID.
    PERFORM FILL_CELL USING H 3 0 IT_SPFLI-CITYFROM.
    PERFORM FILL_CELL USING H 4 0 IT_SPFLI-CITYTO.
    PERFORM FILL_CELL USING H 5 0 IT_SPFLI-DEPTIME.
  ENDLOOP.

* changes by Kishore  - start
*  CALL METHOD OF H_EXCEL 'Workbooks' = H_MAPL.
  CALL METHOD OF H_EXCEL 'Worksheets' = H_MAPL." EXPORTING #1 = 2.

  PERFORM ERR_HDL.
* add a new workbook
  CALL METHOD OF H_MAPL 'Add' = H_MAP  EXPORTING #1 = 2.
  PERFORM ERR_HDL.
* tell user what is going on
  SET PROPERTY OF H_MAP 'NAME' = 'COPY'.
  CALL FUNCTION 'SAPGUI_PROGRESS_INDICATOR'
     EXPORTING
*           PERCENTAGE = 0
            TEXT       = TEXT-009
       EXCEPTIONS
            OTHERS     = 1.
* output column headings to active Excel sheet
  PERFORM FILL_CELL USING 1 1 1 'Flug'(001).
  PERFORM FILL_CELL USING 1 2 0 'Nr'(002).
  PERFORM FILL_CELL USING 1 3 1 'Von'(003).
  PERFORM FILL_CELL USING 1 4 1 'Nach'(004).
```

```
  PERFORM FILL_CELL USING 1 5 1 'Zeit'(005).
  LOOP AT IT_SPFLI.
* copy flights to active EXCEL sheet
    H = SY-TABIX + 1.
    PERFORM FILL_CELL USING H 1 0 IT_SPFLI-CARRID.
    PERFORM FILL_CELL USING H 2 0 IT_SPFLI-CONNID.
    PERFORM FILL_CELL USING H 3 0 IT_SPFLI-CITYFROM.
    PERFORM FILL_CELL USING H 4 0 IT_SPFLI-CITYTO.
    PERFORM FILL_CELL USING H 5 0 IT_SPFLI-DEPTIME.
  ENDLOOP.
* changes by Kishore  - end
* disconnect from Excel
*      CALL METHOD OF H_EXCEL 'FILESAVEAS' EXPORTING  #1 = 'C:\SKV.XLS'.

  FREE OBJECT H_EXCEL.
  PERFORM ERR_HDL.
*---------------------------------------------------------------------*
*       FORM FILL_CELL                                                *
*---------------------------------------------------------------------*
*       sets cell at coordinates i,j to value val boldtype bold       *
*---------------------------------------------------------------------*
FORM FILL_CELL USING I J BOLD VAL.
  CALL METHOD OF H_EXCEL 'Cells' = H_ZL EXPORTING #1 = I #2 = J.
  PERFORM ERR_HDL.
  SET PROPERTY OF H_ZL 'Value' = VAL .
  PERFORM ERR_HDL.
  GET PROPERTY OF H_ZL 'Font' = H_F.
  PERFORM ERR_HDL.
  SET PROPERTY OF H_F 'Bold' = BOLD .
  PERFORM ERR_HDL.
ENDFORM.
*&---------------------------------------------------------------------*
*&      Form  ERR_HDL
*&---------------------------------------------------------------------*
*       outputs OLE error if any                                      *
*---------------------------------------------------------------------*
```

```
*  -->  p1        text
*  <--  p2        text
*---------------------------------------------------------------------*
FORM ERR_HDL.
IF SY-SUBRC <> 0.
  WRITE: / 'Fehler bei OLE-Automation:'(010), SY-SUBRC.
  STOP.
ENDIF.
ENDFORM.                    " ERR_HDL
```

Please note that this example maybe slow at filling the excel table (perhaps four fields per second on a 900 MHz machine - almost 30 seconds for a short example).

To get the data on properties and methods - there is a bit of smoke and mirrors going on here; they are EXCEL properties and methods, not sap ones - so you need to look at excel help to determine how a particular function is structured. then build the block in sap, as shown in the example.

If you only want to transfer the data to Excel like when you transfer the data from ALV to Excel simply use the Function Modules:

XXL_SIMPLE_API

If you want more modifications when you transfer it to Excel use:

XXL_FULL_API

Display a Secondary List using ALV Grid

To display a secondary list when you click on one of the row items in an alv grid. The secondary list should a lso be an alv.

Try out this code. You will have to make a structure ZSTR same as the output internal table.

```
REPORT ZTEST_REP1 .
```

```
TABLES : MARA,
         BHDGD,
         zstr.

TYPES:     BEGIN OF T_MARA,
             MATNR LIKE MARA-MATNR,
             ERNAM LIKE MARA-ERNAM,
           END OF T_MARA.


CLASS LCL_EVENT_RECEIVER DEFINITION DEFERRED.
*Constants for ALV Implementation
CONSTANTS:  C_SET                    VALUE 'X',
            C_RESET                  VALUE '0',
            C_SAVE                   VALUE 'A',
            C_EXIT(4)                VALUE 'EXIT',
            C_BACK(4)                VALUE 'BACK',
            C_CANC(4)                VALUE 'CANC',
            C_PGTOP(5)               VALUE 'PGTOP',
            C_PGUP(4)                VALUE 'PGUP',
            C_PGDN(4)                VALUE 'PGDN',
            C_PGEND(5)               VALUE 'PGEND'.



DATA : I_MARA TYPE STANDARD TABLE OF T_MARA WITH HEADER LINE,

*      Internal table for fields catalouge
       I_FIELDCAT TYPE LVC_T_FCAT WITH HEADER LINE,
*       i_fieldcat2 type lvc_t_fcat with header line,

*      Internal table for cursor position
       I_GT_SELROWS TYPE LVC_T_ROW .

DATA : WA_MARA LIKE I_MARA,
       WA_GRIDROW LIKE LVC_S_ROW,
       WA_GRIDCOL LIKE LVC_S_COL.
```

```
*Data for ALV Implementation.
DATA:  OK_CODE      LIKE SY-UCOMM,
       W_OK_CODE    LIKE SY-UCOMM,
       W_CALL       TYPE I VALUE 1,
       W_TAB        LIKE SY-UCOMM VALUE 'TAB1',
       W_SAVE,                          "For Parameter I_SAVE
       W_VARIANT    TYPE DISVARIANT,        "For parameter IS_VARIANT
       W_GRID       TYPE REF TO CL_GUI_ALV_GRID,
*       w_grid1      type ref to cl_gui_alv_grid,
       W_CONTAINER  TYPE REF TO CL_GUI_CUSTOM_CONTAINER,

*       w_container1  type ref to cl_gui_custom_container,

       W_REPID      LIKE SY-REPID,
       W_GS_PRINT   TYPE LVC_S_PRNT,
       W_GS_LAYOUT  TYPE LVC_S_LAYO,
       W_EVENT_REC  TYPE REF TO LCL_EVENT_RECEIVER,
       W_CONT_MAIN  TYPE SCRFNAME VALUE 'CCCONTAINER',
       W_LN         TYPE I,                    "line number
       W_INDEX      LIKE SY-TABIX,
       W_FLAG,
       W_TEMP_VAL   TYPE I.

*---------------------------------------------------------------------*
*  Definition:
*---------------------------------------------------------------------*
CLASS LCL_EVENT_RECEIVER DEFINITION.

  PUBLIC SECTION.
    METHODS:
    HANDLE_TOP_OF_PAGE
      FOR EVENT PRINT_TOP_OF_PAGE OF CL_GUI_ALV_GRID,
    HANDLE_DOUBLE_CLICK
      FOR EVENT DOUBLE_CLICK OF CL_GUI_ALV_GRID
          IMPORTING E_ROW E_COLUMN.
ENDCLASS.
```

```
*---------------------------------------------------------------------*
* CLASS LCL_EVENT_RECEIVER IMPLEMENTATION
*---------------------------------------------------------------------*
CLASS LCL_EVENT_RECEIVER IMPLEMENTATION.
  METHOD HANDLE_TOP_OF_PAGE.
    PERFORM F_GET_HEADER.
  ENDMETHOD.                    "handle_top_of_page

  METHOD HANDLE_DOUBLE_CLICK.

*   The event DOUBLE_CLICK provides parameters for row and column
*   of the click. We use row parameter to select a line of the
*   corresponding internal table.

*   read selected row from internal table
      READ TABLE I_MARA INDEX E_ROW-INDEX INTO WA_MARA.
      IF SY-SUBRC <> 0.
*       message i001. " Cursor position not correct.
      ELSE.
*       call dialog screen and display the details
        call screen 200 starting at 10 5.
      ENDIF.



  ENDMETHOD.                    "handle_double_click

ENDCLASS.

*---------------------------------------------------------------------
* start-of-selection.
*---------------------------------------------------------------------
START-OF-SELECTION.

SELECT MATNR ERNAM FROM MARA INTO TABLE I_MARA.

*---------------------------------------------------------------------
* End-of-Selection.
```

```
*---------------------------------------------------------------------
END-OF-SELECTION.

* Start of ALV part.
  W_REPID = SY-REPID.
  W_VARIANT-REPORT = W_REPID.
  W_SAVE = C_SAVE.
  W_CONT_MAIN = W_CONT_MAIN.
  W_GS_LAYOUT = W_GS_LAYOUT.
  W_GS_PRINT  = W_GS_PRINT.
  I_FIELDCAT  = I_FIELDCAT.
  CALL SCREEN 100.

*&---------------------------------------------------------------------*
*&      Form  f_get_header
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
FORM F_GET_HEADER.

  DATA: L_LINE1 LIKE BHDGD-LINE1,
        L_LINE2 LIKE BHDGD-LINE2.
  CONSTANTS LC_SPACE VALUE ' '.
  DATA: L_F1(7), L_F2(11), L_F3(9), L_F4(6), L_F5(11), L_F6(4), L_F7(8),
        L_F8(4),L_F9(10), L_F11(11), L_F12(24), L_F13(4),
        L_F14(3).

* take the values of line1 and line2 into two new variables, otherwise
* after coming back to the first screen from the print preview, the
* header shows the condensed lines

  L_LINE1 = BHDGD-LINE1.
  L_LINE2 = BHDGD-LINE2.

  CONDENSE L_LINE1.
  CONDENSE L_LINE2.
```

```
*split the lines to display the whole lines within the
*stipulated report-width
 SPLIT L_LINE1 AT LC_SPACE INTO L_F1 L_F2 L_F3 L_F4 L_F5 L_F6 L_F7 L_F8
                          L_F9 .

  SPLIT L_LINE2 AT LC_SPACE INTO L_F11 L_F12 L_F13 L_F14.
  L_F14 = SY-PAGNO.
    WRITE:/1 L_F1, 9 L_F2, 40 L_F3, 50 L_F4, 57 L_F5, 88 L_F6, 93 L_F7 ,
         103 L_F8 , 108 L_F9 .
    WRITE:/1 L_F11, 40 TEXT-012, 78 L_F12, 103 L_F13, 108 L_F14.


ENDFORM.                    " f_get_header
*&---------------------------------------------------------------------*
*&      Module  STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE STATUS_0100 OUTPUT.
  SET PF-STATUS 'STAT'.
  SET TITLEBAR 'TITL'.

ENDMODULE.                  " STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE USER_COMMAND_0100 INPUT.
  CASE SY-UCOMM .
    WHEN C_EXIT OR C_BACK OR C_CANC.
      IF NOT W_CONTAINER  IS INITIAL.
        CALL METHOD W_CONTAINER->FREE.
      ENDIF.

      LEAVE TO SCREEN 0.
```

```
    WHEN C_PGTOP.
      WA_GRIDROW-INDEX = 1.

    WHEN C_PGUP.
      IF WA_GRIDROW-INDEX <= 15.
        WA_GRIDROW-INDEX = 1.
      ELSE.
        WA_GRIDROW-INDEX = WA_GRIDROW-INDEX - 15.
      ENDIF.

    WHEN C_PGDN.
      PERFORM F_GET_NO_ROWS.
      W_TEMP_VAL = W_LN - WA_GRIDROW-INDEX.
      IF W_TEMP_VAL < 15.
        WA_GRIDROW-INDEX = W_LN.
      ELSE.
        WA_GRIDROW-INDEX = WA_GRIDROW-INDEX + 15.
      ENDIF.

    WHEN C_PGEND.
      PERFORM F_GET_NO_ROWS.
      WA_GRIDROW-INDEX = W_LN.

  ENDCASE.


ENDMODULE.                " USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
*&      Form  f_get_no_rows
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
FORM F_GET_NO_ROWS.

    DESCRIBE TABLE I_MARA  LINES W_LN.
```

```
ENDFORM.                    " f_get_no_rows
*&---------------------------------------------------------------------*
*&      Module  DISPLAY_0100  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE DISPLAY_0100 OUTPUT.

  IF NOT WA_GRIDROW IS INITIAL
  AND NOT WA_GRIDCOL IS INITIAL.

    CALL METHOD W_GRID->SET_SCROLL_INFO_VIA_ID
      EXPORTING
        IS_ROW_INFO = WA_GRIDROW
        IS_COL_INFO = WA_GRIDCOL .

    CALL METHOD W_GRID->SET_CURRENT_CELL_VIA_ID
      EXPORTING
        IS_ROW_ID = WA_GRIDROW
        IS_COLUMN_ID = WA_GRIDCOL .

  ENDIF.

  CALL METHOD W_GRID->GET_SCROLL_INFO_VIA_ID
  IMPORTING
    ES_ROW_INFO = WA_GRIDROW
    ES_COL_INFO = WA_GRIDCOL .

  CALL METHOD W_GRID->GET_SELECTED_ROWS
    IMPORTING
      ET_INDEX_ROWS = I_GT_SELROWS[].

* Build the fieldcat according to structure
  CALL FUNCTION 'LVC_FIELDCATALOG_MERGE'
      EXPORTING
          I_STRUCTURE_NAME = 'ZSTR'
      CHANGING
          CT_FIELDCAT      = I_FIELDCAT[].
```

```
LOOP AT I_FIELDCAT.
  W_INDEX = SY-TABIX.
  CASE I_FIELDCAT-FIELDNAME.

    WHEN 'MATNR'.
      I_FIELDCAT-SCRTEXT_S = 'MATNR'.
      I_FIELDCAT-KEY       = ' '.
      I_FIELDCAT-COL_POS   = '1'.

    WHEN 'ERNAM'.
      I_FIELDCAT-SCRTEXT_S = 'ERDAT'.
      I_FIELDCAT-OUTPUTLEN = '18'.
      I_FIELDCAT-COL_POS   = '2'.



  ENDCASE.
  MODIFY I_FIELDCAT INDEX W_INDEX.

ENDLOOP.

READ TABLE I_FIELDCAT INDEX 1 .

IF W_CALL = 1.

  PERFORM F_STD_HEADER.

  CALL METHOD W_GRID->SET_TABLE_FOR_FIRST_DISPLAY
              EXPORTING
                IS_VARIANT                = W_VARIANT
                I_SAVE                    = W_SAVE
              CHANGING
                IT_OUTTAB                 = I_MARA[]
                IT_FIELDCATALOG           = I_FIELDCAT[]
             EXCEPTIONS
               INVALID_PARAMETER_COMBINATION = 1
               PROGRAM_ERROR               = 2
               OTHERS                      = 3.
```

```
    IF SY-SUBRC <> 0.
     MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
            WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
     EXIT.
    ENDIF.

    CREATE OBJECT W_EVENT_REC.
    SET HANDLER W_EVENT_REC->HANDLE_TOP_OF_PAGE FOR W_GRID.
    CREATE OBJECT W_EVENT_REC.
    SET HANDLER W_EVENT_REC->HANDLE_DOUBLE_CLICK FOR W_GRID.
    W_FLAG = C_RESET.


    CALL METHOD CL_GUI_CONTROL=>SET_FOCUS EXPORTING CONTROL = W_GRID.
    W_CALL = 0.
  ENDIF.


ENDMODULE.              " DISPLAY_0100  OUTPUT
*&---------------------------------------------------------------------*
*&      Form  f_std_header
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*----------------------------------------------------------------------*
FORM F_STD_HEADER.

ENDFORM.                  " f_std_header
*&---------------------------------------------------------------------*
*&      Module  DYNPRONR_CHECK_500  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE DYNPRONR_CHECK_500 OUTPUT.
*  if w_dynpronr is initial.
```

```
*    w_dynpronr = '0100'.
*   endif.

  ENDMODULE.                " DYNPRONR_CHECK_500  OUTPUT
*&---------------------------------------------------------------------*
*&      Module  create_objects_0100  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE create_objects_0100 OUTPUT.
  check w_container is initial .
  create object w_container
    exporting
      container_name = 'CC'.
  create object w_grid
    exporting
      i_parent = w_container.
  w_flag = c_set.
  w_flag = w_flag.
ENDMODULE.                " create_objects_0100  OUTPUT
*&---------------------------------------------------------------------*
*&      Module  STATUS_0200  OUTPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
MODULE STATUS_0200 OUTPUT.
  SET PF-STATUS 'ST20'.
  SET TITLEBAR '200'.
  zstr-matnr    = wa_mara-matnr.
  zstr-ernam    = wa_mara-ernam.

ENDMODULE.                " STATUS_0200  OUTPUT
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0200  INPUT
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
```

```
MODULE USER_COMMAND_0200 INPUT.
  move ok_code to w_ok_code.
  clear ok_code.
  case w_ok_code.
   when c_back or c_exit or c_canc.
    leave to screen 0.
  endcase.
  clear w_ok_code.
ENDMODULE.            " USER_COMMAND_0200  INPUT

*-- End of Program
```

How to use ALV for Hierarchical Lists

Can anyone tell me how to use ALV for hierarchical lists using the function code REUSE_ALV_HIERSEQ_LIST_DISPLAY?

Swarna

Hello, there are some nice examples in SAP which use this function module, so you might want to check them out (where used etc.)

In essence, this is a call in one of my ABAPs

```
  CALL FUNCTION 'REUSE_ALV_HIERSEQ_LIST_DISPLAY'
    EXPORTING
      i_interface_check              = 'I'
      i_callback_program             = gv_repid
*      i_callback_pf_status_set       = 'STATUS_DATA'
      i_callback_user_command        = 'COMMAND_DATA'
*      is_layout                      = gs_layout
      it_fieldcat                    = gt_fieldcat
     i_default                      = ' '
      i_save                         = 'A'
      i_tabname_header               = v_headers_itable
      i_tabname_item                 = v_items_itable
      i_structure_name_header        = v_headers_table
```

```
      i_structure_name_item          = v_items_table
      is_keyinfo                     = gs_keyinfo
      i_bypassing_buffer             = 'X'
    TABLES
      t_outtab_header                = i_headers
*      t_outtab_item                  = i_result
      t_outtab_item                  = i_report
    EXCEPTIONS
      program_error                  = 1
      OTHERS                         = 2.
```

The field cat creation worked like this :

```
FORM fieldcat.

  DATA: ls_fieldcat TYPE slis_fieldcat_alv.

  CALL FUNCTION 'REUSE_ALV_FIELDCATALOG_MERGE'
       EXPORTING
            i_internal_tabname = v_items_itable
            i_structure_name   = v_items_table
       CHANGING
            ct_fieldcat        = gt_fieldcat.

  CALL FUNCTION 'REUSE_ALV_FIELDCATALOG_MERGE'
       EXPORTING
            i_internal_tabname = v_headers_itable
            i_structure_name   = v_headers_table
       CHANGING
            ct_fieldcat        = gt_fieldcat.

ENDFORM.
```

and of course you need to tell the thing what is key and item

```
  gs_keyinfo-header01 = 'PA'.
```

```
gs_keyinfo-item01  = 'PA'.
gs_keyinfo-item02 = 'SAPDOC'.
PERFORM fieldcat.
```

I hope this helps you and not confuse you,

Cheers.

ALV 'Classic' Creating User/Global Layout Variants

You are working with the ALV "Classic" function module REUSE_ALV_LIST_DISPLAY.

Reading the documentation for the function module, it seems there is a way to save an ALV layout variant as "user-specific" and/or "global". But unfortunately, you either can only save "globals" (with the '/' as first character of the layout name) or "user-specific".

You have tried the I_SAVE parameter as 'U' and can only save "user-specific". You tried I_SAVE as 'X' and can only save "global". You tried I_SAVE as 'A', but only can only save as "user-specific". The odd thing is, on the Save Layout pop-up dialog the User-Specific checkbox is always "greyed-out", but has a check (for 'U' and 'A') or is checkless (for 'X').

Can "user" and "global" layout variants be saved together from same program with I_SAVE as 'A'?

Why is the User-Specific checkbox on the Save Layout pop-up always "greyed-out"?

You have the following EXPORTING parameters in my function module:

I_DEFAULT = 'X'

I_SAVE = 'A' "<=== this is to be global & user IS_VARIANT = VARIANT

VARIANT has the program's name in the REPORT field.

The "user-specific saving" needs a special authorization:

Authority-check object **'S_ALV_LAYO' id 'ACTVT' field '23'**, you can

avoid this authorization with **IS_LAYOUT-NO_AUTHOR = 'X'**.

function REUSE_ALV_FIELDCATALOG_MERGE

An example :-

Please note that structure ZSTOCK is a custom table and iline looks as follow :-

```
data: iline type table of zstock with header line.
data: gt_fieldcat        type slis_t_fieldcat_alv.

  perform setup-fieldcatalog using gt_fieldcat[].

form setup-fieldcatalog using _fieldcat type slis_t_fieldcat_alv.
  data: ls_fieldcat type slis_fieldcat_alv.

  call function 'REUSE_ALV_FIELDCATALOG_MERGE'
      exporting
          i_internal_tabname = 'ILINE'
          i_structure_name   = 'ZSTOCK'
      changing
          ct_fieldcat       = _fieldcat.

  loop at _fieldcat into ls_fieldcat.
    case ls_fieldcat-fieldname.
      when 'DEPT'.
       ls_fieldcat-no_out    = 'X'.
      when 'DESCR'.
       ls_fieldcat-no_out    = 'X'.
      when 'GOOD_PRD'.
       ls_fieldcat-do_sum    = 'X'.
    endcase.

    modify _fieldcat from ls_fieldcat.
  endloop.

endform.                  " FIELDCATALOG
```

REUSE_ALV_GRID_DISPLAY Functions Example

I am using "REUSE_ALV_GRID_DISPLAY" to display Report in ALV.

I have to show System Date and Time at the end of Report so I caught Event "END_OF_PAGE" in IT_EVENTS and modified IT_EVENTS field FORM with "F100_TOP_OF_PAGE", but I am not able to see the Date and Time in the END OF PAGE

I wrote follwoing ocde so please suggest me necessary changes

```
*&---------------------------------------------------------------------*
*& Form f100-end_of_page
*&---------------------------------------------------------------------*
* text
*----------------------------------------------------------------------*
* --> p1 text
*  <-- p2 text
*----------------------------------------------------------------------*

FORM f100-end_of_page.
clear: wa_listheader, it_end_listheader, it_end_listheader[].
concatenate 'Date' sy-datum into w_date separated by space.

wa_listheader-key = ''.
wa_listheader-typ = 'S'.
wa_listheader-info = 'Date'.

append wa_listheader to it_end_listheader.

clear: wa_listheader.

concatenate 'Time' sy-uzeit into w_date separated by space.

wa_listheader-key = ''.
wa_listheader-typ = 'S'.
wa_listheader-info = 'Time'.

append wa_listheader to it_end_listheader.

clear: wa_listheader.

CALL FUNCTION 'REUSE_ALV_COMMENTARY_WRITE'
EXPORTING
IT_LIST_COMMENTARY = it_end_listheader[]
* I_LOGO =
* I_END_OF_LIST_GRID =
.

ENDFORM. " f100-end_of_page
```

In grid display there will be no end_of_page and no end_of_list.

You have show in top_of_page itself.

Otherwise use list display instead of grid.

Suresh Avutu

These steps have to be carried out...

```
DATA:  gc_formname_top_of_page TYPE slis_formname
       VALUE 'TOP_OF_PAGE'.
DATA:  gc_formname_end_of_page TYPE slis_formname
       VALUE 'END_OF_PAGE'.

DATA:  gt_list_top_of_page TYPE slis_t_listheader.
DATA:  gt_list_end_of_page TYPE slis_t_listheader.
```

```
PERFORM e03_eventtab_build USING gt_event[].
        PERFORM e04_comment_build  USING
gt_list_top_of_page[].
        PERFORM e06_comment_build  USING
gt_list_end_of_page[].
        PERFORM list_alv_display.
```

```
FORM list_alv_display .

  gt_event-name = slis_ev_top_of_list.
  gt_event-form = 'TOP_OF_PAGE'.
  APPEND gt_event.

  gt_event-name = slis_ev_end_of_list.
  gt_event-form = 'END_OF_PAGE'.
  APPEND gt_event.
```

```
  CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
    EXPORTING
*   I_INTERFACE_CHECK              = ' '
    i_bypassing_buffer             = 'X'
    i_buffer_active                = ' '
    i_callback_program             = ....
---------------------------------------
FORM end_of_page.
  CALL FUNCTION 'REUSE_ALV_COMMENTARY_WRITE'
    EXPORTING
      i_logo           = ''
      it_list_commentary = gt_list_end_of_page
      I_END_OF_LIST_GRID = 1.

ENDFORM.                    "END_OF_PAGE
----------------------------------------
FORM e06_comment_build
 USING e06_lt_end_of_page TYPE slis_t_listheader.
  DATA: ls_line  TYPE slis_listheader.
  DATA: yl_uname(60).

 < here u can concatenate system date and time to
yl_uname>

  CLEAR ls_line.
  ls_line-typ  = 'S'.
  ls_line-info = yl_uname.
  APPEND ls_line TO e06_lt_end_of_page.

endform.                    "e06_comment_build
```

Suman Tyagi

An Example:

```
REPORT ZALV_GRID.
```

```
TABLES :vbap.
type-pools : slis.
data i_events TYPE slis_t_event.
DATA : my_alv TYPE REF TO cl_gui_alv_grid.
TYPES : BEGIN OF itab,
vbeln LIKE vbap-vbeln,
arktx LIKE vbap-arktx,
END OF itab.
TYPES : itab1 TYPE TABLE OF itab.
DATA : display TYPE itab1.
DATA : fcat TYPE SLIS_T_FIELDCAT_ALV.
DATA : wa LIKE LINE OF FCAT.
DATA WA1 LIKE VBAP.
DATA: container TYPE REF TO cl_gui_custom_container.
data report_id like sy-repid.

SELECT-OPTIONS s_vbeln FOR vbap-vbeln.
report_id = sy-repid.

SELECT * FROM vbap INTO CORRESPONDING FIELDS OF TABLE display WHERE
vbeln IN s_vbeln.

wa-fieldname = 'VBELN'.
wa-tabname = 'VBAP'.
wa-key = 'X'.
WA-HOTSPOT = 'X'.
wa-text_fieldname = 'Doc no.'.
APPEND wa TO fcat.
CLEAR wa.
wa-fieldname = 'ARKTX'.
wa-tabname = 'VBAP'.
wa-text_fieldname = 'Item Text'.
APPEND wa TO fcat.

PERFORM f0650_build_event USING 'USER_COMMAND'
'F0750_USER_COMMAND'.
```

```
CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY '
EXPORTING

I_CALLBACK_PROGRAM = report_id

IT_FIELDCAT = FCAT
IT_EVENTS = i_events
TABLES
t_outtab = DISPLAY
.

FORM f0650_build_event USING value(w_c_event_name)
value(w_c_event_form).

DATA: f0650_wa_event TYPE slis_alv_event.

CLEAR f0650_wa_event.
f0650_wa_event-name = w_c_event_name.
f0650_wa_event-form = w_c_event_form.
APPEND f0650_wa_event TO i_events.

ENDFORM.
FORM f0750_user_command USING w_ucomm TYPE sy-ucomm

w_selfield TYPE slis_selfield.

CASE w_ucomm.
WHEN '&IC1'.

READ TABLE DISPLAY INTO WA1 INDEX w_selfield-tabindex.
* MESSAGE E000 WITH
* ' You have no authorization to view the report'.
call transaction 'SE11'.

ENDCASE.
ENDFORM.

*** End of Program
```

Sample programs on ALV Grid

```
report zbnstest.
************************************************************************
* TABLES AND DATA DECLARATION.
************************************************************************
*TABLES: mara,makt.",marc.
data syrepid like sy-repid.
data sydatum(10). " LIKE sy-datum.
data sypagno(3) type n.


* WHEN USING MORE THAN ONE TABLE IN ALV WE NEEED TO DECLARE THE TYPE
* GROUP (TYPE-POOLS--------->SLIS)
type-pools : slis.


************************************************************************
* INTERNAL TABLE DECLARATION.
************************************************************************
* INTERNAL TABLE TO HOLD THE VALUES FROM THE MARA TABLE
data: begin of t_mara occurs 0,
matnr like mara-matnr,
meins like mara-meins,
mtart like mara-mtart,
matkl like mara-matkl,

end of t_mara.

* INTERNAL TABLE TO HOLD THE CONTENTS FROM THE EKKO TABLE
data : begin of t_marc occurs 0,
matnr like mara-matnr,
werks like marc-werks,
minbe like marc-minbe.
data: end of t_marc.

* INTERNAL TABLE TO HOLD THE VALUES FROM MAKT TABLE.
data : begin of t_makt occurs 0,
matnr like mara-matnr,
maktx like makt-maktx,
```

```
spras like makt-spras,
end of t_makt.

* INTERNAL TABLE WHICH ACTUALLY MERGES ALL THE OTHER INTERNAL TABLES.
data: begin of itab1 occurs 0,
matnr like mara-matnr,
meins like mara-meins,
maktx like makt-maktx,
spras like makt-spras,
werks like marc-werks,
minbe like marc-minbe,
end of itab1.


* THE FOLLOWING DECLARATION IS USED FOR DEFINING THE FIELDCAT
* AND THE LAYOUT FOR THE ALV.
* HERE AS slis_t_fieldcat_alv IS A INTERNAL TABLE WITHOUT A HEADER LINE
* WE EXPLICITELY DEFINE AN INTERNAL TABLE OF THE SAME STRUCTURE AS THAT
* OF slis_t_fieldcat_alv BUT WITH A HEADER LINE IN THE DEFINITION.
* THIS IS DONE TO MAKE THE CODE SIMPLER.
* OTHERWISE WE MAY HAVE TO DEFINE THE STRUCTURE AS IN THE NORMAL SAP
* PROGRAMS.
* IN THE FIELDCATALOG TABLE WE ACTUALLY PASS THE FIELDS FROM ONE OR
* MORE TABLES AND CREATE A STRUCTURE
* IN THE LAYOUT STRUCTURE WE BASICALLY DEFINE THE FORMATTING OPTIONS
* LIKE DISPLAY IN THE ZEBRA PATTERN ,THE HOTSPOT OPTIONS ETC.
data: fieldcatalog type slis_t_fieldcat_alv with header line,
fieldlayout type slis_layout_alv.


* DECLARING THE EVENTTABLE INTERNL TABLE FOR USING EVENTS LIKE
* TOP-OF-PAGE ETC.
data : eventstab type slis_t_event with header line.

* DECLARING AN INTERNAL TABLE TO HOLD THE DATA FOR THE TOP-OF-PAGE
data : heading type slis_t_listheader with header line.
data : heading1 type slis_t_listheader with header line.
data : heading2 type slis_t_listheader with header line.
```

```
data : heading3 type slis_t_listheader with header line.
data : heading4 type slis_t_listheader with header line.
data : heading5 type slis_t_listheader with header line.
data : heading6 type slis_t_listheader with header line.
data : heading7 type slis_t_listheader with header line.
data : heading8 type slis_t_listheader with header line.


* STRUCTURE TO PASS THE COLOR ATTRIBUTES FOR DISPLAY.
data : colorstruct type slis_coltypes.
***********************************************************************
* INITIALIZATION. *
***********************************************************************
initialization.
syrepid = sy-repid.
sypagno = sy-pagno.
clear fieldcatalog.


***********************************************************************
* START-OF-SELECTION. *
***********************************************************************
start-of-selection.
* SUBROUTINE TO POPULATE THE COLORSTRUCT
perform fill_colorstruct using colorstruct.
* SUBROUTINE TO POPULATE THE FIELDS OF THE FIELD CATALOGUE
perform populate_fieldcatalog.
* SUBROUTINE TO SELECT DATA FROM VARIOUS TABLES AND POPULATE IT IN THE
* INTERNAL TABLE.
perform selectdata_and_sort.
* SUBROUTINE TO POPULATE THE LAYOUT STRUCTURE.
perform populate_layout using fieldlayout.
* SUBROUTINE TO CALL THE FUNCTION MERGE TO ENSURE PROPER DISPLAY.
perform merge_fieldcatalog.
* SUBROUTINE TO POPULATE THE EVENTSTAB.
perform fill_eventstab tables eventstab.
* SUBROUTINE TO POPULATE THE HEADING TABLES.
```

```
perform fill_headingtable tables heading using 'HEADING'.
perform fill_headingtable tables heading1 using 'HEADING1'.
perform fill_headingtable tables heading2 using 'HEADING2'.
perform fill_headingtable tables heading3 using 'HEADING3'.
perform fill_headingtable tables heading4 using 'HEADING4'.
perform fill_headingtable tables heading5 using 'HEADING5'.
perform fill_headingtable tables heading6 using 'HEADING6'.
perform fill_headingtable tables heading7 using 'HEADING7'.
perform fill_headingtable tables heading8 using 'HEADING8'.

* SUBROUTINE TO DISPLAY THE LIST.
perform display_alv_list.



***********************************************************************
* FORMS
***********************************************************************
* IN THIS SUBROUTINE WE POPULATE THE FIELDCATALOG TABLE WITH THE NAMES
* OF THE TABLE,FIELDNAME,WHETHER IT IS KEY FIELD OR NOT,HEADING AND
* COLUMN JUSTIFICATION.
form populate_fieldcatalog.
perform fill_fields_of_fieldcatalog tables fieldcatalog
using 'ITAB1' 'MATNR' 'X' .
perform fill_fields_of_fieldcatalog tables fieldcatalog
using 'ITAB1' 'MEINS' ' '.
perform fill_fields_of_fieldcatalog tables fieldcatalog
using 'ITAB1' 'MAKTX' ' ' .
perform fill_fields_of_fieldcatalog tables fieldcatalog
using 'ITAB1' 'MTART' ' ' .
perform fill_fields_of_fieldcatalog tables fieldcatalog
using 'ITAB1' 'MATKL' ' ' .
perform fill_fields_of_fieldcatalog tables fieldcatalog
using 'ITAB1' 'SPRAS' ' ' .
perform fill_fields_of_fieldcatalog tables fieldcatalog
using 'ITAB1' 'WERKS' ' ' .
perform fill_fields_of_fieldcatalog tables fieldcatalog
```

```
using 'ITAB1' 'MINBE' ' ' .
endform. " POPULATE_FIELDCATALOG


*---------------------------------------------------------------------*
* FORM FILL_FIELDS_OF_FIELDCATALOG *
*---------------------------------------------------------------------*
* ........ *
*---------------------------------------------------------------------*
* --> FIELDCATALOG *
* --> P_TABNAME *
* --> P_FIELDNAME *
* --> P_KEY *
* --> P_KEY *
*---------------------------------------------------------------------*
form fill_fields_of_fieldcatalog tables fieldcatalog
structure fieldcatalog
using p_tabname
p_fieldname
p_key.
* p_no_out.
fieldcatalog-tabname = p_tabname.
fieldcatalog-fieldname = p_fieldname.
fieldcatalog-key = p_key.
fieldcatalog-emphasize = '1234'.
*fieldcatalog-no_out = p_no_out.
append fieldcatalog.
endform. " FILL_FIELDSOFFIELDCATALOG


*---------------------------------------------------------------------*
* FORM POPULATE_LAYOUT *
*---------------------------------------------------------------------*
* ........ *
*---------------------------------------------------------------------*
* --> FIELDLAYOUT *
*---------------------------------------------------------------------*
```

```
form populate_layout using fieldlayout type slis_layout_alv.
fieldlayout-f2code = '&ETA' .
fieldlayout-zebra = 'X'.
* FOR THE WINDOW TITLE.
fieldlayout-window_titlebar = 'ALV with Events'.

fieldlayout-colwidth_optimize = 'X'.
fieldlayout-no_vline = ' '.
*fieldlayout-no_input = 'X'.
fieldlayout-confirmation_prompt = ''.
fieldlayout-key_hotspot = 'X'.
* This removes the column headings if the flag is set to 'X'
fieldlayout-no_colhead = ' '.
*fieldlayout-hotspot_fieldname = 'MAKTX'.
fieldlayout-detail_popup = 'X'.
* fieldlayout-coltab_fieldname = 'X'.
endform. " POPULATE_LAYOUT

*---------------------------------------------------------------------*
* FORM SELECTDATA_AND_SORT *
*---------------------------------------------------------------------*
* ........ *
*---------------------------------------------------------------------*
form selectdata_and_sort.
select matnr meins mtart matkl from mara
into corresponding fields of t_mara
up to 500 rows .

select matnr maktx spras from makt
into corresponding fields of t_makt
where matnr = t_mara-matnr and
spras = sy-langu.

select matnr werks minbe from marc
into corresponding fields of t_marc
where matnr = t_mara-matnr.
append t_marc.
endselect.
```

```
append t_makt.
endselect.
append t_mara.
endselect.
perform populate_itab1.
sort itab1 by matnr.
endform. " SELECTDATA_AND_SORT

*---------------------------------------------------------------------*
* FORM MERGE_FIELDCATALOG *
*---------------------------------------------------------------------*
* ........ *
*---------------------------------------------------------------------*
form merge_fieldcatalog.
call function 'REUSE_ALV_FIELDCATALOG_MERGE'
exporting
i_program_name = syrepid
i_internal_tabname = 'ITAB1'
* i_structure_name = 'COLORSTRUCT'
* I_CLIENT_NEVER_DISPLAY = 'X'
i_inclname = syrepid
changing
ct_fieldcat = fieldcatalog[]
exceptions
inconsistent_interface = 1
program_error = 2
others = 3.
endform. " MERGE_FIELDCATALOG


* IN THIS FUNCTION THE MINIMUM PARAMETERS THAT WE NEED TO PASS IS AS
* FOLLOWS:-
* i_callback_program --> CALLING PROGRAM NAME
* i_structure_name --> STRUCTURE NAME.
* is_layout --> LAYOUT NAME.
* it_fieldcat ---> BODY OF THE FIELD CATALOGUE INTERNAL TABLE
form display_alv_list.
```

```
call function 'REUSE_ALV_LIST_DISPLAY'
exporting
* I_INTERFACE_CHECK = ' '
i_callback_program = syrepid
* I_CALLBACK_PF_STATUS_SET = ' '
* I_CALLBACK_USER_COMMAND = ' '
i_structure_name = 'ITAB1'
is_layout = fieldlayout
it_fieldcat = fieldcatalog[]
* IT_EXCLUDING =
* IT_SPECIAL_GROUPS =
* IT_SORT =
* IT_FILTER =
* IS_SEL_HIDE =
* I_DEFAULT = 'X'
* THE FOLLOWING PARAMETER IS SET AS 'A' INORDER TO DISPLAY THE STANDARD
* TOOL BAR
i_save = 'A'
* IS_VARIANT = ' '
it_events = eventstab[]
* IT_EVENT_EXIT =
* IS_PRINT =
* I_SCREEN_START_COLUMN = 0
* I_SCREEN_START_LINE = 0
* I_SCREEN_END_COLUMN = 0
* I_SCREEN_END_LINE = 0
* IMPORTING
* E_EXIT_CAUSED_BY_CALLER =
* ES_EXIT_CAUSED_BY_USER =
tables
t_outtab = itab1
exceptions
program_error = 1
others = 2.
endform. " DISPLAY_ALV_LIST
*&---------------------------------------------------------------------*
```

```
*& Form POPULATE_ITAB1
*&---------------------------------------------------------------------*
* text
*---------------------------------------------------------------------*
* --> p1 text
* <-- p2 text
*---------------------------------------------------------------------*
form populate_itab1.
loop at t_mara.
loop at t_makt where matnr = t_mara-matnr.
loop at t_marc where matnr = t_mara-matnr.
move-corresponding t_mara to itab1.
move-corresponding t_makt to itab1.
move-corresponding t_marc to itab1.
append itab1.
endloop.
endloop.
endloop.
endform. " POPULATE_ITAB1
*&---------------------------------------------------------------------*
*& Form FILL_EVENTSTAB
*&---------------------------------------------------------------------*
* text
*---------------------------------------------------------------------*
* -->P_EVENTSTAB text *
*---------------------------------------------------------------------*
form fill_eventstab tables p_eventstab structure eventstab.
* WHEN THE FOLLOWING FUNCTION IS CALLED THE SYSTEM POPULATES THE
* INTERNAL TABLE EVENTSTAB WITH A LIST OF EVENTS NAME.
* AS SHOWN BELOW WHEN USING I_LIST_TYPE = 0 THE FUNCTION RETURNS 14
* EVENTS NAME.
call function 'REUSE_ALV_EVENTS_GET'
exporting
i_list_type = 0
importing
et_events = p_eventstab[]
```

```
exceptions
list_type_wrong = 1
others = 2.

* BY CALLING THE ABOVE FUNCTION WE FIRST POPULATE THE EVENTSTAB WITH
* THE PREDEFINED EVENTS AND THEN WE MOVE THE FORM NAME AS SHOWN BELOW.
* WE ASSIGN A FORM NAME TO THE EVENT AS REQUIRED BY THE USER.
* FORM NAME CAN BE ANYTHING.THE PERFORM STATEMENT FOR THIS FORM
* IS DYNAMICALY CALLED.

read table p_eventstab with key name = slis_ev_top_of_page.
if sy-subrc = 0 .
move 'TOP_OF_PAGE' to p_eventstab-form.
append p_eventstab.
endif.

read table p_eventstab with key name = slis_ev_top_of_coverpage.
if sy-subrc = 0 .
move 'TOP_OF_COVERPAGE' to p_eventstab-form.
append p_eventstab.
endif.

read table p_eventstab with key name = slis_ev_end_of_coverpage .
if sy-subrc = 0 .
move 'END_OF_COVERPAGE' to p_eventstab-form.
append p_eventstab.
endif.

read table p_eventstab with key name = slis_ev_foreign_top_of_page.
if sy-subrc = 0 .
move 'FOREIGN_TOP_OF_PAGE' to p_eventstab-form.
append p_eventstab.
endif.

read table p_eventstab with key name = slis_ev_foreign_end_of_page.
if sy-subrc = 0 .
move 'FOREIGN_END_OF_PAGE' to p_eventstab-form.
append p_eventstab.
endif.
```

```
read table p_eventstab with key name = slis_ev_list_modify.
if sy-subrc = 0 .
move 'LIST_MODIFY' to p_eventstab-form.
append p_eventstab.
endif.

read table p_eventstab with key name = slis_ev_top_of_list.
if sy-subrc = 0 .
move 'TOP_OF_LIST' to p_eventstab-form.
append p_eventstab.
endif.

read table p_eventstab with key name = slis_ev_end_of_page.
if sy-subrc = 0 .
move 'END_OF_PAGE' to p_eventstab-form.
append p_eventstab.
endif.

read table p_eventstab with key name = slis_ev_end_of_list .
if sy-subrc = 0 .
move 'END_OF_LIST' to p_eventstab-form.
append p_eventstab.
endif.
endform. " FILL_EVENTSTAB
*&---------------------------------------------------------------------*
*& Form FILL_HEADINGTABLE
*&---------------------------------------------------------------------*
* text
*----------------------------------------------------------------------*
* -->P_HEADING text *
*----------------------------------------------------------------------*
form fill_headingtable tables p_heading structure heading
using tablename.
case tablename.
when 'HEADING'.
p_heading-typ = 'H'.
concatenate
' REPORT NAME:-' syrepid
```

```
' ABB Industry Pte Ltd' into p_heading-info.
append p_heading.
write sy-datum using edit mask '__/__/____' to sydatum.

concatenate
' DATE:-' sydatum ' USER: ' sy-uname 'PAGE NO:' sypagno
into p_heading-info.
append p_heading.
when 'HEADING1'.
p_heading-typ = 'H'.
p_heading-info = 'TOP-OF-COVER-PAGE'.
append p_heading.
when 'HEADING2'.
p_heading-typ = 'H'.
p_heading-info = 'END-OF-COVER-PAGE'.
append p_heading.
when 'HEADING3'.
p_heading-typ = 'H'.
p_heading-info = 'FOREIGN-TOP-OF-PAGE'.
append p_heading.
when 'HEADING4'.
p_heading-typ = 'H'.
p_heading-info = 'FOREIGN-END-OF-PAGE'.
append p_heading.
* WHEN 'HEADING5'.
* P_HEADING-TYP = 'H'.
* P_HEADING-INFO = 'LIST-MODIFY'.
* APPEND P_HEADING.
when 'HEADING6'.
p_heading-typ = 'H'.
p_heading-info = 'END-OF-PAGE'.
append p_heading.
when 'HEADING7'.
p_heading-typ = 'H'.
p_heading-info = 'END-OF-LIST'.
append p_heading.
when 'HEADING8'.
```

```
p_heading-typ = 'H'.
p_heading-info = 'TOP-OF-LIST'.
append p_heading.
endcase.
endform. " FILL_HEADINGTABLE

*---------------------------------------------------------------------*
* FORM TOP_OF_PAGE *
*---------------------------------------------------------------------*
* ........ *
*---------------------------------------------------------------------*
form top_of_page.
call function 'REUSE_ALV_COMMENTARY_WRITE'
exporting
it_list_commentary = heading[]
exceptions
others = 1.
endform.
*&---------------------------------------------------------------------*
*& Form FILL_COLORSTRUCT
*&---------------------------------------------------------------------*
* text
*---------------------------------------------------------------------*
* -->P_COLORSTRUCT text *
*---------------------------------------------------------------------*
form fill_colorstruct using p_colorstruct type slis_coltypes .
p_colorstruct-heacolfir-col = 6.
p_colorstruct-heacolfir-int = 1.
p_colorstruct-heacolfir-inv = 1.
endform. " FILL_COLORSTRUCT

*---------------------------------------------------------------------*
* FORM TOP_OF_COVERPAGE *
*---------------------------------------------------------------------*
* ........ *
*---------------------------------------------------------------------*
form top_of_coverpage.
```

```
call function 'REUSE_ALV_COMMENTARY_WRITE'
exporting
it_list_commentary = heading1[]
exceptions
others = 1.
endform.

*---------------------------------------------------------------------*
* FORM END_OF_COVERPAGE *
*---------------------------------------------------------------------*
* ........ *
*---------------------------------------------------------------------*
form end_of_coverpage.
call function 'REUSE_ALV_COMMENTARY_WRITE'
exporting
it_list_commentary = heading2[]
exceptions
others = 1.
endform.

*---------------------------------------------------------------------*
* FORM FOREIGN_TOP_OF_PAGE *
*---------------------------------------------------------------------*
* ........ *
*---------------------------------------------------------------------*
form foreign_top_of_page.
call function 'REUSE_ALV_COMMENTARY_WRITE'
exporting
it_list_commentary = heading3[]
exceptions
others = 1.

endform.

*---------------------------------------------------------------------*
* FORM FOREIGN_END_OF_PAGE *
*---------------------------------------------------------------------*
* ........ *
```

```
*---------------------------------------------------------------------*
form foreign_end_of_page.
call function 'REUSE_ALV_COMMENTARY_WRITE'
exporting
it_list_commentary = heading4[]
exceptions
others = 1.
endform.

*---------------------------------------------------------------------*
* FORM LIST_MODIFY *
*---------------------------------------------------------------------*
* ........ *
*---------------------------------------------------------------------*
*FORM LIST_MODIFY.
* CALL FUNCTION 'REUSE_ALV_COMMENTARY_WRITE'
* EXPORTING
* IT_LIST_COMMENTARY = HEADING5[]
* EXCEPTIONS
* OTHERS = 1.
*ENDFORM.

*---------------------------------------------------------------------*
* FORM END_OF_PAGE *
*---------------------------------------------------------------------*
* ........ *
*---------------------------------------------------------------------*
form end_of_page.
call function 'REUSE_ALV_COMMENTARY_WRITE'
exporting
it_list_commentary = heading6[]
exceptions
others = 1.
endform.

*---------------------------------------------------------------------*
* FORM END_OF_LIST *
*---------------------------------------------------------------------*
```

```
* ........ *
*---------------------------------------------------------------------*
form end_of_list.
call function 'REUSE_ALV_COMMENTARY_WRITE'
exporting
it_list_commentary = heading7[]
exceptions
others = 1.
endform.

*---------------------------------------------------------------------*
* FORM TOP_OF_LIST *
*---------------------------------------------------------------------*
* ........ *
*---------------------------------------------------------------------*
form top_of_list.
call function 'REUSE_ALV_COMMENTARY_WRITE'
exporting
it_list_commentary = heading8[]
exceptions
others = 1.
endform.

*--- End of Program
```

How to Refresh ALV List/Grid once it is displayed?

This mean to say that if you have a 'refresh' push button in your gui status, every time you press the button, the list should get refreshed.

In ALV, to refresh the table you have to call the method "refresh_table_display".

It has the syntax very similar to creating the table.

It has two parameters. In the first one, you can mention if you want to refresh only the data (the icons are not refreshed)

or

if you want to refresh only the icons around the grid (the data is not refreshed - this option is mostly not used in day to day applications).

the synatx is :-

call method grid (name of grid)->refresh_table_display

exporting

IS_STABLE = <STRUCT OF TYPE LVC_S_STBL> (THIS IS FOR DATA REFRESHING)

I_SOFT_REFRESH = <VARIABLE OF CHAR 01> (THIS IS FOR ICON REFRESHING).

How can I insert my company logo in the standard report?

It is not possible to print logo in the ordinary report, but it can done through ALV.

Write the code in Top-of-page event in ALV.

The following is the code for inserting the logo in ALV.

```
FORM TOP_OF_PAGE.

  CALL FUNCTION 'REUSE_ALV_COMMENTARY_WRITE'
      EXPORTING
        I_LOGO          = 'ENJOYSAP_LOGO'
         IT_LIST_COMMENTARY = GT_LIST_TOP_OF_PAGE.

ENDFORM.
```

Upload Logo for REUSE_ALV_COMMENTARY_WRITE

For those who wish to upload and use a picture in your ALV abap reports.

Steps for uploading Logo :-:

1. Goto the transaction **OAER**
2. Enter the class name as 'PICTURES'
3. Enter the class type as 'OT'
4. Enter the object key as the name of the logo you wish to give
5. Execute
6. Then in the new screen select Standard doc. types in bottom window

Click on the Screen icon

Now, it will ask for the file path where you have to upload the logo

7. Now you can use this logo in REUSE_ALV_COMMENTARY_WRITE

or

Import Logo and Background Picture for Reporting

In this step, you can import a customer-specific logo and a background picture into the R/3 System. These will be displayed in the header area of reports in HR Funds and Position Management.

From the SPRO:

HR Funds and Position Management --> Dialog Control --> Customize Reporting Interface --> Import Logo and Background Picture for Reporting.

Activities

1. Enter the Name of your logo/background picture as an object key in the initial screen.
2. Make sure that the class name is PICTURES, and the class type is OT.
3. Choose Execute.
4. Double-click the document type Picture on the Create tab page. A dialog box will appear in which you can enter the path in which the logo/background picture can be found.
5. Enter the path and choose Open. The logo will be uploaded into the current R/3 System. If the logo/background picture is to be transported into other systems as well, choose Transport.
6. Return to the initial screen and repeat the procedure after having entered the Name of your background picture as an object key.

Please note that the logo/background picture can only be displayed in ALV-based reports with an HTML header. Manually programmed reports such as business distribution plans are not based on the ALV.

If you have selected several initial objects, ALV-based reports in HR Funds and Position Management will automatically use a hiearchical-sequential display. A logo is not displayed here either. Note also that the logo cannot be printed (see print preview in program).

Make sure that the logo does not exceed a height of 100 pixels because it would mean that the header of the report will be scrollable.

What is ALV Programming?

Content Author: Nimesh Jhanwar

What is ALV programming in ABAP? When is this grid used in ABAP?

ALV is Application List viewer.

Sap provides a set of ALV (ABAP LIST VIEWER) function modules which can be put into use to embellish the output of a report. This set of ALV functions is used to enhance the readability and functionality of any report output. Cases arise in sap when the output of a report contains columns extending more than 255 characters in length.

In such cases, this set of ALV functions can help choose selected columns and arrange the different columns f rom a report output and also save different variants for report display. This is a very efficient tool for dynamical ly sorting and arranging the columns from a report output.

The report output can contain up to 90 columns in the display with the wide array of display options.

The commonly used ALV functions used for this purpose are;

1. REUSE_ALV_VARIANT_DEFAULT_GET
2. REUSE_ALV_VARIANT_F4
3. REUSE_ALV_VARIANT_EXISTENCE
4. REUSE_ALV_EVENTS_GET
5. REUSE_ALV_COMMENTARY_WRITE
6. REUSE_ALV_FIELDCATALOG_MERGE
7. REUSE_ALV_LIST_DISPLAY
8. REUSE_ALV_GRID_DISPLAY
9. REUSE_ALV_POPUP_TO_SELECT

Purpose of the above Functions are differ not all the functions are required in all the ALV Report.

But either no.7 or No.8 is there in the Program.

How you call this function in your report?

After completion of all the data fetching from the database and append this data into an Internal Table. say I_I TAB.

Then use follwing function module.

```
CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
    EXPORTING
      I_CALLBACK_PROGRAM      = 'Prog.name'
```

```
        I_STRUCTURE_NAME        = 'I_ITAB'
        I_DEFAULT               = 'X'
        I_SAVE                  = 'A'
      TABLES
        T_OUTTAB                = I_ITAB.
  IF SY-SUBRC <> 0.
    WRITE: 'SY-SUBRC: ', SY-SUBRC .
  ENDIF.
ENDFORM.                  " GET_FINAL_DATA
```

查看 PO 的 Text

```
DATA: BEGIN OF itab OCCURS 0.
      INCLUDE STRUCTURE thead.
DATA: END OF itab.

DATA: BEGIN OF lines OCCURS 0.
      INCLUDE STRUCTURE tline.
DATA: END OF lines.


START-OF-SELECTION.

  SELECT * FROM stxl
    INTO  CORRESPONDING FIELDS OF TABLE itab
    WHERE  tdobject = 'EKKO'
    AND    tdname IN ('4500004889', '4500004890').

  LOOP AT itab.
    CALL FUNCTION 'READ_TEXT'
       EXPORTING
          id       = itab-tdid                  "'F01'
          language  = itab-tdspras  "'E'
          name      = itab-tdname  "'4500004890'
          object    = itab-tdobject  "'EKKO'
*       IMPORTING
*          header    =
       TABLES
```

```
          lines     = lines
        EXCEPTIONS
          not_found = 1.

    LOOP AT lines.
      WRITE:/ lines-tdline.
      SKIP.
    ENDLOOP.

  ENDLOOP.
```

获取 SAP 表字段说明

```
REPORT  ZGETTABLEFIELD                                  .
* Data declaration
TYPE-POOLS: SLIS.

* Global structure of list
TYPES:      BEGIN OF UD_STRUCT,
            POSITION   LIKE DD03L-POSITION,
            TABNAME    LIKE DD03L-TABNAME,
            FIELDNAME  LIKE DD03L-FIELDNAME,
            DATATYPE   LIKE DD03L-DATATYPE,
            DDLENG     LIKE DD03L-LENG,
            DECIMALS   LIKE DD03L-DECIMALS,
            DDTEXT     LIKE DD03T-DDTEXT,
            EDDTEXT    LIKE DD03T-DDTEXT,
            DDDTEXT    LIKE DD03T-DDTEXT,
          END OF UD_STRUCT.

TABLES: DD03L.

DATA:   GT_FIELDCAT TYPE SLIS_T_FIELDCAT_ALV.
DATA:   GT_OUTTAB  TYPE UD_STRUCT OCCURS 0 WITH HEADER LINE.
DATA:   G_REPID LIKE SY-REPID.
data : begin of exclude occurs 5,
       fcode like sy-ucomm,
      end   of exclude.
DATA  p_ucomm LIKE sy-ucomm.
```

```
PARAMETER P_TNAME LIKE DD02l-TABNAME  DEFAULT 'VBAK'.

*=====================================================================*
* Initialization fieldcatalog
*=====================================================================*
INITIALIZATION.
  G_REPID = SY-REPID.
  PERFORM FIELDCAT_INIT USING GT_FIELDCAT[].


*=====================================================================*
* Ereignis : AT SELECTION-SCREEN OUTPUT (PBO-Zeitpunkt)             *
*=====================================================================*
at selection-screen output.
  data exclude like rsexfcode occurs 0 with header line.
  if sy-dynnr = 1000.
    call function 'RS_SET_SELSCREEN_STATUS'
      EXPORTING
        p_status  = 'ZGETTBFD'
      TABLES
        p_exclude = exclude
      EXCEPTIONS
        others    = 1.
  endif.
  p_ucomm = SPACE.

*=====================================================================*
* Ereignis : AT SELECTION-SCREEN (PAI-Zeitpunkt)                    *
*          letztes PAI-Ereignis                                     *
*=====================================================================*
at selection-screen.
  p_ucomm = sy-ucomm.
  CASE p_ucomm.
    WHEN 'STBL'.
      SET PARAMETER ID 'DTB' FIELD P_TNAME.
*     PERFORM AUTHORITY_CHECK USING 'SE11' .
      CALL TRANSACTION 'SE11' AND SKIP FIRST SCREEN.
  ENDCASE.
```

```
*=====================================================================*
* Data selection
*=====================================================================*
START-OF-SELECTION.
  PERFORM SELECT_DATA TABLES GT_OUTTAB.
  perform function_exclude tables exclude.

*=====================================================================*
* Display list
*=====================================================================*
END-OF-SELECTION.
   CALL FUNCTION 'REUSE_ALV_LIST_DISPLAY'
      EXPORTING
         I_CALLBACK_PROGRAM = G_REPID
         IT_FIELDCAT        = GT_FIELDCAT[]
      TABLES
         T_OUTTAB           = GT_OUTTAB.
*---------------------------------------------------------------------
*    Forms
*---------------------------------------------------------------------

* Initialization fieldcatalog

FORM FIELDCAT_INIT
     USING RT_FIELDCAT TYPE SLIS_T_FIELDCAT_ALV.
  DATA: LS_FIELDCAT TYPE SLIS_FIELDCAT_ALV.
  DATA: POS TYPE I VALUE 1.

  clear ls_fieldcat.
  LS_FIELDCAT-COL_POS       = POS.
  LS_FIELDCAT-FIELDNAME     = 'POSITION'.
  ls_fieldcat-ref_fieldname = 'POSITION'.
  LS_FIELDCAT-REF_TABNAME   = 'DD03L'.
  LS_FIELDCAT-KEY           = 'X'.
    APPEND LS_FIELDCAT TO  RT_FIELDCAT.
  clear ls_fieldcat.
  POS = POS + 1.
  LS_FIELDCAT-COL_POS       = POS.
```

```
  LS_FIELDCAT-FIELDNAME     = 'TABNAME'.
  ls_fieldcat-ref_fieldname = 'TABNAME'.
  LS_FIELDCAT-REF_TABNAME   = 'DD03T'.
*  LS_FIELDCAT-KEY          = 'X'.
    APPEND LS_FIELDCAT TO  RT_FIELDCAT.
  clear ls_fieldcat.
  POS = POS + 1.
  LS_FIELDCAT-COL_POS       =  POS.
  LS_FIELDCAT-FIELDNAME     = 'FIELDNAME'.
  ls_fieldcat-ref_fieldname = 'FIELDNAME'.
  LS_FIELDCAT-REF_TABNAME   = 'DD03T'.
    APPEND LS_FIELDCAT TO RT_FIELDCAT.
  clear ls_fieldcat.
  POS = POS + 1.
  LS_FIELDCAT-COL_POS       =  POS.
  LS_FIELDCAT-FIELDNAME     = 'DATATYPE'.
  ls_fieldcat-ref_fieldname = 'DATATYPE'.
  LS_FIELDCAT-REF_TABNAME   =  'DD03T'.
    APPEND LS_FIELDCAT TO RT_FIELDCAT.
  clear ls_fieldcat.
  POS = POS + 1.
  LS_FIELDCAT-COL_POS       =  POS.
  LS_FIELDCAT-FIELDNAME     = 'DDLENG'.
  ls_fieldcat-ref_fieldname = 'LENG'.
  LS_FIELDCAT-REF_TABNAME   =  'DD03L'.
    APPEND LS_FIELDCAT TO RT_FIELDCAT.
  clear ls_fieldcat.
  POS = POS + 1.
  LS_FIELDCAT-COL_POS       =  POS.
  LS_FIELDCAT-FIELDNAME     = 'DECIMALS'.
  ls_fieldcat-ref_fieldname = 'DECIMALS'.
  LS_FIELDCAT-REF_TABNAME   =  'DD03L'.
    APPEND LS_FIELDCAT TO RT_FIELDCAT.
  clear ls_fieldcat.
  POS = POS + 1.
  LS_FIELDCAT-COL_POS       =  POS.
```

```
  LS_FIELDCAT-FIELDNAME     = 'DDTEXT'.
  ls_fieldcat-ref_fieldname = 'DDTEXT'.
  LS_FIELDCAT-REF_TABNAME   =  'DD03T'.
*  LS_FIELDCAT-NO_OUT       = 'X'.
    APPEND LS_FIELDCAT TO RT_FIELDCAT.
ENDFORM.   "fieldcat_init


* Data selection
FORM SELECT_DATA
  TABLES RT_OUTTAB LIKE GT_OUTTAB[].
  SELECT DD03L~POSITION   DD03L~TABNAME
       DD03L~FIELDNAME  DD03L~DATATYPE
       DD03L~LENG       DD03L~DECIMALS
       DD03T~DDTEXT
       DD04T~DDTEXT     DD01T~DDTEXT
       INTO (RT_OUTTAB-POSITION, RT_OUTTAB-TABNAME,
           RT_OUTTAB-FIELDNAME, RT_OUTTAB-DATATYPE,
           RT_OUTTAB-DDLENG, RT_OUTTAB-DECIMALS,
           RT_OUTTAB-DDTEXT,
           RT_OUTTAB-EDDTEXT,RT_OUTTAB-DDDTEXT)
       FROM DD03L LEFT JOIN DD03T
       ON DD03L~TABNAME = DD03T~TABNAME
         AND DD03L~FIELDNAME = DD03T~FIELDNAME
         AND DD03T~DDLANGUAGE = SY-LANGU
       LEFT JOIN DD04T
        ON DD03L~ROLLNAME = DD04T~ROLLNAME
         AND DD04T~DDLANGUAGE = SY-LANGU
       LEFT JOIN DD01T
        ON DD03L~DOMNAME = DD01T~DOMNAME
        AND DD01T~DDLANGUAGE = SY-LANGU
       WHERE DD03L~TABNAME =  P_TNAME
       ORDER BY DD03L~POSITION.
    IF RT_OUTTAB-DDTEXT = SPACE.
      IF RT_OUTTAB-EDDTEXT = SPACE.
        RT_OUTTAB-DDTEXT = RT_OUTTAB-DDDTEXT.
      ELSE.
```

```
        RT_OUTTAB-DDTEXT = RT_OUTTAB-EDDTEXT.
      ENDIF.
    ENDIF.
    APPEND RT_OUTTAB.
  ENDSELECT.

ENDFORM.

*&---------------------------------------------------------------------*
*&      Form  FUNCTION_EXCLUDE
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
*      -->P_EXCLUDE  text                                              *
*----------------------------------------------------------------------*
form function_exclude tables  p_exclude structure exclude.
*  data : rcode like sy-subrc.
* clear p_exclude.
* refresh p_exclude.
* exclude-fcode = 'EERW'.
* append exclude.
* exclude-fcode = 'EXIT'.
* append exclude.
endform.                          " FUNCTION_EXCLUDE
```

[推荐]获取ＳＡＰ系统用户出口列表

```
*&---------------------------------------------------------------------*
*& Report  Y_FIND_USEREXIT                                            *
*&                                                                     *
*&---------------------------------------------------------------------*
*&                                                                     *
*&                                                                     *
*&---------------------------------------------------------------------*

REPORT  Y_FIND_USEREXIT                    .
tables : tstc, tadir, modsapt, modact, trdir, tfdir, enlfdir.
        tables : tstct.
data : jtab like tadir occurs 0 with header line.
```

```
data : field1(30).
data : v_devclass like tadir-devclass.
parameters : p_tcode like tstc-tcode obligatory.

select single * from tstc where tcode eq p_tcode.
if sy-subrc eq 0.
  select single * from tadir where pgmid = 'R3TR'
              and object = 'PROG'
              and obj_name = tstc-pgmna.
  move : tadir-devclass to v_devclass.
    if sy-subrc ne 0.
      select single * from trdir where name = tstc-pgmna.
      if trdir-subc eq 'F'.
        select single * from tfdir where pname = tstc-pgmna.
        select single * from enlfdir where funcname =
        tfdir-funcname.
        select single * from tadir where pgmid = 'R3TR'
                    and object = 'FUGR'
                    and obj_name eq enlfdir-area.

        move : tadir-devclass to v_devclass.
       endif.
     endif.
     select * from tadir into table jtab
               where pgmid = 'R3TR'
                 and object = 'SMOD'
                 and devclass = v_devclass.
      select single * from tstct where sprsl eq sy-langu and
                          tcode eq p_tcode.
      format color col_positive intensified off.
      write:/(19) 'Transaction Code - ',
          20(20) p_tcode,
          45(50) tstct-ttext.
              skip.
      if not jtab[] is initial.
        write:/(95) sy-uline.
        format color col_heading intensified on.
```

```
        write:/1 sy-vline,
              2 'Exit Name',
             21 sy-vline ,
             22 'Description',
             95 sy-vline.
        write:/(95) sy-uline.
        loop at jtab.
          select single * from modsapt
                 where sprsl = sy-langu and
                       name = jtab-obj_name.
               format color col_normal intensified off.
               write:/1 sy-vline,
                     2 jtab-obj_name hotspot on,
                    21 sy-vline ,
                    22 modsapt-modtext,
                    95 sy-vline.
        endloop.
        write:/(95) sy-uline.
        describe table jtab.
        skip.
        format color col_total intensified on.
        write:/ 'No of Exits:' , sy-tfill.
      else.
        format color col_negative intensified on.
        write:/(95) 'No User Exit exists'.
      endif.
    else.
      format color col_negative intensified on.
      write:/(95) 'Transaction Code Does Not Exist'.
    endif.

at line-selection.
  get cursor field field1.
  check field1(4) eq 'JTAB'.
  set parameter id 'MON' field sy-lisel+1(10).
  call transaction 'SMOD' and skip first  screen.
```

如何限制 SELECT－OPTIONS 的选择屏幕的 OPTION

```
REPORT Z_CONECT_A.
* Include type pool
 SSCRTYPE-POOLS sscr.
TABLES : marc.
*定义选择屏幕
select-options :  s_matnr for marc-matnr,
s_werks for marc-werks.
* Define the object to be passed to the RESTRICTION
parameterDATA restrict TYPE sscr_restrict.
* Auxiliary objects for filling RESTRICT
DATA : optlist TYPE sscr_opt_list,
  ass type sscr_ass.INITIALIZATION.
* 限制MATNR参数只能使用‘EQ’ 和‘BT’.
 optlist-name = 'OBJECTKEY1'.
optlist-options-eq = 'X'.
optlist-options-bt = 'X'.
 APPEND optlist TO restrict-opt_list_tab.
 ass-kind = 'S'.
 ass-name = 'S_MATNR'.
 ass-sg_main = 'I'.
 ass-sg_addy = space.
ass-op_main = 'OBJECTKEY1'.
 APPEND ass TO restrict-ass_tab.
* 限制 WERKS 参数只能使用CP, GE, LT, NE.
 optlist-name = 'OBJECTKEY2'.
 optlist-options-cp = 'X'.
 optlist-options-ge = 'X'.
optlist-options-lt = 'X'.
 optlist-options-ne = 'X'.
 APPEND optlist TO restrict-opt_list_tab.
  ass-kind = 'S'.
ass-name = 'S_WERKS'.
 ass-sg_main = 'I'.
 ass-sg_addy = space.
 ass-op_main = 'OBJECTKEY2'.
```

```
APPEND ass TO restrict-ass_tab.
 CALL FUNCTION 'SELECT_OPTIONS_RESTRICT'
    EXPORTING    restriction                  = restrict
     EXCEPTIONS     TOO_LATE                     = 1
     REPEATED                     = 2
     SELOPT_WITHOUT_OPTIONS       = 3
     SELOPT_WITHOUT_SIGNS         = 4
     INVALID_SIGN                 = 5
     EMPTY_OPTION_LIST            = 6
     INVALID_KIND                 = 7
      REPEATED_KIND_A              = 8
      OTHERS                       = 9            .
     IF sy-subrc <> 0.
       MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
              WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
 ENDIF.
```

一些有用的 ABAP 程序和函数组

BW 设置

名称	类型	描述
SAP_RSADMIN_MAINTAIN	ABAP	维护 RSADMIN 表的 BW 的设置

安全

名称	类型	描述
RSUSR000	ABAP	List who is currently logged in. Equivalent to transaction AL08
RSUSR003	ABAP	List important security parameters (mostly the ones that start login/*, and the snc/* ones), and show the standard users (sap*, ddic, earlywatch and sapcpic) in each client, and whether they have their default (insecure) passwords.

事务管理

Generally these programs are scheduled to run regularly on all SAP systems. These is detailed in OSS note [1 6083].

名称	类型	描述
RSPO1041	ABAP	Deletes old spool requests.
RSPO1043	ABAP	Reorganises spool database.
RSCOLL00	ABAP	Collects and reorganises statistics records from application server, OS and database
RSSNAPDL	ABAP	Deletes old ABAP short dumps
RSBDCREO	ABAP	Reorganises batch input sessions
RSBTCDEL	ABAP	Delete old background job logs

系统管理

名称	类型	描述
RSPFPAR	ABAP	Displays profile parameters and their current values (as set in RZ10 and RZ11)

IDOC 处理

名称	类型	描述
RDBAPP01	ABAP	Reprocess failed or delayed IDOCs. Often IDOCs are not processed due to a blockage on the receiving system. This report can be scheduled to run on a regular basis to pickup any IDOCs that are not yet processed, and process them

性能调整

名称	类型	描述
RSMEMORY	ABAP	Displays and changes SAP memory settings dynamically. Even some that cannot be changed in RZ11. See OSS Note [177226]

[推荐]ABAP 的代码规范标准（某公司使用）

ABAP 代码编写要求

1、单元格式

```
*************************************************************************
* Copyright 2006 C-Bons Wuhan                                           *
* All Rights Reserved                                                   *
```

```
*---------------------------------------------------------------------*
* Program Name : ZXXXXX                                               *
* Project      : C-Bons SAP Implementation Project                    *
* Program Title:                                                      *
* Created by   : DEVXX                                                *
* Created on   : 2006/02/18                                           *
* Version      : 1.0                                                  *
* Function Description:                                               *
*                                                                     *
*---------------------------------------------------------------------*
* Data Table List:                                                    *
*如维护操作的表,数据计算来源等表
*---------------------------------------------------------------------*
* Refrence Table List:                                                *
*如联络处描叙,输入帮助等用到的表
*---------------------------------------------------------------------*
* Modification Log:                                                   *
***********************************************************************
* Date       Programmer     Correction Number     DesingDoc Number    *
* YYYY/MM/DD XXXXXXXX       DEVK9nnnnnn                               *
***********************************************************************
```

注意：版本修改信息中设计文档版本有对应的文档则必须填写。

单元命名规则

对于复杂的程序，可以将其编写为多个 INCLUDE，不同作用的代码放在不同的 INCLUDE 中，各个 INCLUDE 的名称按下表编写

代码的作用 单元名称

Events（TOP－OF－PAGE 等） ZXXXXE01

Subroutines（Form routines） ZXXXXF01

PAI Modules ZXXXXI01

PBO Modules ZXXXXO01

Global Data ZXXXXTOP

Input Help ZXXXXH01

其中：ZXXXX 为程序名。

事件代码必须遵循编写顺序

Initialization

At Selection-Screen OutPut

At Selection-Screen

START-OF-Selection

At Select-Screen on XXXX

At Select-Screen on value request for XXXXX

At Select-Screen on help request for XXXXX

共用函数的格式

```
************************************************************************
* Function Name : XXXXX                                                *
* Created by   : DEVXX                                                 *
* Created on   : 2006/02/18                                            *
* Function Description:                                                *
*                                                                      *
*----------------------------------------------------------------------*
*参数说明                                                              *
*                                                                      *
*----------------------------------------------------------------------*
* Modification Log:                                                    *
************************************************************************
* Date       Programmer   Description                                  *
* YYYY/MM/DD  XXXXXXXX                                                 *
```

子函数的格式

```
************************************************************************
* Form Name : XXXXX                                                    *
* Created by   : DEVXX                                                 *
* Created on   : 2006/02/18                                            *
* Form Description:                                                    *
*                                                                      *
*----------------------------------------------------------------------*
```

```
*参数说明                    *
*                                                    *
*--------------------------------------------------------------------*
* Modification Log:                                  *
**********************************************************************
* Date        Programmer    Description          *
* YYYY/MM/DD   XXXXXXXX                              *
**********************************************************************
```

（注：标为蓝色部分可以省略）

变量命名规则

Type Name:T_

Internal tables (global): gt_

Internal Tables (Local):it

工作区(structure): wa_

全局变量: g_

局部变量: l_

Ranges = r_

Selection screen parameter: p_

select-options : s_

函数参数命名规则：

IMPORTING parameters IM_<parameter name>

EXPORTING parameters EX_<parameter name>

CHANGING parameters CH_<parameter name>

返回参数 RE_<result>

变量定义顺序

```
**********************************************************************
*DATA     DECLARATION                                *
**********************************************************************
*--------------------------------------------------------------------*
* CONSTANTS                          *
*--------------------------------------------------------------------*
CONSTANTS:
     C_TOTAL(8)  TYPE C       VALUE 'WA_TOTAL'.

*--------------------------------------------------------------------*
*TABLES                                *
```

```
*---------------------------------------------------------------------*
TABLES: AFKO,    "Order header data PP orders
        AFPO,    "Order item
        JEST,    "Object status
        RESB,    "Reservation/Dependent Requirements
        AUFK,    "Order master data
        MAKT,    "Material Descriptions
        TJ02T.   "System status texts

*---------------------------------------------------------------------*
* TYPES OR TYPE-POOLS                                                 *
*---------------------------------------------------------------------*
TYPES: BEGIN OF T_AUFK,
        AUFNR   LIKE AUFK-AUFNR,  "Order number
        AUART   LIKE AUFK-AUART,  "Order type
        LOEKZ   LIKE AUFK-LOEKZ,  "Deletion flag
        OBJNR   LIKE AUFK-OBJNR,  "Object number
       END OF T_AUFK.

*---------------------------------------------------------------------*
* INTERNAL TABLES AND WORK AREAS                                      *
*---------------------------------------------------------------------*
DATA: I_AUFK    TYPE T_AUFK    OCCURS 0,
      WA_AUFK   TYPE T_AUFK.

*---------------------------------------------------------------------*
* GLOBAL VARIABLES                                                    *
*---------------------------------------------------------------------*
DATA: G_FLAG.
```

屏幕编号规则

选择屏幕编号范围：1000-1999

录入屏幕编号范围：1－999

子屏幕编号范围：3000－4999

注释规则

1) 任何全局变量要简单解释作用或使用地方

2) 任何函数超过 30 行的函数或 form 必须在关键位置加注释，3) 解释操作意图

4)

消息提示使用规则

5) 消息请使用 ZDEV 的标 6) 准的消息文本；

7) 如果标 8) 准消息文本中没有的请找熊红梅增加，9) 自己不 10) 得维护；

注释里的创建日期和修改日期必须认证填写为实际日期

Status 命名规则为:Menu+屏幕号,共用的为 Menu+Main

[推荐]SAP 的 SCreen 录入的简单模板

1、主程序单元

```
************************************************************************
* Copyright 2003                                                       *
* All Rights Reserved                                                  *
*----------------------------------------------------------------------*
* Program Name : ZFIE0005                                              *
* TYPE         : REPORT                                                *
* Project      :SAP Implementation Project                             *
* Program Title: 管理部门年度预算考核额调整
* Created by   : DEV01                                                 *
* Created on   :
* Version      : 1.0                                                   *
* Function Description:                                                *
* 管理部门年度预算考核额调整
*----------------------------------------------------------------------*
*操作数据表
*ZFI017
*----------------------------------------------------------------------*
*引用数据表
*
*----------------------------------------------------------------------*
* Modification Log:                                                    *
* Date       Programmer     Correction Number      DesingDoc Number    *
************************************************************************
include zfie0032top."全局变量说明但愿
```

include zfie0032o01."输出 Module 说明单元

include zfie0032i01."输入 Module 说明单元

include zfie0032f01."共用函数说明单元

include zfie0032fi1."输入函数说明单元

include zfie0032fi2."输入函数说明单元

include zfie0032h01."输入帮助说明单元

2、 zfie0032top 单元

```
************************************************************************
* Copyright 2003                                                       *
* All Rights Reserved                                                  *
*----------------------------------------------------------------------*
* Program Name : ZFIE0005                                              *
* TYPE         : REPORT                                                *
* Project      :SAP Implementation Project                             *
* Program Title: 管理部门年度预算考核额调整
* Created by   : DEV01                                                 *
* Created on   :
* Version      : 1.0                                                   *
* Function Description:                                                *
* 管理部门年度预算考核额调整
*----------------------------------------------------------------------*
*操作数据表
*ZFI017
*----------------------------------------------------------------------*
*引用数据表
*
*----------------------------------------------------------------------*
* Modification Log:                                                    *
* Date       Programmer     Correction Number     DesingDoc Number     *
************************************************************************
report zfie0030 message-id zdev.
constants: con_show(6)   value 'SHOW',
           con_change(6) value 'CHANGE'.
class cl_gui_cfw definition load.
tables: zfi017v,*zfi017v,bseg,setheadert,zfi010,zfi017,csku,cskt. "
```

```
*公司,年份,次数的新旧值
data: v_bukrs like zfi017-bukrs,
      v_gjahr like zfi017-gjahr,
      v_zmon  like zfi017-zmon,
      v_oldbukrs like zfi017-bukrs,
      v_oldgjahr like zfi017-gjahr,
      v_oldzmon  like zfi017-zmon.
*grid 的控件
controls: tctrl_order  type tableview using screen '0100'.
data: col type cxtab_column.
* Table for all entries loaded from database
data:   begin of order_extract occurs 0100.
        include structure zfi017v.
        include structure vimflagtab.
data: end of order_extract.
* Table for entries selected to show on screen
data: begin of order_total occurs 10.
        include structure zfi017v.
        include structure vimflagtab.
data: end of order_total.
*记录数据的状态信息的变量定义
data: begin of status_order. "state vector
        include structure vimstatus.
data: end of status_order.
data:  grid_item   type i.
data: answer(1)   type c,
      actionmode(6),
      datastate(1),
      mark_total   type i,
      init(1),
      first_flag(1).
data: wa_zfi017v type zfi017v.
data: ok_code like sy-ucomm,
      save_ok like sy-ucomm.          "OK-Code
data  g_return_code type i.
data: vim_marked(1) type c.              "mark-checkbox field
```

```
*存储屏幕选择的字段名
data   indexfldname(30).
data: gt_fieldcatalog   type lvc_t_fcat,
      text(70).
**********************************************************
*用于帮助的变量说明
*屏幕字段的帮助函数使用的变量
*用于成本要素组的属于帮助
data: begin of i_zfi010 occurs 0,
      name like zfi010-name,
end of i_zfi010.
*用于成本中心的输入帮助
data: begin of i_setheadert occurs 0,
        ktext like cskt-ktext,
      end of i_setheadert.
data: butxt like t001-bukrs.
data: ktext like cskt-kostl.
data: name like zfi010-name.
*用于输入帮助,存储帮助函数的通讯内表
data:    begin of fldtab occurs 2.
        include structure help_value.
data:    end of fldtab.
data:    f4dyn          like sy-dynnr.
data:    begin of f4hlp occurs 1.
        include structure dynpread.
data:    end of f4hlp.
```

3、ZFIE0032O01 单元

```
*************************************************************************
* Copyright 2003 C-Bons Wuhan                                          *
* All Rights Reserved                                                  *
*----------------------------------------------------------------------*
* Program Name : ZFIE0005                                              *
* TYPE         : REPORT                                                *
* Project      : SAP Implementation Project                            *
* Program Title: 管理部门年度预算考核额调整
```

```
* Created by   : DEV01                                    *
* Created on   :
* Version      : 1.0                                      *
* Function Description:                                   *
* 管理部门年度预算考核额调整
*-----------------------------------------------------------------*
*操作数据表
*ZFI017
*-----------------------------------------------------------------*
*引用数据表
*
*-----------------------------------------------------------------*
* Modification Log:                                       *
* Date       Programmer     Correction Number      DesingDoc Number   *

***********************************************************************
***********************************************************************
*&Form Name    : init_data
* Created by   : DEV01                                    *
* Created on   :
*-----------------------------------------------------------------*
* Function Description:                                   *
*根据用户输入条件的变化,进行数据读取
*-----------------------------------------------------------------*
*参数说明
*
*-----------------------------------------------------------------*
* Date       Programmer    Description         *

***********************************************************************
module init_data output.
  if first_flag eq space.
    first_flag = 'X'.
    v_gjahr = sy-datum+0(4).
    bseg-pswsl = 2.
  endif.
```

```
  if ( v_oldgjahr ne v_gjahr or v_oldbukrs ne v_bukrs and
    v_oldzmon ne v_zmon ) and
    v_gjahr ne space and  v_bukrs ne space and v_zmon ne space.
    perform fill_data.
  else.
*    SET CURSOR FIELD f LINE lin OFFSET o.
  endif.
endmodule.              " init_data  OUTPUT
*****************************************************************************
*&Form Name    : init_pbo
* Created by   : DEV01                                          *
* Created on   :
*-----------------------------------------------------------------------*
* Function Description:                                          *
*设置标题栏和工具条按钮
*-----------------------------------------------------------------------*
*参数说明
*
*-----------------------------------------------------------------------*
* Date        Programmer   Description        *

*****************************************************************************
module init_pbo output.
  set pf-status 'MAIN100'.
  set titlebar 'MAINTITLE'.
endmodule.                        " PBO  OUTPUT
*****************************************************************************
*&Form Name    :liste_show_liste
* Created by   : DEV01                                          *
* Created on   :
*-----------------------------------------------------------------------*
* Function Description:                                          *
*将显示数据写到显示字段中，并设置主键字段是否可输入
*-----------------------------------------------------------------------*
*参数说明
*
```

```
*-----------------------------------------------------------------*
* Date        Programmer    Description          *

***************************************************************************
module liste_show_liste output.
  if tctrl_order-current_line gt tctrl_order-lines.
    exit from step-loop.
  endif.
  grid_item = sy-loopc.
  zfi017v-kostl = order_extract-kostl.
  zfi017v-ktext = order_extract-ktext.
  zfi017v-kstar = order_extract-kstar.
  zfi017v-descript = order_extract-descript.
  zfi017v-adamt = order_extract-adamt.
  zfi017v-reasn = order_extract-reasn.
  zfi017v-zyearmonth = order_extract-zyearmonth.
  vim_marked = order_extract-mark.
  loop at screen.
    if ( order_extract-action = 'L'
     and screen-name = 'VIM_MARKED' ).
      screen-input = 0.
      modify screen.
    endif.
    if ( zfi017v-kostl ne space
     and screen-name = 'ZFI017V-KOSTL' ) or
     ( zfi017v-kstar ne space
     and screen-name = 'ZFI017V-KSTAR' ) .
      screen-input = 0.
      modify screen.
    endif.
  endloop.
  if vim_marked = 'M'.
    vim_marked = 'X'.
  endif.
endmodule.              " LISTE_SHOW_LISTE   OUTPUT
***************************************************************************
```

```
*&Form Name    : fill_data
* Created by   : DEV01                                    *
* Created on   :
*---------------------------------------------------------------------*
* Function Description:                                   *
*根据用户输入读取数据并初试化状态变量
*---------------------------------------------------------------------*
*参数说明
*
*---------------------------------------------------------------------*
* Date        Programmer    Description        *

***********************************************************************
form fill_data .
  select *
    into corresponding fields of order_extract
  from zfi017
  where zfi017~bukrs = v_bukrs
    and zfi017~gjahr = v_gjahr
    and zfi017~zmon = v_zmon.
    select single cskt~ktext as descript
      into (order_extract-descript)
      from cskt
      where spras = '1'
        and kokrs = '1000'
        and kostl = order_extract-kostl.
    select single ktext
    into (order_extract-ktext)
    from csku
    where spras = '1' and ktopl ='CB00'
          and kstar = order_extract-kstar.
    append order_extract.
  endselect.
  sort order_extract by bukrs gjahr kostl kstar.
*  order_extract[] = order_total[].
  if actionmode = con_change.
```

```
    perform insert_newworkarea using grid_item.
  endif.
  describe table order_extract lines tctrl_order-lines.
  tctrl_order-top_line = 1.
  clear ok_code.
  mark_total = 1.
  clear:datastate,status_order.
  v_oldgjahr = v_gjahr.
  v_oldbukrs = v_bukrs.
endform.                    " fill_data
***********************************************************************
*&Form Name    : init_ctrl
* Created by   : DEV01                                    *
* Created on   :
*---------------------------------------------------------------------*
* Function Description:                                   *
*根据数据状态，设置屏幕字段是否可以输入
*---------------------------------------------------------------------*
*参数说明
*
*---------------------------------------------------------------------*
* Date        Programmer    Description          *

***********************************************************************
module init_ctrl output.
  if actionmode eq space.
    actionmode = con_show.
  endif.
  if con_show eq actionmode.
    loop at screen.
      case screen-name.
        when  'V_BUKRS'.
          screen-input = 1.
          modify screen.
        when  'V_GJAHR'.
          screen-input = 1.
```

```
      modify screen.
    when 'V_ZMON'.
      screen-input = 1.
      modify screen.
  endcase.
endloop.
read table tctrl_order-cols into col
  with key screen-name = 'ZFI017V-KOSTL'.
if sy-subrc = 0.
  col-screen-input = '0'.
  modify tctrl_order-cols index sy-tabix from col.
endif.
read table tctrl_order-cols into col
  with key screen-name = 'ZFI017V-KSTAR'.
if sy-subrc = 0.
  col-screen-input = '0'.
  modify tctrl_order-cols index sy-tabix from col.
endif.
read table tctrl_order-cols into col
  with key screen-name = 'ZFI017V-ADAMT'.
if sy-subrc = 0.
  col-screen-input = '0'.
  modify tctrl_order-cols index sy-tabix from col.
endif.
read table tctrl_order-cols into col
  with key screen-name = 'ZFI017V-REASN'.
if sy-subrc = 0.
  col-screen-input = '0'.
  modify tctrl_order-cols index sy-tabix from col.
endif.
read table tctrl_order-cols into col
  with key screen-name = 'ZFI017V-ZYEARMONTH'.
if sy-subrc = 0.
  col-screen-input = '0'.
  modify tctrl_order-cols index sy-tabix from col.
endif.
```

```
else.
  loop at screen.
    case screen-name.
      when 'V_BUKRS'.
        screen-input = 0.
        modify screen.
      when 'V_GJAHR'.
        screen-input = 0.
        modify screen.
      when 'V_ZMON'.
        screen-input = 0.
        modify screen.
    endcase.
  endloop.
  read table tctrl_order-cols into col
    with key screen-name = 'ZFI017V-KOSTL'.
  if sy-subrc = 0.
    col-screen-input = '1'.
    modify tctrl_order-cols index sy-tabix from col.
  endif.
  read table tctrl_order-cols into col
    with key screen-name = 'ZFI017V-KSTAR'.
  if sy-subrc = 0.
    col-screen-input = '1'.
    modify tctrl_order-cols index sy-tabix from col.
  endif.
  read table tctrl_order-cols into col
    with key screen-name = 'ZFI017V-ADAMT'.
  if sy-subrc = 0.
    col-screen-input = '1'.
    modify tctrl_order-cols index sy-tabix from col.
  endif.
  read table tctrl_order-cols into col
    with key screen-name = 'ZFI017V-REASN'.
  if sy-subrc = 0.
    col-screen-input = '1'.
```

```
      modify tctrl_order-cols index sy-tabix from col.
    endif.
    read table tctrl_order-cols into col
      with key screen-name = 'ZFI017V-ZYEARMONTH'.
    if sy-subrc = 0.
      col-screen-input = '1'.
      modify tctrl_order-cols index sy-tabix from col.
    endif.
  endif.
endmodule.                " init_ctrl OUTPUT
```

4、ZFIE0032I01 单元

```
*********************************************************************
* Copyright 2003                                                    *
* All Rights Reserved                                               *
*-------------------------------------------------------------------*
* Program Name : ZFIE0005                                           *
* TYPE         : REPORT                                             *
* Project      : SAP Implementation Project                         *
* Program Title: 管理部门年度预算考核额调整
* Created by   : DEV01                                              *
* Created on   : *
* Version      : 1.0                                                *
* Function Description:                                             *
* 管理部门年度预算考核额调整
*-------------------------------------------------------------------*
*操作数据表
*ZFI017
*-------------------------------------------------------------------*
*引用数据表
*
*-------------------------------------------------------------------*
* Modification Log:                                                 *
* Date       Programmer     Correction Number     DesingDoc Number  *
*********************************************************************
*********************************************************************
```

```
*&Form Name    : LISTE_EXIT_COMMAND
* Created by   : DEV01                                    *
* Created on   :
*-----------------------------------------------------------------------*
* Function Description:                                    *
*处理系统退出命令，提示用户是否保存数据
*-----------------------------------------------------------------------*
*参数说明
*
*-----------------------------------------------------------------------*
* Date        Programmer   Description        *
***********************************************************************
module liste_exit_command input.
  data:  savestate  type i .
  case ok_code.
    when  'CANC'.
      set screen 0.
      leave screen.
    when 'BACK'.
      if datastate = 'X'.
        call function 'POPUP_TO_CONFIRM_STEP'
          exporting
            titel     = '退出维护'
            textline1 = '数据被修改。'
            textline2 = '是否先保存所做更改？'
          importing
            answer    = answer.
        case answer.
          when 'J'.
            perform save_data changing savestate.
            sy-subrc = savestate.
          when 'n'.
            sy-subrc = 1.
          when 'A'.
            sy-subrc = 0.
        endcase.
      endif.
```

```
      if sy-subrc = 0.
        set screen 0.
        leave screen.
      endif.
    when 'EXIT'.
      if datastate = 'X'.
        call function 'POPUP_TO_CONFIRM_STEP'
          exporting
            titel     = '退出维护'
            textline1 = '数据被修改。'
            textline2 = '是否先保存所做更改？'
          importing
            answer    = answer.
        case answer.
          when 'J'.
            perform save_data changing savestate.
            sy-subrc = savestate.
          when 'n'.
            sy-subrc = 1.
          when 'A'.
            sy-subrc = 0.
        endcase.
      endif.
      if sy-subrc = 0.
        set screen 0.
        leave screen.
        perform exit_program.
      endif.
  endcase.
endmodule.                " LISTE_EXIT_COMMAND  INPUT
**&---------------------------------------------------------------------
*
**&      Module  LISTE_BEFORE_LOOP  INPUT
**&---------------------------------------------------------------------
*
**       text
**----------------------------------------------------------------------
```

```
*
*MODULE  liste_before_loop  INPUT.
*
*ENDMODULE.                " LISTE_BEFORE_LOOP   INPUT
***********************************************************************
*&Form  Name    : do_mark_checkbox
*  Created  by   : DEV01                                    *
*  Created  on   : *
*---------------------------------------------------------------------*
*  Function  Description:                                    *
*处理记录选中标记字段
*---------------------------------------------------------------------*
*参数说明
*
*---------------------------------------------------------------------*
*  Date        Programmer    Description          *
***********************************************************************
module  liste_mark_checkbox  input.
  if   status_order-upd_flag  eq  space.   "only  mouse  mark
    perform  update_entry  using  space  0  tctrl_order-top_line.
  endif.
endmodule.                " LISTE_MARK_CHECKBOX   INPUT
***********************************************************************
*&Form  Name    : set_update_orderkey_flag
*  Created  by   : DEV01                                    *
*  Created  on   : *
*---------------------------------------------------------------------*
*  Function  Description:                                    *
*标记数据被修改,标记关键有新值需要检查
*---------------------------------------------------------------------*
*参数说明
*
*---------------------------------------------------------------------*
*  Date        Programmer    Description          *
***********************************************************************
module  set_update_orderkey_flag  input.
  status_order-upd_flag  =  'X'.
```

```
  status_order-auth_check = 'X'.
  status_order-st_data = 'X'.
  datastate = 'X'.
endmodule.              " set_update_orderkey_flag  INPUT
*********************************************************************
*&Form Name    : set_update_orderkey_group
* Created by   : DEV01                                  *
* Created on   :
*---------------------------------------------------------------------*
* Function Description:                                 *
*标记数据被修改,标记关键有新值需要检查
*---------------------------------------------------------------------*
*参数说明
*
*---------------------------------------------------------------------*
* Date        Programmer   Description          *
*********************************************************************
module set_update_orderkey_group input.
  status_order-upd_flag = 'X'.
  status_order-auth_check = 'X'.
  status_order-st_mode = 'X'.
  datastate = 'X'.
endmodule.              " set_update_orderkey_flag  INPUT
*********************************************************************
*&Form Name    : set_update_order_flag
* Created by   : DEV01                                  *
* Created on   : *
*---------------------------------------------------------------------*
* Function Description:                                 *
*标记数据被修改
*---------------------------------------------------------------------*
*参数说明
*
*---------------------------------------------------------------------*
* Date        Programmer   Description          *
*********************************************************************
module set_update_order_flag input.
```

```
  status_order-upd_flag = 'X'.
  datastate = 'X'.
endmodule.                " SET_UPDATE_FLAG  INPUT
*****************************************************************************
*&Form Name    : liste_update_order
* Created by   : DEV01                                                      *
* Created on   :
*---------------------------------------------------------------------------*
* Function Description:                                                     *
*将工作区数据更新到显示和缓冲内表中
*---------------------------------------------------------------------------*
*参数说明
*
*---------------------------------------------------------------------------*
* Date        Programmer   Description                                      *
*****************************************************************************
module liste_update_order input.
  if status_order-upd_flag = 'X'.
    perform check_key_order.
    perform update_tab_order using sy-subrc sy-tabix.
    clear status_order.
  endif.
endmodule.                " LISTE_UPDATE_LISTE  INPUT
*****************************************************************************
*&Form Name    : check_key_order
* Created by   : DEV01                                                      *
* Created on   :
*---------------------------------------------------------------------------*
* Function Description:                                                     *
*关键字检查，判断是否重复
*---------------------------------------------------------------------------*
*参数说明
*
*---------------------------------------------------------------------------*
* Date        Programmer   Description                                      *
*****************************************************************************
form check_key_order .
```

```
*  CLEAR order_total.
*  READ TABLE order_total WITH KEY
*    kostl = zfi017v-kostl
*    KSTAR = zfi017v-KSTAR BINARY SEARCH.
*  CHECK sy-subrc EQ 0.
*  IF status_order-auth_check = 'X'.
*    IF order_total-action EQ 'D'
*    OR order_total-action  EQ 'X'
*    OR order_total-action  EQ 'Y'.
*      MESSAGE ID 'SV' TYPE 'E' NUMBER '010'.
*    ELSE.
*      MESSAGE ID 'SV' TYPE 'E' NUMBER '009'.
*    ENDIF.
*  ENDIF.
endform.                    "check_key_order
***********************************************************************
*&Form Name    : liste_after_loop
* Created by   : DEV01                                    *
* Created on   : *
*---------------------------------------------------------------------*
* Function Description:                                   *
*处理用户操作
*---------------------------------------------------------------------*
*参数说明
*
*---------------------------------------------------------------------*
* Date        Programmer   Description          *
***********************************************************************
module liste_after_loop input.
*  DATA: savestate TYPE i.
  save_ok = ok_code.
  clear ok_code.
  case save_ok.
    when 'EXIT' or 'BACK' or 'CANC'.
      if datastate = 'C'.
        call function 'POPUP_TO_CONFIRM_STEP'
          exporting
```

```
      titel     = '退出维护'
      textline1 = '数据被修改。'
      textline2 = '是否先保存所做更改？'
    importing
      answer    = answer.
  case answer.
    when 'J'.
      perform save_data changing savestate.
      sy-subrc = savestate.
    when 'n'.
      sy-subrc = 1.
    when 'A'.
      sy-subrc = 0.
  endcase.
endif.
if sy-subrc = 0.
  set screen 0.
  leave screen.
  perform exit_program.
endif.
when 'EDIT'.
if datastate eq space.
  if con_show eq actionmode.
    call function 'ENQUEUE_EZFIE017'
      exporting
        bukrs         = v_bukrs
        gjahr         = v_gjahr
        zmon          = v_zmon
      exceptions
        foreign_lock   = 2
        system_failure = 3.
    if sy-subrc ne 0.
      message i622 with v_bukrs v_gjahr sy-msgv1.
    else.
      actionmode = con_change.
      if ( v_oldgjahr = v_gjahr and v_oldbukrs = v_bukrs
        and v_gjahr ne space and v_bukrs ne space ).
```

```
          perform insert_newworkarea using grid_item.
        endif.
      endif.
    endif.
  endif.
when 'DELE'.
  if con_show ne actionmode.
    perform delete_order.
  endif.
when 'SAVE' or 'SAVV'.
  if datastate ne space.
    perform save_data changing savestate.
    if savestate eq 0.
      call function 'DEQUEUE_EZFIE017'
        exporting
          bukrs = v_bukrs
          gjahr = v_gjahr
          zmon  = v_zmon.
      clear datastate.
    endif.
  else.
    call function 'DEQUEUE_EZFIE017'
      exporting
        bukrs = v_bukrs
        gjahr = v_gjahr
        zmon  = v_zmon.
    perform clear_action.
  endif.
when 'SOUP'.
* search for selected columns.
  loop at tctrl_order-cols into col.
    if col-selected = 'X'.
      indexfldname = col-screen-name+11.
      sort order_total by (indexfldname).
      exit.
    endif.
  endloop.
```

```
    when 'SODO'.
      loop at tctrl_order-cols into col.
        if col-selected = 'X'.
          indexfldname = col-screen-name+11.
          sort order_total by (indexfldname) descending.
          exit.
        endif.
      endloop.
    when others.
      call method cl_gui_cfw=>dispatch.
  endcase.
  call method cl_gui_cfw=>flush.
endmodule.                " LISTE_AFTER_LOOP  INPUT
*****************************************************************************
*&Form Name    : insert_newworkarea
* Created by   : DEV01                                        *
* Created on   : *
*-----------------------------------------------------------------------*
* Function Description:                                        *
*为编辑数据添加临时空记录
*-----------------------------------------------------------------------*
*参数说明
*-->p_entries:插入新空白记录数
*-----------------------------------------------------------------------*
* Date        Programmer   Description          *
*****************************************************************************
form insert_newworkarea using    p_entries.
  if p_entries eq 0.
    p_entries = 20.
  endif.
  clear order_extract.
  move 'L' to order_extract-action.
  do p_entries times.
    append order_extract.
  enddo.
  move tctrl_order-lines to tctrl_order-top_line.
  describe table order_extract lines tctrl_order-lines.
```

```
endform.                    " insert_newworkarea
*****************************************************************************
*&Form Name    : delete_order
* Created by   : DEV01                                        *
* Created on   :
*---------------------------------------------------------------------*
* Function Description:                                       *
*删除的数据,在缓冲数据中设立删除标志(只能删除新建且未保存的定价)
*---------------------------------------------------------------------*
*参数说明
*
*---------------------------------------------------------------------*
* Date        Programmer   Description          *
*                                                *
*****************************************************************************
*---------------------------------------------------------------------*
form delete_order .

  loop at order_extract.
    if order_extract-mark = 'M' and
      order_extract-action ne 'L'.
      delete order_extract.
    endif.
  endloop.
  describe table order_extract lines tctrl_order-lines.
endform.                    " delete_order
```

5、ZFIE0032F01 单元

```
*****************************************************************************
* Copyright 2003 Wuhan                                        *
* All Rights Reserved                                         *
*---------------------------------------------------------------------*
* Program Name : ZFIE0005                                     *
* TYPE         : REPORT                                       *
* Project      : SAP Implementation Project                   *
* Program Title: 管理部门年度预算考核额调整
* Created by   : DEV01                                        *
```

```
* Created on   : *
* Version      : 1.0                                    *
* Function Description:                                 *
* 管理部门年度预算考核额调整
*-----------------------------------------------------------------------*
*操作数据表
*ZFI017
*-----------------------------------------------------------------------*
*引用数据表
*
*-----------------------------------------------------------------------*
* Modification Log:                                     *
* Date        Programmer     Correction Number      DesingDoc Number   *
*************************************************************************
*************************************************************************
*&Form Name    : exit_program
* Created by   : DEV01                                  *
* Created on   : *
*-----------------------------------------------------------------------*
* Function Description:                                 *
*退出程序时，解除锁
*-----------------------------------------------------------------------*
*参数说明
*
*-----------------------------------------------------------------------*
* Date        Programmer   Description          *
*************************************************************************
form exit_program.
  call function 'DEQUEUE_EZFIE017'
    exporting
      bukrs = v_bukrs
      gjahr = v_gjahr
      znum = v_zmon.
  leave program.
endform.                          " exit_program
```

6、ZFIE0032FI1 单元

```
************************************************************************
* Copyright 2003 *
* All Rights Reserved                                    *
*----------------------------------------------------------------------*
* Program Name : ZFIE0005                                *
* TYPE         : REPORT                                  *
* Project      : SAP Implementation Project              *
* Program Title: 管理部门年度预算考核额调整
* Created by   : DEV01                                   *
* Created on   : *
* Version      : 1.0                                     *
* Function Description:                                  *
* 管理部门年度预算考核额调整
*----------------------------------------------------------------------*
*操作数据表
*ZFI017
*----------------------------------------------------------------------*
*引用数据表
*
*----------------------------------------------------------------------*
* Modification Log:                                      *
* Date       Programmer     Correction Number     DesingDoc Number   *
************************************************************************

************************************************************************
*&Form Name    : clear_action
* Created by   : DEV01                                   *
* Created on   :
*----------------------------------------------------------------------*
* Function Description:                                  *
*清除数据操作状态，恢复到初试显示状态
*----------------------------------------------------------------------*
*参数说明
*
*----------------------------------------------------------------------*
* Date       Programmer   Description        *
```

```
*************************************************************************
form clear_action.
*  order_extract[] = order_total[].
  actionmode = con_show.
  describe table order_extract lines tctrl_order-lines.
  tctrl_order-top_line = 1.
  clear ok_code.
  mark_total = 1.
  clear:datastate,status_order.
  clear status_order.
endform.                  " clear_action
*************************************************************************
*&Form Name    : clear_mark
* Created by   : DEV01                                    *
* Created on   :
*-----------------------------------------------------------------------*
* Function Description:                                   *
*清除记录选中标记
*-----------------------------------------------------------------------*
*参数说明
*
*-----------------------------------------------------------------------*
* Date        Programmer    Description          *
*************************************************************************
form clear_mark .
  loop at order_total.
    order_total-mark = space.
    modify order_total index sy-tabix.
  endloop.
endform.                  " clear_mark
*************************************************************************
*&Form Name    : save_data
* Created by   : DEV01                                    *
* Created on   :
*-----------------------------------------------------------------------*
* Function Description:                                   *
*保存数据
```

```
*---------------------------------------------------------------------*
*参数说明
*<--savestate:返回保存是否成功，成功为 0 值
*---------------------------------------------------------------------*
* Date        Programmer    Description          *
***********************************************************************
form save_data changing savestate type i.
  data i_znum type i.
  savestate = -1.
  delete from zfi017
    where bukrs = v_bukrs and
          gjahr = v_gjahr and
          zmon = v_zmon.
  i_znum = 1.
  loop at order_extract.
    if order_extract-action eq space.
      zfi017-bukrs = order_extract-bukrs.
      zfi017-gjahr = order_extract-gjahr.
      zfi017-zmon = order_extract-zmon.
      zfi017-znum = i_znum.
      zfi017-kostl = order_extract-kostl.
      zfi017-kstar = order_extract-kstar.
      zfi017-adamt = order_extract-adamt.
      zfi017-reasn = order_extract-reasn.
      zfi017-zyearmonth = order_extract-zyearmonth.
      insert zfi017.
      if sy-subrc ne 0.
        rollback work.
        exit.
      endif.
      i_znum = i_znum + 1.
    elseif order_extract-action = 'U'.
      zfi017-bukrs = order_extract-bukrs.
      zfi017-gjahr = order_extract-gjahr.
      zfi017-zmon = order_extract-zmon.
      zfi017-znum = i_znum.
      zfi017-kostl = order_extract-kostl.
```

```
  zfi017-kstar = order_extract-kstar.

  zfi017-adamt = order_extract-adamt.

  zfi017-reasn = order_extract-reasn.

  zfi017-zyearmonth = order_extract-zyearmonth.

  insert zfi017.

  if sy-subrc ne 0.

    rollback work.

    exit.

  endif.

  i_znum = i_znum + 1.

  order_total-action = space.

  modify order_extract.

elseif order_extract-action = 'X'.

  delete order_total.

elseif ( ( order_extract-action = 'D' ) or

  ( order_extract-action = 'Y' ) ).

  delete order_extract.

elseif order_extract-action = 'N'.

  zfi017-bukrs = order_extract-bukrs.

  zfi017-gjahr = order_extract-gjahr.

  zfi017-zmon = order_extract-zmon.

  zfi017-znum = i_znum.

  zfi017-kostl = order_extract-kostl.

  zfi017-kstar = order_extract-kstar.

  zfi017-adamt = order_extract-adamt.

  zfi017-reasn = order_extract-reasn.

  zfi017-zyearmonth = order_extract-zyearmonth.

  insert zfi017.

  if sy-subrc ne 0.

    rollback work.

    exit.

  endif.

  i_znum = i_znum + 1.

  order_total-action = space.

  modify order_extract.

endif.
```

7、ZFIE0032FI2 单元

```
************************************************************************
* Copyright 2003 *
* All Rights Reserved                                    *
*----------------------------------------------------------------------*
* Program Name : ZFIE0005                                *
* TYPE         : REPORT                                  *
* Project      : SAP Implementation Project              *
* Program Title: 管理部门年度预算考核额调整
* Created by   : DEV01                                   *
* Created on   : *
* Version      : 1.0                                     *
* Function Description:                                  *
* 管理部门年度预算考核额调整
*----------------------------------------------------------------------*
*操作数据表
*ZFI017
*----------------------------------------------------------------------*
*引用数据表
*
*----------------------------------------------------------------------*
* Modification Log:                                      *
* Date       Programmer     Correction Number     DesingDoc Number   *
************************************************************************

************************************************************************
*&Form Name    : update_tab_order
* Created by   : DEV01                                   *
* Created on   : *
*----------------------------------------------------------------------*
* Function Description:                                  *
*将显示缓冲 work area 的数据更新到内表
*----------------------------------------------------------------------*
*参数说明              *
*  --> p_rc    ：记录定位操作返回的 sy-subrc 的值
*  --> p_index ：记录的索引位置
*----------------------------------------------------------------------*
```

```
* Date        Programmer    Description          *
***********************************************************************
form update_tab_order using value(p_rc)
                      value(p_index).
   perform update_entry using 'X'  p_rc p_index.
endform.                  " update_tab
***********************************************************************
*&Form Name    : update_tab_order
* Created by   : DEV01                                    *
* Created on   :*
*-----------------------------------------------------------------*
* Function Description:                                   *
*将显示缓冲 work area 的数据更新到内表
*-----------------------------------------------------------------*
*参数说明
*  -->  p_workarea : 'X' 更新显示和缓冲双份内表数据
*                    space 只更新显示内表数据
*  -->  p_rc    : 记录定位操作返回的 sy-subrc 的值
*  -->  p_index : 记录的索引位置
*-----------------------------------------------------------------*
* Date        Programmer    Description          *
***********************************************************************
form update_entry using value(p_workarea)
                        value(p_rc)
                        value(p_index).
    read table order_extract index tctrl_order-top_line.
    if order_extract-action = 'L'.
      move-corresponding zfi017v to order_extract.
      order_extract-bukrs = v_bukrs.
      order_extract-gjahr = v_gjahr.
      order_extract-zmon = v_zmon.
      order_extract-action  = 'N'.
      if vim_marked ne space.
        order_extract-mark  = 'M'.
      endif.
      modify order_extract index tctrl_order-top_line.
      clear order_extract.
```

```
      order_extract-action  = 'L'.
      append order_extract.
    else.
      move-corresponding zfi017v to order_extract.
      order_extract-bukrs = v_bukrs.
      order_extract-gjahr = v_gjahr.
      order_extract-zmon = v_zmon.
      order_extract-action  = 'U'.
      if vim_marked ne space.
        order_extract-mark  = 'M'.
      endif.
      modify order_extract index tctrl_order-top_line.
    endif.
    describe table order_extract lines tctrl_order-lines.
endform.                    "update_tab_order
  endloop.
  commit work.
  savestate = 0.
  perform clear_action.
  message s024.
endform.                    " save_data
```

8、ZFIE0032H01 单元

```
*************************************************************************
* Copyright 2003 Wuhan                                  *
* All Rights Reserved                                   *
*-----------------------------------------------------------------------*
* Program Name : ZFIE0005                               *
* TYPE         : REPORT                                 *
* Project      : SAP Implementation Project             *
* Program Title: 管理部门年度预算考核额调整
* Created by   : DEV01                                  *
* Created on   : *
* Version      : 1.0                                    *
* Function Description:                                 *
* 管理部门年度预算考核额调整
*-----------------------------------------------------------------------*
```

```
*操作数据表
*ZFI017
*---------------------------------------------------------------------*
*引用数据表
*
*---------------------------------------------------------------------*
* Modification Log:                                                   *
* Date       Programmer     Correction Number      DesingDoc Number   *
***********************************************************************
***********************************************************************
*&Form Name    : v_bukrs_check
* Created by   : DEV01                                                *
* Created on   : *
*---------------------------------------------------------------------*
* Function Description:                                               *
*公司代码字段输入检查:不能为空且代码不能不存在
*---------------------------------------------------------------------*
*参数说明
*
*---------------------------------------------------------------------*
* Date       Programmer   Description        *
***********************************************************************
module v_bukrs_check input.
  tables: t001.
  if v_bukrs = space.
    message e023 with '公司代码'.
  endif.
  select single * from t001 where bukrs = v_bukrs.
  if sy-subrc ne 0.
    message e014 with v_bukrs.
  endif.
endmodule.              " v_bukrs_check  INPUT
***********************************************************************
*&Form Name    : v_gjahr_check
* Created by   : DEV01                                                *
* Created on   : *
*---------------------------------------------------------------------*
```

```
* Function Description:                                    *
*年份代码字段输入检查:不能为空且年份不能小于 1990
*-----------------------------------------------------------------------*
*参数说明
*
*-----------------------------------------------------------------------*
* Date        Programmer   Description          *
*************************************************************************
module v_gjahr_check input.
  if v_gjahr = space.
    message e023 with '年份'.
  endif.
  if v_gjahr gt 9999 or v_gjahr lt 1000.
    message e046.
  endif.
endmodule.                " v_gjahr_check  INPUT
*************************************************************************
*&Form Name    : v_zmon_check
* Created by   : DEV01                                      *
* Created on   :  *
*-----------------------------------------------------------------------*
* Function Description:                                     *
*月份字段输入检查
*-----------------------------------------------------------------------*
*参数说明
*
*-----------------------------------------------------------------------*
* Date        Programmer   Description          *
*************************************************************************
module v_zmon_check input.
  if v_zmon = space.
    message e023 with '月份'.
  endif.
  if v_zmon gt 12 or v_zmon lt 1.
    message e047.
  endif.
endmodule.                " v_gjahr_check  INPUT
```

```
*************************************************************************
*&Form Name    : select_kostl
* Created by   : DEV01                                    *
* Created on   :   *
*-----------------------------------------------------------------------*
* Function Description:                                   *
*获取成本中心组(帮助输入)
*-----------------------------------------------------------------------*
*参数说明
*
*-----------------------------------------------------------------------*
* Date        Programmer   Description          *
*************************************************************************
module select_kostl input.
  data:
      dynpro_values type table of dynpread,
      field_value like line of dynpro_values,
      l_year like bkpf-gjahr,
      g_kh(10).
  data: begin of i_tree occurs 0,
     setname like setnode-setname,
   end of i_tree.
  data: i_subtree like i_tree occurs 0 with header line,
   i_temptree like i_tree occurs 0 with header line.
  data l_count type i .
  ranges: s_setname for setnode-setname.
  ranges: s_kostl for cskt-kostl.

  refresh dynpro_values.
  field_value-fieldname = 'V_GJAHR'.
  append field_value to dynpro_values.

  call function 'DYNP_VALUES_READ'
   exporting
     dyname           = sy-repid
     dynumb           = sy-dynnr
     translate_to_upper = 'X'
```

```
  tables
    dynpfields        = dynpro_values.

  read table dynpro_values index 1 into field_value.
  l_year = field_value-fieldvalue(4).

  refresh i_setheadert.
  s_setname-option = 'EQ'.
  s_setname-sign = 'I'.
  l_count = 1.
  refresh i_temptree.
  refresh i_subtree.
*特殊处理，2005 年预算用 2006 的机构
  if l_year < 2006.
    l_year = 2006.
  endif.
  concatenate 'KH' l_year into g_kh.
  while l_count > 0.
    select setnode~subsetname as setname
      into (setheadert-setname)
      from setnode
        where setnode~setclass = '0101'
          and setnode~subclass = '1000'
          and setnode~subsetcls = '0101'
          and setnode~subsetscls = '1000'
          and setnode~setname = g_kh.
      move setheadert-setname to s_setname-low.
      append s_setname.
      i_temptree-setname = setheadert-setname.
      append i_temptree.
    endselect.
    append lines of i_temptree to i_subtree.
    describe table i_subtree lines l_count.
    if l_count ne 0.
      read table i_subtree index 1.
      g_kh = i_subtree-setname.
      delete i_subtree index 1.
```

```
  endif.
  refresh i_temptree.
  clear i_temptree.
endwhile.

select valsign as sign   valoption as option
  valfrom as low   valto   as high
  into table s_kostl
from setleaf
where setclass = '0101'
  and subclass = '1000'  and setname in s_setname.

refresh i_setheadert.
select single  bukrs  from t001  into butxt
where bukrs = v_bukrs.
if sy-subrc <> 0.
  message id 'ZDEV' type 'S' number '049'.
  exit.
endif.
select cskt~kostl cskt~ktext
  into (cskt-kostl,cskt-ktext)
from cskt
 where spras = '1'
  and kokrs = '1000'
  and kostl in s_kostl.
  move cskt-kostl to i_setheadert-ktext.
  append i_setheadert.
  move cskt-ktext to i_setheadert-ktext.
  append i_setheadert.
endselect.
refresh fldtab.
clear fldtab.
fldtab-tabname    = 'CSKT'.
fldtab-fieldname  = 'KOSTL'.
fldtab-selectflag = 'X'.
append fldtab.
clear fldtab.
```

```
  fldtab-tabname     = 'CSKT'.
  fldtab-fieldname   = 'KTEXT'.
  append  fldtab.
  call  function  'HELP_VALUES_GET_WITH_TABLE'
    exporting
      display       = space
      fieldname     = 'KOSTL'
      tabname       = 'ZFI017V'
    importing
      select_value  =  ktext
    tables
      fields        = fldtab
      valuetab      = i_setheadert.
  if  not  ktext  is  initial.
    zfi017v-kostl  =  ktext.
  endif.
endmodule.                  "  select_kostl   INPUT

*****************************************************************************
*&Form  Name     :  select_zgroup
*  Created  by    :  DEV01                                          *
*  Created  on    :  *
*-----------------------------------------------------------------------*
*  Function  Description:                                            *
*获取某个年度成本要素组(帮助输入)
*-----------------------------------------------------------------------*
*参数说明
*
*-----------------------------------------------------------------------*
*  Date         Programmer    Description          *
*****************************************************************************
module  select_zgroup  input.
  refresh  i_zfi010.
*********XIONGHM***************
  select  kstar  ktext
    into  (csku-kstar,csku-ktext)
      from  csku
```

```
      where spras = '1' and ktopl ='CB00'.
    i_zfi010-name = csku-kstar.
    append i_zfi010.
    i_zfi010-name = csku-ktext.
    append i_zfi010.
  endselect.

  refresh fldtab.
  clear fldtab.
  fldtab-tabname    = 'ZFI017V'.
  fldtab-fieldname  = 'KSTAR'.
  fldtab-selectflag = 'X'.
  append fldtab.
  clear fldtab.
  fldtab-tabname    = 'ZFI017V'.
  fldtab-fieldname  = 'KTEXT'.
  append fldtab.
  call function 'HELP_VALUES_GET_WITH_TABLE'
    exporting
      display      = space
      fieldname    = 'KSTAR'
      tabname      = 'ZFI017V'
    importing
      select_value = name
    tables
      fields       = fldtab
      valuetab     = i_zfi010.
  if not name is initial.
    zfi017v-kstar = name.
  endif.
endmodule.                " select_prodh  INPUT
*****************************************************************************
*&Form Name    : v_check_kostl
* Created by   : DEV01                                                      *
* Created on   : *
*---------------------------------------------------------------------------*
* Function Description:                                                     *
```

```
*检查成本输入中心是否存在
*---------------------------------------------------------------------*
*参数说明
*
*---------------------------------------------------------------------*
* Date        Programmer    Description          *
************************************************************************
module v_check_kostl input.
  if zfi017v-kostl eq space.
    message e621 with zfi017v-kostl.
  endif.
  if status_order-st_data ne space.
    select single ktext as descript
      into zfi017v-descript
    from cskt
     where spras = '1'
      and kokrs = '1000'
      and kostl = zfi017v-kostl.
    if sy-subrc ne 0.
      message e620 with zfi017v-kostl.
    endif.
  endif.
endmodule.                " v_check_kostl INPUT
************************************************************************
*&Form Name    : v_check_zgroup
* Created by   : DEV01                                    *
* Created on   :   *
*---------------------------------------------------------------------*
* Function Description:                                   *
*检查输入成本要素组是否存在
*---------------------------------------------------------------------*
*参数说明
*
*---------------------------------------------------------------------*
* Date        Programmer    Description          *
************************************************************************
module v_check_zgroup input.
```

```
  if zfi017v-kstar eq space.
    message e621 with zfi017v-kstar.
  endif.
  if status_order-st_mode ne space.
    select single ktext
    into (zfi017v-ktext)
    from csku
    where spras = '1' and ktopl ='CB00'
          and kstar = zfi017v-kstar.
    if sy-subrc ne 0.
      message e621 with zfi017v-kstar.
    endif.
  endif.
endmodule.              " v_check_zgroup  INPUT
```

9、屏幕 PBO 和 PAI

```
*BEFORE OUTPUT
process before output.
  module init_pbo.
  module init_data.
  module init_ctrl.
  loop at order_extract with control tctrl_order
    cursor tctrl_order-top_line.

    module liste_show_liste.
  endloop.

*After input
process after input.
  module liste_exit_command at exit-command.
  field v_bukrs module v_bukrs_check on request.
  field v_gjahr module v_gjahr_check on request.
  field v_zmon module v_zmon_check on request.
  module liste_before_loop.
  loop at order_extract.
    field zfi017v-kostl
      module set_update_orderkey_flag on request.
    field zfi017v-kstar
```

```
      module set_update_orderkey_group on request.
    chain.
      field zfi017v-adamt.
      field zfi017v-reasn.
      field zfi017v-zyearmonth.
      module set_update_order_flag on chain-request.
    endchain.
    field vim_marked module liste_mark_checkbox.
**检查
    field zfi017v-kostl module v_check_kostl.
    field zfi017v-kstar module v_check_zgroup.
* save current data to order_total
    chain.
      field zfi017v-kostl.
      field zfi017v-kstar.
      field zfi017v-adamt.
      field zfi017v-reasn.
      field zfi017v-zyearmonth.
      module liste_update_order.
    endchain.
  endloop.
*deal with function Code
  module liste_after_loop.

process on value-request.
  field zfi017v-kostl module select_kostl.
  field zfi017v-kstar module select_zgroup.
```

正确地使用 SAP 的标准对话框函数

在用户设计 sap 的程序时，经常需要一些对话框，用户可以自己编写，但使用 SAP 系统中提供了的对话框函数将减少许多开发工作。

1、sap 的函数组列表和用途说明

适用情况	Function group
提示用户将可能丢失数据	SPO1
提示用户对某个问题选择 Yes 或者 No	SPO1
提示用户将可能丢失数据，并询问用户是否继续操作	SPO1

提示用户在多个操作中选择一个操作	SPO2
提示用户是继续当前操作或者取消当前操作	SPO2
提示用户输入数据（可以根据一个表检查或者不检查输入值）	SPO4
将数据显示给用户	SPO4
将详细数据显示给用户	SPO6
从列表中选择数据	SP05
用可滚动的对话框显示数据给用户	STAB
从视图或者数据表中打印数据	STPR

2、函数列表和说明

2.1 SPO1 的函数

- **POPUP_TO_CONFIRM_STEP**

  用此函数可以建立一个对话框用于询问用户是否执行某步操作，用户可以选择 Yes No 或者 Cancel。该函数可以传入一个标题和两行的文本（提示问题）。系统在窗口上显示一个绿色问号图标。

  可以设置某个按钮作为默认按钮。

```
CALL FUNCTION 'POPUP_TO_CONFIRM_STEP'
    EXPORTING TITEL        = '确认提示测试'
              TEXTLINE1    = '确实要执行'
              TEXTLINE2    = '测试？'
              CANCEL_DISPLAY = SPACE  "不显示 CANCEL 按钮
    IMPORTING ANSWER       = ANSWER.
```

- **POPUP_TO_CONFIRM_WITH_MESSAGE**

  类似 POPUP_TO_CONFIRM_STEP，只是多三行的文本错误诊断提示。 系统在窗口上显示一个绿色问号图标。

```
call function 'POPUP_TO_CONFIRM_WITH_MESSAGE'
    exporting titel        = '确认提示测试'
              textline1    = '确实要执行                    ;;'
              textline2    = spop-textline2
              diagnosetext1 = spop-diagnose1
              diagnosetext2 = spop-diagnose2
              diagnosetext3 = spop-diagnose3
    importing answer       = answer.
```

- **POPUP_TO_CONFIRM_WITH_VALUE**

  用此函数可以建立一个对话框用于询问用户是否执行某步操作，该操作可能会丢失数据，用户可以选择 Yes No 或者 Cancel。该函数可以传入一个标题，两行的文本（提示问题）和一个对象值（对象值将会插入在提示问

题文本的两部分之间）。系统在窗口上显示一个绿色问号图标。

可以设置某个按钮作为默认按钮。

```
CALL FUNCTION 'POPUP_TO_CONFIRM_WITH_VALUE'
  EXPORTING TITEL       = TITEL
            TEXT_BEFORE  = '确实要执行'
            OBJECTVALUE  = 'TEST'
            TEXT_AFTER   = '?'
  IMPORTING ANSWER      = ANSWER.
```

- **POPUP_TO_CONFIRM_LOSS_OF_DATA**

  用此函数可以建立一个对话框用于询问用户是否执行某步操作，该操作可能会丢失数据，用户可以选择 Yes No 或者 Cancel。该函数可以传入一个标题和一个两行的文本（提示问题）。 系统在窗口上显示一个黄色！图标和一行“数据将丢失。”。

  可以设置某个按钮作为默认按钮。

```
CALL FUNCTION 'POPUP_TO_CONFIRM_LOSS_OF_DATA'
  EXPORTING TITEL       = TITEL
            TEXTLINE1    = SPOP-TEXTLINE1
            TEXTLINE2    = SPOP-TEXTLINE2
  IMPORTING ANSWER      = ANSWER.
```

- **POPUP_TO_CONFIRM**

  该函数是 POPUP 函数的增强版，可以自定义按钮文本和图标。

2.2、Function group SPO2

- **POPUP_TO_DECIDE**

  显示一个对话框，用户可以两个操作中的一个或者取消。可以传入三行提示文本。

- **POPUP_TO_DECIDE_WITH_MESSAGE**

  类同 **POPUP_TO_DECIDE**，程序员可以多传入 With this function module you create a dialog box in which you inform the user about a specific decision point via a diagnosis text, during an action. He or she can choose one of two alternative actions offered or cancel the action.

  The action, the diagnosis text, the question and the alternative actions are passed as parameters.

  The user action (Alternative 1, Alternative 2, or Cancel) is returned in a parameter.

2.3、Function group SPO4

- **POPUP_GET_VALUES**

  This function module sends a dialog box for data display and input.

  The input fields are passed in a structure and must be defined in the Dictionary. You can also speci

fy individual field display attributes and a field text, if the key word from the Dictionary is not to be displayed as field text in the dialog box, in the structure.

The standard help functionality (F1, F4) is supported.

- **POPUP_GET_VALUES_DB_CHECKED**

  This function module sends a dialog box for data to be input und checked against the database.

  The input fields are passed in a structure and must be defined in the Dictionary. You can also specify individual field display attributes and a field text in the structure, if the key word from the Dictionary is not to be displayed as field text in the dialog box.

  A comparison operator for checking the input data in the database is passed. You can specify whether the check is for the existence or absence of an object. A foreign key relationship check is supported.

  The standard help functionality (F1, F4) is supported.

  The user action is returned in a parameter.

- **POPUP_GET_VALUES_USER_CHECKED**

  This function module sends a dialog box for data to be input and checked in an external sub-routine (user exit). The input fields are passed in a structure and must be defined in the dictionary. You can also specify individual field display attributes and a field text in the structure, if the key word from the Dictionary is not to be displayed as field text in the dialog box.

  The Data input by the user in the dialog box are passed to the sub-routine specified in the interface for checking. Errors found by the check are entered in an error structure and are evaluated on return from the sub-routine by the function module.

  The standard help functionality (F1, F4) is supported.

  The user action (*Continue* or *Cancel*) is returned in a parameter.

- **POPUP_GET_VALUES_USER_HELP**

  This function module sends a dialog box for data to be input with the possibility of a check in an external sub-routine (user exit) and branching in a user F1 or F4 help.

  The input fields are passed in a structure and must be defined in the Dictionary. You can also specify individual field display attributes and a field text in the structure, if the key word from the Dictionary is not to be displayed as field text in the dialog box.

  You can pass the data which are entered by the user in a dialog box to a sub-routine which must be specified in the interface for checking. Errors occurring in the check are stored in an error structure and are analyzed by the function module upon return from the sub-routine. The data, and an error message, if appropriate, are displayed again.

  The standard help functionality (F1, F4) is supported.

User exits for a user F1 or F4 help can also be specified.

The user action (*Continue* or *Cancel*) is returned in a parameter.

- **POPUP_GET_VALUES_USER_BUTTONS**

  This function module is like the previous function module POPUP_GET_VALUES_USER_HELP, with the additional possibility of passing one or two additional pushbuttons and a standard pushbutton, which the user can name.

- **POPUP_GET_VALUES_SET_MAX_FIELD**

  With this function module you can specify the maximum number of fields which can be displayed in dialog boxes for this function group (SPO4). The specified value is stored in the function group local memory and applies for the rest of the application. Dialog boxes which display more then this number of fields are displayed with a scroll bar.

2.4、Function group SPO6

- **POPUP_DISPLAY_TEXT**

  With this function module you display a text which exists in the system in a dialog box.

- **POPUP_DISPLAY_TEXT_WITH_PARAMS**

  With this function module you display a text which exists in the system with parameters in a dialog box. The parameter values are passed in a table. The use of numbered texts is recommended, to make the parameter values translatable.

  The parameter names must be passed in upper-case letters.

2.5、Function group SPO5

- **POPUP_TO_DECIDE_LIST**

  从列表中选择数据，样例程序

```
report  rsspo500.
data:  selectlist like spopli occurs 5 with header line.
data:  antwort  type c.
while antwort ne 2.
clear  selectlist.
refresh selectlist.
selectlist-varoption = 显示含有单选按钮的弹出框'.
append selectlist.
selectlist-varoption = '显示含有复选框的弹出框'.
```

```
selectlist-selflag   = 'X'.
append selectlist.
call function 'POPUP_TO_DECIDE_LIST'
     exporting
*        CURSORLINE        = 1
*        MARK_FLAG         = ' '
         mark_max          = 1
         start_col         = 10
         start_row         = 10
         textline1         = 'Text1'
         textline2         = 'POPUP_TO_DECIDE_LIST'
         textline3         = 'TEXT3'
         titel             = 'TITLE '
     importing
         answer            =  antwort
     tables
         t_spopli          =  selectlist
     exceptions
         not_enough_answers = 1
         too_much_answers   = 2
         too_much_marks     = 3
         others             = 4.
  if antwort eq 'A'.
    exit.
  endif.
endwhile.

if antwort ne 'A'.
  clear   selectlist.
  refresh selectlist.
  selectlist-varoption = '最多 15 个选项'.
  selectlist-selflag   = 'X'.
  append selectlist.
  selectlist-varoption = '含有复选框'.
  selectlist-selflag   = 'X'.
  append selectlist.
  selectlist-varoption = '或单选按钮'.
```

```
  selectlist-selflag   = 'X'.
  append  selectlist.
  selectlist-varoption  = '典型的列表功能：'.
  selectlist-selflag   = 'X'.
  append  selectlist.
  selectlist-varoption  = '*  选择'.
  selectlist-selflag   = 'X'.
  append  selectlist.
  selectlist-varoption  = '*  选择全部'.
  selectlist-selflag   = 'X'.
  append  selectlist.
  selectlist-varoption  = '  取消全部选择'.
  selectlist-selflag   = 'X'.
  append  selectlist.
  selectlist-varoption  = '3  70 字符/选项'.
  selectlist-selflag   = 'X'.
  append  selectlist.
  selectlist-varoption  = '65  字符/选项'.
  selectlist-selflag   = 'X'.
  append  selectlist.
  selectlist-varoption  =  '激活/不激活可选行'.
  selectlist-selflag   = 'X'.
  selectlist-inactive  = 'X'.
  append  selectlist.

*  2.  Aufruf  ************************************************************

  call  function  'POPUP_TO_DECIDE_LIST'
      exporting
*        CURSORLINE        =  1
         mark_flag         =  'X'
         mark_max          =  0
        start_col         =  15
        start_row         =  15
         textline1         =  'Das  POPUP  bietet'(c01)
         textline2         =  'folgende  Funktionalit 鏗:'(c02)
         titel             =  'Das  Beispiel  2'(b02)
```

```
    importing
      answer          =  antwort
    tables
      t_spopli        =  selectlist
    exceptions
      not_enough_answers = 1
      too_much_answers   = 2
      too_much_marks     = 3
      others          = 4.

endif.
```

在 SAP 中设计自动刷新的报表代码

```
REPORT zauto_refresh .
DATA: g_init_once,
      ok_code(20),
      g_ref_from_timer.

TYPES: BEGIN OF t_ekko,
  ebeln TYPE ekpo-ebeln,
  ebelp TYPE ekpo-ebelp,
  statu TYPE ekpo-statu,
  aedat TYPE ekpo-aedat,
  matnr TYPE ekpo-matnr,
  menge TYPE ekpo-menge,
  meins TYPE ekpo-meins,
  netpr TYPE ekpo-netpr,
  peinh TYPE ekpo-peinh,
END OF t_ekko.

DATA: it_ekko TYPE STANDARD TABLE OF t_ekko INITIAL SIZE 0,
      wa_ekko TYPE t_ekko.


IF g_init_once <> 'X'.
```

```
  g_init_once = 'X'.
  CALL FUNCTION 'Z_ENQUE_SLEEP'
     STARTING NEW TASK 'WAIT'
     PERFORMING when_finished ON END OF TASK.

ENDIF.

WRITE:/ 'wait for 10 sec....'.

AT USER-COMMAND.
  CASE ok_code.
    WHEN 'FCT_R'.
      SELECT ebeln ebelp statu aedat matnr menge meins netpr peinh
       UP TO 10 ROWS
        FROM ekpo
        INTO TABLE it_ekko.
      WRITE:/ sy-uzeit. "Time
      LOOP AT it_ekko INTO wa_ekko.
        WRITE:/ wa_ekko-ebeln, wa_ekko-ebelp.
      ENDLOOP.
      sy-lsind = 0.
      IF g_ref_from_timer = 'X'.

        CALL FUNCTION 'Z_ENQUE_SLEEP'
          STARTING NEW TASK 'INFO'
          PERFORMING when_finished ON END OF TASK.

        g_ref_from_timer = ''.
      ENDIF.
  ENDCASE.


*---------------------------------------------------------------------*
*       FORM WHEN_FINISHED                                            *
*---------------------------------------------------------------------*
*       ........                                                      *
```

```
*---------------------------------------------------------------------*
*  -->  TASKNAME                                                      *
*---------------------------------------------------------------------*
FORM when_finished USING taskname.
  RECEIVE RESULTS FROM FUNCTION 'Z_ENQUE_SLEEP'.

  g_ref_from_timer = 'X'.

* Trigger an event to run the at user-command
  SET USER-COMMAND 'FCT_R'.
  ok_code = 'FCT_R'.
  sy-ucomm = 'FCT_R'.

ENDFORM.                    " WHEN_FINISHED




FUNCTION Z_ENQUE_SLEEP.
*"----------------------------------------------------------------------
*"*"Local interface:
*"----------------------------------------------------------------------
wait up to 10 seconds.

*CALL FUNCTION 'ENQUE_SLEEP'
*     EXPORTING
*        SECONDS = 1.

ENDFUNCTION.
```

如何在 SAP 的 Screen 中编写 List 报表

1、相关命令

LEAVE TO LIST-PROCESSING [AND RETURN TO SCREEN <nnnn>].

LEAVE LIST-PROCESSING.

2、使用说明

3、推荐设计思路

设计一个空屏幕，在需要调用 list 的屏幕的逻辑流中使用 CALL SCREEN 调用空屏幕。空屏幕的 next screen 设置为 0，不需要 PAI，只需要在 PBO 中设计一个 Module，这个 module 的具体步骤如下：

1.首先调用 LEAVE TO LIST-PROCESSING AND RETURN TO SCREEN 0.

2.为 list 设置 GUI status ;如：设置 SPACE 或者为自己定义的.

3.使用下列语句使空屏幕不输出:

SUPPRESS DIALOG.

or

LEAVE SCREEN.

4.进行 List 输出和处理.

4、示例

```
REPORT demo_leave_to_list_processing .
TABLES sdyn_conn.
DATA: wa_spfli TYPE spfli,
    flightdate TYPE sflight-fldate.

CALL SCREEN 100.

MODULE status_0100 OUTPUT.
 SET PF-STATUS 'SCREEN_100'.
ENDMODULE.

MODULE cancel INPUT.
 LEAVE PROGRAM.
ENDMODULE.

MODULE user_command_0100.
 CALL SCREEN 500.
 SET SCREEN 100.
ENDMODULE.

MODULE call_list_500 OUTPUT.
 LEAVE TO LIST-PROCESSING AND RETURN TO SCREEN 0.
 SET PF-STATUS space.
```

```
  SUPPRESS DIALOG.
  SELECT carrid connid cityfrom cityto
    FROM spfli
    INTO CORRESPONDING FIELDS OF wa_spfli
    WHERE carrid = sdyn_conn-carrid.
    WRITE: / wa_spfli-carrid, wa_spfli-connid,
             wa_spfli-cityfrom, wa_spfli-cityto.
    HIDE: wa_spfli-carrid, wa_spfli-connid.
  ENDSELECT.
  CLEAR: wa_spfli-carrid.
ENDMODULE.

TOP-OF-PAGE.
  WRITE text-001 COLOR COL_HEADING.
  ULINE.

TOP-OF-PAGE DURING LINE-SELECTION.
  WRITE sy-lisel COLOR COL_HEADING.
  ULINE.

AT LINE-SELECTION.
  CHECK not wa_spfli-carrid is initial.
  SELECT fldate
    FROM sflight
    INTO flightdate
    WHERE carrid = wa_spfli-carrid AND
          connid = wa_spfli-connid.
      WRITE / flightdate.
  ENDSELECT.
  CLEAR: wa_spfli-carrid.
```

该示例屏幕 100 只包含一个输入字段（SDYN_CONN－CARRID ），在屏幕 100 的逻辑流中会调用 list 处理.屏幕 100 的逻辑流如下：

```
PROCESS BEFORE OUTPUT.
  MODULE STATUS_0100.
```

```
PROCESS AFTER INPUT.
  MODULE CANCEL AT EXIT-COMMAND.
  MODULE USER_COMMAND_0100.
```

在屏幕 100 的 PAI 的 module USER_COMMAND_100 将使用 CALL SCREEN 调用屏幕 500。屏幕 500 的逻辑流中封装了所有 List 处理，屏幕 500 的逻辑流如下:

```
PROCESS BEFORE OUTPUT.
  MODULE CALL_LIST_500.

PROCESS AFTER INPUT.
```

屏幕 500 的 PBO 的 module CALL_LIST_500 定义了 list 处理代码。另外由于屏幕 500 的屏幕流设置为 0，该程序从 List 处理返回时到屏幕 100 的 USER_COMMAND_100 的调用 CALL SCREEN 500 的语句之后.

该程序会在 list 处理时显示一个明细 List，具体代码参见 AT LINE-SELECTION、TOP-OF-PAGE 和 TOP-OF-PAGE DURING LINE-SELECTION 事件代码.

如何访问 SAP 的 Domain 的 Value Range

访问 Domain 的 Value Range 有两种方法：

1、直接访问表

dd07l 和 dd07T

```
  select * from dd07l
      where domname  = 'domname' and
          as4local  = active.
```

2、使用 SAP 的标准函数

```
    call function 'DD_DOMVALUES_GET'
        exporting
            domname       = p_domname
        importing
            rc            = l_subrc
        tables
            dd07v_tab     = l_dd07v
        exceptions
            wrong_textflag = 1
            others        = 2.
```

获取 SAP 标准函数的说明（含参数和异常）

```
report ZGET_FUNC_DES
TYPE-POOLS : slis.

PARAMETERS: p_func LIKE fupararef-funcname.    " Name of Function Module

DATA : BEGIN OF i_tab OCCURS 0,
        funcname     LIKE    fupararef-funcname,      " Name of Function Module
        paramtype    LIKE    fupararef-paramtype,     " Parameter type
        pposition    LIKE    fupararef-pposition,     " Internal Table, Current Line Index
        optional     LIKE    fupararef-optional,      " Optional parameters
        parameter    LIKE    fupararef-parameter,     " Parameter name
        defaultval   LIKE    fupararef-defaultval,    " Default value for import parameter
        structure    LIKE    fupararef-structure,     " Associated Type of an Interface Parameter
        stext        LIKE    funct-stext,             " Short text
    END OF i_tab.

DATA: BEGIN OF mtab_new_prog OCCURS 0,
        line(172) TYPE c,
      END OF mtab_new_prog.

DATA: funcdesc LIKE tftit-stext,    " Short text for function module
      mylen    TYPE i,
      myrc     TYPE i.

CONSTANTS: myhats(40) VALUE '^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^'.

TRANSLATE p_func TO UPPER CASE.

SELECT SINGLE
       tftit~stext    " Short text for function module
  INTO funcdesc
  FROM tftit
 WHERE tftit~funcname = p_func
   AND tftit~spras    = sy-langu.

TRANSLATE p_func TO LOWER CASE.
```

```
CONCATENATE  'CALL FUNCTION ' p_func '   " ' funcdesc INTO mtab_new_prog-line.
APPEND mtab_new_prog.

TRANSLATE p_func TO UPPER CASE.

SELECT
      fupararef~funcname      " Name of Function Module
      fupararef~paramtype     " Parameter type
      fupararef~pposition     " Internal Table, Current Line Index
      fupararef~optional      " Optional parameters
      fupararef~parameter     " Parameter name
      fupararef~defaultval    " Default value for import parameter
      fupararef~structure     " Associated Type of an Interface Parameter
      funct~stext             " Short text
  INTO TABLE i_tab
  FROM fupararef
      INNER JOIN funct
      ON  fupararef~funcname  = funct~funcname
      AND fupararef~parameter = funct~parameter
      AND funct~spras = sy-langu
 WHERE fupararef~funcname = p_func
 AND fupararef~r3state = 'A'
 ORDER BY fupararef~paramtype
          fupararef~pposition.


LOOP AT i_tab.
  AT NEW paramtype.
    CASE i_tab-paramtype.
      WHEN 'C'.
        MOVE '  CHANGING' TO mtab_new_prog-line.
      WHEN 'E'.
        MOVE '  IMPORTING' TO mtab_new_prog-line.
      WHEN 'I'.
        MOVE '  EXPORTING' TO mtab_new_prog-line.
      WHEN 'T'.
        MOVE '  TABLES' TO mtab_new_prog-line.
      WHEN 'X'.
```

```
    MOVE '  EXCEPTIONS' TO mtab_new_prog-line.
  ENDCASE.
  APPEND mtab_new_prog.
ENDAT.

IF i_tab-optional = 'X'.
  mtab_new_prog-line = '*^^^'.
ELSE.
  mtab_new_prog-line = '^^^^'.
ENDIF.

IF i_tab-paramtype = 'X'.
  MOVE i_tab-pposition TO i_tab-defaultval.
  CONDENSE i_tab-defaultval.
ELSE.
  TRANSLATE i_tab-parameter TO LOWER CASE.
ENDIF.

CONCATENATE mtab_new_prog-line i_tab-parameter '^=^' INTO mtab_new_prog-line.

IF i_tab-defaultval IS NOT INITIAL.
  CONCATENATE mtab_new_prog-line i_tab-defaultval  INTO mtab_new_prog-line.
ENDIF.

mylen = STRLEN( mtab_new_prog-line ).

IF mylen < 31.
  COMPUTE mylen = 31 - mylen.
ELSE.
  MOVE 1 TO mylen.
ENDIF.

TRANSLATE i_tab-structure TO LOWER CASE.

CONCATENATE mtab_new_prog-line myhats+0(mylen) ' " ' i_tab-structure INTO mtab_new_prog-line.

mylen = STRLEN( mtab_new_prog-line ).

IF mylen < 47.
  COMPUTE mylen = 47 - mylen.
```

```
  ELSE.
    MOVE 1 TO mylen.
  ENDIF.

  CONCATENATE mtab_new_prog-line myhats+0(mylen) ' ' i_tab-stext INTO mtab_new_prog-line.

  APPEND  mtab_new_prog.
ENDLOOP.    " LOOP AT I_TAB

CONCATENATE '   . " ' p_func INTO mtab_new_prog-line.
APPEND mtab_new_prog.

LOOP AT mtab_new_prog.
  TRANSLATE mtab_new_prog-line USING '^ '.
  MODIFY mtab_new_prog.

  IF mtab_new_prog = space.
    SKIP 1.
  ENDIF.
  WRITE: / mtab_new_prog.
ENDLOOP.       " LOOP AT MTAB_NEW_PROG

* Write the beautiful program code to ClipBoard from internal table
CALL METHOD cl_gui_frontend_services=>clipboard_export
  IMPORTING
    data = mtab_new_prog[]
  CHANGING
    rc   = myrc.
```

Download ABAP Spool to PDF（代码样例）

```
*** This program receive spool id and destination file name ***
DATA: it_pdf TYPE TABLE OF TLINE WITH HEADER LINE,
      gv_string TYPE string.



PARAMETERS: p_spool LIKE TSP01-RQIDENT,
            p_file LIKE RLGRAP-FILENAME.


AT SELECTION-SCREEN ON VALUE-REQUEST FOR p_file.
```

```
  CALL FUNCTION ' KD_GET_FILENAME_ON_F4'
    EXPORTING
*     PROGRAM_NAME        = SYST-REPID
*     DYNPRO_NUMBER       = SYST-DYNNR
*     FIELD_NAME          = '  '
      STATIC              = ' X'
      MASK                = ' ,*.txt,*.*'
    CHANGING
      FILE_NAME           = p_file
    EXCEPTIONS
      MASK_TOO_LONG       = 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.

START-OF-SELECTION.
  CALL FUNCTION ' CONVERT_ABAPSPOOLJOB_2_PDF'
    EXPORTING
      SRC_SPOOLID                    = p_spool
*     NO_DIALOG                      =
*     DST_DEVICE                     =
*     PDF_DESTINATION                =
*   IMPORTING
*     PDF_BYTECOUNT                  =
*     PDF_SPOOLID                    =
*     LIST_PAGECOUNT                 =
*     BTC_JOBNAME                    =
*     BTC_JOBCOUNT                   =
   TABLES
     PDF                           = it_pdf
   EXCEPTIONS
     ERR_NO_ABAP_SPOOLJOB          = 1
     ERR_NO_SPOOLJOB               = 2
```

```
    ERR_NO_PERMISSION              = 3
    ERR_CONV_NOT_POSSIBLE          = 4
    ERR_BAD_DESTDEVICE             = 5
    USER_CANCELLED                 = 6
    ERR_SPOOLERROR                 = 7
    ERR_TEMSEERROR                 = 8
    ERR_BTCJOB_OPEN_FAILED         = 9
    ERR_BTCJOB_SUBMIT_FAILED       = 10
    ERR_BTCJOB_CLOSE_FAILED        = 11
    OTHERS                         = 12
            .
  IF SY-SUBRC <> 0.
* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
*         WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
  ELSE.
    gv_string = p_file.
    CALL FUNCTION ' GUI_DOWNLOAD'
      EXPORTING
*       BIN_FILESIZE                    =
        FILENAME                        = gv_string
        FILETYPE                        = ' BIN'
*       APPEND                          = '  '
*       WRITE_FIELD_SEPARATOR           = '  '
*       HEADER                          = ' 00'
*       TRUNC_TRAILING_BLANKS           = '  '
*       WRITE_LF                        = ' X'
*       COL_SELECT                      = '  '
*       COL_SELECT_MASK                 = '  '
*       DAT_MODE                        = '  '
*       CONFIRM_OVERWRITE               = '  '
*       NO_AUTH_CHECK                   = '  '
*       CODEPAGE                        = '  '
*       IGNORE_CERR                     = ABAP_TRUE
*       REPLACEMENT                     = ' #'
*       WRITE_BOM                       = '  '
*       TRUNC_TRAILING_BLANKS_EOL       = ' X'
```

```
*       WK1_N_FORMAT                    = ’  ’
*       WK1_N_SIZE                      = ’  ’
*       WK1_T_FORMAT                    = ’  ’
*       WK1_T_SIZE                      = ’  ’
*     IMPORTING
*       FILELENGTH                      =
      TABLES
        DATA_TAB                        = it_pdf
*       FIELDNAMES                      =
     EXCEPTIONS
       FILE_WRITE_ERROR                 = 1
       NO_BATCH                         = 2
       GUI_REFUSE_FILETRANSFER          = 3
       INVALID_TYPE                     = 4
       NO_AUTHORITY                     = 5
       UNKNOWN_ERROR                    = 6
       HEADER_NOT_ALLOWED               = 7
       SEPARATOR_NOT_ALLOWED            = 8
       FILESIZE_NOT_ALLOWED             = 9
       HEADER_TOO_LONG                  = 10
       DP_ERROR_CREATE                  = 11
       DP_ERROR_SEND                    = 12
       DP_ERROR_WRITE                   = 13
       UNKNOWN_DP_ERROR                 = 14
       ACCESS_DENIED                    = 15
       DP_OUT_OF_MEMORY                 = 16
       DISK_FULL                        = 17
       DP_TIMEOUT                       = 18
       FILE_NOT_FOUND                   = 19
       DATAPROVIDER_EXCEPTION           = 20
       CONTROL_FLUSH_ERROR              = 21
       OTHERS                           = 22
              .
    IF SY-SUBRC <> 0.
* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
*         WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
```

```
    ENDIF.
  ENDIF.
```

ABAP 动态生成经典应用之 Table 数据 Upload 程序

开发说明：在 CBO 的程序开发过程中，需要为 Table 准备大量的测试数据，手动录入效率低，不专业，我们可以采用其他的高级编辑工具(例如：EXCEL, EditPlus)按照 Table 数据存储结构准备好数据，最后保存为 ASC 的文本文件，通过执行下面开发的程序，下面的程序执行的功能就是把编辑好的文本文件上的数据上载到 SAP 对应的 Table 中，小程序非常实用，也适用于我们大量更新 Table 数据时使用，ABAPer 们的手头必备啊！

开发技术：

1.文本文件上载技术

2.动态程序代码生成技术

3.ABAP 动态程序执行技术

4.TXT 文本文件对应用 Table 字段编辑技术

注意事项：

文件文件编辑过程中时间日期格式为 20060201 120000 表示 2006.02.01 12：00：00

文本文件中不能包含除字符，数字之外的其他特殊符号。假如 Table 中对应的字段数据为空时，在编辑数据时也对应该列为空。

程序代码：如下

```
**********************************************************************
* (Copyright @2006 Mysingle Digital System Co.Ltd.
* All Rights Reserved|Confidential)
* System Module      : ABAP CBO
* Program Description : Table Upload & Download
* Developer          : ZOU XIN
* Develop Date       : 2006.01.01
* Use Status         : Release 1.0
**********************************************************************
REPORT z_cbo_abap_01 NO STANDARD PAGE HEADING.

TABLES : dd03l,dd02t.
```

```
DATA: BEGIN OF tab OCCURS 10,
      fcode(4),
     END OF tab.
DATA : BEGIN OF itab OCCURS 0,
      chk(1) TYPE c,                  " check box
      tabname    LIKE dd03l-tabname,     " Table name
      fieldname  LIKE dd03l-fieldname,   " Feld Name
      position   LIKE dd03l-position,    " Table
      keyflag    LIKE dd03l-keyflag,     " Primary Key
      datatype   LIKE dd03l-datatype,    " Data Type
      intlen     LIKE dd03l-leng,
      END OF itab.
DATA itab1 LIKE itab OCCURS 0 WITH HEADER LINE.
DATA : BEGIN OF uptab OCCURS 0,
      text(72) TYPE c,
      END OF uptab.
DATA: BEGIN OF error_message,
      line1(72),
      line2(72),
      line3(72),
     END OF error_message.
DATA : cnt1(8) TYPE c,
      fcode LIKE sy-ucomm,
      changed LIKE s38e-buf_varied,
      prog(8) TYPE c,
      msg(120) TYPE c,
      er_include LIKE sy-repid,
      er_line    LIKE sy-index,
      er_off     LIKE sy-tabix,
      er_subrc   LIKE sy-subrc.

* 程序录入界面
PARAMETERS : tabname LIKE dd03l-tabname DEFAULT 'ZP023'.


START-OF-SELECTION.
```

```
*上载 Table 字段分析
  SELECT SINGLE * FROM dd02t WHERE tabname = tabname.
  SET PF-STATUS 'ZOUXIN'.
  SELECT * INTO CORRESPONDING FIELDS OF TABLE itab FROM dd03l
          WHERE tabname = tabname
          AND as4local = 'A'
          ORDER BY position.
  itab-chk = 'X'.
 MODIFY itab INDEX 1 TRANSPORTING chk .
*动态上载程序代码生成函数
  PERFORM generate_upload_code.
*ABAP 程序代码编辑器调用
  PERFORM edit_generator_code.

AT USER-COMMAND .
  IF sy-ucomm = 'EDIT'.
    PERFORM edit_generator_code.
  ELSEIF sy-ucomm = 'EXEC'.
    GENERATE SUBROUTINE POOL uptab NAME prog MESSAGE msg.
*程序代码语法检测
    PERFORM chcek_syntax_error.
*动态程序代码执行
    PERFORM dyn1 IN PROGRAM (prog).
  ENDIF.

*&---------------------------------------------------------------------*
*&      Form  generate_upload_code
*&---------------------------------------------------------------------*
FORM generate_upload_code.
  REFRESH uptab.
  uptab-text = 'REPORT ZUP19800526.'.
  APPEND uptab.
  CONCATENATE 'TABLES :' tabname '.' INTO uptab-text
  SEPARATED BY space.
  APPEND uptab.
  uptab-text = 'DATA : BEGIN OF UPTAB OCCURS 0,'.
```

```
APPEND uptab.
LOOP AT itab WHERE chk = ''.
  CLEAR uptab.
  CONCATENATE ' ' itab-fieldname '(' itab-intlen ')' ' TYPE C ,'
            INTO uptab-text+10(80).
  APPEND uptab.
ENDLOOP.
uptab-text+6(82) = 'END OF UPTAB.'.
APPEND uptab.
uptab-text = 'DATA : BEGIN OF RESULT OCCURS 0.'.
APPEND uptab.
CLEAR uptab.
CONCATENATE 'INCLUDE STRUCTURE' itab-tabname '.'
          INTO uptab-text+10(80) SEPARATED BY space.
APPEND uptab.
uptab-text = 'DATA : END OF RESULT.'.
APPEND uptab.
uptab-text = 'FORM DYN1.'.
APPEND uptab.
CLEAR uptab.
uptab-text+2(88) = 'CALL FUNCTION "UPLOAD"'.
APPEND uptab.
CLEAR uptab.
uptab-text+4(86) = 'EXPORTING'.
APPEND uptab.
CLEAR uptab.
uptab-text+6(84) = 'FILENAME = "C:\"'.
APPEND uptab.
uptab-text+6(84) = 'FILETYPE = "DAT"'.
APPEND uptab.
uptab-text+4(86) = 'TABLES'.
APPEND uptab.
CLEAR uptab.
uptab-text+6(84) = 'DATA_TAB = UPTAB.'.
APPEND uptab.
uptab-text+2(88) = 'LOOP AT UPTAB.'.
```

```
  APPEND uptab.
  CLEAR uptab.
  uptab-text+4(86) = 'CLEAR RESULT.'.
  APPEND uptab.
  uptab-text+4(86) = 'MOVE-CORRESPONDING UPTAB TO RESULT.'.
  APPEND uptab.
  uptab-text+4(86) = 'APPEND RESULT.'.
  APPEND uptab.
  uptab-text+2(88) = 'ENDLOOP.'.
  APPEND uptab.
  CLEAR uptab.
  CONCATENATE 'INSERT'  itab-tabname 'FROM TABLE RESULT.'
          INTO uptab-text+2(88) SEPARATED BY space.
  APPEND uptab.
  uptab-text = 'ENDFORM.' .
  APPEND uptab.
  CLEAR uptab.
ENDFORM.                    " generate_upload_code
*&---------------------------------------------------------------------*
*&      Form  edit_generator_code
*&---------------------------------------------------------------------*
FORM edit_generator_code.
  CALL FUNCTION 'EDITOR_APPLICATION'
      EXPORTING
          application = 'BF'
          display     = ' '
          name        = 'Source Code.....'
      IMPORTING
          fcode       = fcode
          changed     = changed
      TABLES
          content     = uptab.
  LOOP AT uptab.
    WRITE:/1 uptab-text.
  ENDLOOP.
ENDFORM.                          " PRINT_GENERATOR_CODE
```

```
*&---------------------------------------------------------------------*
*&      Form  chcek_syntax_error
*&---------------------------------------------------------------------*
FORM chcek_syntax_error.
  CALL FUNCTION 'EDITOR_SYNTAX_CHECK'
      EXPORTING
          i_global_check   = ' '
          i_global_program = ' '
          i_program        = 'ZUP19800526'
          i_r2_check       = ' '
          i_r2_destination = ' '
          i_trdir          = ' '
      IMPORTING
          o_error_include  = er_include
          o_error_line     = er_line
          o_error_message  = error_message
          o_error_offset   = er_off
          o_error_subrc    = er_subrc
      TABLES
          i_source         = uptab.
  IF er_subrc <> 0.
    er_line = er_line - 2.
    WRITE:/1 'Error Line : ',er_line.
    WRITE:/1 error_message-line1,error_message-line2,
           error_message-line3.
    STOP.
  ENDIF.
ENDFORM.                    " chcek_syntax_error
```

ABAP 动态生成经典应用之 Dynamic SQL Excute 程序

开发说明：在 SAP 的系统维护过程中，有时我们需要修改一些 Table 中的数据，可是很多 Table 又不能直接在 Tcode：SE16 中修改，使用的 SAP ID 又没有调试数据修改权限，这时我们应该怎么样修改数据呢？思路--> ABAP 程序中的 SQL 更新语句谁都有权限执行，只要我们能动态生成修改该 Table 字段的 ABAP CODE 动态执行即可！

开发技术：

1.SQL 代码编写技术

1.动态程序代码生成技术

2.ABAP 动态程序执行技术

注意事项：

SQL 语法一定要准确，修改条件准确，修改数据后不违法数据唯一性原则

程序代码：如下

```
**********************************************************************
* (Copyright @2006 Mysingle Digital System Co.Ltd.
* All Rights Reserved|Confidential)
* System Module  :  ABAP CBO
* Use Status  :  Release 1.0
**********************************************************************
REPORT z_cbo_abap_02 MESSAGE-ID zp NO STANDARD PAGE HEADING.

DATA : fcode LIKE sy-ucomm,
       changed LIKE s38e-buf_varied,
       save_tabix  LIKE sy-tabix,
       tabix_count TYPE i,
       select_key(10) TYPE c,
       etc(80)  TYPE c,
       update_flag  TYPE c,
       line_cnt     TYPE i,
       prog(8)      TYPE c,
       msg(120)     TYPE c,
       msg_text(72) TYPE c,
       confirm_flag TYPE c.
DATA: itab_sql LIKE abapsource OCCURS 0 WITH HEADER LINE,
      itab_prog LIKE abapsource OCCURS 0 WITH HEADER LINE.

START-OF-SELECTION.
*程序执行直接进入 ABAP 代码编辑器
  SET PF-STATUS 'PFSTA00'.
  WRITE: /1 'Edit Your SQL ................' COLOR 2.
```

```
AT USER-COMMAND.
*动态生成程序修改确认
  IF sy-ucomm = 'EDIT'.
    PERFORM editor_sql.
*动态生成程序执行
  ELSEIF sy-ucomm = 'EXEC' OR sy-ucomm = 'EDEX'.
    REFRESH itab_prog.
    CLEAR itab_prog.
    IF update_flag = 'X'.
      PERFORM exec_modify.
    ENDIF.
  ENDIF.
*&---------------------------------------------------------------------*
*&      Form  editor_sql
*&---------------------------------------------------------------------*
FORM editor_sql.
* CALL Editor
  CALL FUNCTION 'EDITOR_APPLICATION'
      EXPORTING
          application = 'BF'
          display     = ' '
          name        = '[Edit Your SQL......]'
      IMPORTING
          fcode       = fcode
          changed     = changed
      TABLES
          content     = itab_sql.
* Translate Code Upper
  LOOP AT itab_sql.
    save_tabix  = sy-tabix.
    tabix_count = tabix_count + 1.
    IF itab_sql-line = space OR itab_sql-line+(1) = '*'.
      DELETE itab_sql INDEX save_tabix.
    ENDIF.
    TRANSLATE itab_sql-line TO UPPER CASE.
```

```
    MODIFY itab_sql INDEX save_tabix.
  ENDLOOP.
* Parsing input SQL code
  LOOP AT itab_sql.
    IF sy-tabix = 1.
      SHIFT itab_sql-line LEFT DELETING LEADING space.
    ENDIF.
    save_tabix = sy-tabix + 1.
    SPLIT itab_sql-line AT space INTO select_key etc.
    IF select_key = 'SELECT'.
      MESSAGE i433 WITH 'Donot support select syntax!^~^'.
      stop.
* hehe~~Don't bother myself.
    ELSEIF select_key = 'DELETE' OR  select_key = 'UPDATE'
       OR  select_key = 'INSERT'.
      update_flag = 'X'.
    ENDIF.
  ENDLOOP.
* Display the SQL code
  sy-lsind = 0.
  DELETE itab_sql WHERE line IS initial.
  DESCRIBE TABLE itab_sql LINES line_cnt.
  IF line_cnt = 0.
    WRITE: /1 'Edit Your SQL ................' COLOR 2.
  ELSE.
    LOOP AT itab_sql.
      WRITE: /1 itab_sql-line.
    ENDLOOP.
  ENDIF.
  IF update_flag = 'Y'.
    EXIT.
  ENDIF.
ENDFORM.                    " editor_sql
*&---------------------------------------------------------------------*
*&      Form  exec_modify
*&---------------------------------------------------------------------*
```

```
FORM exec_modify.
  IF sy-ucomm = 'EXEC'.
* Modify dialog box
    CALL FUNCTION 'POPUP_TO_CONFIRM_STEP'
        EXPORTING
            textline1 = 'Do you want to really UPDATE?'
            titel     = 'Exit'
        IMPORTING
            answer    = confirm_flag.
    CASE confirm_flag.
      WHEN 'N'. EXIT. "NO
      WHEN 'A'. EXIT. "Cancel
      WHEN 'J'. "perform exec_sql_update. "YES
    ENDCASE.
  ENDIF.
* Modify Program ABAP Code.
  itab_prog-line = 'PROGRAM ZSQL19800526 MESSAGE-ID AT.'.
  APPEND itab_prog.
  itab_prog-line = 'DATA: COUNT TYPE I.'.
  APPEND itab_prog.
  itab_prog-line = 'FORM DYN2.'.
  APPEND itab_prog.
  itab_prog-line = 'EXEC SQL.'.
  APPEND itab_prog.
  LOOP AT itab_sql.
    itab_prog-line = itab_sql-line.
    APPEND itab_prog.
  ENDLOOP.
  itab_prog-line = 'ENDEXEC.'.
  APPEND itab_prog.
  itab_prog-line = 'MESSAGE I315 WITH "Performed" SY-DBCNT'.
  CONCATENATE itab_prog-line "" 'Records!^-^' "" '.'
            INTO itab_prog-line SEPARATED BY space.
  APPEND itab_prog.
  itab_prog-line = 'ENDFORM.'.
  APPEND itab_prog.
```

```
* Dynamic Program Display
  IF sy-ucomm = 'EDEX'.
    CALL FUNCTION 'EDITOR_APPLICATION'
       EXPORTING
          application = 'BF'
          display     = ' '
          name        = 'Modify Program...'
       IMPORTING
          fcode       = fcode
       TABLES
          content     = itab_prog.
    STOP.
  ENDIF.
* Dynamic Program Excuted
  GENERATE SUBROUTINE POOL itab_prog NAME prog
                        MESSAGE msg.
  IF sy-subrc <> 0.
    msg_text = msg+(80).
    WRITE: /1 msg_text.
    msg_text = msg+80(40).
    WRITE: /1 msg_text.
  ELSE.
    PERFORM dyn2 IN PROGRAM (prog).
  ENDIF.
ENDFORM.              " exec_modify

     Report to display 12 weeks forecast, Yesterday Stocks, Onhand Stocks, MTD Pull, MTD GR

*
* Report list last 12 Week Forecast, Yesterday, MTD PULL, Onhand Stock,
* MTD PO, MTD GR or NEXT MONTH PO
*
REPORT ZAI_WEEKLY LINE-SIZE 255 NO STANDARD PAGE HEADING
                  LINE-COUNT 065(001).

TABLES: MDKP,  "Header Data for MRP Document
        MDTB,  "MRP table
```

```
        MKPF,  "Header: Material Document
        MSEG,  "Document Segment: Material
        MARD,  "Storage Location Data for Material
        EKKO,  "Purchasing Document Header
        EKPO,  "Purchasing Document Item
        EKET,  "Scheduling Agreement Schedule Lines
        MARC,  "Plant Data for Material
        EINA,  "Purchasing Info Record: General Data
        MARA,  "General Material Data
        PBIM,  "Independent requirements for material
        PBED.  "Independent requirements data

DATA: BEGIN OF INT_MRP OCCURS 100,
       MATNR(18) TYPE C,
       MENGE    TYPE P DECIMALS 0,
       MTDPULL  TYPE P DECIMALS 0,
       ONHAND   TYPE P DECIMALS 0,
       MTDGR    TYPE P DECIMALS 0,
       MTDPOORD TYPE P DECIMALS 0,
       MTDPODEL TYPE P DECIMALS 0,
       NEXPOORD TYPE P DECIMALS 0,
       NEXPODEL TYPE P DECIMALS 0,
       PLNMG01  TYPE P DECIMALS 0,
       PLNMG02  TYPE P DECIMALS 0,
       PLNMG03  TYPE P DECIMALS 0,
       PLNMG04  TYPE P DECIMALS 0,
       PLNMG05  TYPE P DECIMALS 0,
       PLNMG06  TYPE P DECIMALS 0,
       PLNMG07  TYPE P DECIMALS 0,
       PLNMG08  TYPE P DECIMALS 0,
       PLNMG09  TYPE P DECIMALS 0,
       PLNMG10  TYPE P DECIMALS 0,
       PLNMG11  TYPE P DECIMALS 0,
       PLNMG12  TYPE P DECIMALS 0,
      END   OF INT_MRP.
```

```
DATA: BEGIN OF INT_DATE,
       DATE01(12)  TYPE C,
       DATE02(12)  TYPE C,
       DATE03(12)  TYPE C,
       DATE04(12)  TYPE C,
       DATE05(12)  TYPE C,
       DATE06(12)  TYPE C,
       DATE07(12)  TYPE C,
       DATE08(12)  TYPE C,
       DATE09(12)  TYPE C,
       DATE10(12)  TYPE C,
       DATE11(12)  TYPE C,
       DATE12(12)  TYPE C,
      END OF INT_DATE.

DATA: BEGIN OF MDTBX OCCURS 0.
        INCLUDE STRUCTURE MDTB.
DATA: END   OF MDTBX.

DATA: NO01(2)  TYPE N,
      FN01(20),
      FN02(20),
      X_MONTH01(20),
      X_MONTH02(20),
      X_MONTH03(20),
      X_MONTH04(20),
      X_MONTH05(20),
      X_MONTH06(20),
      X_MONTH07(20),
      X_MONTH08(20),
      X_MONTH09(20),
      X_MONTH10(20),
      X_MONTH11(20),
      X_MONTH12(20).
FIELD-SYMBOLS: <FS1>, <FS2>.
```

```
DATA: FDATE LIKE SY-DATUM,
      LDATE LIKE SY-DATUM,
      XDATE LIKE SY-DATUM.

DATA: BEGIN OF INT_PBED,
        BDZEI    TYPE PBED-BDZEI,
        PLNMG    TYPE PBED-PLNMG,
        PDATU    TYPE PBED-PDATU,
      END OF INT_PBED,
      ITAB_PBED LIKE TABLE OF INT_PBED.

DATA: BEGIN OF INT_MKPF,
        VGART    TYPE MKPF-VGART,
        BUDAT    TYPE MKPF-BUDAT,
        MBLNR    TYPE MKPF-MBLNR,
        MJAHR    TYPE MKPF-MJAHR,
      END OF INT_MKPF,
      ITAB_MKPF LIKE TABLE OF INT_MKPF.

DATA: BEGIN OF INT_MSEG,
        MBLNR    TYPE MSEG-MBLNR,
        MJAHR    TYPE MSEG-MJAHR,
        MATNR    TYPE MSEG-MATNR,
        MENGE    TYPE MSEG-MENGE,
        DMBTR    TYPE MSEG-DMBTR,
        SHKZG    TYPE MSEG-SHKZG,
      END OF INT_MSEG,
      ITAB_MSEG LIKE TABLE OF INT_MSEG.

DATA: BEGIN OF INT_MARD,
        MATNR    TYPE MARD-MATNR,
        DISKZ    TYPE MARD-DISKZ,
        LABST    TYPE MARD-LABST,
      END OF INT_MARD,
      ITAB_MARD LIKE TABLE OF INT_MARD.
```

```
DATA: BEGIN OF INT_EKKO,
        BUKRS    TYPE EKKO-BUKRS,
        EBELN    TYPE EKKO-EBELN,
      END OF INT_EKKO,
      ITAB_EKKO LIKE TABLE OF INT_EKKO.

DATA: BEGIN OF INT_EKPO,
        EBELN    TYPE EKPO-EBELN,
        EBELP    TYPE EKPO-EBELP,
        MATNR    TYPE EKPO-MATNR,
      END OF INT_EKPO,
      ITAB_EKPO LIKE TABLE OF INT_EKPO.

SELECT-OPTIONS: X_MATNR  FOR MDKP-MATNR, "Material number
                X_WERKS  FOR MDKP-PLWRK, "Plant
                X_EKGRP  FOR MARC-EKGRP, "Purchasing group
                X_BESKZ  FOR MARC-BESKZ, "Procurement Type
                X_SPART  FOR MARA-SPART, "Procurement Type
                X_LIFNR  FOR EINA-LIFNR, "Vendor's account number
                X_BUDAT  FOR MDTB-DAT00, "Posting Date
                X_BUDAT2 FOR MDTB-DAT00.

START-OF-SELECTION.
  PERFORM DATA_ONHAND.
  PERFORM DATA_MRP.
  PERFORM YESTERDAY_PULL.
  PERFORM DATA_MTD_PULL.
  PERFORM DATA_MTD_GR.
  PERFORM DATA_MTD_PO.
* PERFORM DATA_NEX_PO.
  PERFORM LOOP_INT_MRP.
END-OF-SELECTION.

FORM DATA_ONHAND.
  CLEAR ITAB_MARD.
  INT_MARD-DISKZ = ' '.
```

```
  APPEND INT_MARD TO ITAB_MARD.

  SELECT WERKS MATNR LGORT LABST
     INTO   CORRESPONDING FIELDS OF INT_MARD
     FROM   MARD FOR ALL ENTRIES IN ITAB_MARD
     WHERE DISKZ  = ITAB_MARD-DISKZ  "Storage location MRP indicator
       AND WERKS IN X_WERKS
       AND MATNR IN X_MATNR.

* Check Purchase Group, Procurement Type
    SELECT SINGLE * FROM MARC WHERE LVORM  = ' '
                                AND WERKS IN X_WERKS
                                AND MATNR  = INT_MARD-MATNR
                                AND EKGRP IN X_EKGRP
                                AND BESKZ IN X_BESKZ.
    IF SY-SUBRC <> 0. CONTINUE. ENDIF.

* Check Purchasing Info
    SELECT SINGLE * FROM EINA WHERE LOEKZ = ' '
                                 AND MATNR = INT_MARD-MATNR
                                 AND LIFNR IN X_LIFNR.
    IF SY-SUBRC <> 0. CONTINUE. ENDIF.

* Check Division
    SELECT SINGLE * FROM MARA WHERE LVORM = ''
                                 AND MATNR = INT_MARD-MATNR
                                 AND SPART IN X_SPART.
    IF SY-SUBRC <> 0. CONTINUE. ENDIF.

*      WRITE:/ INT_MARD-MATNR, INT_MARD-LABST.

       INT_MRP-ONHAND = INT_MARD-LABST.
       INT_MRP-MATNR  = INT_MARD-MATNR.
       COLLECT INT_MRP.
       CLEAR   INT_MRP.
  ENDSELECT.
```

```
ENDFORM.

FORM DATA_MRP.
* MDTB-DELKZ :-
* PP - IndReq
* SB - DepReq
* AR - OrdRes
LOOP AT INT_MRP.
SELECT SINGLE * FROM PBIM WHERE WERKS IN X_WERKS
                            AND MATNR =  INT_MRP-MATNR.
    INT_PBED-BDZEI = PBIM-BDZEI.
    APPEND INT_PBED TO ITAB_PBED.
    CLEAR NO01.
    SELECT BDZEI PLNMG PDATU
      INTO   CORRESPONDING FIELDS OF INT_PBED
      FROM   PBED FOR ALL ENTRIES IN ITAB_PBED
      WHERE  BDZEI = ITAB_PBED-BDZEI.
              ADD    1       TO NO01.
              CONCATENATE 'INT_MRP-PLNMG' NO01 INTO FN01.
              CONCATENATE 'INT_DATE-DATE' NO01 INTO FN02.
              ASSIGN (FN01)  TO <FS1>.
              ASSIGN (FN02)  TO <FS2>.

              ADD INT_PBED-PLNMG TO <FS1>.
              <FS2> = INT_PBED-PDATU.

              MODIFY  INT_MRP.

              IF NO01 => 12.
                 EXIT.
              ENDIF.
    ENDSELECT.
ENDLOOP.
ENDFORM.

FORM YESTERDAY_PULL.
```

```
INT_MKPF-BUDAT = SY-DATUM - 1.
APPEND INT_MKPF TO ITAB_MKPF.

SELECT BUDAT MBLNR MJAHR
  INTO   CORRESPONDING FIELDS OF INT_MKPF
  FROM   MKPF FOR ALL ENTRIES IN ITAB_MKPF
  WHERE  BUDAT = ITAB_MKPF-BUDAT.

*  WRITE: / INT_MKPF-BUDAT, INT_MKPF-MBLNR, INT_MKPF-MJAHR.

  CLEAR ITAB_MSEG.
  INT_MSEG-MBLNR = INT_MKPF-MBLNR.
  INT_MSEG-MJAHR = INT_MKPF-MJAHR.
  APPEND INT_MSEG TO ITAB_MSEG.
  SELECT MBLNR MJAHR MATNR MENGE DMBTR SHKZG
     INTO   CORRESPONDING FIELDS OF INT_MSEG
     FROM   MSEG FOR ALL ENTRIES IN ITAB_MSEG
     WHERE  MBLNR = ITAB_MSEG-MBLNR
       AND  MJAHR = ITAB_MSEG-MJAHR
       AND  MATNR IN X_MATNR
       AND  LGORT IN ('G1', 'G2', 'G3', 'G4')
       AND  BWART IN ('311', '312').

        IF INT_MSEG-SHKZG = 'S'.
            MULTIPLY INT_MSEG-MENGE BY -1.
            MULTIPLY INT_MSEG-DMBTR BY -1.
        ENDIF.

*       WRITE: / INT_MSEG-MBLNR, INT_MSEG-MATNR, INT_MSEG-MENGE.
        INT_MRP-MENGE = INT_MSEG-MENGE.
        INT_MRP-MATNR = INT_MSEG-MATNR.
        COLLECT INT_MRP.
        CLEAR   INT_MRP.
   ENDSELECT.
ENDSELECT.
```

```
ENDFORM.

FORM DATA_MTD_PULL.
INT_MKPF-VGART = ' '.
APPEND INT_MKPF TO ITAB_MKPF.
SELECT BUDAT MBLNR MJAHR
  INTO   CORRESPONDING FIELDS OF INT_MKPF
  FROM   MKPF FOR ALL ENTRIES IN ITAB_MKPF
  WHERE  VGART <> ITAB_MKPF-VGART
    AND  BUDAT IN X_BUDAT.

* MTD Pull
       SELECT * FROM MSEG WHERE WERKS IN X_WERKS
                            AND MBLNR  = INT_MKPF-MBLNR
                            AND MJAHR  = INT_MKPF-MJAHR
                            AND MATNR IN X_MATNR
                            AND LGORT IN ('G1', 'G2',
                                          'G3', 'G4')
                            AND BWART IN ('311', '312').
          IF MSEG-SHKZG = 'S'.
             MULTIPLY MSEG-MENGE BY -1.
             MULTIPLY MSEG-DMBTR BY -1.
          ENDIF.

          INT_MRP-MTDPULL = MSEG-MENGE.
          INT_MRP-MATNR   = MSEG-MATNR.

          COLLECT INT_MRP.
          CLEAR   INT_MRP.

       ENDSELECT.

ENDSELECT.
ENDFORM.

FORM DATA_MTD_GR.
```

```
INT_MKPF-VGART = ' '.
APPEND INT_MKPF TO ITAB_MKPF.
SELECT BUDAT MBLNR MJAHR
  INTO   CORRESPONDING FIELDS OF INT_MKPF
  FROM   MKPF FOR ALL ENTRIES IN ITAB_MKPF
  WHERE  VGART <> ITAB_MKPF-VGART
    AND  BUDAT IN X_BUDAT.

* MTD Goods Receipts
       SELECT * FROM MSEG WHERE WERKS IN X_WERKS
                            AND MBLNR  = INT_MKPF-MBLNR
                            AND MJAHR  = INT_MKPF-MJAHR
                            AND MATNR IN X_MATNR
                            AND LGORT IN ('G1')
                            AND BWART IN ('101', '102').
          IF MSEG-SHKZG = 'H'.
             MULTIPLY MSEG-MENGE BY -1.
             MULTIPLY MSEG-DMBTR BY -1.
          ENDIF.

          INT_MRP-MTDGR   = MSEG-MENGE.
          INT_MRP-MATNR   = MSEG-MATNR.

          COLLECT INT_MRP.
          CLEAR   INT_MRP.

       ENDSELECT.

ENDSELECT.
ENDFORM.

FORM DATA_MTD_PO.
CLEAR ITAB_EKKO.
INT_EKKO-BUKRS = '0010'.  "Company Code
APPEND INT_EKKO TO ITAB_EKKO.
SELECT BUKRS EBELN
```

```
  INTO   CORRESPONDING FIELDS OF INT_EKKO
  FROM   EKKO FOR ALL ENTRIES IN ITAB_EKKO
  WHERE  BUKRS = ITAB_EKKO-BUKRS
    AND  LOEKZ = ' '.

     CLEAR ITAB_EKPO.
     INT_EKPO-EBELN = INT_EKKO-EBELN.
     APPEND INT_EKPO TO ITAB_EKPO.
     SELECT EBELN EBELP MATNR
        INTO   CORRESPONDING FIELDS OF INT_EKPO
        FROM   EKPO FOR ALL ENTRIES IN ITAB_EKPO
        WHERE EBELN  = ITAB_EKPO-EBELN
          AND MATNR IN X_MATNR
          AND LOEKZ  = ' '
* "Delivery completed" indicator
          AND ELIKZ  = ' '.

         CLEAR EKET.
         SELECT SINGLE * FROM EKET WHERE EBELN  = INT_EKPO-EBELN
                                     AND EBELP  = INT_EKPO-EBELP
                                     AND EINDT IN X_BUDAT.
         IF SY-SUBRC = 0.
            INT_MRP-MATNR    = INT_EKPO-MATNR.
            INT_MRP-MTDPOORD = EKET-MENGE.
            INT_MRP-MTDPODEL = EKET-WEMNG.
            COLLECT INT_MRP.
            CLEAR   INT_MRP.
         ENDIF.
    ENDSELECT.
ENDSELECT.
ENDFORM.

FORM DATA_NEX_PO.
CLEAR ITAB_EKKO.
INT_EKKO-BUKRS = '0010'.  "Company Code
APPEND INT_EKKO TO ITAB_EKKO.
```

```
SELECT BUKRS EBELN
  INTO   CORRESPONDING FIELDS OF INT_EKKO
  FROM   EKKO FOR ALL ENTRIES IN ITAB_EKKO
  WHERE  BUKRS = ITAB_EKKO-BUKRS
    AND  LOEKZ = ' '.

    CLEAR ITAB_EKPO.
    INT_EKPO-EBELN = INT_EKKO-EBELN.
    APPEND INT_EKPO TO ITAB_EKPO.
    SELECT EBELN EBELP MATNR
       INTO   CORRESPONDING FIELDS OF INT_EKPO
       FROM   EKPO FOR ALL ENTRIES IN ITAB_EKPO
       WHERE EBELN  = ITAB_EKPO-EBELN
         AND MATNR IN X_MATNR
         AND LOEKZ  = ' '
* "Delivery completed" indicator
         AND ELIKZ  = ' '.

        CLEAR EKET.
        SELECT SINGLE * FROM EKET WHERE EBELN  = INT_EKPO-EBELN
                                    AND EBELP  = INT_EKPO-EBELP
                                    AND EINDT IN X_BUDAT2.
        IF SY-SUBRC = 0.
           INT_MRP-MATNR    = INT_EKPO-MATNR.
           INT_MRP-NEXPOORD = EKET-MENGE.
           INT_MRP-NEXPODEL = EKET-WEMNG.
           COLLECT INT_MRP.
           CLEAR   INT_MRP.
        ENDIF.
    ENDSELECT.
ENDSELECT.
ENDFORM.

FORM LOOP_INT_MRP.
   SORT INT_MRP.
   LOOP AT INT_MRP.
```

```
* Check Purchase Group, Procurement Type
    SELECT SINGLE * FROM MARC WHERE LVORM  = ' '
                              AND WERKS IN X_WERKS
                              AND MATNR  = INT_MRP-MATNR
                              AND EKGRP IN X_EKGRP
                              AND BESKZ IN X_BESKZ.
    IF SY-SUBRC <> 0. CONTINUE. ENDIF.

* Check Purchasing Info
    SELECT SINGLE * FROM EINA WHERE LOEKZ = ' '
                               AND MATNR = INT_MRP-MATNR
                               AND LIFNR IN X_LIFNR.
    IF SY-SUBRC <> 0. CONTINUE. ENDIF.

* Check Division
    SELECT SINGLE * FROM MARA WHERE LVORM = ''
                               AND MATNR = INT_MRP-MATNR
                               AND SPART IN X_SPART.
    IF SY-SUBRC <> 0. CONTINUE. ENDIF.

    WRITE: /  INT_MRP-MATNR   UNDER '      Material',
              INT_MRP-PLNMG01 UNDER X_MONTH01,
              INT_MRP-PLNMG02 UNDER X_MONTH02,
              INT_MRP-PLNMG03 UNDER X_MONTH03,
              INT_MRP-PLNMG04 UNDER X_MONTH04,
              INT_MRP-PLNMG05 UNDER X_MONTH05,
              INT_MRP-PLNMG06 UNDER X_MONTH06,
              INT_MRP-PLNMG07 UNDER X_MONTH07,
              INT_MRP-PLNMG08 UNDER X_MONTH08,
              INT_MRP-PLNMG09 UNDER X_MONTH09,
              INT_MRP-PLNMG10 UNDER X_MONTH10,
              INT_MRP-PLNMG11 UNDER X_MONTH11,
              INT_MRP-PLNMG12 UNDER X_MONTH12,
              INT_MRP-MENGE   UNDER 'Yesterday Pull',
              INT_MRP-MTDPULL UNDER '      MTD Pull',
              INT_MRP-ONHAND  UNDER '        On Hand'.
```

```
    INT_MRP-MTDPOORD = INT_MRP-MTDPOORD - INT_MRP-MTDPODEL.
*   INT_MRP-NEXPOORD = INT_MRP-NEXPOORD - INT_MRP-NEXPODEL.
    WRITE: /  INT_MRP-MTDPOORD UNDER '         MTD PO',
              INT_MRP-MTDGR    UNDER '         MTD GR'.
   ENDLOOP.
ENDFORM.

TOP-OF-PAGE.
FORMAT COLOR COL_TOTAL.
    WRITE: / SY-DATUM, SY-UZEIT, SY-REPID,
       110  'Material Consumption',
       200  SY-UNAME, SY-PAGNO.

    SKIP.
    CLEAR NO01.
    DO 12 TIMES.
       ADD    1        TO NO01.
       CONCATENATE 'INT_DATE-DATE' NO01 INTO FN01.
       CONCATENATE 'X_MONTH'       NO01 INTO FN02.
       ASSIGN (FN01)  TO <FS1>.
       ASSIGN (FN02)  TO <FS2>.
* Date conversion to 31 January 2003
*      CALL FUNCTION 'CONVERSION_EXIT_LDATE_OUTPUT'
*          EXPORTING
*             INPUT         = <FS1>
*          IMPORTING
*             OUTPUT        = <FS2>.
*       <FS2> = <FS1>.
        CONCATENATE <FS1>+6(2) <FS1>+4(2) <FS1>(4) INTO <FS2>.
    ENDDO.

    WRITE: /1 '      Material',
           20 X_MONTH01(8),
           35 X_MONTH02(8),
           50 X_MONTH03(8),
           65 X_MONTH04(8),
```

```
          80 X_MONTH05(8),
          95 X_MONTH06(8),
         110 X_MONTH07(8),
         125 X_MONTH08(8),
         140 X_MONTH09(8),
         155 X_MONTH10(8),
         170 X_MONTH11(8),
         185 X_MONTH12(8),
         200 'Yesterday Pull',
         215 '       MTD Pull',
         230 '        On Hand'.
  WRITE: /215 '         MTD PO',
         230 '         MTD GR'.

INITIALIZATION.
   CASE SY-DATUM+4(2).
        WHEN '01'.
          ADD 31 TO SY-DATUM.
        WHEN '02'.
          ADD 28 TO SY-DATUM.
        WHEN '03'.
          ADD 31 TO SY-DATUM.
        WHEN '04'.
          ADD 30 TO SY-DATUM.
        WHEN '05'.
          ADD 31 TO SY-DATUM.
        WHEN '06'.
          ADD 30 TO SY-DATUM.
        WHEN '07'.
          ADD 31 TO SY-DATUM.
        WHEN '08'.
          ADD 31 TO SY-DATUM.
        WHEN '09'.
          ADD 30 TO SY-DATUM.
        WHEN '10'.
          ADD 31 TO SY-DATUM.
```

```
      WHEN '11'.
        ADD 30 TO SY-DATUM.
      WHEN '12'.
        ADD 31 TO SY-DATUM.
      WHEN OTHERS.
        ADD 28 TO SY-DATUM.
  ENDCASE.
  LDATE = FDATE = SY-DATUM.
  LDATE+6(2) = '01'.
  SUBTRACT 1 FROM LDATE.
  FDATE = LDATE.
  FDATE+6(2) = '01'.
  MOVE:   FDATE         TO  X_BUDAT-LOW,
          LDATE         TO  X_BUDAT-HIGH.
  APPEND X_BUDAT.

  LDATE+6(2) = '28'.
  ADD  60 TO LDATE.
  LDATE+6(2) = '01'.
  SUBTRACT 1 FROM LDATE.
  FDATE = LDATE.
  FDATE+6(2) = '01'.
  MOVE:   FDATE         TO  X_BUDAT2-LOW,
          LDATE         TO  X_BUDAT2-HIGH.
  APPEND X_BUDAT2.
```

Mass Select and Print Material Master Changed History

4.6x

```
REPORT ZMMCHGHISTORY NO STANDARD PAGE HEADING
        LINE-SIZE 195 LINE-COUNT 60.

* Change doc listing
* Grouped into 3 chg types: 1. Part revision  2. Price change  3. Others
```

```
TABLES:
      CDHDR, CDPOS, MARA, MAKT, MARD.

FIELD-GROUPS: HEADER.

DATA: BEGIN OF CHGDOC OCCURS 50.
      INCLUDE STRUCTURE CDRED.
DATA: END OF CHGDOC.


DATA:
     CHGTYPE(1),
     PLANT(4),
     MATNR1 LIKE CHGDOC-OBJECTID.


SELECT-OPTIONS:
   XMATNR  FOR CDHDR-OBJECTID,    "Material
   XUDATE  FOR CDHDR-UDATE,       "Change Date
   XUNAME  FOR CDHDR-USERNAME,    "User Name
   XTCODE  FOR CDHDR-TCODE,       "Transaction Code
   XWERKS  FOR MARD-WERKS.        "Plants

SELECTION-SCREEN SKIP.

*Filter change type
SELECTION-SCREEN BEGIN OF BLOCK CHG0 WITH FRAME TITLE TEXT-001.
  PARAMETERS : XCHG1 AS CHECKBOX DEFAULT 'X',
               XCHG2 AS CHECKBOX DEFAULT 'X',
               XCHG3 AS CHECKBOX DEFAULT 'X'.
SELECTION-SCREEN END OF BLOCK CHG0.



START-OF-SELECTION.

INSERT:
      CHGDOC-OBJECTID        "Material
      CHGTYPE                "Change type
```

```
      PLANT
      CHGDOC-CHANGENR
      CHGDOC-USERNAME
      CHGDOC-UDATE
      CHGDOC-TCODE
      CHGDOC-TABNAME
      CHGDOC-TABKEY
      CHGDOC-CHNGIND
      CHGDOC-FNAME
      CHGDOC-FTEXT
      CHGDOC-TEXTART
      CHGDOC-OUTLEN
      CHGDOC-F_OLD
      CHGDOC-F_NEW

INTO HEADER.

SELECT * FROM MARA WHERE MATNR IN XMATNR.

  MATNR1 = MARA-MATNR.

  CALL FUNCTION 'CHANGEDOCUMENT_READ'
    EXPORTING
*       ARCHIVE_HANDLE          = 0
*       CHANGENUMBER            = ' '
*       DATE_OF_CHANGE          = '00000000'
        OBJECTCLASS             = 'MATERIAL'
        OBJECTID                = MATNR1
*       TABLEKEY                = ' '
*       TABLENAME               = ' '
*       TIME_OF_CHANGE          = '000000'
*       USERNAME                = ' '
*       LOCAL_TIME              = ' '
    TABLES
        EDITPOS                 = CHGDOC
    EXCEPTIONS
        NO_POSITION_FOUND       = 1
        WRONG_ACCESS_TO_ARCHIVE = 2
```

```
        TIME_ZONE_CONVERSION_ERROR = 3
        OTHERS                     = 4.

  LOOP AT CHGDOC.

    CHECK:  CHGDOC-UDATE    IN XUDATE,
            CHGDOC-USERNAME IN XUNAME,
            CHGDOC-TCODE    IN XTCODE.

*   Chg type: 1. Part revision, 2. Price change, 3. Others
    CASE CHGDOC-TCODE.
      WHEN 'MM01' OR 'MM02' OR 'MM03'.  CHGTYPE = '1'.
      WHEN 'MR21'.  CHGTYPE = '2'.
      WHEN OTHERS.  CHGTYPE = '3'.
    ENDCASE.

*   Filter chg type
    IF ( CHGTYPE = '1' AND XCHG1 <> 'X' ) OR
      ( CHGTYPE = '2' AND XCHG2 <> 'X' ) OR
      ( CHGTYPE = '3' AND XCHG3 <> 'X' ).
      CONTINUE.
    ENDIF.

*   Plant is a substring of tabkey
    PLANT = CHGDOC-TABKEY+21(4).

    IF NOT ( XWERKS IS INITIAL ) AND NOT ( PLANT IS INITIAL ).
      CHECK PLANT IN XWERKS.
    ENDIF.

    EXTRACT HEADER.

  ENDLOOP.

ENDSELECT.

END-OF-SELECTION.

SORT.
LOOP.
```

```
*  Material
  AT NEW CHGDOC-OBJECTID.
    SELECT SINGLE * FROM MAKT  WHERE MATNR = CHGDOC-OBJECTID.
    FORMAT INTENSIFIED ON.
    SKIP.  SKIP.
    WRITE:/' *** Material:', (18) CHGDOC-OBJECTID, MAKT-MAKTX.
  ENDAT.

*  Change type
  AT NEW CHGTYPE.
    FORMAT INTENSIFIED ON.
    SKIP.
    CASE CHGTYPE.
      WHEN '1'.   WRITE:/ '  **  Change type:  PARTS REVISION'.
      WHEN '2'.   WRITE:/ '  **  Change type:  PRICE CHANGE'.
      WHEN '3'.   WRITE:/ '  **  Change type:  OTHERS'.
    ENDCASE.
    SKIP.
  ENDAT.

  SHIFT CHGDOC-F_OLD LEFT DELETING LEADING SPACE.
  SHIFT CHGDOC-F_NEW LEFT DELETING LEADING SPACE.

  FORMAT INTENSIFIED OFF.
  WRITE:
   /    PLANT         UNDER 'Plant',
    (50) CHGDOC-FTEXT   UNDER 'Field',
    (45) CHGDOC-F_OLD   UNDER 'Old value',
    (45) CHGDOC-F_NEW   UNDER 'New value'.

  AT NEW CHGDOC-CHANGENR.
    FORMAT INTENSIFIED OFF.
    WRITE:
        CHGDOC-CHANGENR   UNDER 'Change doc',
        CHGDOC-TCODE      UNDER 'Tcod',
        CHGDOC-USERNAME   UNDER 'User name   ',
        CHGDOC-UDATE      UNDER 'Date    ' DD/MM/YY.
  ENDAT.
```

```
  AT END OF CHGDOC-OBJECTID.
    SKIP.
    ULINE.
    SKIP.
  ENDAT.
ENDLOOP.


TOP-OF-PAGE.
WRITE: / SY-DATUM, SY-UZEIT,
   50 'ABC PTE LTD',
  100 'page', SY-PAGNO,
     / SY-REPID,
   48 'Change Documents Report',
  100 SY-UNAME.

SKIP.
ULINE.

WRITE:/3
      'Change doc',
      'Tcod',
      'User name   ',
      'Date    ',
      'Plant',
  (50) 'Field',
  (45) 'Old value',
  (45) 'New value'.

ULINE.

*** End of Program
```

Purchase Order History Mass Display

Instead of checking your Purchase Order History one at a time, you can now mass display or print them with t his customized abap report.

```
*
```

```
*  Mass display or print Purchase Order History
*
*  You can request report by :
*  1.  Change date
*  2.  User Name
*  3.  Purchase Order Number
*  4.  Vendor Code
*
* Written by : SAP Basis, ABAP Programming and Other IMG Stuff
*              http://www.sapsky.com
*
REPORT ZPOCHANGE LINE-SIZE 132 NO STANDARD PAGE HEADING
                 LINE-COUNT 065(001)
                 MESSAGE-ID VR.

TABLES: DD04T,
        CDHDR,
        CDPOS,
        DD03L,
        DD41V,
        T685T,
        VBPA,
        TPART,
        KONVC,
        EKKO.

SELECT-OPTIONS: XUDATE FOR CDHDR-UDATE,
                XNAME  FOR CDHDR-USERNAME,
                XEBELN FOR EKKO-EBELN,
                XLIFNR FOR EKKO-LIFNR.

SELECTION-SCREEN SKIP.
* TEXT-001 - Sorting Sequence
SELECTION-SCREEN BEGIN OF BLOCK BLK1 WITH FRAME TITLE TEXT-001.
PARAMETERS: SUDATE RADIOBUTTON GROUP R1,
            SNAME  RADIOBUTTON GROUP R1,
```

```
            SOBID  RADIOBUTTON GROUP R1.
SELECTION-SCREEN END OF BLOCK BLK1.

DATA: WFLAG,
      WCHANGENR LIKE CDHDR-CHANGENR.

DATA: INDTEXT(60) TYPE C.

DATA: BEGIN OF ICDHDR OCCURS 50.
        INCLUDE STRUCTURE CDHDR.
DATA: END OF ICDHDR.

DATA: BEGIN OF ICDSHW OCCURS 50.
        INCLUDE STRUCTURE CDSHW.
DATA: END OF ICDSHW.

DATA: BEGIN OF EKKEY,
        EBELN LIKE EKET-EBELN,
        EBELP LIKE EKET-EBELP,
        ETENR LIKE EKET-ETENR,

      END OF EKKEY.

DATA: BEGIN OF ITAB OCCURS 50,
        BEGIN OF EKKEY,
          EBELN LIKE EKET-EBELN,
          EBELP LIKE EKET-EBELP,

          ETENR LIKE EKET-ETENR,
        END OF EKKEY,
        CHANGENR LIKE CDHDR-CHANGENR,
        UDATE    LIKE CDHDR-UDATE,
        UTIME    LIKE CDHDR-UTIME,
        USERNAME LIKE CDHDR-USERNAME,
        CHNGIND  LIKE CDSHW-CHNGIND,
        FTEXT    LIKE CDSHW-FTEXT,
```

```
        OUTLEN   LIKE CDSHW-OUTLEN,
        F_OLD    LIKE CDSHW-F_OLD,
        F_NEW    LIKE CDSHW-F_NEW,
      END OF ITAB.

DATA: OLD_OBJECTID LIKE CDHDR-OBJECTID.

FIELD-SYMBOLS: <F_OLD>, <F_NEW>.

SELECT * FROM EKKO WHERE EBELN IN XEBELN AND
                         LIFNR IN XLIFNR.
  CLEAR CDHDR.
  CLEAR CDPOS.
  CDHDR-OBJECTCLAS = 'EINKBELEG'.
  CDHDR-OBJECTID   = EKKO-EBELN.
  PERFORM GETCHGDOCS.
ENDSELECT.

IF SUDATE = 'X'.
  SORT ITAB BY UDATE EKKEY-EBELN CHANGENR EKKEY-EBELP
               EKKEY-ETENR.
ELSEIF SNAME = 'X'.
  SORT ITAB BY USERNAME EKKEY-EBELN CHANGENR EKKEY-EBELP
               EKKEY-ETENR.
ELSE.
  SORT ITAB BY EKKEY-EBELN CHANGENR EKKEY-EBELP EKKEY-ETENR.
ENDIF.

LOOP AT ITAB.
  CLEAR: INDTEXT, EKKEY.
  CASE ITAB-CHNGIND.
    WHEN 'U'.
        INDTEXT(50) = ITAB-FTEXT.
        INDTEXT+51  = TEXT-020.
        CONDENSE INDTEXT.
    WHEN 'D'.
```

```
    INDTEXT = TEXT-021.
WHEN 'E'.
    INDTEXT(5) = ITAB-FTEXT.
    INDTEXT+51 = TEXT-021.
    CONDENSE INDTEXT.
  WHEN 'I'.
    INDTEXT = TEXT-022.
ENDCASE.
RESERVE 4 LINES.
IF WCHANGENR NE ITAB-CHANGENR.
  WCHANGENR = ITAB-CHANGENR.
  EKKEY = ITAB-EKKEY.
  WRITE:/ ITAB-UDATE UNDER 'Change Date',
          ITAB-UTIME UNDER 'Time',
          ITAB-USERNAME UNDER 'User Name',
          ITAB-EKKEY-EBELN UNDER 'PO No',
          ITAB-EKKEY-EBELP UNDER 'Item',
          ITAB-EKKEY-ETENR UNDER 'Sch No',
          INDTEXT     UNDER 'Changes'.
ELSEIF ITAB-EKKEY NE EKKEY.
  WRITE:/ ITAB-EKKEY-EBELP UNDER 'Item',
          ITAB-EKKEY-ETENR UNDER 'Sch No',
          INDTEXT     UNDER 'Changes'.

ENDIF.

CASE ITAB-CHNGIND.
  WHEN 'U'.
    ASSIGN ITAB-F_OLD(ITAB-OUTLEN) TO <F_OLD>.
    ASSIGN ITAB-F_NEW(ITAB-OUTLEN) TO <F_NEW>.
    WRITE: / TEXT-023  UNDER 'Changes',
             <F_OLD>.
    WRITE: / TEXT-024 UNDER 'Changes',
             <F_NEW>.
  WHEN 'E'.
    ASSIGN ITAB-F_OLD(ITAB-OUTLEN) TO <F_OLD>.
```

```
        WRITE: TEXT-023 UNDER 'Changes',
               <F_OLD>.
    ENDCASE.
    SKIP.
ENDLOOP.



TOP-OF-PAGE.
WRITE:/ SY-DATUM,SY-UZEIT,
       50 'P U R C H A S E  O R D E R  H I S T O R Y',
      120 'Page', SY-PAGNO.
WRITE: / SY-REPID,
         60 'Purchase Orders Changes'.
SKIP.
ULINE.
IF SUDATE = 'X'.
  WRITE:/001 'Change Date',
         014 'Time',
         024 'User Name',
         038 'PO No',
         050 'Item',
         057 'Sch No',
         065 'Changes'.
ELSEIF SOBID = 'X'.
  WRITE:/001 'PO No',
         013 'Item',
         020 'Sch No',
         028 'Change Date',
         041 'Time',
         051 'User Name',
         065 'Changes'.
ELSE.
  WRITE:/001 'User Name',
         015 'Change Date',
         028 'Time',
         038 'PO No',
```

```
        050 'Item',
        057 'Sch No',
        065 'Changes'.
ENDIF.
ULINE.

FORM GETCHGDOCS.
  CALL FUNCTION 'CHANGEDOCUMENT_READ_HEADERS'
       EXPORTING
            DATE_OF_CHANGE    = CDHDR-UDATE
            OBJECTCLASS       = CDHDR-OBJECTCLAS
            OBJECTID          = CDHDR-OBJECTID
            TIME_OF_CHANGE    = CDHDR-UTIME
            USERNAME          = CDHDR-USERNAME
       TABLES
            I_CDHDR           = ICDHDR
       EXCEPTIONS
            NO_POSITION_FOUND = 1
            OTHERS            = 2.

  CHECK SY-SUBRC EQ 0.
  DELETE ICDHDR WHERE CHANGE_IND EQ 'I'.
  CHECK NOT ICDHDR[] IS INITIAL.
  LOOP AT ICDHDR.
    CHECK ICDHDR-UDATE IN XUDATE.
    CHECK ICDHDR-USERNAME IN XNAME.
    CALL FUNCTION 'CHANGEDOCUMENT_READ_POSITIONS'
                 EXPORTING  CHANGENUMBER       = ICDHDR-CHANGENR
                 IMPORTING  HEADER             = CDHDR
                 TABLES     EDITPOS            = ICDSHW
                 EXCEPTIONS NO_POSITION_FOUND  = 1
                            OTHERS             = 2.
    CHECK SY-SUBRC EQ 0.
    LOOP AT ICDSHW.
      CHECK ICDSHW-TEXT_CASE EQ SPACE.
      MOVE-CORRESPONDING ICDSHW TO ITAB.
```

```
      MOVE-CORRESPONDING ICDHDR TO ITAB.
      MOVE ICDSHW-TABKEY+3 TO ITAB-EKKEY.
      APPEND ITAB.
    ENDLOOP.
  ENDLOOP.
ENDFORM.
*
* END OF PROGRAM
```

Questions for Bar Code Printing in SAP

[日期：2006-10-22]　　来源：sap-img　作者：sapsky　　[字体：大 中 小]

1. Can we print bar codes in SAP only from ZEBRA printers ?

2. I had read that bar code printing is enabled in SAP and only needs to configure device/printer for that. Does this mean that we can use our existing HP 2300 or like printers to print bar codes without any ZEBRA like printers and the printed bar codes are readable through any reader ?

3. Are there any specific steps to print bar codes from SAP (from within Smartforms) including data fetch and printer configurations.

4. What is the process or method for reading data in bar code form into sap (as far as I think that we can upload the txt file created from reader and upload it to SAP). Can we do this without uploading TXT file i.e. directly reading from bar code reader into SAP. In other means what are the normal ways to read bar codes data into SAP.

Furthermore, we have taken a zebra Z4M plus printer from one of vendor on trial basis but even after following the recommended steps as mentioned in the config. manual we are unable to print any thing on bar code printer from within SAP.

I do agree that we can print bar codes from excel or from outside SAP but we wanted to print it out from within SAP and we think SMARTFORM is an easier way as compare to SAPSCRIPT (although we don't have any idea of printing bar codes from SAPSCRIPTs).

About reading from bar code reader .... yes we know that we have to read the bar codes into a ASCII or text file but in that case we need to write a ABAP program to upload that information into SAP to d

o MIGO (in our case) and for that we also need to fill in certain other information into that text file afte r reading bar codes ....?? is there any other simpler way of doing this ???

We are able to print barcodes from smartforms. Doing this way, we print barcodes on laser printers.

We are also printing from sap to zebra printers using two ways:

1. Download the data to an excel sheet, then creating a macro that opens the printer port and sending the co mmands to the printer through this "file". All this is done via vb script provided with excel)..
2. The second way is creating a vbscript (an ascii file from sap) with the printer commands and then runing it using ws_execute.

You need a barcode reader to read the barcodes, and this scanner acts like a keyboard, it sends the data sca nned to the active field on screen. (which might be a notepad, word, excel or an input field or ...).

Back to your problem:

1. We're using Zebra 2746-e (Eltron) to print labels that have some barcodes in their design. There are severa l approaches to solve this problem. We've decided to work with the programming language of the printer (EPL-2), because we use the zebra printers from SAP or from other windows applications.

Our solution was to develope function module that creates an ascii file (a vbscript file) and then use the ws_e xecute to run wscript with this file. The vbscript just opens the port for output and sends a sequence of writelin e, each of them with a command to the printer. After all the commands were sent to the printer, we close the port . Just note that the port acts the same as a file.

This is an EXAMPLE of the visual basic script code;

```
Set fs=CreateObject("Scripting.FileSystemObject")
Rem send the output to COM1 port.
Set a = fs.CreateTextFile("COM1:",True)
Comilla = Chr(34)
a.writeline "O"
a.writeline "ZB"
a.writeline ""
a.writeline ""
a.writeline ""
a.writeline "N"
a.writeline ""
a.writeline ""
```

```
a.writeline "q800"
a.writeline "Q635,24+0"
a.writeline "R32,24"
a.writeline "S3"
a.writeline "D8"
a.writeline "ZT"
a.writeline ""
rem this is an example of barcode ean128-ucc
a.writeline "B126,429,0,1E,3,3,61,B,""01123456789012341005121 5"""
rem this is an example of barcode 3 of 9 rotated
a.writeline "B10,495,3,1,3,3,49,B,""01234567"""
a.writeline ""
a.writeline ""
a.writeline "P1"
a.writeline ""
a.Close
```

We've send this code to an ascii file or as part of a macro to be run within an excel sheet. We have succeed in both cases.

The zebra printer is created as local to the computer running the vbscript. We haven't been able to share the zebra printer so other people in the network can use it. And the zebra printer isn't installed as a SAP printer (we aren't using sap spooler to send jobs to the zebra). Another Note: zebra 2746 comes with a software bar-one (or something like that) that allows you to design the barcode label in a wysiwyg way

If my memory isn't failing, I think that I've read something about controlling zebra printers from smartforms in service.sap.com, but I haven't explored this possibility.

2. **From 4.6c on, you can use smartforms to print barcodes without buying any barcode.dll software nor hardware extention like Bardimm on any laser/inkjet printer (Please Note that I haven't mentioned Zebra printers here!). To do this, you have to create a smartstyle -> character format with the desired barcode font (defined within sap).** Then in the smartform, create a window, put the field and associate it the character format. That's all (I mean, that's all we do at least :-). I think, you have to consider the barcode specifications before sending the barcode value to the smartform (Just an example, if you're using 3 of 9, the code should start and end with an asterisk - '*' -) We're printing an interleaved 2 out of 5 barcode in our invoices due to a legal requirement, and we did it this way.

3. If you have a barcode scanner, then you should not need reading the barcode into an ascii file to get the data read in an standard or custom screen field. You can read it directly to the field you want. (unless... you h

ave complex data coded in the barcode - for example if you're using an ean-ucc 128 compliant code and you're sending several fields in a single code ... In this case, an interface is almost mandatory because you must interpret the data fields according to the ucc standard, split the code into several fields .... and .... pure programming logic).

To put it clear: if you have to read, for example, a barcode that holds the legal number of an invoice using a barcode scanner and this number should be sent to migo-> bktxt then you don't need an interface. The scanner itself acts like a fast operator entering the characters using a keyboard and filling in the field.

We're reading barcodes in several places (when we finish each pallet, when we receive an invoice, and so on. Each case is a different screen. We arent using an ascii file to read these barcodes. Furthermore, we read the invoice legal number into migo bktxt field (Head Text).

关于 SmartForm 和 ScriptForm 的输出格式设置说明

[日期：2006-11-05] 来源：sapsky 作者：sapsky [字体：大 中 小]

Syntax	说明
&field+<offset>&	对于字符变量设置从何位置显示数据,如果 offset 大于字符变量长度时,系统就不会显示任何数据
&field(<length>)&	设置输出长度.
&field(*)&	如果该字段类型是 abap 数据字典里定义的类型,系统将按照字典定义的长度设置输出长度
&field(S)&	禁止输出符号位
&field(<)&	符号位显示在数据的左边
&field(.<nat.number>)&	设置显示小数的位数
&field(E<nat.number>)&	设置为科学标示法
&field(T)&	禁止千分位的显示(适用于: DEC, CURR, INT 和 QUAN 几种数据类型).
&field(Z)&	禁止数字前导 0 的显示
&field(I)&	禁止显示空值
&field(K)&	禁止类型系统按数据字典定义的转换函数进行输出转换
&field(R)&	右对齐(只有在定义了输出长度时才有效)
&field(F<filler>)&	用<filler>指定的字符替换左边的空格.

&field(L)&	将日期转换为本地显示格式,使用 JDAT 指定的格式
&field(C)&	该设置效果和 ABAP 的 CONDENSE 语句相同.
/: SET COUNTRY country_key	设置按某个国家显示小数点,千位符和日期的格式
/: SET DATE MASK = 'date_mask'	设置日期显示格式 DD 天 (two digits) DDD 天名称(缩写) DDDD 天名称 (全称) MM 月 (two digits) MMM 日期名称 (缩写) MMMM 日期名称 (全称) YY 年(two digits) YYYY 年 (four digits) LD 天 (formatted as for the L option) LM 月 (formatted as for the L option) LY 年 (formatted as for the L option) 示例 /: SET DATE MASK = 'Foster City, MM.DD.YY' &DATE& -> Foster City, 03.01.97 &DATE(Z)& -> Foster City, 3.1.97 /: SET DATE MASK = 'MMMM DD, YYYY' &DATE& -> March 01, 1997 取消设置 /: SET DATE MASK = ‘’
/: SET TIME MASK = 'time_mask'	时间设置 HH hours (two digits) MM minutes (two digits) SS seconds (two digits) 假设当前时间是 10:08:12. &TIME& -> 10:08:12 /: SET TIME MASK = 'HH:MM' &TIME& -> 10:08 /: SET TIME MASK = 'HH hours MM minutes' &TIME& -> 10 hours 08 minutes &TIME(Z)& -> 10 hours 8 minutes 取消设置: /: SET TIME MASK = ' '

Internal Table in Smartform

[日期：2006-10-22] 来源：sap-img 作者：sapsky [字体：大 中 小]

Here is a sample program in which used two internal tables:

```
REPORT YPRINTPRG_SMARTFORM1 .
DATA : ITKNA1 LIKE KNA1,
```

```
        ITVBAK LIKE VBAK OCCURS 0 WITH HEADER LINE.
PARAMETERS : PKUNNR LIKE KNA1-KUNNR.

SELECT * FROM KNA1 INTO ITKNA1
  WHERE KUNNR = PKUNNR.
ENDSELECT.

SELECT * FROM VBAK
   INTO TABLE ITVBAK
  WHERE KUNNR = PKUNNR.

CALL FUNCTION '/1BCDWB/SF00000011' “THIS FUNCTION MODULE CALLS THE
                        SMART FORM WE WILL GET THIS AT MENU ENVIRONEMENT ”
  EXPORTING
  ITKNA1              = ITKNA1
  TABLES
    ITVBAK            = ITVBAK.

IN SMART FORM
FORM INERFACE---------IMPORT (TAB)
  Parameter name     Type assignment    Reference type     Default value
  ITKNA1             LIKE               KNA1

FORM INERFACE---------TABLES (TAB)
  ITVBAK             LIKE               VBAK

PAGES & WINDOWS----- MAIN WINDOW-----LOOP 1----DATA(TAB)
  ITVBAK       INTO          ITVBAK

PAGES & WINDOWS-------MAIN WINDOW------LOOP 1-----TEXT 3(EDITOR)
  &ITVBAK-VBELN&   &ITVBAK-ERDAT&  &ITVBAK-ERNAM&  &ITVBAK-NETWR&

PAGES & WINDOWS-------HEADER WINDOW-----TEXT 2(EDITOR)
  Customer No. &itkna1-kunnr&       CustomerName :&itkna1-name1&
```

Display a contents of a table on SmartForm with LOOP

There's a DDIC Table called "Ugyfel" containing 5 rows. I'd like simply to display all the rows on a SF's Main window.

Please follow this process to display the value from your table "Ugyfel"

1. Go with a transaction code : smartforms
2. Enter the form name like : ysmart_forms1
3. Create
4. Enter the Description for the form
5. From the left side window there will be a form interface to provide table .....
6. Go for tables option
7. ugyfel like ugyfel(ref.type)
8. Pages and window---> page1---> main window
9. Go to the form painter adjust the main window.
10. Select main window and right click --> go for create loop
11. Name: loop1, desc: display loop.
12. Internal table ktab into ktab.
13. select loop right click -> create a text
14. name : text1, desc: display text.
15. Go to change editor.
16. Write the mater what ever you want and if you want to display data from the table write the table fields as follows:

&ktab-<field1>& &ktab-<field2>&

save & activate then execute ,, scripts will generate a function module like : '/ibcdw/sf0000031' copy this function module and call in executable program...

For that

1. go with abap editor se38.
2. table: ugyfel.
3. parameters: test like ugyfel-<field1>.
4. data itab like ugyfel occurs 0 with header line.
5. select * from ugyfel into table itab where field1 = test1.
6. call function '/ibcdw/sf0000031'
7. tables

   ktab = itab.

Save and activate the program (^f 3).

Now run the program (f 8)

ALL THE BEST.

Smartforms FAQ Part Two

Smartforms output difference

Problem with Smartforms: in a certain form for two differently configured printers, there seem to be a difference in the output of characters per inch (the distance between characters which gives a layout problem - text in two lines instead of one.

It happens when the two printers having different Printer Controls' if you go to SPAD Menu (Spool Administrator Menu) you can see the difference in the Printer Control and if you make the Printer control setting for both the printers as same. then it will be ok. and also u have to check what is the device type used for both the output devices.

SmartForms Output to PDF

There is a way to download smartform in PDF format.

Please do the following:

1. Print the smartform to the spool.
2. Note the spool number.
3. Download a PDF file (Acrobat Reader) version of the spool by running Program RSTXPDFT4 and entering the

noted spool number.

SmartForm Doublesided printing question

Your customer wants your PO SmartForm to be able to print "Terms and Conditinos" on the back side of each page. They don't want to purchase pre-printed forms with the company's logo on the front and terms & conditions on the back. Now this presents an interesting problem.

Has anyone else ever had a request like this? If for example there was a 3 page PO to be printed, they want 3 pieces of paper, the front side of each to containe the PO information (page 1, 2, and 3) and the back side of each piece of paper to containg the static "Terms & Conditions" information.

Anyone have a clue how to force this out?

Easy - page FRONT lists page CONTACTS as next page and CONTACTS lists FRONT as next page. Since CONTACTS does not contain a MAIN window, it will print the contacts info and then continue on to FRONT for

the rest of the main items. Additionally, set print mode on FRONT to D (duplex) and set CONTACTS to 'blank' (for both resource name and print mode - this is the only way to get to the back of the page).

Transport Smart Forms

How does one transport SMARTFORM? SE01?

How do you make sure that both, the SMARTFORM & it's function module gets transported? Or does the FM with same name gets generated automatically in the transported client?

A smartform is transported no differently than any other object. if it is assigned to a development class that is atteched to a transport layer, it will be transported.
The definition is transported, and when called, the function module is regenerated.
This leads to an interetsing situation. On the new machine, it is very likely the function module name will be different than the name on the source system. Make sure, before you call the function module, you resolve the external name to the internal name using the 'SSF_FUNCTION_MODULE_NAME' function module.
Typically, generate the SF, then use the pattern to being in the interface. Then change the call function to use the name you get back from the above function module.

Smartforms: protect lines in main window.

How to protect lines in the main window from splitting between pages?

It was easy with SAPscript, but how to do it with SF's. For 4.7 version if you are using tables, there are two options for protection against line break:
- You can protect a line type against page break.
- You can protect several table lines against page break for output in the main area.

Protection against page break for line types
- Double-click on your table node and choose the Table tab page.
- Switch to the detail view by choosing the Details pushbutton.
- Set the Protection against page break checkbox in the table for the relevant line type. Table lines that use this line type are output on one page.

Protection against page break for several table lines
- Expand the main area of your table node in the navigation tree.
- Insert a file node for the table lines to be protected in the main area.
- If you have already created table lines in the main area, you can put the lines that you want to protect again page break under the file using Drag&Drop. Otherwise, create the table lines as subnodes of the file.

- Choose the Output Options tab page of the file node and set the Page Protection option. All table lines that are in the file with the Page Protection option set are output on one page.

In 4.6, Alternatively in a paragraph format use the Page protection attribute to determine whether or not to display a paragraph completely on one page. Mark it if you want to avoid that a paragraph is split up by a page break. If on the current page (only in the main window) there is not enough space left for the paragraph, the entire paragraph appears on the next page.

Smart forms Frequently Asked Questions

Forcing a page break within table loop

Create a loop around the table. Put a Command node before the table in the loop that forces a NEWPAGE on whatever condition you want. Then only loop through a subset of the internal table (based on the conditions in the Command node) of the elements in the Table node.

Font style and Font size

Goto Transaction SMARTSTYLES.
There you can create Paragraph formats etc just like in sapscript.

Then in your window under OUTPUT OPTIONS you include this SMARTSTYLE and use the Paragraph and character formats.

Line in Smartform

Either you can use a window that takes up the width of your page and only has a height of 1 mm.

Then you put a frame around it (in window output options).
Thus you have drawn a box but it looks like a line.

Or you can just draw "__" accross the page and play with the fonts so that it joins each UNDER_SCORE.

Difference between 'forminterface' and 'global definitions' in global settings of smart forms

The Difference is as follows.

To put it very simply:

Form Interface is where you declare what must be passed in and out of the smartform (in from the print program to the smartform and out from the smartform to the print program).

Global defs. is where you declare data to be used within the smartform on a global scope.

ie: anything you declare here can be used in any other node in the form.

Smartforms function module name

Once you have activated the smartform, go to the environment -> function module name. There you can get the name of funtion module name.

The key thing is the program that calls it. for instance, the invoice SMARTFORM LB_BIL_INVOICE is ran by the program RLB_INVOICE.

This program uses another FM to determine the name of the FM to use itself. The key thing is that when it calls this FM (using a variable to store the actual name), that the parameters match the paramters in your smartform.

Another thing to note is that the FM name will change wherever the SF is transported to.

So you need to use the FM to determine the name of the SF.

Here is the code that can be use to determine the internal name of the function module:

Code:

```
  if sf_label(1) <> '/'.    " need to resolve by name
    move sf_label to externalname.
    call function 'SSF_FUNCTION_MODULE_NAME'
        exporting
            formname          = externalname
        importing
            fm_name           = internalname
        exceptions
            no_form           = 1
            no_function_module = 2
            others            = 3.
    if sy-subrc <> 0.
      message 'e427'.
    endif.
    move internalname to sf_label.
  endif.
```

It checks to see if the sf_label starts with a '/', which is how the internal names start. if it does, the name has already been converted. If not, it calls the FM and converts the name.

You would then CALL FUNCTION sf_label.

[推荐]A Sample Program Calling Smartforms

You should use 'SSF_FUNCTION_MODULE_NAME' & call function fm_name in your program & not others.

```
*&---------------------------------------------------------------------*
*& Report  ZTACA_DRIVER_SMARTFORM                                      *
*&                                                                     *
*&---------------------------------------------------------------------*
*&                                                                     *
*&                                                                     *
*&---------------------------------------------------------------------*

REPORT  ZTACA_DRIVER_SMARTFORM                    .

Tables : sflight.
Data : fm_name TYPE rs38l_fnam.

*data : Begin of it_flttab occurs 0,
*        carrid type sflight-carrid,
*        connid type sflight-connid,
*        fldate type sflight-fldate,
*        seatsmax type sflight-seatsmax,
*        seatsocc type sflight-seatsocc,
*        End of it_flttab.

data : it_flttab like table  of sflight.
Data : g_salary type i .
*       it_flttab type standard table of ty_flt.
g_salary = 1000.
```

```
select carrid connid fldate seatsmax seatsocc from sflight into
corresponding fields of table it_flttab.

CALL FUNCTION 'SSF_FUNCTION_MODULE_NAME'
  EXPORTING
    formname                  = 'ZTACA_SMFORM2'
*   VARIANT                   = ' '
*   DIRECT_CALL               = ' '
 IMPORTING
   FM_NAME                   = fm_name
 EXCEPTIONS
   NO_FORM                   = 1
   NO_FUNCTION_MODULE        = 2
   OTHERS                    = 3
          .
IF sy-subrc <> 0.
 MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
         WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
ENDIF.

call function fm_name
  Exporting
     salary = g_salary
  TABLES
     it_flttab = it_flttab
  EXCEPTIONS
    FORMATTING_ERROR           = 1
    INTERNAL_ERROR             = 2
    SEND_ERROR                 = 3
    USER_CANCELED              = 4
    OTHERS                     = 5           .

 IF SY-SUBRC <> 0.
 MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
         WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
```

```
ENDIF.
```

Example Forms Available in Standard SAP R/3

SF_EXAMPLE_01

Simple example; invoice with table output of flight booking for one customer

SF_EXAMPLE_02

Similar to SF_EXAMPLE_01 but with subtotals

SF_EXAMPLE_03

Similar to SF_EXAMPLE_02, whereby several customers are selected in the application program; the form is called for each customer and all form outputs are included in an output request

SmartForms System Fields

Within a form you can use the field string SFSY with its system fields. During form processing the system replaces these fields with the corresponding values. The field values come from the SAP System or are results of the processing.

System fields of Smart Forms

&SFSY-DATE&

Displays the date. You determine the display format in the user master record.

&SFSY-TIME&

Displays the time of day in the form HH:MM:SS.

&SFSY-PAGE&

Inserts the number of the current print page into the text. You determine the format of the page number (for example, Arabic, numeric) in the page node.

&SFSY-FORMPAGES&

Displays the total number of pages for the currently processed form. This allows you to include texts such as' Page x of y' into your output.

&SFSY-JOBPAGES&

Contains the total page number of all forms in the currently processed print request.

&SFSY-WINDOWNAME&

Contains the name of the current window (string in the Window field)

&SFSY-PAGENAME&

Contains the name of the current page (string in the Page field)

&SFSY-PAGEBREAK&

Is set to 'X' after a page break (either automatic [Page 7] or command-controlled [Page 46])

&SFSY-MAINEND&

Is set as soon as processing of the main window on the current page ends

&SFSY-EXCEPTION&

Contains the name of the raised exception. You must trigger your own exceptions, which you defined in the form interface, using the user_exception macro (syntax: user_exception <exception name >).

Conversion of SAPSCRIPT to SMARTFORMS

SAP provides a conversion for SAPscript documents to SMARTforms.

This is basically a function module, called FB_MIGRATE_FORM. You can start this function module by hand (via SE37), or create a small ABAP which migrates all SAPscript forms automatically.

You can also do this one-by-one in transaction SMARTFORMS, under

Utilities -> Migrate SAPscript form.

You could also write a small batch program calling transaction SMARTFORMS and running the migration tool.

FAQ on Migrating SAPscript to SmartForms

Is it possible to migrate a SAPscript form to a Smart Form?

Smart Forms provides a migration tool for this purpose which migrates layout and texts of a SAPscript form to a Smart Form. It does not migrate SAPscript form logic of the print program. Using Smart Forms, this logic is described by the tree structure of the Form Builder. The effort involved in migrating it depends on the complexity of the print program.

Which Basis Release do I need to use SAP Smart Forms?

SAP Smart Forms is available as of R/3 Basis Release 4.6C.

I have heard that Smart Forms replaces SAPscript. What does "replace" mean?

It does not mean that SAPscript is removed from the Basis shipment. Even as of Basis Release 4.6C, SAPscri

pt remains part of the SAP standard and there are no plans to remove it. Since Smart Forms is currently, and will continue to be, the tool for form maintenance for mySAP.com solutions, our further development efforts will focus on Smart Forms, not on SAPscript.

Do we have to migrate all SAPscript forms to Smart Forms?

There is no point in migrating all SAPscript forms already in use. Since SAPscript can still be used and will be available in the future, there is no need to. If you plan to migrate a SAPscript form, it is recommended that you check whether benefit is worth the effort involved.

Difference with SMARTFORMS vs. SapScript (SE71)

The Following are the differences :-

a) Multiple page formats are possible in smartforms which is not the case in SAPScripts

b) It is possible to have a smartform without a main window .

c) Labels cannot be created in smartforms.

d) Routines can be written in smartforms tool.

e) Smartforms generates a function module when activated.

[推荐]A Simple Smartform Tutorial

SAP Smartforms can be used for creating and maintaining forms for mass printing in SAP Systems. The output medium for Smartforms support printer, fax, e-mail, or the Internet (by using the generated XML output).

According to SAP, you need neither have any programming knowledge nor use a Script language to adapt standard forms. However, basic ABAP programming skills are required only in special cases (for example, to call a function module you created or for complex and extensive conditions).

1. Create a new smartforms

    Transaction code **SMARTFORMS**

    Create new smartforms call **ZSMART**

2. Define looping process for internal table

    Pages and windows

- First Page -> Header Window (Cursor at First Page then click **Edit -> Node -> Create**)

  Here, you can specify your title and page numbering

  &SFSY-PAGE& (Page 1) of **&SFSY-FORMPAGES(Z4.0)&** (Total Page)
- Main windows -> TABLE -> DATA
- In the Loop section, tick Internal table and fill in
- **ITAB1** (table in ABAP SMARTFORM calling function) INTO **ITAB2**

3. Define table in smartforms

Global settings :

Form interface

Variable name	Type assignment	Reference type
ITAB1	TYPE	Table Structure

Global definitions

Variable name	Type assignment	Reference type
ITAB2	TYPE	Table Structure

4. To display the data in the form

Make used of the Table Painter and declare the Line Type in Tabstrips Table

e.g. HD_GEN for printing header details,

IT_GEN for printing data details.

You have to specify the Line Type in your Text elements in the Tabstrips Output options.

Tick the New Line and specify the Line Type for outputting the data.

Declare your output fields in Text elements

Tabstrips - Output Options

For different fonts use this Style : **IDWTCERTSTYLE**

For Quantity or Amout you can used this variable **&GS_ITAB-AMOUNT(12.2)&**

5. Calling SMARTFORMS from your ABAP program

```
REPORT ZSMARTFORM.

* Calling SMARTFORMS from your ABAP program.
* Collecting all the table data in your program, and pass once to SMARTFORMS
* SMARTFORMS
* Declare your table type in :-
* Global Settings -> Form Interface
* Global Definintions -> Global Data
* Main Window -> Table -> DATA
```

```
*
* Written by :  SAP Hints and Tips on Configuration and ABAP/4 Programming
*                http://sapr3.tripod.com
*

TABLES: MKPF.

DATA: FM_NAME TYPE RS38L_FNAM.

DATA: BEGIN OF INT_MKPF OCCURS 0.
      INCLUDE STRUCTURE MKPF.
DATA: END OF INT_MKPF.

SELECT-OPTIONS S_MBLNR FOR MKPF-MBLNR MEMORY ID 001.

SELECT * FROM MKPF WHERE MBLNR IN S_MBLNR.
  MOVE-CORRESPONDING MKPF TO INT_MKPF.
  APPEND INT_MKPF.

ENDSELECT.

* At the end of your program.
* Passing data to SMARTFORMS

call function 'SSF_FUNCTION_MODULE_NAME'
  exporting
   formname            = 'ZSMARTFORM'
*   VARIANT             = ' '
*   DIRECT_CALL           = ' '
  IMPORTING
   FM_NAME             = FM_NAME
  EXCEPTIONS
   NO_FORM             = 1
   NO_FUNCTION_MODULE      = 2
   OTHERS              = 3.

if sy-subrc <> 0.
  WRITE: / 'ERROR 1'.
* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
```

```
*        WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
endif.

call function FM_NAME
* EXPORTING
*   ARCHIVE_INDEX          =
*   ARCHIVE_INDEX_TAB      =
*   ARCHIVE_PARAMETERS     =
*   CONTROL_PARAMETERS     =
*   MAIL_APPL_OBJ          =
*   MAIL_RECIPIENT         =
*   MAIL_SENDER            =
*   OUTPUT_OPTIONS         =
*   USER_SETTINGS          = 'X'
* IMPORTING
*   DOCUMENT_OUTPUT_INFO   =
*   JOB_OUTPUT_INFO        =
*   JOB_OUTPUT_OPTIONS     =
  TABLES
   GS_MKPF                = INT_MKPF
  EXCEPTIONS
   FORMATTING_ERROR       = 1
   INTERNAL_ERROR         = 2
   SEND_ERROR             = 3
   USER_CANCELED          = 4
   OTHERS                 = 5.

if sy-subrc <> 0.
  MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
      WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
endif.
```

Additional Fonts for your SMARTFORMS

You can create additional fonts and style with transaction SMARTSTYLES

This can then be define in the paragraph and character formats, which you can then be assign to texts and fields in the Smart Form.

The character formats includes effects such as superscript, subscript, barcode and font attributes.

Advantages of SAP Smart Forms

SAP Smart Forms have the following advantages:

1. The adaption of forms is supported to a large extent by graphic tools for layout and logic, so that no programming knowledge is necessary (at least 90% of all adjustments). Therefore, power user forms can also make configurations for your business processes with data from an SAP system. Consultants are only required in special cases.

2. Displaying table structures (dynamic framing of texts)

3. Output of background graphics, for form design in particular the use of templates which were scanned.

4. Colored output of texts

5. User-friendly and integrated Form Painter for the graphical design of forms

6. Graphical Table Painter for drawing tables

7. Reusing Font and paragraph formats in forms (Smart Styles)

8. Data interface in XML format (XML for Smart Forms, in short XSF)

9. Form translation is supported by standard translation tools

10. Flexible reuse of text modules

11. HTML output of forms (Basis release 6.10)

12. Interactive Web forms with input fields, pushbuttons, radio buttons, etc. (Basis-Release 6.10)

Introduction to SAP SmartForms

What is SAP Smart Forms?

SAP Smart Forms is introduced in SAP Basis Release 4.6C as the tool for creating and maintaining forms.

SAP Smart Forms allow you to execute simple modifications to the form and in the form logic by using simple graphical tools; in 90% of all cases, this won't include any programming effort. Thus, a power user without any

programming knowledge can

configure forms with data from an SAP System for the relevant business processes.

To print a form, you need a program for data retrieval and a Smart Form that contains the entire from logic. As **data retrieval** and **form logic** are separated, you must only adapt the Smart Form if changes to the form logic are necessary. The **application program passes the data via a function module interface to the Smart Form**. When **activating** the Smart Form, the **system automatically generates a function module**. At runtime, the system processes this function module.

You can insert static and dynamic tables. This includes line feeds in individual table cells, **triggering events** for **table headings** and **subtotals**, and **sorting data** before output.

You can check individual nodes as well as the entire form and find any existing errors in the tree structure. The **data flow analysis checks** whether all fields (variables) have a defined value at the moment they are displayed.

SAP Smart Forms allow you to **include graphics**, which you can display either as part of the form or as background graphics. You use background graphics to copy the layout of an existing (scanned) form or to lend forms a company-specific look. During printout, you can suppress the background graphic, if desired.

SAP Smart Forms also support postage optimizing.

Also read SAP Note No. 168368 - Smart Forms: New form tool in Release 4.6C

What Transaction to start SAP Smart Forms?

Execute transaction **SMARTFORMS** to start SAP Smart Forms.

Key Benefits of SAP Smart Forms:

SAP Smart Forms allows you to reduce considerably the implementation costs of mySAP.com solutions since forms can be adjusted in minimum time.

You design a form using the graphical Form Painter and the graphical Table Painter. The form logic is represented by a hierarchy structure (tree structure) that consists of individual nodes, such as **nodes for global settings**, **nodes for texts**, **nodes for output tables**, or **nodes for graphics**.

To make changes, use **Drag & Drop**, **Copy & Paste**, and **select different attributes**.

These actions do not include writing of coding lines or using a Script language.

Using your form description maintained in the Form Builder, Smart Forms generates a function module that **encapsulates layout, content and form logic**. So you **do not need a group of function modules** to print a form, but only one.

For Web publishing, the system provides a **generated XML output** of the processed form.

Smart Forms provides a data stream called XML for Smart Forms (XSF) to allow the use of 3rd party printing tools. XSF passes form content from R/3 to an external product without passing any layout information about the Smart Form.

A Sample SAP Scripts Reports

```
CALL FUNCTION 'OPEN_FORM'
* EXPORTING
*   APPLICATION                    = 'TX'
*   ARCHIVE_INDEX                  =
*   ARCHIVE_PARAMS                 =
*   DEVICE                         = 'PRINTER'
*   DIALOG                         = 'X'
*   FORM                           = 'ZSCRIPT1'
*   LANGUAGE                       = SY-LANGU
*   OPTIONS                        =
*   MAIL_SENDER                    =
*   MAIL_RECIPIENT                 =
*   MAIL_APPL_OBJECT               =
*   RAW_DATA_INTERFACE             = '*'
*   SPONUMIV                       =
* IMPORTING
*   LANGUAGE                       =
*   NEW_ARCHIVE_PARAMS             =
*   RESULT                         =
 EXCEPTIONS
   CANCELED                       = 1
   DEVICE                         = 2
   FORM                           = 3
   OPTIONS                        = 4
```

```
   UNCLOSED                          = 5
   MAIL_OPTIONS                      = 6
   ARCHIVE_ERROR                     = 7
   INVALID_FAX_NUMBER                = 8
   MORE_PARAMS_NEEDED_IN_BATCH       = 9
   SPOOL_ERROR                       = 10
   CODEPAGE                          = 11
   OTHERS                            = 12
          .
IF SY-SUBRC <> 0.
 MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
         WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
ENDIF.

CALL FUNCTION 'START_FORM'
EXPORTING
*   ARCHIVE_INDEX          =
   FORM                   = 'ZFORM1'
*   LANGUAGE               = ' '
*   STARTPAGE              = 'X'
   PROGRAM                = 'ZSCRIPT1'
*   MAIL_APPL_OBJECT       =
* IMPORTING
*   LANGUAGE               =
* EXCEPTIONS
*   FORM                   = 1
*   FORMAT                 = 2
*   UNENDED                = 3
*   UNOPENED               = 4
*   UNUSED                 = 5
*   SPOOL_ERROR            = 6
*   CODEPAGE               = 7
*   OTHERS                 = 8
          .
IF SY-SUBRC <> 0.
* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
```

```
*         WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
ENDIF.

CALL FUNCTION 'WRITE_FORM'
 EXPORTING
*   ELEMENT                          = ' '
*   FUNCTION                         = 'SET'
*   TYPE                             = 'BODY'
   WINDOW                           = 'HEADER'
* IMPORTING
*   PENDING_LINES                    =
 EXCEPTIONS
   ELEMENT                          = 1
   FUNCTION                         = 2
   TYPE                             = 3
   UNOPENED                         = 4
   UNSTARTED                        = 5
   WINDOW                           = 6
   BAD_PAGEFORMAT_FOR_PRINT         = 7
   SPOOL_ERROR                      = 8
   OTHERS                           = 9
          .
IF SY-SUBRC <> 0.
write:/ 'ERROR IN HEADER'.

* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
*         WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
ENDIF.

CALL FUNCTION 'WRITE_FORM'
 EXPORTING
*   ELEMENT                          = ' '
*   FUNCTION                         = 'SET'
*   TYPE                             = 'BODY'
   WINDOW                           = 'MAIN'
* IMPORTING
```

```
*   PENDING_LINES                    =
 EXCEPTIONS
   ELEMENT                          = 1
   FUNCTION                         = 2
   TYPE                             = 3
   UNOPENED                         = 4
   UNSTARTED                        = 5
   WINDOW                           = 6
   BAD_PAGEFORMAT_FOR_PRINT         = 7
   SPOOL_ERROR                      = 8
   OTHERS                           = 9
          .
IF SY-SUBRC <> 0.
write:/ 'ERROR IN HEADER'.

* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
*         WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
ENDIF.

CALL FUNCTION 'WRITE_FORM'
 EXPORTING
*   ELEMENT                          = ' '
*   FUNCTION                         = 'SET'
*   TYPE                             = 'BODY'
   WINDOW                           = 'FOOTER'
* IMPORTING
*   PENDING_LINES                    =
 EXCEPTIONS
   ELEMENT                          = 1
   FUNCTION                         = 2
   TYPE                             = 3
   UNOPENED                         = 4
   UNSTARTED                        = 5
   WINDOW                           = 6
   BAD_PAGEFORMAT_FOR_PRINT         = 7
   SPOOL_ERROR                      = 8
```

```
   OTHERS                          = 9
          .
IF SY-SUBRC <> 0.
write:/ 'ERROR IN HEADER'.

* MESSAGE ID SY-MSGID TYPE SY-MSGTY NUMBER SY-MSGNO
*         WITH SY-MSGV1 SY-MSGV2 SY-MSGV3 SY-MSGV4.
ENDIF.

CALL FUNCTION 'END_FORM'
* IMPORTING
*   RESULT                          =
* EXCEPTIONS
*   UNOPENED                        = 1
*   BAD_PAGEFORMAT_FOR_PRINT        = 2
*   SPOOL_ERROR                     = 3
*   CODEPAGE                        = 4
*   OTHERS                          = 5
          .

CALL FUNCTION 'CLOSE_FORM'
* IMPORTING
*   RESULT                          =
*   RDI_RESULT                      =
* TABLES
*   OTFDATA                         =
* EXCEPTIONS
*   UNOPENED                        = 1
*   BAD_PAGEFORMAT_FOR_PRINT        = 2
*   SEND_ERROR                      = 3
*   SPOOL_ERROR                     = 4
*   CODEPAGE                        = 5
*   OTHERS                          = 6
          .
```

Can you explain the difference between

1.open_form and Start form

2.end_form and Close_form.

whether all 4 modules are required in the driver pgm .

Open_form => It assign the form and printer, It should be first.

Start_form => It start Writing mode. You can use write_form in loop to write more than one lines befor End_form.

End_form => It end writing mode of current page and will require to start again through Start_form.

Close_form=> it end the Form. After this you can not start again for created file.

Sample Sapscripts Label Printing Program

```
TABLES : ZPACK,ZTRN.

DATA: BEGIN OF ITAB OCCURS 0,
      ZPKSLIP_NO LIKE ZTRN-ZPKSLIP_NO,
      ZCARTON_NO LIKE ZPACK-ZCARTON_NO,
      END OF ITAB.

DATA MVAR(12) TYPE C.
DATA MCTR(6) TYPE C.

SELECTION-SCREEN BEGIN OF BLOCK B1 WITH FRAME TITLE TEXT-001.
SELECT-OPTIONS: ZSLIP FOR ZTRN-ZPKSLIP_NO NO-EXTENSION NO INTERVALS
OBLIGATORY default 6.
SELECTION-SCREEN END OF BLOCK B1.

SELECT * FROM ZPACK INTO CORRESPONDING FIELDS OF TABLE ITAB WHERE
ZPKSLIP_NO EQ ZSLIP-LOW .

CALL FUNCTION 'OPEN_FORM'
  EXPORTING
    FORM = 'ZTEST_RAJ'.

DO 4 TIMES.
```

```
  MCTR = 100000 + SY-INDEX.
  MCTR = MCTR+1(5).
  CONCATENATE '55C/06/' MCTR INTO MVAR.

  DO 80 TIMES.
    ITAB-ZPKSLIP_NO = MVAR.
    ITAB-ZCARTON_NO = SY-INDEX.
    APPEND ITAB.
    CLEAR ITAB.
  ENDDO.

ENDDO.

SORT ITAB BY ZPKSLIP_NO ZCARTON_NO.

CALL FUNCTION 'START_FORM'
  EXPORTING
    FORM = 'ZTEST_RAJ'.

LOOP AT ITAB.

  AT NEW ZPKSLIP_NO.

    CALL FUNCTION 'WRITE_FORM'
      EXPORTING
        ELEMENT = '101'
        WINDOW  = 'MAIN'.

  ENDAT.

  CALL FUNCTION 'WRITE_FORM'
    EXPORTING
      ELEMENT = '102'
      WINDOW  = 'MAIN'.

  AT END OF ZPKSLIP_NO.

    CALL FUNCTION 'END_FORM'.
```

```
    CALL FUNCTION 'START_FORM'
      EXPORTING
        FORM = 'ZTEST_RAJ'.

  ENDAT.

ENDLOOP.

CALL FUNCTION 'END_FORM'.

CALL FUNCTION 'CLOSE_FORM'.
```

In sap script write :

```
/E   101
P1   ,,&ITAB-ZPKSLIP_NO(R)&
P1
/E   102
P1   ,,&ITAB-ZCARTON_NO(R)&
```

FAQ for Sap Scripts

What is the difference between a script & a report ?

Script is a form which has a layout set as per the company standards and can be used for external use too. Generally reports are designed for internal use for in house users

What are the components/elements in sap script ?

Layout set and Print program and the layout set has windows in it.

Can you create a script with out a main window ?

No

How many main windows can be created for a script ?

99

How can we use tables in sap scripts?

We can access structures and the tables tat are updated during runtime. Else you have to pass the structure v alues to the table in the print program.

How to print a logo in a sap script?

Upolad in the R3 using Se78 and use the Include statement in the script.

When we need to modify standard scripts (eg:medruck) given by sap?

When the client goes for customization of the form

What is the use of transaction code NACE in sap scripts?

You can track the form and the print program used for that form

What is the table name that will contain all the script form names and print program names?

TNAPR

Can you assign your own form to a standard print program? how?

Yes. thru NACE

What is the use of PROTECT & ENDPROTECT?

Keeps the block of text in the same page.

How to add extra functionality to a standard print program with out touching the print program?

Thru subroutine programs

What is sub routine pool in sap script? when it is used?

Its an Abap prog of type sub routine pool, it is used for calculating certain variables, eg DUE date for an Invoi ce. You pass the values from the form thru ITCSY structure intot he prgram.

How to read text in sapscripts?

SO10

What is the difference between paragraph & character format?

Self explanatory definition

How to use a sapscript for multiple languages ?

(english, germany etc) Copy the script in each lang or you have an option to click 'TO all Languages'

How to download/upload sapscripts from & to your PC ?

SE78 or RSTXLDMC

What is the difference between scripts & smart forms?

Scripts are client-dependent but SF are client Independent

Sapscripts and abap programs are client dependent or not? Why?

Scripts are client dependent. / Reports are client Independent.

What is the transaction code for logo uploading?

SE78

What is the standard program used for uploading the logo to script?

RSTXLDMC FM to upload image in tiff format.

How can you send forms from one client to other?

SE71, Utilities -> Copy from client...

What does open_form, write_form, close_form do?

Again its self-explanatory

What is the diffrence between open_form and close_form?

open_form is used to open the form/initiate the form.

close_form is used to conclude the open_form.

How to convert a sapscript to smart form?

tcode SMARTFORMS, I think its menu Utilities you have an option.. Migrate Scripts to Smartforms.

How to send a smartform result through mail?

I think you have to configure the output type. Not sure..

How to select desired paper size in sapscript?

In Basic settings.

How to print the Page Nos in Forms. Every page I want to print 1 of 10 , 2 of 10 , 3 of 0 ...etc.

PAGE &PAGE& OF &SAPSCRIPT-FORMPAGES& *-- Nitin

How to debugg a script?

This can done in two ways:

In the form Utilities->debugger / RSTXDBUG FM for debugging script

The Procedure for debugging SAP script is:

Generally SAP script contains the Layout and corresponding print program.

First go to SE71 and enter ur script name. In the same screen go to Utilities->click on activate debugger optio n.

Now go to SE 38 and enter ur Print Program name and execute the program.

Now you can debug the script Page wise and window wise.

1. When do you modified MEDRUCK? (IF I SAID I HAVE WORKED ON SCRIPTS).

Generally, we modify existing sap scripts provided by SAP rather than creating one. Unless you have to do so mething new for your client like Labels or Packaging card, etc., MEDRUCK is the form for PO.

2. I want to know the procedure to create a purchase order using MEDRUCK.

You don't create a PO using MEDRUCK. MEDRUCK is the form used to print a PO that has been created.

3. What are the usual changes to be done on MEDRUCK?

Goto SE71, there is an option in Utilities as COPY ffrom Source client (000). Copy the from MEDRUCK into a Zname form. The common changes wud b inserting a logo, using Std text for Terms and Conditions, alignmen t of windows as per client requirement, get xtra data if client is asking for somethign more.

4. How can I access my data from DB to SCRIPTS?

There are structures used in Scripts which hold the data entered by the user. These structures are used to get data from Database.

5. Please send me the one examples in full length.

Look at MEDRUCK form and it would have a print program. you can find in tcode NACE.

SapScript Question

1) How do you backup script layout sets?

2) What type of variables normally used in script to o/p data?

3) How do you use tabsets in layouts?

1) Use this Std program RSTXSCRP.

   1) First Export to Presentation file(.doc).

   2) Whenever you need that Export into SAP.

2) Normally we call them as Program symbols. Those are defined in Driver program. We can use in Script as for exp. &itab-matnr&

Other variables ---System symbols : ex &page&

---Std symbols :

---Text symbols :We define them in script editor itself.

Ex : /: Define &mysymbol& = 'XX'

3) We can control the tab feed in a paragraph with tab positions. The tab stops us define in the paragraph format replace the tab spacing we defined in the header data of the form. However, this depends on the extent to which we have defined tab stops in the paragraph format. If there are fewer tabs in the paragraph formats than in the header data, the tab stops of the header data are used for the rest of the line.

SAPscripts Tips by : Venkat O

Q: We get the total number of pages as expected by using 'SAPSCRIPT-FORMPAGES' in a duplex layout. In our case duplex case is always 'Terms & Conditions'. We do not want the number of pages as in duplex printing. What is the best possible solution?

A: On the Terms & Conditions page, Change the Page counter mode to 'HOLD' to keep the page counter from incrementing when you print the Term & Conditions.

Q: Can I Print a logo on an Invoice?

A: Save a Logo using Paintshop Pro or Corel Draw as Tiff file. Use RSTXLDMC to convert the logo to standard text in SapScript. When the program is executed, the path and file name have to be correctly specified.

Process could be like the following:

Run RSTXLDMC

Enter file name C:\MAIL\COMPLOGO.TIF

Resolution for Tiff file

Absolute X-position

Absolute Y-position

Absolute positioning

Reserved height

Shift to right

UOM = CM

Text title

Line width for text = 132

Text name ZHEX-MACRO-COMPLOGO

Text ID ST

Text language = E

Postscript scaling

Width & Height according to PS scaling

Number of Tiff gray levels (2,4,9) 2

Then Create a new window 'COMP' with attributes;

Window COMP description Company Logo

Window type CONST

Left margin 7.00 CH window width 10.00 CH

Upper margin LN window height 8.00 LN

Finally in the text element , mention

/: INCLUDE 'ZHEX-MACRO-COMPLOGO' OBJECT TEXT ID ST LANGUAGE 'E'.

Please note that if object name is not indicated as 'ZHEX...', the logo may not be printed!

You will not be able to see the logo in a test print. The same will be printed in actual printout.

If you are using two logos in the same layout, the names of the logos should be unique. Say 'ZHEX-MACRO-LOGO1' and 'ZHEX-MACRO-LOGO2'. Else all the information will be overwritten.

If the logo is not EXACTLY TIFF 6.0, the same will not be printed.

See OSS notes 5995, 18045, 39031 for some inputs.

Details information about SAP Barcodes

What I need to do to print a barcode in sapscript?

A barcode solution consists of the following:

- a barcode printer
- a barcode reader
- a mobile data collection application/program

A barcode label is a special symbology to represent human readable information such as a material number or batch number
in machine readable format.

There are different symbologies for different applications and different industries. Luckily, you need not worry to much about that as the logistics supply chain has mostly standardized on 3 of 9 and 128 barcode symbologies - which all barcode readers support and which SAP support natively in it's printing protocols.

You can print barcodes from SAP by modifying an existing output form.

Behind every output form is a print program that collects all the data and then pass it to the form. The form contains the layout as well as the font, line and paragraph formats. These forms are designed using SAPScript

(a very easy but frustratingly simplistic form format language) or SmartForms that is more of a graphical form design tool.

Barcodes are nothing more than a font definition and is part of the style sheet associated with a particular SAPScript form. The most important aspect is to place a parameter in the line of the form that points to the data element that you want to represent as barcode on the form, i.e. material number. Next you need to set the font for that parameter value to one of the supported barcode symbologies.

The next part of the equation can be a bit tricky as you will need to get a printer to print that barcode font. Regular laser printers does not normally print barcode fonts, only specialized industrial printers that is specifically designed to support that protocol and that uses specialized label media and heat transfer (resin) ribbon to create the sharp image required for barcodes.

Not to fear though, there are two ways to get around this:
- You can have your IT department do some research -
most laser printers can accept a font cartridge/dimm chip (similar to computer memory), called a **BarDIMM** that will allow a laser printer to support the printing of barcodes.
- Secondly, you can **buy software that you can upload in your SAP print Server** that will convert the barcode symbology as an image that will print on a regular laser printer. I found that this option results in less sharper barcodes. This option is really if you need to convert a large quantity of printers (>10) to support barcodes.
- Thirdly, you can **buy a third party software like Barcode.dll and install on your frontend PC** connected to the laser printer.

Now you have a barcode printed - what next?
Well there are two options, depending on your business requirements:
- You can use an existing SAP transaction on a regular workstation and get a barcode wedge reader to hook up between the keyboard and the PC. These wedge readers comes in a wand or scanner format. There are even wireless wedge scanners available that allows you to roam a few yards from the workstation to scan a label. This approach is mostly used where you want to prevent human errors in typing in long material, batch or serial numbers in receiving or issuing of material. The problem is that it's just replacing the keyboard input and you are basically locked down in one location and have to bring all the material to that location to process.
- Another solution is to use SAPConsole transactions
or write your own ABAP Dialog programs that will fit onto a barcode enabled wireless handheld terminal and that will follow the business logic as executed on the shop floor.

These programs are highly complex exercises in industrial engineering and ergonomics because of the limited s creen sizes and limited ability to accept keyboard input. The user is instructed step-by-step and only scan and push F-keys to interact with the SAP system. Scan, scan, beep, beep, enter - highly automated.

Delete Load program for SAPScript

Occassionally, when you make frequent changes to your SAPScript, the system can get out of sync.

When you view the form, the old data get display without your changes.

This can be fixed by deleting the SAPScript LOAD with program **RSTXDELL**.

Picture doesn't show in Print Preview

You have uploaded the picture as .TIF in Sap using ABAP **RSTXLDMC** and have also add the statement

/: INCLUDE ZHEX-SAMPLE-PICTURE OBJECT TEXT ID ST LANGUAGE EN

in your SapScript but the problem is that in print preview it's not displaying the picture.

It is normal that the picture doesn't show in print preview and you will be able to see the object only after prin ting.

Don't let this bother you as long as the picture is shown on the hardcopy printout.

Import/Export SapScript form from PC file

backup sapscript layout sets? Can you download and upload? How?

Use ABAP program: **RSTXSCRP**

It will download and upload your sapscripts as a text file in your local harddisk.

How to Upload graphics (IMAGE) to your Sapscript?

How to Upload graphics (IMAGE) to your Sapscript?

Command in your Sapscript

/: INCLUDE Z_YOUR_LOGO OBJECT TEXT ID ST LANGUAGE E

These are the steps to be followed for uploading graphics in R/3 system

1. First save the file as BMP
2. Open the BMP file in IMaging (Goto -> Programs -> Accessories -> Imaging) and make it Zoom as 100% and save as *.TIFF
3. Open SE38 and execute program RSTXLDMC
4. Give your TIFF file path name
5. Select Bcol (for Color)
6. TEXT ID will be ZHEX-MACRO-*.
7. Inplace of * write your own logo name (ZCOMPANYLOGO)
8. Execute the program
9. Now Goto SE71 create your ZFORM
10. Create logo window
11. Goto text element of logo window

or

In 4.6x :-

1. Goto SE71 Change the mode to GRAPHICAL
2. Choose the Graph Tabstrips
3. Now type in some name for the LOGO WINDOW
4. Press the IMPORT BUTTON and then IMPORT the BMP file from your DESKTOP
5. The code will be written automatically. You just need to drag and drop wherever you want the graphics to be.

Please note that in 4.6c onwards, you can also used Windows Bitmap file (.BMP).

How to convert Sapscript spools request to PDF?

SAP have created a standard program **RSTXPDFT4** to convert your Sapscripts spools into a PDF format.

Specify the spool number and you will be able to download the sapscripts spool into your local harddisk.

It look exactly like what you see during a spool display.

Please note that it is not restricted to sapsciprts spool only. Any reports in the spool can be converted using t he program '**RSTXPDFT4**'.

SAPScripts - Developing SAPScript in different languages

Developing SAPScript in different languages

You can goto transaction SE63 and translate the scripts into different languages.

In **SE63**, click **Translation -> Long Texts -> Sapscripts -> Forms**

Those language you can convert to have already been pre-installed in the system.

SE63 is the best way to translate since it offers check options.

However, it does not mean that it is 100% full proof that everything is correct

SAPscripts How to calculate Totals and Subtotals

I have some doubs in BDC and SMART FORMS. I want to change the material number using the trans action code MM02 through BDC.

In scripts and smartforms how to calculate totals and subtotals?

To calculate totals and sub totals in sap scripts you have to use subroutines.

Say if you have to add the unit price (KOMVD-KBERT) then in the main window whereever tat value is picked write this routine

/: DEFINE &TOT_PRICE&
/: PERFORM F_GET_PRICE IN PROGRAM <subroutine prog name> /:USING &KOMVD-KBERT& /:CHANGING &TOT_PRICE& /:ENDPERFORM

Then write the variable where ever you want it to be printed (mostly it will be in footer window)

Then create subroutine pool program and you have to write the code.

```
FORM F_GET_PRICE tables int_cond structure itcsy
                        outt_cond structure itcsy. data : value type kbert.

statics   value1 type kbert.
Read int_cond table index 1.
value = int_cond-value.

value1 = value1 + value.

Read outt_cond table index 1.
outt_cond-value = value1.
Modify outt_cond index 1.

ENDFORM.
```

I have given a rough outline, please be aware of the variable conversions as Int_cond-value and outt_cond-value are characters.

Retrieving data without modifying the original called program

```
*
* Retrieving data without modifying the original called program
*
* Put this script code in your sapscripts
* /: PERFORM GET_BARCODE IN PROGRAM ZSCRIPTPERFORM
* /: USING &PAGE&
* /: USING &NEXTPAGE&
* /: CHANGING &BARCODE&
* /: ENDPERFORM
* /  &BARCODE&
*
* Submitted by : SAP Basis, ABAP Programming and Other IMG Stuff
*                http://www.sap-img.com
*
REPORT ZSCRIPTPERFORM.

FORM GET_BARCODE TABLES  IN_PAR STRUCTURE ITCSY
                        OUT_PAR STRUCTURE ITCSY.

DATA: PAGNUM   LIKE SY-TABIX, "page number
      NEXTPAGE LIKE SY-TABIX. "number of next page

READ TABLE IN_PAR WITH KEY 'PAGE'.
CHECK SY-SUBRC = 0.
PAGNUM = IN_PAR-VALUE.

READ TABLE IN_PAR WITH KEY 'NEXTPAGE'.
CHECK SY-SUBRC = 0.
NEXTPAGE = IN_PAR-VALUE.

READ TABLE OUT_PAR WITH KEY 'BARCODE'.
CHECK SY-SUBRC = 0.
```

```
IF PAGNUM = 1.
   OUT_PAR-VALUE = '|'. "First page
ELSE.
   OUT_PAR-VALUE = '||'. "Next page
ENDIF.

IF NEXTPAGE = 0.
   OUT_PAR-VALUE+2 = 'L'. "Flag: last page
ENDIF.

MODIFY OUT_PAR INDEX SY-TABIX.

ENDFORM.

*-- End of Program
```

Orientations in SAPSCRIPT

Hi,

I have 2 pages for a Form in SAPscript .

Can I have 2 different Orientations for 2 pages
Ie Can I assign Page1 as Portrait & page2 as Landscape ???

If so , How ????

Thanks in Advance.
Ashwini Jaokar.

-----Reply Message-----
Subject: Re: Orientations in SAPSCRIPT
From: jmersinger

Ashwini,

Not that I know of in the same layoutset...what you can do is create two layoutsets...one portrait, one landscap e...then in the print program call each one individually.

jjm

-----Reply Message-----

Subject: RE: Orientations in SAPSCRIPT

From: Ralph Klassen

Each form may only have a single orientation but you can create two forms and include them in the same spo ol output.

In your ABAP program:

1. call function 'OPEN_FORM', don't pass a value in 'FORM'

2. call function 'START_FORM', include parameter 'FORM' passing the name of your first form

3. call function 'WRITE_FORM' as normal to output each element

4. call function 'END_FORM'

5. call function 'START_FORM', include parameter 'FORM' passing the name of your second form

6. call function 'WRITE_FORM' as normal to output each element

7. call function 'END_FORM'

8. call function 'CLOSE_FORM'

Repeat the 'START_FORM' ... 'END_FORM' sequence as required.

I have not tried using page numbers with this technique, I suspect that each form will restart at 1.

Regards,

Ralph Klassen Sylogist

-----End of Reply Message-----

Print Footer notes only on the last page

Command to used in your sapscripts :-

/: IF &NEXTPAGE& EQ 0

whatever footer you want.

/: ENDIF

Different font on the same line

You can have different font on the same line by defining a character format.

For example B for bold text and U for Underline.

In your SAPScript apply like this :

<U>Underline Text</> <B>Bold Text</>

SAP Printer commands in SAPScripts

The command line in the editor must be as follows:

/: PRINT-CONTROL xxxxx

or

/: PRINT-CONTROL 'xxxxx'

where xxxxx stands for the five-character name of the print control.

Example:
/: PRINT-CONTROL ZM100

The complete printer command normally resides in the print control.

If characters belonging to the print command follow after the print control in the text (only useful for the HPL2 printer driver for PCL-5 printers), the text line following after the PRINT-CONTROL command should begin with an equals sign (=) in the format column.

Example:
/: PRINT-CONTROL SESCP = *c5G

If you do not use the equals sign, a space character is inserted between the print control SESCP and the cha racter *c5G.

Refer to OSS note **5996 - How can SAPscript include printer commands?**

SAP Scripts Boxes/Lines/Shading

Setting default parameters for a box:

You can use the POSITION and SIZE commands to set default parmeters for a box.

Instead of:

```
/: BOX XPOS '11.21' MM  YPOS '5.31' MM  HEIGHT '10' MM WIDTH '20' MM INTENSITY 10 FRAME 0 TW
```

You can write:

```
/: POSITION XORIGIN '11.21' YORIGIN '5.31' MM
/: SIZE HEIGHT '2' MM WIDTH '76' MM
/: BOX  FRAME 10 TW INTENSITY 10
```

This can be usefull if you gave several boxes that share the same parameters.

If you want to set the position realtively to the window use POSITION WINDOW
to set the position to the top/left start of the window. Then use POSITION
to set the current position relatively to the start of the Window.

Note that you uses "+" or "-" in the ORIGIN position to the set the position relatively.

```
/: POSITION WINDOW
/: POSITION XORIGIN '+5' MM  YORIGIN '+10' MM
```

the position is now 5 MM from the left and 10 MM from the top of the window

NOTE: After using the position command you can move the current position

realtively to the last used position

```
/: POSITION XORIGIN '+10' MM  YORIGIN '+20' MM
```

Now the position will be X = 15 and Y = 30

Drawing a line. You can draw a line by setting the Height or Weidth to 0
and add a frane. E.g. a horizontal line:

```
/: SIZE HEIGHT '0' MM WIDTH '200' MM
/: BOX FRAME 10 TW XPOS '11.21' MM YPOS '14.81' MM INTENSITY 100
```

Reading Text in SAPScripts

If you only need to output the text, you don't need to used READ_TEXT like in an ABAP program, just use the INCLUDE command in SAPScript.

It will read the text and output it to your form.

The Syntax is like this:

/: INCLUDE &T166K-TXNAM& OBJECT &T166K-TDOBJECT& ID &T166K-TDID& LANGUAGE &EKKO-SPRAS&

SAPScript Important Programs

Here are some useful programs for SAPSCRIPT development/search ...

RSTXFCAT - Program to find out SAP Scirpt names (Search Program)

RSTXCDM1 - SAPscript: Sample Program for Form Printing

RSTXCNVR - Converting SAPscript standard text to RAW format (ASCII)

RSTXCPDF - Routines for Converting OTF Format to PDF Format

RSTXDBUG - Activate/Deactivate Form Debugger

RSTXFCAT - Find Forms

RSTXFCPY - Copy Forms Between Clients

RSTXFCOM - Comparison of Two Forms

RSTXFCON - SAPscript: Conversion of Page Format for Forms

RSTXFINF - Comprehensive Information about a Specific Form

RSTXHTML - Conversion of SAPscript Texts (ITF) to HTML

RSTXICON - List of SAP icons and their names and SAP numbers <xxxxx>

RSTXSYMB - List of SAP symbols and their names as well as <xxxxx> SAP number

RSTXR3TR - Transport Program For SAPscript Transport Objects

RSTXSCAT - Find Styles

RSTXSF01 - TrueType font installation for SAPscript/SmartForms

RSTXTCAT - Find Standard Texts

SAPScript Transaction codes

SE71 - Form painter

SE72 - Style maintenance

SE78 - SapScript Graphics Management

SO10 - Create standard text module

文本框 TextEdit Control

- **TIP:** Use SE75 to create your own custom text ID for SAVE_TEXT object

Contributed by Henrik Frank

1. Example 1: Creating the TextEdit control
2. Example 2: Event handling - Application event
3. Example 3: Event handling - System event
4. Example 4: Calling a methods of the control
5. Example 5: Responding to an event
6. Example 6: Protect a line in the TextEdit control and the importance of FLUSH
7. Example 7: Using multiple controls

See the whole program code

Example 1: Creating the TextEdit control

This is a simple example of how to implement a text edit control.

Steps

1. Create a report
2. In the start of selection event add: SET SCREEN '100'.
3. Create screen 100
4. Place a custom control on the screen by choosing the custom control icon which can be recognized by the letter 'C', and give it the name *MYCONTAINER1*.
5. To be able to exit the program, add a pushbutton with the function code EXIT.

6. In the elements list enter the name OK_CODE for the element of type OK.

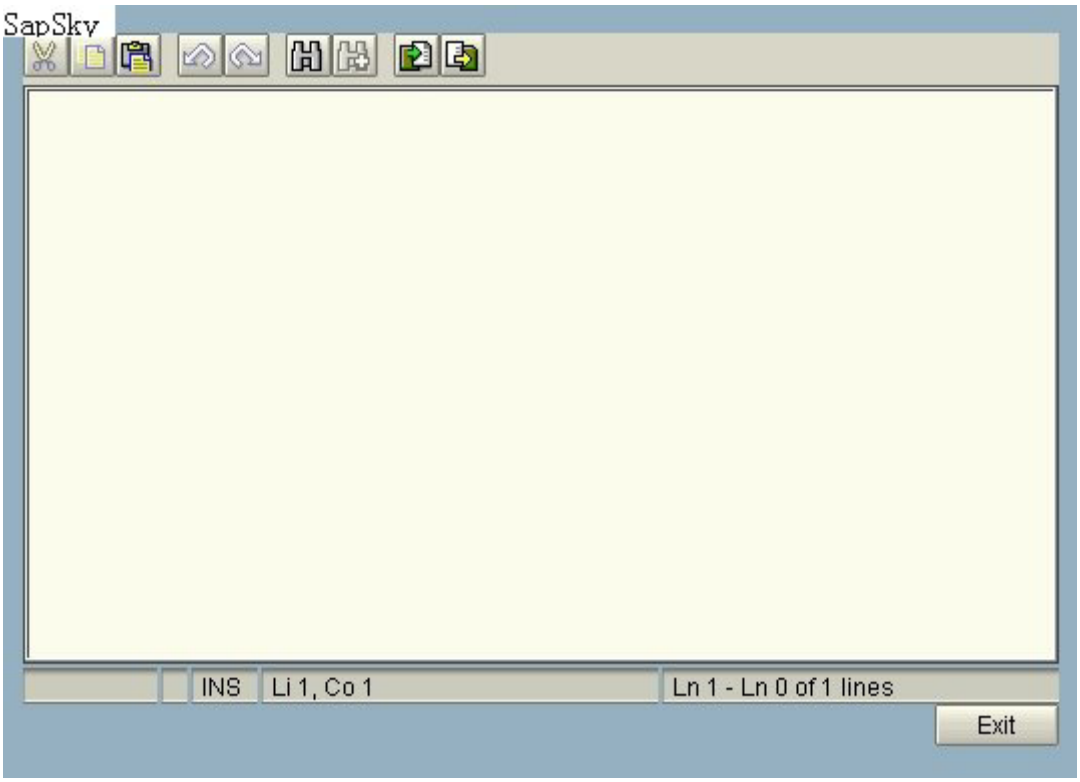
The code

```
REPORT sapmz_hf_controls1 .
CONSTANTS:
  line_length TYPE i VALUE 254.
DATA: ok_code LIKE sy-ucomm.
DATA:
* Create reference to the custom container
  custom_container TYPE REF TO cl_gui_custom_container,
* Create reference to the TextEdit control
  editor TYPE REF TO cl_gui_textedit,
  repid LIKE sy-repid.
START-OF-SELECTION.
  SET SCREEN '100'.
*---------------------------------------------------------------------*

*       MODULE USER_COMMAND_0100 INPUT                                *

*---------------------------------------------------------------------*

MODULE user_command_0100 INPUT.
  CASE ok_code.
    WHEN 'EXIT'.
      LEAVE TO SCREEN 0.
  ENDCASE.
ENDMODULE.                    " USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*

*&      Module  STATUS_0100  OUTPUT

*&---------------------------------------------------------------------*

MODULE status_0100 OUTPUT.
* The TextEdit control should only be initialized the first time the
```

```
* PBO module executes

  IF editor IS INITIAL.
    repid = sy-repid.
*   Create obejct for custom container
    CREATE OBJECT custom_container
      EXPORTING
        container_name              = 'MYCONTAINER1'
      EXCEPTIONS
        cntl_error                  = 1
        cntl_system_error           = 2
        create_error                = 3
        lifetime_error              = 4
        lifetime_dynpro_dynpro_link = 5
        others                      = 6
        .
    IF sy-subrc <> 0.
      MESSAGE ID sy-msgid TYPE 'I' NUMBER sy-msgno
                 WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
    ENDIF.
*   Create obejct for the TextEditor control
    CREATE OBJECT editor
      EXPORTING
         wordwrap_mode          =
                cl_gui_textedit=>wordwrap_at_fixed_position
         wordwrap_position      = line_length
         wordwrap_to_linebreak_mode = cl_gui_textedit=>true
        parent                  = custom_container
      EXCEPTIONS
        error_cntl_create      = 1
        error_cntl_init        = 2
        error_cntl_link        = 3
        error_dp_create        = 4
        gui_type_not_supported = 5
        others                 = 6
        .
```

```
    IF sy-subrc <> 0.
      MESSAGE ID sy-msgid TYPE 'I' NUMBER sy-msgno
                 WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
    ENDIF.
  ENDIF.
ENDMODULE.                    " STATUS_0100  OUTPUT
```

The result



Example 2: Event handling - Application event

There are 2 types of events:

- *System events*. These events are triggerede irrespective of the screen flow-logic.
- *Application events*. The PAI module is processed after an event. The method *CL_GUI_CFW=>DISPATCH* must be called to initiate event handling

In this example an application event is added to the program in example 1. New code is marked with red.

Steps:

1. Create an input/output field on screen 100, where the event type can be output. Name it EVENT_TYPE

The code:

```
REPORT sapmz_hf_controls1 .
CONSTANTS:
  line_length TYPE i VALUE 254.
DATA: ok_code LIKE sy-ucomm.
DATA:
* Create reference to the custom container
  custom_container TYPE REF TO cl_gui_custom_container,
* Create reference to the TextEdit control
  editor TYPE REF TO cl_gui_textedit,
  repid LIKE sy-repid.
*************************************************************************

* Impmenting events

*************************************************************************
DATA:

  event_type(20) TYPE c,

* Internal table for events that should be registred

  i_events TYPE cntl_simple_events,

* Structure for oneline of the table

  wa_events TYPE cntl_simple_event.
*-----------------------------------------------------------------------*

*      CLASS lcl_event_handler DEFINITION

*-----------------------------------------------------------------------*
```

```
CLASS lcl_event_handler DEFINITION.

  PUBLIC SECTION.

    CLASS-METHODS:

      catch_dblclick FOR EVENT dblclick

         OF cl_gui_textedit IMPORTING sender.
ENDCLASS.
CLASS lcl_event_handler IMPLEMENTATION.

  METHOD catch_dblclick.

    event_type = 'Event DBLCLICK raised'.

  ENDMETHOD.

ENDCLASS.



START-OF-SELECTION.
  CLEAR wa_events. refresh i_events.
  SET SCREEN '100'.


*---------------------------------------------------------------------*

*       MODULE USER_COMMAND_0100 INPUT                                *

*---------------------------------------------------------------------*

MODULE user_command_0100 INPUT.
  CASE ok_code.
```

```
    WHEN 'EXIT'.
      LEAVE TO SCREEN 0.
    WHEN OTHERS.
*   Call the Dispacth method to initiate application event handling

    call method cl_gui_cfw=>Dispatch.


  ENDCASE.
ENDMODULE.                    " USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*

*&      Module  STATUS_0100  OUTPUT

*&---------------------------------------------------------------------*

MODULE status_0100 OUTPUT.
* The TextEdit control shoul only be initialized the first time the

* PBO module executes
  IF editor IS INITIAL.
    repid = sy-repid.
*   Create obejct for custom container
    CREATE OBJECT custom_container
      EXPORTING
        container_name              = 'MYCONTAINER1'
      EXCEPTIONS
        cntl_error                  = 1
        cntl_system_error           = 2
        create_error                = 3
        lifetime_error              = 4
        lifetime_dynpro_dynpro_link = 5
        others                      = 6
        .
    IF sy-subrc <> 0.
      MESSAGE ID sy-msgid TYPE 'I' NUMBER sy-msgno
```

```
            WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
  ENDIF.
*  Create obejct for the TextEditor control
  CREATE OBJECT editor
    EXPORTING
      wordwrap_mode          =
             cl_gui_textedit=>wordwrap_at_fixed_position
      wordwrap_position      = line_length
      wordwrap_to_linebreak_mode = cl_gui_textedit=>true
     parent                  = custom_container
    EXCEPTIONS
     error_cntl_create      = 1
     error_cntl_init        = 2
     error_cntl_link        = 3
     error_dp_create        = 4
     gui_type_not_supported = 5
     others                 = 6
     .
  IF sy-subrc <> 0.
    MESSAGE ID sy-msgid TYPE 'I' NUMBER sy-msgno
            WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
  ENDIF.
*  Link the event handler method to the event and the

*  TextEdit control
  SET HANDLER lcl_event_handler=>catch_dblclick FOR editor.
*  Register the event in the internal table i_events
  wa_events-eventid = cl_gui_textedit=>event_double_click.

  wa_events-appl_event = 'X'. "This is an application event

  append wa_events to i_events.
*  Pass the table to the TextEdit control using method

*  set_registred_events
```

```
    call method editor->set_registered_events

       exporting events = i_events.
  ENDIF.
ENDMODULE.                    " STATUS_0100  OUTPUT
```

Result:

When you double click on the TextEdit control, the input/ouput field should show the text: *Event DBLCLICK*

Example 3: Event handling - System event

System events are passed irrespective of the flow-logic of the screen. To implement a system event change the code from example 2 as follows:

Code:

```
CLASS lcl_event_handler IMPLEMENTATION.
  METHOD catch_dblclick.
*---  event_type = 'Event DBLCLICK raised'.
* Reacting to the system event
  call method cl_gui_cfw=>set_new_ok_code

    exporting new_code = 'SHOW'.





MODULE user_command_0100 INPUT.
  CASE ok_code.
   code.........
    WHEN 'SHOW'.
      event_type = 'System dblclick'.
    WHEN OTHERS.
*----    call method cl_gui_cfw=>Dispatch.
  ENDCASE.
ENDMODULE.                    " USER_COMMAND_0100  INPUT
```

```
MODULE status_0100 OUTPUT.
 Code ................
*---    wa_events-appl_event = 'X'. "This is an application event

    wa_events-appl_event = space. "This is a system event

  ENDIF.
ENDMODULE.                    " STATUS_0100  OUTPUT
```

Result:

When you double clicks on the TextEdit control nothing happens, since the flow-logic of the screen an dthe fiel de transport is ignore.

Example 4: Calling methods of the control

In this exercise a function that loads the texts of an internal table into the text window, is implemented.

Steps:

Define anoterh pushbutton on the screen, that activates the method that fills the TextEdit control. Give itname *PUSHBUTTON_IMPORT* and function code *IMP*.

Define a form CREATE_TEXTS that carries out the text import.

Only changes to the code in example 2 is show.

Code:

```
MODULE user_command_0100 INPUT.
  CASE ok_code.

    code.........

     WHEN 'IMP'.

        perform load_texts.
```

```
  ENDCASE.

ENDMODULE.                    " USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*

*&      Form  load_texts

*&---------------------------------------------------------------------*

* This form creates an internal table with texts. The the contents of

* the table is instered into the TextEdit control using method

* set_text_as_r3table

*---------------------------------------------------------------------*
FORM load_texts.
  TYPES:
   BEGIN OF t_texttable,
     line(line_length) TYPE c,
   END OF t_texttable.
  DATA
    i_texttable TYPE TABLE OF t_texttable.
* Create internal table with texts
  APPEND 'This a method that fills the TextEdit control' TO i_texttable.
  APPEND 'with a text.' TO i_texttable.
  DO 10 TIMES.
    APPEND 'hallo world !' TO i_texttable.
  ENDDO.
* Load TextEdit control with texts
  CALL METHOD editor->set_text_as_r3table
    EXPORTING table = i_texttable.
  IF sy-subrc > 0.
*   Display an error message
    EXIT.
  ENDIF.
```

```
* All methods that operates on controls are transferred to the frontend

* by a RFC calls. the method FLUSH is used to determine when this is done.
  CALL METHOD cl_gui_cfw=>flush.
  IF sy-subrc > 0.
*   Display an error message
  ENDIF.
ENDFORM.                    " create_texts
```

Example 5: Responding to an event

When you double click on a text line in the TextEdit control, you want it to be prefixed with a '*'.

The line number of the TextEdit control that is double clicked, is retreived using method GET_SELECTION_POS. The internal text table is reloaded froim the TextEdit control with method GET_TEXT_AS_R3TABLE. The position of the double click in the TextEdit control is used to find the entry in the table, and the entry is prefixed with '*' and loaded into the TextEdit control again.

The program should be changed so that the internal table i_texttable is global, and a global flag g_loaded added. The load of the table should be moved to the PBO module. The changes in thje code are marked with red. The whole program now looks like this:

Code

```
REPORT sapmz_hf_controls1 .
CONSTANTS:
  line_length TYPE i VALUE 254.
DATA: ok_code LIKE sy-ucomm.
DATA:
* Create reference to the custom container
  custom_container TYPE REF TO cl_gui_custom_container,
* Create reference to the TextEdit control
  editor TYPE REF TO cl_gui_textedit,
  repid LIKE sy-repid.
*********************************************************************
```

```
* Utillity table to load texts

*********************************************************************
TYPES:

   BEGIN OF t_texttable,

     line(line_length) TYPE c,

   END OF t_texttable.
DATA:

  i_texttable TYPE TABLE OF t_texttable,

  g_loaded(1) TYPE c.


*********************************************************************

* Impmenting events

*********************************************************************
DATA:
  event_type(20) TYPE c,
* Internal table for events that should be registred
  i_events TYPE cntl_simple_events,
* Structure for oneline of the table
  wa_events TYPE cntl_simple_event.
*---------------------------------------------------------------------*

*       CLASS lcl_event_handler DEFINITION

*---------------------------------------------------------------------*
CLASS lcl_event_handler DEFINITION.
  PUBLIC SECTION.
    CLASS-METHODS:
```

```
      catch_dblclick FOR EVENT dblclick
         OF cl_gui_textedit IMPORTING sender.
ENDCLASS.
CLASS lcl_event_handler IMPLEMENTATION.
  METHOD catch_dblclick.
    DATA:
      from_line TYPE i,
      from_pos  TYPE i,
      to_line TYPE i,
      to_pos TYPE i,
      wa_texttable TYPE t_texttable.
* Used for the sytem event
  call method cl_gui_cfw=>set_new_ok_code
    exporting new_code = 'SHOW'.
* Read the position of the double click
    CALL METHOD sender->get_selection_pos

      IMPORTING

         from_line = from_line

         from_pos  = from_pos

         to_line   = to_line

         to_pos    = to_pos.
*   Texts in the TextEdit control can have been changed, so

*   first reload text from the control into the internal

*   table that contains text
    IF NOT g_loaded IS INITIAL.

      CALL METHOD sender->get_text_as_r3table

           IMPORTING table = i_texttable.
```

```
*   Read the line of the internal table that was clicked
      READ TABLE i_texttable INDEX from_line INTO wa_texttable.

      IF sy-subrc <> 0.

        EXIT.

      ENDIF.
      IF wa_texttable+0(1) CS '*'.

        SHIFT wa_texttable.

      ELSEIF wa_texttable+0(1) NS '*'.

        SHIFT wa_texttable RIGHT.

        wa_texttable+0(1) = '*'.

      ENDIF.

      modify i_texttable from wa_texttable index from_line.
*     Reload texts from h einternal table
      perform load_texts.
    ENDIF.
  ENDMETHOD.
ENDCLASS.


START-OF-SELECTION.
  CLEAR wa_events.
  REFRESH: i_events.
  SET SCREEN '100'.
*---------------------------------------------------------------------*

*       MODULE USER_COMMAND_0100 INPUT                                *
```

```
*---------------------------------------------------------------------*
MODULE user_command_0100 INPUT.
  CASE ok_code.
    WHEN 'EXIT'.
      LEAVE TO SCREEN 0.
    WHEN 'SHOW'.
      event_type = 'System dblclick'.
    WHEN 'IMP'.
      PERFORM Load_texts.
    WHEN OTHERS.
*    CALL METHOD cl_gui_cfw=>dispatch. "Not used for system events
  ENDCASE.
ENDMODULE.                    " USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*

*&      Module  STATUS_0100  OUTPUT

*&---------------------------------------------------------------------*
MODULE status_0100 OUTPUT.
* The TextEdit control shoul only be initialized the first time the

* PBO module executes
  IF editor IS INITIAL.
    repid = sy-repid.
*   Create object for custom container
    CREATE OBJECT custom_container
      EXPORTING
        container_name              = 'MYCONTAINER1'
      EXCEPTIONS
        cntl_error                  = 1
        cntl_system_error           = 2
        create_error                = 3
        lifetime_error              = 4
        lifetime_dynpro_dynpro_link = 5
        others                      = 6
```

```
      .
    IF sy-subrc <> 0.
      MESSAGE ID sy-msgid TYPE 'I' NUMBER sy-msgno
                 WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
    ENDIF.
*   Create obejct for the TextEditor control
    CREATE OBJECT editor
      EXPORTING
        wordwrap_mode          =
               cl_gui_textedit=>wordwrap_at_fixed_position
        wordwrap_position      = line_length
        wordwrap_to_linebreak_mode = cl_gui_textedit=>true
       parent                  = custom_container
      EXCEPTIONS
       error_cntl_create      = 1
       error_cntl_init        = 2
       error_cntl_link        = 3
       error_dp_create        = 4
       gui_type_not_supported = 5
       others                 = 6
       .
    IF sy-subrc <> 0.
      MESSAGE ID sy-msgid TYPE 'I' NUMBER sy-msgno
                 WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
    ENDIF.
*   Link the event handler method to the event and the

*   TextEdit control
    SET HANDLER lcl_event_handler=>catch_dblclick FOR editor.
*   Register the event in the internal table i_events
    wa_events-eventid = cl_gui_textedit=>event_double_click.
*    wa_events-appl_event = 'X'. "This is an application event
    wa_events-appl_event = space. "This is a system event
    APPEND wa_events TO i_events.
*   Pass the table to the TextEdit control uding method
```

```
*   set_registred_events
    CALL METHOD editor->set_registered_events
       EXPORTING events = i_events.
* Create internal table with texts taht can be uploaded to

* the TextEdit control
  APPEND 'This a method that fills the TextEdit control' TO i_texttable.
    APPEND 'with a text.' TO i_texttable.
    DO 10 TIMES.
      APPEND 'hallo world !' TO i_texttable.
    ENDDO.



  ENDIF.
ENDMODULE.                    " STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*

*&      Form Load_texts

*&---------------------------------------------------------------------*

* This form loads the lines of the internal table i_texttable into

* the TextEdit control

*---------------------------------------------------------------------*
FORM Load_texts.
* Load TextEdit control with texts
  CALL METHOD editor->set_text_as_r3table
    EXPORTING table = i_texttable.
  IF sy-subrc > 0.
*   Display an error message
    EXIT.
  ENDIF.
* All methods that operates on controls are transferred to the frontend
```

```
* by a RFC calls. the method FLUSH is used to determine when this is

* done.
  CALL METHOD cl_gui_cfw=>flush.
  IF sy-subrc > 0.
*   Display an error message
  ENDIF.
  g_loaded = 'X'.
ENDFORM.                    " create_texts
```

Example 6: Protect a line in the TextEdit control and the importance of FLUSH

All methods that operates on controls are transfered to the fronend by RFC calls. The *FLUSH* method is used to synchronize control execution and the frontend. This is very important when working e.g. with export parameters from a method, as the parmeters will not be correct before the *FLUSH* method has been called.

The example below portects selected lines in the TextEdit and uses FLUSH to ensure that the correct parameters are returned from method *GET_SELECTION_POS*.

Note: Instead of using method *PROTECT_LINES*, the method *PROTECT_SELECTION* could be used. This method does not need line numbers or a FLUSH statement

Steps

- Add a new pushbutton to the screen with the function code *PROTECT.*

Code

Add the following code to the example:

```
* Global variables

DATA:

    from_idx TYPE i,

    to_idx TYPE i,
```

```
    index TYPE i.

MODULE user_command_0100 INPUT.

  CASE ok_code.

    code.......................

    WHEN 'PROTECT'.

      PERFORM protect.



.     .......................

  ENDCASE.

*&---------------------------------------------------------------------*

*&      Form  protect

*&---------------------------------------------------------------------*

* Protects marked lines in a TextEdit control

*---------------------------------------------------------------------*
FORM protect.
* Determine the area selected by the user
  CALL METHOD editor->get_selection_pos
     IMPORTING
       from_line = from_idx
       to_line   = to_idx
     EXCEPTIONS
       error_cntl_call_method = 1.
* Synchronize execution in the control with the ABAP program.
```

```
* Without this synchronization the variables from_idx and

* to_idx will have obsolutete values (The initial value for

* both, are 0)
  CALL METHOD cl_gui_cfw=>flush.
  IF sy-subrc > 0.
* Errormessage: Error in flush
  ENDIF.
* Protect the selected lines
  IF to_idx > from_idx.
    to_idx = to_idx - 1.
  ENDIF.
  CALL METHOD editor->protect_lines
    EXPORTING
      from_line = from_idx
      to_line   = to_idx.
* The PROTECT_SELECTION method could be used instead, eliminating the

* need of line numbers and the last FLUSH




* call method editor->protect_selection.
* Flush again to protect immidiately

  CALL METHOD cl_gui_cfw=>flush.
  IF sy-subrc > 0.
* Errormessage: Error in flush
  ENDIF.


ENDFORM.                    " protect
```

Example 7: Using multiple controls

In this example a second TextEdit control will be added to the screen. The new TextEdit control will be designed to act as a clipboard for short texts.

Steps:

- Add a new container to the screen and name it *MYCONTAINER2.*

Code:

Insert global datadeclaration:

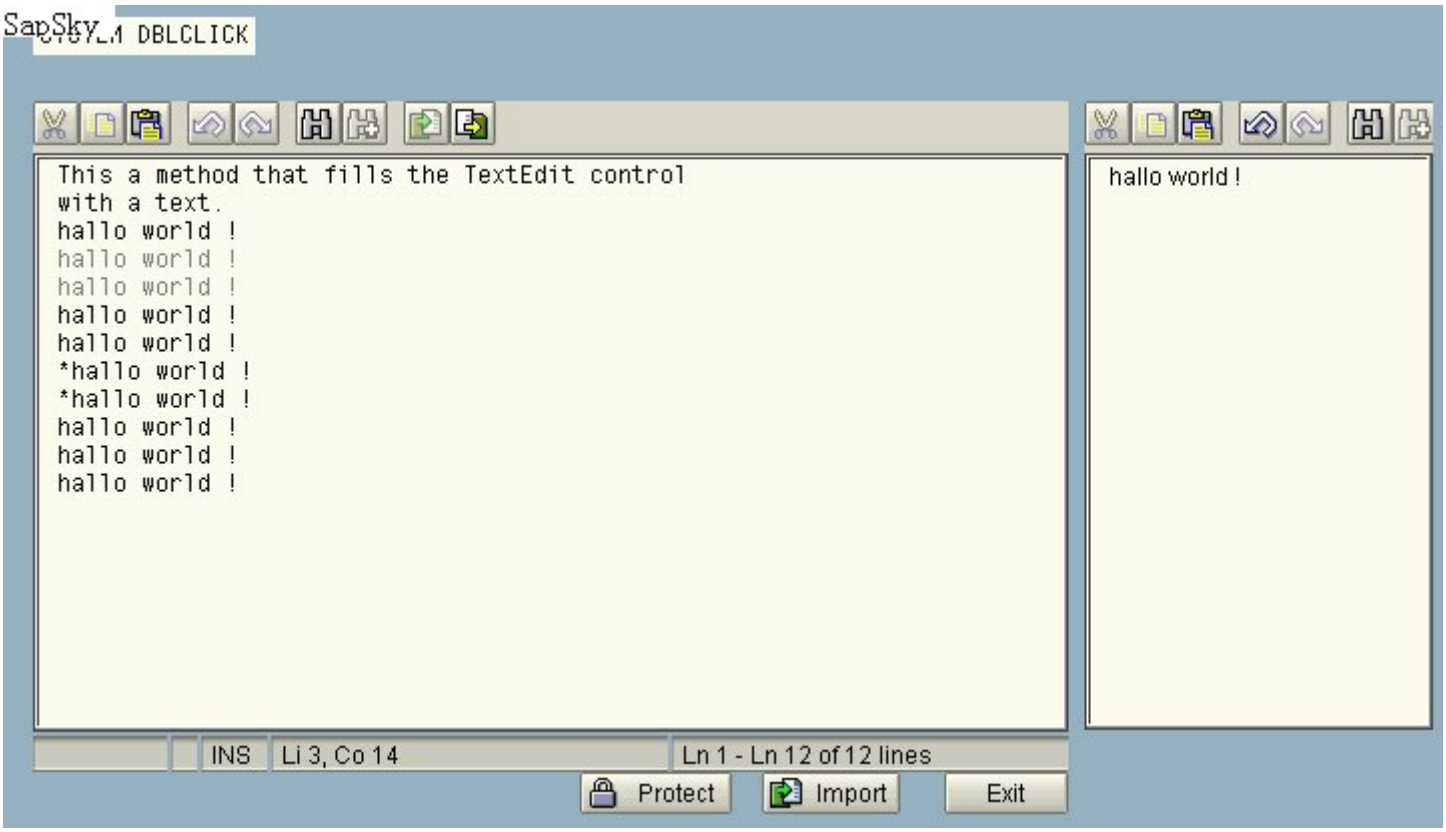
```
**********************************************************************

* Implementing a second Scratch TextEdit control

**********************************************************************

DATA:

  scratch                TYPE REF TO cl_gui_textedit,

  custom_container2   TYPE REF TO cl_gui_custom_container.
```

Insert the following code in the PBO module:

```
*-------------------------------------------------------

* The SCRATCH TextEdit control

*-------------------------------------------------------

  IF scratch IS INITIAL.
*   Create obejct for custom container2
    CREATE OBJECT custom_container2
```

```
      EXPORTING
        container_name              = 'MYCONTAINER2'
      EXCEPTIONS
        cntl_error                  = 1
        cntl_system_error           = 2
        create_error                = 3
        lifetime_error              = 4
        lifetime_dynpro_dynpro_link = 5
        others                      = 6
        .
    IF sy-subrc <> 0.
      MESSAGE ID sy-msgid TYPE 'I' NUMBER sy-msgno
                 WITH sy-msgv1 sy-msgv2 sy-msgv3 sy-msgv4.
    ENDIF.
*   Create obejct for the SCRATCH TextEditor control
    CREATE OBJECT scratch
      EXPORTING
         parent         = custom_container2
         wordwrap_mode  =
                cl_gui_textedit=>wordwrap_at_windowborder
        wordwrap_to_linebreak_mode = cl_gui_textedit=>true.
*   Remove the staus bar
    CALL METHOD scratch->set_statusbar_mode
      EXPORTING statusbar_mode = cl_gui_textedit=>false.
  ENDIF.
```

Result:



[推荐] 很不错的 OOP ALV

[日期：2006-12-01] 来源：sapsky 作者：sapsky [字体：大 中 小]

Contributed by Henrik Frank

Preface

Note that practical examples of using the ALV grid can be found in development class SLIS.

Example of the display of an ALV grid:

SapSky

ID	No.	Flight date	Airfare	Curr.	Plane type	Max. capacity	Occupancy	Booking total
	5555	11.08.2001	1.000,00	GBP	BOING 767	250	200	200.000,00
	756	19.05.2001	2.000,00	GBP	BOING 747	350	200	400.000,00
B				**GBP**				**600.000,00**
	505	10.01.2001	1.000,00	EUR	CRASH II	50	35	35.000,00
L				**EUR**				**35.000,00**
	551	18.05.2001	1.500,00	DKK	MD80	120	100	150.000,00
	7677	11.08.2001	10.000,00	DKK	BOING 767	250	200	2.000.000,00
S				**DKK**				**2.150.000,00**
				GBP				**600.000,00**
				EUR				**35.000,00**
				DKK				**2.150.000,00**

Simple example of how to implement an ALV grid

Note that this example uses table ZSFLIGHT. The table is equivalent to the table SFLIGHT.

Steps:

1. Create an executable program (Report)
2. Create a screen (100) and place a custom container named ALV_CONTAINER on the screen
3. Create a Pushbutton. Give it the text Exit and the functioncode EXIT

```
REPORT sapmz_hf_alv_grid .
TABLES: zsflight.
*---------------------------------------------------------------------
* G L O B A L   I N T E R N A L   T A B L E S
*---------------------------------------------------------------------
DATA: gi_sflight TYPE STANDARD TABLE OF sflight.
*---------------------------------------------------------------------
* G L O B A L   D A T A
*---------------------------------------------------------------------
DATA: ok_code LIKE sy-ucomm,
      g_wa_sflight LIKE sflight.
* Declare reference variables to the ALV grid and the container
DATA:
  go_grid              TYPE REF TO cl_gui_alv_grid,
  go_custom_container TYPE REF TO cl_gui_custom_container.
```

```
*---------------------------------------------------------------------
* S T A R T - O F - S E L E C T I O N.
*---------------------------------------------------------------------
START-OF-SELECTION.
  SET SCREEN '100'.
*&---------------------------------------------------------------------*
*&      Module  USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
MODULE user_command_0100 INPUT.
  CASE ok_code.
    WHEN 'EXIT'.
      LEAVE TO SCREEN 0.
  ENDCASE.
ENDMODULE.                    " USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
*&      Module  STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
MODULE status_0100 OUTPUT.
* Create objects
  IF go_custom_container IS INITIAL.
    CREATE OBJECT go_custom_container
      EXPORTING container_name = 'ALV_CONTAINER'.
    CREATE OBJECT go_grid
      EXPORTING
        i_parent = go_custom_container.
    PERFORM load_data_into_grid.
  ENDIF.
ENDMODULE.                    " STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
*&      Form  load_data_into_grid
*&---------------------------------------------------------------------*
FORM load_data_into_grid.
* Read data from table SFLIGHT
  SELECT *
    FROM zsflight
    INTO TABLE gi_sflight.
```

```
  * Load data into the grid and display them
    CALL METHOD go_grid->set_table_for_first_display
      EXPORTING i_structure_name = 'SFLIGHT'
      CHANGING  it_outtab        = gi_sflight.
  ENDFORM.                       " load_data_into_grid
```

Allow the user to save and reuse the layout

A button can be shown on the grid toolbar, allowing the user to save and reuse a layout. The button looks like this: 

See also example in SAP standard program BCALV_GRID_09.

To do this use the parameters *IS_VARIANT* and *I_SAVE* of the *set_table_for_first_display* method. Note that the IS_VARIANT parameter must have the structure *DISVARIANT*.

The I_SAVE "Options for saving layouts" parameter can have the following values:

- **U** Only user specific layouts can be saved
- **X** Only global layouts can be saved
- **A** Both user specific and global layouts can be saved
- **Space** Layouts can not be saved

Add the following code to the example:

```
FORM load_data_into_grid.
  DATA:
*    For parameter IS_VARIANT
     l_layout TYPE disvariant.
    Code..........
* Load data into the grid and display them
  l_layout-report = sy-repid.
  CALL METHOD go_grid->set_table_for_first_display
     EXPORTING i_structure_name = 'SFLIGHT'
               is_variant       = l_layout
               i_save           = 'A'
     CHANGING  it_outtab        = gi_
```

Integrate user defined functions in the grid toolbar

Possibilities:

- Replace existing functions in the toolbar or context men with user defined functions
- Add new functions to the toolbar or context menu

Note that the whole toolbar can be removed using the IT_TOOLBAR_EXCLUDING parameter of the *set_table_for_first_display* method.

See also example in SAP standard program BCALV_GRID_05

1) To get access to the icons insert the following statement in the top of the program:

TYPE-POOLS: icon.

2) To allow the declaration of o_event_receiver before the lcl_event_receiver class is defined, declare it as deferred in the start of the program

* To allow the declaration of o_event_receiver before the lcl_event_receiver class is defined, declare it as deferred in the
* start of the program
CLASS lcl_event_receiver DEFINITION DEFERRED.

3) Declare reference to the event handler class

```
DATA:
o_event_receiver TYPE REF TO lcl_event_receiver.
```

4) Class for event receiver. This class adds the new button to the toolbar and handles the event when the button is pushed

```
CLASS lcl_event_receiver DEFINITION.
  PUBLIC SECTION.
    METHODS:
     handle_toolbar FOR EVENT toolbar OF cl_gui_alv_grid
       IMPORTING
         e_object e_interactive,
     handle_user_command FOR EVENT user_command OF cl_gui_alv_grid
       IMPORTING e_ucomm.
ENDCLASS.
*---------------------------------------------------------------------*
*       CLASS lcl_event_receiver IMPLEMENTATION
*---------------------------------------------------------------------*
```

```
CLASS lcl_event_receiver IMPLEMENTATION.
  METHOD handle_toolbar.
*   Event handler method for event toolbar.
      CONSTANTS:
*   Constants for button type
        c_button_normal              TYPE i VALUE 0,
        c_menu_and_default_button TYPE i VALUE 1,
       c_menu                        TYPE i VALUE 2,
        c_separator                  TYPE i VALUE 3,
        c_radio_button               TYPE i VALUE 4,
        c_checkbox                    TYPE i VALUE 5,
        c_menu_entry                  TYPE i VALUE 6.
      DATA:
          ls_toolbar   TYPE stb_button.
*    Append seperator to the normal toolbar
      CLEAR ls_toolbar.
      MOVE c_separator TO ls_toolbar-butn_type..
      APPEND ls_toolbar TO e_object->mt_toolbar.
*    Append a new button that to the toolbar. Use E_OBJECT of
*    event toolbar. E_OBJECT is of type CL_ALV_EVENT_TOOLBAR_SET.
*    This class has one attribute MT_TOOLBAR which is of table type
*    TTB_BUTTON. The structure is STB_BUTTON
      CLEAR ls_toolbar.
      MOVE 'CHANGE'           TO ls_toolbar-function.
      MOVE  icon_change      TO ls_toolbar-icon.
      MOVE 'Change flight' TO ls_toolbar-quickinfo.
      MOVE 'Change'           TO ls_toolbar-text.
      MOVE ' '                 TO ls_toolbar-disabled.
      APPEND ls_toolbar      TO e_object->mt_toolbar.
    ENDMETHOD.
    METHOD handle_user_command.
*    Handle own functions defined in the toolbar
      CASE e_ucomm.
        WHEN 'CHANGE'.
          LEAVE TO SCREEN 0.
      ENDCASE.
```

```
  ENDMETHOD.
ENDCLASS.
```

5) In the PBO module, crate object for event handler and set handler

```
    CREATE OBJECT o_event_receiver.
    SET HANDLER o_event_receiver->handle_user_command FOR go_grid.
    SET HANDLER o_event_receiver->handle_toolbar FOR go_grid.
```

6) In the PBO module after the CALL METHOD go_grid->set_table_for_first_display, rai se event

toolbar to show the modified toolbar

```
CALL METHOD go_grid->set_toolbar_interactive.
```

Set focus to the grid

After *CALL METHOD go_grid->set_table_for_first_display* insert the following statement:

```
CALL METHOD cl_gui_control=>set_focus EXPORTING control = go_grid.
```

Set the title of the grid

Fill the *grid_title* field of structure *lvc_s_layo*.

Note that the structure *lvc_s_layo* can be used for to customize the grid appearance i n many ways.

```
DATA:
* ALV control: Layout structure
gs_layout TYPE lvc_s_layo.

* Set grid title
gs_layout-grid_title = 'Flights'.
CALL METHOD go_grid->set_table_for_first_display
            EXPORTING i_structure_name = 'SFLIGHT'
                      is_layout        = gs_layout
            CHANGING  it_outtab        = gi_sflight.
```

Customize the appearence of the grid

The structure *lvc_s_layo* contains fields for setting graphical properties, displaying exce ptions, calculating totals and enabling specific interaction options.

Fill the apporpiate fields of structure *lvc_s_layo* and insert it as a parameter in the CALL METHOD go_grid->set_table_for_first_display. See the example under Set the title of the grid.

If you want to change apperance after list output, use the methods *get_frontend_layout* and *set_frontend_layout*.

Examples of fields in structure *lvc_s_layo:*

GRID_TITLE Setting the title of the grid

SEL_MODE. Selection mode, determines how rows can be selected. Can have the following values:

- **A** Multiple columns, multiple rows with selection buttons.
- **B** Simple selection, listbox, Single row/column
- **C** Multiple rows without buttons
- **D** Multiple rows with buttons and select all ICON

Setting and getting selected rows (Columns) and read line contents

You can read which rows of the grid that has been selected, and dynamic select rows of the grid using methods *get_selected_rows* and *set_selected_rows*. There are similar methods for columns.

Note that the grid table always has the rows in the same sequence as displayed in the grid, thus you can use the index of the selected row(s) to read the information in the rows from the table. In the examples below the grid table is named gi_sflight.

Data declaration:

```
DATA:
* Internal table for indexes of selected rows
gi_index_rows TYPE lvc_t_row,
* Information about 1 row
g_selected_row LIKE lvc_s_row.
```

Example 1: Reading index of selected row(s) and using it to read the grid table

```
  CALL METHOD go_grid->get_selected_rows
```

```
    IMPORTING
      et_index_rows = gi_index_rows.
  DESCRIBE TABLE gi_index_rows LINES l_lines.
  IF l_lines = 0.
    CALL FUNCTION 'POPUP_TO_DISPLAY_TEXT'
          EXPORTING
                textline1 = 'You must choose a valid line'.
    EXIT.
  ENDIF.
  LOOP AT gi_index_rows INTO g_selected_row.
     READ TABLE gi_sflight INDEX g_selected_row-index INTO g_wa_sflight.
    ENDIF.
  ENDLOOP.
```

\

Example 2: Set selected row(s).

```
  DESCRIBE TABLE gi_index_rows LINES l_lines.
  IF l_lines > 0.
    CALL METHOD go_grid->set_selected_rows
         exporting
           it_index_rows = gi_index_rows.
  ENDIF.
```

Make an Exception field (= Traffic lights)

There can be defined a column in the grid for display of traffic lights. This field is of type Char 1, and can contain the following values:

- **1** Red
- **2** Yellow
- **3** Green

The name of the traffic light field is supplied inh the *gs_layout-excp_fname* used by method *set_table_for_first_display*.

Example

```
TYPES: BEGIN OF st_sflight.
          INCLUDE STRUCTURE zsflight.
TYPES:   traffic_light TYPE c.
```

```
TYPES: END OF st_sflight.
TYPES: tt_sflight TYPE STANDARD TABLE OF st_sflight.
DATA: gi_sflight TYPE tt_sflight.
*    Set the exception field of the table
     LOOP AT gi_sflight INTO g_wa_sflight.
       IF g_wa_sflight-paymentsum < 100000.
         g_wa_sflight-traffic_light = '1'.
       ELSEIF g_wa_sflight-paymentsum => 100000 AND
              g_wa_sflight-paymentsum < 1000000.
         g_wa_sflight-traffic_light = '2'.
       ELSE.
         g_wa_sflight-traffic_light = '3'.
       ENDIF.
       MODIFY gi_sflight FROM g_wa_sflight.
     ENDLOOP.
*    Name of the exception field (Traffic light field)
     gs_layout-excp_fname = 'TRAFFIC_LIGHT'.
*    Grid setup for first display
     CALL METHOD go_grid->set_table_for_first_display
       EXPORTING i_structure_name = 'SFLIGHT'
                                 is_layout                    = gs_layout
       CHANGING   it_outtab                        = gi_sflight.
```

Color a line

The steps for coloring a line i the grid is much the same as making a traffic light.

* To color a line the structure of the table must include a Char 4 field for color properties

```
TYPES: BEGIN OF st_sflight.
          INCLUDE STRUCTURE zsflight.
*         Field for line color
types:   line_color(4) type c.
TYPES: END OF st_sflight.
TYPES: tt_sflight TYPE STANDARD TABLE OF st_sflight.
DATA: gi_sflight TYPE tt_sflight.
```

* Loop trough the table to set the color properties of each line. The color properties field is

```
* Char 4 and the characters is set as follows:
* Char 1 = C = This is a color property
* Char 2 = 6 = Color code (1 - 7)
* Char 3 = Intensified on/of = 1 = on
* Char 4 = Inverse display = 0 = of
      LOOP AT gi_sflight INTO g_wa_sflight.
        IF g_wa_sflight-paymentsum < 100000.
          g_wa_sflight-line_color      = 'C610'.
        ENDIF.
        MODIFY gi_sflight FROM g_wa_sflight.
     ENDLOOP.
* Name of the color field
gs_layout-info_fname = 'LINE_COLOR'.
* Grid setup for first display
 CALL METHOD go_grid->set_table_for_first_display
        EXPORTING i_structure_name = 'SFLIGHT'
                                  is_layout                    = gs_layout
        CHANGING   it_outtab                        = gi_sflight.
```

Refresh grid display

Use the grid method *REFRESH_TABLE_DISPLAY*

Example:

```
CALL METHOD go_grid->refresh_table_display.
```

Complete code for the ALV grid example

This example shows and ALV grid with flights. After selecting a line a change button can be pushed to display a change screen. After the changes have been saved, the ALV grid screen is displayed again, and the grid is updated with the changes.

The example shows:

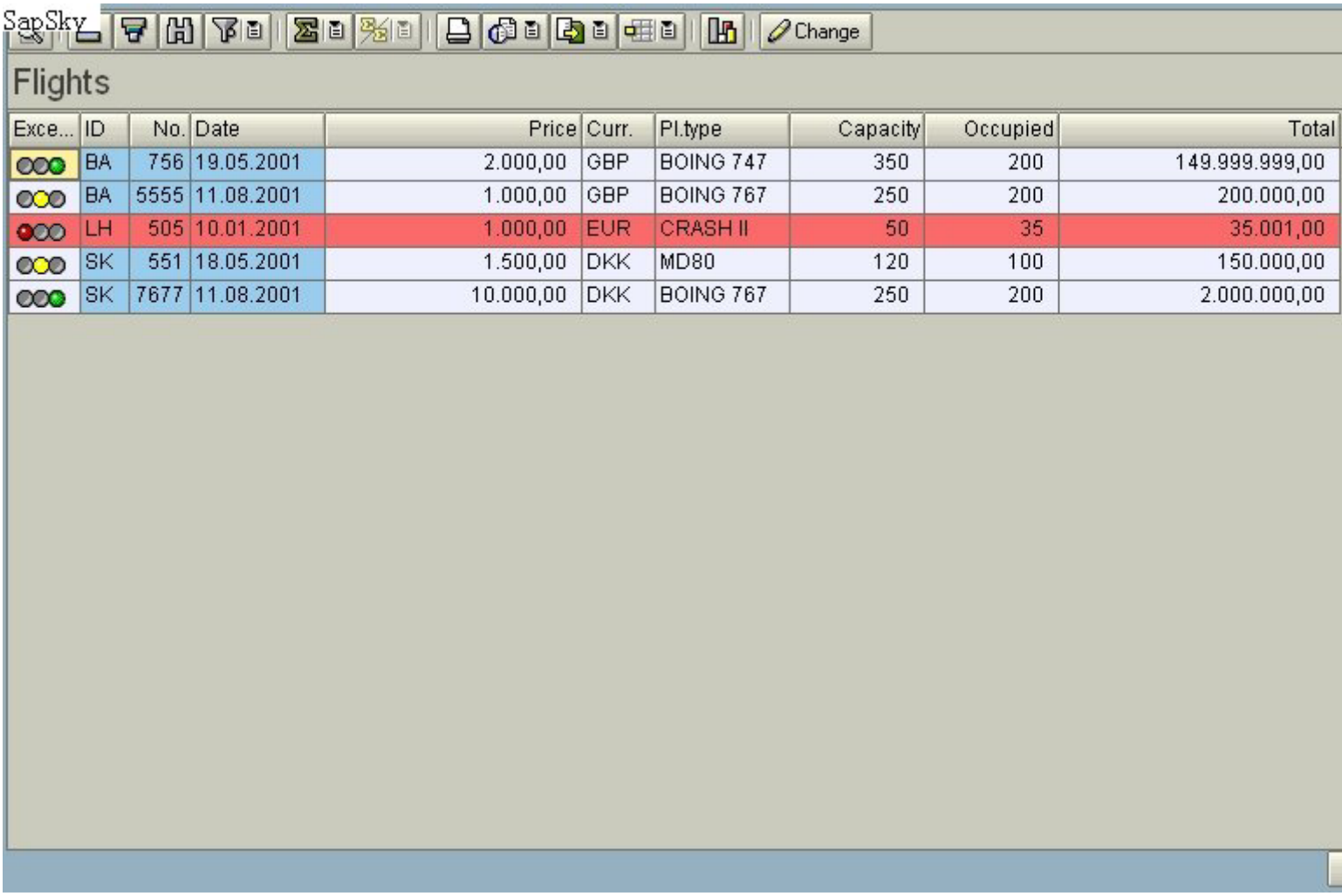
- How to setup the ALV grid
- How to ste focus to the grid
- How to set the title of the grid

- How to allow a user to save and reuse a grid layout (Variant)
- How to customize the ALV grid toolbar
- Refresh the grid
- Set and get row selection and read line contents
- Make and exception field (Traffic light)
- Coloring a line

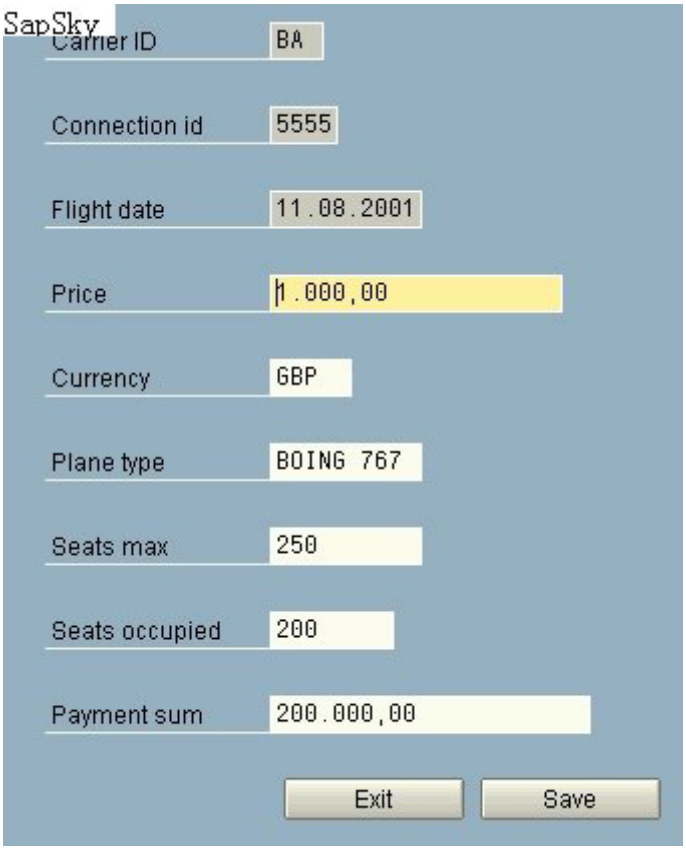
Steps:

- Create screen 100 with the ALV grid. Remember to include an exit button
- Add a change button to the ALV grid toolbar
- Create screen 200 the Change screen

The screens:



Exce...	ID	No.	Date	Price	Curr.	Pl.type	Capacity	Occupied	Total
	BA	756	19.05.2001	2.000,00	GBP	BOING 747	350	200	149.999.999,00
	BA	5555	11.08.2001	1.000,00	GBP	BOING 767	250	200	200.000,00
	LH	505	10.01.2001	1.000,00	EUR	CRASH II	50	35	35.001,00
	SK	551	18.05.2001	1.500,00	DKK	MD80	120	100	150.000,00
	SK	7677	11.08.2001	10.000,00	DKK	BOING 767	250	200	2.000.000,00



The code:

```
    REPORT sapmz_hf_alv_grid .
* Type pool for icons - used in the toolbar
    TYPE-POOLS: icon.
    TABLES: zsflight.
    * To allow the declaration of o_event_receiver before the
    * lcl_event_receiver class is defined, decale it as deferred in the
    * start of the program
    CLASS lcl_event_receiver DEFINITION DEFERRED.
    *---------------------------------------------------------------------
    * G L O B A L   I N T E R N A L   T A B L E S
    *---------------------------------------------------------------------
    *DATA: gi_sflight TYPE STANDARD TABLE OF sflight.
    * To include a traffic light and/or color a line the structure of the
    * table must include fields for the traffic light and/or the color
    TYPES: BEGIN OF st_sflight.
                INCLUDE STRUCTURE zsflight.
    *           Field for traffic light
    TYPES:   traffic_light TYPE c.
    *           Field for line color
```

```
types:   line_color(4) type c.
TYPES: END OF st_sflight.
TYPES: tt_sflight TYPE STANDARD TABLE OF st_sflight.
DATA: gi_sflight TYPE tt_sflight.
*---------------------------------------------------------------------
* G L O B A L    D A T A
*---------------------------------------------------------------------
DATA: ok_code        LIKE sy-ucomm,
*       Work area for internal table
        g_wa_sflight      TYPE st_sflight,
*       ALV control: Layout structure
        gs_layout          TYPE lvc_s_layo.
* Declare reference variables to the ALV grid and the container
DATA:
   go_grid                   TYPE REF TO cl_gui_alv_grid,
   go_custom_container TYPE REF TO cl_gui_custom_container,
   o_event_receiver      TYPE REF TO lcl_event_receiver.
DATA:
* Work area for screen 200
   g_screen200 LIKE zsflight.
* Data for storing information about selected rows in the grid
DATA:
* Internal table
   gi_index_rows TYPE lvc_t_row,
* Information about 1 row
   g_selected_row LIKE lvc_s_row.
*---------------------------------------------------------------------
* C L A S S E S
*---------------------------------------------------------------------
CLASS lcl_event_receiver DEFINITION.
   PUBLIC SECTION.
      METHODS:
        handle_toolbar FOR EVENT toolbar OF cl_gui_alv_grid
          IMPORTING
             e_object e_interactive,
        handle_user_command FOR EVENT user_command OF cl_gui_alv_grid
```

```
          IMPORTING e_ucomm.
ENDCLASS.
*---------------------------------------------------------------------*
*          CLASS lcl_event_receiver IMPLEMENTATION
*---------------------------------------------------------------------*
CLASS lcl_event_receiver IMPLEMENTATION.
  METHOD handle_toolbar.
*  Event handler method for event toolbar.
    CONSTANTS:
*  Constants for button type
      c_button_normal              TYPE i VALUE 0,
      c_menu_and_default_button TYPE i VALUE 1,
      c_menu                        TYPE i VALUE 2,
      c_separator                  TYPE i VALUE 3,
      c_radio_button               TYPE i VALUE 4,
      c_checkbox                   TYPE i VALUE 5,
      c_menu_entry                 TYPE i VALUE 6.
    DATA:
        ls_toolbar   TYPE stb_button.
*   Append seperator to the normal toolbar
    CLEAR ls_toolbar.
    MOVE c_separator TO ls_toolbar-butn_type..
    APPEND ls_toolbar TO e_object->mt_toolbar.
*   Append a new button that to the toolbar. Use E_OBJECT of
*   event toolbar. E_OBJECT is of type CL_ALV_EVENT_TOOLBAR_SET.
*   This class has one attribute MT_TOOLBAR which is of table type
*   TTB_BUTTON. The structure is STB_BUTTON
    CLEAR ls_toolbar.
    MOVE 'CHANGE'          TO ls_toolbar-function.
    MOVE  icon_change      TO ls_toolbar-icon.
    MOVE 'Change flight' TO ls_toolbar-quickinfo.
    MOVE 'Change'          TO ls_toolbar-text.
    MOVE ' '               TO ls_toolbar-disabled.
    APPEND ls_toolbar      TO e_object->mt_toolbar.
  ENDMETHOD.
```

```
  METHOD handle_user_command.
*    Handle own functions defined in the toolbar
     CASE e_ucomm.
       WHEN 'CHANGE'.
         PERFORM change_flight.
*          LEAVE TO SCREEN 0.
     ENDCASE.
  ENDMETHOD.
ENDCLASS.
*---------------------------------------------------------------------
* S T A R T - O F - S E L E C T I O N.
*---------------------------------------------------------------------
START-OF-SELECTION.
  SET SCREEN '100'.
*&---------------------------------------------------------------------*
*&        Module  USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
MODULE user_command_0100 INPUT.
  CASE ok_code.
    WHEN 'EXIT'.
      LEAVE TO SCREEN 0.
  ENDCASE.
ENDMODULE.                          " USER_COMMAND_0100  INPUT
*&---------------------------------------------------------------------*
*&        Module  STATUS_0100  OUTPUT
*&---------------------------------------------------------------------*
MODULE status_0100 OUTPUT.
  DATA:
*    For parameter IS_VARIANT that is sued to set up options for storing
*    the grid layout as a variant in method set_table_for_first_display
     l_layout TYPE disvariant,
*    Utillity field
     l_lines TYPE i.
* After returning from screen 200 the line that was selected before
* going to screen 200, should be selected again. The table gi_index_rows
* was the output table from the GET_SELECTED_ROWS method in form
```

```
* CHANGE_FLIGHT
  DESCRIBE TABLE gi_index_rows LINES l_lines.
  IF l_lines > 0.
    CALL METHOD go_grid->set_selected_rows
         EXPORTING
           it_index_rows = gi_index_rows.
    CALL METHOD cl_gui_cfw=>flush.
    REFRESH gi_index_rows.
  ENDIF.
* Read data and create objects
  IF go_custom_container IS INITIAL.
*   Read data from datbase table
    PERFORM get_data.
*   Create objects for container and ALV grid
    CREATE OBJECT go_custom_container
      EXPORTING container_name = 'ALV_CONTAINER'.
    CREATE OBJECT go_grid
      EXPORTING
        i_parent = go_custom_container.
*   Create object for event_receiver class
*   and set handlers
    CREATE OBJECT o_event_receiver.
    SET HANDLER o_event_receiver->handle_user_command FOR go_grid.
    SET HANDLER o_event_receiver->handle_toolbar FOR go_grid.
*   Layout (Variant) for ALV grid
    l_layout-report = sy-repid. "Layout fo report
*-------------------------------------------------------------
* Setup the grid layout using a variable of structure lvc_s_layo
*-------------------------------------------------------------
*   Set grid title
    gs_layout-grid_title = 'Flights'.
*   Selection mode - Single row without buttons
*   (This is the default  mode
    gs_layout-sel_mode = 'B'.
*   Name of the exception field (Traffic light field) and the color
*   field + set the exception and color field of the table
```

```
    gs_layout-excp_fname = 'TRAFFIC_LIGHT'.
    gs_layout-info_fname = 'LINE_COLOR'.
    LOOP AT gi_sflight INTO g_wa_sflight.
      IF g_wa_sflight-paymentsum < 100000.
*        Value of traffic light field
         g_wa_sflight-traffic_light = '1'.
*        Value of color field:
*        C = Color, 6=Color 1=Intesified on, 0: Inverse display off
     g_wa_sflight-line_color      = 'C610'.
      ELSEIF g_wa_sflight-paymentsum => 100000 AND
               g_wa_sflight-paymentsum < 1000000.
         g_wa_sflight-traffic_light = '2'.
      ELSE.
         g_wa_sflight-traffic_light = '3'.
   ENDIF.
      MODIFY gi_sflight FROM g_wa_sflight.
    ENDLOOP.
*   Grid setup for first display
    CALL METHOD go_grid->set_table_for_first_display
      EXPORTING i_structure_name = 'SFLIGHT'
                is_variant          = l_layout
                i_save                = 'A'
                is_layout           = gs_layout
      CHANGING   it_outtab            = gi_sflight.
*-- End of grid setup ----------------------------------------
*   Raise event toolbar to show the modified toolbar
    CALL METHOD go_grid->set_toolbar_interactive.
*   Set focus to the grid. This is not necessary in this
*   example as there is only one control on the screen
    CALL METHOD cl_gui_control=>set_focus EXPORTING control = go_grid.
  ENDIF.
ENDMODULE.                          " STATUS_0100   OUTPUT
*&---------------------------------------------------------------------*
*&        Module   USER_COMMAND_0200   INPUT
*&---------------------------------------------------------------------*
MODULE user_command_0200 INPUT.
```

```
  CASE ok_code.
    WHEN 'EXIT200'.
      LEAVE TO SCREEN 100.
      WHEN'SAVE'.
      PERFORM save_changes.
  ENDCASE.
ENDMODULE.                          " USER_COMMAND_0200  INPUT
*&---------------------------------------------------------------------*
*&        Form  get_data
*&---------------------------------------------------------------------*
FORM get_data.
* Read data from table SFLIGHT
  SELECT *
    FROM zsflight
    INTO TABLE gi_sflight.
ENDFORM.                            " load_data_into_grid
*&---------------------------------------------------------------------*
*&        Form  change_flight
*&---------------------------------------------------------------------*
*  Reads the contents of the selected row in the grid, ans transfers
*  the data to screen 200, where it can be changed and saved.
*---------------------------------------------------------------------*
FORM change_flight.
  DATA:l_lines TYPE i.
  REFRESH gi_index_rows.
  CLEAR    g_selected_row.
* Read index of selected rows
  CALL METHOD go_grid->get_selected_rows
    IMPORTING
      et_index_rows = gi_index_rows.
* Check if any row are selected at all. If not
* table  gi_index_rows will be empty
  DESCRIBE TABLE gi_index_rows LINES l_lines.
  IF l_lines = 0.
    CALL FUNCTION 'POPUP_TO_DISPLAY_TEXT'
        EXPORTING
```

```
                    textline1 = 'You must choose a line'.
      EXIT.
    ENDIF.
* Read indexes of selected rows. In this example only one
* row can be selected as we are using gs_layout-sel_mode = 'B',
* so it is only ncessary to read the first entry in
* table gi_index_rows
    LOOP AT gi_index_rows INTO g_selected_row.
      IF sy-tabix = 1.
       READ TABLE gi_sflight INDEX g_selected_row-index INTO g_wa_sflight.
      ENDIF.
    ENDLOOP.
* Transfer data from the selected row to screenm 200 and show
* screen 200
    CLEAR g_screen200.
    MOVE-CORRESPONDING g_wa_sflight TO g_screen200.
    LEAVE TO SCREEN '200'.
ENDFORM.                              " change_flight
*&---------------------------------------------------------------------*
*&        Form   save_changes
*&---------------------------------------------------------------------*
* Changes made in screen 200 are written to the datbase table
* zsflight, and to the grid table gi_sflight, and the grid is
* updated with method refresh_table_display to display the changes
*----------------------------------------------------------------------*
FORM save_changes.
    DATA: l_traffic_light TYPE c.
* Update traffic light field
* Update database table
    MODIFY zsflight FROM g_screen200.
* Update grid table , traffic light field and color field.
* Note that it is necessary to use structure g_wa_sflight
* for the update, as the screen structure does not have a
* traffic light field
    MOVE-CORRESPONDING g_screen200 TO g_wa_sflight.
    IF g_wa_sflight-paymentsum < 100000.
```

```
      g_wa_sflight-traffic_light = '1'.
*     C = Color, 6=Color 1=Intesified on, 0: Inverse display off
      g_wa_sflight-line_color      = 'C610'.
    ELSEIF g_wa_sflight-paymentsum => 100000 AND
           g_wa_sflight-paymentsum < 1000000.
      g_wa_sflight-traffic_light = '2'.
      clear g_wa_sflight-line_color.
    ELSE.
      g_wa_sflight-traffic_light = '3'.
      clear g_wa_sflight-line_color.
    ENDIF.
    MODIFY gi_sflight INDEX g_selected_row-index FROM g_wa_sflight.
* Refresh grid
    CALL METHOD go_grid->refresh_table_display.
    CALL METHOD cl_gui_cfw=>flush.
    LEAVE TO SCREEN '100'.
  ENDFORM.                          " save_changes
```

Reading attribute of a BOR (Business Object) in ABAP

[日期：2006-10-18] 来源：sap-img 作者：sapsky [字体：大 中 小]

Reading attribute of a BOR (Business Object) in ABAP

Many times I've looked at transaction SWO1 for a business object and seen how it displays all sorts of useful functions (e.g Line Item Texts etc. etc.). Many of these attributes can only be got at in normal programming via convoluted abap tinkering. -- How many times for example do you want to get texts to include on a smartform etc.

However it's really easy to get the attribute from the BOR itself using very simple coding.

Most people shy away from this since a BOR is usually used in SAP workflow and most abap developers haven't had a lot of exposure to OO programming or SAP workflow.

Anyway here's sample code to return an Attribute of a BOR.

```
program zzreadbor.

* Get an attribute of a business object.
```

```
parameters: p_busobj(10) type c default 'BUSISM007',  "IS Media
customer
            p_key(70) type c,
            p_attr(32) type c default 'Xmediacustomer'.
data:
    i_objtype TYPE swo_objtyp,
    i_objkey TYPE swo_typeid,
    i_element TYPE swo_verb.
     DATA object TYPE swo_objhnd.
    DATA verb TYPE swo_verb.
    DATA return TYPE swotreturn.
 DATA lt_container TYPE STANDARD TABLE OF swcont.
 data line type swcont.

i_objtype = p_busobj.
i_element = p_attr.
i_objkey = p_key.
*  instantiate the business object. I.e give it a key and create it.
  CALL FUNCTION 'SWO_CREATE'
    EXPORTING
      objtype = i_objtype
      objkey  = i_objkey
    IMPORTING
      object  = object.

* return attribute.
  CALL FUNCTION 'SWO_INVOKE'
    EXPORTING
      access     = 'G'
      object     = object
      verb       = i_element

    IMPORTING
      return     = return
      verb       = verb
```

```
    TABLES
      container  = lt_container.
* the attribute value is returned in field line-value.
  IF return-code = 0.
  loop at lt_container into line.
  write: / line-value.
  endloop.
endif.
```

* The above example will return an 'X' if the customer is an IS Media customer or blank otherwise.

Other good business objects to use are sales docs (header / line items), invoices, Info types etc etc.

You can also call methods this way as well -- change the access to 'C' in the SWO_INVOKE and pass the parameters to the method. Method name is in VERB. A bit more complex as in the case of Supertypes you need to find the program -- will post an example later --however to start with even the attributes are useful since on "Instantiation" of the BOR you have access to ALL the attribites of that BOR.

To Use BADI - Business Add In you need to Understand ABAP OO Interface Concept

* For Rajat's OO ans BAdI Education

```
Report Z_CTRY.

* A Shape Interface "Like a shape" "Behaves like a Shape"
Adverb/Adjective

Interface IShape.
Methods:  getArea Returning Value(area) Type F,
          getCircumference Returning Value(circumference) Type F.
Endinterface.

* A Circle Class that behaves like a Shape
Class CCircle Definition.

    Public Section.
```

```
        Interfaces IShape.

        Aliases:   getArea For IShape~getArea,
                   getCircumference For IShape~getCircumference.

        Methods:   Constructor Importing pRadius Type F,
                   setRadius   Importing pRadius Type F,
                   getRadius   Returning Value(pRadius) Type F.

    Private Section.
        Data radius Type F.
        Constants PI Type F Value '3.141592365359'.
EndClass.

Class CCircle Implementation.

    Method Constructor.
        radius = pRadius.
    EndMethod.

    Method setRadius.
        radius = pRadius.
    EndMethod.

    Method getRadius.
        pRadius = radius.
    EndMethod.

    Method IShape~getArea.
        Compute area = 2 * PI * radius.
    EndMethod.

    Method IShape~getCircumference.
        Compute circumference = PI * radius * radius.
    EndMethod.
```

```
EndClass.

* A Square Class
Class CSquare Definition.

    Public Section.

        Interfaces IShape.

        Aliases:   getArea For IShape~getArea,
                   getCircumference For IShape~getCircumference.

        Methods:   Constructor Importing pSide Type F.

    Private Section.
        Data side Type F.
EndClass.

Class CSquare Implementation.

    Method Constructor.
        side = pSide.
    EndMethod.

    Method IShape~getArea.
        Compute area = side * side.
    EndMethod.

    Method IShape~getCircumference.
        Compute circumference = 4 * side.
    EndMethod.

EndClass.

* A Rectangle Class
Class CRectangle Definition.
```

```
    Public Section.

        Interfaces IShape.

        Aliases:   getArea For IShape~getArea,
                   getCircumference For IShape~getCircumference.

        Methods:   Constructor Importing pHeight Type F
                                         pLength Type F.

    Private Section.
        Data: height Type F,
              length Type F.
EndClass.

Class CRectangle Implementation.

    Method Constructor.
        height = pHeight.
        length = pLength.
    EndMethod.

    Method IShape~getArea.
        Compute area = height * length.
    EndMethod.

    Method IShape~getCircumference.
        Compute circumference = 2 * ( height + length ).
    EndMethod.

EndClass.

* START of PROGRAM

* Array of Shapes
```

```
Data:  OneShape   Type Ref To IShape,             " One Object with
Shape Behaviour
       ShapeTable Type Table Of Ref To IShape.    " Array of Objects
with Shape Behaviour

* Concrete Objects with IShape Behaviour!
Data:  C1 Type Ref To CCircle,
       S1 Type Ref To CSquare,
       R1 Type Ref To CRectangle,
       C2 Type Ref To CCircle,
       S2 Type Ref To CSquare,
       R2 Type Ref To CRectangle.


Data:  descr_ref  TYPE ref to CL_ABAP_TYPEDESCR,
       ClassName  Type String,
       Serial     Type I.

Data:  myArea          Type F,
       myCircumference Type F.

START-OF-SELECTION.
    Create Object C1 Exporting pRadius = '2.5'.
    Create Object C2 Exporting pRadius = '5.0'.

    Create Object S1 Exporting pSide = '3.5'.
    Create Object S2 Exporting pSide = '6.0'.

    Create Object R1 Exporting pHeight = '2.8' pLength = '3.4'.
    Create Object R2 Exporting pHeight = '1.7' pLength = '6.3'.

* Append in any order!
    Append S1  to ShapeTable.
    Append R2  to ShapeTable.
    Append R1  to ShapeTable.
    Append C2  to ShapeTable.
```

```
    Append C1  to ShapeTable.
    Append S2  to ShapeTable.

    Serial = 0.

    Loop At ShapeTable into OneShape.
       Call Method OneShape->getArea
                            Receiving area = myArea.
       Call Method OneShape->getCircumference
                            Receiving circumference = myCircumference.

       descr_ref = CL_ABAP_TYPEDESCR=>Describe_By_Object_Ref( OneShape
).
       Call Method descr_ref->get_relative_name
                   Receiving P_RELATIVE_NAME = ClassName.

       Add 1 to Serial.

       Write: / Serial,  ClassName.
       Write: / 'Area          ',  myArea          Decimals 4 Exponent
0.
       Write: / 'Circumference ',  myCircumference Decimals 4 Exponent
0.
       Write: /.

    EndLoop.

** Results
*           1  CSQUARE
*  Area                            12.2500
*  Circumference                   14.0000
*
*           2  CRECTANGLE
*  Area                            10.7100
*  Circumference                   16.0000
*
```

```
*          3  CRECTANGLE
*  Area                              9.5200
*  Circumference                    12.4000
*
*          4  CCIRCLE
*  Area                             31.4159
*  Circumference                    78.5398
*
*          5  CCIRCLE
*  Area                             15.7080
*  Circumference                    19.6350
*
*          6  CSQUARE
*  Area                             36.0000
*  Circumference                    24.0000
```


Splash Screen in ABAP using OO

With good tips from top SDN contributor

Thomas Jung

I made 2 function modules to suit my whims.

SAP being a serious Businesss Software you cannot have too many JPGs floating around! One or two is fun.

In Function group uou need two screens 0806 & 2009 which are essentially blank.

I put 2 title Bars - 0806 "SAP - JOB in Progress"; 2009 - "SAP - JOB OVER!!"

Code listing for function: ZJNC_START_SPLASH

Description: Show Splash at Start

```
FUNCTION zjnc_start_splash.
*"----------------------------------------------------------------------
```

```
*"*"Local interface:
*"  IMPORTING
*"     REFERENCE(IMAGEFILE) TYPE  C DEFAULT 'THANKS.JPG'
*"     REFERENCE(WIDTH) TYPE  I DEFAULT 415
*"     REFERENCE(HEIGHT) TYPE  I DEFAULT 274
*"     REFERENCE(TIMEOUT) TYPE  I DEFAULT 3
*"     REFERENCE(CALLBACK) TYPE  C
*"----------------------------------------------------------------------

*       Global data declarations
  MOVE imagefile TO g_name.
  MOVE width TO picwidth.
  MOVE height TO picheight.
  MOVE timeout TO pictimeout.
  MOVE callback TO piccallback.
  TRANSLATE piccallback TO UPPER CASE.

  PERFORM getpicurl.

  CALL SCREEN 0806.

ENDFUNCTION.
```

Code listing for function: ZJNC_END_SPLASH

Description: Show Splash at End

```
FUNCTION ZJNC_END_SPLASH.
*"----------------------------------------------------------------------
*"*"Local interface:
*"  IMPORTING
*"     REFERENCE(IMAGEFILE) TYPE  C DEFAULT 'THANKS.JPG'
*"     REFERENCE(WIDTH) TYPE  I DEFAULT 415
*"     REFERENCE(HEIGHT) TYPE  I DEFAULT 274
*"     REFERENCE(TIMEOUT) TYPE  I DEFAULT 3
```

```
*"----------------------------------------------------------------------

*       Global data declarations
  MOVE imagefile TO g_name.
  MOVE width TO picwidth.
  MOVE height TO picheight.
  MOVE timeout TO pictimeout.

  PERFORM getpicurl.

  CALL SCREEN 2009.

ENDFUNCTION.
```

Code listing for: LZUTILTOP

```
* TOP level Include of Function Group ZUTIL

*  Author Jayanta Narayan Choudhuri
*         Flat 302
*         395 Jodhpur Park
*         Kolkata 700 068
*       Email sss@cal.vsnl.net.in
*       URL:  http://www.geocities.com/ojnc

FUNCTION-POOL zutil.                        "MESSAGE-ID ..

TYPE-POOLS: abap.

DATA: graphic_url(255),
      g_result   TYPE i,
      g_linesz   TYPE i,
      g_filesz   TYPE i,
      g_name(100).
```

```
TYPES: t_graphic_line(256) TYPE x.

DATA: graphic_line TYPE t_graphic_line,
      graphic_table TYPE TABLE OF t_graphic_line.

DATA: picwidth        TYPE i,
      picheight       TYPE i,
      pictimeout      TYPE i,
      piccallback(60) TYPE c,
      first           TYPE boolean.



*---------------------------------------------------------------------*
*       CLASS ZCL_ES_SPLASH_SCREEN  DEFINITION
*---------------------------------------------------------------------*
CLASS zcl_es_splash_screen DEFINITION.

  PUBLIC SECTION.
    EVENTS on_close.

    METHODS constructor
      IMPORTING
        !i_num_secs  TYPE i DEFAULT 5
        !i_url       TYPE c
        !i_width     TYPE i
        !i_height    TYPE i.

  PROTECTED SECTION.

    METHODS handle_end_of_timer
      FOR EVENT finished OF cl_gui_timer.
  PRIVATE SECTION.

    DATA container TYPE REF TO cl_gui_dialogbox_container.
    DATA image     TYPE REF TO cl_gui_picture.
    DATA timer     TYPE REF TO cl_gui_timer.
```

```
ENDCLASS.                    "ZCL_ES_SPLASH_SCREEN  DEFINITION

*---------------------------------------------------------------------*
* CLASS ZCL_ES_SPLASH_SCREEN IMPLEMENTATION
*---------------------------------------------------------------------*
CLASS zcl_es_splash_screen IMPLEMENTATION.
  METHOD constructor.

    DATA: image_width     TYPE i,
          image_height    TYPE i.

    COMPUTE image_width  = i_width + 30.
    COMPUTE image_height = i_height + 50.

    CREATE OBJECT container
       EXPORTING
         width                       = 10
         height                      = 10
         top                         = 10
         left                        = 10
         name                        = 'DialogSplash'.


    CALL METHOD container->set_caption
      EXPORTING
        caption = g_name.

    CREATE OBJECT image
      EXPORTING
        parent = container.

    CALL METHOD image->load_picture_from_url
      EXPORTING
        url = i_url.
```

```
image->set_display_mode( image->display_mode_normal_center ).

cl_gui_cfw=>flush( ).

container->set_metric( EXPORTING metric = image->metric_pixel ).

DATA: myleft TYPE i,
      mytop  TYPE i.

COMPUTE myleft = ( 800 - image_width ) / 2.
COMPUTE mytop  = ( 600 - image_height ) / 2.

IF myleft < 0.
  MOVE 0 TO myleft.
ENDIF.

IF mytop < 0.
  MOVE 0 TO mytop.
ENDIF.

container->set_position(
  EXPORTING
    height            = image_height
    left              = myleft
    top               = mytop
    width             = image_width ).

cl_gui_cfw=>update_view( ).

CREATE OBJECT timer.
timer->interval = i_num_secs.

SET HANDLER me->handle_end_of_timer FOR timer.
timer->run( ).
cl_gui_cfw=>flush( ).
```

```
  ENDMETHOD.                      "constructor

  METHOD handle_end_of_timer.

* I wanted NAMASTE to remain until JOB was complete.
*    IF container IS NOT INITIAL.
*      container->free( ).
*      CLEAR container.
*      FREE  container.
*    ENDIF.
*
*    IF timer IS NOT INITIAL.
*      timer->free( ).
*      CLEAR timer.
*      FREE  timer.
*    ENDIF.
*
*    cl_gui_cfw=>flush( ).

    RAISE EVENT on_close.

  ENDMETHOD.                      "handle_end_of_timer
ENDCLASS.                      "ZCL_ES_SPLASH_SCREEN  IMPLEMENTATION

*---------------------------------------------------------------------*
*       CLASS lcl_event_handler DEFINITION
*---------------------------------------------------------------------*
CLASS lcl_event_handler DEFINITION.

  PUBLIC SECTION.
    CLASS-METHODS: on_close FOR EVENT on_close OF zcl_es_splash_screen.
ENDCLASS. "lcl_event_handler DEFINITION

*---------------------------------------------------------------------*
* CLASS lcl_event_handler IMPLEMENTATION
*---------------------------------------------------------------------*
```

```
CLASS lcl_event_handler IMPLEMENTATION.
  METHOD on_close.
    IF sy-dynnr = 2009.
      LEAVE PROGRAM.
    ELSE.
      MOVE abap_false TO first.
      PERFORM (piccallback) IN PROGRAM (sy-cprog).
    ENDIF.
  ENDMETHOD. "on_close
ENDCLASS. "lcl_event_handler IMPLEMENTATION

DATA: splash TYPE REF TO zcl_es_splash_screen.

*&---------------------------------------------------------------------*
*&      Module  STATUS_0806  OUTPUT
*&---------------------------------------------------------------------*
MODULE status_0806 OUTPUT.
  IF first IS INITIAL.
    first = abap_true.

    SET TITLEBAR 'TITLE0806'.

    CREATE OBJECT splash
        EXPORTING
          i_num_secs = pictimeout
          i_url      = graphic_url
          i_width    = picwidth
          i_height   = picheight.

    SET HANDLER lcl_event_handler=>on_close FOR splash.

  ENDIF.
ENDMODULE.                 " STATUS_0806  OUTPUT

*&---------------------------------------------------------------------*
*&      Module  STATUS_2009  OUTPUT
```

```
*&---------------------------------------------------------------------*
MODULE status_2009 OUTPUT.
  IF first IS INITIAL.
    first = abap_true.

    SET TITLEBAR 'TITLE2009'.

    CREATE OBJECT splash
        EXPORTING
          i_num_secs = pictimeout
          i_url      = graphic_url
          i_width    = picwidth
          i_height   = picheight.

    SET HANDLER lcl_event_handler=>on_close FOR splash.

  ENDIF.
ENDMODULE.                    " STATUS_2009  OUTPUT

*&---------------------------------------------------------------------*
*&      Form  getpicurl
*&---------------------------------------------------------------------*
FORM getpicurl.

  OPEN DATASET g_name FOR INPUT IN BINARY MODE.

  REFRESH graphic_table.
  CLEAR   g_filesz.

  DO.
    CLEAR graphic_line.
    READ DATASET g_name INTO graphic_line ACTUAL LENGTH g_linesz.

    ADD g_linesz TO g_filesz.

    APPEND graphic_line TO graphic_table.
```

```
    IF sy-subrc <> 0.
      EXIT.
    ENDIF.

  ENDDO.

  CLOSE DATASET g_name.

  CLEAR graphic_url.

  CALL FUNCTION 'DP_CREATE_URL'
    EXPORTING
      type                  = 'IMAGE'
      subtype               = 'GIF'
    TABLES
      data                  = graphic_table
    CHANGING
      url                   = graphic_url
    EXCEPTIONS
      dp_invalid_parameter = 1
      dp_error_put_table   = 2
      dp_error_general     = 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.
    EXIT.
  ENDIF.

ENDFORM.                      "getpicurl
```


ABAP Object Oriented SpreadSheet with "Unlimited" Power

Content Author: Jayanta Narayan Choudhuri

Author Email: sss@cal.vsnl.net.in

Author Website: http://www.geocities.com/ojnc

Objective: SAP On-Line HELP has a section - "Controls and Control Framework (BC-CI)".

Under this refer "Desktop Office Integration (BC-CI)"

In that section read "The Spreadsheet Interface" thoroughly.

The ides is that once a programmer gets hold of a SpreadSheetInterface Object he/she can use the powerful methods to populate Excel in any way setting sheets, ranges, colours, fonts and ofcourse content.

Create a Function Group ZUTIL

Paste TOP Level code into LZUTILTOP

Create 4 Functionn Modules

ZJNC_START_EXCEL.

ZJNC_ADD_SHEET.

ZJNC_ADD_RANGE.

ZJNC_ADD_TABLE.

ZJNC_START_EXCEL - uses the "secret" screen 2307 which a user does not even see to get hold of a Spreadsheet Interface handle. With this alone a user has virtually unlimited power as he she can call all the methods.

But to make life easier I created 4 simple functions:

ZJNC_ADD_SHEET adds a sheet to a work book

ZJNC_ADD_RANGE adds a range to a sheet

ZJNC_ADD_TABLE adds a internal table to a range with specification of all properties like font colour size bold italic etc. In ABAP Objects, you can only declare tables without headers. Hence TABLE[] syntax ensures Header is Stripped.

It is best to have full geometry in mind and fill in the following sequence

For each SHEET Create 1 RANGE & Populate Data immediately

For each SHEET Reapeat for all Ranges

Before creating a range you will need to consider size based on table.

The no. of Rows & Columns will decide size.

The cumulative rows will gixe the corner co-ordinates.

Attached Files:

ZJNCEXCEL_Test.ab4 is the Test Program

ZJNCEXCEL_FUNC.ab4 is the Function Group

ZEXCEL_WRITEUP.txt is this write-up


```
*  Author Jayanta Narayan Choudhuri
*         Flat 302
*         395 Jodhpur Park
*         Kolkata 700 068
*       Email sss@cal.vsnl.net.in
*       URL:  http://www.geocities.com/ojnc
```


```
*-----------------------------------------------------------------------------------------
-
* Screen 2307 has only 1 Custom Control MYCONTROL
* Screen 2307 Flow Logic

PROCESS BEFORE OUTPUT.
  MODULE ZJNCPBO.
*
PROCESS AFTER INPUT.
* MODULE ZJNCPAI.

1 Custom Control MYCONTROL
OK ZJNC_OK_CODE

*-----------------------------------------------------------------------------------------
-
```

```
FUNCTION ZJNC_START_EXCEL.
*"----------------------------------------------------------------------
*"*"Local interface:
*"  EXPORTING
*"     REFERENCE(SPREADSHEETINTF) TYPE REF TO  I_OI_SPREADSHEET
*"----------------------------------------------------------------------

Move SY-REPID to ZJNC_REPID.
CALL SCREEN 2307.

spreadsheetintf = zjncspreadsheet.

ENDFUNCTION.

FUNCTION ZJNC_ADD_SHEET.
*"----------------------------------------------------------------------
*"*"Local interface:
*"  IMPORTING
*"     REFERENCE(PSHEET) TYPE  C
*"     REFERENCE(SPREADSHEETINTF) TYPE REF TO  I_OI_SPREADSHEET
*"----------------------------------------------------------------------

    Move SY-REPID to ZJNC_REPID.

    CALL METHOD SPREADSHEETINTF->add_sheet
           EXPORTING name     = psheet
                     no_flush  = ' '
           IMPORTING error     = zjncerror
                     retcode   = zjncretcode.

ENDFUNCTION.

FUNCTION ZJNC_ADD_RANGE.
*"----------------------------------------------------------------------
*"*"Local interface:
*"  IMPORTING
```

```
*"     REFERENCE(PRANGE) TYPE  C
*"     REFERENCE(STARTROW) TYPE  I
*"     REFERENCE(STARTCOL) TYPE  I
*"     REFERENCE(NUMROWS) TYPE  I
*"     REFERENCE(NUMCOLS) TYPE  I
*"     REFERENCE(PSHEET) TYPE  C
*"     REFERENCE(SPREADSHEETINTF) TYPE REF TO  I_OI_SPREADSHEET
*"----------------------------------------------------------------------


    Move SY-REPID to zjnc_repid.

    CALL METHOD SPREADSHEETINTF->select_sheet
           EXPORTING name      = psheet
           no_flush  = ' '
           IMPORTING error     = zjncerror
                     retcode   = zjncretcode.

    CALL METHOD SPREADSHEETINTF->set_selection
           EXPORTING top       = StartRow
                     left      = StartCol
                     rows      = 1
                     columns   = 1
                     no_flush  = ' '
           IMPORTING error     = zjncerror
                     retcode   = zjncretcode.

    CALL METHOD SPREADSHEETINTF->insert_range
           EXPORTING name      = prange
                     rows      = numRows
                     columns   = numCols
                     no_flush  = ' '
           IMPORTING error     = zjncerror
                     retcode   = zjncretcode.
```

```
ENDFUNCTION.

FUNCTION ZJNC_ADD_TABLE.
*"----------------------------------------------------------------------
*"*"Local interface:
*"  IMPORTING
*"     REFERENCE(PTABLE) TYPE  TABLE
*"     REFERENCE(PRANGE) TYPE  C
*"     REFERENCE(PSIZE) TYPE  I DEFAULT -1
*"     REFERENCE(PBOLD) TYPE  I DEFAULT -1
*"     REFERENCE(PITALIC) TYPE  I DEFAULT -1
*"     REFERENCE(PALIGN) TYPE  I DEFAULT -1
*"     REFERENCE(PFRONT) TYPE  I DEFAULT -1
*"     REFERENCE(PBACK) TYPE  I DEFAULT -1
*"     REFERENCE(PFORMAT) TYPE  C DEFAULT 'NA'
*"     REFERENCE(SPREADSHEETINTF) TYPE REF TO  I_OI_SPREADSHEET
*"----------------------------------------------------------------------



** TYPES: SOI_zjnc_fields_table TYPE STANDARD TABLE OF RFC_FIELDS.
DATA: zjnc_fields_table   Type TABLE OF rfc_fields.
DATA: zjncwa_zjnc_fields_table TYPE     rfc_fields.

    Move SY-REPID to zjnc_repid.

    CALL FUNCTION 'DP_GET_FIELDS_FROM_TABLE'
         TABLES
              data   = ptable
              fields = zjnc_fields_table.

    CALL METHOD SPREADSHEETINTF->insert_one_table
           EXPORTING
*                    ddic_name    = ddic_name
                     data_table   = ptable
                     fields_table = zjnc_fields_table
                     rangename    = prange
```

```
                wholetable   = 'X'
                no_flush     = ' '
        IMPORTING error        = zjncerror
                retcode      = zjncretcode.

    CALL METHOD SPREADSHEETINTF->set_font
        EXPORTING rangename = prange
                family    = 'Arial'
                size      = psize
                bold      = pbold
                italic    = pitalic
                align     = palign
                no_flush  = ' '
        IMPORTING error     = zjncerror
                retcode   = zjncretcode.

    CALL METHOD SPREADSHEETINTF->set_color
        EXPORTING rangename = prange
                front     = pfront
                back      = pback
                no_flush  = ' '
        IMPORTING error     = zjncerror
                retcode   = zjncretcode.

    If pFormat <> 'NA'.
        CALL METHOD SPREADSHEETINTF->set_format_string
            EXPORTING rangename    = prange
                      formatstring = pformat
                      no_flush     = ' '
            IMPORTING error     = zjncerror
                      retcode   = zjncretcode.
    EndIf.

ENDFUNCTION.
```

```
*---------------------------------------------------------------------------------------
* TOP level Include of Function Group ZUTIL


FUNCTION-POOL ZUTIL.                          "MESSAGE-ID ..

* Global ZUTIL Data for ZJNCEXCEL
DATA zjnccontainer   TYPE REF TO cl_gui_custom_container.

DATA zjnccontrol     TYPE REF TO i_oi_container_control.

DATA zjncdocument    TYPE REF TO i_oi_document_proxy.

DATA zjncspreadsheet TYPE REF TO i_oi_spreadsheet.

DATA zjncerror       TYPE REF TO i_oi_error.
DATA zjncretcode     TYPE SOI_RET_STRING.

DATA zjncexcelsheet  TYPE soi_document_type VALUE
SOI_DOCTYPE_EXCEL_SHEET.

DATA:   zjnc_ok_code LIKE sy-ucomm,          " return code from screen
        zjnc_repid   LIKE sy-repid.


*********************************************************************
*   P B O
*********************************************************************
MODULE zjncpbo OUTPUT.

*    SET PF-STATUS 'ZJNCSTATUS'.
*    SET TITLEBAR  'ZJNCTITLE'.

    IF zjncdocument IS NOT INITIAL.
       RETURN.
    EndIf.
```

```
    Perform ZJNC_INIT_EXCEL.

    Leave to Screen 0.

ENDMODULE.                             " PBO


*&---------------------------------------------------------------------*
*&      Form  ZJNC_INIT_EXCEL
*&---------------------------------------------------------------------*
Form ZJNC_INIT_EXCEL.

    CALL METHOD c_oi_container_control_creator=>get_container_control
         IMPORTING control = zjnccontrol
                   error   = zjncerror.

    IF sy-subrc NE 0.
      CALL FUNCTION 'POPUP_TO_INFORM'
           EXPORTING
                titel = zjnc_repid
                txt2  = 'Create OLE zjnccontrol Failed'
                txt1  = 'to make Excel zjnccontrol'.
      Leave Program.
    ENDIF.

    CREATE OBJECT zjnccontainer
      EXPORTING
         CONTAINER_NAME = 'MYCONTROL'
      EXCEPTIONS
         CNTL_ERROR = 1
         CNTL_SYSTEM_ERROR = 2
         CREATE_ERROR = 3
         LIFETIME_ERROR = 4
         LIFETIME_DYNPRO_DYNPRO_LINK = 5.
```

```
    IF sy-subrc NE 0.
*      add your handling
    ENDIF.

    CALL METHOD zjnccontrol->init_control
         EXPORTING  r3_application_name      = 'R/3 Basis' "#EC NOTEXT
                    inplace_enabled          = 'X'
                    inplace_scroll_documents = 'X'
                    parent                   =  zjnccontainer
                    register_on_close_event  = 'X'
                    register_on_custom_event = 'X'
                    no_flush                 = 'X'
          IMPORTING error = zjncerror.

    IF sy-subrc NE 0.
      CALL FUNCTION 'POPUP_TO_INFORM'
           EXPORTING
                titel = zjnc_repid
                txt2  = 'INIT OLE zjnccontrol Failed'
                txt1  = 'to init Excel zjnccontrol'.
      Leave Program.
    ENDIF.

    CALL METHOD zjnccontrol->get_document_proxy
       EXPORTING document_type       = zjncexcelsheet
*                document_format     = document_format
*                register_container  = register_container
                 no_flush            = ' '
       IMPORTING document_proxy       = zjncdocument
                 retcode              = zjncretcode
                 error                = zjncerror.

    IF sy-subrc NE 0.
      CALL FUNCTION 'POPUP_TO_INFORM'
           EXPORTING
                titel = zjnc_repid
```

```
            txt2  = 'Create zjncdocument PROXY Failed'
            txt1  = 'to make Excel zjncdocument'.
      Leave Program.
    ENDIF.

    CALL METHOD zjncdocument->create_document
      EXPORTING open_inplace     = ' '
*             create_view_data = create_view_data
*             onsave_macro     = onsave_macro
*             startup_macro    = startup_macro
              document_title   = 'JNC'
              no_flush         = ' '
     IMPORTING error            = zjncerror
*             retcode          = retcode
              .

    IF sy-subrc NE 0.
      CALL FUNCTION 'POPUP_TO_INFORM'
           EXPORTING
              titel = zjnc_repid
              txt2  = 'Create zjncdocument Failed'
              txt1  = 'to make Excel zjncdocument'.
      Leave Program.
    ENDIF.

   CALL METHOD zjncdocument->get_spreadsheet_interface
     EXPORTING no_flush        = ' '
     IMPORTING sheet_interface = zjncspreadsheet
               error           = zjncerror
               retcode         = zjncretcode.

    IF sy-subrc NE 0.
      CALL FUNCTION 'POPUP_TO_INFORM'
           EXPORTING
              titel = zjnc_repid
              txt2  = 'Create zjncspreadsheet INTERFACE Failed'
```

```
                txt1  = 'to make Excel zjncspreadsheet'.
      Leave Program.
    ENDIF.

ENDFORM.                                  " ZJNC_INIT_EXCEL



Report ZExcelTest.

DATA spreadsheetintf TYPE REF TO i_oi_spreadsheet.

DATA: numRows    type I,
      maxRows    type I.

DATA: usa_sales TYPE i VALUE 1000,
      europe_sales TYPE i VALUE 2000,
      japan_sales TYPE i VALUE 1000,
      asia_sales TYPE i VALUE 100,
      america_sales TYPE i VALUE 100,
      africa_sales TYPE i VALUE 100.

  DATA: BEGIN OF head_table Occurs 0,
            hd_region(10),
            hd_sales(10),
            hd_date(10),
            hd_time(10),
            hd_weight(10),
            hd_amount(10),
            hd_id(10),
        END OF head_table.

  DATA: BEGIN OF sales_table Occurs 0,
            region(60),
            sales TYPE i,
            date  TYPE d,
            time  TYPE t,
```

```
          weight TYPE f,
          amount TYPE p DECIMALS 3,
          id(10) TYPE n,
      END OF sales_table.


DATA: ind TYPE i.

CLEAR: head_table.

Head_Table-hd_region = 'Region'.
Head_Table-hd_sales = 'Sales'.
Head_Table-hd_date = 'Date'.
Head_Table-hd_time = 'Time'.
Head_Table-hd_weight = 'Weight in MT'.
Head_Table-hd_amount = 'Value in Rupees'.
Head_Table-hd_id = 'Sytem ID'.

Append Head_Table.

CALL FUNCTION 'ZJNC_START_EXCEL'
  IMPORTING
    SPREADSHEETINTF       =  SPREADSHEETINTF.

CALL FUNCTION 'ZJNC_ADD_SHEET'
  EXPORTING
    PSHEET                = 'Sheet ONE'
    SPREADSHEETINTF       = spreadsheetintf.

maxRows = 1.

CALL FUNCTION 'ZJNC_ADD_RANGE'
  EXPORTING
    PRANGE                = 'HeadRange1'
    STARTROW              = maxRows
    STARTCOL              = 1
```

```
      NUMROWS                = 1
      NUMCOLS                = 7
      PSHEET                 = 'Sheet ONE'
      SPREADSHEETINTF        = spreadsheetintf.

* In ABAP Objects, you can only declare tables without headers.
* Hence sales_table[] ensures Header is Stripped

  CALL FUNCTION 'ZJNC_ADD_TABLE'
    EXPORTING
      PTABLE                 = head_table[]
      PRANGE                 = 'HeadRange1'
*     PSIZE                  = -1
      PBOLD                  = 1
*     PITALIC                = -1
*     PALIGN                 = -1
*     PFRONT                 = -1
*     PBACK                  = -1
*     PFORMAT                = 'NA'
      SPREADSHEETINTF        = spreadsheetintf.

  Add 1 to maxrows.

  CLEAR: sales_table.

  sales_table-region = 'USA'(usa).
  sales_table-sales = usa_sales.
  APPEND sales_table.

  sales_table-region = 'Europe'(eur).
  sales_table-sales = europe_sales.
  APPEND sales_table.

  sales_table-region = 'Japan'(jap).
  sales_table-sales = japan_sales.
  APPEND sales_table.
```

```
  sales_table-region = 'Asia'(asi).
  sales_table-sales = asia_sales.
  APPEND sales_table.

  LOOP AT sales_table.
    ind = sy-tabix.
    sales_table-date = sy-datum + ind.
    sales_table-time = sy-uzeit + ind.
    sales_table-weight = 100000 * ind.
    sales_table-amount = 11111 * ind.
    sales_table-id = ind.
    MODIFY sales_table.
  ENDLOOP.

  Describe Table  sales_table Lines numRows.

  CALL FUNCTION 'ZJNC_ADD_RANGE'
    EXPORTING
      PRANGE                = 'DataRange1'
      STARTROW              = maxRows
      STARTCOL              = 1
      NUMROWS               = numRows
      NUMCOLS               = 7
      PSHEET                = 'Sheet ONE'
      SPREADSHEETINTF       = spreadsheetintf.

  CALL FUNCTION 'ZJNC_ADD_TABLE'
    EXPORTING
      PTABLE                = sales_table[]
      PRANGE                = 'DataRange1'
*     PSIZE                 = -1
      PBOLD                 = 0
*     PITALIC               = -1
*     PALIGN                = -1
      PFRONT                = 3
```

```
*     PBACK                 = -1
*     PFORMAT               = 'NA'
      SPREADSHEETINTF       = spreadsheetintf.

* Start NewSheet on TOP
  Move 1 to maxRows.

  CALL FUNCTION 'ZJNC_ADD_SHEET'
    EXPORTING
      PSHEET                = 'Sheet TWO'
      SPREADSHEETINTF       = spreadsheetintf.

  CALL FUNCTION 'ZJNC_ADD_RANGE'
    EXPORTING
      PRANGE                = 'HeadRange2'
      STARTROW              = maxRows
      STARTCOL              = 1
      NUMROWS               = 1
      NUMCOLS               = 7
      PSHEET                = 'Sheet TWO'
      SPREADSHEETINTF       = spreadsheetintf.

* In ABAP Objects, you can only declare tables without headers.
* Hence sales_table[] ensures Header is Stripped

  CALL FUNCTION 'ZJNC_ADD_TABLE'
    EXPORTING
      PTABLE                = head_table[]
      PRANGE                = 'HeadRange2'
*     PSIZE                 = -1
      PBOLD                 = 1
*     PITALIC               = -1
*     PALIGN                = -1
*     PFRONT                = -1
*     PBACK                 = -1
*     PFORMAT               = 'NA'
```

```
    SPREADSHEETINTF        = spreadsheetintf.

Add 1 to maxrows.

CLEAR: sales_table.

sales_table-region = 'America'(ame).
sales_table-sales = america_sales.
APPEND sales_table.

sales_table-region = 'Africa'(afr).
sales_table-sales = africa_sales.
APPEND sales_table.

LOOP AT sales_table.
  ind = sy-tabix.
  sales_table-date = sy-datum + ind.
  sales_table-time = sy-uzeit + ind.
  sales_table-weight = 700000 * ind.
  sales_table-amount = 123456 * ind.
  sales_table-id = ind.
  MODIFY sales_table.
ENDLOOP.

Describe Table  sales_table Lines numRows.

CALL FUNCTION 'ZJNC_ADD_RANGE'
  EXPORTING
    PRANGE                 = 'DataRange2'
    STARTROW               = maxRows
    STARTCOL               = 1
    NUMROWS                = numRows
    NUMCOLS                = 7
    PSHEET                 = 'Sheet TWO'
    SPREADSHEETINTF        = spreadsheetintf.
```

```
  CALL FUNCTION 'ZJNC_ADD_TABLE'
    EXPORTING
      PTABLE                = sales_table[]
      PRANGE                = 'DataRange2'
*     PSIZE                 = -1
      PBOLD                 = 0
*     PITALIC               = -1
*     PALIGN                = -1
      PFRONT                = 55
      PBACK                 = 6
*     PFORMAT               = 'NA'
      SPREADSHEETINTF       = spreadsheetintf.


  CALL FUNCTION 'POPUP_TO_INFORM'
   EXPORTING
        titel = sy-repid
        txt2  = 'See EXCEL & SAVE if Needed'
        txt1  = 'Jai Hind ....'.
```


ABAP Object Oriented SpreadSheet with "Unlimited" Power

Content Author: Jayanta Narayan Choudhuri

Author Email: sss@cal.vsnl.net.in

Author Website: http://www.geocities.com/ojnc

Objective: SAP On-Line HELP has a section - "Controls and Control Framework (BC-CI)".

Under this refer "Desktop Office Integration (BC-CI)"

In that section read "The Spreadsheet Interface" thoroughly.

The ides is that once a programmer gets hold of a SpreadSheetInterface Object he/she can use the powerful methods to populate Excel in any way setting sheets, ranges, colours, fonts and ofcourse content.

Create a Function Group ZUTIL

Paste TOP Level code into LZUTILTOP

Create 4 Functionn Modules

ZJNC_START_EXCEL.

ZJNC_ADD_SHEET.

ZJNC_ADD_RANGE.

ZJNC_ADD_TABLE.

ZJNC_START_EXCEL - uses the "secret" screen 2307 which a user does not even see to get hold of a Spreadsheet Interface handle. With this alone a user has virtually unlimited power as he she can call all the methods.

But to make life easier I created 4 simple functions:

ZJNC_ADD_SHEET adds a sheet to a work book

ZJNC_ADD_RANGE adds a range to a sheet

ZJNC_ADD_TABLE adds a internal table to a range with specification of all properties like font colour size bold italic etc. In ABAP Objects, you can only declare tables without headers. Hence TABLE[] syntax ensures Header is Stripped.

It is best to have full geometry in mind and fill in the following sequence

For each SHEET Create 1 RANGE & Populate Data immediately
For each SHEET Reapeat for all Ranges

Before creating a range you will need to consider size based on table.
The no. of Rows & Columns will decide size.
The cumulative rows will gixe the corner co-ordinates.

Attached Files:

ZJNCEXCEL_Test.ab4 is the Test Program

ZJNCEXCEL_FUNC.ab4 is the Function Group

ZEXCEL_WRITEUP.txt is this write-up

* Author Jayanta Narayan Choudhuri

```
*         Flat 302
*         395 Jodhpur Park
*         Kolkata 700 068
*       Email sss@cal.vsnl.net.in
*       URL:  http://www.geocities.com/ojnc

*-----------------------------------------------------------------------------------------
-
* Screen 2307 has only 1 Custom Control MYCONTROL
* Screen 2307 Flow Logic

PROCESS BEFORE OUTPUT.
  MODULE ZJNCPBO.
*
PROCESS AFTER INPUT.
* MODULE ZJNCPAI.

1 Custom Control MYCONTROL
OK ZJNC_OK_CODE

*-----------------------------------------------------------------------------------------
-
FUNCTION ZJNC_START_EXCEL.
*"----------------------------------------------------------------------
*"*"Local interface:
*"  EXPORTING
*"     REFERENCE(SPREADSHEETINTF) TYPE REF TO  I_OI_SPREADSHEET
*"----------------------------------------------------------------------

Move SY-REPID to ZJNC_REPID.
CALL SCREEN 2307.

spreadsheetintf = zjncspreadsheet.

ENDFUNCTION.
```

```
FUNCTION ZJNC_ADD_SHEET.
*"----------------------------------------------------------------------
*"*"Local interface:
*"  IMPORTING
*"     REFERENCE(PSHEET) TYPE  C
*"     REFERENCE(SPREADSHEETINTF) TYPE REF TO  I_OI_SPREADSHEET
*"----------------------------------------------------------------------

    Move SY-REPID to ZJNC_REPID.

    CALL METHOD SPREADSHEETINTF->add_sheet
           EXPORTING name     = psheet
                     no_flush  = ' '
           IMPORTING error     = zjncerror
                     retcode   = zjncretcode.

ENDFUNCTION.

FUNCTION ZJNC_ADD_RANGE.
*"----------------------------------------------------------------------
*"*"Local interface:
*"  IMPORTING
*"     REFERENCE(PRANGE) TYPE  C
*"     REFERENCE(STARTROW) TYPE  I
*"     REFERENCE(STARTCOL) TYPE  I
*"     REFERENCE(NUMROWS) TYPE  I
*"     REFERENCE(NUMCOLS) TYPE  I
*"     REFERENCE(PSHEET) TYPE  C
*"     REFERENCE(SPREADSHEETINTF) TYPE REF TO  I_OI_SPREADSHEET
*"----------------------------------------------------------------------


    Move SY-REPID to zjnc_repid.

    CALL METHOD SPREADSHEETINTF->select_sheet
           EXPORTING name      = psheet
```

```
          no_flush  = ' '
          IMPORTING error     = zjncerror
                    retcode   = zjncretcode.

    CALL METHOD SPREADSHEETINTF->set_selection
          EXPORTING top       = StartRow
                    left      = StartCol
                    rows      = 1
                    columns   = 1
                    no_flush  = ' '
          IMPORTING error     = zjncerror
                    retcode   = zjncretcode.

    CALL METHOD SPREADSHEETINTF->insert_range
          EXPORTING name      = prange
                    rows      = numRows
                    columns   = numCols
                    no_flush  = ' '
          IMPORTING error     = zjncerror
                    retcode   = zjncretcode.


ENDFUNCTION.

FUNCTION ZJNC_ADD_TABLE.
*"----------------------------------------------------------------------
*"*"Local interface:
*"  IMPORTING
*"     REFERENCE(PTABLE) TYPE  TABLE
*"     REFERENCE(PRANGE) TYPE  C
*"     REFERENCE(PSIZE) TYPE  I DEFAULT -1
*"     REFERENCE(PBOLD) TYPE  I DEFAULT -1
*"     REFERENCE(PITALIC) TYPE  I DEFAULT -1
*"     REFERENCE(PALIGN) TYPE  I DEFAULT -1
*"     REFERENCE(PFRONT) TYPE  I DEFAULT -1
*"     REFERENCE(PBACK) TYPE  I DEFAULT -1
```

```
*"     REFERENCE(PFORMAT) TYPE  C DEFAULT 'NA'
*"     REFERENCE(SPREADSHEETINTF) TYPE REF TO  I_OI_SPREADSHEET
*"----------------------------------------------------------------------


** TYPES: SOI_zjnc_fields_table TYPE STANDARD TABLE OF RFC_FIELDS.
DATA: zjnc_fields_table   Type TABLE OF rfc_fields.
DATA: zjncwa_zjnc_fields_table TYPE     rfc_fields.

    Move SY-REPID to zjnc_repid.

    CALL FUNCTION 'DP_GET_FIELDS_FROM_TABLE'
         TABLES
              data   = ptable
              fields = zjnc_fields_table.

    CALL METHOD SPREADSHEETINTF->insert_one_table
           EXPORTING
*                   ddic_name    = ddic_name
                    data_table   = ptable
                    fields_table = zjnc_fields_table
                    rangename    = prange
                    wholetable   = 'X'
                    no_flush     = ' '
           IMPORTING error        = zjncerror
                    retcode      = zjncretcode.

    CALL METHOD SPREADSHEETINTF->set_font
           EXPORTING rangename = prange
                    family    = 'Arial'
                    size      = psize
                    bold      = pbold
                    italic    = pitalic
                    align     = palign
                    no_flush  = ' '
           IMPORTING error     = zjncerror
```

```
                    retcode   = zjncretcode.

    CALL METHOD SPREADSHEETINTF->set_color
           EXPORTING rangename = prange
                     front     = pfront
                     back      = pback
                     no_flush  = ' '
           IMPORTING error     = zjncerror
                     retcode   = zjncretcode.

    If pFormat <> 'NA'.
        CALL METHOD SPREADSHEETINTF->set_format_string
               EXPORTING rangename    = prange
                         formatstring = pformat
                         no_flush     = ' '
               IMPORTING error     = zjncerror
                         retcode   = zjncretcode.
    EndIf.

ENDFUNCTION.


*---------------------------------------------------------------------------------------
* TOP level Include of Function Group ZUTIL


FUNCTION-POOL ZUTIL.                          "MESSAGE-ID ..

* Global ZUTIL Data for ZJNCEXCEL
DATA zjnccontainer   TYPE REF TO cl_gui_custom_container.

DATA zjnccontrol     TYPE REF TO i_oi_container_control.

DATA zjncdocument    TYPE REF TO i_oi_document_proxy.

DATA zjncspreadsheet TYPE REF TO i_oi_spreadsheet.
```

```
DATA zjncerror       TYPE REF TO i_oi_error.
DATA zjncretcode     TYPE SOI_RET_STRING.

DATA zjncexcelsheet  TYPE soi_document_type VALUE
SOI_DOCTYPE_EXCEL_SHEET.

DATA:   zjnc_ok_code LIKE sy-ucomm,          " return code from screen
        zjnc_repid   LIKE sy-repid.


***********************************************************************
*   P B O
***********************************************************************
MODULE zjncpbo OUTPUT.

*    SET PF-STATUS 'ZJNCSTATUS'.
*    SET TITLEBAR  'ZJNCTITLE'.

    IF zjncdocument IS NOT INITIAL.
       RETURN.
    EndIf.

    Perform ZJNC_INIT_EXCEL.

    Leave to Screen 0.

ENDMODULE.                             " PBO


*&---------------------------------------------------------------------*
*&      Form  ZJNC_INIT_EXCEL
*&---------------------------------------------------------------------*
Form ZJNC_INIT_EXCEL.

    CALL METHOD c_oi_container_control_creator=>get_container_control
```

```
      IMPORTING control = zjnccontrol
                error   = zjncerror.

    IF sy-subrc NE 0.
      CALL FUNCTION 'POPUP_TO_INFORM'
           EXPORTING
                titel = zjnc_repid
                txt2  = 'Create OLE zjnccontrol Failed'
                txt1  = 'to make Excel zjnccontrol'.
      Leave Program.
    ENDIF.

    CREATE OBJECT zjnccontainer
      EXPORTING
         CONTAINER_NAME = 'MYCONTROL'
      EXCEPTIONS
         CNTL_ERROR = 1
         CNTL_SYSTEM_ERROR = 2
         CREATE_ERROR = 3
         LIFETIME_ERROR = 4
         LIFETIME_DYNPRO_DYNPRO_LINK = 5.

    IF sy-subrc NE 0.
*     add your handling
    ENDIF.

    CALL METHOD zjnccontrol->init_control
         EXPORTING  r3_application_name      = 'R/3 Basis' "#EC NOTEXT
                    inplace_enabled          = 'X'
                    inplace_scroll_documents = 'X'
                    parent                   =  zjnccontainer
                    register_on_close_event  = 'X'
                    register_on_custom_event = 'X'
                    no_flush                 = 'X'
          IMPORTING error = zjncerror.
```

```
    IF sy-subrc NE 0.
      CALL FUNCTION 'POPUP_TO_INFORM'
           EXPORTING
                titel = zjnc_repid
                txt2  = 'INIT OLE zjnccontrol Failed'
                txt1  = 'to init Excel zjnccontrol'.
      Leave Program.
    ENDIF.

    CALL METHOD zjnccontrol->get_document_proxy
       EXPORTING document_type       = zjncexcelsheet
*                document_format     = document_format
*                register_container  = register_container
                 no_flush            = ' '
       IMPORTING document_proxy       = zjncdocument
                 retcode              = zjncretcode
                 error                = zjncerror.

    IF sy-subrc NE 0.
      CALL FUNCTION 'POPUP_TO_INFORM'
           EXPORTING
                titel = zjnc_repid
                txt2  = 'Create zjncdocument PROXY Failed'
                txt1  = 'to make Excel zjncdocument'.
      Leave Program.
    ENDIF.

    CALL METHOD zjncdocument->create_document
      EXPORTING open_inplace     = ' '
*               create_view_data = create_view_data
*               onsave_macro     = onsave_macro
*               startup_macro    = startup_macro
                document_title   = 'JNC'
                no_flush         = ' '
     IMPORTING error            = zjncerror
*               retcode          = retcode
```

```
                .

    IF sy-subrc NE 0.
      CALL FUNCTION 'POPUP_TO_INFORM'
           EXPORTING
                titel = zjnc_repid
                txt2  = 'Create zjncdocument Failed'
                txt1  = 'to make Excel zjncdocument'.
      Leave Program.
    ENDIF.

   CALL METHOD zjncdocument->get_spreadsheet_interface
     EXPORTING no_flush        = ' '
     IMPORTING sheet_interface = zjncspreadsheet
               error           = zjncerror
               retcode         = zjncretcode.

    IF sy-subrc NE 0.
      CALL FUNCTION 'POPUP_TO_INFORM'
           EXPORTING
                titel = zjnc_repid
                txt2  = 'Create zjncspreadsheet INTERFACE Failed'
                txt1  = 'to make Excel zjncspreadsheet'.
      Leave Program.
    ENDIF.

ENDFORM.                              " ZJNC_INIT_EXCEL


Report ZExcelTest.

DATA spreadsheetintf TYPE REF TO i_oi_spreadsheet.

DATA: numRows    type I,
      maxRows    type I.
```

```
DATA: usa_sales TYPE i VALUE 1000,
      europe_sales TYPE i VALUE 2000,
      japan_sales TYPE i VALUE 1000,
      asia_sales TYPE i VALUE 100,
      america_sales TYPE i VALUE 100,
      africa_sales TYPE i VALUE 100.

  DATA: BEGIN OF head_table Occurs 0,
            hd_region(10),
            hd_sales(10),
            hd_date(10),
            hd_time(10),
            hd_weight(10),
            hd_amount(10),
            hd_id(10),
        END OF head_table.

  DATA: BEGIN OF sales_table Occurs 0,
            region(60),
            sales TYPE i,
            date  TYPE d,
            time  TYPE t,
            weight TYPE f,
            amount TYPE p DECIMALS 3,
            id(10) TYPE n,
        END OF sales_table.


  DATA: ind TYPE i.

  CLEAR: head_table.

  Head_Table-hd_region = 'Region'.
  Head_Table-hd_sales = 'Sales'.
  Head_Table-hd_date = 'Date'.
  Head_Table-hd_time = 'Time'.
```

```
  Head_Table-hd_weight = 'Weight in MT'.
  Head_Table-hd_amount = 'Value in Rupees'.
  Head_Table-hd_id = 'Sytem ID'.

  Append Head_Table.

  CALL FUNCTION 'ZJNC_START_EXCEL'
    IMPORTING
      SPREADSHEETINTF       =  SPREADSHEETINTF.

  CALL FUNCTION 'ZJNC_ADD_SHEET'
    EXPORTING
      PSHEET                = 'Sheet ONE'
      SPREADSHEETINTF       = spreadsheetintf.

  maxRows = 1.

  CALL FUNCTION 'ZJNC_ADD_RANGE'
    EXPORTING
      PRANGE                = 'HeadRange1'
      STARTROW              = maxRows
      STARTCOL              = 1
      NUMROWS               = 1
      NUMCOLS               = 7
      PSHEET                = 'Sheet ONE'
      SPREADSHEETINTF       = spreadsheetintf.

* In ABAP Objects, you can only declare tables without headers.
* Hence sales_table[] ensures Header is Stripped

  CALL FUNCTION 'ZJNC_ADD_TABLE'
    EXPORTING
      PTABLE                = head_table[]
      PRANGE                = 'HeadRange1'
*     PSIZE                 = -1
      PBOLD                 = 1
```

```
*     PITALIC                = -1
*     PALIGN                 = -1
*     PFRONT                 = -1
*     PBACK                  = -1
*     PFORMAT                = 'NA'
      SPREADSHEETINTF        = spreadsheetintf.

  Add 1 to maxrows.

  CLEAR: sales_table.

  sales_table-region = 'USA'(usa).
  sales_table-sales = usa_sales.
  APPEND sales_table.

  sales_table-region = 'Europe'(eur).
  sales_table-sales = europe_sales.
  APPEND sales_table.

  sales_table-region = 'Japan'(jap).
  sales_table-sales = japan_sales.
  APPEND sales_table.

  sales_table-region = 'Asia'(asi).
  sales_table-sales = asia_sales.
  APPEND sales_table.

  LOOP AT sales_table.
    ind = sy-tabix.
    sales_table-date = sy-datum + ind.
    sales_table-time = sy-uzeit + ind.
    sales_table-weight = 100000 * ind.
    sales_table-amount = 11111 * ind.
    sales_table-id = ind.
    MODIFY sales_table.
  ENDLOOP.
```

```
  Describe Table  sales_table Lines numRows.

  CALL FUNCTION 'ZJNC_ADD_RANGE'
    EXPORTING
      PRANGE                = 'DataRange1'
      STARTROW              = maxRows
      STARTCOL              = 1
      NUMROWS               = numRows
      NUMCOLS               = 7
      PSHEET                = 'Sheet ONE'
      SPREADSHEETINTF       = spreadsheetintf.

  CALL FUNCTION 'ZJNC_ADD_TABLE'
    EXPORTING
      PTABLE                = sales_table[]
      PRANGE                = 'DataRange1'
*     PSIZE                 = -1
      PBOLD                 = 0
*     PITALIC               = -1
*     PALIGN                = -1
      PFRONT                = 3
*     PBACK                 = -1
*     PFORMAT               = 'NA'
      SPREADSHEETINTF       = spreadsheetintf.

* Start NewSheet on TOP
  Move 1 to maxRows.

  CALL FUNCTION 'ZJNC_ADD_SHEET'
    EXPORTING
      PSHEET                = 'Sheet TWO'
      SPREADSHEETINTF       = spreadsheetintf.

  CALL FUNCTION 'ZJNC_ADD_RANGE'
    EXPORTING
```

```
      PRANGE                  = 'HeadRange2'
      STARTROW                = maxRows
      STARTCOL                = 1
      NUMROWS                 = 1
      NUMCOLS                 = 7
      PSHEET                  = 'Sheet TWO'
      SPREADSHEETINTF         = spreadsheetintf.

* In ABAP Objects, you can only declare tables without headers.
* Hence sales_table[] ensures Header is Stripped

  CALL FUNCTION 'ZJNC_ADD_TABLE'
    EXPORTING
      PTABLE                  = head_table[]
      PRANGE                  = 'HeadRange2'
*     PSIZE                   = -1
      PBOLD                   = 1
*     PITALIC                 = -1
*     PALIGN                  = -1
*     PFRONT                  = -1
*     PBACK                   = -1
*     PFORMAT                 = 'NA'
      SPREADSHEETINTF         = spreadsheetintf.

  Add 1 to maxrows.

  CLEAR: sales_table.

  sales_table-region = 'America'(ame).
  sales_table-sales = america_sales.
  APPEND sales_table.

  sales_table-region = 'Africa'(afr).
  sales_table-sales = africa_sales.
  APPEND sales_table.
```

```
  LOOP AT sales_table.
    ind = sy-tabix.
    sales_table-date = sy-datum + ind.
    sales_table-time = sy-uzeit + ind.
    sales_table-weight = 700000 * ind.
    sales_table-amount = 123456 * ind.
    sales_table-id = ind.
    MODIFY sales_table.
  ENDLOOP.

  Describe Table  sales_table Lines numRows.

  CALL FUNCTION 'ZJNC_ADD_RANGE'
    EXPORTING
      PRANGE                = 'DataRange2'
      STARTROW              = maxRows
      STARTCOL              = 1
      NUMROWS               = numRows
      NUMCOLS               = 7
      PSHEET                = 'Sheet TWO'
      SPREADSHEETINTF       = spreadsheetintf.

  CALL FUNCTION 'ZJNC_ADD_TABLE'
    EXPORTING
      PTABLE                = sales_table[]
      PRANGE                = 'DataRange2'
*     PSIZE                 = -1
      PBOLD                 = 0
*     PITALIC               = -1
*     PALIGN                = -1
      PFRONT                = 55
      PBACK                 = 6
*     PFORMAT               = 'NA'
      SPREADSHEETINTF       = spreadsheetintf.
```

```
CALL FUNCTION 'POPUP_TO_INFORM'
 EXPORTING
      titel = sy-repid
      txt2  = 'See EXCEL & SAVE if Needed'
      txt1  = 'Jai Hind ....'.
```


[日期：2006-10-18] 来源：sap-img 作者：sapsky [字体：大 中 小]

What are ABAP Objects?

ABAP Objects is a new concept in R/3 Release 4.0. The term has two meanings. On the one hand, it stands for the entire ABAP runtime environment. On the other hand, it represents the object-oriented extension of the ABAP language.

The following is a simple example shows the object-oriented aspect of function groups in the simple case of a counter.

Suppose we have a function group called COUNTER:

Create an abap program with this code :-

```
FUNCTION-POOL COUNTER.

DATA COUNT TYPE I.

FUNCTION SET_COUNTER.
* Local Interface IMPORTING VALUE(SET_VALUE)
  COUNT = SET_VALUE.
ENDFUNCTION.

FUNCTION INCREMENT_COUNTER.
  ADD 1 TO COUNT.
ENDFUNCTION.

FUNCTION GET_COUNTER.
* Local Interface: EXPORTING VALUE(GET_VALUE)
```

```
  GET_VALUE = COUNT.
ENDFUNCTION.

* End of program code
```

The function group has a global integer field COUNT, and three function modules,

- SET_COUNTER,
- INCREMENT_COUNTER, and
- GET_COUNTER, that work with the field.

Two of the function modules have input and output parameters. These form the data interface of the function group.

Any ABAP program can then work with this function group. For example:

```
REPORT ZABAPOO.

DATA NUMBER TYPE I VALUE 5.

CALL FUNCTION 'SET_COUNTER' EXPORTING SET_VALUE = NUMBER.

DO 3 TIMES.
  CALL FUNCTION 'INCREMENT_COUNTER'.
ENDDO.

CALL FUNCTION 'GET_COUNTER' IMPORTING GET_VALUE = NUMBER.

WRITE: / 'After processing NUMBER is :- ', NUMBER.

* End of program code
```

After this section of the program has been processed, the program variable NUMBER will have the value 8.

The program itself cannot access the COUNT field in the function group. Operations on this field are fully encapsulated in the function module. The program can only communicate with the function group by calling its function modules.

SAP Business Objects & BAPI

SAP Business Objects (类似于 java 中的对象有 attributes 和 methods,另外还有 tables).

1, An SAP Business Objects is the representation of a central business object in the real world, such as an employee, sales order, purchase requisition,

purchase order, applicant,invoice, and so on.

2, A business object is composed of :

(1), [tables] that are related in a business context,

(2), [Attributes] are characteristics that specify the business object, The attribute can only be modified by the methods that belong to the business object.

(3), [the related application programs].

3, Business Objects are maintained by SAP in the Business Object Repository(BOR).

BOR 是一个工具被用来 create, delete, modify 业务对象。

BAPI (Business Application Programming Interface)业务应用编程接口

SAP R/3 为了集成第三方软件，为软件厂商提供了接口 BAPI，为了 access R/3 中的业务数据(business data)和业务流程(business),必须使用 BAPI.

可以这样说一个 BAPI 就是某个 Business Object 的一个 public attribute 或一个 public method.

GL A/C posting using BAPI_ACC_GL_POSTING_POST.

Requirement:

An interface needs to be developed to upload large journals either from a tab delimited file. Currently it's being done manually using FB50 transaction. Various Cost of Capital & Operational Property journals have to be posted at month end. A file interface needs to be developed.

Processing:

It involves following development/configurations:

- File format determination (Required / optional fields and field checks).
- Logical File Path configuration through transaction 'FILE'. A new physical file path should be created on operating system level or an existing one can be used if agreed. The Basis team member should create a new file path at operating system level, if required. The file path will have three directories: /GL_FILE /GL_Processed /GL_Error
- Program Z_BAPI_GL_AC_DOC_POST needs to be developed to do the processing as described below:
    - The processing can be done in foreground as well as in background mode.
    - In case of background: File can only be read from Application Server.
    - In case of foreground: User will have an option to choose from Presentation or Application Server File.

- Logical File Path / Name needs to be configured using FILE transaction for application server file processing. It is required to identify the Application server directory and file. Further it gives the flexibility to change the path and file name by using the transaction FILE, without making any changes to the program. It should not be hard-coded as directory structure might be different for Testing and production development servers.
- Read the input file from presentation or application server as chosen.
- Prepare Account doc header and detail internal tables.
- Call 'BAPI_ACC_GL_POSTING_POST' to post the GL accounting document.
- For application server case, Processed file can be flagged archived by using Function module DX_FILE_COPY and then it can be deleted using EPS_DELETE_FILE .
- Error file can be created (in /GL_Error directory for application server and with .err extension in case of PC processing) for error records and can be re-processed after correction.
- The list of successful and error transaction will be displayed after processing.

代码:

```
report z_test_bapi_gl_ac_doc LINE-SIZE 200.
*-----------------------------------------------------------------------*
*      Written By: Ram Manohar Tiwari
*-----------------------------------------------------------------------*
*      Presented By: http://www.rmtiwari.com
*-----------------------------------------------------------------------*

data:
    obj_type like bapiache02-obj_type,
    obj_key like bapiache02-obj_key,
    obj_sys like bapiache02-obj_sys,
    documentheader like bapiache08,

    accountgl like bapiacgl08
            occurs 0 with header line,
    currencyamount like bapiaccr08
            occurs 0 with header line,
    return like bapiret2
            occurs 0 with header line,
    extension1 like bapiextc
            occurs 0 with header line,
```

```
    t_edidd  like  edidd  occurs  0  with  header  line,
    bapi_retn_info  like  bapiret2  occurs  0  with  header  line.
data:  error_flag.


*documentheader-obj_type     =  'BKPFF'.
*documentheader-obj_key      =  '18000000002002004'.
*documentheader-obj_type     =  'BKPFF'.
*documentheader-obj_key      =  '180000000010002004'.

*documentheader-obj_sys      =  'RD1CLNT200'.


documentheader-username      =  sy-uname.
documentheader-header_txt    =  'Test  using  BAPI'.
documentheader-comp_code     =  '1000'.

*documentheader-ac_doc_no
*documentheader-fisc_year    =  '2005'.

documentheader-doc_date      =  sy-datum.
documentheader-pstng_date    =  sy-datum.

*documentheader-trans_date
*documentheader-fis_period

documentheader-doc_type      =  'SA'.


*documentheader-ref_doc_no
*documentheader-compo_acc
*documentheader-reason_rev


accountgl-itemno_acc  =  '1'.
```

```
accountgl-gl_account = '0000160100'.
accountgl-comp_code = '1000'.
accountgl-pstng_date = sy-datum.
accountgl-doc_type   = 'SA'.
accountgl-profit_ctr = '0000010000'.
append accountgl.

accountgl-itemno_acc = '2'.
accountgl-gl_account = '0000160100'.
accountgl-comp_code = '1000'.
accountgl-pstng_date = sy-datum.
accountgl-doc_type   = 'SA'.
accountgl-profit_ctr = '0000010000'.
append accountgl.


*AC_DOC_NO
*FISC_YEAR
*FIS_PERIOD
*accountgl-STAT_CON = 'X'.
*REF_KEY_1
*REF_KEY_2
*REF_KEY_3
*CUSTOMER
*VENDOR_NO
*ALLOC_NMBR
*ITEM_TEXT
*BUS_AREA
*COSTCENTER
*ACTTYPE
*ORDERID
*ORIG_GROUP
*COST_OBJ
*PROFIT_CTR
*PART_PRCTR
*WBS_ELEMENT
```

```
*NETWORK
*ROUTING_NO
*ORDER_ITNO


currencyamount-itemno_acc = '1'.
currencyamount-currency = 'GBP'.
currencyamount-amt_doccur = '100.00'.
append currencyamount.

currencyamount-itemno_acc = '2'.
currencyamount-currency = 'GBP'.
currencyamount-amt_doccur = '-100.00'.
append currencyamount.


*   call BAPI-function in this system
call function 'BAPI_ACC_GL_POSTING_POST'
     exporting
         documentheader = documentheader
*     importing
*         obj_type       = obj_type
*         obj_key        = obj_key
*         obj_sys        = obj_sys
     tables
         accountgl      = accountgl
         currencyamount = currencyamount
         return         = return
         extension1     = extension1
     exceptions
         others         = 1.
if sy-subrc <> 0.
  message e999(re) with 'Problem occured'.
else.
      loop at return.
         if not return is initial.
```

```
        clear bapi_retn_info.
        move-corresponding return to bapi_retn_info.
        if return-type = 'A' or return-type = 'E'.
          error_flag = 'X'.
        endif.
        append bapi_retn_info.
      endif.
    endloop.
  if error_flag = 'X'.
    message e999(re) with  'Problem occured'.
    rollback work.
  else.
  commit work.
  endif.
endif.
```

HR-TM Object

-----Original Message-----

Subject: HR-TM Object

My question will be more applicable for those people working with the HR Time Management module.

I want to ask is there a way to populate the Time Clocking data for employee using the BAPI/RFC. Though there are many partners solution available in the market, we are unable to do so due to some legal contract issue.

Presently, we have a non SAP certified system to capture the employee clocking data. This system will then download a ASCII file to interface to the SAP system before running the time evaluation program.

Any advise from someone out there? Thanks.

-----Reply Message-----

Subject: RE: HR-TM Object

The way we handled it was to create an RFC that can be invoked by the UNIX startrfc command. This is triggered by the transport process we use to fetch and move files. This RFC is modeled after the channel 1 (CC1) import provided by SAP. The ASCII file seems to be the smoothest solution at this time. I am told there are timeclocks out there that have direct connectivity to SAP but I have not seen one at this time.

-----Reply Message-----
Subject: RE: HR-TM Object

We have a similar need. It looks like the TimeMgtConfirmation.Post BAPI will do this. Take a look at it. I haven't used it yet, but it appears to be what we need. We plan to build a web application to report time that would use this (or similar) BAPI.

-----Reply Message-----
Subject: RE: HR-TM Object

We have found that response time of BAPI's is too ong for mass input. I have approx 1 million punches a week coming into the system. This would take tooooooooo long via a BAPI. Using the channel which is SAP supplied, I can import 25,000 punches in 4 minutes. The only reason I am using my own RFC is to remove some of the validation checks being done by SAP (I have pre checked the data before the import step) to speed things up.

I still do not trust HR BAPI's to work as advertised. They do NOT.

If your WEB application is using the BAPI directly (as the user types) then the response time issue is probably not as important. People can olnly type so fast. You may want to watch out for licesnse issues with this though. The license is very expensive if you have a large number of people using them. One license per person is required!

-----End of Message-----

BAPI_EMPLOYEE_CHECKPASSWORD

-----Original Message-----

Subject: BAPI_EMPLOYEE_CHECKPASSWORD

All

Is the Password parameter in BAPI_EMPLOYEE_CHECKPASSWORD a structure?

Also, Do you have some sample code on how to use this?

Thanks for your help.

-----Reply Message-----

Subject: RE: BAPI_EMPLOYEE_CHECKPASSWORD

Hi,

It's a simple parameter. If you go into transaction "se37" and type in the RFC name (e.g. BAPI_EMPLOYEE_CHECKPASSWORD), select the Import/Export radio button and then click the Display button. You'll see the PASSWORD parameter listed. If you look under the "Reference Type" column you see that is BAPIUID-PASSWORD, which indicates that is the PASSWORD field of the BAPIUID structure. If you double-click on this column you'll see the whole structure and the line will be positioned to this field.

-----End of Message-----

How to find function module or Bapi for particular transaction in sap?

If you mean that you need to know what BAPI's a particular tranx uses, which I can only assume that's what you mean, then you should access the code behind the transaction and search for 'CALL'. That normally is the standard method that think that most people use.

Suppose you want to find the bapi for creating a sales order, you usually use transaction VA01 for this.

1. Find out the package of the transaction.

Start Va01 go to system --> status.

Double click on transaction

Package is VA

Open this package in SE80

Open business engineering-->Business object types

Find the BO which sounds the most appropriate

I would suggest BUS2032 Sales Order

Double click.

Open methods.

Find the released method with from data or something similar in the name

, Createfromdat2

Position the cursor in it and click the program button

Scroll down to find the bapi used in this method

With this way you can also find out programs and FM's

2. Start va01 go to system-->status

Double click transaction VA01

Double click on package

Read the application component. (this is SD-SLS Sales)

Then open the transaction BAPI

Sales and distribution-->Sales-->sales order

createfromdat2

Questions on BAPI and RFC Programming

One BAPI to get status back of the delivery no ! >> RFC programming

1) I am creating purchase order(PO) and creating inbound delivery for that purchase order(PO) which will be distibuted to NONSAP((warehouse management) .

> ----------

Then NONSAP((ware house management) sends back the confirmation (after updating its Data base of that inbound delivery) to SAP through IDoc which uses delivery number. I am sending delivery no(in IDoc) from Non Sap(ware house management) to Sap through JCO.

If you use Jco, which is the Java Connector, you know how to program

Java or you have a resource available that knows how to program java.

Then in SAP [after getting back that IDoc from NON-SAP((ware house management)], the status of that delivery number will changes to "Confirmed". We can see that in SAP through transaction code : vl33n.

> ----------

I am not so familiar with VL33N, so I could not find this status change. Am I right to think that this status is s tored in table LIKP field VLSTK (Distribution Status (Decentralized Warehouse
Processing)).

NOW , I need one BAPI which I can use from NON-SAP, to get back the status of that delivery number. M y idoc sending from NON-SAP, since it is through idoc , it is not returning back the delivery number's status f rom SAP, I need to use one BAPI from nonsap (java prg) which takes input as delivery number and gives ba ck the status of that delivery number from SAP.

> ----------

You need either a BAPI (there is non that does what you want), or you need an RFC.

2) What is RFC sever programming ? In which language is it in? I am an ABAP programmer. How can I deal that issue please guide me .

> ----------

Ahah. RFC ! So, RFC's are actually function modules, but in the Attributes tab you enable 'Remote-enabled mo dule' and you enable 'Start Immediately'. RFC's are created with SE37, you can also create them with SE80.

In the import section you would create a p_vbeln like likp-vbeln, in the export section you would create a p_sta tus like likp-vlstk.

In the source code you would have something like

```
select single vlstk
into p_status
from likp
where vbeln eq p_vbeln.
```

And then in JCO you call this Function Module/RFC. The java person should know how to do this.

Bapi’s Customizing

Hello,

Is it true SAP systems often are so customized that you can not trust on Bapi's if they exist yes or no ? Can a SAP system be modified without changes in the repository ? Wher is the repository located from a world wid e firm ?

Everybody felt free to answer me:

------Reply Message-----

Subject: RE: Bapi's Customizing

BAPI has nothing to do with customizing. Business Objects are used based of course on the Customizing settings made on an SAP client. There is no way you can use a Business Object without any customizing made on R/3.For instance, to place a Sales Order thru a customer-designed BAPI application, you have to set the sales organization, the product code and so on and these are all part of customizing. When doing customizing, customizing tables are created according to needs, automatically by R/3 and you don't have to know about these. This are all transparent to the BAPI programmer. All you need to know are the BAPI classes, methods and parameters. That's it .....

-----End of Message-----

BAPI C++ Library

-----Original Message-----

Subject: BAPI C++ Library

Hi,

I've just subscribed to this list in the hopes of getting an answer to a problem which I'm having. I'm trying to statically link to the BAPI C++ class library. In MSVC++ 5.0 I'm getting undefined external symbol errors from the linker on all the CBO references, which I didn't get in a previous iteration, using just the RFC class library and its C++ objects. My project currently includes the lib file, RFCClass.lib, which I gather from the linker errors doesn't resolve the CBO calls. I'm wondering if anyone knows what the name of the CBO lib file is, and where in the Automation Kit I can find it. Thanks in advance for any assistance.

Regards,

-----Reply Message-----

Subject: RE: BAPI C++ Library

Hi,

The lib file is named "cbo.lib" and you'll find it in the SAP Automation Kit in the directory: ..\Bapi Class\lib\ (in version 4.0b or 4.0a).
Further information you'll find in the ..\Bapi Class\Help directory.

-----End of Message-----

ODBC and MsAccess

-----Original Message----
Subject: ODBC and MsAccess

Hello All,

We are building an interface between R/3 and an application a company build for us in Ms. Access. Since they build the application for us we want to subcontract this interface to this company. The consultant came to with the idea of using ODBC to read/write directly to the Oracle database.
Can this be done ???? (I thought not !)

anybody has experience with Access/R/3

Thanks !

-----Reply Message----
Subject: RE: ODBC and MsAccess

Hello!

Dont write directly to the SAP database because you can get inconsistent information in your system. Always use the standard functionality with its checks. For this reason you can use the BAPI-method.

-----End of Message-----

BAPI ActiveX Control

-----Original Message-----
Subject: BAPI ActiveX Control

Hi!

Where can I find the BAPI ActiveX Control ?

Thanks in advance

-----Reply Message-----

Subject: RE: BAPI ActiveX Control

Hi!

All necessary files for using BAPI are installed together with the SAP
GUI on your PC (from 3.1H on).
If it does not work or you can not find any SAP ActiveX Control you have to
install the SAP GUI again.
The BAPI ActiveX Control might be the file "wdobapi.ocx ".

-----End of Message-----

Difference Between BAPI and RFC

What is the main difference between BAPI and RFC and difference between BAPI and BDC?

BAPI is used only when it is available for the particular transaction like Delivery Sales order. but BDC can be used for any
transaction which have screen and fields.

BAPI is directly updated the database instead BDC run through the screen flow.

So BAPI can't handle all the flow logic checking and enhancement put by programmer to faciliate the user requirement.

Difference between BAPI and BDC:

BAPI is a higher end usage for tranfering the data from SAP to non-SAP and vice-versa. for ex: if we are using VB application,where in that we want to connect to SAP and retireve the data,and then change and update the data in SAP for that purpose we can use that.

Apart from that, we can also use it for Uploading/Downloading the data from SAP to Non-SAP like BDC, provided we have an existing BAPI for that.

BAPI function modules will also do all the checks required for data integrity like Transactions for BDC.

There is one more advantage using BAPI instead of BDC. When we go for upgradation, there might be pozzibility to change the screen elements for transactions depending on the requirement. In that case,our BDC pgm may or may not work (depending on the screen changes they have made). Unless and until we prepare new BDC we cant use the old BDC pgm. But in BAPI, SAP promises that they are going to keep the old BAPI and for new functionality they will provide an upgraded BAPI. Until we write a new BAPI pgm, we can use the exisitng BAPI pgm.

Read function module definition

-----Original Message-----
Subject: read function module definition

hi,

did someone know, where the import, export, tables, exception definition
of a function module is storeed?
I try to write a RFC call to read the interface of a function module (or
BAPI) to generate a C++ interface.
Therefore I write a function in Abap, which could be called via RFC.

Thx

-----Reply Message-----
Subject: RE: read function module definition

Try RFC_GET_FUNCTION_INTERFACE

-----End of Message-----

Persistent key

-----Original Message-----
Subject: persistent key

Hi all,
i'm new in bapi programming, i'm trying to call GetList method from
GeneralLedgerAccount Business Object, and got this error message 'The
persistent key for an bussiness object instance of type GeneralLedgerAccount
has not been set. Cannot invoke method GetList'.
How can i set the persistent key, and what is the persistent key exactly ?

How can i check the persistent key in the BOR ?

I'm using SAP R/3 version 3.1g and VB version 6.0

-----Reply Message-----

Subject: RE: persistent key

Although we are currently using SAP 4.0B, the GeneralLedgerAccount BAPI has not changed significantly (if at all). I have had this error message many times working with HR BAPI's and it took a lot of research to finally puzzle out what SAP actually wants in these situations. On the other hand, I took a look at the GetList method for this BAPI and I cannot see why you would be getting this error! There are three kinds of BAPI's I have encountered so far - those that return actual tables of data based on key fields you fill in (ie. Employee.GetList), those that return tables of keys that point to the actual data within SAP (think of C pointers! ie. EmployeePersonalData.GetList), and BAPI's that have no input parameters at all but can retrieve extra information directly related to a particular 'instance' of a business object that you have previously 'instantiated' (translation - the last 'record' you 'read' from the table! ie. EmployeePersonalData.GetDetail). GeneralLedgerAccount is of the first type, while the error you get is related to BAPI's of the third type. It is usually caused by calling a GetDetail method without calling GetSAPObject with a complete, unique key value first. The following code fragment shows how I read Employee.GetList - this may shed some light on what you might be missing. The second fragment shows how to 'instantiate' a 'persistant' object (don't we just love this terminology?) before calling a GetDetail BAPI. Hope this helps you out!

==========

```
Dim oBapiControl As Object
Dim oConnection As Object
Dim oEmployee As Object
Dim oReturn As Object
Dim otabPersonalData As Object
Dim otabOrgAssignment As Object
Dim oRow As Object

oBapiControl = CreateObject("SAP.BAPI.1")
Set oConnection = oBapiControl.Connection
'SET UP ALL oConnection PARAMETERS HERE AND CALL oConnection.Logon(0, True)
Set oEmployee = oBapiControl.GetSAPObject("Employee")
oEmployee.Getlist Lastname:="*", _
                  Return:=oReturn, _
```

```
                    PersonalData:=otabPersonalData, _
                    OrgAssignment:=otabOrgAssignment
For Each oRow In otabPersonalData.Rows
   Print "Personnel Number = " + oRow.Value("PERNO")
...
Next oRow
```

- Substitute the word 'GeneralLedgerAccount' for 'Employee' and this should work for you!

==========

```
Private Sub LoadFamily(sPerno As String)
  Dim cSep          As String * 1
  Dim sToday        As Date
  Dim oCol          As Object
  Dim oDelRow       As Object
  Dim oReturn       As Object
  Dim oFReturn      As Object
  Dim otabFKeyList  As Object
  Dim oFamilyKey    As Object
  Dim oFamilyMem    As Object
  Dim sFirstName    As String
  Dim sSecondName   As String
  Dim sLastName     As String
  Dim sGender       As String
  Dim sBirthdate    As Date

  cSep = Chr(9)
  sToday = Date

  ' RETRIEVE ALL FAMILY MEMBERS FOR A GIVEN PERSONNEL NUMBER.
  ' NOTE THAT THE BAPI RETURNS A TABLE OF KEYS (Familykey) AND
  ' NOT THE ACTUAL DATA!
  Set otabFKeyList = Nothing
  oFamilyMembers.Getlist EmployeeNumber:=sPerno, _
                    Subtype:="", _
                    Timeintervallow:=sToday, _
                    Timeintervalhigh:=sToday, _
                    Return:=oFReturn, _
```

```
                    Familykey:=otabFKeyList
If oFReturn.Value("TYPE") <> "E" Then          'IF THE CALL SUCCEEDED,
  For Each oFamilyKey In otabFKeyList.Rows       THEN FOR EACH KEY IN THE TABLE...
    On Error Resume Next
    ' INSTANTIATE A PERSISTANT LOCAL BUSINESS OBJECT HERE USING THE
    ' FULL KEY VALUE
    Set oFamilyMem = oBapiControl.GetSAPObject("EmployeeFamilyMember", _
                                oFamilyKey.Value("EMPLOYEENO"), _
                                oFamilyKey.Value("SUBTYPE"), _
                                oFamilyKey.Value("OBJECTID"), _
                                oFamilyKey.Value("LOCKINDIC"), _
                                oFamilyKey.Value("VALIDEND"), _
                                oFamilyKey.Value("VALIDBEGIN"), _
                                oFamilyKey.Value("RECORDNR"))
    If Err.Number = 0 Then                      ' AND IF THAT WORKED, THEN FINALLY
      ' WE CAN GET THE INFORMATION WE WANT BY CALLING GETDETAIL. THIS BAPI
      ' USES THE KEY INFORMATION IN THE FamilyMem OBJECT TO IDENTIFY THE SAP
      ' DATA THAT SHOULD BE RETURNED....VERY COMPLEX!!!
      oFamilyMem.GetDetail Return:=oReturn, _
                    Firstname:=sFirstName, _
                    Initials:=sSecondName, _
                    Lastname:=sLastName, _
                    Gender:=sGender, _
                    Dateofbirth:=sBirthdate
      If oReturn.Type <> "E" Then
        Print #2, sPerno; cSep; _
              sFirstName; cSep; _
              sSecondName; cSep; _
              sLastName; cSep; _
              sGender; cSep; _
              sBirthdate; cSep; _
              " "
      End If
    Else
      Err.Clear
    End If
```

```
    Next oFamilyKey
  End If
End Sub
```

==========

-----End of Message-----

Exponential form to general

-----Original Message-----

Subject: Exponential form to general

hi,

can anyone help me in BAPI

1)In BAPI_BILLINGDOC_CANCEL1

i am not able to under TESTRUN.how can we test it

2)BAPI_SALESORDER_CHANGE

3)BAPI_TRANSACTION_COMMIT

can anyone help me out on these.i dont know BAPI.Can any one advice me where can i read it?

-----Reply Message-----

Subject: RE: Exponential form to general

Hi,

You can test the bapi's by passing the required parameters, Actually you can use the BAPI_SALESORDER_CHANGE when you want to change the existing sales order. BAPI_TRANSACTION_COMMIT is necessary to commit the changes, if you won't call the BAPI_TRANSACTION_COMMIT the values won't store in database.

The doucmentation is not available in English version, if yo want to see the similar once check SD_SALESDOCUMENT_CHANGE function module, both will do same work. You can get the documentation for this function module in GOTO-->DOCUMENTAION.

regards

-----End of Message-----

COMMIT WORK and BAPI_TRANSACTION_COMMIT

-----Original Message-----

Subject: COMMIT WORK and BAPI_TRANSACTION_COMMIT

Does anybody know what's the difference between the two?

-----Reply Message-----

Subject: RE: COMMIT WORK and BAPI_TRANSACTION_COMMIT

Hi,

Commit work is used when you code directly in ABAP and make changes in the database and want to commit the database.

BAPI_TRANSACTION_COMMIT is used when you make changes to the SAP database by calling a BAPI from outside SAP and want to commit the database. When you use a BAPI, you can not directly use commit work, instead you are allowed to use only BAPI_TRANSACTION_COMMIT.

Regards,

-----Reply Message-----

Subject: RE: COMMIT WORK and BAPI_TRANSACTION_COMMIT

I would sasy the diff lies more in the way u want to call Commit Work.

With BAPI_TRANSACTION_COMMIT ..the external systems have a way of deciding on whether to Commit or to Roll back Changes.

But with Commit Work u have to code it inside ure BAPI and the outside systems then have no chance or have

any hold over the commit...

so i guess the diff lies more in the way how u want to call commit ,either from outside or from within ure BAPI.

u can use both...

SAP though recommends using BAPI_Transaction_Commit and not using Commit_work in the BAPI...

but its upto u and ure middleware guy to decide how u want to do it..

BAPI vs Call transaction

-----Original Message-----

Subject: BAPI vs Call transaction

Hi all!

Could you explain me why a BAPI is faster than a call transaction?.

E.g. If i have the BAPI: create_sales_document and I could also do a call transaction to va01. Wich one is better?. Why?.

Thanks in advance...

-----Reply Message-----

Subject: RE: BAPI vs Call transaction

Hi, As of I know BAPI's R internally they RFC functions and they were implemented with Objects... ..May be because of this reason they R faqster..if it is wrong ..please don't mind...Regards...

-----Reply Message-----

Subject: RE: BAPI vs Call transaction

A BAPI is faster because it is updating the DB "directly" through ABAP code. A BDC with call transaction goes through the whole screen sequence like any user would do, simply put, it is filling screens.

Use BAPIs whenever possible.

-----Reply Message-----

Subject: RE: BAPI vs Call transaction

you use a special BAPI, cause this one uses CALL TRANSACTION to create a sales order.
A lot of BAPIS and IDOC - input FM use DIRECT INPUT instead of CALL TA. Much faster. And you can do a lot with them, you can't do as easy in CALL TA.
But the best reason for BAPIs is, that they are farely safe on release change or support package change.

Bye

-----Reply Message-----

Subject: RE: BAPI vs Call transaction

Thanks for your answers... They have been very useful...

-----End of Message-----

Difference and/or similarities between BAPI and IDOC's

-----Original Message-----

Subject: Difference and/or simmelarities between BAPI and IDOC's

Hello,

Can someone explain to me the difference and/or similarities between BAPI and IDOC's?

With regards,

-----Reply Message-----

Subject: RE: Difference and/or simmelarities between BAPI and IDOC's

There are many differences between IDOCs and BAPIs.

BAPIs in 3.1 are synchronous; in 4.+ they can be asynchronous (and I believe they then drive certain ALE/IDOCs).

BAPIs are called from the outside-in. That is, an external program invokes a BAPI that gets data from SAP to display or updates data in SAP. The BAPI concept does not include an event concept -- you cannot tell SAP that when certain events happen to a "business object", to fire a message or a file to an external system.

BAPIs are invokable from Java or C/C++ or Visual Basic (and I think some people are using Delphi).

In 3.1x there are very few BAPIs to use. In 4.+ SAP has added a large number.

BAPIs are not totally immune to upgrades but if they are to be retired you supposedly will have them supported for two releases. Whether those

are point or letter releases, I don't know. I believe that IDOCs may be more changable from release to release.

BAPIs are reasonably well documented and there is a common place to look to see what is available. IDOCs -- I have heard -- are poorly documented in terms of finding them, and IDOCs were done differently by different groups in SAP.

BTW, you can also use Java, C/C++, Visual Basic, ... to invoke RFCs in SAP and get or update data. That's how the BAPIs work since they utimately are sets of RFC calls (written to a design spec for BAPIs).

Hope I haven't misstated any of the details.

-----End of Message-----

BAPI and ABAP objects

-----Original Message-----
Subject: BAPI and ABAP objects

Hi,

Can anybody tell why BAPI is not integrated with ABAP objects?

Regards,

-----Reply Message-----
Subject: RE: BAPI and ABAP objects

Hello,

I'm curious to know myself. In general, BAPI is more in the macro level and ABAP objects in the micro. I hope they unify
those two in future releases.
I think that the technical reason is that BAPI can be used from outside of R/3. The method this is based on is RFC and
therefore all the BAPIs are function modules whereas methods of ABAP objects are not.

If you get interesting answers please let me know too.

Thanks,

-----End of Message-----

BAPI Licensing

-----Original Message-----

Subject: BAPI Licensing

Hello,

I was wondering if anyone out there has had exposure to the licensing issues relating to BAPI's. We are developing an internet application (using BAPI's not Web Studio) primarily focussed on enabled sales order entry on the Web. We seem to be getting mixed signals from SAP as to the appropriate licensing fees that need to be paid for this. Thus far three models have been floated, including:

1. Each user of the Web application attracts the same annual license fee as normal SAP user - i.e. $5000 per user 2. Each time a BAPI is called a fixed fee is charged - i.e. $10 per BAPI call 3. Each BAPI used in the application attracts an annual per user fee - i.e. $500 per BAPI per user

What I can't understand is that a model has not been adopted somewhere already. Surely there are some sites using BAPI's in a production environment somewhere. And if so, surely the licensing issues have been resolved for those sites.

Is there anyone out there who has a live application who knows about these issues? If so it would be useful to have a reference for the SAP guys here downunder.

Personally I don't see models 1 and 2 as particularly feasible, model 1 as the costs for this would be astronomical (we're supposed to be saving money with Web orders) and model 2 as it would be a nightmare gathering the appropriate data for billing. Model 3, however, presents the issue of whether a custom developed BAPI attracts the same annual fee. If so, you might be inclined to

develop a lot of RFC functions and not BAPIs to plug functionality gaps. :) You might also use the underlying WWW_**** function modules in the BAPIs instead of the actual BAPIs. :)

The model that SAP have not presented is model 4 in which you only pay per concurrent user logged into SAP as per a standard license agreement. The Web application only uses one SAP username but if it is getting good usage there will be concurrent users reflected in SAP.

Anyone have any thoughts, or specifics on their own licensing agreements?

Cheers,

-----Reply Message-----
Subject: RE: BAPI Licensing

We have had a similar problem and went the route of individual RFC's in a batch mode (text file import) to get around this extortion. The apps we are using are employee self service for HR and would have cost $5000 per employee (30,000+ employees) who may access the system once or twice a year.

If you manage to get a licensing agreement with SAP for direct BAPI use that makes since please post a message.

-----End of Message-----

BAPI Conventions

Methods
Parameters
Standardized BAPIs
Standardized Parameters
Important things to remember..
BAPI/ALE Integration

Methods

- If the BAPI to be implemented is a standardized BAPI, use the generic names, for example, GetList, GetDetail.
- The method name must be in English (maximum 30 characters).
- The individual components of a BAPI name are separated by the use of upper and lower case.Example: GetList

  Underscores ("_") are not allowed in BAPI names.
- Each BAPI has a return parameter that is either an export parameter or an export table.
- So that customers can enhance BAPIs, each BAPI must have an ExtensionIn and an ExtensionOut parameter.

Parameters

- If standardized parameters are used, you have to use the names specified for standardized parameters.
- BAPI parameter names should be as meaningful as possible. Poorly chosen names include abbreviations and technical names (e.g. "flag", table names, etc.).

  The parameter and field names must be in English with a maximum of 30 characters.
- The components of a parameter name in the BOR are separated by upper and lower case letters to make them easier to read. Example: CompanyCodeDetail
- Values that belong to each other semantically should be grouped together in one structured parameter, instead of using several scalar parameters.
- For ISO-relevant fields (country, language, unit of measure, currency), additional fields for ISO codes are provided.
- Unit of measure fields must accompany all quantity fields and currency identifiers must accompany currency amount fields.

Standardized BAPIs

Some BAPIs provide basic functions and can be used for most SAP business object types. These BAPIs should be implemented the same for all business object types. Standardized BAPIs are easier to use and prevent users having to deal with a number of different BAPIs. Whenever possible, a standardized BAPI must be used in preference to an individual BAPI.

The following standardized BAPIs are provided:

Reading instances of SAP business objects

GetList ( )	With the BAPI GetList you can select a range of object key values, for example, company codes and material numbers. The BAPI GetList() is a class method.
GetDetail()	With the BAPI GetDetail() the details of an instance of a business object type are retrieved and returned to the calling program. The instance is identified via its key. The BAPI GetDetail() is an instance method.

BAPIs that can create, change or delete instances of a business object type

The following BAPIs of the same object type have to be programmed so that they can be called several times within one transaction. For example, if, after sales order 1 has been created, a second sales order 2 is created in the same transaction, the second BAPI call must not affect the consistency of the sales order 2. After completing the transaction with a COMMIT WORK, both the orders are saved consistently in the database.

Create( ) and CreateFromData( )	The BAPIs Create() and CreateFromData() create an instance of an SAP business object type, for example, a purchase order. These BAPIs are class methods.
Change( )	The BAPI Change() changes an existing instance of an SAP business object type, for example, a purchase order. The BAPI Change () is an instance method.
Delete( ) and Undelete( )	The BAPI Delete() deletes an instance of an SAP business object type from the database or sets a deletion flag. The BAPI Undelete() removes a deletion flag. These BAPIs are instance methods.
Cancel ( )	Unlike the BAPI Delete(), the BAPI Cancel() cancels an instance of a business object type. The instance to be cancelled remains in the database and an additional instance is created and this is the one that is actually canceled. The Cancel() BAPI is an instance method.
Add<subobject> ( ) and Remove<subobject> ( )	The BAPI Add<subobject> adds a subobject to an existing object instance and the BAPI and Remove<subobject> removes a subobject from an object instance. These BAPIs are instance methods.

BAPIs for Mass Data Processing

The BAPIs listed above for creating and changing data can also be used for mass processing. For more information see BAPIs for Mass Data Transfer [Extern]

BAPIs for Replicating Business Object Instances

Replicate( ) and SaveReplica( )	The BAPIs Replicate() and SaveReplica() are implemented as methods of replicable business object types. They enable specific instances of an object type to be copied to one or more different systems. These BAPIs are used mainly to transfer data between distributed systems within the context of Application Link Enabling (ALE). These BAPIs are class methods.

Other Less Used Standardized BAPIs

- Programming GetStatus() BAPIs [Extern]
- Programming ExistenceCheck() BAPIs [Extern]

Standardized Parameters

There are some parameters that can be created for various BAPIs because they contain the same or the equivalent data in all BAPIs. They should be implemented the same in all BAPIs.

Address parameters	Specific reference structures are defined for address parameters in BAPIs. You should copy these structures to use in your BAPI, especially if the underlying object type uses the central address management (CAM).
Change Parameters	In BAPIs that cause database changes (for example, Change() and Create() BAPIs) you must be able to distinguish between parameter fields that contain modified values and parameter fields that have not been modified. This distinction is made through the use of standardized parameters.
Extension parameters	The parameters ExtensionIn and ExtensionOut provides customers with a mechanism that enables BAPIs to be enhanced without modifications.
Return Parameters	Each BAPI must have an export return parameter for returning messages to the calling application. To provide application programmers with a consistent error handling process for BAPI calls, all return parameters must be implemented in the same, standardized way.
Selection Parameters	Standardized selection parameters are used in BAPIs that can be used to search for specific instances of a business object type (e.g. in GetList()). These parameters enable the BAPI caller to specify the relevant selection criteria.
Test Run Parameters	The parameter TestRun is used in write BAPIs (Create() and Change()), to check the entries for the object instance in the database before actually creating the object instance. The creation of the object instance is only simulated and data is not updated.
Text Transfer Parameters	To transfer BAPI documentation texts (e.g. the documentation of a business object type), you have to create standardized text transfer parameters.

Important things to remember..

It is important to follow the guidelines below when develop9ng BAPIs:

- BAPIs must not contain CALL TRANSACTIO or SUBMIT REPORT
- BAPIs must not invoke a COMMIT WORK. instead use the BAPI TransactionCommit to execute the com mit after the BAPI has executed.
- BAPI structures must not use includes.
- There should be no functional dependecies between two BAPIs
- BAPIs must perform there own authorization check
- BAPIs should not use dialogs
- BAPIs must not cause the program to abort or terminate. re4levant messages must be communicated th rough the return parameter.

BAPI/ALE Integration

When you use the BAPIs for asynchronous messagning, the application in the sendig systen calls the generate d ALE IDoc interface isntead of the BAPI.

Asynchronous BAPIs use the ALE interface this way:

- Creates an IDOC from the BAPI data
- Sends the IDOC to the target system
- Receives the IDOC in trhe target system, crreates the BAPI data from the IDoc and calls the BAPI

An ALE interface for a BAPi is created in transaction *BDBG*.

What is the different between ALE, IDOC and BAPI?

ALE

ALE is SAP proprietary technology that enables data communications between two or more SAP R/3 systems a nd/or R/3 and external systems. When a new enterprise resource planning (ERP) solution such as R/3 is imple mented, companies have to interface the ERP system with legacy systems or other ERP systems.

ALE provides intelligent mechanisms where by clients can achieve integration as well as distribution of applicati ons and data.

ALE technology facilitates rapid application prototyping and application interface development, thus reducing impl ementation time.

The ALE components are inherently integrated with SAP applications and are robust, leading to a highly reliable system.

ALE comes with application distribution/integration scenarios as well as a set of tools, programs, data definitions, and methodologies that you can easily configure to get an interface up and running.

BAPI

BAPIs provide a stable, standardized method for third-party applications and components to integrate into the Business Framework. These interfaces are being specified as part of SAP's initiative with customers, partners and leading standards organizations. Also, SAP has implemented the emerging Object Application Group (OAG) specifications with BAPIs.

Pros and Cons for both BAPI and Call Transaction

BAPI

One of the big plusses for BAPIs is that the interface and function are not supposed to change. This is a big plus when you do upgrades or hot packs because the transaction can change (format, required inputs etc) which means you then need to update the call transaction.

Some of the BAPIs are better documented and easier to use than others.

You usually need to perform the BAPI that actually does the COMMIT after you call your BAPI.

The Program coding for calling a BAPI is usually cleaner than setting up the screen flow etc for the Call Transaction.

You don't need to worry about special data circumstances interrupting the normal data flow of the screens and causing errors because of that.

BAPIs probably have better performance since they don't do the screen flow processing.

In general if the BAPI exists for the transaction you want to perform and you can figure out how to use it the BAPI is probably the best way to go.

This is just from my experience working with both BAPI and Call Transaction. I have had some very good successes with BAPIs, but very occasionally found that I could not get the BAPI to perform the update I needed.

ABAP Tips by: Heather R Woytash

The interface concept of the classic R/3 is based on two different strategies: Remote Function Calls (RFC) and data exchange through IDoc message documents. RFC makes direct and synchronous calls of a program in the remote system. If the caller is an external program it will call an RFC-enabled function in R/3 and if the calling program is the R/3 system it will call an
RFC-function in another R/3-system or it will call a non-R/3 program through a gateway-proxy (usually rfcexec.exe). BAPIs are a subset of the RFC-enabled function modules, especially designed as Application Programming Interface (API) to the SAP business object, or in other words: are function modules officially released by SAP to be called from external programs.

IDocs are text encoded documents with a rigid structure that are used to exchange data between R/3 and a foreign system. Instead of calling a program in the destination system directly, the data is first packed into an IDoc and then sent to the receiving system, where it is analyzed and properly processed. Therefore an IDoc data exchange is always an
asynchronous process. The significant difference between simple RFC-calls and IDoc data exchange is the fact, that every action performed on IDocs are protocolled by R/3 and IDocs can be reprocessed if an error occurred in one of the message steps.

While IDocs have to be understood as a data exchange protocol, EDI and ALE are typical use cases for IDocs. R/3 uses IDocs for both EDI and ALE to deliver data to the receiving system. ALE is basically the scheduling mechanism that defines when and between which partners and what kind of data will be exchanged on a regular or event triggered basis. Such a set-up is called an ALE-scenario.

The philosophical difference between EDI and ALE can be pinned as follows: If we send data to an external partner, we generally speak of EDI, while ALE is a mechanism to reliable replicate data between trusting systems to store a redundant copy of the IDoc data. The difference is made clear, when we think of a purchase order that is sent as an IDoc. If we send the purchase order to a supplier then the supplier will store the purchase order as a sales order. However, if we send the purchase order via ALE to another R/3 system, then the receiving system will store the purchase order also as a purchase order.

ALE (Application Link Enabling)

许多大企业把整个 SAP system 分布在多个 instance 上,每个 instance 专注于特定的业务.

例如: plant 中的物料采购在一个本地的 instance 上进行处理(the local system).

向供应商付款和总帐处理在公司中心 instance 上处理(the central system).

分布(distribute)不是简单的数据分割!!!

分布就是把数据分布在多个地点,但同时又能保证数据在整体上的一致性和完整性.

分布用到分割技术, 但是分布不是简单的分割.为了保证不同地点的数据的完整性

和一致性 different systems 利用 ALE 进行沟通.

The ALE concept always relates to an enterprise structure with areas that have central tasks and areas with tasks that are decentralized.

It may be practical for organizations to use separate application systems so that application components can be installed and operated on decentralized systems that are technically independent of each other.

The ALE concept supports the implementation and operation of distributed SAP applications. It is based on business-controlled messaging with consistent data storage on loosely coupled systems. The applications are integrated through the message exchange, not via a central database.

To implement a distributed, yet integrated system, the customer must specify in a logical model, which applications are to run on which systems and how the applications are to exchange data with each other.

On the technical side, the data exchange is carried out via IDocs (intermediate documents) as used in the EDI (Electronic Data Interchange) interface. On the application side, EDI supports information exchange between R/3 systems in different enterprises, whereas ALE supports information exchange within one enterprise. The ALE distribution mechanism is similar to the EDI mechanism. In ALE, business processes are distributed at the transaction level.

The procurement of materials via the plant in a company code can be handled in a local system. However, payments to vendors and the general ledger are managed in a central system.

In order that both organizations use the same master data, the head office distributes the master data to the local system. Thus the transaction data in the local systems for which IDocs exist can be sent to the head office without inconsistencies.

Examples of data that can be distributed:

Master data: Customers, vendors, G/L accounts, cost centers, cost elements, activity types

Transaction data: Incoming invoices, outgoing invoices, Financial Accounting line items, Controlling documents

The SAP standard system is delivered with some ALE scenarios. For example:

Central contract management in Purchasing

Distributed Inventory Management

Central Materials Management master data

The message types required for the distribution via IDocs are also delivered by SAP. You can also define your own ALE scenarios.

Business to Business Procurement works through ALE with an R/3 backend system

生成SO。Inbound IDOC Status Report for Sales Orders

Requirement:

Sales Orders are being created through inbound IDocs using FM 'EDI_DATA_INCOMING'. Now a Report is required to check the status of these Inbound IDocs along with Sales Orders generated against customer Purchase Orders.

Processing:

The report selects, 'ORDERS' IDoc numbers & status, generated between given time range, from table EDIDC. Further, it calls Function Module 'IDOC_READ_COMPLETELY' to get the IDoc details. Then required information is extracted by reading relevant field data of IDoc segments.

代码:

```
REPORT  Z_EDI_FILE_LOAD_STATUS_REPORT          .
*---------------------------------------------------------------------
* Staus Report for Inbound IDOCs ( Sales Orders )
*---------------------------------------------------------------------
* Program       : Z_EDI_FILE_LOAD_STATUS_REPORT
* Presented By   : www.rmtiwari.com
*---------------------------------------------------------------------
TABLES : EDIDC.
*---------------------------------------------------------------------
* ALV stuff
TYPE-POOLS: SLIS.
DATA: GT_FIELDCAT TYPE SLIS_T_FIELDCAT_ALV,
      GS_LAYOUT   TYPE SLIS_LAYOUT_ALV,
      GT_SORT     TYPE SLIS_T_SORTINFO_ALV,
      GT_LIST_TOP_OF_PAGE TYPE SLIS_T_LISTHEADER.

DATA : BEGIN OF T_REPORT OCCURS 0,
        IDOC_NO       TYPE EDI_DOCNUM,
        IDOC_DATE     TYPE SY-DATUM,
        IDOC_TIME     TYPE SY-UZEIT,
        SORDER_NO     TYPE VBELN,
        STP_NO        TYPE KNA1-KUNNR,
        STP_NAME(35)  TYPE C,
        STP_PHONE(12) TYPE C,
        PO_NO(15)     TYPE C,
        STATUS        TYPE C,
        S_TEXT(70)    TYPE C,
        ERROR(70)     TYPE C,
      END OF T_REPORT.

*------------------PARAMETER-----------------------------------------*
```

```
selection-screen begin of block date with frame title TEXT-S01.
select-options: UDATE for EDIDC-UPDDAT
                default SY-datum obligatory,    "Changed On
            UTIME for EDIDC-UPDTIM .            "Changed Time

selection-screen end   of block date.

*---------------------------------------------------------------------*
INITIALIZATION.

START-OF-SELECTION.
PERFORM SHOW_STATUS_REPORT.

*&---------------------------------------------------------------------*
*&      Form  alv_grid
*&---------------------------------------------------------------------*
*       text
*---------------------------------------------------------------------*
*  -->  p1        text
*  <--  p2        text
*---------------------------------------------------------------------*
FORM ALV_GRID.

  IF GT_FIELDCAT[] IS INITIAL.
    PERFORM FIELDCAT_INIT.
    PERFORM LAYOUT_INIT.
    PERFORM SORT_INIT.
  ENDIF.

  PERFORM GRID_DISPLAY.

ENDFORM.                    "alv_grid
*&---------------------------------------------------------------------*
*&      Form  layout_init
*&---------------------------------------------------------------------*

FORM LAYOUT_INIT.
  GS_LAYOUT-ZEBRA            = 'X'.
  GS_LAYOUT-CELL_MERGE       = 'X'.
```

```
  GS_LAYOUT-COLWIDTH_OPTIMIZE = 'X'.
  GS_LAYOUT-NO_VLINE          = ' '.
  GS_LAYOUT-TOTALS_BEFORE_ITEMS = ' '.

ENDFORM.                    " layout_init
*&---------------------------------------------------------------------*
*&      Form  fieldcat_init
*&---------------------------------------------------------------------*

FORM FIELDCAT_INIT.
  DATA: LS_FIELDCAT TYPE SLIS_FIELDCAT_ALV.

   CLEAR LS_FIELDCAT.
  LS_FIELDCAT-FIELDNAME    = 'IDOC_NO'.
  LS_FIELDCAT-KEY          = 'X'.
  LS_FIELDCAT-REPTEXT_DDIC = 'IDOC'.
  LS_FIELDCAT-OUTPUTLEN    = 10.
* Fix for ALV print bug, which puts 'N/A' over last digit
* Set inttype to 'N' to stop corruption of printed ALV cell.
  LS_FIELDCAT-INTTYPE = 'N'.
  APPEND LS_FIELDCAT TO GT_FIELDCAT.

  CLEAR LS_FIELDCAT.
  LS_FIELDCAT-FIELDNAME    = 'IDOC_DATE'.
  LS_FIELDCAT-REPTEXT_DDIC = 'Creation Date'.
  LS_FIELDCAT-OUTPUTLEN    = 10.
  APPEND LS_FIELDCAT TO GT_FIELDCAT.

  CLEAR LS_FIELDCAT.
  LS_FIELDCAT-FIELDNAME    = 'IDOC_TIME'.
  LS_FIELDCAT-REPTEXT_DDIC = 'Creation Time'.
  LS_FIELDCAT-OUTPUTLEN    = 8.
  APPEND LS_FIELDCAT TO GT_FIELDCAT.

  CLEAR LS_FIELDCAT.
  LS_FIELDCAT-FIELDNAME    = 'STATUS'.
  LS_FIELDCAT-REPTEXT_DDIC = 'St'.
```

```
  LS_FIELDCAT-OUTPUTLEN    = 2.
  APPEND LS_FIELDCAT TO GT_FIELDCAT.

  CLEAR LS_FIELDCAT.
  LS_FIELDCAT-FIELDNAME    = 'ERROR'.
  LS_FIELDCAT-REPTEXT_DDIC = 'Message'.
  LS_FIELDCAT-OUTPUTLEN    = 70.
  APPEND LS_FIELDCAT TO GT_FIELDCAT.


  CLEAR LS_FIELDCAT.
  LS_FIELDCAT-FIELDNAME    = 'STP_NO'.
  LS_FIELDCAT-REPTEXT_DDIC = 'S.T.Party No'.
  LS_FIELDCAT-OUTPUTLEN    = 10.
  APPEND LS_FIELDCAT TO GT_FIELDCAT.

  CLEAR LS_FIELDCAT.
  LS_FIELDCAT-FIELDNAME    = 'STP_NAME'.
  LS_FIELDCAT-REPTEXT_DDIC = 'Sold to Party Name'.
  LS_FIELDCAT-OUTPUTLEN    = 35.
  APPEND LS_FIELDCAT TO GT_FIELDCAT.


  CLEAR LS_FIELDCAT.
  LS_FIELDCAT-FIELDNAME    = 'PO_NO'.
  LS_FIELDCAT-REPTEXT_DDIC = 'Purch Order'.
  LS_FIELDCAT-OUTPUTLEN    = 15.
  APPEND LS_FIELDCAT TO GT_FIELDCAT.

  CLEAR LS_FIELDCAT.
  LS_FIELDCAT-FIELDNAME    = 'STP_PHONE'.
  LS_FIELDCAT-REPTEXT_DDIC = 'S.T.Party Phone'.
  LS_FIELDCAT-OUTPUTLEN    = 15.
  APPEND LS_FIELDCAT TO GT_FIELDCAT.

ENDFORM.                    "fieldcat_init

*&---------------------------------------------------------------------*
*&      Form  sort_init
```

```
*&---------------------------------------------------------------------*

FORM SORT_INIT.
  DATA: LS_SORT TYPE SLIS_SORTINFO_ALV.
*
  CLEAR LS_SORT.
  LS_SORT-FIELDNAME = 'IDOC_DATE'.
  LS_SORT-SPOS      = 1.
  LS_SORT-UP        = 'X'.
  APPEND LS_SORT TO GT_SORT.

  CLEAR LS_SORT.
  LS_SORT-FIELDNAME = 'IDOC_TIME'.
  LS_SORT-SPOS      = 2.
  LS_SORT-UP        = 'X'.
  APPEND LS_SORT TO GT_SORT.

  CLEAR LS_SORT.
  LS_SORT-FIELDNAME = 'STATUS'.
  LS_SORT-SPOS      = 3.
  LS_SORT-UP        = 'X'.
  APPEND LS_SORT TO GT_SORT.

  CLEAR LS_SORT.
  LS_SORT-FIELDNAME = 'IDOC_NO'.
  LS_SORT-SPOS      = 4.
  LS_SORT-UP        = 'X'.
  APPEND LS_SORT TO GT_SORT.

ENDFORM.                    "sort_init
*&---------------------------------------------------------------------*
*&      Form  grid_display
*&---------------------------------------------------------------------*

FORM GRID_DISPLAY.

  CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
    EXPORTING
```

```
      IS_LAYOUT     = GS_LAYOUT
      IT_FIELDCAT   = GT_FIELDCAT
      IT_SORT       = GT_SORT
      i_callback_program      = SY-REPID
      I_CALLBACK_TOP_OF_PAGE = 'TOP_OF_PAGE'
      I_DEFAULT     = ' '
      I_SAVE        = 'X'
    TABLES
      T_OUTTAB      = T_REPORT
    EXCEPTIONS
      PROGRAM_ERROR = 1
      OTHERS        = 2.

ENDFORM.                    "grid_display
*&---------------------------------------------------------------------*
*&      Form  COMMENT_BUILD
*&---------------------------------------------------------------------*
*       Processing of listheader
*----------------------------------------------------------------------*
FORM COMMENT_BUILD USING P_FK_LIST_TOP_OF_PAGE TYPE SLIS_T_LISTHEADER.

  DATA: LS_LINE TYPE SLIS_LISTHEADER.
  REFRESH P_FK_LIST_TOP_OF_PAGE.
* List Heading : Typ H
  CLEAR LS_LINE.
  LS_LINE-TYP  = 'H'.
  LS_LINE-INFO  = 'Sales Order Interface: Z_EDI_FILE_LOAD'.
  APPEND LS_LINE TO P_FK_LIST_TOP_OF_PAGE.

* List : Typ S
  clear LS_LINE.
  LS_LINE-typ  = 'S'.
  LS_LINE-key  = 'Date Range:'.
  LS_LINE-info  = UDATE-low.
  if not UDATE-high is initial.
    write ' To ' to  LS_LINE-info+30.
    LS_LINE-info+36 = UDATE-high.
  endif.
```

```
  APPEND LS_LINE TO P_FK_LIST_TOP_OF_PAGE.

ENDFORM.                        " COMMENT_BUILD

*---------------------------------------------------------------------*
*       FORM TOP_OF_PAGE                                              *
*---------------------------------------------------------------------*
*       Ereigniss TOP_OF_PAGE                                         *
*       event     TOP_OF_PAGE
*---------------------------------------------------------------------*
FORM TOP_OF_PAGE.

  PERFORM COMMENT_BUILD  USING gt_LIST_TOP_OF_PAGE[].
  CALL FUNCTION 'REUSE_ALV_COMMENTARY_WRITE'
    EXPORTING
      IT_LIST_COMMENTARY = GT_LIST_TOP_OF_PAGE.

ENDFORM.                "TOP_OF_PAGE



*&---------------------------------------------------------------------*
*&       Form  show_status_report
*&---------------------------------------------------------------------*

FORM SHOW_STATUS_REPORT .
* Report to show status.

  DATA: BEGIN OF T_TEDS2 OCCURS 0.
        INCLUDE STRUCTURE TEDS2.
  DATA: END OF T_TEDS2.

  DATA: BEGIN OF T_IDOC_CONTROL_TMP OCCURS 0.
        INCLUDE STRUCTURE EDIDC.
  DATA: END OF T_IDOC_CONTROL_TMP.

  CONSTANTS: C_STATUS_IN_IDOC_POSTED      LIKE EDIDC-STATUS VALUE '53'.
```

```
  DATA : T_EDIDS TYPE STANDARD TABLE OF EDIDS WITH HEADER LINE.
  DATA : T_EDIDD TYPE STANDARD TABLE OF EDIDD WITH HEADER LINE.

  DATA : GV_PARTNER_SEG TYPE E1EDKA1,
       GV_PO_REF_SEG  TYPE E2EDK02.


* Get text for status values
  SELECT * FROM TEDS2 INTO TABLE T_TEDS2 WHERE LANGUA = SY-LANGU.

* Read the IDoc's status after processing
  SELECT * FROM EDIDC
    INTO TABLE T_IDOC_CONTROL_TMP
   WHERE UPDDAT IN UDATE
     AND UPDTIM IN UTIME
     AND MESTYP = 'ORDERS'.

  LOOP AT T_IDOC_CONTROL_TMP.

*   IDoc has been processed, since control record changed.
    READ TABLE T_TEDS2 WITH KEY STATUS = T_IDOC_CONTROL_TMP-STATUS.
    T_REPORT-IDOC_NO = T_IDOC_CONTROL_TMP-DOCNUM.
    T_REPORT-IDOC_DATE = T_IDOC_CONTROL_TMP-CREDAT.
    T_REPORT-IDOC_TIME = T_IDOC_CONTROL_TMP-CRETIM.
    T_REPORT-S_TEXT = T_TEDS2-DESCRP.
    IF T_IDOC_CONTROL_TMP-STATUS = C_STATUS_IN_IDOC_POSTED.
*     ok status
      T_REPORT-STATUS = 'S'.
    ELSE.
*     error status
      T_REPORT-STATUS = 'E'.
    ENDIF.

*     Get IDoc details.

    CALL FUNCTION 'IDOC_READ_COMPLETELY'
      EXPORTING
        DOCUMENT_NUMBER        = T_REPORT-IDOC_NO
      TABLES
```

```
    INT_EDIDS          = T_EDIDS
    INT_EDIDD          = T_EDIDD
  EXCEPTIONS
    DOCUMENT_NOT_EXIST     = 1
    DOCUMENT_NUMBER_INVALID = 2
    OTHERS             = 3.

*  Get Error status
  READ TABLE T_EDIDS WITH KEY STATUS = T_IDOC_CONTROL_TMP-STATUS.
  IF SY-SUBRC EQ 0.
   REPLACE FIRST OCCURRENCE OF '&1' IN T_EDIDS-STATXT
                WITH T_EDIDS-STAPA1.
   REPLACE FIRST OCCURRENCE OF '&2' IN T_EDIDS-STATXT
                WITH T_EDIDS-STAPA2.
   REPLACE FIRST OCCURRENCE OF '&3' IN T_EDIDS-STATXT
                WITH T_EDIDS-STAPA3.
   REPLACE FIRST OCCURRENCE OF '&4' IN T_EDIDS-STATXT
                WITH T_EDIDS-STAPA4.

   REPLACE FIRST OCCURRENCE OF '&' IN T_EDIDS-STATXT
                WITH T_EDIDS-STAPA1.
   REPLACE FIRST OCCURRENCE OF '&' IN T_EDIDS-STATXT
                WITH T_EDIDS-STAPA2.
   REPLACE FIRST OCCURRENCE OF '&' IN T_EDIDS-STATXT
                WITH T_EDIDS-STAPA3.
   REPLACE FIRST OCCURRENCE OF '&' IN T_EDIDS-STATXT
                WITH T_EDIDS-STAPA4.


   T_REPORT-ERROR = T_EDIDS-STATXT.
  ENDIF.
  LOOP AT T_EDIDD.

   CASE T_EDIDD-SEGNAM.

    WHEN 'E1EDKA1'.
     GV_PARTNER_SEG = T_EDIDD-SDATA.

     CLEAR : T_REPORT-STP_NAME.
```

```
        CALL FUNCTION 'CONVERSION_EXIT_ALPHA_INPUT'
          EXPORTING
            INPUT  = GV_PARTNER_SEG-PARTN
          IMPORTING
            OUTPUT = T_REPORT-STP_NO.

        SELECT SINGLE NAME1 TELF1
          INTO (T_REPORT-STP_NAME,T_REPORT-STP_PHONE)
          FROM KNA1
         WHERE KUNNR = T_REPORT-STP_NO.

      WHEN 'E1EDK02'.
        GV_PO_REF_SEG = T_EDIDD-SDATA.
        T_REPORT-PO_NO = GV_PO_REF_SEG-BELNR.
    ENDCASE.

   ENDLOOP.

   APPEND T_REPORT.
  ENDLOOP .

  SORT T_REPORT BY STATUS IDOC_NO.

* Show Report
  PERFORM ALV_GRID.

ENDFORM.                    " show_status_report
```

如何从 SAP 中查找 BADI

［日期：2006-11-05］ 来源：sapsky 作者：sapsky ［字体：大 中 小］

BADI 作为 SAP 的第三代用户出口，他的应用也越来越广泛，但如何找到合适的 badi 是许多 abap 程序员的困惑。我这里就介绍一下我个人的应用的经验，供大家参考。

1、badi 对象的信息存储在 SXS_INTER, SXC_EXIT, SXC_CLASS 和 SXC_ATTR 这四个表中（参见 SECE 包）；

2、sap 程序都会调用 cl_exithandler=>get_instance 来判断对象是否存在，并返回实例；其实 get_instance 就是对上述几个表和他们的视图（V_EXT_IMP 和 V_EXT_ACT）进行查询和搜索。

3、基于这个机理，我查用 ST05 来监控一个 TCODE 来跟踪，然后选择查找有关上述几个表和视图的操作，就可获得相关 BADI。

4、se18 查找接口，se19 实现接口就可以实现用户增强。

示例：用 LE_SHP_DELIVERY_PROC 控制跨月 Cancel

```
METHOD IF_EX_LE_SHP_DELIVERY_PROC~CHANGE_DELIVERY_HEADER .
data : thismonth(2) type c.
data : wa_likp type line of SHP_LIKP_T.
data : wa_log type line of SHP_BADI_ERROR_LOG_T.
clear ct_log[],thismonth.
thismonth = sy-datum+4(2). "----->這一個月的月份
loop at it_xlikp into wa_likp.
check IS_V50AGL-WARENAUSG_STORNO ='X'."--->代表作 GI cancel
if wa_likp-WADAT_IST+4(2) < thismonth.
wa_log-VBELN = cs_likp-vbeln.
wa_log-MSGTY = 'E'. "錯誤訊息
wa_log-MSGID = 'ZDN_ERROR'. "這一個 class 要自己建
wa_log-MSGNO = '001'.
append wa_log to ct_log. "Error log 寫入
endif.
endloop.
```

SAP 用户出口的类型

sap 的用户出口总共有三代:

1、第一代

sap 提供一个空代码的子过程，在这个子过程中用户可以添加自己的代码，控制自己的需求。这类增强都需要修改 sap 的标准代码。

示例：USEREXIT.. in SAPMV45A

2、第二代

sap 提供的是 CUSTOMER-FUNCTION，它是通过 SMOD 和 CMOD 完成实现。参见我的 http://blog.csdn.net/CompassButton/archive/2006/08/31/1150258.aspx

3、第三代

sap 提供的第三代的用户出口就是 BADI，他的调用方式是 CALL METHOD (instance)，（相关的 TCODE 是 SE18 和 SE19），你可以通过 EXIT_HANDLER 这个单词查找 BADI。

SD 相关的 BADI

[日期：2006-11-05] 来源：sapsky 作者：sapsky [字体：大 中 小]

HU_BADI	Business Add-Ins for Handling Units
LE_SHP_BADI	Business Add-Ins in Shipping
LE_TRA_BADI	Business Add-Ins in Transportation
LE_WM_BADI	Business Add-Ins in Warehouse Management
MRM_BADI	Business Add-Ins in Invoice Verification
PL_PACKINST_BADI	Business Add-In in the Packing Instruction
S_BADI_FORMULA_BUILDER	BADI Implementation with Formula Builder
VA_BADI	BADIs R/3 Sales
VF_BADI	BAdIs for Billing

如何使用ＢＡＤＩ修改ＰＯ

Requirement:

Populate EVERS [Shipping Point] at the time of purchase order ceration. Shipping point should be derivated from the shipping point on sales order [if PO created from in reference to a sales order].

Processing:

This is an example to show - how to achieve post-processing [follow-on processing] functionality using BADIs [Business Add-inn] or user-exits.

- Find the relevant BADI using transaction SE18. In this case BADI ME_PURCHDOC_POSTED is used.
- Further, implement the BADI using transaction SE19.
- In Attributes section of BADI, define a STATIC attribute as PO_NUMBER. Static means the attribute will keep its value between the calls. This will be checked to ensure that same PO will not be processed twice. Also these kind of user-exits and BADIs might get called recursively and get caught into an infinite loop, if not coded properly. Rememeber that this BADI is at the time of PO save and then you are again trying to change & save the Purchase Order from within the BADI.
- BAPI to change Purchase Order 'BAPI_PO_CHANGE' will be called IN BACKGROUND TASK to ensure that it will be called when COMMIT WORK is encountered.
- Don't forget to activate the BADI implementation in SE19.

method IF_EX_ME_PURCHDOC_POSTED~POSTED .

```
DATA: wa_ekpo like line of IM_EKPO,
      lt_po_item type standard table of BAPIMEPOITEM,
      lt_po_itemx type standard table of BAPIMEPOITEMX,
      wa_po_item type BAPIMEPOITEM,
      wa_po_itemx type BAPIMEPOITEMX,
      lt_return type standard table of BAPIRET2.


*data: ls_ebeln type BAPIMEPOHEADER-PO_NUMBER.

check im_ekko-ebeln ne PO_NUMBER.

PO_NUMBER = im_ekko-ebeln.

LOOP AT IM_EKPO into wa_ekpo.

  wa_po_item-PO_ITEM = wa_ekpo-ebelp.
* EVERS to be derived
  wa_po_item-SHIPPING = 'C'.
  APPEND wa_po_item to lt_po_item .

  wa_po_itemx-PO_ITEM = wa_ekpo-ebelp.
  wa_po_itemx-SHIPPING = 'X'.
  APPEND wa_po_itemx to lt_po_itemx.
ENDLOOP.

 CALL FUNCTION 'BAPI_PO_CHANGE' IN BACKGROUND TASK
   EXPORTING
    purchaseorder              = PO_NUMBER
*    POHEADER                   =
*    POHEADERX                  =
*    POADDRVENDOR               =
*    TESTRUN                    =
*    MEMORY_UNCOMPLETE          =
*    MEMORY_COMPLETE            =
*    POEXPIMPHEADER             =
*    POEXPIMPHEADERX            =
*    VERSIONS                   =
    NO_MESSAGING               = 'X'
```

```
    NO_MESSAGE_REQ            = 'X'
    NO_AUTHORITY              = 'X'
    NO_PRICE_FROM_PO          = 'X'
*   IMPORTING
*    EXPHEADER                =
*    EXPPOEXPIMPHEADER        =
   TABLES
    RETURN                    = lt_return
    POITEM                    = lt_po_item
    POITEMX                   = lt_po_itemx
*    POADDRDELIVERY           =
*    POSCHEDULE               =
*    POSCHEDULEX              =
*    POACCOUNT                =
*    POACCOUNTPROFITSEGMENT   =
*    POACCOUNTX               =
*    POCONDHEADER             =
*    POCONDHEADERX            =
*    POCOND                   =
*    POCONDX                  =
*    POLIMITS                 =
*    POCONTRACTLIMITS         =
*    POSERVICES               =
*    POSRVACCESSVALUES        =
*    POSERVICESTEXT           =
*    EXTENSIONIN              =
*    EXTENSIONOUT             =
*    POEXPIMPITEM             =
*    POEXPIMPITEMX            =
*    POTEXTHEADER             =
*    POTEXTITEM               =
*    ALLVERSIONS              =
*    POPARTNER                =
          .

endmethod.
```

ALE, EDI & IDoc 处理的重要 T-codes

A.主菜单

WEDI Main menu for EDI-related activities

BALE Main menu for ALE-related activities

SWLD Main menu for workflow-related activities

SALE Main area for ALE configuration

NACE Main menu for Message control configuration

.

B.IDoc 监视/检查

WE02 IDoc display

WE05 IDoc list

WE07 IDoc statistics

.

C.测试

WE19 Test tool for IDocs

WE12 Convert an outbound IDoc to an inbound IDoc

WE16 Process an incoming IDoc file

WE17 Process an incoming status file

.

D.IDocs 再处理

BD87 Manual processing of IDocs

ALE, EDI 处理中的几个重要系统标准程序(System Standard Programs)

RSNAST00: Selection Program for Issuing Output

RBDAPP01: Inbound Processing of IDocs Ready for Transfer

RBDMANIN: Start error handling for non-posted IDocs

RBDMIDOC: Creating IDoc Type from Change Pointers

RBDMOIND: Status Conversion with Successful tRFC Execution

RSEOUT00: Process all selected IDocs (EDI)

RSEIDOCM: CA-EDI: Active monitoring for IDoc processing

[推荐]创建动态内表

```
REPORT  zg_dynamic_ex LINE-SIZE 300.
TYPE-POOLS:   abap.
FIELD-SYMBOLS:<dyn_table> TYPE STANDARD TABLE,
            <dyn_wa>,
            <dyn_field>.
DATA: dy_table TYPE REF TO data,
      dy_line  TYPE REF TO data,
      xfc      TYPE       lvc_s_fcat,
      ifc      TYPE       lvc_t_fcat.
SELECTION-SCREEN BEGIN OF BLOCK b1 WITH FRAME.
PARAMETERS: p_table(30)  TYPE c DEFAULT 'T001'.
SELECTION-SCREEN END OF BLOCK b1.
START-OF-SELECTION.
  PERFORM get_structure.
  PERFORM create_dynamic_itab.
  PERFORM get_data.
  PERFORM write_out.
*&---------------------------------------------------------------------*
*&      Form  get_structure
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM get_structure.
  DATA: idetails      TYPE       abap_compdescr_tab,
        xdetails      TYPE       abap_compdescr.
  DATA: ref_table_des TYPE REF TO cl_abap_structdescr.
  ref_table_des ?=
   cl_abap_typedescr=>describe_by_name( p_table ).
```

```
  idetails[] = ref_table_des->components[].
  LOOP AT idetails INTO xdetails.
    CLEAR xfc.
    xfc-fieldname = xdetails-name.
    xfc-datatype  = xdetails-type_kind.
    xfc-inttype   = xdetails-type_kind.
    xfc-intlen    = xdetails-length.
    xfc-decimals  = xdetails-decimals.
    APPEND xfc TO ifc.
  ENDLOOP.
ENDFORM.                    "get_structure
*&---------------------------------------------------------------------*
*&      Form  create_dynamic_itab
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM create_dynamic_itab.
  CALL METHOD cl_alv_table_create=>create_dynamic_table
    EXPORTING
      it_fieldcatalog = ifc
    IMPORTING
      ep_table        = dy_table.
  ASSIGN dy_table->* TO <dyn_table>.
  CREATE DATA dy_line LIKE LINE OF <dyn_table>.
  ASSIGN dy_line->* TO <dyn_wa>.
ENDFORM.                    "create_dynamic_itab
*&---------------------------------------------------------------------*
*&      Form  get_data
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM get_data.
  SELECT * INTO TABLE <dyn_table> FROM (p_table).
ENDFORM.                    "get_data
```

```
*&---------------------------------------------------------------------*
*&      Form  write_out
*&---------------------------------------------------------------------*
*       text
*----------------------------------------------------------------------*
FORM write_out.
  LOOP AT <dyn_table> INTO <dyn_wa>.
    DO.
      ASSIGN COMPONENT sy-index OF STRUCTURE <dyn_wa> TO <dyn_field>.
      IF sy-subrc <> 0.
        EXIT.
      ENDIF.
      IF sy-index = 1.
        WRITE:/ <dyn_field>.
      ELSE.
        WRITE: <dyn_field>.
      ENDIF.
    ENDDO.
  ENDLOOP.
ENDFORM.                    "write_out
```

VB 调用 SAP RFC

调用 RFC 中的"RFC_READ_TABLE"方法，读取 R/3 系统中 CSKT）表中的数据。

方便起见，这里用 VBA 实现此功能。其 CLICK 事件代码如下：

```
Private Sub CommandButton1_Click()
ll = ConnectSAP()
ll = CallSAPFUNC()
End Sub

Function ConnectSAP()
Dim strStatus As String
Set oFunction = CreateObject("SAP.LogonControl.1")
Set oConnection = oFunction.NewConnection
oConnection.Client = "300"
oConnection.Language = "JA"
oConnection.user = "DEV00018"
```

```
oConnection.Password = "XUYT0520"
oConnection.ApplicationServer = "10.10.2.91"
oConnection.SystemNumber = "00"
result = oConnection.Logon(0, True)
If result <> True Then
Set oFunction = Nothing
Set oConnection = Nothing
strStatus = "No access to R/3 System"
Else
strStatus = "SAP R/3 Access OK "
End If
ll = MsgBox(strStatus, , "LOGIN")
End Function

Function CallSAPFUNC()
Dim oConnection As Object
Dim ofun As Object
Dim func As Object

Set ofun = CreateObject("SAP.FUNCTIONS") ' FUNCTION NAME
Set ofun.Connection = oConnection
Set func = ofun.Add("RFC_READ_TABLE")
func.Exports("QUERY_TABLE") = "CSKT" ' TABLE NAME
If func.Call = True Then
Set oline = func.tables.Item("DATA")
Row = oline.rowcount
Sheet2.Cells(1, 1) = "0000"
Sheet2.Cells(1, 2) = "言語"
Sheet2.Cells(1, 3) = "管理領域"
Sheet2.Cells(1, 4) = "原"
Sheet2.Cells(1, 5) = "有効終了日"
Sheet2.Cells(1, 6) = "一般名称"
Sheet2.Cells(1, 7) = "0"
Sheet2.Cells(1, 8) = "条件"
i = 2
Do While i <= 100
```

```
Sheet2.Cells(i, 1) = Mid(Trim(oline.Value(i, 1)), 1, 3)
Sheet2.Cells(i, 2) = Mid(Trim(oline.Value(i, 1)), 4, 1)
Sheet2.Cells(i, 3) = Mid(Trim(oline.Value(i, 1)), 5, 4)
Sheet2.Cells(i, 4) = Mid(Trim(oline.Value(i, 1)), 9, 10)
Sheet2.Cells(i, 5) = Mid(Trim(oline.Value(i, 1)), 19, 8)
Sheet2.Cells(i, 6) = Mid(Trim(oline.Value(i, 1)), 27, 20)
Sheet2.Cells(i, 7) = Mid(Trim(oline.Value(i, 1)), 47, 40)
Sheet2.Cells(i, 8) = Mid(Trim(oline.Value(i, 1)), 87, 26)
i = i + 1
Loop
End If
ll = MsgBox("END PROCESS", , "END")
End Function
```

用户出口

用户出口就是 SAP 中的 Customer Exits 或者 User Exits

什么叫用户出口呢？打个比方说吧，SAP 软件就象一根晾衣服的绳子，上面有数不清的衣架，多数衣架上已经挂上了衣服，就些衣服就 SAP 的标准程序，还有些衣架是空着的，这些就是“用户出口”，你可以把自己做的衣服(比如程序代码)挂到这些衣架上去--如果你觉得 SAP 给你准备的衣服不够穿或者不合身的话。

使用用户出口可以：
-不影响标准 SAP 源代码
-不影响软件升级

SAP 有四种基本用户出口的类型：
1.菜单出口-Menu Exits
定义自己的菜单
2.屏幕出口-Screen Exits
定义自己的屏幕
3.功能模块出口-Function Module Exits
在 SAP 应用程序中添加功能
4.关键字出口-Keyword Exits
在 ABAP/4 字典中的关键字数据元素添加文档。结果是你在使用这些数据元素的字段处按 F1 后会出现你自定义的说明文档

使用的方法是：首先定义(T-Code:CMOD)一个项目 Project(以管理你的增强，这里的项目和 PS 模块的项目可是两回事)，把你要使用的系统增加 Enhancement 分配给这个项目，编辑系统增强中的用户出口对象。

SAP 的用户出口和其它模块不太一样，其他模块基本采用上面说到的系统增强方法，SD 的子模块则是罗列了一大堆已经定义好的子程序(Include)--说实话，我比较喜欢这种方式，你可以直接在 SE38 中修改这些子程序，然后激活就可以了。

要编辑用户出口，你必须有开发的权限，另外，除了关键字出口外，其他的出口都需要你有一定的 ABAP/4 编程能力。

[推荐]如何调整 ABAP 程序的性能

1、使用 where 语句

不推荐

```
Select * from zflight.
 Check : zflight-airln = 'LF' and zflight-fligh = 'BW222'.
Endselect.
```

推荐

```
Select * from zflight where airln = 'LF' and fligh = '222'.
Endselect.
```

2、使用聚合函数

不推荐

```
Maxnu = 0.
Select * from zflight where airln = 'LF' and cntry = 'IN'.
 Check zflight-fligh > maxnu.
 Maxnu = zflight-fligh.
Endselect.
```

推荐

```
Select max( fligh ) from zflight into maxnu where airln = 'LF' and cntry = 'IN'.
```

3、使用视图代替基本表查询

不推荐

```
Select * from zcntry where cntry like 'IN%'.
 Select single * from zflight where cntry = zcntry-cntry and airln = 'LF'.
Endselect.
```

推荐

```
Select * from zcnfl where cntry like 'IN%' and airln = 'LF'.
Endselect.
```

4、使用 INTO table 代替 select endselect

不推荐

```
Refresh: int_fligh.
Select * from zflight into int_fligh.
 Append int_fligh. Clear int_fligh.
Endselect.
```

推荐

```
Refresh: int_fligh.
Select * from zflight into table int_fligh.
```

5、使用批量修改内表代替逐行修改

不推荐

```
Loop at int_fligh.
    If int_fligh-flag is initial.
        Int_fligh-flag = 'X'.
    Endif.
    Modify int_fligh.
Endloop.
```

推荐

```
Int_fligh-flag = 'X'.
Modify int_fligh transporting flag where flag is initial.
```

6、使用二分法查询，提高查询内表数据速度

不推荐

```
Read table int_fligh with key airln = 'LF'.
```

推荐

```
Read table int_fligh with key airln = 'LF' binary search.
```

7、两个内表添加使用批量增加代替逐行

不推荐

```
Loop at int_fligh1.
 Append int_fligh1 to int_fligh2.
Endloop.
```

推荐

```
Append lines of int_fligh1 to int_fligh2.
```

8、使用 table buffering

Use of buffered tables is recommended to improve the performance considerably. The buffer is bypassed while using the following statements

Select distinct

Select … for update

Order by, group by, having clause

Joins

Use the Bypass buffer addition to the select clause in order to explicitly bypass the buffer while selecting the data.

9、 使用 FOR ALL Entries

不推荐

```
Loop at int_cntry.
    Select single * from zfligh into int_fligh
        where cntry = int_cntry-cntry.
        Append int_fligh.
    Endloop.
```

推荐

```
Select * from zfligh appending table int_fligh
For all entries in int_cntry
Where cntry = int_cntry-cntry.
```

10、正确地使用 where 语句，使查询能使用索引

When a base table has multiple indices, the where clause should be in the order of the index, either a primary or a secondary index

To choose an index, the optimizer checks the field names specified in the where clause and then uses an index that has the same order of the fields. One more tip is that if a table begins with MANDT, while an index does not, there is a high possibility that the optimizer might not use that index.

11、正确地使用 MOVE 语句

Instead of using the move-corresponding clause it is advisable to use the move statement instead. Attempt should be made to move entire internal table headers in a single shot, rather than moving the fields one by one.

12、正确地使用 inner join

Let us take an example of 2 tables, zairln and zflight. The table zairln has the field airln, which is the airline code and the field lnnam, which is the name of the airline. The table zflight has the field airln, the airline code and other fields which hold the details of the flights that an airline operates.

Since these 2 tables a re logically joined by the airln field, it is advisable to use the inner join.

```
Select a~airln a~lnnam b~fligh b~cntry into table int_airdet
From zairln as a inner join zflight as b on a~airln = b~airln.
```

In order to restrict the data as per the selection criteria, a where clause can be added to the above inner join.

13、使用 sort by 代替 order by

14、避免使用 SELECT DISTINCT 语句

使用的 ABAP SORT + DELETE ADJACENT DUPLICATES 代替.

取得服务器活动的程序，包括用户、命令、正在运行的程序

有时需要知道此时服务器的状态，如 background job 在运行时，需要确定下相关的程序是否也在运行中。。

```
DATA: WITH_CPU TYPE X VALUE 0.

  DATA: BEGIN OF WP_TABL OCCURS 10.
        INCLUDE STRUCTURE WPINFO.
  DATA: END OF WP_TABL.

  REFRESH WP_TABL.
  CALL FUNCTION 'TH_WPINFO'
     EXPORTING
        WITH_CPU   = WITH_CPU
     TABLES
        WPLIST     = WP_TABL
     EXCEPTIONS
        SEND_ERROR = 1
        OTHERS     = 2.
```

常用的 Function Module

Indeed these powerful ABAP/4 functions are very interesting and can bring some advantages. Improve your home development easily. They belong to standard objects and should never be changed.

- Bp_event_raise
  Trigger an event from ABAP/4 program.
- Bp_joblog_read
  Fetch job log executions filling the structure TBTC5.
- G_set_get_all_values
  Fetch values from a set filling the structure RGSB4.
- Popup_to_confirm_loss_of_data
  Create a dialog box in which you make a question whether the user wishes to perform a processing step with loss of data.
- Popup_to_confirm_step
  Create a dialog box in which you make a question whether the user wishes to perform the step.
- Popup_to_confirm_with_message
  Create a dialog box in which you inform the user about a specific decision point during an action.

- Popup_to_confirm_with_value

  Create a dialog box in which you make a question whether the user wishes to perform a processing step with a particular object.
- Popup_to_decide

  Create a dialog box in which you require the user between the two processing alternatives, or to cancel the action.
- Popup_to_decide_with_message

  Create a dialog box in which you inform the user about a specific decision point via a diagnosis text.
- Popup_to_display_text

  Create a dialog box in which you display a two-line message.
- Rfc_system_info

  Fetch information from the current instance filling the structure FRCSI.
- Rs_send_mail_for_spoollist

  Send messages from ABAP/4 programs to SAPoffice. The structure SOLI may contain the message.
- Rzl_sleep

  Hang the current application from 1 to 5 seconds.
- Rzl_submit

  Submit a remote report.
- Sapgui_progress_indicator

  Set progress indicator on the left lower corner of the current window.
- Sd_print_terms_of_payment

  Format terms of payment according to base line date and payment terms.
- So_wind_spool_list

  Browse printer spool numbers according to user informed.
- So_spool_read

  Fetch printer spool according to the spool number informed.
- So_user_list_read

  List of all users filling the structure SOUD3.
- Spell_amount

  Return the amount in words filling the structure SPELL.
- Th_saprel

  Gather information from the current system including upgrade activities. It completes fields from the structure KKS03.

- Th_server_list

  Gather information of all instances filling the structure MSXXLIST.
- Th_user_list

  List of logged users filling the structure UINFO.
- Th_user_info

  Information about the current user. It completes fields from the structure KKS03.
- Th_wpinfo

  List of work processes filling the structure WPINFO.
- Ws_upload

  Transfer files from the frontend to the application server.
- Ws_download

  Transfer files from the application server to the frontend.
- Ws_excel

  Download files at the frontend in excel format.
- Ws_execute

  Execute an external program on the presentation server.
- Ws_file_delete

  Delete file at the frontend.
- Ws_volume_get

  Get the label from a frontend device.
- Ws_msg

  Create a dialog box in which you display an one-line message.

Note: These functions have been used and have worked as well as they were supposed to do.

一个只有 9 句的程序却可更改 SAP 标准程序

```
REPORT ZMODISAP .

DATA:ITAB_CODE(72) OCCURS 0 WITH HEADER LINE.

***Change client status,under this status,No access key is asked

UPDATE T000

SET CCCATEGORY = 'C'

CCCORACTIV = '2'

CCNOCLIIND = '3'.

*** Modify LSTRDU34

READ REPORT 'LSTRDU34' INTO ITAB_CODE .

INSERT 'SY-SUBRC = 0 .' INTO ITAB_CODE  INDEX 102.
```

```
INSERT REPORT 'LSTRDU34' FROM ITAB_CODE .
*** Modify LSTRDU44
READ REPORT 'LSTRDU44' INTO ITAB_CODE .
INSERT 'SY-SUBRC = 0 .' INTO itab_code index 101.
INSERT REPORT 'LSTRDU44' FROM ITAB_CODE .
```

如何在用户登录时 sap 时触发一特定程序执行

有时候需要在用户登录时 sap 时触发一特定程序去执行，sap 提供了两种方法 t.

1) 使用函数 **'NAVIGATION_SET_START_TCODE'** ,用该函数可以设置用户和一个 TCODE，该 TCODE 当用户登录 sap 系统后就可以自动执行。sap 对应的程序：ADMIN_SET_START_TRANSACTION_FO

2)使用用户出口

该用户出口的模块名称是：**SUSR0001**，你可以在该单元里增加你的代码进行相应的控制。.

SAP 用户登录增强示例

```
* Transaction CMOD -> Utiliteis -> SAP Enhancements
* Exit Name SUSR0001
* Double click EXIT_SAPLSUSF_001
* Double click ZXUSRU01
* Insert -> include zsesschk.
*
* zsesschk limits the number of login sessions per user
* in a certain client
* It runs from user exit SUSR0001 after the SAP Login
* n-1 is the number of concurrent sessions allowed

TABLES: UINFO.
DATA: N TYPE I VALUE 2.          "Upper limit of login sessions
DATA: OPCODE TYPE X VALUE 2, I TYPE I, A(60).
DATA: BEGIN OF BDC_TAB1 OCCURS 5.
      INCLUDE STRUCTURE BDCDATA.
DATA: END OF BDC_TAB1.
```

```
DATA: BEGIN OF USR_TABL OCCURS 10.
      INCLUDE STRUCTURE UINFO.
DATA: END OF USR_TABL.

* Exclude Limit login by Users
IF  SY-UNAME <> 'XXX'
AND SY-UNAME <> 'XXX'.

CALL 'ThUsrInfo' ID 'OPCODE' FIELD OPCODE
  ID 'TAB' FIELD USR_TABL-*SYS*.

LOOP AT USR_TABL.
  IF SY-UNAME = USR_TABL-BNAME AND SY-MANDT = USR_TABL-MANDT.
   I = I + 1.
  ENDIF.

ENDLOOP.

IF I >= N.

A = 'You have already '.
A+17(2) = I - 1.
A+19(25) = 'login sessions in client '.
A+44(4) = SY-MANDT.

  CALL FUNCTION 'POPUP_TO_INFORM'
     EXPORTING
        TITEL = 'UNSUCCESSFUL LOGIN'
        TXT1  = A
        TXT2  = 'You are not allowed to log in'.

  MOVE: 'SAPMSSY0' TO BDC_TAB1-PROGRAM,
       '120' TO BDC_TAB1-DYNPRO,
       'X' TO BDC_TAB1-DYNBEGIN.
  APPEND BDC_TAB1.CLEAR BDC_TAB1.
  MOVE: 'BDC_OKCODE' TO BDC_TAB1-FNAM,
      '/nex' TO BDC_TAB1-FVAL.
  APPEND BDC_TAB1.CLEAR BDC_TAB1.

  CALL TRANSACTION 'SM04' USING BDC_TAB1 MODE 'N'.
```

ENDIF.

ENDIF.

转载一个 SAP 下载工具的代码，仅用于学习 上

```
program zdown.
*=====================================================
* Direct Download Enterprise version 1.3.1.
*
* THIS SOFTWARE IS FOR PERSONAL USE ONLY.
* THIS PROGRAM IS FREEWARE AND IS PROVIDED ON AN AS-IS BASIS WITHOUT WARRANTY OF ANY
 KIND.
* THE PROVIDER SPECIFICALLY DISCLAIMS ANY OTHER WARRANTY, EXPRESS OR IMPLIED, INCLUDIN
G ANY WARRANTY OF MERCHANTABILITY
* OR FITNESS FOR A PARTICULAR PURPOSE.
*
* IN NO EVENT SHALL THE PROVIDER BE LIABLE FOR ANY CONSEQUENTIAL, INDIRECT, SPECIAL OR
INCIDENTAL DAMAGES, EVEN IF PROVIDER
* HAS BEEN ADVISED BY CLIENT OF THE POSSIBILITY OF SUCH POTENTIAL LOSS OR DAMAGE.
* CLIENT AGREES TO HOLD PROVIDER HARMLESS FROM AND AGAINST ANY AND ALL CLAIMS, LOSS
ES, LIABILITIES AND EXPENSES. BY
* INSTALLING OR RUNNING THIS PROGRAM YOU ARE AGREEING TO THE TERMS AND CONDITONS ST
ATED ABOVE.
*
*-----------------------------------------------------------------------------------------------------------------
* PROGRAM DESCRIPTION & USE
* Allows a user to download programs, Functions, DD definitions, etc to the presentation server. This version
searches
* recursively for nested includes and function modules, and allows you to download the resulting code as stan
dard text
* or HTML web pages within a suitable directory structure.
*
* You can either search by object name, using wildcards if you wish, or a combination of Author and object n
ame. If
* you want all objects returned for a particular author then select the author name and choose the most suitab
le
* radiobutton. All objects will be returned if the fields to the right hand side of the radiobutton are left complet
```

```
ely
*   blank.
*
*   Compatible with R/3 Enterprise only, for older versions of SAP you will need Direct Download version 5.xx.
*   This version removes the programming limitations imposed by developing across SAP releases 3 to 4.6.
*
*   In order to be able to download files to the SAP server you must first set up a logical filepath within transac
tion
*   'FILE', or use an existing one.  You must also create a external operating system command in SM69 called
ZMKDIR.  This
*   will then be used to create any directories needed on the SAP server

*   This program is intended to allow a person to keep a visual representation of a program for backup purpose
s only as
*   has not been designed to allow programs to be uploaded to SAP systems.
*---------------------------------------------------------------------------------------------------------------
*
*  author          :  E.G.Mellodew
*
*
*  program  contact  :  direct@dalestech.com
*                       www.dalestech.com
*
*---------------------------------------------------------------------------------------------------------------

*---------------------------------------------------------------------------------------------------------------
*   SAP  Tables
*---------------------------------------------------------------------------------------------------------------
tables:  trdir,  seoclass,  tfdir,  enlfdir,  dd02l.

*---------------------------------------------------------------------------------------------------------------
*   Types
*---------------------------------------------------------------------------------------------------------------
*  text  element  structure
types:  tTextTable  like  textpool.
*  GUI  titles
types:  tGUITitle  like  d347t.
```

```
* Message classes
types: begin of tMessage,
        arbgb like t100-arbgb,
        stext like t100a-stext,
        msgnr like t100-msgnr,
        text  like t100-text,
      end of tMessage.

* Screen flow.
types: begin of tScreenFlow,
        screen like d020s-dnum,
        code like d022s-line,
      end of tScreenFlow.

* Holds a table\structure definition
types: begin of tDictTableStructure,
        fieldname like dd03l-fieldname,
        position  like dd03l-position,
        keyflag   like dd03l-keyflag,
        rollname  like dd03l-rollname,
        domname   like dd03l-domname,
        datatype  like dd03l-datatype,
        leng      like dd03l-leng,
        ddtext    like dd04t-ddtext,
      end of tdictTableStructure.

* Holds a tables attributes + its definition
types: begin of tDictTable,
        tablename    like dd03l-tabname,
        tableTitle   like dd02t-ddtext,
        iStructure type tDictTableStructure occurs 0,
      end of tDictTable.

* Include program names
types: begin of tInclude,
        includeName like trdir-name,
        includeTitle like tftit-stext,
      end of tInclude.
```

```
* Exception class texts
types: begin of tConcept,
        constName type string,
        concept type sotr_conc,
      end of tConcept.

* Method
types: begin of tMethod,
        cmpName like vseomethod-cmpname,
        descript like vseomethod-descript,
        exposure like vseomethod-exposure,
        methodKey type string,
      end of tMethod.

* Class
types: begin of tClass,
        scanned(1),
        clsname like vseoclass-clsname,
        descript like vseoclass-descript,
        msg_id like vseoclass-msg_id,
        exposure like vseoclass-exposure,
        state like vseoclass-state,
        clsfinal like vseoclass-clsfinal,
        r3release like vseoclass-r3release,
        iMethods type tMethod occurs 0,
        iDictStruct type tDictTable occurs 0,
        iTextElements type tTextTable occurs 0,
        iMessages type tMessage occurs 0,
        iConcepts type tConcept occurs 0,
        textElementKey type string,
        publicClassKey type string,
        privateClassKey type string,
        protectedClassKey type string,
        typesClassKey type string,
        exceptionClass type i,
      end of tClass.
```

```
* function modules
types: begin of tFunction,
        functionName like tfdir-funcName,
        functionGroup like enlfdir-area,
        includeNumber like tfdir-include,
        functionMainInclude like tfdir-funcName,
        functionTitle like tftit-stext,
        topIncludeName like tfdir-funcName,
        progname like tfdir-pname,
        programLinkName like tfdir-pname,
        messageClass like t100-arbgb,
        iTextElements type tTextTable occurs 0,
        iSelectiontexts type tTextTable occurs 0,
        iMessages type tMessage occurs 0,
        iIncludes type tInclude occurs 0,
        iDictStruct type tDictTable occurs 0,
        iGUITitle type tGUITitle occurs 0,
        iScreenFlow type tScreenFlow occurs 0,
      end of tFunction.

types: begin of tProgram,
        progname like trdir-name,
        programTitle like tftit-stext,
        subc like trdir-subc,
        messageClass like t100-arbgb,
        iMessages type tMessage occurs 0,
        iTextElements type tTextTable occurs 0,
        iSelectiontexts type tTextTable occurs 0,
        iGUITitle type tGUITitle occurs 0,
        iScreenFlow type tScreenFlow occurs 0,
        iIncludes type tInclude occurs 0,
        iDictStruct type tDictTable occurs 0,
      end of tProgram.

*---------------------------------------------------------------------------------------------------------------
*  Internal tables
*---------------------------------------------------------------------------------------------------------------
```

```
*  Dictionary object
data: iDictionary type standard table of tDictTable with header line.
* Function modules.
data: iFunctions type standard table of tFunction with header line.
* Tree display structure.
data: iTreeDisplay type standard table of snodetext with header line.
* Message class data
data: iMessages type standard table of tMessage with header line.
* Holds a single message class an all of its messages
data: iSingleMessageClass type standard table of tMessage with header line.
* Holds program related data
data: iPrograms type standard table of tProgram with header line.
* Classes
data: iClasses type standard table of tClass with header line.
* Table of paths created on the SAP server
data: iServerPaths type standard table of string with header line.

*-----------------------------------------------------------------------------------------------------------
*  Table prototypes
*-----------------------------------------------------------------------------------------------------------
data: dumiDictStructure type standard table of tDictTableStructure.
data: dumiTextTab type standard table of tTextTable.
data: dumiIncludes type standard table of tInclude.
data: dumiHtml type standard table of string.
data: dumiHeader type standard table of string .
data: dumiScreen type standard table of tScreenFlow .
data: dumIGUITitle type standard table of tGUITitle.
data: dumiMethods type standard table of tMethod.
data: dumiConcepts type standard table of tConcept.

*-----------------------------------------------------------------------------------------------------------
*   Global objects
*-----------------------------------------------------------------------------------------------------------
data: objFile type ref to cl_gui_frontend_services.
data: objRuntimeError type ref to cx_root.

*-----------------------------------------------------------------------------------------------------------
*  Constants
```

```
*-----------------------------------------------------------------------------------------------------------------------
constants: VERSIONNO type string value '1.3.1'.
constants: TABLES type string value 'TABLES'.
constants: TABLE type string value 'TABLE'.
constants: LIKE type string value 'LIKE'.
constants: TYPE type string value 'TYPE'.
constants: TYPEREFTO type string value 'TYPE REF TO'.
constants: STRUCTURE type string value 'STRUCTURE'.
constants: LOWSTRUCTURE type string value 'structure'.
constants: OCCURS type string value 'OCCURS'.
constants: FUNCTION type string value 'FUNCTION'.
constants: CALLFUNCTION type string value ' CALL FUNCTION'.
constants: MESSAGE type string  value 'MESSAGE'.
constants: INCLUDE type string value 'INCLUDE'.
constants: LOWINCLUDE type string value 'include'.
constants: DESTINATION type string value 'DESTINATION'.
constants: IS_TABLE type string value 'T'.
constants: IS_PROGRAM type string value 'P'.
constants: IS_SCREEN type string value 'S'.
constants: IS_GUITITLE type string value 'G'.
constants: IS_DOCUMENTATION type string value 'D'.
constants: IS_MESSAGECLASS type string value 'MC'.
constants: IS_FUNCTION type string value 'F'.
constants: IS_CLASS type string value 'C'.
constants: IS_METHOD type string value 'M'.
constants: ASTERIX type string value '*'.
constants: COMMA type string value ','.
constants: PERIOD type string value '.'.
constants: DASH type string value '-'.
constants: TRUE type i value 1.
constants: FALSE type i value 0.
constants: LT type string value '&lt;'.
constants: GT type string value '&gt;'.
constants: UNIX type string value 'UNIX'.
constants: NON_UNIX type string value 'not UNIX'.
constants: BACKGROUND_COLOUR type string value '#FFFFE0'.
```

```
constants: COLOUR_WHITE type string value '#FFFFFF'.
constants: COLOUR_BLACK type string value '#000000'.
constants: COLOUR_YELLOW type string value '#FFFF00'.
constants: COMMENT_COLOUR type string value '#0000FF'.
constants: HTMLEXTENSION type string value 'html'.
constants: TEXTEXTENSION type string value 'txt'.

*---------------------------------------------------------------------------------------------------------------
*  Global variables
*---------------------------------------------------------------------------------------------------------------
data: statusBarMessage(100).
data: forcedExit type i value 0.
data: startTime like sy-uzeit.
data: runTime like sy-uzeit.
data: downloadFileExtension type string.
data: downloadFolder type string.
data: serverSlashSeparator type string.
data: frontendSlashSeparator type string.
data: slashSeparatorToUse type string.
data: serverFilesystem type filesys_d.
data: serverFolder type string.
data: frontendOpSystem type string.
data: serverOpSystem type string.
data: customerNameSpace type string.
ranges: soProgramName for trdir-name.
ranges: soAuthor for usr02-bname.
ranges: soTableNames for dd02l-tabname.
ranges: soFunctionName  for tfdir-funcName.
ranges: soClassName for vseoclass-clsname.
ranges: soFunctionGroup for enlfdir-area.
field-symbols: <waDictStruct> type tDictTable.

*---------------------------------------------------------------------------------------------------------------
*  Selection screen declaration
*---------------------------------------------------------------------------------------------------------------
* Author
selection-screen: begin of block b1 with frame title tBlock1.
```

```
  selection-screen begin of line.
    selection-screen comment 5(23) tAuth.
    parameters: pAuth like usr02-bname memory id MAUTH.
  selection-screen end of line.

  selection-screen begin of line.
    selection-screen comment 5(36) tPmod.
    parameters: pMod as checkbox.
  selection-screen end of line.

* Local objects
  selection-screen begin of line.
    selection-screen comment 5(36) t$tmp.
    parameters: p$tmp as checkbox default ''.
  selection-screen end of line.
selection-screen: end of block b1.

selection-screen begin of block b2 with frame title tBlock2.
* Tables
  selection-screen begin of line.
    parameters: rTable radiobutton group r1.
    selection-screen comment 5(15) tRtable.
  selection-screen end of line.

  selection-screen begin of line.
    selection-screen comment 10(15) tPtable.
    select-options: soTable for dd02l-tabname.
  selection-screen end of line.

  selection-screen begin of line.
    selection-screen comment 10(79) tTnote.
  selection-screen end of line.

* Message classes
  selection-screen begin of line.
    parameters: rMess radiobutton group r1.
    selection-screen comment 5(18) tPMes.
  selection-screen end of line.
```

```
  selection-screen begin of line.
    selection-screen comment 10(18) tMname.
    parameters: pMname like t100-arbgb memory id MMNAME.
  selection-screen end of line.

* Function modules
  selection-screen begin of line.
    parameters: rFunc radiobutton group r1.
    selection-screen comment 5(30) tRfunc.
  selection-screen end of line.

  selection-screen begin of line.
    selection-screen comment 10(15) tPfname.
    select-options: soFname for tfdir-funcName.
  selection-screen end of line.

  selection-screen begin of line.
    selection-screen comment 10(15) tFgroup.
    select-options: soFgroup for enlfdir-area.
  selection-screen end of line.

* Classes
  selection-screen begin of line.
    parameters: rClass radiobutton group r1.
    selection-screen comment 5(30) tRClass.
  selection-screen end of line.

  selection-screen begin of line.
    selection-screen comment 10(15) tPcName.
    select-options: soClass for seoclass-clsname.
  selection-screen end of line.

* Programs / includes
  selection-screen begin of line.
    parameters: rProg radiobutton group r1 default 'X'.
    selection-screen comment 5(18) tProg.
  selection-screen end of line.
```

```
  selection-screen begin of line.
    selection-screen comment 10(15) tRpname.
    select-options: soProg for trdir-name.
  selection-screen end of line.

  selection-screen skip.
* Language
  selection-screen begin of line.
    selection-screen comment 1(18) tMLang.
    parameters: pMLang like t100-sprsl default 'EN'.
  selection-screen end of line.

* Package
  selection-screen begin of line.
    selection-screen comment 1(18) tPack.
    parameters: pPack like tadiv-devclass memory id MPACK.
  selection-screen end of line.

* Customer objects
  selection-screen begin of line.
    selection-screen comment 1(27) tCust.
    parameters: pCust as checkbox default 'X'.
    selection-screen comment 32(25) tNRange.
    parameters: pCName type namespace memory id MNAMESPACE.
  selection-screen end of line.
selection-screen: end of block b2.

* Additional things to download.
selection-screen: begin of block b3 with frame title tBlock3.
  selection-screen begin of line.
    selection-screen comment 1(33) tPtext.
    parameters: pText as checkbox default 'X' memory id MTEXT.
  selection-screen end of line.

  selection-screen begin of line.
    selection-screen comment 1(33) tMess.
    parameters: pMess as checkbox default 'X' memory id MMESS.
  selection-screen end of line.
```

```
  selection-screen begin of line.
    selection-screen comment 1(33) tPinc.
    parameters: pInc as checkbox default 'X' memory id MINC.
    selection-screen comment 40(20) tRecc.
    parameters: pReci as checkbox default 'X' memory id MRECI.
  selection-screen end of line.

  selection-screen begin of line.
    selection-screen comment 1(33) tPfunc.
    parameters: pFunc as checkbox default 'X' memory id MFUNC.
    selection-screen comment 40(20) tRecf.
    parameters: pRecf as checkbox default 'X' memory id MRECF.
  selection-screen end of line.

  selection-screen begin of line.
    selection-screen comment 1(33) tDoc.
    parameters: pDoc as checkbox default 'X' memory id MDOC.
  selection-screen end of line.

  selection-screen begin of line.
    selection-screen comment 1(33) tPscr.
    parameters: pScr as checkbox default 'X' memory id MSCR.
  selection-screen end of line.

  selection-screen begin of line.
    selection-screen comment 1(33) tPdict.
    parameters: pDict as checkbox default 'X' memory id MDICT.
  selection-screen end of line.

  selection-screen begin of line.
    selection-screen comment 1(33) tSortT.
    parameters: pSortT as checkbox default ' ' memory id MSORTT.
  selection-screen end of line.
selection-screen: end of block b3.

* File details
selection-screen: begin of block b4 with frame title tBlock4.
  selection-screen begin of line.
    selection-screen comment 1(20) tPhtml.
```

```
  parameters: pHtml radiobutton group g1 default 'X'.
 selection-screen end of line.

 selection-screen begin of line.
  selection-screen comment 5(29) tComm.
  parameters: pComm as checkbox default 'X'.
 selection-screen end of line.

 selection-screen begin of line.
  selection-screen comment 5(29) tBack.
  parameters: pBack as checkbox default 'X'.
 selection-screen end of line.

 selection-screen begin of line.
  selection-screen comment 1(20) tPtxt.
  parameters: pTxt radiobutton group g1.
 selection-screen end of line.

 selection-screen skip.

* Download to SAP server
 selection-screen begin of line.
  selection-screen comment 1(25) tServ.
  parameters: pServ radiobutton group g2.
 selection-screen end of line.
 selection-screen begin of line.
  selection-screen comment 8(20) tSPath.
  parameters: pLogical like filename-fileintern memory id MLOGICAL.
 selection-screen end of line.
 selection-screen comment /28(60) tSDPath.

* Download to PC
 selection-screen begin of line.
  selection-screen comment 1(25) tPc.
  parameters: pPc radiobutton group g2 default 'X'.
 selection-screen end of line.
 selection-screen begin of line.
  selection-screen comment 8(20) tPpath.
  parameters: pFolder like rlgrap-filename memory id MFOLDER.
```

```
  selection-screen end of line.
selection-screen: end of block b4.

* Display options
selection-screen: begin of block b5 with frame title tBlock5.
* Display final report
  selection-screen begin of line.
    selection-screen comment 1(33) tRep.
    parameters: pRep as checkbox default 'X'.
  selection-screen end of line.
* Display progress messages
  selection-screen begin of line.
    selection-screen comment 1(33) tProMess.
    parameters: pProMess as checkbox default 'X'.
  selection-screen end of line.
selection-screen: end of block b5.

*---------------------------------------------------------------------------------------------------------------
* Display a directory picker window
*---------------------------------------------------------------------------------------------------------------
at selection-screen on value-request for pFolder.

data: objFile type ref to cl_gui_frontend_services.
data: pickedFolder type string.
data: initialFolder type string.

  if sy-batch is initial.
    create object objFile.

    if not pFolder is initial.
      initialFolder = pFolder.
    else.
      objFile->get_temp_directory( changing temp_dir = initialFolder
                                   exceptions cntl_error = 1
                                              error_no_gui = 2
                                              not_supported_by_gui = 3 ).
    endif.
```

```
    objFile->directory_browse( exporting initial_folder = initialFolder
                               changing selected_folder = pickedFolder
                               exceptions cntl_error = 1
                                          error_no_gui = 2
                                          not_supported_by_gui = 3 ).

    if sy-subrc = 0.
      pFolder = pickedFolder.
    else.
      write: / 'An error has occured picking a folder'.
    endif.
  endif.

*-----------------------------------------------------------------------------------------------------------------
at selection-screen.
*-----------------------------------------------------------------------------------------------------------------
  case 'X'.
    when pPc.
      if pFolder is initial.
*       User must enter a path to save to
        message e000(oo) with 'You must enter a file path'.
      endif.

    when pServ.
      if pLogical is initial.
*       User must enter a logical path to save to
        message e000(oo) with 'You must enter a logical file name'.
      endif.
  endcase.

*-----------------------------------------------------------------------------------------------------------------
at selection-screen on pLogical.
*-----------------------------------------------------------------------------------------------------------------

  if not pServ is initial.
    call function 'FILE_GET_NAME' exporting logical_filename = pLogical
                                  importing file_name = serverFolder
                                  exceptions file_not_found = 1
```

```
                                  others = 2.
    if sy-subrc = 0.
      if serverFolder is initial.
        message e000(oo) with 'No file path returned from logical filename'.
      else.
*       Path to display on the selection screen
        tSDPath = serverFolder.
*       Remove the trailing slash off the path as the subroutine buildFilename will add an extra one
        shift serverFolder right deleting trailing serverSlashSeparator.
        shift serverFolder left deleting leading space.
      endif.
    else.
      message e000(oo) with 'Logical filename does not exist'.
    endif.
  endif.

* ---------------------------------------------------------------------------------------------------------
at selection-screen on value-request for soProg-low.
* ---------------------------------------------------------------------------------------------------------
  call function 'REPOSITORY_INFO_SYSTEM_F4' exporting object_type  = 'PROG'
                                        object_name  = soProg-low
                                        suppress_selection   = 'X'
                                        use_alv_grid = ''
                                        without_personal_list = ''
                                  importing object_name_selected = soProg-low
                                  exceptions cancel = 1.

* ---------------------------------------------------------------------------------------------------------
at selection-screen on value-request for soProg-high.
* ---------------------------------------------------------------------------------------------------------
  call function 'REPOSITORY_INFO_SYSTEM_F4' exporting object_type  = 'PROG'
                                        object_name  = soProg-high
                                        suppress_selection   = 'X'
                                        use_alv_grid = ''
                                        without_personal_list = ''
                                  importing object_name_selected = soProg-high
                                  exceptions cancel = 1.
```

```
* ---------------------------------------------------------------------------------------------------------------
at selection-screen on value-request for soClass-low.
* ---------------------------------------------------------------------------------------------------------------
  call function 'F4_DD_ALLTYPES' exporting object = soClass-low
                                suppress_selection = 'X'
                                display_only = ''
                                only_types_for_clifs = 'X'
                          importing result = soClass-low.

* ---------------------------------------------------------------------------------------------------------------
at selection-screen on value-request for soClass-high.
* ---------------------------------------------------------------------------------------------------------------
  call function 'F4_DD_ALLTYPES' exporting object = soClass-high
                                suppress_selection = 'X'
                                display_only = ''
                                only_types_for_clifs = 'X'
                          importing result = soClass-high.

* ---------------------------------------------------------------------------------------------------------------
at selection-screen on value-request for soFName-low.
* ---------------------------------------------------------------------------------------------------------------
  call function 'REPOSITORY_INFO_SYSTEM_F4' exporting object_type  = 'FUNC'
                                      object_name  = soFname-low
                                      suppress_selection   = 'X'
                                      use_alv_grid = ''
                                      without_personal_list = ''
                                 importing object_name_selected = soFName-low
                                 exceptions cancel = 1.

* ---------------------------------------------------------------------------------------------------------------
at selection-screen on value-request for soFName-high.
* ---------------------------------------------------------------------------------------------------------------
  call function 'REPOSITORY_INFO_SYSTEM_F4' exporting object_type  = 'FUNC'
                                      object_name  = soFname-high
                                      suppress_selection   = 'X'
                                      use_alv_grid = ''
                                      without_personal_list = ''
```

```
                                  importing object_name_selected = soFName-high
                                  exceptions cancel = 1.

* ---------------------------------------------------------------------------------------------------------
at selection-screen on value-request for soFGroup-low.
* ---------------------------------------------------------------------------------------------------------
  call function 'REPOSITORY_INFO_SYSTEM_F4' exporting object_type  = 'FUGR'
                                        object_name  = soFGroup-low
                                        suppress_selection   = 'X'
                                        use_alv_grid = ''
                                        without_personal_list = ''
                                  importing object_name_selected = soFGroup-low
                                  exceptions cancel = 1.

* ---------------------------------------------------------------------------------------------------------
at selection-screen on value-request for soFGroup-high.
* ---------------------------------------------------------------------------------------------------------
  call function 'REPOSITORY_INFO_SYSTEM_F4' exporting object_type  = 'FUGR'
                                        object_name  = soFGroup-high
                                        suppress_selection   = 'X'
                                        use_alv_grid = ''
                                        without_personal_list = ''
                                  importing object_name_selected = soFGroup-high
                                  exceptions cancel = 1.

*---------------------------------------------------------------------------------------------------------
* initialisation
*---------------------------------------------------------------------------------------------------------
initialization.
* Parameter screen texts.
  tBlock1 = 'Author (Optional)'.
  t$tmp   = 'Programs only: include local objects'.
  tBlock2 = 'Objects to download'.
  tBlock3 = 'Additional downloads for programs, function modules and classes'.
  tBlock4 = 'Download parameters'.
  tBlock5 = 'Display options'.
  tAuth   = 'Author name'.
  tPmod   = 'Include programs modified by author'.
```

```
tCust   = 'Only customer objects'.
tNRange = 'Alt customer name range'.
tRtable = 'Tables / Structures'.
tPtable = 'Table name'.
tTnote  = 'Note: tables are stored under the username of the last person who modified them'.
tRfunc  = 'Function modules'.
tPfname = 'Function name'.
tFgroup = 'Function group'.
tRClass  = 'Classes'.
tPcname = 'Class name'.
tMess   = 'Message class'.
tMName  = 'Class name'.
tMLang  = 'Language'.
tProg   = 'Programs'.
tRpname = 'Program name'.
tPack   = 'Package'.
tPtxt   = 'Text document'.
tPhtml  = 'HTML document'.
tComm   = 'Highlight comments'.
tBack   = 'Include background colour'.
tPtext  = 'Text elements'.
tPinc   = 'Include programs'.
tRecc   = 'Recursive search'.
tPpath  = 'File path'.
tSPath  = 'Logical file name'.
tPmes   = 'Message classes'.
tPfunc  = 'Function modules'.
tDoc    = 'Function module documentation'.
tRecf   = 'Recursive search'.
tPscr   = 'Screens'.
tPdict  = 'Dictionary structures'.
tSortT  = 'Sort table fields alphabetically'.
tServ   = 'Download to server'.
tPc     = 'Download to PC'.
tRep    = 'Display download report'.
tProMess  = 'Display progress messages'.
```

```
* Determine the frontend operating system type.
  if sy-batch is initial.
    perform determineFrontendOPSystem using frontendSlashSeparator frontendOpSystem.
  endif.
  perform determineServerOpsystem using serverSlashSeparator serverFileSystem serverOpsystem.

* Determine if the external command exists.  If it doesn't then disable the server input field
  perform findExternalCommand.

*---------------------------------------------------------------------------------------------------------------
* start-of-selection.
*---------------------------------------------------------------------------------------------------------------
start-of-selection.

  perform checkComboBoxes.
  perform fillSelectionRanges.
  startTime = sy-uzeit.

* Don't display status messages if we are running in the background
  if not sy-batch is initial.
    pProMess = ''.
  endif.

* Fool the HTML routines to stop them hyperlinking anything with a space in them
  if pCName is initial.
    customerNameSpace  = '^'.
  else.
    customerNameSpace = pCName.
  endif.

* Determine which operating slash and download directory to use
  case 'X'.
    when pPc.
      slashSeparatorToUse = frontendSlashSeparator.
      downloadFolder = pFolder.
    when pServ.
      slashSeparatorToUse = serverSlashSeparator.
      downloadFolder = serverFolder.
  endcase.
```

```
* Main program flow.
  case 'X'.
*   Select tables
    when rTable.
      perform retrieveTables using iDictionary[]
                                   soTableNames[]
                                   soAuthor[].

*   Select message classes tables
    when rMess.
      perform retrieveMessageClass using iMessages[]
                                         soAuthor[]      "Author
                                         pMname          "Message class name
                                         pMLang          "Message class language
                                         pMod.           "Modified by author

*   Select function modules
    when rFunc.
      perform retrieveFunctions using soFunctionName[]   "Function name
                                      soFunctionGroup[]  "Function group
                                      iFunctions[]       "Found functions
                                      soAuthor[]         "Author
                                      pText              "Get text elements
                                      pScr               "Get screens
                                      pCust              "Customer data only
                                      customerNameSpace. "Customer name range


      loop at iFunctions.
*       Find Dict structures, messages, functions, includes etc.
        perform scanForAdditionalFuncStuff using iFunctions[]
                                                 pRecI          "Search for includes recursively
                                                 pRecF          "Search for functions recursively
                                                 pInc           "Search for includes
                                                 pFunc          "Search for functions
                                                 pDict          "search for dictionary objects
                                                 pMess          "Search for messages
                                                 pCust          "Customer data only
```

```
                                    customerNameSpace.      "Customer name range
    endloop.

*   Select Classes
   when rClass.
    perform retrieveClasses using iClasses[]
                                  iFunctions[]
                                  soClassName[]       "Class name
                                  soAuthor[]          "Author
                                  customerNameSpace   "Customer name range
                                  pMod                "Also modified by author
                                  pCust               "Customer object only
                                  pMess               "Find messages
                                  pText               "Text Elements
                                  pDict               "Dictionary structures
                                  pFunc               "Get functions
                                  pInc                "Get includes
                                  pRecF               "Search recursively for functions
                                  pRecI               "Search recursively for includes
                                  'X'                 "Search recursively for classes
                                  pMLang.             "Language

    loop at iFunctions.
*     Find Dict structures, messages, functions, includes etc.
     perform scanForAdditionalFuncStuff using iFunctions[]
                                    pRecI               "Search for includes recursively
                                    pRecF               "Search for functions recursively
                                    pInc                "Search for includes
                                    pFunc               "Search for functions
                                    pDict               "search for dictionary objects
                                    pMess               "Search for messages
                                    pCust               "Customer data only
                                    customerNameSpace.      "Customer name range
    endloop.

*   Select programs
   when rProg.
    perform retrievePrograms using iPrograms[]
```

```
                    iFunctions[]
                    soProgramName[]    "Program name
                    soAuthor[]         "Author
                    customerNamespace  "Customer name range
                    pMod               "Also modified by author
                    pCust              "Customer object only
                    pMess              "Find messages
                    pText              "Text Elements
                    pDict              "Dictionay structures
                    pFunc              "Get functions
                    pInc               "Get includes
                    pScr               "Get screens
                    pRecF              "Search recursively for functions
                    pRecI              "Search recursively for includes
                    p$Tmp              "local objects
                    pPack.             "Package
  endcase.

*-----------------------------------------------------------------------------------------------------------
* end-of-selection
*-----------------------------------------------------------------------------------------------------------
end-of-selection.

  if forcedExit = 0.
*   Set the file extension and output type of the file
    if pTxt is initial.
      downloadFileExtension = HTMLEXTENSION.
    else.
      downloadFileExtension = TEXTEXTENSION.
    endif.

*   Decide what to download
    case 'X'.
*     Download tables
      when rTable.
        if not ( iDictionary[] is initial ).
          perform downloadDDStructures using iDictionary[]
                                             downloadFolder
```

```
                                HTMLEXtension
                                space
                                pSortT
                                slashSeparatorToUse
                                pServ
                                pProMess.

*       Free up any memory used for caching HTML versions of tables
        loop at iDictionary.
          free memory id iDictionary-tablename.
        endloop.

*       Display donwload report
        if not pRep is initial.
          get time.
          runTime = sy-uzeit - startTime.
          perform fillTreeNodeTables using iDictionary[]
                                iTreeDisplay[]
                                runTime.
        endif.

        clear iDictionary[].
      endif.

*     Download message class
      when rMess.
        if not ( iMessages[] is initial ).
          sort iMessages ascending by arbgb msgnr.
          loop at iMessages.
            append iMessages to iSingleMessageClass.
            at end of arbgb.
              perform downloadMessageClass using iSingleMessageClass[]
                                  iMessages-arbgb
                                  downloadFolder
                                  downloadFileExtension
                                  pHtml
                                  space
                                  pComm
```

```
                                    customerNameSpace
                                    pInc
                                    pDict
                                    pMess
                                    slashSeparatorToUse
                                    pServ
                                    pProMess.
          clear iSingleMessageClass[].
        endat.
      endloop.

*       Display download report
      if not pRep is initial.
        get time.
        runTime = sy-uzeit - startTime.
        perform fillTreeNodeMessages using iMessages[]
                                           iTreeDisplay[]
                                           runTime.
      endif.

      clear iMessages[].
    endif.

*   Download functions
    when rFunc.
      if not ( iFunctions[] is initial ).
        perform downloadFunctions using iFunctions[]
                                        downloadFolder
                                        downloadFileExtension
                                        space
                                        pDoc
                                        pHtml
                                        pComm
                                        customerNameSpace
                                        pInc
                                        pDict
                                        TEXTEXTENSION
                                        HTMLEXTENSION
```

```
                                pSortT
                                slashSeparatorToUse
                                pServ
                                pProMess.

*       Free up any memory used for caching HTML versions of tables
        loop at iFunctions.
          loop at iFunctions-iDictStruct assigning <waDictStruct>.
            free memory id <waDictStruct>-tablename.
          endloop.
        endloop.

*       Display donwload report
        if not pRep is initial.
          get time.
          runTime = sy-uzeit - startTime.
          perform fillTreeNodeFunctions using iFunctions[]
                                              iTreeDisplay[]
                                              runTime.
        endif.

        clear iFunctions[].
      endif.

*     Download Classes
      when rClass.
        if not ( iClasses[] is initial ).
          perform downloadClasses using iClasses[]
                                        iFunctions[]
                                        downloadFolder
                                        downloadFileExtension
                                        HTMLEXTENSION
                                        TEXTEXTENSION
                                        pHtml
                                        pComm
                                        customerNameSpace
                                        pInc
                                        pDict
```

```
                    pDoc
                    pSortT
                    slashSeparatorToUse
                    pServ
                    pProMess.

*     Free up any memory used for caching HTML versions of tables
      loop at iFunctions.
        loop at iFunctions-iDictStruct assigning <waDictStruct>.
          free memory id <waDictStruct>-tablename.
        endloop.
      endloop.

*     Free up any memory used for caching HTML versions of tables
      loop at iPrograms.
        loop at iPrograms-iDictStruct assigning <waDictStruct>.
          free memory id <waDictStruct>-tablename.
        endloop.
      endloop.

*     Display donwload report
      if not pRep is initial.
        get time.
        runTime = sy-uzeit - startTime.
        perform fillTreeNodeClasses using iClasses[]
                                          iFunctions[]
                                          iTreeDisplay[]
                                          runTime.
      endif.

      clear iClasses[].
      clear iFunctions[].
    endif.

*   Download programs
    when rProg.
      if not ( iPrograms[] is initial ).
        perform downloadPrograms using iPrograms[]
```

```
                    iFunctions[]
                    downloadFolder
                    downloadFileExtension
                    HTMLEXTENSION
                    TEXTEXTENSION
                    pHtml
                    pComm
                    customerNameSpace
                    pInc
                    pDict
                    pDoc
                    pSortT
                    slashSeparatorToUse
                    pServ
                    pProMess.

*     Free up any memory used for caching HTML versions of tables
      loop at iFunctions.
        loop at iFunctions-iDictStruct assigning <waDictStruct>.
          free memory id <waDictStruct>-tablename.
        endloop.
      endloop.

*     Free up any memory used for caching HTML versions of tables
      loop at iPrograms.
        loop at iPrograms-iDictStruct assigning <waDictStruct>.
          free memory id <waDictStruct>-tablename.
        endloop.
      endloop.

*     Display donwload report
      if not pRep is initial.
        get time.
        runTime = sy-uzeit - startTime.
        perform fillTreeNodePrograms using iPrograms[]
                                           iFunctions[]
                                           iTreeDisplay[]
```

```
                                runTime.
        endif.

        clear iPrograms[].
        clear iFunctions[].
      endif.
    endcase.

    if not pRep is initial.
      if not ( iTreeDisplay[] is initial ).
        perform displayTree using iTreeDisplay[].
      else.
        statusBarMessage = 'No items found matching selection criteria'.
        perform displayStatus using statusBarMessage 2.
      endif.
    endif.
  endif.

*--- Memory IDs
* User name
  set parameter id 'MAUTH' field pAuth.
* Message class
  set parameter id 'MMNAME' field pMname.
* Customer namespace
  set parameter id 'MNAMESPACE' field pCName.
* Folder
  set parameter id 'MFOLDER' field pFolder.
* Logical filepath
  set parameter id 'MLOGICAL' field pLogical.
* Package
  set parameter id 'MPACK' field pPack.
* Text element checkbox
  set parameter id 'MTEXT' field pText.
* Messages checkbox
  set parameter id 'MMESS' field pMess.
* Includes checkbox
  set parameter id 'MINC' field pInc.
* Recursive includes checkbox.
```

```
  set parameter id 'MRECI' field pReci.
* Functions checkbox
  set parameter id 'MFUNC' field pFunc.
* Recursive functions checkbox
  set parameter id 'MRECF' field pRecf.
* Function module documntation checkbox
  set parameter id 'MDOC' field pDoc.
* Screens checkbox
  set parameter id 'MSCR' field pScr.
* Dictionary checkbox
  set parameter id 'MDICT' field pDict.
* Sort table ascending checkBox
  set parameter id 'MSORTT' field pSortT.

*****************************************************************************************************************
*************************************************SUBROUTINES*************************************************
*****************************************************************************************************************

*---------------------------------------------------------------------------------------------------------------
*  checkComboBoxes...  Check input parameters
*---------------------------------------------------------------------------------------------------------------
form checkComboBoxes.

  if pAuth is initial.
    case 'X'.
      when rTable.
        if soTable[] is initial.
          statusBarMessage = 'You must enter either a table name or author.'.
        endif.
      when rFunc.
        if ( soFName[] is initial ) and ( soFGroup[] is initial ).
          if soFName[] is initial.
            statusBarMessage = 'You must enter either a function name or author.'.
          else.
            if soFGroup[] is initial.
              statusBarMessage = 'You must enter either a function group, or an author name.'.
            endif.
          endif.
```

```
        endif.
      when rProg.
        if soProg[] is initial.
           statusBarMessage = 'You must enter either a program name or author name.'.
        endif.
    endcase.
  else.
*   Check the user name of the person objects are to be downloaded for
    if pAuth = 'SAP*' or pauth = 'SAP'.
      statusBarMessage = 'Sorry cannot download all objects for SAP standard user'.
    endif.
  endif.

  if not statusBarMessage is initial.
    perform displayStatus using statusBarMessage 3.
    forcedExit = 1.
    stop.
  endif.
endform.                                                    "checkComboBoxes

*---------------------------------------------------------------------------------------------------------------
* fillSelectionRanges...      for selection routines
*---------------------------------------------------------------------------------------------------------------
form fillSelectionRanges.

data: strLength type i.

  strLength = strlen( pcName ).

  if not pAuth is initial.
    soAuthor-sign = 'I'.
    soAuthor-option = 'EQ'.
    soAuthor-low = pAuth.
    append soAuthor.
  endif.

* Tables
  if not soTable is initial.
    soTableNames[] = soTable[].
```

```
*   Add in the customer namespace if we need to
    if not pcName is initial.
      loop at soTableNames.
       if soTableNames-low+0(strLength) <> pcName.
         concatenate pcName soTableNames-low into soTableNames-low.
       endif.

       if soTableNames-high+0(strLength) <> pcName.
         concatenate pcName soTableNames-high into soTableNames-high.
       endif.

       modify soTableNames.
      endloop.
    endif.
  endif.

* Function names
  if not soFName is initial.
    soFunctionName[] = soFname[].
*   Add in the customer namespace if we need to
    if not pcName is initial.
      loop at soFunctionName.
       if soFunctionName-low+0(strLength) <> pcName.
         concatenate pcName soFunctionName-low into soFunctionName-low.
       endif.

       if soFunctionName-high+0(strLength) <> pcName.
         concatenate pcName soFunctionName-high into soFunctionName-high.
       endif.

       modify soFunctionName.
      endloop.
    endif.
  endif.

* Function group
  if not soFGroup is initial.
    soFunctionGroup[] = soFGroup[].
*   Add in the customer namespace if we need to
```

```
    if not pcName is initial.
      loop at soFunctionName.
       if soFunctionGroup-low+0(strLength) <> pcName.
         concatenate pcName soFunctionGroup-low into soFunctionGroup-low.
       endif.

       if soFunctionGroup-high+0(strLength) <> pcName.
         concatenate pcName soFunctionGroup-high into soFunctionGroup-high.
       endif.

       modify soFunctionGroup.
      endloop.
    endif.
  endif.

* Class names
  if not soClass is initial.
    soClassName[] = soClass[].
*   Add in the customer namespace if we need to
    if not pcName is initial.
      loop at soClassName.
       if soClassName-low+0(strLength) <> pcName.
         concatenate pcName soClassName-low into soClassName-low.
       endif.

       if soClassName-high+0(strLength) <> pcName.
         concatenate pcName soClassName-high into soClassName-high.
       endif.

       modify soClassName.
      endloop.
    endif.
  endif.

* Program names
  if not soProg is initial.
    soProgramName[] = soProg[].
*   Add in the customer namespace if we need to
    if not pcName is initial.
```

```
      loop at soProgramName.
       if soProgramName-low+0(strLength) <> pcName.
         concatenate pcName soProgramName-low into soProgramName-low.
       endif.

       if soProgramName-high+0(strLength) <> pcName.
         concatenate pcName soProgramName-high into soProgramName-high.
       endif.

       modify soProgramName.
      endloop.
    endif.
  endif.
endform.                                                          " fillSelectionRanges

*---------------------------------------------------------------------------------------------------------------
*  retrieveTables...          Search for tables in dictionary
*---------------------------------------------------------------------------------------------------------------
form retrieveTables using iLocDictStructure like iDictionary[]
                          soTable like soTable[]
                          soAuthor like soAuthor[].

data: waDictStructure type tDictTable.

  select tabname
        from dd02l
        into waDictStructure-tablename
        where tabname in soTable
          and tabclass <> 'CLUSTER'
          and tabclass <> 'POOL'
          and tabclass <> 'VIEW'
          and as4user in soAuthor
          and as4local = 'A'.

    perform findTableDescription using waDictStructure-tablename
                                       waDictStructure-tableTitle.

    perform findTableDefinition using waDictStructure-tableName
                                       waDictStructure-iStructure[].
```

```
    append waDictStructure to iLocDictStructure.
    clear waDictStructure.
  endselect.
endform.                                                    "retrieveTables

*---------------------------------------------------------------------------------------------------------
*  findTableDescription...  Search for table description in dictionary
*---------------------------------------------------------------------------------------------------------
form findTableDescription using value(tableName)
                                tableDescription.

    select single ddtext
                from dd02t
                into tableDescription
                where tabname = tableName
                 and ddlanguage = sy-langu.
endform.                                                    "findTableDescription

*---------------------------------------------------------------------------------------------------------
*  findTableDefinition... Find the structure of a table from the SAP database.
*---------------------------------------------------------------------------------------------------------
form findTableDefinition using value(tablename)
                            iDictStruct like dumIDictStructure[].

data gotstate like dcobjif-gotstate.
data: definition type standard table of DD03P with header line.
data: waDictStruct type tDictTableStructure.

  call function 'DDIF_TABL_GET'
      exporting
          name        = tablename
          state       = 'A'
          langu       = sy-langu
      importing
          gotstate    = gotstate
      tables
          dd03p_tab   = definition
      exceptions
```

```
        illegal_input = 1
        others        = 2.

  if sy-subrc = 0 and gotstate = 'A'.
    loop at definition.
      move-corresponding definition to waDictStruct.
      perform removeLeadingZeros changing waDictStruct-position.
      perform removeLeadingZeros changing waDictStruct-leng.
      append waDictStruct to iDictStruct.
    endloop.
  endif.
endform.                                                    "findTableDefinition

*---------------------------------------------------------------------------------------------------------------
*  retrieveMessageClass...   Retrieve a message class from the SAP database
*---------------------------------------------------------------------------------------------------------------
form retrieveMessageClass using iLocMessages like iMessages[]
                          rangeAuthor like soAuthor[]
                          value(messageClassName)
                          value(messageClassLang)
                          value(modifiedBy).

data: waMessage type tMessage.

  if not messageClassName is initial.
    select * from t100
          appending corresponding fields of table iLocMessages
          where sprsl = messageClassLang
            and arbgb = messageClassName.

    loop at iLocMessages into waMessage.
      select single stext
                from t100a              "#EC CI_BUFFJOIN
                into waMessage-stext
                where arbgb = waMessage-arbgb.
      modify iLocMessages from waMessage index sy-tabix.
    endloop.
  else.
```

```
    if modifiedBy is initial.
*     Select by author
      select t100~arbgb                    "#EC CI_BUFFJOIN
           t100~msgnr
           t100~text
           t100a~stext
           appending corresponding fields of table iLocMessages
           from t100
           inner join t100a on t100a~arbgb = t100~arbgb
           where t100a~masterLang = messageClassLang
             and t100a~respUser in rangeAuthor[].
    else.
*     Select also by the last person who modified the message class
      select t100~arbgb                    "#EC CI_BUFFJOIN
           t100~msgnr
           t100~text
           t100a~stext
           appending corresponding fields of table iLocMessages
           from t100
           inner join t100a on t100a~arbgb = t100~arbgb
           where t100a~masterLang = messageClassLang
             and t100a~respUser in rangeAuthor[]
             and t100a~lastUser in rangeAuthor[].
    endif.
  endif.
endform.                                                    "retrieveMessageClass

*---------------------------------------------------------------------------------------------------------------
*  retrieveFunctions...  Retrieve function modules from SAP DB.  May be called in one of two ways
*---------------------------------------------------------------------------------------------------------------
form retrieveFunctions using soFName like soFunctionName[]
                        soFGroup like soFunctionGroup[]
                        iLocFunctionNames like iFunctions[]
                        value(solocAuthor) like soAuthor[]
                        value(getTextElements)
                        value(getScreens)
```

```
                    value(customerOnly)
                    value(customerNameRange).

ranges: rangeFuncName  for tfdir-funcName.
ranges: rangefuncGroup for enlfdir-area.
data: waFunctionName type tFunction.
data: noGroupsFound type i value TRUE.
data: previousFG type v_fdir-area.

  rangeFuncName[] = soFName[].
  rangeFuncGroup[] = soFGroup[].

  if not soLocAuthor[] is initial.
*-- Need to select all function groups by author
    select area
          from tlibv
          into rangeFuncGroup-low
          where uname in soLocAuthor
            and area in soFGroup[].

      rangeFuncGroup-sign = 'I'.
      rangeFuncGroup-option = 'EQ'.
      append rangeFuncGroup.
      noGroupsFound = FALSE.
    endselect.
  else.
    noGroupsFound = FALSE.
  endif.

  if noGroupsFound = FALSE.
*   select by function name and/or function group.
    select funcName area
                from v_fdir
                into (waFunctionName-functionName,
                      waFunctionName-functionGroup)
                where funcName in rangeFuncName
                  and area in rangeFuncGroup
```

```
                and generated = ''
                order by area.

      append waFunctionName to iLocFunctionNames.
    endselect.
  endif.

  loop at iLocFunctionNames into waFunctionName.
    perform retrieveFunctionDetail using waFunctionName-functionName
                                         waFunctionName-progname
                                         waFunctionName-includeNumber
                                         waFunctionName-functionTitle.

    perform findMainFunctionInclude using waFunctionName-progname
                                          waFunctionName-includeNumber
                                          waFunctionName-functionMainInclude.

    perform findFunctionTopInclude using waFunctionName-progname
                                         waFunctionName-topIncludeName.

*   Find all user defined includes within the function group
    perform scanForFunctionIncludes using waFunctionName-progname
                                          customerOnly
                                          customerNameRange
                                          waFunctionName-iIncludes[].
*   Find main message class
    perform findMainMessageClass using waFunctionName-progname
                                        waFunctionName-messageClass.

*   Find any screens declared within the main include
    if not getScreens is initial.
      if previousFG is initial or previousFG <> waFunctionName-functionGroup.
        perform findFunctionScreenFlow using waFunctionName.

*       Search for any GUI texts
        perform retrieveGUITitles using waFunctionName-iGUITitle[]
                                        waFunctionName-progname.
      endif.
    endif.
```

```
    if not getTextElements is initial.
*     Find the program texts from out of the database.
      perform retrieveProgramTexts using waFunctionName-iSelectionTexts[]
                                         waFunctionName-iTextElements[]
                                         waFunctionName-progname.
    endif.

    previousFG = waFunctionName-functionGroup.
    modify iLocFunctionNames from waFunctionName.
  endloop.
endform.                                                        "retrieveFunctions

*---------------------------------------------------------------------------------------------------------------
*  retrieveFunctionDetail...   Retrieve function module details from SAP DB.
*---------------------------------------------------------------------------------------------------------------
form retrieveFunctionDetail using value(functionName)
                                  progname
                                  includeName
                                  titleText.

  select single pname
           include
           from tfdir
           into (progname, includeName)
           where funcName = functionName.

  if sy-subrc = 0.
    select single stext
             from tftit
             into titleText
             where spras = sy-langu
               and funcName = functionName.
  endif.
endform.                                                      "retrieveFunctionDetail

*---------------------------------------------------------------------------------------------------------------
*  findMainFunctionInclude...  Find the main include that contains the source code
*---------------------------------------------------------------------------------------------------------------
```

```
form findMainFunctionInclude using value(programName)
                                  value(includeNo)
                                      internalIncludeName.
data: newIncludeNumber type string.

  concatenate '%U' includeNo into newIncludeNumber.
  select single include
            from d010inc
            into internalIncludeName
            where master = programName
              and include like newIncludeNumber.
endform.                                                        "findMainFunctionInclude

*---------------------------------------------------------------------------------------------------------------
*  findFunctionTopInclude...  Find the top include for the function group
*---------------------------------------------------------------------------------------------------------------
form findFunctionTopInclude using value(programName)
                                  topIncludeName.

  select single include
            from d010inc
            into topIncludeName
            where master = programName
              and include like '%TOP'.
endform.                                                        "findFunctionTopInclude

*---------------------------------------------------------------------------------------------------------------
* scanForAdditionalFuncStuff... Search for additional things relating to functions
*---------------------------------------------------------------------------------------------------------------
form scanForAdditionalFuncStuff using iLocFunctions like iFunctions[]
                                    value(recursiveIncludes)
                                    value(recursiveFunctions)
                                    value(searchForIncludes)
                                    value(searchForFunctions)
                                    value(searchForDictionary)
                                    value(searchForMessages)
                                    value(customerOnly)
                                    value(customerNameRange).
```

```
data: waFunction type tFunction.
data: waInclude type tInclude.

  loop at iLocFunctions into waFunction.
    if not searchForIncludes is initial.
*     Search in the main include
      perform scanForIncludePrograms using waFunction-functionMainInclude
                                           recursiveIncludes
                                           customerOnly
                                           customerNameRange
                                           waFunction-iIncludes[].

*     Search in the top include
      perform scanForIncludePrograms using waFunction-topIncludeName
                                           recursiveIncludes
                                           customerOnly
                                           customerNameRange
                                           waFunction-iIncludes[].
    endif.

    if not searchForFunctions is initial.
      perform scanForFunctions using waFunction-functionMainInclude
                                     waFunction-programLinkName
                                     recursiveIncludes
                                     recursiveFunctions
                                     customerOnly
                                     customerNameRange
                                     iLocFunctions[].
    endif.

    modify iLocFunctions from waFunction.
  endloop.

* Now we have everthing perhaps we had better find all the dictionary structures
  if not searchForDictionary is initial.
    loop at iLocFunctions into waFunction.
      perform scanForTables using waFunction-progname
                                  customerOnly
```

```
                        customerNameRange
                        waFunction-iDictStruct[].

      perform scanForLikeOrType using waFunction-progname
                          customerOnly
                          customerNameRange
                          waFunction-iDictStruct[].

      loop at waFunction-iIncludes into waInclude.
        perform scanForTables using waInclude-includeName
                        customerOnly
                        customerNameRange
                        waFunction-iDictStruct[].

        perform scanForLikeOrType using waInclude-includeName
                          customerOnly
                          customerNameRange
                          waFunction-iDictStruct[].
      endloop.

    modify iLocFunctions from waFunction.
    endloop.
  endif.

* Now search for all messages
  if not searchForMessages is initial.
    loop at iLocFunctions into waFunction.
      perform scanForMessages using waFunction-progName
                        waFunction-messageClass
                        waFunction-iMessages[].
    modify iLocFunctions from waFunction.
    endloop.
  endif.
endform.                                                    "scanForAdditionalFuncStuff

*---------------------------------------------------------------------------------------------------------------
* scanForClasses... Search each class or method for other classes
*---------------------------------------------------------------------------------------------------------------
form scanForClasses using value(className)
```

```
                    value(classLinkName)
                    value(customerOnly)
                    value(customerNameRange)
                        iLocClasses like iClasses[].

data iLines type standard table of string with header line.
data: head type string.
data: tail type string.
data: lineLength type i value 0.
data: waLine type string.
data: waClass type tClass.
data: castClassName type program.
data: exceptionCustomerNameRange type string.

* Build the name of the possible cusotmer exception classes
  concatenate customerNameRange 'CX_' into  exceptionCustomerNameRange.

* Read the program code from the textpool.
  castClassName = className.
  read report castClassName into iLines.

  loop at iLines into waLine.
*   Find custom tables.
    lineLength = strLen( waLine ).
    if lineLength > 0.
      if waLine(1) = ASTERIX.
        continue.
      endif.

      translate waLine to upper case.

      find TYPEREFTO in waLine ignoring case.
      if sy-subrc = 0.
*       Have found a reference to another class
        split waLine at TYPE into head tail.
        shift tail left deleting leading space.
        split tail at 'REF' into head tail.
        shift tail left deleting leading space.
        split tail at 'TO' into head tail.
```

```
    shift tail left deleting leading space.
    if tail cs PERIOD.
      split tail at PERIOD into head tail.
    else.
      if tail cs COMMA.
        split tail at COMMA into head tail.
      endif.
    endif.
  else.
*     Try and find classes which are only referenced through static mehods
    find '=>' in waLine match offset sy-fdpos.
    if sy-subrc = 0.
      head = waline+0(sy-fdpos).
      shift head left deleting leading space.
      condense head.
      find 'call method' in head ignoring case.
      if sy-subrc = 0.
        shift head left deleting leading space.
        split head at space into head tail.
        split tail at space into head tail.
*         Should have the class name here
        head = tail.
      else.
*         Still have a class name even though it does not have the words call method in front
        if waLine cs '='.
          split waLine at '=' into tail head.
          shift head left deleting leading space.
          split head at '=' into head tail.
        endif.
        sy-subrc = 0.
      endif.
    endif.
  endif.

  if sy-subrc = 0.
    try.
      if head+0(1) = 'Y' or head+0(1) = 'Z' or head cs customerNameRange.
```

```
*         We have found a class best append it to our class table if we do not already have it.
          read table iLocClasses into waClass with key clsName = head.
          if sy-subrc <> 0.
            if head+0(3) = 'CX_'
              or head+0(4) = 'ZCX_'
              or head+0(4) = 'YCX_'
              or head cs exceptionCustomerNameRange.

             waClass-exceptionClass = TRUE.
            endif.

            waClass-clsname = head.
            append waClass to iLocClasses.
          endif.
        endif.
        catch cx_sy_range_out_of_bounds.
      endtry.
    endif.
  endif.
 endloop.
endform.                                                                    "scanForClasses

*---------------------------------------------------------------------------------------------------------------
* scanForIncludePrograms... Search each program for include programs
*---------------------------------------------------------------------------------------------------------------
form scanForIncludePrograms using value(programName)
                                  value(recursiveIncludes)
                                  value(customerOnly)
                                  value(customerNameRange)
                                      iLocIncludes like dumiIncludes[].

data: iIncludeLines type standard table of string with header line.
data: iTokens type standard table of stokes with header line.
data: iKeywords type standard table of text20 with header line.
data: iStatements type standard table of sstmnt with header line.
data: waTokens type stokes.
data: waInclude type tInclude.
data: waIncludeExists type tInclude.
```

```
data: maxLines type i.
data: nextLine type i.
data: castProgramName type program.

* Read the program code from the textpool.
  castProgramName = programName.
  read report castProgramName into iIncludeLines.

  append INCLUDE to iKeywords.
  scan abap-source iIncludeLines tokens into iTokens with includes statements into iStatements keywords from i
Keywords.

  clear iIncludeLines[].

  maxLines = lines( iTokens ).
  loop at iTokens where str = INCLUDE and type = 'I'.
    nextLine = sy-tabix + 1.
    if nextLine <= maxLines.
      read table iTokens index nextLine into waTokens.

*     Are we only to find customer includes?
      if not customerOnly is initial.
        try.
          if waTokens-str+0(1) = 'Y' or waTokens-str+0(1) = 'Z' or waTokens-str cs customerNameRange
            or waTokens-str+0(2) = 'MZ' or waTokens-str+0(2) = 'MY'.

          else.
            continue.
          endif.
          catch cx_sy_range_out_of_bounds into objRuntimeError.
        endtry.
      endif.

      waInclude-includeName = waTokens-str.

*     Best find the program title text as well.
      perform findProgramOrIncludeTitle using waInclude-includeName
                                              waInclude-includeTitle.
```

```
*     Don't append the include if we already have it listed
      read table iLocIncludes into waIncludeExists with key includeName = waInclude-includeName.
      if sy-subrc <> 0.
        append waInclude to iLocIncludes.

        if not recursiveIncludes is initial.
*         Do a recursive search for other includes
          perform scanForIncludePrograms using waInclude-includeName
                                              recursiveIncludes
                                              customerOnly
                                              customerNameRange
                                              iLocIncludes[].
        endif.
      endif.
    endif.
  endloop.
endform.                                                        "scanForIncludePrograms

*---------------------------------------------------------------------------------------------------------
* scanForFunctions... Search each program for function modules
*---------------------------------------------------------------------------------------------------------
form scanForFunctions using value(programName)
                      value(programLinkName)
                      value(recursiveIncludes)
                      value(recursiveFunctions)
                      value(customerOnly)
                      value(customerNameRange)
                          iLocFunctions like iFunctions[].

data: iIncludeLines type standard table of string with header line.
data: iTokens type standard table of stokes with header line.
data: iStatements type standard table of sstmnt with header line.
data: waTokens type stokes.
data: waFunction type tFunction.
data: waFunctionComparison type tFunction.
data: maxLines type i.
data: nextLine type i.
```

```
data: castProgramName type program.
data: skipThisloop type i.

* Read the program code from the textpool.
  castProgramName = programName.
  read report castProgramName into iIncludeLines.
  scan abap-source iIncludeLines tokens into iTokens with includes statements into iStatements.
  clear iIncludeLines[].

  maxLines = lines( iTokens ).
  loop at iTokens where str = FUNCTION and type = 'I'.

    nextLine = sy-tabix + 1.
    if nextLine <= maxLines.
      read table iTokens index nextLine into waTokens.

*     Are we only to find customer functions
      skipThisLoop = FALSE.
      if not customerOnly is initial.
        try.
          if waTokens-str+1(1) = 'Y' or waTokens-str+1(1) = 'Z' or waTokens-str cs customerNameRange.
          else.
            skipThisLoop = TRUE.
          endif.
        catch cx_sy_range_out_of_bounds into objRuntimeError.
        cleanup.
          skipThisLoop = TRUE.
        endtry.
      endif.

      if skipThisLoop = FALSE.
        waFunction-functionName = waTokens-str.
        replace all occurrences of "" in waFunction-functionName with ' '.
        condense waFunction-functionName.

*       Don't add a function if we alread have it listed.
        read table iLocFunctions with key functionName = waFunction-functionName into waFunctionComparison.
        if sy-subrc <> 0.
```

```
*       Add in the link name if the function is linked to a program
        waFunction-programLinkName = programLinkName.

*       Don't download functions which are called through an RFC destination
        nextline = sy-tabix + 2.
        read table iTokens index nextLine into waTokens.
        if waTokens-str <> DESTINATION.

*         Find the function group
          select single area from v_fdir into wafunction-functionGroup where funcName = waFunction-functionNam
e.

          if sy-subrc = 0.
*           Best find the function number as well.
            perform retrieveFunctionDetail using waFunction-functionName
                                                waFunction-progname
                                                waFunction-includeNumber
                                                waFunction-functionTitle.

            perform findMainFunctionInclude using waFunction-progname
                                                  waFunction-includeNumber
                                                  waFunction-functionMainInclude.

            perform findFunctionTopInclude using waFunction-progname
                                                waFunction-topIncludeName.

*           Find main message class
            perform findMainMessageClass using waFunction-progname
                                               waFunction-messageClass.

            append waFunction to iLocFunctions.

*           Now lets search a little bit deeper and do a recursive search for other includes
            if not recursiveIncludes is initial.
              perform scanForIncludePrograms using waFunction-functionMainInclude
                                                   recursiveIncludes
                                                   customerOnly
                                                   customerNameRange
```

```
                                    waFunction-iIncludes[].
            endif.

*           Now lets search a little bit deeper and do a recursive search for other functions
            if not recursiveFunctions is initial.
              perform scanForFunctions using waFunction-functionMainInclude
                                  space
                                  recursiveIncludes
                                  recursiveFunctions
                                  customerOnly
                                  customerNameRange
                                  iLocFunctions[].
            endif.
            clear waFunction.
           endif.
          endif.
         endif.

         clear waFunction.
        endif.
      endif.
    endloop.
endform.                                                        "scanForFunctions

*---------------------------------------------------------------------------------------------------------------
*  scanForFunctionIncludes... Find all user defined includes within the function group
*---------------------------------------------------------------------------------------------------------------
form scanForFunctionIncludes using poolName
                          value(customerOnly)
                          value(customerNameRange)
                          iLocIncludes like dumiIncludes[].

data: iIncludeLines type standard table of string with header line.
data: iTokens type standard table of stokes with header line.
data: iKeywords type standard table of text20 with header line.
data: iStatements type standard table of sstmnt with header line.
data: waTokens type stokes.
data: waInclude type tInclude.
```

```
data: waIncludeExists type tInclude.
data: maxLines type i.
data: nextLine type i.
data: castProgramName type program.

* Read the program code from the textpool.
  castProgramName = poolName.
  read report castProgramName into iIncludeLines.

  append INCLUDE to iKeywords.
  scan abap-source iIncludeLines tokens into iTokens with includes statements into iStatements keywords from i
Keywords.

  clear iIncludeLines[].

  maxLines = lines( iTokens ).
  loop at iTokens where str = INCLUDE and type = 'I'.
    nextLine = sy-tabix + 1.
    if nextLine <= maxLines.
      read table iTokens index nextLine into waTokens.

      if waTokens-str cp '*F++'.
*       Are we only to find customer includes?
        if not customerOnly is initial.
          try.
            if waTokens-str+0(2) = 'LY' or waTokens-str+0(2) = 'LZ' or waTokens-str cs customerNameRange.
            else.
              continue.
            endif.
            catch cx_sy_range_out_of_bounds into objRuntimeError.
          endtry.
        endif.

        waInclude-includeName = waTokens-str.

*       Best find the program title text as well.
        perform findProgramOrIncludeTitle using waInclude-includeName
                                                waInclude-includeTitle.
```

```
*       Don't append the include if we already have it listed
        read table iLocIncludes into waIncludeExists with key includeName = waInclude-includeName.
        if sy-subrc <> 0.
          append waInclude to iLocIncludes.
        endif.
       endif.
     endif.
   endloop.
endform.                                                    "scanForFunctionIncludes

*---------------------------------------------------------------------------------------------------------
*  findProgramOrIncludeTitle...   Finds the title text of a program.
*---------------------------------------------------------------------------------------------------------
form findProgramOrIncludeTitle using value(programName)
                                     titleText.

  select single text
            from trdirt
            into titleText
            where name = programName
              and sprsl = sy-langu.
endform.                                                    "findProgramOrIncludeTitle

*---------------------------------------------------------------------------------------------------------
* retrievePrograms...   find programs and sub objects from SAP DB
*---------------------------------------------------------------------------------------------------------
form retrievePrograms using iLocProgram like iPrograms[]
                      iLocFunctions like iFunctions[]
                      rangeProgram like soProgramName[]
                      rangeAuthor like soAuthor[]
                      value(custNameRange)
                      value(alsoModifiedByauthor)
                      value(customerProgsOnly)
                      value(getMessages)
                      value(getTextElements)
                      value(getCustDictStructures)
                      value(getFunctions)
                      value(getIncludes)
```

```
                    value(getScreens)
                    value(recursiveFuncSearch)
                    value(recursiveIncludeSearch)
                    value(getLocalObjects)
                    value(package).

data: waRangeProgram like line of rangeProgram.

  if rangeProgram[] is initial.
*   We are finding all programs by an author
    perform findAllProgramsForAuthor using iLocProgram[]
                                rangeProgram[]
                                rangeAuthor[]
                                custNameRange
                                alsoModifiedByAuthor
                                customerProgsOnly
                                getLocalObjects
                                package.
  else.
    read table rangeProgram index 1 into waRangeProgram.
    if waRangeProgram-low cs ASTERIX.
      perform findProgramsByWildcard using iLocProgram[]
                                rangeProgram[]
                                rangeAuthor[]
                                custNameRange
                                customerProgsOnly
                                getLocalObjects
                                package.
    else.
      perform checkProgramDoesExist using iLocProgram[]
                               rangeProgram[].
    endif.
  endif.

* Find extra items
  perform scanForAdditionalProgStuff using iLocProgram[]
                                iLocFunctions[]
                                getTextElements
```

```
                                    getMessages
                                    getScreens
                                    getCustDictStructures
                                    getFunctions
                                    getIncludes
                                    customerProgsOnly
                                    custNameRange
                                    recursiveIncludeSearch
                                    recursiveFuncSearch.
endform.                                                    "retrievePrograms

*-----------------------------------------------------------------------------------------------------
*   scanForAdditionalProgStuff...
*-----------------------------------------------------------------------------------------------------
form scanForAdditionalProgStuff using iLocProgram like iPrograms[]
                                iLocFunctions like iFunctions[]
                                value(getTextElements)
                                value(getMessages)
                                value(getScreens)
                                value(getCustDictStructures)
                                value(getFunctions)
                                value(getIncludes)
                                value(customerOnly)
                                value(customerNameRange)
                                value(recursiveIncludeSearch)
                                value(recursiveFuncSearch).

data: waProgram type tProgram.
data: waInclude type tInclude.
data: myTabix type syTabix.

* Best to find all the includes used in a program first
  if not getIncludes is initial.
    loop at iLocProgram into waProgram.
      myTabix = sy-tabix.
      perform scanForIncludePrograms using waProgram-progName
                                    recursiveIncludeSearch
                                    customerOnly
```

```
                                  customerNameRange
                                  waProgram-iIncludes[].

      modify iLocProgram from waProgram index myTabix.
    endloop.
  endif.

* Once we have a list of all the includes we need to loop round them an select all the other objects
  loop at iLocProgram into waProgram.
    myTabix = sy-tabix.
    perform findProgramDetails using waProgram-progName
                                     waProgram-subc
                                     waProgram-programTitle
                                     waProgram
                                     getTextElements
                                     getMessages
                                     getScreens
                                     getCustDictStructures
                                     customerOnly
                                     customerNameRange.

*   Find any screens
    if not getScreens is initial.
      perform findProgramScreenFlow using waProgram.
    endif.

    loop at waProgram-iIncludes into waInclude.
      perform findProgramDetails using waInclude-includeName
                                       'I'
                                       waInclude-includeTitle
                                       waProgram
                                       getTextElements
                                       getMessages
                                       getScreens
                                       getCustDictStructures
                                       customerOnly
                                       customerNameRange.
    endloop.
```

```
    modify iLocProgram from waProgram index myTabix.
  endloop.

* Now we have all the program includes and details we need to find extra functions
  if not getFunctions is initial.
    loop at iLocProgram into waProgram.
*     Find any functions defined in the code
      perform scanForFunctions using waProgram-progname
                                     waProgram-progname
                                     space
                                     space
                                     customerOnly
                                     customerNameRange
                                     iLocFunctions[].
    endloop.
  endif.

* We have a list of all the functions so lets go and find details and other function calls
  perform scanForAdditionalFuncStuff using iLocFunctions[]
                                           recursiveIncludeSearch
                                           recursiveFuncSearch
                                           getIncludes
                                           getFunctions
                                           getCustDictStructures
                                           getMessages
                                           customerOnly
                                           customerNameRange.
endform.                                                    "scanForAdditionalProgStuff

*---------------------------------------------------------------------------------------------------
*   findProgramDetails...
*---------------------------------------------------------------------------------------------------
form findProgramDetails using value(programName)
                              value(programType)
                                  programTitle
                                  waProgram type tProgram
                              value(getTextElements)
                              value(getMessages)
```

```
                    value(getScreens)
                    value(getCustDictStructures)
                    value(customerOnly)
                    value(customerNameRange).

  perform findProgramOrIncludeTitle using programName
                                          programTitle.

  if not getTextElements is initial.
*   Find the program texts from out of the database.
    perform retrieveProgramTexts using waProgram-iSelectionTexts[]
                                       waProgram-iTextElements[]
                                       programName.
  endif.

* Search for any GUI texts
  if not getScreens is initial and ( programType = 'M' or programType = '1' ).
    perform retrieveGUITitles using waProgram-iGUITitle[]
                                    programName.
  endif.

* Find individual messages
  if not getMessages is initial.
    if programType = 'M' or programType = '1'.
      perform findMainMessageClass using programName
                                         waProgram-messageClass.
    endif.

    perform scanForMessages using programName
                                  waProgram-messageClass
                                  waProgram-iMessages[].
  endif.

  if not getCustDictStructures is initial.
    perform scanForTables using programName
                                customerOnly
                                customerNameRange
                                waProgram-iDictStruct[].
```

```
    perform scanForLikeOrType using programName
                                    customerOnly
                                    customerNameRange
                                    waProgram-iDictStruct[].
  endif.
endform.                                                    "findProgramDetails

*---------------------------------------------------------------------------------------------------
*   findAllProgramsForAuthor...
*---------------------------------------------------------------------------------------------------
form findAllProgramsForAuthor using iLocProgram like iPrograms[]
                                    rangeProgram like soProgramName[]
                                    rangeAuthor like soAuthor[]
                                    value(custNameRange)
                                    value(alsoModifiedByauthor)
                                    value(customerProgsOnly)
                                    value(getLocalObjects)
                                    value(package).

data: altCustomerNameRange type string.
field-symbols: <waProgram> type tProgram.
data: genFlag type genFlag.

* build up the customer name range used for select statements
  concatenate custNameRange '%' into altCustomerNameRange.

* select by name and author
  if not alsoModifiedByAuthor is initial.
*   Programs modified by author
*   Program to search for is an executable program
    if customerProgsOnly is initial.
*     Select all programs
      select progname
             subc
             from reposrc
             appending corresponding fields of table ilocProgram
             where progname in rangeProgram
```

```
          and cnam in rangeAuthor
          and ( subc = '1' or subc = 'M' or subc = 'S' ).

    else.
*     Select only customer specific programs
      select progname
           subc
           from reposrc
           appending corresponding fields of table iLocProgram
           where progname  in rangeProgram
             and ( progname like altCustomerNameRange
                  or progname like 'Z%'
                  or progname like 'Y%'
                  or progname like 'SAPMZ%'
                  or progname like 'SAPMY%')
             and cnam in rangeAuthor
             and ( subc = '1' or subc = 'M' or subc = 'S' ).
    endif.
  else.

*   Programs created by author
    if customerProgsOnly is initial.
*     Select all programs
      select progname
           subc
           from reposrc
           appending corresponding fields of table iLocProgram
           where progname in rangeProgram
             and ( subc = '1' or subc = 'M' or subc = 'S' )
             and ( cnam in rangeAuthor or unam in rangeAuthor ).
    else.
*     Select only customer specific programs
      select progname
           subc
           from reposrc
           appending corresponding fields of table iLocProgram
           where progname in rangeProgram
```

```
          and ( progname like altCustomerNameRange
              or progname like 'Z%'
              or progname like 'Y%'
              or progname like 'SAPMZ%'
              or progname like 'SAPMY%')
          and ( subc = '1' or subc = 'M' or subc = 'S' )
          and ( cnam in rangeAuthor or unam in rangeAuthor ).
    endif.
  endif.

* Delete any programs which are local objects
  if getLocalObjects is initial.
    loop at iLocProgram assigning <waProgram>.
      select single genflag
                from tadiv
                into genflag
                where pgmid = 'R3TR'
                  and object = 'PROG'
                  and obj_name = <waProgram>-progName
                  and devclass = '$TMP'.
      if sy-subrc = 0.
        delete iLocProgram.
      endif.
    endloop.
  endif.

* Delete any programs which are not in the specified package
  if not package is initial.
    if package cs '*'.
      translate package using '*%'.
    endif.

    loop at iLocProgram assigning <waProgram>.
      select single genflag
                from tadiv
                into genflag
                where pgmid = 'R3TR'
                  and object = 'PROG'
```

```
                and obj_name = <waProgram>-progName
                and devclass like package.
      if sy-subrc <> 0.
        delete iLocProgram.
      endif.
    endloop.
  endif.
endform.                                                "findAllProgramsForAuthor

*---------------------------------------------------------------------------------------------------
*  checkProgramDoesExist...
*---------------------------------------------------------------------------------------------------
form checkProgramDoesExist using iLocProgram like iPrograms[]
                          rangeProgram like soProgramName[].

data: waProgram type tProgram.

*  Check to see if the program is an executable program
   select single progname
            subc
            into (waProgram-progname, waProgram-subc)
            from reposrc
            where progname in rangeProgram
              and ( subc = '1' or
                    subc = 'I' or
                    subc = 'M' or
                    subc = 'S' ).

   if not waProgram-progname is initial.
     append waProgram to iLocProgram.
   endif.
endform.                                                "checkProgramDoesExist

*---------------------------------------------------------------------------------------------------
*  findProgramsByWildcard.. Search in the system for programs
*---------------------------------------------------------------------------------------------------
form findProgramsByWildcard using iLocProgram like iPrograms[]
                            value(rangeProgram) like soProgramName[]
```

```
                    value(rangeAuthor) like soAuthor[]
                    value(custNameRange)
                    value(customerProgsOnly)
                    value(getLocalObjects)
                    value(package).

data: altCustomerNameRange type string.
field-symbols: <waProgram> type tProgram.
data: genFlag type genFlag.

  if customerProgsOnly is initial.
*   build up the customer name range used for select statements
    if custNameRange <> '^'.
      concatenate custNameRange '%' into altCustomerNameRange.

      select progname
             subc
             from reposrc
             appending corresponding fields of table iLocProgram
             where progname  in rangeProgram
               and progname like altCustomerNameRange
               and ( subc = '1' or subc = 'M' or subc = 'S' )
               and ( cnam in rangeAuthor or unam in rangeAuthor ).
    else.
      select progname
             subc
             from reposrc
             appending corresponding fields of table iLocProgram
             where progname  in rangeProgram
               and ( subc = '1' or subc = 'M' or subc = 'S' )
               and ( cnam in rangeAuthor or unam in rangeAuthor ).
    endif.
  else.
*   Only customer programs
    if custNameRange <> '^'.
      concatenate custNameRange '%' into altCustomerNameRange.
```

```
    select progname
           subc
           from reposrc
           appending corresponding fields of table iLocProgram
           where progname  in rangeProgram
             and ( progname like altCustomerNameRange
                   or progname like 'Z%'
                   or progname like 'Y%'
                   or progname like 'SAPMZ%'
                   or progname like 'SAPMY%')
             and ( subc = '1' or subc = 'M' or subc = 'S' )
             and ( cnam in rangeAuthor or unam in rangeAuthor ).
  else.
    select progname
           subc
           from reposrc
           appending corresponding fields of table iLocProgram
           where progname  in rangeProgram
           and ( progname like 'Z%'
               or progname like 'Y%'
               or progname like 'SAPMZ%'
               or progname like 'SAPMY%')
           and ( subc = '1' or subc = 'M' or subc = 'S' )
           and ( cnam in rangeAuthor or unam in rangeAuthor ).
  endif.
 endif.

* Delete any programs which are local objects
 if getLocalObjects is initial.
  loop at iLocProgram assigning <waProgram>.
    select single genflag
                from tadiv
                into genflag
                where pgmid = 'R3TR'
                  and object = 'PROG'
                  and obj_name = <waProgram>-progName
                  and devclass = '$TMP'.
```

```
      if sy-subrc = 0.
        delete iLocProgram.
      endif.
    endloop.
  endif.

* Delete any programs which are not in the specified package
  if not package is initial.
    loop at iLocProgram assigning <waProgram>.
      select single genflag
                from tadiv
                into genflag
                where pgmid = 'R3TR'
                  and object = 'PROG'
                  and obj_name = <waProgram>-progName
                  and devclass <> package.
      if sy-subrc = 0.
        delete iLocProgram.
      endif.
    endloop.
  endif.
endform.                                                    "findProgramsByWildcard

*---------------------------------------------------------------------------------------------------
*  retrieveProgramTexts... Find the text elements and selection texts for a program
*---------------------------------------------------------------------------------------------------
form retrieveProgramTexts using iLocSelectionTexts like dumiTextTab[]
                                iLocTextElements like dumiTextTab[]
                                value(programName).

data: iTextTable type standard table of tTextTable with header line.
data: waTexts type tTextTable.
data: castProgramName(50).

  move programName to castProgramName.

  read textpool castProgramName into iTextTable language sy-langu.
  delete iTextTable where key = 'R'.
```

```
* Selection texts.
  loop at iTextTable where id = 'S'.
    move iTextTable-key to waTexts-key.
    move iTextTable-entry to waTexts-entry.
    append waTexts to iLocSelectiontexts.
    clear waTexts.
  endloop.

* Text elements.
  delete iTextTable where key = 'S'.
  loop at iTextTable where id = 'I'.
    move iTextTable-key to waTexts-key.
    move iTextTable-entry to waTexts-entry.
    append waTexts to iLocTextElements.
  endloop.
endform.                                                    "retrieveProgramTexts

*---------------------------------------------------------------------------------------------------
*  retrieveGUITitles...  Search for any GUI texts
*---------------------------------------------------------------------------------------------------
form retrieveGUITitles using iLocGUITitle like dumIGUITitle[]
                        value(programName).

  select obj_code
        text
        from d347t
        appending corresponding fields of table iLocGUItitle
        where progname = programName.
endform.                                                    "retrieveGUITitles

*---------------------------------------------------------------------------------------------------
*   findMainMessageClass... find the message class stated at the top of  program.
*---------------------------------------------------------------------------------------------------
form findMainMessageClass using value(programName)
                              messageClass.

  select single msgid
              from trdire into messageClass
```

```
            where report = programName.
endform.                                                    "findMainMessageClass

*---------------------------------------------------------------------------------------------------
* retrieveClasses...   find classes and sub objects from SAP DB
*---------------------------------------------------------------------------------------------------
form retrieveClasses using iLocClasses like iClasses[]
                    iLocFunctions like iFunctions[]
                    rangeClass like soClassName[]
                    rangeAuthor like soAuthor[]
                    value(custNameRange)
                    value(alsoModifiedByauthor)
                    value(customerProgsOnly)
                    value(getMessages)
                    value(getTextElements)
                    value(getCustDictStructures)
                    value(getFunctions)
                    value(getIncludes)
                    value(recursiveFuncSearch)
                    value(recursiveIncludeSearch)
                    value(recursiveClassSearch)
                    value(language).

data: waRangeClass like line of rangeClass.

  if rangeClass[] is initial.
*   We are finding all programs by an author
    perform findAllClassesForAuthor using iLocClasses[]
                                  rangeClass[]
                                  rangeAuthor[]
                                  custNameRange
                                  alsoModifiedByAuthor
                                  customerProgsOnly
                                  language.
  else.
    read table rangeClass index 1 into waRangeClass.
    if waRangeClass-low cs ASTERIX.
      perform findClassesByWildcard using iLocClasses[]
```

```
                                rangeClass[]
                                rangeAuthor[]
                                custNameRange
                                customerProgsOnly
                                language.
    else.
      perform checkClassDoesExist using iLocClasses[]
                                rangeClass[].
    endif.
  endif.

* Find extra items
  if not iLocClasses[] is initial.
    perform scanForAdditionalClassStuff using iLocClasses[]
                                    iLocFunctions[]
                                    getTextElements
                                    getMessages
                                    getCustDictStructures
                                    getFunctions
                                    getIncludes
                                    customerProgsOnly
                                    custNameRange
                                    recursiveIncludeSearch
                                    recursiveFuncSearch
                                    recursiveClassSearch.
  endif.
endform.                                                    "retrieveClasses

*---------------------------------------------------------------------------------------------------
*   findAllClassesForAuthor...
*---------------------------------------------------------------------------------------------------
form findAllClassesForAuthor using iLocClass like iClasses[]
                              rangeClass like soClassName[]
                              rangeAuthor like soAuthor[]
                              value(custNameRange)
                              value(alsoModifiedByauthor)
```

```
                    value(customerClassesOnly)
                    value(language).

data: altCustomerNameRange(2).

* build up the customer name range used for select statements
  concatenate custNameRange '%' into altCustomerNameRange.

* select by name and author
  if not alsoModifiedByAuthor is initial.
*   Classes modified by author
    if customerClassesOnly is initial.
*     Select all classes
      select clsname descript msg_id
          from vseoclass
          appending corresponding fields of table ilocClass
          where clsname in rangeClass
            and langu = language
            and ( author in rangeAuthor or changedby in rangeAuthor )
            and version = '1'
            and ( state = '0' or state = '1' ).

      if sy-subrc <> 0.
        select clsname descript msg_id
            from vseoclass
            appending corresponding fields of table ilocClass
            where clsname in rangeClass
            and langu = language
              and ( author in rangeAuthor or changedby in rangeAuthor )
              and version = '0'
              and ( state = '0' or state = '1' ).
      endif.
    else.
*     Select only customer specific classes
      select clsname descript msg_id
          from vseoclass
          appending corresponding fields of table ilocClass
          where clsname in rangeClass
```

```
          and ( clsname like altCustomerNameRange
              or clsname like 'Z%'
              or clsname like 'Y%')
          and langu = language
          and ( author in rangeAuthor or changedby in rangeAuthor )
          and version = '1'
          and ( state = '0' or state = '1' ).

      if sy-subrc <> 0.
        select clsname descript msg_id
          from vseoclass
          appending corresponding fields of table ilocClass
          where clsname in rangeClass
            and ( clsname like altCustomerNameRange
                or clsname like 'Z%'
                or clsname like 'Y%')
            and langu = language
            and ( author in rangeAuthor or changedby in rangeAuthor )
            and version = '0'
            and ( state = '0' or state = '1' ).
      endif.
    endif.
  else.
*   Programs created by author
    if customerClassesOnly is initial.
*     Select all classes
      select clsname descript msg_id
          from vseoclass
          appending corresponding fields of table ilocClass
          where clsname in rangeClass
            and langu = language
            and author in rangeAuthor
            and version = '1'
            and ( state = '0' or state = '1' ).

      if sy-subrc <> 0.
        select clsname descript msg_id
```

```
        from vseoclass
        appending corresponding fields of table ilocClass
        where clsname in rangeClass
          and langu = language
          and author in rangeAuthor
          and version = '0'
          and ( state = '0' or state = '1' ).
  endif.
 else.
*   Select only customer specific classes
  select clsname descript msg_id
       from vseoclass
       appending corresponding fields of table ilocClass
       where clsname in rangeClass
        and ( clsname like altCustomerNameRange
            or clsname like 'Z%'
            or clsname like 'Y%')
        and langu = language
        and author in rangeAuthor
        and version = '1'
        and ( state = '0' or state = '1' ).

  if sy-subrc <> 0.
   select clsname descript msg_id
        from vseoclass
        appending corresponding fields of table ilocClass
        where clsname in rangeClass
          and ( clsname like altCustomerNameRange
              or clsname like 'Z%'
              or clsname like 'Y%')
          and langu = language
          and author in rangeAuthor
          and version = '0'
          and ( state = '0' or state = '1' ).
  endif.
 endif.
```

```
  endif.
endform.                                                    "findAllClassesForAuthor

*---------------------------------------------------------------------------------------------------
*  findClassesByWildcard...  Find classes using a wildcard search
*---------------------------------------------------------------------------------------------------
form findClassesByWildcard using iLocClass like iClasses[]
                          rangeClass like soClassName[]
                          value(rangeAuthor) like soAuthor[]
                          value(custNameRange)
                          value(customerClassesOnly)
                          value(language).

data: altCustomerNameRange(2).

  if customerClassesOnly is initial.
*   Searching for customer and SAP classes
    if custNameRange <> '^'.
*     build up the customer name range used for select statements
      concatenate custNameRange '%' into altCustomerNameRange.

      select clsname descript msg_id
           from vseoclass
           appending corresponding fields of table ilocClass
           where clsname in rangeClass
             and clsname like custNameRange
             and langu = language
             and ( author in rangeAuthor or changedby in rangeAuthor )
             and version = '1'
             and ( state = '0' or state = '1' ).
      if sy-subrc <> 0.
        select clsname descript msg_id
             from vseoclass
             appending corresponding fields of table ilocClass
             where clsname in rangeClass
               and clsname like custNameRange
               and langu = language
               and ( author in rangeAuthor or changedby in rangeAuthor )
```

```
            and version = '0'
            and ( state = '0' or state = '1' ).
      endif.
    else.
*     Searching using normal name ranges
      select clsname descript msg_id
          from vseoclass
          appending corresponding fields of table ilocClass
          where clsname in rangeClass
            and langu = language
            and ( author in rangeAuthor or changedby in rangeAuthor )
            and version = '1'
            and ( state = '0' or state = '1' ).
      if sy-subrc <> 0.
        select clsname descript msg_id
            from vseoclass
            appending corresponding fields of table ilocClass
            where clsname in rangeClass
              and langu = language
              and ( author in rangeAuthor or changedby in rangeAuthor )
              and version = '0'
              and ( state = '0' or state = '1' ).
      endif.
    endif.
  else.
*   searching for only customer classes
    if custNameRange <> '^'.
*     build up the customer name range used for select statements
      concatenate custNameRange '%' into altCustomerNameRange.

      select clsname descript msg_id
          from vseoclass
          appending corresponding fields of table ilocClass
          where clsname in rangeClass
            and clsname like custNameRange
            and langu = language
            and ( clsname like 'ZC%' or clsname like 'YC%' )
```

```
        and ( author in rangeAuthor or changedby in rangeAuthor )
        and version = '1'
        and ( state = '0' or state = '1' ).
  if sy-subrc <> 0.
   select clsname descript msg_id
        from vseoclass
        appending corresponding fields of table ilocClass
        where clsname in rangeClass
         and langu = language
         and ( clsname like 'ZC%' or clsname like 'YC%' )
         and ( author in rangeAuthor or changedby in rangeAuthor )
         and version = '0'
         and ( state = '0' or state = '1' ).
  endif.
 else.
*  Searching using normal name ranges
  select clsname descript msg_id
       from vseoclass
       appending corresponding fields of table ilocClass
       where clsname in rangeClass
        and ( clsname like 'ZC%' or clsname like 'YC%' )
        and ( author in rangeAuthor or changedby in rangeAuthor )
        and version = '1'
        and ( state = '0' or state = '1' ).
  if sy-subrc <> 0.
   select clsname descript msg_id
        from vseoclass
        appending corresponding fields of table ilocClass
        where clsname in rangeClass
         and ( clsname like 'ZC%' or clsname like 'YC%' )
         and ( author in rangeAuthor or changedby in rangeAuthor )
         and version = '0'
         and ( state = '0' or state = '1' ).
  endif.
 endif.
```

```
  endif.
endform.                                                    "findClassesByWildcard

*---------------------------------------------------------------------------------------------------
*   checkClassDoesExist...
*---------------------------------------------------------------------------------------------------
form checkClassDoesExist using iLocClass like iClasses[]
                               rangeClass like soClassName[].

data: waClass type tClass.

  select single clsname descript msg_id
        from vseoclass
        into corresponding fields of waClass
        where clsname in rangeClass
          and version = '1'
          and ( state = '0' or state = '1' ).

  if sy-subrc <> 0.
    select single clsname descript msg_id
        from vseoclass
        into corresponding fields of waClass
        where clsname in rangeClass
          and version = '0'
          and ( state = '0' or state = '1' ).
  endif.

   if not waClass-clsname is initial.
     append waClass to iLocClass.
   endif.
endform.                                                    "checkClassDoesExist

*---------------------------------------------------------------------------------------------------
*   scanForAdditionalClassStuff...
*---------------------------------------------------------------------------------------------------
form scanForAdditionalClassStuff using iLocClasses like iClasses[]
                                       iLocFunctions like iFunctions[]
                                       value(getTextElements)
                                       value(getMessages)
```

```
                    value(getCustDictStructures)
                    value(getFunctions)
                    value(getIncludes)
                    value(customerOnly)
                    value(customerNameRange)
                    value(recursiveIncludeSearch)
                    value(recursiveFuncSearch)
                    value(recursiveClassSearch).

data: waClass type tClass.
data: waMethod type tMethod.
data: myTabix type syTabix.
data: scanningForClasses type i value FALSE.
data: classNewLines type i value 0.
data: classCurrentLines type i value 0.

  loop at iLocClasses into waClass where scanned is initial.
*  Once we have a list of all the classes we need to loop round them an select all the other objects
    myTabix = sy-tabix.
    perform findClassDetails using waClass-clsName
                        waClass
                        iLocFunctions[]
                        getTextElements
                        getMessages
                        getFunctions
                        getCustDictStructures
                        customerOnly
                        customerNameRange.

*   Set the scanned class so we do not check them again when running recursively.
    waClass-scanned = 'X'.
    modify iLocClasses from waClass index myTabix.
  endloop.

* Now we have all the classes and details we need to find extra classes
  if not recursiveClassSearch is initial.
    classCurrentLines = lines( iLocClasses ).
    loop at iLocClasses into waClass.
```

```
*    Don't try and find any other details for an exception class
     if ( waClass-clsName ns 'ZCX_' or waClass-clsName ns 'CX_' ).
*     Find any classes defined in the main class definition
      perform scanForClasses using waClass-privateClassKey
                                   waClass-clsname
                                   customerOnly
                                   customerNameRange
                                   iLocClasses[].

      perform scanForClasses using waClass-publicClassKey
                                   waClass-clsname
                                   customerOnly
                                   customerNameRange
                                   iLocClasses[].

      perform scanForClasses using waClass-protectedClassKey
                                   waClass-clsname
                                   customerOnly
                                   customerNameRange
                                   iLocClasses[].

      loop at waClass-iMethods into waMethod.
*       Find any classes defined in any of the methods
        perform scanForClasses using waMethod-methodKey
                                     waClass-clsname
                                     customerOnly
                                     customerNameRange
                                     iLocClasses[].
      endloop.
     endif.
    endloop.

*   We have a list of all the classes so lets go and find their details
    classNewLines = lines( iLocClasses ).
    if classNewLines > classCurrentLines.
     perform scanForAdditionalClassStuff using iLocClasses[]
                                               iLocFunctions[]
                                               getTextElements
```

```
                                    getMessages
                                    getCustDictStructures
                                    getFunctions
                                    getIncludes
                                    customerOnly
                                    customerNameRange
                                    recursiveIncludeSearch
                                    recursiveFuncSearch
                                    recursiveClassSearch.
    endif.
  endif.
endform.                                                    "scanForAdditionalClassStuff

*-----------------------------------------------------------------------------------------------------
*   findClassDetails...
*-----------------------------------------------------------------------------------------------------
form findClassDetails using value(className)
                            waClass type tClass
                        iLocFunctions like iFunctions[]
                        value(getTextElements)
                        value(getMessages)
                        value(getFunctions)
                        value(getCustDictStructures)
                        value(customerOnly)
                        value(customerNameRange).

data: iEmptySelectionTexts type standard table of tTextTable.
data: myTabix type syTabix.
data: waMethod type tMethod.

* Build up the keys we will use for finding data
  perform buildClassKeys using waClass.

  if waClass-descript is initial.
    perform findClassDescription using className
                                waClass-descript.
  endif.
```

```
* Find the class attributes.
  select single exposure msg_id state clsfinal r3release
            from vseoclass
            into (waClass-exposure, waClass-msg_id, waClass-state,
                 waClass-clsfinal, waClass-r3release)
            where clsName = waClass-clsName.

* Don't try and find any other details for an exception class
  if ( waClass-clsName cs 'ZCX_' or waClass-clsName cs 'CX_' ).
*   Exception texts
    perform findExceptionTexts using waClass-publicClassKey
                                     waClass-iConcepts[].
    waClass-scanned = 'X'.
  else.
    if not getTextElements is initial.
*     Find the class texts from out of the database.
      perform retrieveProgramTexts using iEmptySelectionTexts[]
                                         waClass-iTextElements[]
                                         waClass-textElementKey.
    endif.

*   Find any declared dictionary structures
    if not getCustDictStructures is initial.
      perform scanForTables using waClass-privateClassKey
                                  customerOnly
                                  customerNameRange
                                  waClass-iDictStruct[].

      perform scanForTables using waClass-publicClassKey
                                  customerOnly
                                  customerNameRange
                                  waClass-iDictStruct[].

      perform scanForTables using waClass-protectedClassKey
                                  customerOnly
                                  customerNameRange
                                  waClass-iDictStruct[].
```

```
  perform scanForTables using waClass-typesClassKey
                              customerOnly
                              customerNameRange
                              waClass-iDictStruct[].

  perform scanForLikeOrType using waClass-privateClassKey
                                 customerOnly
                                 customerNameRange
                                 waClass-iDictStruct[].

  perform scanForLikeOrType using waClass-publicClassKey
                                 customerOnly
                                 customerNameRange
                                 waClass-iDictStruct[].

  perform scanForLikeOrType using waClass-protectedClassKey
                                 customerOnly
                                 customerNameRange
                                 waClass-iDictStruct[].

  perform scanForLikeOrType using waClass-typesClassKey
                                 customerOnly
                                 customerNameRange
                                 waClass-iDictStruct[].
endif.


*  Methods
*  Find all the methods for this class
perform findClassMethods using className
                               waClass-iMethods[].

loop at waClass-iMethods[] into waMethod.
  myTabix = sy-tabix.
*   Find individual messages
  if not getMessages is initial.
    perform scanForMessages using waMethod-methodKey
                                 waClass-msg_id
```

```
                                waClass-iMessages[].
      endif.

      if not getCustDictStructures is initial.
*       Find any declared dictionary structures
        perform scanForTables using waMethod-methodKey
                                    customerOnly
                                    customerNameRange
                                    waClass-iDictStruct[].

        perform scanForLikeOrType using waMethod-methodKey
                                     customerOnly
                                     customerNameRange
                                     waClass-iDictStruct[].
      endif.

      if not getfunctions is initial.
        perform scanForFunctions using waMethod-methodKey
                                    waClass-clsName
                                    space
                                    space
                                    customerOnly
                                    customerNameRange
                                    iLocFunctions[].
      endif.

      modify waClass-iMethods from waMethod index myTabix.
    endloop.
  endif.
endform.                                                    "findClassDetails

*---------------------------------------------------------------------------------------------------
*  buildClassKeys...   Finds the title text of a class.
*---------------------------------------------------------------------------------------------------
form buildClassKeys using waClass type tClass.

data: classNameLength type i.
data: loops type i.
```

```
  classNameLength = strlen( waClass-clsName ).

  cl_oo_classname_service=>get_pubsec_name( exporting clsname = waClass-clsName
                                receiving result = waClass-publicClassKey ).

  cl_oo_classname_service=>get_prisec_name( exporting clsname = waClass-clsName
                                receiving result = waClass-privateClassKey ).

  cl_oo_classname_service=>get_prosec_name( exporting clsname = waClass-clsName
                                receiving result = waClass-protectedClassKey ).


* Text element key - length of text element key has to be 32 characters.
  loops = 30 - classNameLength.
  waClass-textElementKey = waClass-clsName.
  do loops times.
    concatenate waClass-textElementKey '=' into waClass-textElementKey.
  enddo.
* Save this for later.
  concatenate waClass-textElementKey 'CP' into waClass-textElementKey.

* Types Class key - length of class name has to be 32 characters.
  loops = 30 - classNameLength.
  waClass-typesClassKey = waClass-clsName.
  do loops times.
    concatenate waClass-typesClassKey '=' into waClass-typesClassKey.
  enddo.
* Save this for later
  concatenate waClass-typesClassKey 'CT' into waClass-typesClassKey.
endform.                                                    "buildClassKeys

*---------------------------------------------------------------------------------------------------
*  findClassDescription...   Finds the title text of a class.
*---------------------------------------------------------------------------------------------------
form findClassDescription using value(className)
                              titleText.

  select single descript
           from vseoclass
```

```
            into titleText
            where clsname = className
              and langu = sy-langu.
  if sy-subrc <> 0.
    select single descript
              from vseoclass
              into titleText
              where clsname = className.
  endif.
endform.                                                    "findClassDescription

*---------------------------------------------------------------------------------------------------
*  findExceptionTexts...   Fiond the texts of an exception class.
*---------------------------------------------------------------------------------------------------
form findExceptionTexts using publicClassKey
                        iConcepts like dumiConcepts[].

data: castClassName type program.
data: iTempLines type standard table of string with header line.
data: iTokens type standard table of stokes with header line.
data: iKeywords type standard table of text20 with header line.
data: iStatements type standard table of sstmnt with header line.
data: waTokens type stokes.
data: waCurrentToken type stokes.
data: waConcept like line of iConcepts.
data: tokenLength type i.
data: myRow type i.

  castClassName = publicClassKey.
  read report castClassName into iTempLines.

  append 'CONSTANTS' to iKeywords.
  scan abap-source iTempLines tokens into iTokens statements into iStatements keywords from iKeywords.

  delete iTokens where str = 'CONSTANTS'.
  delete iTokens where str = 'VALUE'.
  delete iTokens where str = 'TYPE'.
```

```
  loop at iTokens into waTokens where str = 'SOTR_CONC'.
*   The loop before holds the constant name
    myRow = sy-tabix - 1.
    read table iTokens index myRow into waCurrentToken.
    waConcept-constName = waCurrentToken-str.

*   The loop after holds the constant name
    myRow = myRow + 2.
    read table iTokens index myRow into waCurrentToken.
    tokenLength = strLen( waCurrentToken-str ).
    if tokenLength = 34.
*     Most likely an exception text.
      replace all occurrences of "" in waCurrentToken-str with ' ' .
      waConcept-concept = waCurrentToken-str.
      append waConcept to iConcepts.
    endif.
  endloop.
endform.

*---------------------------------------------------------------------------------------------------
*  findClassMethods...   Finds the methods of a class.
*---------------------------------------------------------------------------------------------------
form findClassMethods using value(className)
                        iLocMethods like dumiMethods[].

data: iMethods type standard table of tMethod with header line.

  select cmpName descript exposure
       from vseomethod
       into corresponding fields of table iMethods
         where clsname = className
           and version = '1'
           and langu = sy-langu
           and ( state = '0' or state = '1' ).

  if sy-subrc <> 0.
    select cmpName descript exposure
         from vseomethod
```

```
        into corresponding fields of table iMethods
        where clsname = className
          and version = '0'
          and langu = sy-langu
          and ( state = '0' or state = '1' ).
  endif.

* Find the method key so that we can acces the source code later
  loop at iMethods.
    perform findMethodKey using className
                                iMethods-cmpName
                                iMethods-methodKey.
    modify iMethods.
  endloop.

  iLocMethods[] = iMethods[].
endform.                                                    "findClassMethods

*---------------------------------------------------------------------------------------------------
* findMethodKey... find the unique key which identifes this method
*---------------------------------------------------------------------------------------------------
form findMethodKey using value(className)
                         value(methodName)
                              methodKey.

data: methodID type seocpdkey.
data: locMethodKey type program.

  methodID-clsname = className.
  methodID-cpdName = methodName.

  cl_oo_classname_service=>get_method_include( exporting mtdkey = methodID
                                       receiving result = locMethodKey
                                       exceptions class_not_existing = 1
                                                  method_not_existing = 2 ).

  methodKey = locMethodKey.
endform.                                                    "findMethodKey
```

```
*---------------------------------------------------------------------------------------------------
* scanForMessages... Search each program for messages
*---------------------------------------------------------------------------------------------------
form scanForMessages using value(programName)
                        value(mainMessageClass)
                            iLocMessages like iMessages[].

data: iIncludeLines type standard table of string with header line.
data: iTokens type standard table of stokes with header line.
data: iStatements type standard table of sstmnt with header line.
data: iKeywords type standard table of text20 with header line.
data: waMessage type tMessage.
data: waMessageComparison type tMessage.
data: watokens type stokes.
data: nextLine type i.
data: stringLength type i value 0.
data: workingOnMessage type i value FALSE.
data: castProgramName type program.

* Read the program code from the textpool.
  castProgramName = programName.
  read report castProgramName into iIncludeLines.

  append MESSAGE to iKeywords.
  scan abap-source iIncludeLines tokens into iTokens with includes statements into iStatements keywords from i
Keywords.

  clear iIncludeLines[].

  loop at iTokens.
    if iTokens-str = MESSAGE.
      workingOnMessage = TRUE.
      continue.
    endif.

    if workingOnMessage = TRUE.
      stringLength = strlen( iTokens-str ).
```

```
*     Message declaration 1
      if stringLength = 4 and iTokens-str+0(1) ca sy-abcde.
        waMessage-msgnr = iTokens-str+1(3).
        waMessage-arbgb = mainMessageClass.
      else.
        if iTokens-str cs "" or iTokens-str cs '`'.
*         Message declaration 2
          translate iTokens-str using '" '.
          translate iTokens-str using '` '.
          condense iTokens-str.
          shift iTokens-str left deleting leading space.
          waMessage-text = iTokens-str.
          waMessage-arbgb = 'Hard coded'.
        else.
          if iTokens-str = 'ID'.
*           Message declaration 3
            nextLine = sy-tabix + 1.
            read table iTokens index nextLine into waTokens.
            translate waTokens-str using '" '.
            condense iTokens-str.
            shift waTokens-str left deleting leading space.
            if not waTokens-str = 'SY-MSGID'.
              waMessage-arbgb = waTokens-str.

              nextLine = nextLine + 4.
              read table iTokens index nextLine into waTokens.
              translate waTokens-str using '" '.
              condense waTokens-str.
              shift waTokens-str left deleting leading space.
              waMessage-msgnr = waTokens-str.
            else.
              workingOnMessage = FALSE.
            endif.
          else.
            if stringLength >= 5 and iTokens-str+4(1) = '('.
*             Message declaration 4
              waMessage-msgnr = iTokens-str+1(3).
```

```
            shift iTokens-str left up to '('.
            replace '(' into iTokens-str with space.
            replace ')' into iTokens-str with space.
            condense iTokens-str.
            waMessage-arbgb = iTokens-str.
          endif.
        endif.
      endif.
    endif.

*     find the message text
      if not waMessage-arbgb is initial and not waMessage-msgnr is initial and waMessage-text is initial.
        select single text
                 from t100
                 into waMessage-text
                 where sprsl = sy-langu
                   and arbgb = waMessage-arbgb
                   and msgnr = waMessage-msgnr.
      endif.

*     Append the message
      if not waMessage is initial.
*       Don't append the message if we already have it listed
        read table iLocMessages with key arbgb = waMessage-arbgb
                                 msgnr = waMessage-msgnr
                                 into waMessageComparison.
        if sy-subrc <> 0.
          append waMessage to iLocMessages.
        endif.
        clear waMessage.
        workingOnMessage = FALSE.
      endif.
    endif.
  endloop.
endform.                                                    "scanForMessages
```