# ABAP Code

## Global Declaration

TYPE-POOLS: lvc.  
  
" Selection Screen  
SELECT-OPTIONS:  
 s\_matnr FOR mara-matnr,  
 s\_werks FOR ekpo-werks OBLIGATORY,  
 s\_lifnr FOR ekko-lifnr,  
 s\_mtart FOR mara-mtart.  
  
PARAMETERS:  
 p\_layout TYPE disvariant-variant,  
 p\_save TYPE char1 DEFAULT 'A'.  
  
" Types  
TYPES: BEGIN OF ty\_key,  
 ebeln TYPE ekpo-ebeln,  
 ebelp TYPE ekpo-ebelp,  
 END OF ty\_key.  
  
TYPES: BEGIN OF ty\_out,  
 sel TYPE boole\_d, " Checkbox for bulk close  
 ebeln TYPE ekpo-ebeln,  
 ebelp TYPE ekpo-ebelp,  
 lifnr TYPE ekko-lifnr,  
 name1 TYPE lfa1-name1,  
 matnr TYPE ekpo-matnr,  
 mtart TYPE mara-mtart,  
 txz01 TYPE ekpo-txz01,  
 werks TYPE ekpo-werks,  
 lgort TYPE ekpo-lgort,  
 menge TYPE ekpo-menge,  
 meins TYPE ekpo-meins,  
 eindt TYPE eket-eindt,  
 elikz TYPE ekpo-elikz,  
 END OF ty\_out.  
  
" Data: Internal Tables and Work Areas  
DATA:  
 gt\_out TYPE STANDARD TABLE OF ty\_out WITH DEFAULT KEY,  
 gs\_out TYPE ty\_out,  
 gt\_to\_close TYPE SORTED TABLE OF ty\_key WITH UNIQUE KEY ebeln ebelp.  
  
DATA:  
 gt\_ekko TYPE TABLE OF ekko,  
 gt\_ekpo TYPE TABLE OF ekpo,  
 gt\_eket TYPE TABLE OF eket,  
 gt\_mara TYPE TABLE OF mara,  
 gt\_lfa1 TYPE TABLE OF lfa1.  
  
" ALV Objects and Structures  
DATA:  
 go\_container TYPE REF TO cl\_gui\_custom\_container,  
 go\_alv TYPE REF TO cl\_gui\_alv\_grid,  
 go\_parent TYPE REF TO cl\_gui\_container.  
  
DATA:  
 gt\_fieldcat TYPE lvc\_t\_fcat,  
 gs\_layout TYPE lvc\_s\_layo,  
 gt\_sort TYPE lvc\_t\_sort,  
 gt\_filter TYPE lvc\_t\_filt,  
 gs\_variant TYPE disvariant,  
 gv\_save TYPE char1.  
  
" Event Handler  
CLASS lcl\_event\_handler DEFINITION FINAL.  
 PUBLIC SECTION.  
 METHODS handle\_user\_command  
 FOR EVENT user\_command OF cl\_gui\_alv\_grid  
 IMPORTING e\_ucomm.  
 METHODS handle\_toolbar  
 FOR EVENT toolbar OF cl\_gui\_alv\_grid  
 IMPORTING e\_object e\_interactive.  
 METHODS handle\_hotspot\_click  
 FOR EVENT hotspot\_click OF cl\_gui\_alv\_grid  
 IMPORTING e\_row\_id e\_column\_id.  
 METHODS handle\_double\_click  
 FOR EVENT double\_click OF cl\_gui\_alv\_grid  
 IMPORTING e\_row e\_column es\_row\_no.  
 METHODS handle\_data\_changed  
 FOR EVENT data\_changed\_finished OF cl\_gui\_alv\_grid  
 IMPORTING e\_modified et\_good\_cells.  
ENDCLASS.  
  
DATA go\_events TYPE REF TO lcl\_event\_handler.  
  
" General  
DATA:  
 gv\_okcode TYPE syucomm,  
 gv\_repid TYPE syrepid.  
  
" Constants  
CONSTANTS:  
 c\_fcode\_close TYPE syucomm VALUE 'CLOSE\_PO',  
 c\_fcode\_refresh TYPE syucomm VALUE 'REFRESH',  
 c\_true TYPE boole\_d VALUE 'X',  
 c\_false TYPE boole\_d VALUE ''.

## Selection Screen

SELECTION-SCREEN BEGIN OF BLOCK b1 WITH FRAME TITLE text-t01.  
SELECT-OPTIONS:  
 s\_matnr FOR mara-matnr,  
 s\_werks FOR ekpo-werks OBLIGATORY,  
 s\_lifnr FOR ekko-lifnr,  
 s\_mtart FOR mara-mtart.  
SELECTION-SCREEN END OF BLOCK b1.  
  
SELECTION-SCREEN BEGIN OF BLOCK b2 WITH FRAME TITLE text-t02.  
PARAMETERS:  
 p\_layout TYPE disvariant-variant,  
 p\_save TYPE char1 DEFAULT 'A'.  
SELECTION-SCREEN END OF BLOCK b2.

## Processing Logic

INITIALIZATION.  
 gv\_repid = sy-repid.  
 gv\_save = p\_save.  
 gs\_variant-report = gv\_repid.  
 gs\_variant-variant = p\_layout.  
  
START-OF-SELECTION.  
 PERFORM frm\_get\_data.  
  
END-OF-SELECTION.  
 CALL SCREEN 0100.  
  
CLASS lcl\_event\_handler IMPLEMENTATION.  
 METHOD handle\_toolbar.  
 DATA: ls\_button TYPE stb\_button.  
  
 CLEAR ls\_button.  
 ls\_button-function = c\_fcode\_close.  
 ls\_button-icon = icon\_system\_save.  
 ls\_button-quickinfo = 'Close Selected PO Items'.  
 ls\_button-text = 'Close Items'.  
 ls\_button-disabled = abap\_false.  
 APPEND ls\_button TO e\_object->mt\_toolbar.  
  
 CLEAR ls\_button.  
 ls\_button-butn\_type = 3. " Separator  
 APPEND ls\_button TO e\_object->mt\_toolbar.  
  
 CLEAR ls\_button.  
 ls\_button-function = c\_fcode\_refresh.  
 ls\_button-icon = icon\_refresh.  
 ls\_button-quickinfo = 'Refresh Data'.  
 ls\_button-text = 'Refresh'.  
 ls\_button-disabled = abap\_false.  
 APPEND ls\_button TO e\_object->mt\_toolbar.  
 ENDMETHOD.  
  
 METHOD handle\_user\_command.  
 CASE e\_ucomm.  
 WHEN c\_fcode\_close.  
 IF go\_alv IS BOUND.  
 go\_alv->check\_changed\_data( ).  
 ENDIF.  
 PERFORM frm\_close\_selected.  
 WHEN c\_fcode\_refresh.  
 IF go\_alv IS BOUND.  
 go\_alv->check\_changed\_data( ).  
 ENDIF.  
 PERFORM frm\_get\_data.  
 PERFORM frm\_refresh\_alv.  
 WHEN OTHERS.  
 ENDCASE.  
 ENDMETHOD.  
  
 METHOD handle\_hotspot\_click.  
 " Not used  
 ENDMETHOD.  
  
 METHOD handle\_double\_click.  
 " Not used  
 ENDMETHOD.  
  
 METHOD handle\_data\_changed.  
 " Not used, we explicitly call CHECK\_CHANGED\_DATA before actions  
 ENDMETHOD.  
ENDCLASS.  
  
MODULE status\_0100 OUTPUT.  
 SET TITLEBAR 'TIT\_0100'.  
  
 IF go\_alv IS INITIAL.  
 PERFORM frm\_build\_fieldcat.  
 PERFORM frm\_build\_layout.  
 PERFORM frm\_build\_sort.  
  
 CREATE OBJECT go\_container  
 EXPORTING  
 container\_name = 'CC\_ALV'.  
  
 CREATE OBJECT go\_alv  
 EXPORTING  
 i\_parent = go\_container.  
  
 CREATE OBJECT go\_events.  
 SET HANDLER go\_events->handle\_toolbar FOR go\_alv.  
 SET HANDLER go\_events->handle\_user\_command FOR go\_alv.  
 SET HANDLER go\_events->handle\_hotspot\_click FOR go\_alv.  
 SET HANDLER go\_events->handle\_double\_click FOR go\_alv.  
 SET HANDLER go\_events->handle\_data\_changed FOR go\_alv.  
  
 PERFORM frm\_display\_alv.  
 ELSE.  
 PERFORM frm\_refresh\_alv.  
 ENDIF.  
ENDMODULE.  
  
MODULE user\_command\_0100 INPUT.  
 gv\_okcode = sy-ucomm.  
 CLEAR sy-ucomm.  
  
 CASE gv\_okcode.  
 WHEN 'BACK' OR 'CANCEL' OR 'EXIT'.  
 LEAVE TO SCREEN 0.  
 WHEN c\_fcode\_close.  
 PERFORM frm\_close\_selected.  
 WHEN c\_fcode\_refresh.  
 PERFORM frm\_get\_data.  
 PERFORM frm\_refresh\_alv.  
 WHEN OTHERS.  
 ENDCASE.  
ENDMODULE.  
  
FORM frm\_get\_data.  
 DATA: lt\_matnr\_rng TYPE RANGE OF mara-matnr,  
 lt\_ebeln\_rng TYPE RANGE OF ekko-ebeln,  
 ls\_rng LIKE LINE OF lt\_matnr\_rng,  
 ls\_rng2 LIKE LINE OF lt\_ebeln\_rng.  
  
 CLEAR: gt\_out, gt\_to\_close, gt\_ekpo, gt\_ekko, gt\_mara, gt\_lfa1, gt\_eket.  
  
 " 1) If material type provided, fetch materials  
 IF s\_mtart[] IS NOT INITIAL.  
 SELECT matnr, mtart  
 FROM mara  
 INTO TABLE @gt\_mara  
 WHERE mtart IN @s\_mtart  
 AND matnr IN @s\_matnr. " respects explicit MATNR filter if given  
 SORT gt\_mara BY matnr.  
 DELETE ADJACENT DUPLICATES FROM gt\_mara COMPARING matnr.  
  
 LOOP AT gt\_mara ASSIGNING FIELD-SYMBOL(<ls\_mara>).  
 CLEAR ls\_rng.  
 ls\_rng-sign = 'I'.  
 ls\_rng-option = 'EQ'.  
 ls\_rng-low = <ls\_mara>-matnr.  
 APPEND ls\_rng TO lt\_matnr\_rng.  
 ENDLOOP.  
 ELSEIF s\_matnr[] IS NOT INITIAL.  
 " If MTART not given, use MATNR range directly  
 lt\_matnr\_rng = s\_matnr[].  
 ENDIF.  
  
 " 2) If vendor provided, fetch EBELN list from EKKO  
 IF s\_lifnr[] IS NOT INITIAL.  
 SELECT ebeln, lifnr  
 FROM ekko  
 INTO TABLE @gt\_ekko  
 WHERE lifnr IN @s\_lifnr.  
 SORT gt\_ekko BY ebeln.  
 DELETE ADJACENT DUPLICATES FROM gt\_ekko COMPARING ebeln.  
  
 LOOP AT gt\_ekko ASSIGNING FIELD-SYMBOL(<ls\_ekko\_v>).  
 CLEAR ls\_rng2.  
 ls\_rng2-sign = 'I'.  
 ls\_rng2-option = 'EQ'.  
 ls\_rng2-low = <ls\_ekko\_v>-ebeln.  
 APPEND ls\_rng2 TO lt\_ebeln\_rng.  
 ENDLOOP.  
 ENDIF.  
  
 " 3) Fetch open EKPO items based on filters (ELIKZ = space)  
 SELECT ebeln, ebelp, matnr, txz01, werks, lgort, menge, meins, elikz  
 FROM ekpo  
 INTO TABLE @gt\_ekpo  
 WHERE elikz = @c\_false  
 AND werks IN @s\_werks  
 AND ( ( @lt\_matnr\_rng IS INITIAL AND matnr IN @s\_matnr ) OR  
 ( @lt\_matnr\_rng IS NOT INITIAL AND matnr IN @lt\_matnr\_rng ) OR  
 ( @lt\_matnr\_rng IS INITIAL AND @s\_matnr[] IS INITIAL ) )  
 AND ( @lt\_ebeln\_rng IS INITIAL OR ebeln IN @lt\_ebeln\_rng ).  
  
 IF gt\_ekpo IS INITIAL.  
 RETURN.  
 ENDIF.  
  
 " Restrict related headers (EKKO) to the selected EBELN if not already fetched  
 DATA: lt\_ebeln TYPE SORTED TABLE OF ekko-ebeln WITH UNIQUE KEY table\_line.  
 lt\_ebeln = VALUE #( FOR wa IN gt\_ekpo ( wa-ebeln ) ).  
  
 IF gt\_ekko IS INITIAL.  
 SELECT ebeln, lifnr  
 FROM ekko  
 INTO TABLE @gt\_ekko  
 FOR ALL ENTRIES IN @gt\_ekpo  
 WHERE ebeln = @gt\_ekpo-ebeln.  
 ELSE.  
 " Keep only relevant headers  
 DELETE gt\_ekko WHERE ebeln NOT IN lt\_ebeln.  
 ENDIF.  
 SORT gt\_ekko BY ebeln.  
  
 " Fetch vendor names (LFA1)  
 DATA: lt\_lifnr TYPE SORTED TABLE OF lfa1-lifnr WITH UNIQUE KEY table\_line.  
 lt\_lifnr = VALUE #( FOR wa IN gt\_ekko ( wa-lifnr ) ).  
  
 IF lt\_lifnr IS NOT INITIAL.  
 SELECT lifnr, name1  
 FROM lfa1  
 INTO TABLE @gt\_lfa1  
 FOR ALL ENTRIES IN @gt\_ekko  
 WHERE lifnr = @gt\_ekko-lifnr.  
 SORT gt\_lfa1 BY lifnr.  
 ENDIF.  
  
 " Ensure material master for MTART if not fetched yet  
 IF gt\_mara IS INITIAL.  
 DATA(lt\_matnr\_all) = VALUE SORTED TABLE OF mara-matnr WITH UNIQUE KEY table\_line(  
 FOR wa IN gt\_ekpo ( wa-matnr ) ).  
 IF lt\_matnr\_all IS NOT INITIAL.  
 SELECT matnr, mtart  
 FROM mara  
 INTO TABLE @gt\_mara  
 FOR ALL ENTRIES IN @gt\_ekpo  
 WHERE matnr = @gt\_ekpo-matnr.  
 SORT gt\_mara BY matnr.  
 ENDIF.  
 ELSE.  
 " Keep only relevant materials  
 DELETE gt\_mara WHERE matnr NOT IN VALUE #( FOR wa IN gt\_ekpo ( wa-matnr ) ).  
 ENDIF.  
  
 " Fetch earliest schedule date from EKET per item  
 CLEAR gt\_eket.  
 SELECT ebeln, ebelp, MIN( eindt ) AS eindt  
 FROM eket  
 INTO TABLE @gt\_eket  
 FOR ALL ENTRIES IN @gt\_ekpo  
 WHERE ebeln = @gt\_ekpo-ebeln  
 AND ebelp = @gt\_ekpo-ebelp  
 GROUP BY ebeln, ebelp.  
 SORT gt\_eket BY ebeln ebelp.  
  
 " Build output  
 LOOP AT gt\_ekpo ASSIGNING FIELD-SYMBOL(<ls\_ekpo>).  
 CLEAR gs\_out.  
 gs\_out-sel = c\_false.  
 gs\_out-ebeln = <ls\_ekpo>-ebeln.  
 gs\_out-ebelp = <ls\_ekpo>-ebelp.  
 gs\_out-matnr = <ls\_ekpo>-matnr.  
 gs\_out-txz01 = <ls\_ekpo>-txz01.  
 gs\_out-werks = <ls\_ekpo>-werks.  
 gs\_out-lgort = <ls\_ekpo>-lgort.  
 gs\_out-menge = <ls\_ekpo>-menge.  
 gs\_out-meins = <ls\_ekpo>-meins.  
 gs\_out-elikz = <ls\_ekpo>-elikz.  
  
 READ TABLE gt\_ekko ASSIGNING FIELD-SYMBOL(<ls\_ekko>) WITH KEY ebeln = <ls\_ekpo>-ebeln BINARY SEARCH.  
 IF sy-subrc = 0.  
 gs\_out-lifnr = <ls\_ekko>-lifnr.  
 READ TABLE gt\_lfa1 ASSIGNING FIELD-SYMBOL(<ls\_lfa1>) WITH KEY lifnr = <ls\_ekko>-lifnr BINARY SEARCH.  
 IF sy-subrc = 0.  
 gs\_out-name1 = <ls\_lfa1>-name1.  
 ENDIF.  
 ENDIF.  
  
 READ TABLE gt\_mara ASSIGNING FIELD-SYMBOL(<ls\_mara2>) WITH KEY matnr = <ls\_ekpo>-matnr BINARY SEARCH.  
 IF sy-subrc = 0.  
 gs\_out-mtart = <ls\_mara2>-mtart.  
 ENDIF.  
  
 READ TABLE gt\_eket ASSIGNING FIELD-SYMBOL(<ls\_eket>) WITH KEY ebeln = <ls\_ekpo>-ebeln ebelp = <ls\_ekpo>-ebelp BINARY SEARCH.  
 IF sy-subrc = 0.  
 gs\_out-eindt = <ls\_eket>-eindt.  
 ENDIF.  
  
 APPEND gs\_out TO gt\_out.  
 ENDLOOP.  
ENDFORM.  
  
FORM frm\_build\_fieldcat.  
 CLEAR gt\_fieldcat.  
  
 DATA ls\_fcat TYPE lvc\_s\_fcat.  
  
 DEFINE add\_fcat.  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = &1.  
 ls\_fcat-coltext = &2.  
 ls\_fcat-scrtext\_l = &2.  
 ls\_fcat-scrtext\_m = &2.  
 ls\_fcat-scrtext\_s = &2.  
 ls\_fcat-key = &3.  
 ls\_fcat-edit = &4.  
 ls\_fcat-checkbox = &5.  
 ls\_fcat-hotspot = &6.  
 ls\_fcat-outputlen = &7.  
 APPEND ls\_fcat TO gt\_fieldcat.  
 END-OF-DEFINITION.  
  
 add\_fcat 'SEL' 'Sel.' '' 'X' 'X' '' 3.  
 add\_fcat 'EBELN' 'PO' 'X' '' '' 'X' 10.  
 add\_fcat 'EBELP' 'Item' 'X' '' '' '' 5.  
 add\_fcat 'LIFNR' 'Vendor' '' '' '' '' 10.  
 add\_fcat 'NAME1' 'Vendor Name' '' '' '' '' 25.  
 add\_fcat 'MATNR' 'Material' '' '' '' '' 18.  
 add\_fcat 'MTART' 'Mat. Type' '' '' '' '' 10.  
 add\_fcat 'TXZ01' 'Short Text' '' '' '' '' 30.  
 add\_fcat 'WERKS' 'Plant' '' '' '' '' 4.  
 add\_fcat 'LGORT' 'SLoc' '' '' '' '' 4.  
 add\_fcat 'MENGE' 'Quantity' '' '' '' '' 13.  
 add\_fcat 'MEINS' 'UoM' '' '' '' '' 3.  
 add\_fcat 'EINDT' 'Deliv. Date' '' '' '' '' 10.  
 add\_fcat 'ELIKZ' 'Delivery Completed' '' '' '' '' 1.  
ENDFORM.  
  
FORM frm\_build\_layout.  
 CLEAR gs\_layout.  
 gs\_layout-zebra = abap\_true.  
 gs\_layout-cwidth\_opt = abap\_true.  
 gs\_layout-sel\_mode = 'A'. " multiple row selection  
ENDFORM.  
  
FORM frm\_build\_sort.  
 CLEAR gt\_sort.  
 DATA ls\_sort TYPE lvc\_s\_sort.  
  
 CLEAR ls\_sort.  
 ls\_sort-fieldname = 'EBELN'.  
 ls\_sort-up = abap\_true.  
 ls\_sort-subtot = abap\_false.  
 APPEND ls\_sort TO gt\_sort.  
  
 CLEAR ls\_sort.  
 ls\_sort-fieldname = 'EBELP'.  
 ls\_sort-up = abap\_true.  
 ls\_sort-subtot = abap\_false.  
 APPEND ls\_sort TO gt\_sort.  
ENDFORM.  
  
FORM frm\_display\_alv.  
 IF go\_alv IS NOT BOUND.  
 RETURN.  
 ENDIF.  
  
 DATA: ls\_stbl TYPE lvc\_s\_stbl.  
 ls\_stbl-row = abap\_true.  
 ls\_stbl-col = abap\_true.  
  
 CALL METHOD go\_alv->set\_table\_for\_first\_display  
 EXPORTING  
 is\_layout = gs\_layout  
 i\_save = gv\_save  
 is\_variant = gs\_variant  
 CHANGING  
 it\_outtab = gt\_out  
 it\_fieldcatalog = gt\_fieldcat  
 it\_sort = gt\_sort  
 it\_filter = gt\_filter.  
  
 CALL METHOD go\_alv->set\_ready\_for\_input  
 EXPORTING  
 i\_ready\_for\_input = 1.  
ENDFORM.  
  
FORM frm\_refresh\_alv.  
 IF go\_alv IS BOUND.  
 DATA ls\_stbl TYPE lvc\_s\_stbl.  
 ls\_stbl-row = abap\_true.  
 ls\_stbl-col = abap\_true.  
 go\_alv->refresh\_table\_display( EXPORTING is\_stable = ls\_stbl ).  
 ENDIF.  
ENDFORM.  
  
FORM frm\_close\_selected.  
 DATA: lt\_sel TYPE STANDARD TABLE OF ty\_out WITH DEFAULT KEY,  
 ls\_key TYPE ty\_key,  
 lv\_cnt TYPE i.  
  
 " Make sure latest checkbox changes are captured  
 IF go\_alv IS BOUND.  
 go\_alv->check\_changed\_data( ).  
 ENDIF.  
  
 LOOP AT gt\_out ASSIGNING FIELD-SYMBOL(<ls\_out>) WHERE sel = c\_true.  
 APPEND <ls\_out> TO lt\_sel.  
 ENDLOOP.  
  
 IF lt\_sel IS INITIAL.  
 MESSAGE 'No items selected to close.' TYPE 'S' DISPLAY LIKE 'E'.  
 RETURN.  
 ENDIF.  
  
 " Build keys to close  
 CLEAR gt\_to\_close.  
 LOOP AT lt\_sel ASSIGNING FIELD-SYMBOL(<ls\_sel>).  
 CLEAR ls\_key.  
 ls\_key-ebeln = <ls\_sel>-ebeln.  
 ls\_key-ebelp = <ls\_sel>-ebelp.  
 INSERT ls\_key INTO TABLE gt\_to\_close.  
 ENDLOOP.  
  
 " Update EKPO-ELIKZ = 'X' for selected items  
 lv\_cnt = 0.  
 LOOP AT gt\_to\_close ASSIGNING FIELD-SYMBOL(<ls\_key2>).  
 UPDATE ekpo  
 SET elikz = @c\_true  
 WHERE ebeln = @<ls\_key2>-ebeln  
 AND ebelp = @<ls\_key2>-ebelp  
 AND elikz = @c\_false.  
 IF sy-subrc = 0.  
 lv\_cnt = lv\_cnt + 1.  
 ENDIF.  
 ENDLOOP.  
  
 IF lv\_cnt > 0.  
 COMMIT WORK AND WAIT.  
 MESSAGE |Closed { lv\_cnt } item(s).| TYPE 'S'.  
 ELSE.  
 ROLLBACK WORK.  
 MESSAGE 'Nothing was updated.' TYPE 'S'.  
 ENDIF.  
  
 " Refresh dataset to show only open items  
 PERFORM frm\_get\_data.  
 PERFORM frm\_refresh\_alv.  
ENDFORM.

## Output Display

CLASS lcl\_event\_handler DEFINITION FINAL.  
 PUBLIC SECTION.  
 METHODS handle\_user\_command  
 FOR EVENT user\_command OF cl\_gui\_alv\_grid  
 IMPORTING e\_ucomm.  
 METHODS handle\_toolbar  
 FOR EVENT toolbar OF cl\_gui\_alv\_grid  
 IMPORTING e\_object e\_interactive.  
 METHODS handle\_hotspot\_click  
 FOR EVENT hotspot\_click OF cl\_gui\_alv\_grid  
 IMPORTING e\_row\_id e\_column\_id.  
 METHODS handle\_double\_click  
 FOR EVENT double\_click OF cl\_gui\_alv\_grid  
 IMPORTING e\_row e\_column es\_row\_no.  
 METHODS handle\_data\_changed  
 FOR EVENT data\_changed\_finished OF cl\_gui\_alv\_grid  
 IMPORTING e\_modified et\_good\_cells.  
ENDCLASS.  
  
CLASS lcl\_event\_handler IMPLEMENTATION.  
 METHOD handle\_toolbar.  
 DATA: ls\_button TYPE stb\_button.  
  
 CLEAR ls\_button.  
 ls\_button-function = c\_fcode\_close.  
 ls\_button-icon = icon\_system\_save.  
 ls\_button-quickinfo = 'Close Selected PO Items'.  
 ls\_button-text = 'Close Items'.  
 ls\_button-disabled = abap\_false.  
 APPEND ls\_button TO e\_object->mt\_toolbar.  
  
 CLEAR ls\_button.  
 ls\_button-butn\_type = 3.  
 APPEND ls\_button TO e\_object->mt\_toolbar.  
  
 CLEAR ls\_button.  
 ls\_button-function = c\_fcode\_refresh.  
 ls\_button-icon = icon\_refresh.  
 ls\_button-quickinfo = 'Refresh Data'.  
 ls\_button-text = 'Refresh'.  
 ls\_button-disabled = abap\_false.  
 APPEND ls\_button TO e\_object->mt\_toolbar.  
 ENDMETHOD.  
  
 METHOD handle\_user\_command.  
 CASE e\_ucomm.  
 WHEN c\_fcode\_close.  
 IF go\_alv IS BOUND.  
 go\_alv->check\_changed\_data( ).  
 ENDIF.  
 PERFORM frm\_close\_selected.  
 WHEN c\_fcode\_refresh.  
 IF go\_alv IS BOUND.  
 go\_alv->check\_changed\_data( ).  
 ENDIF.  
 PERFORM frm\_get\_data.  
 PERFORM frm\_refresh\_alv.  
 WHEN OTHERS.  
 ENDCASE.  
 ENDMETHOD.  
  
 METHOD handle\_hotspot\_click.  
 ENDMETHOD.  
  
 METHOD handle\_double\_click.  
 ENDMETHOD.  
  
 METHOD handle\_data\_changed.  
 ENDMETHOD.  
ENDCLASS.  
  
MODULE status\_0100 OUTPUT.  
 SET TITLEBAR 'TIT\_0100'.  
  
 IF go\_alv IS INITIAL.  
 PERFORM frm\_build\_fieldcat.  
 PERFORM frm\_build\_layout.  
 PERFORM frm\_build\_sort.  
  
 CREATE OBJECT go\_container  
 EXPORTING  
 container\_name = 'CC\_ALV'.  
  
 CREATE OBJECT go\_alv  
 EXPORTING  
 i\_parent = go\_container.  
  
 CREATE OBJECT go\_events.  
 SET HANDLER go\_events->handle\_toolbar FOR go\_alv.  
 SET HANDLER go\_events->handle\_user\_command FOR go\_alv.  
 SET HANDLER go\_events->handle\_hotspot\_click FOR go\_alv.  
 SET HANDLER go\_events->handle\_double\_click FOR go\_alv.  
 SET HANDLER go\_events->handle\_data\_changed FOR go\_alv.  
  
 PERFORM frm\_display\_alv.  
 ELSE.  
 PERFORM frm\_refresh\_alv.  
 ENDIF.  
ENDMODULE.  
  
MODULE user\_command\_0100 INPUT.  
 gv\_okcode = sy-ucomm.  
 CLEAR sy-ucomm.  
  
 CASE gv\_okcode.  
 WHEN 'BACK' OR 'CANCEL' OR 'EXIT'.  
 LEAVE TO SCREEN 0.  
 WHEN c\_fcode\_close.  
 PERFORM frm\_close\_selected.  
 WHEN c\_fcode\_refresh.  
 PERFORM frm\_get\_data.  
 PERFORM frm\_refresh\_alv.  
 WHEN OTHERS.  
 ENDCASE.  
ENDMODULE.  
  
FORM frm\_build\_fieldcat.  
 CLEAR gt\_fieldcat.  
 DATA ls\_fcat TYPE lvc\_s\_fcat.  
  
 " Selection checkbox  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'SEL'.  
 ls\_fcat-coltext = 'Sel.'.  
 ls\_fcat-scrtext\_l = 'Select'.  
 ls\_fcat-checkbox = abap\_true.  
 ls\_fcat-edit = abap\_true.  
 ls\_fcat-outputlen = 3.  
 APPEND ls\_fcat TO gt\_fieldcat.  
  
 " PO  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'EBELN'.  
 ls\_fcat-coltext = 'PO'.  
 ls\_fcat-scrtext\_l = 'PO'.  
 ls\_fcat-key = abap\_true.  
 ls\_fcat-hotspot = abap\_true.  
 ls\_fcat-outputlen = 10.  
 APPEND ls\_fcat TO gt\_fieldcat.  
  
 " Item  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'EBELP'.  
 ls\_fcat-coltext = 'Item'.  
 ls\_fcat-scrtext\_l = 'Item'.  
 ls\_fcat-key = abap\_true.  
 ls\_fcat-outputlen = 5.  
 APPEND ls\_fcat TO gt\_fieldcat.  
  
 " Vendor  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'LIFNR'.  
 ls\_fcat-coltext = 'Vendor'.  
 ls\_fcat-scrtext\_l = 'Vendor'.  
 ls\_fcat-outputlen = 10.  
 APPEND ls\_fcat TO gt\_fieldcat.  
  
 " Vendor Name  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'NAME1'.  
 ls\_fcat-coltext = 'Vendor Name'.  
 ls\_fcat-scrtext\_l = 'Vendor Name'.  
 ls\_fcat-outputlen = 25.  
 APPEND ls\_fcat TO gt\_fieldcat.  
  
 " Material  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'MATNR'.  
 ls\_fcat-coltext = 'Material'.  
 ls\_fcat-scrtext\_l = 'Material'.  
 ls\_fcat-outputlen = 18.  
 APPEND ls\_fcat TO gt\_fieldcat.  
  
 " Material Type  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'MTART'.  
 ls\_fcat-coltext = 'Mat. Type'.  
 ls\_fcat-scrtext\_l = 'Material Type'.  
 ls\_fcat-outputlen = 10.  
 APPEND ls\_fcat TO gt\_fieldcat.  
  
 " Short Text  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'TXZ01'.  
 ls\_fcat-coltext = 'Short Text'.  
 ls\_fcat-scrtext\_l = 'Short Text'.  
 ls\_fcat-outputlen = 30.  
 APPEND ls\_fcat TO gt\_fieldcat.  
  
 " Plant  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'WERKS'.  
 ls\_fcat-coltext = 'Plant'.  
 ls\_fcat-scrtext\_l = 'Plant'.  
 ls\_fcat-outputlen = 4.  
 APPEND ls\_fcat TO gt\_fieldcat.  
  
 " Storage Location  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'LGORT'.  
 ls\_fcat-coltext = 'SLoc'.  
 ls\_fcat-scrtext\_l = 'Storage Loc'.  
 ls\_fcat-outputlen = 4.  
 APPEND ls\_fcat TO gt\_fieldcat.  
  
 " Quantity (summable)  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'MENGE'.  
 ls\_fcat-coltext = 'Quantity'.  
 ls\_fcat-scrtext\_l = 'Quantity'.  
 ls\_fcat-outputlen = 13.  
 ls\_fcat-do\_sum = abap\_true.  
 APPEND ls\_fcat TO gt\_fieldcat.  
  
 " UoM  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'MEINS'.  
 ls\_fcat-coltext = 'UoM'.  
 ls\_fcat-scrtext\_l = 'UoM'.  
 ls\_fcat-outputlen = 3.  
 APPEND ls\_fcat TO gt\_fieldcat.  
  
 " Delivery Date  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'EINDT'.  
 ls\_fcat-coltext = 'Deliv. Date'.  
 ls\_fcat-scrtext\_l = 'Delivery Date'.  
 ls\_fcat-outputlen = 10.  
 APPEND ls\_fcat TO gt\_fieldcat.  
  
 " Delivery Completed  
 CLEAR ls\_fcat.  
 ls\_fcat-fieldname = 'ELIKZ'.  
 ls\_fcat-coltext = 'Delivery Completed'.  
 ls\_fcat-scrtext\_l = 'Deliv. Completed'.  
 ls\_fcat-outputlen = 1.  
 APPEND ls\_fcat TO gt\_fieldcat.  
ENDFORM.  
  
FORM frm\_build\_layout.  
 CLEAR gs\_layout.  
 gs\_layout-zebra = abap\_true.  
 gs\_layout-cwidth\_opt = abap\_true.  
 gs\_layout-sel\_mode = 'A'.  
 gs\_layout-totals\_bef = abap\_true.  
ENDFORM.  
  
FORM frm\_build\_sort.  
 CLEAR gt\_sort.  
 DATA ls\_sort TYPE lvc\_s\_sort.  
  
 CLEAR ls\_sort.  
 ls\_sort-fieldname = 'EBELN'.  
 ls\_sort-up = abap\_true.  
 ls\_sort-subtot = abap\_true.  
 APPEND ls\_sort TO gt\_sort.  
  
 CLEAR ls\_sort.  
 ls\_sort-fieldname = 'EBELP'.  
 ls\_sort-up = abap\_true.  
 ls\_sort-subtot = abap\_false.  
 APPEND ls\_sort TO gt\_sort.  
ENDFORM.  
  
FORM frm\_display\_alv.  
 IF go\_alv IS NOT BOUND.  
 RETURN.  
 ENDIF.  
  
 CALL METHOD go\_alv->set\_table\_for\_first\_display  
 EXPORTING  
 is\_layout = gs\_layout  
 i\_save = gv\_save  
 is\_variant = gs\_variant  
 CHANGING  
 it\_outtab = gt\_out  
 it\_fieldcatalog = gt\_fieldcat  
 it\_sort = gt\_sort  
 it\_filter = gt\_filter.  
  
 go\_alv->set\_ready\_for\_input( i\_ready\_for\_input = 1 ).  
ENDFORM.  
  
FORM frm\_refresh\_alv.  
 IF go\_alv IS BOUND.  
 DATA ls\_stbl TYPE lvc\_s\_stbl.  
 ls\_stbl-row = abap\_true.  
 ls\_stbl-col = abap\_true.  
 go\_alv->refresh\_table\_display( EXPORTING is\_stable = ls\_stbl ).  
 ENDIF.  
ENDFORM.  
  
FORM frm\_close\_selected.  
 DATA: lt\_sel TYPE STANDARD TABLE OF ty\_out WITH DEFAULT KEY,  
 ls\_key TYPE ty\_key,  
 lv\_cnt TYPE i.  
  
 IF go\_alv IS BOUND.  
 go\_alv->check\_changed\_data( ).  
 ENDIF.  
  
 LOOP AT gt\_out ASSIGNING FIELD-SYMBOL(<ls\_out>) WHERE sel = c\_true.  
 APPEND <ls\_out> TO lt\_sel.  
 ENDLOOP.  
  
 IF lt\_sel IS INITIAL.  
 MESSAGE 'No items selected to close.' TYPE 'S' DISPLAY LIKE 'E'.  
 RETURN.  
 ENDIF.  
  
 CLEAR gt\_to\_close.  
 LOOP AT lt\_sel ASSIGNING FIELD-SYMBOL(<ls\_sel>).  
 CLEAR ls\_key.  
 ls\_key-ebeln = <ls\_sel>-ebeln.  
 ls\_key-ebelp = <ls\_sel>-ebelp.  
 INSERT ls\_key INTO TABLE gt\_to\_close.  
 ENDLOOP.  
  
 lv\_cnt = 0.  
 LOOP AT gt\_to\_close ASSIGNING FIELD-SYMBOL(<ls\_key2>).  
 UPDATE ekpo  
 SET elikz = @c\_true  
 WHERE ebeln = @<ls\_key2>-ebeln  
 AND ebelp = @<ls\_key2>-ebelp  
 AND elikz = @c\_false.  
 IF sy-subrc = 0.  
 lv\_cnt = lv\_cnt + 1.  
 ENDIF.  
 ENDLOOP.  
  
 IF lv\_cnt > 0.  
 COMMIT WORK AND WAIT.  
 MESSAGE |Closed { lv\_cnt } item(s).| TYPE 'S'.  
 ELSE.  
 ROLLBACK WORK.  
 MESSAGE 'Nothing was updated.' TYPE 'S'.  
 ENDIF.  
  
 PERFORM frm\_get\_data.  
 PERFORM frm\_refresh\_alv.  
ENDFORM.