\*&---------------------------------------------------------------------\*  
\*& Report  ZAA\_BACKUP\_REPORT\_TO\_TXT.  
\*&  
\*&---------------------------------------------------------------------\*  
\*& Description:  
\*&  Download source code of selected programs and save  
\*&  them into txt files  
\*&  
\*& Technical description:  
\*&  Program automatically recognizes on what operation system does SAP  
\*&  system run. According to this information it dynamically creates  
\*&  OS compatible path.  
\*&  (Program assumes that SAP system runs on client's PC)  
\*&  
\*&---------------------------------------------------------------------\*  
\*& Change log:  
\*& Date       Author        Action  
\*& 2017-02-01 ABAP\_ACADEMY  Created  
\*&  
\*&---------------------------------------------------------------------\*  
  
REPORT  zaa\_backup\_report\_to\_txt.  
  
\*-----------------------------------\*  
\* DECLARATIONS  
  
" Constants  
  
CONSTANTS:  
  BEGIN OF gc\_download\_status,  
    success TYPE c VALUE 'S',  
    error   TYPE c VALUE 'E',  
  END OF gc\_download\_status.  
  
" Types  
  
TYPES:  
  tv\_path              TYPE c LENGTH 255, " Download Path  
  tv\_program\_name      TYPE sobj\_name,  
  tv\_source\_code\_line  TYPE string.  
  
TYPES:  
  BEGIN OF ts\_log,     " Logging Download Process to This Structure  
    object TYPE char4, " Name of the Program  
    status TYPE c,     " Status can contain 2 states:  
                       " 'S' - Successful Download of Program  
                       " 'E' - Error in Download  
    ddtext TYPE ddtext, " Long Text Description  
  END OF ts\_log.  
  
TYPES:  
  tt\_program\_names  TYPE TABLE OF tv\_program\_name,  
  tt\_source\_code    TYPE TABLE OF tv\_source\_code\_line,  
  tt\_logs           TYPE TABLE OF ts\_log.  
  
" Variables  
  
DATA:  
  gv\_devclass              TYPE devclass,         " Development Class/Package where programs are stored  
  gv\_program\_name          TYPE tv\_program\_name,  " Program Name for Download  
  gv\_folder                TYPE string.           " Target Folder Where Files Will Be Stored  
  
" Internal Tables  
  
DATA:  
  gt\_program\_names  TYPE tt\_program\_names,        " All Program Names Ready to Be Downloaded  
  gt\_logs           TYPE tt\_logs.                 " Download Process Log  
  
  
\*-----------------------------------\*  
\* SELECTION-SCREEN  
  
" Download Either All Programs from Package or Select Specific Programs  
SELECTION-SCREEN BEGIN OF BLOCK b01 WITH FRAME TITLE text-b01.  
SELECT-OPTIONS: so\_devc FOR gv\_devclass.  
SELECT-OPTIONS: so\_prog FOR gv\_program\_name.  
SELECTION-SCREEN END OF BLOCK b01.  
  
" Choose Download Path to Local Hard Drive  
SELECTION-SCREEN BEGIN OF BLOCK b02 WITH FRAME TITLE text-b02.  
PARAMETERS: p\_path TYPE tv\_path OBLIGATORY.  
SELECTION-SCREEN END OF BLOCK b02.  
  
" Display Log Information on the Screen  
SELECTION-SCREEN BEGIN OF BLOCK b03 WITH FRAME TITLE text-b03.  
PARAMETERS: p\_succ AS CHECKBOX DEFAULT abap\_true.  
PARAMETERS: p\_err  AS CHECKBOX DEFAULT abap\_true.  
SELECTION-SCREEN END OF BLOCK b03.  
  
  
\*-----------------------------------\*  
\* AT SELECTION-SCREEN Event Implementation  
  
AT SELECTION-SCREEN ON VALUE-REQUEST FOR p\_path.  
  
  " Display Window to Choose Target Path on Local Hard Drive  
  CALL METHOD cl\_gui\_frontend\_services=>directory\_browse  
    CHANGING  
      selected\_folder = gv\_folder  
    EXCEPTIONS  
      cntl\_error      = 1  
      error\_no\_gui    = 2  
      OTHERS          = 3.  
  
  IF sy-subrc = 0.  
    p\_path = gv\_folder.  
  ENDIF.  
  
  
\*-----------------------------------\*  
\* MAIN LOGIC  
  
START-OF-SELECTION.  
  
  " Check packages or programs are selected for download  
  IF so\_devc IS INITIAL AND so\_prog IS INITIAL.  
    WRITE: icon\_red\_light, 'Firstly choose either development package or program to download!'.  
    EXIT. " Program  
  ENDIF.  
  
  " Search for programs in the system  
  PERFORM get\_programs\_to\_download  
    TABLES so\_devc  
           so\_prog  
    CHANGING gt\_program\_names.  
  
  " When no programs found to download, leave the program  
  IF gt\_program\_names IS INITIAL.  
    WRITE: /, icon\_red\_light, 'No programs found for download'.  
    EXIT. " Program  
  ENDIF.  
  
  " Download programs to local hard drive  
  PERFORM download\_programs\_to\_hdd  
    USING  
      p\_path  
      gt\_program\_names  
    CHANGING  
      gt\_logs.  
  
  " Display log on the screen  
  PERFORM display\_logs  
   USING gt\_logs  
         p\_succ  
         p\_err.  
  
END-OF-SELECTION.  
  
  
\*-----------------------------------\*  
\* SUBROUTINE IMPLEMENTATIONS  
  
  
\*&---------------------------------------------------------------------\*  
\*&      Form  download\_reports\_to\_hdd  
\*&---------------------------------------------------------------------\*  
\*       Download Source Codes of All Programs and Save Them to HDD  
\*       Log Results for All Program Downloads  
\*----------------------------------------------------------------------\*  
\*      -->IV\_PATH           (INPUT) Target Path  
\*      -->IT\_PROGRAM\_NAMES  (INPUT) Program Names Ready for Download  
\*      -->ET\_LOGS           (OUTPUT)Download Activity Log  
\*----------------------------------------------------------------------\*  
FORM download\_programs\_to\_hdd  
  USING    iv\_path                      TYPE tv\_path  
           it\_program\_names             TYPE tt\_program\_names  
  CHANGING et\_logs                      TYPE tt\_logs.  
  
  DATA:  
    lb\_is\_source\_code\_get TYPE abap\_bool.  
  DATA:  
    lv\_file\_path          TYPE string,  
    lv\_program\_name       TYPE tv\_program\_name.  
  DATA:  
    ls\_log                TYPE ts\_log.  
  DATA:  
    lt\_source\_code        TYPE TABLE OF tv\_source\_code\_line.  
  
  " Initialize exporting parameters  
  CLEAR et\_logs.  
  
  " For each program separately do the actions inside of this cycle  
  LOOP AT it\_program\_names INTO lv\_program\_name.  
  
    " Initialize variable before next cycle starts  
    CLEAR:  
      lb\_is\_source\_code\_get,  
      lt\_source\_code.  
  
    " Get source code of the program  
    PERFORM get\_source\_code  
      USING  
        lv\_program\_name  
      CHANGING  
        lb\_is\_source\_code\_get  
        lt\_source\_code  
        et\_logs.  
    IF lb\_is\_source\_code\_get = abap\_false.  
      CONTINUE. " Skip upcomming functionality and go to next cycle  
    ENDIF.  
  
    " Create path with target filename where source code will be downloaded  
    " E.g. when program name is ZAA\_TEST\_PROGRAM, final file path will be the following:  
    " <IV\_PATH - Path to Target Folder><OS Separator><File Name - NSP\_ZAA\_TEST\_PROGRAM.abap>  
    PERFORM create\_path\_to\_file  
      USING iv\_path  
            lv\_program\_name  
      CHANGING lv\_file\_path.  
  
    " Download source code to file on PC  
    CALL FUNCTION 'GUI\_DOWNLOAD'  
      EXPORTING  
        filename                = lv\_file\_path  
      TABLES  
        data\_tab                = lt\_source\_code  
      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.  
  
      " Add Error Information to Log  
      ls\_log-status = gc\_download\_status-error.  
      CONCATENATE lv\_program\_name ' was not downloaded succesfully'  
        INTO ls\_log-ddtext.  
      APPEND ls\_log TO et\_logs.  
  
    ELSE.  
  
      " Add Error Information to Log  
      ls\_log-status = gc\_download\_status-success.  
      CONCATENATE lv\_program\_name ' was downloaded succesfully'  
        INTO ls\_log-ddtext.  
      APPEND ls\_log TO et\_logs.  
  
    ENDIF.  
  
  ENDLOOP.  
  
ENDFORM.                    " download\_reports\_to\_hdd  
  
\*&---------------------------------------------------------------------\*  
\*&      Form  get\_programs\_to\_download  
\*&---------------------------------------------------------------------\*  
\*       Get all program names from TADIR system table  
\*----------------------------------------------------------------------\*  
\*      -->IT\_DEVC           (INPUT) Development Classes/Packages  
\*      -->IT\_PROG           (INPUT) Program Names  
\*      -->CT\_PROGRAM\_NAMES  (OUTPUT)Program Names Ready to Be Downloaded  
\*----------------------------------------------------------------------\*  
FORM get\_programs\_to\_download  
  TABLES   it\_devc  
           it\_prog  
  CHANGING ct\_program\_names  TYPE tt\_program\_names.  
  
  " Get all reports from TADIR by provided devclass and programs  
  " chosen from selection screen  
  SELECT obj\_name AS name FROM tadir  
    APPENDING TABLE ct\_program\_names  
    WHERE pgmid = 'R3TR'  
      AND object = 'PROG'  
      AND obj\_name IN it\_prog  
      AND devclass IN it\_devc  
      AND delflag <> 'X'.  
  
ENDFORM.                    " GET\_REPORTS  
  
\*&---------------------------------------------------------------------\*  
\*&      Form  get\_source\_code  
\*&---------------------------------------------------------------------\*  
\*       Get Source Code of the Requested Program Name  
\*----------------------------------------------------------------------\*  
\*      -->IV\_PROGRAM\_NAME        (INPUT) Program Name  
\*      -->EB\_IS\_SOURCE\_CODE\_GET  (OUTPUT)(BOOL) Was this soubroutine  
\*                                able to get source code?  
\*      -->ET\_SOURCE\_CODE         (OUTPUT)Exported Source Code  
\*      -->CT\_LOGS                (OUTPUT)Log Information  
\*----------------------------------------------------------------------\*  
FORM get\_source\_code  
  USING  
    iv\_program\_name       TYPE tv\_program\_name  
  CHANGING  
    eb\_is\_source\_code\_get TYPE abap\_bool  
    et\_source\_code        TYPE tt\_source\_code  
    ct\_logs               TYPE tt\_logs.  
  
  DATA:  
    ls\_log TYPE ts\_log.  
  
  " Initialize exporting parameters  
  CLEAR: et\_source\_code, eb\_is\_source\_code\_get.  
  
  " Read source code of report  
  READ REPORT iv\_program\_name INTO et\_source\_code.  
  IF sy-subrc <> 0.  
  
    " We were not able to get source code  
    eb\_is\_source\_code\_get = abap\_false.  
  
    " Add information about problem to log  
    ls\_log-status = 'R'.  
    CONCATENATE iv\_program\_name ' - problem reading report'  
      INTO ls\_log-ddtext.  
    APPEND ls\_log TO ct\_logs.  
  
    EXIT. " FORM execution  
  
  ENDIF.  
  
  " We have got source code successfuly  
  eb\_is\_source\_code\_get = abap\_true.  
  
ENDFORM.                    "get\_source\_code  
  
\*&---------------------------------------------------------------------\*  
\*&      Form  create\_path\_to\_file  
\*&---------------------------------------------------------------------\*  
\*       Create Target Path to File on Local Hard Drive  
\*----------------------------------------------------------------------\*  
\*      -->IV\_PATH          (INPUT) Path  
\*      -->IV\_PROGRAM\_NAME  (INPUT) Program Name  
\*      -->EV\_FILE\_PATH     (OUTPUT)Target File Path  
\*----------------------------------------------------------------------\*  
FORM create\_path\_to\_file  
  USING iv\_path         TYPE tv\_path  
        iv\_program\_name TYPE tv\_program\_name  
  CHANGING ev\_file\_path TYPE string.  
  
  DATA:  
   lv\_filename  TYPE string,  
   lv\_separator TYPE c.  
  
  CLEAR ev\_file\_path.  
  
  " Create filename for the file containing the source code  
  " E.g. when program name is ZAA\_TEST\_PROGRAM, filename will be called:  
  " 'NSP\_ZAA\_TEST\_PROGRAM.abap'  
  PERFORM create\_filename  
    USING iv\_program\_name  
    CHANGING lv\_filename.  
  
  " Get folder separator of current operation system  
  PERFORM get\_os\_separator  
    CHANGING lv\_separator.  
  
  " Create file path, where file will be stored  
  CONCATENATE iv\_path lv\_separator lv\_filename  
    INTO ev\_file\_path.  
  
ENDFORM.                    "create\_path\_to\_file  
  
\*&---------------------------------------------------------------------\*  
\*&      Form  create\_filename  
\*&---------------------------------------------------------------------\*  
\*       Create Filename of Where Program Will Be Stored  
\*----------------------------------------------------------------------\*  
\*      -->IV\_PROGRAM\_NAME  (INPUT) Program Name  
\*      -->EV\_FILENAME      (OUTPUT)Target Filename  
\*----------------------------------------------------------------------\*  
FORM create\_filename  
  USING iv\_program\_name TYPE tv\_program\_name  
  CHANGING ev\_filename  TYPE string.  
  
  DATA:  
    lv\_program\_name TYPE tv\_program\_name.  
  
  " Initilialize exporting parameter  
  CLEAR ev\_filename.  
  
  lv\_program\_name = iv\_program\_name.  
  
  " In case that program name starts with '/', remove it from the beginning of the filename  
  IF iv\_program\_name(1) = '/'.  
    SHIFT lv\_program\_name LEFT DELETING LEADING '/'.  
  ENDIF.  
  
  " Filename Pattern  
  " <System ID>\_<Program Name>.abap  
  CONCATENATE sy-sysid '\_' lv\_program\_name '.abap'  
    INTO ev\_filename.  
  
  " If program name contains '/' sign, change it to '\_'  
  REPLACE ALL OCCURRENCES OF '/' IN ev\_filename WITH '\_'.  
  
ENDFORM.                    "create\_filename  
  
\*&--------------------------------------------------------------------------------------\*  
\*&      Form  display\_logs  
\*&--------------------------------------------------------------------------------------\*  
\*       Display Activity Log  
\*---------------------------------------------------------------------------------------\*  
\*      -->IT\_LOGS                 (INPUT) Activity Log  
\*      -->IB\_IS\_SUCCESS\_REQUESTED (INPUT) Is Display of Success Log Messages Requested?  
\*      -->IB\_IS\_ERROR\_REQUESTED   (INPUT) Is Display of Error Log Messages Requested?  
\*---------------------------------------------------------------------------------------\*  
FORM display\_logs  
  USING   it\_logs                 TYPE tt\_logs  
          ib\_is\_success\_requested TYPE abap\_bool  
          ib\_is\_error\_requested   TYPE abap\_bool.  
  
  DATA: ls\_log TYPE ts\_log.  
  
  " Display Error Messages if Requested  
  IF ib\_is\_error\_requested = abap\_true.  
  
    LOOP AT it\_logs INTO ls\_log WHERE status = gc\_download\_status-error.  
      WRITE: /, icon\_red\_light, ls\_log-object, ls\_log-ddtext.  
    ENDLOOP.  
  
  ENDIF.  
  
  " Display Success Messages if Requested  
  IF ib\_is\_success\_requested = abap\_true.  
  
    LOOP AT it\_logs INTO ls\_log WHERE status = gc\_download\_status-success.  
      WRITE: /, icon\_green\_light, ls\_log-object, ls\_log-ddtext.  
    ENDLOOP.  
  
  ENDIF.  
  
ENDFORM.                    " DISPLAY\_LOGS  
  
\*&---------------------------------------------------------------------\*  
\*&      Form  get\_os\_separator  
\*&---------------------------------------------------------------------\*  
\*       Get OS Specific Folder Separator  
\*----------------------------------------------------------------------\*  
\*      -->EV\_SEPARATOR  (OUTPUT) OS Specific Separator  
\*----------------------------------------------------------------------\*  
FORM get\_os\_separator  
  CHANGING ev\_separator TYPE c.  
  
  CONSTANTS:  
    BEGIN OF lc\_separator,  
      windows TYPE c VALUE '\',  
      unix    TYPE c VALUE '/',  
    END OF lc\_separator.  
  
  CLEAR ev\_separator.  
  
  IF sy-opsys CA 'Win'.  
    ev\_separator = lc\_separator-windows.  
  ELSE.  
    ev\_separator = lc\_separator-unix.  
  ENDIF.  
  
ENDFORM.                    "get\_os\_separator