1 System

This chapter contains functions that could be associated with BASIS type of operations, such as finding the IP address of a terminal and opening FTP sessions.

ABAP4_CALL_TRANSACTION

Summary

Initiates a transaction in a separate window.

Description

Basically a wrapper to CALL TRANSACTION. Within an ABAP program, this will start an additional transaction. The normal rules of authorisation to run the transaction naturally still apply.

Parameters

```
EXPORTING
       TCODE             Contains the transaction code to be called.
       SKIP_SCREEN       If set, will skip the first screen of the transaction.
       MODE_VAL          Display mode:
                         Value             Meaning
                         A (default)       Display the screens
                         E                 Only display screens if an error occurs
                         N                 Do not display (background mode)
       UPDATE_VAL        Update mode:
                         Value             Meaning
                         A (default)       Asynchronous update
                         S                 Synchronous update
                         L                 Local update
TABLES
       USING_TAB         BDC data for the transaction
       SPAGPA_TAB        Holds SPA\GPA parameters to fill input fields
       MESS_TAB          Contains any error messages from the transaction
```

Example

```
REPORT ZEXAMPLE.
DATA: BEGIN OF IMESS OCCURS 0.
       INCLUDE STRUCTURE BDCMSGCOLL.
DATA: END OF IMESS.


CALL FUNCTION 'ABAP4_CALL_TRANSACTION' STARTING NEW TASK 'ZTSK'
       EXPORTING
         TCODE                     = 'SE38'       "START ABAP DEVELOPMENT
       TABLES
         MESS_TAB                  = IMESS
       EXCEPTIONS
         CALL_TRANSACTION_DENIED   = 1
         TCODE_INVALID             = 2
         OTHERS                    = 3.


IF SY-SUBRC <> 0.
  LOOP AT IMESS.
    WRITE:/IMESS-MSGV1,
           IMESS-MSGV2,
           IMESS-MSGV3.
  ENDLOOP.
ENDIF.
```

See Also

HLP_MODE_CREATE, TH_REMOTE_TRANSACTION, TRANSACTION_CALL

ARFC_GET_TID

Summary

Returns the IP address of the server in hexadecimal.

Description

The IP address is returned from the function in hexadecimal, so this should be formatted to the normal dotted notation of an IP address before being displayed to the user. The example will do this for you.

Parameters

```
IMPORTING
       TID  Contains the IP address of the user's computer that runs the function.
```

Example

```
REPORT ZEXAMPLE.
DATA: TERM_IP        LIKE ARFCTID,
      IP_ADDR(20)    TYPE C,
      IP_BIT(3)      TYPE C,
      HOSTADDR(4)    TYPE X,
      HEX_CHAR       TYPE X,
      HADDR_X(8)     TYPE X,
      IP_LEN         TYPE I,
      HEXIP_LEN      TYPE I VALUE 0,
      HEXIP          TYPE I,
      CHAR_HEX       TYPE I.

CALL FUNCTION 'ARFC_GET_TID'
     IMPORTING
          TID = TERM_IP.

HOSTADDR = TERM_IP(8).
HADDR_X = HOSTADDR.

DESCRIBE FIELD HOSTADDR LENGTH HEXIP_LEN.
HEXIP_LEN = HEXIP_LEN - 1.

DO HEXIP_LEN TIMES.
  HEX_CHAR = HADDR_X + HEXIP(1).
  CHAR_HEX = HEX_CHAR.
  IP_BIT = CHAR_HEX.
  CONDENSE IP_BIT.
  IP_LEN = STRLEN(IP_ADDR).
  IP_ADDR + IP_LEN = IP_BIT.
  IP_LEN = STRLEN(IP_ADDR).
  IP_ADDR + IP_LEN = '.'.
  HEXIP = HEXIP + 1.
ENDDO.

HEX_CHAR = HADDR_X + HEXIP(1).
CHAR_HEX = HEX_CHAR.
IP_BIT = CHAR_HEX.
CONDENSE IP_BIT.
IP_LEN = STRLEN(IP_ADDR).
IP_ADDR + IP_LEN = IP_BIT.

WRITE:/ 'SERVER IP ADDRESS IS:', IP_ADDR.
```

See Also

TERMINAL_ID_GET, TH_USER_INFO

AUTHORITY_CHECK_DATASET

Summary

Checks file access authorisation.

Description

This function module allows you to check the user's authorisation to access files (with commands OPEN DATASET, READ DATASET, TRANSFER and DELETE DATASET). A check should be performed before opening a file. This function is well documented.

Parameters

```
EXPORTING
       PROGRAM      Program containing file access command (default: current program)
       ACTIVITY     Access type required to file:
                    Value                Meaning
                    READ                 Read file
                    WRITE                Change file
                    READ_WITH_FILTER     Read file with filter function
                    WRITE_WITH_FILTER    Change file with filter function
                    DELETE               Delete file
       FILENAME     Name of accessed file
```

Example

```
REPORT ZEXAMPLE.
DATA: BEGIN OF ITAB OCCURS 0,
         ATYPE(20),
      END OF ITAB.

PARAMETER  P_FNAME LIKE AUTHB-FILENAME.
PARAMETERS:P_READ AS CHECKBOX DEFAULT 'X',
           P_WRITE AS CHECKBOX DEFAULT 'X',
           P_RWF AS CHECKBOX DEFAULT 'X',
           P_WWF AS CHECKBOX DEFAULT 'X',
           P_DELETE AS CHECKBOX DEFAULT 'X'.

CLEAR: ITAB, ITAB[].

IF P_READ EQ 'X'.
  ITAB-ATYPE = 'READ'.
  APPEND ITAB.
ENDIF.

IF P_WRITE EQ 'X'.
  ITAB-ATYPE = 'WRITE'.
  APPEND ITAB.
ENDIF.

IF P_RWF EQ 'X'.
  ITAB-ATYPE = 'READ_WITH_FILTER'.
  APPEND ITAB.
ENDIF.

IF P_WWF EQ 'X'.
  ITAB-ATYPE = 'WRITE_WITH_FILTER'.
  APPEND ITAB.
ENDIF.
```

```
IF P_DELETE EQ 'X'.
  ITAB-ATYPE = 'DELETE'.
  APPEND ITAB.
ENDIF.

LOOP AT ITAB.
  CALL FUNCTION 'AUTHORITY_CHECK_DATASET'
       EXPORTING
            ACTIVITY              = ITAB-ATYPE
            FILENAME              = P_FNAME
       EXCEPTIONS
            NO_AUTHORITY          = 1
            ACTIVITY_UNKNOWN      = 2
            OTHERS                = 3.
  CASE SY-SUBRC.
    WHEN 0.
      WRITE:/ 'You have', ITAB-ATYPE, 'access to', P_FNAME.
    WHEN 1.
      WRITE:/ 'You do not have', ITAB-ATYPE, 'access to', P_FNAME.
    WHEN OTHERS.
      WRITE:/ 'Error with function'.
  ENDCASE.
ENDLOOP.
```

BP_EVENT_RAISE

Summary

Triggers an event in the background-processing system from an ABAP program.

Description

Events let you start background jobs under defined conditions. The event IDs are defined in transaction SM62 (event arguments are specified when the job is scheduled).

When you define a new event, a transport request must be manually created if it is to be transported to another system.

Parameters

```
EXPORTING
       EVENTID    The event name, defined in SM62
       EVENTPARM  Job can be scheduled to wait for an EVENTID or combination of EVENTID
                  and EVENTPARM
```

Example

```
REPORT ZEXAMPLE.
DATA: Q_EVENT      LIKE TBTCJOB-EVENTID   VALUE 'SAP_QEVENT',
      Q_EVENTPARM  LIKE TBTCJOB-EVENTPARM.
```

```
CALL FUNCTION 'BP_EVENT_RAISE'
       EXPORTING
            EVENTID               = Q_EVENT
            EVENTPARM             = Q_EVENTPARM
       EXCEPTIONS
            BAD_EVENTID           = 1
            EVENTID_DOES_NOT_EXIST = 2
            EVENTID_MISSING       = 3
            RAISE_FAILED          = 4
            OTHERS                = 5.

IF SY-SUBRC NE 0.
       WRITE:/ 'EVENT', Q_EVENT, 'NOT RAISED'.
ELSE.
       WRITE:/ 'EVENT', Q_EVENT, 'RAISED SUCCESSFULLY'.
ENDIF.
```

See Also

GET_JOB_RUNTIME_INFO

CAT_CHECK_RFC_DESTINATION

Summary

Checks for the RFC destinations and connections on a client.

Description

RFC destinations are defined within SAP using transaction code SM59.

Parameters

```
EXPORTING
       RFCDESTINATION    System to be tested
IMPORTING
       MSGV1             RFC message
       MSGV2             RFC message
       RFC_SUBRC         RFC return code
```

Example

```
REPORT ZEXAMPLE.
DATA:  RFCDESTINATION   LIKE  RSCAT-RFCDEST,
       V_MSGV1          LIKE  SY-MSGV1,
       V_MSGV2          LIKE  SY-MSGV2,
       V_SUBRC          LIKE  SYST-SUBRC.
```

```
CALL FUNCTION 'CAT_CHECK_RFC_DESTINATION'
       EXPORTING
              RFCDESTINATION  = RFCDESTINATION
       IMPORTING
              MSGV1           = V_MSGV1
              MSGV2           = V_MSGV2
              RFC_SUBRC       = V_SUBRC.

IF V_SUBRC NE 0.
       WRITE:/ 'ERROR:', V_MSGV1, V_MSGV2.
ELSE.
       SET PARAMETER ID 'RFC' FIELD RFCDESTINATION.
       WRITE:/ 'CONNECTION TO', RFCDESTINATION, 'IS WORKING'.
ENDIF.
```

See Also

CAT_PING, TH_SERVER_LIST

CAT_PING

Summary

Checks RFC system and configuration.

Description

Tests if an RFC system is reachable and returns configuration data if possible.

Parameters

```
EXPORTING
       RFCDESTINATION   System to be tested
IMPORTING
       SYSINFO          Structure with RFC system configuration information
```

Example

```
REPORT ZEXAMPLE.
DATA: BEGIN OF SYSINFO.
       INCLUDE STRUCTURE CATFR.
DATA: END OF SYSINFO.
DATA RFC_DESTINATION LIKE RFCDES-RFCDEST.

SYSINFO = SPACE.

CALL FUNCTION 'CAT_PING' DESTINATION RFC_DESTINATION
       IMPORTING
              SYSINFO = SYSINFO
```

```
     EXCEPTIONS
               COMMUNICATION_FAILURE  = 1
               SYSTEM_FAILURE         = 2.

IF SY-SUBRC NE 0.
     WRITE:/ 'COULD NOT CONNECT TO', RFC_DESTINATION.
ELSE.
     WRITE:/ SYSINFO.
ENDIF.
```

See Also

CAT_CHECK_RFC_DESTINATION

DEQUEUE_ES_PROG

Summary

Releases program locks.

Description

This function releases a lock in a program that has been set by ENQUEUE_ES_PROG.

Parameters

```
EXPORTING
     NAME    Program name to lock
```

Example

```
REPORT ZEXAMPLE.
DATA V_PGM TYPE PROGRAMM.

CALL FUNCTION 'DEQUEUE_ES_PROG'
     EXPORTING
             NAME   = V_PGM.

     WRITE:/ 'PROGRAM', V_PGM, 'IS UNLOCKED'.
```

See Also

DEQUEUE_ESFUNCTION, ENQUEUE_ES_PROG

ENQUEUE_ES_PROG

Summary

Prevents the parallel execution of a program.

Description

This function creates a lock in a program that should not be processed more than once, simultaneously. The lock remains in place until either the DEQUEUE_ES_PROG function module is called or the transaction is completed (with an implicit DEQUEUE_ALL call).

Parameters

```
EXPORTING
       NAME        Program name to lock
       SCOPE       Controls how the lock is passed to the update program:
                   Value          Meaning
                   1              The lock is not passed to the update program. The lock
                                  is removed when the transaction ends.
                   2 (default)    The lock is passed to the update program. The update
                                  program is responsible for removing the lock.
                   3              The lock is passed to the update program. The lock must
                                  be removed in both the interactive program and in the
                                  update program.
```

Example

```
REPORT ZEXAMPLE.
DATA V_PGM TYPE PROGRAMM.

CALL FUNCTION 'ENQUEUE_ES_PROG'
     EXPORTING
          NAME            = V_PGM
          SCOPE           = '3'
     EXCEPTIONS
          FOREIGN_LOCK    = 1
          SYSTEM_FAILURE  = 2
          OTHERS          = 3.

IF SY-SUBRC NE 0.
     WRITE:/ 'LOCK FAILED ON PROGRAM ZPROGRAM'.
ELSE.
     WRITE:/ V_PGM, 'SUCCESSFULLY LOCKED AGAINST SIMULTANEOUS PROCESSING'.
ENDIF.
```

See Also

DEQUEUE_ES_PROG, ENQUEUE_ESFUNCTION

FTP_COMMAND

Summary

Executes a command on an FTP server.

Description

Passes an FTP command to an FTP server for processing.

Parameters

```
EXPORTING
       HANDLE       Unique ID identifying FTP session (from FTP_CONNECT)
       COMMAND      Any FTP command. For example, DIR lists files in a directory
TABLES
       DATA         Results from FTP command. For example, filenames in a directory
```

Example

See FTP_CONNECT

See Also

FTP_CONNECT

FTP_CONNECT

Summary

Opens a connection to the FTP server.

Description

FTP_CONNECT requires an encrypted password to work. It returns a unique ID (handle) that can be used with other FTP functions (e.g. FTP_COMMAND).

For the RFC_DESTINATION value, you can use CAT_CHECK_RFC_DESTINATION to determine the FTP server as defined in SAP.

Parameters

```
EXPORTING
     USER                   Username to the FTP server
     PASSWORD               Password valid for the FTP server (encrypted)
     HOST                   FTP server name
     RFC_DESTINATION        The server name as configured in SAP
IMPORTING
     HANDLE                 Unique ID created for this FTP session
```

Example

```
REPORT ZEXAMPLE.
DATA: FTP_USER(64)      VALUE 'FTPUSER',
```

```
      FTP_PWD(64)       VALUE 'FTPPWD',
      FTP_HOST(50)      VALUE 'FTPSERVER',
      RFC_DEST          LIKE RSCAT-RFCDEST VALUE 'RFC_SERVER'.

DATA: HDL TYPE I,
      KEY TYPE I VALUE 26101957,
      DSTLEN TYPE I.

DATA: BEGIN OF FTP_DATA OCCURS 0,
          LINE(132) TYPE C,
      END OF FTP_DATA.

DESCRIBE FIELD FTP_PWD LENGTH DSTLEN.

CALL 'AB_RFC_X_SCRAMBLE_STRING'
      ID 'SOURCE'          FIELD FTP_PWD
      ID 'KEY'             FIELD KEY
      ID 'SCR'             FIELD 'X'
      ID 'DESTINATION'     FIELD FTP_PWD
      ID 'DSTLEN'          FIELD DSTLEN.


CALL FUNCTION 'FTP_CONNECT'
      EXPORTING
           USER              = FTP_USER
           PASSWORD          = FTP_PWD
           HOST              = FTP_HOST
           RFC_DESTINATION   = RFC_DEST
      IMPORTING
           HANDLE            = HDL
      EXCEPTIONS
           NOT_CONNECTED     = 1
           OTHERS            = 2.

IF SY-SUBRC NE 0.
      WRITE:/ 'COULD NOT CONNECT TO', FTP_HOST.
ELSE.
      WRITE:/ 'CONNECTED SUCCESSFULLY. SESSION HANDLE IS', HDL.

      CALL FUNCTION 'FTP_COMMAND'
             EXPORTING
                    HANDLE         = HDL
                    COMMAND        = 'DIR'
             TABLES
                    DATA           = FTP_DATA
             EXCEPTIONS
                    TCPIP_ERROR    = 1
                    COMMAND_ERROR  = 2
                    DATA_ERROR     = 3
                    OTHERS         = 4.

  IF SY-SUBRC NE 0.
        WRITE:/ 'COULD NOT EXECUTE FTP COMMAND'.
```

```
  ELSE.
        LOOP AT FTP_DATA.
        WRITE: / FTP_DATA.
        ENDLOOP.

        CALL FUNCTION 'FTP_DISCONNECT'
               EXPORTING
                      HANDLE          = HDL
               EXCEPTIONS
                      OTHERS          = 1.
      IF SY-SUBRC NE 0.
       WRITE:/ 'COULD NOT DISCONNECT FROM FTP SERVER'.
    ELSE.
       WRITE:/ 'DISCONNECTED FROM FTP SERVER'.
    ENDIF.
  ENDIF.
ENDIF.
```

See Also

CAT_CHECK_RFC_DESTINATION, FTP_COMMAND, FTP_DISCONNECT

FTP_DISCONNECT

Summary

Closes the connection and logs off the FTP server.

Description

This function also destroys the handle created by FTP_CONNECT, so it is no longer valid for subsequent commands.

Parameters

```
EXPORTING
       HANDLE     Unique ID created by FTP_CONNECT to identify the FTP session
```

Example

See FTP_CONNECT

See Also

FTP_CONNECT, FTP_COMMAND

GET_JOB_RUNTIME_INFO

Summary

Gets information about a job.

Description

This function can also determine what event and argument triggered the start of a background job from within the background job. This is possible only in job steps that start ABAP programs.

Parameters

```
IMPORTING
        EVENTID                Event that started the job (if any)
        EVENTPARM              Event parameters
        JOBNAME                Job returned by function
```

Example

```
REPORT ZEXAMPLE.
PARAMETERS:V_EVTID    LIKE TBTCM-EVENTID,
           V_EVTPRM   LIKE TBTCM-EVENTPARM,
           V_JOBNAM   LIKE TBTCM-JOBNAME.

CALL FUNCTION 'GET_JOB_RUNTIME_INFO'
     IMPORTING
           EVENTID          = V_EVTID
           EVENTPARM        = V_EVTPRM
           JOBNAME          = V_JOBNAM
     EXCEPTIONS
           NO_RUNTIME_INFO  = 1
           OTHERS           = 2.

IF SY-SUBRC NE 0.
     WRITE:/ 'ERROR IN FUNCTION'.
ELSE.
     WRITE:/ 'JOB', V_JOBNAM, 'STARTED WITH EVENT', V_EVTID, 'AND PARAMETER', V_EVTPRM.
ENDIF.
```

See Also

BP_EVENT_ RAISE

GUI_EXEC

Summary

Starts an external program asynchronously.

Description

Replaces WS_EXECUTE to start an external application. This function is only available for Windows 32-bit clients and does not associate applications with file extensions.

Parameters

```
EXPORTING
       COMMAND      Program name
       PARAMETER    Optional field for parameters (if not specified in COMMAND)
IMPORTING
       RETURNCODE   Function return code, assigned by Windows system
```

Example

```
REPORT ZEXAMPLE.
DATA: PROGRAM(255) TYPE C VALUE 'NOTEPAD',
      PARAMETER(255) TYPE C VALUE 'C:\DATAFILE.TXT',
RETCODE TYPE I.

CALL FUNCTION 'GUI_EXEC'
       EXPORTING
              COMMAND     = PROGRAM
              PARAMETER   = PARAMETER
       IMPORTING
              RETURNCODE  = RETCODE.

IF RETCODE NE 0.
  WRITE:/ 'PROGRAM', PROGRAM, 'NOT FOUND OR COULD NOT BE STARTED'.
ENDIF.
```

See Also

GUI_RUN, EXECUTE_WINWORD, WS_EXECUTE, WS_EXCEL

GUI_GET_DESKTOP_INFO

Summary

Returns information about the end-users client (the desktop).

Description

This function is platform specific. Replaces WS_QUERY.

Parameters

```
EXPORTING
       TYPE:  Indicates the information to be returned:
              Value        Meaning
              -2           SAP system directory
               1           Computer name
               2           Windows directory
               3           Windows system directory
               4           Temporary directory
               5           Windows user name
               6           Windows OS
               7           Windows build number
               8           Windows version
               9           SAP GUI program name
               10          SAP GUI program path
               11          SAP current directory
               12          Desktop directory
CHANGING
     RETURN  Text information from function
```

Example

```
REPORT ZEXAMPLE.
DATA: V_VALU(255)    TYPE C,
      INFOREQ        TYPE I VALUE '-2'.

WHILE INFOREQ NE 13.
  CALL FUNCTION 'GUI_GET_DESKTOP_INFO'
         EXPORTING
               TYPE    = INFOREQ
         CHANGING
               RETURN  = V_VALU.

  CASE INFOREQ.
    WHEN '-2'.
      WRITE:/ 'SAP SYSTEM DIRECTORY:', V_VALU.
    WHEN '1'.
      WRITE:/ 'COMPUTER NAME:', V_VALU.
    WHEN '2'.
      WRITE:/ 'WINDOWS DIRECTORY:', V_VALU.
    WHEN '3'.
      WRITE:/ 'WINDOWS SYSTEM DIRECTORY:', V_VALU.
    WHEN '4'.
      WRITE:/ 'TEMPORARY DIRECTORY:', V_VALU.
    WHEN '5'.
      WRITE:/ 'WINDOWS USER NAME:', V_VALU.
    WHEN '6'.
      WRITE:/ 'WINDOWS OS:', V_VALU.
    WHEN '7'.
      WRITE:/ 'WINDOWS BUILD NUMBER:', V_VALU.
    WHEN '8'.
      WRITE:/ 'WINDOWS VERSION:', V_VALU.
```

```
    WHEN '9'.
      WRITE:/ 'SAP GUI PROGRAM NAME:', V_VALU.
    WHEN '10'.
      WRITE:/ 'SAP GUI PROGRAM PATH:', V_VALU.
    WHEN '11'.
      WRITE:/ 'SAP CURRENT DIRECTORY:', V_VALU.
    WHEN '12'.
      WRITE:/ 'DESKTOP DIRECTORY:', V_VALU.
  ENDCASE.

  INFOREQ = INFOREQ + 1.
ENDWHILE.
```

See Also

IW_C_GET_FRONTEND_VERSION, WS_QUERY

GUI_RUN

Summary

Starts program asynchronously with ShellExecute.

Description

This function is only available for Windows 32-bit clients. If you enter a document name for COMMAND, the document is displayed in its corresponding application.

Parameters

```
EXPORTING
      COMMAND       File or program name
      PARAMETER     Optional field for parameters (if not specified in COMMAND)
IMPORTING
      RETURNCODE    Function return code, assigned by the Windows system
```

Example

```
REPORT ZEXAMPLE.
DATA:  PROGRAM(255)     TYPE C VALUE 'NOTEPAD',
       PARAMETER(255)   TYPE C VALUE 'C:\DATAFILE.TXT',
RETCODE TYPE I.

CALL FUNCTION 'GUI_RUN'
       EXPORTING
            COMMAND     = PROGRAM
            PARAMETER   = PARAMETER
       IMPORTING
            RETURNCODE  = RETCODE.
```

```
IF RETCODE NE 0.
  WRITE:/ 'PROGRAM', PROGRAM, 'NOT FOUND OR COULD NOT BE STARTED'.
ENDIF.
```

See Also

GUI_EXEC

GWY_READ_CONNECTIONS

Summary

Checks if the gateway connection is open.

Description

When the connection is broken, the entry is still in the connection tables. The length of time this takes to clear (usually a few seconds) is defined in R/3.

Parameters

```
EXPORTING
     GWHOST          Local host
     GWSERV          Remote server
TABLES
     CONNECTIONS     List of gateway connections
```

Example

```
REPORT ZEXAMPLE.
DATA:   GW_HOST          LIKE  GWY_STRUCT-GWHOST,
        GW_SERV          LIKE  GWY_STRUCT-GWSERV,
        SAPSYS(2),
        CONVERSATION_ID(8).

DATA: BEGIN OF GWCONN OCCURS 0.
        INCLUDE STRUCTURE GWY_CONN.
DATA: END OF GWCONN.

* 1 GET HOST AND SERVICE
CALL 'C_SAPGPARAM' ID 'NAME'  FIELD 'SAPLOCALHOST'
                   ID 'VALUE' FIELD GW_HOST.

CALL 'C_SAPGPARAM' ID 'NAME'  FIELD 'SAPSYSTEM'
                   ID 'VALUE' FIELD SAPSYS.

GW_SERV     = 'SAPGW'.
GW_SERV + 5 = SAPSYS.
```

```
CALL  FUNCTION 'GWY_READ_CONNECTIONS'
      EXPORTING
          GWHOST                     = GW_HOST
          GWSERV                     = GW_SERV
      TABLES
          CONNECTIONS                = GWCONN
      EXCEPTIONS
          GWY_UNKNOWN_OPCODE         = 01
          GWY_COMMUNICATION_FAILURE  = 02
          GWY_GET_TAB_FAILED         = 03
          GWY_NEWLINE_FAILED         = 04
          GWY_TABLEN_TOO_SHORT       = 05
          GWY_GET_OPCODE_FAILED      = 06
          GWY_GET_GWHOST_FAILED      = 07
          GWY_GET_GWSERV_FAILED      = 08.
IF SY-SUBRC NE 0.
  WRITE:/ 'ERROR IN FUNCTION'.
ELSE.
  WRITE:/2 'UNIT', 11 'PROGRAM', 20 'USER', 33 'SYSTEM', 50 'LAST REQUEST'.
  ULINE AT /1(70).

  LOOP AT GWCONN.
    WRITE:   /2 GWCONN-LU,
             11 GWCONN-TP,
             20 GWCONN-GWUSER,
             33 GWCONN-SYMDEST,
             50 GWCONN-LAST_REQ.
  ENDLOOP.
ENDIF.
```

HLP_MODE_CREATE

Summary

Creates another session in the system.

Description

This function creates another session in your system with the transaction passed as parameter to the function module.

Parameters

```
EXPORTING
     TCODE  Transaction to call in new session
```

Example

```
REPORT ZEXAMPLE.
```

```
CALL  FUNCTION   'HLP_MODE_CREATE'
      EXPORTING
                 TCODE = 'SE38'.     "ABAP DEVELOPMENT
```

See Also

ABAP4_CALL_TRANSACTION, TH_REMOTE_TRANSACTION, TRANSACTION_CALL

IW_C_GET_FRONTEND_VERSION

Summary

Version of the SAP frontend installed on a PC.

Description

Display the SAP logon screen (where you choose the server you want to logon), select the vend diagram (top right). Then choose "About Frontend". There is a file version number: 4640.2.0.2071 (as an example). The function returns this value.

Find the path and name by clicking on "Loaded DLLs" pushbutton on the popup box. On scrolling to the bottom of this list, you will see the path to the FRONT.EXE application.

Parameters

```
EXPORTING
      COMPPATH               Path to the SAP GUI
      COMPNAME               Name of the SAP GUI program
IMPORTING
      FILEVERSION            Release version of the GUI
```

Example

```
REPORT ZEXAMPLE.
DATA FILEVERSION LIKE CNTLSTRINF-VERSION.

CALL FUNCTION 'IW_C_GET_FRONTEND_VERSION'
     EXPORTING
          COMPPATH        = 'C:\PROGRAM FILES\SAPPC\SAPGUI\'
          COMPNAME        = 'FRONT.EXE'
     IMPORTING
          FILEVERSION     = FILEVERSION.

WRITE:/ 'THE FRONT-END PROGRAM VERSION IS:', FILEVERSION.
```

See Also

GUI_GET_DESKTOP_INFO

RFC_MAIL

Summary

Sends an e-mail to another SAP system.

Description

To view the function (as it cannot be seen in SE37), go to SM59, select "TCP/IP connections", select and open "SERVER_EXEC" or "LOCAL_EXEC", look on the pull down menu SYSTEM INFORMATION->FUNCTION LIST.

Parameters

```
EXPORTING
     USER          E-mail address of recipient
TABLES
     MAIL          E-mail message
```

Example

```
REPORT ZEXAMPLE.
DATA V_EMAIL(200) TYPE C OCCURS 0 WITH HEADER LINE.

V_EMAIL = 'MESSAGE BODY LINE 1'.     APPEND V_EMAIL.
V_EMAIL = 'MESSAGE BODY LINE 2'.     APPEND V_EMAIL.

CALL FUNCTION 'RFC_MAIL' DESTINATION 'LOCAL_EXEC'
       EXPORTING
            USER = 'USERNAME@SOMEWHERE.COM'
       TABLES
            MAIL = V_EMAIL.

IF SY-SUBRC EQ 0.
       WRITE 'E-MAIL SENT SUCCESSFULLY.'.
ELSE.
       WRITE 'ERROR SENDING E-MAIL.'.
ENDIF.
```

See Also

CAT_CHECK_RFC_DESTINATION, SO_NEW_DOCUMENT_ATT_SEND_API1, SO_ NEW_DOCUMENT_SEND_API1

RSPO_FIND_SPOOL_REQUESTS

Summary

Finds a spool number.

Description

Returns spool number(s) for a user, SAP client, and\or printer ID.

Parameters

```
EXPORTING
     RQOWNER          Spool owner
     ALLCLIENTS       SAP clients
     RQDEST           Printer name
TABLES
     SPOOLREQUESTS    List of spool(s) information
```

Example

```
REPORT ZEXAMPLE.
TYPE-POOLS:SLIS,
           SPO1R.
DATA: SPOOL_OWNER    LIKE SY-UNAME,
     PRN             LIKE TSP03-PADEST VALUE 'LOCL',
     SPOOL_NUMBER    LIKE TSP01-RQIDENT.

DATA: BEGIN OF IRQTAB OCCURS 0.
       INCLUDE STRUCTURE RSPORQ.
DATA: END OF IRQTAB.

DATA: LTAB TYPE SPO1R_TVIEW WITH HEADER LINE,
     RS_SELFIELD TYPE SLIS_SELFIELD.
SPOOL_OWNER = SY-UNAME.
CALL  FUNCTION 'RSPO_FIND_SPOOL_REQUESTS'
     EXPORTING
          RQOWNER         = SPOOL_OWNER
          RQDEST          = PRN
     TABLES
          SPOOLREQUESTS   = IRQTAB.

WRITE:/ 'PRINTING SPOOLS OF', SPOOL_OWNER, 'ON PRINTER', PRN, 'NOW.'.
LOOP AT IRQTAB.
  SPOOL_NUMBER = IRQTAB-RQIDENT.

  CALL FUNCTION 'RSPO_OUTPUT_SPOOL_REQUEST'
     EXPORTING
          DEVICE           = PRN
          SPOOL_REQUEST_ID  = SPOOL_NUMBER
     EXCEPTIONS
          OTHERS           = 1.

  SKIP.
  IF SY-SUBRC EQ 0.
     WRITE: IRQTAB-RQIDENT,'PRINTED ON', PRN.
     EXIT.                   "ONLY PRINT ONE IN THIS EXAMPLE
```

```
  ELSE.
       WRITE: IRQTAB-RQIDENT, 'NOT PRINTED ON', PRN.
  ENDIF.
ENDLOOP.

* PRINT THE LAST ONE, WITH NO PRINT DIALOG BOX
MOVE-CORRESPONDING IRQTAB TO LTAB.
APPEND LTAB.

CALL  FUNCTION 'RSPO_RPRINT_SPOOLREQ'
      EXPORTING
          TEND        = SPOOL_NUMBER
          POPUP       = ' '
      TABLES
          REQ_VIEW    = LTAB
      CHANGING
          RS_SELFIELD = RS_SELFIELD
      EXCEPTIONS
          OTHERS      = 1.

IF SY-SUBRC EQ 0.
  WRITE:/ LTAB-RQIDENT, 'PRINTED ON DEFAULT PRINTER'.
ELSE.
  WRITE:/ LTAB-RQIDENT, 'NOT PRINTED'.
ENDIF.

* DOWNLOAD A SPOOLJOB TO A FILE
CALL  FUNCTION 'RSPO_DOWNLOAD_SPOOLJOB'
      EXPORTING
          ID     = LTAB-RQIDENT
          FNAME  = 'C:\TEMP\SPOOLFILE.TXT'.

WRITE:/ LTAB-RQIDENT, 'DOWNLOADED TO C:\TEMP\SPOOLFILE.TXT'.
```

RSPO_OUTPUT_SPOOL_REQUEST

Summary

Outputs the same request on a different printer.

Description

Picks up a spool on one printer and sends it to another for printing.

Parameters

```
EXPORTING
       DEVICE            Printer name
       SPOOL_REQUEST_ID  Spool number
```

Example

See RSPO_FIND_SPOOL_REQUESTS

See Also

GET_PRINT_PARAMETERS, RSPO_FIND_SPOOL_REQUESTS

RSPO_RPRINT_SPOOLREQ

Summary

Triggers spool to print automatically.

Description

Prints the data from the spool number which is passed into it.

Parameters

```
EXPORTING
     TEND           Spool request number
     POPUP          Display print parameters (default X = Yes)
TABLES
     REQ_VIEW       Spool information
CHANGING
     RS_SELFIELD    Spool ALV display information
```

Example

See RSPO_FIND_SPOOL_REQUESTS

See Also

RSPO_FIND_SPOOL_REQUESTS, GET_PRINT_PARAMETERS

SAPWL_GET_SUMMARY_STATISTIC

Summary

Object usage statistics summary.

Description

Returns summary of usage statistics on a variety of objects, such as users of transactions and reports.

Parameters

```
EXPORTING
     PERIODTYPE            Period to report:
                           Value      Meaning
                           D          Daily
                           W          Weekly
                           M          Monthly
                           Y          Yearly
     HOSTID                Client
     STARTDATE             Start date to calculate statistics
TABLES
     SUMMARY               Statistical summary
```

Example

```
REPORT ZEXAMPLE.
DATA: BEGIN OF SUMMARY OCCURS 0.
        INCLUDE STRUCTURE SAPWLSUMRY.
DATA: END OF SUMMARY.

DATA: PERIODTYPE LIKE  SAPWLACCTP-PERIODTYPE VALUE 'D',
      HOSTID     LIKE  SAPWLSERV-HOSTSHORT,
      STARTDAT   LIKE  SAPWLACCTP-STARTDATE.

DATA AVG(5).

STARTDAT = SY-DATUM.
HOSTID   = SY-HOST.

CALL  FUNCTION 'SAPWL_GET_SUMMARY_STATISTIC'
      EXPORTING
          PERIODTYPE  = PERIODTYPE
          HOSTID      = HOSTID
          STARTDATE   = STARTDAT
      TABLES
          SUMMARY     = SUMMARY.

LOOP AT SUMMARY.
  IF SUMMARY-TASKTYPE = 'DIALOG'.
    AVG = SUMMARY-RESPTI / SUMMARY-COUNT.
    EXIT.
  ENDIF.
ENDLOOP.

WRITE:/ 'AVERAGE RESPONSE TIME:', AVG.
```

See Also

SAPWL_WORKLOAD_GET_DIRECTORY, SAPWL_WORKLOAD_GET_STATISTIC

SAPWL_WORKLOAD_GET_DIRECTORY

Summary

Timeframe of statistics on SAP database.

Parameters

```
TABLES
    DIRECTORY                Table of contents
```

Example

```
REPORT ZEXAMPLE.
DATA: WL_DIR LIKE SAPWLDIR OCCURS 1 WITH HEADER LINE,
      NUM_LINES LIKE SYST-INDEX.

DATA: BEGIN OF ITIME_FRAME OCCURS 1,
         FILLER(14) TYPE C,
         VALUE(7)   TYPE C,
END OF ITIME_FRAME.

CALL  FUNCTION 'SAPWL_WORKLOAD_GET_DIRECTORY'
      TABLES
         DIRECTORY = WL_DIR.

DESCRIBE TABLE WL_DIR LINES NUM_LINES.
IF NUM_LINES > 0.
  LOOP AT WL_DIR WHERE PERIODTYPE = 'M' AND HOSTID <> 'TOTAL'.
    CONCATENATE  WL_DIR-STARTDATE + 4(2)
                 WL_DIR-STARTDATE + 0(4)
    INTO ITIME_FRAME-VALUE SEPARATED BY '/'.
    APPEND ITIME_FRAME.
  ENDLOOP.
ELSE.
  CONCATENATE SYST-DATUM + 4(2) SYST-DATUM + 0(4)
  INTO ITIME_FRAME-VALUE SEPARATED BY '/'.
  APPEND ITIME_FRAME.
ENDIF.

* DELETE DUPLICATE INFO FROM MULTIPLE APP. SERVERS
SORT ITIME_FRAME BY VALUE + 3(4) VALUE(2) ASCENDING.
DELETE ADJACENT DUPLICATES FROM ITIME_FRAME COMPARING VALUE.

LOOP AT ITIME_FRAME.
  WRITE:/ ITIME_FRAME-VALUE.
ENDLOOP.
```

See Also

SAPWL_GET_SUMMARY_STATISTIC, SAPWL_WORKLOAD_GET_STATISTIC

SAPWL_WORKLOAD_GET_STATISTIC

Summary

Object usage statistics.

Description

Returns usage statistics on a variety of objects, such as users of transactions and reports.

Parameters

```
EXPORTING
     PERIODTYPE            Period to report:
                           Value      Meaning
                           D          Daily
                           W          Weekly
                           M          Monthly
                           Y          Yearly
     HOSTID                Client
     STARTDATE             Start date to calculate statistics
TABLES
     USER_STATISTIC        Users of transactions and reports
```

Example

```
REPORT ZEXAMPLE.
DATA: BEGIN OF USER_STATISTIC OCCURS 0.
        INCLUDE STRUCTURE SAPWLUENTI.
DATA: END OF USER_STATISTIC.

DATA: PERIODTYPE LIKE  SAPWLACCTP-PERIODTYPE VALUE 'D',
      HOSTID     LIKE  SAPWLSERV-HOSTSHORT,
      STARTDAT   LIKE  SAPWLACCTP-STARTDATE.

STARTDAT = SY-DATUM.
HOSTID   = SY-HOST.
CALL  FUNCTION 'SAPWL_WORKLOAD_GET_STATISTIC'
      EXPORTING
          PERIODTYPE      = PERIODTYPE
          HOSTID          = HOSTID
          STARTDATE       = STARTDAT
      TABLES
          USER_STATISTIC  = USER_STATISTIC
      EXCEPTIONS
          NO_DATA_FOUND   = 1.

IF SY-SUBRC EQ 0.
  LOOP AT USER_STATISTIC.
      WRITE:/  USER_STATISTIC-TTYPE,
               USER_STATISTIC-ENTRY_ID,
```

```
              USER_STATISTIC-ACCOUNT.
  ENDLOOP.
ELSE.
  WRITE:/ 'NO USER STATISTICS'.
ENDIF.
```

See Also

SAPWL_GET_SUMMARY_STATISTIC, SAPWL_WORKLOAD_GET_STATISTIC

SHOW_JOBSTATE

Summary

Checks the status of a job.

Parameters

```
EXPORTING
     JOBCOUNT            ID number of job
     JOBNAME             Job name
IMPORTING
     ABORTED             Job terminated abnormally
     FINISHED            Job completed successfully
     PRELIMINARY         Job not released to run or no start condition
     READY               Job scheduled, released, start condition fulfilled, but job not
                         yet started
     RUNNING             Job in progress
     SCHEDULED           Job scheduled and released, waiting for start condition to be
                         fulfilled
```

Example

```
REPORT ZEXAMPLE.
TABLES: TBIST, TBIER.

DATA: BEGIN OF I_TJOBS OCCURS 0.
        INCLUDE STRUCTURE TBIER_S.
DATA: END OF I_TJOBS.
DATA: I_TBIZU LIKE TBIZU OCCURS 0 WITH HEADER LINE.

DATA: ABORTED,FINISHED,PRELIMINARY,READY,RUNNING,SCHEDULED.

* JOB NAMES
SELECT * FROM TBIZU INTO TABLE I_TBIZU.

* CURRENT JOBS
SELECT * FROM TBIST.
  MOVE-CORRESPONDING TBIST TO I_TJOBS.
  I_TJOBS-JOBID = TBIST-JOBNAME.
```

```
  APPEND I_TJOBS.
ENDSELECT.

* COMPLETED JOBS
SELECT * FROM TBIER.
  MOVE-CORRESPONDING TBIER TO I_TJOBS.
  I_TJOBS-JOBID = TBIER-JOBNAME.
  APPEND I_TJOBS.
ENDSELECT.

LOOP AT I_TJOBS.
  READ TABLE I_TBIZU WITH KEY JOBID = I_TJOBS-JOBID BINARY SEARCH.
  IF SY-SUBRC        = 0.
    I_TJOBS-JOBTEXT  = I_TBIZU-JOBTEXT.
    I_TJOBS-BTCJOB   = I_TBIZU-BTCJOB.
  ENDIF.

  CALL  FUNCTION 'SHOW_JOBSTATE'
        EXPORTING
            JOBCOUNT          = I_TJOBS-JOBCOUNT
            JOBNAME           = I_TJOBS-BTCJOB
        IMPORTING
            ABORTED           = ABORTED
            FINISHED          = FINISHED
            PRELIMINARY       = PRELIMINARY
            READY             = READY
            RUNNING           = RUNNING
            SCHEDULED         = SCHEDULED
        EXCEPTIONS
            JOBCOUNT_MISSING  = 01
            JOBNAME_MISSING   = 02
            JOB_NOTEX         = 03
            OTHERS            = 99.
  IF SY-SUBRC <> 0.
    I_TJOBS-STATUSTEXT = 'ERROR IN FUNCTION CALL'.
  ELSE.
    IF ABORTED = 'X'.
      I_TJOBS-STATUSTEXT   = 'JOB ABORTED'.
    ELSEIF FINISHED        = 'X'.
      I_TJOBS-STATUSTEXT   = 'JOB FINISHED'.
    ELSEIF SCHEDULED       = 'X'.
      I_TJOBS-STATUSTEXT   = 'JOB SCHEDULED'.
    ELSEIF RUNNING         = 'X'.
      I_TJOBS-STATUSTEXT   = 'JOB RUNNING'.
    ELSEIF READY           = 'X'.
      I_TJOBS-STATUSTEXT   = 'JOB READY'.
    ELSE.
      I_TJOBS-STATUSTEXT   = 'JOB UNKNOWN'.
    ENDIF.
    MODIFY I_TJOBS.
  ENDIF.
ENDLOOP.

WRITE:/'JOB ID', 10 'JOB STATUS'.
LOOP AT I_TJOBS.
```

```
  WRITE:/ I_TJOBS-JOBID,
          I_TJOBS-STATUSTEXT.
ENDLOOP.
```

SO_SPOOL_READ

Summary

Returns printer spool information.

Description

Retrieves data from a spool.

Parameters

```
EXPORTING
     SPOOL_NUMBER            Printer spool number
TABLES
     OBJCONT                 Data from printer spool
```

Example

```
REPORT ZEXAMPLE.
DATA: BEGIN OF OBJCONT OCCURS 0.
        INCLUDE STRUCTURE SOLI.
DATA: END OF OBJCONT.

DATA: SPOOL_NUMBER     LIKE  RSPOTYPE-RQNUMBER,
      OWNER            LIKE  SOUD-USRNAM,
      CONT_SIZE        LIKE  RSTSTYPE-LINELENGTH.

OWNER = SY-UNAME.
CALL  FUNCTION 'SO_WIND_SPOOL_LIST' "DISPLAY ALL SPOOL NUMBERS FOR OWNER
      EXPORTING
           OWNER         = OWNER
      IMPORTING
           SPOOL_NUMBER  = SPOOL_NUMBER.

CALL  FUNCTION 'SO_SPOOL_READ'       "GET SPOOL INFOMATION
      EXPORTING
           SPOOL_NUMBER                  = SPOOL_NUMBER
      TABLES
           OBJCONT                       = OBJCONT
      EXCEPTIONS
           CONVERT_ERROR                 = 1
           OBJECT_NOT_EXIST              = 2
           OPERATION_NO_AUTHORIZATION    = 3
           SPOOL_CLOSE_ERROR             = 4
```

```
          SPOOL_OPEN_ERROR               = 5
          SPOOL_READ_ERROR               = 6
          OTHERS                         = 7.

IF SY-SUBRC EQ 0.
  CALL FUNCTION 'RSPO_SPOOLDATA_WRITE_INIT'.

  LOOP AT OBJCONT.
    CONT_SIZE = STRLEN(OBJCONT).
    CALL FUNCTION 'RSPO_SPOOLDATA_WRITE' "DISPLAY SPOOL
         EXPORTING
            SPOOL_DATA  = OBJCONT
            DATA_LENGTH = CONT_SIZE
            START_POS   = 1
         EXCEPTIONS
            OTHERS      = 1.
  ENDLOOP.
ELSE.
  WRITE:/ 'COULD NOT DISPLAY SPOOL:', SPOOL_NUMBER.
ENDIF.
```

See Also

RSPO_RETURN_ABAP_SPOOLJOB, SO_WIND_SPOOL_LIST

SO_WIND_SPOOL_LIST

Summary

Popup dialogue to browse printer spool numbers.

Parameters

```
EXPORTING
       OWNER                     Name of printer spool creator
IMPORTING
       SPOOL_NUMBER              Printer spool number
```

Example

See SO_SPOOL_READ

See Also

SO_SPOOL_READ

SXPG_CALL_SYSTEM

Summary

Calls a command external to the SAP system.

Description

The function checks the user's authorisation to run a command and runs the command on the system on which the function module is executed. Internally, it calls SXPG_COMMAND_LIST_GET and SXPG_COMMAND_EXECUTE. Use with caution!!

Parameters

```
IMPORTING
     COMMANDNAME          Name of the external command, as defined in the maintenance
                          function (transaction SM69)
     PARAMETERS           Arguments for the external command
EXPORTING
     STATUS               Returns the final execution status of the external command:
                          Value       Meaning
                          0           Command started and completed successfully
                          E           Command failed
TABLES
     EXEC_PROTOCOL        Contains output of the command and host system
```

Example

```
REPORT ZEXAMPLE.
DATA: BEGIN OF COMMAND_LIST OCCURS 0.
        INCLUDE STRUCTURE SXPGCOLIST.
DATA: END OF COMMAND_LIST .

DATA: BEGIN OF EXEC_PROTOCOL OCCURS 0.
        INCLUDE STRUCTURE BTCXPM.
DATA: END OF EXEC_PROTOCOL.

DATA: STATUS LIKE BTCXP3-EXITSTAT,
      COMMANDNAME LIKE SXPGCOLIST-NAME VALUE '*',
      SEL_NO LIKE SY-TABIX.

* GET LIST OF EXTERNAL COMMANDS
CALL  FUNCTION 'SXPG_COMMAND_LIST_GET'
      EXPORTING
          COMMANDNAME       = COMMANDNAME
          OPERATINGSYSTEM   = SY-OPSYS
      TABLES
          COMMAND_LIST      = COMMAND_LIST
```

```
     EXCEPTIONS
        OTHERS           = 1.

IF SY-SUBRC EQ 0.
  CALL  FUNCTION 'POPUP_WITH_TABLE_DISPLAY'
       EXPORTING
           ENDPOS_COL    = 100
           ENDPOS_ROW    = 20
           STARTPOS_COL  = 2
           STARTPOS_ROW  = 2
           TITLETEXT     = 'CHOOSE A COMMAND TO EXECUTE:'
       IMPORTING
           CHOISE        = SEL_NO
       TABLES
           VALUETAB      = COMMAND_LIST
       EXCEPTIONS
           BREAK_OFF     = 1
           OTHERS        = 2.

  IF SY-SUBRC EQ 0.
    READ TABLE COMMAND_LIST INDEX SEL_NO.

* CHECK AUTHORIZATION
    CALL  FUNCTION 'SXPG_COMMAND_CHECK'
         EXPORTING
              COMMANDNAME                   = COMMAND_LIST-NAME
              OPERATINGSYSTEM               = SY-OPSYS
         EXCEPTIONS
              NO_PERMISSION                 = 1
              COMMAND_NOT_FOUND             = 2
              PARAMETERS_TOO_LONG           = 3
              SECURITY_RISK                 = 4
              WRONG_CHECK_CALL_INTERFACE    = 5
              X_ERROR                       = 6
              TOO_MANY_PARAMETERS           = 7
              PARAMETER_EXPECTED            = 8
              ILLEGAL_COMMAND               = 9
              COMMUNICATION_FAILURE         = 10
              SYSTEM_FAILURE                = 11
              OTHERS                        = 12.

  CASE SY-SUBRC.
    WHEN 0.
      CALL FUNCTION 'SXPG_COMMAND_EXECUTE'
           EXPORTING
               COMMANDNAME                    = COMMAND_LIST-NAME
           TABLES
               EXEC_PROTOCOL                  = EXEC_PROTOCOL
           EXCEPTIONS
               NO_PERMISSION                  = 1
               COMMAND_NOT_FOUND              = 2
               PARAMETERS_TOO_LONG            = 3
               SECURITY_RISK                  = 4
               WRONG_CHECK_CALL_INTERFACE     = 5
               PROGRAM_START_ERROR            = 6
```

```
                PROGRAM_TERMINATION_ERROR       = 7
                X_ERROR                         = 8
                PARAMETER_EXPECTED              = 9
                TOO_MANY_PARAMETERS             = 10
                ILLEGAL_COMMAND                 = 11
                WRONG_ASYNCHRONOUS_PARAMETERS   = 12
                CANT_ENQ_TBTCO_ENTRY            = 13
                JOBCOUNT_GENERATION_ERROR       = 14
                OTHERS                          = 15.

    IF SY-SUBRC EQ 0.
      WRITE:/ COMMAND_LIST-NAME, 'RAN SUCCESSFULLY'.
    ELSE.
      WRITE:/ 'ERROR WITH COMMAND', COMMAND_LIST-NAME.
    ENDIF.

    WHEN 1.
      WRITE:/'YOU ARE NOT AUTHORIZED TO RUN', COMMAND_LIST-NAME.

     WHEN OTHERS.
      WRITE:/'ERROR WITH FUNCTION WITH COMMAND', COMMAND_LIST-NAME.
    ENDCASE.
  ENDIF. "POPUP_WITH_TABLE_DISPLAY
ENDIF.   "SXPG_COMMAND_LIST_GET
```

See Also

SXPG_COMMAND_EXECUTE

SXPG_COMMAND_CHECK

Summary

Checks authorisation to run a command.

Description

Checks user's authorisation to execute the command on the host system with the specified arguments, and then carry out the command.

Parameters

```
IMPORTING
     ADDITIONAL_PARAMETERS    Arguments for the external command
     COMMANDNAME              Name of the external command, as defined in the maintenance
                              function (transaction SM69)
     OPERATINGSYSTEM          The host system on which the command is to be executed
                              OPERATINGSYSTEM value is defined as part of the command
                              definition (transaction SM69)
     TARGETSYSTEM             System upon which the command is to run
```

```
EXPORTING
     PROGRAMNAME              Name and path to the command to be executed. Arguments are
                              not included
     DEFINED_PARAMETERS       Returns argument string from the command definition
     ALL_PARAMETERS           Returns complete argument string for the command,
                              consisting of ADDITIONAL_PARAMETERS and DEFINED_PARAMETERS
```

Example

See SXPG_CALL_SYSTEM

SXPG_COMMAND_EXECUTE

Summary

Executes a command on remote system.

Description

Checks user's authorisation to run the command and if the authorisation check is successful, then the command is executed on the target host system. Commands are defined with SM69 and can be tested with SM49. Use with caution!

Parameters

```
IMPORTING
     COMMANDNAME              Name of the external command, as defined in the maintenance
                              function (transaction SM69)
     OPERATINGSYSTEM          The host system on which the command is to be executed
                              OPERATINGSYSTEM value is defined as part of the command
                              definition (transaction SM69)
     TARGETSYSTEM             System upon which the command is to run
     STDOUT                   Log STDOUT output from the external command in parameter
                              EXEC_PROTOCOL
     STDERR                   Log STDERR output from the external command in parameter
                              EXEC_PROTOCOL
     TERMINATIONWAIT          Wait for termination of external command and event log
     TRACE                    Trace execution through CALL "writetrace" and through the
                              local trace function of the command
     ADDITIONAL_PARAMETERS    Arguments for the external command
     ABAPPROG                 ABAP program name
     ABAPFORM                 ABAP procedure (FORM) in program ABAPPROG
EXPORTING
     STATUS                   Returns the final status of the execution of the external
                              command:
                              Value        Meaning
                              0            Command started and completed successfully
                              E            Command failed
TABLES
     EXEC_PROTOCOL            Contains output of the command and host system.
```

Example

See SXPG_CALL_SYSTEM

See Also

SXPG_CALL_SYSTEM

SXPG_COMMAND_LIST_GET

Summary

Reads a list of the external commands that have been defined in R/3 into an internal table.

Description

You can loop through the table to select a command, or offer the list to your user for selection. You can pass the selection onto SXPG_COMMAND_EXECUTE for an authorisation check and execution of the command.

Parameters

```
IMPORTING
    COMMANDNAME             Name of the external command, as defined in the maintenance
                            function (transaction SM69).
    OPERATINGSYSTEM         The host system on which the command is to be executed.
                            OPERATINGSYSTEM value is defined as part of the command
                            definition (transaction SM69).
    TARGETSYSTEM            System upon which the command is to run.
TABLES
    COMMAND_LIST            Contains list of selected commands in the format shown in
                            transaction SM49 or SM69.
```

Example

See SXPG_CALL_SYSTEM

TERMINAL_ID_GET

Summary

Returns the IP address and terminal ID.

Parameters

```
EXPORTING
    USERNAME            SAP user
```

```
IMPORTING
    TERMINAL            IP address and terminal information
```

Example

```
REPORT ZEXAMPLE.
DATA TERMINAL  LIKE  USR41-TERMINAL.

CALL FUNCTION 'TERMINAL_ID_GET'
    EXPORTING
       USERNAME               = SY-UNAME
    IMPORTING
       TERMINAL               = TERMINAL
    EXCEPTIONS
       MULTIPLE_TERMINAL_ID   = 1
       NO_TERMINAL_FOUND      = 2
       OTHERS                 = 3.

IF SY-SUBRC EQ 0.
  WRITE:/'USER', SY-UNAME, 'IS USING TERMINAL', TERMINAL.
ELSE.
  WRITE:/'ERROR IN FUNCTION'.
ENDIF.
```

See Also

ARFC_GET_TID, TH_USER_INFO

TH_DELETE_USER

Summary

Logoffs a user.

Description

Results are similar to using transaction SM04. Does not actually delete the user's ID from SAP!

Parameters

```
EXPORTING
    USER                SAP username to logoff the system
    CLIENT              SAP client to log users off
```

Example

```
REPORT ZEXAMPLE.

PARAMETERS V_BNAME LIKE SY-UNAME.
```

```
CALL FUNCTION 'TH_DELETE_USER'
    EXPORTING
        USER             = V_BNAME
        CLIENT           = SY-MANDT
    EXCEPTIONS
        AUTHORITY_ERROR  = 1
        OTHERS           = 2.

IF SY-SUBRC EQ 0.
  WRITE:/ V_BNAME, 'LOGGED OFF THE SYSTEM'.
ELSE.
  WRITE:/ V_BNAME, 'NOT LOGGED OFF THE SYSTEM'.
ENDIF.
```

TH_ENVIRONMENT

Summary

Gets values in SAP environment variables.

Description

Returns the values of SAP systems environment variables.

Parameters

```
TABLES

      ENVIRONMENT  Holds names and values of environment variables.
```

Example

```
REPORT ZEXAMPLE.
DATA: BEGIN OF IENV OCCURS 0.
      INCLUDE STRUCTURE  THENV.
DATA: END OF IENV.

DATA: V_VAR(20), V_VALUE(235).

CALL  FUNCTION 'TH_ENVIRONMENT'
      TABLES
      ENVIRONMENT = IENV.

WRITE:/ 'VARIABLE', 30 'VALUE'.
ULINE.
LOOP AT IENV.
  SPLIT IENV-LINE AT '=' INTO V_VAR V_VALUE.
  WRITE:/ V_VAR, 30 V_VALUE.
  CLEAR: V_VAR, V_VALUE.
ENDLOOP.
```

TH_REMOTE_TRANSACTION

Summary

Runs a transaction on a remote server.

Description

The transaction may be run as a BDC by filling in the BDCTAB table parameter.

Parameters

```
EXPORTING
     TCODE                    Transaction code to be run
     DEST                     RFC-capable server on which to run transaction
IMPORTING
     COMM_MESSAGE             Error messages
     SYST_MESSAGE             Success messages
TABLES
     BDCTAB                   BDC data
```

Example

```
REPORT ZEXAMPLE.
DATA: BEGIN OF SERVER_LIST OCCURS 0 .
        INCLUDE STRUCTURE MSXXLIST .
DATA: END OF SERVER_LIST .

DATA: MESSAGE_SERVER LIKE MSXXLIST-HOST,
      V_LINE TYPE I,
      COMM_MESSAGE(256),
      SYST_MESSAGE(256).

CALL  FUNCTION 'TH_SERVER_LIST'
      TABLES
          LIST    = SERVER_LIST
      EXCEPTIONS
          OTHERS  = 1.

IF SY-SUBRC NE 0.
  CALL 'C_SAPGPARAM' ID 'NAME'  FIELD 'RDISP/MSHOST'
                     ID 'VALUE' FIELD MESSAGE_SERVER.

  SERVER_LIST-NAME = MESSAGE_SERVER.
  APPEND SERVER_LIST.
ENDIF.

DESCRIBE TABLE SERVER_LIST LINES V_LINE.
IF V_LINE GE 1.
  READ TABLE SERVER_LIST INDEX 1.         "RUN TXN ON FIRST SERVER FOUND
```

```
  CALL FUNCTION 'TH_REMOTE_TRANSACTION'
       EXPORTING
           TCODE         = 'SE37'
           DEST          = SERVER_LIST-NAME
       IMPORTING
           COMM_MESSAGE  = COMM_MESSAGE
           SYST_MESSAGE  = SYST_MESSAGE.

  IF COMM_MESSAGE <> SPACE.
    WRITE:/ COMM_MESSAGE.
  ELSE.
    WRITE:/ SYST_MESSAGE.
  ENDIF.
ELSE.
  WRITE:/ 'NO SERVERS FOUND'.
ENDIF.
```

See Also

ABAP4_CALL_TRANSACTION, HLP_MODE_CREATE, TRANSACTION_CALL

TH_SERVER_LIST

Summary

List of RFC servers.

Parameters

```
TABLES
    LIST                  List of servers
```

Example

See TH_REMOTE_TRANSACTION

See Also

CAT_CHECK_RFC_DESTINATION

TH_USER_INFO

Summary

Returns information about user.

Parameters

```
IMPORTING
    HOSTADDR                    IP address of the frontend computer
    TERMINAL                    Terminal name of the user
    ACT_SESSIONS                Current number of open sessions
    MAX_SESSIONS                Maximum number of sessions allowed
    MY_SESSION                  Currently active session
    MY_INTERNAL_SESSION         Current internal session
    TASK_STATE                  Task state
```

Example

```
REPORT ZEXAMPLE.
TABLES: MSXXLIST.

DATA:  HOSTADDR                 LIKE   MSXXLIST-HOSTADR,
       TERMINAL(255),
       ACT_SESSIONS             LIKE   SMO4DIC-COUNTER,
       MAX_SESSIONS             LIKE   SMO4DIC-COUNTER,
       MY_SESSION               LIKE   SMO4DIC-COUNTER,
       MY_INTERNAL_SESSION      LIKE   SMO4DIC-COUNTER,
       TASK_STATE               LIKE   SMO4DIC-COUNTER.

DATA:    DOT           VALUE '.',
         IP1           TYPE I,
         IP2           TYPE I,
         IP3           TYPE I,
         IP4           TYPE I,
         C_IP1(3)      TYPE C,
         C_IP2(3)      TYPE C,
         C_IP3(3)      TYPE C,
         C_IP4(3)      TYPE C,
         V_IPADDR(15).

CALL  FUNCTION 'TH_USER_INFO'
      IMPORTING
           HOSTADDR             = HOSTADDR
           TERMINAL             = TERMINAL
           ACT_SESSIONS         = ACT_SESSIONS
           MAX_SESSIONS         = MAX_SESSIONS
           MY_SESSION           = MY_SESSION
           MY_INTERNAL_SESSION  = MY_INTERNAL_SESSION
           TASK_STATE           = TASK_STATE.

* FORMAT THE IP ADDRESS
C_IP1 = IP1 = HOSTADDR(1).
C_IP2 = IP2 = HOSTADDR + 1(1).
C_IP3 = IP3 = HOSTADDR + 2(1).
C_IP4 = IP4 = HOSTADDR + 3(1).
CONCATENATE C_IP1 DOT C_IP2 DOT C_IP3 DOT C_IP4 INTO V_IPADDR.
CONDENSE V_IPADDR NO-GAPS.
```

```
WRITE:/ SY-UNAME, 'INFORMATION:'.
ULINE.

WRITE:/ 'IP ADDRESS:', 20 V_IPADDR,
      / 'TERMINAL:', 20 TERMINAL,
      / 'OPEN SESSIONS:', 20 ACT_SESSIONS,
      / 'MAXIMUM SESSIONS:', 20 MAX_SESSIONS,
      / 'CURRENT SESSION:', 20 MY_SESSION,
      / 'INTERNAL SESSION:', 20 MY_INTERNAL_SESSION,
      / 'TASK STATE:', 20 TASK_STATE.
```

See Also

ARFC_GET_TID, TERMINAL_ID_GET, TH_USER_LIST

TH_USER_LIST

Summary

Displays users logged onto a server.

Parameters

```
TABLES
    LIST                  List of users logged into the system
```

Example

```
REPORT ZEXAMPLE.
DATA: BEGIN OF IUSRTBL OCCURS 0.
        INCLUDE STRUCTURE UINFO.
DATA: END OF IUSRTBL.

DATA NUMBER_OF_USERS TYPE I.

CALL  FUNCTION 'TH_USER_LIST'
      TABLES
          LIST    = IUSRTBL
      EXCEPTIONS
          OTHERS  = 1.

DESCRIBE TABLE IUSRTBL LINES NUMBER_OF_USERS.
WRITE:/ NUMBER_OF_USERS, 'ARE LOGGED IN NOW'.

LOOP AT IUSRTBL.
  WRITE:/ IUSRTBL-BNAME.
ENDLOOP.
```

See Also

TH_USER_INFO

TRANSACTION_CALL

Summary

Initiates a transaction in a separate window.

Description

An extremely simple function, which issues a CALL TRANSACTION command. This function cannot process exceptions (so will crash if a non-existent transaction code is passed into it). Use one of the alternative functions in the "See Also:" section instead.

Parameters

```
EXPORTING
    TRANSACTION_NAME        Transaction code to call
```

Example

```
REPORT ZEXAMPLE.

CALL  FUNCTION 'TRANSACTION_CALL'
      EXPORTING
         TRANSACTION_NAME  = 'SE37'       "FUNCTION MODULES
      EXCEPTIONS
         OTHERS            = 1.

WRITE:/ 'FUNCTION CALLED'.
```

See Also

ABAP4_CALL_TRANSACTION, HLP_MODE_CREATE, TH_REMOTE_TRANSACTION

USER_EXISTS

Summary

Checks whether user ID is valid.

Parameters

```
EXPORTING
     BNAME            User ID
     CLIENT           Client to check for user ID
IMPORTING
     LOCKED           Flag whether user ID is locked
```

Example

See F4_USER

See Also

F4_USER

WS_EXECUTE

Summary

Calls an external program from ABAP.

Description

Replaced by GUI_EXEC.

Parameters

```
EXPORTING
     COMMANDLINE      File to load with external application
     PROGRAM          Path and name of external application
```

Example

```
REPORT ZEXAMPLE.
DATA: V_PGM(100)    VALUE 'C:\PROGRAM FILES\MICROSOFT OFFICE\OFFICE\WINWORD.EXE',
     V_FNAME        LIKE RLGRAP-FILENAME VALUE 'C:\DOCUMENT.DOC'.

CALL FUNCTION 'WS_EXECUTE'
     EXPORTING
         COMMANDLINE          = V_FNAME
         PROGRAM              = V_PGM
     EXCEPTIONS
         FRONTEND_ERROR       = 1
         NO_BATCH             = 2
         PROG_NOT_FOUND       = 3
         ILLEGAL_OPTION       = 4
         GUI_REFUSE_EXECUTE   = 5
         OTHERS               = 6.
```

```
IF SY-SUBRC NE 0.
  WRITE:/ V_FNAME, 'NOT OPENED WITH', V_PGM.
ELSE.
  WRITE:/ 'EXTERNAL APPLICATION CALLED SUCCESSFULLY'.
ENDIF.
```

See Also

SXPG_COMMAND_EXECUTE, GUI_RUN, GUI_EXEC

WS_QUERY

Summary

Executes query function on frontend.

Description

Replaced by GUI_GET_DESKTOP_INFO. The query command "OS" under UNIX returns the output of the UNIX command "uname".

Parameters

```
EXPORTING
       ENVIRONMENT          Environment variable for EN query, e.g. TEMP
       FILENAME             Variable requirement for FC\FE\FL\FT\DE queries
       QUERY                Query command:
                            Value        Meaning
                            CD           Current directory
                            DE           Directory exists
                            EN           Value of environment variable
                            FC           Filename creator (Apple only)
                            FE           Filename exists
                            FL           File length
                            FT           Filetype and status (UNIX only)
                            GM           GMUX version (internal use only)
                            LF           Linefeed code
                            OS           Operating system of presentation server
                            RC           Last return code
                            WI           Window ID graphics program
                            WS           Windows System of presentation server
                            XP           Execution path (UNIX only)
                            WINID        Variable requirement for WI query
IMPORTING
       RETURN               Result of the query:
                            Command      Return value (non-exhaustive)
                            OS           AIX
                            DOS
                                         HP-UX
                                         MAC
                                         NT
                                         OS2
```

```
                                    OS\2
                                    OSF1
                                    SINIX
                                    SunOS
                                    ULTRIX
                                    VMS
                          WS        MC              (Macintosh)
                                    MF              (UNIX-Motif)
                          NT        (Windows NT)
                          PM        (OS/2)
                          WN        (DOS\Windows 3.xx)
                                    WN32   (32-bit Windows)
                                    WN32_95
                                    WN32_98
                                    WN32_S
```

Example

```
REPORT ZEXAMPLE.
DATA: WINSYS(20),
      OSSYS(20).

START-OF-SELECTION.
  CALL  FUNCTION 'WS_QUERY'
        EXPORTING
             QUERY                = 'WS'
        IMPORTING
             RETURN               = WINSYS
        EXCEPTIONS
             INV_QUERY            = 1
             NO_BATCH             = 2
             FRONTEND_ERROR       = 3
             OTHERS               = 4.

  CALL  FUNCTION 'WS_QUERY'
        EXPORTING
             QUERY                = 'OS'
        IMPORTING
             RETURN               = OSSYS
        EXCEPTIONS
             INV_QUERY            = 1
             NO_BATCH             = 2
             FRONTEND_ERROR       = 3
             OTHERS               = 4.

  PERFORM WSQUERY_CODE CHANGING WINSYS OSSYS.

  WRITE:/ 'RUNNING', WINSYS, 'ON THE', OSSYS, 'OPERATING SYSTEM'.

*&---------------------------------------------------------------------*
*& FORM WSQUERY_CODE
*&---------------------------------------------------------------------*
FORM WSQUERY_CODE CHANGING  P_WINSYS
                            P_OSSYS.
```

```
* READABLE TEXTS:
  CASE P_WINSYS.
    WHEN 'MC'.
      P_WINSYS = 'MACINTOSH'.
    WHEN 'MF'.
      P_WINSYS = 'UNIX'.
    WHEN 'NT'.
      P_WINSYS = 'WINDOWS NT'.
    WHEN 'PM'.
      P_WINSYS = 'OS\2'.
    WHEN 'WN'.
      P_WINSYS = 'WINDOWS 3.XX'.
    WHEN 'WN32'.
      P_WINSYS = '32-BIT WINDOWS'.
    WHEN 'WN32_95'.
      P_WINSYS = 'WINDOWS 95'.
    WHEN 'WN32_98'.
      P_WINSYS = 'WINDOWS 98'.
    WHEN OTHERS.
      P_WINSYS = 'UNKNOWN'.
  ENDCASE.

  CASE P_OSSYS.
    WHEN 'AIX'.
      P_OSSYS = 'AIX'.
    WHEN 'DOS'.
      P_OSSYS = 'DOS'.
    WHEN 'HP-UX'.
      P_OSSYS = 'HP-UX'.
    WHEN 'MAC'.
      P_OSSYS = 'MAC'.
    WHEN 'NT'.
      P_OSSYS = 'NT'.
    WHEN 'OS2' OR 'OS\2'.
      P_OSSYS = 'OS\2'.
    WHEN 'OSF1'.
      P_OSSYS = 'OSF1'.
    WHEN 'SINIX'.
      P_OSSYS = 'SINIX'.
    WHEN 'SUNOS'.
      P_OSSYS = 'SUNOS'.
    WHEN 'ULTRIX'.
      P_OSSYS = 'ULTRIX'.
    WHEN 'VMS'.
      P_OSSYS = 'VMS'.
    WHEN OTHERS.
      P_OSSYS = 'UNKNOWN'.
  ENDCASE.
ENDFORM. " WSQUERY_CODE
```

See Also

GUI_GET_DESKTOP_INFO