Classic Interactive Report

Interactive program generally interacts with user on the output screen. Let us suppose we have an output for a particular program/report. Now if we want details description by double clicking on a particular field or clicking on a button then it will be called interaction with output. By doing this we can go to the detailed report or secondary output. When we are on the secondary output then we can back to our first out by clicking on Back button. **We can generate 19 secondary list of output. Hence a total of 20 output screen (1 Primary + 19 Secondary) can be generated by Interactive program.**

We have a requirement where we have to prepare a primary output containing Purchase Order, Company Code, Category, Type and Vendor with PO Info. The secondary list will be generated with PO Item, Material, Plant, Storage Location, Material Group, Quantity and Unit of measure. The secondary list will be generated if we double click on PO field in primary output.

We have mentioned the tables EKKO, EKPO and T161T. Now we have declared the structure types of internal tables of it\_ekko, it\_text, it\_out1 and it\_ekpo. Here it\_out1 contains the combination of two internal tables it\_ekko and it\_text and it\_ekpo is for the secondary output. For internal operation we have declared work area of every internal tables.

Next we have mentioned the event INITIALIZATION. Under this we are calling program name, user name and system date of the program. We have declared here selection screen begin with block b1 and under this we are mentioning obligatory select option.

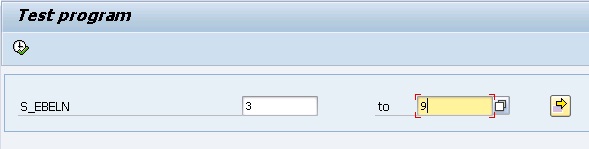
Now under START-OF-SELECTION we are declaring all the operations for primary listing. We have made 4 subroutines. In get\_ekko we select fields from EKKO into internal table it\_ekko. Then in get\_t161t we select fields from t161t into internal table it\_text for all entries in it\_ekko. After that we are preparing the primary output in the basic\_output subroutine. Now we have made another subroutine for displaying the primary output and that is disp\_basic.

Hence the primary list is prepared now and now we have to create the secondary list. To make the secondary list we are calling event AT LINE-SELECTION which raises functionality when user double clicks in any field. Now we are mentioning GET CURSOR for that field and for that value of the field. When these two are matches with Purchase Order (EBELN) field then we are calling two subroutines for secondary list.

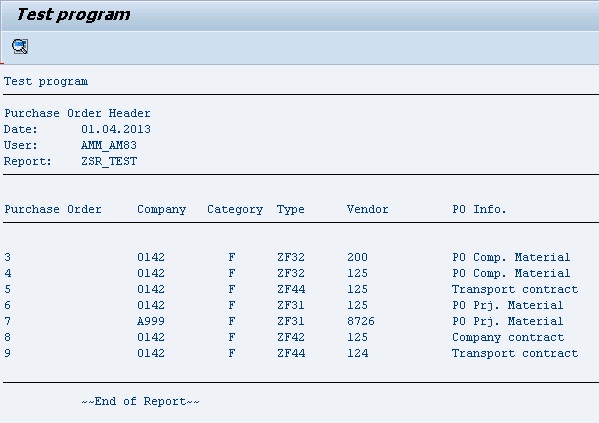
Subroutine get\_ekpo Selects fields from EKPO into table it\_ekpo for all entries in it\_ekko. Then it prepares the secondary output display in the subroutine of ekpo\_output.

We also have raised events TOP-OF-PAGE which shows the header list of primary output and TOP-OF-PAGE DURING LINE-SELECTION which shows the top list of secondary output. Here TOP-OF-PAGE DURING LINE-SELECTION is used only for interactive reports. That means when user double clicks on the field the TOP-OF-PAGE DURING LINE-SELECTION event triggers to make the secondary output header.

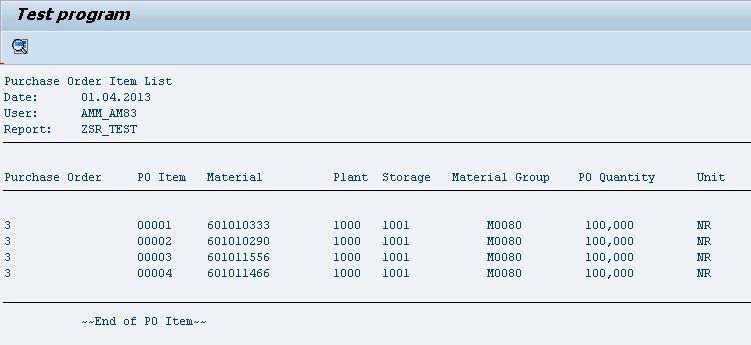
\*&---------------------------------------------------------------------\*  
\*& Report  ZSR\_TEST  
\*&  
\*&---------------------------------------------------------------------\*  
\*&  
\*&  
\*&---------------------------------------------------------------------\*  
  
REPORT  zsr\_test.  
  
TABLES: ekko, ekpo, t161t.  
TYPES: BEGIN OF ty\_ekko,  
        ebeln TYPE ekko-ebeln, "Purchase Order  
        bukrs TYPE ekko-bukrs, "Company Code  
        bstyp TYPE ekko-bstyp, "Category  
        bsart TYPE ekko-bsart, "Type  
        lifnr TYPE ekko-lifnr, "Vendor  
       END OF ty\_ekko,  
  
       BEGIN OF ty\_text,  
        spras TYPE t161t-spras,  
        bsart TYPE t161t-bsart,  
        bstyp TYPE t161t-bstyp,  
        batxt TYPE t161t-batxt, "PO Info  
       END OF ty\_text,  
  
       BEGIN OF ty\_ekpo,  
        ebeln TYPE ekpo-ebeln,  
        ebelp TYPE ekpo-ebelp, "PO Item  
        matnr TYPE ekpo-matnr, "Material  
        werks TYPE ekpo-werks, "Plant  
        lgort TYPE ekpo-lgort, "Storage Location  
        matkl TYPE ekpo-matkl, "Material Group  
        menge TYPE ekpo-menge, "Quantity  
        meins TYPE ekpo-meins, "Unit  
       END OF ty\_ekpo,  
  
       BEGIN OF ty\_out1,  
        ebeln TYPE ekko-ebeln,  
        bukrs TYPE ekko-bukrs,  
        bstyp TYPE ekko-bstyp,  
        bsart TYPE ekko-bsart,  
        lifnr TYPE ekko-lifnr,  
        batxt TYPE t161t-batxt,  
       END OF ty\_out1.  
  
DATA: wa\_ekko TYPE ty\_ekko,  
      it\_ekko TYPE STANDARD TABLE OF ty\_ekko,  
      wa\_text TYPE ty\_text,  
      it\_text TYPE STANDARD TABLE OF ty\_text,  
      wa\_out1 TYPE ty\_out1,  
      it\_out1 TYPE STANDARD TABLE OF ty\_out1,  
      wa\_ekpo TYPE ty\_ekpo,  
      it\_ekpo TYPE STANDARD TABLE OF ty\_ekpo,  
  
      v\_repid TYPE sy-repid,  
      v\_user  TYPE sy-uname,  
      v\_date  TYPE sy-datum,  
  
      v\_field1 TYPE char40,  
      v\_field2 TYPE char40,  
      v\_value1 TYPE char40,  
      v\_value2 TYPE char40.  
  
INITIALIZATION.  
  v\_repid = sy-repid.  
  v\_user  = sy-uname.  
  v\_date  = sy-datum.  
  
  SELECTION-SCREEN BEGIN OF BLOCK b1 WITH FRAME TITLE text-001.  
  SELECT-OPTIONS   s\_ebeln FOR ekko-ebeln OBLIGATORY.  
  SELECTION-SCREEN END OF BLOCK b1.  
  
START-OF-SELECTION.  
  PERFORM get\_ekko.  
  PERFORM get\_t161t.  
  PERFORM basic\_output.  
  PERFORM disp\_basic.  
  
AT LINE-SELECTION.  
  GET CURSOR FIELD v\_field1 VALUE v\_value1.  
  CASE v\_field1.  
    WHEN 'WA\_OUT1-EBELN'.  
      PERFORM get\_ekpo.  
      PERFORM ekpo\_output.  
  ENDCASE.  
  
  GET CURSOR FIELD v\_field2 VALUE v\_value2.  
  CASE v\_field2.  
    WHEN 'WA\_EKPO-MATNR'.  
      PERFORM get\_mara.  
  ENDCASE.  
  
TOP-OF-PAGE.  
  PERFORM top\_page1.  
  
TOP-OF-PAGE DURING LINE-SELECTION.  
  PERFORM top\_page2.  
  
\*&---------------------------------------------------------------------\*  
\*&      Form  get\_ekko  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM get\_ekko .  
  
  SELECT ebeln bukrs bstyp bsart lifnr  
  FROM ekko INTO TABLE it\_ekko  
  WHERE ebeln IN s\_ebeln.  
  
  IF sy-subrc = 0.  
    SORT it\_ekko BY ebeln.  
  ELSE.  
    MESSAGE 'Purchase Order doesn''t exist.' TYPE 'I'.  
    LEAVE LIST-PROCESSING.  
  ENDIF.  
  
ENDFORM.                    " get\_ekko  
\*&---------------------------------------------------------------------\*  
\*&      Form  get\_t161t  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM get\_t161t .  
  
  IF it\_ekko IS NOT INITIAL.  
    SELECT spras bsart bstyp batxt  
    FROM t161t INTO TABLE it\_text  
    FOR ALL ENTRIES IN it\_ekko  
    WHERE spras = sy-langu  
      AND bsart = it\_ekko-bsart  
      AND bstyp = it\_ekko-bstyp.  
  
    IF sy-subrc = 0.  
      SORT it\_text BY bsart bstyp.  
    ENDIF.  
  ENDIF.  
  
ENDFORM.                                                    " get\_t161t  
\*&---------------------------------------------------------------------\*  
\*&      Form  basic\_output  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM basic\_output .  
  
  IF it\_ekko IS NOT INITIAL.  
    LOOP AT it\_ekko INTO wa\_ekko.  
      wa\_out1-ebeln = wa\_ekko-ebeln.  
      wa\_out1-bukrs = wa\_ekko-bukrs.  
      wa\_out1-bstyp = wa\_ekko-bstyp.  
      wa\_out1-bsart = wa\_ekko-bsart.  
      wa\_out1-lifnr = wa\_ekko-lifnr.  
  
      READ TABLE it\_text INTO wa\_text  
      WITH KEY bsart = wa\_ekko-bsart  
               bstyp = wa\_ekko-bstyp BINARY SEARCH.  
      IF sy-subrc = 0.  
        wa\_out1-batxt = wa\_text-batxt.  
      ENDIF.  
  
      APPEND wa\_out1 TO it\_out1.  
      CLEAR: wa\_out1, wa\_ekko, wa\_text.  
    ENDLOOP.  
  ENDIF.  
  
ENDFORM.                    " basic\_output  
\*&---------------------------------------------------------------------\*  
\*&      Form  disp\_basic  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM disp\_basic .  
  
  IF it\_out1 IS NOT INITIAL.  
    LOOP AT it\_out1 INTO wa\_out1.  
      AT FIRST.  
        WRITE: /  'Purchase Order',  
               20 'Company',  
               30 'Category',  
               40 'Type',  
               50 'Vendor',  
               65 'PO Info.'.  
        ULINE.  
        SKIP.  
      ENDAT.  
  
      WRITE: /  wa\_out1-ebeln,  
             20 wa\_out1-bukrs,  
             33 wa\_out1-bstyp,  
             40 wa\_out1-bsart,  
             50 wa\_out1-lifnr,  
             65 wa\_out1-batxt.  
  
      AT LAST.  
        SKIP.  
        ULINE.  
        WRITE: /12 '~~End of Report~~'.  
      ENDAT.  
    ENDLOOP.  
  ENDIF.  
  
ENDFORM.                    " disp\_basic  
\*&---------------------------------------------------------------------\*  
\*&      Form  get\_ekpo  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM get\_ekpo .  
  
  DATA: lv\_ebeln TYPE ekko-ebeln.  
  
  IF v\_value1 IS NOT INITIAL.  
    CALL FUNCTION 'CONVERSION\_EXIT\_ALPHA\_INPUT'  
      EXPORTING  
        input  = v\_value1  
      IMPORTING  
        output = lv\_ebeln.  
  
    IF lv\_ebeln IS NOT INITIAL.  
      SELECT ebeln ebelp matnr werks lgort  
             matkl menge meins  
        FROM ekpo INTO TABLE it\_ekpo  
        WHERE ebeln = lv\_ebeln.  
  
      IF sy-subrc <> 0.  
        MESSAGE 'PO Item doesn''t Exist.' TYPE 'I'.  
        LEAVE LIST-PROCESSING.  
      ENDIF.  
    ENDIF.  
  ENDIF.  
  
ENDFORM.                    " get\_ekpo  
\*&---------------------------------------------------------------------\*  
\*&      Form  ekpo\_output  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM ekpo\_output .  
  
  IF it\_ekpo IS NOT INITIAL.  
    LOOP AT it\_ekpo INTO wa\_ekpo.  
  
      AT FIRST.  
        WRITE: /  'Purchase Order',  
               20 'PO Item',  
               30 'Material',  
               48 'Plant',  
               55 'Storage',  
               65 'Material Group',  
               83 'PO Quantity',  
              100 'Unit'.  
        ULINE.  
        SKIP.  
      ENDAT.  
  
      WRITE: /  wa\_ekpo-ebeln,  
             20 wa\_ekpo-ebelp,  
             30 wa\_ekpo-matnr,  
             48 wa\_ekpo-werks,  
             55 wa\_ekpo-lgort,  
             70 wa\_ekpo-matkl,  
             75 wa\_ekpo-menge,  
            100 wa\_ekpo-meins.  
  
      AT LAST.  
        SKIP.  
        ULINE.  
        WRITE: /12 '~~End of PO Item~~'.  
      ENDAT.  
  
    ENDLOOP.  
  ENDIF.  
  
ENDFORM.                    " ekpo\_output  
\*&---------------------------------------------------------------------\*  
\*&      Form  get\_mara  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM get\_mara .  
  
  DATA: lv\_matnr TYPE mara-matnr.  
  
  IF v\_value2 IS NOT INITIAL.  
    CALL FUNCTION 'CONVERSION\_EXIT\_MATN1\_INPUT'  
      EXPORTING  
        input        = v\_value2  
      IMPORTING  
        output       = lv\_matnr  
      EXCEPTIONS  
        length\_error = 1  
        OTHERS       = 2.  
  
    IF lv\_matnr IS NOT INITIAL.  
      SET PARAMETER ID 'MAT' FIELD lv\_matnr.  
      CALL TRANSACTION 'MM03' AND SKIP FIRST SCREEN.  
    ENDIF.  
  
  ENDIF.  
  
ENDFORM.                    " get\_mara  
\*&---------------------------------------------------------------------\*  
\*&      Form  top\_page1  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM top\_page1 .  
  
  WRITE: / 'Purchase Order Header',  
         / 'Date: ',   12 v\_date DD/MM/YYYY,  
         / 'User: ',   12 v\_user,  
         / 'Report: ', 12 v\_repid.  
  ULINE.  
  SKIP.  
  
ENDFORM.                                                    " top\_page1  
\*&---------------------------------------------------------------------\*  
\*&      Form  top\_page2  
\*&---------------------------------------------------------------------\*  
\*       text  
\*----------------------------------------------------------------------\*  
\*  -->  p1        text  
\*  <--  p2        text  
\*----------------------------------------------------------------------\*  
FORM top\_page2 .  
  
  WRITE: / 'Purchase Order Item List',  
         / 'Date: ',   12 v\_date DD/MM/YYYY,  
         / 'User: ',   12 v\_user,  
         / 'Report: ', 12 v\_repid.  
  ULINE.  
  SKIP.  
  
ENDFORM.                                                    " top\_page2  
  
Below is the output:

[](http://4.bp.blogspot.com/-Q01rCfsgXW8/UVmCp89K8_I/AAAAAAAAAeI/thAROmglOrI/s1600/1.jpg)

Primary Listing:

[](http://1.bp.blogspot.com/-0cigq9BKBsQ/UVmDR6vTUCI/AAAAAAAAAeY/m0948G5rKsU/s1600/2.jpg)

If we double click on the PO in this list then following will be generated:

[](http://4.bp.blogspot.com/-wu-vl_V-590/UVmDiMVX1PI/AAAAAAAAAeg/KPdMsv91Ul4/s1600/3.jpg)