z/OS
3.2

ISPF Planning and Customizing

GC19-3623-70

Note

Before using this information and the product it supports, read the information in "Notices" on page 321.

This edition applies to IBM® z/OS® 3.2 (5655-ZOS) and to all subsequent releases and modifications until otherwise indicated in new editions.

Last updated: 2025-09-30

© Copyright International Business Machines Corporation 1984, 2025.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Preface

The Interactive System Productivity Facility (ISPF) product assists in program development. It is designed to take advantage of the characteristics of IBM display terminals, and to increase programmer productivity in an interactive environment.

About this document

This document provides guidance and reference information for the planning and customizing of ISPF under z/OS with the Time Sharing Option Extensions (TSO/E). It consists of three parts:

Part 1. Planning and customizing
Contains detailed planning information and procedures you need to install and customize ISPF. It explains how to specify configuration options, modify the distributed menus and options, and tune ISPF performance.

Part 2. Reference
Provides a lookup reference for static information such as the ISPF Configuration Table keywords and values, exits, PDF translation tables, dialog development model listings, and so on.

Part 3. Appendixes
This part provides information about ISPF enqueue processing for data integrity, accessibility, notices, and index.

Who should use this document

z/OS ISPF Planning and Customizing is designed for system programmers or other people whose responsibilities include installing, customizing, and tuning ISPF. You should be familiar with MVS, ISPF concepts and terminology as described in the *z/OS ISPF Dialog Developer's Guide and Reference*, and with the System Modification Program Extended (SMP/E).

What is in this document?

This publication contains:

- Chapter 1, "z/OS 3.1 ISPF planning," on page 3 describes planning procedures you need to know before installing ISPF.
- Chapter 2, "The ISPF Configuration Table," on page 9 describes the ISPF Configuration Utility.
- Chapter 3, "Customizing ISPF," on page 37 describes how to modify items that affect both the DM and PDF components to suit the needs of your installation.
- Chapter 4, "Improving ISPF performance," on page 93 describes how to tune ISPF to reduce system load and improve response times for interactive users.
- Chapter 5, "Customizing DM," on page 105 describes how to modify items that affect the DM function of ISPF to suit the needs of your installation.
- Chapter 6, "ISPF installation-wide exits," on page 135 describes how you can write exit routines to collect system-related information.
- Chapter 7, "Customizing PDF," on page 141 describes how to modify the distributed release of PDF to suit the needs of your installation.
- Chapter 8, "PDF installation-wide exits," on page 191 describes the installation-wide exits that allow you to customize the behavior of ISPF utilities such as Edit, Browse, Print, and Compress, as well as data set lists and member lists.
- Chapter 9, "ISPF Configuration Table keywords and values," on page 215 lists the keywords in the ISPF Configuration Table and their allowable and default values.
- Chapter 10, "Exits," on page 255 lists ISPF the installation-wide exits.

- Chapter 11, “PDF translation tables,” on page 273 lists the PDF translation tables.
- Chapter 12, “Dialog development model listings,” on page 281 lists all of the edit models included in the ISPF SKELS library.
- Chapter 13, “Programming interface macros for customers,” on page 311 lists the macros provided by ISPF as programming interfaces.
- Chapter 14, “ISPF data set descriptions,” on page 313 lists the target and distribution data sets used by ISPF, and their contents.
- Appendix A, “ISPF enqueue processing for data integrity,” on page 315 explains how ISPF ensures data integrity.

How to read the syntax diagrams

The syntactical structure of commands described in this document is shown by means of syntax diagrams.

Figure 1 on page xi shows a sample syntax diagram that includes the various notations used to indicate such things as whether:

- An item is a keyword or a variable.
- An item is required or optional.
- A choice is available.
- A default applies if you do not specify a value.
- You can repeat an item.

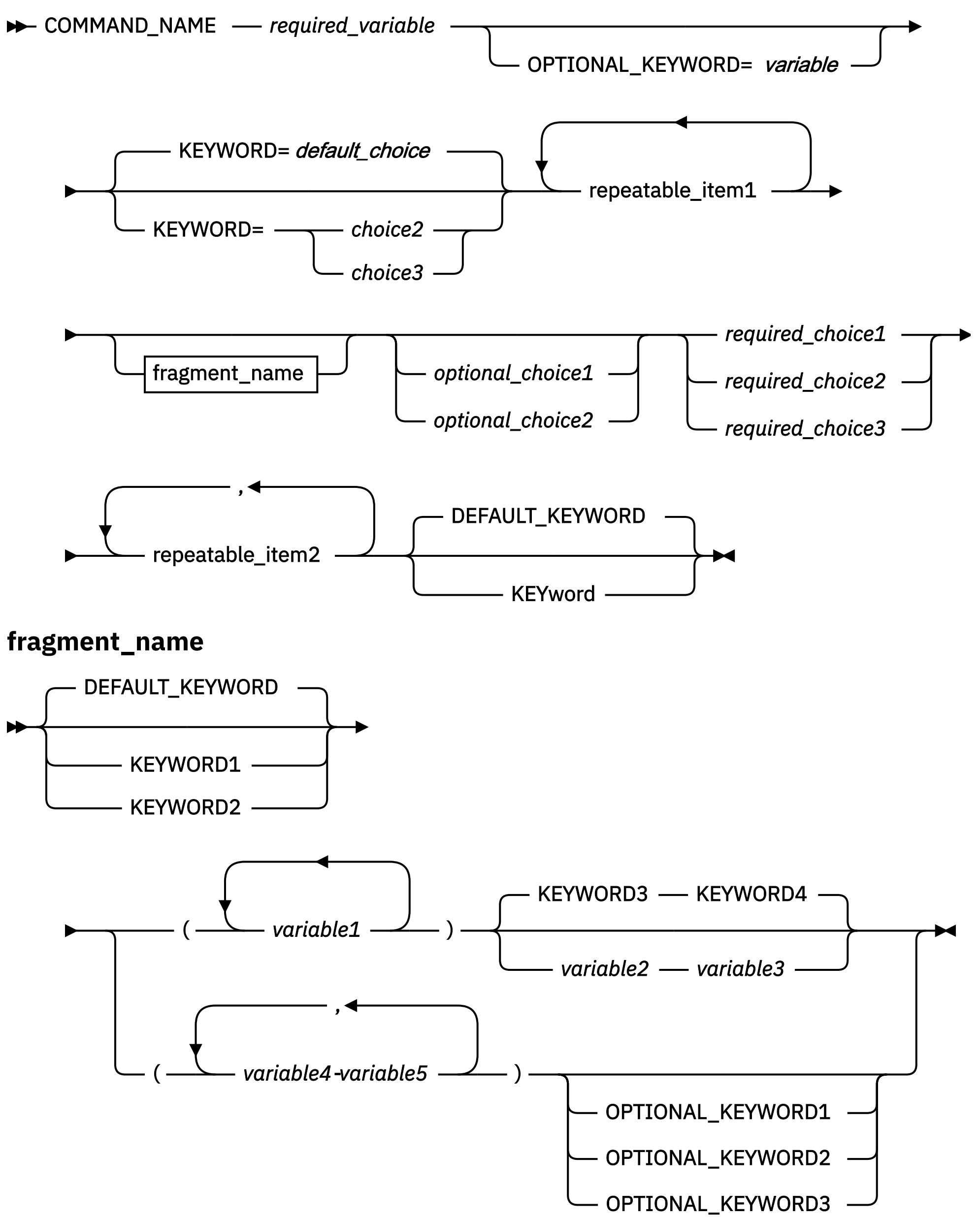


Figure 1. Sample syntax diagram

Here are some tips for reading and understanding syntax diagrams:

Order of reading

Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

The ►►─ symbol indicates the beginning of a statement.

The ─► symbol indicates that a statement is continued on the next line.

The ►─ symbol indicates that a statement is continued from the previous line.

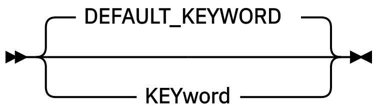
The ─►◄ symbol indicates the end of a statement.

Keywords

Keywords appear in uppercase letters.

►►─ COMMAND_NAME ─►◄

Sometimes you only need to type the first few letters of a keyword, The required part of the keyword appears in uppercase letters.

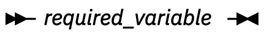


In this example, you could type "KEY", "KEYW", "KEYWO", "KEYWOR" or "KEYWORD".

The abbreviated or whole keyword you enter must be spelled exactly as shown.

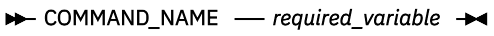
Variables

Variables appear in lowercase letters. They represent user-supplied names or values.



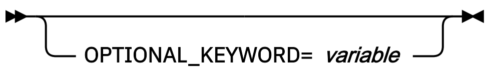
Required items

Required items appear on the horizontal line (the main path).



Optional items

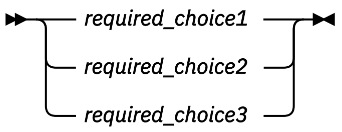
Optional items appear below the main path.



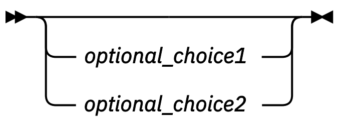
Choice of items

If you can choose from two or more items, they appear vertically, in a stack.

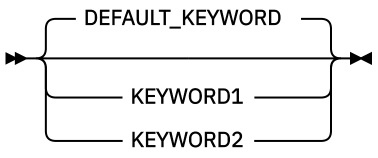
If you *must* choose one of the items, one item of the stack appears on the main path.



If choosing one of the items is optional, the entire stack appears below the main path.

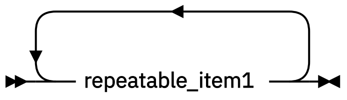


If a default value applies when you do not choose any of the items, the default value appears above the main path.

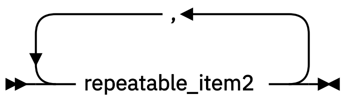


Repeatable items

An arrow returning to the left above the main line indicates an item that can be repeated.

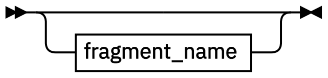


If you need to specify a separator character (such as a comma) between repeatable items, the line with the arrow returning to the left shows the separator character you must specify.



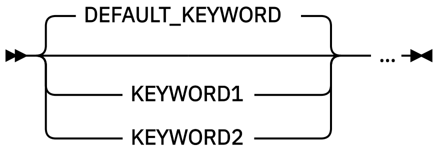
Fragments

Where it makes the syntax diagram easier to read, a section or *fragment* of the syntax is sometimes shown separately.



⋮

fragment_name



z/OS information

This information explains how z/OS references information in other documents and on the web.

When possible, this information uses cross-document links that go directly to the topic in reference using shortened versions of the document title. For complete titles and order numbers of the documents for all products that are part of z/OS, see *z/OS Information Roadmap*.

How to provide feedback to IBM

We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information. For more information, see How to send feedback to IBM.

Summary of changes

This information includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations for the current edition are indicated by a vertical line to the left of the change.

Note: IBM z/OS policy for the integration of service information into the z/OS product documentation library is documented on the z/OS Internet Library under IBM z/OS Product Documentation Update Policy (www.ibm.com/docs/en/zos/latest?topic=zos-product-documentation-update-policy).

Summary of changes for z/OS 3.2

The following content is new, changed, or no longer included in z/OS 3.2.

New

The following content is new.

September 2025 release

- None.

Changed

The following content is changed.

September 2025 release

- "Build SMP/E USERMOD" on page 34 is updated with the FMID for z/OS 3.2.
- "Edit-related settings" on page 220 is updated in support of enhancements to PDSE V2 member generation.

Deleted

The following content is deleted.

September 2025 release

- None.

Summary of changes for z/OS 3.1

The following changes are made for z/OS 3.1.

New

September 2023 release

- None.

Changed

October 2024 refresh

- "Build SMP/E USERMOD" on page 34 is updated with the correct FMID for z/OS 3.1 as it was previously incorrect.

September 2023 release

- Externalize Edit HILITE settings to ISPF z variables, see the following topics:

  - "Action bars and extended color in Edit" on page 142

Deleted

September 2023 release

- None.

What's in the library?

You can order the ISPF books using the numbers provided below.

Title	**Order Number**
z/OS ISPF Dialog Developer's Guide and Reference	SC19-3619–40
z/OS ISPF Dialog Tag Language Guide and Reference	SC19-3620–40
z/OS ISPF Edit and Edit Macros	SC19-3621–40
z/OS ISPF Messages and Codes	SC19-3622–40
z/OS ISPF Planning and Customizing	GC19-3623–40
z/OS ISPF Reference Summary	SC19-3624–40
z/OS ISPF Software Configuration and Library Manager Guide and Reference	SC19-3625–40
z/OS ISPF Services Guide	SC19-3626–40
z/OS ISPF User's Guide Vol I	SC19-3627–40
z/OS ISPF User's Guide Vol II	SC19-3628–40

© Copyright IBM Corp. 1984, 2025

Part 1. Planning and customizing

Chapter 1. z/OS 3.1 ISPF planning

Application dialogs that were created to run under earlier versions of ISPF will run under z/OS 3.1 ISPF with no change.

All the components of ISPF (DM, PDF, and SCLM) are considered one element in all releases of z/OS. Attempting to run one of the components from one release of z/OS with a component from a different release of z/OS is not supported.

As documented in the *Choosing the z/OS base and optional features* topic under *Preparing the target system* in *z/OS Planning for Installation*, because the z/OS base elements and optional features are integrated into a single package with compatible service levels, you must install, with few exceptions, the entire z/OS product. Using ISPF from one z/OS release with z/OS base elements or optional features from another z/OS release is not supported.

Hardware and software requirements

For hardware and software requirements for ISPF, refer to *z/OS Planning for Installation*, number GA22–7504.

Notes on migrating from previous releases to z/OS 3.1 ISPF

For information about migrating from a previous supported release of ISPF, refer to *z/OS Upgrade Workflow*.

The z/OS 3.1 ISPF data sets are all named ISP.SISP*xxxx* and ISP.AISP*xxxx*.

Using SMP to install z/OS 3.1 ISPF deletes previous releases of ISPF.

If you have customized the ISPF configuration table for your installation, copy the customized configuration load module (ISPCFIGU) to a location that is in the standard MVS search sequence, or allocated to ISPLLIB, so that z/OS 3.1 ISPF will use the customized configuration. After you are running z/OS 3.1 ISPF, you can use the ISPF Configuration utility to set any new configuration options.

If you are running with an ISPLLIB DD statement and the z/OS 3.1 ISPF SISPLOAD and SISPLPA are not in the current LPA or the linklist, you must include the z/OS 3.1 ISPF SISPLOAD and SISPLPA data sets in the ISPLLIB concatenation and remove any load library data sets from a previous version or release of ISPF.

If you are using the z/OS UNIX Directory List utility, include the Language Environment® run-time library data sets SCEERUN and SCEERUN2 in the STEPLIB if they are not already in the linklist.

Remove the statement SUBSYS SUBNAME(ISPF) INITRTN(ISPDTSSI) from your IEFSSN*xx* member in SYS1.PARMLIB, if you added it in a previous release to support TSO line mode. If you have changed the logon procedure to use alternate entry point IKJEFT1I to support TSO line mode, you can change the EXEC statement to use IKJEFT01, or whatever your site requires.

Installation, maintenance, and migration diagnostics

This information describes common situations you may encounter during installation, maintenance, and migration.

Problem: Unpredictable results such as 0C4 or 0C1 abends upon invocation of ISPF.

Cause:
This problem can occur when internal LINKs and LOADs are used in conjunction with the ISPF ISPLLIB DCB.

Solution:
When using STEPLIB to test new maintenance and an ISPLLIB is allocated, data sets allocated to STEPLIB should also be allocated to ISPLLIB.

Problem: New levels of code residing in STEPLIB do not appear to be executed. This includes changes to customer applications and changes made by PTFs.

Cause:
This problem can occur when internal LINKS and LOADs are used in conjunction with the ISPF ISPLLIB DCB.

Solution:
When using STEPLIB to test new maintenance and an ISPLLIB is allocated, data sets allocated to STEPLIB should also be allocated to ISPLLIB.

Problem: Abend code 806 for ISPLINK or Abend code 0C1 in user application.

Cause:
During installation, the SMP/E install logic for ISPF deletes the existing ISPLINK load module.

Solution:
Relink-edit ISPLINK to user application.

Problem: Abends in load modules IGC0009C and IGC0009D.

Cause:
Failure to install ISPF in the same zone as TSO/E will result in the supplied ISPF SVC 93 and SVC 94 exits (ISPSC93, ISPSC94), creating SVC 93 and SVC 94 load modules without the proper link-edit information.

Failure to use the correct version of either IGC0009C or IGC0009D or both.

Solution:
TSO/E, ISPF, and ISPF/PDF must all be installed in the same zone. Verify that the correct versions are copied from a test system to the production system.

Language Environment considerations

The z/OS UNIX Directory List utility include code are compiled with the IBM C®/C++ compiler. Therefore, the Language Environment SCEERUN and SCEERUN2 run-time library data sets must be available to this code through either STEPLIB or the linklist at execution time. Using the usual ISPLLIB allocation will not make the run-time library available to this code.

Software Configuration and Library Manager (SCLM)

Use the Software Configuration and Library Manager (SCLM) to create, control, maintain, and track software components for a project.

The SCLM project database consists of a series of related ISPF libraries (partitioned data sets). These contain source and non-source software components. SCLM project definition and control information is contained in an assembled and linked PROJDEFS data set. SCLM project cross-reference and accounting data sets are VSAM clusters.

SCLM requires RACF® (or equivalent) to protect the SCLM-controlled data sets. Individual developers should have READ authority to all levels of the hierarchy to draw down members from the hierarchy using the SCLM edit option. Those developers who promote the modified members up the hierarchy will need UPDATE authority for the libraries into which they are promoting. The Build Coordinator should have UPDATE access to the libraries containing the non-editable parts (non-editable parts are the output from the build process) and READ access to all levels of the hierarchy to allow the builds to be done.

For detailed information about SCLM, refer to *z/OS ISPF Software Configuration and Library Manager Guide and Reference*.

Maintenance considerations

ISPF is packaged using the features of SMP/E that allow multiple multicultural support language features to be installed in a single target zone. We recommend that you take advantage of this by installing all of the base code and all multicultural support language features in one target zone and one distribution zone. This simplifies the installation of maintenance.

If you do not install your multicultural support features in the same target and distribution zones as the base, SMP/E cannot properly track maintenance (PTFs) that contains REQ or IF REQ statements. When you install PTFs for the multicultural support features that contain REQ statements for PTFs that apply to the base or PTFs for the base that contain IF REQ statements for PTFs that apply to the multicultural support features, you must use this procedure:

1. APPLY the base PTF that is REQed by the multicultural support feature PTF or contains an IF REQ for an multicultural support feature PTF.
2. APPLY the multicultural support feature PTF using BYPASS(REQ).
3. Test the maintenance according to your normal procedures.
4. ACCEPT the base PTF that is REQed by the multicultural support feature PTF or contains and IF REQ for a multicultural support feature PTF.
5. ACCEPT the multicultural support feature PTF using BYPASS(REQ).

PTFs for ISPF do not contain preprocessed panels. If you are using preprocessed panels in your execution data sets, after installing PTFs that contain panels you must preprocess the panels in the PTF into your execution data set to fully install the maintenance.

Load module search order

When using STEPLIB to test new maintenance and an ISPLLIB is allocated, those data sets allocated to STEPLIB that contain ISPF load modules should also be allocated to ISPLLIB. This prevents the possibility of mixed code (production code versus code to be tested).

The exceptions to this search order are the Language Environment run-time library data sets SCEERUN and SCEERUN2. Modules in these data sets are not searched for using the ISPLLIB task library. SCEERUN and SCEERUN2 must be in STEPLIB or LNKLST if you are using the ISPF C/S feature or the z/OS UNIX Directory List utility. See “Language Environment considerations” on page 4 for more information about SCEERUN and SCEERUN2.

For more information about search order, refer to *z/OS ISPF User's Guide Vol I*.

TSO logon procedure

Figure 2 on page 6 is a sample TSO logon procedure (logon proc) to allocate the ISPF data sets required to execute ISPF.

```
//ISPF     PROC
//ISPF     EXEC PGM=IKJEFT01,DYNAMNBR=100
//*
//STEPLIB  DD DSN=ISP.SISPLPA,DISP=SHR
//         DD DSN=ISP.SISPLOAD,DISP=SHR
//         DD DSN=CEE.SCEERUN,DISP=SHR
//         DD DSN=CEE.SCEERUN2,DISP=SHR
//*
//ISPMLIB  DD DSN=ISP.SISPMxxx,DISP=SHR
//         DD DSN=SYS1.SBPXMENU,DISP=SHR
//*
//*SYSHELP DD DSN=ISP.SISPHELP,DISP=SHR
//*
//ISPPLIB  DD DSN=ISP.SISPPxxx,DISP=SHR
//         DD DSN=SYS1.SBPXPENU,DISP=SHR
//*
//ISPSLIB  DD DSN=ISP.SISPSxxx,DISP=SHR
//         DD DSN=ISP.SISPSLIB,DISP=SHR
//*
//ISPTLIB  DD DSN=userid.ISPTABLE,DISP=SHR
//         DD DSN=ISP.SISPTxxx,DISP=SHR
//         DD DSN=SYS1.SBPXTENU,DISP=SHR
//*
//SYSPROC  DD DSN=ISP.SISPCLIB,DISP=SHR
//         DD DSN=SYS1.SBPXEXEC,DISP=SHR
//*
//SYSEXEC  DD DSN=ISP.SISPEXEC,DISP=SHR
//*
//ISPTABL  DD DSN=userid.ISPTABLE,DISP=SHR
//*
//ISPPROF  DD DSN=userid.ISPTABLE,DISP=OLD
//*
```

Figure 2. TSO logon procedure

The default names ISP.SISP*yxxx* are used for the ISPF data sets, where *xxx* represents the national language:

Language	xxx
US English	ENU
Japanese	JPN
Uppercase English	ENP

Note: For information about data sets that are required by other products for them to run under ISPF, see the *z/OS Program Directory* in the z/OS Internet library (www.ibm.com/servers/resourcelink/svc00100.nsf/pages/zosInternetLibrary) and the documentation for the specific product.

This logon procedure assumes that all data sets are cataloged and that SYSEXEC is included in the search order for REXX execs on your system. See *z/OS TSO/E REXX User's Guide* and *z/OS TSO/E Command Reference* for details about controlling the search order for REXX execs.

If you plan to allow the use of multiple ISPF sessions, the user's logon procedure must be configured to allow profile sharing. This option avoids enqueue lock outs and loss of profile updates when the same profile data set is used for concurrent ISPF sessions. With profile sharing enabled, the user's logon procedure is required to allocate ISPF profile data sets with the disposition SHARED, rather than NEW, OLD, or MOD, and the data sets must already exist. Or, these data sets must be temporary data sets. For more information, see "Customizing for profile sharing" on page 43.

If an ISPLLIB is allocated when you use a STEPLIB to test new ISPF maintenance, you must include the data sets allocated to STEPLIB that contain ISPF load modules in the ISPLLIB allocation. This prevents mixing production code with code to be tested. For more information about library search order, refer to the *z/OS ISPF Services Guide*.

The SCEERUN and SCEERUN2 libraries are required in STEPLIB or LNKLST if you plan to use the z/OS UNIX Directory List utility. SCEERUN and SCEERUN2 are the Language Environment run-time library data sets and contain run-time modules required by C/C++ code used by these ISPF features. These libraries must be in STEPLIB or LNKLST for the modules to be loaded at execution time. These modules are not searched for using the ISPLLIB allocation. See "Language Environment considerations" on page 4 for more information.

If you plan to use the z/OS UNIX Directory List utility, these libraries must be allocated as shown in Figure 2 on page 6: SYS1.SBPXMENU, SYS1.SBPXPENU, SYS1.SBPTMENU, SYS1.SBPXEXEC.

You must use a unique user profile data set (userid.ISPTABLE) in the sample logon procedure for each national language. If you do not, the result is mixed language messages and prompts. The data set *userid*.ISPTABLE must have record format FB, LRECL 80, and a block size that is a multiple of 80. You might want to allocate the ISPPROF, ISPTABLE, and ISPTLIB DDs in a REXX exec that is executed by the PARM on the EXEC statement of the logon procedure.

Edit backup and recovery considerations

Edit backup and recovery helps you to recover data that might otherwise be lost. For example, you would use edit recovery to re-establish the edit session at the point of failure after a power outage or system failure.

Edit backup and recovery is enabled when you enter Edit mode and recovery mode is on. You can set recovery mode on or off with the RECOVERY command.

If recovery mode is on, Edit writes data to an *edit recovery data set* which is used if a recovery is required. Some restrictions apply to the type of data set that can be used as an edit recovery data set.

For more information about backup and recovery, see "Edit backup and recovery" on page 143.

Chapter 2. The ISPF Configuration Table

The ISPF Configuration table is a keyword-driven flat file. Each entry is in the format KEYWORD = value. A slash and asterisk (/*) in columns 1 and 2 indicate a comment line.

Use the ISPF Configuration table to change site-wide defaults and to indicate that installation exit routines are provided for some of the ISPF functions. The ISPF functions that allow installation-written exit routines are data set allocation, print utility, data set compression, the data set list utility, member list filter, and data set name change. ISPF checks the configuration table to determine, first, if exit routines are provided, and second, whether those routines are programs or CLISTs. If you specify both a CLIST and a program, ISPF uses the program.

A default configuration load module (ISPCFIG) is included with ISPF. It is used if you choose to make no customizations for your installation. If you do want to create your own configuration table, simply use the ISPF Configuration utility to create or modify the settings and regenerate the keyword file. You can then convert the keyword file into a configuration table load module called ISPCFIGU. Optionally, you can use the same file to create load module called ISPCFIGV for VSAM support. The converted load module must be placed in the standard MVS search sequence or allocated to ISPLLIB for ISPF to use the values specified. For more information about the ISPF Configuration utility, see "The ISPF Configuration utility" on page 9.

It is recommended that you always use the latest release as the common configuration, though older releases will work with the newest table. Newer releases will also work with an older table, but the ability to configure the settings added in the latest release is lost.

Related references

Chapter 9, "ISPF Configuration Table keywords and values," on page 215

The ISPF Configuration utility

The ISPF Configuration utility enables you to:

- modify the configuration settings saved in the keyword files
- generate a configuration table load module or SMP/E USERMOD for use on your ISPF system
- convert existing ISPF configuration table assembler files from earlier releases of ISPF (SAMPLIB member ISRCNFIG) into the keyword file format
- convert an existing ISPF configuration table load module into the keyword file format

The ISPF Configuration utility sets two types of values: system-wide values and user-specific values. System-wide values are values that are used for all users and are re-read from the Configuration Table at the beginning of each ISPF session. For example, the PDF Exits and PDF Default Unit are system-wide values.

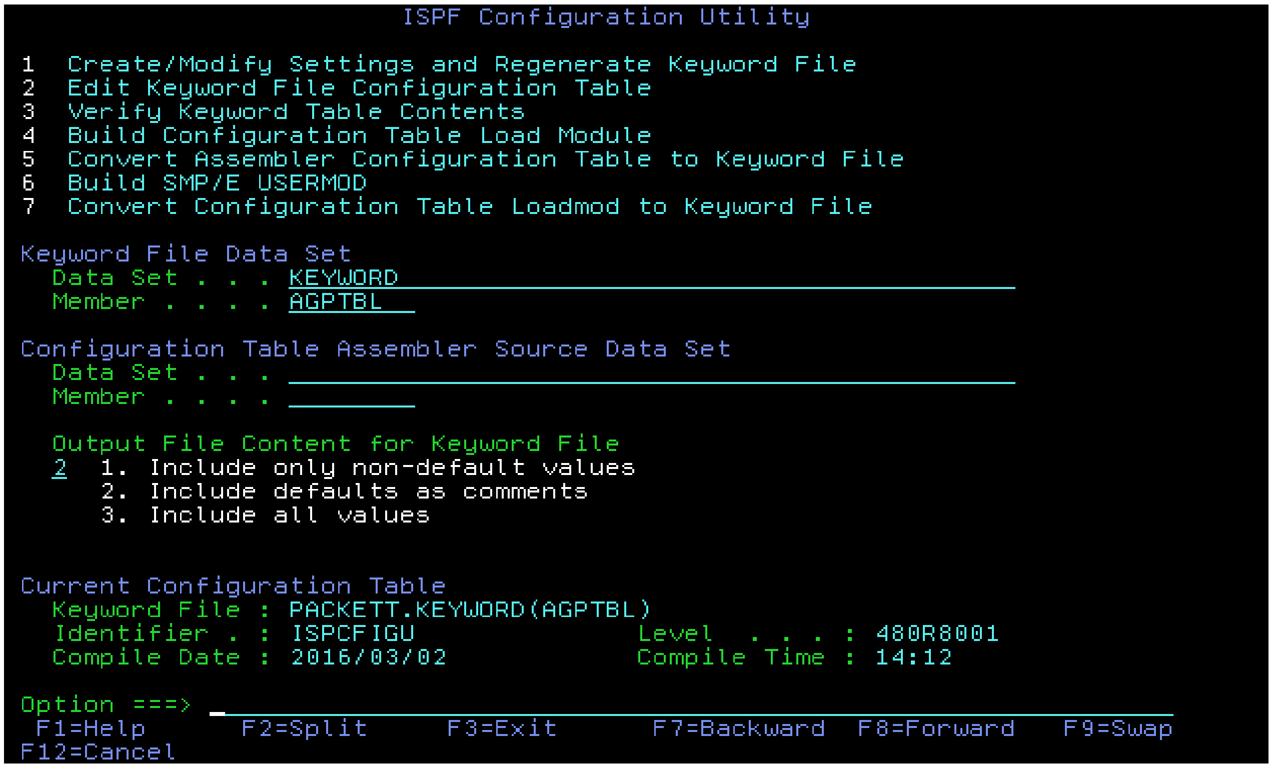
User-specific values are used as initial values when a new profile is created. For example, if a user has an ISPSPROF or Edit Profile, it is used instead of the Configuration Table values. These profiles hold user-specific values.

As system administrator you can, if you choose, ensure that even user-specific values are set according to your specifications. Some of the user-specific Edit Profile fields have corresponding FORCE fields. The FORCE fields enable you to require the user to use the specified configuration values even if the user already has an edit profile. If the user attempts to modify one of the "forced" values, an error message is displayed. Some of the user-specific ISPSPROF fields have corresponding RESET fields. The RESET fields enable you to reset these values once for each user. The reset is done when the Sitewide Defaults Version Level is incremented. Any ISPSPROF fields that do not have RESET fields are not used by a user unless they are creating a new ISPSPROF table.

You can generate the keyword file in one of three ways:

- Use the **Convert Assembler Configuration Table to Keyword File** option on the ISPF Configuration utility main menu.
- Use the **Create/Modify Settings and Regenerate Keyword File** option on the ISPF Configuration utility main menu.
- Use the **Convert Configuration Table Loadmod to Keyword File** option on the ISPF Configuration utility main menu.

To start the ISPF Configuration utility run the command `TSO ISPCCONF` while in ISPF. This displays the ISPF Configuration utility main menu panel, as shown in Figure 3 on page 10.



```
                          ISPF Configuration Utility

 1  Create/Modify Settings and Regenerate Keyword File
 2  Edit Keyword File Configuration Table
 3  Verify Keyword Table Contents
 4  Build Configuration Table Load Module
 5  Convert Assembler Configuration Table to Keyword File
 6  Build SMP/E USERMOD
 7  Convert Configuration Table Loadmod to Keyword File

Keyword File Data Set
  Data Set . . . KEYWORD
  Member . . . . AGPTBL

Configuration Table Assembler Source Data Set
  Data Set . . .
  Member . . . .

  Output File Content for Keyword File
  2   1. Include only non-default values
      2. Include defaults as comments
      3. Include all values


Current Configuration Table
  Keyword File : PACKETT.KEYWORD(AGPTBL)
  Identifier . : ISPCFIGU               Level  . . . : 480R8001
  Compile Date : 2016/03/02             Compile Time : 14:12

Option ===> _
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 3. ISPF configuration utility main menu panel (ISPPCONF)

These options are available on the ISPF Configuration utility main menu:

1. Create/Modify Settings and Regenerate Keyword File (see "Create/Modify Settings and Regenerate Keyword File" on page 11).
2. Edit Keyword File Configuration Table (see "Edit Keyword File Configuration Table" on page 30).
3. Verify Keyword Table Contents (see "Verify Keyword Table Contents" on page 31).
4. Build Configuration Table Load Module (see "Build Configuration Table Load Module" on page 32).
5. Convert Assembler Configuration Table to Keyword File (see "Convert Assembler Configuration Table to Keyword File" on page 33).
6. Build SMP/E USERMOD (see "Build SMP/E USERMOD" on page 34).
7. Convert Configuration Table LoadMod to Keyword File (see "Convert Configuration Table LoadMod to Keyword File" on page 36).

These input fields are available on the ISPF Configuration utility main menu panel:

Keyword File Data Set
The name of the data set and member containing the keyword file you want to use. The data set must exist and have a record length of at least 255 bytes for variable-length files, and 251 bytes for fixed-length files. This must be a Partitioned Data Set (PDS).

Configuration Table Assembler Source Data Set
The name of the sequential data set or partitioned data set and member containing the Configuration Table assembler source code to be converted into a keyword file.

Output File Content for Keyword File
The entry in this field determines the type of data written to the keyword file when the Convert Assembler Configuration Table to Keyword file option, or the Create/Modify Settings and Regenerate Keyword File option is used. Valid values are:

1
Only those values that are different than the default values are included in the keyword file.

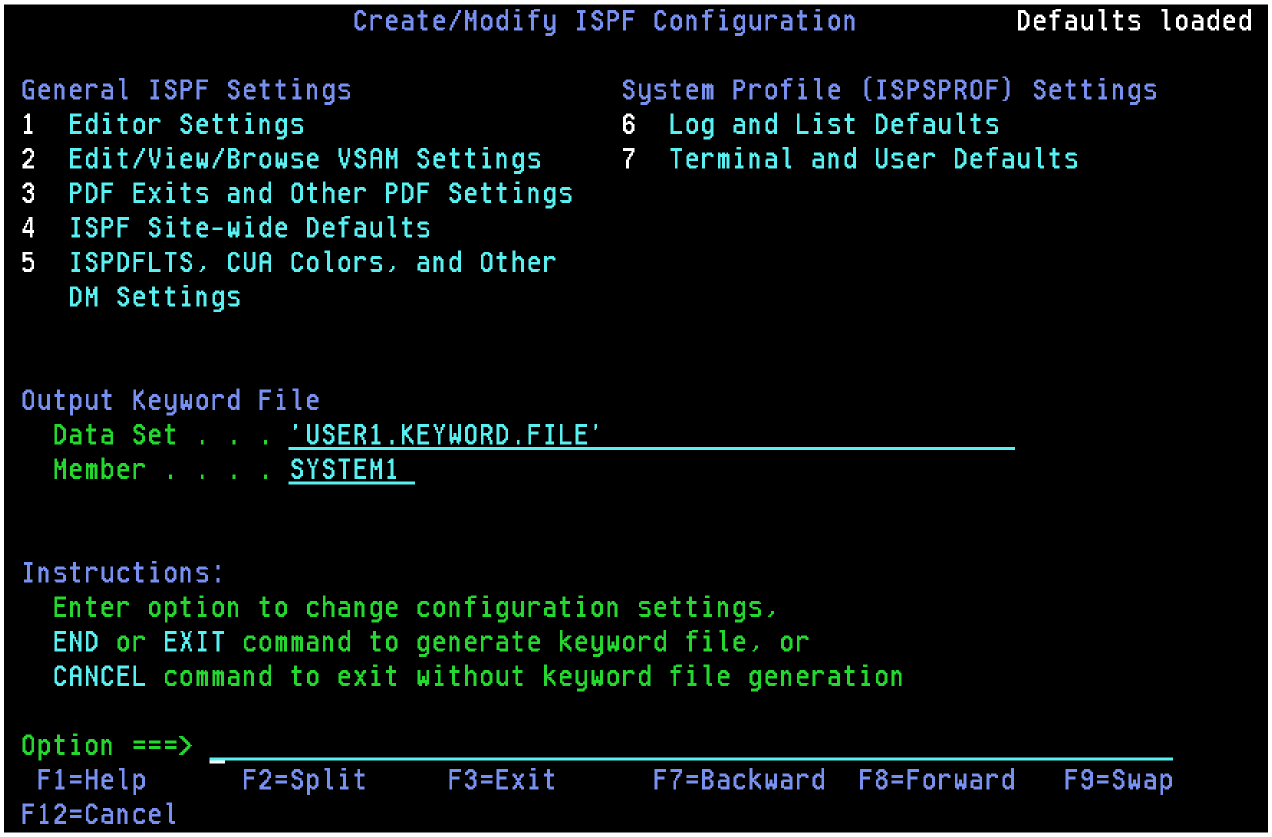
2
Those values different than the default values are included in the keyword file, and all default values are included as comment lines.

3
All values are included in the keyword file.

Create/Modify Settings and Regenerate Keyword File

If you choose Option 1, **Create/Modify Settings and Regenerate Keyword File** on the ISPF Configuration utility main menu panel, the Create/Modify ISPF Configuration panel appears, as shown in Figure 4 on page 11.

This option enables you to create a new keyword file or modify an existing keyword file containing ISPF configuration settings. Each time you use this option the keyword file is regenerated with the new or modified settings values. Note that any comments that have been added, or any reformatting changes that you might have made to the keyword file by editing it directly *do not* appear in the regenerated file.



```
                      Create/Modify ISPF Configuration         Defaults loaded

General ISPF Settings                      System Profile (ISPSPROF) Settings
1  Editor Settings                         6  Log and List Defaults
2  Edit/View/Browse VSAM Settings          7  Terminal and User Defaults
3  PDF Exits and Other PDF Settings
4  ISPF Site-wide Defaults
5  ISPDFLTS, CUA Colors, and Other
   DM Settings


Output Keyword File
  Data Set . . . 'USER1.KEYWORD.FILE'
  Member . . . . SYSTEM1


Instructions:
  Enter option to change configuration settings,
  END or EXIT command to generate keyword file, or
  CANCEL command to exit without keyword file generation

Option ===> _
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 4. Create/modify ISPF Configuration panel (ISPPMOD)

From this panel you can choose to modify these groups of configuration options by entering the option number on the command line.

- General ISPF Settings (see "General ISPF Settings Panels" on page 12)
  1. Editor Settings
  2. Edit/View/Browse VSAM Settings
  3. PDF Exits and Other PDF Settings
  4. ISPF Site-wide Defaults
  5. ISPDFLTS, CUA Colors, and Other DM Settings
- System Profile (ISPSPROF) Settings (see "System Profile (ISPSPROF) Settings panels" on page 25)
  6. Log and List Defaults
  7. Terminal and User Defaults

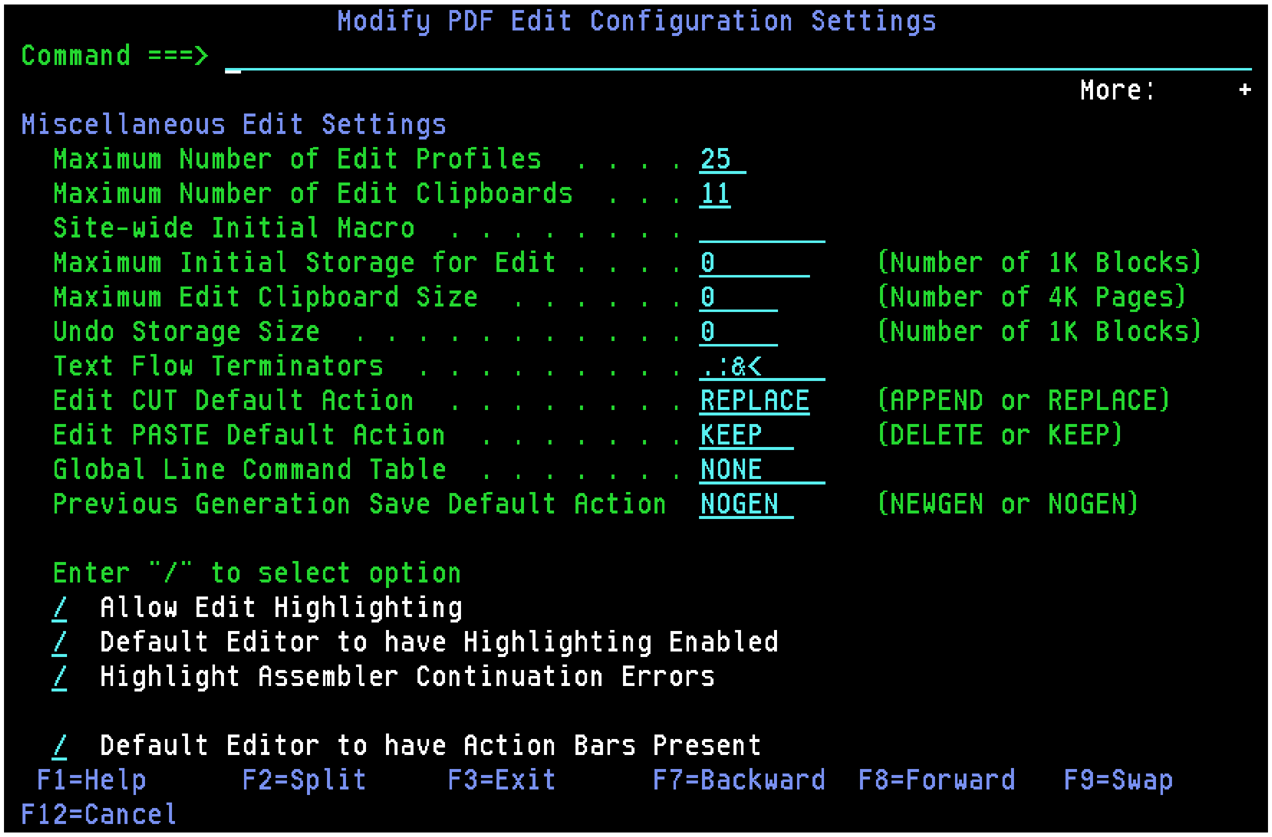
As you choose a group of settings to modify, a panel specific to that group of settings is displayed. If you are creating a new keyword file, the panel fields are initialized with the default values for each setting. If you are modifying an existing keyword file, the panel reflects the current values in that keyword file.

Each of the selected panels has field-level help for the individual fields on the panel. To display a pop-up help panel for a field, put the cursor on the field and press the Help function key, or enter HELP on the command line.

The Output Keyword File entry fields enable you to save the generated keyword file to a data set or member other than the data set or member used as input. If you leave this field blank, the data is saved back to the input data set and member specified on the ISPF Configuration utility main menu panel (ISPPCONF).

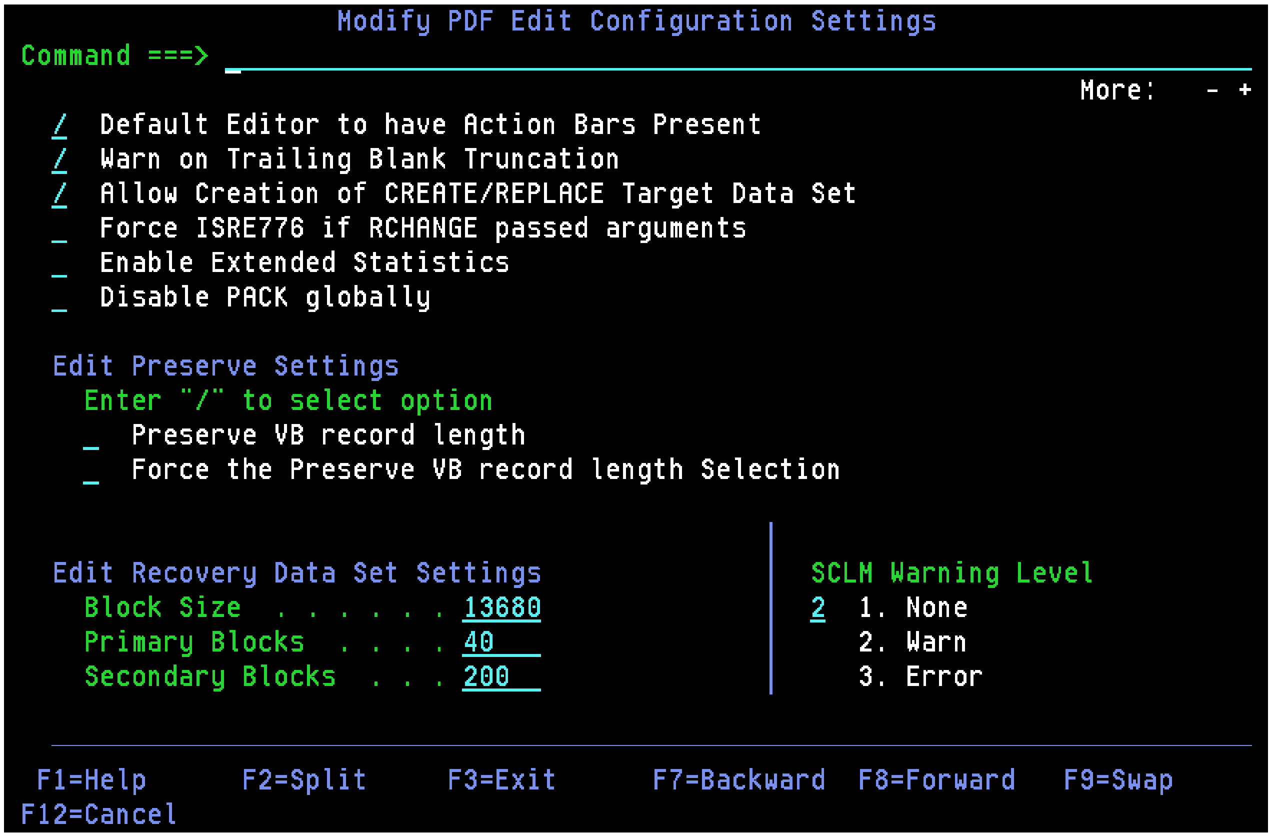
General ISPF Settings Panels

Selecting Option 1, **Editor Settings** on the Create/Modify ISPF Configuration panel (ISPPMOD) displays the Modify PDF Edit Configuration Settings panel (ISPPMOD1). Scroll down to display all the fields on this panel.



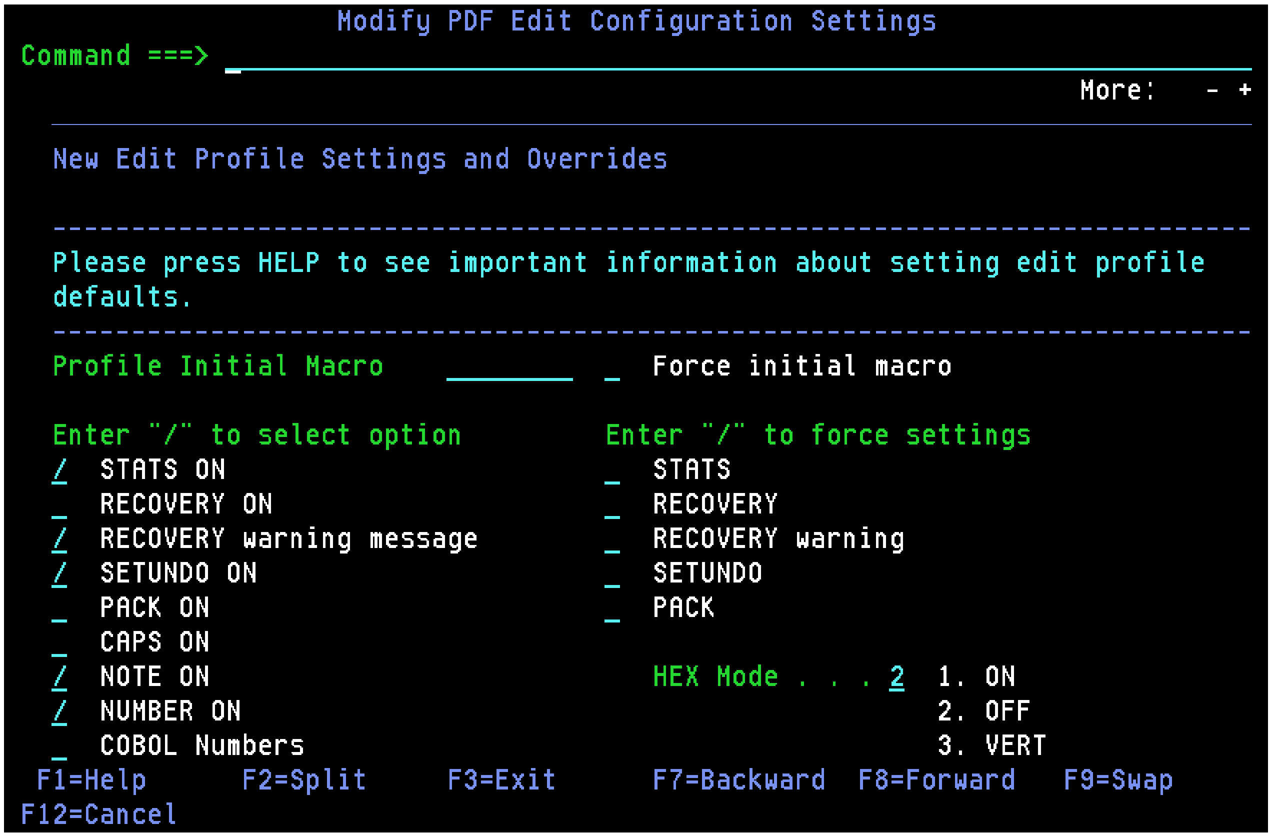
```
                    Modify PDF Edit Configuration Settings
Command ===> _
                                                                  More:     +
Miscellaneous Edit Settings
  Maximum Number of Edit Profiles  . . . . 25
  Maximum Number of Edit Clipboards  . . . 11
  Site-wide Initial Macro  . . . . . . . .
  Maximum Initial Storage for Edit . . . . 0           (Number of 1K Blocks)
  Maximum Edit Clipboard Size  . . . . . . 0           (Number of 4K Pages)
  Undo Storage Size  . . . . . . . . . . . 0           (Number of 1K Blocks)
  Text Flow Terminators  . . . . . . . . . .:&<
  Edit CUT Default Action  . . . . . . . . REPLACE     (APPEND or REPLACE)
  Edit PASTE Default Action  . . . . . . . KEEP        (DELETE or KEEP)
  Global Line Command Table  . . . . . . . NONE
  Previous Generation Save Default Action  NOGEN       (NEWGEN or NOGEN)

  Enter "/" to select option
  /  Allow Edit Highlighting
  /  Default Editor to have Highlighting Enabled
  /  Highlight Assembler Continuation Errors

  /  Default Editor to have Action Bars Present
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 5. Modify PDF EDIT Configuration Settings panel 1 (ISPPMOD1)



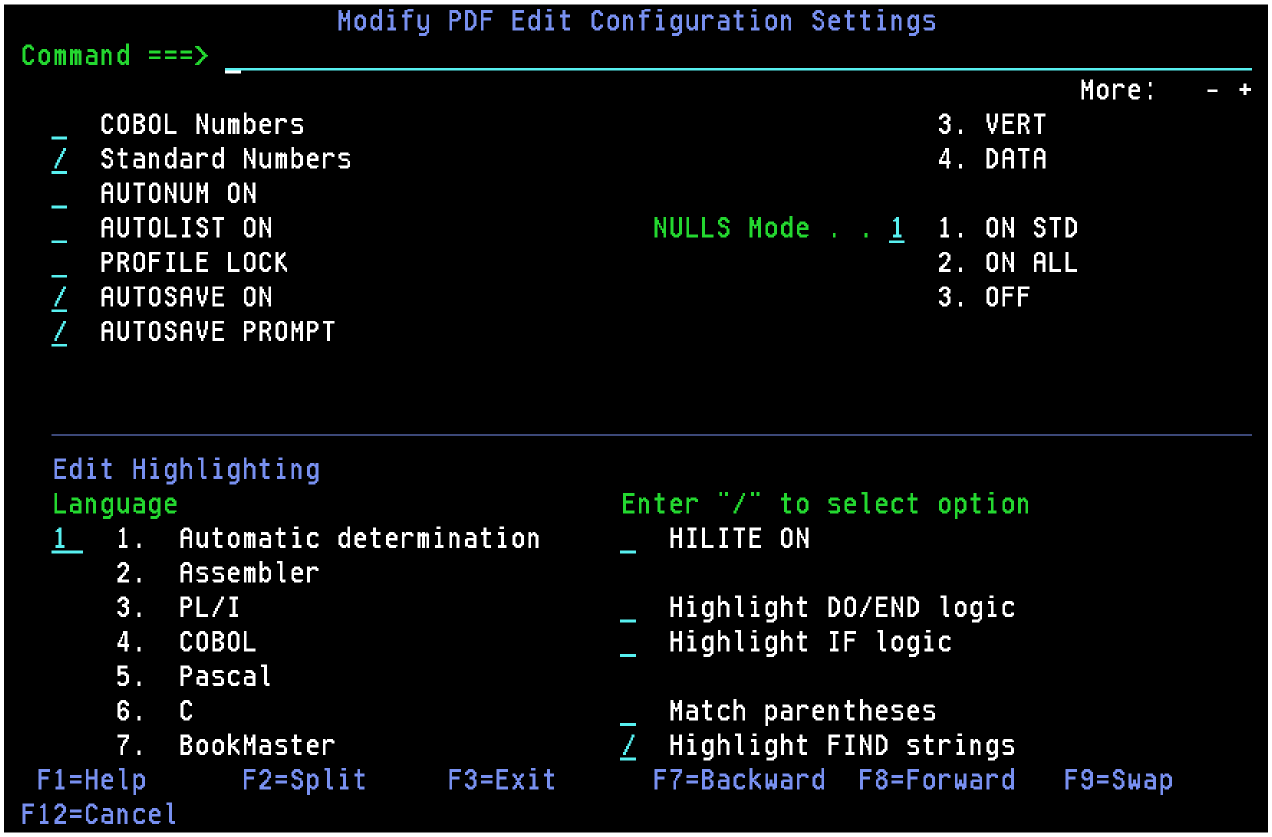
```
                       Modify PDF Edit Configuration Settings
Command ===> _
                                                                    More:   - +
  /  Default Editor to have Action Bars Present
  /  Warn on Trailing Blank Truncation
  /  Allow Creation of CREATE/REPLACE Target Data Set
  _  Force ISRE776 if RCHANGE passed arguments
  _  Enable Extended Statistics
  _  Disable PACK globally

  Edit Preserve Settings
    Enter "/" to select option
    _  Preserve VB record length
    _  Force the Preserve VB record length Selection


  Edit Recovery Data Set Settings                          SCLM Warning Level
    Block Size . . . . . . 13680                           2  1. None
    Primary Blocks  . . . . 40                                2. Warn
    Secondary Blocks  . . . 200                               3. Error


 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 6. Modify PDF Edit Configuration Settings panel 2 (ISPPMOD1)



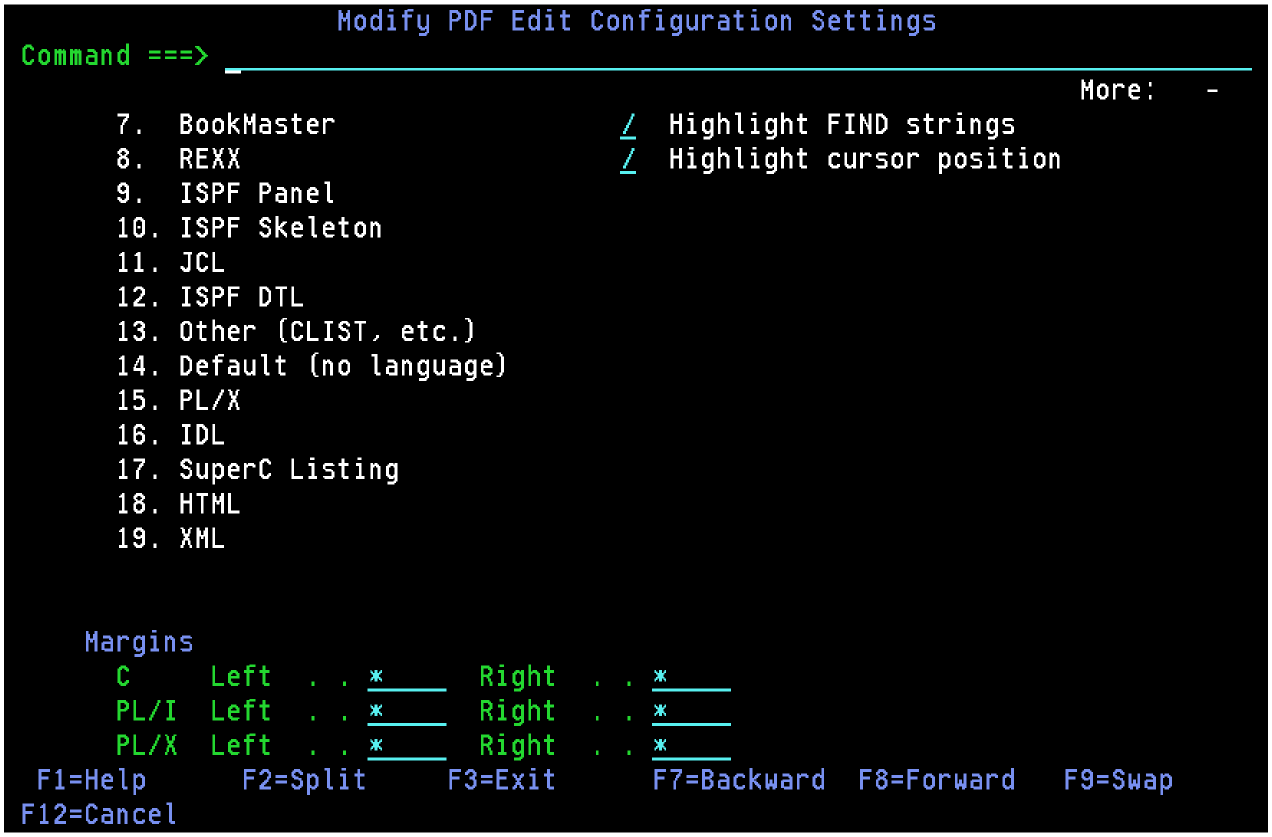
```
                       Modify PDF Edit Configuration Settings
Command ===> _
                                                                    More:   - +

  New Edit Profile Settings and Overrides


  ----------------------------------------------------------------------------
  Please press HELP to see important information about setting edit profile
  defaults.
  ----------------------------------------------------------------------------
  Profile Initial Macro         ________  _  Force initial macro

  Enter "/" to select option           Enter "/" to force settings
  /  STATS ON                          _  STATS
  _  RECOVERY ON                       _  RECOVERY
  /  RECOVERY warning message          _  RECOVERY warning
  /  SETUNDO ON                        _  SETUNDO
  _  PACK ON                           _  PACK
  _  CAPS ON
  /  NOTE ON                              HEX Mode . . . 2  1. ON
  /  NUMBER ON                                              2. OFF
  _  COBOL Numbers                                          3. VERT
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 7. Modify PDF Edit Configuration Settings panel 3 (ISPPMOD1)



```
                     Modify PDF Edit Configuration Settings
Command ===> _
                                                                     More:    - +
  _  COBOL Numbers                                         3. VERT
  /  Standard Numbers                                      4. DATA
  _  AUTONUM ON
  _  AUTOLIST ON                          NULLS Mode . . 1 1. ON STD
  _  PROFILE LOCK                                          2. ON ALL
  /  AUTOSAVE ON                                           3. OFF
  /  AUTOSAVE PROMPT


  Edit Highlighting
  Language                             Enter "/" to select option
  1    1.  Automatic determination     _  HILITE ON
       2.  Assembler
       3.  PL/I                        _  Highlight DO/END logic
       4.  COBOL                       _  Highlight IF logic
       5.  Pascal
       6.  C                           _  Match parentheses
       7.  BookMaster                  /  Highlight FIND strings
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

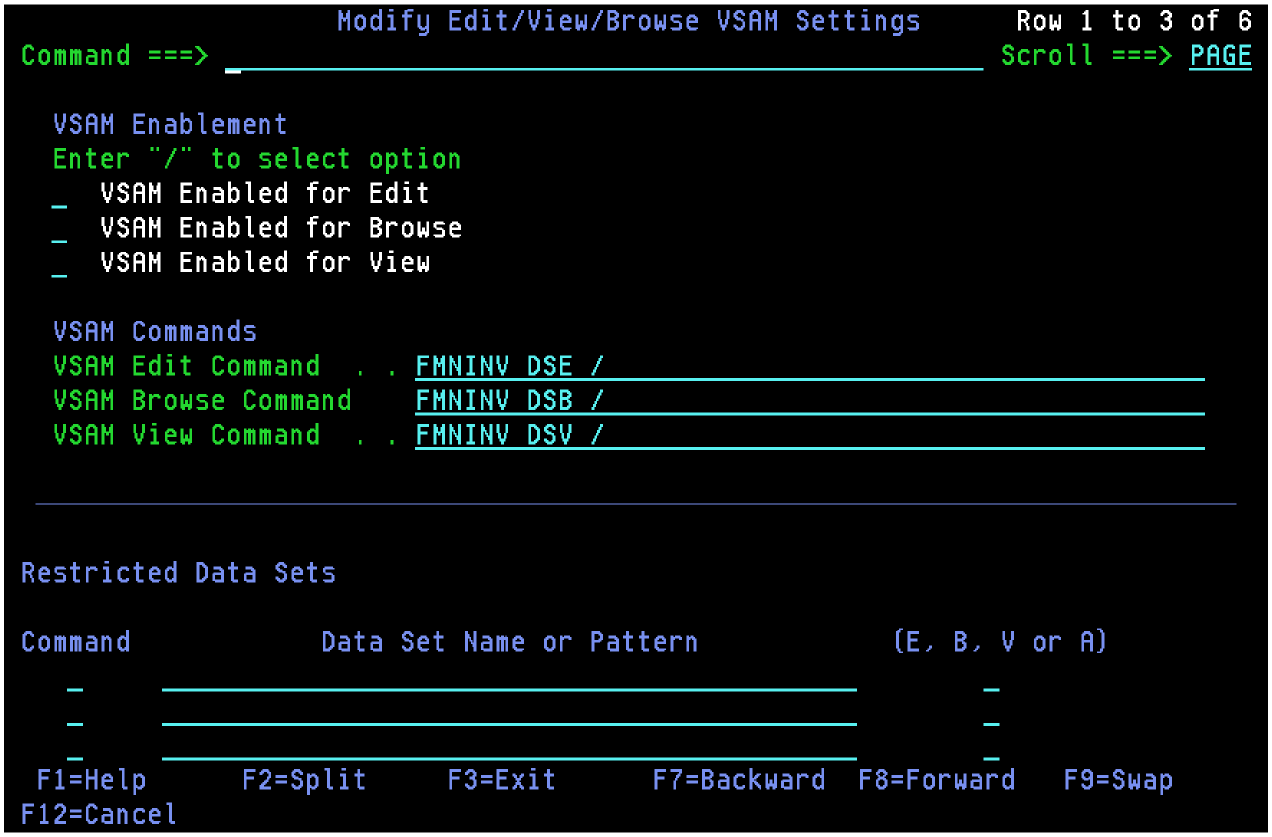
Figure 8. Modify PDF Edit Configuration Settings panel 4 (ISPPMOD1)



```
                     Modify PDF Edit Configuration Settings
Command ===> _
                                                                     More:    -
       7.  BookMaster                  /  Highlight FIND strings
       8.  REXX                        /  Highlight cursor position
       9.  ISPF Panel
       10. ISPF Skeleton
       11. JCL
       12. ISPF DTL
       13. Other (CLIST, etc.)
       14. Default (no language)
       15. PL/X
       16. IDL
       17. SuperC Listing
       18. HTML
       19. XML


    Margins
       C     Left  . . *     Right  . . *
       PL/I  Left  . . *     Right  . . *
       PL/X  Left  . . *     Right  . . *
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 9. Modify PDF Edit Configuration Settings panel 5 (ISPPMOD1)

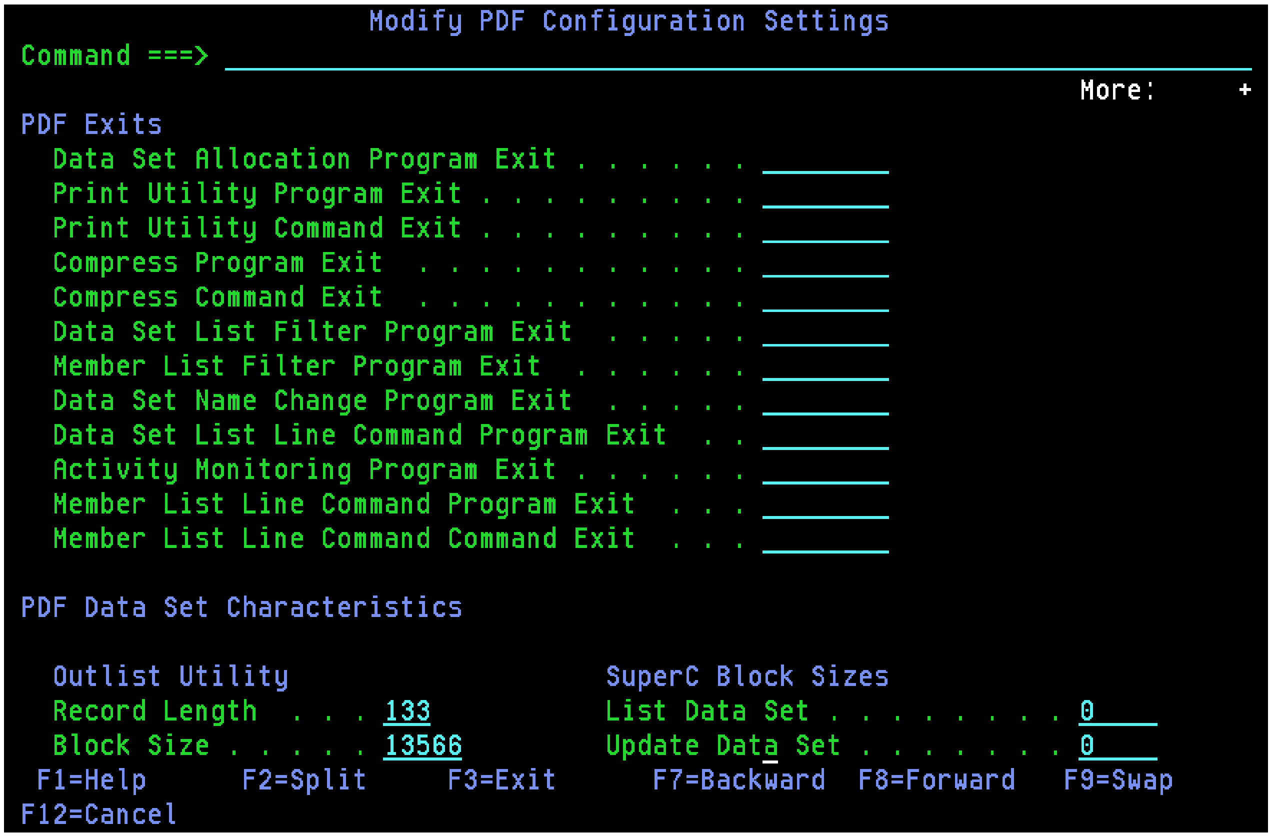
Selecting Option 2, **Edit/View/Browse VSAM Settings** on the Create/Modify ISPF Configuration panel (ISPPMOD) displays the Modify Edit/View/Browse VSAM Settings panel (ISPPMOD2).



```
                    Modify Edit/View/Browse VSAM Settings      Row 1 to 3 of 6
Command ===> _                                               Scroll ===> PAGE

  VSAM Enablement
  Enter "/" to select option
  _  VSAM Enabled for Edit
  _  VSAM Enabled for Browse
  _  VSAM Enabled for View

  VSAM Commands
  VSAM Edit Command  . . FMNINV DSE /
  VSAM Browse Command    FMNINV DSB /
  VSAM View Command  . . FMNINV DSV /


Restricted Data Sets

Command            Data Set Name or Pattern            (E, B, V or A)
   _                                                          _
   _                                                          _
   _                                                          _
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

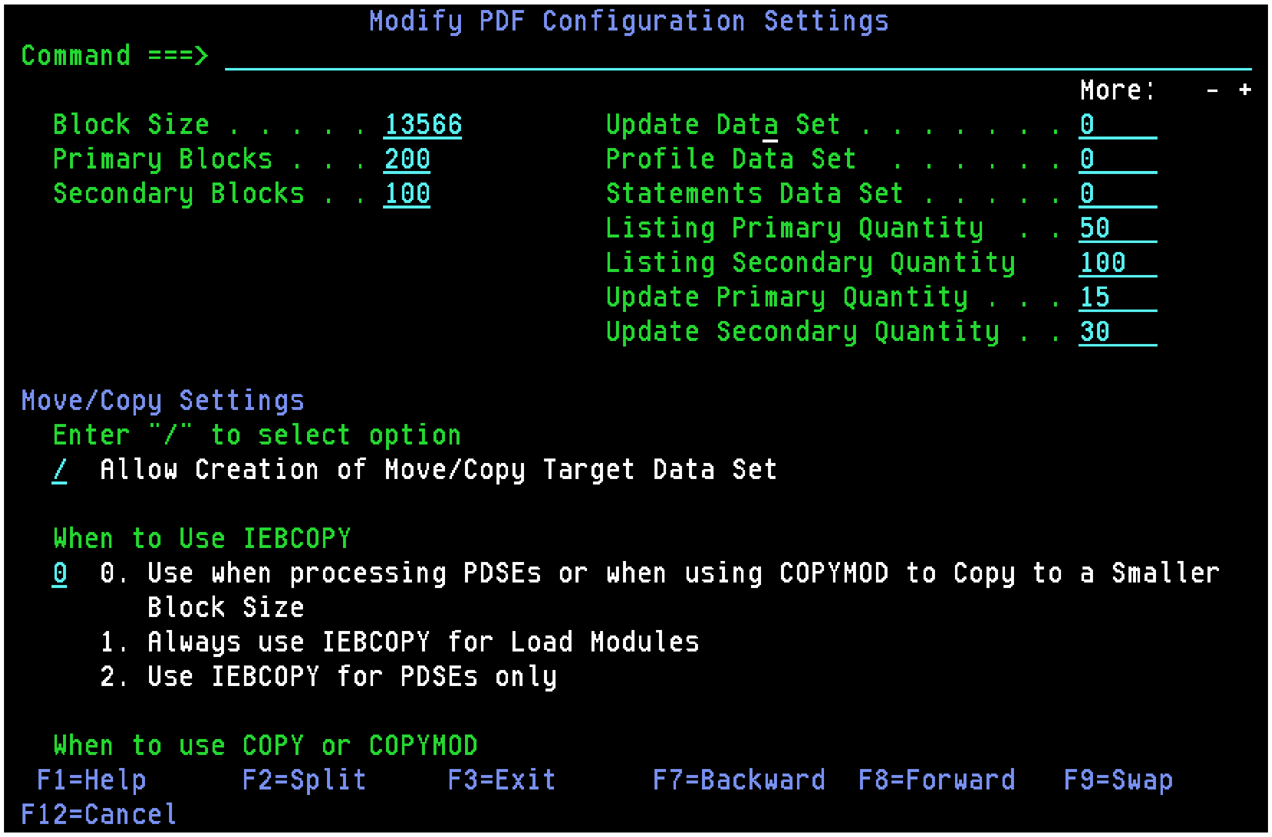
Figure 10. Modify Edit/View/Browse VSAM Settings panel (ISPPMOD2)

Selecting Option 3, **PDF Exits and Other PDF Settings** on the Create/Modify ISPF Configuration panel (ISPPMOD) displays the **Modify PDF Configuration Settings** panel (ISPPMOD3). Scroll down to display all the fields on this panel.



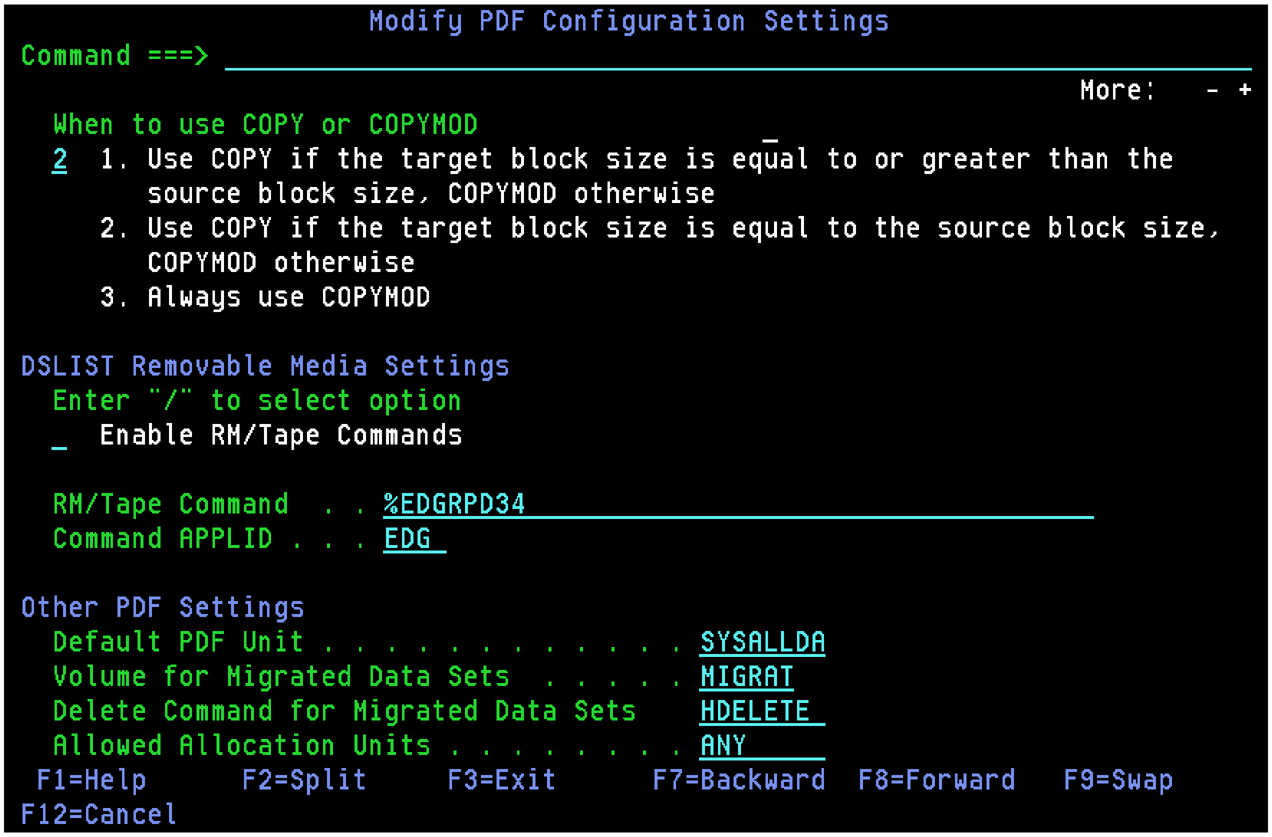
```
                       Modify PDF Configuration Settings
Command ===> ________________________________________________________________
                                                                  More:     +
PDF Exits
  Data Set Allocation Program Exit . . . . . . ________
  Print Utility Program Exit . . . . . . . . . ________
  Print Utility Command Exit . . . . . . . . . ________
  Compress Program Exit  . . . . . . . . . . . ________
  Compress Command Exit  . . . . . . . . . . . ________
  Data Set List Filter Program Exit  . . . . . ________
  Member List Filter Program Exit  . . . . . . ________
  Data Set Name Change Program Exit  . . . . . ________
  Data Set List Line Command Program Exit  . . ________
  Activity Monitoring Program Exit . . . . . . ________
  Member List Line Command Program Exit  . . . ________
  Member List Line Command Command Exit  . . . ________

PDF Data Set Characteristics

  Outlist Utility                  SuperC Block Sizes
  Record Length . . . 133          List Data Set . . . . . . . . 0
  Block Size . . . . . 13566       Update Data Set . . . . . . . 0
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 11. Modify PDF Configuration Settings panel 1 (ISPPMOD3)



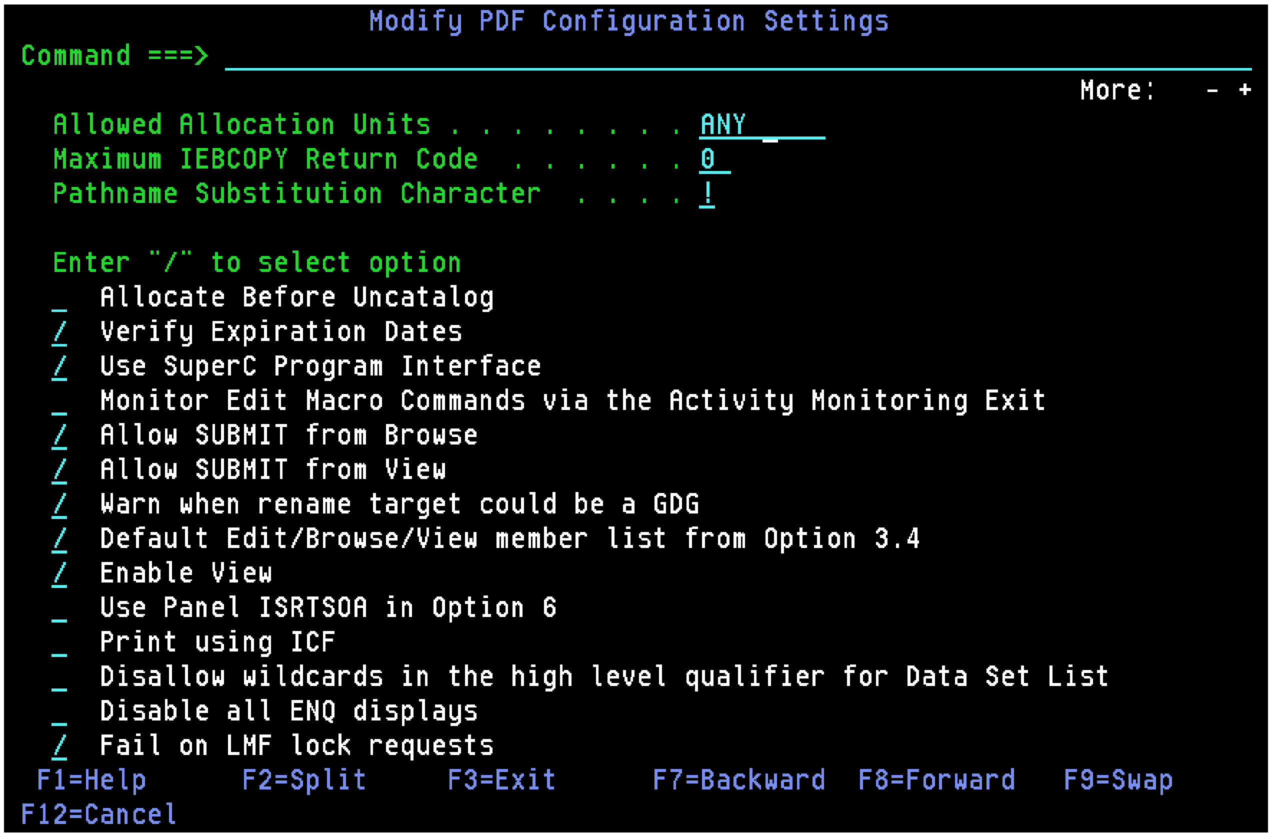
```
                       Modify PDF Configuration Settings
Command ===> ________________________________________________________________
                                                                  More:   - +
  Block Size . . . . . 13566       Update Data Set . . . . . . . 0
  Primary Blocks . . . 200         Profile Data Set  . . . . . . 0
  Secondary Blocks . . 100         Statements Data Set . . . . . 0
                                   Listing Primary Quantity  . . 50
                                   Listing Secondary Quantity    100
                                   Update Primary Quantity . . . 15
                                   Update Secondary Quantity . . 30

Move/Copy Settings
  Enter "/" to select option
  /  Allow Creation of Move/Copy Target Data Set

  When to Use IEBCOPY
  0  0. Use when processing PDSEs or when using COPYMOD to Copy to a Smaller
        Block Size
     1. Always use IEBCOPY for Load Modules
     2. Use IEBCOPY for PDSEs only

  When to use COPY or COPYMOD
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 12. Modify PDF Configuration Settings panel 2 (ISPPMOD3)



```
                       Modify PDF Configuration Settings
Command ===> ________________________________________________________________
                                                                 More:    - +
  When to use COPY or COPYMOD
  2  1. Use COPY if the target block size is equal to or greater than the
        source block size, COPYMOD otherwise
     2. Use COPY if the target block size is equal to the source block size,
        COPYMOD otherwise
     3. Always use COPYMOD

DSLIST Removable Media Settings
  Enter "/" to select option
  _  Enable RM/Tape Commands

  RM/Tape Command  . . %EDGRPD34_________________________________
  Command APPLID . . . EDG_

Other PDF Settings
  Default PDF Unit . . . . . . . . . . . . . SYSALLDA
  Volume for Migrated Data Sets  . . . . . . MIGRAT
  Delete Command for Migrated Data Sets      HDELETE_
  Allowed Allocation Units . . . . . . . . . ANY_____
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 13. Modify PDF Configuration Settings panel 3 (ISPPMOD3)



```
                       Modify PDF Configuration Settings
Command ===> ________________________________________________________________
                                                                 More:    - +
  Allowed Allocation Units . . . . . . . . . ANY_____
  Maximum IEBCOPY Return Code  . . . . . . . 0_
  Pathname Substitution Character  . . . . . !

  Enter "/" to select option
  _  Allocate Before Uncatalog
  /  Verify Expiration Dates
  /  Use SuperC Program Interface
  _  Monitor Edit Macro Commands via the Activity Monitoring Exit
  /  Allow SUBMIT from Browse
  /  Allow SUBMIT from View
  /  Warn when rename target could be a GDG
  /  Default Edit/Browse/View member list from Option 3.4
  /  Enable View
  _  Use Panel ISRTSOA in Option 6
  _  Print using ICF
  _  Disallow wildcards in the high level qualifier for Data Set List
  _  Disable all ENQ displays
  /  Fail on LMF lock requests
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 14. Modify PDF Configuration Settings panel 4 (ISPPMOD3)

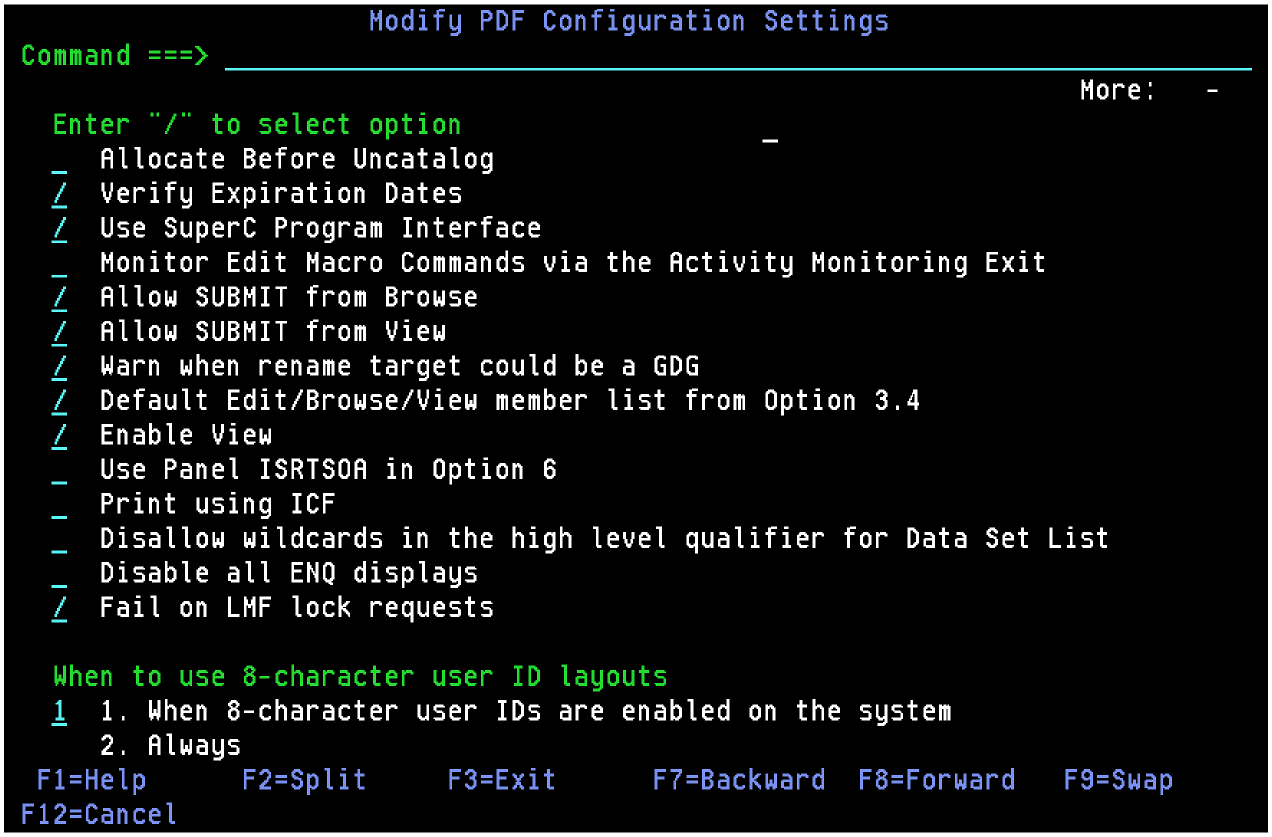
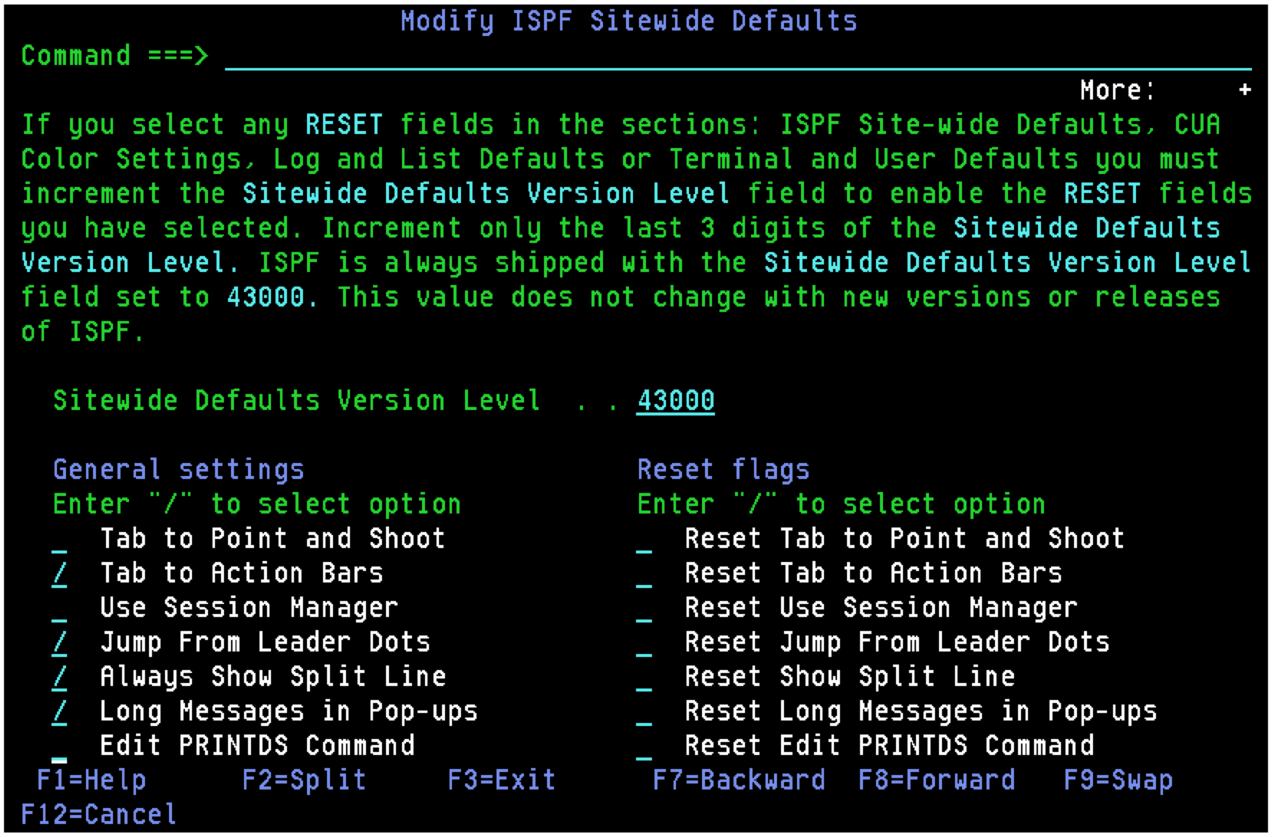

```
                   Modify PDF Configuration Settings
Command ===> ___________________________________________________________________
                                                                  More:    -
  Enter "/" to select option                             _
  _  Allocate Before Uncatalog
  /  Verify Expiration Dates
  /  Use SuperC Program Interface
  _  Monitor Edit Macro Commands via the Activity Monitoring Exit
  /  Allow SUBMIT from Browse
  /  Allow SUBMIT from View
  /  Warn when rename target could be a GDG
  /  Default Edit/Browse/View member list from Option 3.4
  /  Enable View
  _  Use Panel ISRTSOA in Option 6
  _  Print using ICF
  _  Disallow wildcards in the high level qualifier for Data Set List
  _  Disable all ENQ displays
  /  Fail on LMF lock requests

  When to use 8-character user ID layouts
  1  1. When 8-character user IDs are enabled on the system
     2. Always
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

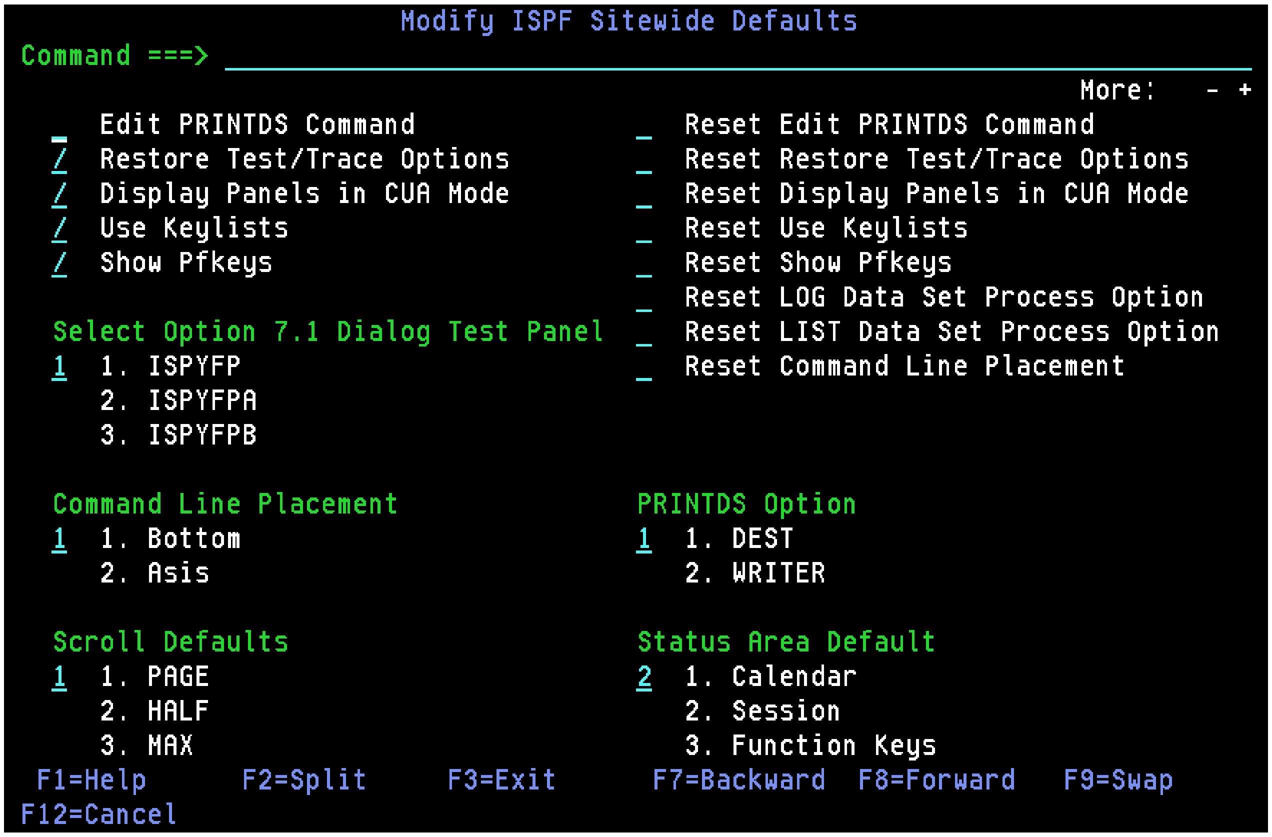
Figure 15. Modify PDF Configuration Settings panel 5 (ISPPMOD3)

Selecting Option 4, **ISPF Site-wide Defaults** on the Create/Modify ISPF Configuration panel (ISPPMOD) displays the Modify ISPF Sitewide Defaults panel (ISPPMOD4). Scroll down to display all the fields on this panel.



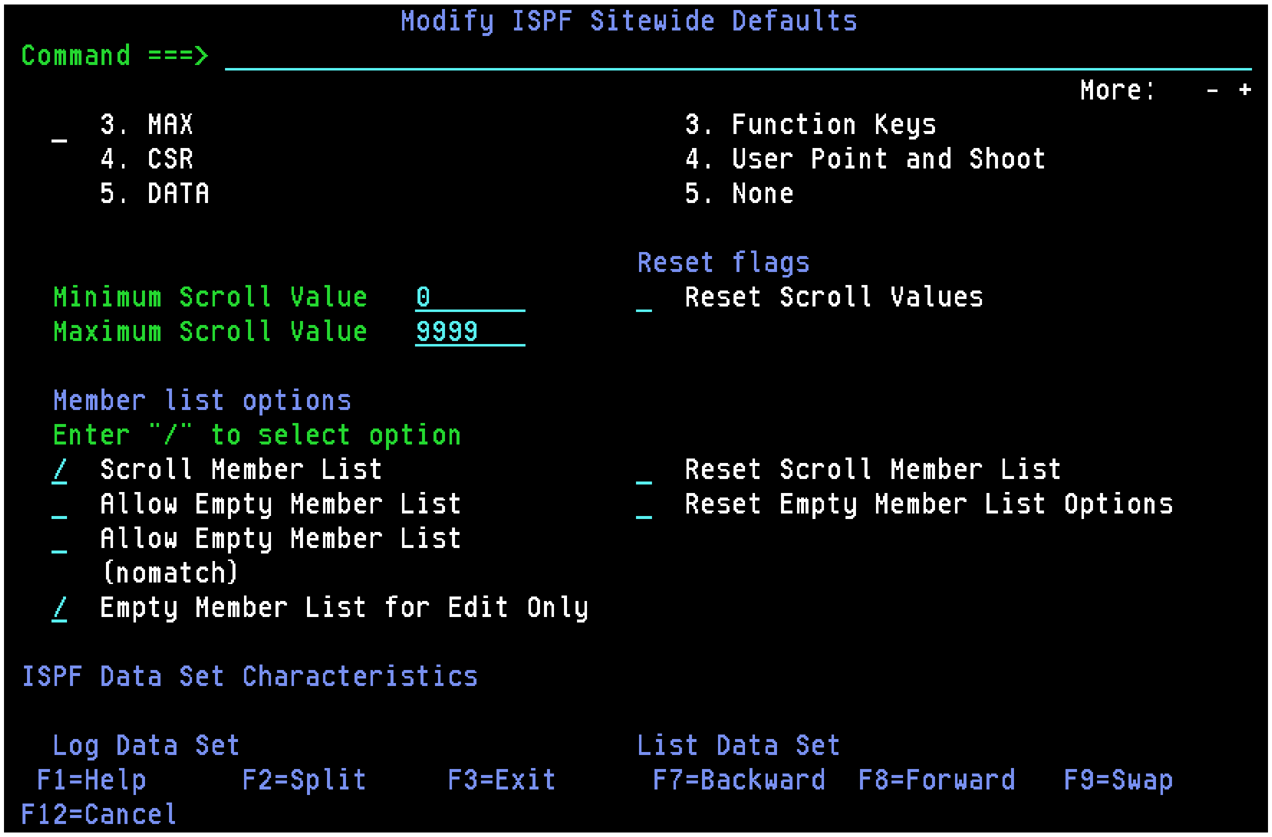
```
                         Modify ISPF Sitewide Defaults
Command ===> ________________________________________________________________
                                                                  More:     +
If you select any RESET fields in the sections: ISPF Site-wide Defaults, CUA
Color Settings, Log and List Defaults or Terminal and User Defaults you must
increment the Sitewide Defaults Version Level field to enable the RESET fields
you have selected. Increment only the last 3 digits of the Sitewide Defaults
Version Level. ISPF is always shipped with the Sitewide Defaults Version Level
field set to 43000. This value does not change with new versions or releases
of ISPF.

  Sitewide Defaults Version Level  . . 43000

  General settings                     Reset flags
  Enter "/" to select option           Enter "/" to select option
  _  Tab to Point and Shoot            _  Reset Tab to Point and Shoot
  /  Tab to Action Bars                _  Reset Tab to Action Bars
  _  Use Session Manager               _  Reset Use Session Manager
  /  Jump From Leader Dots             _  Reset Jump From Leader Dots
  /  Always Show Split Line            _  Reset Show Split Line
  /  Long Messages in Pop-ups          _  Reset Long Messages in Pop-ups
  _  Edit PRINTDS Command              _  Reset Edit PRINTDS Command
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 16. Modify ISPF Sitewide Defaults panel 1 (ISPPMOD4)



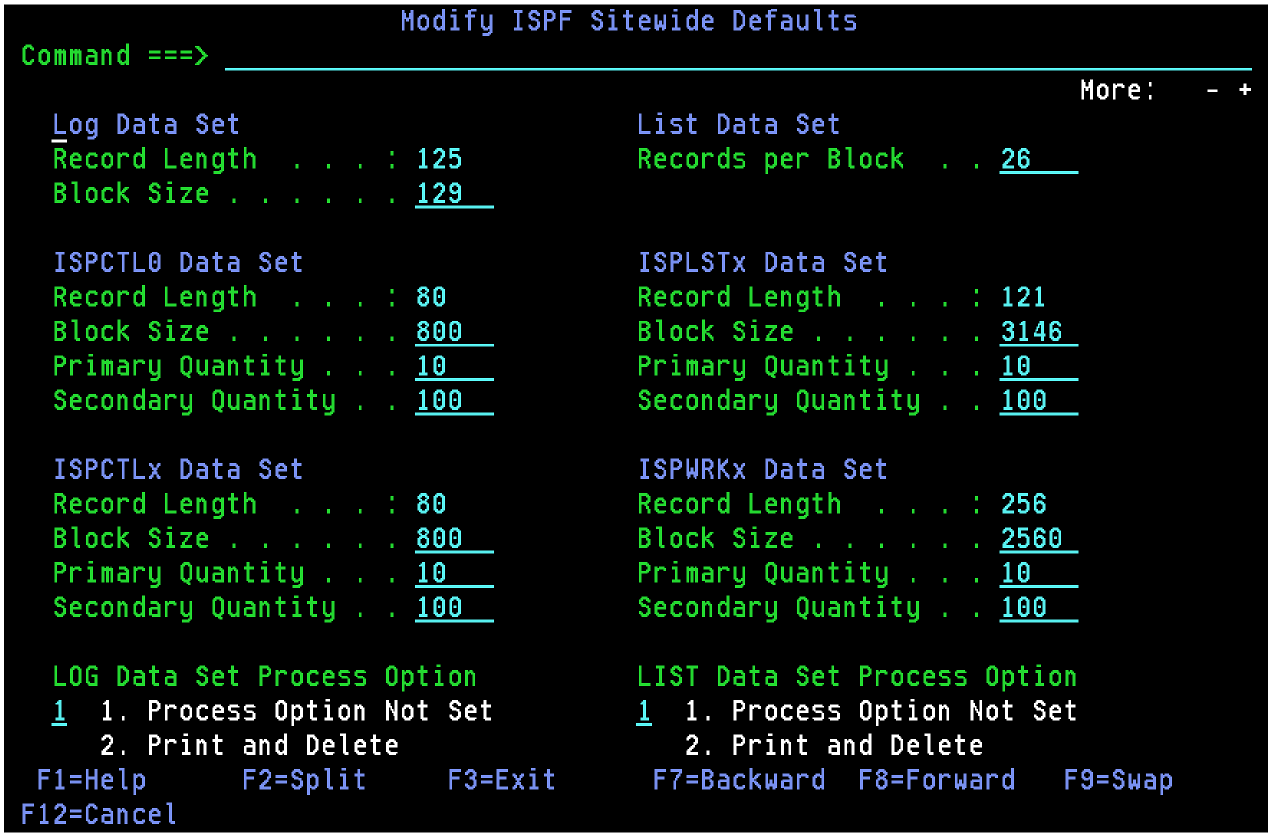
```
                         Modify ISPF Sitewide Defaults
Command ===> ________________________________________________________________
                                                                  More:   - +
  _  Edit PRINTDS Command              _  Reset Edit PRINTDS Command
  /  Restore Test/Trace Options        _  Reset Restore Test/Trace Options
  /  Display Panels in CUA Mode        _  Reset Display Panels in CUA Mode
  /  Use Keylists                      _  Reset Use Keylists
  /  Show Pfkeys                       _  Reset Show Pfkeys
                                       _  Reset LOG Data Set Process Option
  Select Option 7.1 Dialog Test Panel  _  Reset LIST Data Set Process Option
  1  1. ISPYFP                         _  Reset Command Line Placement
     2. ISPYFPA
     3. ISPYFPB

  Command Line Placement               PRINTDS Option
  1  1. Bottom                         1  1. DEST
     2. Asis                              2. WRITER

  Scroll Defaults                      Status Area Default
  1  1. PAGE                           2  1. Calendar
     2. HALF                              2. Session
     3. MAX                               3. Function Keys
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 17. Modify ISPF Sitewide Defaults panel 2 (ISPPMOD4)



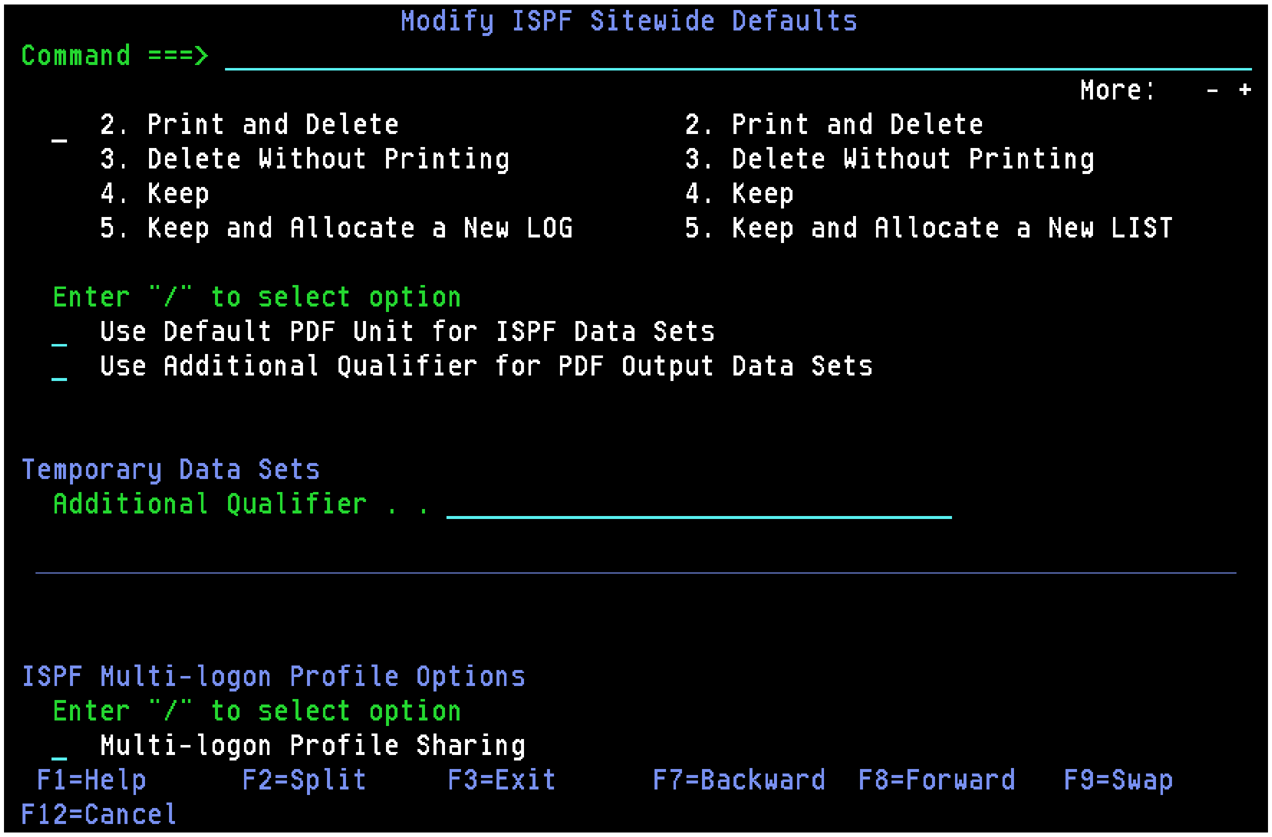
```
                       Modify ISPF Sitewide Defaults
Command ===> ____________________________________________________________________
                                                                  More:    - +
   _   3. MAX                                   3. Function Keys
       4. CSR                                   4. User Point and Shoot
       5. DATA                                  5. None

                                             Reset flags
  Minimum Scroll Value    0_______           _  Reset Scroll Values
  Maximum Scroll Value    9999____

  Member list options
  Enter "/" to select option
  /  Scroll Member List                      _  Reset Scroll Member List
  _  Allow Empty Member List                 _  Reset Empty Member List Options
  _  Allow Empty Member List
     (nomatch)
  /  Empty Member List for Edit Only

ISPF Data Set Characteristics

  Log Data Set                               List Data Set
 F1=Help      F2=Split      F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 18. Modify ISPF Sitewide Defaults panel 3 (ISPPMOD4)



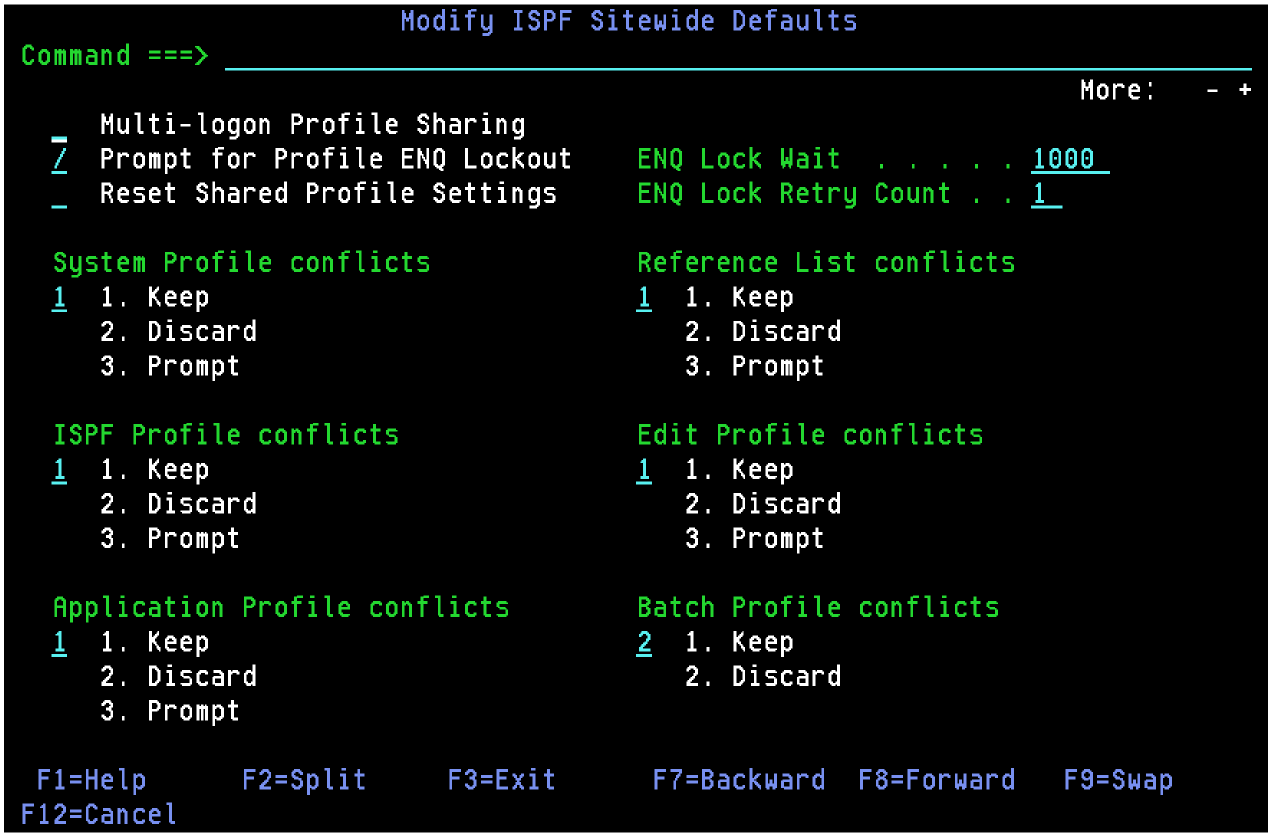
```
                       Modify ISPF Sitewide Defaults
Command ===> ____________________________________________________________________
                                                                  More:    - +
  Log Data Set                               List Data Set
  Record Length . . . : 125                  Records per Block . . 26____
  Block Size . . . . . . 129__

  ISPCTL0 Data Set                           ISPLSTx Data Set
  Record Length . . . : 80                   Record Length . . . : 121
  Block Size . . . . . . 800__               Block Size . . . . . . 3146_
  Primary Quantity . . . 10___               Primary Quantity . . . 10___
  Secondary Quantity . . 100__               Secondary Quantity . . 100__

  ISPCTLx Data Set                           ISPWRKx Data Set
  Record Length . . . : 80                   Record Length . . . : 256
  Block Size . . . . . . 800__               Block Size . . . . . . 2560_
  Primary Quantity . . . 10___               Primary Quantity . . . 10___
  Secondary Quantity . . 100__               Secondary Quantity . . 100__

  LOG Data Set Process Option                LIST Data Set Process Option
  1  1. Process Option Not Set               1  1. Process Option Not Set
     2. Print and Delete                        2. Print and Delete
 F1=Help      F2=Split      F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 19. Modify ISPF Sitewide Defaults panel 4 (ISPPMOD4)



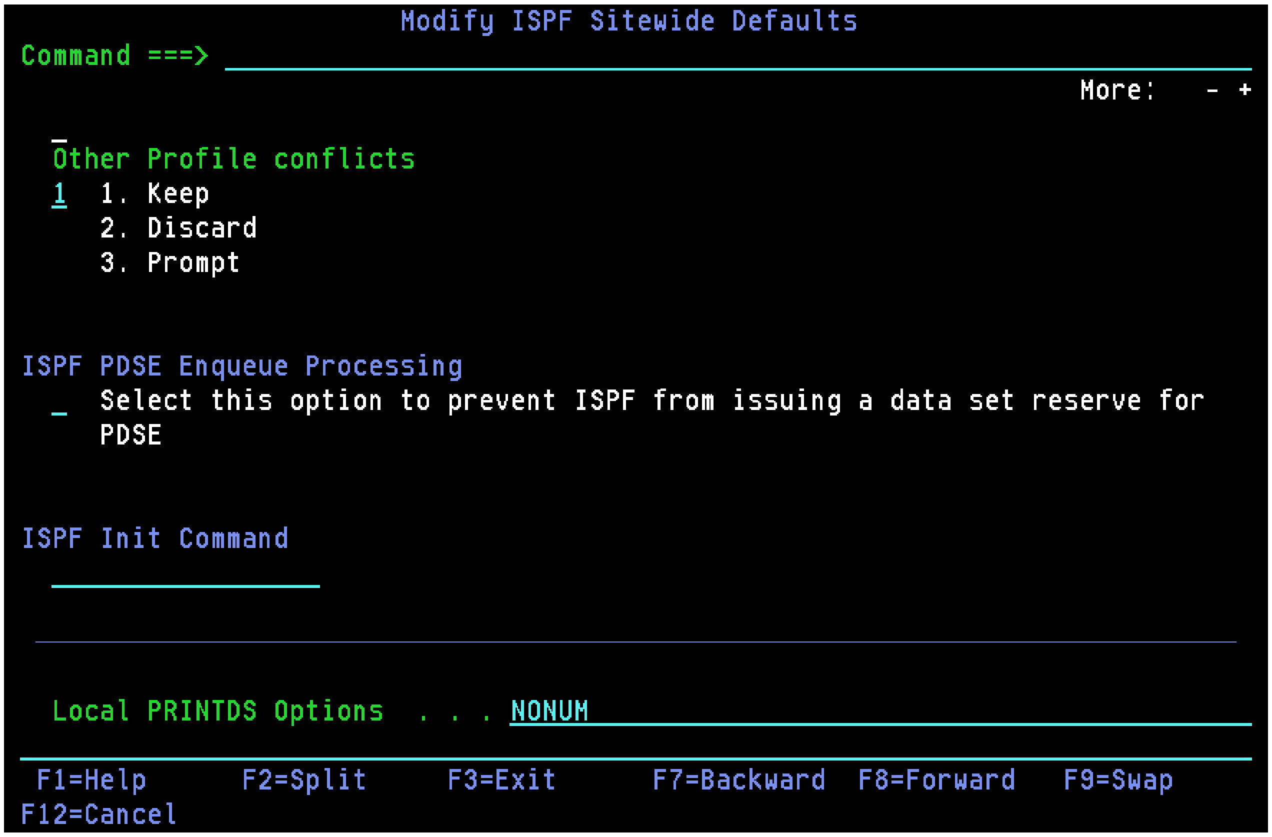
```
                         Modify ISPF Sitewide Defaults
Command ===> ____________________________________________________________
                                                                More:    - +
  _   2. Print and Delete                     2. Print and Delete
      3. Delete Without Printing              3. Delete Without Printing
      4. Keep                                 4. Keep
      5. Keep and Allocate a New LOG          5. Keep and Allocate a New LIST

  Enter "/" to select option
  _   Use Default PDF Unit for ISPF Data Sets
  _   Use Additional Qualifier for PDF Output Data Sets


Temporary Data Sets
  Additional Qualifier . . ________________________________



ISPF Multi-logon Profile Options
  Enter "/" to select option
  _   Multi-logon Profile Sharing
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 20. Modify ISPF Sitewide Defaults panel 5 (ISPPMOD4)



```
                         Modify ISPF Sitewide Defaults
Command ===> ____________________________________________________________
                                                                More:    - +
  _   Multi-logon Profile Sharing
  /   Prompt for Profile ENQ Lockout     ENQ Lock Wait . . . . . 1000
  _   Reset Shared Profile Settings      ENQ Lock Retry Count . . 1

  System Profile conflicts               Reference List conflicts
  1   1. Keep                            1   1. Keep
      2. Discard                             2. Discard
      3. Prompt                              3. Prompt

  ISPF Profile conflicts                 Edit Profile conflicts
  1   1. Keep                            1   1. Keep
      2. Discard                             2. Discard
      3. Prompt                              3. Prompt

  Application Profile conflicts          Batch Profile conflicts
  1   1. Keep                            2   1. Keep
      2. Discard                             2. Discard
      3. Prompt

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

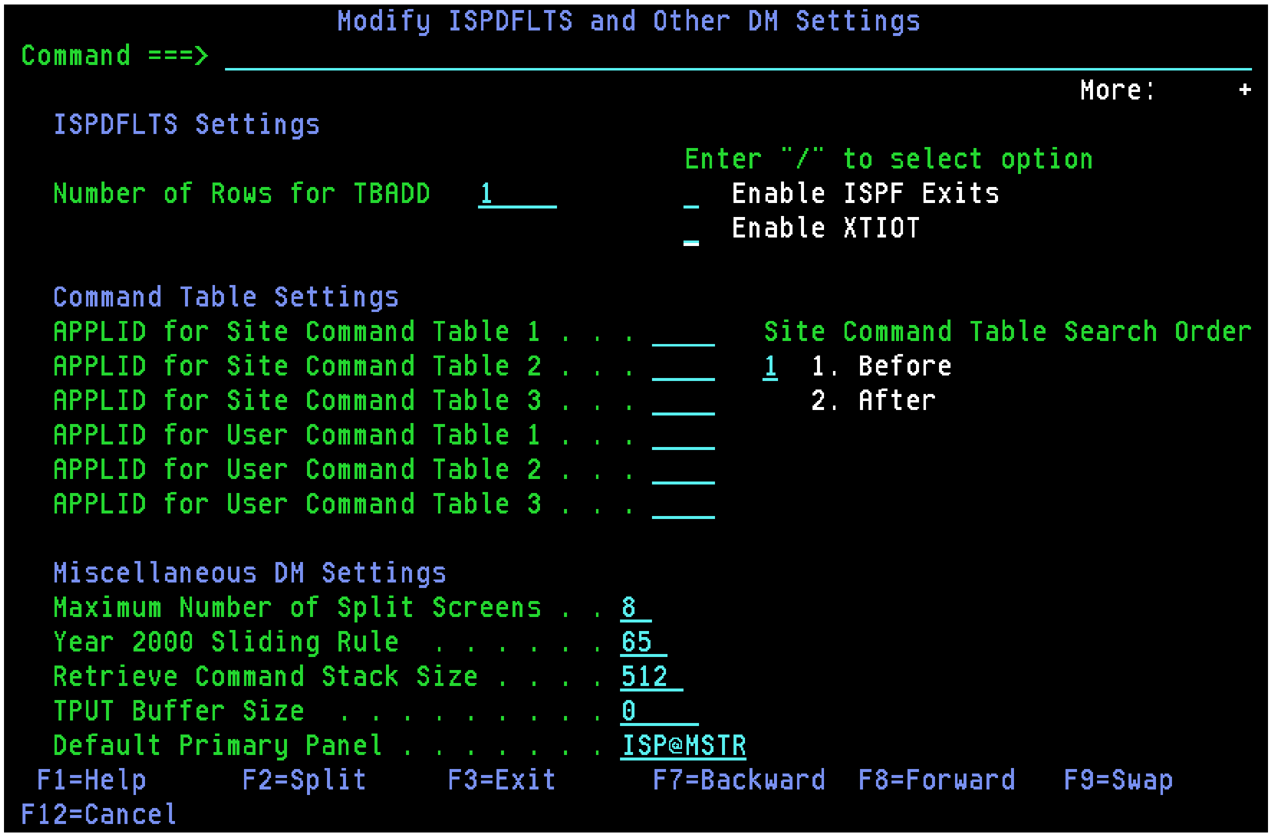
Figure 21. Modify ISPF Sitewide Defaults panel 6 (ISPPMOD4)



```
                          Modify ISPF Sitewide Defaults
Command ===> _____________________________________________________________
                                                             More:    - +

  Other Profile conflicts
  1  1. Keep
     2. Discard
     3. Prompt


ISPF PDSE Enqueue Processing
  _  Select this option to prevent ISPF from issuing a data set reserve for
     PDSE


ISPF Init Command
  _________________

 ____________________________________________________________________________


  Local PRINTDS Options  . . . NONUM
_____________________________________________________________________________
 F1=Help       F2=Split      F3=Exit       F7=Backward  F8=Forward    F9=Swap
F12=Cancel
```

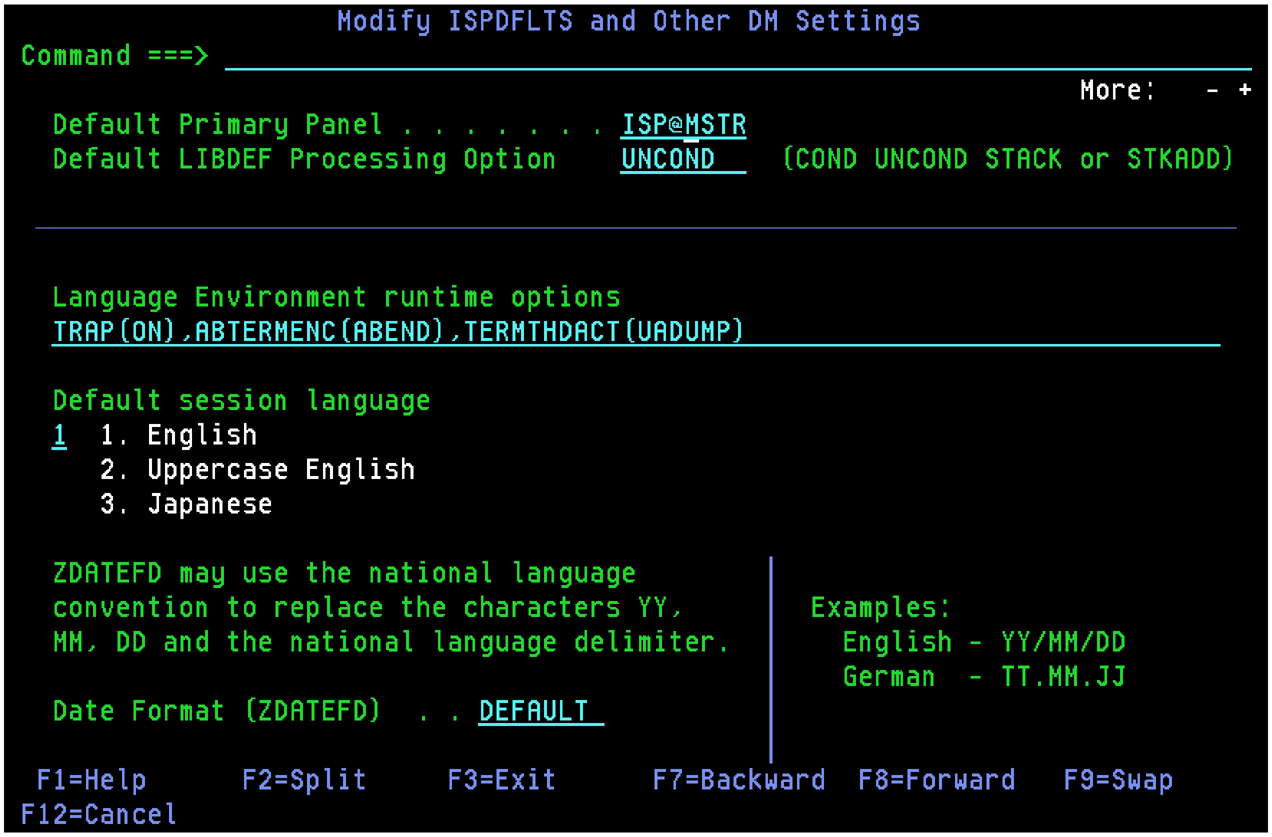
Figure 22. Modify ISPF Sitewide Defaults panel 7 (ISPPMOD4)

Selecting Option 5, **ISPDFLTS, CUA Colors, and Other DM Settings** on the Create/Modify ISPF Configuration panel (ISPPMOD) displays the Modify ISPDFLTS and Other DM Settings panel (ISPPMOD5). Scroll down to display all the fields on this panel.



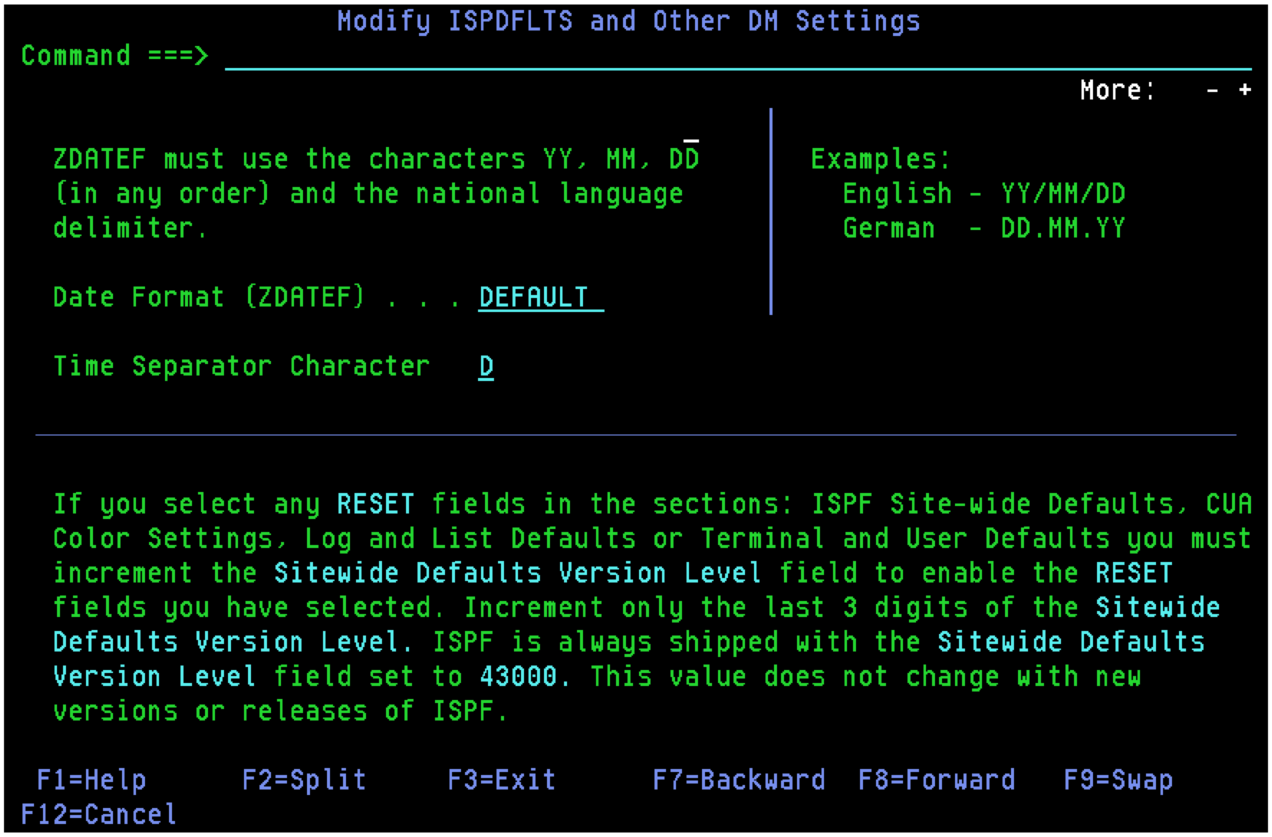
```
                     Modify ISPDFLTS and Other DM Settings
Command ===> _________________________________________________________________
                                                                   More:    +
  ISPDFLTS Settings
                                        Enter "/" to select option
  Number of Rows for TBADD    1____         _  Enable ISPF Exits
                                            _  Enable XTIOT

  Command Table Settings
  APPLID for Site Command Table 1 . . . ____     Site Command Table Search Order
  APPLID for Site Command Table 2 . . . ____     1  1. Before
  APPLID for Site Command Table 3 . . . ____        2. After
  APPLID for User Command Table 1 . . . ____
  APPLID for User Command Table 2 . . . ____
  APPLID for User Command Table 3 . . . ____

  Miscellaneous DM Settings
  Maximum Number of Split Screens . . 8_
  Year 2000 Sliding Rule  . . . . . . 65_
  Retrieve Command Stack Size . . . . 512_
  TPUT Buffer Size  . . . . . . . . . 0___
  Default Primary Panel . . . . . . . ISP@MSTR
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 23. Modify ISPDFLTS and Other DM Settings panel 1 (ISPPMOD5)



```
                     Modify ISPDFLTS and Other DM Settings
Command ===> _________________________________________________________________
                                                                   More:  - +
  Default Primary Panel . . . . . . . ISP@MSTR
  Default LIBDEF Processing Option    UNCOND__  (COND UNCOND STACK or STKADD)



  Language Environment runtime options
  TRAP(ON),ABTERMENC(ABEND),TERMTHDACT(UADUMP)_____________________________

  Default session language
  1  1. English
     2. Uppercase English
     3. Japanese

  ZDATEFD may use the national language       |
  convention to replace the characters YY,   |  Examples:
  MM, DD and the national language delimiter. |    English - YY/MM/DD
                                              |    German  - TT.MM.JJ
  Date Format (ZDATEFD)   . . DEFAULT_        |

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 24. Modify ISPDFLTS and Other DM Settings panel 2 (ISPPMOD5)



```
                    Modify ISPDFLTS and Other DM Settings
Command ===> ________________________________________________________________
                                                                  More:   - +

  ZDATEF must use the characters YY, MM, DD  |  Examples:
  (in any order) and the national language   |    English - YY/MM/DD
  delimiter.                                 |    German  - DD.MM.YY
                                             |
  Date Format (ZDATEF) . . . DEFAULT_        |

  Time Separator Character   D

 ____________________________________________________________________________


  If you select any RESET fields in the sections: ISPF Site-wide Defaults, CUA
  Color Settings, Log and List Defaults or Terminal and User Defaults you must
  increment the Sitewide Defaults Version Level field to enable the RESET
  fields you have selected. Increment only the last 3 digits of the Sitewide
  Defaults Version Level. ISPF is always shipped with the Sitewide Defaults
  Version Level field set to 43000. This value does not change with new
  versions or releases of ISPF.

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

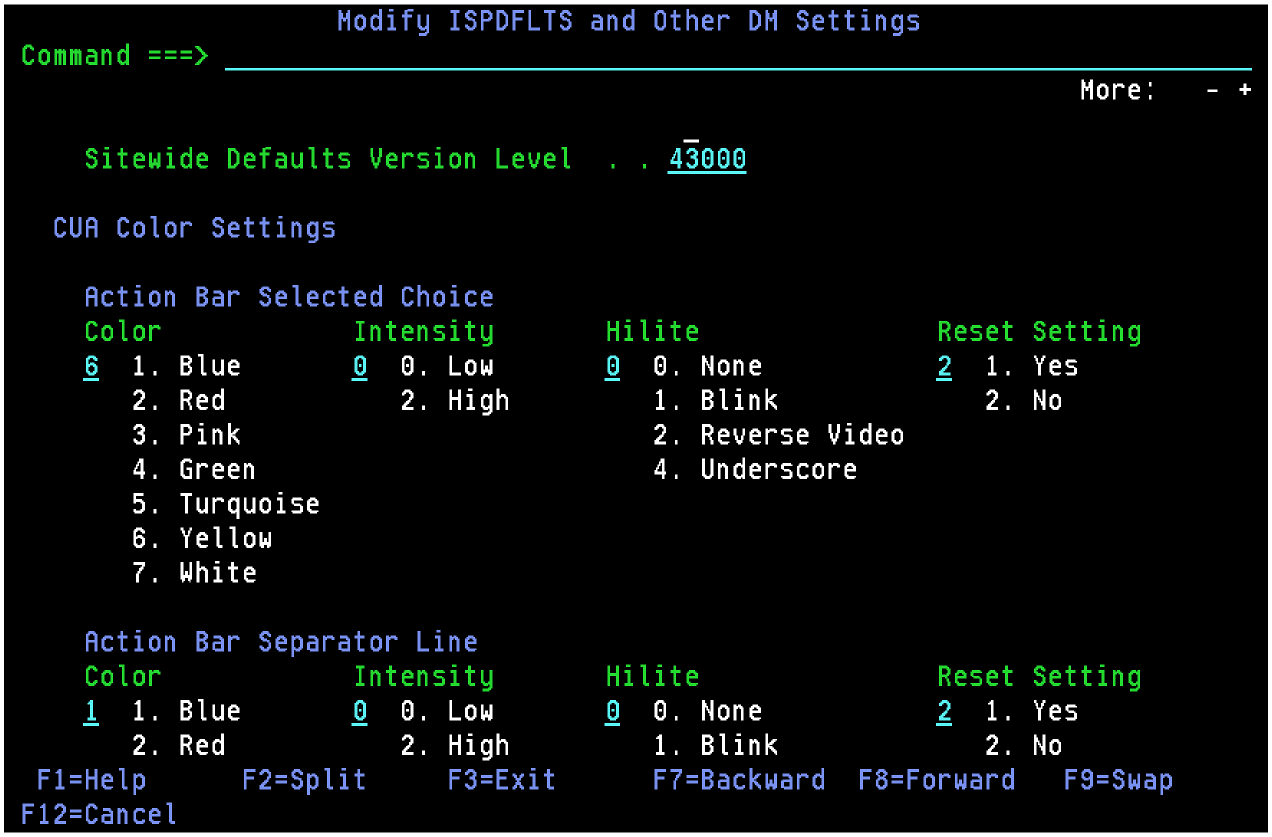
Figure 25. Modify ISPDFLTS and Other DM Settings panel 3 (ISPPMOD5)

The remaining screens enable you to specify the CUA Color settings for these panel elements:

- Action Bar Selected Choice
- Action Bar Separator Line
- Action Bar Unselected Choice
- Action Message Text
- Caution Text
- Choice Entry Field
- Column Heading
- Descriptive Text
- Emphasized Text
- Error Emphasis
- Field Prompt
- Function Keys
- Informational Message Text
- List Entry Field
- List Item Description
- List Item
- Normal Entry Field
- Normal Text
- Panel ID
- Panel Information
- Panel Title

- Point and Shoot
- Pulldown Available Choice
- Pulldown Unavailable Choice
- Reference Phrase
- Scroll Information
- Selection Available Choice
- Selection Unavailable Choice
- Variable Output Information
- Warning Message Text
- Warning Message
- Workarea Separator Line

Figure 26 on page 25 shows the options for the first two panel elements.

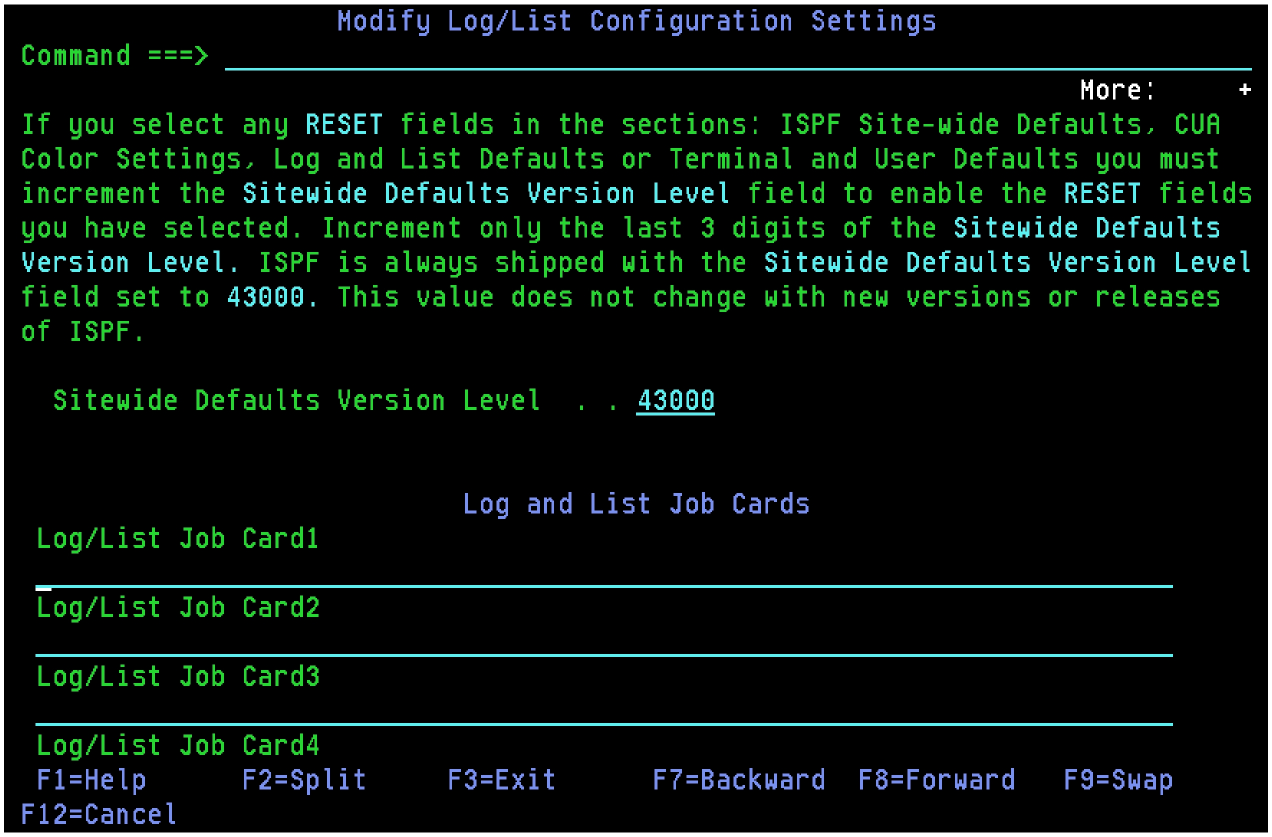


```
                    Modify ISPDFLTS and Other DM Settings
Command ===> ____________________________________________________________________
                                                                    More:   - +

    Sitewide Defaults Version Level  . . 43000

  CUA Color Settings

    Action Bar Selected Choice
    Color            Intensity       Hilite                Reset Setting
    6  1. Blue       0  0. Low       0  0. None            2  1. Yes
       2. Red           2. High         1. Blink              2. No
       3. Pink                          2. Reverse Video
       4. Green                         4. Underscore
       5. Turquoise
       6. Yellow
       7. White

    Action Bar Separator Line
    Color            Intensity       Hilite                Reset Setting
    1  1. Blue       0  0. Low       0  0. None            2  1. Yes
       2. Red           2. High         1. Blink              2. No
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 26. Modify ISPDFLTS and Other DM Settings panel 4 (ISPPMOD5)

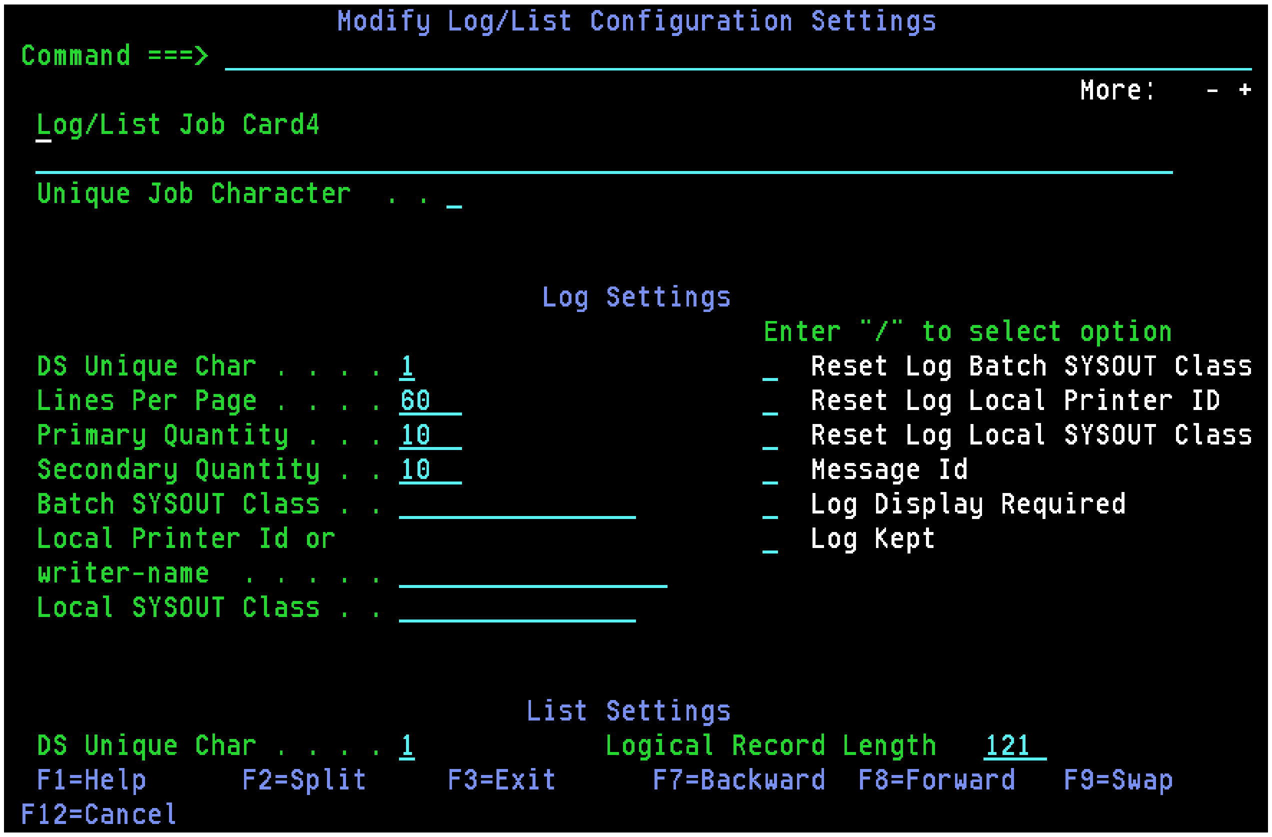
System Profile (ISPSPROF) Settings panels

Selecting Option 6, **Log and List Defaults** on the Create/Modify ISPF Configuration panel (ISPPMOD) displays the Modify Log/List Configuration Settings panel (ISPPMOD6). Scroll down to display all the fields on this panel.



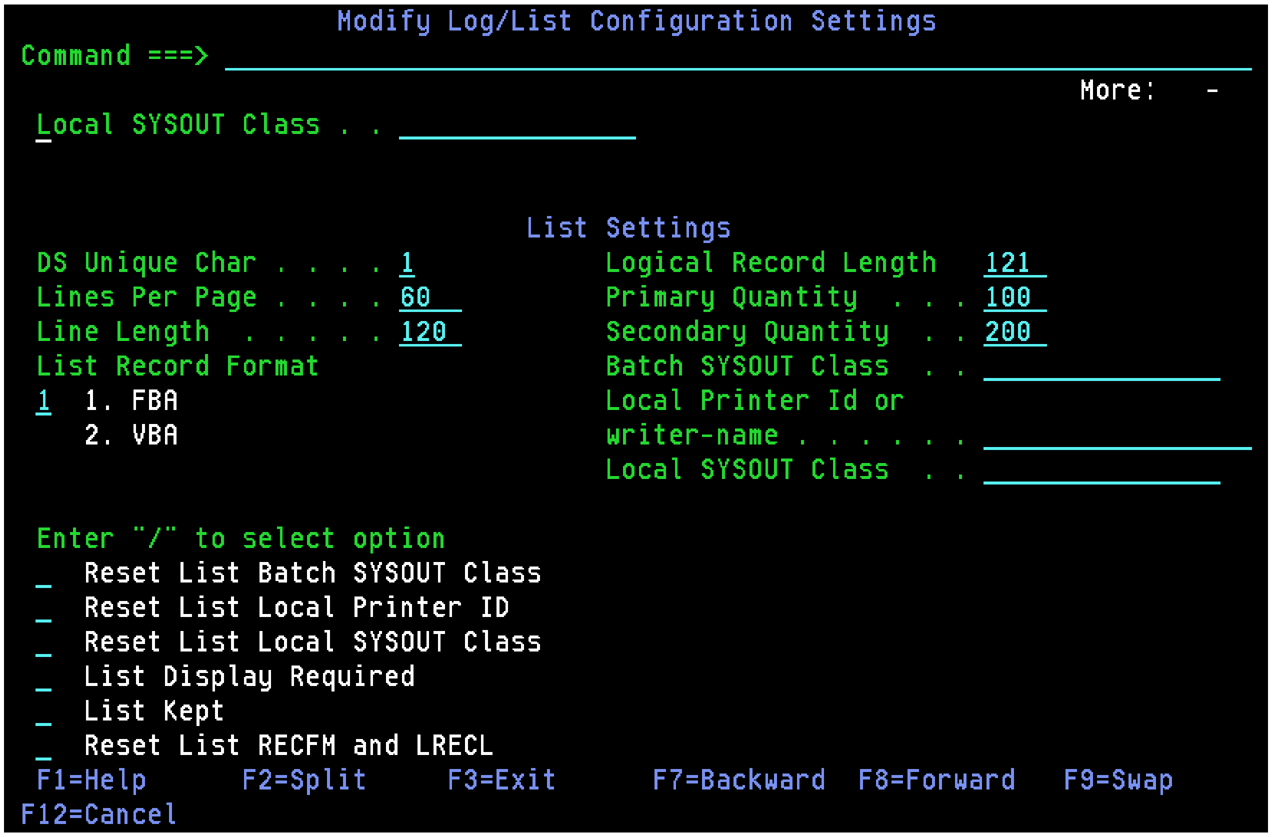
```
                     Modify Log/List Configuration Settings
Command ===> _________________________________________________________________
                                                                    More:     +
If you select any RESET fields in the sections: ISPF Site-wide Defaults, CUA
Color Settings, Log and List Defaults or Terminal and User Defaults you must
increment the Sitewide Defaults Version Level field to enable the RESET fields
you have selected. Increment only the last 3 digits of the Sitewide Defaults
Version Level. ISPF is always shipped with the Sitewide Defaults Version Level
field set to 43000. This value does not change with new versions or releases
of ISPF.

  Sitewide Defaults Version Level  . . 43000


                          Log and List Job Cards
 Log/List Job Card1

 ________________________________________________________________________
 Log/List Job Card2

 ________________________________________________________________________
 Log/List Job Card3

 ________________________________________________________________________
 Log/List Job Card4
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 27. Modify Log/List Configuration Settings panel 1 (ISPPMOD6)



```
                     Modify Log/List Configuration Settings
Command ===> _________________________________________________________________
                                                                    More:   - +
 Log/List Job Card4

 ________________________________________________________________________
 Unique Job Character  . . _



                               Log Settings
                                                Enter "/" to select option
 DS Unique Char . . . . 1                       _  Reset Log Batch SYSOUT Class
 Lines Per Page . . . . 60                      _  Reset Log Local Printer ID
 Primary Quantity . . . 10                      _  Reset Log Local SYSOUT Class
 Secondary Quantity . . 10                      _  Message Id
 Batch SYSOUT Class . . _______________         _  Log Display Required
 Local Printer Id or                            _  Log Kept
 writer-name  . . . . . _________________
 Local SYSOUT Class . . _______________


                              List Settings
 DS Unique Char . . . . 1            Logical Record Length   121
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

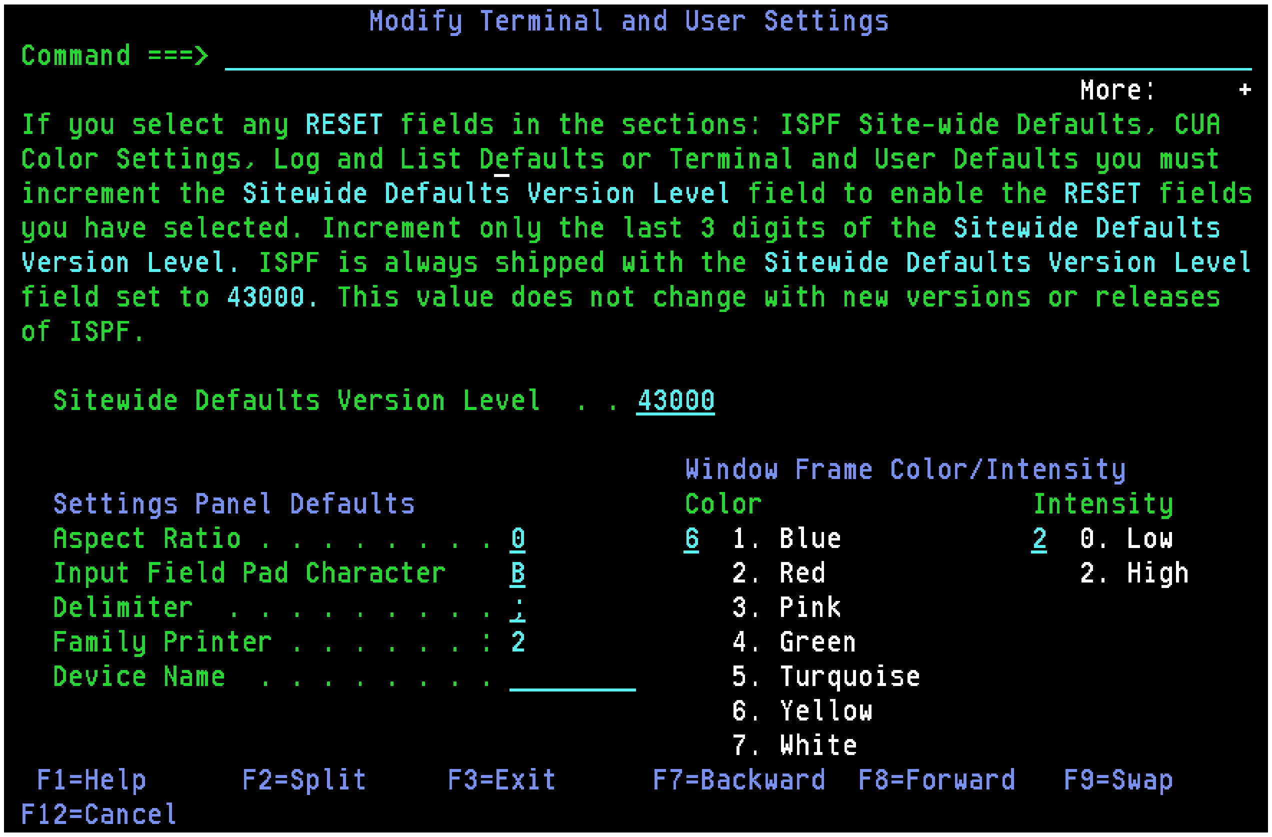
Figure 28. Modify Log/List Configuration Settings panel 2 (ISPPMOD6)



```
                         Modify Log/List Configuration Settings
Command ===> ________________________________________________________________
                                                                   More:    -
 Local SYSOUT Class . . _______________


                                 List Settings
 DS Unique Char . . . . 1            Logical Record Length   121
 Lines Per Page . . . . 60           Primary Quantity  . . . 100
 Line Length  . . . . . 120          Secondary Quantity  . . 200
 List Record Format                  Batch SYSOUT Class  . . _______________
 1  1. FBA                           Local Printer Id or
    2. VBA                           writer-name . . . . . . _________________
                                     Local SYSOUT Class  . . _______________

 Enter "/" to select option
 _  Reset List Batch SYSOUT Class
 _  Reset List Local Printer ID
 _  Reset List Local SYSOUT Class
 _  List Display Required
 _  List Kept
 _  Reset List RECFM and LRECL
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

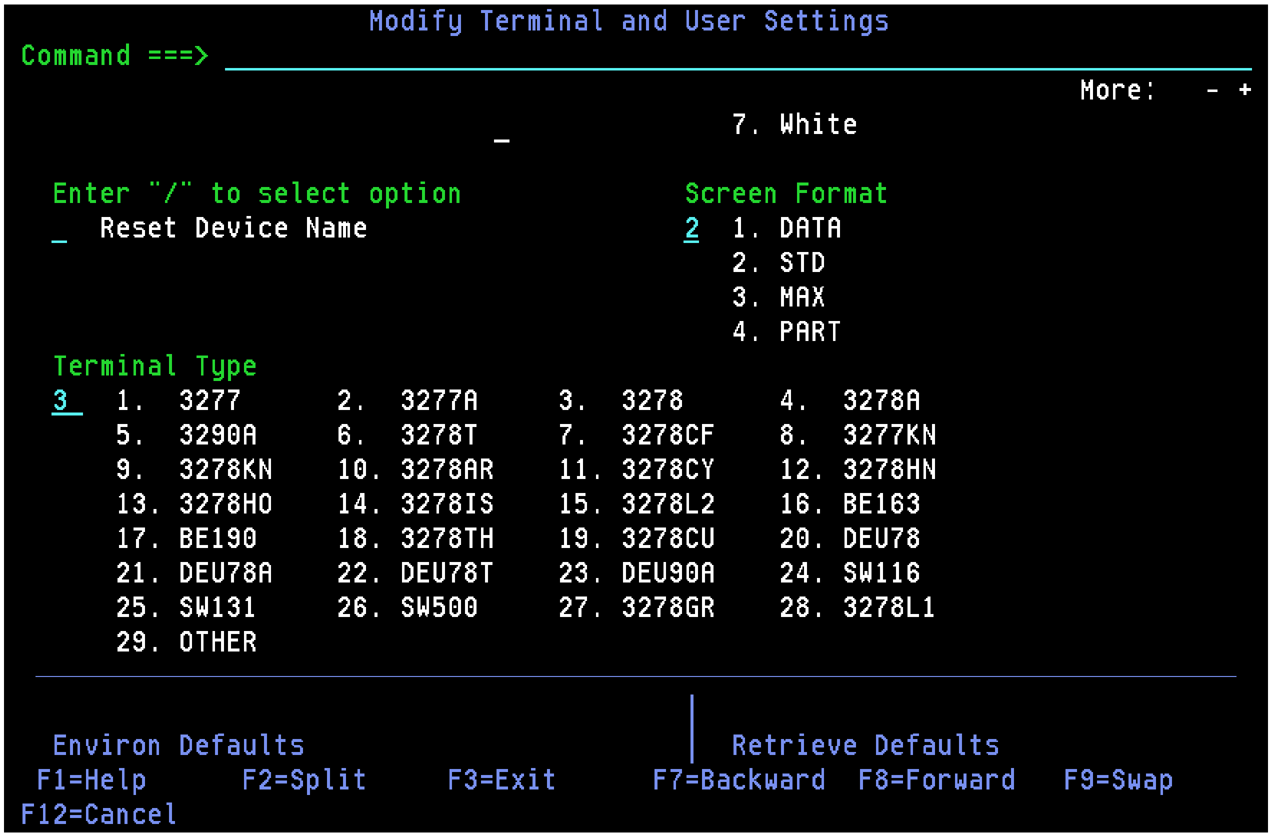
Figure 29. Modify Log/List Configuration Settings panel 3 (ISPPMOD6)

Selecting Option 7, **Terminal and User Defaults** on the Create/Modify ISPF Configuration panel (ISPPMOD) displays the Modify Terminal and User Settings panel (ISPPMOD7). Scroll down to display all the fields on this panel.



```
                      Modify Terminal and User Settings
Command ===> ________________________________________________________________
                                                                  More:     +
If you select any RESET fields in the sections: ISPF Site-wide Defaults, CUA
Color Settings, Log and List Defaults or Terminal and User Defaults you must
increment the Sitewide Defaults Version Level field to enable the RESET fields
you have selected. Increment only the last 3 digits of the Sitewide Defaults
Version Level. ISPF is always shipped with the Sitewide Defaults Version Level
field set to 43000. This value does not change with new versions or releases
of ISPF.

  Sitewide Defaults Version Level  . . 43000

                                          Window Frame Color/Intensity
  Settings Panel Defaults                 Color                  Intensity
  Aspect Ratio . . . . . . . . . 0        6  1. Blue             2  0. Low
  Input Field Pad Character      B           2. Red                 2. High
  Delimiter  . . . . . . . . . . ;           3. Pink
  Family Printer . . . . . . . : 2           4. Green
  Device Name  . . . . . . . . . ________    5. Turquoise
                                             6. Yellow
                                             7. White
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 30. Modify Terminal and User Settings panel 1 (ISPPMOD7)



```
                      Modify Terminal and User Settings
Command ===> ________________________________________________________________
                                                                  More:   - +
                                             7. White
                               _

  Enter "/" to select option              Screen Format
  _  Reset Device Name                    2  1. DATA
                                             2. STD
                                             3. MAX
                                             4. PART
  Terminal Type
  3_  1.  3277       2.  3277A      3.  3278       4.  3278A
      5.  3290A      6.  3278T      7.  3278CF     8.  3277KN
      9.  3278KN     10. 3278AR     11. 3278CY     12. 3278HN
      13. 3278HO     14. 3278IS     15. 3278L2     16. BE163
      17. BE190      18. 3278TH     19. 3278CU     20. DEU78
      21. DEU78A     22. DEU78T     23. DEU90A     24. SW116
      25. SW131      26. SW500      27. 3278GR     28. 3278L1
      29. OTHER
 ____________________________________________________________________________
                                          |
  Environ Defaults                        |  Retrieve Defaults
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 31. Modify Terminal and User Settings panel 2 (ISPPMOD7)

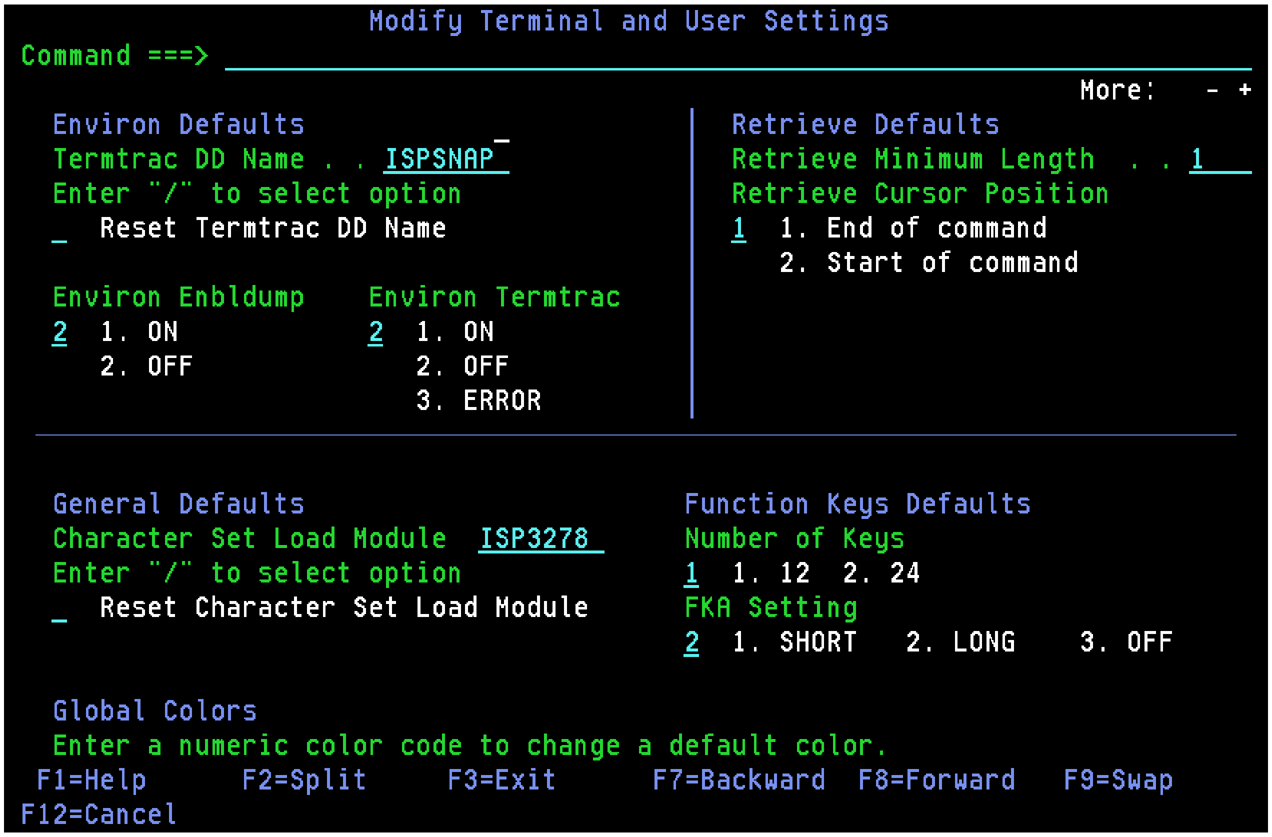

```
                      Modify Terminal and User Settings
Command ===> ______________________________________________________________
                                                                More:    - +
  Environ Defaults                    |  Retrieve Defaults
  Termtrac DD Name . . ISPSNAP_       |  Retrieve Minimum Length  . . 1___
  Enter "/" to select option          |  Retrieve Cursor Position
  _  Reset Termtrac DD Name           |  1  1. End of command
                                      |     2. Start of command
  Environ Enbldump    Environ Termtrac|
  2  1. ON            2  1. ON        |
     2. OFF              2. OFF       |
                         3. ERROR     |
 ____________________________________________________________________________

  General Defaults                      Function Keys Defaults
  Character Set Load Module  ISP3278_   Number of Keys
  Enter "/" to select option            1  1. 12  2. 24
  _  Reset Character Set Load Module    FKA Setting
                                        2  1. SHORT   2. LONG    3. OFF

  Global Colors
  Enter a numeric color code to change a default color.
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 32. Modify Terminal and User Settings panel 3 (ISPPMOD7)

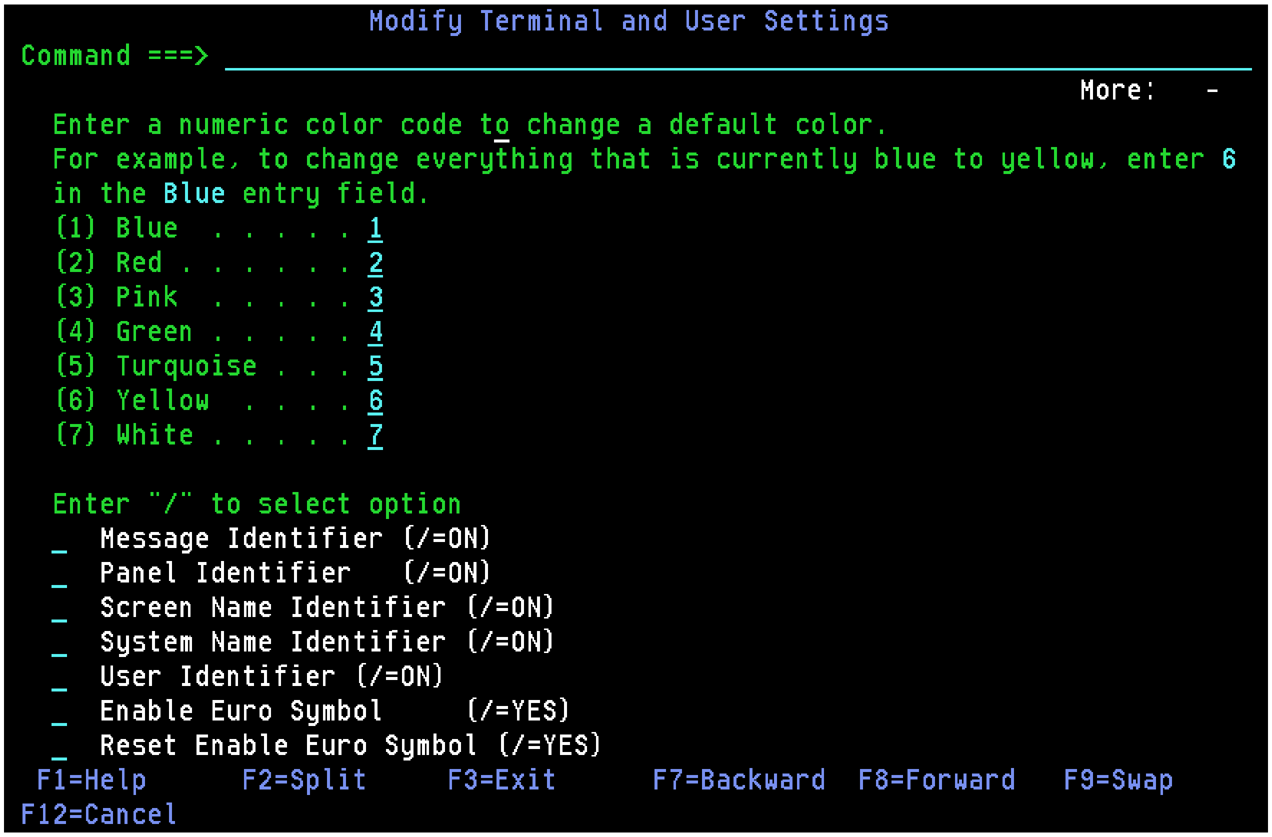

```
                      Modify Terminal and User Settings
Command ===> ______________________________________________________________
                                                                More:    -
  Enter a numeric color code to change a default color.
  For example, to change everything that is currently blue to yellow, enter 6
  in the Blue entry field.
  (1) Blue  . . . . . . 1
  (2) Red . . . . . . . 2
  (3) Pink  . . . . . . 3
  (4) Green . . . . . . 4
  (5) Turquoise . . . . 5
  (6) Yellow  . . . . . 6
  (7) White . . . . . . 7

  Enter "/" to select option
  _  Message Identifier (/=ON)
  _  Panel Identifier   (/=ON)
  _  Screen Name Identifier (/=ON)
  _  System Name Identifier (/=ON)
  _  User Identifier (/=ON)
  _  Enable Euro Symbol      (/=YES)
  _  Reset Enable Euro Symbol (/=YES)
 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 33. Modify Terminal and User Settings panel 4 (ISPPMOD7)

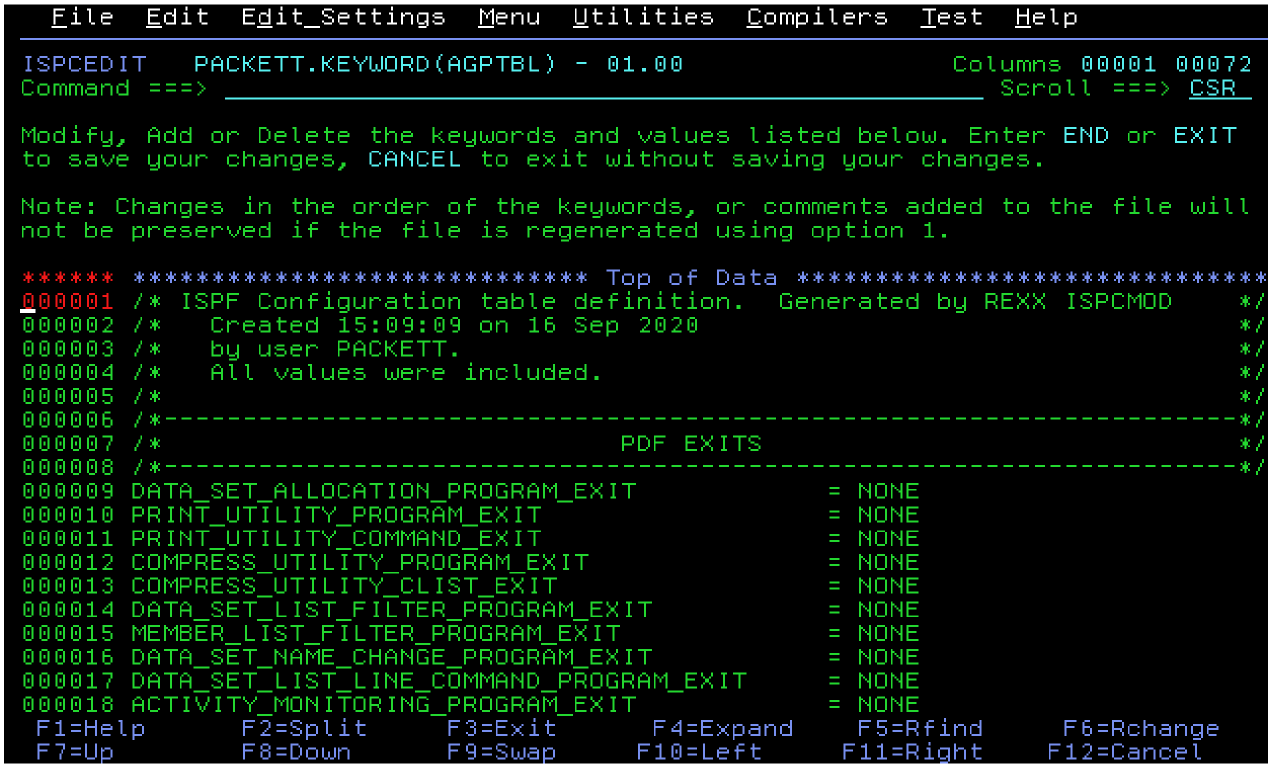
Edit Keyword File Configuration Table

If you choose Option 2, **Edit Keyword File Configuration Table** on the ISPF Configuration utility main menu panel, the Edit Keyword File panel appears, as shown in Figure 34 on page 30.

This option enables you to directly edit the keyword file and manually modify the configuration settings.

Comments can be added to the keyword file and the keywords can be rearranged, but such additions and changes are not preserved if you later use Option 1 **Create/Modify Settings and Regenerate Keyword File**. Only the changed values of keywords are carried forward. Full line comments are denoted by an asterisk (*) in column one, or a forward slash and asterisk (/*) in columns one and two. Comments on the same line as a keyword begin with a forward slash and asterisk, and end at the end of the line.

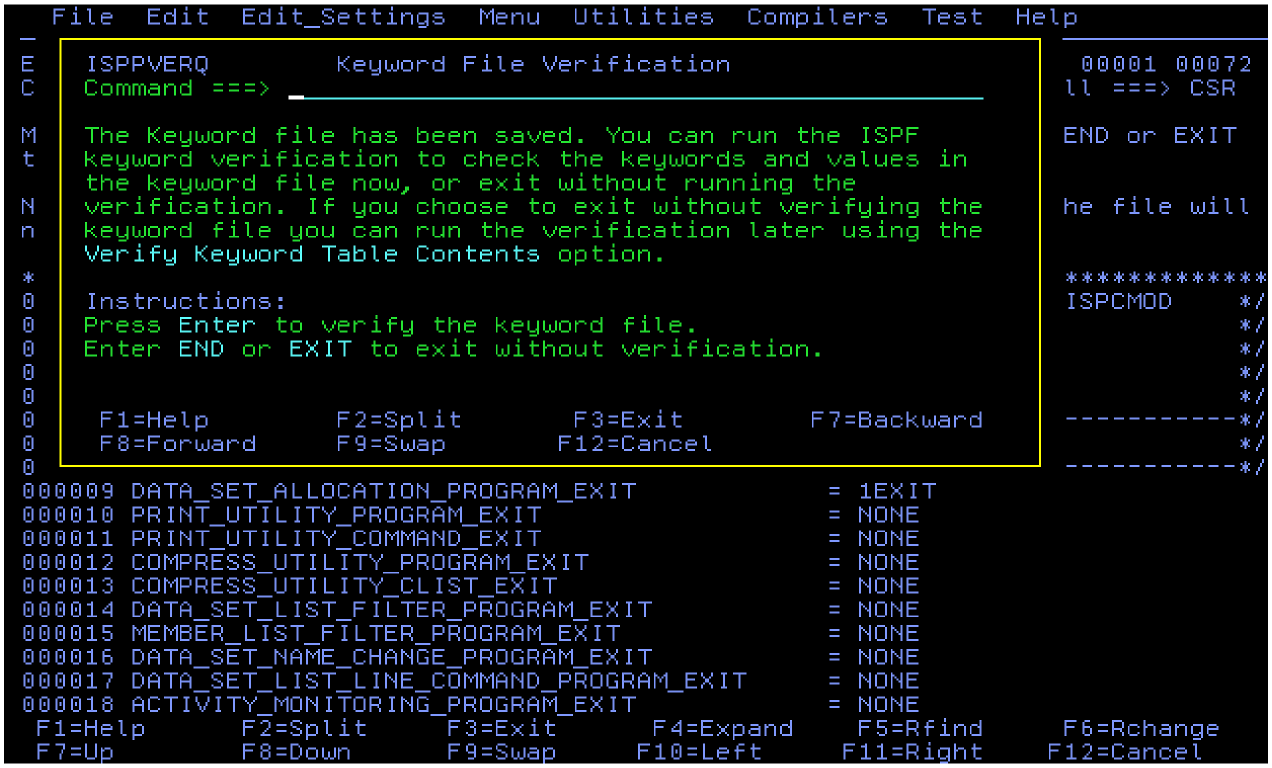
Note: If Workstation Defaults keywords or Workstation Download Defaults keywords appear in the keyword file, they are ignored.



```
   File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ISPCEDIT   PACKETT.KEYWORD(AGPTBL) - 01.00                 Columns 00001 00072
 Command ===> ______________________________________________   Scroll ===> CSR

 Modify, Add or Delete the keywords and values listed below. Enter END or EXIT
 to save your changes, CANCEL to exit without saving your changes.

 Note: Changes in the order of the keywords, or comments added to the file will
 not be preserved if the file is regenerated using option 1.

 ****** ***************************** Top of Data ******************************
 000001 /* ISPF Configuration table definition.  Generated by REXX ISPCMOD    */
 000002 /*   Created 15:09:09 on 16 Sep 2020                                  */
 000003 /*   by user PACKETT.                                                 */
 000004 /*   All values were included.                                        */
 000005 /*                                                                    */
 000006 /*--------------------------------------------------------------------*/
 000007 /*                              PDF EXITS                             */
 000008 /*--------------------------------------------------------------------*/
 000009 DATA_SET_ALLOCATION_PROGRAM_EXIT            = NONE
 000010 PRINT_UTILITY_PROGRAM_EXIT                  = NONE
 000011 PRINT_UTILITY_COMMAND_EXIT                  = NONE
 000012 COMPRESS_UTILITY_PROGRAM_EXIT               = NONE
 000013 COMPRESS_UTILITY_CLIST_EXIT                 = NONE
 000014 DATA_SET_LIST_FILTER_PROGRAM_EXIT           = NONE
 000015 MEMBER_LIST_FILTER_PROGRAM_EXIT             = NONE
 000016 DATA_SET_NAME_CHANGE_PROGRAM_EXIT           = NONE
 000017 DATA_SET_LIST_LINE_COMMAND_PROGRAM_EXIT     = NONE
 000018 ACTIVITY_MONITORING_PROGRAM_EXIT            = NONE
  F1=Help      F2=Split     F3=Exit      F4=Expand    F5=Rfind     F6=Rchange
  F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 34. Edit Keyword File panel (ISPCEDIT)

If you make changes to the keyword file, when you end the edit session ISPF displays the confirmation window in Figure 35 on page 31.



```
   File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ─  ┌──────────────────────────────────────────────────────────────┐ ─────────────
 E  │ ISPPVERQ        Keyword File Verification                   │  00001 00072
 C  │ Command ===> _                                               │ ll ===> CSR
    │                                                              │
 M  │ The Keyword file has been saved. You can run the ISPF        │ END or EXIT
 t  │ keyword verification to check the keywords and values in     │
    │ the keyword file now, or exit without running the            │
 N  │ verification. If you choose to exit without verifying the    │ he file will
 n  │ keyword file you can run the verification later using the    │
    │ Verify Keyword Table Contents option.                        │
 *  │                                                              │ *************
 0  │ Instructions:                                                │ ISPCMOD    */
 0  │ Press Enter to verify the keyword file.                      │            */
 0  │ Enter END or EXIT to exit without verification.              │            */
 0  │                                                              │            */
 0  │                                                              │            */
 0  │  F1=Help        F2=Split       F3=Exit        F7=Backward    │ -----------*/
 0  │  F8=Forward     F9=Swap       F12=Cancel                     │            */
 0  └──────────────────────────────────────────────────────────────┘ -----------*/
 000009 DATA_SET_ALLOCATION_PROGRAM_EXIT            = 1EXIT
 000010 PRINT_UTILITY_PROGRAM_EXIT                  = NONE
 000011 PRINT_UTILITY_COMMAND_EXIT                  = NONE
 000012 COMPRESS_UTILITY_PROGRAM_EXIT               = NONE
 000013 COMPRESS_UTILITY_CLIST_EXIT                 = NONE
 000014 DATA_SET_LIST_FILTER_PROGRAM_EXIT           = NONE
 000015 MEMBER_LIST_FILTER_PROGRAM_EXIT             = NONE
 000016 DATA_SET_NAME_CHANGE_PROGRAM_EXIT           = NONE
 000017 DATA_SET_LIST_LINE_COMMAND_PROGRAM_EXIT     = NONE
 000018 ACTIVITY_MONITORING_PROGRAM_EXIT            = NONE
  F1=Help      F2=Split     F3=Exit      F4=Expand    F5=Rfind     F6=Rchange
  F7=Up        F8=Down      F9=Swap     F10=Left     F11=Right    F12=Cancel
```

Figure 35. Keyword File Verification panel (ISPPVERQ)

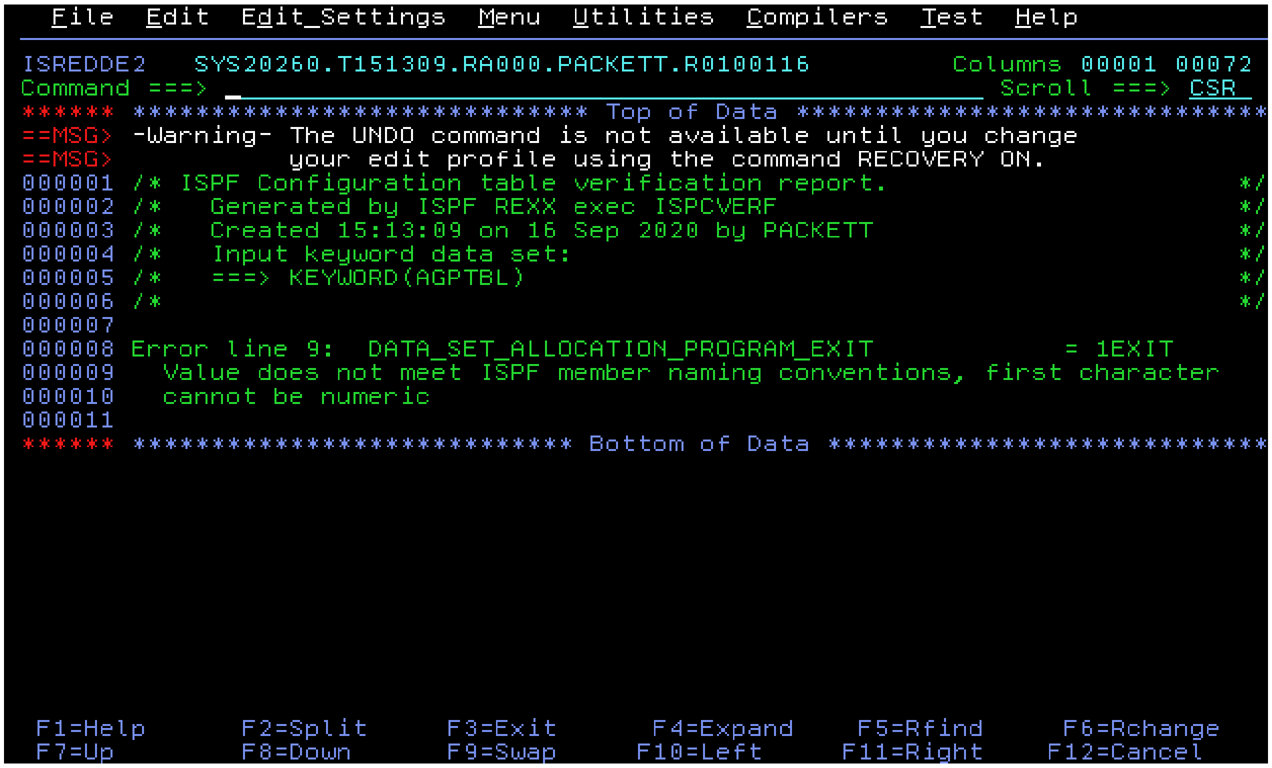
On this Keyword File Verification window, you can indicate whether you want to run ISPF's keyword verification routine to check the keywords and values in your keyword file. If you want to run the verification exec, press the Enter key. If you do not want to verify your changes, enter the END command, or EXIT.

Verify Keyword Table Contents

If you choose Option 3, **Verify Keyword Table Contents** on the ISPF Configuration utility main menu panel, ISPF verifies that the keyword file is correct in these respects:

- each record is in the correct format
- each keyword specified is a recognized keyword
- each value is syntactically correct.

If any verification errors are found, a listing similar to the one shown in Figure 36 on page 32 is displayed. The listing shows the line in error and the reason that ISPF flagged it as incorrect.



```
  File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
 ISREDDE2   SYS20260.T151309.RA000.PACKETT.R0100116         Columns 00001 00072
 Command ===> _                                                Scroll ===> CSR
 ****** ***************************** Top of Data ******************************
 ==MSG> -Warning- The UNDO command is not available until you change
 ==MSG>           your edit profile using the command RECOVERY ON.
 000001 /* ISPF Configuration table verification report.                      */
 000002 /*   Generated by ISPF REXX exec ISPCVERF                             */
 000003 /*   Created 15:13:09 on 16 Sep 2020 by PACKETT                       */
 000004 /*   Input keyword data set:                                          */
 000005 /*   ===> KEYWORD(AGPTBL)                                             */
 000006 /*                                                                    */
 000007
 000008 Error line 9:  DATA_SET_ALLOCATION_PROGRAM_EXIT            = 1EXIT
 000009   Value does not meet ISPF member naming conventions, first character
 000010   cannot be numeric
 000011
 ****** **************************** Bottom of Data ****************************

  F1=Help       F2=Split      F3=Exit       F4=Expand     F5=Rfind      F6=Rchange
  F7=Up         F8=Down       F9=Swap      F10=Left      F11=Right     F12=Cancel
```

Figure 36. Verification Failure Listing

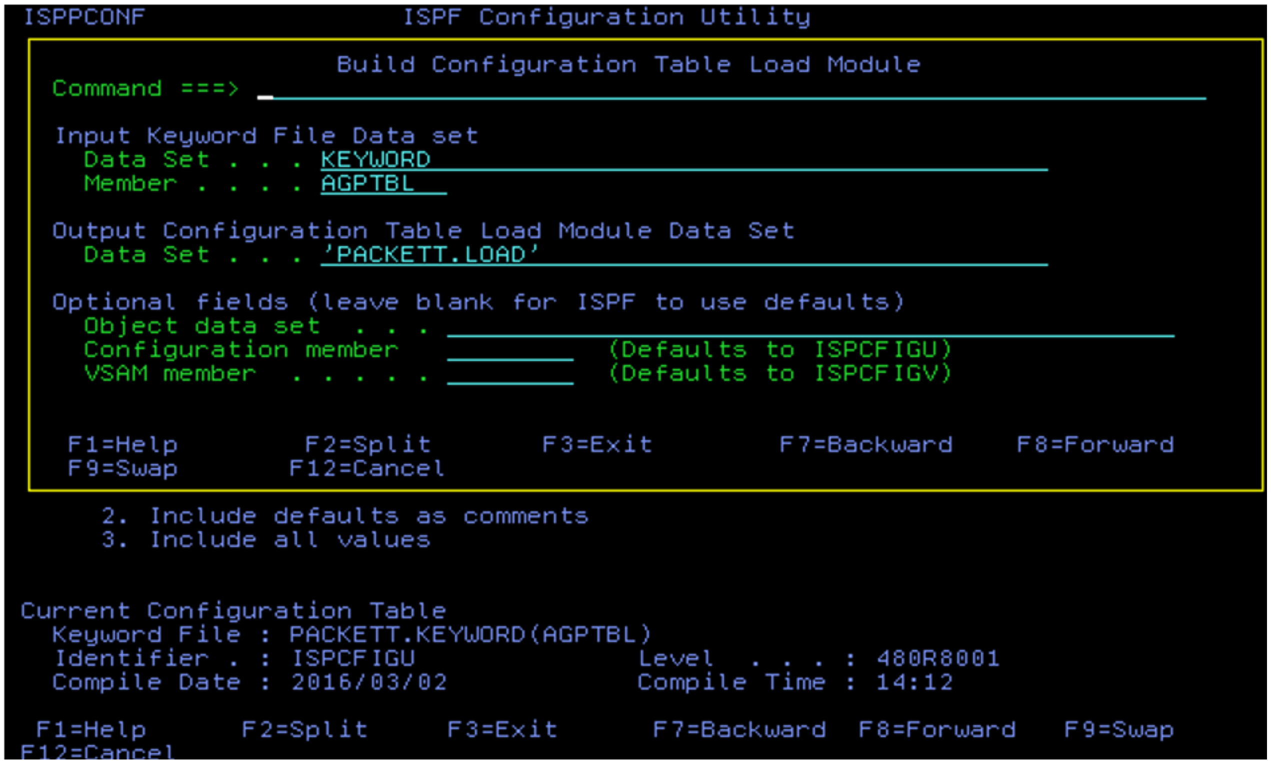
Build Configuration Table Load Module

If you choose Option 4, **Build Configuration Table Load Module** on the ISPF Configuration utility main menu panel, ISPF displays the Build Configuration Table Load Module panel, as shown in Figure 37 on page 33.

This option enables you to convert the keyword file you specify into a load module for ISPF to use to determine the session settings. You can create two separate load modules: the configuration table load module (default name ISPCFIGU), and the VSAM configuration load module (default name ISPCFIGV). The configuration table load module is always created, the VSAM configuration load module is only created if any of the VSAM restriction keywords (VSAM_RESTRICTED_EDIT_DATASET, VSAM_RESTRICTED_BROWSE_DATASET, VSAM_RESTRICTED_VIEW_DATASET, or VSAM_RESTRICTED_ALL_DATASET) is used in your keyword file. These VSAM restriction keywords are associated with the fields in the Restricted Data Sets section of the Modify Edit/View/Browse VSAM Settings panel (ISPPMOD2) of the ISPF Configuration Utility.

Note: If you convert your keyword file when it contains any of the VSAM restriction keywords, a VSAM configuration load module is created. If you later remove every VSAM restriction keyword from the keyword file and convert it again, the VSAM configuration load module is not created. However, the previously created VSAM configuration load module is not deleted. Because the ISPF Configuration Utility does not know the name that was used for the previously created VSAM configuration load module, it is your responsibility to delete it from all locations where it exists.

While you can create the load modules with any name initially (for example, if you want to create several different configurations for your installation), ISPF only recognizes the default names of ISPCFIGU and ISPCFIGV. If you create load modules with other names and want to use them, you must rename them when they are moved into the ISPF execution data sets.



```
ISPPCONF                    ISPF Configuration Utility

                      Build Configuration Table Load Module
  Command ===> _

  Input Keyword File Data set
    Data Set . . . KEYWORD
    Member . . . . AGPTBL

  Output Configuration Table Load Module Data Set
    Data Set . . . 'PACKETT.LOAD'

  Optional fields (leave blank for ISPF to use defaults)
    Object data set  . . .
    Configuration member            (Defaults to ISPCFIGU)
    VSAM member  . . . . .          (Defaults to ISPCFIGV)


   F1=Help        F2=Split       F3=Exit        F7=Backward    F8=Forward
   F9=Swap       F12=Cancel
      2. Include defaults as comments
      3. Include all values


Current Configuration Table
  Keyword File : PACKETT.KEYWORD(AGPTBL)
  Identifier . : ISPCFIGU                     Level  . . . : 480R8001
  Compile Date : 2016/03/02                   Compile Time : 14:12

 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 37. Build Configuration Table Load Module panel (ISPPBLD)

When the Build Configuration Table Load Module panel appears, the Input Keyword File Data Set field value is initialized with the name found on the ISPF Configuration utility main menu panel. Verification of the keyword file specified is automatically run before the keyword file is converted into a load module. If ISPF finds any errors during the verification process, they are logged to a temporary data set, and ISPF puts you into View mode on the messages data set when the verification is complete. The build *is not* done until the verification completes without error.

The other entry fields found on the Build Configuration Table Load Module panel are:

Output Configuration Table Load Module Data Set
The name of the pre-existing partitioned data set into which the generated configuration table load module is to be stored.

Object Data Set
Optional. The name of the pre-existing partitioned data set into which the generated configuration table object module is to be stored. If no object data set is specified, ISPF uses a temporary data set.

Configuration Member
Optional. The member name of the configuration table load module created. If no member name is specified, a value of ISPCFIGU is used.

VSAM Member
Optional. The member name of the VSAM configuration load module created. If no member name is specified, a value of ISPCFIGV is used.

Convert Assembler Configuration Table to Keyword File

If you choose Option 5, **Convert Assembler Configuration Table to Keyword File** on the ISPF Configuration utility main menu panel, ISPF converts a configuration table assembler source file used in releases of ISPF before OS/390® Version 2 Release 8.0 into the new keyword format.

You specify the assembler source file in the Configuration Table Assembler Source Data Set field on the ISPF Configuration utility main menu panel. The keyword file that is generated is placed into the data set and member you specify in the Keyword File Data Set field.

The Output File Content for Keyword File field controls how much data is written to the keyword file during the conversion. Valid values for this field are:

1
Only those values that are different than the default values are included in the keyword file.

2
Those values different than the default values are included in the keyword file, and all default values are included as comment lines.

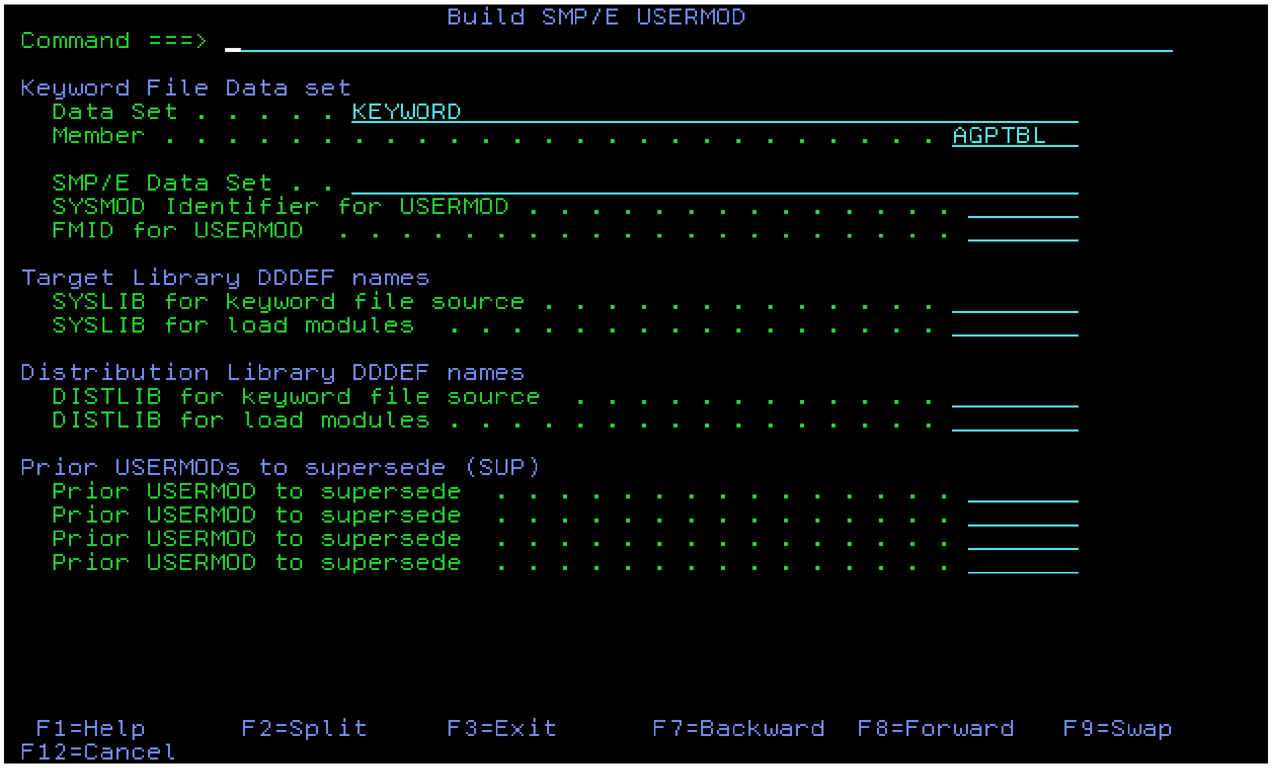
3
All values are included in the keyword file.

If the output keyword file specified in the Keyword File Data Set field already exists, the Confirm Conversion panel asks if you want to overwrite it. If you want to overwrite the file press the Enter key, otherwise enter END or CANCEL to exit.

Build SMP/E USERMOD

The FMID for the FMID operand of the SMP/E USERMOD.

If you choose Option 6, **Build SMP/E USERMOD** on the ISPF Configuration utility main menu panel, ISPF displays the Build SMP/E USERMOD panel (ISPPSMP), as shown in Figure 38 on page 34.



```
                            Build SMP/E USERMOD
Command ===> _

Keyword File Data set
  Data Set . . . . . KEYWORD
  Member . . . . . . . . . . . . . . . . . . . . . . . . AGPTBL

  SMP/E Data Set . .
  SYSMOD Identifier for USERMOD . . . . . . . . . . . . . .
  FMID for USERMOD  . . . . . . . . . . . . . . . . . . . .

Target Library DDDEF names
  SYSLIB for keyword file source . . . . . . . . . . . . .
  SYSLIB for load modules  . . . . . . . . . . . . . . . .

Distribution Library DDDEF names
  DISTLIB for keyword file source  . . . . . . . . . . . .
  DISTLIB for load modules . . . . . . . . . . . . . . . .

Prior USERMODs to supersede (SUP)
  Prior USERMOD to supersede  . . . . . . . . . . . . . . .
  Prior USERMOD to supersede  . . . . . . . . . . . . . . .
  Prior USERMOD to supersede  . . . . . . . . . . . . . . .
  Prior USERMOD to supersede  . . . . . . . . . . . . . . .




 F1=Help      F2=Split     F3=Exit      F7=Backward  F8=Forward   F9=Swap
F12=Cancel
```

Figure 38. Build SMP/E USERMOD panel (ISPPSMP)

This option enables you to convert the keyword file you specify into load modules (a configuration table load module and a VSAM configuration load module), and to package the keyword file source code and the generated load modules in an SMP/E USERMOD. The configuration table load module is always created. The VSAM configuration load module is only created if one of the VSAM restriction keywords is used in your keyword file. The default names of ISPCFIGU (for the configuration load module) and ISPCFIGV (for the VSAM configuration load module) are always used when the load modules are packaged in an SMP/E USERMOD.

When the Build SMP/E USERMOD panel appears, the Keyword File Data Set field is initialized with the name found on the ISPF Configuration utility main menu panel. All entry fields on the Build SMP/E

USERMOD panel are required except the "Prior USERMOD to supersede" fields. The other entry fields on the Build SMP/E USERMOD panel are:

SMP/E Data Set
The name of the pre-existing partitioned data set into which the SMP/E USERMOD is to be stored. The USERMOD is stored in a member named the same as the SYSMOD ID used for the USERMOD. This data set must be record format FB with LRECL 80.

SYSMOD Identifier for USERMOD
A seven-character identifier for the SMP/E USERMOD. The identifier is used in the `++USERMOD` statement and is used for the member name in the SMP/E Data set.

FMID for USERMOD
The FMID for the FMID operand of the SMP/E USERMOD. The FMID for ISPF z/OS Version 3 Release 2.0 is HIF83B2.

Target Library DDDEF names

SYSLIB for keyword file source
The DDDEF name to be used as the SYSLIB for the keyword source in the SMP/E USERMOD. The DDDEF must exist in the target zone that has ISPF installed. The data set in the DDDEF entry must have the same record format and LRECL as your keyword source data set.

SYSLIB for load modules
The DDDEF name to be used as the SYSLIB for the load modules in the SMP/E USERMOD. The DDDEF must exist in the target zone that has ISPF installed. The data set in the DDDEF entry must have record format U and LRECL 0 with a block size equal to or greater than 6144.

Distribution Library DDDEF names

DISTLIB for keyword file source
The DDDEF name to be used as the DISTLIB for the keyword source in the SMP/E USERMOD. The DDDEF must exist in the target zone and distribution zone that has ISPF installed. The data set in the DDDEF entry must have the same record format and LRECL as your keyword source data set.

DISTLIB for load modules
The DDDEF name to be used as the DISTLIB for the load modules in the SMP/E USERMOD. The DDDEF must exist in the target zone and distribution zone that has ISPF installed. The data set in the DDDEF entry must have record format U and LRECL 0 with a block size equal to or greater than 6144.

Prior USERMODs to supersede (SUP)
Four optional fields for the seven-character name of a previous USERMOD to be superseded by this SMP/E USERMOD. The previous USERMODs (up to four) are included in the SUP operand of the SMP/E USERMOD. The four fields must be filled from top to bottom.

When you complete the required fields on the panel and press Enter, the keyword file specified is automatically verified before it is converted into load modules and an SMP/E USERMOD is built. If ISPF finds any errors during verification, the errors are logged to a temporary data set and ISPF puts you into View mode on the messages data set.

If the verification process completes without error the keyword file is packaged in an SMP/E USERMOD as **++DATA (ISPCFIGU)**. The configuration load module is generated and packaged as **++PROGRAM (ISPCFIGU)**.

The VSAM load module, if required, is generated and packaged as **++PROGRAM (ISPCFIGV)**.

The SMP/E USERMOD is stored in the data set specified in the SMP/E Data Set field on the Build SMP/E USERMOD panel (ISPPSMP), using a member name equal to the SYSMOD ID specified in the SYSMOD Identifier for USERMOD field.

Installing the USERMOD on your system

To install the resulting SMP/E USERMOD on your system, perform these steps.

1. Create target and distribution libraries for the keyword source file. The target and distribution libraries for the keyword source must have the same record format and LRECL as the keyword file that you used to build the USERMOD. One or two tracks will be sufficient space.
2. Create target and distribution libraries for the load modules. The target and distribution load libraries must be record format U, LRECL 0, with a block size equal to or greater than 6144. One or two tracks will be sufficient space.
3. Add DDDEFs to your target zone for the target and distribution libraries using the DDDEF names you specified on the Build SMP/E USERMOD panel, with the names of the appropriate target or distribution libraries you created.
4. Add DDDEFs to your distribution zone for the distribution libraries using the DDDEF names you specified on the Build SMP/E USERMOD panel, with the names of the appropriate distribution libraries you created.
5. SMP/E RECEIVE and APPLY the USERMOD.
6. Place the load module target library in the normal MVS search order on your system or allocate it to ISPLLIB in your logon procedure.

Note:

1. You only need to create the distribution libraries and distribution DDDEFs if you SMP/E ACCEPT the USERMOD. However, the Distribution Library DDDEF names fields on the panel must be filled in when building the USERMOD to make the MCS statements in the USERMOD complete.
2. You might need an SMPTLOAD DDDEF in your target and distribution zone. The temporary data set for the ISPCFIGU and ISPCFIGV load modules is created with DSORG(PO) UNIT(SYSALLDA) and will be either a PDS or a PDSE depending on the setup of your system. The temporary load data set is IEBCOPY unloaded to produce the ++PROGRAM elements in the USERMOD. See the *z/OS SMP/E Reference* for more information about the requirement for SMPTLOAD.

Convert Configuration Table LoadMod to Keyword File

If you choose Option 7, **Convert Configuration Table LoadMod to Keyword File** on the ISPF Configuration utility main menu panel, ISPF converts a configuration table load module to a keyword file that can be used as input to Options 1 and 2. The active configuration or a module from a data set can be chosen.

This can be used to reconstruct source that may have been lost.

All keywords are produced.

Chapter 3. Customizing ISPF

Here are the ways in which you can customize ISPF.

Customizing action bars

Many ISPF action bar items are common to multiple panels. These action bar definitions are coded in separate files that are embedded into the various panels. The embedded DTL source fields are written in Generalized Markup Language (GML), as part of the DTL panel source.

To customize one or more action bar items:

1. Update the appropriate embed file from Table 1 on page 37. These embed files are distributed in the ISP.SISPGENU library.

Table 1. Embedded DTL source files

File name	**DTL File for...**
ISPDFIL1	File action bar choice
ISPFMENU	SCLM menu action bar choice
ISPDFUNC	Functions action bar choice
ISPDHUCM	Action bar choice
ISPDHYXM	Action bar choice
ISPFJOBC	Jobcard action bar choice
ISPDLANG	Compilers action bar choice
ISPDMENU	Menu action bar choice
ISPDREFE	Reflist action bar choice
ISPDREFL	Reflist action bar choice
ISPDREFM	Refmode action bar choice
ISPDSAVE	Save action bar choice
ISPFSCLM	SCLM action bar choice
ISPDTEST	Test action bar choice
ISPDUTIL	Utilities action bar choice

2. Reconvert the affected panels by specifying the appropriate list of members (Table 2 on page 37) and running the ISPDTLC conversion utility.

Table 2. DTL list of panels

File name	**DTL List for...**
ISPFAB	All action bar GMLs
ISPFALL	All product GMLs
ISPFIL1	File action bar choice
ISPMENU1	SCLM menu action bar choice
ISPFUNC	Functions action bar choice
ISPHUCM	Action bar choice

Table 2. DTL list of panels (continued)

File name	DTL List for...
ISPHYXM	Action bar choice
ISPJOBC	Jobcard action bar choice
ISPLANG	Compilers action bar choice
ISPMENU	Menu action bar choice
ISPREFE	Reflist action bar choice
ISPREFL	Reflist action bar choice
ISPREFM	Refmode action bar choice
ISPSAVEB	Save action bar choice
ISPSCLM	SCLM action bar choice
ISPTEST	Test action bar choice
ISPUTIL	Utilities action bar choice

The lists of panels for different types of modifications are contained in ISP.SISPSAMP. ISPDTLC will convert all of the members in a list that are used as input member names.

Invoking the ISPDTLC conversion utility

ISPDTLC can be run either interactively or in a batch job. Because of the number of panels to be converted, a batch job might be the best choice. The information that follows describes both interactive and batch conversions.

Note: After conversion, you might want to preprocess the panels to improve performance. For more information about preprocessing panels, see "Preprocess all ISPF panels" on page 95.

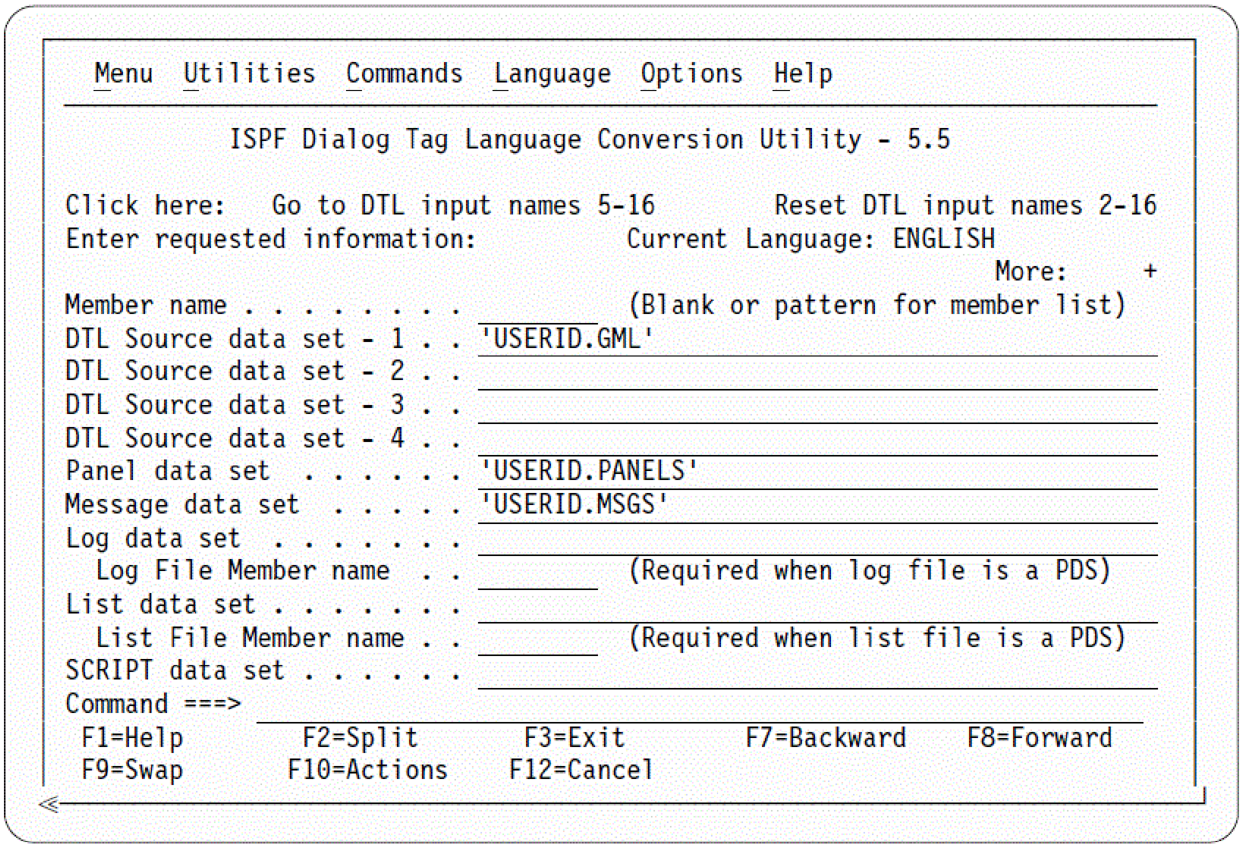
For more information about ISPDTLC invocation, refer to *z/OS ISPF Dialog Tag Language Guide and Reference*.

Running interactively

To run interactively, invoke the conversion utility from any command line by entering:

```
ISPDTLC
```

Figure 39 on page 39 shows the first screen of the ISPDTLC invocation panel with the default data set names and with the required options selected.



```
  Menu  Utilities  Commands  Language  Options  Help
 ------------------------------------------------------------------------------
            ISPF Dialog Tag Language Conversion Utility - 5.5

 Click here:   Go to DTL input names 5-16       Reset DTL input names 2-16
 Enter requested information:          Current Language: ENGLISH
                                                                More:     +
 Member name . . . . . . . . ________  (Blank or pattern for member list)
 DTL Source data set - 1 . . 'USERID.GML'
 DTL Source data set - 2 . . ______________________________________________
 DTL Source data set - 3 . . ______________________________________________
 DTL Source data set - 4 . . ______________________________________________
 Panel data set  . . . . . . 'USERID.PANELS'
 Message data set  . . . . . 'USERID.MSGS'
 Log data set  . . . . . . . ______________________________________________
   Log File Member name  . . ________  (Required when log file is a PDS)
 List data set . . . . . . . ______________________________________________
   List File Member name . . ________  (Required when list file is a PDS)
 SCRIPT data set . . . . . . ______________________________________________
 Command ===> _____________________________________________________________
  F1=Help        F2=Split       F3=Exit        F7=Backward     F8=Forward
  F9=Swap       F10=Actions    F12=Cancel
```

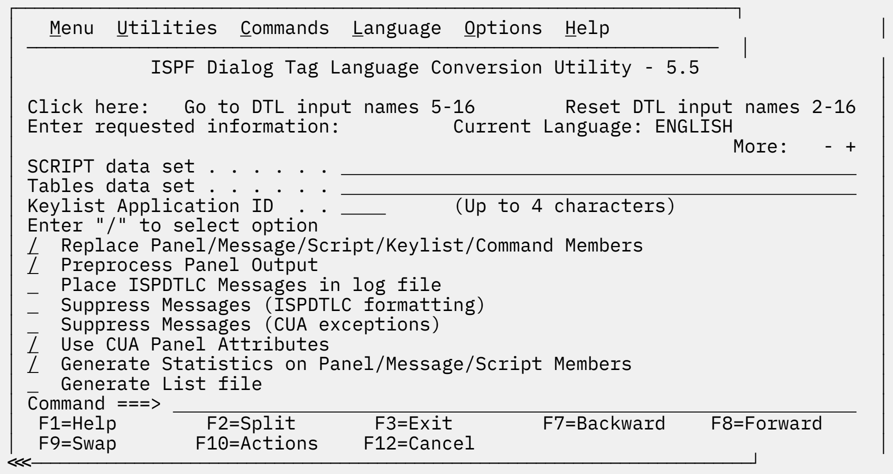
Figure 39. First ISPDTLC screen

The member name ISPMENU will cause all of the panels with the Menu action bar item to be converted.

When the interactive panel is displayed, change the data set names shown as required for your installation. Fill in the name of your input and output files where indicated. The first DTL source file should contain any local panel modifications. The second DTL source file contains the product panel source. The third DTL source file contains the conversion list member ISPMENU.

Note: The current language selection appears on all screens. A different language can be selected from the action bar.

After typing in your file names, scroll to the next screen (Figure 40 on page 39).

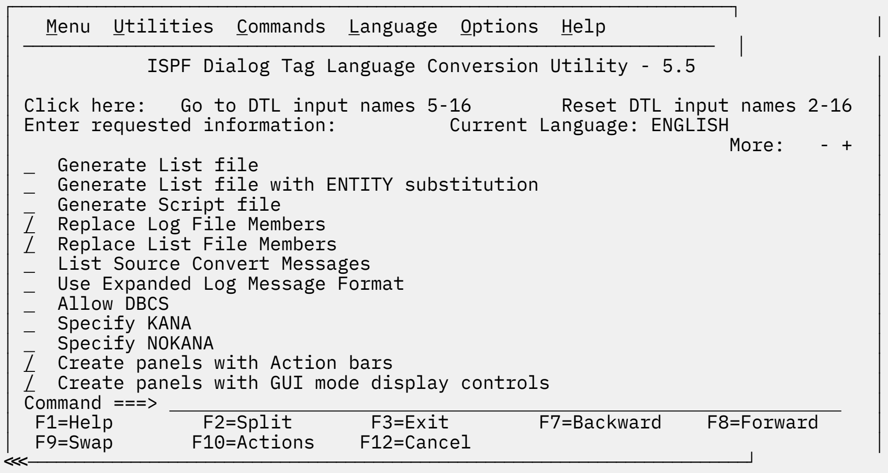


```
  Menu  Utilities  Commands  Language  Options  Help
 --------------------------------------------------------------
           ISPF Dialog Tag Language Conversion Utility - 5.5

 Click here:   Go to DTL input names 5-16       Reset DTL input names 2-16
 Enter requested information:          Current Language: ENGLISH
                                                           More:    - +
 SCRIPT data set . . . . . . ______________________________________________
 Tables data set . . . . . . ______________________________________________
 Keylist Application ID  . . ____       (Up to 4 characters)
 Enter "/" to select option
 /  Replace Panel/Message/Script/Keylist/Command Members
 /  Preprocess Panel Output
 _  Place ISPDTLC Messages in log file
 _  Suppress Messages (ISPDTLC formatting)
 _  Suppress Messages (CUA exceptions)
 /  Use CUA Panel Attributes
 /  Generate Statistics on Panel/Message/Script Members
 _  Generate List file
 Command ===> ____________________________________________________________
  F1=Help        F2=Split       F3=Exit        F7=Backward     F8=Forward
  F9=Swap       F10=Actions    F12=Cancel
```

Figure 40. Second ISPDTLC screen

Note: To reduce the amount of screen output you can select options Suppress Messages (ISPDTLC formatting) and Suppress Messages (CUA exceptions).

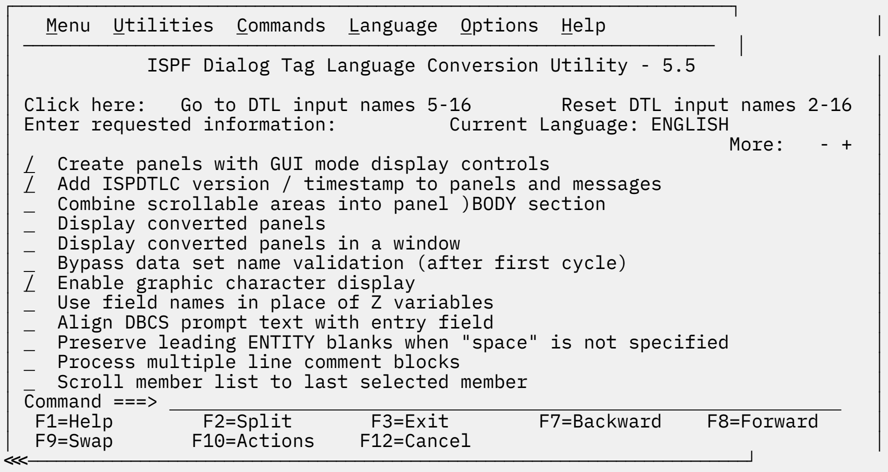
Enter ISR as the Keylist Application ID and select the options as shown. Scroll to the next screen (Figure 41 on page 40).



```
  Menu  Utilities  Commands  Language  Options  Help
 
           ISPF Dialog Tag Language Conversion Utility - 5.5

Click here:   Go to DTL input names 5-16       Reset DTL input names 2-16
Enter requested information:          Current Language: ENGLISH
                                                               More:   - +
_  Generate List file
_  Generate List file with ENTITY substitution
_  Generate Script file
/  Replace Log File Members
/  Replace List File Members
_  List Source Convert Messages
_  Use Expanded Log Message Format
_  Allow DBCS
_  Specify KANA
_  Specify NOKANA
/  Create panels with Action bars
/  Create panels with GUI mode display controls
Command ===> ____________________________________________________________
 F1=Help        F2=Split       F3=Exit        F7=Backward    F8=Forward
 F9=Swap       F10=Actions    F12=Cancel
```

Figure 41. Third ISPDTLC screen

Scroll to the next screen (Figure 42 on page 40).



```
  Menu  Utilities  Commands  Language  Options  Help
 
           ISPF Dialog Tag Language Conversion Utility - 5.5

Click here:   Go to DTL input names 5-16       Reset DTL input names 2-16
Enter requested information:          Current Language: ENGLISH
                                                               More:   - +
/  Create panels with GUI mode display controls
/  Add ISPDTLC version / timestamp to panels and messages
_  Combine scrollable areas into panel )BODY section
_  Display converted panels
_  Display converted panels in a window
_  Bypass data set name validation (after first cycle)
/  Enable graphic character display
_  Use field names in place of Z variables
_  Align DBCS prompt text with entry field
_  Preserve leading ENTITY blanks when "space" is not specified
_  Process multiple line comment blocks
_  Scroll member list to last selected member
Command ===> ____________________________________________________________
 F1=Help        F2=Split       F3=Exit        F7=Backward    F8=Forward
 F9=Swap       F10=Actions    F12=Cancel
```

Figure 42. Fourth ISPDTLC screen

Scroll to the last screen (Figure 43 on page 41).

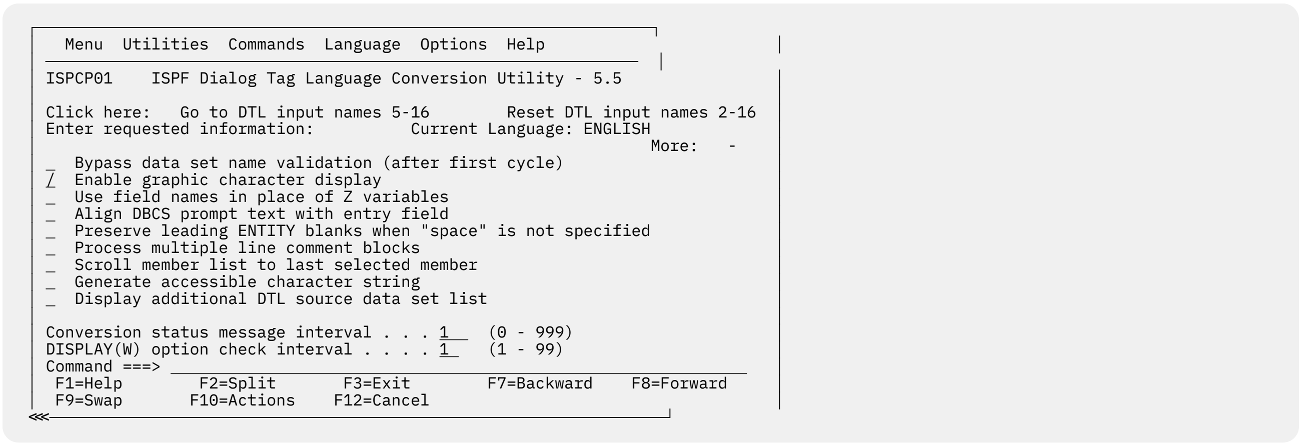

```
   Menu  Utilities  Commands  Language  Options  Help
 ──────────────────────────────────────────────────────────────
 ISPCP01    ISPF Dialog Tag Language Conversion Utility - 5.5

 Click here:   Go to DTL input names 5-16        Reset DTL input names 2-16
 Enter requested information:          Current Language: ENGLISH
                                                               More:   -
 _  Bypass data set name validation (after first cycle)
 /  Enable graphic character display
 _  Use field names in place of Z variables
 _  Align DBCS prompt text with entry field
 _  Preserve leading ENTITY blanks when "space" is not specified
 _  Process multiple line comment blocks
 _  Scroll member list to last selected member
 _  Generate accessible character string
 _  Display additional DTL source data set list

 Conversion status message interval . . . 1    (0 - 999)
 DISPLAY(W) option check interval . . . . 1    (1 - 99)
 Command ===> ____________________________________________________________
  F1=Help        F2=Split       F3=Exit        F7=Backward    F8=Forward
  F9=Swap       F10=Actions    F12=Cancel
```

Figure 43. Fifth ISPDTLC screen

Note: ISPF product panels require that the options "Preserve leading ENTITY blanks when "space" is not specified" and "Process multiple line comment blocks" be selected as shown in the figures here.

If a double byte language conversion is in process, the DBCS and KANA options might be required.

If you are recompiling the English version of the panels, a second conversion should be made for the uppercased English panels. You will need to enter a different output panel library name and select the UPPERENG language from the action bar. All of the other panel information and options should be the same as the first conversion.

Running in batch mode

The batch conversion requires a profile data set that contains ddnames and data set names for the input and output files. The sample JCL shown in Figure 44 on page 41 refers to the profile data set. The ISPDTLC invocation syntax specifies all of the required options for the conversion.

Change the data set names shown in both the profile and the JCL as required for your location.

```
//your jobcard here
//CONVERT  EXEC PGM=IKJEFT01,DYNAMNBR=50
//SYSPRINT DD SYSOUT=*
//SYSPROC  DD  DISP=SHR,DSN=your.exec.dataset
//SYSEXEC  DD  DISP=SHR,DSN=ISP.SISPEXEC
//STEPLIB  DD  DISP=SHR,DSN=ISP.SISPLOAD
//         DD  DISP=SHR,DSN=ISP.SISPLPA
//ISPLLIB  DD  DISP=SHR,DSN=ISP.SISPLOAD
//         DD  DISP=SHR,DSN=ISP.SISPLPA
//ISPMLIB  DD  DISP=SHR,DSN=ISP.SISPMENU
//ISPPLIB  DD  DISP=SHR,DSN=ISP.SISPPENU
//ISPTLIB  DD  DISP=SHR,DSN=your.profile.dataset
//         DD  DISP=SHR,DSN=ISP.SISPTENU
//ISPSLIB  DD  DISP=SHR,DSN=ISP.SISPSENU
//ISPTABL  DD  DISP=OLD,DSN=your.profile.dataset
//ISPPROF  DD  DISP=OLD,DSN=your.profile.dataset
//ISPLOG   DD  DISP=OLD,DSN=your.log.dataset
//ISPLIST  DD  DISP=OLD,DSN=your.list.dataset
//SYSTSPRT DD  SYSOUT=*
//SYSTSIN  DD  *
  PROFILE PREFIX(USERAA)
  ISPSTART CMD(CONVACTB)
```

Figure 44. Sample JCL for Batch conversion

The CONVACTB EXEC (written in REXX) is:

```
'ISPDTLC ISPMENU (DISK KEYLAPPL=ISR MSGSUPP CUAATTR CUASUPP PLEB MCOMMENT
    NOSTATS NOACTBAR PROFILE=your.gml.dataset(profile)' ;
```

This EXEC should be placed on *your.exec.dataset*.

Note:

1. If you are recompiling the English version of the panels, you must run ISPDTLC two times. The second run is to create the uppercase English version of the panel. The UPPERENG language keyword is specified to create the uppercase panel version. A different profile is also used with a different panel library defined to DTLPAN.

   ```
   'ISPDTLC ISPMENU (DISK KEYLAPPL=ISR MSGSUPP CUAATTR CUASUPP PLEB MCOMMENT
       UPPERENG NOSTATS NOACTBAR PROFILE=your.gml.dataset(profileu)' ;
   ```

2. If you are recompiling DBCS versions of the panels, you must add the DBCS option to the command syntax. Japanese panels require the KANA option as well.

   ```
   'ISPDTLC ISPFALL (DISK KEYLAPPL=ISR MSGSUPP CUAATTR CUASUPP PLEB MCOMMENT
       JAPANESE DBCS KANA NOSTATS NOACTBAR PROFILE=your.gml.dataset(profile)' ;
   ```

your.gml.dataset(profile) should include:

DTLGML
    your.appl.gml.dataset

DTLGML
    ISP.SISPGENU

DTLGML
    ISP.SISPGMLI

DTLGML
    ISP.SISPSAMP

DTLPAN
    your.output.panel.dataset

DTLMSG
    your.output.message.dataset

DTLLOG
    your.log.dataset

your.gml.dataset(profileu) for creating uppercased English panels should include:

DTLGML
    your.appl.gml.dataset

DTLGML
    ISP.SISPGENU

DTLGML
    ISP.SISPGMLI

DTLGML
    ISP.SISPSAMP

DTLPAN
    your.output.uppercase.panel.dataset

DTLMSG
    your.output.message.dataset

DTLLOG
    your.log.dataset

Your updated GML files should be stored in *your.appl.gml.dataset*. The converted files will be in *your.output.panel.dataset* or *your.output.uppercase.panel.dataset*, and the messages generated will be in *your.log.dataset*.

The log file will contain the results of the conversion processing. Most of the messages generated by ISPDTLC are related to ISPF extensions to the Dialog Tag Language and have been suppressed. The expected message number from compiling the panel source will appear in a comment near the top of the DTL source file for that panel source.

Customizing for profile sharing

Customizing for profile sharing shows how you can customize shared profiles within ISPF.

Note: Profile sharing is intended to be used by the same user logging concurrently onto multiple systems. The contents of the ISPF profile are often user-specific.

Customizing the ISPF Configuration utility

You can customize the ISPF Configuration utility with these configuration options:

- PROFILE_SHARING
- PROFILE_ENQLOCK_WAIT
- PROFILE_ENQLOCK_RETRY_COUNT
- PROFILE_ENQLOCK_PROMPT
- PROFILE_SYSPROF_CONFLICT
- PROFILE_ISPPROF_CONFLICT
- PROFILE_APPPROF_CONFLICT
- PROFILE_REFLIST_CONFLICT
- PROFILE_EDIT_CONFLICT
- PROFILE_BATCH_CONFLICT
- PROFILE_OTHER_CONFLICT
- RESET_PROFILE_SHARING_SETTINGS

For further details about configuration options, see "ISPF site-wide profile customizations" on page 228.

Temporary data set names

The ISPF Configuration utility provides an option to specify an additional qualifier for ISPF temporary data sets, including Log, List and temporary control and work data sets. The use of this qualifier also includes trace data sets from ISPVCALL, ISPDPTRC and ISPFTTRC. When Profile Sharing has been enabled and no value has been specified for the additional qualifier, a value of "ISP&SEQ" is used, where &SEQ is a system symbolic variable that has the same value as defined by the ISPF dialog variable ZSEQ.

SYSIKJUA enqueue

To enable the ISPF multi-logon support in a sysplex, you must comment out or remove the RNLDEF statement specifying SYSIKJUA in the GRSNL*xx* member of the system parmlib concatenation:

```
/* RNLDEF RNL(INCL) TYPE(GENERIC)           */
/* QNAME(SYSIKJUA)                          */
```

Limitation of other products

The following limitation applies to shared profiles:

TSO/E Reconnect
TSO/E users can only reconnect to their sessions from the same system where they were already logged on, not from a remote z/OS system.

Note: Other ISPF applications may experience unexpected results when they attempt to use the ISPF profile data set to store application specific information.

TSO/ISPF client gateway

ISPF provides a thin gateway service that allows applications to start a TSO address space, send a TSO or ISPF command as input, and receive a response as output. Callers of this service are typically applications that are serving remote clients that want to issue TSO or ISPF commands. This section describes how the gateway works, how to install and customize the gateway, and how client applications can use the gateway.

Starting from z/OS V2R2, the “Interactive ISPF Gateway” provides an improved ISPF gateway service for callers to start a TSO address space. The new service provides all of the same capabilities as the existing “Legacy ISPF Gateway” and many more. While the Legacy ISPF Gateway is still provided, users should consider migrating to the Interactive ISPF Gateway to benefit from the new features. Future enhancements for the TSO/ISPF client gateway will be made to the Interactive ISPF Gateway.

Advantages of using the Interactive ISPF Gateway:

- The TSO/E address spaces are started by using a TSO logon procedure. Sites can choose to use their existing TSO logon procedures for TSO/E address spaces started through the gateway.
- There is support for callers to execute programs that are interactive, using a conversational pattern.
- Attention interrupts can be sent to terminate commands in progress.
- The Interactive ISPF Gateway uses z/OS Common Event Adapter (CEA) TSO/E address space services to start and manage TSO/E address spaces. The z/OS CEA TSO/E address space services provide support for the reuse of TSO/E address spaces, improving performance when a single TSO/E address space is used to issue multiple TSO or ISPF commands.
- The gateway can capture, and return to the client, messages that are issued using the TPUT and WTO macros and messages issued by TSO/E REXX.
- TSO/E address spaces that are started using CEA TSO/E address space services have the same characteristics as TSO/E address spaces for users who have logged in through the TSO/E logon screen. Therefore, existing processes used to manage TSO/E address spaces also apply to address spaces that are started using CEA TSO/E address space services. For example the operator command:

  ```
  C U=user-id,A=asid
  ```

  can be used to cancel an address space that is started using CEA TSO/E address space services.

Requirements for using the Interactive ISPF Gateway:

- Callers must define the CGI_CEATSO environment variable and set it to the value TRUE.
- Callers must use Amode 64 because this is required by the underlying CEA TSO/E address space services.
- A system administrator must configure and start the z/OS CEA address space.

For more information on CEA TSO/E address space services, refer to **CEA TSO/E address space services** in **z/OS MVS Programming: Callable Services for High-Level Languages**. For information on preparing to use the Interactive ISPF Gateway, see “Preparing to use the Interactive ISPF Gateway” on page 46.

Table 3 on page 44 describes the load modules that comprise the TSO/ISPF client gateway. The footnotes indicate the required load modules for using the Interactive ISPF Gateway and the required load modules for using the Legacy ISPF Gateway.

Table 3. Load modules comprising the gateway

Name	**Library**	**Description**
ISPZINT	ISP.SISPLPA	Gateway initialization - routes processing to ISPZINO or ISPZINL. (1,2)
ISPZINO	ISP.SISPLPA	Legacy ISPF Gateway - manages TSO sessions using spawned processes, routes command requests to these sessions. (2)
ISPZINL	SYS1.SIEALNKE	Interactive ISPF Gateway – manages TSO sessions by using CEA TSO/E address space services and routing command requests to these sessions. (1)

Table 3. Load modules comprising the gateway (continued)

Name	Library	Description
ISPZTSO	SYS1.LINKLIB	TSO initialization - attaches a TSO session to run a command. ISPZTSO must run APF-authorized. (2)
ISPZCNT	ISP.SISPLOAD	ISPF initialization - allocates data sets and starts an ISPF session. (2)
ISPZCMD	ISP.SISPLOAD	ISPF command interface - invokes an ISPF command. (1,2)
ISPZTMO	ISP.SISPLOAD	Reusable session time-out processor. (2)

1 – Load module is used by the Interactive ISPF Gateway.
2 – Load module is used by the Legacy ISPF Gateway.

Table 4 on page 45 describes the TSO/ISPF client gateway files installed into the z/OS UNIX file system. The footnotes indicate the required files for using the Interactive ISPF Gateway and the required files for using the Legacy ISPF Gateway.

Table 4. Gateway files installed into the z/OS UNIX file system

Name	Description
/usr/lpp/ispf/bin/ISPZINL	A stub file with the sticky bit set on to enable invocation of the Interactive ISPF Gateway load module ISPZINL. (1)
/usr/lpp/ispf/bin/ISPZINO	A stub file with the sticky bit set on to enable invocation of the Legacy ISPF Gateway load module ISPZINO. (2)
/usr/lpp/ispf/bin/ISPZINT	A stub file with the sticky bit set on to enable invocation of the gateway initialization load module ISPZINT. (1,2)
/usr/lpp/ispf/bin/ISPZTSO	A stub file with the sticky bit set on to enable invocation of the TSO initialization load module ISPZTSO. (2)
/usr/lpp/ispf/bin/ISPZXENV	A REXX routine called by the gateway XML API routine, ISPZXML, to set up environment variables. (1,2)
/usr/lpp/ispf/bin/ISPZXML	A REXX routine that is the XML API for the gateway. (1,2)

1 – File is used by the Interactive ISPF Gateway.
2 – File is used by the Legacy ISPF Gateway.

If you are using the Interactive ISPF Gateway, see "Interactive ISPF Gateway" on page 45 for additional information.

If you are using the Legacy ISPF Gateway, see "Legacy ISPF Gateway" on page 68 for additional information.

Interactive ISPF Gateway

This section provides information about preparing for and using the Interactive ISPF Gateway. For information about the Interactive ISPF Gateway and the Legacy ISPF Gateway and the benefits of using the Interactive ISPF Gateway, see "TSO/ISPF client gateway" on page 43.

To use the Interactive ISPF Gateway, define the environment variable CGI_CEATSO and set it to the value TRUE before invoking the gateway.

The Interactive ISPF Gateway is invoked and runs under the caller's address space. It then uses the z/OS CEA TSO/E address space services to start and communicate with the TSO address space on behalf of the caller. The types of requests that can be issued by the caller are:

- Start a TSO/E address space. Available options allow the client to specify whether ISPF is started in the address space and whether a dormant TSO session is used if available.
- Issue a TSO command (or ISPF command, if ISPF is started) in the TSO/E address space.
- Communicate interactively with the command running in the TSO/E address space.
- Ping a started TSO/E address space. If 15 minutes pass without receipt of a ping for the address space, CEA TSO/E address space services ends the address space.
- Send an attention interrupt to the command running in the TSO/E address space.
- End a TSO/E address space. Available options allow the client to specify whether the address space is cancelled, logged off, or allowed to become dormant.

Some types of gateway requests (for example, start a TSO/E address space, issue a command in the address space, and end the address space) can be combined in a single request. Upon completion of a request, the gateway returns the results to the client. Examples of items returned are the return code value, the output from the command run in the address space, and the information needed to reuse the address space for a subsequent command.

After a TSO/E address space is created, it can be used for a single TSO/ISPF command and then ended or it can be kept active and used for subsequent commands. The advantage of keeping the address space active is that the overhead of establishing a new address space for every TSO/ISPF command request is avoided.

The gateway does not provide the network communications and data transport between the client and z/OS host. The gateway is a thin service designed to interface with a local z/OS application that is serving remote users that have been authenticated to z/OS. The local application is responsible for any network communications when serving the remote client connections.

Note: ISPF commands invoked by means of the gateway run in an ISPF batch environment.

Preparing to use the Interactive ISPF Gateway

Consider these points as you prepare your system to use the Interactive ISPF Gateway:

- An OMVS segment must be defined in the security system (RACF® or equivalent) for each user of the gateway. It must specify a valid non-zero uid, home directory, and shell command.
- The prerequisites described in **System prerequisites for the CEA TSO/E address space services** in **z/OS MVS Programming: Callable Services for High-Level Languages** must be completed.
- Set MAXPROCUSER in BPXPRMxx parmlib member to a minimum of 50. This can be checked and set dynamically (until the next IPL) with the following commands (as described in z/OS MVS System Commands, SA22-7627):

  ```
  DISPLAY OMVS,O
  SETOMVS MAXPROCUSER=50.
  ```

  Setting a value that is too low can cause the gateway to fail.
- Table 3 on page 44 describes the load modules that comprise the gateway. The load modules that are used by the Interactive ISPF Gateway are indicated by the footnotes. These load modules must be available to run under the server used to communicate between the client and z/OS® host.

  A PROGRAM class profile that identifies ISPZINT as being a controlled program should be defined using the command:

  ```
  RDEFINE PROGRAM ISPZINT  ADDMEM('ISP.SISPLPA'//NOPADCHK) UACC(READ)
  ```

For information on using this command, refer to your security server's documentation. Failure to define ISPZINT as being controlled might result in messages, such as CSV042I or ICH422I, being issued.

- Table 4 on page 45 describes the gateway files installed into the z/OS UNIX file system. The files that are used by the Interactive ISPF Gateway are indicated by the footnotes.
- The CEA TSO/E address space services programs are in directory /usr/lib. This directory must be included in the LIBPATH environment variable available to the gateway.
- If a user has a TN3270 ISPF session active while using the Interactive ISPF Gateway, the TN3270 session must use ISPF profile sharing when it uses the same ISPF profile data set as is used by the logon procedure specified for the gateway. Profile sharing is enabled by specifying the SHRPROF parameter on the ISPF, PDF, or ISPSTART command used to start ISPF. Sites can also use the ISPF configuration table to define SHRPROF as a default parameter for these commands. An alternative to using ISPF profile sharing is to use different ISPF profile data sets for TN3270 ISPF sessions and the gateway. This can be done by using different logon procedures for TN3270 and the gateway.
- For additional steps that must be completed depending on the environment from which you call the gateway, see "Customizing the gateway calling environment" on page 82.

Using the Interactive ISPF Gateway

This section describes how to provide requests to the Interactive ISPF Gateway and the format of the output returned by the gateway.

Using the Interactive ISPF Gateway XML API

Requests can be passed to the Interactive ISPF Gateway in XML format. The client passes the request XML, by means of STDIN, to the XML API module ISPZXML. After processing the input XML, ISPZXML calls module ISPZXENV to define the necessary environment variables. After defining the environment variables, ISPZXENV calls module ISPZINT to initialize the gateway. If environment variable CGI_CEATSO is defined and is set to the value TRUE, ISPZINT calls module ISPZINL to invoke the Interactive ISPF Gateway to process the request. The output from the request is passed by means of STDOUT back to the client.

The XML schema shown here, which is supplied in member ISPZXCEI in the ISPF samples data set ISP.SISPSAMP, describes the request XML format and can be used to validate the XML for a request:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="ISPF-INPUT">
 <xs:complexType>
  <xs:all>
   <xs:element name="SERVICE-REQUEST">
    <xs:complexType>
     <xs:all>

   <!-- REQUEST: Request type. Required on all gateway calls       -->
   <xs:element name="request" minOccurs="1">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <!-- NEWTSO:     Start a new TSO session without ISPF active.-->
      <xs:enumeration value="NEWTSO"/>
      <!-- NEWTSOISPF: Start a new TSO session with ISPF active.   -->
      <xs:enumeration value="NEWTSOISPF"/>
      <!-- RECONNTSO:  If a dormant TSO session is available,      -->
      <!--             reconnect to it without ISPF active.        -->
      <!--             Otherwise, this is equivalent to NEWTSO.    -->
      <xs:enumeration value="RECONNTSO"/>
      <!-- RECONNTSOISPF: If a dormant TSO session is available,   -->
      <!--                reconnect to it with ISPF active. Other- -->
      <!--                wise, this is equivalent to NEWTSOISPF.  -->
      <xs:enumeration value="RECONNTSOISPF"/>
      <!-- REUSE:      Send command to a started session.          -->
      <xs:enumeration value="REUSE"/>
      <!-- RESPOND:    Respond to a TSO prompt.                    -->
      <xs:enumeration value="RESPOND"/>
      <!-- PING:       Ping a started session.                     -->
      <xs:enumeration value="PING"/>
```

```
       <!-- ATTN:       ATTN a started session.                     →
       <xs:enumeration value="ATTN"/>
       <!-- DORMANT:    If dormant sessions are enabled and max      -->
       <!--             dormant sessions is not reached, put the     -->
       <!--             TSO session in a dormant state. Otherwise,   -->
       <!--             this is equivalent to LOGOFF.                -->
       <xs:enumeration value="DORMANT"/>
       <!-- LOGOFF:     LOGOFF the started TSO session.              -->
       <xs:enumeration value="LOGOFF"/>
       <!-- CANCEL:     CANCEL the started TSO session.              -->
       <xs:enumeration value="CANCEL"/>
      </xs:restriction>
     </xs:simpleType>
    </xs:element>

     <!-- PROCNAME: Procedure name. Required with request of         -->
     <!--           NEWTSO | NEWTSOISPF | RECONNTSO | RECONNTSOISPF -->
    <xs:element name="procname" type="xs:string" minOccurs="0"/>

     <!-- ACCTNUM:  Account number. Required with request of         -->
     <!--           NEWTSO | NEWTSOISPF | RECONNTSO | RECONNTSOISPF -->
    <xs:element name="acctnum" type="xs:string" minOccurs="0"/>

     <!-- GROUPID:  Group identifier. Required with request of       -->
     <!--           NEWTSO | NEWTSOISPF | RECONNTSO | RECONNTSOISPF -->
    <xs:element name="groupid" type="xs:string" minOccurs="0"/>

     <!-- REGIONSZ: Region size. Required with request of            -->
     <!--           NEWTSO | NEWTSOISPF | RECONNTSO | RECONNTSOISPF -->
    <xs:element name="regionsz" type="xs:integer" minOccurs="0"/>

     <!-- SESSID:   Session identifier. Required with request of     -->
     <!--           REUSE | RESPOND | PING | ATTN | DORMANT |        -->
     <!--           LOGOFF | CANCEL (unless request of NEWTSO |      -->
     <!--           NEWTSOISPF | RECONNTSO | RECONNTSOISPF is        -->
     <!--           specified on same call).                         -->
    <xs:element name="sessid" type="xs:string" minOccurs="0"/>

     <!-- CMDRESP:  TSO/ISPF command or response                     -->
     <!--           Response format: RESPONSE "response text"        -->
    <xs:element name="cmdresp" type="xs:string" minOccurs="0"/>

     <!-- LOGLEVEL: Logging level. Add values for multiple types     -->
     <!--            1 - Log error information                       -->
     <!--            2 - Log debug information                       -->
     <!--            4 - Log communication information               -->
     <!--            8 - Log time information                        -->
     <!--           16 - Log information to the system console       -->
     <!--           32 - Include time stamps in log information      -->
    <xs:element name="loglevel" type="xs:integer" minOccurs="0"/>

      </xs:all>
     </xs:complexType>
    </xs:element>

   </xs:all>
  </xs:complexType>
 </xs:element>

 </xs:schema>
```

The output from a service request is returned by the gateway to the client in XML format. The XML schema shown here, which is supplied in member ISPZXCEO in the ISPF samples data set ISP.SISPSAMP, describes the format and can be used to process the XML returned for a service request:

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="ISPF-OUTPUT">
 <xs:complexType>
  <xs:all>
   <xs:element name="SERVICE-REQUEST"/>
   <xs:element name="SERVICE-RESPONSE">
    <xs:complexType>
     <xs:all>
      <xs:element name="ISPF-COMMAND"/>
      <xs:element name="RETURN-CODE"/>
      <xs:element name="ISPF"/>
      <xs:element name="TSO"/>
```

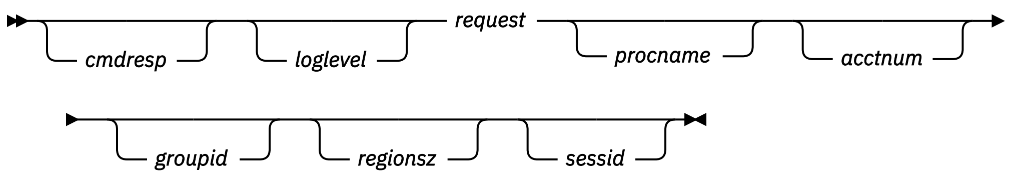
```
        <xs:element name="SESSION-INFO"/>
        <xs:element name="SESS"/>
        <xs:element name="TSOPROMPT"/>
       </xs:all>
      </xs:complexType>
     </xs:element>

     <xs:element name="OPERATIONS-LOG"/>

    </xs:all>
   </xs:complexType>
  </xs:element>

  </xs:schema>
```

Using the Interactive ISPF Gateway native API

Requests can be passed to the Interactive ISPF Gateway in native format. Native API requests consist of a parameter string passed by means of STDIN to the gateway module ISPZXENV. ISPZXENV defines the necessary environment variables and then calls module ISPZINT to initialize the gateway. If environment variable CGI_CEATSO is defined and is set to the value TRUE, ISPZINT calls module ISPZINL to invoke the Interactive ISPF Gateway to process the request. The output from a request is passed by means of STDOUT back to the client.

Note: If the client defines environment variables using a method other than by calling gateway module ISPZXENV, the native API request can be passed by means of STDIN directly to gateway module ISPZINT.

Native API requests are of the format:



The only parameter required on all requests is the request parameter. Other parameters are required depending on the request parameter value.

The cmdresp parameter is the only parameter that can contain embedded blanks. All other parameters must be of the format &KEYWORD=value, with no embedded blanks. Blanks are allowed between parameters, but are not required.

The parameters on a request can be specified in any order with the following exceptions:

- The cmdresp parameter, when used, must be specified as the first parameter.
- The loglevel parameter, when used, should be specified immediately following the cmdresp parameter, or specified first when the cmdresp parameter is not specified, to provide the most comprehensive logging.

cmdresp

For this parameter, specify one of the following:

- A TSO or ISPF command to be issued in a TSO/E address space. The command format is free form.
- A response to a TSO prompt received from a TSO/E address space. A response must be in the format: RESPONSE "*response text*".

When used, this parameter must be specified as the first parameter in the request.

loglevel

The level of logging performed by the gateway during processing of the request. Logging information is included in the information returned to the client by means of STDOUT.

Specify this parameter using the format: &LOGLEVEL=*log_level*

The logging levels are:

0
: No logging. This is the default level.

1
: Log error information. Provides additional information when errors occur during request processing.

2
: Log debug information. Provides information about the request processing that can be useful when debugging problems.

4
: Log communication information. Provides information about the messages that are sent by the gateway to TSO or ISPF and received by the gateway from TSO or ISPF.

8
: Log time information. Provides information about the elapsed time during various phases of gateway processing.

16
: Log information to the system console. Causes the logging information generated by the gateway to be written to the system console.

32
: Include time stamps in the log information. Causes a time stamp value to be included in some lines of log information.

Notes:

- The logging level values can be added together to request multiple types of logging information. For example, specify &LOGLEVEL=7 to request the logging of error, debug, and communication information.
- This parameter should be specified immediately following the cmdresp parameter, or first when the cmdresp parameter is not specified, in order to provide the most comprehensive logging information.

request

This parameter specifies an action to be taken by the gateway.

Specify this parameter using the format: &REQUEST=*action*

The possible values for *action* are:

NEWTSO
: Start a new TSO/E address space. Do not reconnect to a dormant address space. Do not start ISPF in the address space. This address space can only be used for issuing TSO commands. The procname, acctnum, groupid, and regionsz parameters must be specified with this request value.

NEWTSOISPF
: Start a new TSO/E address space. Do not reconnect to a dormant address space. Start ISPF in the address space. This address space can be used for issuing TSO and ISPF commands. The procname, acctnum, groupid, and regionsz parameters must be specified with this request value.

RECONNTSO
: Reconnect to a dormant TSO/E address space if one is available. Otherwise, start a new address space. Do not start ISPF in the address space. This address space can only be used for issuing TSO commands. The procname, acctnum, groupid, and regionsz parameters must be specified with this request value.

RECONNTSOISPF
: Reconnect to a dormant TSO/E address space if one is available. Otherwise, start a new address space. Start ISPF in the address space. This address space can be used for issuing TSO and ISPF commands. The procname, acctnum, groupid, and regionsz parameters must be specified with this request value.

Note: ISPF cannot be active in a dormant address space. Therefore, you cannot reconnect to a dormant address space in which ISPF is already active.

REUSE
Reuse a TSO/E address space for a new command. The address space to reuse is identified by the sessid parameter. The sessid parameter must be specified with this request value.

Note: REUSE does NOT reset the timeout timer which might cause the session to timeout prematurely. Setting CGI_PING=TRUE will result in a PING request to be sent along with REUSE so that the timer will be reset. As a result, the session will continue to exist.

RESPOND
Respond to a prompt from a TSO/E address space. The target address space for the response is identified by the sessid parameter. The sessid and cmdresp parameters must be specified with this request value.

PING
Ping the TSO/E address space that is started or reconnected by the same client request or that is identified by the sessid parameter. The sessid parameter must be specified with this request value unless the address space is started or reconnected by the same client request. If 15 minutes pass without receipt of a ping for the address space, CEA TSO/E address space services ends the address space.

ATTN
Send an attention interrupt to the TSO/E address space that is identified by the sessid parameter. The sessid parameter must be specified with this request value.

DORMANT
Put the TSO/E address space in a dormant state. This request is satisfied only if dormant sessions are enabled and the maximum number of dormant sessions is not reached. Otherwise, this is equivalent to LOGOFF. The TSO/E address space is the address space that was started or reconnected by the same client request or that is identified by the sessid parameter. The sessid parameter must be specified with this request value unless the address space is started or reconnected by the same client request.

LOGOFF
Log off the TSO/E address space that is started or reconnected by the same client request or that is identified by the sessid parameter. The sessid parameter must be specified with this request value unless the address space is started or reconnected by the same client request.

CANCEL
Cancel the TSO/E address space that is started or reconnected by the same client request or that is identified by the sessid parameter. The sessid parameter must be specified with this request value unless the address space is started or reconnected by the same client request.

procname
The name of the TSO/E logon procedure to be used for a TSO/E session. This parameter is required on a request for the start of a new TSO/E session (NEWTSO or NEWTSOISPF) or a reconnect to a dormant TSO/E session (RECONNTSO or RECONNTSOISPF). Callers can obtain this information by using z/OS Security Manager services, such as RACROUTE REQUEST=EXTRACT. Valid values are 1 to 8 characters long.

Specify this parameter using the format: &PROCNAME=*procedure_name*

acctnum
The TSO/E account number to be used for a TSO/E session. This parameter is required on a request for the start of a new TSO/E session (NEWTSO or NEWTSOISPF) or a reconnect to a dormant TSO/E session (RECONNTSO or RECONNTSOISPF). Callers can obtain this information by using z/OS Security Manager services, such as RACROUTE REQUEST=EXTRACT. Valid values are 1 to 40 characters long.

Specify this parameter using the format: &ACCTNUM=*account_number*

groupid
The TSO/E group name to be used for a TSO/E session. This parameter is required on a request for the start of a new TSO/E session (NEWTSO or NEWTSOISPF) or a reconnect to a dormant TSO/E session (RECONNTSO or RECONNTSOISPF). Callers can obtain this information by using z/OS Security Manager services, such as RACROUTE REQUEST=EXTRACT. Valid values are 1 to 8 characters long.

Specify this parameter using the format: &GROUPID=*group_id*

regionsz

The region size to be used for a TSO/E session. This parameter is required on a request for the start of a new TSO/E session (NEWTSO or NEWTSOISPF) or a reconnect to a dormant TSO/E session (RECONNTSO or RECONNTSOISPF). Callers can obtain this information by using z/OS Security Manager services, such as RACROUTE REQUEST=EXTRACT. Valid values are 0 to 2,096,128.

Specify this parameter using the format: &REGIONSZ=*region_size*

sessid

The information that uniquely identifies a started TSO/E session. This information is returned to the client by the gateway when a TSO/E session is started or reconnected and consists of the following:

- Address space information version
- Address space ID
- Address space token
- Address space index
- Message queue ID
- Address space type

Specify this parameter using the format:

&VER=*ver*&ASID=*asid*&STOKEN=*stoken*&INDEX=*index*&MSGQID=*msgqid*&TYPE=*type*

Note: This is the format used by the gateway to return the session identifying information to the client when the session is started or reconnected. The information is returned by the gateway between the <SESS> and </SESS> tags. The client can provide the information on subsequent requests exactly as it was returned by the gateway.

With the exception described in the following note, the sessid parameter is required with any request parameter value where the gateway must identify the TSO/E session to be used. This includes all of the following types of requests:

&REQUEST=REUSE
Reuse the session for a new command.

&REQUEST=RESPOND
Respond to a TSO prompt from the session.

&REQUEST=PING
Ping the session.

&REQUEST=ATTN
Send an attention interrupt to the session.

&REQUEST=DORMANT
Put the session in a dormant state.

&REQUEST=LOGOFF
Log off of the session.

&REQUEST=CANCEL
Cancel the session.

Note: If a single input from the client requests both the start or reconnect of a TSO/E session and one of the request types above, the session identifying information is not required on the request. For information on combining request types, see "Combining requests using the Interactive ISPF Gateway" on page 62.

Starting a TSO/E address space using the Interactive ISPF Gateway

To start a TSO/E address space, send a request to the gateway with the request parameter set to one of these values:

NEWTSO
Start a new TSO/E address space. Do not reconnect to a dormant address space. Do not start ISPF in the address space. This address space can only be used for issuing TSO commands.

NEWTSOISPF
Start a new TSO/E address space. Do not reconnect to a dormant address space. Start ISPF in the address space. This address space can be used for issuing TSO and ISPF commands.

RECONNTSO
Reconnect to a dormant TSO/E address space if one is available. Otherwise, start a new address space. Do not start ISPF in the address space. This address space can only be used for issuing TSO commands.

RECONNTSOISPF
Reconnect to a dormant TSO/E address space if one is available. Otherwise, start a new address space. Start ISPF in the address space. This address space can be used for issuing TSO and ISPF commands.

Note: Dormant TSO/E address spaces and the ability to reconnect to them are functions provided by CEA TSO/E address space services. To enable this feature, which is disabled by default, specify non-zero values for the RECONSESSIONS and RECONTIME statements in the TSOASMGR parmlib statement in the CEAPRMxx parmlib member. For additional information about dormant sessions and enabling their use, refer to **Reconnecting to CEA TSO/E address spaces** in **z/OS MVS Programming: Callable Services for High-Level Languages**.

Four items must be provided with a request to start (or reconnect to) a TSO/E address space. These are the same items that are entered when a user logs on to a TSO/E session:

- Procedure name – specified with the procname parameter.
- Account number – specified with the acctnum parameter.
- Group name – specified with the groupid parameter.
- Region size – specified with the regionsz parameter.

Callers can obtain this information by using z/OS Security Manager services, such as RACROUTE REQUEST=EXTRACT.

Using the XML API to start a TSO/E address space

The following example shows the input and resulting output using the XML API to start a TSO/E address space. In this example, a new TSO/E address space is requested. ISPF is not started in the address space. Therefore, the address space can only be used for issuing TSO commands:

Input:

```
<?xml version="1.0"?>
<ISPF-INPUT>
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"ispf.xsd\">
<SERVICE-REQUEST>
<request>NEWTSO</request>
<procname>ISPFPROC</procname>
<acctnum>SAMPACCT</acctnum>
<groupid>DEFAULT</groupid>
<regionsz>2000000</regionsz>
</SERVICE-REQUEST>
</ISPF-INPUT>
```

Output:

```
<?xml version="1.0"?>
<ISPF-OUTPUT>
 <SERVICE-REQUEST>
<request>NEWTSO</request>
<procname>ISPFPROC</procname>
<acctnum>SAMPACCT</acctnum>
```

```
<groupid>DEFAULT</groupid>
<regionsz>2000000</regionsz>
 </SERVICE-REQUEST>
 <SERVICE-RESPONSE>
  <RETURN-CODE>0</RETURN-CODE>
  <SESSION-INFO>
  <SESS>&VER=1&ASID=88&STOKEN=000001600000006C&INDEX=1&MSGQID=3342342&TYPE=TSO</SESS>
  <TSOPROMPT>NO</TSOPROMPT>
  </SESSION-INFO>
 </SERVICE-RESPONSE>
 <OPERATIONS-LOG>
 <![CDATA[
  Content-type: text/plain

  ISPZINL started - z/OS 3.1 01JAN23 Base
  Data read from STDIN is
  &REQUEST=NEWTSO&PROCNAME=ISPFPROC&ACCTNUM=SAMPACCT&GROUPID=DEFAULT&REGIONSZ=2000000
  *** XML-NOTE *** Reference tagged SERVICE-RESPONSE
 ]]>
 </OPERATIONS-LOG>
</ISPF-OUTPUT>
```

Note: The value returned between the <SESS> and </SESS> tags is the sessid value that must be returned on subsequent requests to use the session.

Using the native API to start a TSO/E address space

The following example shows the input and resulting output using the native API to start a TSO/E address space. In this example, a new TSO/E address space is requested. ISPF is started in the address space. Therefore, it can be used for issuing TSO and ISPF commands:

Input:

```
&REQUEST=NEWTSOISPF &PROCNAME=ISPFPROC &ACCTNUM=SAMPACCT &GROUPID=DEFAULT
&REGIONSZ=2000000
```

Output:

```
Content-type: text/plain

ISPZINL started - z/OS 3.1 01JAN23 Base
Data read from STDIN is &REQUEST=NEWTSOISPF &PROCNAME=ISPFPROC &ACCTNUM=SAMPACCT
&GROUPID=DEFAULT &REGIONSZ=2000000
<ISPINFO>
RC=0
</ISPINFO>
<SESSION-INFO>
<SESS>&VER=1&ASID=106&STOKEN=000001A80000004E&INDEX=2&MSGQID=9764869&TYPE=ISPF</SESS>
<TSOPROMPT>NO</TSOPROMPT>
</SESSION-INFO>
```

Note: The value returned between the <SESS> and </SESS> tags is the sessid value that must be returned on subsequent requests to use the session.

Issuing a TSO/ISPF command using the Interactive ISPF Gateway

To issue a TSO or ISPF command in a TSO/E address space, send a request to the gateway with the cmdresp parameter set to the command string.

You must also include the request parameter to indicate whether you want the command to be issued in a new TSO/E address space, in a reconnected TSO/E address space, or in an active TSO/E address space. The request parameter must be set to one of the following values:

NEWTSO
: The command is issued in a new TSO/E address space in which ISPF is not started. Only use this value if your command does not require that ISPF be started in the address space. For this value, the procname, acctnum, groupid, and regionsz parameters must also be specified.

NEWTSOISPF

The command is issued in a new TSO/E address space in which ISPF is started. Use this value if your command requires that ISPF be started in the address space. For this value, the procname, acctnum, groupid, and regionsz parameters must also be specified.

RECONNTSO

If a dormant TSO/E address space is available, the address space is reconnected and the command is issued in that address space. Otherwise, the command is issued in a new TSO/E address space. ISPF is not started in the address space. Only use this value if your command does not require that ISPF be started in the address space. For this value, the procname, acctnum, groupid, and regionsz parameters must also be specified.

RECONNTSOISPF

If a dormant TSO/E address space is available, the address space is reconnected and the command is issued in that address space. Otherwise, the command is issued in a new TSO/E address space. ISPF is started in the address space. Use this value if your command requires that ISPF be started in the address space. For this value, the procname, acctnum, groupid, and regionsz parameters must also be specified.

REUSE

The command is issued in an active TSO/E address space. You must be aware of whether ISPF is started in the address space, based upon the request that was previously issued to start or reconnect to the address space. For this value, the sessid parameter must also be specified.

Note: REUSE does NOT reset the timeout timer which might cause the session to timeout prematurely. Setting CGI_PING=TRUE will result in a PING request to be sent along with REUSE so that the timer will be reset. As a result, the session will continue to exist.

Using the XML API to issue a TSO/ISPF command in a TSO/E address space

The following example shows the input and resulting output using the XML API to issue a TSO/ISPF command in a TSO/E address space. In this example, the TIME command is issued in a new TSO/E address space in which ISPF is not started.

Input:

```
<?xml version="1.0"?>
<ISPF-INPUT>
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"ispf.xsd\">
<SERVICE-REQUEST>
<request>NEWTSO</request>
<procname>ISPFPROC</procname>
<acctnum>SAMPACCT</acctnum>
<groupid>DEFAULT</groupid>
<regionsz>2000000</regionsz>
<cmdresp>TIME</cmdresp>
</SERVICE-REQUEST>
</ISPF-INPUT>
```

Output:

```
<?xml version="1.0"?>
<ISPF-OUTPUT>
 <SERVICE-REQUEST>
<request>NEWTSO</request>
<procname>ISPFPROC</procname>
<acctnum>SAMPACCT</acctnum>
<groupid>DEFAULT</groupid>
<regionsz>2000000</regionsz>
<cmdresp>TIME</cmdresp>
 </SERVICE-REQUEST>
 <SERVICE-RESPONSE>
  <TSO>
  <![CDATA[
   IKJ56650I TIME-10:40:59 PM. CPU-00:00:00 SERVICE-311 SESSION-00:00:00 JUNE 27,2014
  ]]>
  </TSO>
  <RETURN-CODE>0</RETURN-CODE>
  <SESSION-INFO>
  <SESS>&VER=1&ASID=88&STOKEN=0000016000000071&INDEX=2&MSGQID=10092549&TYPE=TSO</SESS>
  <TSOPROMPT>NO</TSOPROMPT>
  </SESSION-INFO>
 </SERVICE-RESPONSE>
 <OPERATIONS-LOG>
 <![CDATA[
  Content-type: text/plain

  ISPZINL started - z/OS 3.1 01JAN23 Base
  Data read from STDIN is
  TIME&REQUEST=NEWTSO&PROCNAME=ISPFPROC&ACCTNUM=SAMPACCT&GROUPID=DEFAULT&REGIONSZ=2000000
  *** OUTPUT FROM TSO SESSION ***
  *** XML-NOTE *** Reference tagged SERVICE-RESPONSE
 ]]>
 </OPERATIONS-LOG>
</ISPF-OUTPUT>
```

Using the native API to issue a TSO/ISPF command in a TSO/E address space

The following example shows the input and resulting output using the native API to issue a TSO/ISPF command in a TSO/E address space. In this example, the TSO TIME command is issued in an already active TSO/E address space in which ISPF is started:

Input:

```
TSO TIME &REQUEST=REUSE&VER=1&ASID=106&STOKEN=000001A80000004E&INDEX=2&MSGQID=9764869&TYPE=ISPF
```

Output:

```
Content-type: text/plain

ISPZINL started - z/OS 3.1 01JAN23 Base
Data read from STDIN is
TSO TIME &REQUEST=REUSE&VER=1&ASID=106&STOKEN=000001A80000004E&INDEX=2&MSGQID=9764869&TYPE=ISPF
*** OUTPUT FROM ISPF SESSION ***
<ISPINFO>
Command passed to SELECT -
CMD(TIME)
<ISPF>
IKJ56650I TIME-10:50:49 PM. CPU-00:00:00 SERVICE-3661 SESSION-00:01:01 JUNE 27,2014
</ISPF>
RC=0
</ISPINFO>
<SESSION-INFO>
<TSOPROMPT>NO</TSOPROMPT>
</SESSION-INFO>
```

Interactive communication using the Interactive ISPF Gateway

When the client requests that a command be issued in a TSO/E address space and that command prompts the client for input, the gateway processing of the request that issued the command ends. Any data that is written by the command before it issued the prompt is returned to the client in the gateway output, along with an indication that the command is waiting for input. The indication that the command is waiting for input is a TSOPROMPT line with the value YES:

```
<TSOPROMPT>YES</TSOPROMPT>
```

When included in the output, the TSOPROMPT line can be found in the information that is surrounded by the <SESSION-INFO> and </SESSION-INFO> tags. If the TSOPROMPT line is not included in the output or the value on the line is NO, the command has ended and is not waiting for input.

To respond to a prompt from the command running in a TSO/E address space, send a request to the gateway with the request parameter set to RESPOND. The target TSO/E address space for the response must be identified using the sessid parameter. Use the cmdresp parameter to provide the response text. The cmdresp parameter must be specified using the format:

RESPONSE "*response text*"

Using the XML API for interactive communication

The following example shows the input and resulting output using the XML API for interactive communication.

In this example, the client issues the command EX 'TEST.EXEC(GWSVMULT)' to run a REXX exec. The exec writes the line “Hello client, what is your name?” and then prompts for input. This is the output returned to the client when the exec prompts for input:

```
<?xml version="1.0"?>
<ISPF-OUTPUT>
 <SERVICE-REQUEST>
<request>REUSE</request>
<sessid>&VER=1&ASID=88&STOKEN=000016000000076&INDEX=1&MSGQID=10944520&TYPE=TSO</sessid>
<cmdresp>EX 'TEST.EXEC(GWSVMULT)'</cmdresp>
 </SERVICE-REQUEST>
 <SERVICE-RESPONSE>
  <TSO>
  <![CDATA[
   Hello client, what is your name?
  ]]>
  </TSO>
  <RETURN-CODE>0</RETURN-CODE>
  <SESSION-INFO>
  <TSOPROMPT>YES</TSOPROMPT>
  </SESSION-INFO>
 </SERVICE-RESPONSE>
 <OPERATIONS-LOG>
 <![CDATA[
  Content-type: text/plain

  ISPZINL started - z/OS 3.1 01JAN23 Base
  Data read from STDIN is
  EX 'TEST.EXEC(GWSVMULT)'&REQUEST=REUSE
  &VER=1&ASID=88&STOKEN=000016000000076&INDEX=1&MSGQID=10944520&TYPE=TSO
  *** OUTPUT FROM TSO SESSION ***
  *** XML-NOTE *** Reference tagged SERVICE-RESPONSE
 ]]>
 </OPERATIONS-LOG>
</ISPF-OUTPUT>
```

The following XML input from the client provides the response:

```
<?xml version="1.0"?>
<ISPF-INPUT>
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"ispf.xsd\">
<SERVICE-REQUEST>
<request>RESPOND</request>
<cmdresp>RESPONSE "JOHN"</cmdresp>
<sessid>&VER=1&ASID=88&STOKEN=000016000000076&INDEX=1&MSGQID=10944520&TYPE=TSO</
sessid>
</SERVICE-REQUEST>
</ISPF-INPUT>
```

Output:

```
<?xml version="1.0"?>
<ISPF-OUTPUT>
 <SERVICE-REQUEST>
<request>RESPOND</request>
<cmdresp>RESPONSE "JOHN"</cmdresp>
<sessid>&VER=1&ASID=88&STOKEN=000016000000076&INDEX=1&MSGQID=10944520&TYPE=TSO</sessid>
 </SERVICE-REQUEST>
 <SERVICE-RESPONSE>
  <TSO>
  <![CDATA[
   Hello JOHN
  ]]>
  </TSO>
```

```
   <RETURN-CODE>0</RETURN-CODE>
   <SESSION-INFO>
   <TSOPROMPT>NO</TSOPROMPT>
   </SESSION-INFO>
  </SERVICE-RESPONSE>
  <OPERATIONS-LOG>
  <![CDATA[
   Content-type: text/plain

   ISPZINL started - z/OS 3.1 01JAN23 Base
   Data read from STDIN is
   RESPONSE "JOHN"&REQUEST=RESPOND
   &VER=1&ASID=88&STOKEN=0000016000000076&INDEX=1&MSGQID=10944520&TYPE=TSO
   *** XML-NOTE *** Reference tagged SERVICE-RESPONSE
  ]]>
  </OPERATIONS-LOG>
</ISPF-OUTPUT>
```

As indicated in the output by the TSOPROMPT line with the value NO, the command did not prompt the client a second time. However, if the client is prompted a second time, the TSOPROMPT line would have the value YES and the example above could be repeated to provide the new response.

Using the native API for interactive communication

The following example shows the input and resulting output using the native API for interactive communication.

In this example, the client issues the command TSO PING to run the PING program. The program writes the line “Enter host name or address” and prompts for input. The following output is returned to the client when the program prompts for input:

```

Content-type: text/plain

ISPZINL started - z/OS 3.1 01JAN23 Base
Data read from STDIN is TSO PING &REQUEST=REUSE
&VER=1&ASID=104&STOKEN=000001A00000019B&INDEX=1&MSGQID=10616837&TYPE=ISPF
*** OUTPUT FROM ISPF SESSION ***
<ISPINFO>
Command passed to SELECT -
CMD(PING)
<ISPF>
Enter host name or address
</ISPF>
</ISPINFO>
<SESSION-INFO>
<TSOPROMPT>YES</TSOPROMPT>
</SESSION-INFO>
```

The following native input from the client provides the response:

```

RESPONSE "TEST-GWY.US.XYZ.COM" &REQUEST=RESPOND
&VER=1&ASID=104&STOKEN=000001A00000019B&INDEX=1&MSGQID=10616837&TYPE=ISPF
```

Output:

```

Content-type: text/plain

ISPZINL started - z/OS 3.1 01JAN23 Base
Data read from STDIN is RESPONSE "TEST-GWY.US.XYZ.COM" &REQUEST=RESPOND
&VER=1&ASID=104&STOKEN=000001A00000019B&INDEX=1&MSGQID=10616837&TYPE=ISPF
<ISPINFO>
<ISPF>
CS 3.1: Pinging host TEST-GWY.US.XYZ.COM (192.0.2.61)
Ping #1 response took 0.001 seconds.
</ISPF>
RC=0
</ISPINFO>
<SESSION-INFO>
<TSOPROMPT>NO</TSOPROMPT>
```

```
</SESSION-INFO>
```

As indicated in the output by the TSOPROMPT line with the value NO, the program did not prompt the client a second time. If the client is prompted a second time, the TSOPROMPT line would have the value YES and the example above could be repeated to provide the new response.

Pinging a TSO/E address space using the Interactive ISPF Gateway

To ping a TSO/E address space, send a request to the gateway with the request parameter set to PING. The session to be pinged must be identified using the sessid parameter.

Note: If 15 minutes pass without receipt of a ping for the address space, CEA TSO/E address space services ends the address space.

Using the XML API to ping a TSO/E address space

The following example shows the input and resulting output using the XML API to ping a TSO/E address space:

Input:

```
<?xml version="1.0"?>
<ISPF-INPUT>
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"ispf.xsd\">
<SERVICE-REQUEST>
<request>PING</request>
<sessid>&VER=1&ASID=88&STOKEN=000001600000006C&INDEX=1&MSGQID=3342342&TYPE=TSO</
sessid>
</SERVICE-REQUEST>
</ISPF-INPUT>
```

Output:

```
<?xml version="1.0"?>
<ISPF-OUTPUT>
 <SERVICE-REQUEST>
<request>PING</request>
<sessid>&VER=1&ASID=88&STOKEN=000001600000006C&INDEX=1&MSGQID=3342342&TYPE=TSO</sessid>
 </SERVICE-REQUEST>
 <SERVICE-RESPONSE>
  <RETURN-CODE>0</RETURN-CODE>
  <SESSION-INFO>
  <TSOPROMPT>NO</TSOPROMPT>
  </SESSION-INFO>
 </SERVICE-RESPONSE>
 <OPERATIONS-LOG>
 <![CDATA[
  Content-type: text/plain

  ISPZINL started - z/OS 3.1 01JAN23 Base
  Data read from STDIN is
  &REQUEST=PING&VER=1&ASID=88&STOKEN=000001600000006C&INDEX=1&MSGQID=3342342&TYPE=TSO
  *** XML-NOTE *** Reference tagged SERVICE-RESPONSE
 ]]>
 </OPERATIONS-LOG>
</ISPF-OUTPUT>
```

Using the native API to ping a TSO/E address space

The following example shows the input and resulting output using the native API to ping a TSO/E address space:

Input:

```
&REQUEST=PING &VER=1&ASID=106&STOKEN=000001A80000004E&INDEX=2&MSGQID=9764869&TYPE=ISPF
```

Output:

```
Content-type: text/plain

ISPZINL started - z/OS 3.1 01JAN23 Base
Data read from STDIN is  &REQUEST=PING
&VER=1&ASID=106&STOKEN=000001A80000004E&INDEX=2&MSGQID=9764869&TYPE=ISPF
<ISPINFO>
RC=0
</ISPINFO>
<SESSION-INFO>
<TSOPROMPT>NO</TSOPROMPT>
</SESSION-INFO>
```

Sending an attention interrupt using the Interactive ISPF Gateway

To send an attention interrupt to a TSO/E address space, send a request to the gateway with the request parameter set to ATTN. The session must be identified using the sessid parameter.

Using the XML API to send an attention interrupt to a TSO/E address space

The following example shows the input and resulting output using the XML API to send an attention interrupt to a TSO/E address space:

Input:

```
<?xml version="1.0"?>
<ISPF-INPUT>
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"ispf.xsd\">
<SERVICE-REQUEST>
<request>ATTN</request>
<sessid>&VER=1&ASID=88&STOKEN=000001600000006C&INDEX=1&MSGQID=3342342&TYPE=TSO</
sessid>
</SERVICE-REQUEST>
</ISPF-INPUT>
```

Output:

```
<?xml version="1.0"?>
<ISPF-OUTPUT>
 <SERVICE-REQUEST>
<request>ATTN</request>
<sessid>&VER=1&ASID=88&STOKEN=000001600000006C&INDEX=1&MSGQID=3342342&TYPE=TSO</sessid>
 </SERVICE-REQUEST>
 <SERVICE-RESPONSE>
  <TSO>
  <![CDATA[
   EZZ3112I Host name or address not entered
  ]]>
  </TSO>
  <RETURN-CODE>0</RETURN-CODE>
  <SESSION-INFO>
  <TSOPROMPT>NO</TSOPROMPT>
  </SESSION-INFO>
 </SERVICE-RESPONSE>
 <OPERATIONS-LOG>
 <![CDATA[
  Content-type: text/plain

  ISPZINL started - z/OS 3.1 01JAN23 Base
  Data read from STDIN is
  &REQUEST=ATTN&VER=1&ASID=88&STOKEN=000001600000006C&INDEX=1&MSGQID=3342342&TYPE=TSO
  *** XML-NOTE *** Reference tagged SERVICE-RESPONSE
 ]]>
 </OPERATIONS-LOG>
</ISPF-OUTPUT>
```

Using the native API to send an attention interrupt to a TSO/E address space

The following example shows the input and resulting output using the native API to send an attention interrupt to a TSO/E address space:

Input:

```
&REQUEST=ATTN &VER=1&ASID=106&STOKEN=000001A80000004E&INDEX=2&MSGQID=9764869&TYPE=ISPF
```

Output:

```
Content-type: text/plain

ISPZINL started - z/OS 3.1 01JAN23 Base
Data read from STDIN is
&REQUEST=ATTN &VER=1&ASID=106&STOKEN=000001A80000004E&INDEX=2&MSGQID=9764869&TYPE=ISPF
<ISPINFO>
<ISPF>
EZZ3112I Host name or address not entered
</ISPF>
RC=100
</ISPINFO>
<SESSION-INFO>
<TSOPROMPT>NO</TSOPROMPT>
</SESSION-INFO>
```

Ending a TSO/E address space using the Interactive ISPF Gateway

To end a TSO/E address space, send a request to the gateway with the request parameter set to one of these values:

DORMANT
If dormant sessions are enabled and the maximum number of dormant sessions is not reached, put the TSO/E session in a dormant state. Otherwise, this is equivalent to LOGOFF. The TSO/E address space is the address space that is started or reconnected by this request or that is identified by the sessid parameter.

LOGOFF
Log off the TSO/E address space that is started or reconnected by this request or that is identified by the sessid parameter.

CANCEL
Cancel the TSO/E address space that is started or reconnected by this request or that is identified by the sessid parameter.

Note: Dormant TSO/E address spaces and the ability to reconnect to them are functions provided by CEA TSO/E address space services. To enable this feature, which is disabled by default, specify non-zero values for the RECONSESSIONS and RECONTIME statements in the TSOASMGR parmlib statement of the CEAPRMxx parmlib member. For additional information about dormant sessions and enabling their use, refer to **Reconnecting to CEA TSO/E address spaces** in **z/OS MVS Programming: Callable Services for High-Level Languages**.

Using the XML API to end a TSO/E address space

The following example shows the input and resulting output using the XML API to end a TSO/E address space. In this example, the TSO/E address space, which is started by a previous gateway request, is put in a dormant state if dormant sessions are enabled and the maximum number of dormant sessions is not reached. Otherwise, the session is logged off:

Input:

```
<?xml version="1.0"?>
<ISPF-INPUT>
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
```

```
xsi:noNamespaceSchemaLocation=\"ispf.xsd\">
<SERVICE-REQUEST>
<request>DORMANT</request>
<sessid>&VER=1&ASID=88&STOKEN=000001600000006C&INDEX=1&MSGQID=3342342&TYPE=TSO</
sessid>
</SERVICE-REQUEST>
</ISPF-INPUT>
```

Output:

```
<?xml version="1.0"?>
<ISPF-OUTPUT>
 <SERVICE-REQUEST>
<request>DORMANT</request>
<sessid>&VER=1&ASID=88&STOKEN=000001600000006C&INDEX=1&MSGQID=3342342&TYPE=TSO</sessid>
<cmdresp></cmdresp>
 </SERVICE-REQUEST>
 <SERVICE-RESPONSE>
  <RETURN-CODE>0</RETURN-CODE>
 </SERVICE-RESPONSE>
 <OPERATIONS-LOG>
 <![CDATA[
  Content-type: text/plain

  ISPZINL started - z/OS 3.1 01JAN23 Base
  Data read from STDIN is
  &REQUEST=DORMANT&VER=1&ASID=88&STOKEN=000001600000006C&INDEX=1&MSGQID=3342342&TYPE=TSO
  *** XML-NOTE *** Reference tagged SERVICE-RESPONSE
 ]]>
 </OPERATIONS-LOG>
</ISPF-OUTPUT>
```

Using the native API to end a TSO/E address space

The following example shows the input and resulting output using the native API to end a TSO/E address space. In this example, the TSO/E address space, which is started by a previous gateway request, is logged off:

Input:

```
&REQUEST=LOGOFF &VER=1&ASID=106&STOKEN=000001A80000004E&INDEX=2&MSGQID=9764869&TYPE=ISPF
```

Output:

```
Content-type: text/plain

ISPZINL started - z/OS 3.1 01JAN23 Base
Data read from STDIN is
&REQUEST=LOGOFF &VER=1&ASID=106&STOKEN=000001A80000004E&INDEX=2&MSGQID=9764869&TYPE=ISPF
<ISPINFO>
RC=0
</ISPINFO>
```

Combining requests using the Interactive ISPF Gateway

Some types of requests can be combined into a single gateway input. For example, the client can send in to the gateway a single input that requests all of the following:

- Start a TSO/E address space
- Issue a command in the address space
- End the address space.

Combining request types in a single input has the following restrictions:

- Only one of NEWTSO, NEWTSOISPF, RECONNTSO, RECONNTSOISPF, and REUSE can be specified in a single input
- Only one of CANCEL, LOGOFF, and DORMANT can be specified in a single input

- RESPOND cannot be combined with any other request type
- ATTN cannot be combined with any other request type.

Using the XML API to combine requests

The following example shows the input and resulting output using the XML API to combine requests. This example combines two requests:

- Issue the TIME command by reusing the active TSO/E address space identified by the sessid parameter
- Ping the TSO/E address space to keep it active.

Input:

```
<?xml version="1.0"?>
<ISPF-INPUT>
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"ispf.xsd\">
<SERVICE-REQUEST>
<request>REUSE</request>
<sessid>&VER=1&ASID=88&STOKEN=0000016000000077&INDEX=2&MSGQID=10682373&TYPE=TSO</
sessid>
<request>PING</request>
<cmdresp>TIME</cmdresp>
</SERVICE-REQUEST>
</ISPF-INPUT>
```

Output:

```
<?xml version="1.0"?>
<ISPF-OUTPUT>
 <SERVICE-REQUEST>
<request>REUSE</request>
<sessid>&VER=1&ASID=88&STOKEN=0000016000000077&INDEX=2&MSGQID=10682373&TYPE=TSO</sessid>
<request>PING</request>
<cmdresp>TIME</cmdresp>
 </SERVICE-REQUEST>
 <SERVICE-RESPONSE>
  <TSO>
  <![CDATA[
   IKJ56650I TIME-10:01:24 PM. CPU-00:00:00 SERVICE-326 SESSION-00:00:54 JULY 3,2014
  ]]>
  </TSO>
  <RETURN-CODE>0</RETURN-CODE>
 </SERVICE-RESPONSE>
 <OPERATIONS-LOG>
 <![CDATA[
  Content-type: text/plain

  ISPZINL started - z/OS 3.1 01JAN23 Base
  Data read from STDIN is
  TIME&REQUEST=REUSE&VER=1&ASID=88&STOKEN=0000016000000077&INDEX=2
  &MSGQID=10682373&TYPE=TSO&REQUEST=PING
  *** OUTPUT FROM TSO SESSION ***
  *** XML-NOTE *** Reference tagged SERVICE-RESPONSE
 ]]>
 </OPERATIONS-LOG>
</ISPF-OUTPUT>
```

Using the native API to combine requests

The following example shows the input and resulting output using the native API to combine requests. This example combines two requests:

- Issue the TSO LISTCAT command in a new TSO/E address space in which ISPF is started
- After completing the command, logoff the address space.

Input:

```
TSO LISTC ENT('SYS1.LINKLIB') &REQUEST=NEWTSOISPF &PROCNAME=ISPFPROC &ACCTNUM=SAMPACCT
&GROUPID=DEFAULT &REGIONSZ=2000000 &REQUEST=LOGOFF
```

Output:

```
Content-type: text/plain

ISPZINL started - z/OS 3.1 01JAN23 Base
Data read from STDIN is TSO LISTC ENT('SYS1.LINKLIB') &REQUEST=NEWTSOISPF
&PROCNAME=ISPFPROC &ACCTNUM=SAMPACCT &GROUPID=DEFAULT &REGIONSZ=2000000 &REQUEST=LOGOFF
*** OUTPUT FROM ISPF SESSION ***
<ISPINFO>
Command passed to SELECT -
CMD(LISTC ENT('SYS1.LINKLIB'))
<ISPF>
NONVSAM ------- SYS1.LINKLIB
     IN-CAT --- CATALOG.MASTER.SYSPLEXA
</ISPF>
RC=0
</ISPINFO>
```

Controlling the Interactive ISPF Gateway logging level

To control the level of logging performed by the gateway during the processing of a request, include the loglevel parameter on the request. For a description of the logging levels provided by the gateway, see "Using the Interactive ISPF Gateway native API" on page 49.

Using the XML API to control the gateway logging level

The following example shows the input and resulting output using the XML API to control the gateway logging level. In this example, debug log information is requested:

Input:

```
<?xml version="1.0"?>
<ISPF-INPUT>
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"ispf.xsd\">
<SERVICE-REQUEST>
<loglevel>2</loglevel>
<request>NEWTSO</request>
<procname>ISPFPROC</procname>
<acctnum>SAMPACCT</acctnum>
<groupid>DEFAULT</groupid>
<regionsz>2000000</regionsz>
</SERVICE-REQUEST>
</ISPF-INPUT>
```

Output:

```
<?xml version="1.0"?>
<ISPF-OUTPUT>
 <SERVICE-REQUEST>
<loglevel>2</loglevel>
<request>NEWTSO</request>
<procname>ISPFPROC</procname>
<acctnum>SAMPACCT</acctnum>
<groupid>DEFAULT</groupid>
<regionsz>2000000</regionsz>
 </SERVICE-REQUEST>
 <SERVICE-RESPONSE>
  <RETURN-CODE>0</RETURN-CODE>
  <SESSION-INFO>
  <SESS>&VER=1&ASID=106&STOKEN=000001A800000052&INDEX=1&MSGQID=10223621&TYPE=TSO</
SESS>
  <TSOPROMPT>NO</TSOPROMPT>
  </SESSION-INFO>
 </SERVICE-RESPONSE>
 <OPERATIONS-LOG>
 <![CDATA[
  Content-type: text/plain

  ISPZINL started - z/OS 3.1 01JAN23 Base
  Data read from STDIN is

&LOGLEVEL=2&REQUEST=NEWTSO&PROCNAME=ISPFPROC&ACCTNUM=SAMPACCT&GROUPID=DEFAULT&REGIONS
Z=2000000
  ISPZTrc2 Received LogLevel  = 2
  ISPZTrc2 - Client requested log level: DEBUG
  ISPZTrc2 Received REQUEST   = NEWTSO
  ISPZTrc2 - Client requested start of new TSO session without ISPF active
  ISPZTrc2 Received PROCNAME  = ISPFPROC
  ISPZTrc2 Received ACCTNUM   = SAMPACCT
  ISPZTrc2 Received GROUPID   = DEFAULT
  ISPZTrc2 Received REGIONSZ  = 2000000
  ISPZTrc2 No command received
  ISPZTrc2 ****************************************
  ISPZTrc2 Calling CEATsoRequest() with CeaTsoStart
  ISPZTrc2         CEATSO_NORECONN ON
  ISPZTrc2 ****************************************
  ISPZTrc2 Connected to a new TSO/E session
  ISPZTrc2 CeaTsorequest Asid       = 106
  ISPZTrc2 CeaTsoRequest Stoken     = 000001A800000052
  ISPZTrc2 CeaTsorequest Index      = 1
  ISPZTrc2 CeaTsorequest Msgqueueid = 10223621
  ISPZTrc2 ****** START OF TSO LOGON MESSAGES ******
  ISPZTrc2 IKJ56455I USER1 LOGON IN PROGRESS AT 00:25:34 ON JULY 17, 2021
  ISPZTrc2 IKJ56951I NO BROADCAST MESSAGES
  ISPZTrc2 ******  END OF TSO LOGON MESSAGES  ******
  *** XML-NOTE *** Reference tagged SERVICE-RESPONSE
 ]]>
 </OPERATIONS-LOG>
</ISPF-OUTPUT>
```

Using the native API to control the gateway logging level

The following example shows the input and resulting output using the native API to control the gateway logging level. In this example, debug and communication log information is requested:

Input:

```
TSO TIME &LOGLEVEL=6 &REQUEST=REUSE
&VER=1&ASID=97&STOKEN=0000018400000046&INDEX=2&MSGQID=3801094&TYPE=ISPF
```

Output:

```
Content-type: text/plain


ISPZINL started - z/OS 3.1 01JAN23 Base


Data read from STDIN is
TSO TIME &LOGLEVEL=6 &REQUEST=REUSE
&VER=1&ASID=97&STOKEN=0000018400000046&INDEX=2&MSGQID=3801094&TYPE=ISPF
ISPZTrc2 Received LogLevel = 6
ISPZTrc2 - Client requested log level: DEBUG COMM
ISPZTrc2 Received REQUEST  = REUSE
ISPZTrc2 - Client requested reuse of existing session
ISPZTrc2 Received VER      = 1
ISPZTrc2 Received ASID     = 97
ISPZTrc2 Received STOKEN   = 0000018400000046
ISPZTrc2 Received INDEX    = 2
ISPZTrc2 Received MSGQID   = 3801094
ISPZTrc2 Received TYPE     = ISPF
ISPZTrc2 - ISPF session specified
ISPZTrc2 Received command  = TSO TIME
ISPZTrc2 Sending message (type=32771): TSO TIME
ISPZTrc4   Sent Message Type:    32771
```

```
ISPZTrc4   Sent Message Length:  9
ISPZTrc4   Sent Message Text:
ISPZTrc4       TSO TIME
*** OUTPUT FROM ISPF SESSION ***
ISPZTrc4   Received Message Type:     2
ISPZTrc4   Received Message Length:  122
ISPZTrc4   Received Message Text:
ISPZTrc4 {"TSO MESSAGE":{"VERSION":"0100","DATA":"Entering ISPZCMD: TSO/ISPF session
initiated
ISPZTrc4 2015071-BASE z22 15/07/17 01:35 "}}  Entering ISPZCMD: TSO/ISPF session initiated
2015071-BASE z22 15/07/17 01:35
ISPZTrc4   Received Message Type:     2
ISPZTrc4   Received Message Length:  89
ISPZTrc4   Received Message Text:
ISPZTrc4 {"TSO MESSAGE":{"VERSION":"0100","DATA":"Entering service call processing 01:35:00.27
"}}  Entering service call processing 01:35:00.27
ISPZTrc4   Received Message Type:     2
ISPZTrc4   Received Message Length:  85
ISPZTrc4   Received Message Text:
ISPZTrc4 {"TSO MESSAGE":{"VERSION":"0100","DATA":"Waiting for message on UNIX message queue"}}
Waiting for message on UNIX message queue
ISPZTrc4   Received Message Type:     2
ISPZTrc4   Received Message Length:  72
ISPZTrc4   Received Message Text:
ISPZTrc4 {"TSO MESSAGE":{"VERSION":"0100","DATA":"Message received: TSO TIME  "}}  Message
received: TSO TIME
ISPZTrc4   Received Message Type:     2
ISPZTrc4   Received Message Length:  53
ISPZTrc4   Received Message Text:
ISPZTrc4 {"TSO MESSAGE":{"VERSION":"0100","DATA":"<ISPINFO>"}}
<ISPINFO>
ISPZTrc4   Received Message Type:     2
ISPZTrc4   Received Message Length:  71
ISPZTrc4   Received Message Text:
ISPZTrc4 {"TSO MESSAGE":{"VERSION":"0100","DATA":"Command passed to SELECT - "}}   Command
passed to SELECT -
ISPZTrc4   Received Message Type:     2
ISPZTrc4   Received Message Length:  53
ISPZTrc4   Received Message Text:
ISPZTrc4 {"TSO MESSAGE":{"VERSION":"0100","DATA":"CMD(TIME)"}}
CMD(TIME)
ISPZTrc4   Received Message Type:     2
ISPZTrc4   Received Message Length:  50
ISPZTrc4   Received Message Text:
ISPZTrc4 {"TSO MESSAGE":{"VERSION":"0100","DATA":"<ISPF>"}}
<ISPF>
ISPZTrc4   Received Message Type:     2
ISPZTrc4   Received Message Length:  127
ISPZTrc4   Received Message Text:
ISPZTrc4 {"TSO MESSAGE":{"VERSION":"0100","DATA":"IKJ56650I TIME-01:35:41 AM.
ISPZTrc4 CPU-00:00:00 SERVICE-3671 SESSION-00:00:41 JULY 17,2015"}}  IKJ56650I TIME-01:35:41
AM. CPU-00:00:00 SERVICE-3671 SESSION-00:00:41 JULY 17,2015
ISPZTrc4   Received Message Type:     2
ISPZTrc4   Received Message Length:  51
ISPZTrc4   Received Message Text:
ISPZTrc4 {"TSO MESSAGE":{"VERSION":"0100","DATA":"</ISPF>"}}                       </
ISPF>
ISPZTrc4   Received Message Type:     2
ISPZTrc4   Received Message Length:  49
ISPZTrc4   Received Message Text:
ISPZTrc4 {"TSO MESSAGE":{"VERSION":"0100","DATA":"RC=0 "}}
RC=0
ISPZTrc4   Received Message Type:     2
ISPZTrc4   Received Message Length:  54
ISPZTrc4   Received Message Text:
ISPZTrc4 {"TSO MESSAGE":{"VERSION":"0100","DATA":"</ISPINFO>"}}                </
ISPINFO>
ISPZTrc4   Received Message Type:     2
ISPZTrc4   Received Message Length:  68
ISPZTrc4   Received Message Text:
ISPZTrc4 {"TSO MESSAGE":{"VERSION":"0100","DATA":"Complete sent by ISPZCMD"}}
Complete sent by ISPZCMD
<SESSION-INFO>
<TSOPROMPT>NO</TSOPROMPT>
</SESSION-INFO>
```

Note: Each line of log information that is included in the output is prepended with the string ISPZTrcn, where n is the decimal log level that caused that line to be included in the output. For example, a line that is the output for error logging is prepended with the string ISPZTrc1.

Interactive ISPF Gateway return codes

When an error is detected in the input received by the gateway or an error occurs during the processing of a request, a return code is included in the output returned to the client. In most cases, repeating the command with a loglevel parameter value that includes error logging information provides additional information about the error.

Some errors that occur during the processing of a request are detected by the CeaTsoRequest() function of CEA TSO/E address space services. Information about the return codes, reason codes and diagnostic codes returned for these problems can be found in **Return, reason, and diagnostic codes** in **z/OS MVS Programming: Callable Services for High-Level Languages**.

Some errors that occur during the processing of a request are detected by C functions that are called by the gateway. When this occurs, the return code returned by the gateway indicates the failing function. The additional information provided in the error logging information includes the return code and reason code returned by the function. Information about these return codes and reason codes can be found in the section documenting the function in **z/OS XL C/C++ Runtime Library Reference**.

The following return codes are returned by the ISPF gateway itself:

-87
A TSO logon parameter (for example, procedure name or account number) is not valid.

-88
Conflicting parameters are specified on the gateway request.

-89
A message was received on the message queue with a message type of ISPF. This is not expected.

-90
An error occurred on a call to the iconv_close() function.

-91
An error occurred on a call to the iconv_open() function.

-92
An error occurred on a call to the iconv() function.

-93
An error occurred on a call to the msgsnd() function.

-94
An error occurred on a call to the __msgrcv_timed() function.

-95
An error occurred on a call to the setenv() function.

-96
The response contains no text.

-97
No data was read from STDIN.

-98
A parameter value on the gateway request is bad.

-99
A required parameter is missing on the gateway request.

Receiving return code information using the XML API

The following example shows the return code information received using the XML API. In this example, the procname parameter is not provided on a request to start a new TSO/E address space:

```
<?xml version="1.0"?>
<ISPF-OUTPUT>
 <SERVICE-REQUEST>
```

```
<request>NEWTSO</request>
<acctnum>SAMPACCT</acctnum>
<groupid>DEFAULT</groupid>
<regionsz>2000000</regionsz>
 </SERVICE-REQUEST>
 <SERVICE-RESPONSE>
  <RETURN-CODE>-99</RETURN-CODE>
 </SERVICE-RESPONSE>
 <OPERATIONS-LOG>
 <![CDATA[
  Content-type: text/plain

  ISPZINL started - z/OS 3.1 01JAN23 Base
  Data read from STDIN is &REQUEST=NEWTSO&ACCTNUM=SAMPACCT&GROUPID=DEFAULT
  *** XML-NOTE *** Reference tagged SERVICE-RESPONSE
      82 *-* 'ISPZINL'
         +++ RC(157) +++
 ]]>
 </OPERATIONS-LOG>
</ISPF-OUTPUT>
```

Repeating the command with a loglevel value that includes error information causes the following line to be included in the OPERATIONS_LOG section of the output:

```
***ERROR: PROCNAME, ACCTNUM, GROUPID, and REGIONSZ must be provided to start a session
```

Receiving return code information using the native API

The following example shows the return code information received using the native API. In this example, the session identification information provided on a call to CeaTsoRequest() to end an address space does not identify a known address space:

```
Content-type: text/plain

ISPZINL started - z/OS 3.1 01JAN23 Base
Data read from STDIN is
&REQUEST=PING &VER=1&ASID=97&STOKEN=0000018400000046&INDEX=2&MSGQID=3801094&TYPE=ISPF
<ISPINFO>
RC=-1
RSN=0483100A
</ISPINFO>
    82 *-* 'ISPZINL'
       +++ RC(255) +++
```

Repeating the command with a loglevel value that includes error information causes the following lines to be included in the output:

```
*** CEATsoPing failed ***
CEATsoRequest Asid     = 97
CeaTsoRequest Stoken   = 0000018400000046
CEATSORequest Index    = 2
CEATsoRequest rc(dec)  = -1
CEATsoRequest rc(hex)  = FFFFFFFF
CEATsoError rc(hex)    = FFFFFFFF
CEATsoerror rsn(hex)   = 0483100A
CEATsoError diag1(hex) = 00000013
CEATsoError diag2(hex) = 00000001
```

Legacy ISPF Gateway

This section provides information about installing, customizing, and using the Legacy ISPF Gateway. For information about the Interactive ISPF Gateway and the Legacy ISPF Gateway and the benefits of using the Interactive ISPF Gateway, see “TSO/ISPF client gateway” on page 43 .

To use the Legacy ISPF Gateway, ensure that the environment variable CGI_CEATSO is not defined or that it is set to a value other than TRUE before invoking the gateway.

Legacy ISPF Gateway description

The gateway runs within z/OS UNIX and is invoked upon a request for a TSO or ISPF command from the client application. To run a requested command, the gateway establishes a TSO address space and, if the request is for an ISPF command, data sets are allocated and an ISPF session is also initialized. After completion of the command, the results are returned to the client.

If requested by the client, the gateway maintains the state of the TSO/ISPF session for any subsequent client function requests for the user. Otherwise, the interface terminates the TSO/ISPF session. The advantages of maintaining a user's TSO/ISPF session on z/OS are in bypassing the overhead of having to establish a new session for every TSO or ISPF command request and in adding an inherent ability to maintain state data between calls from the client.

The gateway does not provide the data transport between the client and z/OS host. The gateway is designed to interface with an authenticated connection between the client and z/OS host such as the HTTP protocol (web-based services), direct TCP/IP socket connections, or any other means through which data and commands can be exchanged between the client and z/OS host.

Note: ISPF commands invoked by means of the gateway run in an ISPF batch environment.

Installing and customizing the Legacy ISPF Gateway

This section describes installing and customizing the gateway.

Installation considerations

Consider these points before you configure your system:

- A RACF OMVS segment (or equivalent) that specifies a valid non-zero uid, home directory, and shell command must be defined for each user of the interface.
- Set MAXPROCUSER in BPXPRM*xx* parmlib member to a minimum of 50. This can be checked and set dynamically (until the next IPL) with the following commands (as described in *z/OS MVS System Commands*, SA22-7627):

  ```
  DISPLAY OMVS,O
  SETOMVS MAXPROCUSER=50.
  ```

  Setting a value that is too low can cause the interface to fail.
- Table 3 on page 44 describes the load modules that comprise the gateway. These load modules must be available to run under the server used to communicate between the client and z/OS host.
- A PROGRAM class profile that identifies ISPZINT as being a controlled program should be defined using the command:

  ```
  RDEFINE PROGRAM ISPZINT  ADDMEM('ISP.SISPLPA'//NOPADCHK) UACC(READ)
  ```

  (Refer to your security server's documentation for details.) Failure to define ISPZINT as being controlled might result in messages, such as CSV042I or ICH422I, being issued.
- Table 4 on page 45 describes the gateway files installed on the z/OS UNIX file system.
- For additional steps that must be completed depending on the environment from which the gateway is called, see "Customizing the gateway calling environment" on page 82.

Installing the Legacy ISPF Gateway

To install the gateway, customize and run the JCL in the member ISPZINS1 in the ISPF samples data set ISP.SISPSAMP. Customize the JCL according to the instructions in the member.

This job performs these tasks:

- Creates CONFIG and WORKAREA directories in the z/OS UNIX file system at the directory you specify.
- Copies the sample ISPF configuration table from member ISPZISPC in the samples data set ISP.SISPSAMP to file ISPF.conf in the CONFIG directory. This file requires customization.

The recommended base directory for the configuration files is /etc/ispf. The part of the directory up to ISPF must exist before running this job.

You must have read and write access to the WORKAREA directory /var/ispf/WORKAREA. The WORKAREA is used for the transfer of files. Temporary directories of the format /var/ispf/WORKAREA/userid/* are created during use of the interface.

Note: Some temporary session files might be created in the /tmp directory. Ensure all users have write access to the /tmp directory.

The interface removes any temporary files it creates in the WORKAREA directory. However, temporary output is sometimes left over, for example, if there is a communication error while processing. For this reason, it is recommended that you clear out the WORKAREA directory from time to time. To do this, use these commands in OMVS:

```
cd /var/ispf/WORKAREA
rm -r *
```

Where `/var/ispf/WORKAREA` depends on where you create the WORKAREA directory.

Customizing the Legacy ISPF Gateway

You must customize the ISPF configuration file ISPF.conf to host site requirements for ISPF data set allocation. This file is stored, by default, in directory `/etc/ispf`. The provided sample ISPF.conf has instructions to complete customization so your user site can:

- Include the minimum ISPF data set allocations for the gateway to operate. This means allocating the minimum ISPF data sets required to initialize an ISPF session. In the sample provided, these are specified as the isp.sisp* data sets. You might need to change these for your site-specified data set names.
- Add additional DDNAME file allocations or concatenate additional ISPF data sets.
- Launch a customer-defined allocation executable (exec) to provide further data set allocation by user ID. A sample exec is provided in member ISPZISP2 in the ISPF samples data set ISP.SISPSAMP.

The allocations for each of the ISPF DDs must be specified on a single line with each data set separated by a comma. The maximum length of the string defining the data sets for a DD is 255 characters. If the data sets you want to define for a DD exceed this length, use the customer defined allocation exec to allocate the DD. Comment lines can be added by beginning the line with an asterisk (*). Here is a sample ISPF.conf:

```
* REQUIRED:
* Below is the minimum requirements for ISPF allocation.
* Change the default ISPF data set names below to match your
* host site.
* Add additional dsn concatenations on same line and separate
* by comma.
* Order of data sets listed is search order in concatenation.
*
sysproc=ISP.SISPCLIB
sysexec=SYS1.LOCAL.EXEC,ISP.SISPEXEC
ispmlib=ISP.SISPMENU
isptlib=ISP.SISPTENU
ispplib=ISP.SISPPENU
ispslib=ISP.SISPSLIB
ispllib=ISP.SISPLOAD,SYS1.LOCAL.LOAD
```

The ISPF_timeout option in the ISPF configuration file can be used to specify a time out value for reusable ("stateful") ISPF sessions. The time out value specifies the number of seconds a user's ISPF session can remain idle between service call requests. If the idle time exceeds the time out value, the session is terminated and the next service request for the user results in a new ISPF session being established. The default reusable ISPF session time out value is 900 seconds (15 minutes).

Here is an example of setting a time out value of 300 seconds (5 minutes):

```
ISPF_timeout = 300
```

Note: If you have the PDSMAN product installed, it must be disabled for use with the gateway by including the record in the ISPF.conf file:

```
ezyoff=nullfile
```

Using the Legacy ISPF Gateway

This section describes how to provide TSO/ISPF service and command requests to the gateway and the format of the output returned by the gateway.

Passing requests to the Legacy ISPF Gateway

TSO/ISPF service and command requests are passed to the gateway in XML format. The XML schema shown here, which is supplied in member ISPZXSDI in the ISPF samples data set ISP.SISPSAMP, describes the format and can be used to validate the XML for a service request:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="ISPF-INPUT">
 <xs:complexType>
  <xs:all>

   <xs:element name="SERVICE-REQUEST">
    <xs:complexType>
     <xs:all>

   <xs:element name="service">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <!-- Specifies native TSO or ISPF service call -->
      <xs:enumeration value="ISPF"/>
      <xs:enumeration value="TSO"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>

   <xs:element name="session" minOccurs="0">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <!-- Default NONE : Session terminates after service call -->
      <xs:enumeration value="NONE"/>
      <!-- Reusable ISPF session stays active between calls -->
      <xs:enumeration value="REUSE"/>
     </xs:restriction>
    </xs:simpleType>
   </xs:element>

    <!-- Use existing ISPF profile in call -->
   <xs:element name="ispprof" type="xs:string" minOccurs="0"/>

    <!-- Free form TSO/ISPF command -->
   <xs:element name="command" type="xs:string"/>

     </xs:all>
    </xs:complexType>
   </xs:element>

  </xs:all>
 </xs:complexType>
</xs:element>

</xs:schema>
```

This is an example of the XML to request a TSO LISTCAT command:

```
<?xml version=\"1.0\"?>
<ISPF-INPUT>
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"ispf.xsd\">
<SERVICE-REQUEST>
<service>TSO</service>
<session>NONE</session>
<command>LISTC ENT('SYS1.LINKLIB')</command>
```

```
</SERVICE-REQUEST>
</ISPF-INPUT>
```

This is an example of the XML to request to run REXX program DINFO in a "reusable" ISPF session which remains active to run subsequent commands for this user:

```
<?xml version=\"1.0\"?>
<ISPF-INPUT>
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:noNamespaceSchemaLocation=\"ispf.xsd\">
<SERVICE-REQUEST>
<service>ISPF/service>
<session>REUSE</session>
<command>TSO DINFO sys1.linklib</command>
<ispprof>USER.ISPPROF</ispprof>
</SERVICE-REQUEST>
</ISPF-INPUT>
```

This shows the REXX program DINFO:

```
/* REXX */
parse upper arg dsn .
address "ISPEXEC" "dsinfo dataset('"dsn"')"
if rc = 0 then do
  say 'Volume              = ' zdsvol
  say 'Primary allocation  = ' strip(zds1ex) zdsspc
  say 'Secondary allocation = ' strip(zds2ex) zds2spc
end
else do
  say 'DSINFO rc = ' rc
  say 'ZERRMSG   = ' zerrmsg
end
exit rc
```

Note: System symbols &SYSUID and &SYSPREF can be used as the high-level qualifier for the ISPF profile data set name specified with the <ispprof> tag.

Receiving output from the Legacy ISPF Gateway

The output from a service request is returned by the gateway to the client in XML format. The XML schema shown here, which is supplied in member ISPZXSDO in the ISPF samples data set ISP.SISPSAMP, describes the format and can be used to process the XML returned for a service request:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="ISPF-OUTPUT">
 <xs:complexType>
  <xs:all>

   <xs:element name="SERVICE-REQUEST"/>

   <xs:element name="SERVICE-RESPONSE">
    <xs:complexType>
     <xs:all>
      <xs:element name="ISPF-COMMAND"/>
      <xs:element name="RETURN-CODE"/>
      <xs:element name="ISPF"/>
     </xs:all>
    </xs:complexType>
   </xs:element>

   <xs:element name="OPERATIONS-LOG"/>

  </xs:all>
 </xs:complexType>
</xs:element>

</xs:schema>
```

Here is the XML returned for the TSO LISTCAT command above:

```
<?xml version="1.0"?>
<ISPF-OUTPUT>
 <SERVICE-REQUEST>
```

```
<service>TSO</service>
<session>NONE</session>
<command>LISTC ENT('SYS1.LINKLIB')</command>
 </SERVICE-REQUEST>
 <SERVICE-RESPONSE>
  <ISPF-COMMAND>

  </ISPF-COMMAND>
  <![CDATA[
   NONVSAM ------- SYS1.LINKLIB
        IN-CAT --- CATALOG.MASTER.SYSPLEXD
   READY
   END
  ]]>
  </TSO>
 </SERVICE-RESPONSE>
 <OPERATIONS-LOG>
 <![CDATA[
  Content-type: text/plain

  Entering ISPZINT (Service initialization)
  About to read from fileno(stdin) = 0
  Data read from STDIN is TSO LISTC ENT('SYS1.LINKLIB')
  EPOCH secs = 1202109526
  Local Date & time: Mon Feb  4 02:18:46 2008
  Hour: 2 Min: 18 Sec 46
  Function ID timestamp = ID008326
  Environment variables:
  0 QUERY_STRING=
  1 CONTENT_TYPE=application/x-www-form-urlencoded
  2 PATH=/bin:.:/usr/sbin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin:
  /usr/lpp/products/java142/J1.4/bin/
  3 AUTH_TYPE=Basic
  4 DOCUMENT_URI=/ISPZXML
  5 SHELL=/bin/sh
  6 HTTPS=OFF
  7 HTTP_ACCEPT=text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
  8 HTTP_USER_AGENT=Java1.3.1
  9 SERVER_PORT=1726
  10 RULE_FILE=//DD:CONF
  11 GATEWAY_INTERFACE=CGI/1.1
  12 PATH_INFO=
  13 CONTENT_LENGTH=274
  14 _CEE_RUNOPTS=ENVAR("_CEE_ENVFILE=//DD:ENV")
  15 _BPX_SPAWN_SCRIPT=YES
  16 REFERER_URL=
  17 _=./ISPZINT
  18 CLASSPATH=.:/usr/lpp/internet/server_root/CAServlet
  19 STEPLIB=CURRENT
  20 REQUEST_METHOD=POST
  21 REMOTE_ADDR=9.190.236.239
  22 LANG=C
  23 LIBPATH=/bin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin
  24 REMOTE_USER=USERNME
  25 SERVER_ADDR=192.168.123.11
  26 FSCP=IBM-1047
  27 PATH_TRANSLATED=
  28 HTTP_CONNECTION=keep-alive
  29 SERVER_TOKEN=1
  30 HTTP_HOST=PTHISD1.AU.IBM.COM
  31 _BPX_SHAREAS=YES
  32 CGI_ISPCONF=/etc/ispf
  33 SERVER_SOFTWARE=IBM HTTP Server/V5R3M0
  34 REPORTBITS=77
  35 DOCUMENT_ROOT=usr/lpp/ispf/bin
  36 NETCP=ISO8859-1
  37 CGI_ISPWORK=/var/ispf
  38 COUNTERDIR=NULL
  39 LC_ALL=en_US.IBM-1047
  40 CGI_DTCONF=/etc/ispf
  41 _BPX_USERID=USERNME
  42 SERVER_PROTOCOL=HTTP/1.1
  43 JAVA_HOME=/usr/lpp/products/java142/J1.4/
  44 HTTPS_KEYSIZE=
  45 TZ=EST5EDT
  46 _CEE_ENVFILE=//DD:ENV
  47 _BPX_BATCH_SPAWN=SPAWN
  48 SCRIPT_NAME=/ISPZXML
  49 NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lpp/internet/%L/%N:
  /usr/lib/nls/msg/En_US.IBM-1047/%N
  50 CGI_DTWORK=/var/ispf
  51 DOCUMENT_NAME=usr/lpp/ispf/bin/ISPZXML
```

```
   52 SERVER_NAME=PTHISD1.AU.IBM.COM
   Number of environment variables is 53
   Protocol = HTTP
   FSCP = IBM-1047
   NETCP = ISO8859-1
   CGI_ISPCONF = /etc/ispf
   CGI_ISPWORK = /var/ispf
   CGI_TRANTABLE =
   Server PATH = /bin:.:/usr/sbin:/usr/lpp/internet/bin:
   /usr/lpp/internet/sbin:/usr/lpp/products/java142/J1.4/bin/
   About to spawn task for ISPZTSO
   Parameters passed to ISPZTSO - LISTC ENT('SYS1.LINKLIB')
   Return code from ISPZTSO is 0
   About to open /tmp/USERNME.ID008326.SYSTSPRT
   *** XML-NOTE *** Reference tagged SERVICE-RESPONSE
   OUT.84
  ]]>
  </OPERATIONS-LOG>
 <ISPF-OUTPUT>
```

Shown here is the XML returned from running the DINFO REXX program in a "reusable" ISPF session (see above):

```
<?xml version="1.0"?>
<ISPF-OUTPUT>
 <SERVICE-REQUEST>
<service>ISPF</service>
<session>REUSE</session>
<command>TSO DINFO sys1.linklib</command>
<ispprof>USERNME.TEST.ISPPROF</ispprof>
 </SERVICE-REQUEST>
 <SERVICE-RESPONSE>
  <ISPF-COMMAND>
   SELECT CMD(DINFO sys1.linklib) NEST
  </ISPF-COMMAND>
  <RETURN-CODE>0</RETURN-CODE>
  <ISPF>
  <![CDATA[
   Volume              =  $$SR86
   Primary allocation  =  2398 BLOCK
   Secondary allocation =  0 BLOCK
  ]]>
  </ISPF>
 </SERVICE-RESPONSE>
 <OPERATIONS-LOG>
 <![CDATA[
  Content-type: text/plain

  Entering ISPZINT (Service initialization)
  About to read from fileno(stdin) = 0
  Data read from STDIN is ISPF TSO DINFO sys1.linklib&SESSION=SPAWN
  &ISPPROF=USERNME.TEST.ISPPROF
  EPOCH secs = 1202113932
  Local Date & time: Mon Feb  4 03:32:12 2008
  Hour: 3 Min: 32 Sec 12
  Function ID timestamp = ID012732
  Environment variables:
  0 QUERY_STRING=
  1 CONTENT_TYPE=application/x-www-form-urlencoded
  2 PATH=/bin:.:/usr/sbin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin:
  /usr/lpp/products/java142/J1.4/bin/
  3 AUTH_TYPE=Basic
  4 DOCUMENT_URI=/ISPZXML
  5 SHELL=/bin/sh
  6 HTTPS=OFF
  7 HTTP_ACCEPT=text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
  8 HTTP_USER_AGENT=Java1.3.1
  9 SERVER_PORT=1726
  10 RULE_FILE=//DD:CONF
  11 GATEWAY_INTERFACE=CGI/1.1
  12 PATH_INFO=
  13 CONTENT_LENGTH=313
  14 _CEE_RUNOPTS=ENVAR("_CEE_ENVFILE=//DD:ENV")
  15 _BPX_SPAWN_SCRIPT=YES
  16 REFERER_URL=
  17 _=./ISPZINT
  18 CLASSPATH=.:/usr/lpp/internet/server_root/CAServlet
  19 STEPLIB=CURRENT
  20 REQUEST_METHOD=POST
  21 REMOTE_ADDR=9.190.236.239
```

```
22 LANG=C
23 LIBPATH=/bin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin
24 REMOTE_USER=USERNME
25 SERVER_ADDR=192.168.123.11
26 FSCP=IBM-1047
27 PATH_TRANSLATED=
28 HTTP_CONNECTION=keep-alive
29 SERVER_TOKEN=1
30 HTTP_HOST=PTHISD1.AU.IBM.COM
31 _BPX_SHAREAS=YES
32 CGI_ISPCONF=/etc/ispf
33 SERVER_SOFTWARE=IBM HTTP Server/V5R3M0
34 REPORTBITS=77
35 DOCUMENT_ROOT=usr/lpp/ispf/bin
36 NETCP=ISO8859-1
37 CGI_ISPWORK=/var/ispf
38 COUNTERDIR=NULL
39 LC_ALL=en_US.IBM-1047
40 CGI_DTCONF=/etc/ispf
41 _BPX_USERID=USERNME
42 SERVER_PROTOCOL=HTTP/1.1
43 JAVA_HOME=/usr/lpp/products/java142/J1.4/
44 HTTPS_KEYSIZE=
45 TZ=EST5EDT
46 _CEE_ENVFILE=//DD:ENV
47 _BPX_BATCH_SPAWN=SPAWN
48 SCRIPT_NAME=/ISPZXML
49 NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lpp/internet/%L/%N:
/usr/lib/nls/msg/En_US.
IBM-1047/%N
50 CGI_DTWORK=/var/ispf
51 DOCUMENT_NAME=usr/lpp/ispf/bin/ISPZXML
52 SERVER_NAME=PTHISD1.AU.IBM.COM
Number of environment variables is 53
Protocol = HTTP
FSCP = IBM-1047
NETCP = ISO8859-1
CGI_ISPCONF = /etc/ispf
CGI_ISPWORK = /var/ispf
CGI_TRANTABLE =
Server PATH = /bin:.:/usr/sbin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin:
/usr/lpp/products/java142/J1.4/bin/
ISPF standalone function invoked
&ISPPROF value = USERNME.TEST.ISPPROF
ISPF COMMAND = ISPF TSO DINFO sys1.linklib
ISPF PROFILE = USERNME.TEST.ISPPROF
Re-usable ISPF session = SPAWN
About to spawn task for ISPZTSO
Parameters passed to ISPZTSO - PROFILE
Return code from ISPZTSO is 0
About to process PROFILE data in /tmp/USERNME.ID012732.SYSTSPRT
About to malloc() 252 bytes for profdat
*** PROFILE data: 1IKJ56688I CHAR(0)  LINE(0)    PROMPT   INTERCOM
NOPAUSE MSGID  MODE
  WTPMSG   NORECOVER PREFIX(USERNME) PLANGUAGE(ENU) SLANGUAG E(ENU)
  VARSTORAGE(LOW)
  IKJ56689I
  DEFAULT LINE/CHARACTER DELETE CHARACTERS IN EFFECT FOR THIS TERMINAL READY END
Temporary data set prefix set to : USERNME
About to call bpxwdyn to allocate VCMTEMP
Allocating data set USERNME.SCLMDT.VCMISPF.ID012732 to the VCMTEMP DD
1024 bytes of ISPF TSO DINFO sys1.linklib written to VCMTEMP
1024 bytes of /etc/SCLMDTIS;/var/ispf written to VCMTEMP
1024 bytes of /bin:.:/usr/sbin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin:
/usr/lpp/products/java142/J1.4/bin/~~NONE written to VCMTEMP
Parameter to be passed to ISPZTSO CALL *(ISPZCNT)
'+ISPF ID012732 USERNME NONE NONE
NONE NONE NONE NONE NONE USERNME.TEST.ISPPROF ISPF TSO DINFO sys1.linklib'
Entering Spawn Processing: 03:32:12
PID of this process = 999
RESPfile is /var/ispf/WORKAREA/USERNME/ISPFPID.999
Group PID of this process = 50332642
About to issue system command: ps -u USERNME -o pid,pgid,jobname,xasid,comm
>/var/ispf/WORKAREA/USERNME/ISPZINT.pidlog
PID=6 PGID=50332642 JOBNAME=USERNME ASID=7A COMMAND=/bin/ps
PID=50332642 PGID=50332642 JOBNAME=USERNME ASID=90
COMMAND=/usr/lpp/ispf/bin/ISPZXML
PID=67109859 PGID=50332642 JOBNAME=USERNME ASID=8B COMMAND=/bin/sh
PID=16778213 PGID=50332642 JOBNAME=USERNME ASID=8B COMMAND=./ISPZXENV
PID=998 PGID=50332642 JOBNAME=USERNME ASID=80 COMMAND=/bin/sh
PID=999 PGID=50332642 JOBNAME=USERNME ASID=92 COMMAND=./ISPZINT
PID=16778216 PGID=50332642 JOBNAME=USERNME ASID=7A COMMAND=/bin/sh
```

```
    No active ISPF session found - new TSO/ISPF session started
    About to issue system command: rm /var/ispf/WORKAREA/USERNME/ISPFPID*
    mkdir /var/ispf/WORKAREA/USERNME rc = -1
    New SIGfile = /var/ispf/WORKAREA/USERNME/ISPFPID.signal
    New CMDfile = /var/ispf/WORKAREA/USERNME/ISPFPID.cmd
    RUN directory = /usr/lpp/ispf/bin/
    About to spawn task for ISPZTSO
    Parameters passed to ISPZTSO - CALL *(ISPZCNT) '+ISPF ID012732
    USERNME NONE NONE
    NONE NONE NONE NONE NONE USERNME.TEST.ISPPROF ISPF TSO DINFO sys1.linklib'
    Return code from ISPZTSO is 0
    Waiting on SIGNAL return 03:32:12
    Read signal file: Output = COMPLETE
    *** OUTPUT FROM ISPF SESSION ***
    Entering ISPZCNT (ISPF Initialization)
    Parameters +ISPF ID012732 USERNME NONE NONE NONE NONE NONE NONE NONE USERNME.
    TEST.ISPPROF ISPF TSO DINFO SYS1.L
    REC: sysproc=isp.sispclib,bzz.v1r1.sbzzclib,SYS2.CLIST.ISD1.USER,SCLMDW.V710.
    SFEKPROC,WD4Z.V710.SFEKPROC
    Allocation successful for SYSPROC
    REC: sysexec=sclmdw.v3.test.exec,USERNME.sclmdw.exec,USERNME.exec
    Allocation successful for SYSEXEC
    REC: ispmlib=isp.sispmenu
    Allocation successful for ISPMLIB
    REC: isptlib=isp.sisptenu
    Allocation successful for ISPTLIB
    REC: ispplib=isp.sisppenu
    Allocation successful for ISPPLIB
    REC: ispslib=sclmdw.v3.dev.skels,bzz.v1r1.sbzzsenu,isp.sispsenu,isp.sispslib
    Allocation successful for ISPSLIB
    REC: isptrace=nullfile
    Allocation successful for ISPTRACE
    REC: ISPF_timeout = 300
    IEBCOPY of ISPF profile RC = 00
    NOTE: Data set allocations took 0.49 elapsed seconds
    *** XML-NOTE *** Reference tagged SERVICE-RESPONSE
    Current Process List:
           PID       PGID JOBNAME  ASID COMMAND
      50331654   50332642 USERNME    7a /bin/sh
            10   50332642 USERNME    7a /bin/ps
      50332642   50332642 USERNME    90 /usr/lpp/ispf/bin/ISPZXML
      67109859   50332642 USERNME    8b /bin/sh
      16778213   50332642 USERNME    8b ./ISPZXENV
           998   50332642 USERNME    80 /bin/sh
           999   50332642 USERNME    92 ./ISPZINT
      50332648   50332642 USERNME4   89 ./ISPZTSO
  ]]>
  </OPERATIONS-LOG>
<ISPF-OUTPUT>
```

Return and reason codes from the Legacy ISPF Gateway

The RETURN-CODE tag provides the value of the return code from the TSO or ISPF service. If there is an error in the gateway, a value of 8 or 12 is returned with the RETURN-CODE tag and REASON-CODE tags of this format are also provided to describe the error:

```
<REASON-CODE id="isppcISPZ0007">ISPF OR SCLM SERVICE HAS ENDED
ABNORMALLY</REASON-CODE>
```

Table 5 on page 76 lists the possible reason codes when there is a return code of 8 from the gateway:

Table 5. Possible reason codes for a return code of 8 from the gateway

Reason Code	Description
ISPZ0001	Error in ALLOCATION/WRITE to *dsname*.
ISPZ0002	Processing terminates.
ISPZ0003	Following function has failed: MSG: *function*
ISPZ0009	Error reading the ISPF.conf configuration file.
ISPZ0010	Ensure file exists in directory specified by the environment variable CGI_ISPCONF in the server configuration.

Table 5. Possible reason codes for a return code of 8 from the gateway (continued)

Reason Code	Description
ISPZ0012	Error in ISPF data set allocation: See error message below
ISPZ0013	Error in allocating the following DD and data set names: MSG: DD=*ddname* MSG: *dsname1 dsname2... dsnamen*
ISPZ0014	Verify that the ISPF configuration file ISPF.conf on the host is correct.
ISPZ0224	*** Operation Cancelled ***
ISPZ0225	Possible previous request still active. Either cancel host session or retry once processing terminates.

Table 6 on page 77 lists the possible reason codes when there is a return code of 12 from the gateway:

Table 6. Possible reason codes for a return code of 12 from the gateway

Reason Code	Description
ISPZ0007	ISPF or SCLM service has ended abnormally.
ISPZ0008	Review log for error details.

Using the native API of the Legacy ISPF Gateway

This section documents the native API used by some IBM products to invoke the gateway. It is recommended, however, that applications use the XML structured API to invoke the gateway.

Service request format

Service call requests consist of a parameter string passed by means of STDIN to the interface module ISPZINT. The first word of the service request string identifies if the request is for a TSO or ISPF service:

```
TSO service-request
```

or

```
ISPF service-request
```

A TSO service request results in the TSO service running in a TSO address space without the overhead of establishing an ISPF environment. Shown here are some examples of TSO service requests:

```
TSO time
```

```
TSO alloc fi(temp1) da('hlq.filetemp') shr
```

An ISPF service request can be for a TSO or ISPF service. The service runs in an ISPF environment established in a TSO address space. The caller can request the ISPF environment be "reusable" (that is, remain active for running subsequent service requests) by including &SESSION=SPAWN in the parameter string. Also the caller can request an existing ISPF profile be dynamically allocated by including &ISPPROF= `my.profile.dataset` in the parameter string. Shown here are some examples of ISPF service requests:

```
ISPF tso profile
ISPF select cmd(%myexec)
ISPF tso listalc&SESSION=SPAWN&ISPPROF=HLQ.ISPPROF
ISPF tso ex 'my.dataset(exec1)'&SESSION=SPAWN
```

ISPPROF may contain the strings &SYSUID., &SYSPREF. and &SYSNAME. in any order. These strings can also be used in the CGI_ISPPREF environment variable. Note that the trailing dot (.) must exist for the variable to be recognized correctly.

&SYSUID.
: Replaced by the userid of the active session.

&SYSPREF.
: Replaced by the TSO PREFIX, if that can be determined; otherwise, it has the same value as &SYSUID..

&SYSNAME.
: Replaced by the SYSNAME IPL parameter.

For example, for this command (where PREFIX is `mypref` and SYSUID is `myuser`):

```
ISPF tso mycmd&SESSION=SPAWN&ISPPROF=HLQ.&SYSPREF..&SYSUID..LLQ
```

where &SYSPREF is "mypref" and &SYSUID is "myuse" the string becomes:

```
ISPF tso mycmd&SESSION=SPAWN&ISPPROF=HLQ.mypref.myuser.LLQ
```

To terminate a "reusable" ISPF session, pass the parameter string shown here to ISPZINT:

```
ISPFFUNC=SHUTDOWN
```

or

```
ISPFFUNC=CANCEL
```

Service response format

Service response data is returned in a tagged format as shown in this section. However this is not valid XML format. Tagged data may be incorporated within system log messages and be stripped out at the client end by means of the encapsulating tags <ISPINFO> </ISPINFO>.

```
System logging data
⋮
<ISPINFO>
ISPF COMMAND :  ISPF command issued
RC=X
MSG: xxxxxx
<Specific function request tags>
xxxxxx
xxxxxx
</End of specific function request tags>
</ISPINFO>
System logging data
⋮
```

Service request and response examples

Shown here is the request parameter string and response data returned for an ISPF service request:

Request

```
ISPF TSO PROFILE&SESSION=SPAWN&ISPPROF=&SYSUID..TEST.ISPPROF
```

Response

```
START Transfer DATE/TIME is :Tue Mar 18 12:25:59 WST 2008
At url openConnection
SCLMFunc returned from post is Connect
About to accept response from Server
Response from Server received
Entering ISPZINT (Service initialization)
About to read from fileno(stdin) = 0
Data read from STDIN is ISPF TSO PROFILE&SESSION=SPAWN&ISPPROF=
&SYSUID..TEST.ISPPROF
EPOCH secs = 1205814289
```

```
Local Date & time: Tue Mar 18 00:24:49 2008
Hour: 0 Min: 24 Sec 49
Function ID timestamp = ID001489
Environment variables:
0 _CEE_ENVFILE=//DD:ENV
1 # PATH=/bin:.:/usr/sbin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin:
/usr/lpp/products/java531-UK14829/J5.0/bin
2 PATH=/bin:.:/usr/sbin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin:
/usr/lpp/products/java142/J1.4/bin/
3 SHELL=/bin/sh
4 TZ=EST5EDT
5 LANG=C
6 LC_ALL=en_US.IBM-1047
7 NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lpp/internet/%L/%N:
/usr/lib/nls/msg/En_US.
IBM-1047/%N
8 LIBPATH=/bin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin
9 # JAVA_HOME=/usr/lpp/products/java531-UK14829/J5.0/bin
10 JAVA_HOME=/usr/lpp/products/java142/J1.4/
11 CLASSPATH=.:/usr/lpp/internet/server_root/CAServlet
12 STEPLIB=CURRENT
13 _BPX_BATCH_SPAWN=SPAWN
14 _BPX_SHAREAS=YES
15 # _BPX_SHAREAS=YES
16 # _BPX_SPAWN_SCRIPT=NO
17 CGI_DTCONF=/etc/SCLMDTIS
18 CGI_DTWORK=/var/SCLMDTIS
19 CGI_ISPCONF=/etc/SCLMDTIS
20 CGI_ISPWORK=/var/SCLMDTIS
21 # CGI_TRANTABLE=DOHERTL.LSTRANS.FILE
22 COUNTERDIR=NULL
23 REPORTBITS=77
24 SERVER_SOFTWARE=IBM HTTP Server/V5R3M0
25 SERVER_NAME=PTHISD1.AU.IBM.COM
26 SERVER_PORT=1726
27 _BPX_SPAWN_SCRIPT=YES
28 _BPX_USERID=USERNME
29 RULE_FILE=//DD:CONF
30 SERVER_PROTOCOL=HTTP/1.1
31 REQUEST_METHOD=POST
32 GATEWAY_INTERFACE=CGI/1.1
33 PATH_INFO=
34 PATH_TRANSLATED=
35 QUERY_STRING=
36 SERVER_ADDR=192.168.123.11
37 SERVER_TOKEN=1
38 SCRIPT_NAME=/ISPZINT
39 REMOTE_ADDR=192.168.128.253
40 AUTH_TYPE=Basic
41 REMOTE_USER=USERNME
42 CONTENT_TYPE=application/x-www-form-urlencoded
43 CONTENT_LENGTH=61
44 REFERER_URL=
45 DOCUMENT_ROOT=usr/lpp/ispf/bin
46 DOCUMENT_URI=/ISPZINT
47 DOCUMENT_NAME=usr/lpp/ispf/bin/ISPZINT
48 FSCP=IBM-1047
49 NETCP=ISO8859-1
50 HTTPS_KEYSIZE=
51 HTTPS=OFF
52 HTTP_CONNECTION=keep-alive
53 HTTP_ACCEPT=text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
54 HTTP_HOST=PTHISD1.AU.IBM.COM
55 HTTP_USER_AGENT=Java/1.5.0
56 HTTP_PRAGMA=no-cache
57 HTTP_CACHE_CONTROL=no-cache
58 _CEE_RUNOPTS=ENVAR("_CEE_ENVFILE=//DD:ENV")
Number of environment variables is 59
Connection Protocol = HTTP
Server Name  = PTHISD1.AU.IBM.COM
Server Port  = 1726
FSCP = IBM-1047
NETCP = ISO8859-1
CGI_ISPCONF = /etc/SCLMDTIS
CGI_ISPWORK = /var/SCLMDTIS
Server PATH = /bin:.:/usr/sbin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin:
/usr/lpp/products/java142/J1.4/bin/
ISPF standalone function invoked
&ISPPROF value = &SYSUID..TEST.ISPPROF
ISPF COMMAND = ISPF TSO PROFILE
ISPF PROFILE = USERNME.TEST.ISPPROF
Re-usable ISPF session = SPAWN
```

```
About to spawn task for ISPZTSO
Parameters passed to ISPZTSO - PROFILE
Return code from ISPZTSO is 0
About to process PROFILE data in /tmp/USERNME.ID001489.ISPF.SYSTSPRT
About to malloc() 252 bytes for profdat
Temporary data set prefix set to : USERNME
About to call bpxwdyn to allocate VCMTEMP
Allocating data set USERNME.ISPF.VCMISPF.ID001489 to the VCMTEMP DD
1024 bytes of ISPF TSO PROFILE written to VCMTEMP
1024 bytes of /etc/SCLMDTIS;/var/SCLMDTIS written to VCMTEMP
1024 bytes of /bin:.:/usr/sbin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin:
/usr/lpp/products/java142/J1.4/bin/~~ written to VCMTEMP
Parameter to be passed to ISPZTSO CALL *(ISPZCNT) '+ISPF ID001489 USERNME
USERNME.TEST.ISPPROF ISPF TSO PROFILE'
Entering Spawn Processing: 00:24:49
PID of this process = 67109485
RESPfile is /var/SCLMDTIS/WORKAREA/USERNME/ISPFPID.67109485
Group PID of this process = 67109485
No ISPZINT.pidlog detected so no existing ISPF session running
No active ISPF session found - new TSO/ISPF session started
About to issue system command: rm /var/SCLMDTIS/WORKAREA/USERNME/ISPFPID*
mkdir /var/SCLMDTIS/WORKAREA/USERNME rc = -1
New SIGfile = /var/SCLMDTIS/WORKAREA/USERNME/ISPFPID.signal
New CMDfile = /var/SCLMDTIS/WORKAREA/USERNME/ISPFPID.cmd
RUN directory = /usr/lpp/ispf/bin/
About to spawn task for ISPZTSO
Parameters passed to ISPZTSO - CALL *(ISPZCNT) '+ISPF ID001489 USERNME
USERNME.TEST.ISPPROF ISPF TSO PROFILE'
Return code from ISPZTSO is 0
Waiting on SIGNAL return 00:24:49
Read signal file: Output = COMPLETE
*** OUTPUT FROM ISPF SESSION ***
<ISPINFO>
ISPF COMMAND : SELECT CMD(PROFILE) NEST
RC=0
<ISPF>
IKJ56688I CHAR(0)  LINE(0)    PROMPT   INTERCOM   NOPAUSE MSGID
MODE    WTPMSG   NORECOVER PREFIX(USERNME) PLANGUAGE(ENU)
SLANGUAGE(ENU) VARSTORAGE(LOW)
IKJ56689I DEFAULT LINE/CHARACTER DELETE CHARACTERS IN EFFECT FOR
THIS TERMINAL
</ISPF>
</ISPINFO>
Current Process List:
       PID       PGID JOBNAME  ASID COMMAND
        94   67109485 USERNME    8a /bin/ps
  50332025   67109485 USERNME3   98 ./ISPZTSO
  16777835   16777835 USERNME    9b ISRUUDL
  67109485   67109485 USERNME    95 /usr/lpp/ispf/bin/ISPZINT
  50332289   67109485 USERNME    8a /bin/sh
RC from doGet then writefile :ok
TOTAL Completion DATE/TIME is :Tue Mar 18 12:26:02 WST 2008
```

Shown here is the request parameter string and response data returned for a TSO service request.

Request

```
TSO LISTALC
```

Response

```
START Transfer DATE/TIME is :Tue Mar 18 12:33:55 WST 2008
At url openConnection
SCLMFunc returned from post is Connect
About to accept response from Server
Response from Server received
Entering ISPZINT (Service initialization)
About to read from fileno(stdin) = 0
Data read from STDIN is TSO LISTALC
EPOCH secs = 1205814765
Local Date & time: Tue Mar 18 00:32:45 2008
Hour: 0 Min: 32 Sec 45
Function ID timestamp = ID001965
Environment variables:
0 _CEE_ENVFILE=//DD:ENV
1 # PATH=/bin:.:/usr/sbin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin:
/usr/lpp/products/java531-UK14829/J5.0/bin
2 PATH=/bin:.:/usr/sbin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin:
/usr/lpp/products/java142/J1.4/bin/
```

```
3 SHELL=/bin/sh
4 TZ=EST5EDT
5 LANG=C
6 LC_ALL=en_US.IBM-1047
7 NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lpp/internet/%L/%N:
/usr/lib/nls/msg/En_US.
IBM-1047/%N
8 LIBPATH=/bin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin
9 # JAVA_HOME=/usr/lpp/products/java531-UK14829/J5.0/bin
10 JAVA_HOME=/usr/lpp/products/java142/J1.4/
11 CLASSPATH=.:/usr/lpp/internet/server_root/CAServlet
12 STEPLIB=CURRENT
13 _BPX_BATCH_SPAWN=SPAWN
14 _BPX_SHAREAS=YES
15 # _BPX_SHAREAS=YES
16 # _BPX_SPAWN_SCRIPT=NO
17 CGI_DTCONF=/etc/SCLMDTIS
18 CGI_DTWORK=/var/SCLMDTIS
19 CGI_ISPCONF=/etc/SCLMDTIS
20 CGI_ISPWORK=/var/SCLMDTIS
21 # CGI_TRANTABLE=DOHERTL.LSTRANS.FILE
22 COUNTERDIR=NULL
23 REPORTBITS=77
24 SERVER_SOFTWARE=IBM HTTP Server/V5R3M0
25 SERVER_NAME=PTHISD1.AU.IBM.COM
26 SERVER_PORT=1726
27 _BPX_SPAWN_SCRIPT=YES
28 _BPX_USERID=USERNME
29 RULE_FILE=//DD:CONF
30 SERVER_PROTOCOL=HTTP/1.1
31 REQUEST_METHOD=POST
32 GATEWAY_INTERFACE=CGI/1.1
33 PATH_INFO=
34 PATH_TRANSLATED=
35 QUERY_STRING=
36 SERVER_ADDR=192.168.123.11
37 SERVER_TOKEN=1
38 SCRIPT_NAME=/ISPZINT
39 REMOTE_ADDR=192.168.128.253
40 AUTH_TYPE=Basic
41 REMOTE_USER=USERNME
42 CONTENT_TYPE=application/x-www-form-urlencoded
43 CONTENT_LENGTH=12
44 REFERER_URL=
45 DOCUMENT_ROOT=usr/lpp/ispf/bin
46 DOCUMENT_URI=/ISPZINT
47 DOCUMENT_NAME=usr/lpp/ispf/bin/ISPZINT
48 FSCP=IBM-1047
49 NETCP=ISO8859-1
50 HTTPS_KEYSIZE=
51 HTTPS=OFF
52 HTTP_CONNECTION=keep-alive
53 HTTP_ACCEPT=text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
54 HTTP_HOST=PTHISD1.AU.IBM.COM
55 HTTP_USER_AGENT=Java/1.5.0
56 HTTP_PRAGMA=no-cache
57 HTTP_CACHE_CONTROL=no-cache
58 _CEE_RUNOPTS=ENVAR("_CEE_ENVFILE=//DD:ENV")
Number of environment variables is 59
Connection Protocol = HTTP
Server Name  = PTHISD1.AU.IBM.COM
Server Port  = 1726
FSCP = IBM-1047
NETCP = ISO8859-1
CGI_ISPCONF = /etc/SCLMDTIS
CGI_ISPWORK = /var/SCLMDTIS
Server PATH = /bin:.:/usr/sbin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin:
/usr/lpp/products/java142/J1.4/bin/
About to spawn task for ISPZTSO
Parameters passed to ISPZTSO - LISTALC
Return code from ISPZTSO is 0
About to open /tmp/USERNME.ID001965.ISPF.SYSTSPRT
<ISPINFO>
<TSO>
PDFTDEV.USERNME.LOAD
PDFTDEV.STG.LOAD
PDFTDEV.INT.LOAD
PDFTDEV.SVT.LOAD
USERNME.SCLMDW.LOAD
SCLMDW.V3TEST.LOAD
SCLMDW.V3BASE.LOAD
/tmp/USERNME.ID001965.ISPF.SYSTSPRT
```

```
</TSO>
</ISPINFO>
RC from doGet then writefile :ok
TOTAL Completion DATE/TIME is :Tue Mar 18 12:33:56 WST 2008
```

Customizing the gateway calling environment

If you plan to use IBM HTTP server powered by Apache to invoke the gateway, see "Customizing IBM HTTP server powered by Apache" on page 82 for information about customizing that environment for invoking the gateway.

If you plan to use IBM z/OS Explorer's Remote Systems Explorer (RSE) to invoke the gateway, see "Customizing Remote Systems Explorer" on page 90 for information about customizing that environment for invoking the gateway.

If you plan to invoke the gateway by means other than through IBM HTTP server powered by Apache or RSE (for example, from the OMVS shell), see "Customizing other environments" on page 90 for information about customizing the environment for invoking the gateway.

Customizing IBM HTTP server powered by Apache

If you plan to use IBM HTTP server powered by Apache to invoke the gateway, changes must be made to the HTTP configuration and environment files.

To invoke the gateway as installed, make these changes to the HTTP configuration file, httpd.conf:

- Include Alias and ScriptAlias directives to map the gateway URLs to their file system locations. The path specified in these directives must be the path where the gateway was installed. For example:

```
Alias        /ISPZIVP.html  /usr/lpp/ispf/bin/ISPZIVP.html
ScriptAlias  /ISPZIVP.cgi   /usr/lpp/ispf/bin/ISPZIVP.cgi
ScriptAlias  /ISPZXML       /usr/lpp/ispf/bin/ISPZXML
```

- If the gateway modules (see Table 3 on page 44) are not in the LINKLIST, include a STEPLIB directive to indicate the load library data sets that contain these modules. For example, if the libraries were DEV.USER.LOAD, DEV.STG.LOAD, and DEV.BASE.LOAD:

```
setenv STEPLIB DEV.USER.LOAD:DEV.STG.LOAD:DEV.BASE.LOAD
```

- Include LoadModule directives and a Directory directive to:
  - cause IBM HTTP server powered by Apache to prompt users to enter their user ID and password
  - invoke the gateway under the user's user ID
  - allow the gateway directory to serve content only to users who are authenticated using the System Authorization Facility (SAF) security product
  - Security headers that should be set unless otherwise mitigated are X-XSS-Protection, Strict-Transport-Security, X-Frame-Options, X-Content-Type-Options, Cache-Control, Pragma, and Content-Security-Policy.

  The path specified in the Directory directive must be the path where the gateway was installed. For example:

```
LoadModule auth_basic_module modules/mod_auth_basic.so
LoadModule authnz_saf_module modules/mod_authnz_saf.so
LoadModule headers_module modules/mod_headers.so
<Directory /usr/lpp/ispf/bin>
  AuthName "SAF auth ISPF Gateway"
  AuthType Basic
  AuthBasicProvider saf
  Require valid-user
  SAFRunAs %%CLIENT%%
  <IfModule mod_headers.c>
   Header set X-XSS-Protection "1; mode=block"
   Header always set Strict-Transport-Security "max-age=31536000 ; includeSubDomains"
   Header always set X-Frame-Options SAMEORIGIN
```

```
    Header always set X-Content-Type-Options nosniff
    Header set Cache-Control no-store
    Header set Pragma no-cache
    Header set Content-Security-Policy "default-src 'self';"
   </IfModule>
</Directory>
```

- Update the configuration to enable SSL for your TSO/ISPF gateway traffic.

  By default, an HTTPS connection is required to access the TSO/ISPF Gateway and unencrypted HTTP connections are rejected with an error message:

  ```
  ***ERROR: Connection is NOT using HTTPS. HTTPS is required.
  ```

  Although not recommended, you can override this default by adding the environment variable CGI_SECURECONN to your HTTP configuration and setting it to "FALSE".

  1. Use the gskkyman utility to create a key database and password stash file. See *z/OS Cryptographic Services System Secure Sockets Layer Programming* for information on the gskkyman utility.
  2. Store the key database file and password stash file in your HTTP ServerRoot directory.
  3. Include directives to enable SSL support:

     ```
     # Replace @@ServerRoot@@ with your ServerRoot directory name
     # Replace ihsserverkey.kdb with your database file name
     LoadModule ibm_ssl_module modules/mod_ibm_ssl.so
     Listen 443
     <VirtualHost *:443>
     SSLEnable
     </VirtualHost>
     KeyFile @@ServerRoot@@/ihsserverkey.kdb
     SSLDisable
     ```

  4. Include the LoadModule directive and rewrite rules to redirect your TSO/ISPF gateway traffic to use HTTPS:

     ```
     LoadModule rewrite_module modules/mod_rewrite.so
     RewriteEngine on
     RewriteCond %{SERVER_PORT} =80
     RewriteRule ^(.*) https://%{SERVER_NAME}%{REQUEST_URI} [R,L]
     ```

- To use the Interactive ISPF Gateway, include the following directives:
  - Set the CGI_CEATSO directive to the value TRUE:

    ```
    
    setenv CGI_CEATSO  TRUE
    ```

    When the CGI_CEATSO directive is not included or is set to a value other than TRUE, the Legacy ISPF Gateway is invoked.
  - Set the LIBPATH directive to include the directory where the CEA TSO/E address space services programs are located (/usr/lib). For example:

    ```
    
    setenv LIBPATH     /usr/lib
    ```

- To use the Legacy ISPF Gateway, include the following directives:
  - Set the CGI_ISPWORK directive to the path for the WORKAREA directory used by the gateway. For example, this directive specifies the default path for the WORKAREA directory:

    ```
    setenv CGI_ISPWORK /var/ispf
    ```

  - Set the CGI_ISPCONF directive to the path for the CONFIG directory where the ISPF configuration file ISPF.conf is stored. For example, this directive specifies the default path for the CONFIG directory:

    ```
    setenv CGI_ISPCONF /etc/ispf
    ```

- Set the CGI_ISPLOGLEVEL directive to the level of logging requested from the gateway. See "Customizing other environments" on page 90 for information on the available log levels. For example, this directive requests default logging:

  ```
  setenv CGI_ISPLOGLEVEL 0
  ```

To invoke the gateway as installed, make this change to the HTTP environment file, envvars:

- Update the path statement to include the path where the gateway was installed. If dot(.), indicating the current directory, is already specified then no update is required. In this example, the gateway path /usr/lpp/ispf/bin is added to the existing path statement:

  ```
  export PATH=$PATH:/etc/ihs9_rw/bin:/usr/lpp/ispf/bin
  ```

For additional information about configuring IBM HTTP server powered by Apache, review the manuals at IBM HTTP Server in IBM Documentation (www.ibm.com/docs/en/ibm-http-server).

Verifying the IBM HTTP server powered by Apache customization

This Installation Verification Process (IVP) applies if you have configured IBM HTTP server powered by Apache to invoke the gateway. IBM HTTP server powered by Apache must be running and the IVP Alias or ScriptAlias directives must be configured in the httpd.conf file for successful verification processing.

From a browser, type the location URL address:

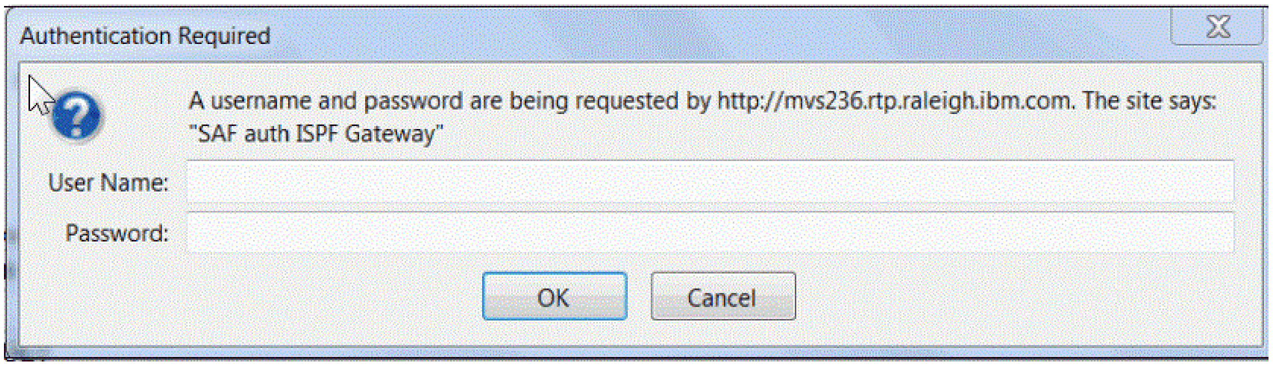
```
http://hostname:portnumber/ISPZIVP.html
```

where:

hostname
Is the name of the TCP/IP host on which IBM HTTP server powered by Apache is running.

port number
Is the port used in the httpd.conf file (the default port is 80).

If IBM HTTP server powered by Apache is running, you are prompted for a valid TSO user ID and password for the system on which the Web server is running:

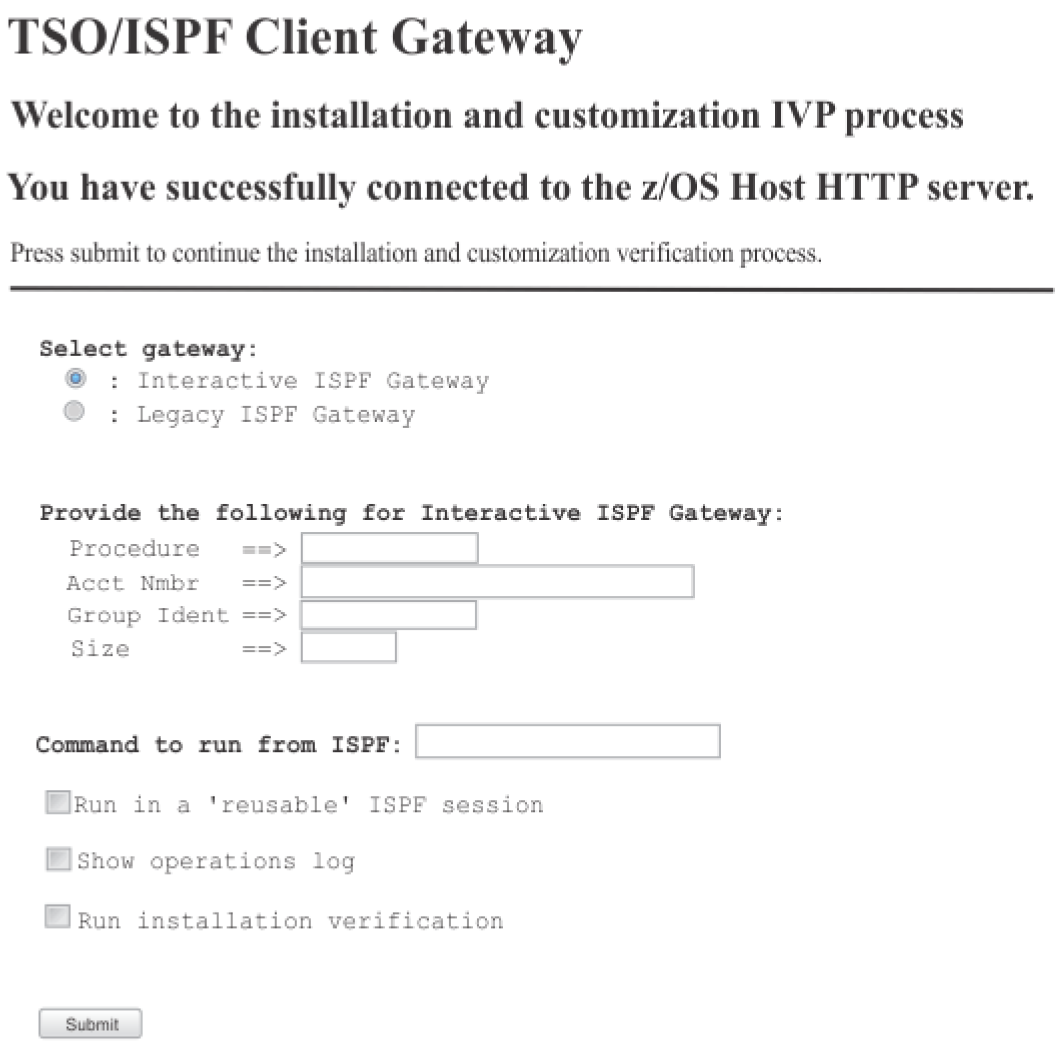

Authentication Required

A username and password are being requested by http://mvs236.rtp.raleigh.ibm.com. The site says: "SAF auth ISPF Gateway"

User Name:

Password:

After you enter your TSO user ID and password, the browser displays the HTTP welcome screen:


TSO/ISPF Client Gateway

Welcome to the installation and customization IVP process

You have successfully connected to the z/OS Host HTTP server.

Press submit to continue the installation and customization verification process.

Select gateway:

[✓ : Interactive ISPF Gateway]

: Legacy ISPF Gateway

Provide the following for Interactive ISPF Gateway:

Procedure ==>

Acct Nmbr ==>

Group Ident ==>

Size ==>

Command to run from ISPF:

☐ Run in a 'reusable' ISPF session

☐ Show operations log

☐ Run installation verification

Figure 45. HTTP welcome screen

If your connection fails, check to ensure that:

- IBM HTTP server powered by Apache has successfully initialized.
- The z/OS UNIX System Services file system mount point containing the TSO/ISPF Client Gateway installation is mounted.
- The hostname:portnumber value used in the URL is correct. If it appears to be correct, try pinging the hostname).
- There are no firewall restrictions.
- The Alias directive in the httpd.conf file is set correctly:

```
Alias  /ISPZIVP.html  /usr/lpp/ispf/bin/ISPZIVP.html
```

After you receive the welcome screen, continue with the IVP, which checks and validates your installation and customization process, by taking these steps:

To test using the Interactive ISPF Gateway:

1. Select the radio button for **Interactive ISPF Gateway**.
2. Enter the procedure name, account number, group ID, and region size to be used for the created TSO/E session.
3. Enter a TSO command (for example, TSO TIME or TSO LISTA) in the **Command to run from ISPF** field.
4. Select the **Show operations log** checkbox if you want the operations log included in the response. If the test does not complete successfully, the operations log might contain information to help you determine the reason.
5. Select the **Run installation verification** checkbox.
6. Click on the **Submit** push-button.

To test using the Legacy ISPF Gateway:

1. Select the radio button for **Legacy ISPF Gateway**.
2. Enter a TSO command (for example, TSO TIME or TSO LISTA) in the **Command to run from ISPF** field.
3. Select the **Show operations log** checkbox if you want the operations log included in the response. If the test does not complete successfully, the operations log might contain information to help you determine the reason.
4. Select the **Run installation verification** checkbox.
5. Click on the **Submit** push-button.

Figure 46 on page 87 shows an example of the expected validation responses from a test using the Interactive ISPF Gateway. In this example, TSO TIME was entered in the **Command to run from ISPF** field, the **Show operations log** checkbox was not checked, and the **Run installation verification** checkbox was checked. For this example, the CGI_ISPLOGLEVEL is also set to 8. If the CGI_ISPLOGLEVEL is below 8 or not set, PROCNAME, ACCTNUM, GROUPID, and REGIONSZ will not be displayed:

```
TSO/ISPF Client Gateway install verification for HTTP

Review IVP log messages from HOST below :

Using Interactive ISPF Gateway
  PROCNAME = ISPFPROC
  ACCTNUM  = SAMPACCT
  GROUPID  = DEFAULT
  REGIONSZ = 2000000

-------------------------------------------------------------
CHECK1 : ENVIRONMENT VARIABLES - key variables displayed below :

Server Name             = mvs236.rtp.raleigh.ibm.com
Server Port             = 80
Server PATH             = /usr/sbin:/bin:/usr/local/bin:.:/etc/ihs9_rw/bin:/usr/lpp/ispf/bin
STEPLIB                 =
LIBPATH                 = /usr/lib
CGI_ISPCONF             =
CGI_ISPWORK             =
CGI_CEATSO              = TRUE
Current login name      = USER1
Home directory          = /u/user1
Current working directory (HOMEDIR) = /usr/lpp/ispf/bin

-------------------------------------------------------------
CHECK2 : z/OS UNIX MODULES

Checking for Interactive ISPF Gateway modules in current directory
ISPZINT
ISPZINL
ISPZXML
ISPZXENV

-------------------------------------------------------------
CHECK3 : TSO/ISPF INITIALIZATION

( TSO/ISPF session will be initialized and run command : TSO TIME )
Command output -

   TIME-02:18:46 AM. CPU-00:00:00 SERVICE-55768 SESSION-00:00:02 JULY 25,2014
-------------------------------------------------------------

Host installation verification completed successfully

-------------------------------------------------------------
```

Figure 46. Example of expected validation response using the Interactive ISPF Gateway

Figure 47 on page 88 shows the expected validation responses from a test using the Legacy ISPF Gateway. In this example, TSO TIME was entered in the **Command to run from ISPF** field, the **Show operations log** checkbox was not checked, and the **Run installation verification** checkbox was checked:

```
TSO/ISPF Client Gateway install verification for HTTP

Review IVP log messages from HOST below :

Using Legacy ISPF Gateway

-------------------------------------------------------------
CHECK1 : ENVIRONMENT VARIABLES - key variables displayed below :

Server Name             = mvs236.rtp.raleigh.ibm.com
Server Port             = 80
Server PATH             = /usr/sbin:/bin:/usr/local/bin:.:/etc/ihs9_rw/bin:/usr/lpp/ispf/bin
STEPLIB                 =
LIBPATH                 =
CGI_ISPCONF             = /etc/ispf
CGI_ISPWORK             = /var/ispf
CGI_CEATSO              =
Current login name      = USER1
Home directory          = /u/user1
Current working directory (HOMEDIR) = /usr/lpp/ispf/bin

-------------------------------------------------------------
CHECK2 : z/OS UNIX MODULES

Checking for Legacy ISPF Gateway modules in current directory
ISPZINT
ISPZINO
ISPZTSO
ISPZXML
ISPZXENV

-------------------------------------------------------------
CHECK3 : TSO/ISPF INITIALIZATION

( TSO/ISPF session will be initialized and run command : TSO TIME )
Command output -

   IKJ56650I TIME-02:43:49 AM. CPU-00:00:00 SERVICE-68102 SESSION-00:00:01 JULY 25,2014
-------------------------------------------------------------

Host installation verification completed successfully

-------------------------------------------------------------
```

Figure 47. Example of expected validation response using the Legacy ISPF Gateway

Java sample to invoke the gateway through a HTTP server

The Java™ program shown here, which is supplied in member ISPZXJAV in the ISPF samples data set ISP.SISPSAMP, shows an example of invoking the gateway through a HTTP server and passing XML to request running the TSO PROFILE command.

Before using this sample, you must take the actions described on the lines indicated with //**.

```
package com.ibm.ispfcall;

import java.io.*;
import java.net.*;
import java.util.*;

//*******************************************************************//
// Before using this sample, you must take the actions described on  //
// the lines indicated with //**.                                    //
//*******************************************************************//
public class XmlIspf {

 public static void main(String[] args) {

  try {

   //** Create file C:\logon.txt, containing your user ID and password
   //** in the format USERID:PASSWORD.
   BufferedReader in = new BufferedReader(new FileReader("C:\\logon.txt"));
   String logon = in.readLine();
   in.close();
   String encodedlogon =
       Base64.getEncoder().encodeToString(logon.getBytes("utf-8"));
```

```
    Date d = new Date() ;
    System.out.println("START Transfer DATE/TIME is :" + d.toString()  );

    // URL details for CGI POST
    //** Replace SITE.COM with name of TCP/IP host where HTTP server
    //** is running.
    //** Replace 1234 with port number on which HTTP server is listening.
    URL url = new URL("http", "SITE.COM", 1234, "/ISPZXML");
    HttpURLConnection con = (HttpURLConnection) url.openConnection();

    con.setUseCaches(false);
    con.setDoInput(true);
    con.setDoOutput(true);
    con.setRequestMethod("POST");
    con.setRequestProperty( "Authorization", "Basic "
                           + encodedlogon );

    System.out.println("At url openConnection.. ");

    // POST CGI routines
    doPut(url, con, "POST", "");
    doGet(url, con);

    Date c = new Date() ;
    System.out.println("TOTAL Completion DATE/TIME is :" + c.toString()  );

   }
   catch (IOException exception)
   {
    System.out.println("Error: " + exception);
   }
  }

  public static void doPut(URL url, HttpURLConnection con, String ISPFFUNC,
                           String fileIn) throws IOException
  {
   if (ISPFFUNC.equals("POST"))
   {
    PrintWriter out = new PrintWriter(con.getOutputStream());
    // Below is a sample inline XML input for an ISPF service request
    // This could alternatively be read from an external file
    out.println( "<?xml version=\"1.0\"?>"           );
    out.println( "<ISPF-INPUT>"            );
    out.println( "xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"" );
    out.println( "xsi:noNamespaceSchemaLocation=\"ispf.xsd\">"    );
    out.println( "<SERVICE-REQUEST>"          );

    //** Uncomment these lines if you want to invoke the
    //** Interactive ISPF Gateway. Otherwise, see below.
    //** Replace YOURPROC, YOURACCT, YOURGRP, AND YOURSIZE
    //** with your TSO logon values.
    // out.println( "<loglevel>3</loglevel>"         );
    // out.println( "<request>NEWTSO</request>"         );
    // out.println( "<procname>YOURPROC</procname>"         );
    // out.println( "<acctnum>YOURACCT</acctnum>"         );
    // out.println( "<groupid>YOURGRP</groupid>"         );
    // out.println( "<regionsz>YOURSIZE</regionsz>"         );
    // out.println( "<cmdresp>TSO PROFILE</cmdresp>"    );
    // out.println( "<request>LOGOFF</request>"         );

    //** Uncomment these lines to invoke the Legacy ISPF Gateway.
    //** Otherwise, see above.
    // out.println( "<service>ISPF</service>" );
    // out.println( "<session>NONE</session>" );
    // out.println( "<command>TSO PROFILE</command>" );
    // out.println( "<ispprof>USERID.ISPPROF</ispprof>" );

    out.println( "</SERVICE-REQUEST>"          );
    out.println( "</ISPF-INPUT>"           );
    out.flush();
    out.close();
   }
  }

  public static void doGet(URL url, HttpURLConnection con) throws IOException
  {
   BufferedReader in;
   try
   {
    System.out.println("About to accept response from Server");
    in = new BufferedReader(new InputStreamReader(con.getInputStream()));
    System.out.println("Response from Server received");
   }
   catch (FileNotFoundException exception)
   {
    InputStream err = ((HttpURLConnection)con).getErrorStream();
    if (err == null) throw exception ;
    in = new BufferedReader(new InputStreamReader(err));
   }

   String line;
   while ((line = in.readLine()) != null)
    System.out.println(line);

   in.close();
  }
 }
```

Customizing Remote Systems Explorer

If you plan to invoke the gateway using the Remote Systems Explorer (RSE) function from IBM z/OS Explorer, customize the RSE environment. For information about installing and configuring RSE, see *Basic customization* in IBM Developer for z Systems (www.ibm.com/docs/en/adfz/developer-for-zos).

Customizing other environments

If you plan to invoke the gateway by means other than through the HTTP server or RSE (for example, from the OMVS shell), configuration is required to ensure values are set for environment variables used by the gateway. This requires the environment set-up routine, ISPZXENV, to be modified to set values for the environment variables shown in Table 7 on page 90. The footnotes indicate which environment variables are required for the Interactive ISPF Gateway and which environment variables are required for the Legacy ISPF Gateway.

Table 7. Environment variables to be set by the environment set-up routine, ISPZXENV

Environment variable	**Description**
STEPLIB	Specifies the STEPLIB data sets from where MVS load modules are run. This environment variable must be set to the names of the load library data sets containing the gateway modules (see Table 3 on page 44) if these modules are not in the LINKLIST. (1,2)
CGI_ISPCONF	Specifies the CONFIG directory path where the configuration files are stored. (2)
CGI_ISPWORK	Specifies the WORKAREA directory path that is used for temporary files. (2)
CGI_ISPPREF	Specifies the temporary data set prefix. (2)
CGI_CEATSO	Indicates whether the Legacy ISPF Gateway or the Interactive ISPF Gateway is invoked. (1)
CGI_PING	Control whether or not to PING when using REUSE. (1)
CGI_ISPDEBUG_MIN	Controls whether the list of defined environment variables is written to STDOUT by the Legacy ISPF Gateway. **Note:** CGI_ISPLOGLEVEL is recommended as a replacement for this variable.
CGI_ISPLOGLEVEL	Specifies the level of logging performed by the ISPF Gateway (both Legacy and Interactive).

Table 7. Environment variables to be set by the environment set-up routine, ISPZXENV (continued)

Environment variable	Description
CGI_ISPDEBUG	Specifies whether debug information is included in the log information produced by the ISPF Gateway. **Note:** CGI_ISPLOGLEVEL is recommended as a replacement for this variable.

1 – Variable is required for the Interactive ISPF Gateway.
2 – Variable is required for the Legacy ISPF Gateway.

A sample of the REXX routine ISPZXENV is provided in member ISPZXENV in the ISPF installation directory (/usr/lpp/ispf/bin). These lines in ISPZXENV may require modification:

`STEPLIB = 'ISP.SISPLPA:ISP.SISPLOAD'`
The `STEPLIB` variable should specify the data sets containing the load modules for the TSO/ISPF client gateway (see Table 3 on page 44). If these load modules reside in the LINKLIST, then the `STEPLIB` variable should be set to a null value (that is, '').

`CGI_ISPCONF = '/etc/ispf'`
The `CGI_ISPCONF` variable specifies the HOME directory path where the configuration files reside for the gateway. For more information on the CONFIG directory, see "Installing the Legacy ISPF Gateway" on page 69. This variable is not used by the Interactive ISPF Gateway.

`CGI_ISPWORK = '/var/ispf'`
The `CGI_ISPWORK` variable specifies the pathname of the directory where work files for the gateway are stored. The value must be no longer than 80 characters. For more information on the work directory, see "Installing the Legacy ISPF Gateway" on page 69. This variable is not used by the Interactive ISPF Gateway.

`CGI_ISPPREF`
The `CGI_ISPPREF` variable specifies the prefix to be used to allocate cataloged z/OS work files. It supports the use of system variables &SYSUID, &SYSPREF and &SYSNAME. These system variables are described in "Using the native API of the Legacy ISPF Gateway" on page 77. If not specified, &SYSPREF..ISPF.VCMISPF is used. The prefix must resolve to a value of 35 characters or less. This variable is not used by the Interactive ISPF Gateway.

CGI_CEATSO
The CGI_CEATSO variable specifies which version of the gateway is used. If the variable is defined and is set to a value of TRUE, the Interactive ISPF Gateway is used. If the variable is not defined or is set to a value other than TRUE, the Legacy ISPF Gateway is used.

CGI_PING
The CGI_PING variable is directly related to the use of REUSE request. REUSE does NOT reset the timeout timer which might cause the session to timeout prematurely. Setting CGI_PING=TRUE will result in a PING request to be sent along with REUSE so that the timer will be reset. As a result, the session will continue to exist.

CGI_ISPDEBUG_MIN
The CGI_ISPDEBUG_MIN variable controls whether the list of defined environment variables is written to STDOUT by the Legacy ISPF Gateway. If the variable is defined and is set to a value of TRUE, the list of defined environment variables is not written to STDOUT. This can result in improved performance for a client that has many environment variables defined.

Note: CGI_ISPLOGLEVEL is recommended as a replacement for this variable.

CGI_ISPLOGLEVEL
The CGI_ISPLOGLEVEL variable specifies the level of logging performed by the ISPF Gateway (both Legacy and Interactive). Specify a non-zero value to activate more than the default logging. Add logging level values together to request multiple types of logging information.

1
: Log debug information. Provides information about the request processing that can be useful when debugging problems. Equivalent to specifying CGI_ISPDEBUG='TRUE'.

2
: Controls whether the list of defined environment variables is written to STDOUT. Include this logging level if you do not want the list of environment variables written to STDOUT. This can result in improved performance for a client that has many environment variables defined. Equivalent to specifying CGI_ISPDEBUG_MIN='TRUE'.

4
: Include time stamps. Causes a time stamp value to be included in some lines of log information.

8
: Include additional log information when the gateway is invoked via HTTP. This log level should be included only when debugging gateway problems in the HTTP environment.

CGI_ISPDEBUG
: The CGI_ISPDEBUG variable specifies whether debug information is included in the log information produced by the Legacy ISPF Gateway. Provides information about the request processing that can be useful when debugging problems. Specify TRUE to include debug information in the log information.

  Note: CGI_ISPLOGLEVEL is recommended as a replacement for this variable.

Place the amended version of ISPZXENV in a directory referenced before /usr/lpp/ispf/bin in your PATH variable, otherwise you may lose your changes when system maintenance is applied.

Chapter 4. Improving ISPF performance

To improve ISPF performance consider using these methods, which are described in detail here:

- Use virtual I/O (VIO) (see "Use virtual I/O" on page 93).
- Remove or tailor edit functions (see "Remove or tailor Edit functions" on page 93).
- Remove action bars from ISPF panels (see "Remove action bars from ISPF panels" on page 94).
- Remove scrollable areas from ISPF panels (see "Remove scrollable areas from ISPF panels" on page 95).
- Preprocess ISPF panels (see "Preprocess all ISPF panels" on page 95).
- Add load modules to LPALST (see "Add load modules to the pageable link-pack area (LPA)" on page 96).
- Allocate execution data sets (see "Allocate execution data sets" on page 99).
- Disable generic searches (see "Disable generic high-level qualifiers" on page 99).
- Preallocate ISPF temporary data sets to VIO (see "Preallocate ISPF temporary data sets to VIO" on page 101).

Here are some other suggestions, which require no further description:

- Use Browse instead of View for large files
- Use an action bar to call other functions from Edit or Browse if you intend to return to the Edit or Browse session
- Use the IBM REXX compiler to compile ISPF dialogs written in REXX.

Use virtual I/O

Unit name SYSALLDA is the default specified in the ISPF Configuration table (keyword name PDF_DEFAULT_UNIT). When defining SYSALLDA, or another unit if you modify the configuration table, use the VIO=YES option on the UNITNAME macro. PDF allocates its temporary data sets using VIO if it is available. See Chapter 2, "The ISPF Configuration Table," on page 9 for information about modifying the configuration table.

Remove or tailor Edit functions

Using the ISPF configuration table, you can tailor or remove functions that are likely to increase resource use. These include:

- Edit recovery data set attributes (RECOVERY ON)
- Edit in-storage change tracking (SETUNDO STORAGE)
- Edit enhanced highlighting (HILITE)

Change the block size of the edit recovery data set

The performance of the edit recovery function can be enhanced by changing the block size of the edit recovery data set. This change is to the keyword EDIT_RECOVERY_BLOCK_SIZE in the ISPF configuration table. Similar fields exist in the ISPF configuration table for other less frequently used data sets. See Chapter 2, "The ISPF Configuration Table," on page 9 for more information about the ISPF configuration table.

Disable SETUNDO STORAGE

The SETUNDO STORAGE function of the editor lets users undo changes made within edit sessions by keeping a record of those changes in storage. This provides better performance for individual users but can have a negative impact on overall system performance on systems where the editor is heavily used.

The function of saving the record of changes in storage can be completely disabled by setting the keyword UNDO_STORAGE_SIZE to zero in the ISPF configuration table. The UNDO function will still be available to users who are running with RECOVERY ON.

By default, existing user edit profiles that have never had SETUNDO explicitly changed, will have SETUNDO STORAGE enabled. Newly created profiles will have SETUNDO STORAGE turned off. If you want to force most of the existing, heavily used profiles to specify SETUNDO OFF, disable the function by changing the configuration table keyword UNDO_STORAGE_SIZE to zero and run the system like that for several days or weeks. When the function is disabled, edit profiles are changed to reflect SETUNDO OFF when they are used.

After several days or weeks, you might want to enable the function by again changing the configuration table. See Chapter 2, “The ISPF Configuration Table,” on page 9 for more information on the ISPF configuration table.

Note: If you normally place a copy of the edit profile table (normally ISREDIT) containing a ZDEFAULT profile in your ISPTLIB concatenation, you should ensure that it reflects the SETUNDO setting of your choice, because this will override the programmed defaults for newly created edit profiles.

Disable Edit extended highlighting

Although edit extended highlighting provides powerful features (such as language sensitive color), it might require an unacceptable amount of CPU time system wide. To disable the feature, change the value of the keyword DEFAULT_EDIT_DISPLAY in the ISPF configuration table to a value of 0 or 1.

To completely disable extended highlighting, even for applications that use their own edit panels, set the value of keyword ALLOW_EDIT_HIGHLIGHTING in the ISPF configuration table to NO. See Chapter 2, “The ISPF Configuration Table,” on page 9 for more information about the ISPF configuration table.

Remove action bars from ISPF panels

Some of the panels that are provided with the Dialog Tag Language (DTL) source can be recompiled to remove the action bars. This can result in some performance benefits. To recompile the DTL source:

1. Invoke ISPDTLC with the NOACTBAR option by specifying:

   ```
   ISPDTLC (NOACTBAR
   ```

   or by de-selecting the option "Create panels with Action bars" on the invocation panel.
2. Modify the convert EXEC to include the NOACTBAR parameter:

   ```
   'ISPDTLC ISPFAB (DISK KEYLAPPL=ISR MSGSUPP CUAATTR CUASUPP PLEB MCOMMENT
       NOSTATS NOACTBAR PROFILE=your.gml.dataset(profile)' ;
   ```

3. If you are recompiling DBCS versions of the panels, you must add the DBCS option to the command syntax. Japanese panels require the KANA option as well.

   ```
   'ISPDTLC ISPFALL (DISK KEYLAPPL=ISR MSGSUPP CUAATTR CUASUPP PLEB MCOMMENT
        JAPANESE DBCS KANA NOSTATS NOACTBAR PROFILE=your.gml.dataset(profile)' ;
   ```

For more information about invoking ISPDTLC, see “Invoking the ISPDTLC conversion utility” on page 38.

Remove scrollable areas from ISPF panels

Some of the panels that are provided with the Dialog Tag Language (DTL) source can be recompiled to remove scrollable areas (where the resulting panel fits within a standard 24 line screen). This can result in some performance benefits. To recompile the DTL source:

1. Invoke ISPDTLC with the MERGESAREA option by specifying:

   ```
   ISPDTLC (MERGESAREA
   ```

   or by selecting the option "Combine scrollable areas into panel body" on the invocation panel.
2. Modify the convert EXEC to include the MERGESAREA parameter:

   ```
   'ISPDTLC ISPFALL (DISK KEYLAPPL=ISR MSGSUPP CUAATTR CUASUPP PLEB MCOMMENT
       NOSTATS MERGESAREA PROFILE=your.gml.dataset(profile)' ;
   ```

3. If you are recompiling DBCS versions of the panels, you must add the DBCS option to the command syntax. Japanese panels require the KANA option as well.

   ```
   'ISPDTLC ISPFALL (DISK KEYLAPPL=ISR MSGSUPP CUAATTR CUASUPP PLEB MCOMMENT
       JAPANESE DBCS KANA NOSTATS MERGESAREA PROFILE=your.gml.dataset(profile)' ;
   ```

Note: The NOACTBAR AND MERGESAREA options can be combined to provide both results.

For more information about invoking ISPDTLC, see “Invoking the ISPDTLC conversion utility” on page 38.

Preprocess all ISPF panels

Preprocessed panels are displayed much faster than if they are not preprocessed. However, panels that must be sized at run time (dynamic panels) cannot be preprocessed. After preprocessing the ISPF panels, copy all panels into your execution data set, specifying NOREPLACE. This will copy only those members that could not be preprocessed.

Note: The preprocessed versions of the ISPF panels are not updated by PTFs from IBM. If a panel is updated by a PTF, you must preprocess it into your execution data set again.

Preprocessed panel utility

The preprocessed panel utility is an ISPF dialog called ISPPREP. This utility converts panel definitions to a form ISPF can display more quickly. The converted panels can be defined to ISPF in place of normal panel definitions. This improves ISPF's performance.

When you install ISPF, consider creating a panel library to contain the preprocessed versions of the ISPF panels. Be aware that a preprocessed panel is in an encoded form and cannot be changed through the normal edit procedure. For a general discussion of ISPPREP and its use, refer to the *z/OS ISPF Dialog Developer's Guide and Reference*.

This example shows how to create preprocessed versions of the ISPF panels contained in the data set ISP.SISPPENU. The example assumes this data set is cataloged. However, this is not necessary and both the panel input and panel output data sets can be uncataloged.

Before continuing, you must allocate the data set that will contain the preprocessed panels. For this example, catalog the data set and name it ISP.PREPPLIB. Allocate the data set with the same record format, logical record length, and block size of the ISPF panel data set, ISP.SISPPENU. However, the entire ISPF panel data set in the preprocessed format will take up about 25% less space than the original data set, so allocate the new data set accordingly.

To convert the ISPF panel library, issue the ISPSTART command as follows:

```
ISPSTART PGM(ISPPREP) PARM(INPAN('ISP.SISPPENU'),
                           OUTPAN('ISP.PREPPLIB'))
```

Note: This command creates all members in the output library with ISPF statistics included. Be sure that you have enough directory blocks to contain the members and their statistics, or use the NOSTAT parameter for ISPPREP. See the *z/OS ISPF Dialog Developer's Guide and Reference* for more information about ISPPREP.

The panel output data set name in the previous example is a suggested naming convention. Regardless of how you name the data set, be sure you modify your TSO LOGON procedure or the CLIST that allocates the ISPF data sets to allocate the new panel library. See the *z/OS ISPF User's Guide Vol I* for more information about allocating ISPF libraries.

The previous example shows the use of ISPPREP in foreground batch mode. You can also run ISPPREP as a background job, as an interactive dialog by selecting option 2 on the ISPF Primary Option Menu, or by entering the ISPPREP command on any Command line.

These restrictions apply to the panels you can convert to preprocessed panels. You cannot convert any panel that contains these items in the panel definition:

- A dialog variable specified with the WIDTH keyword in the)BODY header statement of a panel.
- A dialog variable that defines a model line in a table display panel definition.
- A dynamic or graphic area that has EXTEND(ON) specified for the attribute character.
- An INEXT section.

If ISPPREP is passed a panel definition that does not meet these restrictions, a message is issued to the ISPF log data set that specifies the name of the panel that violated the restrictions. In this case, the panel definition is not converted to the preprocessed format. However, you can copy the original panel definitions into the new panel library to keep your panels grouped accordingly. In any case, be sure to check the ISPF log data set after invoking ISPPREP. The messages in the log data set help you identify any problems ISPPREP encounters during the conversion. For more information, refer to the *z/OS ISPF Dialog Developer's Guide and Reference*.

Add load modules to the pageable link-pack area (LPA)

Once you have installed and verified ISPF, you can enhance its performance by adding the LPA-eligible load modules (in the SISPLPA library) to the LPA list in an LPALST*xx* member of PARMLIB. Add those load modules not eligible for LPA (that is, those in the SISPLOAD library that are not re-entrant) to the link list in an LNKLST*xx* member of PARMLIB. For information about adding data sets to the Link and LPA lists, see *z/OS MVS Initialization and Tuning Reference*. You can then remove these data sets from the STEPLIB in your TSO LOGON procedure. After adding SISPLPA to LPALST and SISPLOAD to LNKLST, specify CLPA as an initial program load (IPL) parameter to force the SISPLPA modules into the link pack area and to have SISPLOAD added as a system link library.

ISPF performance directly relates to the number of load modules that reside in the pageable LPA because these modules are not loaded into the user's private storage when they are called. For optimum performance, all of the eligible ISPF licensed program load modules should be in the LPA. However, to load all eligible modules to the LPA, your system would need a very large LPA.

Any module that is re-entrant (that is, has the RN attribute) is eligible for the LPA.

No modules are required in the LPA. Table 8 on page 97 lists the load modules recommended for inclusion in the LPA.

To conserve LPA space:

- Move some of the infrequently used modules to a system-link library. Keep the modules you most frequently use in the LPA.
- If you are not using SCLM, move all load modules that begin with FLM from the SISPLPA library to the SISPLOAD library so that they are in a system link library instead of the Link-Pack area.

Moving load modules

To move load modules from one library to another, use SMP/E ++MOVE statements. SMP/E moves the load modules and all associated aliases and updates the SMP CSI. This ensures that the load modules are available to SMP/E for subsequent maintenance.

Sample usermods that contain SMP/E ++MOVE statements are included with ISPF to help you customize your system if you decide to put only part of the SISPLPA load modules into LPA.

Usermod ISPMOVE1 contains a **sample** ++MOVE statement to move a load module from SISPLPA to SISPLOAD. Usermod ISPMOVE2 contains a **sample** ++MOVE statement to move a load module from SISPLOAD to SISPLPA.

Note:

1. These sample usermods are not complete. They do not contain ++MOVE statements for the possible load modules that may be moved. They contain a sample ++MOVE statement to show you the format required. Before using them, you must edit them for applicability to your environment. Provide statements for those load modules that you want to move from one library to another. See the instructions in the samples for more information about changes you need to make.
2. Be aware that some PTFs may change the JCLIN for ISPF load modules. If the JCLIN is changed for a load module that you have moved, after application of the PTF, the load module may reside back in the library you moved it from. You should carefully examine the output from the application of any PTF, and also the information in your SMP/E zone, and if necessary re-apply the usermod.

If you move the modules without using SMP/E ++MOVE statements, consider:

- You can use the ISPF Move and Copy utility (option 3.3) to move load modules between the SISPLOAD and SISPLPA data sets. If you do this, you must also update the load module SYSLIB subentries in the SMP CSI to reflect the move, otherwise the modules will not be available to SMP/E for maintenance.
- Many of the load modules have aliases. You can determine the alias of any load module by viewing the data set list (from ISPF option 3.4) of SISPLPA and SISPLOAD. When you use ISPF option 3.3 to move modules that have aliases, make sure that 'Process member aliases' is selected in the panel where you specify the "To" data set.

Five SCLM load modules are shipped as ++PROGRAMs. They are: FLMCQBLD, FLMCQCNT, FLMCQINT, FLMCQPRM, and FLMCQSTQ. You cannot use the usermods to move these load modules as SMP/E does not allow for moving a ++PROGRAM by a ++MOVE statement.

ISPF load module descriptions

This section contains information describing ISPF load modules.

Table 8 on page 97 lists the frequently used ISPF load modules that should be in the LPA and a brief description of each.

Any load module distributed in SISPLOAD that does not have the re-entrant (RN) attribute is not eligible for the LPA and should not be moved to SISPLPA.

Load modules shipped with the attribute RMODE(24) reside below the 16M line.

Note: Load module ISPLINK is shipped with the attribute RMODE(24) to provide compatibility with dialogs that are AMODE(24) and use a LOAD and CALL interface to ISPLINK. However, programs that will reside above the 16MB line (RMODE(ANY)) and include ISPLINK in their load module can override the RMODE(24) at link-edit time. ISPLINK code can reside and execute above the 16MB line.

Table 8. Minimum recommended load modules for the LPA

Load Module	Description
FLM$CPI	SCLM
FLMB	SCLM

Table 8. Minimum recommended load modules for the LPA (continued)

Load Module	**Description**
FLMCPCS	SCLM
FLMDDL	SCLM
FLMIO24	SCLM
FLMP	SCLM
FLMRTLIB	SCLM
FLMS$LNK	SCLM
FLMS7C	SCLM
ISPICP	ISPF driver (alias name: ISPSTART)
ISPISM	Settings processor
ISPKEY	KEYLIST command processor
ISPKLU	KEYLIST Utility
ISPLLP	Log/List processor
ISPMAIN	ISPF controller
ISPNL*xxx*	where *xxx* designates the languages you are using
ISPOMI	MSGID processor
ISPOPF	PFSHOW processor
ISPOPI	Panel ID processor
ISPOPT	Options processor
ISPSUBS	Common subroutines (alias name: ISPCIP)
ISPSUBX	Common subroutines
ISPTASK	Dialog driver
ISPTCM	TSO command table
ISPTUTOR	Tutorial processor (option T)
ISPYL*xxx*	where *xxx* designates the languages you are using
ISP32*nnn*	where *nnn* designates the terminal types in use
ISRBRO	Main Browse module
ISREDIT	Main Edit module
ISRNLxxx	where xxx designates the languages you are using
ISRPLEX	Edit extended highlight module
ISRPX	Edit extended highlight customization module
ISRRCL	Action bar router utility
ISRSEPRM	More SuperC
ISRSUBS	Main PDF subroutine module
ISRSUBX	Main below 16M PDF subroutine module
ISRSUPC	Superc
ISRUDA	PDF Utilities

Table 8. Minimum recommended load modules for the LPA (continued)

Load Module	Description
ISRUMC	PDF Move Copy
ISR32*nnn*	where *nnn* designates the terminal types in use

ISRSUPC is shipped RMODE(24) and is large. Installations that do not use Edit Compare or Superc functions frequently could consider moving this module from LPA to free up below the line storage.

Include FLMS7C to enhance the performance of SCLM batch jobs and tools or dialogs that use the FLMCMD interface to SCLM services. Include FLMS$LNK for tools or dialogs using the FLMLNK interface to SCLM services. Include FLMLPCBL if you are using COBOL source; FLMLPFRT if you are using FORTRAN source; or FLMLPGEN if you are using Assembler, REXX, PLI, or text source.

Allocate execution data sets

Consider these points when creating data sets that are used at execution time and when setting up ISPF applications.

- System-Determined Block size (SDB) should be used for the ISPxLIB data sets. To allocate a data set with SDB, code `DCB=BLKSIZE=0` in JCL, or specify a block size of zero in ISPF Option 3.2 when allocating the data set. The exception to this is ISPLLIB data sets with a record format of U. These should be allocated using a block size consistent with your system conventions.
- If you do not use a system-determined block size, use a block size for the ISPxLIB data sets that is a half of a 3380/3390 track. Use a block size of 32760 for load module data sets including those allocated to ISPLLIB, if any.
- Use cached controllers for the ISPxLIB data sets.
- Use PDSEs for the ISPxLIB data sets. Pay careful attention to the parameters with which the PDSEs are defined. The direct MSR (millisecond response time) parameter of the storage class affects PDSE performance. Low values of this parameter will improve PDSE performance at the expense of increased storage use.
- Make sure the ISPxLIB data sets are on lightly used volumes. You can spread them out over multiple volumes.
- Minimize the number of ISPLLIB data sets:
  - If ISPF modules are in the LPA and LNKLIST, do not include them in ISPLLIB. It is not necessary.
  - Put commonly used ISPF applications in the LPA and LNKLIST.
  - Use LIBDEFs where possible for infrequently used applications.

Disable generic high-level qualifiers

ISPF allows a generic high-level qualifier when using option 3.4 to print or display a list of data set names. A generic high-level qualifier is one that contains a wildcard character. You can restrict the search limits (and improve performance) by disabling the use of wildcard characters in the first qualifier. You can do this in two ways:

- Through the DISALLOW_WILDCARDS_IN_HLQ setting in the ISPF Configuration table.
- By writing an exit that sets a return code to prevent the list from being generated when it finds a wildcard character. This method allows more flexibility, for example if you wish to allow a wildcard character at the end of a high-level qualifier but not at the start.

For more information about using the data set list filter exit, see "Data set list filter exit" on page 191.

Here is an example of a data set list exit that disables wildcards. A copy of this exit is included in your SISPSAMP data set in member ISRNOGEN. Note that this example as written is nonreentrant, so it should not be put in the LPA. If nonreentrant exits are put in the LPA, abends can occur.

```
   TITLE ' ISRNOGEN:   PROC (PARM1,PARM2,PARM3,PARM4);'
* This program is an example of a data set list exit.  It checks
* for a generic high-level qualifier and sets the return code
* to prevent the list from being generated when there is a generic
* high-level qualifier.
ISRNOGEN CSECT ,
ISRNOGEN AMODE  31
ISRNOGEN RMODE  ANY
@PROLOG  STM   @14,@12,12(@13)
         BALR  @12,0
@PSTART  DS    0H
         USING @PSTART,@12
         MVC   PARMADDR(16),0(@01)
* If the first parameter is one, check the data set name for
* a generic character (% or *)
CHECK1   L     @04,PARMADDR
         CLC   PARM1(4,@04),ONE
         BNE   EXITRTN
* Check current Dsname
         L     @05,PARMADDR+4
         LR    @01,@05
         XR    @02,@02
* Translate and test for *, %, blank, or period
         TRT   DSNAME(44,@05),TABLE
         SR    @01,@05
         AR    @05,@01
* Check the character that stopped the translate and test for an
* asterisk or a percent sign.  If either of these is the first
* character found, the high-level qualifier is generic.
         CLC   DSNAME(1,@05),ASTERISK
         BE    ERROR
         CLC   DSNAME(1,@05),PERCENT
         BE    ERROR
* If the high-level qualifier is not an asterisk or percent sign,
* set the return code to allow the list to be displayed.
NAMEOK   SLR   @06,@06
         ST    @06,EXITRC
EXITRTN  DS    0H
         L     @15,EXITRC
         L     @14,12(,@13)
         LM    @00,@12,20(@13)
         BR    @14
* If the high-level qualifier is generic, set the return code to 8
* to prevent searching all catalogs.
ERROR    MVC   EXITRC(4),EIGHT
         B     EXITRTN
* Data for checkname                /*                                */
@DATA    DS    0H
         DS    0F
PARMADDR DS    4F
         DS    0F
ONE      DC    F'1'
TWO      DC    F'2'
FOUR     DC    F'4'
EIGHT    DC    F'8'
         DS    0D
EXITRC   DS    F
ASTERISK DC    C'*'
PERCENT  DC    C'%'
```

```
* Translate and test table with blank, period, asterisk, and
* percent signs
TABLE    DC    64X'00'
         DC    X'40'
         DC    10X'00'
         DC    X'4B'
         DC    16X'00'
         DC    X'5C'
         DC    15X'00'
         DC    X'6C'
         DC    147X'00'
@00      EQU   00                    Equates for registers 0-15
@01      EQU   01
@02      EQU   02
@03      EQU   03
```

```
@04      EQU   04
@05      EQU   05
@06      EQU   06
@07      EQU   07
@08      EQU   08
@09      EQU   09
@10      EQU   10
@11      EQU   11
@12      EQU   12
@13      EQU   13
@14      EQU   14
@15      EQU   15
VOLUME   EQU   0
DSNAME   EQU   0
EXIT1    EQU   0
LEVEL    EQU   0
PARM1    EQU   0
PARM2    EQU   0
PARM3    EQU   0
PARM4    EQU   0
         DS    0D
@ENDDATA EQU   *
         END   ISRNOGEN
```

Preallocate ISPF temporary data sets to VIO

ISPF uses temporary data sets to generate JCL or utility control statements or to generate listings. To preallocate these data sets to VIO, include the DD statements in Figure 48 on page 102 in the TSO LOGON procedure.

Preallocation of these data sets to VIO is not mandatory; ISPF automatically allocates them to real data sets if required. However, preallocation is recommended, because it reduces overhead and eliminates potential problems from insufficient space.

```
//ISPCTL0  DD DISP=NEW,UNIT=VIO,SPACE=(CYL,(1,1)),
//            DCB=(LRECL=80,BLKSIZE=800,RECFM=FB)

//ISPCTL1  DD DISP=NEW,UNIT=VIO,SPACE=(CYL,(1,1)),
//            DCB=(LRECL=80,BLKSIZE=800,RECFM=FB)

⋮
//ISPCTLW  DD DISP=NEW,UNIT=VIO,SPACE=(CYL,(1,1)),
//            DCB=(LRECL=80,BLKSIZE=800,RECFM=FB)


//* In this section of JCL, there is one DD for each screen
//* defined, based on the value of keyword MAXIMUM_NUMBER_OF_
//*  SPLIT_SCREENS in the configuration table.
//* The DD name is in the form ISPCTLx, where x can be
//* 1-9, A-W.  For example, if the keyword value = 8, only
//* ISPCTL1 to ISPCTL8 need to be coded.
//* ISPCTL0 is a special case, used only by Edit for the Submit
//* command.

//ISPWRK1  DD DISP=NEW,UNIT=VIO,SPACE=(CYL,(1,1)),
//            DCB=(LRECL=256,BLKSIZE=2560,RECFM=FB)

//ISPWRK2  DD DISP=NEW,UNIT=VIO,SPACE=(CYL,(1,1)),
//            DCB=(LRECL=256,BLKSIZE=2560,RECFM=FB)

⋮
//ISPWRKW  DD DISP=NEW,UNIT=VIO,SPACE=(CYL,(1,1)),
//            DCB=(LRECL=256,BLKSIZE=2560,RECFM=FB)

//* In this section of JCL, there is one DD for each screen
//* defined, based on the value of keyword MAXIMUM_NUMBER_OF_
//* SPLIT_SCREENS in the configuration table.
//* The DD name is in the form ISPWRKx, where x can be
//* 1-9, A-W.  For example, if the value of the keyword = 8,
//* only ISPWRK1 to ISPWRK8 need to be coded.

//ISPLST1  DD DISP=NEW,UNIT=VIO,SPACE=(CYL,(1,1)),
//            DCB=(LRECL=121,BLKSIZE=1210,RECFM=FBA)

//ISPLST2  DD DISP=NEW,UNIT=VIO,SPACE=(CYL,(1,1)),
//            DCB=(LRECL=121,BLKSIZE=1210,RECFM=FBA)

⋮
//ISPLSTW  DD DISP=NEW,UNIT=VIO,SPACE=(CYL,(1,1)),
//            DCB=(LRECL=121,BLKSIZE=1210,RECFM=FBA)

//* In this section of JCL, there is one DD for each screen
//* defined, based on the value of keyword MAXIMUM_NUMBER_OF_
//* SPLIT_SCREENS in the configuration table.
//* The DD name is in the form ISPLSTx, where x can be
//* 1-9, A-W.  For example, if the value of the keyword = 8,
//* only ISPLST1 to ISPLST8 need to be coded.
```

Figure 48. DD statements to preallocate data sets

Note:

1. When allocating to VIO, make sure that enough auxiliary storage is dedicated to VIO so that system availability is not affected.
2. Use of the BUFNO parameter on allocation of ISPF libraries is not supported.
3. The ISPF temporary data set default names associated with the ISPCTL*x* are SPFTEMP*x*.CNTL, respectively, where *x*= value 0-9, A-W.
4. The ISPF temporary data set default names associated with the ISPWRK*x* are SPFTEMP*x*.WORK, respectively, where *x*= value 1-9, A-W.
5. The ISPF temporary data set default names associated with the ISPLST*x* are SPFTEMP*x*.LIST, respectively, where *x*= value 1-9, A-W.
6. The ddnames ISPWRK*x* are used by ISPF for file tailoring services with ISPFILE allocated to a PDS. The ddnames ISPLST*x* are used for generated listings.

7. The ddname ISPCTL0 is used with the edit SUBMIT command. The ddnames ISPCTLx are used with ISPF compress (both interactive and the LMCOMP service) and ISPF background (option 5), and for processing the file tailoring service FTOPEN TEMP.
8. ISPCTL1 is a required ddname when using SCLM. The data set should be allocated as an ISPF temporary data set for SCLM foreground processing and should be added to the FLMLIBS skeleton for batch processing.
9. ISPF does not support multivolume temporary data sets. If SMS is enabled, temporary data sets dynamically allocated by ISPF must be assigned a data class with a volume count of one. The storage administrator can control this in the data class ACS routine by testing the execution mode (&XMODE) for a value of TSO.
10. If you have dialogs that need to edit or browse temporary data sets, use the LMINIT service to associate a DATAID with ddname ZTEMPN and invoke edit or browse using the DATAID parameter. For more information, refer to the BROWSE and EDIT services in the *z/OS ISPF Services Guide*.

Compress SCLM listings

The ISPF sample library ISP.SISPSAMP contains two sample members that demonstrate how to compress SCLM listings using the PACK option on the LMCOPY service. FLM03ASM is the language definition. FLM03LMC is a sample REXX exec. Comments explaining how to use these members are included in the code.

Chapter 5. Customizing DM

Customizing Dialog Manager (DM) describes procedures you can use to customize the DM component of ISPF to suit the particular needs of your installation:

See also Chapter 6, “ISPF installation-wide exits,” on page 135.

SMF command accounting

The MVS System Management Facility (SMF) collects and records a variety of system and job-related information. ISPF uses SMF to format information for a type 32 record. This type 32 record contains the names of the program functions and TSO commands being executed and the number of times each is used during the session. SMF also allows the installation to specify that the record is to include resources such as the total processor time under TCBs and SRBs and the total number of TGETs, TPUTs, and transactions associated with each name. The record is written when a TSO user logs off or when an SMF recording interval expires.

ISPF issues an SVC 109 (the extended service router, code X'19') to start and stop this functional accounting before and after the link to a program function, and before and after attaching a command. When processing command (CLIST) functions, the SVC 109 is issued before and after attaching the EXEC command processor. Therefore, the command name EXEC will be recorded, rather than the actual CLIST name.

The calls to SVC 109 can be nested depending on the nature of the program function or CLIST. The commands (and service units attributed to each command, if recorded) are accounted to the appropriate logical screen and are recorded as such. The command name ISPFSWAP is passed to SMF to indicate the user has swapped screens.

You must specify in module IEEMB846 the name of each module that is invoked for ISPF subfunctions or subcommands. You should also specify the command used to invoke ISPF. *z/OS MVS System Management Facilities (SMF)* provides details on including additional names. Without these additions to IEEMB846, ISPF will cause extensive counts of ***OTHER to be recorded in the SMF type 32 records.

Table 9 on page 106 lists the ISPF options and their related module names plus any additional commands that can be implicitly or explicitly invoked by a particular option.

© Copyright IBM Corp. 1984, 2025

Table 9. ISPF Options and related module names

Option	Module Names	TSO Commands
0	ISPISM	
1	ISRBRO	
2	ISREDIT	SUBMIT
3.1 and 3.2	ISRUDA	
3.3	ISRUMC	
3.4	ISRUDL	
3.5	ISRURS	
3.6	ISRUHC	SUBMIT, ICQCPC00
3.7	ISPWSD	
3.8	ISRUOLP	STATUS, OUTPUT, SUBMIT, ICQCPC00
3.9	ISPUCM	
3.11	ISRFMT	
3.12	ISRSSM	
3.13 and 3.15	ISRSEPRM	
3.14	ISRSFM	
3.16	ISRUTABL	
3.17	ISRUUDL	
4	ISRFPR	ASM, FORT, PLI, PLIC, LINK, TESTCOB, TESTFORT, CALL, SCRIPT, ALLOC, FREE, ICQCPC00
5	ISRJB1	SUBMIT
6	ISRPTC	All commands except those prohibited by ISPF in ISPTCM
7	ISPYXDR	
7.1	ISPYFI	
7.2	ISPYPI	
7.3	ISPYVI	
7.4	ISPYTI	
7.5	ISPYLI	
7.6	ISPYSI	
7.7.1	ISPYRFI	
7.7.2	ISPYRVI	
7.8	ISPYBI	
7.T	ISPTUTOR	
9	ISRALTDI	
10	FLMDDL	
10.1	FLMEB$	

Table 9. ISPF Options and related module names (continued)

Option	Module Names	TSO Commands
10.2	FLMED$	
10.3	FLMUDU$	
10.4	FLMBD$	
10.5	FLMPD$	
10.6	FLMPTC	
10.A	FLMA	
11	ISRUDA	
X	ISPLLP	SUBMIT

Table 10 on page 107 lists the ISPF commands with the related modules:

Table 10. ISPF commands

Command	Module Name
COLOR	ISPOPT
CUAATTR	ISPOPT
ENVIRON	ISPENV
EXHELP	ISPTUTOR
FKA	ISPOPF
HELP	ISPTUTOR
ISPFVAR	ISPISM
ISPLIBD	ISPLLS
ISPPREP	ISPPREP
ISRRLIST	ISRDSLST
ISRROUTE	ISRRCL
KEYLIST	ISPKLU
KEYS	ISPOPT
KEYSHELP	ISPTUTOR
LIST	ISPLLP
LOG	ISPLLP
MSGID	ISPOMI
PANELID	ISPOPI
PFSHOW	ISPOPF
PSCOLOR	ISPOPT
REFACTD	ISRDSLST
REFACTL	ISRDSLST
REFADDD	ISRRSLST

Table 10. ISPF commands (continued)

Command	Module Name
REFADDL	ISRRSLST
REFLISTD	ISRDSLST
REFLISTL	ISRDSLST
REFOPEND	ISRDSLST
REFOPENL	ISRDSLST
SAREA	ISPSAM
SETTINGS	ISPISM
START	ISPSTRT
TSOCMD	ISRPTC
TUTOR	ISPTUTOR
ZKEYS	ISPOPT

Pre-allocation of List/Log data sets

ISPF normally allocates the ISPF list and log data sets (sequential data sets) the first time a user requests printed output or takes action that generates log output. The user can control the printing and disposition of these data sets at ISPF termination and by issuing the ISPF LOG and LIST commands.

You can pre-allocate the list or log data set to a sequential data set. If you do this, the data sets are automatically saved when the user logs off TSO. If both data sets are pre-allocated, the termination menu is bypassed when the user exits from ISPF. If the user reenters ISPF before logging off, any new output is added to the end of the sequential data sets.

You can pre-allocate the list and log data sets directly to SYSOUT by including these DD statements in the TSO LOGON procedure:

```
//ISPLIST  DD SYSOUT=A,
//        DCB=(LRECL=121,BLKSIZE=1210,RECFM=FBA)

//ISPLOG   DD SYSOUT=A,
//        DCB=(LRECL=125,BLKSIZE=129,RECFM=VA)
```

If you pre-allocate these data sets to SYSOUT, they are automatically printed when the user logs off TSO. If both data sets are pre-allocated, the termination menu is bypassed when the user exits from ISPF. If the user reenters ISPF before logging off, any new output is added to the end of the SYSOUT data sets.

Note:

1. You cannot use ISPF option 7.5 to browse log data sets allocated to SYSOUT.
2. You can use the ISPF Log/List pop-up on the Settings panel to specify either the number of lines per page or to bypass logging altogether (by specifying zero primary pages). The rest of the information on these panels is ignored if the list and log data sets are allocated to SYSOUT.
3. The defaults for the list data set are LRECL=121, line length=120, RECFM=FBA. However, you can use the Log/List pop-up on the Settings panel to change the characteristics of the list data set so screen images wider than 121 characters can be printed.
4. You cannot issue the LOG or LIST command to process a preallocated log or list data set.
5. The ISPF temporary data set default names associated with the ISPLOG ddname are SPFLOG*x*.LIST, where *x*=numeric value 0-9.
6. The ISPF temporary data set default names associated with the ISPLIST ddname are SPF*x*.LIST, where *x*=numeric value 0-9.

See *Terminating a dialog* in *z/OS ISPF Dialog Developer's Guide and Reference* for more information.

Specifying the maximum number of split screens

ISPF can run up to 32 logical screens at one time. You can specify the maximum number of logical screens allowed for your installation by modifying the ISPF configuration table. See Chapter 2, "The ISPF Configuration Table," on page 9 for more information about modifying the ISPF configuration table. Set the value of keyword MAXIMUM_NUMBER_OF_SPLIT_SCREENS to any number from 4 to 32. The default value is 8. Be sure to consider your users' region size when you set the limit for the maximum number of screens.

Setting ISPF site-wide defaults

You can set these site-wide defaults for ISPF by changing the ISPF configuration table:

- ISPF's Settings options
- CUA panel elements
- KEYLIST ON|OFF
- PFSHOW ON|OFF
- Log and List final disposition
- Default Primary panel
- SCROLL_DEFAULT (CSR|DATA|HALF|MAX|PAGE)
- SCROLL_MIN (minimum scroll value)
- SCROLL_MAX (maximum scroll value)
- STATUS_AREA_DEFAULT (CAL|FUN|OFF|SES|UPS|USE)

For this list of defaults, except SCROLL and STATUS AREA, ISPF provides a force option which indicates the default set in the ISPF configuration table should be used, even though users may have a value set in their system or user profile. The SCROLL and STATUS AREA defaults are included in the ISPSPROF when the initial profile is built. The user can later override the site-wide default.

You can also set these site-wide defaults by changing the ISPF configuration table:

- Log, list, and temporary data set block size.
- Log and temporary data set logical record length.
- PRINTDS operands, DEST, or WRITER

This list of defaults cannot be overridden by the end user.

See Chapter 2, "The ISPF Configuration Table," on page 9 for more information about modifying the Configuration table.

Customizing command tables

While running an application, you can use commands defined in eight different command tables. These command tables are:

- Application command table
- 3 User command tables
- 3 Site command tables
- System command table.

The user command tables and the site command tables are optional to use. They must be defined for your installation and present when ISPF is initialized. To make use of them, you must update the ISPF configuration table to include their application identification.

ISPF uses a specific order when searching the tables for commands you enter. However, you do have some control over the search order when using the optional site command tables. The site command tables can be searched either before or after the system command table. To define the search order relative to the site and system command tables, update the ISPF configuration table.

The keywords in the ISPF configuration table that determine the search order between the site command tables and the system command table, and whether or not user command tables and site command tables are defined, are:

APPLID_FOR_USER_COMMAND_TABLE
: The application ID for up to 3 user command tables. The default for each is NONE (no user command tables). The user command tables are searched after the command table for the current application, that is, the command table for the current APPLID, and before the site-wide and default system command tables.

APPLID_FOR_SITE_COMMAND_TABLE
: The application ID for up to 3 site command tables. The default for each is NONE (no site-wide command tables). The search order for the site command tables depends on the SITE_COMMAND_TABLE_SEARCH_ORDER_SETTING.

SITE_COMMAND_TABLE_SEARCH_ORDER
: Determines if the site-wide command tables are to be searched before or after the default ISPF command table. Valid values are BEFORE and AFTER. The default is BEFORE.

When you enter a command, the application command table is searched first. If the command is found, no further searching is necessary. If the command is not found in the application command table, up to 3 user command tables are searched. If the command is not found in the user command tables, up to 3 site command tables are searched or the system command table is searched (depending on the search order defined in the ISPF configuration table). Finally, if the command is still not found, the remaining command table(s), site or system, are searched. User command tables and site command tables are only searched if they have been defined in the ISPF configuration table and are present at ISPF initialization.

See Chapter 2, "The ISPF Configuration Table," on page 9 for more information about modifying the ISPF configuration table.

Application command table

Commands in the application command table are in effect only for the application you are running. Defining commands in the application command table lets you customize the set of commands you need for a particular application without redefining the system command table for each application.

ISPF provides a utility to create and modify the command tables. Enter ISPF option 3.9 from the ISPF Primary Option Menu. For more information about creating application command tables, refer to the *z/OS ISPF User's Guide Vol II* or the *z/OS ISPF Dialog Developer's Guide and Reference*.

User command tables

Commands in up to 3 user command tables are in effect for all applications. These commands can override identically named commands in the site or system command tables, and are themselves overridden by identically named commands in the application command table.

You must specify the application ID of the user command tables in the ISPF configuration table, or the user command tables cannot be used. If you do not have a user command table that matches the application ID, the ID is ignored until the table is present and ISPF is reinitialized.

Site command tables

Commands in up to 3 site command tables are in effect for all applications. An option in the ISPF configuration table enables a site to specify that up to 3 site command tables are searched before or after the system command table. If the site command tables are searched first, then commands in the site command tables can override identically named commands in the system command table, and in turn can be overridden by identically named commands in the application or user command tables. If the site command tables are searched after the system command table, then their commands override no others, but can be overridden by identically named commands in either the application, user, or system command tables.

You must specify the application ID of up to 3 site command tables in the ISPF configuration table, or the site command tables cannot be used. If you do not have a site command table that matches the application ID, the ID is ignored until the table is present and ISPF is reinitialized.

System command table

Commands in the system command table are in effect for all applications. However, these commands can be overridden by an identically named command in an application or user command table, or a site command table if it is defined to be searched first.

Creating ISPF terminal translation tables

Creating ISPF terminal translation tables describes how to perform these tasks:

- Create a set of ISPF translation tables.
- Modify existing ISPF panels to use with the set of translation tables.

The ISP.SISPSAMP library includes sample assembler source programs, ISPOWNTT and ISPAPLTT. Use these as examples of what a completed module should look like. You can modify the sample module to suit your requirements and supply your own values for each of the translation tables.

ISPF uses these translation tables:

- 2-byte input translation table
- 2-byte output translation table
- Uppercase character translation table
- Lowercase character translation table
- Valid terminal output translation table
- Generic string master translation table
- Alphabetic character translation table
- Collating sequence translation table

The sample assembler modules include all of these translation tables. Each translation table consists of 32 consecutive DC instructions. Each DC instruction consists of eight hexadecimal values. You must supply the 256 hexadecimal values that make up each of the translation tables. The address of each table is at the start of the assembler module. Ignore addresses such as TBIP, which are set to zero.

The sample ISPOWNTT corresponds to an English 3278/3279 terminal except that the collating sequence translation table is not used in English.

Note: If the set of terminal translation tables has to support Katakana characters, you must perform these steps:

1. Change the DPRP pointer in the source from A(0) to A(TTDPR).
2. Rename table TTUPP to TTDPR.
3. Add another 256-character table labeled TTUPP in which all characters translate to themselves except X'08', X'1C', X'1D', and X'1E', which translate to X'40'.

Uppercase character translation table

The uppercase character translation table (TTUPP in the example shown in Table 11 on page 112) translates screen input data as follows:

- Lowercase alphabetic characters translate to uppercase.
- X'08', X'1C', X'1D', and X'1E' translate to blank (X'40').

- All other hexadecimal values translate to themselves.

Table 11. Uppercase character translation table example

Table	Hexadecimal Code	Position

TTUPP	`DC X'0001020304050607'`
	`DC X'40090A0B0C0D0E0F'`
	`DC X'1011121314151617'`
	`. . .`
	`DC X'78797A7B7C7D7E7F'`
	`DC X'80C1C2C3C4C5C6C7'`
	`. . .`
	`DC X'E8E9EAEBECEDEEEF'`
	`DC X'F0F1F2F3F4F5F6F7'`
	`DC X'F8F9FAFBFCFDFEFF'`
		`(X'00' to X'07')`
		`(X'08' to X'0F')`
		`(X'10' to X'17')`
		
		`(X'78' to X'7F')`
		`(X'80' to X'87')`
		
		`(X'E8' to X'EF')`
		`(X'F0' to X'F7')`
		`(X'F8' to X'FF')`

Table 11 on page 112 shows how the uppercase character translation table might be represented in the assembler module. For example, this is true for Table 11 on page 112:

- The hexadecimal position for a lowercase ‘a’ (X'81'), contains the hexadecimal value for an uppercase ‘A’ (X'C1').
- The hexadecimal position for an uppercase ‘A’ (X'C1') contains the hexadecimal value for an uppercase ‘A’ (X'C1').

Enter the values you want in the 256 hexadecimal positions of the uppercase translation table. When you finish with the table, you are ready to move on to the second translation table.

Lowercase character translation table

The lowercase character translation table (TTLOW) must be left as it is. Its function is internal to ISPF.

Valid terminal output translation table

The valid terminal output translation table (TTVAL in the example shown in Table 12 on page 112) represents display characters, in hexadecimal, as follows:

- Valid display characters are represented with X'00'.
- Invalid display characters are represented with X'FF'.

Table 12. Valid terminal output translation table example

Table	Hexadecimal Code	Position

TTVAL	`DC X'FFFFFFFFFFFFFFFF'`
	`DC X'FFFFFFFFFFFFFFFF'`
	`DC X'FFFFFFFFFFFFFFFF'`
	`...`
	`DC X'FF00000000000000'`
	`DC X'FF00000000000000'`
	`...`
	`DC X'0000FFFFFFFFFFFF'`
	`DC X'0000000000000000'`
	`DC X'0000FFFFFFFFFFFF'`
		`(X'00' to X'07')`
		`(X'08' to X'0F')`
		`(X'10' to X'17')`
		
		`(X'78' to X'7F')`
		`(X'80' to X'87')`
		
		`(X'E8' to X'EF')`
		`(X'F0' to X'F7')`
		`(X'F8' to X'FF')`

Generic string master translation table

The positions in the generic string master translation table (TTGSM in the example shown in Table 13 on page 113) are filled in as follows:

X'00'
: Blank character

X'01'
: Invalid character

X'02'
: Special character

X'04'
: APL/TEXT special characters (only for APL and TEXT keyboards)

X'08'
: APL/TEXT alphabetic characters (only for APL and TEXT keyboards)

X'10'
: Lowercase alphabetic character

X'20'
: Uppercase alphabetic character

X'40'
: Numeric character

X'80'
: User character subset

Table 13. Generic string master translation table example

Table	Hexadecimal Code	Position
TTGSM	`DC X'0101010101010101'`	`(X'00' to X'07')`
	`DC X'0101010101010101'`	`(X'08' to X'0F')`
	`DC X'0101010101010101'`	`(X'10' to X'17')`
	`...`	
	`DC X'0102020202020202'`	`(X'78' to X'7F')`
	`DC X'0110101010101010'`	`(X'80' to X'87')`
	`...`	
	`DC X'2020010101010101'`	`(X'E8' to X'EF')`
	`DC X'4040404040404040'`	`(X'F0' to X'F7')`
	`DC X'4040010101010101'`	`(X'F8' to X'FF')`

Modifying the GSM to use the user character subset

The Generic String Master (GSM) translation table and its related tables can be modified to add an additional character subset to be used in picture string processing by the ISPF EDIT, FIND, and CHANGE commands.

These steps allow you to modify the GSM to use a character subset:

1. Choose which character is used to represent your subset. For example, Edit uses an @ to stand for alphabetic.
2. Modify the entry in the Generic String Special Character (GSS) table found in "Translation table for generic string special characters" on page 275 corresponding to the character you wish to use for a value of X'08'. This indicates where in the Generic String Character Code (GSC) table the mask for your character set is located. The GSC does not need to be changed as it is initially set for user character sets.
3. Modify the GSM entries of those characters you wish to include in your special character set so the high order bit is on.

Alphabetic character translation tables

There are two alphabetic translation tables:

- TTALP, which includes the number sign (#), the dollar sign ($), and the at sign (@) (Table 14 on page 114).
- TTALB, which does not include the number sign (#), the dollar sign ($), and the at sign (@) (Table 15 on page 114).

Note: Valid alphabetic characters are represented with X'00'. For example, the hexadecimal position for an uppercase ‘A’ contains X'00'. Non-alphabetic characters are represented with X'FF'. For example, the hexadecimal position for a blank contains X'FF'.

Table 14. Sample alphabetic character translation table including #, $, and @

Table	Hexadecimal Code	Position
TTALP	`DC X'FFFFFFFFFFFFFFFF'`	`(X'00' to X'07')`
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'08' to X'0F')`
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'10' to X'17')`
	`...`	
	`DC X'FFFFFF0000FFFFFF'`	`(X'78' to X'7F')`
	`DC X'FF00000000000000'`	`(X'80' to X'87')`
	`...`	
	`DC X'0000FFFFFFFFFFFF'`	`(X'E8' to X'EF')`
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'F0' to X'F7')`
	`DC X'0000FFFFFFFFFFFF'`	`(X'F8' to X'FF')`

Table 15. Sample alphabetic character translation table excluding #, $, and @

Table	Hexadecimal Code	Position
TTALB	`DC X'FFFFFFFFFFFFFFFF'`	`(X'00' to X'07')`
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'08' to X'0F')`
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'10' to X'17')`
	`...`	
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'50' to X'57')`
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'58' to X'5F')`
	`...`	
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'78' to X'7F')`
	`DC X'FF00000000000000'`	`(X'80' to X'87')`
	`DC X'0000FFFFFFFFFFFF'`	`(X'88' to X'8F')`
	`...`	
	`DC X'0000FFFFFFFFFFFF'`	`(X'E8' to X'EF')`
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'F0' to X'F7')`
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'F8' to X'FF')`

Collating sequence translation table

The collating sequence translation table (TTCOL in the example shown in Table 16 on page 115) contains the sort order of the 256 table entries, represented in hexadecimal (0-255). This table is used by TBSORT/TBADD services when the sort type is ‘C’, and by the ISPF Edit SORT command. See *z/OS ISPF Dialog Developer's Guide and Reference* for further information.

Example 1: If you want an ‘A’ to sort before a ‘B’, as shown in Table 16 on page 115, the hexadecimal position for an ‘A’ will contain a value that is less than the sort value found in the hexadecimal position for ‘B’. Similarly, to sort a blank (X'40') before an ‘A’, the hexadecimal table position for a blank will contain a sort value less than that of the ‘A’.

Example 2: If you want a blank (X'40') to sort last (not shown here), set the hexadecimal table position for a blank (X'40') to X'FF'.

Table 16 on page 115 shows an example of how the table would look if you wanted to sort strictly on the basis of hexadecimal values (for example, ‘a’ before ‘b’, ‘A’ before ‘A’).

Table 16. Collating sequence translation table example

Table	Hexadecimal Code	Position

TTCOL	`DC X'0001020304050607'`
	`DC X'08090A0B0C0D0E0F'`
	`DC X'1011121314151617'`
	`...`
	`DC X'78797A7B7C7D7E7F'`
	`DC X'8081828384858687'`
	`...`
	`DC X'E8E9EAEBECEDEEEF'`
	`DC X'F0F1F2F3F4F5F6F7'`
	`DC X'F8F9FAFBFCFDFEFF'`
		`(X'00' to X'07')`
		`(X'08' to X'0F')`
		`(X'10' to X'17')`
		
		`(X'78' to X'7F')`
		`(X'80' to X'87')`
		
		`(X'E8' to X'EF')`
		`(X'F0' to X'F7')`
		`(X'F8' to X'FF')`

If you want to sort strictly on the basis of hexadecimal codes (as ISPF does for English), set the pointer to the collating sequence table (COLP) to zero. In the case of a pure hexadecimal sort, ISPF does not require a table.

After you supply the values for these six tables, assemble and link-edit the module. After the load module is created, go to the next step.

Specifying terminal types

Having created a new set of translation tables you must provide a way for users to select the appropriate terminal type to refer to it. The simplest way is for the user to select the option OTHER in the ISPF Settings panel (ISPISMMN) and then enter the name of the load module (see Settings (Option 0)in the *z/OS ISPF User's Guide Vol II*). The load module is a member of a partitioned data set that is either allocated to the ISPLLIB ddname, or exists in STEPLIB, JOBLIB, or LINKLIB.

Validity checks are conducted on the load module to help prevent a user from loading bad translation tables. This could cause ISPF to fail until the faulty specification is removed by manually editing or deleting the user system profile. These checks are performed:

- verify that the module name matches an 8-byte constant at the start of the module (as in the sample program ISPOWNTT)
- verify that the uppercase translate table immediately follows the address pointer for the collating sequence translate table
- verify that the uppercase and lowercase translate tables are 256 bytes long

If the load of the new translation tables fails, ISPF reverts to the previous terminal type setting.

Older methods for specifying translation tables are still supported. See these topics for more information:

- "Changing the DTL source for ISPISMMN" on page 115
- "Changing the DTL source for ISPOPTxx panels" on page 117
- "Invoking ISPTTDEF" on page 118

Changing the DTL source for ISPISMMN

To change the ISPISMMN Dialog Tag Language source, use the Edit option to update the DTL source file members ISPZMMCH and ISPZMMSO. The member ISPISMMN defines the basic panel. This panel does not have to be modified but it does have to be reconverted with ISPDTLC after the changes to the imbed members ISPZMMCH and ISPZMMSO are complete.

The last <SELFLD tag in DTL source member ISPISMMN (see Figure 49 on page 116) defines the list of terminal types. Four columns of choices are specified. The number of choices in each column is calculated by the conversion utility. The choice definitions are found in file imbed ISPZMMCH.

```
<selfld type=single name=ztm pmtloc=before listtype=ddlist
        required=yes msg=ispo901 help=ispo901h autotab=no
        entwidth=2 selfmt=end selwidth=60 choicecols=4 choicedepth=*>
        &selfld_3_prompt;

&ispzmmch;        <:-- include CHOICE tags for terminal types -->

</selfld>
```

Figure 49. DTL source for terminal type selection - SELFLD tag

This example illustrates adding terminal type XXXX using module ISPOWNTT to panel ISPISMMN. Alternatively, you can replace an existing set of translation tables by typing over the terminal type and the name of the load module that the newly defined set replaces.

The modification to the DTL source member ISPZMMCH adds the terminal type selection to the panel display (see Figure 50 on page 116) and creates panel logic to determine which selection number will match a certain terminal type.

```
 Terminal Type   3    1. 3277       2. 3277A      3. 3278       4. 3278A
                      5. 3290A      6. 3278T      7. 3278CF     8. 3277KN
                      9. 3278KN    10. 3278AR    11. 3278CY    12. 3278HN
                     13. 3278HO    14. 3278IS    15. 3278L2    16. BE163
                     17. BE190     18. 3278TH    19. 3278CU    20. DEU78
                     21. DEU78A    22. DEU78T    23. DEU90A    24. SW116
                     25. SW131     26. SW500     27. 3278GR    28. 3278L1
                     29. OTHER     30. XXXX
```

Figure 50. Example: adding a terminal type to panel ISPISMMN

Modified DTL source for the English-language section of the member ISPZMMCH follows:

```
  ⋮
  <condexec lang=english>
    <choice selchar=1  checkvar=ztermp match=3277>3277       <:-- 01 -->
    <choice selchar=5  checkvar=ztermp match=3290A>3290A     <:-- 02 -->
    <choice selchar=9  checkvar=ztermp match=3278KN>3278KN   <:-- 03 -->
    <choice selchar=13 checkvar=ztermp match=3278HO>3278HO   <:-- 04 -->
    <choice selchar=17 checkvar=ztermp match=BE190>BE190     <:-- 05 -->
    <choice selchar=21 checkvar=ztermp match=DEU78A>DEU78A   <:-- 06 -->
    <choice selchar=25 checkvar=ztermp match=SW131>SW131     <:-- 07 -->
    <choice selchar=2  checkvar=ztermp match=3277A>3277A     <:-- 08 -->
    <choice selchar=6  checkvar=ztermp match=3278T>3278T     <:-- 09 -->
    <choice selchar=10 checkvar=ztermp match=3278AR>3278AR   <:-- 10 -->
    <choice selchar=14 checkvar=ztermp match=3278IS>3278IS   <:-- 11 -->
    <choice selchar=18 checkvar=ztermp match=3278TH>3278TH   <:-- 12 -->
    <choice selchar=22 checkvar=ztermp match=DEU78T>DEU78T   <:-- 13 -->
    <choice selchar=26 checkvar=ztermp match=SW500>SW500     <:-- 14 -->
    <choice selchar=3  checkvar=ztermp match=3278>3278       <:-- 15 -->
    <choice selchar=7  checkvar=ztermp match=3278CF>3278CF   <:-- 16 -->
    <choice selchar=11 checkvar=ztermp match=3278CY>3278CY   <:-- 17 -->
    <choice selchar=15 checkvar=ztermp match=3278L2>3278L2   <:-- 18 -->
    <choice selchar=19 checkvar=ztermp match=3278CU>3278CU   <:-- 19 -->
    <choice selchar=23 checkvar=ztermp match=DEU90A>DEU90A   <:-- 20 -->
    <choice selchar=27 checkvar=ztermp match=3278GR>3278GR   <:-- 21 -->
    <choice selchar=4  checkvar=ztermp match=3278A>3278A     <:-- 22 -->
    <choice selchar=8  checkvar=ztermp match=3277KN>3277KN   <:-- 23 -->
    <choice selchar=12 checkvar=ztermp match=3278HN>3278HN   <:-- 24 -->
    <choice selchar=16 checkvar=ztermp match=BE163>BE163     <:-- 25 -->
    <choice selchar=20 checkvar=ztermp match=DEU78>DEU78     <:-- 26 -->
    <choice selchar=24 checkvar=ztermp match=SW116>SW116     <:-- 27 -->
    <choice selchar=28 checkvar=ztermp 3278L1>3278L1         <:-- 28 -->
    <choice selchar=30 checkvar=ztermp match=XXXX>XXXX       <:-- 29 -->
  </condexec>
  ⋮
```

Figure 51. DTL source for terminal type selection - CHOICE tags

The conversion utility formats CHOICE tags in a top-to-bottom, left-to-right order, placing the first seven CHOICE tags in column 1 (choice numbers 1, 5, 9, 13, 17, 21, and 25), CHOICE tags 8 through 14 in column 2 (selection numbers 2, 6, 10, 14, 18, 22, and 26), and so on. The number of entries in each column is based on 28 total lines of CHOICE and CHDIV tags divided by the specified number of choice columns, in our example, 4 (as defined in Figure 49 on page 116 by the CHOICECOLS keyword). These tags are arranged so that the choices appear in a left-to-right, top-to-bottom order).

By modifying the DTL source member ISPZMMSO, the new terminal type XXXX will be associated with its load module name ISPOWNTT. Modified DTL source for the English-language section of the member ISPZMMSO follows:

```
/* set translate load module name based on terminal type */
⋮
&ZCHARLM = TRANS(&ZTERM
                 3277  ,  ISP3277
                 3277A ,  ISP3277A
</source>
<condexec lang=english
<source>
                 3278  ,  ISP3278
                 3278A ,  ISP3278A
                 3290A ,  ISP3278A
                 3278T ,  ISP3278T
</source>
</condexec>
<source>
                 3278CF,  ISP3278C
                 3277KN,  ISP3277K
                 3278KN,  ISP3278K
                 3278AR,  ISPAR78
                 3278CY,  ISPCY78
                 3278HN,  ISPHN78
                 3278HO,  ISPHO78
                 3278IS,  ISPIS78
                 3278L2,  ISPL278
                 BE163,   ISPB678
                 BE190,   ISPB978
                 3278TH,  ISPTH78
                 3278CU,  ISPCU78
                 UE3278,  ISP3278
                 UE3278A, ISP3278A
                 UE3290A, ISP3278A
                 UE3278T, ISP3278T
                 DEU78,   ISPGE78
                 DEU78A,  ISPGE78A
                 DEU90A,  ISPGE78A
                 DEU78T,  ISPGE78T
                 SW116 ,  ISPSW116
                 SW131 ,  ISPSW131
                 SW500 ,  ISPSW500
                 3278L1,  ISPL178
                 3278GR,  ISPGR78
                 OTHER,   ISPOTHR
                 XXXX,    ISPOWNTT)
</source>
```

Figure 52. DTL source for valid terminal types and associated load module names

In Figure 52 on page 117, the left entries (for example 3277, 3277A and 3278) are valid terminal types a user can specify. The right entries (for example ISP3277, ISP3277A and ISP3278) are the associated load module names.

The delivered ISPF terminal table names start with the prefix "ISP". ISPF does not require that user-defined terminal table names begin with the prefix "ISP"; however, PDF terminal names require the "ISR" prefix. PDF searches for the load module beginning with the fourth position of the actual table name and prefixes it with "ISR". See “Creating PDF translation tables” on page 146 for a discussion of PDF translation tables.

Changing the DTL source for ISPOPTxx panels

If you still use the old options panels ISPOPT1 or ISPOPT1A, use ISPF Edit to update the DTL source file imbed member ISPZPTTT. ISPZPTTT is used by both ISPOPT1 and ISPOPT1A. It is a conversion-language-sensitive file that includes both the terminal type verification statement and the terminal type load module selection translation construct. ISPZPTTT is distributed in ISP.SISPGMLI.

After this member is updated, run ISPDTLC to convert all of the listed panels.

To add a new terminal type XXXX using load module ISPOWNTT to these panels, update ISPZPTTT as follows:

```
<source type=proc>
  VER (&ZTERM NB LIST
          3277,3277A,3278,3278A,3278T,3278CF,3277KN,3278KN,3290A,3278AR,
          BE163,BE190,3278CY,3278HN,3278HO,3278IS,3278L2,3278TH,3278CU,
          DEU78,DEU78A,DEU90A,SW116,SW131,SW500,3278L1,DEU78T,3278GR,
          UE3278,UE3278A,UE3290A,UE3278T,XXXX
          MSG=ISPO004)

  /* set translate load module name based on terminal type */
  &ZCHARLM = TRANS(&ZTERM
                 3277  ,  ISP3277
                 3277A ,  ISP3277A
                 3278  ,  ISP3278
                 3278A ,  ISP3278A
                 3290A ,  ISP3278A
                 3278T ,  ISP3278T
                 3278CF,  ISP3278C
                 3277KN,  ISP3277K
                 3278KN,  ISP3278K
                 3278AR,  ISPAR78
                 3278CY,  ISPCY78
                 3278HN,  ISPHN78
                 3278HO,  ISPHO78
                 3278IS,  ISPIS78
                 3278L2,  ISPL278
                 BE163,   ISPB678
                 BE190,   ISPB978
                 3278TH,  ISPTH78
                 3278CU,  ISPCU78
                 UE3278,  ISP3278
                 UE3278A, ISP3278A
                 UE3290A, ISP3278A
                 UE3278T, ISP3278T
                 DEU78,   ISPGE78
                 DEU78T,  ISPGE78T
                 DEU78A,  ISPGE78A
                 DEU90A,  ISPGE78A
                 SW116 ,  ISPSW116
                 SW131 ,  ISPSW131
                 SW500 ,  ISPSW500
                 3278GR,  ISPGR78
                 3278L1,  ISPL178
                 OTHER,   ISPOTHR
                 XXXX,    ISPOWNTT)
</source>
```

Figure 53. ISPZPTTT modified to add new terminal type XXXX

Invoking ISPTTDEF

The ISPTTDEF program offers you an alternative approach to specifying the terminal type or the corresponding set of translation tables (the load module). You can invoke ISPTTDEF from a selection panel, command table, or dialog function. To invoke the program, enter:

```
SELECT PGM(ISPTTDEF) PARM(xxx)
```

where *xxx* is the terminal type (one of the types listed on the distributed panel ISPISMMN) or the name of the load module. When you specify a terminal type, the ISPTTDEF program loads and uses the appropriate load module for that terminal type. If you specify a load module, the program attempts to load a module with that name.

Creating ISPF code page translation tables

ISPF supports extended code pages that allow ISPF to display panels, messages, and variable application data correctly on terminals using any of the supported code pages. For example, ISPF can display a German panel on a French CECP (Country Extended Code Page) terminal, with all common characters displayed correctly. Any characters in the panel that do not exist in the terminal code page are displayed as periods.

The code page and character set are specified by CCSID (Coded Character Set Identifier) as defined by Character Data Representation Architecture (CDRA). ISPF supports these EXTENDED CODE PAGE CCSIDs for the TRANS service and also with the use of the CCSID keyword on panels and messages.

Table 17. Extended CCSID1 Supported

CCSID	Character Set	Code Page	Country/Language
00037	697	37	U.S.A. Canada Netherlands Portugal Brazil Australia New Zealand
00273	697	273	Austria Germany
00277	697	277	Denmark Norway
00278	697	278	Finland Sweden
00280	697	280	Italy
00284	697	284	Spain L.A. Spanish
00285	697	285	United Kingdom
00297	697	297	France
00420	235	420	Arabic
00424	941	424	Hebrew
00500	697	500	Switzerland Belgium
00838	1176	838	Thailand
00870	959	870	Latin-2
00871	697	871	Iceland
00875	923	875	Greece
00880	960	880	Cyrillic
01025	1150	1025	Cyrillic
01026	1126	1026	Turkey
01047	697	1047	Latin1
01123	1326	1123	Ukraine

Table 18. Extended CCSID1 Supported (EURO)

CCSID	Character Set	Code Page	Country/Language
00924	1353	0924	Latin9
01140	695	1140	U.S.A. Canada Netherlands Portugal Brazil Australia New Zealand
01141	695	1141	Austria Germany
01142	695	1142	Denmark Norway
01143	695	1143	Finland Sweden
01144	695	1144	Italy
01145	695	1145	Spain L.A. Spanish
01146	695	1146	United Kingdom
01147	695	1147	France
01148	695	1148	Switzerland Belgium
01149	695	1149	Iceland
01153	1375	1153	Latin2
01154	1381	1154	Cyrillic
01155	1378	1155	Turkey
01158	1388	1158	Ukraine
01160	1395	1160	Thailand
04899	1356	0803	Hebrew
04971	1371	0875	Greece
12712	1357	0424	Hebrew
16804	1461	0420	Arabic

These Extended CCSIDs (shown in Table 19 on page 121) are also supported for panels and messages that specify an extended code page. These are the mixed SBCS/DBCS CCSIDs for these languages.

Japanese (Katakana) and Simplified Chinese EXTENDED CODE PAGES are not supported on any terminal but these EXTENDED CODE PAGES are supported for the TRANS service and with the CCSID keyword on panels and messages.

Table 19. Extended SBCS and DBCS CCSIDs Supported

CCSID	Character Set	Code Page	Country
00930	1172	290	Japanese (Katakana)
00939	1172	1027	Japanese (Latin)
00933	1173	833	Korean
00935	1174	836	Simplified Chinese
00937	1175	037	Traditional Chinese
01159	65535	1159	Traditional Chinese
01364	65535	0834	Korean
01371	65535	0835	Traditional Chinese
01388	65535	0837	Simplified Chinese
01390	65535	0300	Japanese
01399	65535	0300	Japanese
05123	65535	1027	Japanese
08482	65535	0290	Japanese

Base code pages for terminals

ISPF provides direct translation between each BASE CODE PAGE and its EXTENDED CODE PAGE for panels or messages. It also provides direct translation between extended Japanese (Latin or English) and both base Japanese (English) and base Japanese (Katakana). All translation between the single-byte EXTENDED CODE PAGEs for the double-byte languages and the CECP code pages is through CCSID 00500.

ISPF supports the base code pages (including mixed SBCS/DBCS CCSIDs for the DBCS languages) shown in Table 20 on page 121.

Table 20. Base CCSIDs Supported

CCSID	Character Set	Code Page	Country/Language
00803	1147	424	Hebrew (Old)
00931	101	037	Japan (English)
04369	265	273	Germany and Austria
04371	273	275	Brazil
04373	281	277	Denmark and Norway
04374	285	278	Finland and Sweden
04376	293	280	Italy
04380	309	284	L.A. (Spanish Speaking)
04381	313	285	U.K. English
04393	1129	297	France
04934	938	838	Thailand
04966	959	870	Latin-2

Table 20. Base CCSIDs Supported (continued)

CCSID	Character Set	Code Page	Country/Language
04976	960	880	Cyrillic
05029	933	833	Korean
05031	936	836	Simplified Chinese
05033	101	037	Traditional Chinese
08229	101	037	U.S. English and Netherlands
08476	650	284	Spain
09122	332	290	Japan (Katakana)
41460	904	500	Switzerland
45556	908	500	Switzerland

ISPCCSID translation load modules

ISPCCSID translation load modules translate data from one CCSID to another. There is one translation load module for each of the supported CCSIDs. The name, or alias, of each CCSID translation load module is made up of a 5-digit CCSID, prefixed with "ISP". For example, load module ISP00111 supports translation of the CCSID 00111. Each CCSID translation load module must contain at least two translation tables. These translation tables convert data between the respective CCSID and CCSID 00500. In addition each CCSID load module can contain up to 256 pairs of optional “direct” translation tables. ISPF uses the direct translation tables when available. Otherwise, ISPF translates the characters through CCSID 00500. Translating through CCSID 00500 can result in valid characters being lost as CCSID 00500 does not have all possible code points defined.

You can add direct "To" and "From" translation tables for direct translation to prevent possible loss of characters through CCSID 00500 for character sets other than 697 or to augment the extended code page translation tables provided by ISPF. The direct translation CCSID must be one of the CCSIDs supported by ISPF (see “Extended code page translation tables provided by ISPF” on page 123) or added by the user.

Both "To" and "From" translation tables must be provided for direct translation tables as well as CCSID 00500 tables, even though there might be no translation needed. For example, to translate from a base CCSID to an extended CCSID for the same code page, all characters will translate to themselves.

Adding translation tables for extended code page support

You can provide support for additional code pages by creating or modifying translation tables using the sample assembler module ISPEXCP in the ISP.SISPSAMP library.

Any translation tables that are added must be named ISP*nnnnn*, where *nnnnn* is the CCSID, and must be a CCSID defined in the Character Data Representation Architecture Registry. This CCSID must be different from any of the supported CCSIDs. The translation tables should include code points X'40' through X'FE'.

Table 21 on page 123 and Table 22 on page 123 show examples of the "To" and "From" translation tables needed to translate characters between CCSID 00500 and CCSID 00037.

Table 21. Table for translating from CCSID 00037 to CCSID 00500.

Table	Hexadecimal Code	Position
TO_500	`DC X'4041424344454647'`	`(X'40' to X'47')`
	`DC X'4849B04B4C4D4EBB'`	`(X'48' to X'4F')`
	`DC X'5051525354555657'`	`(X'50' to X'57')`
	`DC X'58594F5B5C5D5EBA'`	`(X'58' to X'5F')`
	`. . .`	
	`DC X'78797A7B7C7D7E7F'`	`(X'78' to X'7F')`
	`DC X'8081828384858687'`	`(X'80' to X'87')`
	`. . .`	
	`DC X'E8E9EAEBECEDEEEF'`	`(X'E8' to X'EF')`
	`DC X'F0F1F2F3F4F5F6F7'`	`(X'F0' to X'F7')`
	`DC X'F8F9FAFBFCFDFE'`	`(X'F8' to X'FE')`

Table 22. Table for translating from CCSID 00500 to CCSID 00037.

Table	Hexadecimal Code	Position
FROM_500	`DC X'4041424344454647'`	`(X'40' to X'47')`
	`DC X'4849BA4B4C4D4E5A'`	`(X'48' to X'4F')`
	`DC X'5051525354555657'`	`(X'50' to X'57')`
	`DC X'5859BB5B5C5D5EB0'`	`(X'58' to X'5F')`
	`. . .`	
	`DC X'78797A7B7C7D7E7F'`	`(X'78' to X'7F')`
	`DC X'8081828384858687'`	`(X'80' to X'87')`
	`. . .`	
	`DC X'E8E9EAEBECEDEEEF'`	`(X'E8' to X'EF')`
	`DC X'F0F1F2F3F4F5F6F7'`	`(X'F0' to X'F7')`
	`DC X'F8F9FAFBFCFDFE'`	`(X'F8' to X'FE')`

The source for these modules is provided in ISPEXCP, in the ISP.SISPSAMP library.

Extended code page translation tables provided by ISPF

ISPF provides the translation tables shown in Table 23 on page 123, which you can update. They are distributed in ISP.SISPSAMP.

Table 23. Translation tables provided with ISPF

Table name	CCSID	Description
ISPSTC1	00037	U.S.A, Canada, Netherlands, Portugal, Brazil, Australia, and New Zealand
ISPSTC2	00273	Austria and Germany
ISPSTC3	00277	Denmark and Norway
ISPSTC4	00278	Finland and Sweden
ISPSTC5	00280	Italy
ISPSTC6	00284	Spain and L.A. (Spanish-speaking)
ISPSTC7	00285	United Kingdom
ISPSTC8	00297	France
ISPSTC9	00500	Switzerland and Belgium
ISPSTC10	00939	Japan (Latin)
ISPSTC11	00930	Japan (Katakana)
ISPSTC12	00933	Korea

Table 23. Translation tables provided with ISPF (continued)

Table name	CCSID	Description
ISPSTC13	00935	Simplified Chinese
ISPSTC14	00937	Traditional Chinese
ISPSTC15	00870	Latin 2
ISPSTC16	00880	Cyrillic
ISPSTC17	01025	Cyrillic

ISPCCSID translation load module generation macro

You can use the assembler macro, ISPCCSID, to generate custom ISPCCSID translation load modules. The macro also allows you to add "direct" translation tables to the ISPCCSID translation load modules ISPF supplies with the product. Calls to this macro must also be coded for the To_500 and From_500 tables and any "To" and "From" tables for direct translation. The load module must have either the name ISP*xxxxx* (where *xxxxx* is new CCSID) or an alias of ISP*xxxxx*. In both cases, the load module should be a CCSID defined in the Character Data Representation Architecture Registry.

Note that only the values for the hexadecimal digits X'40' through X'FE' are defined in a given translation table. These are the only code points that vary from CCSID to CCSID.

The first time you use the ISPCCSID macro, you must identify the CCSID of the ISPCCSID translation load module and provide the addresses of the "To" and "From" CCSID 00500 translation tables.

You can use the ISPCCSID macro again with the same ISPCCSID translation load module generation to identify the CCSID and translation table addresses of optional direct "To" and "From" translation tables.

The format of calls to the ISPCCSID assembler macro is:

```
ISPCCSID  CCSID=nnnnn,TO=to-address,FROM=from-address
```

The required parameters of the ISPCCSID macro are:

nnnnn
This parameter is a 5-digit decimal (5 characters) number that specifies a CCSID. The *nnnnn* value on the first or only ISPCCSID macro definition is the CCSID associated with the ISPCCSID translation load module. The *nnnnn* value on other than the first ISPCCSID macro definition is the CCSID associated with direct "To" and "From" translation tables. If this parameter is not 5 digits, it causes an assembly error.

to-address
On the first or only ISPCCSID macro definition, this parameter specifies the address of the translation table that converts data from the CCSID associated with the respective ISPCCSID translation load module to CCSID 00500. On subsequent ISPCCSID macro definitions within the same ISPCCSID translation load module, this parameter specifies the address of the translation table that converts data from the CCSID associated with the respective ISPCCSID translation load module to the CCSID specified on this ISPCCSID macro definition.

from-address
On the first or only ISPCCSID macro definition, this parameter specifies the address of the translation table that converts data from CCSID 00500 to the CCSID associated with the respective ISPCCSID translation load module. On subsequent ISPCCSID macro definitions within the same ISPCCSID translation load module, this parameter specifies the address of the translation table that converts data from the CCSID specified on this ISPCCSID macro definition to the CCSID associate with the respective ISPCCSID translation load module.

ISPCCSID translation load module definition examples

Each ISPCCSID translation load module must be compiled separately using High Level Assembler (or a functional equivalent).

This example shows the ISPCCSID macro used with the Basic ISP00111 translation module.

```
          ISPCCSID CCSID=00111,TO=TRTO500,FROM=TRFR500
*
*
TRTO500   DC    XL191'...            00111 TO 00500
TRFR500   DC    XL191'...            00111 FROM 00500 (00500 TO 00111)
          END
```

This example shows the ISPCCSID macro used with the ISP00222 translation module with two direct CCSID entries.

```
          ISPCCSID CCSID=00222,TO=TRTO500,FROM=TRFR500
          ISPCCSID CCSID=00333,TO=TRT00333,FROM=TRF00333
          ISPCCSID CCSID=00444,TO=TRT00444,FROM=TRF00444
*
*
TRTO500   DC    XL191'...            00222 TO 00500
TRFR500   DC    XL191'...            00222 FROM 00500 (00500 TO 00222)
*
*
TRT00333  DC    XL191'...            00222 TO 00333
```

Example of user-modifiable ISPF translation table

This is the module for CCSID 00037 (ISPSTC1). The existing tables can be modified, or more pairs of direct translation tables can be added. To add direct translation tables, add a new ISPCCSID macro call for the new direct translation tables, and add the new tables. The assembler program should be renamed to ISPTTC*nn*, where *nn* is the last 1-digit or 2-digit number of the ISPSTC*nn* name. For example, ISPSTC1 should be renamed ISPTTC1, and ISPSTC14 renamed ISPTTC14.

```
*  THESE MACROS WILL GENERATE THE CCSID 00037 MODULE.
*
*
         ISPCCSID CCSID=00037,TO=TTC1T5H,FROM=TTC1F5H
         ISPCCSID CCSID=08229,TO=TTC1TB1,FROM=TTC1FB2
         ISPCCSID CCSID=04371,TO=TTC1TB2,FROM=TTC1FB2
*
*    TTC1T5H - CCSID 00037 TO CCSID 00500 Table
*
TTC1T5H  DS   0XL191
         DC X'4041424344454647'          (X'40' TO X'47')
         DC X'4849B04B4C4D4EBB'          (X'48' TO X'4F')
         DC X'5051525354555657'          (X'50' TO X'57')
         DC X'58594F5B5C5D5EBA'          (X'58' TO X'5F')
         DC X'6061626364656667'          (X'60' TO X'67')
         DC X'68696A6B6C6D6E6F'          (X'68' TO X'6F')
         DC X'7071727374757677'          (X'70' TO X'77')
         DC X'78797A7B7C7D7E7F'          (X'78' TO X'7F')
         DC X'8081828384858687'          (X'80' TO X'87')
         DC X'88898A8B8C8D8E8F'          (X'88' TO X'8F')
         DC X'9091929394959697'          (X'90' TO X'97')
         DC X'98999A9B9C9D9E9F'          (X'98' TO X'9F')
         DC X'A0A1A2A3A4A5A6A7'          (X'A0' TO X'A7')
         DC X'A8A9AAABACADAEAF'          (X'A8' TO X'AF')
         DC X'5FB1B2B3B4B5B6B7'          (X'B0' TO X'B7')
         DC X'B8B94A5ABCBDBEBF'          (X'B8' TO X'BF')
         DC X'C0C1C2C3C4C5C6C7'          (X'C0' TO X'C7')
         DC X'C8C9CACBCCCDCECF'          (X'C8' TO X'CF')
         DC X'D0D1D2D3D4D5D6D7'          (X'D0' TO X'D7')
         DC X'D8D9DADBDCDDDEDF'          (X'D8' TO X'DF')
         DC X'E0E1E2E3E4E5E6E7'          (X'E0' TO X'E7')
         DC X'E8E9EAEBECEDEEEF'          (X'E8' TO X'EF')
         DC X'F0F1F2F3F4F5F6F7'          (X'F0' TO X'F7')
         DC X'F8F9FAFBFCFDFE'            (X'F8' TO X'FE')
*
*    TTC1F5H - CCSID 00037 FROM CCSID 00500 Table
*
TTC1F5H  DS   0XL191
```

```
         DC X'4041424344454647'          (X'40' TO X'47')
         DC X'4849BA4B4C4D4E5A'          (X'48' TO X'4F')
         DC X'5051525354555657'          (X'50' TO X'57')
         DC X'5859BB5B5C5D5EB0'          (X'58' TO X'5F')
         DC X'6061626364656667'          (X'60' TO X'67')
         DC X'68696A6B6C6D6E6F'          (X'68' TO X'6F')
         DC X'7071727374757677'          (X'70' TO X'77')
         DC X'78797A7B7C7D7E7F'          (X'78' TO X'7F')
         DC X'8081828384858687'          (X'80' TO X'87')
         DC X'88898A8B8C8D8E8F'          (X'88' TO X'8F')
         DC X'9091929394959697'          (X'90' TO X'97')
         DC X'98999A9B9C9D9E9F'          (X'98' TO X'9F')
         DC X'A0A1A2A3A4A5A6A7'          (X'A0' TO X'A7')
         DC X'A8A9AAABACADAEAF'          (X'A8' TO X'AF')
         DC X'4AB1B2B3B4B5B6B7'          (X'B0' TO X'B7')
         DC X'B8B95F4FBCBDBEBF'          (X'B8' TO X'BF')
         DC X'C0C1C2C3C4C5C6C7'          (X'C0' TO X'C7')
         DC X'C8C9CACBCCCDCECF'          (X'C8' TO X'CF')
         DC X'D0D1D2D3D4D5D6D7'          (X'D0' TO X'D7')
         DC X'D8D9DADBDCDDDEDF'          (X'D8' TO X'DF')
         DC X'E0E1E2E3E4E5E6E7'          (X'E0' TO X'E7')
         DC X'E8E9EAEBECEDEEEF'          (X'E8' TO X'EF')
         DC X'F0F1F2F3F4F5F6F7'          (X'F0' TO X'F7')
         DC X'F8F9FAFBFCFDFE'            (X'F8' TO X'FE')
*
*    TTC1TB1 - CCSID 00037 TO CCSID 08229 Table
*
TTC1TB1  DS   0XL191
         DC X'404B4B4B4B4B4B4B'          (X'40' TO X'47')
         DC X'4B4B4A4B4C4D4E4F'          (X'48' TO X'4F')
         DC X'504B4B4B4B4B4B4B'          (X'50' TO X'57')
         DC X'4B4B5A5B5C5D5E5F'          (X'58' TO X'5F')
         DC X'60614B4B4B4B4B4B'          (X'60' TO X'67')
         DC X'4B4B6A6B6C6D6E6F'          (X'68' TO X'6F')
         DC X'4B4B4B4B4B4B4B4B'          (X'70' TO X'77')
         DC X'4B797A7B7C7D7E7F'          (X'78' TO X'7F')
         DC X'4B81828384858687'          (X'80' TO X'87')
         DC X'88894B4B4B4B4B4B'          (X'88' TO X'8F')
         DC X'4B91929394959697'          (X'90' TO X'97')
         DC X'98994B4B4B4B4B4B'          (X'98' TO X'9F')
         DC X'4BA1A2A3A4A5A6A7'          (X'A0' TO X'A7')
         DC X'A8A94B4B4B4B4B4B'          (X'A8' TO X'AF')
         DC X'4B4B4B4B4B4B4B4B'          (X'B0' TO X'B7')
         DC X'4B4B4B4B4B4B4B4B'          (X'B8' TO X'BF')
         DC X'C0C1C2C3C4C5C6C7'          (X'C0' TO X'C7')
         DC X'C8C94B4B4B4B4B4B'          (X'C8' TO X'CF')
         DC X'D0D1D2D3D4D5D6D7'          (X'D0' TO X'D7')
         DC X'D8D94B4B4B4B4B4B'          (X'D8' TO X'DF')
         DC X'E04BE2E3E4E5E6E7'          (X'E0' TO X'E7')
         DC X'E8E94B4B4B4B4B4B'          (X'E8' TO X'EF')
         DC X'F0F1F2F3F4F5F6F7'          (X'F0' TO X'F7')
         DC X'F8F94B4B4B4B4B'            (X'F8' TO X'FE')
*
*    TTC1FB1 - CCSID 00037 FROM CCSID 08229 Table
*
TTC1FB1  DS   0XL191
         DC X'4041424344454647'          (X'40' TO X'47')
         DC X'48494A4B4C4D4E4F'          (X'48' TO X'4F')
         DC X'5051525354555657'          (X'50' TO X'57')
         DC X'58595A5B5C5D5E5F'          (X'58' TO X'5F')
         DC X'6061626364656667'          (X'60' TO X'67')
         DC X'68696A6B6C6D6E6F'          (X'68' TO X'6F')
         DC X'7071727374757677'          (X'70' TO X'77')
         DC X'78797A7B7C7D7E7F'          (X'78' TO X'7F')
         DC X'8081828384858687'          (X'80' TO X'87')
         DC X'88898A8B8C8D8E8F'          (X'88' TO X'8F')
         DC X'9091929394959697'          (X'90' TO X'97')
         DC X'98999A9B9C9D9E9F'          (X'98' TO X'9F')
         DC X'A0A1A2A3A4A5A6A7'          (X'A0' TO X'A7')
         DC X'A8A9AAABACADAEAF'          (X'A8' TO X'AF')
         DC X'B0B1B2B3B4B5B6B7'          (X'B0' TO X'B7')
         DC X'B8B9BABBBCBDBEBF'          (X'B8' TO X'BF')
         DC X'C0C1C2C3C4C5C6C7'          (X'C0' TO X'C7')
         DC X'C8C9CACBCCCDCECF'          (X'C8' TO X'CF')
         DC X'D0D1D2D3D4D5D6D7'          (X'D0' TO X'D7')
         DC X'D8D9DADBDCDDDEDF'          (X'D8' TO X'DF')
         DC X'E0E1E2E3E4E5E6E7'          (X'E0' TO X'E7')
         DC X'E8E9EAEBECEDEEEF'          (X'E8' TO X'EF')
         DC X'F0F1F2F3F4F5F6F7'          (X'F0' TO X'F7')
         DC X'F8F9FAFBFCFDFE'            (X'F8' TO X'FE')
*
*    TTC1TB2 - CCSID 00037 TO CCSID 04371 Table
```

```
*
TTC1TB2  DS   0XL191
         DC X'404B4B4B4B4B794B'          (X'40' TO X'47')
         DC X'4B4B4B4B4C4D4E4B'          (X'48' TO X'4F')
         DC X'50D04B4B4B4B4B4B'          (X'50' TO X'57')
         DC X'4B4B4F5A5C5D5E4B'          (X'58' TO X'5F')
         DC X'60614B4B4B4B7C4B'          (X'60' TO X'67')
         DC X'5B4B4B6B6C6D6E6F'          (X'68' TO X'6F')
         DC X'4B4A4B4B4B4B4B4B'          (X'70' TO X'77')
         DC X'4B4B7A4B4B7D7E7F'          (X'78' TO X'7F')
         DC X'4B81828384858687'          (X'80' TO X'87')
         DC X'88894B4B4B4B4B4B'          (X'88' TO X'8F')
         DC X'4B91929394959697'          (X'90' TO X'97')
         DC X'98994B4B4B4B4B4B'          (X'98' TO X'9F')
         DC X'4BA1A2A3A4A5A6A7'          (X'A0' TO X'A7')
         DC X'A8A94B4B4B4B4B4B'          (X'A8' TO X'AF')
         DC X'5F44B4BB4B4B4B4B'          (X'B0' TO X'B7')
         DC X'4B4B4B4B4B4B4B4B'          (X'B8' TO X'BF')
         DC X'4BC1C2C3C4C5C6C7'          (X'C0' TO X'C7')
         DC X'C8C94B4B4B4B4BC0'          (X'C8' TO X'CF')
         DC X'4BD1D2D3D4D5D6D7'          (X'D0' TO X'D7')
         DC X'D8D94B4B4B4B4B4B'          (X'D8' TO X'DF')
         DC X'E04BE2E3E4E5E6E7'          (X'E0' TO X'E7')
         DC X'E8E94B4B4B4B4B7B'          (X'E8' TO X'EF')
         DC X'F0F1F2F3F4F5F6F7'          (X'F0' TO X'F7')
         DC X'F8F94B4B4B4B4B'            (X'F8' TO X'FE')
*
*    TTC1FB2 - CCSID 00037 FROM CCSID 04371 Table
*
TTC1FB2  DS   0XL191
         DC X'4041424344454647'          (X'40' TO X'47')
         DC X'4849714B4C4D4E5A'          (X'48' TO X'4F')
         DC X'5051525354555657'          (X'50' TO X'57')
         DC X'58595B685C5D5EB0'          (X'58' TO X'5F')
         DC X'6061626364656667'          (X'60' TO X'67')
         DC X'6869486B6C6D6E6F'          (X'68' TO X'6F')
         DC X'7071727374757677'          (X'70' TO X'77')
         DC X'78467AEF667D7E7F'          (X'78' TO X'7F')
         DC X'8081828384858687'          (X'80' TO X'87')
         DC X'88898A8B8C8D8E8F'          (X'88' TO X'8F')
         DC X'9091929394959697'          (X'90' TO X'97')
         DC X'98999A9B9C9D9E9F'          (X'98' TO X'9F')
         DC X'A0A1A2A3A4A5A6A7'          (X'A0' TO X'A7')
         DC X'A8A9AAABACADAEAF'          (X'A8' TO X'AF')
         DC X'B0B1B2B3B4B5B6B7'          (X'B0' TO X'B7')
         DC X'B8B9BABBBCBDBEBF'          (X'B8' TO X'BF')
         DC X'CFC1C2C3C4C5C6C7'          (X'C0' TO X'C7')
         DC X'C8C9CACBCCCDCECF'          (X'C8' TO X'CF')
         DC X'51D1D2D3D4D5D6D7'          (X'D0' TO X'D7')
         DC X'D8D9DADBDCDDDEDF'          (X'D8' TO X'DF')
         DC X'E0E1E2E3E4E5E6E7'          (X'E0' TO X'E7')
         DC X'E8E9EAEBECEDEEEF'          (X'E8' TO X'EF')
         DC X'F0F1F2F3F4F5F6F7'          (X'F0' TO X'F7')
         DC X'F8F9FAFBFCFDFE'            (X'F8' TO X'FE')
         END
```

Displaying square brackets used in C programs

The standard non-APL terminals that ISPF supports do not have the left and right brackets used in a C program. Therefore, the translation tables provided with ISPF are defined so that these characters are not valid and are displayed as periods.

If you have a terminal that supports these characters, you can modify the translation tables TTVAL and TTGSM. To do so, simply indicate that these characters are valid.

Note that the C/370 compiler expects the brackets at code points AD and BD. On an APL or TEXT terminal, if you use the ISPF-supplied terminal type 3278A, the code points AD and BD are displayed correctly as left and right brackets.

ISPEXEC processing

ISPEXEC is an external entry point in module ISPLINK. This is how ISPF supports the Call ISPEXEC interface in module dialogs. As a result of executing a CLIST that is not under ISPF, if the CLIST contains

ISPEXEC dialog service statements, CLIST might try to invoke the ISPEXEC module as a command processor. The results of this change are as follows:

- If the ISPLINK module (or its alias entry points ISPLNK, ISPEXEC, ISPEX, or ISPQRY) is not invoked under ISPF, TSO issues an error message with a return code of 20.
- The ISPEXEC entry point can interfere with your installation's setup if someone creates a CLIST called ISPEXEC, or if a CLIST specifically checks for a return code of 12 (TSO issues return code 12 if you try to invoke ISPEXEC when not under ISPF). To eliminate this problem, you make this change:

  Move the ISPLINK load module (and alias entry points ISPLNK, ISPEXEC, ISPEX and ISPQRY) to a library that is not defined in the search sequence for attaching commands under ISPF.
- The ISPLINK load module is usually link-edited with dialog functions coded in a programming language. Therefore, you should copy ISPLINK to an "automatic call" link-edit library. Modify existing CLISTs to recognize the ISPEXEC return code of 20 for invocation outside of an ISPF environment (testing for a return code not equal to 0 is recommended). If you choose, you can turn off (NOP) the error message issued from the ISPLINK (and ISPEXEC) module by applying this SUPERZAP:

```
NAME      ISPLINK    ISPLINK
VER       00FC       0A5D       TPUT SVC
REP       00FC       0700       NOP INSTRUCTION
```

  Remember that the location of the `0A5D` instruction can change from `00FC` as a result of maintenance.

ISPF-to-APL2 terminal type mappings

ISPF-to-APL2® terminal type mappings provides information about how to add or change mappings, if you need to do so. See *z/OS ISPF Dialog Developer's Guide and Reference* for a description of the ISPF/APL2 terminal type dialog, ISPAPTT.

ISPAPTT consists of an 8-character header (with value ISPAPTT), and fifteen 20-character entries. Each 20-character entry contains three fields. All fields are left-justified and padded on the right with blanks.

- The first 13 entries each appear as follows:
  - A 4-character EBCDIC sequence number; for example, "0001," "0002"
  - An 8-character EBCDIC ISPF terminal type
  - An 8-character EBCDIC APL2 terminal type
- The 14th entry consists of:
  - A 4-character EBCDIC sequence number; for example, "0014"
  - An 8-character EBCDIC value of "BATCH" noting the ISPF terminal type used if executing in the background (dialog ISPAPTT looks for this value)
  - An 8-character EBCDIC value of "1" noting the APL2 terminal type to be used if executing in the background
- The last (15th) entry contains:
  - A 4-character EBCDIC value equal to "LAST"
  - An 8-character value composed of all hexadecimal Fs, indicating the end of the list (dialog ISPAPTT looks for this value)
  - An 8-character EBCDIC value of "3277" indicating the APL2 terminal type to be used if the ISPF terminal type is not found in the list

You can change any of these values by performing a zap of the module. Include the APL2 terminal type to use in the background, and the APL2 terminal type to use if the ISPF terminal type is not found in the list.

Note: Do not alter the ISPF terminal type "BATCH" in the 14th entry or the 8-character hexadecimal Fs in the 15th entry.

Several of the first 13 entries can be changed to allow new terminal types. These entries contain a 4-character EBCDIC sequence number followed by 16 characters (two 8-character areas) of binary zeros.

To change these entries use a zap and enter the ISPF terminal type into the first 8-character area, and the corresponding APL2 terminal type into the second 8-character area.

Load APL2 workspace

If you run APL2 with ISPF, each ISPF/APL user must load an APL2 workspace from the ISPALIB library. This assumes APL2 is correctly installed and the VSAM cluster needed to hold existing workspaces is defined. The ISPF/APL user only has to perform this step once.

The workspace to use is in the ISP.SISPALIB data set. To place the workspace into the VSAM cluster with your existing APL2 workspaces:

1. Enter `apl2`
2. Specify `)IN 'ISP.SISPALIB(ISPFWS)'` to bring in the workspace information
3. Specify `)WSID` *nnnnnnnn* to name the workspace
4. Specify `)SAVE` to save the workspace and put it in the VSAM cluster
5. Specify `)OFF HOLD` to leave APL2

Tailoring ISPF defaults

In earlier versions of ISPF (before OS/390 V2R8.0), default values were set in the ISRCONFG table and the ISPDFLTA and ISPMTAIL macros. These values are now set in the ISPF Configuration table. See Chapter 2, "The ISPF Configuration Table," on page 9 for more information.

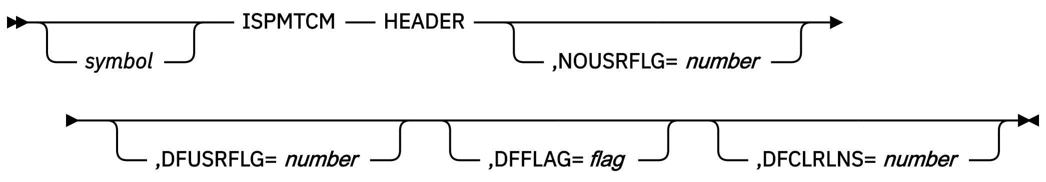
Customizing the ISPF TSO command table (ISPTCM)

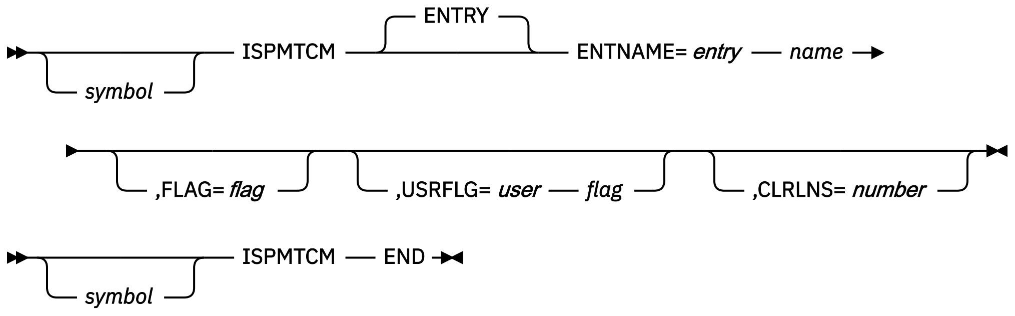
The ISPF TSO command table (ISPTCM) describes the TSO commands that are invoked under ISPF. When a TSO command is issued, ISPF searches ISPTCM. If the command is found, it uses the information in the table to process the command. If the command is not in ISPTCM, ISPF uses default values, which are in the table.

To change the list of TSO commands, their characteristics, or both, customize ISPTCM using the ISPMTCM macro. The assembler source of ISPTCM is member ISPTCMA in ISP.SISPSAMP. You must rename ISPTCMA to ISPTCM before assembling it. The macro ISPMTCM is in ISP.SISPMACS. ISP.SISPSAMP and ISP.SISPMACS are the default data set names for the ISPF data sets. Contact your system programmer for the names of the data sets containing ISPTCMA and ISPMTCM on your system. Customizing the ISPTCM involves modifying ISPTCMA, assembling it, and link-editing the ISPTCM module. When you use the ISPMTCM macro, remember that:

- If you modified ISPTCM in prior ISPF releases, you have to regenerate the table by using this macro.
- The first macro call must be HEADER, followed by one ENTRY macro call for each table entry desired, followed by a macro call of END.
- Use High Level Assembler to assemble ISPTCM.
- You can add up to four additional user-flag bytes for each entry in ISPTCM for your installation's use.
- The entry names in the IBM-supplied ISPTCM source, ISPTCMA, are arranged in alphabetical order. You must maintain alphabetical order for the entries within the ISPTCM.
- You can delete or modify any of the ISPF-provided entries in the table. ISPF allows you to have up to 1000 entries in ISPTCM.

The syntax of the macro is as follows:





Operand
    Description

HEADER
Parameter values are as follows:

NOUSRFLG
The number of user flag bytes defined. The value can be a number between 0 and 4, inclusive. The default is 0.

DFUSRFLG
The user flag that is to be used for an entry when the USRFLG operand is not specified. This must be a hexadecimal string, whose length in bytes must be NOUSRFLG. If you define a default user flag with this operand, the value for the NOUSRFLG operand must be greater than 0. If you do not define a default user flag and you indicated on the NOUSRFLG operand that a user flag exists, ISPF uses a string of binary zeros of whatever the length is in NOUSRFLG for the default user flag.

DFFLAG
The value of the ISPF flag to be used for the default entry in ISPTCM. The default entry determines the characteristics of commands not found in ISPTCM. The value should be a 1-byte hexadecimal string. The default is 61.

DFCLRLNS
The number of lines to clear from the bottom of the physical screen for line I/O when the CLRLNS operand is not specified. The value must be in the range from 0 to 99. The default is 3.

ENTRY
Parameter values are as follows:

ENTNAME
A valid TSO command name. This operand is required for ENTRY calls. The alphabetic characters in ENTNAME must be in uppercase letters. Duplicate entry names cause an error message to be issued.

FLAG
The value of the ISPF flag byte for the current entry. The default is 02.

Flag Field
    Flag Field Description

`B'1.......'`
Reserved.

`B'.1......'`
Command requires function pool. Set this bit on for a command processor program that issues dialog services.

`B'..1.....'`
Command requires authorization check. Set this bit on for a command processor that must be invoked as an authorized command.

`B'...1....'`
Command is not to be logged. Set this bit on if the TSO command buffer should not be written to the ISPLOG data set.

`B'....1...'`
Command is not supported by ISPF. Set this bit on for commands that cannot be invoked under ISPF.

`B'.....1..'`
Command is command procedure (CLIST). Set this bit on if this is the name of a CLIST member.

`B'......1.'`
Command is a command processor. Set this bit on if this is the name of a command processor program module.

`B'.......1'`
Command requires a BLDL to be issued. Set this bit on if a BLDL is to be issued to determine whether this is a command processor module or a CLIST.

USRFLG
The value of the user flag for the current entry. Specify this parameter only if NOUSRFLG is not 0. If you do not specify a value and NOUSRFLG is greater than 0, ISPF uses the default user flag (DFUSRFLG). If specified, USRFLG must be a hexadecimal string with a length equal to the value of NOUSRFLG.

CLRLNS
The minimum number of lines to clear if line mode is entered for this entry. The value should be an integer from 0 to 99. Specifying a value of 0 causes the entire physical screen to be erased. If you do not specify a value for CLRLNS, the DFCLRLNS value in the type HEADER call is used.

When the value for the number of lines to clear is nonzero, ISPF determines where to clear the screen according to:

1. ISPF calculates a value = (number of lines in the visible portion of the active logical screen - 1) - CLRLNS
2. The lesser of this calculated value and the number of lines in the panel displayed on the active logical screen (but not less than 0) is the number of the line after which the screen is cleared. Thus, ISPF will clear more than the CLRLNS number of lines if it can do so without overlaying the displayed panel.

Note:

1. If a CLRLNS value is larger than the visible portion of the active logical screen, ISPF erases the screen beginning at the top of the logical screen.
2. In split-screen mode, if line mode is entered when the top screen is active, the bottom screen will always be cleared. If the bottom screen is active, the bottom screen will be cleared (even if CLRLNS is greater than the physical screen size).
3. For DBCS devices, ISPF always erases the screen beginning at the top of the current logical screen, regardless of the CLRLNS value.
4. For 3290 devices, the entire physical screen is always cleared before going into line mode.

END
Must be last macro call.

Sample ISPTCM definition

An example of how to use the ISPMTCM macro to build ISPTCM is shown in Figure 54 on page 132.

```
ISPMTCM  HEADER,DFCLRLNS=5,NOUSRFLG=2,DFFLAG=61

ISPMTCM  ENTRY,ENTNAME=ALLOCATE,CLRLNS=4

ISPMTCM  ENTNAME=MYCMDA,USRFLG=FFFF
ISPMTCM  ENTNAME=MYCMD1,CLRLNS=10,FLAG=14

ISPMTCM  END
```

Figure 54. Sample ISPTCM definition

ISPTCM usage notes

- ISPTCM is a load module that contains a list of command names and their characteristics. ISPF processes each command in ISPTCM according to the FLAG field defined for its entry. If a command is not in ISPTCM, the DFFLAG parameter is used. The default value of DFFLAG is 61, which indicates to ISPF that commands not contained in ISPTCM require a function pool, an authorization check, and must be logged. Also, a BLDL should be done to locate the command. If the BLDL cannot locate the command, ISPF assumes it to be CLIST and attaches the EXEC command processor.

  You can alter DFFLAG to suit the needs of your installation. If you have not changed DFFLAG, or changed it so that it still calls for a BLDL, if a command processor is to run from the link pack area, its name *must* be in ISPTCM.
- Certain commands, such as, LOGON and ISPF are invalid under ISPF. Do not attempt to make these entries valid by changing the FLAG. The results are unpredictable.
- The USRFLG is an optional field for an entry in ISPTCM. If you define exit routines for TSO command start or TSO command end user exits, or both, you can define USRFLG according to your installation's needs. If you do, ISPF passes these flags to the exit routines. These flags do not affect ISPF execution. See Chapter 6, "ISPF installation-wide exits," on page 135 for a description of how the parameters are passed.
- One, and only one, of the last 3 bits of FLAG and DFFLAG must be 1. Otherwise, the results will be unpredictable.
- The presence or absence of a command in the TCM can affect the search sequence, as depicted in this table.

Search sequence for attaching commands

ISPF attaches a command invoked through the SELECT command, ISPF option 6, or TSO command. The search sequence for locating the command is shown in Figure 55 on page 133.

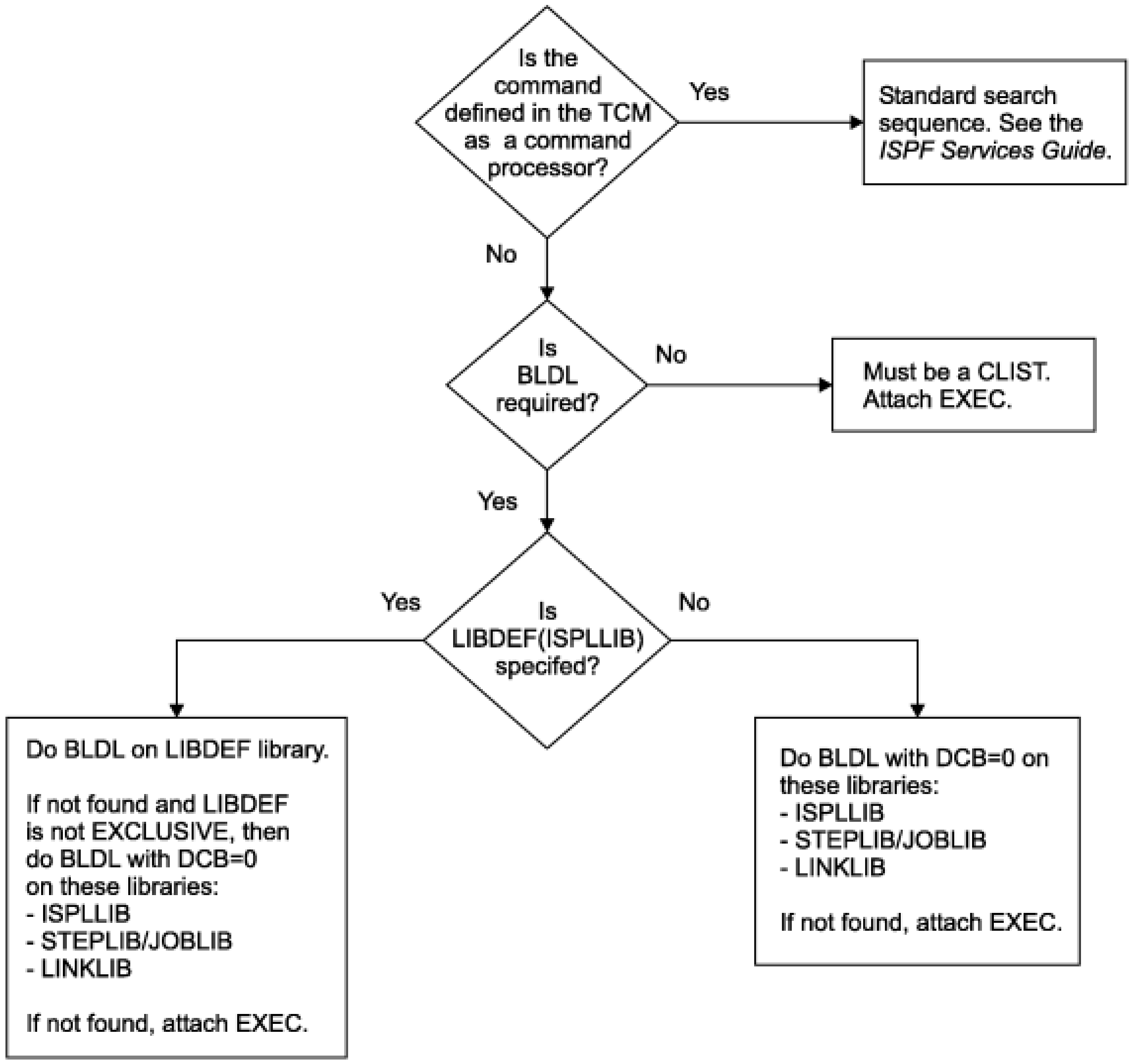

Is the command defined in the TCM as a command processor?
Yes
Standard search sequence. See the *ISPF Services Guide*.
No
Is BLDL required?
No
Must be a CLIST. Attach EXEC.
Yes
Is LIBDEF(ISPLLIB) specifed?
Yes
No
Do BLDL on LIBDEF library.
If not found and LIBDEF is not EXCLUSIVE, then do BLDL with DCB=0 on these libraries:
- ISPLLIB
- STEPLIB/JOBLIB
- LINKLIB
If not found, attach EXEC.
Do BLDL with DCB=0 on these libraries:
- ISPLLIB
- STEPLIB/JOBLIB
- LINKLIB
If not found, attach EXEC.

Figure 55. Search sequence for attaching commands

Alternate option 7.1 panels

Panel ISPYFP, the normal panel displayed with option 7.1, requires extra scrolling to display some of the test entry fields and options. Because this can be inconvenient, ISPF provides alternate panels for ISPYFP. The alternate panels present all of the same entry fields and options in revised formats.

Panel ISPYFPA places most of the commonly used information within the first 24 panel lines. This format often eliminates the need for panel scrolling.

Panel ISPYFPB is similar to panel ISPYFPA, but it has a selection field that enables the user to select a function: panel, command, program, or request. Unlike panels ISPYFP or ISPYFPA, on panel ISPYFPB the panel, command, program, or request fields can all contain values.

All three panels are included in the product panel library. To select which panel to use, set the ISPF Configuration table keyword USE_ALTERNATE_DIALOG_TEST_PANEL to one of these options:

1
: ISPYFP panel

2
: ISPYFPA panel

3
: ISPYFPB panel

ISPF multicultural support

Note: The term "multicultural support" has replaced the previous term "National Language Support" (or "NLS").

To make ISPF available at your installation in a language other than or in addition to English, follow these instructions. You can install as many of the supported languages as needed by your installation.

1. Allocate and load libraries specific to the language. See the *z/OS Program Directory* in the z/OS Internet library (www.ibm.com/servers/resourcelink/svc00100.nsf/pages/zosInternetLibrary) for installation instructions.
2. Optionally, set up a default language for your installation.

   With the standard version of ISPF, the default session language is English. You must invoke ISPF (ISPSTART) with a language keyword to get the session in a different language. If ISPF is used at your installation primarily in a non-English language, it is recommended that you change the default language. The benefits of doing this are:

   - Not having to enter the language keyword when invoking ISPSTART
   - Improved initialization time
   - Possibly smaller LPA use
   - Initialization error messages are issued in the default language.
   - See "Changing the session language default value" on page 134 for further details.
3. Set up the execution environment.

   To run ISPF in any session language, perform these steps:

   - Allocate panel, message, skeleton, table, and profile table libraries according to the language desired. The ddnames ISPPLIB, ISPMLIB, ISPSLIB, ISPTLIB do not change with the language used.
   - Issue the ISPSTART command (with the desired language keyword if this language is different from the default language).

Changing the session language default value

The Session Language Default value is a keyword, DEFAULT_SESSION_LANGUAGE, in the ISPF Configuration Table (see "DEFAULT_SESSION_LANGUAGE" on page 245). For information about changing Configuration Table settings, see Chapter 2, "The ISPF Configuration Table," on page 9.

Chapter 6. ISPF installation-wide exits

ISPF provides 16 installation-wide exits that allow you to collect system-related information such as accounting, activity monitoring, authorization checking, and installation tailoring. These installation-wide exits occur at selected places during ISPF execution and pass control to your exit routines in the order you define them when you install the routines. You can define one or more exit routines to process at each user exit.

Table 24 on page 135 summarizes the ISPF installation-wide exits. The exit ID is a unique number that identifies that installation-wide exit to ISPF.

Table 24. ISPF installation-wide exits

Exit ID	Installation Exit Name	Possible Uses for the Routine
1	ISPF initialization	Provides accounting and monitoring capabilities before ISPF initialization.
2	ISPF termination	Provides accounting and monitoring capabilities before ISPF termination.
3	SELECT service start	Provides monitoring information and lets you restrict access to applications selected through ISPF.
4	SELECT service end	Marks the end of a program, command, or menu invoked through any of the SELECT services.
5	TSO command start	Provides for monitoring and restricting commands invoked through ISPF; allows commands newly added to the system to be invoked without updating ISPTCM.
6	TSO command end	Provides for monitoring of TSO commands invoked through ISPF.
7	LIBDEF service	Provides for restrictions on the use of the LIBDEF service.
8	RESERVE	Allows use of your own method of serializing resources in addition to the RESERVE done by ISPF.
9	RELEASE	Provides for the release of any resources acquired at the RESERVE user exit.
10	Logical screen start	Allows for installation-wide exits to gather accounting and monitoring information for each logical screen.
11	Logical screen end	Gathers accounting and monitoring information for each logical screen.
12	ISPF/PDF service start	Monitors ISPF and PDF dialog services invoked through the ISPLINK or ISPEXEC interfaces.
13	ISPF/PDF service end	Marks the termination of ISPF or PDF dialog services invoked through the ISPLINK or ISPEXEC interfaces.
14	SWAP logical screens	Indicates a change of the active logical screen. Together with the logical screen start and end installation-wide exits, the routine can monitor resource use for each ISPF logical screen.
15	DISPLAY service start	Provides for tailoring of panels to be displayed.
16	Log, list, and temporary data set allocation	Controls data set naming conventions for log, list, and temporary data sets.

The ISPF installation-wide exits are linked together in the ISPEXITS load module. The main entry point of ISPEXITS is ISPXDT. ISPXDT defines which of the 16 installation-wide exits you plan to use and the names of the exit routines that receive control at each user exit. To assist you in building ISPXDT, ISPF provides seven assembler macros. These macros are described in "Exit macros" on page 255.

As you write exit routines for the ISPF installation-wide exits, remember that:

- Even though you can define multiple exit routines for a user exit, if an exit routine returns a return code of 12 or greater in register 15, control returns to ISPF without executing the remaining routines.
- You cannot activate or deactivate a user exit while an ISPF session is in progress. However, you can make changes to the ISPEXITS load module at any time, but remember that ISPF loads the module only once at session initialization. Therefore, the changes are not recognized until the next ISPF session.
- Because ISPF loads the ISPEXITS load module only once, consider making any exit routines that you write re-usable, and preferably reentrant. If you write exits that are *not* reentrant, they cannot be put in the Link Pack Area (LPA) library. Non-reentrant exit routines placed in the LPA can cause abend errors.
- You cannot invoke an ISPF service from within the exit routines.
- The installation-written exit routines receive control in an addressing mode of AMODE=31. The exit routines must support 31-bit addressing. ISPF does not restrict the residency mode of the installation-wide exits.
- Input-output exit routines can modify parameters. If there are multiple exit routines, successive routines receive these parameters as modified by the previous routine. The first exit routine receives parameters as described.
- You can define a data area to be used by each exit routine. These data areas can be shared by different exit routines.
- At initialization, ISPF gets the storage for the data area. It provides the length and the address of the data area in the parameter list when calling an exit routine that uses that data area. (ISPF obtains the storage from subpool 0 below the 16 MB line.) The first time ISPF calls the exit routine, the data area contains binary zeros. The information in the data area is retained between invocations of the exit routines and the storage is available for the entire ISPF session (independent of logical screen abends and restarts). It is released at ISPF termination.
- Data areas are on double-word boundaries.

Related references

Chapter 10, "Exits," on page 255

How to install the installation-wide exits

To install an ISPF installation-wide exit:

1. Indicate in installation tailoring that ISPF installation-wide exits are defined. When you install ISPF, you indicate that exit routines might exist by setting the ENABLE_ISPF_EXITS keyword in the ISPF Configuration table to YES. As a result, you can replace, add, or remove exit routines by simply reassembling ISPXDT and relinking ISPEXITS. Otherwise, ISPF uses the default value of NO and does not load the ISPEXITS module. In this case, you cannot invoke any exit routines and you have to repeat part of the installation process if you want to change this value. See Chapter 2, "The ISPF Configuration Table," on page 9 for more information.
2. Use the exit macros to assemble the exit definition table (ISPXDT). This process is discussed in detail in "Exit macros" on page 255. These macros reside in ISP.SISPMACS.
3. Link-edit the ISPEXITS load module. ISPXDT is the main entry point of this module. There is no need to include any other ISPF-supplied CSECTS in ISPEXITS. Figure 56 on page 137 contains sample JCL to link-edit ISPEXITS.

```
//LKED EXEC PGM=IEWL,REGION=1024K,
// PARM='XREF,LET,LIST,RENT,SIZE=(512K,128K)'
//SYSPRINT DD SYSOUT=A
//SYSUT1 DD UNIT=SYSDA,SPACE=(TRK,(10,5))
//SYSLMOD DD DISP=SHR,DSN=ISP.SISPLOAD
//* All exit routines and ISPXDT CSECT must be in SYSLIB,
//* but an INCLUDE SYSLIB statement is only required for
//* ISPXDT and not for the exit routines.
//SYSLIB DD DSN=ISP.LOCOBJ,DISP=SHR
//SYSLIN DD *
   ORDER ISPXDT
   ENTRY ISPXDT
   INCLUDE SYSLIB(ISPXDT)
 NAME ISPEXITS(R)
/*
//
```

Figure 56. Sample JCL to Link-Edit ISPEXITS

4. ISPXDT does not contain any executable code. If your exit routines are reentrant, ISPEXITS can be copied into the LPA library. Otherwise, it must be copied into the system link library.

Related references

"Exit macros" on page 255

How to use the macros to define ISPXDT

To define ISPXDT using the exit macros:

1. Indicate the beginning of the exit entry definition section of ISPXDT by coding one ISPMXED macro with the START operand. For example:

```
ISPMXED START
⋮
```

2. Define the installation-wide exits you plan to use. Code the ISPMXLST macro, listing the numeric codes for each user exit in ascending order, enclosed within parentheses. This example shows that exit routines will be installed at ISPF installation-wide exits 3, 7, and 14.

```
ISPMXED START
ISPMXLST (3,7,14)
⋮
```

   If no operand is specified, a dummy ISPXDT is built.

3. For each user exit that you define on the ISPMXLST macro, use the ISPMXDEF, ISPMEPT, and ISPMXEND macros to define one or more exit routines. These three macros are a set and must be coded for each user exit you identify.

   - Code the ISPMXDEF macro with the exit ID as its operand.
   - Code the ISPMEPT macro for each exit routine and, if required, its data areas.
   - Code the ISPMXEND macro to end the exit routine definitions for this user exit.

   In this example, three exit routines are defined at user exit 7. The first two share a common data area (DAREA015) and the third uses a different data area (DAREA027). ISPF calls entry points EXPT7EP1, EXPT7EP2, and EXPT7EP3 in that order unless one of the routines returns a return code of 12 or greater. For example, if entry point EXPT7EP2 returns a return code of 12, control returns to ISPF without executing entry point EXPT7EP3. Therefore, if you define more than one exit routine at a given user exit, define them in the correct order of priority.

```
ISPMXED START
ISPMXLST (3,7,14)
⋮
ISPMXDEF 7
ISPMEPT EXPT7EP1,DAREA015
ISPMEPT EXPT7EP2,DAREA015
ISPMEPT EXPT7EP3,DAREA027
```

```
ISPMXEND
⋮
```

If you use the ISPMXDEF macro to define an exit ID that is not listed on the ISPMXLST macro, ISPF issues a warning message. You can still use ISPXDT, but the user exit is defined. As a result, you can easily disable a user exit by deleting its exit ID on the ISPMXLST macro. However, for every exit point listed on the ISPMXLST macro, there must be a corresponding ISPMXDEF macro.

4. Indicate the end of the exit entry definition section of ISPXDT by coding the END operand on the ISPMXED macro. For example:

```
ISPMXED START
ISPMXLST (3,7,14)
⋮
ISPMXED END
```

5. Define the data areas used by the exit routines in the exit data area definition section of ISPXDT. Indicate the start of this section by using the START operand on the ISPMXDD macro. For example:

```
⋮
ISPMXED END
ISPMXDD START
⋮
```

6. Use the ISPMDAD macro to define the name and size of each data area defined on an ISPMEPT macro. Every data area referenced on an ISPMEPT macro must be defined by an ISPMDAD macro.

   This example defines DAREA015 to be 100 bytes long. (ISPF rounds the length to 104 in this case.)

```
⋮
ISPMXDD START
ISPMDAD DAREA015,100
⋮
```

   If you use the ISPMDAD macro to define a data area that is not coded on an ISPMEPT macro, a warning message is issued during assembly and storage for the area is not obtained during execution. If none of your exit routines require a data area, you need not code an ISPMDAD macro, but you must still code ISPMXDD START and ISPMXDD END.

7. Use the ISPMXDD macro to indicate the end of the exit data area definition section. For example:

```
⋮
ISPMXDD START
ISPMDAD DAREA015,100
⋮
ISPMXDD END
```

Sample ISPXDT definition

Figure 57 on page 139 shows a sample ISPXDT definition that defines exit routines for installation-wide exits 2 and 12.

```
<<ISPMXED<START
*
ISPMXLST<(2,12)
*
ISPMXDEF<2
ISPMEPT<MYEXT021,MYAREA01
ISPMEPT<MYEXT022,MYAREA02
ISPMEPT<MYEXT023,MYAREA02
ISPMXEND
*
ISPMXDEF<12
ISPMEPT<MYEXT121,MYAREA03
ISPMEPT<MYEXT122,MYAREA01
ISPMEPT<MYEXT123
ISPMEPT<MYEXT124,MYAREA02
ISPMXEND
*
ISPMXED<END
*
ISPMXDD<START
*
ISPMDAD<MYAREA01,1024
ISPMDAD<MYAREA02,2048
ISPMDAD<MYAREA03,256
*
ISPMXDD<END
```

Figure 57. Sample ISPXDT definition

At user exit 2, exit routines with entry points MYEXT021, MYEXT022, and MYEXT023 are called, in that order.

At user exit 12, exit routines with entry points MYEXT121, MYEXT122, MYEXT123, and MYEXT124 are called, in that order.

MYAREA01 (1024 bytes) is used by exit routines at MYEXT021 and MYEXT122.

MYAREA02 (2048 bytes) is used by exit routines at MYEXT022, MYEXT023, and MYEXT124.

MYAREA03 (256 bytes) is used by the exit routine at MYEXT121.

Exit routine MYEXT123 does not require a data area.

Exit parameter list

Each of the ISPF exit routines is passed a parameter list that contains parameters common to all the installation-wide exits and parameters that are specific to the user exit. The parameters are in the order shown here, followed by exit-specific parameters in the order shown under the individual user exit description. The common parameters are:

exitid
A fullword binary number that identifies the numeric code for the user exit.

userid
An 8-character field that contains the TSO user ID (left-justified) of the current TSO session. If ISPF is running in batch and there is no TSO user ID, the DSN prefix (as set by the TSO PROFILE PREFIX command) is placed in the Userid field. If there is no DSN prefix, the Userid field contains the characters 'BATCH '.

screenid
An 8-character field that identifies the active ISPF logical screen. Because ISPF supports up to 32 logical screens, the possible values are 0 through 9 and A through W.

At initialization and termination exit points, 0 indicates no active logical screens.

These identifiers are left-justified and the field is padded with blanks.

ZENVIR
32 characters of environmental information that is provided by the ZENVIR ISPF system variable. This variable is described in *z/OS ISPF Reference Summary*.

datalen
A fullword binary number that identifies the length of the exit data area, in bytes.

dataptr
The fullword address of the data area ISPF acquires for the exit routine. If you do not provide a data area name on the ISPMEPT macro, this address, as well as the data length, is 0. Note that the data area is always on a doubleword boundary and data length is a multiple of eight.

Standard OS linkage conventions are followed. For example: Register 1 (which points to a list of addresses; each address points to a different parameter) is used to pass the parameter list to the exit program. For more information about linkage conventions, refer to *z/OS MVS Programming: Assembler Services Guide*.

Error processing

The installation-wide exit interface is a system programmer interface and, as such, no special error protection is provided for the exit routines. The ESTAE and ESTAI exit routines ISPF uses for abend recovery are used to recover from errors in the code for the user exit.

Abends within exit routines at these installation-wide exits cause ISPF to terminate:

- ISPF session initialization exit
- ISPF session termination exit
- Logical screen start exit
- Logical screen end exit
- SWAP exit

Abends within exit routines at any other user exit do not terminate ISPF, but do result in a logical screen restart, unless running in test mode or with ENBLDUMP on.

Chapter 7. Customizing PDF

Customizing PDF describes procedures you can use to customize PDF.

Installation options modify the distributed release of PDF to suit your installation's particular needs. Most of the installation options are in the configuration table described in Chapter 2, "The ISPF Configuration Table," on page 9. You make some of these modifications by editing the panel descriptions in the ISPF panel library (ISPPLIB data set).

All panels in the ISPPLIB data set are in ISPF panel format. All messages in the ISPMLIB data set are in ISPF message format. These formats are described in the *z/OS ISPF Dialog Developer's Guide and Reference*.

These topics describe how to modify the distributed release of PDF:

See also Chapter 8, "PDF installation-wide exits," on page 191.

Edit mode defaults

PDF saves several different edit modes in an edit profile. The user can specify the desired edit profile on the Edit Entry Panel. If the Profile field is left blank, the data set type is used as the profile name. For more information about edit profiles, refer to *z/OS ISPF Edit and Edit Macros*.

To preinitialize a set of edit profiles for first-time users, perform these steps:

1. Enter PDF.
2. Select the Edit option.
3. Set the edit profile with the defaults you chose.

   For example, to set "COBOL FIXED 80" in your profile, edit a member of a partitioned data set that has a RECFM of F or FB, a LRECL of 80, and a type qualifier of COBOL (or enter COBOL as the profile name on the Edit Entry Panel).

ISPF provides two methods for initializing new edit profiles; you can set up a profile called ZDEFAULT in the ISPTLIB concatenation, or you can modify the edit profile defaults in the ISPF configuration table. The ISPF configuration table method is recommended because it is easier to maintain than the ZDEFAULT method. The ZDEFAULT method can still be used by individual users.

Site-wide edit profile initialization

When no ZDEFAULT profile exists in the ISPTLIB concatenation and the user has no edit profile member in the ISPPROF concatenation, new edit profiles are created based on the settings in the ISPF configuration table. Using the configuration table, you can change any of the defaults for new edit profiles and you may override (force) settings for PACK, RECOVERY, WARN, SETUNDO, STATS, and IMACRO in existing profiles. When a setting is forced the editor WILL CHANGE the users' profiles, so be very careful if you override the IMACRO setting. IBM recommends that you use the site-wide initial macro instead of forcing the initial macro in each user's profile.

It is helpful to understand when the ZDEFAULT profile is used and where it exists in a user's concatenations. The ZDEFAULT profile exists as a row of the edit profile table named *xxx*EDIT where *xxx* is the application profile.

If ZDEFAULT exists in the edit profile table in the ISPTLIB concatenation, and the user has NO edit profile table in the ISPPROF allocation, the ZDEFAULT profile is copied from ISPTLIB into the user's edit profile when the user's edit profile is created. Therefore, many of your existing users may already have a ZDEFAULT profile in their edit profile. Individual users may delete their ZDEFAULT profiles using the PROFILE RESET command from within an edit session. Doing so will allow them to use the site-wide configuration for new profiles. You may also use a site-wide edit initial macro to issue a PROFILE RESET for all users. ISPF does not ship any edit profiles.

Note: If you use the force settings such as PACK OFF, edit macro commands which attempt to change forced settings will not get a failing return code, but the settings will not change.

Creating a ZDEFAULT profile

Set up a special edit profile named ZDEFAULT (enter ZDEFAULT as the profile name on the Edit Entry Panel). The ZDEFAULT profile is the one used for the initial settings whenever a new edit profile is generated, regardless of the RECFM and LRECL values. For example, if you do not have an ASM profile and you edit an ASM data set, an ASM profile is generated using ZDEFAULT for the initial settings. If no ZDEFAULT profile exists, one is automatically generated with settings obtained from the ISPF Configuration Table.

The number of profiles you can establish also is described in the configuration table. See Chapter 2, "The ISPF Configuration Table," on page 9 for more details. When you finish, exit PDF. Your entire set of edit profiles is saved in your profile library (referenced by ddname ISPPROF) as the ISREDIT member.

The previous discussion assumes you are using Edit from the Edit Entry Panel, which is option 2 of the ISPF Primary Option Menu. If Edit is invoked from another dialog or if this dialog is altered, Edit might be invoked using a NEWAPPL value other than ISR. If this is the case, the table name begins with the NEWAPPL ID rather than ISR. Therefore, you must create a new set of defaults for the NEWAPPL ID. Copy this member to the table input library (referenced by ddname ISPTLIB). When a first-time user enters ISPF, there is no ISREDIT member in that user's profile library. As a result, Edit searches the table input library for member ISREDIT and uses it as the initial set of profiles for the new user. No ISREDIT member is distributed with the ISPF table input library.

Action bars and extended color in Edit

Two ISPF configuration table keywords control whether action bars and highlighting are displayed in ISPF Edit. Disabling action bars provides more space to display data in the Edit panel. Disabling extended color support can improve system performance.

The keyword ALLOW_EDIT_HIGHLIGHTING can disable extended color support for all applications, including PDF itself and applications that use their own panels enabled for extended highlighting.

The DEFAULT_EDIT_DISPLAY keyword can be used to set the attributes of edit sessions invoked directly by PDF or by programs that invoke the edit service panel name other than ISREDDE2, ISREDDE3, ISREDDE4, or ISREDDE5. Using DEFAULT_EDIT_DISPLAY, you can make the edit session have:

- Neither action bars nor extended highlighting
- Extended highlighting with no action bars
- Action bars with no extended highlighting
- Both action bars and extended highlighting

DEFAULT_EDIT_DISPLAY can also be used to configure the editor to use the display method used in previous releases. This method does not support action bars or extended highlighting, but it performs much faster than the other methods. For more information, see "Disable Edit extended highlighting" on page 94.

If ALLOW_EDIT_HIGHLIGHTING is set to NO, the extended highlight support is disabled regardless of how DEFAULT_EDIT_DISPLAY is set.

See "Edit-related settings" on page 220 for more information about these keywords.

The following z Hilite variables represent the values of the active edit profile and are available for reference in the shared memory pool. These are ZHIAUTO, ZHILANG, ZHICOLOR, ZHIPAREN, ZHIFIND, and ZHICURSR. See *Dialog variables* in *z/OS ISPF Reference Summary* for more information.

Edit backup and recovery

Edit backup and recovery is controlled by two edit recovery tables. Table ISREDRT is used for PDF edit without the Edit Interface (EDIF) service. Table ISREIRT is used for PDF edit with the EDIF service. A copy of the tables is automatically saved in each user's profile library. The number of entries (rows) in each table controls the number of recursion levels supported for backup and recovery. CLIST ISREDRTI or ISREIRTI for the EDIF service, builds the edit recovery table (*aaaa*EDRT, where *aaaa* is the ZAPPLID) the first time edit recovery is used for a given user and application ID.

The default table allows eight levels of recursion. After the table is built, you can execute CLIST ISREDRTS or ISREIRTS for the Edit Interface service to display it. To change the default size of the table, change the 'SET N = ' statement in CLIST ISREDRTI or ISREIRTI to any number from 1 to 99.

When a user enters Edit mode and recovery mode is on, or when a user attempts to turn on recovery mode, Edit automatically allocates a recovery data set if there is an unused entry in the edit recovery table. The edit recovery data set name is generated and placed in the table. The names generated are ZPREFIX.ZUSER.*AAAAxxxx*.BACKUP, (where *AAAA* is the application ID and *xxxx* is a 4-digit number between 0000 and 9999) if ZPREFIX and ZUSER are different, and ZUSER.*AAAAxxxx*.BACKUP if ZPREFIX and ZUSER are the same.

The edit interface recovery table contains data set names with a last qualifier of BACKUPI instead of BACKUP. These data set names are passed to the data set name change exit, if one exists, and the installation can change the names, if desired. Whenever the data set name change exit changes the recovery data set name, the PDF-generated recovery data set name is deleted and reused. The data set name change exit should check the recovery table and generate a unique data set name. The modified name is placed in the recovery table. If there is no unused entry, a message is displayed indicating recovery mode is not available. The user can continue editing with recovery mode off.

When you enter Edit, the edit profile controls the initial setting of recovery mode. When you terminate Edit, the system automatically deletes the recovery data set and frees the corresponding entry in the edit recovery table.

These restrictions apply to edit recovery data sets:

- They must be allocated as sequential data sets of record format U.
- They cannot be striped, or striped and compressed data sets.
- They cannot be multivolume data sets.

Edit recovery will not delete edit recovery data sets listed in the edit recovery table when the table has a disposition field set to 'K' associated with the recovery data set name. This is an obsolete facility for preallocating edit recovery files or enforcing naming conventions for edit recovery files. Instead, use the data set name change exit for this purpose.

Data set allocation defaults for the Outlist utility

The Outlist utility (option 3.8) is available as either a CLIST (ISRUOL) or a program (ISRUOLP). The default used by ISPF is the program. To use the CLIST instead, perform these steps:

1. Modify panel ISRUTIL, changing:

```
        8, 'PGM(ISRUOLP)'
to
        8, 'CMD(ISRUOL)'
```

2. Modify the utilities action bar member (ISPDUTIL), changing:

```
       <pdc unavail=zut7 acc1=alt acc2=8>Outlist
           <action run=ISRROUTE parm=U8>
to
       <pdc unavail=zut7 acc1=alt acc2=8>Outlist
           <action run=ISRUOL type=CMD>
```

3. Use Option 3.9 to create a command table entry called ISRUOL, specifying this as the action:

```
        SELECT CMD(ISRUOL)
```

The Outlist utility invokes the TSO OUTPUT command to retrieve data from the SYSOUT queue. For the browse and print options of this utility, ISPF allocates a print data set and passes its name to the OUTPUT command by means of the PRINT parameter. The parameters in CLIST ISRUOL or the ISPF configuration table for program ISRUOLP determine the attributes of the print data set.

You can modify these print data set attributes to meet the needs of your installation. The default value for each attribute is shown in parentheses:

```
LRECL (133)
BLKSIZE (3059)
Primary space in tracks (200)
Secondary space in tracks (100)
```

To change the attributes for CLIST ISRUOL, edit these lines as shown:

```
ATTR SPFUOL1 BLKSIZE(3059) LRECL(133)+
DSORG(PS) RECFM(FB &ZR)
ALLOC DA('&DSN') TRACKS    /*                    */+
USING(SPFUOL1) RELEASE     /*                    */+
SPACE(200 100) CATALOG     /*                    */
```

To change the attributes for program ISRUOLP, modify the appropriate fields in the ISPF configuration table.

Using the Hardcopy utility with DBCS support

When double-byte character set (DBCS) data and field-ruling information are in a data set, you cannot use the print program provided with Hardcopy utility (option 3.6). You must modify the ISRUHCP panel and the ISRUHCS1 skeleton to print the contents of the data set.

Use the ISRUHCP panel to specify print information to ISPF. Also, include the printout destination. Modify the panel as follows:

1. Add these input fields to the)BODY section:
   - A field to indicate a user-supplied print program is going to execute
   - Fields to provide information to the user-supplied program.
2. Add this logic to the)PROC section:
   - After all input checking is finished, if the user-supplied program is requested and a local terminal ID is specified, copy the terminal ID to a variable and clear out the original variable. Set a dummy SYSOUT class, if not set already.
   - VPUT all newly defined variables to the profile pool.

For information on modifying the)BODY and)PROC sections of a panel, see the *z/OS ISPF Dialog Developer's Guide and Reference*.

The ISRUHCS1 skeleton uses IEBGENER to print the data set. Make these changes to this skeleton:

1. If the user-supplied print program is requested, modify the SYSUT2 DD statement from SYSOUT to a temporary data set. Modify the DCB information also.

2. If the user-supplied program is requested, add a job step for initiating the user-supplied program that prints the temporary data set. You can get all information for the user-supplied program from the variables in the profile pool.

SCLM Batch considerations

Before using the SCLM batch facility, modify the FLMLIBS skeleton to allow for batch submissions. FLMLIBS is found in the ISPF skeleton target data set ISP.SISPSLIB. It is the common imbed for the other SCLM skeletons used for batch submission. Modify the data set names in FLMLIBS to match your installation's naming conventions.

Note: In Figure 58 on page 146, the 'ISP' data set high-level qualifier represents your ISPF data sets. *xxx* corresponds to a national language as follows:

Language	**xxx**
US English	ENU (the default)
Uppercase English	ENP
Japanese	JPN

```
)CM
)CM THIS DEFINES THE STEPLIB AND ISPF LIBRARIES
)CM TO BE USED DURING SCLM BATCH OPERATIONS
)CM
)CM BE SURE TO INCLUDE THE LOAD LIBRARIES CONTAINING ISPF.
//*
//*******************************************************************
//* STEPLIB LIBRARIES
//*******************************************************************
//*
//STEPLIB  DD  DSN=ISP.SISPLPA,DISP=SHR
//         DD  DSN=ISP.SISPLOAD,DISP=SHR
//         DD  DSN=CEE.SCEERUN,DISP=SHR
//         DD  DSN=CEE.SCEERUN2,DISP=SHR
//*
//*******************************************************************
//* ISPF LIBRARIES
//*******************************************************************
//*
//ISPMLIB  DD  DSN=ISP.SISPMXXX,DISP=SHR ISPF MSGS
//*
//ISPSLIB  DD  DSN=ISP.SISPSXXX,DISP=SHR ISPF SKELS
//         DD  DSN=ISP.SISPSLIB,DISP=SHR ISPF SKELS
//*
//ISPPLIB  DD  DSN=ISP.SISPPXXX,DISP=SHR ISPF PANELS
//*
//ISPTLIB  DD  UNIT=&VIOUNIT;,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//             DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB),
//             DSN=&TABLESP                  TEMPORARY TABLE LIBRARY
//         DD  DSN=ISP.SISPTXXX,DISP=SHR ISPF TABLES
//*
//ISPTABL  DD  UNIT=&VIOUNIT;,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//             DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB),
//             DSN=&TABLESP                  TEMPORARY TABLE LIBRARY
//*
//ISPPROF  DD  UNIT=&VIOUNIT;,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//             DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB),
//             DSN=&TABLESP                  TEMPORARY TABLE LIBRARY
//*
//ISPLOG   DD  SYSOUT=*,
//             DCB=(LRECL=120,BLKSIZE=2400,DSORG=PS,RECFM=FB)
//*
//ISPCTL1  DD DISP=NEW,UNIT=VIO,SPACE=(CYL,(1,1)),
//            DCB=(LRECL=80,BLKSIZE=800,RECFM=FB)    TEMPORARY FILE
//*                                                  TAILORING DATASET
//*                                                           OW01230
//SYSTERM  DD SYSOUT=*
//*
//*----------------------------------------------------------------------
//* TEMPORARY CLIST CONTAINING COMMAND TO BE EXECUTED
//*----------------------------------------------------------------------
//SYSPROC  DD  DSN=&&&&CLIST&STEP,DISP=(OLD,DELETE)
//         DD  DSN=ISP.SISPCLIB,DISP=SHR          CLIST LIBRARY OW01230
//*
)CM
)CM 5647-A01 (C) COPYRIGHT IBM CORP 1989, 2005 */
```

Figure 58. Sample FLMLIBS skeleton

Creating PDF translation tables

Creating PDF translation tables describes how to create a set of PDF translation tables.

Sample assembler modules are included in the ISP.SISPSAMP sample library (members ISROWNTT and ISRAPLTT). Module ISROWNTT contains a complete set of translation tables for an English 3278/3279 terminal, and module ISRAPLTT contains a complete set of translation tables for the 3278/3276 APL terminals. Use these as an example of what a completed module should look like. You can modify the samples to suit your requirements, supplying the desired values for each of the translation tables.

PDF uses these translation tables:

- Valid data set name character translation table
- Invalid data set name character translation table
- Hexadecimal character translation table

- Numeric character translation table
- Alphanumeric character translation table
- Edit terminal output character translation table
- Generic string character translation table
- Generic string special character translation table
- Uppercase character translation table
- Lowercase character translation table.

The sample Assembler module includes all of these translation tables. Each translation table consists of 32 consecutive DC statements, where each DC statement consists of eight hexadecimal values. You are free to supply the desired 256 hexadecimal values that comprise each of the translation tables. The only exception to this is the generic string character translation table that consists of 32 consecutive DC statements, each consisting of one hexadecimal value. This table should not be modified. The address of each table is located at the start of the Assembler module. If a table is not used, the address for that table is set to 0 (for example: EDIP DC A(0)).

Related references

Chapter 11, “PDF translation tables,” on page 273

PDF Foreground and Batch customizing

With PDF you can customize the Foreground and Batch processing options. You can add and modify existing Foreground or Batch options, develop new primary options to provide an interface to the Foreground or Batch processing mechanisms, and modify the tutorial to reflect installation-developed modifications.

Attention: Do not try to customize PDF Foreground and Batch options unless you are a system programmer who is thoroughly familiar with ISPF.

Review the *z/OS ISPF Dialog Developer's Guide and Reference* for a description of panel and message definition formats, and also for specific requirements for selection panels (menus) and tutorial panels. The distributed panels provide examples of ISPF selection and tutorial panels. The ISPF Primary Option Menu is named ISR@PRIM and the first tutorial panel is named ISR00000.

The PDF Foreground and Batch options use selection and data entry panels in combination with CLISTs and file skeletons (Batch only). The specific requirements for foreground and batch panels, CLISTs, and skeletons are described in these sections.

Foreground processing panels and CLISTs

The Foreground processing option uses ISPF dialog management services. The following code block shows the Foreground selection panel definition. See the *z/OS ISPF Dialog Developer's Guide and Reference* for a general description of panel definition formats.

Note: Attribute characters have been replaced by blanks. Also, some of the ISPF-supplied CLISTs contain a specific library (such as SYS1.LINKLIB) on the program call. If the called program does not reside in that library, you might need to customize the CLIST.

```
)PANEL KEYLIST(ISRSAB,ISR)
)ATTR DEFAULT(...) FORMAT(MIX)             /* ISRFPA - ENGLISH - 7.5 */
 0B TYPE(AB)
 0D TYPE(PS)
 04 TYPE(ABSL) GE(ON)
 05 TYPE(PT)
 09 TYPE(FP)
 0A TYPE(NT)
 11 TYPE(SAC)
 22 TYPE(WASL) SKIP(ON) GE(ON)
 10 TYPE(ET)
 26 AREA(SCRL) EXTEND(ON)
 27 TYPE(CEF) PADC(USER) CKBOX(ON)
 28 TYPE(NEF) CAPS(ON) PADC(USER)
```

```
)ABC DESC('Menu') MNEM(1)
PDC DESC('Settings') UNAVAIL(ZPM1) MNEM(1) ACC(CTRL+S)
 ACTION RUN(ISRROUTE) PARM('SET')
PDC DESC('View') UNAVAIL(ZPM2) MNEM(1) ACC(CTRL+V)
 ACTION RUN(ISRROUTE) PARM('BR1')
PDC DESC('Edit') UNAVAIL(ZPM3) MNEM(1) ACC(CTRL+E)
 ACTION RUN(ISRROUTE) PARM('ED1')
PDC DESC('ISPF Command Shell') UNAVAIL(ZPM4) MNEM(6) ACC(CTRL+C)
 ACTION RUN(ISRROUTE) PARM('C1')
PDC DESC('Dialog Test...') UNAVAIL(ZPM5) MNEM(8) ACC(CTRL+T)
 ACTION RUN(ISRROUTE) PARM('DAL')
PDC DESC('Other IBM Products...') UNAVAIL(ZPM6) MNEM(1) ACC(CTRL+O)
 ACTION RUN(ISRROUTE) PARM('OIB')
PDC DESC('SCLM') UNAVAIL(ZPM7) MNEM(3) ACC(CTRL+L)
 ACTION RUN(ISRROUTE) PARM('SCL')
PDC DESC('ISPF Workplace') UNAVAIL(ZPM8) MNEM(6) ACC(CTRL+W)
 ACTION RUN(ISRROUTE) PARM('WRK')
PDC DESC('Status Area...') UNAVAIL(ZPMS) MNEM(8) ACC(CTRL+A)
 ACTION RUN(ISRROUTE) PARM('SAM')
PDC DESC('Exit') MNEM(2) PDSEP(ON) ACC(CTRL+X) ACTION RUN(EXIT)
)ABCINIT
.ZVARS=ISR@OPT
)ABC DESC('Utilities') MNEM(1)
PDC DESC('Library') UNAVAIL(ZUT1) MNEM(1) ACC(ALT+1)
 ACTION RUN(ISRROUTE) PARM('U1')
PDC DESC('Data set') UNAVAIL(ZUT2) MNEM(1) ACC(ALT+2)
 ACTION RUN(ISRROUTE) PARM('U2')
PDC DESC('Move/Copy') UNAVAIL(ZUT3) MNEM(1) ACC(ALT+3)
 ACTION RUN(ISRROUTE) PARM('U3')
PDC DESC('Data Set List') UNAVAIL(ZUT4) MNEM(2) ACC(ALT+4)
 ACTION RUN(ISRROUTE) PARM('U4')
PDC DESC('Reset Statistics') UNAVAIL(ZUT5) MNEM(5) ACC(ALT+5)
 ACTION RUN(ISRROUTE) PARM('U5')

PDC DESC('Hardcopy') UNAVAIL(ZUT6) MNEM(8) ACC(ALT+6)
 ACTION RUN(ISRROUTE) PARM('U6')
PDC DESC('Reserved') UNAVAIL(ZUTDT) MNEM(1) ACTION RUN(ISRROUTE) PARM('UDT')
PDC DESC('Outlist') UNAVAIL(ZUT7) MNEM(1) ACC(ALT+8)
 ACTION RUN(ISRROUTE) PARM('U8')
PDC DESC('Commands...') UNAVAIL(ZUT8) MNEM(1) ACC(ALT+9)
 ACTION RUN(ISRROUTE) PARM('U9')
PDC DESC('Reserved') UNAVAIL(ZUT9) MNEM(6) ACTION RUN(ISRROUTE) PARM('U10')
PDC DESC('Format') UNAVAIL(ZUT10) MNEM(1) ACC(ALT+F1)
 ACTION RUN(ISRROUTE) PARM('U11')
PDC DESC('SuperC') UNAVAIL(ZUT11) MNEM(1) PDSEP(ON) ACC(CTRL+F2)
 ACTION RUN(ISRROUTE) PARM('U12')
```

```
PDC DESC('SuperCE') UNAVAIL(ZUT12) MNEM(2) ACC(CTRL+F3)
 ACTION RUN(ISRROUTE) PARM('U13')
PDC DESC('Search-For') UNAVAIL(ZUT13) MNEM(2) ACC(CTRL+F4)
 ACTION RUN(ISRROUTE) PARM('U14')
PDC DESC('Search-ForE') UNAVAIL(ZUT14) MNEM(6) ACC(CTRL+F5)
 ACTION RUN(ISRROUTE) PARM('U15')
PDC DESC('Table Utility') UNAVAIL(ZUT15) MNEM(3) ACC(CTRL+F6)
 ACTION RUN(ISRROUTE) PARM('U16')
PDC DESC('Directory List') UNAVAIL(ZUT16) MNEM(2) ACC(CTRL+F7)
 ACTION RUN(ISRROUTE) PARM('U17')
)ABCINIT
.ZVARS=PDFUTIL
&zutdt = '1'
&zut9 = '1'
)ABC DESC('Help') MNEM(1)
PDC DESC('General') MNEM(1) ACTION RUN(TUTOR) PARM('ISR40010')
PDC DESC('Assembler') MNEM(1) ACTION RUN(TUTOR) PARM('ISR41000')
PDC DESC('COBOL') MNEM(1) ACTION RUN(TUTOR) PARM('ISR42000')
PDC DESC('VS Fortran') MNEM(4) ACTION RUN(TUTOR) PARM('ISR43000')
PDC DESC('PL/I') MNEM(2) ACTION RUN(TUTOR) PARM('ISR45000')
PDC DESC('VS Pascal') MNEM(4) ACTION RUN(TUTOR) PARM('ISR46000')
PDC DESC('Binder/Link editor') MNEM(1) ACTION RUN(TUTOR) PARM('ISR47000')
PDC DESC('SCRIPT VS') MNEM(1) ACTION RUN(TUTOR) PARM('ISR49000')
PDC DESC('VS COBOL II debug') MNEM(1) ACTION RUN(TUTOR) PARM('ISR4AA00')
PDC DESC('OS/VS COBOL debug') MNEM(1) ACTION RUN(TUTOR) PARM('ISR4A000')
PDC DESC('FORTRAN debug') MNEM(7) ACTION RUN(TUTOR) PARM('ISR4B000')
PDC DESC('Member parts list') MNEM(1) ACTION RUN(TUTOR) PARM('ISR4C000')
PDC DESC('C/370') MNEM(3) ACTION RUN(TUTOR) PARM('ISR4D000')
PDC DESC('REXX/370') MNEM(1) ACTION RUN(TUTOR) PARM('ISR4E000')
PDC DESC('ADA/370') MNEM(2) ACTION RUN(TUTOR) PARM('ISR4F000')
PDC DESC('AD/Cycle C/370') MNEM(5) ACTION RUN(TUTOR) PARM('ISR4G000')
PDC DESC('ISPDTLC') MNEM(5) ACTION RUN(TUTOR) PARM('ISR4I000')
PDC DESC('OS/390 C/C++') MNEM(5) ACTION RUN(TUTOR) PARM('ISR4J000')
```

```
)ABCINIT
.ZVARS=FPAHELP
)BODY  CMD(ZCMD)
.. Menu. Utilities. Help.
.----------------------------------------------------------------------
.                          .Foreground Selection Panel.
.Option ===>.Z
.SAREA38
)AREA SAREA38
.1  . Assembler             .          ..11 .*FORTRAN debug      .
.2  . COBOL                 .          ..12 . Member Parts List  .
.3  . VS FORTRAN            .          ..13 .*C/370              .
.5  . PL/I                  .          ..14 .*REXX/370           .
.6  . VS PASCAL             .          ..15 .*ADA/370            .
.7  .*Binder/Link editor    .          ..16 .*AD/Cycle C/370     .
.9  . SCRIPT/VS             .          ..18 . ISPDTLC            .
.10 .*VS COBOL II debug     .          ..19 .*OS/390 C/C++       .
.10A.*OS/VS COBOL debug     .
.. .
.  ..&multipmt                   ..*.No packed data support.             .
.  ..Z..Source Data Packed              .
)INIT
.ZVARS = '(ZCMD ZFPKEDV)'
&ZWINTTL = ' '
.HELP = ISR40000
&ZFPKEDV = ' '
  &ZFPKED = TRANS(TRUNC(&ZFPKED,1),Y,YES,*,NO) /*  DATA FORMAT CHECK  */
  &ZFPKEDV = TRANS(&ZFPKED YES,'/' NO,' ')
IF (&ZGUI = ' ')
  &MULTIPMT='Enter "/" to select option    '
ELSE
  &MULTIPMT='Check box to select option    '
.CURSOR = 'ZCMD'
```

```
)REINIT
REFRESH(ZFPKEDV)
)PROC
  &DSN     = ' '             /* INITIALIZE DATA SET NAME FIELD            */
  &ZORG    = ' '             /* INITIALIZE DATA SET ORGANIZATION VARIABLE */
  IF (&ZFPKEDV = ' ')
    &ZFPKED = 'NO'
  ELSE
    &ZFPKED = 'YES'
  &ZFPKED  = TRUNC(&ZFPKED,1)
  VER (&ZFPKED,NB,LIST,Y,N)               /*  Y = EXPAND PACKED DATA  */
  &ZFPKED = TRANS(TRUNC(&ZFPKED,1),Y,YES,N,NO)
  &ZFPACK = TRANS(TRUNC(&ZFPKED,1),Y,YES,N,NO)
  VPUT (ZFPACK,ZFPKED) PROFILE
&ZCMDWRK = &Z
IF (&ZCMD ¬= &Z)
  &ZCMDWRK = TRUNC(&ZCMD,'.')
  &ZTRAIL=.TRAIL
  IF (&ZCMDWRK = &Z)
    .MSG = ISRU000
&ZSEL = TRANS (TRUNC (&ZCMD,'.')
  1,'PGM(ISRFPR) PARM((ISRFP01) 1) NEWPOOL'
  2,'PGM(ISRFPR) PARM((ISRFP02) 2) NEWPOOL'
  3,'PGM(ISRFPR) PARM((ISRFP03) 3) NEWPOOL'
  5,'PGM(ISRFPR) PARM((ISRFP05) 5) NEWPOOL'
  6,'PGM(ISRFPR) PARM((ISRFP06) 6) NEWPOOL'
  7,'PGM(ISRFPR) PARM((ISRFP07) 7) NEWPOOL'
  9,'PGM(ISRFPR) PARM((ISRFP09) 9) NEWPOOL'
 10,'PGM(ISRFPR) PARM((IGZTPIN2,ISRFP10) 10) NEWPOOL'
10A,'PGM(ISRFPR) PARM((ISRFP10A) 10A) NEWPOOL'
 11,'PGM(ISRFPR) PARM((AFFFP11,DDBFP11,ISRFP11) 11) NEWPOOL'
 12,'PGM(ISRFPR) PARM((ISRFP12) 12) NEWPOOL'
 13,'PGM(ISRFPR) PARM((EDCFP13,ISRFP13) 13) NEWPOOL'
 14,'PGM(ISRFPR) PARM((FANFP14,ISRFP14) 14) NEWPOOL'
 15,'PGM(ISRALTDI) PARM(EVGFP15,,ISRFP15,*) NOCHECK'
 16,'PGM(ISRFPR) PARM((EDCFP16,ISRFP16) 16) NEWPOOL'
 18,'CMD(ISPDTLC (PANEL RETURN)) MODE(FSCR)'
 19,'PGM(ISRALTDI) PARM(CBC3PE4A,+,ISRFP19,*,ISRFP19A) NEWPOOL'
 ' ',' '
   *,'?')
)PNTS
FIELD(ZPS01001) VAR(ZCMD) VAL(1)
FIELD(ZPS01002) VAR(ZCMD) VAL(11)
FIELD(ZPS01003) VAR(ZCMD) VAL(2)
FIELD(ZPS01004) VAR(ZCMD) VAL(12)
FIELD(ZPS01005) VAR(ZCMD) VAL(3)
FIELD(ZPS01006) VAR(ZCMD) VAL(13)
FIELD(ZPS01007) VAR(ZCMD) VAL(5)
FIELD(ZPS01008) VAR(ZCMD) VAL(14)
FIELD(ZPS01009) VAR(ZCMD) VAL(6)
FIELD(ZPS01010) VAR(ZCMD) VAL(15)
FIELD(ZPS01011) VAR(ZCMD) VAL(7)
FIELD(ZPS01012) VAR(ZCMD) VAL(16)
FIELD(ZPS01013) VAR(ZCMD) VAL(9)
FIELD(ZPS01014) VAR(ZCMD) VAL(18)
FIELD(ZPS01015) VAR(ZCMD) VAL(10)
FIELD(ZPS01016) VAR(ZCMD) VAL(19)
FIELD(ZPS01017) VAR(ZCMD) VAL(10A)
)END

 /* 5694-A01 COPYRIGHT IBM CORP 1980, 2011 */ /* ISPDTLC Release: 7.5.  Level:
PID                                  */ /*  z/OS 02.05.00.  Created - Date: 18 Feb 2020, Time:
08:48          */
```

Each Foreground option has an associated panel and CLIST. For example, option 1 has the option panel name ISRFP01 with the corresponding CLIST name ISRFC01. Foreground Assembler definition (ISRFP01) shows the ISRFP01 option panel definition.

Note: In Foreground Assembler definition (ISRFP01), attribute characters have been replaced by blanks.

Foreground Assembler definition (ISRFP01)

```
)PANEL KEYLIST(ISRSAB,ISR)
)ATTR DEFAULT(...) FORMAT(MIX)             /* ISRFP01 - ENGLISH - 7.5 */
 0B TYPE(AB)
 04 TYPE(ABSL) GE(ON)
 05 TYPE(PT)
 09 TYPE(FP)
 0A TYPE(NT)
 0C TYPE(NT) SKIP(ON)
 11 TYPE(SAC)
 12 TYPE(CEF) PADC(USER)
 19 TYPE(DT)
 22 TYPE(WASL) SKIP(ON) GE(ON)
 08 TYPE(CH)
 26 TYPE(NEF) CAPS(ON) PADC(USER)
 27 AREA(SCRL) EXTEND(ON)
 28 TYPE(SAC) CSRGRP(99) RADIO(ON)
)ABC DESC('Menu') MNEM(1)
PDC DESC('Settings') UNAVAIL(ZPM1) MNEM(1) ACC(CTRL+S)
 ACTION RUN(ISRROUTE) PARM('SET')
PDC DESC('View') UNAVAIL(ZPM2) MNEM(1) ACC(CTRL+V)
 ACTION RUN(ISRROUTE) PARM('BR1')
PDC DESC('Edit') UNAVAIL(ZPM3) MNEM(1) ACC(CTRL+E)
 ACTION RUN(ISRROUTE) PARM('ED1')
PDC DESC('ISPF Command Shell') UNAVAIL(ZPM4) MNEM(6) ACC(CTRL+C)
 ACTION RUN(ISRROUTE) PARM('C1')
PDC DESC('Dialog Test...') UNAVAIL(ZPM5) MNEM(8) ACC(CTRL+T)
 ACTION RUN(ISRROUTE) PARM('DAL')
PDC DESC('Other IBM Products...') UNAVAIL(ZPM6) MNEM(1) ACC(CTRL+O)
 ACTION RUN(ISRROUTE) PARM('OIB')
PDC DESC('SCLM') UNAVAIL(ZPM7) MNEM(3) ACC(CTRL+L)
 ACTION RUN(ISRROUTE) PARM('SCL')
PDC DESC('ISPF Workplace') UNAVAIL(ZPM8) MNEM(6) ACC(CTRL+W)
 ACTION RUN(ISRROUTE) PARM('WRK')
PDC DESC('Status Area...') UNAVAIL(ZPMS) MNEM(8) ACC(CTRL+A)
 ACTION RUN(ISRROUTE) PARM('SAM')
PDC DESC('Exit') MNEM(2) PDSEP(ON) ACC(CTRL+X) ACTION RUN(EXIT)
)ABCINIT
.ZVARS=ISR@OPT
)ABC DESC('RefList') MNEM(1)
PDC DESC('Current Data Set List &ZDSCURT') MNEM(1) ACC(CTRL+ALT+P)
 ACTION RUN(ISRRLIST) PARM('PL1')
PDC DESC('Current Library List &ZDSCURLT') MNEM(2) ACC(CTRL+SHIFT+P)
 ACTION RUN(ISRRLIST) PARM('LL1')
PDC DESC('List of Personal Data Set Lists') MNEM(1) PDSEP(ON) ACC(CTRL+ALT+O)
 ACTION RUN(ISRRLIST) PARM('PL2')
PDC DESC('List of Personal Library Lists') MNEM(2) ACC(CTRL+SHIFT+O)
 ACTION RUN(ISRRLIST) PARM('LL2')
```

```
)ABCINIT
.ZVARS=REFLIST
      VGET (ZCURTB ZCURLTB) PROFILE
      IF (&ZCURTB = &Z) &ZDSCURT = &Z
      ELSE &ZDSCURT= '(&ZCURTB)'
```

```
       IF (&ZCURLTB = &Z) &ZDSCURLT = &Z
       ELSE &ZDSCURLT= '(&ZCURLTB)'
)ABC DESC('Utilities') MNEM(1)
PDC DESC('Library') UNAVAIL(ZUT1) MNEM(1) ACC(ALT+1)
 ACTION RUN(ISRROUTE) PARM('U1')
PDC DESC('Data set') UNAVAIL(ZUT2) MNEM(1) ACC(ALT+2)
 ACTION RUN(ISRROUTE) PARM('U2')
PDC DESC('Move/Copy') UNAVAIL(ZUT3) MNEM(1) ACC(ALT+3)
 ACTION RUN(ISRROUTE) PARM('U3')
PDC DESC('Data Set List') UNAVAIL(ZUT4) MNEM(2) ACC(ALT+4)
 ACTION RUN(ISRROUTE) PARM('U4')
PDC DESC('Reset Statistics') UNAVAIL(ZUT5) MNEM(5) ACC(ALT+5)
 ACTION RUN(ISRROUTE) PARM('U5')

PDC DESC('Hardcopy') UNAVAIL(ZUT6) MNEM(8) ACC(ALT+6)
 ACTION RUN(ISRROUTE) PARM('U6')
PDC DESC('Reserved') UNAVAIL(ZUTDT) MNEM(1) ACTION RUN(ISRROUTE) PARM('UDT')
PDC DESC('Outlist') UNAVAIL(ZUT7) MNEM(1) ACC(ALT+8)
 ACTION RUN(ISRROUTE) PARM('U8')
PDC DESC('Commands...') UNAVAIL(ZUT8) MNEM(1) ACC(ALT+9)
 ACTION RUN(ISRROUTE) PARM('U9')
PDC DESC('Reserved') UNAVAIL(ZUT9) MNEM(6) ACTION RUN(ISRROUTE) PARM('U10')
PDC DESC('Format') UNAVAIL(ZUT10) MNEM(1) ACC(ALT+F1)
 ACTION RUN(ISRROUTE) PARM('U11')
PDC DESC('SuperC') UNAVAIL(ZUT11) MNEM(1) PDSEP(ON) ACC(CTRL+F2)
 ACTION RUN(ISRROUTE) PARM('U12')
PDC DESC('SuperCE') UNAVAIL(ZUT12) MNEM(2) ACC(CTRL+F3)
 ACTION RUN(ISRROUTE) PARM('U13')
PDC DESC('Search-For') UNAVAIL(ZUT13) MNEM(2) ACC(CTRL+F4)
 ACTION RUN(ISRROUTE) PARM('U14')
PDC DESC('Search-ForE') UNAVAIL(ZUT14) MNEM(6) ACC(CTRL+F5)
 ACTION RUN(ISRROUTE) PARM('U15')
PDC DESC('Table Utility') UNAVAIL(ZUT15) MNEM(3) ACC(CTRL+F6)
 ACTION RUN(ISRROUTE) PARM('U16')
PDC DESC('Directory List') UNAVAIL(ZUT16) MNEM(2) ACC(CTRL+F7)
 ACTION RUN(ISRROUTE) PARM('U17')
)ABCINIT
.ZVARS=PDFUTIL
&zutdt = '1'
&zut9 = '1'
)ABC DESC('Help') MNEM(1)
PDC DESC('Input Data Set') MNEM(1) ACTION RUN(TUTOR) PARM('ISR41001')
PDC DESC('Macro and copy libraries') MNEM(1) ACTION RUN(TUTOR) PARM('ISR41002')
PDC DESC('Object Data Set') MNEM(1) ACTION RUN(TUTOR) PARM('ISR41003')
PDC DESC('Listing Data Set') MNEM(1) ACTION RUN(TUTOR) PARM('ISR41004')
PDC DESC('Password protection') MNEM(1) ACTION RUN(TUTOR) PARM('ISR41005')
PDC DESC('Assembler selection') MNEM(1) ACTION RUN(TUTOR) PARM('ISR41006')
PDC DESC('Example') MNEM(1) ACTION RUN(TUTOR) PARM('ISR41007')
PDC DESC('Appendices') MNEM(5) ACTION RUN(TUTOR) PARM('ISR00004')
PDC DESC('Index') MNEM(3) ACTION RUN(TUTOR) PARM('ISR91000')
)ABCINIT
.ZVARS=FP1HELP
```

```
)BODY  CMD(ZCMD)
.. Menu. RefList. Utilities. Help.
.-------------------------------------------------------------------------------
.                            .Foreground Assembler.                            .
.Command ===>.Z                                                                .
.SAREA39                                                                       .
)AREA SAREA39
.ISPF Library:.                                                                .
. ..Project . . ..Z       .
. ..Group . . . ..Z       .. . ..Z       .. . ..Z       .. . ..Z       .
. ..Type  . . . ..Z       .
. ..Member  . . ..Z       ..(Blank or pattern for member selection list).
.
.Other Partitioned or Sequential Data Set:.                                    .
. ..Data Set Name  . ..Z                                                       .
.. .
.List ID . . ..Z       .   .Assembler.
.Password  . ..Z       .   .Z..1..High Level Assembler..2..Assembler H         .
.
.Assembler Options: (Options OBJECT and LIST generated automatically).         .
. ..    ===>.Z                                                            .
.
.Additional input libraries:.                                                  .
. ..    ===>.Z                                                                .
. ..    ===>.Z                                                                .
. ..    ===>.Z                                                                .
)INIT
.ZVARS = '(ZCMD PRJ1 LIB1 LIB2 LIB3 LIB4 TYP1 MEM DSN LID PSWD ZASMOPT FHASM +
            FHAL1 FHAL2 FHAL3)'
.HELP = ISR41000
.ATTR(ZASMOPT)='CSRGRP(99) RADIO(ON)'
.ATTR(PSWD)='INTENS(NON)'
  &ZUT6  = 1
  &ZUT7  = 1
  &ZUT11 = 1
  &ZUT12 = 1
  &ZUT13 = 1
  &ZUT14 = 1
  &ZMLCSR = '        '                   /*                       @M1A*/
  .HELP = ISR41A00
  &TYP1 = &ASMT                         /*ASSEMBLER TYPE VARIABLE NAME*/
  IF (&ZORG = 'PS')
    IF (&LID = ' ')      .CURSOR = LID
  IF (&DSN ¬= ' ')
    &MEM = ' '                           /*                       @M1A*/
    IF (.CURSOR = ' ')   .CURSOR = DSN
  &LID  = ' '
  IF (&ZFTEMP = '')
    &ZFPACK = &ZFPACK
    VPUT (ZFPACK) SHARED
    &ZFTEMP = '0'
  IF (&ZASMOPT ¬= 1)
    IF (&ZASMOPT ¬= 2)
      &ZASMOPT = 1
```

```
)REINIT
REFRESH(PRJ1 LIB1 LIB2 LIB3 LIB4 TYP1 MEM DSN FHAL1 FHAL2 FHAL3)
IF (&ZNXTMSG='ISRT') .CSRPOS = &ZCSRP  /* AUTOTYPE */
                     .CURSOR = &ZCSRV        /* AUTOTYPE */
ELSE &ZXZX = &Z                         /* AUTOTYPE */
&ZUT6  = 1
&ZUT7  = 1
&ZUT11 = 1
&ZUT12 = 1
&ZUT13 = 1
&ZUT14 = 1
IF (&ZMLCSR ¬= ' ')                          /*                       @M1A*/
  .CURSOR = &ZMLCSR                          /*                       @M1A*/
)PROC
&ZCSRV = .CURSOR                             /* AUTOTYPE */
&ZCSRP = .CSRPOS                             /* AUTOTYPE */
&ZODSNLN = 0                                 /* AUTOTYPE */
&ZODSNMB = &Z                          /* AUTOTYPE */
&ZNAMES='ZCSRV ZCSRP PRJ1 LIB1 LIB2 LIB3 LIB4 TYP1 MEM '
&ZNAMES='&ZNAMES *.&ZODSNLN&ZODSNMB ZCMD'
IF (.CURSOR = DSN, FHAL1, FHAL2, FHAL3)
  &ZODSNLN = 56
  &ZODSNMB = &Z
  IF (.CURSOR = FHAL1) &ZODSNLN = 65
  IF (.CURSOR = FHAL1) &ZODSNMB = '%'
  IF (.CURSOR = FHAL2) &ZODSNLN = 65
  IF (.CURSOR = FHAL2) &ZODSNMB = '%'
  IF (.CURSOR = FHAL3) &ZODSNLN = 65
  IF (.CURSOR = FHAL3) &ZODSNMB = '%'
  &ZNAMES='ZCSRV ZCSRP * * * * * * * &ZCSRV&ZODSNLN&ZODSNMB ZCMD'
PANEXIT((ZNAMES),LOAD,ISRAUTOT)                 /* AUTOTYPE */
IF (&ZNXTMSG='ISRT') EXIT                      /* AUTOTYPE */
VER(&ZASMOPT,NONBLANK)
VER(&ZASMOPT RANGE,1,2)
&ZUT6  = 0
&ZUT7  = 0
&ZUT11 = 0
&ZUT12 = 0
&ZUT13 = 0
&ZUT14 = 0
 VGET (ZRDSN) SHARED                           /* REFERENCE LIST CODE        */
 IF (&ZRDSN ¬= ' ')                            /* IF DATA SET SELECTED       */
   &DSN = &ZRDSN                               /*    PUT DSN VARIABLE        */
   &ZRDSN = ' '                                /*     INTO PANEL             */
   &ZRVOL = ' '                                /*     INTO PANEL             */
   VPUT (ZRDSN ZRVOL) SHARED                   /*                            */
   .CURSOR = DSN
   .MSG = ISRDS003                             /*    MSG PENDING             */
```

```
VGET (DSALSEL) SHARED                         /*                           */
 IF (&DSALSEL ¬= ' ')                          /* IF LIBRARY SELECTED        */
   VGET (DSA1,DSA2,DSA3,DSA4,DSA5,DSA6,DSA7) SHARED
   &PRJ1 = &DSA1                               /*    PUT LIBRARY VARIABLES   */
```

```
    &LIB1 = &DSA2                               /*      INTO PANEL            */
    &LIB2 = &DSA3                               /*         .                  */
    &LIB3 = &DSA4                               /*         .                  */
    &LIB4 = &DSA5                               /*         .                  */
    &TYP1 = &DSA6                               /*         .                  */
    &MEM  = &DSA7                               /*         .                  */
    &DSN = ' '                                  /*    BLANK OUT DSN           */
    &DSALSEL = ' '                         /*    CLEAR LIBRARY SELECTION */
    VPUT (DSALSEL) SHARED                       /*                            */
    .CURSOR = MEM
    .MSG = ISRDS003                             /*    MSG PENDING             */
                                                /*                            */
  IF (&ZCMD ¬= ' ')  .MSG = ISPZ001             /* INVALID COMMAND            */
  IF (&DSN = ' ')                               /* IF NOT OTHER DATA SET,     */
    VER (&PRJ1,NB)                              /* VERIFY LIBRARY FIELDS ARE  */
    VER (&LIB1,NB)                              /* INPUT                      */
    VER (&TYP1,NB)
  IF (&DSN ¬= ' ')                              /*                        @M1A*/
    VER(&DSN DSNAMEFM)
  IF (&LID ¬= '*')  VER (&LID,NAME)             /* LIST ID MUST BE VALID NAME */
  VER (&PSWD,INCLUDE,ALPHA,NUM,MSG=ISRC609)     /* Check password     @V9A*/
  IF (&ZASMOPT = 1)
    &FHALEV = 'HLASM'
  IF (&ZASMOPT = 2)
    &FHALEV = 'HASM'
  IF (&FHAL1 ¬= ' ')                            /*                        @M2A*/
    VER (&FHAL1,DSNAMEPQ)                       /* VERIFY ADDITIONAL LIB1@DSNQ*/
  IF (&FHAL2 ¬= ' ')                            /*                        @M2A*/
    VER (&FHAL2,DSNAMEPQ)                       /* VERIFY ADDITIONAL LIB2@DSNQ*/
  IF (&FHAL3 ¬= ' ')                            /*                        @L2A*/
    VER (&FHAL3,DSNAMEPQ)                       /* VERIFY ADDITIONAL LIB2@DSNQ*/
  &ASMT = &TYP1                                 /* SAVE ASSEMBLER TYPE        */
  &ZSEL = 'CMD(%ISRFC01)'                       /* EXECUTE ASSEMBLE CLIST     */
                                                /* Begin @L3A                 */
  &ZSYSDS1 = &FHAL1                             /* Fill fields for ISRJFSYS   */
  &ZSYSDS2 = &FHAL2                             /* to use as input.           */
  &ZSYSDS3 = &FHAL3                             /* ZSYSDS? is a qualified     */
  &ZSYSCUR1 = 'FHAL1'                           /*     dataset.               */
  &ZSYSCUR2 = 'FHAL2'                           /* ZSYSCUR? is were the cursor*/
  &ZSYSCUR3 = 'FHAL3'                           /*     is placed on a error.  */
  VPUT (ZSYSDS1 ZSYSDS2 ZSYSDS3 ZSYSCUR1 ZSYSCUR2 ZSYSCUR3) SHARED
                                                /* End   @L3A                 */
  VPUT (PRJ1,LIB1,LIB2,LIB3,LIB4,ASMT,FHASM) PROFILE        /* OY14824*/
  VPUT (FHAL1,FHAL2,FHAL3,FHALEV,ZASMOPT) PROFILE
  VPUT (PRJ1,LIB1,LIB2,LIB3,LIB4,ASMT,FHASM,DSN,LID) SHARED /*    @L2C*/
  &ZFBROWS = 'ISRBROB ' /*BROWSE LISTING   , IF BLANK NO AUTO BROWSE  */
  &ZFPRINT = 'ISRFPPRT' /*PRINT PANEL NAME, IF BLANK NO AUTO PRINT PNL*/
  VPUT (ZFBROWS,ZFPRINT,FHAL1,FHAL2,FHAL3,FHALEV) SHARED    /* OW10516*/
 )END

  /* 5694-A01 COPYRIGHT IBM CORP 1980, 2011                         */ /* ISPDTLC Release: 7.5.
 Level: PID                                 */ /* z/OS 02.05.00.  Created - Date: 18 Feb 2020,
 Time: 06:36           */ /* OW21977 - 960813 - OS/390 R2 ROLLUP APAR - OW19891, OW20382
 */ /* OW10516 - 950113 - Add High Level Assembler.  GT4045 - MOS         */
```

Table 25 on page 153 lists the names of the ISPF-supplied panels and CLISTs for the Foreground processing option.

Table 25. ISPF-supplied panels and CLISTs for foreground processing option

Option	Description	Panel ID	CLIST ID
-	FOREGROUND SELECTION MENU	ISRFPA	--
1	ASSEMBLER	ISRFP01	ISRFC01
2	VS COBOL II	ISRFP02	ISRFC02
3	FORTRAN COMPILE	ISRFP03	ISRFC03
5	PLI OPTIMIZER COMPILE	ISRFP05	ISRFC05
6	VS PASCAL COMPILE	ISRFP06	ISRFC06
7	BINDER/LINK EDIT	ISRFP07	ISRFC07

Table 25. ISPF-supplied panels and CLISTs for foreground processing option (continued)

Option	Description	Panel ID	CLIST ID
9	SCRIPT/VS	ISRFP09	ISRFC09
10	VS COBOL II DEBUG	--	--
10A	COBOL INTERACTIVE DEBUG	ISRFP10A	ISRFC10A
11	FORTRAN INTERACTIVE DEBUG	ISRFP11	ISRFC11
12	MEMBER PARTS LIST	ISRFP12	ISRFC12
13	C/370 COMPILE	--	--
14	REXX/370 COMPILE	--	--
15	ADA/370 COMPILE	--	--
16	AD/CYCLE C/370 COMPILE	--	--
17	AD/CYCLE C/370 COBOL/370	--	--
18	ISPDTLC	ISPCP01	--
19	OS/390 C/C++	--	--

Figure 59 on page 155 shows the overall flow of control for foreground processing.

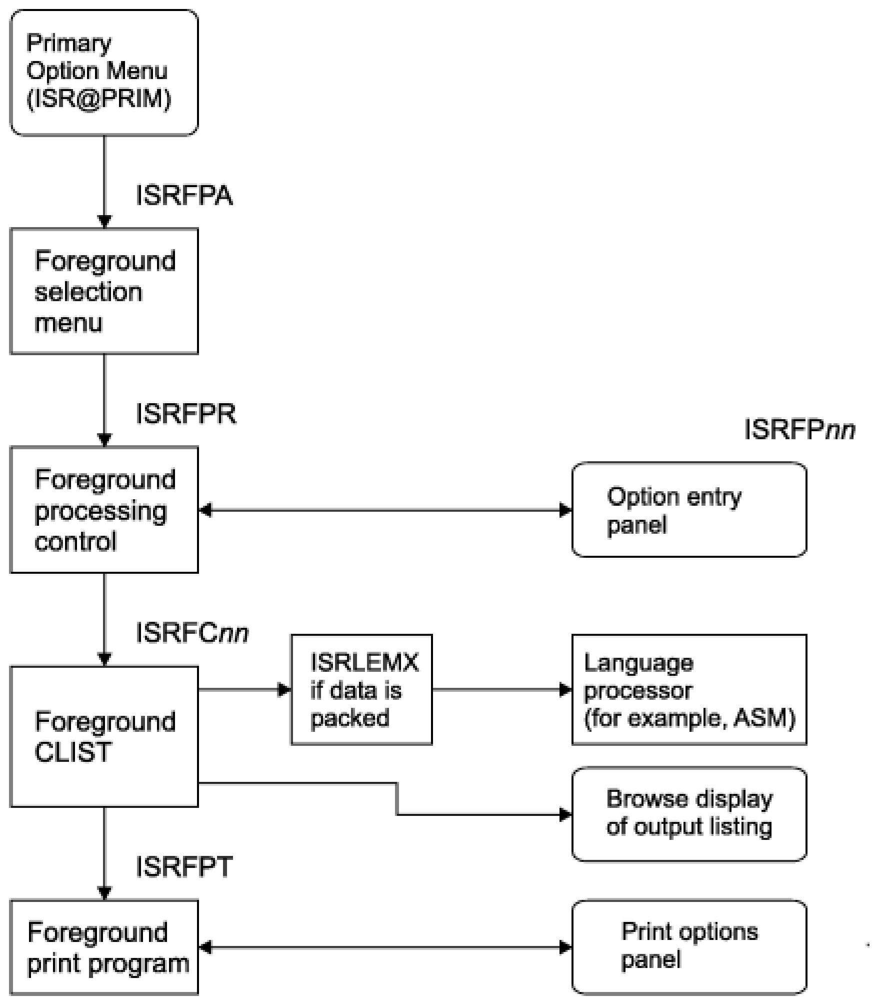

Primary Option Menu (ISR@PRIM)
ISRFPA
Foreground selection menu
ISRFPR
ISRFPnn
Foreground processing control
Option entry panel
ISRFCnn
ISRLEMX if data is packed
Language processor (for example, ASM)
Foreground CLIST
Browse display of output listing
ISRFPT
Foreground print program
Print options panel

Figure 59. Foreground processing flow

ISRFPA is the Foreground Selection Panel. Each option on this menu translates to these selection keywords:

```
'PGM(ISRFPR) PARM((ISRFPnn) nn) NEWPOOL'
```

For all options, program ISRFPR receives control and is passed a parameter containing a list of panel names and the option number. The list of panel names is parsed and the first panel found in the ISPPLIB concatenation sequence is displayed.

Attention: To avoid possible conflicts with ISPF coding, do not use the numbers 12 or 19 for your options.

Each option panel (ISRFP01-ISRFP12) sets certain dialog variables that the ISRFPR program interprets. See "Required option entry panel variables" on page 156. One of these variables, ZSEL, is set to a string of selection keywords (following the rules for selection panels) that indicate which CLIST name (CMD) or load module name (PGM) receives control next.

Note: For foreground processing, ZSEL cannot be set to the PANEL keyword.

The designated CLIST or program does not receive control immediately, because ISRFPR uses the DISPLAY service to display the option panel, rather than the SELECT service. After the option panel is displayed, ISRFPR allocates the user-designated libraries or data sets, displays a member list (if one is required), and scans the concatenated sequence of libraries (if specified) to find the designated member. Then ISRFPR invokes the SELECT service, passing as input the string of selection keywords specified in ZSEL. This causes the corresponding CLIST or program to receive control. (The distributed Foreground option uses CLISTs exclusively.)

The CLIST sets up and issues the appropriate TSO commands to invoke the language processor. The CLIST also initiates automatic browsing of the list data set and subsequent display of the Foreground Print Options panel. When the CLIST finishes, ISRFPR regains control and redisplays the option entry panel.

The PDF interface to VS COBOL II Interactive Debug uses the Debug Productivity Aid (DPA) integrated into VS COBOL II. If your installation does not have VS COBOL II Release 2 or if the panel for DPA (IGZTPIN2) is not in your ISPPLIB concatenation, PDF displays a panel stating DPA is not installed or could not be accessed.

The PDF FORTRAN Interactive Debug option supports both FORTRAN Interactive Debug Version 2 (5668-903) and FORTRAN Interactive Debug Version 1 (5734-F05). If FORTRAN Interactive Debug Version 2 is installed (panel AFFFP11 is found in the ISPPLIB concatenation), PDF executes the Debug Dialog supplied by FORTRAN Interactive Debug Version 2. If FORTRAN Interactive Debug Version 1 is installed (panel DDBFP11), PDF executes that Debug Dialog. PDF looks for FORTRAN Interactive Debug Version 2 first, then Version 1, and finally its own Debug Dialog.

Required option entry panel variables

These dialog variables are explicitly required and must be defined either in the)BODY,)PROC or)INIT section of the suboption panel displayed by ISRFPR. For example, if your application does not require libraries two through four, initialize LIB2, LIB3, and LIB4 to the system variable &Z (blank) in the)INIT section of the suboption panel. This would fill the requirement and allow you to leave these fields out the)BODY section of the suboption panel.

PRJ1
ISPF project name

LIB1
First ISPF library

TYP1
ISPF library type (initially set to nulls in ISRFPR)

MEM
ISPF member name (required if DSN is blank)

DSN
"Other" data set name

ZSEL
Selection string used by ISRFPR to select either a CLIST (CMD) or a program (PGM) with parameters.

LIB2
Second ISPF library

LIB3
Third ISPF library

LIB4
Fourth ISPF library

This field is optional:

PSWD
OS password

The ISRFPR program verifies and processes these variables and then stores these variables in the shared variable pool, so that they can be referenced by the selected CLIST or program.

ZDSQMEM
fully qualified input data set name (without quotes), with the member name in parentheses.

ZDSQ
fully qualified input data set name (without quotes), for the first input data set (without member name)

ZDSQ2
Same as ZDSQ for the second input data set (library)

ZDSQ3
Same as ZDSQ for the third input data set (library)

ZDSQ4
Same as ZDSQ for the fourth input data set (library

ZDS
Same as ZDSQ without the last qualifier (if ZDSQ has only a one level qualifier, then ZDSQ = ZDS.)

ZDS2
Same as ZDS for the second input data set (library)

ZDS3
Same as ZDS for the third input data set (library)

ZDS4
Same as ZDS for the fourth input data set (library)

ZMEM
Input member name (blank for a sequential data set)

ZORG
Input data set organization:

PO=partitioned
PS=sequential

Note: If an asterisk (*) is entered as the member name (from the option panel), ISRFPR does not process the member name. Instead, it sets the ZMEM variable to blank. The CLIST must handle this case.

Other option entry panel variables

ISRFPR does not verify or process other variables from the option panel, such as list ID, compiler options, and additional input libraries. Instead, ISRFPR stores these variables in the shared variable pool (by including VPUT statements in the panel definition) so the CLIST can process them.

Several variables in the option panel are saved in the user profile (again by including VPUT statements in the panel definition) so that they are retained across sessions. In general, variables saved in the profile are also saved in the shared pool to prevent their being modified by another process in split-screen mode. Variables in the profile can be referenced by either screen, but those in the shared pool can be referenced only by the screen in which they are set.

An optional variable, ZSEL2, can be set in the panel. It causes ISRFPR to select a second CLIST or program after successful completion (return code = 0) of the first CLIST or program. ZSEL2 is used with the COBOL and FORTRAN Interactive Debug options.

These sections describe additional variables.

Variables that control automatic Browse and Print

The process of automatically invoking the Browse function for a generated list data set, and subsequently displaying the Foreground Print Options panel is completely external to the ISRFPR program. The ZFBROWS and ZFPRINT variables control these functions. These variables reside in the shared variable pool. Select the distributed Foreground option on the ISRFPA Foreground Selection panel to store ZFBROWS and ZFPRINT in the shared variable pool. ZFBROWS is set to "ISRBROB" (use any nonblank name) and ZFPRINT is set to "ISRFPPRT" (the print panel name).

Use the BROWSE service to invoke the browse function from each CLIST. Use the SELECT service to invoke the print function from each CLIST. SELECT service invokes the ISRFPT program (load module). ISRFPT is distributed with PDF.

ZFPRINT is the variable that is set to the name of the panel to set the print options panel name the dialog developer calls.

The ZFLID variable in ISRFPPRT (Foreground Print Options panel) should be set to the data set name you are going to print. Use the VPUT service to put both the ZFPRINT and ZFLID variables in the shared variable pool. If you do not want to invoke automatic browse or print, set the corresponding variables to blank in either the individual option panels or CLISTs. If you want to set the variables from a panel, remember to use VPUT to store the variables in the shared variable pool so the CLISTs can access them.

When the language processor sends a return code greater than 12 to the ISPF-supplied CLISTs, CLIST bypasses automatic browse and display of the print panel. The exceptions to this are the PL/I Checkout compiler and the COBOL and FORTRAN Interactive Debug programs. The CLISTs for these programs do not display browse or print if the return code is a system return code (for example, "S0C1").

Variables that control option panel redisplay

When an option is processed, normally the original option panel is redisplayed when the specified CLIST or program finishes. ISRFPR stores the original panel name in ZNEXTPN, a variable that controls the next panel to be displayed. You can change the value of ZNEXTPN in your CLIST or panel. For example, to display the linkage edit panel (ISRFP07) after processing the FORTRAN panel (ISRFP03), set ZNEXTPN to ISRFP07 either in the)PROC section of the FORTRAN panel or in the FORTRAN CLIST (ISRFC03). Then place the variable in the shared pool. You can extend this type of panel linkage to any length you want.

Variables used by Foreground CLISTs

In the distributed Foreground option, all option panels select a CLIST to set up and issue the TSO commands. (Only a CLIST or a program can be selected from the option panel.) To obtain variables set from the panel and by program ISRFPR, the CLIST references the shared variable pool. Based on these variables, the CLIST sets up and issues the TSO command required to invoke the language processor. The return code from the TSO command is saved in variable ZFPRFC, which is referenced in various error messages. After completion of the TSO command, the check for automatic browse and print is made. The CLIST issues any log messages. The CLIST also does some specific error checking and can override ISRFPR messages by setting variable ZFPRFC to the return code, issuing a VPUT of ZFPRFC, and invoking the SETMSG service.

The FORTRAN and COBOL interactive CLISTs differ from other CLISTs. These interactive programs attempt to read subsequent lines from the CLIST as input. Therefore, the CLIST is divided into two parts. All functions after the TSO command are in the second CLIST, which is referenced by the ZSEL2 variable described in "Required option entry panel variables" on page 156.

Foreground modifications

Most of the Foreground processing logic resides in the option panels and associated CLISTs to make modifications as easy as possible. Before modifying existing options or adding new ones, you should study one or more of the distributed panel/CLIST pairs to understand the relationships that exist among the panel, the CLIST, and the program ISRFPR. In particular, be aware of these items:

- Additional variables such as LANG and ZORG are used to pass information to log messages, error messages, and panels.
- These commands cannot be invoked under the Foreground option: SPF, PDF, ISPF, ISPSTART, LOGON, LOGOFF, TEST, or a CALL to an authorized program.
- If the CLIST returns a nonzero return code in the EXIT statement, ISRFPR does not attempt to process any second CLIST that was specified by ZSEL2.

Steps to add a new Foreground primary option to PDF

The steps required to add a new PDF primary option that uses the Foreground processing mechanism are listed here.

1. Add the new option (for example, '10') to the ISPF Primary Option Menu, panel ISR@PRIM. In the translated value for option 10, use the PANEL keyword to specify the name of the selection panel to be displayed next. For example:

```
)PROC
  &ZSEL = TRANS( TRUNC(&ZCMD,'.')
                 . . .
```

```
                10,'PANEL(XYZ)'
```

2. Add new selection panel XYZ to the panel library. Use panel ISRFPA as a model. For each option, use the PGM keyword to specify that program ISRFPR is to receive control, and use the PARM keyword to pass the name of the option panel. For example:

```
)PROC
  &ZSEL = TRANS( TRUNC(&ZCMD,'.')
                 . . .

                 2,'PGM(ISRFPR) PARM((FORNEW) 2)'
```

3. Proceed as specified in "Steps to add a new Foreground option to the Foreground Selection Panel" on page 159, starting at step 2.

Steps to add a new Foreground option to the Foreground Selection Panel

The steps required to add a new option to the PDF Foreground Selection Panel are listed here.

1. Add new option (for example, '99') to the Foreground Selection Panel, ISRFPA. Use the PGM keyword to specify that program ISPFPR is to receive control, and use the PARM keyword to pass the name of the new option panel. For example:

```
)PROC
  &ZSEL = TRANS( TRUNC(&ZCMD,'.')
                 . . .
                 99,'PGM(ISRFPR) PARM((FORNEW) 99)'
                 . . .
```

2. Add new option panel FORNEW to the panel library. Use one of the distributed option panels (for example, ISRFP01) as a model.
3. Develop a corresponding CLIST (referenced from panel FORNEW by the ZSEL variable). Use one of the distributed CLISTs (for example, ISRFC01) as a model. Add the CLIST to a library accessible to ddname SYSPROC.

Batch processing panels, CLISTs, and skeletons

The Batch option uses ISPF dialog management services. The following code show the Batch Selection and Batch JCL generation panel formats. See the *z/OS ISPF Dialog Developer's Guide and Reference* for a general description of panel definition formats.

Note: In ISRJPA and ISRJPAB, attribute characters have been replaced by blanks.

Batch selection panel definition (ISRJPA) (Part 1 of 4):

```
)PANEL KEYLIST(ISRSAB,ISR)
)ATTR DEFAULT(...) FORMAT(MIX)              /* ISRJPA - ENGLISH - 7.5 */
 0B TYPE(AB)
 0D TYPE(PS)
 04 TYPE(ABSL) GE(ON)
 05 TYPE(PT)
 09 TYPE(FP)
 0A TYPE(NT)
 0C TYPE(NT) SKIP(ON)
 11 TYPE(SAC)
 22 TYPE(WASL) SKIP(ON) GE(ON)
 08 TYPE(CH)
 10 TYPE(ET)
 26 AREA(SCRL) EXTEND(ON)
 27 TYPE(CEF) PADC(USER) CKBOX(ON)
 28 TYPE(NEF) CAPS(ON) PADC(USER)
)ABC DESC('Menu') MNEM(1)
PDC DESC('Settings') UNAVAIL(ZPM1) MNEM(1) ACC(CTRL+S)
 ACTION RUN(ISRROUTE) PARM('SET')
PDC DESC('View') UNAVAIL(ZPM2) MNEM(1) ACC(CTRL+V)
 ACTION RUN(ISRROUTE) PARM('BR1')
PDC DESC('Edit') UNAVAIL(ZPM3) MNEM(1) ACC(CTRL+E)
 ACTION RUN(ISRROUTE) PARM('ED1')
PDC DESC('ISPF Command Shell') UNAVAIL(ZPM4) MNEM(6) ACC(CTRL+C)
 ACTION RUN(ISRROUTE) PARM('C1')
```

```
PDC DESC('Dialog Test...') UNAVAIL(ZPM5) MNEM(8) ACC(CTRL+T)
 ACTION RUN(ISRROUTE) PARM('DAL')
PDC DESC('Other IBM Products...') UNAVAIL(ZPM6) MNEM(1) ACC(CTRL+O)
 ACTION RUN(ISRROUTE) PARM('OIB')
PDC DESC('SCLM') UNAVAIL(ZPM7) MNEM(3) ACC(CTRL+L)
 ACTION RUN(ISRROUTE) PARM('SCL')
PDC DESC('ISPF Workplace') UNAVAIL(ZPM8) MNEM(6) ACC(CTRL+W)
 ACTION RUN(ISRROUTE) PARM('WRK')
PDC DESC('Status Area...') UNAVAIL(ZPMS) MNEM(8) ACC(CTRL+A)
 ACTION RUN(ISRROUTE) PARM('SAM')
PDC DESC('Exit') MNEM(2) PDSEP(ON) ACC(CTRL+X) ACTION RUN(EXIT)
)ABCINIT
.ZVARS=ISR@OPT
)ABC DESC('Utilities') MNEM(1)
PDC DESC('Library') UNAVAIL(ZUT1) MNEM(1) ACC(ALT+1)
 ACTION RUN(ISRROUTE) PARM('U1')
PDC DESC('Data set') UNAVAIL(ZUT2) MNEM(1) ACC(ALT+2)
 ACTION RUN(ISRROUTE) PARM('U2')
PDC DESC('Move/Copy') UNAVAIL(ZUT3) MNEM(1) ACC(ALT+3)
 ACTION RUN(ISRROUTE) PARM('U3')
PDC DESC('Data Set List') UNAVAIL(ZUT4) MNEM(2) ACC(ALT+4)
 ACTION RUN(ISRROUTE) PARM('U4')
PDC DESC('Reset Statistics') UNAVAIL(ZUT5) MNEM(5) ACC(ALT+5)
 ACTION RUN(ISRROUTE) PARM('U5')

PDC DESC('Hardcopy') UNAVAIL(ZUT6) MNEM(8) ACC(ALT+6)
 ACTION RUN(ISRROUTE) PARM('U6')
```

Batch selection panel definition (ISRJPA) (Part 2 of 4)

```

PDC DESC('Reserved') UNAVAIL(ZUTDT) MNEM(1) ACTION RUN(ISRROUTE) PARM('UDT')
PDC DESC('Outlist') UNAVAIL(ZUT7) MNEM(1) ACC(ALT+8)
 ACTION RUN(ISRROUTE) PARM('U8')
PDC DESC('Commands...') UNAVAIL(ZUT8) MNEM(1) ACC(ALT+9)
 ACTION RUN(ISRROUTE) PARM('U9')
PDC DESC('Reserved') UNAVAIL(ZUT9) MNEM(6) ACTION RUN(ISRROUTE) PARM('U10')
PDC DESC('Format') UNAVAIL(ZUT10) MNEM(1) ACC(ALT+F1)
 ACTION RUN(ISRROUTE) PARM('U11')
PDC DESC('SuperC') UNAVAIL(ZUT11) MNEM(1) PDSEP(ON) ACC(CTRL+F2)
 ACTION RUN(ISRROUTE) PARM('U12')
PDC DESC('SuperCE') UNAVAIL(ZUT12) MNEM(2) ACC(CTRL+F3)
 ACTION RUN(ISRROUTE) PARM('U13')
PDC DESC('Search-For') UNAVAIL(ZUT13) MNEM(2) ACC(CTRL+F4)

 ACTION RUN(ISRROUTE) PARM('U14')
PDC DESC('Search-ForE') UNAVAIL(ZUT14) MNEM(6) ACC(CTRL+F5)
 ACTION RUN(ISRROUTE) PARM('U15')
PDC DESC('Table Utility') UNAVAIL(ZUT15) MNEM(3) ACC(CTRL+F6)
 ACTION RUN(ISRROUTE) PARM('U16')
PDC DESC('Directory List') UNAVAIL(ZUT16) MNEM(2) ACC(CTRL+F7)
 ACTION RUN(ISRROUTE) PARM('U17')
)ABCINIT
.ZVARS=PDFUTIL
&zutdt = '1'
&zut9 = '1'
)ABC DESC('Help') MNEM(1)
PDC DESC('General') MNEM(1) ACTION RUN(TUTOR) PARM('ISR50010')
PDC DESC('Assembler') MNEM(1) ACTION RUN(TUTOR) PARM('ISR51000')
PDC DESC('COBOL') MNEM(1) ACTION RUN(TUTOR) PARM('ISR52000')
PDC DESC('VS Fortran') MNEM(4) ACTION RUN(TUTOR) PARM('ISR53000')
PDC DESC('PL/I') MNEM(2) ACTION RUN(TUTOR) PARM('ISR55000')
PDC DESC('VS Pascal') MNEM(4) ACTION RUN(TUTOR) PARM('ISR56000')
PDC DESC('Binder/Link editor') MNEM(1) ACTION RUN(TUTOR) PARM('ISR57000')
PDC DESC('VS COBOL II debug') MNEM(1) ACTION RUN(TUTOR) PARM('ISR5A000')
PDC DESC('Member parts list') MNEM(1) ACTION RUN(TUTOR) PARM('ISR5C000')
PDC DESC('C/370') MNEM(3) ACTION RUN(TUTOR) PARM('ISR5D000')
PDC DESC('REXX/370') MNEM(1) ACTION RUN(TUTOR) PARM('ISR5E000')
PDC DESC('ADA/370') MNEM(2) ACTION RUN(TUTOR) PARM('ISR5F000')
PDC DESC('AD/Cycle C/370') MNEM(5) ACTION RUN(TUTOR) PARM('ISR5G000')
PDC DESC('ISPDTLC') MNEM(5) ACTION RUN(TUTOR) PARM('ISR5I000')
PDC DESC('OS/390 C/C++') MNEM(1) ACTION RUN(TUTOR) PARM('ISR5J000')
PDC DESC('Appendices') MNEM(4) ACTION RUN(TUTOR) PARM('ISR00004')
)ABCINIT
.ZVARS=JPAHELP
)BODY  CMD(ZCMD)
.. Menu. Utilities. Help.
.-------------------------------------------------------------------------------
.                           .Batch Selection Panel.                            .
.Option ===>.Z                                                                 .
```

```
.SAREA38                                                                         .
)AREA SAREA38
.1 .Assembler           ...7 .*Binder/Link editor ...15.*ADA/370            ..
.2 .COBOL               ...10.*VS COBOL II debug  ...16.*AD/Cycle C/370     ..
.3 .VS FORTRAN          ...12. Member Parts List  ...18. ISPDTLC            ..
.5 .PLI                 ...13.*C/370              ...19.*OS/390 C/C++       ..
.6 .VS PASCAL           ...14.*REXX/370           .
.. .
.  ..&multipmt                        ..*.No packed data support.             .
.  ..Z..Source data online             .
.  ..Z..Source data packed             .
.
.Job Statement Information:.Verify before proceeding.                            .
.                                                                                .
.===>.Z                                                                         .
.===>.Z                                                                         .
.===>.Z                                                                         .
.===>.Z                                                                         .
```

Batch selection panel definition (ISRJPA) (Part 3 of 4)

```
)INIT
.ZVARS = '(ZCMD ZDSCKOV ZBPKEDV BJC1 BJC2 BJC3 BJC4)'
&ZWINTTL = ' '
.HELP = ISR50000
&ZDSCKOV = ' '
&ZBPKEDV = ' '
&ZUT6  = 1
&ZUT7  = 1
&ZUT11 = 1
&ZUT12 = 1
&ZUT13 = 1
&ZUT14 = 1
IF (&ZDSCKO = ' ')
  &ZDSCKO = Y
&ZDSCKO = TRANS(TRUNC(&ZDSCKO,1),N,NO,*,YES)
&ZDSCKOV = TRANS(&ZDSCKO YES,'/' NO,' ')
IF (&ZBPKED = &Z)
  &ZBPKED = N
&ZBPKED = TRANS(TRUNC(&ZBPKED,1),Y,YES,*,NO)
&ZBPKEDV = TRANS(&ZBPKED YES,'/' NO,' ')
IF (&ZGUI = ' ')
  &MULTIPMT='Enter "/" to select option    '
ELSE
  &MULTIPMT='Check box to select option    '
.CURSOR = 'ZCMD'
)REINIT
REFRESH(ZDSCKOV ZBPKEDV)
&ZUT6  = 1
&ZUT7  = 1
&ZUT11 = 1
&ZUT12 = 1
&ZUT13 = 1
&ZUT14 = 1
)PROC
&ZUT6  = 0
&ZUT7  = 0
&ZUT11 = 0
&ZUT12 = 0
&ZUT13 = 0
&ZUT14 = 0
IF (&ZDSCKOV = ' ')
  &ZDSCKO = 'NO'
ELSE
  &ZDSCKO = 'YES'
&ZDSCKO = TRUNC(&ZDSCKO,1)      /*DATA SET (DS) CHECK:             */
&ZDSCHK = TRANS(TRUNC(&ZDSCKO,1),Y,Y,N,N)
&ZDSCKO = TRANS(TRUNC(&ZDSCKO,1),Y,YES,N,NO)
IF (&ZBPKEDV = ' ')
  &ZBPKED = 'NO'
ELSE
  &ZBPKED = 'YES'
```

Batch selection panel definition (ISRJPA) (Part 4 of 4)

```
&ZBPKED  = TRUNC(&ZBPKED,1)
&ZBPKED = TRANS(TRUNC(&ZBPKED,1),Y,YES,N,NO)
&ZBPACK = TRANS(TRUNC(&ZBPKED,1),Y,YES,N,NO)
VER (&ZDSCKO,NB,LIST,YES,NO)           /* Y= VERIFY DSN;N= NO VERIFICATION*/
VER (&ZBPKED,NB,LIST,YES,NO)           /*  Y = EXPAND PACKED DATA  */
&DSN  = ' '                            /* INITIALIZE DATA SET NAME FIELD  */
VPUT (ZDSCHK,ZDSCKO,DSN) SHARED              /* PLACE IN SHARED POOL FOR ISRJB2 */
&RTNPNL = ISRJPB
VPUT (BJC1,BJC2,BJC3,BJC4,ZBPACK,ZBPKED) PROFILE
&ZCMDWRK = &Z
IF (&ZCMD ¬= &Z)
  &ZCMDWRK = TRUNC(&ZCMD,'.')
  &ZTRAIL=.TRAIL
  IF (&ZCMDWRK = &Z)
    .MSG = ISRU000
&ZSEL = TRANS (TRUNC (&ZCMD,'.')
  1,'PGM(ISRJB2) PARM((ISRJP01) 1) NEWPOOL'
  2,'PGM(ISRJB2) PARM((ISRJP02) 2) NEWPOOL'
  3,'PGM(ISRJB2) PARM((ISRJP03) 3) NEWPOOL'
  5,'PGM(ISRJB2) PARM((ISRJP05) 5) NEWPOOL'
  6,'PGM(ISRJB2) PARM((ISRJP06) 6) NEWPOOL'
  7,'PGM(ISRJB2) PARM((ISRJP07) 7) NEWPOOL'
 10,'PGM(ISRJB2) PARM((ISRJP10) 10) NEWPOOL'
 12,'PGM(ISRJB2) PARM((ISRJP12) 12) NEWPOOL'
 13,'PGM(ISRJB2) PARM((EDCJP13,ISRJP13) 13) NEWPOOL'
 14,'PGM(ISRJB2) PARM((FANJP14,ISRJP14) 14) NEWPOOL'
 15,'PGM(ISRALTDI) PARM(EVGJP15,,ISRJP15,*) NOCHECK'
 16,'PGM(ISRJB2) PARM((EDCJP16,ISRJP16) 16) NEWPOOL'
 18,'CMD(ISPDTLC (PANEL SUBMIT RETURN)) MODE(FSCR)'
 19,'PGM(ISRALTDI) PARM(CBC3PE5A,+,ISRJP19,*,ISRJP19A) NOCHECK'
 ' ',' '
   *,'?')
)PNTS
FIELD(ZPS01001) VAR(ZCMD) VAL(1)
FIELD(ZPS01002) VAR(ZCMD) VAL(7)
FIELD(ZPS01003) VAR(ZCMD) VAL(15)
FIELD(ZPS01004) VAR(ZCMD) VAL(2)
FIELD(ZPS01005) VAR(ZCMD) VAL(10)
FIELD(ZPS01006) VAR(ZCMD) VAL(16)
FIELD(ZPS01007) VAR(ZCMD) VAL(3)
FIELD(ZPS01008) VAR(ZCMD) VAL(12)
FIELD(ZPS01009) VAR(ZCMD) VAL(18)
FIELD(ZPS01010) VAR(ZCMD) VAL(5)
FIELD(ZPS01011) VAR(ZCMD) VAL(13)
FIELD(ZPS01012) VAR(ZCMD) VAL(19)
FIELD(ZPS01013) VAR(ZCMD) VAL(6)
FIELD(ZPS01014) VAR(ZCMD) VAL(14)
)END

 /* 5694-A01     COPYRIGHT IBM CORP 1980, 2011 */ /* ISPDTLC Release: 7.5.  Level:
PID                                   */ /* z/OS 02.05.00.  Created - Date: 17 Feb 2020, Time:
23:45           */
```

Batch JCL generation panel definition (ISRJPB) (Part 1 of 4)

```
)PANEL KEYLIST(ISRSAB,ISR)
)ATTR DEFAULT(...) FORMAT(MIX)             /* ISRJPB - ENGLISH - 7.5 */
 0B TYPE(AB)
 0D TYPE(PS)
 04 TYPE(ABSL) GE(ON)
 05 TYPE(PT)
 09 TYPE(FP)
 0A TYPE(NT)
 0C TYPE(NT) SKIP(ON)
 11 TYPE(SAC)
 22 TYPE(WASL) SKIP(ON) GE(ON)
 08 TYPE(CH)
 10 TYPE(ET)
 26 AREA(SCRL) EXTEND(ON)
 27 TYPE(CEF) PADC(USER) CKBOX(ON)
 28 TYPE(NEF) CAPS(ON) PADC(USER)
)ABC DESC('Menu') MNEM(1)
PDC DESC('Settings') UNAVAIL(ZPM1) MNEM(1) ACC(CTRL+S)
 ACTION RUN(ISRROUTE) PARM('SET')
PDC DESC('View') UNAVAIL(ZPM2) MNEM(1) ACC(CTRL+V)
 ACTION RUN(ISRROUTE) PARM('BR1')
PDC DESC('Edit') UNAVAIL(ZPM3) MNEM(1) ACC(CTRL+E)
 ACTION RUN(ISRROUTE) PARM('ED1')
PDC DESC('ISPF Command Shell') UNAVAIL(ZPM4) MNEM(6) ACC(CTRL+C)
 ACTION RUN(ISRROUTE) PARM('C1')
PDC DESC('Dialog Test...') UNAVAIL(ZPM5) MNEM(8) ACC(CTRL+T)
 ACTION RUN(ISRROUTE) PARM('DAL')
PDC DESC('Other IBM Products...') UNAVAIL(ZPM6) MNEM(1) ACC(CTRL+O)
 ACTION RUN(ISRROUTE) PARM('OIB')
```

```
PDC DESC('SCLM') UNAVAIL(ZPM7) MNEM(3) ACC(CTRL+L)
 ACTION RUN(ISRROUTE) PARM('SCL')
PDC DESC('ISPF Workplace') UNAVAIL(ZPM8) MNEM(6) ACC(CTRL+W)
 ACTION RUN(ISRROUTE) PARM('WRK')
PDC DESC('Status Area...') UNAVAIL(ZPMS) MNEM(8) ACC(CTRL+A)
 ACTION RUN(ISRROUTE) PARM('SAM')
PDC DESC('Exit') MNEM(2) PDSEP(ON) ACC(CTRL+X) ACTION RUN(EXIT)
)ABCINIT
.ZVARS=ISR@OPT
)ABC DESC('Utilities') MNEM(1)
PDC DESC('Library') UNAVAIL(ZUT1) MNEM(1) ACC(ALT+1)
 ACTION RUN(ISRROUTE) PARM('U1')
PDC DESC('Data set') UNAVAIL(ZUT2) MNEM(1) ACC(ALT+2)
 ACTION RUN(ISRROUTE) PARM('U2')
PDC DESC('Move/Copy') UNAVAIL(ZUT3) MNEM(1) ACC(ALT+3)
 ACTION RUN(ISRROUTE) PARM('U3')
PDC DESC('Data Set List') UNAVAIL(ZUT4) MNEM(2) ACC(ALT+4)
 ACTION RUN(ISRROUTE) PARM('U4')
PDC DESC('Reset Statistics') UNAVAIL(ZUT5) MNEM(5) ACC(ALT+5)
 ACTION RUN(ISRROUTE) PARM('U5')
PDC DESC('Hardcopy') UNAVAIL(ZUT6) MNEM(8) ACC(ALT+6)
 ACTION RUN(ISRROUTE) PARM('U6')
PDC DESC('Reserved') UNAVAIL(ZUTDT) MNEM(1) ACTION RUN(ISRROUTE) PARM('UDT')
PDC DESC('Outlist') UNAVAIL(ZUT7) MNEM(1) ACC(ALT+8)
 ACTION RUN(ISRROUTE) PARM('U8')
PDC DESC('Commands...') UNAVAIL(ZUT8) MNEM(1) ACC(ALT+9)
 ACTION RUN(ISRROUTE) PARM('U9')
PDC DESC('Reserved') UNAVAIL(ZUT9) MNEM(6) ACTION RUN(ISRROUTE) PARM('U10')
PDC DESC('Format') UNAVAIL(ZUT10) MNEM(1) ACC(ALT+F1)
 ACTION RUN(ISRROUTE) PARM('U11')
```

Batch JCL generation panel definition (ISRJPB) (Part 2 of 4)

```

PDC DESC('SuperC') UNAVAIL(ZUT11) MNEM(1) PDSEP(ON) ACC(CTRL+F2)
 ACTION RUN(ISRROUTE) PARM('U12')
PDC DESC('SuperCE') UNAVAIL(ZUT12) MNEM(2) ACC(CTRL+F3)
 ACTION RUN(ISRROUTE) PARM('U13')
PDC DESC('Search-For') UNAVAIL(ZUT13) MNEM(2) ACC(CTRL+F4)
 ACTION RUN(ISRROUTE) PARM('U14')
PDC DESC('Search-ForE') UNAVAIL(ZUT14) MNEM(6) ACC(CTRL+F5)
 ACTION RUN(ISRROUTE) PARM('U15')
PDC DESC('Table Utility') UNAVAIL(ZUT15) MNEM(3) ACC(CTRL+F6)
 ACTION RUN(ISRROUTE) PARM('U16')
PDC DESC('Directory List') UNAVAIL(ZUT16) MNEM(2) ACC(CTRL+F7)
 ACTION RUN(ISRROUTE) PARM('U17')
)ABCINIT
.ZVARS=PDFUTIL
&zutdt = '1'
&zut9 = '1'
)ABC DESC('Help') MNEM(1)
PDC DESC('General') MNEM(1) ACTION RUN(TUTOR) PARM('ISR50000')
PDC DESC('Assembler') MNEM(1) ACTION RUN(TUTOR) PARM('ISR51000')
PDC DESC('COBOL') MNEM(1) ACTION RUN(TUTOR) PARM('ISR52000')
PDC DESC('VS Fortran') MNEM(4) ACTION RUN(TUTOR) PARM('ISR53000')
PDC DESC('PL/I') MNEM(2) ACTION RUN(TUTOR) PARM('ISR55000')
PDC DESC('VS Pascal') MNEM(4) ACTION RUN(TUTOR) PARM('ISR56000')
PDC DESC('Binder/Link editor') MNEM(1) ACTION RUN(TUTOR) PARM('ISR57000')
PDC DESC('VS COBOL II debug') MNEM(1) ACTION RUN(TUTOR) PARM('ISR5A000')
PDC DESC('Member parts list') MNEM(1) ACTION RUN(TUTOR) PARM('ISR5C000')
PDC DESC('C/370') MNEM(3) ACTION RUN(TUTOR) PARM('ISR5D000')
PDC DESC('REXX/370') MNEM(1) ACTION RUN(TUTOR) PARM('ISR5E000')
PDC DESC('ADA/370') MNEM(2) ACTION RUN(TUTOR) PARM('ISR5F000')
PDC DESC('AD/Cycle C/370') MNEM(5) ACTION RUN(TUTOR) PARM('ISR5G000')
PDC DESC('ISPDTLC') MNEM(5) ACTION RUN(TUTOR) PARM('ISR5I000')
PDC DESC('OS/390 C/C++') MNEM(1) ACTION RUN(TUTOR) PARM('ISR5J000')
PDC DESC('Appendices') MNEM(4) ACTION RUN(TUTOR) PARM('ISR00004')
)ABCINIT
.ZVARS=JPBHELP
)BODY  CMD(ZCMD)
.. Menu. Utilities. Help.
.-------------------------------------------------------------------------------
.                           .Batch Selection Panel.                            .
.Option ===>.Z                                                                 .
.SAREA38                                                                       .
)AREA SAREA38
.1 .Assembler           ...7 .*Binder/Link editor ...15.*ADA/370            ..
.2 .COBOL               ...10.*VS COBOL II debug  ...16.*AD/Cycle C/370     ..
.3 .VS FORTRAN          ...12. Member Parts List  ...18. ISPDTLC            ..
.5 .PLI                 ...13.*C/370              ...19.*OS/390 C/C++       ..
```

```
.6 .VS PASCAL           ...14.*REXX/370           .
.              ..
.              ..*.No packed data support.                                      .
.Instructions:.                                                                 .
. ..Enter option to continue generating JCL,.CANCEL.command to exit without.    .
. ..submitting job or.END.command to &ZBMSG                                     .
. ... .
. ..&multipmt                      .
. ..Z..Source data online           .
. ..Z..Source data packed           .
.
.Job Statement Information:.                                                    .
.                                                                               .
.===>.Z                                                                        .
.===>.Z                                                                        .
.===>.Z                                                                        .
.===>.Z                                                                        .
```

Batch JCL generation panel definition (ISRJPB) (Part 3 of 4)

```
)INIT
.ZVARS = '(ZCMD ZDSC ZBPK BJC1 BJC2 BJC3 BJC4)'
&ZWINTTL = ' '
.HELP = ISR50000
&ZDSC = ' '
&ZBPK = ' '
&ZUT6  = 1
&ZUT7  = 1
&ZUT11 = 1
&ZUT12 = 1
&ZUT13 = 1
&ZUT14 = 1
IF (&ZDSCKO = ' ')
  &ZDSCKO = Y
&ZDSCKO = TRANS(TRUNC(&ZDSCKO,1),N,NO,*,YES)
&ZDSC = TRANS(&ZDSCKO NO,' ' YES,'/')
IF (&ZBPKED = &Z)
  &ZBPKED = N
&ZBPKED = TRANS(TRUNC(&ZBPKED,1),Y,YES,*,NO)
&ZBPK = TRANS(&ZBPKED NO,' ' YES,'/')
IF (&ZJOBSTEP = 'YES')
  &ZBMSG = 'submit job.'
IF (&ZJOBSTEP = 'NO ')
  &ZBMSG = 'exit without submitting job.'
IF (&ZGUI = ' ')
  &MULTIPMT='Enter "/" to select option    '
ELSE
  &MULTIPMT='Check box to select option    '
.CURSOR = 'ZCMD'
)REINIT
REFRESH(ZDSC ZBPK)
&zut6 = 1
&zut7 = 1
&zut11 = 1
&zut12 = 1
&zut13 = 1
&zut14 = 1
)PROC
&zut6 = 0
&zut7 = 0
&zut11 = 0
&zut12 = 0
&zut13 = 0
&zut14 = 0
&ZDSCKO = TRANS(&ZDSC ' ','NO' *,'YES')
&ZBPKED = TRANS(&ZBPK ' ','NO' *,'YES')
&ZDSCKO = TRUNC(&ZDSCKO,1)      /*DATA SET (DS) CHECK:              */
VER (&ZDSCKO,NB,LIST,Y,N)       /* Y= VERIFY DSN;N= NO VERIFICATION*/
&ZDSCHK = TRANS(TRUNC(&ZDSCKO,1),Y,Y,N,N)
&ZBPKED  = TRUNC(&ZBPKED,1)
VER (&ZBPKED,NB,LIST,Y,N)               /*  Y = EXPAND PACKED DATA  */
&ZBPKED = TRANS(TRUNC(&ZBPKED,1),Y,YES,N,NO)
&ZBPACK = TRANS(TRUNC(&ZBPKED,1),Y,YES,N,NO)
&DSN  = ' '                     /* INITIALIZE DATA SET NAME FIELD  */
VPUT (ZDSCHK,ZDSCKO,DSN) SHARED /* PLACE IN SHARED POOL FOR ISRJB2 */
VPUT (ZBPACK,ZBPKED) PROFILE
&ZSEL = TRANS (&ZCMD
                C,C
              CAN,C
```

```
            CANCEL,C
                 *,'*' )
```

Batch JCL generation panel definition (ISRJPB) (Part 4 of 4)

```
if (&ZSEL = 'C') goto ENDD
&ZCMDWRK = &Z
IF (&ZCMD ¬= &Z)
  &ZCMDWRK = TRUNC(&ZCMD,'.')
  &ZTRAIL=.TRAIL
  IF (&ZCMDWRK = &Z)
    .MSG = ISRU000
&ZSEL = TRANS (TRUNC (&ZCMD,'.')
  1,'PGM(ISRJB2) PARM((ISRJP01) 1) NEWPOOL'
  2,'PGM(ISRJB2) PARM((ISRJP02) 2) NEWPOOL'
  3,'PGM(ISRJB2) PARM((ISRJP03) 3) NEWPOOL'
  5,'PGM(ISRJB2) PARM((ISRJP05) 5) NEWPOOL'
  6,'PGM(ISRJB2) PARM((ISRJP06) 6) NEWPOOL'
  7,'PGM(ISRJB2) PARM((ISRJP07) 7) NEWPOOL'
 10,'PGM(ISRJB2) PARM((ISRJP10) 10) NEWPOOL'
 12,'PGM(ISRJB2) PARM((ISRJP12) 12) NEWPOOL'
 13,'PGM(ISRJB2) PARM((EDCJP13,ISRJP13) 13) NEWPOOL'
 14,'PGM(ISRJB2) PARM((FANJP14,ISRJP14) 14) NEWPOOL'
 15,'PGM(ISRALTDI) PARM(EVGJP15,,ISRJP15,*) NOCHECK'
 16,'PGM(ISRJB2) PARM((EDCJP16,ISRJP16) 16) NEWPOOL'
 18,'CMD(ISPDTLC (PANEL SUBMIT RETURN)) MODE(FSCR)'
 19,'PGM(ISRALTDI) PARM(CBC3PE5A,+,ISRJP19,*,ISRJP19A) NOCHECK'
 ' ',' '
   *,'?')
ENDD:
)PNTS
FIELD(ZPS01001) VAR(ZCMD) VAL(1)
FIELD(ZPS01002) VAR(ZCMD) VAL(7)
FIELD(ZPS01003) VAR(ZCMD) VAL(15)
FIELD(ZPS01004) VAR(ZCMD) VAL(2)
FIELD(ZPS01005) VAR(ZCMD) VAL(10)
FIELD(ZPS01006) VAR(ZCMD) VAL(16)
FIELD(ZPS01007) VAR(ZCMD) VAL(3)
FIELD(ZPS01008) VAR(ZCMD) VAL(12)
FIELD(ZPS01009) VAR(ZCMD) VAL(18)
FIELD(ZPS01010) VAR(ZCMD) VAL(5)
FIELD(ZPS01011) VAR(ZCMD) VAL(13)
FIELD(ZPS01012) VAR(ZCMD) VAL(19)
FIELD(ZPS01013) VAR(ZCMD) VAL(6)
FIELD(ZPS01014) VAR(ZCMD) VAL(14)
)END
 /* 5694-A01     COPYRIGHT IBM CORP 1980, 2011 */ /* ISPDTLC Release: 7.5.  Level:
PID                                  */ /* z/OS 02.05.00.  Created - Date: 17 Feb 2020, Time:
23:45           */
```

The Batch option includes eight suboptions. Each Batch option has an associated panel, CLIST, and skeleton. For option 1, for example, the option panel name is ISRJP01, the CLIST name is ISRJC01, and the skeleton name is ISRJS01. The following code shows the Batch Assembler panel definition.

Note: In ISRJP01, attribute characters have been replaced by blanks.

Batch Assembler definition (ISRJP01) (Part 1 of 5)

```
)PANEL KEYLIST(ISRSAB,ISR)
)ATTR DEFAULT(...) FORMAT(MIX)             /* ISRJP01 - ENGLISH - 7.5 */
 0B TYPE(AB)
 04 TYPE(ABSL) GE(ON)
 05 TYPE(PT)
 09 TYPE(FP)
 0A TYPE(NT)
 0C TYPE(NT) SKIP(ON)
 11 TYPE(SAC)
 12 TYPE(CEF) PADC(USER)
 19 TYPE(DT)
 22 TYPE(WASL) SKIP(ON) GE(ON)
 08 TYPE(CH)
 26 TYPE(NEF) CAPS(ON) PADC(USER)
 27 AREA(SCRL) EXTEND(ON)
 28 TYPE(SAC) CSRGRP(99) RADIO(ON)
)ABC DESC('Menu') MNEM(1)
PDC DESC('Settings') UNAVAIL(ZPM1) MNEM(1) ACC(CTRL+S)
 ACTION RUN(ISRROUTE) PARM('SET')
PDC DESC('View') UNAVAIL(ZPM2) MNEM(1) ACC(CTRL+V)
```

```
 ACTION RUN(ISRROUTE) PARM('BR1')
PDC DESC('Edit') UNAVAIL(ZPM3) MNEM(1) ACC(CTRL+E)
 ACTION RUN(ISRROUTE) PARM('ED1')
PDC DESC('ISPF Command Shell') UNAVAIL(ZPM4) MNEM(6) ACC(CTRL+C)
 ACTION RUN(ISRROUTE) PARM('C1')
PDC DESC('Dialog Test...') UNAVAIL(ZPM5) MNEM(8) ACC(CTRL+T)
 ACTION RUN(ISRROUTE) PARM('DAL')
PDC DESC('Other IBM Products...') UNAVAIL(ZPM6) MNEM(1) ACC(CTRL+O)
 ACTION RUN(ISRROUTE) PARM('OIB')
PDC DESC('SCLM') UNAVAIL(ZPM7) MNEM(3) ACC(CTRL+L)
 ACTION RUN(ISRROUTE) PARM('SCL')
PDC DESC('ISPF Workplace') UNAVAIL(ZPM8) MNEM(6) ACC(CTRL+W)
 ACTION RUN(ISRROUTE) PARM('WRK')
PDC DESC('Status Area...') UNAVAIL(ZPMS) MNEM(8) ACC(CTRL+A)
 ACTION RUN(ISRROUTE) PARM('SAM')
PDC DESC('Exit') MNEM(2) PDSEP(ON) ACC(CTRL+X) ACTION RUN(EXIT)
)ABCINIT
.ZVARS=ISR@OPT
)ABC DESC('RefList') MNEM(1)
PDC DESC('Current Data Set List &ZDSCURT') MNEM(1) ACC(CTRL+ALT+P)
 ACTION RUN(ISRRLIST) PARM('PL1')
PDC DESC('Current Library List &ZDSCURLT') MNEM(2) ACC(CTRL+SHIFT+P)
 ACTION RUN(ISRRLIST) PARM('LL1')
PDC DESC('List of Personal Data Set Lists') MNEM(1) PDSEP(ON) ACC(CTRL+ALT+O)
 ACTION RUN(ISRRLIST) PARM('PL2')
PDC DESC('List of Personal Library Lists') MNEM(2) ACC(CTRL+SHIFT+O)
 ACTION RUN(ISRRLIST) PARM('LL2')
```

Batch Assembler definition (ISRJP01) (Part 2 of 5)

```
)ABCINIT
.ZVARS=REFLIST
      VGET (ZCURTB ZCURLTB) PROFILE
      IF (&ZCURTB = &Z) &ZDSCURT = &Z
      ELSE &ZDSCURT= '(&ZCURTB)'
      IF (&ZCURLTB = &Z) &ZDSCURLT = &Z
      ELSE &ZDSCURLT= '(&ZCURLTB)'
)ABC DESC('Utilities') MNEM(1)
PDC DESC('Library') UNAVAIL(ZUT1) MNEM(1) ACC(ALT+1)
 ACTION RUN(ISRROUTE) PARM('U1')
PDC DESC('Data set') UNAVAIL(ZUT2) MNEM(1) ACC(ALT+2)
 ACTION RUN(ISRROUTE) PARM('U2')
PDC DESC('Move/Copy') UNAVAIL(ZUT3) MNEM(1) ACC(ALT+3)
 ACTION RUN(ISRROUTE) PARM('U3')
PDC DESC('Data Set List') UNAVAIL(ZUT4) MNEM(2) ACC(ALT+4)
 ACTION RUN(ISRROUTE) PARM('U4')
PDC DESC('Reset Statistics') UNAVAIL(ZUT5) MNEM(5) ACC(ALT+5)
 ACTION RUN(ISRROUTE) PARM('U5')

PDC DESC('Hardcopy') UNAVAIL(ZUT6) MNEM(8) ACC(ALT+6)
 ACTION RUN(ISRROUTE) PARM('U6')
PDC DESC('Reserved') UNAVAIL(ZUTDT) MNEM(1) ACTION RUN(ISRROUTE) PARM('UDT')
PDC DESC('Outlist') UNAVAIL(ZUT7) MNEM(1) ACC(ALT+8)
 ACTION RUN(ISRROUTE) PARM('U8')
PDC DESC('Commands...') UNAVAIL(ZUT8) MNEM(1) ACC(ALT+9)

 ACTION RUN(ISRROUTE) PARM('U9')
PDC DESC('Reserved') UNAVAIL(ZUT9) MNEM(6) ACTION RUN(ISRROUTE) PARM('U10')
PDC DESC('Format') UNAVAIL(ZUT10) MNEM(1) ACC(ALT+F1)
 ACTION RUN(ISRROUTE) PARM('U11')
PDC DESC('SuperC') UNAVAIL(ZUT11) MNEM(1) PDSEP(ON) ACC(CTRL+F2)
 ACTION RUN(ISRROUTE) PARM('U12')
PDC DESC('SuperCE') UNAVAIL(ZUT12) MNEM(2) ACC(CTRL+F3)
 ACTION RUN(ISRROUTE) PARM('U13')
PDC DESC('Search-For') UNAVAIL(ZUT13) MNEM(2) ACC(CTRL+F4)
 ACTION RUN(ISRROUTE) PARM('U14')
PDC DESC('Search-ForE') UNAVAIL(ZUT14) MNEM(6) ACC(CTRL+F5)
 ACTION RUN(ISRROUTE) PARM('U15')
PDC DESC('Table Utility') UNAVAIL(ZUT15) MNEM(3) ACC(CTRL+F6)
 ACTION RUN(ISRROUTE) PARM('U16')
PDC DESC('Directory List') UNAVAIL(ZUT16) MNEM(2) ACC(CTRL+F7)
 ACTION RUN(ISRROUTE) PARM('U17')
)ABCINIT
.ZVARS=PDFUTIL
&zutdt = '1'
&zut9 = '1'
)ABC DESC('Help') MNEM(1)
PDC DESC('Input Data Set') MNEM(1) ACTION RUN(TUTOR) PARM('ISR51001')
PDC DESC('SYSLIB Data Sets') MNEM(1) ACTION RUN(TUTOR) PARM('ISR51002')
PDC DESC('Object Data Set') MNEM(1) ACTION RUN(TUTOR) PARM('ISR51003')
PDC DESC('Listing') MNEM(1) ACTION RUN(TUTOR) PARM('ISR51004')
PDC DESC('Password protection') MNEM(1) ACTION RUN(TUTOR) PARM('ISR51007')
PDC DESC('Assembler selection') MNEM(1) ACTION RUN(TUTOR) PARM('ISR51008')
PDC DESC('Appendices') MNEM(4) ACTION RUN(TUTOR) PARM('ISR00004')
PDC DESC('Index') MNEM(2) ACTION RUN(TUTOR) PARM('ISR91000')
)ABCINIT
.ZVARS=JP1HELP
```

Batch Assembler definition (ISRJP01) (Part 3 of 5)

```
)BODY  CMD(ZCMD)
.. Menu. RefList. Utilities. Help.
.-------------------------------------------------------------------------------
.                              .Batch Assembler.                               .
.Command ===>.Z                                                                .
.SAREA39                                                                       .
)AREA SAREA39
.ISPF Library:.                                                                .
. ..Project . . ..Z       .
. ..Group . . . ..Z       .. . ..Z       .. . ..Z       .. . ..Z       .
. ..Type  . . . ..Z       .
. ..Member  . . ..Z       ..(Blank or pattern for member selection list).
.
.Other Partitioned or Sequential Data Set:.                                    .
. ..Data Set Name  . ..Z                                                       .
.. .
                                                     .Assembler.
.List ID  . . . . ..Z       ..(Blank for hardcopy). .Z..1..High Level Assembler.
.SYSOUT class . . ..Z              ..(For hardcopy).   .2..Assembler H         .
.
.Assembler options:.                                                           .
...Term  . . ..Z     ..(TERM or NOTERM).
...Other . . ..Z                                                            .
.
.Additional input libraries:.                                                  .
. ..    ===>.Z                                                                .
. ..    ===>.Z                                                                .
. ..    ===>.Z                                                                .
)INIT
.ZVARS = '(ZCMD PRJ1 LIB1 LIB2 LIB3 LIB4 TYP1 MEM DSN LID ZASMOPT BCLA +
           BHASMT BHASM BHAL1 BHAL2 BHAL3)'
.HELP = ISR51000
.ATTR(ZASMOPT)='CSRGRP(99) RADIO(ON)'
&ZUT6  = 1
&ZUT7  = 1
&ZUT11 = 1
&ZUT12 = 1
&ZUT13 = 1
&ZUT14 = 1
&ZMLCSR = '        '                /*                        @M1A*/
&TYP1 = &ASMT                       /*ASSEMBLER TYPE VARIABLE NAME*/
IF (&DSN ¬= ' ')
  &MEM = ' '                        /*                        @M1A*/
  IF (.CURSOR = ' ')   .CURSOR = DSN
  IF (&ZASMOPT ¬= 1)
    IF (&ZASMOPT ¬= 2)
      &ZASMOPT = 1
IF (&BASMT = ' ')
&BASMT = NOTERM                     /* DEFAULT TO "NOTERM"        */
IF (&ZBTEMP = '')
  &ZBPACK = &ZBPACK
  VPUT (ZBPACK) SHARED
  &ZBTEMP = '0'
IF (&BHALEV = &Z)                                      /* @OW19891*/
  &BHALEV = 'HLASM'                                    /* @OW19891*/
VGET (BHAL1,BHAL2,BHAL3) PROFILE            /*                    OW22979*/
)REINIT
REFRESH(PRJ1 LIB1 LIB2 LIB3 LIB4 TYP1 MEM DSN BHAL1 BHAL2 BHAL3)
```

Batch Assembler definition (ISRJP01) (Part 4 of 5)

```
IF (&ZNXTMSG='ISRT') .CSRPOS = &ZCSRP       /* AUTOTYPE */
                     .CURSOR = &ZCSRV       /* AUTOTYPE */
ELSE &ZXZX = &Z                            /* AUTOTYPE */
&ZUT6  = 1
&ZUT7  = 1
&ZUT11 = 1
&ZUT12 = 1
&ZUT13 = 1
```

```
&ZUT14 = 1
IF (&ZMLCSR ¬= ' ')                          /*                        @M1A*/
  .CURSOR = &ZMLCSR                          /*                        @M1A*/
IF (.MSG = ISRDS003)                         /*                        @M1A*/
  REFRESH (PRJ1,LIB1,LIB2,LIB3,LIB4,TYP1,MEM,DSN)
)PROC
&ZCSRV = .CURSOR                             /* AUTOTYPE */
&ZCSRP = .CSRPOS                             /* AUTOTYPE */
&ZODSNLN = 0                                 /* AUTOTYPE */
&ZODSNMB = &Z                                /* AUTOTYPE */
&ZNAMES='ZCSRV ZCSRP PRJ1 LIB1 LIB2 LIB3 LIB4 TYP1 MEM '
&ZNAMES='&ZNAMES *.&ZODSNLN&ZODSNMB ZCMD'
IF (.CURSOR = DSN, BHAL1, BHAL2, BHAL3)
  &ZODSNLN = 56
  &ZODSNMB = &Z
  IF (.CURSOR = BHAL1) &ZODSNLN = 65
  IF (.CURSOR = BHAL1) &ZODSNMB = '%'
  IF (.CURSOR = BHAL2) &ZODSNLN = 65
  IF (.CURSOR = BHAL2) &ZODSNMB = '%'
  IF (.CURSOR = BHAL3) &ZODSNLN = 65
  IF (.CURSOR = BHAL3) &ZODSNMB = '%'
  &ZNAMES='ZCSRV ZCSRP * * * * * * * &ZCSRV&ZODSNLN&ZODSNMB ZCMD'
PANEXIT((ZNAMES),LOAD,ISRAUTOT)               /* AUTOTYPE */
IF (&ZNXTMSG='ISRT') EXIT                    /* AUTOTYPE */
VER(&ZASMOPT,NONBLANK)
VER(&ZASMOPT RANGE,1,2)
&ZUT6  = 0
&ZUT7  = 0
&ZUT11 = 0
&ZUT12 = 0
&ZUT13 = 0
&ZUT14 = 0
VGET (ZRDSN) SHARED                          /* REFERENCE LIST CODE        */
IF (&ZRDSN ¬= ' ')                           /* IF DATA SET SELECTED       */
  &DSN = &ZRDSN                              /*    PUT DSN VARIABLE        */
  &ZRDSN = ' '                               /*     INTO PANEL             */
  &ZRVOL = ' '                               /*     INTO PANEL             */
VPUT (ZRDSN ZRVOL) SHARED                    /*                            */
  .CURSOR = DSN
  .MSG = ISRDS003                            /*    MSG PENDING             */
VGET (DSALSEL) SHARED                        /*                            */
IF (&DSALSEL ¬= ' ')                         /* IF LIBRARY SELECTED        */
   VGET (DSA1,DSA2,DSA3,DSA4,DSA5,DSA6,DSA7) SHARED
  &PRJ1 = &DSA1                              /*    PUT LIBRARY VARIABLES   */
  &LIB1 = &DSA2                              /*      INTO PANEL            */
  &LIB2 = &DSA3                              /*          .                 */
```

Batch Assembler definition (ISRJP01) (Part 5 of 5)

```
  &LIB3 = &DSA4                              /*          .                 */
  &LIB4 = &DSA5                              /*          .                 */
  &TYP1 = &DSA6                              /*          .                 */
  &MEM  = &DSA7                              /*          .                 */
  &DSN = ' '                                 /*    BLANK OUT DSN           */
  &DSALSEL = ' '                             /*    CLEAR LIBRARY SELECTION */
  VPUT (DSALSEL) SHARED                      /*                            */
  .CURSOR = MEM
  .MSG = ISRDS003                            /*    MSG PENDING             */
                                             /*                            */
IF (&ZCMD ¬= ' ') .MSG = ISPZ001             /* INVALID COMMAND            */
VER (&LID,NAME)                              /* LIST ID MUST BE VALID NAME */
IF (&ZDSCHK = 'N' )                          /* IF NO SPF CHECK OF DATASET */
  IF (&DSN = ' ' )                           /*  AND SPF LIBRARY SPECIFIED */
    VER (&PRJ1,NB,NAME)                      /*     REQUIRED FIELD         */
    VER (&LIB1,NB,NAME)                      /*     REQUIRED FIELD         */
    VER (&TYP1,NB,NAME)                      /*     REQUIRED FIELD         */
  IF (&DSN ¬= ' ')                           /*                        @M1A*/
    VER(&DSN DSNAMEPQ)
ELSE
  IF (&DSN ¬= ' ')
    VER(&DSN DSNAMEFM)
IF (&ZASMOPT = 1)
  &BHALEV = 'HLASM'
IF (&ZASMOPT = 2)
  &BHALEV = 'HASM'
VER (&BHALEV,NB,LIST,HASM,HLASM)             /* Assembler level     OW10516*/
VER (&BHASMT,NB,LIST,TERM,NOTERM)            /* TERM OR NOTERM REQUIRED    */
IF (&LID = ' ' )                             /* IF HARDCOPY DESIRED SPECIFY*/
  VER (&BCLA,NONBLANK)                       /*  VALID LOCAL SYSOUT CLASS  */
IF (&BHAL1 ¬= ' ')                           /* IF LIB #1 SPECIFIED    @M2A*/
```

```
  VER (&BHAL1,DSNAMEPQ)                       /* VERIFY ADDITIONAL LIB1@DSNQ*/
IF (&BHAL2 ¬= ' ')                            /* IF LIB #2 SPECIFIED    @M2A*/
  VER (&BHAL2,DSNAMEPQ)                       /* VERIFY ADDITIONAL LIB2@DSNQ*/
IF (&BHAL3 ¬= ' ')                            /* IF LIB #2 SPECIFIED    @L2A*/
  VER (&BHAL3,DSNAMEPQ)                       /* VERIFY ADDITIONAL LIB2@DSNQ*/
&ASMT = &TYP1                                 /* SAVE ASSEMBLER TYPE        */
&ZSEL = 'CMD(%ISRJC01)'                       /* EXECUTE ASSEMBLE CLIST     */
IF (&LID ¬= ' ' )                             /* IF &BCLA NOT REQD          */
  IF (&BCLA = ' ')                            /*  AND NOT SET               */
    &BCLA = *                                 /*  DEFAULT TO MSGCLASS       */
                                              /* Begin @L3A                 */
&ZSYSDS1 = &BHAL1                             /* Fill fields for ISRJFSYS   */
&ZSYSDS2 = &BHAL2                             /* to use as input.           */
&ZSYSDS3 = &BHAL3                             /* ZSYSDS? is a qualified     */
&ZSYSCUR1 = 'BHAL1'                           /*     dataset.               */
&ZSYSCUR2 = 'BHAL2'                           /* ZSYSCUR? is were the cursor*/
&ZSYSCUR3 = 'BHAL3'                           /*     is placed on a error.  */
VPUT (ZSYSDS1 ZSYSDS2 ZSYSDS3 ZSYSCUR1 ZSYSCUR2 ZSYSCUR3) SHARED
                                              /* End   @L3A                 */
VPUT (PRJ1,LIB1,LIB2,LIB3,LIB4,ASMT,BCLA,BHASMT,BHASM) PROFILE
VPUT (BHAL1,BHAL2,BHAL3,BHALEV,ZASMOPT) PROFILE
VPUT (PRJ1,LIB1,LIB2,LIB3,LIB4,ASMT,BCLA,BHASMT,BHASM) SHARED
VPUT (DSN,LID,BHAL1,BHAL2,BHAL3,BHALEV) SHARED            /* OW10516*/
)END /* 5694-A01     COPYRIGHT IBM CORP 1980, 2011                         */ /* ISPDTLC
Release: 7.5.  Level: PID                                  */ /* z/OS 02.05.00.  Created -
Date: 18 Feb 2020, Time: 07:41           */
```

The names of the ISPF-supplied panels, CLISTs, and skeletons for the Batch processing option are shown in the Table 26 on page 169.

Table 26. ISPF-supplied panels, CLISTs, and skeletons for Batch option

Option	Description	Panel ID	CLIST ID	SKEL ID
-	BATCH SELECTION MENU	ISRJPA	-	ISRJSJC
--	BATCH TERMINATION MENU	ISRJPB	--	--
1	ASSEMBLER	ISRJP01	ISRJC01	ISRJS01
2	VS COBOL II	ISRJP02	ISRJC02	ISRJS02
3	FORTRAN COMPILE	ISRJP03	ISRJC03	ISRJS03
5	PLI OPTIMIZER COMPILE	ISRJP05	ISRJC05	ISRJS05
6	VS PASCAL COMPILE	ISRJP06	ISRJC06	ISRJS06
7	BINDER/LINK EDIT	ISRJP07	ISRJC07	ISRJS07
10	VS COBOL II DEBUG	ISRJP10	ISRJC10	ISRJS10
12	MEMBER PARTS LIST	ISRJP12	ISRJC12	ISRJS12
13	C/370 COMPILE	--	--	--
14	REXX/370 COMPILE	--	--	--
15	ADA/370 COMPILE	--	--	--
16	AD/CYCLE C/370 COMPILE	--	--	--
17	AD/CYCLE C/370 COBOL/370	--	--	--
18	ISPDTLC	ISPCP01	--	ISPDTLB
19	OS/390 C C++	--	--	--

The overall flow of control for Batch processing is shown in Figure 60 on page 170.

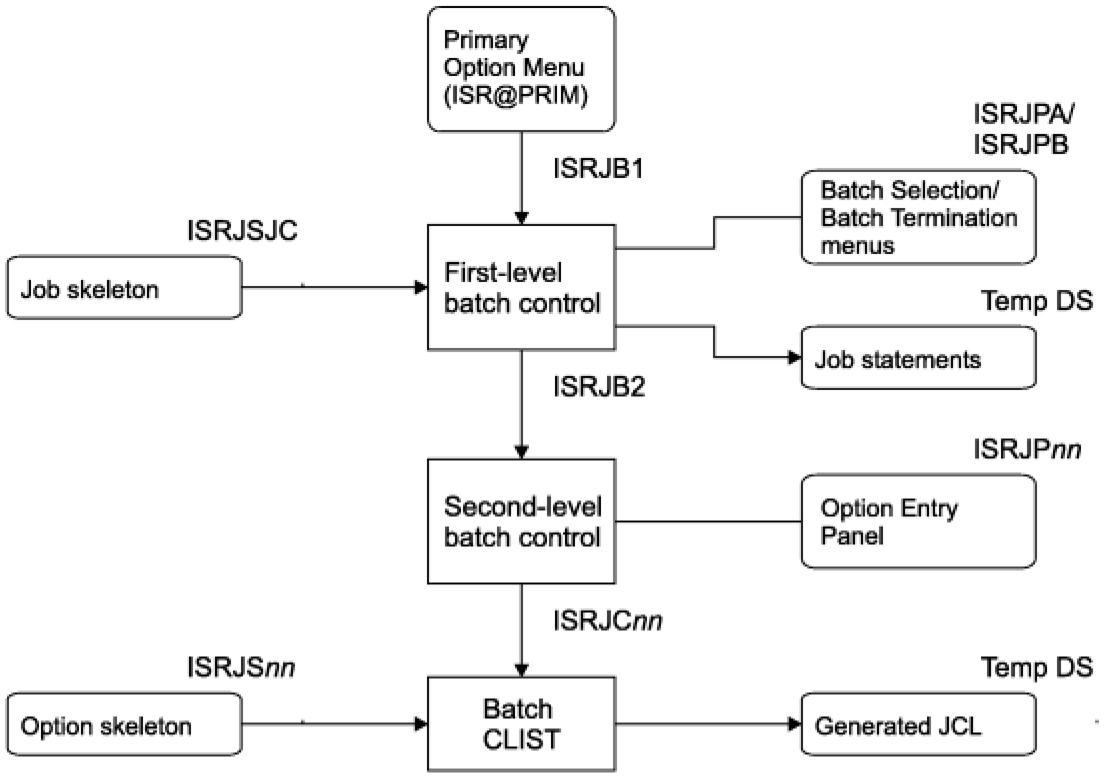


Figure 60. Batch processing flow

Two PDF programs control Batch processing: ISRJB1 and ISRJB2. Program ISRJB1 receives control directly from the ISPF Primary Option Menu, ISR@PRIM, through specification of these selection keywords when you select primary option 5:

```
'PGM(ISRJB1) PARM(ISRJPA) NOCHECK'
```

The parameter specifies the name of the Batch Selection panel. The NOCHECK keyword allows ISRJB1 to receive control if you specify option 5.n, where "n" is a batch option.

ISRJB1 either displays the Batch Selection panel (if you entered option 5 on the ISPF Primary Option Menu) or processes the Batch Selection panel in non-display mode (if you entered 5.n). ISRJB1 makes this determination by examining the ZTRAIL variable, which is set from the ISPF Primary Option Menu. ZTRAIL contains either the option number, or a blank if no option was specified.

Regardless of whether the Batch Selection panel is actually displayed, ISRJB1 invokes file tailoring services to write the four job statements from the Batch Selection panel to a temporary data set. Skeleton ISRJSJC is used to generate the job statement output. ISRJB1 then invokes program ISRJB2 (through the SELECT service) and passes a parameter containing the name of the first (or only) option panel to be displayed.

The selection keywords used to invoke ISRJB2 actually come from the Batch Selection panel, ISRJPA, where they are stored in variable ZSEL. ISRJB1 uses the DISPLAY service, rather than the SELECT service, to display the Batch Selection panel, and subsequently passes ZSEL as input to the SELECT service when it is ready to invoke ISRJB2.

Program ISRJB2 is analogous to the Foreground control program ISRFPR. See "Foreground processing panels and CLISTs" on page 147. ISRJB2 displays and processes the option panel and invokes the corresponding CLIST. The CLIST, in turn, uses file tailoring services to generate JCL statements for the particular option. All Batch JCL is accumulated in the same temporary data set that contains the four job statements.

Eventually, control returns to ISRJB1, the first batch program. ISRJB1 then either displays the Batch Selection panel (ISRJPB) with Job Step Generated, or processes it in non-display mode if display of the Batch Selection panel was bypassed. The Job Step Generated panel is similar in appearance to the Batch

Selection Panel, except that the job statements can no longer be modified. From the Job Step Generated panel, you can select additional options, causing additional job steps to be generated.

Finally, ISRJB1 closes the temporary data set in which all the batch JCL was accumulated and submits it using the TSO SUBMIT command. The submission will be bypassed if either of these conditions occurs:

- No JCL was generated by the options.
- You entered CANCEL on the Batch Selection panel with Job Step Generated.

Variables that control batch processing

The Batch Selection panel, ISRJPA, includes two variables that control Batch processing. The variable RTNPNL contains the name of the panel that corresponds to the Batch Selection panel (for ISRJPA the corresponding panel is ISRJPB). The variable ZDSCKO is associated with the Source Data Online field on the Batch Selection panel. If you enter YES in that field, the Batch option provides error checking and verification of the input data sets. If NO is entered, no data set verification is performed. The NO option allows you to submit a job to be run at a later time if the data sets are not online.

If the Source Data Online field, variable ZDSCKO, is set to YES, the dialog variables required for batch option panels are the same as those for foreground processing. That is, variables PRJ1, LIB1, LIB2, LIB3, LIB4, TYP1, MEM, DSN, and ZSEL must be defined in either the)INIT,)BODY, or)PROC section of the panel displayed by ISRJB2.

If the Source Data Online, variable ZDSCKO, is set to NO, these variables are not required. However, if a data set name is not supplied to ISRJB2 in these variables, the output variables ZDSQMEM, ZDSQ, ZDSQ2, ZDSQ3, ZDSQ4, ZDS, ZDS2, ZDS3, ZDS4, ZMEM, and ZORG are blank when control is passed to the CLIST referenced in variable ZSEL in the option panel. See “Required option entry panel variables” on page 156.

Generated JCL

The job control language (JCL) statements generated by the Batch processing option are accumulated in a temporary data set named *userid*.SPFTEMP*n*.CNTL or *prefix*.*userid*.SPFTEMP*n*.CNTL, where *n* is the screen number. This data set contains the job cards generated by ISRJB1 and job steps generated by the option CLISTs. In all cases, file tailoring services are used to generate the JCL. If VIO is used, the generated JCL is in a VIO data set allocated to DDname ISPCTL1.

ISRJB1 generates up to four job statement lines from information that you supply in the Batch Selection panel, ISRJPA. A job statement line is not generated if the corresponding field on the panel is blank. An attempt is made to provide unique jobnames by using this algorithm:

1. The four lines entered in the selection panel are scanned for the first ‘//’ card.
2. If the string following the ‘//’ is equal to the TSO user ID, and if the user ID is followed by an alphabetic character or a numeric character, that character is automatically incremented each time a job is submitted.
3. If the string following the ‘//’ is not equal to the TSO user ID, the job name will not be automatically incremented each time a job is submitted. ISRJB1 increments and changes jobnames only if they begin with the user ID.

If you bypass the Batch Selection panel (for example, by selecting option 5.n from the ISPF Primary Option Menu), the job statement lines that would have been displayed on the selection panel are used as if you had not modified them.

The file skeleton associated with each option controls the JCL generated for that option. See "Batch (Option 5)" in the *z/OS ISPF User's Guide Vol II* for a general description of skeleton formats. Each variable in the skeleton is replaced by its current value (the contents of the corresponding panel input field, as entered by you, or a previously entered value). Following variable substitution, each record in the skeleton is written to the temporary data set, which is eventually submitted to the job stream by the TSO SUBMIT command. Skeletons must be coded so that the maximum length of a record after substitution does not exceed 71 characters; otherwise, invalid JCL might be generated.

Variables that control option panel redisplay

After an option has been processed, the Batch Selection panel with Job Step Generated is normally displayed, thereby allowing you to add another job step to the one just generated. It is possible, however, to link options together automatically by using the shared variable ZNEXTPN. The program ISRJB2 sets ZNEXTPN to blank. After the completion of a option CLIST, ISRJB2 references the variable again to determine the next panel to display. If you wish, for example, to display the linkage edit panel (ISRJP07) after processing the FORTRAN panel (ISRJP03), set ZNEXTPN to ISRJP07 either in the)PROC section of the FORTRAN panel or in the FORTRAN CLIST (ISRJC03), and place the variable in the shared pool. You can extend this type of panel linkage to any length you want.

Variable used by Batch CLISTs

In the distributed Batch option, all option panels select a CLIST that uses file tailoring services to generate the JCL job steps. (Only a CLIST or program can be selected from the option panel.) To obtain variables set from the panel and by program ISRJB2, the CLIST references the shared variable pool. The CLIST performs basic error checking on some of the variables associated with the panel fields. After generating the JCL, the CLIST returns to ISRJB2. If, because of some error condition, the CLIST does not generate a job step, a nonzero return code is returned to ISRJB2.

Batch skeletons

The skeleton library contains the JCL and file tailoring control statements that are used to generate the JCL for a particular Batch option. A primary skeleton is associated with each option. This skeleton embeds other skeletons based on input data set type. For option 1, for example, skeleton ISRJS01 is used. If the data set was defined in the ISPF library section of the panel, ISRJS01 embeds skeleton ISRJS01I for appropriate JCL. If an "other" partitioned data set were specified, skeleton ISRJS01P would be embedded. If an "other" sequential data set were specified, skeleton ISRJS01S would be embedded.

Batch modifications

Most of the Batch processing logic is in the option panels, CLISTs, and skeletons to make modifications as easy as possible. Before modifying existing options or adding new ones, you should study the distributed panels, CLISTs, and skeletons to understand the relationships among them.

ISPF does not attempt to diagnose all possible types of skeleton coding errors. Some types of errors are detected, and the appropriate error messages are displayed, but in many cases coding errors must be debugged by inspection of the generated JCL. To perform this inspection, enter the CANCEL command when the Batch Selection panel with Job Step Generated is displayed, and then browse the temporary data set that contains the generated JCL. (The temporary data set is closed, but not freed or erased, when the CANCEL command is specified.)

Note: This procedure cannot be used if the temporary CNTL data sets have been allocated to VIO because they have MVS-generated data set names.

Steps to add a new Batch primary option to PDF

The steps required to add a new PDF primary option that uses the Batch processing mechanism are listed here.

1. Add the new option (for example, '99') to the ISPF Primary Option Menu, panel name ISR@PRIM. In the translated value for option 99, use the PGM keyword to specify that program ISRJB1 is to receive control, and use the PARM keyword to pass the name of the Batch Selection panel. Specify the NOCHECK option to allow ISRJB1 to select subsequent panels. Set the ZTRAIL variable to the value remaining from the truncation function. (ZTRAIL is already set in panel ISR@PRIM, but you must set it yourself if you invoke ISRJB1 from some other selection panel.) For example:

```
)PROC
&ZSEL; = TRANS( TRUNC(&ZCMD,'.'
   . . .

  99,'PGM(ISRJB1) PARM(ZNEW99A) NOCHECK '
```

```
    )
&ZTRAIL; = .TRAIL
```

2. Add a new Batch Selection panel (ZNEW99A in this example) to the panel library. Use panel ISRJPA as a model. For each option on the panel, use the PGM keyword to specify that program ISRJB2 is to receive control, and use the PARM keyword to pass the name of the option panel.
3. In the panel definition for ZNEW99A, set the variable RTNPNL to ZNEW99B (as an example) to specify the corresponding Batch Selection panel with Job Step Generated. Add new panel ZNEW99B to the panel library, using panel ISRJPB as a model.
4. Develop new option panels, CLISTs, and skeletons as described in "Steps to add a new Batch option to the PDF Batch Selection Panel" on page 173.

Steps to add a new Batch option to the PDF Batch Selection Panel

The steps required to add a new option to the Batch Selection Panel and the panel showing Job Step Generated are listed here.

1. Add the new option (for example, '99') to panels (ISRJPA and ISRJPB). Use the PGM keyword to specify that program ISRJB2 is to receive control and use the PARM keyword to pass the name of the new option panel. For example:

```

)PROC
&ZSEL = TRANS(&ZCMD
   . . .

   99,'PGM(ISRJB2) PARM(ZNEW99 99)'
```

2. Add new option panel ZNEW99 to the panel library. Use one of the distributed option panels (for example, ISRJP01) as a model.
3. Develop a corresponding CLIST (referenced from panel ZNEW99 by the ZSEL variable) and skeleton (referenced from the CLIST by the file tailoring FTINCL service). Use the distributed CLISTs and skeletons (for example, ISRJC01 and ISRJS01) as models. Add the CLIST to a library accessible to ddname SYSPROC, and add the skeleton to the skeleton library.

You can also develop a new option that displays a lower-level selection panel, from which user selections invoke the Batch CLISTs.

For example:

```
)PROC
&ZSEL = TRANS (&ZCMD
    1,'PGM(ISRJB2) PARM(ISRJP01)'
    2,'PGM(ISPJB2) PARM(ISRJP02)'
    . . .

    9,'CMD(ZNEWCMD &ZCMD)'
```

In this example, the CLIST ZNEWCMD has been selected, and the parameter ZCMD (the option entered on the panel) has been passed to the CLIST. ZNEWCMD can then use the SELECT service to display a lower-level selection panel similar to panel ISRJPA, but without job statement information. When the options have been processed, the CLIST should end and return control to ISRJB1 to close the temporary data set and submit the job to TSO. If JCL was generated the invoked dialog (CLIST or PANEL) must end with a return code of 0. If no JCL was generated the dialog must either end with a return code greater than 0 or set dialog variable ZADARC to a Y in either the ISPF SHARED or PROFILE pool.

Note: If the Source Data Online field, variable ZDSCKO, is set to YES, the dialog variables required for batch option panels are the same as those for foreground processing. That is, variables PRJ1, LIB1, LIB2, LIB3, LIB4, TYP1, MEM, DSN, and ZSEL must be defined in either the)INIT,)BODY, or)PROC section of the panel displayed by ISRJB2.

If the Source Data Online, variable ZDSCKO, is set to NO, these variables are not required. However, if a data set name is not supplied to ISRJB2 in these variables, the output variables ZDSQMEM, ZDSQ, ZDSQ2, ZDSQ3, ZDSQ4, ZDS, ZDS2, ZDS3, ZDS4, ZMEM, and ZORG are blank when control is passed to the CLIST referenced in variable ZSEL in the option panel.

Batch processing options considerations

The SISPLPA data set includes the ISRSCAN load module. It is not executed under TSO but is invoked from batch jobs submitted through the ISPF PDF batch processing option. The ISRSUPC and ISRLEMX load modules are executed in both foreground and batch.

To submit jobs through the ISPF batch processing option, the load modules ISRSCAN, ISRSUPC, and ISRLEMX must be available. If they are not in your LPA or link library, insert a STEPLIB DD statement for the library that contains them in each skeleton member ISRJS*xxx* and ISRSBJCL in the ISP.SISPSLIB data set. Insert the statement following the EXEC PGM=ISRSCAN, EXEC PGM=ISRLEMX, or EXEC PGM=ISRSUPC statement, whichever is in the skeleton (see the comments in each of these members).

ISRSCAN and ISRLEMX programs

The ISPF-supplied Batch skeletons each generate one or more job steps. The first step executes either the ISRSCAN program or the ISRLEMX program. Both programs find the input member and copy it from the library hierarchy (up to four partitioned data sets) to a temporary data set for input to the processing program executed in the second step. The difference between them is that ISRSCAN copies only one member to the output data set, while ISRLEMX copies the primary member and expands any included members as part of the output data set. All members copied by ISRLEMX are unpacked in the output data set and ISRSCAN does not unpack the input while ISRLEMX does. The input to ISRSCAN must not be packed. ISRLEMX also creates a member parts list (see ISPF Options 4.12 and 5.12).

If your input is a sequential data set, ISRLEMX only copies the data to the output data set. **It unpacks the input but does not unpack any included members**.

The first step is essential for the operation of library hierarchies because the ISPF-supplied processing programs typically accept primary input only from fully qualified data sets (that is, from either a sequential data set or a single member of a single partitioned data set, not from concatenations of partitioned data sets).

Using ISRSCAN: The Batch job steps for using ISRSCAN are as follows:

```
//*ISRSCAN step
//SCANSTEP EXEC PGM=ISRSCAN,PARM='member name'
//STEPLIB DD DSN= (Library name if ISRSCAN is not in
       your system library)
//IN      DD ... Input data set in which the source
       member is found.
//OUT     DD ... Sequential output data set.
/*
```

Return Codes: ISRSCAN sets one of these return codes in register 15:

0
Normal completion

8
One of these:

- DDNAME OUT not found.
- Error retrieving data set information for OUT data set.
- OUT data set is a PDS without a member specified.

12
Member not found

16
Unable to open input DCB

20
: I/O error on input data set

24
: Unable to open output DCB

28
: I/O error on output data set

Using ISRLEMX: ISRLEMX Batch job steps are shown here.

```
//*ISRLEMX step
//LEMXSTEP EXEC PGM=ISRLEMX,PARM='parm1,parm2,...,parm15'
//STEPLIB DD DSN= (Library name if ISRLEMX is not in your
       system library)
//ISRLCODE DD ... Input data set in which the source member(s)
       are found.
//ISRLEXPD DD ... Sequential output data set for expansion.
//ISRLXREF DD ... Sequential output data set for mem prts lst.
//ISRLMSG DD SYSOUT=A /* program error messages print here */

  - or -

//ISRLMSG DD DSN=&ZPREFIX;.&LID;.LMSG,UNIT=SYSDA,
             SPACE=(TRK,(1,1)),DISP=(MOD,CATLG),
             DCB=(RECFM=FBA,LRECL=133,BLKSIZE=3059)
/*
```

Note: DDNAME ISRLCODE can be a concatenation of more than 8 data sets, but only the first 8 will be listed in the member parts list.

Return Codes: ISRLEMX sets one of these return codes in register 15:

0
: Normal completion

n
: Parameter *n* is 1-15 (too long)

16
: Too many parameters

17
: Too few parameters

20
: Severe error in expand module—an error is printed in the ISRLMSG data set.

Note:

1. DDNAME ISRLEXPD is needed if parameter 5 is either ‘E’ or blank. DDNAME ISRLXREF is needed if parameter 5 is ‘L’. See this figure for additional information.
2. ISRLEMX reads data that is presented in a BSAM compatible form. If the SUBSYS parameter (or any parameter that will cause the UCB pointer in the TIOT to be 0) is used, PDF cannot verify that the input is on DASD. It is the responsibility of the user to ensure ISRLEMX will see the data in the correct form. If the data is not presented in a BSAM compatible form, results will be unpredictable.

The ISRLEMX parameter string contains up to 15 parameters, each separated from the next by commas. The parameters are:

Table 27. Parameters in the ISRLEMX parameter string

Parameter	Length	Description
1	CHAR(3)	Language type of the input member to be processed: ASM - Assembler COB - COBOL FOR - FORTRAN PAS - Pascal PLI - PL/I SCR - Script
2	CHAR(8)	Member name of member to be expanded or of first member to be processed to create a member parts list.
3	CHAR(1)	'B' - The request is being run in batch 'F' - The request is being run in foreground
4	CHAR(1)	'Y' - Allocate a temporary sort data set if needed 'N' - Do not allocate a sort data set
5	CHAR(1)	'E' - Expand and unpack the specified member and all included members into one sequential data set 'L' - Create a member parts list starting with the specified member blank - Copy the specified input member to a temporary data set, unpacking the member but not expanding. This is the default that is used when a comma is entered to skip this parameter.
6	CHAR(3)	Number of concatenated input libraries that should be scanned to find the specified input member. The value can be 1 - 255. If the number specified is equal to or larger than the number of concatenated input libraries, all concatenated input libraries are scanned.
7	CHAR(20)	User trigger, a character string of maximum length 20 to be processed as an INCLUDE, COPY or IMBED statement when found in the member being processed. Enter a comma to skip this parameter if no user trigger is being used.
8	CHAR(2)	User trigger start column, specifies which column the user trigger listed here will start in the member being processed. Enter a comma to skip this parameter if no user trigger is being used.
9	CHAR(3)	Indicates the National language in use: ENU - English DEU - German (Deutsch) KAN - Japanese (Kanji) DES - Swiss German This is used to build the name of the literal load module that is loaded by member expansion. If the language is not English and the load fails, the English table is loaded.
10	CHAR(1)	Position of the month value in the date string; for example, *yy/mm/dd* (default '4')
11	CHAR(1)	Position of the day value in the date string; for example, *yy/mm/dd* (default '7')

Table 27. Parameters in the ISRLEMX parameter string (continued)

Parameter	Length	Description
12	CHAR(1)	Position of the year value in the date string; for example, *yy/mm/dd* (default '1')
13	CHAR(1)	Delimiter to use in the date string; for example, *yy/mm/dd* (default '/')
14	CHAR(8)	Unit name to be used for all temporary data sets used by ISRLEMX. The unit name must be specified.
15	CHAR(4)	The number of blocks used when allocating the temporary sort data sets. These are allocated to ddnames ISRKLWKnn, where nn is 1 through 4. The default for this parameter is 0100. It can be increased for very large or complex expansions.

Note: You can specify the date notation in the national language format (for example, `yy/mm/dd, mm/dd/yy, dd/mm/yy`). Therefore, you must specify the index for each portion. Use any valid character to delimit the date string (for example, `yy/mm/dd, dd.mm.yy`). This delimiter is required.

Adding user-defined triggers

During installation time, you can add a user-defined trigger to the member expansion or member parts list functions, or to the foreground or batch options of PDF. The user defined trigger is interpreted as though it were an INCLUDE, COPY, or `.im` keyword.

Adding a user-defined trigger to the member expansion function

The user can define the member expansion function trigger by using up to 20 characters. The trigger must start in the user-defined start column and be followed by at least one blank. The next nonblank string is interpreted as the included member name. For more information about member expansion and triggers, see *z/OS ISPF User's Guide Vol II*.

Adding a user-defined trigger to the Foreground options

To define a user trigger for the PDF Foreground options, change the ISRLEMX SELECT statement in the CLIST for the appropriate option.

Change from:

```
ISPEXEC SELECT PGM(ISRLEMX) +
 PARM('XXX,&ZMEM,F,N,E,4, ,00,&ZFPRLANG,&ZFPRMMIX,+
 &ZFPRDDIX,&ZFPRYYIX,&ZFPRDLIM,&Z4UNIT')
```

To:

```
ISPEXEC SELECT PGM(ISRLEMX) +
 PARM('XXX,&ZMEM,F,N,E,4,trigger,nn,&ZFPRLANG,&ZFPRMMIX,+
 &ZFPRMMIX,&ZFPRDDIX,&ZFPRYYIX,&ZFPRDLIM,&Z4UNIT')
```

where *nn* is the trigger start column.

The CLIST names are as follows:

```
OPTION 4.1     ISRFC01
OPTION 4.2     ISRFC02
OPTION 4.3     ISRFC03
OPTION 4.5     ISRFC05
OPTION 4.6     ISRFC06
OPTION 4.9     ISRFC09
```

Adding a user-defined trigger to the Batch options

To define a user trigger for the PDF Batch options, change the ISRLEMX EXEC statement in the skeleton for the appropriate option.

Change from:

```
//EXPAND EXEC PGM=ISRLEMX,COND=(12,LE),
//  PARM=('XXX,&ZMEM,B,N,E,4, ,00,&ZJB2LANG,;
//  &ZJB2MMIX,&ZJB2DDIX,&ZJB2YYIX,&ZJB2DLIM,&Z5UNIT')
```

To:

```
//EXPAND EXEC PGM=ISRLEMX,COND=(12,LE),
//  PARM=('XXX,&ZMEM,B,N,E,4,trigger,nn,&ZJB2LANG,;
//  &ZJB2MMIX,&ZJB2DDIX,&ZJB2YYIX,&ZJB2DLIM,&Z5UNIT')
```

where *nn* is the trigger start column.

The skeleton names are as follows:

```
OPTION 5.1    ISRJS01I, ISRJS01J, ISRJS01P, ISRJS01S
OPTION 5.2    ISRJS02I, ISRJS02J, ISRJS02P, ISRJS02S
OPTION 5.3    ISRJS03I, ISRJS03J, ISRJS03P, ISRJS03S
OPTION 5.5    ISRJS05I, ISRJS05J, ISRJS05P, ISRJS05S
OPTION 5.6    ISRJS06I, ISRJS06J, ISRJS06P, ISRJS06S
```

Adding a user-defined trigger to the member parts list function

The user can define the trigger for the member parts list function by using up to 20 characters. The trigger must start in the user-defined start column and be followed by at least one blank. The next nonblank string is interpreted as the included member name.

To define the trigger, change two SET statements in the CLISTs ISRFC12 (Foreground) and ISRJC12 (Batch), as follows:

Change from:

```
SET &UT = &STR(&Z)
SET &UTC = &STR(00)
```

To:

```

SET &UT = &STR(trigger)
SET &UTC = &STR(nn)
```

where *nn* is the trigger start column.

Changing the size of the data set for the member parts list function

The member parts list function allocates a temporary data set for collecting cross-reference data. The default size of this data set is 100 blocks each of primary and secondary storage. To change this size, change the SSPACE variable in the panels ISRFP12 (foreground) and ISRJP12 (batch), as follows:

Change from:

```
)PROC
&SSPACE = '0100'
```

To:

```
)PROC
&SSPACE = 'nnnn'
```

where *nnnn* is the number of blocks to be allocated.

Customizing Browse and Edit

Customizing Browse and Edit describes how to provide customized Browse and Edit panels, and how to enable users to browse data that is stored in a Unicode format.

Providing customized Browse and Edit panels

Dialog developers can provide customized Browse or Edit data display panels for the dialogs they create. The name of the customized panel is passed as a parameter to the BROWSE or EDIT dialog service.

Note: Do not use the names ISREDDE, ISREDDE2, ISREDDE3, ISREDDE4, or ISREDDE5 as a panel name passed to the EDIT or VIEW services. When ISPF is using any of these panels, it can dynamically switch among any of these panels based on system configuration and edit profile settings.

Customized Browse panels must be patterned after the distributed panel ISRBROB (for Browse, as shown in Figure 61 on page 180). Customized Edit panels must be patterned after these panels:

ISREFR01 - Edit without action bars or extended highlighting (as shown in Figure 62 on page 181)
ISREFR02 - Edit with action bars and extended highlighting
ISREFR03 - Edit with action bars and no extended highlighting
ISREFR04 - Edit with extended highlighting but no action bars

The customized panel must include a Command field named ZCMD and a dynamic area named ZDATA. As shown in Figure 62 on page 181, this ZDATA dynamic area should be coded with an attribute that is defined with AREA(DYNAMIC). It is not acceptable to code this area with an attribute defined as AREA(SCRL). To enable extended highlighting, the customized edit panel must have a shadow variable called ZSHADOW and must have the attributes as defined in the attribute section of panel ISREFR04. Customized panels should meet the requirements for a movable Command line, so that the Command line can be displayed at the bottom of the screen at your request. See *z/OS ISPF Dialog Developer's Guide and Reference* for a discussion of these requirements.

Inclusion of a Scroll field is optional. The scroll fields in the distributed panels are named ZSCBR (for Browse) and ZSCED (for Edit). You can use these same names for scroll fields on customized panels. Inclusion of the (protected) variables that appear in the title line of the distributed panels is optional. If you want to display the volume of the data set, you can use variable ZDSVOL.

Because Edit and Browse translate non-displayable characters into blanks (for Edit) or a user-specified character (for Browse), hex codes that represent these characters cannot be placed in the data with the intent of using them as attribute characters. If you want to modify the Edit or Browse display, you must either use displayable characters, or redefine the existing Edit and Browse attributes. If you redefine an existing attribute, do not change the TYPE or FORMAT values of the attribute.

The HIDE EXCLUDED command uses reserved attribute characters to underscore the number field of the line preceding the excluded message line. As some customized Edit panels might already use these attribute characters, ISPF uses the dialog variable ZHIDEX to determine whether the attribute characters are available and correctly defined in the Edit panel. An example of setting ZHIDEX and providing the reserved attribute bytes 13, 16, 17, and 1D is shown in Figure 62 on page 181.

If the HIDE command is used in an initial macro, then the macro must set the ISPF dialog variable ZHIDEX to a value of ?Y? and the Edit panel in use must contain the required attribute character definitions as previously described.

Note, even when the HIDE EXCLUDED command is enabled using the ZHIDEX variable, the number field of the line preceding the excluded message is only underscored on ISPF-supported terminals with the extended highlighting feature available.

```
)ATTR DEFAULT(   ) FORMAT(MIX)             /* ISRBROB - ENGLISH - 7.1 */
05 TYPE(PT)
09 TYPE(FP)
0A TYPE(NT)
13 TYPE(NEF) PADC(USER)
16 TYPE(VOI) PADC(USER)
26 AREA(DYNAMIC) EXTEND(ON) SCROLL(ON)
01 TYPE(DATAOUT) INTENS(LOW)
02 TYPE(DATAOUT)
0B TYPE(DATAOUT) FORMAT(DBCS) OUTLINE(L)
0C TYPE(DATAOUT) FORMAT(EBCDIC) OUTLINE(L)
0D TYPE(DATAOUT) FORMAT(&MIXED) OUTLINE(L)
10 TYPE(DATAOUT) INTENS(LOW) FORMAT(DBCS) OUTLINE(L)
11 TYPE(DATAOUT) INTENS(LOW) FORMAT(EBCDIC) OUTLINE(L)
12 TYPE(DATAOUT) INTENS(LOW) FORMAT(&MIXED) OUTLINE(L)
)BODY  EXPAND(//) WIDTH(&ZWIDTH)  CMD(ZCMD)
BROWSE    Z/ /                                       Line Z         Col Z
Command ===> Z/ /                                             Scroll ===> Z
ZDATA/ /
/ /
/ /
)INIT
.ZVARS = '(ZTITLB ZLINES ZCOLUMS ZCMD ZSCBR)'
.HELP = ISR1B000
&ZCMD = ' '
VGET (ZSCBR) PROFILE     /* Fill Scroll Vars if       */
IF (&ZSCBR = ' ')  &ZSCBR  = 'PAGE'  /* Blank with Page */
IF (&ZMEMB ^= ' ') &ZTITLB = '&ZDSNT(&ZMEMB)&ZLEVEL ' /* OZ91708 */
IF (&ZMEMB = ' ')  &ZTITLB = '&ZDSNT&ZLEVEL '
&MIXED = MIX
IF (&ZPDMIX = N) &MIXED = EBCDIC
*REXX(ZSCR,ZTITLB)
if length(ztitlb) > 53-9-2
then zscr = 'ON'
else zscr = 'OFF'
*ENDREXX
)REINIT
REFRESH(ZCMD,ZSCBR,ZDATA,ZLINES,ZCOLUMS,ZTITLB)
)PROC
&ZCURSOR = .CURSOR
&ZCSROFF = .CSRPOS
VPUT (ZSCBR) PROFILE     /*                           */
&ZLVLINE = LVLINE(ZDATA)
)FIELD
FIELD(ZTITLB) SCROLL(ZSCR)
)END
/* 5694-A01     COPYRIGHT IBM CORP 1995, 2011 */
/* ISPDTLC Release: 7.1.  Level: PID                                  */
```

Figure 61. Browse panel (ISRBROB)

```
)ATTR /* EDIT PANEL WITH NO ACTION BAR & NO HIGHLIGHTING */
  _ TYPE(INPUT) CAPS(OFF) INTENS(HIGH) FORMAT(&MIXED)
  | AREA(DYNAMIC) EXTEND(ON) SCROLL(ON) USERMOD(20)
  ! TYPE(OUTPUT) INTENS(HIGH) PAD(-)
 01 TYPE(DATAOUT) INTENS(LOW)
 02 TYPE(DATAOUT) INTENS(HIGH)
 03 TYPE(DATAOUT) SKIP(ON) /* FOR TEXT ENTER CMD. FIELD */
 04 TYPE(DATAIN)  INTENS(LOW)  CAPS(OFF) FORMAT(&MIXED)
 05 TYPE(DATAIN)  INTENS(HIGH) CAPS(OFF) FORMAT(&MIXED)
 06 TYPE(DATAIN)  INTENS(LOW)  CAPS(IN)  FORMAT(&MIXED)
 07 TYPE(DATAIN)  INTENS(HIGH) CAPS(IN)  FORMAT(&MIXED)
 08 TYPE(DATAIN)  INTENS(LOW)  FORMAT(DBCS) OUTLINE(L)
 09 TYPE(DATAIN)  INTENS(LOW)  FORMAT(EBCDIC) OUTLINE(L)
 0A TYPE(DATAIN)  INTENS(LOW)  FORMAT(&MIXED) OUTLINE(L)
 0D TYPE(DATAIN)  INTENS(LOW)  CAPS(IN)  FORMAT(&MIXED) COLOR(BLUE)
 13 TYPE(DATAOUT) SKIP(ON) HILITE(USCORE)
 16 TYPE(DATAIN) INTENS(LOW) CAPS(IN) HILITE(USCORE) FORMAT(&MIXED)
 17 TYPE(DATAIN) CAPS(IN) HILITE(USCORE) FORMAT(&MIXED)
 1D TYPE(DATAIN) INTENS(LOW) CAPS(IN) COLOR(BLUE) HILITE(USCORE)
   FORMAT(&MIXED)
 20 TYPE(DATAIN)  INTENS(LOW) CAPS(IN) FORMAT(&MIXED)
)BODY WIDTH(&ZWIDTH) EXPAND(//)
%EDIT -----!ZTITLE -------------------/-/------------------%COLUMNS!ZCL
%COMMAND ===>_ZCMD                    / /                     %SCROLL ==
|ZDATA -------------------------------/-/------------------------------
|                                     / /
| -----------------------------------/-/------------------------------
)INIT
  &ZHIDEX = 'Y'            /* Indicate the presence of  */
                           /* attrs 13, 16, 17 and 1D   */
                           /* for HIDE EXCLUDED         */
  IF (&ZVMODET = 'VIEW')   /* VIEW MODE                 */
    .HELP = ISR10000       /* DEFAULT TUTORIAL NAME     */
  ELSE                     /* EDIT MODE                 */
    .HELP = ISR20000       /* DEFAULT TUTORIAL NAME     */
  .ZVARS = 'ZSCED'         /* SCROLL AMT VARIABLE NAME  */
  &MIXED = MIX             /* SET FORMAT MIX            */
  IF (&ZPDMIX = N)         /* IF EBCDIC MODE REQUESTED  */
    &MIXED = EBCDIC        /*  SET FORMAT EBCDIC        */
  VGET (ZSCED) PROFILE     /* FILL SCROLL VARS IF       */
  IF (&ZSCED = ' ')        /* BLANK WITH PAGE.          */
   &ZSCED = 'PAGE'         /*                           */
)REINIT
  REFRESH(ZCMD,ZSCED,ZDATA,ZTITLE,ZCL,ZCR)
  IF (&ZVMODET = 'VIEW')   /* VIEW MODE                 */
    .HELP = ISR10000       /* DEFAULT TUTORIAL NAME     */
  ELSE                     /* EDIT MODE                 */
    .HELP = ISR20000       /* DEFAULT TUTORIAL NAME     */
)PROC
  &ZCURSOR = .CURSOR
  &ZCSROFF = .CSRPOS
  &ZLVLINE = LVLINE(ZDATA)
  VPUT (ZSCED) PROFILE
)END
/*******************************************************************/
/* Use variable ZDSVOL to display the volume of the data set       */
/*******************************************************************/
/*  DYNAMIC AREA SCREEN WIDTH FROM PQUERY. (80,132,160)
/*  DYNAMIC AREA SCREEN DEPTH FROM PQUERY. (24,32,43,27,60)
/*                                                                  */
/* 5645-001, 5655-042 (C) COPYRIGHT IBM CORP 1980, 1996             */
```

Figure 62. Edit panel (ISREFR01)

The distributed panels have a dynamic area that automatically fills the available width and depth of the screen (after the first two lines). A customized panel can have a dynamic area that is fixed in width (in which case the WIDTH and EXPAND keywords should be omitted from the)BODY header statement), fixed in depth (in which case the EXTEND keyword should be omitted on the AREA statement in the)ATTR section), or fixed in both width and depth.

Note: If the dynamic area is less than the full width of the screen, the panel definition must include attribute bytes for protected fields on either side of the area.

See the description of dynamic areas in the *z/OS ISPF Dialog Developer's Guide and Reference*.

A customized panel can have additional text, input, or output fields outside the dynamic area. Any additional variable fields are transparent to the Browse and Edit programs. They can, however, be processed by the dialog that invoked Browse or Edit, or by edit macros provided by the dialog developer. Additional variable fields that appear on the panel should be refreshed (by using a REFRESH statement) in the)REINIT section of the panel definition. This ensures that the display screen is updated with the current contents of the variables each time the panel is displayed, which is after execution of each macro or built-in command, including SCROLL commands.

Any variables that are to be passed from the panel to an edit macro should be stored in the shared or profile pool by including a VPUT statement in the)PROC section of the panel definition. The macro must then issue a VGET to obtain them. The reason is that macros operate as nested dialogs, with a separate function pool from that of the dialog that invokes Edit.

The statements that appear in the)ATTR,)INIT,)REINIT, and)PROC sections of the distributed panels should be included in customized panels, with the possible exception of the EXTEND, WIDTH, and EXPAND keywords (discussed here) and the initialization of the scroll fields. Additional keywords can be added to the attribute definitions (for example, to produce different colors), but the same hexadecimal representation of the attribute bytes must be maintained, including the X'20' specified on the USERMOD keyword in the Edit panel.

The attribute keywords of the last attribute byte set before the dynamic area and of the attribute byte represented by X'01' should be the same for panel ISRBROB and any panels using it for an example.

Table 28 on page 182 and Table 29 on page 182 list the ISPF-provided output-only variables you can use on Browse and Edit panels.

Table 28. Browse output-only variables

Variable	Description	Format/Length
ZCOLUMS	First and last columns being displayed	CHAR 7
ZCSROFF	Cursor offset within cursor field	CHAR 4
ZCURSOR	Name of field where cursor is placed	CHAR 8
ZDADWD	Width of the dynamic area (ZDATA)	CHAR 4
ZDAMLN	Length of the dynamic area (ZDATA)	CHAR 4
ZDSN	Name of data set being displayed	CHAR 44
ZDSNT	Name of data set or file being displayed	CHAR 1023
ZLEVEL	Version and mod level of member	CHAR 8
ZLINES	Top line of the data display	CHAR 8
ZLVLINE	Last visible line of ZDATA after last interaction	CHAR 4
ZMEMB	Name of member being displayed	CHAR 8
ZWIDTH	Width of the panel	CHAR 4
ZDSVOL	Volume of the data set or first library in the concatenation	CHAR 6

Table 29. Edit output-only variables

Variable	Description	Format/Length
ZCL	Left column of the data display	CHAR 5
ZCR	Right column of the data display	CHAR 5
ZCSROFF	Cursor offset within cursor field	CHAR 4
ZCURSOR	Name of field where cursor is placed	CHAR 8

Table 29. Edit output-only variables (continued)

Variable	Description	Format/Length
ZDADLN	Length of the dynamic area (ZDATA)	CHAR 4
ZDADWD	Width of the dynamic area (ZDATA)	CHAR 4
ZDSN	Name of data set being displayed	CHAR 44
ZLEVEL	Version and mod level of member	CHAR 8
ZLVLINE	Last visible line of ZDATA after last interaction	CHAR 4
ZMEMB	Name of member being displayed	CHAR 8
ZTITLE	The title line DSN(MEMB) -VER.MOD	CHAR 1023
ZWIDTH	Width of the panel	CHAR 4
ZDSVOL	Volume of the data set or first library in the concatenation.	CHAR 6

Enabling Browse panels to display Unicode data

You can enable users to browse data that is stored in a Unicode format. To do this, MVS Conversion Services must first be set up for the appropriate conversions. See *z/OS Unicode Services User's Guide and Reference*.

Customizing member list panels

Dialog developers can also provide customized member list display panels for their dialogs. The name of the customized member list display panel is passed as a parameter to the LMMDISP service.

A customized member list panel should be modeled after the LMMDISP default member list panel, ISRML000.

Note: Panel ISRML000 is provided as both source code (written in Dialog Tag Language, or DTL) and as generated output. If you use this panel as a model for creating your own customized panel, be sure to copy the DTL source code for modification, as it is easier to modify than the generated output panel. The DTL source code can be found in one of the SISPG*xxx* libraries, where *xxx* is the designator for a specified language.

The customized panel must include a Command field named ZCMD and a dynamic area named ZDATA. In addition, for customized member list panels, the dynamic area must be 80 characters wide. The Scroll field is optional. The name of the Scroll field on member list panels is ZSCML. This same name can be used on a customized member list panel. Inclusion of the variables that output the data set name, relative row number, and total number of rows is optional. If you want to display the volume of the data set you can use variable ZDSVOL. Customized member list panels should also meet the requirements for a movable Command line. Figure 63 on page 184 shows an example of a customized member list panel that was generated by Dialog Tag Language code.

See the *z/OS ISPF Dialog Developer's Guide and Reference* for a discussion of these needs.

```
)PANEL KEYLIST(ISRSPBC,ISR)
)ATTR DEFAULT(???) FORMAT(MIX)
 0B TYPE(AB)
 0D TYPE(PS)
 2D TYPE(ABSL) GE(ON)
 2E TYPE(PT)
 28 TYPE(FP)
 0A TYPE(NT)
 13 TYPE(NEF) PADC(USER)
 16 TYPE(VOI) PADC(USER)
 26 AREA(DYNAMIC)
 08 TYPE(DATAOUT) PAS(ON) CSRGRP(99)
 09 TYPE(DATAOUT)
 29 AREA(DYNAMIC) EXTEND(ON) SCROLL(ON)
 01 TYPE(DATAIN) CAPS(ON) PADC(&ZMLPAD) PAS(ON)
 02 TYPE(DATAOUT) INTENS(&MLI2) SKIP(ON) COLOR(&MLC2) HILITE(&MLH2)
 03 TYPE(DATAIN) INTENS(&MLI5) CAPS(ON) COLOR(&MLC5) HILITE(&MLH5)
 04 TYPE(DATAOUT) INTENS(&MLI3) COLOR(&MLC3) HILITE(&MLH3)
 05 TYPE(DATAOUT)
 06 TYPE(DATAOUT) INTENS(LOW)
 14 TYPE(NEF) CAPS(ON) PADC(USER)
)ABC DESC('Menu') MNEM(1)
PDC DESC('Settings') UNAVAIL(ZPM1) MNEM(1) ACC(CTRL+S)
 ACTION RUN(ISRROUTE) PARM('SET')
PDC DESC('View') UNAVAIL(ZPM2) MNEM(1) ACC(CTRL+V)
 ACTION RUN(ISRROUTE) PARM('BR1')
PDC DESC('Edit') UNAVAIL(ZPM3) MNEM(1) ACC(CTRL+E)
 ACTION RUN(ISRROUTE) PARM('ED1')
PDC DESC('ISPF Command Shell') UNAVAIL(ZPM4) MNEM(6) ACC(CTRL+C)
 ACTION RUN(ISRROUTE) PARM('C1')
PDC DESC('Dialog Test...') UNAVAIL(ZPM5) MNEM(8) ACC(CTRL+T)
 ACTION RUN(ISRROUTE) PARM('DAL')
PDC DESC('Other IBM Products...') UNAVAIL(ZPM6) MNEM(1) ACC(CTRL+O)
 ACTION RUN(ISRROUTE) PARM('OIB')
PDC DESC('SCLM') UNAVAIL(ZPM7) MNEM(3) ACC(CTRL+L)
 ACTION RUN(ISRROUTE) PARM('SCL')
PDC DESC('ISPF Workplace') UNAVAIL(ZPM8) MNEM(6) ACC(CTRL+W)
 ACTION RUN(ISRROUTE) PARM('WRK')
PDC DESC('Status Area...') UNAVAIL(ZPMS) MNEM(8) ACC(CTRL+A)
 ACTION RUN(ISRROUTE) PARM('SAM')
PDC DESC('Exit') MNEM(2) PDSEP(ON) ACC(CTRL+X) ACTION RUN(EXIT)
)ABCINIT
.ZVARS=ISR@OPT
)ABC DESC('Functions') MNEM(1)
PDC DESC('Save List') MNEM(1) ACTION RUN(>SAVE)
PDC DESC('Change Colors') MNEM(1) ACTION RUN(>MLC)
)ABCINIT
.ZVARS=MEMOPT
)ABC DESC('Utilities') MNEM(1)
```

Figure 63. Customized member list panel (Part 1 of 3)

Customized member list panel (Part 2 of 3)

```
PDC DESC('Library') UNAVAIL(ZUT1) MNEM(1) ACC(ALT+1)
 ACTION RUN(ISRROUTE) PARM('U1')
PDC DESC('Data set') UNAVAIL(ZUT2) MNEM(1) ACC(ALT+2)
 ACTION RUN(ISRROUTE) PARM('U2')
PDC DESC('Move/Copy') UNAVAIL(ZUT3) MNEM(1) ACC(ALT+3)
 ACTION RUN(ISRROUTE) PARM('U3')
PDC DESC('Data Set List') UNAVAIL(ZUT4) MNEM(2) ACC(ALT+4)
 ACTION RUN(ISRROUTE) PARM('U4')
PDC DESC('Reset Statistics') UNAVAIL(ZUT5) MNEM(5) ACC(ALT+5)
 ACTION RUN(ISRROUTE) PARM('U5')

PDC DESC('Hardcopy') UNAVAIL(ZUT6) MNEM(8) ACC(ALT+6)
 ACTION RUN(ISRROUTE) PARM('U6')
PDC DESC('Reserved') UNAVAIL(ZUTDT) MNEM(1) ACTION RUN(ISRROUTE) PARM('UDT')
PDC DESC('Outlist') UNAVAIL(ZUT7) MNEM(1) ACC(ALT+8)
 ACTION RUN(ISRROUTE) PARM('U8')
PDC DESC('Commands...') UNAVAIL(ZUT8) MNEM(1) ACC(ALT+9)
 ACTION RUN(ISRROUTE) PARM('U9')
PDC DESC('Reserved') UNAVAIL(ZUT9) MNEM(6) ACTION RUN(ISRROUTE) PARM('U10')
PDC DESC('Format') UNAVAIL(ZUT10) MNEM(1) ACC(ALT+F1)
 ACTION RUN(ISRROUTE) PARM('U11')
PDC DESC('SuperC') UNAVAIL(ZUT11) MNEM(1) PDSEP(ON) ACC(CTRL+F2)
 ACTION RUN(ISRROUTE) PARM('U12')
```

```
PDC DESC('SuperCE') UNAVAIL(ZUT12) MNEM(2) ACC(CTRL+F3)
 ACTION RUN(ISRROUTE) PARM('U13')
PDC DESC('Search-For') UNAVAIL(ZUT13) MNEM(2) ACC(CTRL+F4)
 ACTION RUN(ISRROUTE) PARM('U14')
PDC DESC('Search-ForE') UNAVAIL(ZUT14) MNEM(6) ACC(CTRL+F5)
 ACTION RUN(ISRROUTE) PARM('U15')
PDC DESC('Table Utility') UNAVAIL(ZUT15) MNEM(3) ACC(CTRL+F6)
 ACTION RUN(ISRROUTE) PARM('U16')
PDC DESC('Directory List') UNAVAIL(ZUT16) MNEM(2) ACC(CTRL+F7)
 ACTION RUN(ISRROUTE) PARM('U17')
)ABCINIT
.ZVARS=PDFUTIL
&zutdt = '1'
&zut9 = '1'
)ABC DESC('Help') MNEM(1)
PDC DESC('General') MNEM(1) ACTION RUN(TUTOR) PARM('ISR01130')
PDC DESC('Scrolling') MNEM(1) ACTION RUN(TUTOR) PARM('ISR01131')
PDC DESC('Pattern matching') MNEM(1) ACTION RUN(TUTOR) PARM('ISR01232')
PDC DESC('LOCATE command') MNEM(1) ACTION RUN(TUTOR) PARM('ISR01132')
PDC DESC('SORT command') MNEM(2) ACTION RUN(TUTOR) PARM('ISR01226')
PDC DESC('SAVE command') MNEM(2) ACTION RUN(TUTOR) PARM('ISR01229')
PDC DESC('REFRESH command') MNEM(1) ACTION RUN(TUTOR) PARM('ISR01142')
PDC DESC('RESET command') MNEM(2) ACTION RUN(TUTOR) PARM('ISR01138')
PDC DESC('SELECT command') MNEM(5) ACTION RUN(TUTOR) PARM('ISR01133')
PDC DESC('MLC command') MNEM(1) ACTION RUN(TUTOR) PARM('ISR01237')
PDC DESC('S line command') MNEM(4) ACTION RUN(TUTOR) PARM('ISR01134')
PDC DESC('Statistics') MNEM(2) ACTION RUN(TUTOR) PARM('ISR01140')
PDC DESC('Appendices') MNEM(5) ACTION RUN(TUTOR) PARM('ISR00004')
PDC DESC('Index') MNEM(3) ACTION RUN(TUTOR) PARM('ISR91000')
)ABCINIT
```

Customized member list panel (Part 3 of 3)

```
.ZVARS=MEMLHELP
)BODY  CMD(ZCMD)
## Menu# Functions# Utilities# Help#
#-------------------------------------------------------------------------------
#MEMBER LIST #Z                                        #Row#Z      #of#Z      #
#Command ===>#Z                                              #Scroll ===>#Z   #
#ZMLCOLD                                                                      #
#ZDATA                                                                        #
#                                                                             #
)INIT
.ZVARS = '(ZDSN ZMLCR ZMLTR ZCMD ZSCML)'
.HELP = ISR01130
.ATTR(ZMLCR)='JUST(RIGHT) PAD(''0'')'
.ATTR(ZMLTR)='JUST(RIGHT) PAD(''0'')'
&zds   = &zdsn
&zscr = 'OFF'
&zt = LENGTH(zds)
IF (&zt > 42)
  &zscr = 'ON'
VGET (MLC1 MLC2 MLC3 MLH1 MLH2 MLH3 ZMLPD ZSCML) PROFILE
VGET (ZGUI)
&ZMLPAD = 'USER'                     /* Init to user pad char OW18007*/
IF (&ZGUI = ' ')                     /* JSON API              OW18007*/
  IF (&ZMLPD = ' ')                  /* User pad not wanted   OW18007*/
    &ZMLPAD = '.'                    /* Use default pad char  OW18007*/
IF (&MLC1 = ' ') &MLC1 = 'TURQ'
IF (&MLC2 = ' ') &MLC2 = 'BLUE'
IF (&MLC3 = ' ') &MLC3 = 'GREEN'
                                         /* Fill Scroll Vars if       */
IF (&ZSCML = ' ') &ZSCML = 'PAGE'        /* Blank with page.          */
)PROC
VPUT (ZSCML) PROFILE
IF (.CURSOR = ZDATA OR .CURSOR = ZMLCOLD) &ZMSCPOS = &ZCURPOS
ELSE &ZMSCPOS = '0000'
)FIELD
FIELD(ZDSN) SCROLL(ZSCR)
)PNTS
)END
/* 5650-ZOS     COPYRIGHT IBM CORP 1994, 2021 */ /* ISPDTLC Release: 7.5.  Level:
PID                                  */ /* z/OS 02.05.00.  Created - Date: 2 Jun 2020, Time:
08:21            */
```

As with edit and browse customized panels, the distributed member list panels have a dynamic area that extends to fill the available depth of the screen. But, unlike edit and browse panels, (the dynamic area on a member list panel does not expand to fill the width of the screen), it must be 80 characters wide. The dynamic area can also be fixed in length by removing the EXTEND keyword from the)ATTR section of the panel definition.

See the description of dynamic areas in the *z/OS ISPF Dialog Developer's Guide and Reference*.

See "Customizing Browse and Edit" on page 179 for a more complete discussion on customizing panels.

Table 30 on page 186 lists the ISPF-provided output-only variables you can use on member list panels.

Table 30. Member list panel output-only variables

Variable	Description	Format/Length
ZDAZWD	Width of dynamic area (ZDATA)	CHAR 4
ZDAMLN	Length of dynamic area (ZDATA)	CHAR 4
ZDSN	First or only data set in concatenation	CHAR 44
ZMLCOLS	Member statistics column headings	CHAR 80
ZMLCR	Relative row number of top row	FIXED 4
ZMLTR	Total number of rows in member list	FIXED 4
ZDSVOL	Volume of the data set or first library in the concatenation.	CHAR 6

Customizing the z/OS UNIX directory list panel

Dialog developers can also provide a customized z/OS UNIX directory list display panel for their dialogs. The name of the customized directory list display panel is passed as a parameter to the DIRLIST service.

A customized directory list panel must be modeled after the DIRLIST default directory list panel, ISRUUDL0.

Note: Panel ISRUUDL0 is shipped as both source code (written in Dialog Tag Language, or DTL) and as generated output. If you use this panel as a model for creating your own customized panel, you have the option of modifying a copy of the DTL source code and using this to generate the customized panel. The DTL source code can be found in one of the SISPG*xxx* libraries, where *xxx* is the designator for a specified language.

The customized panel must include a Command field named ZCMD and a dynamic area named ZDATA which is used to display the column headings. ISPF uses the TBDISPL service to display the directory list and dynamically builds a model line in the value of variable ZULMODL. Therefore, the customized panel must specify variable ZULMODL as a variable model line. The panel is designed to use the available screen width and both the model line and column heading line are required to expand for the width of the screen. ISPF dynamically builds a list of ZVARs for the panel in the value of variable ZULZVARS.

For information on variable model lines refer to the description of the TBDISPL service in the *ISPF Services Guide*, and the section titled "Requirements for Model Section" in the *ISPF Dialog Developer's Guide and Reference*.

IBM Products option

Module ISRALTDI is used to determine if products are installed for the IBM Products Option (option 9). Products are considered to be installed if a specific panel for the product can be found in the panel library. If the panel is found, then ISPF will attempt to invoke it. No other checks are made to see if the products are correctly and completely installed or available.

For Information/System invocation, all parameters are allowed to default. To add parameters to the invocation, modify panel ISRDINFX by placing this statement immediately after the string PGM(BLGINIT):

```
PARM(your additional parameters)
```

To pass suboptions to the product panels, use the TRUNC and TRAIL functions with the ZALTTR variable before invoking program ISRALTDI. ZALTTR can hold up to 80 characters. Use this code:

```
&ZALTTR; = TRUNC(&ZCMD,'.')
&ZALTTR; = .TRAIL
&ZSEL; = TRANS(TRUNC*&ZCMD,'.')
```

The operation of ISRALTDI is:

1. Search the panel library for the primary panel.
2. If the primary panel exists, SELECT or DISPLAY the primary or alternate panel. If the primary panel does not exist, search the panel library for the secondary panel.
3. If the secondary panel exists, SELECT or DISPLAY the secondary or alternate panel. If the secondary panel does not exist, set an error message.

ISRALTDI is intended to be invoked from a selection panel. The parameter list follows:

```
SELECT PGM(ISRALTDI) PARM(
    Primary-panel
, < Primary-application-id   >
,   Secondary-panel
, < Secondary-application id >
, < Alternate-panel          >
, < Alternate-application-id >
) NOCHECK
```

After the last nonblank parameter, trailing commas are optional. The parameters are described here.

Primary-panel and secondary-panel are panel names and are required.

Primary- and secondary-application-id can be:

APPLICATION ID
: 0 to 4 character APPLID under which a panel is to be SELECTed.

*****
: DISPLAY this panel.

+
: Use alternate panel.

Alternate-panel is an optional panel name.

Alternate APPLID can be:

APPLICATION ID
: 0 to 4 character APPLID under which a panel is to be SELECTed.

*****
: DISPLAY this panel.

Some examples of the parameter string for ISRALTDI follow:

- If PANEL1 exists, SELECT it with application ID ‘AP’. Otherwise, DISPLAY panel ‘NOTHERE’.

  ```
  PANEL1,AP,NOTHERE,*
  ```

- If PANEL1 exists, DISPLAY it. Otherwise, DISPLAY panel ‘NOTHERE’.

  ```
  PANEL1,*,NOTHERE,*
  ```

- If PANEL1 exists, SELECT PANEL3 with application ID ‘CSP’. Otherwise, SELECT panel ‘UNAVAIL’ with application ID ‘ISR’.

  ```
  PANEL1,+,UNAVAIL,ISR,PANEL3,CSP
  ```

- If PANEL1 exists, SELECT PANEL3 with application ID ‘CSP’. Otherwise, SELECT panel ‘UNAVAIL’ with current APPLID.

```
PANEL1,+,UNAVAIL,,PANEL3,CSP
```

- If PANEL1 exists, SELECT PANEL1 with current application ID. Otherwise, SELECT panel ‘UNAVAIL’ with current APPLID.

```
PANEL1,,UNAVAIL
```

Part 2. Reference

Chapter 8. PDF installation-wide exits

PDF allows installations to satisfy unique processing needs by providing installation-wide exits at these points:

- Data set list and member list
- Data set allocation
- Print utility
- Compress request
- Data set name and member name change.

The sections that follow describe each of these installation-wide exits and list the parameters, return codes, and any error processing.

Note: All PDF exits should be AMODE(31). Consider making any exit routines that you write reusable, and preferably reentrant. If you write exits that are *not* reentrant, they cannot be put in the Link Pack Area (LPA) library. Non-reentrant exit routines placed in the LPA can cause abend errors.

Data set list filter exit

In the ISPF configuration table keyword file, this exit is set with keyword DATA_SET_LIST_FILTER_PROGRAM_EXIT.

The data set list filter exit is used by option 3.4 to allow the installation to determine what data set names appear in the data set list. The exit is called at two different points:

- First, with the DSNAME LEVEL and VOLUME specified on option 3.4 panel
- Second, for each data set whose name matches the DSNAME LEVEL and VOLUME.

If an exit routine is not provided, PDF uses the dsname level and volume specified on the panel and adds all data sets that match the data set name level and volume to the list.

PDF calls this exit routine using the standard conventions. The routine must be a program. All ISPF and system services are available to the routine.

See "Disable generic high-level qualifiers" on page 99 for an example of a data set list filter exit.

Analyze Dsname and Volume

Table 31. Details of the parameters used to analyze Dsname and Volume

Parameter	Type	Len	Description	Modifiable
CODE	FIXED	4	Code that indicates that this is the analyze dsname and volume call. Call=1.	No
DSNAME LEVEL	CHAR	44	Data set name level that was specified on the option 3.4 panel.	Yes
VOLUME	CHAR	6	Volume that was specified on the option 3.4 panel.	Yes

Analyze data set name

Table 32. Details of the parameters used to analyze the data set name

Parameter	Type	Len	Description	Modifiable
CODE	FIXED	4	Code that indicates that this is the analyze data set name call. Call=2	No
DSNAME LEVEL	CHAR	44	Data set name level that you specified on the option 3.4 panel.	No
VOLUME	CHAR	6	Volume that you specified on the option 3.4 panel.	No
DATA SET NAME	CHAR	44	Data set name that is to be added to the list.	No
VOLUME NAME	CHAR	6	Volume on which this data set resides.	No

Return codes

With the DSNAME LEVEL and VOLUME specified on option 3.4 panel.

0
: The data set name and volume is used as entered on the panel.

4
: Either the data set name or the volume has been modified by the exit and the modified data set name or volume is used.

8
: A list is not displayed.

For each data set whose name matches the DSNAME LEVEL and VOLUME.

0
: Data set name and volume is included in the list without any modification.

4
: Data set name is not added to the list.

8
: The list is stopped, no more items are added to the list.

Data set allocation exit

In the ISPF configuration table keyword file, this exit is set with keyword DATA_SET_ALLOCATION_PROGRAM_EXIT.

You can specify an installation-written exit routine to create, delete, allocate, and deallocate data sets instead of using those functions provided by PDF. However, allocations done by ISPF, the TSO ALLOCATE command, or TSO commands are not handled by the exit. If you use your own data set allocation exit routine, it must be a program. CLISTs are not allowed.

Exit parameters

PDF passes the SVC 99 parameter list/dynamic allocation request table as input to the exit routine. The information in the parameter list depends on the type of request being processed and is obtained from the user. Register 1 points to the parameter list.

PDF uses these parameters to communicate with the allocation exit:

SVC99 parmlist pointer
: A pointer to the SVC99 parmlist/dynamic allocation request table.

User storage pointer
A pointer to a 120-byte area that can be used to add new or changed text units.

Path name pointer
A pointer to an area of storage containing the absolute path name of the z/OS UNIX file to be allocated. Set to zero if this is not a request to allocation a z/OS UNIX file.

Path name length
A fullword binary integer that is the length of the z/OS UNIX file path name. Set to zero if this is not a request to allocation a z/OS UNIX file.

In addition to these parameters, PDF allocates enough space for a total of 30 text unit pointers in the text unit pointer list. The unused pointers can be used in conjunction with the 120-byte user data area for adding or changing text units.

If the request is to *allocate* a data set, the parameter list contains the information in Table 33 on page 193.

Table 33. Parameter list during allocation

Key	#Parms	Length	Parameter	Description	Exit may modify these parameters
0001	1	8	DDNAME	ddname to allocate data set to	No
0002	1	44	DSNAME	Data set name to allocate	No
0004	1	1	STATUS	Data set status	Yes
0005	1	1	NDISP	Normal data set disposition	Yes
0006	1	1	CDISP	Conditional data set disposition	Yes
0010	1	6	VLSER	Volume serial	No
0015	1	8	UNIT	unit name	No
0050	1	8	PASSW	Password for protected data set if a password was specified	No
0057	1	8	RTORG	Return data set organization	No
005D	1	6	RTVOL	Return volume serial from allocation	No
8017	1	1023 (max)	PATHNAME	Path name of the z/OS UNIX file to allocate. Pathname is of the form: `/dev/fd`*nnn* where *nnn* is the file descriptor number. The real path name can be obtained via the path name pointer and path name length parameters.	No
8018	1	4	PATHOPT	The file options for the z/OS UNIX file	No
801D	1	1	FILEORG	The organization of the z/OS UNIX file	No

If the request is to *concatenate* a data set, the parameter list contains the information in Table 34 on page 194.

Table 34. Parameter list during concatenation

Key	#Parms	Length	Parameter	Description	Exit may modify these parameters
0001	1	8	DDNAMEx	List of ddnames corresponding to the data sets being concatenated	No
0004	0	0	PERMCC	Permanently concatenated attribute	No

If the request is to *create* a data set, the parameter list contains the information in Table 35 on page 194.

Table 35. Create data set allocation parameter list

Key	#Parms	Length	Parameter	Description	Modifiable?
0001	1	8	DDNAME	ddname to allocate data set to	No
0002	1	44	DSNAME	Data set name to allocate	No
0004	1	1	STATUS	Data set status	Yes
0005	1	1	NDISP	Normal data set disposition	Yes
0006	1	1	CDISP	Conditional data set disposition	Yes
0007 0008 0009	1 1 1	0 0 3	TRKS CYLS BLKS	Space allocated in tracks or Space allocated in cylinders, or Space allocated in blocks	Yes Yes Yes
000A	1	3	PSPACE	Primary space quantity	Yes
000B	1	3	SSPACE	Secondary space quantity	Yes
000C	1	3	DBLKS	Number of directory blocks	Yes
0010	1	6	VOLSER	Volume serial	Yes
0015	1	8	UNIT	Unit group (esoteric) name Device type Specific unit address	Yes
0020	1	1	PASPR	Data set is password protected: X'10' Data set cannot be read, changed, extended, or deleted. X'30' Data set can be read, but not changed, extended, or deleted.	Yes
0022 006D 0023	1 1 1	5 7 2	EXPDT EXPDTL RETPD	Expiration date - YYDDD or Expiration date - YYYYDDD or Retention period	Yes Yes Yes
0030	1	2	BLKSZ	Block size	Yes
003C	1	2	DSORG	Data set organization	Yes
0042	1	2	LRECL	Logical record length	Yes
0049	1	1	RECFM	Record format	Yes
0050	1	8	PASSWORD	Data set password if specified	Yes
0052	0	0	PERM	Permanently allocated attribute	No

Table 35. Create data set allocation parameter list (continued)

Key	#Parms	Length	Parameter	Description	Modifiable?
0057	1	2	RTORG	Return data set organization	No
005D	1	6	RTVOL	Return volume serial from allocation	No
8004	1	8	STORCLAS	Storage class used to allocate the data set	Yes
8005	1	8	MGMTCLAS	Management class used to allocate the data set	Yes
8006	1	8	DATACLAS	Data class used to allocate the data set	Yes
8010	1	1	AVGREC	Unit of allocation in terms of average record size: X'80' Bytes X'40' Kilobytes X'20' Megabytes	Yes
8012	1	1	DSNTYPE	Data set name type used to allocate the data set: X'80' Library X'40' PDS X'08' Extreq X'04' Extpref X'02' Basic X'01' Large	Yes

If the request is to *deallocate* a data set, the parameter list contains the information in Table 36 on page 195.

Table 36. Parameter list during deallocation

Key	#Parms	Length	Parameter	Description	Exit may modify these parameters
0001	1	8	DDNAME	ddname to free	No
0007	0	0	UNALC	Unallocate option	No

If the request is to *delete* a data set, the parameter list contains the information in Table 37 on page 195.

Table 37. Parameter list during deletion

Key	#Parms	Length	Parameter	Description	Exit may modify these parameters
0002	1	44	DSNAME	DSNAME to free	No
0007	0	0	UNALL	Unallocate option	No

If the request is to *delete* a data set, it is done through the MVS SCRATCH macro. This allows unexpired data sets to be deleted.

Return codes

0
: No errors; the exit has issued the SVC 99.

4
: No errors; PDF issues the SVC 99.

8
: Error occurred; the exit has formatted a message.

20
: Severe error from the exit. The exit formats a message and PDF displays it in an error box.

If the allocation exit issues the SVC 99 and requests PDF to evaluate the error, the exit should return the SVC 99 return code to PDF in register 15 with the high-order bit of the register turned on.

If PDF issued the SVC 99, it places (by way of a VPUT) these variables into the shared pool in character format:

Z99RC
: Return code from SVC 99

Z99ERROR
: SVC 99 error code

Z99INFO
: SVC 99 information code.

Activity monitoring exits

In the ISPF configuration table keyword file, this exit is set with keyword ACTIVITY_MONITORING_PROGRAM_EXIT.

The activity monitoring exits provide monitoring information for these PDF functions:

- Primary commands invoked from BROWSE
- Primary commands invoked from EDIT
- Primary commands invoked from VIEW
- Edit macros invoked from EDIT
- Line commands invoked from EDIT
- Line commands invoked from VIEW
- Library/Data Set utility sub-functions
- Data Set List utility built-in line commands.

The activity monitoring exit routine is given control at the start and end of each command or function.

Exit parameters

PDF uses these parameters to communicate with the monitoring exit.

Table 38. Details of the parameters used to communicate with the monitoring exit

Parameter	Type	Len	Description	Modifiable
EXIT ID	FIXED	4	Numeric code for the exit point.	No
COMMAND/ SUB-FUNCTION ADDRESS	PTR	4	Storage location containing the command name or sub-function identifier.	No

Table 38. Details of the parameters used to communicate with the monitoring exit (continued)

Parameter	Type	Len	Description	Modifiable
COMMAND/ SUB-FUNCTION LENGTHS	FIXED	4	Length of the command name or sub-function identifier. The value ranges from 1-255.	No

Usage notes

- Activity monitoring exits are the same for VIEW as for EDIT. From within the VIEW function, the primary and line commands are monitored exactly like they are from EDIT.
- A single exit routine is allowed for all monitoring PDF exit points. The name of the module is placed into the ISPF configuration table when the product is installed.
- 31-bit addressing must be supported by the exit routine as it is given control in AMODE 31. ISPF/PDF does not restrict the RMODE of the exit, but RMODE ANY is recommended.
- Standard OS linkage conventions are used to branch to the exit routine that is defined.
- You can make PDF service calls from within the exit routine, with the exception of those services that could cause recursion (for example, calling the EDIT service for an EDIT primary command exit call).
- You cannot activate or deactivate the exit while an PDF session is in progress. You can make changes to the exit routine load module at any time, but because PDF loads the module only once at session initialization, the changes are not recognized until the next ISPF session.
- PDF does not provide data areas. You can use a predetermined ddname as a work area. The ISPF initialization exit could allocate and open a data set to this ddname. Each PDF exit could read and write to the data set as needed. The ISPF termination exit could close and free the data set.

Exit 1: BROWSE primary command start

This exit point marks the start of a valid BROWSE primary command. The exit routine is given control immediately before the function is performed.

Exit 2: BROWSE primary command end

This exit point marks the end of a valid BROWSE primary command. The exit routine is given control immediately after the command is performed. It can be used together with the BROWSE primary command start exit point for monitoring purposes.

Exit 3: EDIT primary command start

This exit point marks the start of a valid, user-entered EDIT primary command. The exit routine is given control immediately before the command is performed. All edit macro statements (such as ISREDIT SAVE) are processed as if they were primary commands. Individual edit macro statements executed from within the edit macro optionally invoke the exit routine. A field in the configuration table determines whether the activity monitoring exit is invoked at the start and stop of each edit macro command. If an alias has been defined for the command (via the DEFINE command of EDIT), and if the alias name is specified, the actual command that is invoked is passed to the exit.

Exit 4: EDIT primary command end

This exit point marks the end of a user-entered EDIT primary command. The exit routine is given control immediately after the command is performed. It can be used together with the EDIT primary command start exit point for monitoring purposes.

Exit 5: EDIT macro start

This exit point marks the start of a user-entered EDIT macro or an invalid EDIT primary command. The exit routine is given control immediately before the macro is invoked. Individual edit macro statements executed from within an edit macro optionally invoke the edit primary command exit routines (exits 3 and 4). Line commands processed by an edit macro optionally invoke the Edit line command exit routines (exits 7 and 8).

Exit 6: EDIT macro end

This exit point marks the end of a user-entered EDIT primary command. The exit routine is given command immediately after the macro is performed. It can be used together with the EDIT macro command start exit point for monitoring purposes.

Exit 7: EDIT line command start

This exit point marks the start of a valid user-entered EDIT line command. Line commands that are being processed by an edit macro optionally invoke the exit routine and indicate that it is a line command executed by an edit macro. The exit routine is given control immediately before the command is performed.

If multiple line commands are entered, the exit routine is given control once for each line command. A line command entered in block form (CC, for example) causes the exit to be called only once. Destination line commands (A,B,O) and defining line labels do not call the exit.

Exit 8: EDIT line command end

This exit point marks the end of an Edit line command. The exit routine is given control immediately after the command is performed.

If multiple line commands are entered, the exit routine is given control once for each line command. A line command entered in the block form (CC, for example) causes the exit to be called only once. Destination line commands (A, B, O) and defining line labels do not cause the exit to be called.

This exit point can be used together with the Edit line command start exit point for monitoring purposes.

Exit 9: Library/Data Set utility sub-function start

This exit point marks the start of a sub-function from either the Library (option 3.1) or Data Set (option 3.2) Utilities. These are the sub-function identifiers and their descriptions:

A
: Allocate data set

C
: Catalog data set

D
: Delete data set

I
: Data set information

M
: Display member list

MB
: Browse member

MD
: Delete member

ME
Edit member

MG
Reset member statistics

MJ
Submit member

MP
Print member

MR
Rename member

MT
TSO Command member

MV
View member

P
Print entire data set

R
Rename data set

RU
Rename data set using ISPF option 3.2 (catalog is not updated)

S
Data set information (short)

U
Uncatalog data set

VS
VSAM

X
Print data set index listing

Z
Compress data set

The exit routine is given control immediately before the sub-function is performed.

If multiple line commands (member list) are entered, the exit routine is given control once for each sub-function.

Exit 10: Library/Data Set utility sub-function end

This exit point marks the end of a Library or Data Set utility sub-function. The exit routine is given control immediately after the sub-function is performed. It can be used together with the Library/Data Set utility sub-function start exit point for monitoring purposes.

Exit 11: Data Set List utility line command start

This exit point marks the start of a line command from the Data Set List (option 3.4) utility. These are the subfunction identifiers and their descriptions:

Table 39. Subfuction identifiers and descriptions for exit 11

Identifier	Description
E	Edit data set
V	View data set

Table 39. Subfuction identifiers and descriptions for exit 11 (continued)

Identifier	**Description**
B	Browse data set
M	Display member list
D	Delete data set
R	Rename data set
I	Data set information (long)
S	Data set information (short)
P	Print entire data set
C	Catalog data set
U	Uncatalog data set
Z	Compress data set
F	Free unused space
H	Print Index
G	Reset
O	Move data set
Y	Copy data set
A	Reference add to personal list
X	Exclude data set
J	Unexclude 'NX'
K	Unexclude first 'NXF'
L	Unexclude last 'NXL'
N	SuperC
Q	SuperC Extended
T	Search-For
W	Search-For Extended
0	Allocate
MB	Browse member
MC	Copy member
MD	Delete member
ME	Edit member
MG	Reset member
MI	Info member
MJ	Submit member
MM	Move member
MN	Display Member Generation list

Table 39. Subfuction identifiers and descriptions for exit 11 (continued)

Identifier	Description
MO	Open member
MP	Print member
MR	Rename member
MT	TSO Command member
MV	View member

In addition to the one or two character command identifier, the name of the data set is provided for all commands that operate on entire data sets. For example, when an R is typed next to data set MY.TEST.DATA, the command pointed to by the command parameter is:

```
    R 'MY.TEST.DATA'
```

Note that the data set name provided is the one shown in the data set list, even if the name has been changed by a data set name change exit.

For the M (member list) command, the command parameter includes the data set name. However, for commands executed within the member list, just the command code is given (such as MB for the B (browse) line command).

The exit routine is given control immediately before the command is performed.

If multiple line commands are entered, the exit routine is given control once for each line command. Only built-in commands cause the exit to be called.

Exit 12: Data Set List utility line command end

The exit routine is given control immediately after the command is performed.

If multiple line commands are entered, the exit routine is given control once for each line command. Only built-in commands cause the exit to be called.

This exit point is provided to mark the end of a Data Set List utility line command. It can be used together with the Data Set List utility line command start exit point for monitoring purposes.

Error processing

If a nonzero return code is returned from an exit call, a message is conditionally set.

Return codes

0
: Normal Completion

Nonzero
: Severe Error.

The ISPF Select Exit can be used by installations to monitor TSO commands and CLISTs issued from the Data Set List utility.

Data set list line command exit

In the ISPF configuration table keyword file, this exit is set with keyword DATA_SET_LIST_LINE_COMMAND_PROGRAM_EXIT.

The data set list line command exit allows installations to change or restrict the line commands entered in the Data Set List utility.

Exit parameters

ISPF uses these parameters to communicate with the line command exit routine.

Table 40. Details of the parameters used to communicate with the line command exit routine

Parameter	Type	Len	Description	Modifiable
LINE COMMAND	CHAR	255	A 255-line character field containing the data set list line command. The field is left-justified with trailing blanks.	Yes

Usage notes

- The name of the module is placed into the ISPF configuration table when the product is installed.
- 31-bit addressing must be supported by the exit routine as it is given control in AMODE 31. PDF does not restrict the RMODE of the exit, but RMODE ANY is recommended.
- Standard OS linkage conventions are used to branch to the exit routine that is defined.
- You can make PDF service calls from within the exit routine.
- You cannot activate or deactivate the exit while a PDF session is in progress. You can make changes to the exit routine load module at any time, but because PDF loads the module only once at session initialization, the changes are not recognized until the next ISPF session.
- On input to the exit, the 255-character field contains the line command (left justified) after ISPF substitutes the data set name. If you explicitly specify a slash character (/) in place of the data set name, the slash is replaced by the data set name before the exit receives the command. If you do not specify a slash, the data set name is appended to the end of the command that you enter. One or more spaces will precede the data set name in this case.
- The exit can change the command buffer, subject to these rules:
  - If the final command to be executed is not one of the ISPF built-in commands (E, C, V, and so on), then the buffer can be changed to anything. ISPF does no validation before sending the command to TSO.
  - If the final command to be executed *is* one of the built-in commands, the data set name cannot be changed.

  Thus, user commands can be converted to built-in commands, built-in commands can be converted to user commands (like CLISTs), or the command string can be blanked out entirely to ignore the command.
- The Slash (/) command is considered a single command. Selections from the pop-up list are not reprocessed by the line command exit.
- Because the format of the data set list line command is free form, the exit should not have any dependencies on specific columns, or on specific numbers of blanks in the string.
- The intended interface for saving data across exit invocations is through standard ISPF methods (variables or tables).
- If the command buffer is changed, the exit must return a code of **4** in register 15 to notify ISPF.

If the exit name has been specified in the ISPF configuration table the exit is called before each line command is processed. The exit can analyze the line command and determine whether you should invoke the command as entered, as modified by the exit, or not at all.

If the exit modifies the line command, it is responsible for ensuring that the modified line command is valid.

Error processing

If a nonzero return code is returned from an exit call, a message is conditionally set.

Return codes

0
: Use line command as specified by the user

4
: Use line command as changed by the exit

8
: Do not invoke this command

20
: Severe error occurred.

Print utility exit

The Print utility exit receives control from ISPF in two distinct modes:

- Print utility exit for general printing (described here).
- Print utility exit on ISPF termination and LOG/LIST commands (see “Print utility exit on ISPF termination and LOG/LIST commands” on page 205).

In the ISPF configuration table keyword file, this exit is set with keywords PRINT_UTILITY_PROGRAM_EXIT and PRINT_UTILITY_COMMAND_EXIT.

The print utility exit lets you define your own print facility to replace or supplement PDF's print facilities. If you define a print routine for this exit, PDF calls that routine from options 3.6, 3.8, and all 4.x options. The print routine is also called on ISPF termination and for the LOG and LIST commands as described in “Print utility exit on ISPF termination and LOG/LIST commands” on page 205. The print routine can:

- Define its own print facility or use the TSO/E Information Center Facility printer definitions. To indicate that you are using the TSO/E Information Center Facility printer definitions, set the PRINT_USING_ICF keyword in the ISPF configuration table to YES.
- Specify additional print options.
- Supply print parameters automatically.
- Censor print requests.

Exit parameters

PDF communicates with the installation-written print routine through dialog variables in the ISPF shared pool. PDF passes data set, job card, and print option information to the print routine that you supply on the PDF print panels. As a system programmer, you can modify those print panels so that the print routines receive more or less information.

If variable ZPRLGLST is not defined or does not have a value of LOG or LIST, then the variables shown in Table 41 on page 204 are available to the print exit routine. However, if variable ZPRLGLST has a value of LOG or LIST, then the variables shown in Table 42 on page 205 are available to the print exit routine.

Table 41. Variables available to the print exit routine.

Variable	Type	Len	Description	Modifiable
ZPRDSN	CHAR	44	Fully qualified data set name (no quotes) of the data set to be printed	No
ZPRMEM	CHAR	8	Member to be printed, if the data set is a partitioned data set and a member is defined	No
ZPRPMD	CHAR	5	Print mode - BATCH, LOCAL, or TSO/E ICF	Yes
ZPROPT	CHAR	1	Disposition of the data set as specified by the user - K (keep) or D (delete)	Yes
ZPRDSORG	CHAR	8	Organization of the data set - PO or PS	No
ZPRRECFM	CHAR	6	Record format of the data set	No
ZPRLRECL	CHAR	5	Logical record length of the data set	No
ZPRBLKSZ	CHAR	5	Physical block size of the data set	No
ZPRVOLSE	CHAR	6	Volume serial of the data set	No
ZPRPASS	CHAR	8	Data set password (if it is password protected)	No
ZPRSYSO	CHAR	80	Batch SYSOUT class as specified by the user	Yes
ZPRSYSOL	CHAR	80	Local SYSOUT class as specified by the user	Yes
ZPRLPRT	CHAR	8	Local printer ID as specified by the user	Yes
ZPRICFPL	CHAR	15	TSO/E Information Center Facility printer location as specified by the user	Yes
ZPRICFPF	CHAR	8	TSO/E Information Center Facility printer format as specified by the user	Yes
ZPRICFPT	CHAR	8	TSO/E Information Center Facility printer type as specified by the user	Yes
ZPRICFNC	CHAR	3	TSO/E Information Center Facility number of copies as specified by the user	Yes
ZPRICFEF	CHAR	1	TSO/E Information Center Facility enable font selection	Yes
ZPRJB1	CHAR	72	Print job card 1 as specified by the user	Yes
ZPRJB2	CHAR	72	Print job card 2 as specified by the user	Yes
ZPRJB3	CHAR	72	Print job card 3 as specified by the user	Yes
ZPRJB4	CHAR	72	Print job card 4 as specified by the user	Yes

Return codes

0
: Successful termination of print routine. PDF does no further processing.

2
: Successful termination of print routine. PDF submits any generated JCL.

4
: Successful termination of print routine. PDF prints the data according to the information in the dialog variables located in the ISPF shared pool. If no message is waiting to be displayed, PDF issues a message.

20
Installation print routine failed. PDF keeps the data and, if the installation print exit has issued an ISPF SETMSG service request, ISPF displays the requested message.

Note: ISPF processes the data set based on the updated value of the ZPROPT variable for return codes 0, 2, or 4. This allows the exit to delete the data set or instruct ISPF to delete the data set. ZPROPT cannot be set to a value of 'D' for PDS or PDSE data sets.

Error processing

The installation-wide exit routine is responsible for handling all errors that occur while it is in control. The TSO/E Information Center Facility handles all error conditions while it is in control. In addition, PDF displays any error messages generated by the TSO/E Information Center Facility on the current print panel.

Print utility exit on ISPF termination and LOG/LIST commands

In the ISPF configuration table keyword file, this exit is set with keywords PRINT_UTILITY_PROGRAM_EXIT (if your exit is a program) and PRINT_UTILITY_COMMAND_EXIT (if your exit is a CLIST).

The print utility exit lets you define your own print facility to replace or supplement ISPF's print facilities. If you define a print routine for this exit, ISPF calls that routine upon ISPF termination and from ISPF's LOG and LIST commands. The print routine is also called from options 3.6, 3.8, and all 4.x options as described in "Print utility exit" on page 203. To define a print routine create your own JCL for batch jobs or your own local print routine such as PRINTDS. The exit parameters defined in Table 42 on page 205 that are marked with an asterisk (*) are system variables. They can be modified in the exit, but they are not saved in the system profile. The variables defined as not modifiable should not be modified.

Note: When calling the print utility exit upon ISPF termination and ISPF's LOG and LIST commands, the print log is turned off. The print utility operates without a log.

If the print utility exit is invoked, a new LOG or LIST is allocated if ISPF produces further LOG or LIST data.

The print routine can:

- Define its own print facility and submit the job.
- Specify additional print options.
- Supply print parameters automatically.
- Censor print requests.

Exit parameters

ISPF communicates with the installation-written print routine through dialog variables in the ISPF shared pool. PDF passes data set, job card, and print option information to the print routine that you supply on the ISPF LOG/LIST termination and command panels.

If variable ZPRLGLST has a value of LOG or LIST, then the variables shown in Table 42 on page 205 are available to the print exit routine. However, if variable ZPRLGLST is not available or blank, then the variables shown in Table 41 on page 204 are available to the print exit routine.

Table 42. Print exit routine variables

Variable	Type	Len	Description	Modifiable
ZPRLGLST	CHAR	4	Value is LOG if processing the LOG, LIST if processing the LIST.	No

Table 42. Print exit routine variables (continued)

Variable	Type	Len	Description	Modifiable
ZPRLGDSN	CHAR	44	Fully qualified log data set name (no quotes) of the data set to print.	No
ZPRLSDSN	CHAR	44	Fully qualified list data set name (no quotes) of the data set to print.	No
ZPRLGPMD	CHAR	5	Log print mode, BATCH or LOCAL.	Yes
ZPRLSPMD	CHAR	5	List print mode, BATCH or LOCAL.	Yes
*ZPRLGOPT	CHAR	1	Disposition of the log data set as specified by the user D (delete)	Yes
*ZPRLSOPT	CHAR	1	Disposition of the list data set as specified by the user D (delete)	Yes
*ZPRRECFM	CHAR	6	Record format of the list data set.	No
*ZPRLRECL	CHAR	5	Logical record length of the list data set.	No
*ZPRLGSYS	CHAR	15	Batch SYSOUT class for the log data set as specified by the user.	Yes
ZPRLGSYSL	CHAR	15	Local SYSOUT class for the log data set as specified by the user	Yes
*ZPRLSSYS	CHAR	15	Batch SYSOUT class for the list data set as specified by the user	Yes
ZPRLSSYL	CHAR	15	Local SYSOUT class for the list data set as specified by the user	Yes
*ZPRLGPRT	CHAR	17	Local printer ID for the log data set as specified by the user.	Yes
*ZPRLSPRT	CHAR	17	Local printer ID for the list data set as specified by the user.	Yes
*ZPRJB1	CHAR	72	Print job card 1 as specified by the user	Yes
*ZPRJB2	CHAR	72	Print job card 2 as specified by the user	Yes
*ZPRJB3	CHAR	72	Print job card 3 as specified by the user	Yes
*ZPRJB4	CHAR	72	Print job card 4 as specified by the user	Yes
ZPRMSGF	CHAR	1	Data set disposition message suppression. A value of 'S' stops ISPF setting a final disposition message.	Yes

Note: The user can only specify the disposition of the data set as D (delete). The exit can change the disposition of the data set to K (keep).

Return codes

When the print exit routine is invoked upon ISPF termination or ISPF LOG or LIST commands these return codes can occur.

0

Successful termination of print routine.

Batch

Exit provides job card and other JCL needed to print the job. Exit submits the job.

Local
Exit provides necessary information to print the job. Exit issues the print command.

ISPF issues a message on return from the print utility exit, indicating the return code from the exit.

2
Successful termination of the print routine.

Exit provides job card and JCL needed to print the job. ISPF submits the JCL contained in the file pointed to by the ZTEMPF variable.

ISPF issues a message on return from the print utility exit, indicating the return code from the exit.

4
Successful termination of the print routine.

Batch
Exit may alter the modifiable variables and place them in ISPF's shared pool. ISPF submits the print job using the information in the dialog variables located in the ISPF shared pool.

Local
Exit may alter the modifiable variables and place them in ISPF's shared pool. ISPF issues the PRINTDS command using the information in the dialog variables located in the ISPF shared pool.

ISPF issues a message on return from the print utility exit, indicating the return code from the exit.

Note:

ISPF processes the data set based on the updated value of the disposition variable (ZPRLGOPT or ZPRLSOPT) for return codes 0, 2, or 4. This allows the exit to either delete the data set itself, or instruct ISPF to delete the data set. ISPF does not delete pre-allocated LOG or LIST data sets.

For return codes 0, 2, and 4, ISPF issues a message regarding the disposition of the data set. The message is based on the dialog variables ZPRLGOPT, ZPRLSOPT, ZPRLGPRT, and ZPRLSPRT located in the ISPF shared pool. The exit can suppress this message by setting variable ZPRMSGF to a value of 'S'.

20
The print routine failed. ISPF keeps the data and displays an error message.

Error processing

The exit routine is responsible for handling all errors that occur while it is in control.

Compress request exit

In the ISPF configuration table keyword file, this exit is set with the keywords COMPRESS_UTILITY_PROGRAM_EXIT (if your exit is a program) and COMPRESS_UTILITY_CLIST_EXIT (if your exit is a CLIST).

Instead of using the IEBCOPY system utility interface that PDF provides, you can specify that PDF use an installation-written exit routine to handle requests to compress partitioned data sets under options 3.1, 3.4, and the LMCOMP service. When the compress exit gets control, the data set will have already been allocated and enqueued as exclusive. The exit will have to re-allocate the data set to a specific ddname. As part of termination processing, the exit must free the allocated ddname. The FREE operation must not be done by data set name. PDF allows the exit routine to:

- Use an alternate compression technique
- Provide backup before allowing normal PDF compression
- Selectively prevent compression.

PDF calls the exit routine with the SELECT service. The routine can be either a program or CLIST. All ISPF and system services are available.

Exit parameters

PDF communicates with the exit routine through variables in the shared pool. PDF considers these variables to be read-only and ignores any changes the exit routines make to the parameters. A VERASE is done at the end of the compress to delete the variables from all variable pools.

The variables available to the compress exit routine are:

Table 43. Details of the variables available to the compress exit routine

Variable	Type	Len	Description
ZCMPDSN	CHAR	44	Fully qualified data set name with no quotes
ZCMPVOL	CHAR	6	Volume serial
ZCMPPSWD	CHAR	8	Data set password
ZCMPORIG	CHAR	8	Origin of the call. **LMCOMP** The compress service (LMCOMP) called the exit routine. **OPTION31** The compress exit routine is being called from option 3.1, the compress utility. **OPTION34** The compress exit routine is being called from option 3.4, the data set list utility.

Return codes

0
Data set successfully compressed

2
PDF should compress the data set

4
Data set is not eligible for compression

8
Data set could not be allocated

12
Data set not cataloged

16
Error in exit; PDF should continue processing

17
LMCOMP recursion error

18
Unknown return code from exit

19
Error encountered invoking exit

20
Severe error; PDF should not continue processing

Other
PDF treats the error as return code 16.

Error processing

The exit routine is responsible for handling all errors that occur while it is in control. Failure to use correct allocation, serialization, and deallocation causes unpredictable results.

Data set name change exit

In the ISPF configuration table keyword file, this exit is set with keyword DATA_SET_NAME_CHANGE_PROGRAM_EXIT.

You can use the data set name change exit to change the name of the data set entered on the data entry panel. This allows PDF to process using the new data set name without the variables originally entered on the panel being changed. Anywhere the data set name is displayed, other than the data entry panel, can reflect either the original or the changed data set name. The data entry fields will not reflect any changes made to the data set name. If the exit is not specified, all data set names are used exactly as they are entered.

PDF calls this exit routine using the standard conventions. The exit is invoked for every data set specified on a data entry panel or in a PDF service call at the point in the PDF processing flow when the data set name is built from its component parts. This includes the library access services, EDIT and BROWSE services, and the PDF product dialogs themselves. The routine must be a program. All ISPF, PDF and system services are available to it.

If more than one ISPF library is specified on a data entry panel, the exit is invoked once for each library specified before each library is allocated. The libraries are not concatenated until each library is processed by the exit and allocated.

Exit parameters

PDF passes the data set name information as it was entered on the panel to the exit routine. Register 1 points to the parameter list.

PDF uses these parameters to communicate with the data set name change exit:

Table 44. Details of the parameters used to communicate with the data set name change exit

Variable	Type	Len	Description
PROJECT	CHAR	8	Project name as it was entered on the panel
GROUP	CHAR	8	Group name as it was entered on the panel
TYPE	CHAR	8	Type name as it was entered on the panel
MEMBER	CHAR	8	Member name as it was entered on the panel
OTHER DATA SET	CHAR	56	On input to the exit, the other data set name as entered on the panel. On output from the exit, blank or the data set name to be used. This can be used to replace an ISPF library.

Table 44. Details of the parameters used to communicate with the data set name change exit (continued)

Variable	Type	Len	Description
REASON	CHAR	8	Reason the data set is being allocated: **RECOVERY** The data set is the edit recovery data set. The Other Data Set field contains the fully qualified data set name. The data set name change exit is not called for edit recovery data sets which are listed in the edit recovery table with an associated disposition of 'K'. **TEMP** The data set is a temporary PDF data set. This includes: • The data set name specified on the member list SAVE command • The data set name specified on the data set list SAVE command • The data set used during the outlist utility processing (ISPF option 3.8, when using program ISRUOLP instead of CLIST ISRUOL). **blank** • Any data set entered on an PDF data entry panel • Data sets specified on the LMINIT service • Data sets specified on any service that does not need an LMINIT to have been previously performed.

Any of the parameters can be modified and are picked up by PDF with the exception of the Reason field, which is for the information of the exit only. Both an ISPF library (a PROJECT/GROUP/TYPE combination) and an OTHER DATA SET can be present in the parameter list. In this case, the OTHER DATA SET name supersedes the ISPF library name. If an ISPF library combination is specified as input to the exit and the exit wishes to create a new name that does not fit the ISPF library naming convention, the ISPF library name can be replaced by the exit filling in the Other Data Set field.

After the data set name has been changed by the exit, PDF does its normal data set processing, such as appending the user's prefix if an OTHER DATA SET name without quotes is returned by the exit. The changed data set name values are not saved in any variable pool, but are saved internally by PDF. The original input fields on the data entry panel are not changed, but any title line on a panel that contains a data set name can reflect either the original or the changed data set name.

Usage notes

1. Those data sets marked with a reason of TEMP can go through the exit more than once. The exit should not blindly add qualifiers to the TEMP data set names.
2. Any sequential data set that is used for output, such as being edited, the target of a Move/Copy (option 3.3), the source of an Edit MOVE command or the target of an Edit REPLACE command, or being reallocated by Edit after a space ABEND (B37), is sent through the exit twice. In addition, if a member is being browsed via the Library utility browse line command (browse under option 3.1), the data set is sent through the exit twice.
3. Data sets that are allocated in a CLIST as well as a program or that have their data set names built in a CLIST or skeleton cannot be modified by the data set name exit unless the CLIST or skeleton has also been modified to change the data set name in the same way that the exit does. These data sets include:

- The list data set name produced by the language processors in Foreground and Batch, and the term data sets in Batch.
- The input and output data sets for SuperC. The CLISTs to be modified are ISRSFORG and ISRSSRCH.
- The data set created by the Outlist utility (option 3.8) if CLIST ISRUOL is used rather than program ISRUOLP.

4. If the data set is being allocated for recovery, the data set name change exit should check the recovery table and generate a unique data set name.

Member list filter exit

In the ISPF configuration table keyword file, this exit is set with keyword MEMBER_LIST_FILTER_PROGRAM_EXIT.

The member list filter exit provides you with two capabilities:

1. You can dictate which members of a partitioned data set or concatenation of partitioned data sets are to be included in a member list when it is created.
2. You can specify which members of a member list are to be selected when the SELECT primary command is issued.

If a member list filter exit routine is not defined, PDF uses its default pattern matching conventions to determine which members are displayed, and which members are selected if the SELECT primary command is issued.

For each case in which the member list filter exit routine would be invoked, it is called first to allow verification or modification of the specified pattern ("Analyzing patterns with the member list filter exit" on page 211), and then once for each member in the data set that matches the pattern ("Analyzing member names with the member list filter exit" on page 212). The exit routine, through return codes, dictates which members are to be included in the member list, or selected if the SELECT primary command is issued.

PDF invokes the member list filter exit routine using standard linkage conventions. The exit routine must be a program.

From the exit routine, any of the PDF or ISPF services can be invoked. However, be careful when invoking services that generate a member list as part of their internal processing (LMMLIST, LMMDISP, LMMOVE, LMPROM, LMPRINT, and LMMSTATS). These invocations would result in a recursive call to the exit routine.

Analyzing patterns with the member list filter exit

Here are the exit parameters and return codes associated with analyzing patterns.

Exit parameters

PDF uses these parameters to communicate with the exit routine when it is invoked to verify/modify the pattern that you entered either on a member list entry panel, as a parameter to an PDF service, or with the SELECT primary command.

Table 45. Details of the parameters used to communicate with the member list filter exit when it is invoked to verify or modify a pattern

Parameter	Type	Len	Description	Modifiable
CODE	FIXED	4	Code that indicates that this is the analyze pattern call. Call=1.	No
PATTERN	CHAR	8	Pattern as entered by the user	Yes

Return codes

These return codes are expected from the member list filter exit routing to indicate the described conditions.

0
: The pattern that you entered generates the member list.

4
: The pattern is updated by the member list exit program and generates the member list.

8
: The member list request or SELECT command request is canceled and a conditional SETMSG is issued.

Analyzing member names with the member list filter exit

Here are the exit parameters and return codes associated with analyzing member lists.

Exit parameters

PDF uses these parameters to communicate with the exit routine when the routine determines which members that matched the pattern are to be included in the member list, or which members are to be selected if a SELECT command was issued.

Table 46. Details of the parameters used to communicate with the member list filter exit when it is invoked to match members

Parameter	Type	Len	Description	Modifiable
CODE	FIXED	4	Code indicating that this is the analyze member name call. Call=2.	No
MEMNAME	CHAR	8	Member name.	No

Return codes

These return codes are expected from the member list filter exit program to indicate the described conditions.

0
: Include this member in the member list.

4
: Do not include this member in the member list.

8
: The list is stopped; no more items are added to the list.

Member list built-in line command exit

In the ISPF configuration table keyword file, this exit
is set with keywords MEMBER_LIST_LINE_COMMAND_PROGRAM_EXIT and
MEMBER_LIST_LINE_COMMAND_COMMAND_EXIT.

The member list line command exit enables installations to change or restrict the behavior of the built-in line commands entered in a PDF member list. This exit is invoked for single-character member list built-in line commands invoked from ISPF option 3.1, ISPF option 3.4, and the ISPF Workplace (option 11).

This exit can be used to modify the behavior of built-in line commands. For example, when the E (edit) command is entered next to a member, the exit might examine the contents of the member and invoke a program other than the ISPF editor to process that member. If the exit then determines that the ISPF editor does not need to be invoked, it must return a return code of 8.

Exit parameters

PDF uses these parameters to communicate with the line command exit routine. These parameters are passed as a single 65-character string. A single blank is placed between parameters.

Table 47. Details of the parameters used to communicate with the line command exit routine

Parameter	Type	Len	Description	Modifiable
LINE COMMAND	CHAR	1	A 1-character field containing the data set list line command.	No
DATA SET NAME AND MEMBER	CHAR	56	A 56-character field containing data set name and member against which the line command is run. The format of this field is: 'DATA.SET.NAME(MEMBER)'. The field is left-justified with trailing blanks.	No
VOLUME SERIAL	CHAR	6	A 6-character field containing the volume on which the data set specified in DATA SET NAME AND MEMBER can be found.	No

Usage notes

- The name of the module is placed into the ISPF configuration table when the product is installed.
- 31-bit addressing must be supported by the exit routine as it is given control in AMODE 31. PDF does not restrict the RMODE of the exit, but RMODE ANY is recommended.
- For the program version of the exit, standard OS linkage conventions are used to branch to the exit routine that is defined.
- For the command version of the exit, the parameters are passed as operands.
- You can make PDF service calls from within the exit routine.
- You cannot activate or deactivate the exit while a PDF session is in progress. You can make changes to the exit routine load module at any time, but because PDF loads the module only once at session initialization, the changes are not recognized until the next ISPF session.
- If the exit name has been specified in the ISPF configuration table and the member list was generated from ISPF option 3.1, 3.4, or 11, the exit is called before each built-in line command is processed. The exit can analyze the line command and determine whether you should invoke the command as entered, or not at all.

Error processing

If a nonzero return code is returned from an exit call, a message is conditionally set.

Return codes

0
Use line command as specified by the user

8
Do not invoke this built-in command

20
Severe error occurred.

Chapter 9. ISPF Configuration Table keywords and values

ISPF Configuration Table keywords and values explains the keywords in the ISPF Configuration Table and their allowable and default values.

PDF exits

These fields specify the name of the program or command exit to be invoked at each exit point. COMMAND exits can be either CLIST or REXX. Exit names can have a maximum length of 8 characters. If both a program exit and a command exit are specified for the same exit point, the program exit is used.

DATA_SET_ALLOCATION_PROGRAM_EXIT
The program to be invoked as the data set allocation exit. This should be the name of a load module in your standard MVS search sequence.

PRINT_UTILITY_PROGRAM_EXIT
The program to be invoked as the print utility exit. This should be the name of a load module in your standard MVS search sequence.

PRINT_UTILITY_ COMMAND_ EXIT
The command to be invoked as the print utility exit. This should be the name of a member in your SYSPROC or SYSEXEC allocation.

COMPRESS_UTILITY_PROGRAM_EXIT
The program to be invoked as the compress utility exit. This should be the name of a load module in your standard MVS search sequence.

COMPRESS_UTILITY_CLIST_EXIT
The command to be invoked as the compress utility exit. This should be the name of a member in your SYSPROC or SYSEXEC allocation.

DATA_SET_LIST_FILTER_PROGRAM_EXIT
The program to be invoked as the data set list filter exit. This should be the name of a load module in your standard MVS search sequence.

MEMBER_LIST_FILTER_PROGRAM_EXIT
The program to be invoked as the member list filter exit. This should be the name of a load module in your standard MVS search sequence.

DATA_SET_NAME_CHANGE_PROGRAM_EXIT
The program to be invoked as the data set name change exit. This should be the name of a load module in your standard MVS search sequence.

DATA_SET_LIST_LINE_COMMAND_PROGRAM_EXIT
The program to be invoked as the data set list line command exit. This should be the name of a load module in your standard MVS search sequence.

ACTIVITY_MONITORING_PROGRAM_EXIT
The program to be invoked as the activity monitoring exit. This should be the name of a load module in your standard MVS search sequence.

MEMBER_LIST_LINE_COMMAND_PROGRAM_EXIT
The program to be invoked as the member list line command exit. This should be the name of a load module in your standard MVS search sequence.

MEMBER_LIST_LINE_COMMAND_COMMAND_EXIT
The command to be invoked as the member list line command exit. This should be the name of a member in your SYSPROC or SYSEXEC allocation.

Data set allocation settings

PDF_DEFAULT_UNIT
The unit name used by PDF when allocating work data sets. This value is used any time PDF needs to allocate a new data set on behalf of the user with the exception of option 3.2, edit recovery and ISPF work, control, and list data sets (ISPWRKx, ISPCTLx, and ISPLSTx). These new data sets may be temporary or permanent depending on the option of PDF being used.

For improved performance it is recommended that the VIO=YES option be added to the UNITNAME macro for the unit you specify in this field, but a VIO-only unit name is not recommended. Several of the ISPF options (including the Move/Copy utility and Outlist utility) will not function with VIO data sets.

The default is SYSALLDA.

ALLOWED_ALLOCATION_UNITS
This field controls which unit names are eligible to a user when the user is creating a data set through option 3.2. Valid values are:

ANY
Any unit may be used

UADS
Indicates the UNIT parameter in the users UADS entry should control the unit used

unit-name
Indicates that specific unit should be used

The default is ANY.

ALLOCATE_BEFORE_UNCATALOG
Indicates whether data sets to be uncataloged should first be allocated to accommodate those security packages the process during allocation. Valid values are YES or NO.

The default is NO.

VERIFY_EXPIRATION_DATE
Should expiration dates entered in option 3.2 when creating a data set be validated to ensure they are not in the past. Valid values are YES or NO.

The default is YES.

VOLUME_OF_MIGRATED_DATA_SETS
The volume name that indicates a data set is migrated.

The default is MIGRAT.

COMMAND_TO_DELETE_MIGRATED_DATA_SETS
The command that should be invoked when the D line command is used in option 3.4 to delete a migrated data set.

The default is HDELETE.

Outlist data set specifications

OUTLIST_RECORD_LENGTH
Record length of the temporary data set used by the Outlist utility.

The default is 133.

OUTLIST_BLOCK_SIZE
Block Size of the temporary data set used by the Outlist utility. Block size should be an even multiple of the record length unless a zero is specified for system determined block size.

The default is 13566.

OUTLIST_PRIMARY_QUANTITY
Primary number of tracks to be allocated for the Outlist Utility.

The default is 200.

OUTLIST_SECONDARY_QUANTITY
Secondary number of tracks to be allocated for the Outlist Utility.

The default is 100.

SuperC data set specifications

SUPERC_LIST_DATA_SET_BLOCK_SIZE
The block size for the SuperC listing data set. The list data set is a record format FBA, record length 133 data set. The block size should be an even multiple of 133 unless 0 is specified for system determined block size.

The default is 0.

This field is only used when the USE_SUPERC_PROGRAM_INTERFACE field is set to YES.

SUPERC_UPDATE_DATA_SET_BLOCK_SIZE
The block size for the SuperC update data set. The update data set is a record format FB, record length 80 data set. The block size should be an even multiple of 80 unless 0 is specified for system determined block size.

The default is 0.

This field is only used when the USE_SUPERC_PROGRAM_INTERFACE field is set to YES.

SUPERC_PROFILE_DATA_SET_BLOCK_SIZE
The block size for the SuperC profile data set. The profile data set is a record format FB, record length 80 data set. The block size should be an even multiple of 80 unless 0 is specified for system determined block size.

The default is 0.

This field is only used when the USE_SUPERC_PROGRAM_INTERFACE field is set to YES.

SUPERC_STATEMENTS_DATA_SET_BLOCK_SIZE
The block size for the SuperC statements data set. The statements data set is a record format FB, record length 80 data set. The block size should be an even multiple of 80 unless 0 is specified for system determined block size.

The default is 0.

This field is only used when the USE_SUPERC_PROGRAM_INTERFACE field is set to YES.

USE_SUPERC_PROGRAM_INTERFACE
Should SuperC be invoked directly from ISPF rather than invoked via clists ISRSFORG OR ISRSSRCH. Specifying YES in this field will improve the performance of the SuperC interface.

The default is YES.

SUPERC_LISTING_PRIMARY_QUANTITY
Primary number of blocks for the SuperC Listing data set.

The default is 50.

This field is only used when the USE_SUPERC_PROGRAM_INTERFACE field is set to YES.

SUPERC_LISTING_SECONDARY_QUANTITY
Secondary number of blocks for the SuperC Listing data set.

The default is 100.

This field is only used when the USE_SUPERC_PROGRAM_INTERFACE field is set to YES.

SUPERC_UPDATE_PRIMARY_QUANTITY
Primary number of blocks for the SuperC Update data set.

The default is 15.

This field is only used when the USE_SUPERC_PROGRAM_INTERFACE field is set to YES.

SUPERC_UPDATE_SECONDARY_QUANTITY
Secondary number of blocks for the SuperC Update data set.

The default is 30.

This field is only used when the USE_SUPERC_PROGRAM_INTERFACE field is set to YES.

LMF

FAIL_ON_LMF_LOCK
The default is YES.

Edit recovery data set specifications

EDIT_RECOVERY_BLOCK_SIZE
The block size for the edit recovery data set. This data set is a record format U, record length 0 data set. System determined block size is not supported for this field.

The default is 13680, the minimum allowed value is 3120.

EDIT_RECOVERY_PRIMARY_QUANTITY
The primary number of blocks that should be allocated for the edit recovery data set.

The default is 40.

EDIT_RECOVERY_SECONDARY_QUANTITY
The secondary number of blocks that should be allocated for the edit recovery data set.

The default is 200.

Move/Copy settings

MAXIMUM_GOOD_IEBCOPY_RETURN_CODE
Indicates the maximum return code from IEBCOPY that will allow Move/Copy processing to continue. Any return code higher than this value will be considered an error.

The default is 0.

USE_IEBCOPY_COPY_OR_COPYMOD_OPTION
Should ISPF use COPY or COPYMOD when invoking IEBCOPY to process load modules. Valid values are:

1
Use COPY if the target library block size is the same or greater than the source library block size, COPYMOD if the target block size is smaller.

2
Use COPY if the target library block size and source library block size are the same, COPYMOD if they are different.

3
Always use COPYMOD.

The default is 2.

WHEN_TO_USE_IEBCOPY
When should ISPF use IEBCOPY instead of a read/write loop to process load modules. Valid values are:

0

Only use IEBCOPY when processing a PDSE, or when copying from a larger block size to a smaller block size and COPYMOD was requested (see USE_IEBCOPY_COPY_OR_COPYMOD_OPTION).

1

Always use IEBCOPY for load modules.

2

Only use IEBCOPY for PDSEs.

The default is 0.

ALLOW_DATA_SET_CREATION_FOR_MOVE_COPY

If the target data set for Move/Copy does not exist, should the data set be created for the user. The user can specify either that the data set be created with the same characteristics as the original, or can specify the characteristics for the new data set. Valid values are YES and NO.

The default is YES.

DSLIST removable media interface settings

The keywords DSLIST_RM_ENABLED, DSLIST_RM_COMMAND, and DSLIST_RM_APPLID define an external interface to DFSMSrmm or an equivalent product. This can be used to allow the Data Set List utility (ISPF Option 3.4) to process selected line commands for data sets stored on tape or some other removable media.

The Data Set List utility handles return codes from the removable media interface command as follows:

0 or 4

Normal completion. Data Set List utility continues processing.

8

Error. Data Set List utility stops processing. Any further line commands are not processed. No error message is issued, allowing the tape/removable media interface to issue an error message via the ISPF SETMSG service.

12

Error. Data Set List utility issues an error message and stops processing. Any further line commands are not processed.

Other

Severe Error. Data Set List utility stops processing. Any further line commands are not processed. If the command was not found, message ISPD223 is issued, otherwise message ISRU671 is issued.

The command specified by DSLIST_RM_COMMAND should set the return code to 12 for any line commands that are not supported by the interface.

DSLIST_RM_ENABLED

Controls whether support is enabled in ISPF option 3.4 to call the removable media interface for these line commands: I, S, D, R, C, M, P, X, CO, MO. Valid values are YES and NO.

The default is NO.

DSLIST_RM_COMMAND

Specifies a command name, including parameters, to be invoked using the ISPF SELECT CMD service. The command name can be up to 45 characters in length, and may contain 2 special characters:

?

Is replaced with the option 3.4 line command. Two consecutive '?' characters will be replaced with a single '?' and no substitution of the line command will be performed. If '?' is not included within the command, the line command is automatically appended as a parameter to the end of the command.

/

Is replaced with the data set name (ZDLDSN) and the first volume (ZDLVOL), separated by a space. Two consecutive '/' characters will be replaced with a single '/' and no substitution of the data set name and first volume will be performed. Where the '/' is not included within the command, both the data set name and first volume will be automatically appended as a parameters to the end of the command. If a '%' character is specified before the command, the ISPF SELECT CMD service will bypass any attempt to load the command as a load module.

The default value for command is %EDGRPD34, which is equivalent to %EDGRPD34 ? /.

Here are some examples. The examples demonstrate different ways of specifying the DSLIST_RM_COMMAND parameter:

```
tapecmd ? /

tapecmd OPT(?) /

tapecmd OPT(?) DSV(/) ALTLIB(YES)
```

DSLIST_RM_APPLID
The syntax for the DSLIST_RM_APPLID keyword is:

```
DSLIST_RM_ APPLID = applid | NONE
```

Where *applid* is the application ID used to invoke the tape interface command defined by DSLIST_RM_COMMAND. *Applid* is a 1-character to 4-character name. If *applid* is set to NONE, no application ID will be used. Where the *applid* is specified, the PASSLIB parameter will also be included on the ISPF SELECT CMD service.

The default is EDG.

Edit-related settings

MAXIMUM_EDIT_PROFILES
The maximum number of ISPF Edit profiles. If the number of profiles exceeds this number, the least recently used unlocked profile is deleted from the profile table. The value must be between 1 and 255.

The default is 25.

SCLM_WARNING_LEVEL
Indicates the level of SCLM checking that is done when SCLM-controlled members are processed outside SCLM. The following values are valid:

NONE
No checking is done. SCLM-controlled members can be edited and processed outside SCLM.

WARN
If Edit or Reset Statistics processes an SCLM-controlled member, a message is displayed to warn the user that the SCLM accounting data is invalidated by the pending request.

ERROR
When Edit of an SCLM member is attempted, an error message is displayed and the edit is denied.

The default is WARN.

UNDO_STORAGE_SIZE
The maximum number of kilobytes of storage available to the edit UNDO command to be used for keeping a history of edit changes. A minimum value of 1024 (1024 KB) is recommended. If this value is 0, then UNDO is available from the edit recovery data set only. If the field is between 0 and 128, a value of 128 is used. This value will be rounded down to the nearest multiple of 64 (64 KB).

The default is 0.

Note: Use of storage for saving the record of changes provides better response time for individual users, but may have a slight detrimental effect on overall system performance.

ALLOW_EDIT_HIGHLIGHTING

Should ISPF Edit highlighting be available to all users and applications. Valid values are YES or NO. A value of NO disables Edit highlighting for all applications. A value of YES enables Edit highlighting for any dialog that uses a panel that is enabled for highlighting.

The default is YES.

DEFAULT_EDIT_DISPLAY

This field determines how the editor appears when it is started by using ISPF either interactively, or by using an Edit service call that does not specify a user edit panel. It controls the availability of action bars and edit highlighting. The following values are valid:

0

No actions bars are displayed, and Edit highlighting is not available.

1

Actions bars are displayed, but Edit highlighting is available.

2

No actions bars are displayed, but Edit highlighting is available.

3

Actions bars are displayed, and Edit highlighting is available.

The default is 3.

MAXIMUM_STORAGE_ALLOWED_FOR_EDIT

The maximum number of KB of storage that the editor can use when initially reading in data. If the initial read of the data requires more storage than this value, browse will be substituted instead. To allow edit to use as much storage as is available, set the value to 0.

The default is 0.

ENABLE_ASSEMBLER_CONTINUATION_ERRORS

Enable the use of reverse video pink to highlight assembly language continuations that start before column 16. Set this value to NO if your site uses a different start column for assembly language continuation (by using the ICTL assembly language instruction) or if assembly language highlighting is used for data other than assembly language programs (such as SCLM architecture definitions).

The default is YES.

WARN_ON_TRUNCATION_OF_TRAILING_BLANKS

Specifies that a warning message is displayed if a user edits variable data (record format = V) with one or more records that end in a blank. The editor truncates these blanks when the edit data is saved unless the editor is told to preserve the blanks.

The following conditions preserve blanks:

- The "Preserve VB record length" field on the **Edit Entry** panel is selected.
- The PRESERVE keyword is specified on the EDIT service invocation.
- The PRESERVE ON Edit command is entered.

The default is YES.

SITE_WIDE_INITIAL_MACRO

Site-wide Edit initial macro. The macro that is specified here is run before any user-specified macros. This option can be used to alter or disallow edit sessions. You can use a macro that does a PROFILE RESET to force all new profiles to use the settings in this configuration table.

The default is NONE (no macro).

TEXT_FLOW_TERMINATORS

What characters cause the edit text flow function (line command TF, edit macro command TFLOW) to stop processing. These characters generally indicate a new paragraph or section of a document.

The following characters are the default:

```
.:&<
```

EDIT_CUT_DEFAULT
Valid values are REPLACE and APPEND. Selecting REPLACE means that the cut information replaces whatever information is already in the clipboard. Selecting APPEND means that cut information is added at the end of existing information in the clipboard.

The default is REPLACE.

EDIT_PASTE_DEFAULT
Valid values are KEEP and DELETE. Selecting KEEP means that the information remains in the clipboard even after it is pasted it into a separate file. Selecting DELETE means that the information is deleted from the clipboard after it is pasted into a separate file.

The default is KEEP.

ALLOW_DATA_SET_CREATION_FOR_CREATE_REPLACE
If the target data set for the Edit CREATE or REPLACE command does not exist, is the data set created for the user. The user can specify either that the data set is created with the same characteristics as the original, or can specify the characteristics for the new data set. Valid values are YES and NO.

The default is YES.

FORCE_ISRE776_FOR_RCHANGE
Helps ensure that when RCHANGE is issued from a PF key, it does not try to process input from the command line. In this case RCHANGE treats anything that you type on the command line as an invalid parameter and returns an error message ISRE776.

This keyword sets a site default for the EDITSET option "Force ISRE776 if RCHANGE passed arguments". Valid values are YES and NO.

The default is NO.

FORCE_PRESERVE_VB_RECORD_LENGTH
Forces the users' Preserve VB Record Length setting in Edit to be the value selected in the PRESERVE_VB_RECORD_LENGTH field. Valid values are YES and NO.

The default is NO.

PRESERVE_VB_RECORD_LENGTH
The Preserve VB Record Length option in Edit is selected. This option causes the editor to save trailing blanks for variable-length files. Valid values are YES and NO.

The default is NO.

MAXIMUM_NUMBER_OF_EDIT_CLIPBOARDS
The maximum number of Edit clipboards allowed. Edit clipboards are used by the Edit CUT and PASTE commands and are kept in data spaces that last the life of the TSO session. Data spaces are allocated by the CUT command and released by the PASTE command, and are paged out when not in use. This value can be a number between 1 and 11.

The default is 11.

MAXIMUM_EDIT_CLIPBOARD_SIZE
The maximum size for the Edit clipboards, in 4 K increments. A value of 1 means 4 KB. A value of 0 indicates that the IBM default data space size (239 4-K blocks) of the value set through IEFUSI is used.

The default is 0.

VSAM_EDIT_ENABLED
Is editing of the VSAM data set enabled on the system. ISPF Edit starts the command in the VSAM_EDIT_COMMAND field for any VSAM data sets specified unless restricted (see VSAM_RESTRICTED_EDIT_DATASET). Valid values are YES and NO.

The default is NO.

VSAM_EDIT_COMMAND
The command to be started when a VSAM data set is specified to ISPF Edit. A slash (/) can be used to specify the data set name specified. The maximum length is 50 characters.

The default is **FMNINV DSE /**

VSAM_EDIT_LIMITED.
Are users be restricted from editing certain VSAM data sets? If this field is set to YES, the VSAM_RESTRICTED_EDIT_DATASET field must be used to specify the restricted data sets. Valid values are YES and NO.

The default is NO.

VSAM_BROWSE_ENABLED
Is browsing of VSAM data set enabled on the system. ISPF Browse starts the command in the VSAM_BROWSE_COMMAND field for any VSAM data sets specified unless restricted (see VSAM_RESTRICTED_BROWSE_DATASET). Valid values are YES and NO.

The default is NO.

VSAM_BROWSE_COMMAND
The command to be started when a VSAM data set is specified to ISPF Browse. A slash (/) can be used to specify the data set name specified. The maximum length is 50 characters.

The default is **FMNINV DSB /**

VSAM_VIEW_ENABLED
Is viewing of VSAM data set enabled on the system. ISPF View starts the command in the VSAM_VIEW_COMMAND field for any VSAM data sets specified unless restricted (see VSAM_RESTRICTED_VIEW_DATASET). Valid values are YES and NO.

The default is NO.

VSAM_VIEW_COMMAND
The command to be started when a VSAM data set is specified to ISPF View. A slash (/) may be used to specify the data set name specified. The maximum length is 50 characters.

The default is **FMNINV DSV /**

GLOBAL_LINE_COMMAND_TABLE
Defines the line command table that is active when a line command table is not otherwise specified by the user or supplied as a parameter on the EDIF, EDIT, VIEW, or VIIF service call.

GLOBAL_DISABLE_PACK
Disables the PACK operation that is used by the editor. Any currently packed data is unpacked if saved. This option also disables the PACK operation from affecting with COPY and MOVE services.

PREV_GEN_DEFAULT_SAVE_ACTION
This setting is applicable only when you edit a member in a PDSE version 2 data set that is configured for member generations. The setting determines the default action that is taken when a previous generation of a member is saved.

Valid values are:

NEWGEN
Saves the member in a new generation. This new generation becomes the current generation. The generation that is edited is left unchanged.

NOGEN
Saves the member to the same generation that is being edited.

The default is NOGEN.

PROTECT_MEMGENS_FROM_EDIT
This setting is applicable only when you edit a member in a PDSE version 2 data set that is configured for member generations. The setting determines whether noncurrent member generations are protected from editing. When this option is enabled, noncurrent generations are always opened in View mode. Valid values are YES and NO.

The default is NO.

Edit site-wide profile customizations

These fields set the defaults for new Edit Profiles created by any user. For information about the values for each field, refer to the *z/OS ISPF Edit and Edit Macros* . These processing rules apply:

- Some items in this section of the ISPF Configuration table can be *forced*. The default for all of the FORCE fields is NO. To force users to use the value specified, change the FORCE field value to YES. If a user then attempts to change one of the forced settings from within the editor an error message is displayed.
- If the user has a ZDEFAULT Edit profile or if a ZEDFAULT profile exists in the ISPTLIB concatenation, then these settings have no effect *except* for those options that are forced.
- If a user has no ZDEFAULT profile and no ZDEFAULT profile exists in ISPTLIB these settings are used to create all new profiles. They have no effect on existing profiles.
- The ZDEFAULT profile is no longer automatically created the first time a user enters the editor, as it was in releases of ISPF before OS/390 Release 5.0. Profiles used with previous versions might contain a ZDEFAULT profile, and it is honored. Users can also specifically create a ZEDFAULT profile so that they can establish their own defaults.

STATS
Control whether the editor maintains ISPF statistics for PDS members. Valid values are ON and OFF.

The default is ON.

FORCE_STATS
Force the specified value.

The default is NO.

STATS_EXT_ENABLED
Allows extended statistics to be generated when statistics are saved for a PDS member and any of the line count values for the member exceed 65535. Statistics can be saved under any of the following conditions:

- A member is saved from ISPF Edit
- A member is added or updated by using the LMMADD, LMMREP, or LMMSTATS service
- The reset member statistics panels are used

The default is NO.

RECOVERY
Determines whether the editor maintains its recovery file to keep track of the edit session, making it possible for users to recover from system failures. Valid values are ON and OFF.

The default is OFF.

FORCE_RECOVERY
Force the specified value.

The default is NO.

RECOVERY_WARNING_MESSAGE
Determines whether the editor should warn users when they enter an edit session with RECOVERY set OFF. Valid values are WARN and NOWARN.

The default is WARN.

FORCE_RECOVERY_WARNING_MESSAGE
Force the specified value.

The default is NO.

SETUNDO
Determines whether the UNDO command will be available in Edit. Valid values are ON and OFF.

The default is ON.

FORCE_SETUNDO
Force the specified value.

The default is NO.

PACK
Determines whether the editor will save data in packed or unpacked format. Valid values are ON and OFF.

The default is OFF.

FORCE_PACK
Force the specified value.

The default is NO.

IMACRO
Specifies the sitewide initial macro to be run for all users.

The default is NONE (no macro).

FORCE_IMACRO
Force the specified value.

The default is NO.

CAPS
Determines whether the editor automatically rolls text to uppercase. Valid values are ON and OFF.

The default is OFF.

NOTE
Determines whether the editor displays ==NOTE== lines when the Edit MODEL command is used. Valid values are ON and OFF.

The default is ON.

HEX
Determines whether the edit data is displayed in hex mode.

- ON
- OFF
- VERT
- DATA

The default is OFF.

NULLS
Determines whether trailing spaces on edit data are written to the screen as nulls or blanks.

- STD
- ALL
- OFF

The default is STD.

DISPLAY_SEQUENCE_NUMBERS
Determines whether the editor will maintain sequence numbers. Valid values are ON and OFF.

The default is ON.

COBOL_NUMBERS
Determines whether the sequence numbers should be maintained in COBOL format (in columns 1-6). Valid values are ON and OFF.

The default is OFF.

STANDARD_NUMBERS
Determines whether the sequence numbers should be maintained in standard format (in columns 1-8 for variable data, the last 8 columns for fixed data). Valid values are ON and OFF.

The default is ON.

AUTONUM
Determines whether the editor automatically renumbers edit data. Valid values are ON and OFF.

The default is OFF.

AUTOLIST
Determines whether the editor writes the edit data to the ISPF LIST data set when a user ends an edit session in which data has been changed or saved. Valid values are ON and OFF.

The default is OFF.

PROFILE
Determines whether an edit profile can be deleted if it is the least recently used profile. Specify LOCK to prevent profiles from being deleted. Valid values are LOCK and UNLOCK.

The default is UNLOCK.

AUTOSAVE
Determines whether the editor automatically saves changes when the END command is entered. Valid values are ON and OFF.

The default is ON.

AUTOSAVE_PROMPT
Determines whether the editor prompts the user to have the data changes saved of AUTOSAVE is OFF and the END command is entered. Valid values are PROMPT and NOPROMPT.

The default is PROMPT.

HILITE
Determines whether the editor uses color to highlight the data being edited. Valid values are ON and OFF.

The default is OFF.

HILITE_DOLOGIC
Determines whether editor highlighting should use color to match DO/END statements. Valid values are ON and OFF.

The default is OFF.

HILITE_IFLOGIC
Determines whether editor highlighting should use color to match IF/ELSE statements. Valid values are ON and OFF.

The default is OFF.

HILITE_PAREN
Determines whether editor highlighting should use color to match open and close parentheses. Valid values are ON and OFF.

The default is OFF.

HILITE_FIND
Determines whether editor highlighting should use color to highlight the target of the FIND command. Valid values are ON and OFF.

The default is ON.

HILITE_CURSOR
Determines whether editor highlighting should use color to highlight current cursor location. Valid values are ON and OFF.

The default is ON.

HILITE_LANGUAGE
The default language to be used by edit highlighting. Valid values are:

1
Automatic language determination

2
Assembler

3
PL/I

4
COBOL

5
Pascal

6
C

7
BookMaster

8
Rexx

9
ISPF Panel language

10
ISPF Skeleton language

11
JCL

12
ISPF Dialog Tag Language (DTL)

13
Other (CLIST, etc.)

14
Default (no highlighting)

15
PL/X

16
IDL

17
SuperC Listing

18
HTML

19
XML

The default is 1.

HILITE_MARGIN_C
Determines the left and right margins to be used by edit highlighting when processing C statements. Values are specified in the form (*left-margin*,*right-margin*). Although the IBM C/C++ compiler supports left and right margins in the range 1 to 32760, ISPF only supports highlighting data sets up to a record length of 255. Therefore, the *left-margin* must be between 1 and 254 and the *right-margin* must be between 1 and 255.

You can specify (*,*) to use the default margins defined for the language. The default *left-margin* is 1. The default *right-margin* is the lesser of 255 and the last input column. The last input column is 8 less than the record length when the data set has fixed records and is in standard number mode; otherwise it is the same as the record length.

HILITE_MARGIN_PLI
Determines the left and right margins to be used by edit highlighting when processing PL/I statements. Values are specified in the form (*left-margin*,*right-margin*). *left-margin* must be between 1 and 100. *right-margin* must be between 1 and 200.

You can specify (*,*) to use the default margins defined for the language. The default *left-margin* is 2. The default *right-margin* is the lesser of 200 and the last input column. The last input column is 8 less than the record length when the data set has fixed records and is in standard number mode; otherwise it is the same as the record length.

HILITE_MARGIN_PLX
Determines the left and right margins to be used by edit highlighting when processing PL/X statements. Values are specified in the form (*left-margin*,*right-margin*). *left-margin* must be between 1 and 65. *right-margin* must be between 15 and 80.

You can specify (*,*) to use the default margins defined for the language. The default *left-margin* is 2. The default *right-margin* is the lesser of 80 and the last input column. The last input column is 8 less than the record length when the data set has fixed records and is in standard number mode; otherwise it is the same as the record length.

ISPF site-wide profile customizations

The RESET fields described in this section have no effect unless the VERSION_LEVEL_OF_SITEWIDE_DEFAULTS field is modified.

A RESET field causes ISPF to modify the associated value for each user. The value is reset to the value specified in the configuration field that is associated with the RESET field. For example, the RESET_TAB_TO_ACTIONS_BARS field causes the Tab to action bar choices setting to be reset using the value specified in the TAB_TO_ACTION_BARS field. This resetting is done once each time the VERSION_LEVEL_OF_SITEWIDE_DEFAULTS field is incremented. Users can change the values of their fields after the incrementation has caused the reset.

VERSION_LEVEL_OF_SITEWIDE_DEFAULTS
This field indicates the modification level of the current RESET values. This field is set to 43000 initially by ISPF and will not be changed with new versions of ISPF. Each time you modify any of the RESET values, increment this value by 1. For example, the first time you change any RESET values you should set this field to 43001. This value is then saved in the ISPF system profile table.

When ISPF is initialized it checks the value saved in the system profile against the value in this field. If the value of this field is greater than that in the system profile, ISPF will use the values in the configuration fields that are associated with the RESET fields to update the user's values. You must increment this value every time you modify a value associated with a RESET field or it will not be picked up by ISPF.

TAB_TO_POINT_AND_SHOOT
Enable tabbing to point and shoot fields. Valid values are YES and NO.

The default is NO.

RESET_TAB_TO_POINT_AND_SHOOT
Reset the value specified here.

The default is NO.

TAB_TO_ACTION_BARS
Enable tabbing to action bars. Valid values are YES and NO.

The default is YES.

RESET_TAB_TO_ACTION_BARS
Reset the value specified here.

The default is NO.

USE_SESSION_MANAGER
Value is used to set system variable ZSESS and is used to initialize system variable ZSM. Valid values are YES and NO.

The default is NO.

RESET_USE_SESSION_MANAGER
Reset the value of system variable ZSM to the value of the USE_SESSION_MANAGER keyword.

JUMP_FROM_LEADER_DOTS
Enable the ISPF jump command (for example, =2) from fields with leader dots. Valid values are YES and NO.

The default is YES.

RESET_JUMP_FROM_LEADER_DOTS
Reset the value specified here.

The default is NO.

SHOW_SPLIT_LINE
Should ISPF show the split line when a user is running in split screen mode. Valid values are YES and NO.

The default is YES.

RESET_SHOW_SPLIT_LINE
Reset the value specified here.

The default is NO.

LONG_MESSAGES_IN_POPUP
Should ISPF long messages always be shown in pop-up windows, or only when they are longer than 78 characters. Valid values are YES and NO.

The default is YES.

RESET_LONG_MESSAGES_IN_POPUP
Reset the value specified here.

The default is NO.

EDIT_PRINTDS_COMMAND
Should the user be allowed to modify the PRINTDS command generated by ISPF before its submission. Valid values are YES and NO.

The default is NO.

RESET_EDIT_PRINTDS_COMMAND
Reset the value specified here.

The default is NO.

RESTORE_TEST_TRACE_OPTIONS
Should the original TEST and TRACE options specified on ISPF invocation be restored when a user exits from Dialog Test. Dialog test will set TEST mode on. Valid values are YES and NO.

The default is YES.

RESET_RESTORE_TEST_TRACE_OPTIONS
Reset the value specified here.

The default is NO.

DISPLAY_PANELS_IN_CUA_MODE
Should ISPF panels be displayed in CUA mode. Valid values are YES and NO.

The default is YES.

RESET_DISPLAY_PANELS_IN_CUA_MODE
Reset the value specified here.

The default is NO.

LOG_DATA_SET_DISPOSITION
The default disposition for the ISPF Log data set. Valid defaults are:

1
Process option not set

2
Print and delete

3
Delete without printing

4
Keep

5
Keep and allocate a new log

The default is 1, process option not set.

Note: When manually editing the configuration table the valid options are:

NONE
Process option not set

J
Print and delete

D
Delete without printing

K
Keep

R
Keep and allocate a new log data set

The default is NONE, process option not set.

RESET_LOG_DATA_SET_DISPOSITION
Reset the value specified here. The default is NO.

LIST_DATA_SET_DISPOSITION
The default disposition for the ISPF List data set. Valid defaults are:

1
Process option not set

2
Print and delete

3
Delete without printing

4
Keep

5
Keep and allocate a new list

The default is 1, process option not set.

Note: When manually editing the configuration table the valid options are:

NONE
Process option not set

J
Print and delete

D
Delete without printing

K
Keep

R
Keep and allocate a new list data set

The default is NONE, process option not set.

RESET_LIST_DATA_SET_DISPOSITION
Reset the value specified here.

The default is NO.

COMMAND_LINE_PLACEMENT
Placement of the ISPF command line. Valid values are:

BOTTOM
Float the command line to the bottom of the panel.

ASIS
Leave command line as coded on the panel.

The default is BOTTOM.

RESET_COMMAND_LINE_PLACEMENT
Reset the value specified here. The default is NO.

USE_KEYLISTS
Specifies whether ISPF uses Keylists for pfkey definitions. Valid values are YES and NO.

The default is YES.

RESET_USE_KEYLISTS
Reset the value specified here.

The default is NO.

SHOW_PFKEYS
Specifies whether ISPF will display the current PFKEY settings. Valid values are ON and OFF.

The default is ON.

RESET_SHOW_PFKEYS
Reset the value specified here.

The default is NO.

SCROLL_MEMBER_LIST
Specifies if ISPF should scroll to the first member selected in the member list after processing or disable the member list from automatic scrolling and instead place the cursor in front of the last member selected.

The default is YES.

RESET_SCROLL_MEMBER_LIST
Reset the value specified here.

The default is NO.

SCROLL_DEFAULT
Select the default scroll value. The valid settings are:

CSR
When scrolling forward, move the line that currently contains the cursor to the first data line. When scrolling backward, move the line that currently contains the cursor to the last data line.

DATA
Scroll by a full page minus one line when scrolling up or down, or by a full page minus one column when scrolling left or right.

HALF
Half a screenful of data.

MAX
Scroll to the top, bottom, left, or right margin.

PAGE
One screenful of data.

The default is PAGE.

SCROLL_MIN
Select the minimum scroll value allowed. This can be a 1 to 7 digit number.

The default is 0.

SCROLL_MAX
Select the maximum scroll value allowed. This can be a 1 to 7 digit number.

The default is 9999.

RESET_SCROLL_VALUE
Force an update of ISPSPROF from the configuration table values at ISPF initialization.

The default is NO.

DISPLAY_EMPTY_MEMBER_LIST
Controls whether an empty member list is displayed.

The default is NO.

DISPLAY_EMPTY_MEMBER_LIST_PATTERN
If the DISPLAY_EMPTY_MEMBER_LIST option is set, this field controls whether an empty list that results from a nonmatching pattern will be displayed.

The default is NO.

DISPLAY_EMPTY_MEMBER_LIST_FUNCTION
Whether empty member list options apply to non-edit functions such as View and Browse.

The default is YES.

RESET_EMPTY_MEMBER_LIST_OPTIONS
Reset the values specified in the DISPLAY_EMPTY_MEMBER_LIST fields.

The default is NO.

STATUS_AREA_DEFAULT
Select the default status area value. The valid settings are:

SES
Session

FUN
Function Keys

CAL
Calendar

USE
User status

UPS
User Point and Shoot

OFF
None

The default is SESSION.

LIST_DATA_SET_RECORDS_PER_BLOCK
The number of records per block for the ISPF List data set. This value must be in the range 0 to 32760. A value of 0 will result in an ISPF list data set being allocated using a system-determined block size.

The default is 26.

LOG_DATA_SET_BLOCK_SIZE
The block size of the ISPF Log data set. This value must be in the range 0 to 32760. A value of 0 will result in an ISPF log data set being allocated using a system-determined block size.

The default is 129.

LOG_DATA_SET_RECORD_LENGTH
The record length of the ISPF Log data set.

The default is 125.

BLOCK_SIZE_FOR_TEMPORARY_CNTL_DATA_SETS
The block size for ISPF temporary control (CNTL) data sets. This value must be in the range 0 to 32760. A value of 0 will result in the data sets being allocated using a system-determined block size.

The default is 800.

RECORD_LENGTH_FOR_TEMPORARY_CNTL_DATA_SETS
The record length for ISPF temporary control (CNTL) data sets.

The default is 80.

BLOCK_SIZE_FOR_TEMPORARY_LIST_DATA_SETS
The block size for ISPF temporary list data sets. This value must be in the range 0 to 32760. A value of 0 will result in the data sets being allocated using a system-determined block size.

The default is 3146.

RECORD_LENGTH_FOR_TEMPORARY_LIST_DATA_SETS
The record length for ISPF temporary list data sets.

The default is 121.

BLOCK_SIZE_FOR_TEMPORARY_WORK_DATA_SETS
The block size for ISPF temporary work data sets. This value must be in the range 0 to 32760. A value of 0 will result in the data sets being allocated using a system-determined block size.

The default is 2560.

RECORD_LENGTH_FOR_TEMPORARY_WORK_DATA_SETS
The record length for ISPF temporary work data sets.

The default is 256.

ISPCTL0_BLOCK_SIZE
The block size for ISPCTL0 temporary control (CNTL) data set. This value must in the range 0 to 32760.

The default is 800.

ISPCTL0_RECORD_LENGTH
The record length for the ISPCTL0 temporary control (CNTL) data set.

The default is 80.

ISPCTL0_PRIMARY_QUANTITY
Primary number of blocks to be allocated for the ISPCTL0 temporary control (CNTL) data set.

The default is 10.

ISPCTL0_SECONDARY_QUANTITY
Secondary number of blocks to be allocated for the ISPCTL0 temporary control (CNTL) data set.

The default is 100.

ISPCTL_PRIMARY_QUANTITY
Primary number of blocks to be allocated for file tailoring CNTL data sets.

The default is 10.

ISPCTL_SECONDARY_QUANTITY
Secondary number of blocks to be allocated for file tailoring CNTL data sets.

The default is 100.

ISPLST_PRIMARY_QUANTITY
Primary number of blocks to be allocated for ISPF temporary list data sets.

The default is 10.

ISPLST_SECONDARY_QUANTITY
Secondary number of blocks to be allocated for ISPF temporary list data sets.

The default is 100.

ISPWRK_PRIMARY_QUANTITY
Primary number of blocks to be allocated for the file tailoring WORK data sets.

The default is 10.

ISPWRK_SECONDARY_QUANTITY
Secondary number of blocks to be allocated for the File Tailoring WORK data sets.

The default is 100.

USE_PDFCUNIT_FOR_TEMP_ISPF_DATA_SETS
Indicates whether to use the PDFCUNIT value for the units field when allocating ISPF temporary data sets. Valid values are YES and NO.

The default is NO.

ISPF_TEMPORARY_DATA_SET_QUALIFIER
An additional qualifier that will be appended to the ISPF log, list, and temporary control data set names. The qualifier will come after the ISPF assigned prefix, but before the suffix area. If Exit 16 is active, this qualifier will be part of the 26-byte prefix area passed to the exit. If the configuration table field USE_ADDITIONAL_QUAL_FOR_PDF_DATA_SETS is set to YES, the qualifier is also appended to the default names of data sets created by PDF utilities.

The qualifier can be either:

1. A valid data set qualifier, comprising 1 to 8 alphanumeric characters, the first being alphabetic (not numeric)
2. A string containing 1 or more system symbolic variables.

   The string may be up to 32 characters in length, but when resolved it will be truncated to 8 characters. Truncation errors are ignored.

   Other characters may be included between the symbolic variables, providing they are alphanumeric characters and the first character is nonnumeric. The use of any of the date and time symbols requires an alphabetic character before the symbol name to ensure that the qualifier is valid. If the resulting qualifier is invalid, it is ignored without an error message being issued.

Examples:

```
&SYSNAME.
SYS&SYSNAME(1:4).
```

See the *z/OS MVS Initialization and Tuning Reference* for details on valid system symbols.

The default is `ISP&SEQ.` where ISPF Profile Sharing is enabled, otherwise the default is NONE (no qualifier).

PROFILE_SHARING
Enable the ISPF Profile Sharing facility. Valid values are YES and NO. The default is NO.

PROFILE_ENQLOCK_WAIT
The time in milliseconds that ISPF is to wait when it is unable to obtain an enqueue for an ISPF profile member. The value is an integer from 0 to 9999. The default is 1000 milliseconds (1 second). A value of 0 results in no wait being issued.

PROFILE_ENQLOCK_RETRY_COUNT
The number of attempts to retry the enqueue request when ISPF is unable to obtain the enqueue for an ISPF profile member. The value is an integer from 0 to 99. The default value is 1. A value of 0 results in no further attempt to obtain a failed enqueue.

PROFILE_ENQLOCK_PROMPT
Enable the prompt for further action when ISPF is unable to obtain the enqueue for an ISPF profile member. Valid values are YES and NO. The default is YES. When ISPF is running in a background environment or a value of NO is specified and ISPF is unable to obtain the enqueue for an ISPF profile member, the enqueue request is failed. When a value of YES is specified and ISPF is unable to obtain the enqueue for an ISPF profile member, a pop-up window is displayed giving you the option to either retry or fail the enqueue request.

PROFILE_SYSPROF_CONFLICT
The initial action to be taken when the ISPF system profile member, ISPSPROF, has been updated after it was read from disk. Valid values are:

PROMPT
Prompts you to either KEEP or DISCARD the changes to the ISPF system profile from the current ISPF session. You can also disable the prompt so that the same action is taken for further changes to the ISPF system profile during the current ISPF session.

KEEP
Updates the System profile with changes from the current session. Other updates to the ISPF system profile made on other systems may be lost.

DISCARD
Updates made to the ISPF system profile during the current session are discarded.

The default is KEEP.

PROFILE_ISPPROF_CONFLICT
The initial action to be taken when the ISPF profile member, normally ISPPROF, has been updated after it was read from disk. Valid values are:

PROMPT
Prompts you to either KEEP or DISCARD the changes to the ISPF system profile from the current ISPF session. You can also disable the prompt so that the same action is taken for further changes to the ISPF system profile during the current ISPF session.

KEEP
Updates the profile with changes from the current session. Other updates to the ISPF profile made on other systems may be lost.

DISCARD
Updates made to the ISPF profile during the current session are discarded.

The default is KEEP.

PROFILE_APPPROF_CONFLICT

The initial action to be taken when an ISPF application profile member has been updated after it was read from disk. Valid values are:

PROMPT

Prompts you to either KEEP or DISCARD the changes to the ISPF system profile from the current ISPF session. You can also disable the prompt so that the same action is taken for further changes to the ISPF system profile during the current ISPF session.

KEEP

Updates the Application profile with changes from the current session. Other updates to the ISPF application profile made on other systems may be lost.

DISCARD

Updates made to the ISPF application profile during the current session are discarded.

The default is KEEP.

PROFILE_REFLIST_CONFLICT

The initial action to be taken when an ISPF reference list member, ISRLLIST, ISRPLIST, or ISRSLIST, has been updated after it was read from disk. Valid values are:

PROMPT

Prompts you to either KEEP or DISCARD the changes to the ISPF system profile from the current ISPF session. You can also disable the prompt so that the same action is taken for further changes to the ISPF system profile during the current ISPF session.

KEEP

Updates the reference list with changes from the current session. Other updates to the ISPF reference list made on other systems may be lost.

DISCARD

Updates made to the ISPF reference list during the current session are discarded.

The default is KEEP.

PROFILE_EDIT_CONFLICT

The initial action to be taken when an ISPF edit profile, *xxxx*EDRT, has been updated after it was read from disk. Valid values are:

PROMPT

Prompts you to either KEEP or DISCARD the changes to the ISPF system profile from the current ISPF session. You can also disable the prompt so that the same action is taken for further changes to the ISPF system profile during the current ISPF session.

KEEP

Updates the edit profile with changes from the current session. Other updates to the ISPF edit profile made on other systems may be lost.

DISCARD

Updates made to the ISPF edit profile during the current session are discarded.

The default is KEEP.

PROFILE_BATCH_CONFLICT

The action to be taken when ISPF is executing in a background environment and any profile has been updated after it was read from disk. Valid values are:

KEEP

Updates the profile member with changes from the current session. Other updates to the ISPF profile member made on other systems may be lost.

DISCARD

Updates made to the ISPF profile member during the current session are discarded. The default is DISCARD.

PROFILE_OTHER_CONFLICT

The initial action to be taken when any other ISPF table in the ISPF ISPF profile data set has been updated after it was read from disk. Valid values are:

PROMPT

Prompts you to either KEEP or DISCARD the changes to the ISPF system profile from the current ISPF session. You can also disable the prompt so that the same action is taken for further changes to the ISPF system profile during the current ISPF session.

KEEP

Updates the ISPF table with changes from the current session. Other updates to the ISPF table made on other systems may be lost.

DISCARD

Updates made to the ISPF table during the current session are discarded.

The default is KEEP.

Any conflicts that occur during ISPF termination where PROMPT is specified and the ISPF display services are no longer available are converted to the default action.

RESET_PROFILE_SHARING_SETTINGS

Option to specify that all the Profile Sharing settings are to be reset to the ISPF Configuration defined values. Valid values are YES and NO. The default is NO.

PDSE_RESERVE_PROCESSING

Option to specify whether ISPF issues a RESERVE to serialize access to PDSE. Valid values are OFF and ON. The default value is ON. See Appendix A, "ISPF enqueue processing for data integrity," on page 315 for further information.

ISPF_INIT_COMMAND_TABLE_ENTRY

Identifies a command table entry to be invoked when the value of VERSION_LEVEL_OF_SITEWIDE_DEFAULTS has been changed. One possible use for this could be to reset the value of the ISPF ZSTART profile variable. Note that neither ZSTART nor any other start option is processed when this command table entry is invoked. Therefore the invoked function should process ZSTART or other option. ZSTART, or the option, is appended to any other parameters that may be passed via this keyword. The command table entry name plus all arguments is truncated if the length exceeds 255 bytes.

PRINTDS_DEST_OR_WRITER_OPTION

PRINTDS option, valid values are:

DEST

Indicates a local printer ID is being used.

WRITER

Indicates an external writer name is being used.

The default is DEST.

LOCAL_PRINTDS_OPTIONS

Parameters appended to the PRINTDS command when a local print is done. No verification of these fields is done. The maximum length of this value is 128 characters. To disable printing through PRINTDS, specify a value of DISABLE.

The default is NONUM.

USE_ALTERNATE_DIALOG_TEST_PANEL

Select alternate dialog test panels ISPYFP, ISPYFPA, or ISPYFPB. Panel ISPYFPA is formatted with the most frequently used fields at the top of the panel. Panel ISPYFPB is similar to panel ISPYFPA, but it has a selection field that allows the user to select a function: panel, command, program, or request. Unlike the panels ISPYFP or ISPYFPA, on panel ISPYFPB the panel, command, program, or request fields can all contain values. Valid options are:

1

ISPYFP panel

2
: ISPYFPA panel

3
: ISPYFPB panel

The default is 1.

USE_ADDITIONAL_QUAL_FOR_PDF_DATA_SETS

An additional qualifier is included in the default data set name for data sets generated by PDF utilities. The data sets affected by this setting are:

- Listing and update data sets produced by the SuperC and Srchfor utilities.
- Listing data sets produced by the data set list and member list SRCHFOR commands.
- Trace data sets generated by the ISPVCALL, ISPDPTRC, and ISPFTTRC commands.
- Data set created by the table utility EXPORT and FEXPORT commands.

The value specified for the ISPF_TEMPORARY_DATA_SET_QUALIFIER field is used as the additional qualifier which is included at the start of the default suffix for these data set names.

Valid values are YES and NO. The default is NO.

Default CUA color settings

These fields in the ISPF Configuration table define the color and highlighting for each of the CUA panel elements. The setting for each element consists of a 3-character numeric field, with each position meaning:

Position 1 — COLOR

1
: Blue

2
: Red

3
: Pink

4
: Green

5
: Turquoise

6
: Yellow

7
: White

Position 2 — INTENSITY

0
: Low

2
: High

Position 3 — HILITE

0
: None

1
: Blink

2
: Reverse Video

4
Underscore

The RESET fields described in this section have no effect unless the VERSION_LEVEL_OF_SITEWIDE_DEFAULTS field is modified. See "ISPF site-wide profile customizations" on page 228 for more details. If you select any RESET fields here, you *must* increment the value in the VERSION_LEVEL_OF_SITEWIDE_DEFAULTS field.

A RESET field causes ISPF to modify the associated value for each user. The value is reset to the value specified in the configuration field that is associated with the RESET field. For example, the RESET_COLUMN_HEADING field causes the color, intensity, and highlighting used for column headers to be reset using the value specified in the COLUMN_HEADING field. This resetting is done once each time the VERSION_LEVEL_OF_SITEWIDE_DEFAULTS field is incremented. Users can change the values of their fields after the incrementation has caused the reset.

ACTION_BAR_SELECTED_CHOICE
Default is 600 (yellow, low intensity, no highlighting).

RESET_ACTION_BAR_SELECTED_CHOICE
Reset the value specified here.

The default is NO.

ACTION_BAR_SEPARATOR_LINE
Default is 100 (Blue, low intensity, no highlighting).

RESET_ACTION_BAR_SEPARATOR_LINE
Reset the value specified here.

The default is NO.

ACTION_BAR_UNSELECTED_CHOICE
Default is 720 (White, high intensity, no highlighting).

RESET_ACTION_BAR_UNSELECTED_CHOICE
Reset the value specified here.

The default is NO.

ACTION_MESSAGE_TEXT
Default is 220 (Red, high intensity, no highlighting).

RESET_ACTION_MESSAGE_TEXT
Reset the value specified here.

The default is NO.

CAUTION_TEXT
Default is 620 (Yellow, high intensity, no highlighting).

RESET_CAUTION_TEXT
Reset the value specified here.

The default is NO.

CHOICE_ENTRY_FIELD
Default is 504 (Turquoise, low intensity, underscored)

RESET_CHOICE_ENTRY_FIELD
Reset the value specified here.

The default is NO.

COLUMN_HEADING
Default is 120 (Blue, high intensity, no highlighting).

RESET_COLUMN_HEADING
Reset the value specified here.

The default is NO.

DESCRIPTIVE_TEXT
Default is 400 (Green, low intensity, no highlighting).

RESET_DESCRIPTIVE_TEXT
Reset the value specified here.

The default is NO.

EMPHASIZED_TEXT
Default is 520 (Turquoise, high intensity, no highlighting).

RESET_EMPHASIZED_TEXT
Reset the value specified here.

The default is NO.

ERROR_EMPHASIS
Default is 622 (Yellow, high intensity, reverse video)

RESET_ERROR_EMPHASIS
Reset the value specified here.

The default is NO.

FIELD_PROMPT
Default is 400 (Green, low intensity, no highlighting).

RESET_FIELD_PROMPT
Reset the value specified here.

The default is NO.

FUNCTION_KEYS
Default is 100 (Blue, low intensity, no highlighting).

RESET_FUNCTION_KEYS
Reset the value specified here.

The default is NO.

INFORMATIONAL_MESSAGE_TEXT
Default is 720 (White, high intensity, no highlighting).

RESET_INFORMATIONAL_MESSAGE_TEXT
Reset the value specified here.

The default is NO.

LIST_ENTRY_FIELD
Default is 504 (Turquoise, low intensity, underscored)

RESET_LIST_ENTRY_FIELD
Reset the value specified here.

The default is NO.

LIST_ITEM_DESCRIPTION
Default is 400 (Green, low intensity, no highlighting).

RESET_LIST_ITEM_DESCRIPTION
Reset the value specified here.

The default is NO.

LIST_ITEM
Default is 700 (White, low intensity, no highlighting).

RESET_LIST_ITEM
Reset the value specified here.

The default is NO.

NORMAL_ENTRY_FIELD
Default is 504 (Turquoise, low intensity, underscored)

RESET_NORMAL_ENTRY_FIELD
Reset the value specified here.

The default is NO.

NORMAL_TEXT
Default is 400 (Green, low intensity, no highlighting).

RESET_NORMAL_TEXT
Reset the value specified here.

The default is NO.

PANEL_ID
Default is 100 (Blue, low intensity, no highlighting).

RESET_PANEL_ID
Reset the value specified here.

The default is NO.

PANEL_INFORMATION
Default is 400 (Green, low intensity, no highlighting).

RESET_PANEL_INFORMATION
Reset the value specified here.

The default is NO.

PANEL_TITLE
Default is 100 (Blue, low intensity, no highlighting).

RESET_PANEL_TITLE
Reset the value specified here.

The default is NO.

POINT_AND_SHOOT
Default is 520 (Turquoise, high intensity, no highlighting).

RESET_POINT_AND_SHOOT
Reset the value specified here.

The default is NO.

PULLDOWN_AVAILABLE_CHOICE
Default is 700 (White, low intensity, no highlighting).

RESET_PULLDOWN_AVAILABLE_CHOICE
Reset the value specified here.

The default is NO.

PULLDOWN_UNAVAILABLE_CHOICE
Default is 100 (Blue, low intensity, no highlighting).

RESET_PULLDOWN_UNAVAILABLE_CHOICE
Reset the value specified here.

The default is NO.

REFERENCE_PHRASE
Default is 720 (White, high intensity, no highlighting).

RESET_REFERENCE_PHRASE
Reset the value specified here.

The default is NO.

SCROLL_INFORMATION
Default is 720 (White, high intensity, no highlighting).

RESET_SCROLL_INFORMATION
Reset the value specified here.

The default is NO.

SELECTION_AVAILABLE_CHOICE
Default is 700 (White, low intensity, no highlighting).

RESET_SELECTION_AVAILABLE_CHOICE
Reset the value specified here.

The default is NO.

SELECTION_UNAVAILABLE_CHOICE
Default is 100 (Blue, low intensity, no highlighting).

RESET_SELECTION_UNAVAILABLE_CHOICE
Reset the value specified here.

The default is NO.

VARIABLE_OUTPUT_INFORMATION
Default is 500 (Turquoise, low intensity, no highlighting).

RESET_VARIABLE_OUTPUT_INFORMATION
Reset the value specified here.

The default is NO.

WARNING_MESSAGE_TEST
Default is 620 (Yellow, high intensity, no highlighting).

RESET_WARNING_MESSAGE_TEST
Reset the value specified here.

The default is NO.

WARNING_MESSAGE
Default is 220 (Red, high intensity, no highlighting).

RESET_WARNING_MESSAGE
Reset the value specified here.

The default is NO.

WORKAREA_SEPARATOR_LINE
Default is 100 (Blue, low intensity, no highlighting).

RESET_WORKAREA_SEPARATOR_LINE
Reset the value specified here.

The default is NO.

Miscellaneous settings

MONITOR_EDIT_MACRO_COMMANDS
Should the ISPF Activity Monitoring Exit be invoked for ISREDIT commands invoked from an ISPF Edit Macro. Valid values are YES and NO.

The default is NO.

ALLOW_SUBMIT_FROM_BROWSE
Should users be allowed to issue the SUBMIT command from with a Browse session. Valid values are YES and NO.

The default is YES.

ALLOW_SUBMIT_FROM_VIEW
Should users be allowed to issue the SUBMIT command from with a View session. Valid values are YES and NO.

The default is YES.

WARN_ON_RENAME_TO_GDG_NAME
Should a warning panel be displayed when a user attempts to rename a data set to a new name that matches the naming convention of a GDG generation. The renamed data set may become a valid generation if it matches the naming convention of an existing GDG data set. If that new generation causes the LIMIT parameter value specified when the GDG was defined to be exceeded, the system will take action based on the SCRATCH/NOSCRATCH and EMPTY/NOEMPTY parameters that were specified when the GDG was defined. This action may result in one or all of the existing generations being deleted or uncataloged. Valid values are YES and NO.

The default is YES.

DEFAULT_EDIT/BROWSE/VIEW_MEMBER_LIST
Should option 3.4 (Data Set List utility) use the enhanced member list for the Edit, Browse and View actions. Performance is improved if the traditional member list ID used, but capability is improved using the enhanced member list. Valid values are YES and NO.

The default is YES.

IS_VIEW_SUPPORTED
Should users be allowed to use the View function from option 1 or only the Browse function. Because it is based on ISPF Edit, View can impact system resource utilization. Valid values are YES and NO.

The default is YES.

USE_ALTERNATE_PANEL_ISRTSOA
Should alternate ISPF Command Shell panel ISRTSOA be used in place of panel ISRTSO. ISRTSOA contains both an ISPF command line and a TSO command line, ISRTSO contains only one input field for both ISPF and TSO commands. Valid values are YES and NO.

The default is NO.

PRINT_USING_ICF
Indicates whether foreground print requests should be processed using an ICF printer definition. Valid values are YES and NO.

The default is NO.

DISALLOW_WILDCARDS_IN_HLQ
Indicates whether wildcards (* or %) are allowed in the high-level qualifier for data set list. Valid values are YES and NO.

The default is NO.

MAXIMUM_NUMBER_OF_SPLIT_SCREENS
Maximum number of separate logical screens a user can have active. The maximum value for this field is 32, and minimum value is 4.

The default is 8.

APPLID_FOR_USER_COMMAND_TABLE
The application ID for 1 to 3 user command tables. The application ID must be 1 to 4 alphanumeric or special characters, with the first character being either alphabetic or a special character. The application ID values can be specified in either of the formats:

User
to specify a single application ID.

(usr1) or (usr1,usr2) or (usr1,usr2,usr3)
to specify 1 to 3 application IDs.

The default value is NONE (no user command tables).

In addition, a special format can be used to obtain the application ID from the current system name (ISPF dialog variable ZSYSID). The special format is:

```
*, *m, or *m:n.
```

As the system name can be up to 8 characters, *m* and *n* are the start and end positions within the system name used to determine the application ID for the user command tables. The values for *m* and *n* must be in the range 1 to 8, where *m* is less than or equal to *n* and the difference in their values is no more than 3.

The default value for *m* is 1. The default value for *n* is *m*+3, to a maximum value of 8.

Compatibility Issue:
For compatibility of the ISPF configuration options with previous releases, do not specify any of the special formats (*, *m, or *m:n) as either the first user or site application ID.

APPLID_FOR_SITE_COMMAND_TABLE
The application ID for 1 to 3 site command tables. The application ID must be 1 to 4 alphanumeric or special characters, with the first character being either alphabetic or a special character. The application ID values can be specified in either of the formats:

Site
to specify a single application ID.

(sit1) or (sit1,sit2) or (sit1,sit2,sit3)
to specify 1 to 3 application IDs.

The default value is NONE (No site-wide command tables).

In addition, a special format can be used to obtain the application ID from the current system name (ISPF dialog variable ZSYSID). The special format is:

```
*, *m, or *m:n.
```

As the system name can be up to 8 characters, *m* and *n* are the start and end positions within the system name used to determine the application ID for the site command tables. The values for *m* and *n* must be in the range 1 to 8, where *m* is less than or equal to *n* and the difference in their values is no more than 3.

The default value for *m* is 1. The default value for *n* is *m*+3, to a maximum value of 8.

Compatibility Issue:
For compatibility of the ISPF configuration options with previous releases, do not specify any of the special formats (*, *m, or *m:n) as either the first user or site application ID.

SITE_COMMAND_TABLE_SEARCH_ORDER
Determines whether the site-wide command tables are searched before or after the default ISP command table. Valid values are AFTER and BEFORE.

The default is BEFORE.

Depending on this setting, the search order will be:

Table 48. Search order details

Search Order = BEFORE	Search Order = AFTER
1. Application	1. Application
2. USER (1 to 3)	2. USER (1 to 3)
3. SITE (1 to 3)	3. System
4. System	4. SITE (1 to 3)

YEAR_2000_SLIDING_RULE
The cutoff value used by ISPF to determine whether a 2-character year date specified to ISPF should be considered a 19xx or 20xx date. Values less than or equal to this date will be considered 20xx,

values greater will be considered 19xx. Value can be an absolute number or a number preceded by a minus sign to indicate the cutoff should be the specified number of years before the current year. For example, specifying 72 indicates that any 2-character year less than or equal to 72 should be considered 20xx, anything greater should be 19xx. Specifying -40 (assuming the current year is 1999) will yield a cutoff value of 59.

The default is 65.

SHOW_ENQ_DISPLAYS
Used to indicate that users should not be able to see who has existing data set ENQs when they press the help key or when they use the ISRDDN utility.

The default is blank.

DEFAULT_SESSION_LANGUAGE
Selects the language to use as the default language for ISPF. The value chosen will be the language used by ISPF if no language is specified on invocation. Valid values are:

1. English
2. Uppercase English
3. Japanese

The default is 1 (English).

PATHNAME_SUBSTITUTION_CHARACTER
This field defines the character which can be used to represent the full path name of a selected file. This character shows the position of the file name when it is specified as an argument in a z/OS UNIX command, TSO command, CLIST, or REXX exec. The substitution character can be used with commands that are invoked through the z/OS UNIX Directory List utility (ISPF option 3.17).

The default character is ! (exclamation point).

NUMBER_OF_ROWS_FOR_TBADD
The number of rows to be used in calculating the amount of storage required when a TBADD service is invoked. The value can be an integer from 1 to 1000.

The default is 1.

RETRIEVE_COMMAND_STACK_SIZE
The size, in bytes, of the command stack that RETRIEVE command uses. ISPF uses the command stack to hold commands (stripped of leading and trailing blanks) and some ISPF internal information. A stack of 512 bytes holds approximately 20 commands with a length of 10 characters each. The value can be an integer from 312 to 4096.

The default is 512.

ENABLE_ISPF_EXITS
Indicates whether exit routines are available:

YES
Indicates that installation-written routines are provided or planned, and that the ISPEXITS load module is to be loaded at ISPF initialization. If you plan to use exit routines but those routines have not been written, you can code YES for this field, causing the IBM-provided defaults module, ISPEXITS, to be loaded. Later, you can replace ISPEXITS without having to repeat this part of the installation process.

NO
Indicates that exit routines are not provided or planned, and that ISPEXITS is not to be loaded, reducing startup time.

The default is NO.

ENABLE_XTIOT
Indicates whether ISPF can process data sets dynamically allocated with an XTIOT and whether ISPF itself requests an XTIOT via dynamic allocation:

YES

XTIOT capability is requested. ISPF checks whether NON_VSAM_XTIOT support has been activated before enabling XTIOT support fully. For an explanation of this keyword in the DEVSUPxx member of parmlib, see *Using BSAM, BPAM, and QSAM support for XTIOT, uncaptured UCBs, and DSAB above the 16 MB line* in *z/OS DFSMS Using Data Sets*. The purpose of the XTIOT option of dynamic allocation is to allow very many data sets to be allocated at the same time. Without this option, you might be limited to a few hundred data sets. Some programs do not support the XTIOT option. You might have to experiment.

The ISRDDN utility indicates whether an allocation uses XTIOT and disables line command input if XTIOT support is disabled.

NO

XTIOT data sets cannot be processed.

The default is NO.

USE_EIGHT_CHARACTER_USER_ID_LAYOUTS

Specify when to use the layouts that can display 8-character TSO user IDs. Several ISPF panels and saved lists require a modified layout to display 8-character TSO user IDs. The following values are supported:

1

Use the layouts that can display 8-character TSO user IDs when 8-character user IDs are enabled on the system.

2

Always use the layouts that can display 8-character TSO user IDs, whether or not 8-character user IDs are enabled on the system.

The default value is 1.

LANGUAGE_ENVIRONMENT_RUNTIME_OPTIONS

The runtime options used for any Language Environment running under ISPF. For information on Language Environment runtime options, see *z/OS Language Environment Customization*.

The default is TRAP(ON),ABTERMENC(ABEND),TERMTHDACT(UADUMP).

VSAM data set restrictions

VSAM_RESTRICTED_EDIT_DATASET

The names of the data sets that are restricted from use in Edit. Wildcards may be used in the data set name (as in option 3.4) to specify sets of restricted data sets.

The default is NONE (all data sets allowed).

VSAM_RESTRICTED_BROWSE_DATASET

The names of the data sets that are restricted from use in Browse. Wildcards may be used in the data set name (as in option 3.4) to specify sets of restricted data sets.

The default is NONE (all data sets allowed).

VSAM_RESTRICTED_VIEW_DATASET

The names of the data sets that are restricted from use in View. Wildcards may be used in the data set name (as in option 3.4) to specify sets of restricted data sets.

The default is NONE (all data sets allowed).

VSAM_RESTRICTED_ALL_DATASET

The names of the data sets that are restricted from use in View, Edit, and Browse. Wildcards might be used in the data set name (as in option 3.4) to specify sets of restricted data sets.

The default is NONE (all data sets allowed).

ISPSPROF general values

The RESET fields described in this section are not enabled until the VERSION_LEVEL_OF_SITEWIDE_DEFAULTS field is modified. See "ISPF site-wide profile customizations" on page 228 for more details. If you select any RESET fields here, you *must* increment the value in the VERSION_LEVEL_OF_SITEWIDE_DEFAULTS field.

A RESET field causes ISPF to modify the associated value for each user. The value is reset to the value specified in the configuration field that is associated with the RESET field. For example, the RESET_LOCAL_LOG_SYSOUT_CLASS field causes the Local SYSOUT class for the log data set to be reset using the value specified in the LOCAL_LOG_SYSOUT_CLASS field. This resetting is done once each time the VERSION_LEVEL_OF_SITEWIDE_DEFAULTS field is incremented. Users can change the values of their fields after the incrementation has caused the reset.

LOG/LIST_JOB_CARD1
First job card for log/list.

The default is NONE (blank).

LOG/LIST_JOB_CARD2
Second job card for log/list.

The default is NONE (blank).

LOG/LIST_JOB_CARD3
Third job card for log/list.

The default is NONE (blank).

LOG/LIST_JOB_CARD4
Fourth job card for log/list.

The default is NONE (blank).

LOG_DATA_SET_UNIQUE_CHARACTER
This unique character is used as the default only when a first-time user profile is created.

The default is 1.

LOG_SYSOUT_CLASS
Log data set sysout class.

The default is NONE (blank).

RESET_LOG_SYSOUT_CLASS
Reset the Log data set sysout class.

The default is NO.

LOCAL_LOG_SYSOUT_CLASS
Local log data set sysout class.

The default is NONE (blank).

RESET_LOCAL_LOG_SYSOUT_CLASS
Reset the Local log data set sysout class.

The default is NO.

LOG_DISPLAY_REQUIRED
Log panel display required on termination. Valid values are YES or NO.

The default is NO.

LOG_KEPT
The log data set is to be kept. Valid values are YES or NO.

The default is NO.

LOG_LINES_PER_PAGE
The number of log lines per page.

The default is 60.

LOG_MESSAGE_ID
The log message ID. Valid values are YES or NO.

The default is NO.

LOG_LOCAL_PRINTER_ID
The local printer ID (CHAR(17)) for the log.

The default is NONE (blank).

RESET_LOG_LOCAL_PRINTER_ID
Reset the log local printer ID for the log.

The default is NO.

LOG_PAGES_PRIMARY_QUANTITY
The primary quantity of log pages.

The default is 10. Set to zero to prevent allocation of the LOG data set.

LOG_PAGES_SECONDARY_QUANTITY
The secondary quantity of log pages.

The default is 10.

LIST_DATA_SET_UNIQUE_CHARACTER
This unique character is used as the default only when a first-time user profile is created.

The default is 1.

LIST_SYSOUT_CLASS
List data set sysout class.

The default is NONE (blank).

RESET_LIST_SYSOUT_CLASS
Reset the List data set sysout class.

The default is NO.

LOCAL_LIST_SYSOUT_CLASS
Local list data set sysout class.

The default is NONE (blank).

RESET_LOCAL_LIST_SYSOUT_CLASS
Reset Local list data set sysout class.

The default is NO.

LIST_DISPLAY_REQUIRED
List panel display required on termination. Valid values are YES or NO.

The default is NO.

LIST_KEPT
The list data set is to be kept. Valid values are YES or NO.

The default is NO.

LIST_LINES_PER_PAGE
The number of list lines per page.

The default is 60.

LIST_LINE_LENGTH
The line length of the list data set.

The default is 120.

LIST_LOCAL_PRINTER_ID
The local printer ID (CHAR(17)) for the list.

The default is NONE (blank).

RESET_LIST_LOCAL_PRINTER_ID
Reset the list local printer ID for the list. Valid values are YES or NO.

The default is NO.

LIST_RECORD_FORMAT
The record format for the list.

The default is FBA.

LIST_LOGICAL_RECORD_LENGTH
The logical record length for the list.

The default is 121.

RESET_LIST_LRECL_AND_RECFM
Reset the LIST_LOGICAL_RECORD_LENGTH and LIST_RECORD_FORMAT.

The default is NO.

LIST_PAGES_PRIMARY_QUANTITY
The primary quantity of list pages.

The default is 100. Set to zero to prevent allocation of the LIST data set.

LIST_PAGES_SECONDARY_QUANTITY
The secondary quantity of list pages.

The default is 200.

UNIQUE_JOB_CHARACTER
The unique job character.

The default is NONE (blank).

SCREEN_FORMAT
The screen format. The screen format you choose depends on the type of terminal you are using or the type of terminal your emulator is emulating. These formats are available:

DATA
Format based on data width (only 3278 model 5 terminal)

STD
Format 24 lines by 80 characters

MAX
Format 27 lines by 132 characters

PART
Format using hardware partitions (only 3290 terminal)

The default is STD.

TERMINAL_TYPE
The terminal type depends on the type of terminal you are using or the type of terminal your emulator is emulating.

The default is 3278.

FAMILY_PRINTER
The printer type of the destination device. Two (2) is the only valid value and represents a QUEUED printer.

DEVICE_NAME
The device name is the destination of printed output. On MVS, this is the VTAM® node name for the printer and is installation-dependent.

The default is NONE.

RESET_DEVICE_NAME
Reset the device name specified.

The default is NO.

ASPECT_RATIO
Allows the user to preserve the Graphics Aspect Ratio for a "true" picture (0) or to preserve the positional relationship between the graphics and alphanumerics (1).

The default is zero (0).

PAD_CHARACTER
The character entered here will be used to fill input fields on a panel. It must be different than the command delimiter and it cannot be a-z, A-Z, 0-9 or /, except N and B can be used to indicate nulls and blanks respectively.

The default is B.

DELIMITER
The character entered here will be used to separate multiple commands entered on a command line. Alphabetic and alphanumeric characters as well as = (equal sign) and `.` (period) are not valid.

The default is `;` (semicolon).

RETRIEVE_MINIMUM_LENGTH
The minimum number of characters ISPF should save in the retrieve stack. Valid values are 1 through 99.

The default is 1.

RETRIEVE_CURSOR_POSITION
Cursor position relative to the retrieved command. Valid values are:

1
Place cursor at the beginning of the string.

2
Place cursor at the end of the string.

The default is 1.

ENABLE_DUMP
Enable a dump for a subtask ABEND when not in ISPF TEST mode. Valid values are ON and OFF.

The default is OFF.

TERMTRAC_DD_NAME
The terminal tracing (TERMTRAC) ddname.

The default is ISPSNAP.

RESET_TERMTRAC_DD_NAME
Reset the TERMTRAC ddname specified. Valid values are YES and NO.

The default is NO.

ENVIRON_TERMTRAC_VALUE
Enable terminal tracing. Valid values are ON, OFF and ERROR.

The default is OFF.

FKA_SETTING
Current state of the function key form. Valid values are LONG, SHORT and OFF (no display).

The default is LONG.

NUMBER_OF_PFKEYS
Number of function keys. Valid values are 12 or 24.

The default is 12.

CHARACTER_SET_LOAD_MODULE
Character set load module name.

The default is ISP3278.

RESET_CHARACTER_SET_LOAD_MODULE
Reset the character set load module.

The default is NO.

FRAME_COLOR
The color for window frames. Valid values are:

1
Blue

2
Red

3
Pink

4
Green

5
Turquoise

6
Yellow

7
White

The default is 6 (Yellow).

FRAME_INTENSITY
The intensity for window frames. Valid values are 0 (Low) and 2 (High).

The default is 2 (High).

TPUT_BUFFER_BLOCKSIZE
The TPUT buffer block size can range from 512 to 32767. If defined as 0, ISPF will use its own calculations to determine the TPUT buffer size. If defined within the range 512 to 32767, ISPF will use the defined value as the buffer block size.

The default is 0.

GLOBAL_COLORS
Determines the colors that are displayed while running ISPF in host mode. The string is a 7 digit number, each digit specifying the color to be substituted for another host color. The order of the digits in the string, and the number that represents each color is:

1. Blue
2. Red
3. Pink
4. Green
5. Turquoise
6. Yellow
7. White

For example, to substitute blue for green and accept all other default colors, specify the number for blue (**1**) in the position for green (fourth position): 123**1**567.

The default is 1234567.

DEFAULT_MESSAGE_ID
Select to display the message identifier. The valid values are OFF or ON, the default is OFF.

DEFAULT_PANEL_ID[1]
Select to display the panel identifier. The valid values are OFF or ON, the default is OFF.

DEFAULT_SCREEN_NAME[1]
Select to display the screen name. The valid values are OFF or ON.

The default is OFF.

DEFAULT_SYSTEM_NAME[1]
Select to display the system name. The valid values are OFF or ON.

The default is OFF.

DEFAULT_USERID_DISP[1]
Select to display the user identifier. The valid values are OFF or ON.

The default is OFF.

DEFAULT_PRIMARY_PANEL
Enter a panel name for the default primary panel.

The default is ISP@MSTR.

DEFAULT_LIBDEF_PROCESSING_OPTION
Enter the default option for processing LIBDEF requests. The valid values are COND, UNCOND, STACK, or STKADD.

The default is UNCOND.

Note: Use caution when changing the value specified for this keyword. A change to the value can cause unexpected results for ISPF dialogs that use the LIBDEF service.

ENABLE_EURO_SYMBOL
Enable the Euro currency symbol. Valid values are YES or NO.

The default is NO.

RESET_ENABLE_EURO_SYMBOL
Reset enable the Euro currency symbol field.

The default is NO.

DATE_FORMAT_ZDATEFD
The date format using the national language convention for the day, month, year and the national language separator. If the value is DEFAULT ISPF will use the value defined in the ISPF Literal Load Module.

The default is DEFAULT.

DATE_FORMAT_ZDATEF
The date format using the characters DD for day, MM for month, and YY for year and the national language separator. The order of year month and day may change. If the value is DEFAULT, ISPF uses the value defined in the ISPF Literal Load Module.

The default is DEFAULT.

DEFAULT_TIME_SEPARATOR
The separator used in the time of day format. For example, the colon (**:**) in hh**:**mm**:**ss. If the value is D, ISPF uses the value defined in the ISPF Literal Load Module.

The default is the D.

Note:

1. These options control a shared 17-byte information area that overlays any panel display. For a description and references to the system commands that also affect this area, see "Understanding ISPF panels" in *ISPF User's Guide Volume 1*.

Chapter 10. Exits

This section lists the exit macros and installation-wide exits.

Exit macros

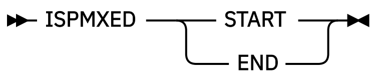
You define the ISPF installation-wide exits and installation-written exit routines you want to use by placing entries in the exit definition table, ISPXDT. ISPXDT consists of two sections:

Exit entry definitions
Defines the installation-wide exits and their associated exit routines

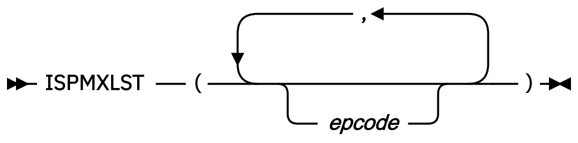
Exit data area definitions
Defines the data areas that are used by the exit routines and their size specifications

The ISPMXED, ISPMXLST, ISPMXDEF, ISPMEPT, and ISPMXEND macros define the exit entry definition section of the table, and the ISPMXDD and ISPMDAD macros define the exit data area definitions.

ISPMXED
Defines the start or the end of the exit entry definition section.



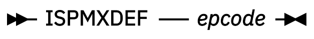
ISPMXLST
Defines the installation-wide exits where you provide exit routines.



where:

epcode
The numeric code for a user exit provided by ISPF. The list of codes must be enclosed in parentheses, and must be in ascending order. See Table 24 on page 135 for a list of the numeric codes.

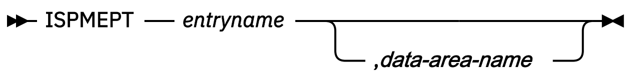
ISPMXDEF
Begins the definition of exit routines for a particular user exit.



where:

epcode
The numeric code for an exit point provided by ISPF. You must include this code as an operand for the ISPMXLST macro in order for the user exit to be defined. See Table 24 on page 135 for a list of the numeric codes.

ISPMEPT
Identifies an exit routine to call at a particular user exit.



where:

entryname
Identifies the entry point of the exit routine. This is usually a CSECT name.

data-area-name
Identifies the name of the data area the exit routine uses. The name can be up to 8 characters long. If you list the data area on an ISPMEPT macro, you must define it using the ISPMDAD macro. If you do not specify a data area on the ISPMEPT macro, ISPF does not provide a data area for the exit routine being defined.

Note: Do not use 'NULLAREA' as a *data-area-name*.

ISPMXEND
Ends the definition of routines for a particular user exit. This macro explicitly ends the ISPMXDEF macro and must be included.

```
►► ISPMXEND ►◄
```

ISPMXDD
Indicates the start or the end of the data-area definition section of ISPXDT. Even if an exit routine does not require a data area, you must code the ISPMXDD START and ISPMXDD END macros.

```
►► ISPMXDD ─┬─ START ─┬─►◄
            └─ END ───┘
```

ISPMDAD
Defines a data area and its size. The data area can be used by one or more exit routines.

```
►► ISPMDAD ── data-area-name ,size ►◄
```

where:

data-area-name
Identifies the name of the data area an exit routine uses. The name can be up to 8 characters long.

size
Specifies, in bytes, the size of the data area. If the size is not a multiple of eight, ISPF rounds it up to the next multiple of eight.

Exit 1: ISPF session initialization exit

The ISPF session initialization exit provides accounting and monitoring capabilities. ISPF gives control to the exit routine after successfully opening all required data sets, determining session language, and before the first logical screen is started.

Exit parameters

In addition to the standard exit parameters described in "Exit parameter list" on page 139, the exit routines at this user exit receive these parameters.

Flags
4 bytes of bit flags defined as follows:

0
1 = TEST option specified

1
1 = TESTX option specified

2
1 = TRACE option specified

3
: 1 = TRACEX option specified

4-31
: Reserved.

CPPL
: Pointer to the address of the TSO command processor parameter list (CPPL) that was passed to ISPF.

Lang
: An 8-character field that contains the name of the national language for this ISPF session. The value is left-justified and padded with blanks.

Return codes

No return codes are acknowledged at this user exit. When this user exit returns to ISPF, normal processing continues.

Note: For multiple exit routines, return codes still affect the processing flow.

Exit 2: ISPF session termination exit

This user exit also provides accounting and monitoring capabilities. ISPF invokes routines at this exit after the last logical screen terminates and just before ISPF terminates.

If the ISPF main task terminates abnormally, routines at this user exit are not invoked. You can create an ESTAE exit in the ISPF initialization exit if you want to detect abnormal product termination. However, this applies only to product abnormal termination and not to abends of the logical screen task.

Exit parameters

In addition to the standard exit parameters described in "Exit parameter list" on page 139, the exit routines at this user exit receive these parameters.

ISPF Return Code:
: A fullword binary number that contains the value of ZISPFRC as set by the application on the ISPSTART command or by ISPF. See the *z/OS ISPF Dialog Developer's Guide and Reference* for return codes set by ISPF. Any change made to this parameter by the exit routine is ignored.

Return codes

No return codes are acknowledged at this user exit. ISPF termination continues upon return from the exit routine.

Note: For multiple exit routines, return codes still affect the processing flow.

Exit 3: SELECT service start exit

In addition to providing information you can use to monitor ISPF, this user exit lets you restrict access to applications selected through ISPF. ISPF invokes the exit routines when the SELECT service is invoked. The SELECT service is invoked by:

- ISPSTART command
- SPLIT command (not when already in split-screen mode)
- SPLIT NEW command
- START command

- ISPSTRT program interface
- Selection menu entry
- Command table entry
- SELECT dialog service

The exit routines are given control after the SELECT request has been parsed and syntax checking is complete.

Changes you make to the Screen name are **not** reflected in the SCRNAME value passed in this exit.

Exit parameters

In addition to the standard exit parameters described in "Exit parameter list" on page 139, the exit routines at this user exit receive these parameters. ISPF ignores any changes the exit routines make to these parameters.

Flags
4 bytes of bit flags defined as follows:

0
1 = PGM keyword specified

1
1 = CMD keyword specified

2
1 = PANEL keyword specified

3-4
Reserved

5
1 = ADDPOP keyword specified

6
1 = BARRIER keyword specified

7
1 = NEST keyword specified

8
1 = NEWAPPL keyword specified

9
1 = NEWPOOL or NEWAPPL keyword specified

10
1 = PASSLIB keyword specified

11
1 = Lang (CREX) keyword specified

12
1 = Lang (APL) keyword specified

13
1 = MODE(FSCR) specified

14
1 = MODE(LINE) specified

15
Reserved

16
1 = PARM value passed to the exit was truncated to 258 bytes, which include a halfword length field and 256 bytes of data.

17
: 1 = SUSPEND specified

18
: 1 = LOGO keyword specified

19-31
: Reserved.

elemname
: An 8-character field that contains the name of the element (PGM, CMD, or PANEL) to be selected.

APPLID
: A 4-character field that contains the current application ID. If NEWAPPL was specified on the SELECT request, this field contains the new APPLID as specified. If APPLID field is blank, the previously specified application ID is implied. The value is left justified and padded with blanks.

PARM
: Input parameters being passed to a program dialog in the same format as when passed to the program; single character string preceded by a halfword containing its length (the length value does not include itself). FLAGS bit 16 is set when this parameter represents a truncated PARM value. The Length field is not affected by truncation.

logoname
: An 8-character field that contains the name of the LOGO panel if it was specified.

Screen name
: Screen name as set by the SCRNAME command and shown by the SCRNAME ON function.

Return codes

0
: Normal completion; ISPF continues processing.

8
: Authorization failure; The SELECT request is not processed. Instead, ISPF issues a line mode message stating that the user exit indicated an authorization failure and terminates the select service, but allows the active application to continue.

16
: Authorization failure; the SELECT request is not processed. Instead, ISPF issues a message indicating that the user exit indicated an authorization failure and terminates the SELECT service processing with a severe error (RC=20).

Other
: ISPF treats the error as severe and issues a message indicating that the exit routine returned an incorrect return code.

Exit 4: select service end exit

You can use this user exit to mark the end of a program, command, or menu invoked through any of the SELECT services, with or without the SELECT service start exit.

Changes you make to the screen name are not reflected in the SCRNAME value passed in this exit.

Exit parameters

In addition to the standard exit parameters described in "Exit parameter list" on page 139, the exit routines at this user exit receive these parameters. ISPF ignores any changes the exit routines make to the parameters.

Flags
: 4 bytes of bit flags defined as follows:

0
: 1 = PGM keyword specified

1
: 1 = CMD keyword specified

2
: 1 = PANEL keyword specified

3-4
: Reserved

5
: 1 = ADDPOP keyword specified

6
: 1 = BARRIER keyword specified

7
: 1 = NEST keyword specified

8
: 1 = new application being invoked

9
: 1 = new shared pool is to be created

10
: 1 = PASSLIB keyword specified

11
: 1 = Lang (CREX) keyword specified

12
: 1 = APL keyword specified

13
: 1 = MODE(FSCR) specified

14
: 1 = MODE(LINE) specified

15-16
: Reserved

17
: 1 = SUSPEND specified

18
: 1 = LOGO keyword specified

19-31
: Reserved.

Note: If no parameters were passed, the flag bytes will be zero.

elemname
An 8-character field that contains the name of the element (PGM, CMD, or PANEL) to be selected. In case of CMD, if the element is a CLIST prefixed with %, the % symbol is removed from the elemname.

APPLID
A 4-character field that contains the application ID of the element that just terminated.

STATS
Two contiguous fullwords containing:

Total "think time" in hundredths of a second for this SELECT level only. "Think® time" means all of the time intervals between the time when ISPF unlocked the keyboard for input and the time when input was received by ISPF. If there are other SELECTs nested within this one, their statistics are given at the corresponding select end exits, but not included here.

The number of times the screen was read by ISPF in this SELECT level.

logoname
An 8-character field that contains the name of the LOGO panel if it was specified.

Screen name
Screen name as set by the SCRNAME command and shown by the SCRNAME ON function.

Return codes

No return codes are acknowledged at this user exit. ISPF termination continues upon return from the exit routine.

Note: For multiple exit routines, return codes still affect the processing flow.

Exit 5: TSO command start exit

You can use this user exit to monitor and restrict commands invoked through ISPF. In addition, you can also use the user exit to allow TSO commands newly added to the system to be invoked from within ISPF without updating the ISPF TSO command table (ISPTCM). If the invoked command is not an implicit CLIST (not prefixed by %), the TCM is searched. This exit is called immediately after searching the TCM. For implicit CLISTs, the exit is called before attaching the exec processor.

Note: This user exit will not be invoked for TSO commands issued from REXX execs.

Exit parameters

In addition to the standard exit parameters described in "Exit parameter list" on page 139, the exit routines at this user exit receive these parameters:

cmdname
An 8-character field that contains the name of the command that was invoked. The name is left-justified within the field and padded with blanks.

If the command is a CLIST, this field contains the CLIST name. The CLIST name will not include the % prefix if one was used.

user flags
1 or more bytes of installation data as defined when ISPTCM is generated. See "Customizing the ISPF TSO command table (ISPTCM)" on page 129 for more information.

TCM flags
The flag byte in ISPTCM that ISPF uses to determine what processing should be done for the command. If the current command is not in ISPTCM, the default flag byte is provided. The exit routine can change this value and ISPF uses the value if it receives a return code of 4. See "Customizing the ISPF TSO command table (ISPTCM)" on page 129 and "ISPTCM usage notes" on page 132 for information about TCM flags.

CPPL
The address of the TSO command processor parameter list (CPPL) that was passed to ISPF.

When the command is an implicit CLIST (prefixed by %), the TCM is not searched. In those cases, the user flags and TCM flags parameters will be binary zeros.

Return codes

0
Normal completion; ISPF continues processing.

4
The exit routine has changed the value of the TCM flag byte. In this case, ISPF uses the value of the flag to process the command. However, changes to the TCM flag bit indicating whether the command should be logged are ignored.

16
: Authorization failure. The exit routine indicated that this command should not be attached. ISPF issues an error message indicating that the exit routine rejected the command and returns to the caller.

Other
: ISPF treats the error as severe and issues a message indicating that the exit routine returned an incorrect return code.

Exit 6: TSO command end exit

You can use this user exit to monitor TSO commands invoked from within ISPF. This exit is called after the attached command completes (or if attach fails).

Note: This user exit will not be invoked for TSO commands issued from REXX execs.

Exit parameters

In addition to the standard exit parameters described in "Exit parameter list" on page 139, the exit routines at this user exit receive these parameters. ISPF ignores any changes made to these parameters.

cmdname
: An 8-character field that contains the name of the command to be attached. The name is left-justified within the field.

CLIST name
: An 8-character field that contains the name of the CLIST if the command was a CLIST. In that case, cmdname is 'EXEC'. If the command is not a CLIST, this field is the same as cmdname.

user flags
: 1 or more bytes of installation data as defined when ISPTCM is generated. See "Customizing the ISPF TSO command table (ISPTCM)" on page 129 for more information.

TCM flags
: The flag byte in ISPTCM that ISPF uses to determine what processing should be done for the command. If the current command is not in ISPTCM, the default flag byte is provided.

CPPL
: The address of the TSO command processor parameter list (CPPL) that was passed to ISPF.

When the command is an implicit CLIST (prefixed by %), the TCM is not searched. In those cases, the user flags and TCM flags parameters will be binary zeros.

Return codes

No return codes are acknowledged at this user exit. When this exit returns to ISPF, normal processing continues.

Note: For multiple exit routines, return codes still affect the processing flow.

Exit 7: LIBDEF service exit

This user exit lets you restrict the use of the LIBDEF service. ISPF passes control to the exit routines at this user exit after determining that a syntactically valid call has been made and before allocating and opening the alternate library.

Exit parameters

In addition to the standard exit parameters described in "Exit parameter list" on page 139, the exit routines at this user exit receive these parameters. ISPF ignores any changes the exit routines make to the parameters.

libtype
An 8-character field that contains the library type as specified on the LIBDEF request. The value is left-justified and padded with blanks.

Flags
4 bytes of bit flags defined as follows:

0
1 = DATASET keyword specified

1
1 = LIBRARY keyword specified

2
1 = EXCLDATA keyword specified

3
1 = EXCLLIBR keyword specified

4
1 = COND request

5
1 = STACK request. If bit 4 and 5 are both 0, an UNCOND request was done

6
1 = STKADD request. Valid if bit 0 equals 1

7-31
Reserved

dsname#
A fullword binary number indicating the number of elements in the lengths and names arrays that follow. The number will not exceed 15.

lengths
An array of fullwords indicating the lengths of the corresponding elements of the names array. The maximum length of a dsname is 44. If a libname is provided using the LIBRARY or EXCLLIBR keyword, the length will be eight.

names
An array containing the data set names or library name as specified in the ID parameter of the LIBDEF service. Data set names are fully qualified without quotes. Each element can be up to 44 characters long and those names less than 44 characters are padded on the right with blanks.

Return codes

0
Normal completion; ISPF continues processing.

16
Authorization failure. ISPF issues a message indicating an authorization failure has occurred and LIBDEF terminates with a severe error.

Other
LIBDEF treats return codes other than 0 or 16 as a severe error and returns a code of 20 to the caller.

Exit 8: RESERVE exit

This user exit lets you use your own method for serializing resources in addition to the RESERVE done by ISPF. See Appendix A, “ISPF enqueue processing for data integrity,” on page 315 for information about ISPF's use of ENQ and RESERVE. ISPF gives control to the exit routine before it does the RESERVE. If your serialization mechanism cannot acquire the resource, or if the exit routine returns a code greater than 0, ISPF does not attempt to get it.

The RESERVE exit is still called if Configuration table keyword PDSE_RESERVE_PROCESSING is set to OFF.

Exit parameters

In addition to the standard exit parameters described in “Exit parameter list” on page 139, the exit routines at this user exit receive these parameters. ISPF ignores any changes made to these parameters.

qname
An 8-character field that contains the queue name against which the RESERVE is done. (ISPF uses a qname of 'SPFEDIT '.)

rname
The 44-character resource name (data set name) that is reserved.

ucbaddr
A fullword field containing the address of the unit control block (UCB) for the reserved device.

Flags
4 bytes of bit flags defined as follows:

0
1 = Exclusive reserve requested. 0 = Shared reserve requested. (ISPF always requests exclusive use of the resource at RESERVE.)

1
1 = A RESERVE or ENQ will be done on qname 'SYSIEWLP' (the linkage editor qname).

2
1 = The resource being reserved is a partitioned data set.

3
1 = The resource being reserved is a sequential data set.

4
1 = The resource being reserved is a PDSE.

5
1 = PDSE_RESERVE_PROCESSING setting. On by default

6-31
Reserved.

Note: The exit routine should not use the same (qname, rname) combination (in a RESERVE macro call) that ISPF uses. If that happens, an abend occurs when ISPF attempts to do its own RESERVE.

Return codes

0
Normal completion; ISPF does its own RESERVE and continues processing.

16
Resource not available. ISPF issues a message showing that the exit routine indicated the resource is not available. This results in a failure of the service requesting the RESERVE.

Other
ISPF treats other return codes similar to a return code of 16 and issues a message indicating that the exit routine returned an incorrect return code.

Exit 9: RELEASE exit

Use this user exit to release any resources acquired at the RESERVE user exit. ISPF relies on task termination to release any resources reserved by the abending logical screen task. This user exit does not get control in the case of abnormal termination of the logical screen.

If your serialization mechanism is such that task termination will not release the reserved resources, an exit routine is provided at the logical screen end user exit to clean up any unreleased resources for that task.

The RESERVE done by ISPF is released before the exit routine is given control.

The RELEASE exit is still called if Configuration table keyword PDSE_RESERVE_PROCESSING is set to OFF.

Exit parameters

In addition to the standard exit parameters described in "Exit parameter list" on page 139, the exit routines at this user exit receive these parameters. ISPF ignores any changes made to these parameters.

qname
An 8-character field that contains the queue name against which the RESERVE is done. (ISPF uses a qname of 'SPFEDIT '.)

rname
The 44-character resource name (data set name) that is reserved.

Return codes

No return codes are acknowledged at this user exit. When this exit returns to ISPF, normal processing continues.

Note: For multiple exit routines, return codes still affect the processing flow.

Exit 10: logical screen start exit

You can use this user exit to gather accounting and monitoring information for each ISPF logical screen. The exit routine is given control just before the logical screen task is attached.

Exit parameters

This user exit uses only the standard exit parameters described in "Exit parameter list" on page 139, which includes the logical screen identifier (screenid).

Return codes

No return codes are acknowledged at this user exit. When this exit returns to ISPF, normal processing continues.

Note: For multiple exit routines, return codes still affect the processing flow.

Exit 11: logical screen end exit

This user exit, which is similar to the logical screen start user exit, lets you gather accounting and monitoring information for each ISPF logical screen. It gives you control for both normal and abnormal termination of a logical screen. You can use it to perform necessary cleanup required as a result of exits being bypassed because of abnormal termination. Specifically, these end user exits cannot gain control because of a logical screen abend even though the corresponding start installation exits did get control:

- SELECT service end exit
- RELEASE exit
- ISPF service end exit.

Note: The TSO command end exit gets control for both normal and abnormal termination of attached commands.

Exit parameters

In addition to the standard exit parameters described in "Exit parameter list" on page 139, the exit routines at this user exit receive these parameters:

Flags
4 bytes of bit flags defined as follows:

0
0 = Normal termination 1 = Abnormal termination.

1-31
Reserved

Next logical screen
An 8-character field that identifies the next active ISPF logical screen. In the case of exiting ISPF, the next logical screen ID will be 0.

Return codes

No return codes are acknowledged at this user exit. When this exit returns to ISPF, normal processing continues.

Note: For multiple exit routines, return codes still affect the processing flow.

Exit 12: ISPF service start exit

The ISPF exit for the service start lets you monitor all external ISPF service requests. Exit routines at this user exit are notified of all service requests (including PDF services) made through the ISPLINK or ISPEXEC interfaces. The exit routines are not notified for ISPF internal service requests (those that do not use the ISPLINK or ISPEXEC interfaces.)

After initial verification of parameters and syntax, ISPF calls the exit routines at this user exit before the requested service is performed.

Exit parameters

In addition to the standard exit parameters described in "Exit parameter list" on page 139, the exit routines at this user exit receive these parameters. ISPF ignores any changes the routines make to these parameters.

servname
An 8-character field containing the name of the ISPF service being invoked. The name is left-justified within the field.

Flags
4 bytes of bit flags defined as follows:

0
1 = ISPF service 0 = PDF service

1-31
Reserved.

Return codes

No return codes are acknowledged at this user exit. When this exit returns to ISPF, normal processing continues.

Note: For multiple exit routines, return codes still affect the processing flow.

Exit 13: ISPF service end exit

You can use this user exit to mark the termination of ISPF dialog services invoked through the ISPLINK or ISPEXEC interfaces.

If a severe error occurs causing a logical screen abend, ISPF does not give control to the exit routines at this user exit. You can also stack exit routines at the logical screen end user exit to ensure that service termination is correctly recorded.

Exit parameters

In addition to the standard exit parameters described in "Exit parameter list" on page 139, the exit routines at this user exit receive these parameters. ISPF ignores any changes the exit routines make to the parameters.

servname
An 8-character field containing the name of the ISPF or PDF service being invoked. The name is left-justified within the field.

Flags
4 bytes of bit flags defined as follows:

0
1 = ISPF service 0 = PDF service

1-31
Reserved.

Return codes

No return codes are acknowledged at this user exit. When this exit returns to ISPF, normal processing continues.

Note: For multiple exit routines, return codes still affect the processing flow.

Exit 14: SWAP exit

Using this user exit, you can indicate a change of the active logical screen. Together with the logical screen start and end user exits, it allows resource use to be monitored for each ISPF logical screen.

ISPF calls the exit routines at this user exit just after the logical screen to be given control is activated. In addition, ISPF calls the exit routines if the SPLIT command is entered when the screen is already split.

Exit parameters

This user exit uses only the standard exit parameters described in "Exit parameter list" on page 139, which includes the logical screen identifier (screenid). The screen ID identifies the logical screen activated as a result of the SWAP command.

Return codes

No return codes are acknowledged at this user exit. When this exit returns to ISPF, normal processing continues.

Note: For multiple exit routines, return codes still affect the processing flow.

Exit 15: DISPLAY service exit

This user exit is provided for installation tailoring purposes. This allows you to selectively replace ISPF, PDF, or other ISPF-based panels with your own versions of the panels. You can control the amount of information presented to your users based on their experience with the panels. The exit routines can change a number of parameters at this user exit.

ISPF calls the exit routines at this user exit before the display of all the panels (as a result of internal or external display requests) except for the display of severe error or abend panels or display requests where a panel ID is not specified. This includes displays caused by the DISPLAY, TBDISPL, EDIT, SELECT, and BROWSE services.

Changes you make to the Screen name are **not** reflected in the SCRNAME value passed in this exit.

Exit parameters

In addition to the standard exit parameters described in "Exit parameter list" on page 139, the exit routines at this user exit receive these parameters.

panel-id

An 8-character field that contains the name of the panel to be displayed. If the display request did not specify a panel-id (indicating that the previously displayed panel is to be redisplayed), the exit routine is not invoked.

This parameter can be changed. If the parameter is changed, the name of the panel in the field must be left-justified.

message-id

An 8-character field that contains the message-id to be displayed on the panel, as specified on the display request. Messages displayed as a result of the SETMSG service are not identified in this field. If a message-id is not specified, the parameter contains blanks.

This parameter can be changed. If the parameter is changed, the message-id in the field must be left-justified.

cursor-field

An 8-character field that contains the field name on which the cursor is to be positioned. This field contains blanks if the cursor-field is not explicitly specified (the cursor is placed by defaults).

This parameter can be changed. If the parameter is changed, the field name must be left-justified.

cursor-offset

A fullword binary number that contains the offset within the cursor-field where the cursor is to be positioned. If not explicitly specified on the display request, this parameter has a value of zero.

This parameter can be changed.

table-name

An 8-character field that contains the name of the table to be displayed if the display request is a result of the TBDISPL service. Otherwise, the field contains blanks.

This parameter *cannot* be changed.

Flags

4 bytes of bit flags defined as follows:

0
1 = Non-display mode is active; the panel is processed but not displayed.

1
1 = COMMAND option was specified on the DISPLAY request.

2-31
Reserved.

message-field
An 8-character field that contains the name of the panel field the message pop-up window is to be positioned adjacent to. This field contains blanks if the MSGLOC parameter is not specified.

This parameter can be changed by the exit routine.

Screen name
Screen name as set by the SCRNAME command and shown by the SCRNAME ON function.

Return codes

0
Normal completion; ISPF continues processing.

4
The exit routine changed one or more of the parameters. ISPF continues processing using the changed parameter values.

Other
ISPF treats the error as severe and issues a message indicating that the exit routine issued an incorrect return code.

Note: Panel functions, such as RESP, that are coded in the panel)INIT section are not processed before the exit is entered. This could prevent a display although non-display is not indicated by the exit 15 parameter.

Exit 16: log, list, trace, and temporary data set allocation exit

This user exit lets you maintain your own data set naming conventions. ISPF calls the routines at this user exit before allocating the log, list, or temporary control data sets. As a result, you can provide a prefix for the name of the data set to be allocated. However, if the data set has been preallocated, ISPF does not use this prefix.

The exit routine can provide a prefix up to 26 characters long. A zero length prefix is flagged as an error. ISPF reserves the remaining 18 characters of the data set name for its own use.

ISPF builds the names of the log, list, and temporary control data sets according to these rules:

1. If a data set prefix is specified in the TSO user profile table (UPT) and it is different from the user ID, the data set name is of the form:

   uptpfx.userid.ISPF-specific-suffix

2. If a prefix is not specified in the UPT or if it is the same as the TSO user ID, the data set name is:

   userid.ISPF-specific-suffix

3. If a user ID is not available (executing ISPF in BATCH), ISPF recommends that the TSO PROFILE command be used to place the user ID in the UPT prefix field. In that case, the data set name has the this form:

   uptpfx.ISPF-specific-suffix

   Note that UPTPFX is assumed to be the user ID in this case.

If you provide an exit routine at this user exit, ISPF allows the data set names to be of the form:

```
Exit-provided-prefix.ISPF-specific-suffix
```

The 18 characters reserved by ISPF begin with the period separator.

Because the user ID is also passed to the exit routine as part of the prefix, the exit routine is responsible for maintaining unique data set names.

Exit parameters

In addition to the standard exit parameters described in "Exit parameter list" on page 139, the exit routines at this user exit receive these parameters.

prefix-len
: A fullword binary number that identifies the length of the prefix field. On entry to the exit routine, it is the length of the prefix including the UPT prefix, the user ID or both. On return to ISPF, it should contain the length of the prefix provided by the exit routine.

  The value of this parameter must be in the range 1 to 26, inclusive.

prefix
: A 26-character field that contains the data set name prefix used by ISPF. On entry to the exit routine, it contains the UPT prefix, the user ID, or both left-justified within the field.

  On return to ISPF, it can contain any prefix (up to 26 characters) chosen by the exit. ISPF does not do any validity checking of the specified prefix. This prefix must be left-justified. If an incorrect prefix is provided, allocation of the data set fails.

suffix-type
: A fullword of bit flags indicating the type of data set that ISPF allocates:

  0
  : 1 = List data set

  1
  : 1 = Log data set

  2
  : 1 = Temporary listing data set

  3
  : 1 = Temporary control data set

  4
  : 1 = Temporary work data set

  5
  : 1 = ISPVCALL trace data set

  6
  : 1 = ISPDPTRC trace data set

  7
  : 1 = ISPFTTRC trace data set

  8-31
  : Reserved.

suffix-len
: A fullword binary number containing the length of the value within the suffix field. ISPF ignores any changes the exit routine makes to this field.

suffix
: An 18-character field containing the name of the data set name suffix that ISPF uses. ISPF ignores any changes the exit routine makes to this field.

Return codes

No return codes are acknowledged. Upon return to ISPF, the data set name is generated using the prefix provided by the exit routine (if any) and normal processing continues.

Note: For multiple exit routines, return codes still affect the processing flow.

Chapter 11. PDF translation tables

PDF translation tables perform many functions for PDF. Each section describes one particular use.

Translation table for valid data set name characters

The translation table for valid data set name characters (Table 49 on page 273) specifies which characters are allowed in a data set name, as follows:

- Valid characters are represented with X'00'
- Invalid characters are represented with X'FF'.

Table 49. Example of translation table for valid data set characters

Table	Hexadecimal Code	Position
TTVDSN	`DC X'FFFFFFFFFFFFFFFF'`	`(X'00' to X'07')`
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'08' to X'0F')`
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'10' to X'17')`
	`...`	
	`DC X'FFFF000000FFFFFF'`	`(X'78' to X'7F')`
	`DC X'FF00000000000000'`	`(X'80' to X'87')`
	`...`	
	`DC X'0000FFFFFFFFFFFF'`	`(X'E8' to X'EF')`
	`DC X'0000000000000000'`	`(X'F0' to X'F7')`
	`DC X'0000FFFFFFFFFFFF'`	`(X'F8' to X'FF')`

Translation table for invalid data set name characters

The translation table for invalid data set name characters (Table 50 on page 273) specifies which characters are not allowed in a data set name, as follows:

- Valid characters are represented by their EBCDIC hexadecimal code
- Invalid characters are represented with X'00'.

Table 50. Example of translation table for invalid data set name characters

Table	Hexadecimal Code	Position
TTIDSN	`DC X'0000000000000000'`	`(X'00' to X'07')`
	`DC X'0000000000000000'`	`(X'08' to X'0F')`
	`DC X'0000000000000000'`	`(X'10' to X'17')`
	`...`	
	`DC X'00007A7B7C000000'`	`(X'78' to X'7F')`
	`DC X'0081828384858687'`	`(X'80' to X'87')`
	`...`	
	`DC X'E8E9000000000000'`	`(X'E8' to X'EF')`
	`DC X'F1F2F3F4F5F6F700'`	`(X'F0' to X'F7')`
	`DC X'F8F9000000000000'`	`(X'F8' to X'FF')`

Translation table for hexadecimal characters

The translation table for hexadecimal characters (Table 51 on page 274) specifies the valid hexadecimal characters as follows:

- Valid characters are represented with X'00'
- Invalid characters are represented with X'FF'.

Table 51. Example of translation table for hexadecimal characters

Table	Hexadecimal Code	Position

TTHEX	`DC X'FFFFFFFFFFFFFFFF'`
	`DC X'FFFFFFFFFFFFFFFF'`
	`DC X'FFFFFFFFFFFFFFFF'`
	`...`
	`DC X'FFFFFFFFFFFFFFFF'`
	`DC X'FF000000000000FF'`
	`...`
	`DC X'FFFFFFFFFFFFFFFF'`
	`DC X'0000000000000000'`
	`DC X'0000FFFFFFFFFFFF'`
		`(X'00' to X'07')`
		`(X'08' to X'0F')`
		`(X'10' to X'17')`
		
		`(X'78' to X'7F')`
		`(X'80' to X'87')`
		
		`(X'E8' to X'EF')`
		`(X'F0' to X'F7')`
		`(X'F8' to X'FF')`

Translation table for numeric characters

The translation table for numeric characters (Table 52 on page 274) specifies the valid numeric characters as follows:

- Valid characters are represented with X'00'
- Invalid characters are represented with X'FF'.

Table 52. Example of translation table for numeric characters

Table	Hexadecimal Code	Position

TTNUM	`DC X'FFFFFFFFFFFFFFFF'`
	`DC X'FFFFFFFFFFFFFFFF'`
	`DC X'FFFFFFFFFFFFFFFF'`
	`...`
	`DC X'FFFFFFFFFFFFFFFF'`
	`DC X'FFFFFFFFFFFFFFFF'`
	`...`
	`DC X'FFFFFFFFFFFFFFFF'`
	`DC X'0000000000000000'`
	`DC X'0000FFFFFFFFFFFF'`
		`(X'00' to X'07')`
		`(X'08' to X'0F')`
		`(X'10' to X'17')`
		
		`(X'78' to X'7F')`
		`(X'80' to X'87')`
		
		`(X'E8' to X'EF')`
		`(X'F0' to X'F7')`
		`(X'F8' to X'FF')`

Translation table for alphanumeric characters

The translation table for alphanumeric characters (Table 53 on page 274 specifies the valid alphanumeric characters as follows:

- Valid characters are represented with X'00'.
- Invalid characters are represented with X'FF'.

Table 53. Example of translation table for alphanumeric characters

Table	Hexadecimal Code	Position

TTALN	`DC X'FFFFFFFFFFFFFFFF'`
	`DC X'FFFFFFFFFFFFFFFF'`
	`DC X'FFFFFFFFFFFFFFFF'`
	`...`
	`DC X'FFFFFF0000FFFFFF'`
	`DC X'FF00000000000000'`
	`...`
	`DC X'0000FFFFFFFFFFFF'`
	`DC X'0000000000000000'`
	`DC X'0000FFFFFFFFFFFF'`
		`(X'00' to X'07')`
		`(X'08' to X'0F')`
		`(X'10' to X'17')`
		
		`(X'78' to X'7F')`
		`(X'80' to X'87')`
		
		`(X'E8' to X'EF')`
		`(X'F0' to X'F7')`
		`(X'F8' to X'FF')`

Translation table for edit terminal output characters

The translation table for edit terminal output characters (Table 54 on page 275) is used to translate invalid edit display characters to an attribute byte as follows:

- Valid characters are represented by their EBCDIC value
- Invalid characters are represented with X'15'.

Table 54. Example of translation table for edit terminal output character

Table	Hexadecimal Code	Position
TTETO	`DC X'1515151515151515'`	`(X'00' to X'07')`
	`DC X'1515151515151515'`	`(X'08' to X'0F')`
	`DC X'1515151515151515'`	`(X'10' to X'17')`
	...	
	`DC X'15797A7B7C7D7E7F'`	`(X'78' to X'7F')`
	`DC X'1581828384858687'`	`(X'80' to X'87')`
	...	
	`DC X'E8E9151515151515'`	`(X'E8' to X'EF')`
	`DC X'F0F1F2F3F4F5F6F7'`	`(X'F0' to X'F7')`
	`DC X'F8F9151515151515'`	`(X'F8' to X'FF')`

Translation table for generic string characters

The translation table for generic string characters (Table 55 on page 275) is used to assign a mask value to a character representing a subset of characters. The characters are defined in the Generic String Special Character (GSS) table, Table 56 on page 276.

Table 55. Example of translation table for generic string characters

Table	Hexadecimal Code (Mask Value)	Code Mask Offset
TTGSC	`DC X'00'`	`X'00' Any character`
	`DC X'01'`	`X'01' Invalid characters`
	`DC X'02'`	`X'02' Special characters`
	`DC X'04'`	`X'03' APL/TEXT Special`
	`DC X'08'`	`X'04' APL/TEXT Alpha`
	`DC X'10'`	`X'05' Lower alpha`
	`DC X'20'`	`X'06' Upper alpha`
	`DC X'40'`	`X'07' Numeric`
	`DC X'80'`	`X'08' User defined character set`
	`DC X'FF'`	`X'09' (RESERVED)`
	`DC X'30'`	`X'0A' Alpha`
	`DC X'7F'`	`X'0B' Nonblank`
	`DC X'FF'`	`X'0C' (RESERVED)`
	`DC X'FF'`	`X'0D' (RESERVED)`
	`DC X'FF'`	`X'0E' (RESERVED)`
	`DC X'FF'`	`X'0F' (RESERVED)`
	`DC X'40'`	`X'10' Not numeric`
	`DC X'20'`	`X'11' Not upper`
	`DC X'10'`	`X'12' Not lower`
	`DC X'30'`	`X'13' Not alpha`
	`DC X'FF'`	`X'14' (RESERVED)`
	`DC X'FF'`	`X'15' (RESERVED)`
	`DC X'FF'`	`X'16' (RESERVED)`

Translation table for generic string special characters

The translation table for generic string special characters (Table 56 on page 276) is used to assign a code of X'01' to X'16' to generic string special characters according to the generic string character (GSC) table (Table 55 on page 275) as follows:

- Numbers and letters translate to their EBCDIC hexadecimal codes.
- Other valid characters, that is, characters used to represent a character subset for Edit and Browse picture strings, are represented by the offset from the generic string character table corresponding to

the subset they represent. In Table 56 on page 276, (X'7B') has a value of ‘07’ because ‘07’ is the offset in the generic string character table for the subset of numeric characters, which the # is used to represent.

- Characters that are invalid in a generic string are represented with X'FF'.

Table 56. Example of translation table for generic string special characters

Table	Hexadecimal Code	Position
TTGSS	DC X'FFFFFFFFFFFFFFFF' DC X'FFFFFFFFFFFFFFFF' DC X'FFFFFFFFFFFFFFFF' ... DC X'FFFFFF070AFF00FF' DC X'FF81828384858687' ... DC X'E8E9FFFFFFFFFFFF' DC X'F0F1F2F3F4F5F6F7' DC X'F8F9FFFFFFFFFFFF'	(X'00' to X'07') (X'08' to X'0F') (X'10' to X'17') (X'78' to X'7F') (X'80' to X'87') (X'E8' to X'EF') (X'F0' to X'F7') (X'F8' to X'FF')

Usage notes for the GSC and GSS tables

The generic string special (GSS) table is used to determine if a character is valid in a picture string and, if so, what offset into the generic string character (GSC) table describes its subset. The mask value from the GSC table is used to determine which characters satisfy the subset. Each character in the text being scanned is represented in the generic string master (GSM) table by a hexadecimal code that indicates which subsets that character belongs to (see “Creating ISPF terminal translation tables” on page 111). The hexadecimal code in the GSM is ANDed with the mask value from the GSC. Any nonzero result is considered a match.

In Table 56 on page 276, a # (‘7B’) has a value in the GSS of ‘07’. At offset ‘07’ in the GSC (actually the eighth entry in the GSC, the first being offset ‘00’), the mask value is a X'40'. This means any character that is a member of the numeric subset, when ANDed with a X'40', will produce a nonzero result.

Translation table for uppercase characters

The translation table for uppercase characters (Table 57 on page 276) translates data as follows:

- Lowercase alphabetic characters translate to uppercase
- All other characters translate to themselves.

Table 57. Example of translation table for uppercase characters

Table	Hexadecimal Code	Position
TTUPP	DC X'0001020304050607' DC X'08090A0B0C0D0E0F' DC X'1011121314151617' ... DC X'78797A7B7C7D7E7F' DC X'80C1C2C3C4C5C6C7' ... DC X'E8E9EAEBECEDEEEF' DC X'F0F1F2F3F4F5F6F7' DC X'F8F9FAFBFCFDFEFF'	(X'00' to X'07') (X'08' to X'0F') (X'10' to X'17') (X'78' to X'7F') (X'80' to X'87') (X'E8' to X'EF') (X'F0' to X'F7') (X'F8' to X'FF')

Translation table for lowercase characters

The translation table for lowercase characters (Table 58 on page 277) translates data as follows:

- Uppercase alphabetic characters translate to lowercase
- All other characters translate to themselves.

Table 58. Example of translation table for lowercase characters

Table	Hexadecimal Code	Position
TTLOW	`DC X'0001020304050607'`	`(X'00' to X'07')`
	`DC X'08090A0B0C0D0E0F'`	`(X'08' to X'0F')`
	`DC X'1011121314151617'`	`(X'10' to X'17')`
	`...`	
	`DC X'78797A7B7C7D7E7F'`	`(X'78' to X'7F')`
	`DC X'8081828384858687'`	`(X'80' to X'87')`
	`...`	
	`DC X'A8A9EAEBECEDEEEF'`	`(X'E8' to X'EF')`
	`DC X'F0F1F2F3F4F5F6F7'`	`(X'F0' to X'F7')`
	`DC X'F8F9FAFBFCFDFEFF'`	`(X'F8' to X'FF')`

Modifying the GSM to use the user character subset

The generic string master (GSM) translation table (Table 59 on page 277) and its related tables can be modified to add an additional character subset to be used in Edit picture string processing for the FIND and CHANGE commands. The GSM table is found in the ISPF translation tables, member ISPOWNTT in ISP.SISPSAMP.

Generic string master translation table

The positions in the generic string master translation table (Table 59 on page 277) are filled in as follows:

X'00'
Blank character

X'01'
Invalid character

X'02'
Special character

X'04'
APL/TEXT special characters (only for APL and TEXT keyboards)

X'08'
APL/TEXT alphabetic characters (only for APL and TEXT keyboards)

X'10'
Lowercase alphabetic character

X'20'
Uppercase alphabetic character

X'40'
Numeric character

X'80'
User character subset.

Table 59. Example of Generic String Master translation table

Table	Hexadecimal Code	Position
TTGSM	`DC X'0101010101010101'`	`(X'00' to X'07')`
	`DC X'0101010101010101'`	`(X'08' to X'0F')`
	`DC X'0101010101010101'`	`(X'10' to X'17')`
	`...`	
	`DC X'0102020202020202'`	`(X'78' to X'7F')`
	`DC X'0110101010101010'`	`(X'80' to X'87')`
	`...`	
	`DC X'2020010101010101'`	`(X'E8' to X'EF')`
	`DC X'4040404040404040'`	`(X'F0' to X'F7')`
	`DC X'4040010101010101'`	`(X'F8' to X'FF')`

To modify the GSM table to use a user character subset, follow these steps:

1. Choose a character to represent your subset. For example, Edit uses an @ to stand for alphabetic.
2. Modify the entry in the generic string special character (GSS) table that corresponds to the character you want to use so that it has a value of X'08'. This indicates where in the generic string character (GSC) table the mask for your character is located. The GSC does not need to be changed. It is initially set for user character sets.
3. Modify the GSM entries of those characters you want to include in your special character set so the high order bit is on.

For example:

If you want to define a character set of special attribute characters consisting of hexadecimal codes X'10' through X'17', and you want to use a (" ") as the picture string identifier for them, you would:

1. Modify the entry for X'4F' in the GSS table so it has a value of X'08' as shown in Table 60 on page 278. Compare to Table 56 on page 276.

Table 60. Example of translation table for modified Generic String Special Characters

Table	Hexadecimal Code	Position
TTGSS	`DC X'FFFFFFFFFFFFFFFF'`	`(X'00' to X'07')`
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'08' to X'0F')`
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'10' to X'17')`
	`...`	
	`DC X'FFFFFF0105FFFF08'`	`(X'78' to X'7F')`
	`DC X'FFFFFFFFFFFFFFFF'`	`(X'80' to X'87')`
	`...`	
	`DC X'E8E9FFFFFFFFFFFF'`	`(X'E8' to X'EF')`
	`DC X'F0F1F2F3F4F5F6F7'`	`(X'F0' to X'F7')`
	`DC X'F8F9FFFFFFFFFFFF'`	`(X'F8' to X'FF')`

2. Modify the GSM entries for hexadecimal codes X'10' through X'17' to turn the high order bit on as shown in Table 61 on page 278. Compare to Table 59 on page 277.

Table 61. Example of modified Generic String Master translation table

Table	Hexadecimal Code	Position
TTGSM	`DC X'0101010101010101'`	`(X'00' to X'07')`
	`DC X'0101010101010101'`	`(X'08' to X'0F')`
	`DC X'8181818181818181'`	`(X'10' to X'17')`
	`...`	
	`DC X'0102020202020202'`	`(X'78' to X'7F')`
	`DC X'0110101010101010'`	`(X'80' to X'87')`
	`...`	
	`DC X'2020010101010101'`	`(X'E8' to X'EF')`
	`DC X'4040404040404040'`	`(X'F0' to X'F7')`
	`DC X'4040010101010101'`	`(X'F8' to X'FF')`

You could locate the special attribute bytes by issuing the edit command FIND P‘"’. If you do not want these bytes to be found under any other picture string, set the hexadecimal value to X'80'. These characters can be included in multiple character sets by setting the appropriate bits to on, according to the GSM table.

ISPF and PDF terminal translation table relationship

The terminal translation table relationship for ISPF and PDF is shown in Table 62 on page 279.

Table 62. ISPF terminal translation table relationship

ISPF	PDF
ISP3277 ISP3277A ISP3277K ISP3278 ISP3278A ISP3278C ISP3278K ISP3278T NEW32TBL	ISR3277 ISR3277A ISR3277K ISR3278 ISR3278A ISR3278C ISR3278K ISR3278T ISR32TBL

The delivered ISPF terminal table names start with the prefix "ISP". ISPF does not require that user-defined terminal table names begin with the prefix "ISP"; however, PDF terminal names require the "ISR" prefix. PDF searches for the load module beginning with the fourth position of the actual table name and prefixes it with "ISR".

Chapter 12. Dialog development model listings

This topic lists all of the models included with PDF. The table shows the external model name, any qualifiers, a short description, and the internal member name in the SKELS library provided with PDF. See *z/OS ISPF Edit and Edit Macros* for more information about adding or changing a model.

DM and PDF services in CLIST commands

Table 63. DM and PDF services in CLIST commands

Model Name	Description	Member Name
Display		
ADDPOP	Display pop-up window	ISREMCD5
DISPLAY	Display option	ISREMCD1
PQUERY	Get panel information	ISREMCD4
REMPOP	Remove pop-up window	ISREMCD6
SETMSG	Set message display	ISREMCD3
TBDISPL	Table display information	ISREMCD2
File tailoring		
FTCLOSE	End file tailoring	ISREMCF3
FTERASE	File tailor erase	ISREMCF4
FTINCL	File tailor include skeleton	ISREMCF2
FTOPEN	File tailor open	ISREMCF1
Library access		
DIRLIST	Displays z/OS UNIX directory list	ISREMCLD
LMCLOSE	Close a data set	ISREMCL1
LMCOMP	Compress a data set	ISREMCLU
LMCOPY	Copy a data set	ISREMCLQ
LMDDISP	Data set list	ISREMCLZ
LMDFREE	Release a data set list	ISREMCLW
LMDINIT	Establish a data set ID	ISREMCLX
LMDLIST	Obtain a list of data sets	ISREMCLV
LMERASE	Erase a data set or library	ISREMCL2
LMFREE	Release a data set	ISREMCL3
LMGET	Read a record	ISREMCL4
LMINIT	Establish a data ID	ISREMCL5
LMMADD	Add a member	ISREMCL6
LMMDEL	Delete a member	ISREMCL7

Table 63. DM and PDF services in CLIST commands (continued)

Model Name	**Description**	**Member Name**
LMMDISP	Display member list	ISREMCL0
LMMFIND	Find a member	ISREMCL8
LMMLIST	Create a member list	ISREMCL9
LMMOVE	Move a data set or member	ISREMCLP
LMMREN	Rename a member	ISREMCLA
LMMREP	Replace a member	ISREMCLB
LMMSTATS	Set member statistics	ISREMCLR
LMOPEN	Open a data set	ISREMCLC
LMPRINT	Write member to list data set	ISREMCLT
LMPUT	Write a record	ISREMCLE
LMQUERY	Provide data set information	ISREMCLF
LMRENAME	Rename a library	ISREMCLG
MEMLIST	Displays Option 3.1 member list	ISREMCLH
Miscellaneous		
BROWSE	Browse service	ISREMCM3
CONTROL	Control service	ISREMCM2
DSINFO	Returns data set information	ISREMCME
EDIT	Edit service	ISREMCM4
EDREC	Edit recovery services	ISREMCM7
GETMSG	Get message service	ISREMCM6
LIBDEF	LIBDEF service	ISREMCM8
LIST	Write data to list data set	ISREMCMA
LOG	Write message or log data set	ISREMCM5
QBASELIB	Query base library information	ISREMCMC
QLIBDEF	Query LIBDEF library information	ISREMCMD
SELECT	Select service	ISREMCM1
Table functions (general)		
TBCLOSE	Table close	ISREMCG5
TBCREATE	Table create	ISREMCG1
TBEND	Table end	ISREMCG6
TBERASE	Table erase	ISREMCG7
TBOPEN	Table open	ISREMCG2
TBQUERY	Table query	ISREMCG3
TBSAVE	Table save	ISREMCG4

Table 63. DM and PDF services in CLIST commands (continued)

Model Name	Description	Member Name
TBSTATS	Table statistics	ISREMCG8
Table functions (row)		
TBADD	Table row add	ISREMCR1
TBBOTTOM	Table row pointer to bottom	ISREMCRA
TBDELETE	Table delete	ISREMCR2
TBEXIST	Table exist	ISREMCR6
TBGET	Table get	ISREMCR3
TBMOD	Table modify	ISREMCR5
TBPUT	Table put	ISREMCR4
TBSARG	Table search parameter	ISREMCR7
TBSCAN	Table scan	ISREMCR8
TBSKIP	Table skip	ISREMCRB
TBSORT	Table sort	ISREMCRD
TBTOP	Table top	ISREMCR9
TBVCLEAR	Table variable clear	ISREMCRC
Variables		
VERASE	Variable erase	ISREMCV8
VGET	Variable get	ISREMCV1
VPUT	Variable put	ISREMCV2

DM and PDF services in COBOL programs

Table 64. DM and PDF services in COBOL programs

Model Name	Description	Member Name
Display		
ADDPOP	Display pop-up window	ISREMBD5
DISPLAY	Display option	ISREMBD1
PQUERY	Get panel information	ISREMBD4
REMPOP	Remove pop-up window	ISREMBD6
SETMSG	Set message display	ISREMBD3
TBDISPL	Table display information	ISREMBD2
File tailoring		
FTCLOSE	End file tailoring	ISREMBF3
FTERASE	File tailor erase	ISREMBF4
FTINCL	File tailor include skeleton	ISREMBF2

Table 64. DM and PDF services in COBOL programs (continued)

Model Name	Description	Member Name
FTOPEN	File tailor open	ISREMBF1
Graphics		
GRERROR	Graphics error block service	ISREMBS3
GRINIT	Graphics initialization	ISREMBS1
GRTERM	Graphics completion service	ISREMBS2
Library access		
DIRLIST	Displays z/OS UNIX directory list	ISREMBLD
LMCLOSE	Close a data set	ISREMBL1
LMCOMP	Compress a data set	ISREMBLS
LMCOPY	Copy a data set	ISREMBLQ
LMDDISP	Data set list	ISREMBLZ
LMDFREE	Release a data set list	ISREMBLW
LMDINIT	Establish a data set ID	ISREMBLU
LMDLIST	Obtain a list of data sets	ISREMBLV
LMERASE	Erase a data set or library	ISREMBL2
LMFREE	Release a data set	ISREMBL3
LMGET	Read a record	ISREMBL4
LMINIT	Establish a data ID	ISREMBL5
LMMADD	Add a member	ISREMBL6
LMMDEL	Delete a member	ISREMBL7
LMMDISP	Display member list	ISREMBLO
LMMFIND	Find a member	ISREMBL8
LMMLIST	Create a member list	ISREMBL9
LMMOVE	Move a data set or member	ISREMBLP
LMMREN	Rename a member	ISREMBLA
LMMREP	Replace a member	ISREMBLB
LMMSTATS	Set member statistics	ISREMBLR
LMOPEN	Open a data set	ISREMBLC
LMPRINT	Write member to list data set	ISREMBLT
LMPUT	Write a record	ISREMBLE
LMQUERY	Provide data set information	ISREMBLF
LMRENAME	Rename a library	ISREMBLG
MEMLIST	Displays Option 3.1 member list	ISREMBLH
Miscellaneous		

Table 64. DM and PDF services in COBOL programs (continued)

Model Name	Description	Member Name
BRIF	Browse interface service	ISREMCB4
BROWSE	Browse service (MVS)	ISREMCB1
BROWSE	Browse service (VM)	ISREMCB2
CONTROL	Control service	ISREMBM2
DSINFO	Returns data set information	ISREMBME
EDIF	Edit interface service	ISREMCE4
EDIREC	Edit recovery for EDIF	ISREMBM9
EDIT	Edit service (MVS)	ISREMCE1
EDIT	Edit service (VM)	ISREMCE2
EDREC	Edit recovery services	ISREMBM7
GETMSG	Get message service	ISREMBM6
LIBDEF	LIBDEF service	ISREMBM8
LIST	Write to list data set	ISREMBMA
LOG	Write message or log data set	ISREMBM5
QBASELIB	Query base library information	ISREMBMC
QLIBDEF	Query LIBDEF library information	ISREMBMD
SELECT	Select service	ISREMBM1
VIIF	View Interface service	ISREMCE6
Table functions (general)		
TBCLOSE	Table close	ISREMBG5
TBCREATE	Table create	ISREMBG1
TBEND	Table end	ISREMBG6
TBERASE	Table erase	ISREMBG7
TBOPEN	Table open	ISREMBG2
TBQUERY	Table query	ISREMBG3
TBSAVE	Table save	ISREMBG4
TBSTATS	Table statistics	ISREMBG8
Table functions (row)		
TBADD	Table row add	ISREMBR1
TBBOTTOM	Table row pointer to bottom	ISREMBRA
TBDELETE	Table delete	ISREMBR2
TBEXIST	Table exist	ISREMBR6
TBGET	Table get	ISREMBR3
TBMOD	Table modify	ISREMBR5

Table 64. DM and PDF services in COBOL programs (continued)

Model Name	**Description**	**Member Name**
TBPUT	Table put	ISREMBR4
TBSARG	Table search parameter	ISREMBR7
TBSCAN	Table scan	ISREMBR8
TBSKIP	Table skip	ISREMBRB
TBSORT	Table sort	ISREMBRD
TBTOP	Table top	ISREMBR9
TBVCLEAR	Table variable clear	ISREMBRC
Variables		
VCOPY	Copy variable	ISREMBV5
VDEFINE	Variable define	ISREMBV3
VDELETE	Variable delete	ISREMBV4
VERASE	Variable erase	ISREMBV8
VGET	Variable get	ISREMBV1
VMASK	Variable mask	ISREMBV9
VPUT	Variable put	ISREMBV2
VREPLACE	Variable replace	ISREMBV6
VRESET	Variable reset	ISREMBV7
Working storage		
WORKSTOR	Working storage definition	ISREMBW1

DM and PDF services in EXEC commands

Table 65. DM and PDF services in EXEC commands

Model Name	**Description**	**Member Name**
Display		
ADDPOP	Display pop-up window	ISREMED5
DISPLAY	Display option	ISREMED1
PQUERY	Get panel information	ISREMED4
REMPOP	Remove pop-up window	ISREMED6
SETMSG	Set message display	ISREMED3
TBDISPL	Table display information	ISREMED2
File tailoring		
FTCLOSE	End file tailoring	ISREMEF3
FTERASE	File tailor erase	ISREMEF4
FTINCL	File tailor include skeleton	ISREMEF2

Table 65. DM and PDF services in EXEC commands (continued)

Model Name	Description	Member Name
FTOPEN	File tailor open	ISREMEF1
Library access		
LMCLOSE	Close a data set	ISREMEL1
LMERASE	Erase a data set or library	ISREMEL2
LMFREE	Release a data set	ISREMEL3
LMGET	Read a record	ISREMEL4
LMINIT	Establish a data ID	ISREMEL5
LMMADD	Add a member	ISREMEL6
LMMDEL	Delete a member	ISREMEL7
LMMFIND	Find a member	ISREMEL8
LMMLIST	Create a member list	ISREMEL9
LMMREN	Rename a member	ISREMELA
LMMREP	Replace a member	ISREMELB
LMOPEN	Open a data set	ISREMELC
LMPUT	Write a record	ISREMELE
LMQUERY	Provide data set information	ISREMELF
LMRENAME	Rename a library	ISREMELG
LMSPEC	Specify a new ISPF library	ISREMELH
LMUNSPEC	Unspecify an ISPF library	ISREMELI
Miscellaneous		
BROWSE	Browse service	ISREMEM3
CONTROL	Control service	ISREMEM2
EDIT	Edit service	ISREMEM4
EDREC	Edit recovery services	ISREMEM7
GETMSG	Get message service	ISREMEM6
LIBDEF	LIBDEF service	ISREMEM8
LIST	Write list data set	ISREMEMA
LOG	Write message or log data set	ISREMEM5
QBASELIB	Query base library information	ISREMEMC
QLIBDEF	Query LIBDEF library information	ISREMEMD
SELECT	Select service	ISREMEM1
Table functions (general)		
TBCLOSE	Table close	ISREMEG5
TBCREATE	Table create	ISREMEG1

Table 65. DM and PDF services in EXEC commands (continued)

Model Name	Description	Member Name
TBEND	Table end	ISREMEG6
TBERASE	Table erase	ISREMEG7
TBOPEN	Table open	ISREMEG2
TBQUERY	Table query	ISREMEG3
TBSAVE	Table save	ISREMEG4
TBSTATS	Table statistics	ISREMEG8
Table functions (row)		
TBADD	Table row add	ISREMER1
TBBOTTOM	Table row pointer to bottom	ISREMERA
TBDELETE	Table delete	ISREMER2
TBEXIST	Table exist	ISREMER6
TBGET	Table get	ISREMER3
TBMOD	Table modify	ISREMER5
TBPUT	Table put	ISREMER4
TBSARG	Table search parameter	ISREMER7
TBSCAN	Table scan	ISREMER8
TBSKIP	Table skip	ISREMERB
TBSORT	Table sort	ISREMERD
TBTOP	Table top	ISREMER9
TBVCLEAR	Table variable clear	ISREMERC
Variables		
VGET	Variable get	ISREMEV1
VPUT	Variable put	ISREMEV2
VERASE	Variable erase	ISREMEV8

DM and PDF services in FORTRAN programs

Table 66. DM and PDF services in FORTRAN programs

Model Name	Description	Member Name
Display		
ADDPOP	Display pop-up window	ISREMFD5
DISPLAY	Display option	ISREMFD1
PQUERY	Get panel information	ISREMFD4
REMPOP	Remove pop-up window	ISREMFD6
SETMSG	Set message display	ISREMFD3

Table 66. DM and PDF services in FORTRAN programs (continued)

Model Name	Description	Member Name
TBDISPL	Table display information	ISREMFD2
File tailoring		
FTCLOSE	End file tailoring	ISREMFF3
FTERASE	File tailor erase	ISREMFF4
FTINCL	File tailor include skeleton	ISREMFF2
FTOPEN	File tailor open	ISREMFF1
Graphics		
GRERROR	Graphics error block service	ISREMFS3
GRINIT	Graphics initialization	ISREMFS1
GRTERM	Graphics completion service	ISREMFS2
Library access		
DIRLIST	Displays z/OS UNIX directory list	ISREMFLD
LMCLOSE	Close a data set	ISREMFL1
LMCOMP	Compress a data set	ISREMFLS
LMCOPY	Copy a data set	ISREMFLQ
LMDDISP	Data set list	ISREMFLZ
LMDFREE	Release a data set list	ISREMFLW
LMDINIT	Establish a data set ID	ISREMFLU
LMDLIST	Obtain a list of data sets	ISREMFLV
LMERASE	Erase a data set or library	ISREMFL2
LMFREE	Release a data set	ISREMFL3
LMGET	Read a record	ISREMFL4
LMINIT	Establish a data ID	ISREMFL5
LMMADD	Add a member	ISREMFL6
LMMDEL	Delete a member	ISREMFL7
LMMDISP	Display member list	ISREMFLO
LMMFIND	Find a member	ISREMFL8
LMMLIST	Create a member list	ISREMFL9
LMMOVE	Move a data set or member	ISREMFLP
LMMREN	Rename a member	ISREMFLA
LMMREP	Replace a member	ISREMFLB
LMMSTATS	Set member statistics	ISREMFLR
LMOPEN	Open a data set	ISREMFLC
LMPRINT	Write member to list data set	ISREMFLT

Table 66. DM and PDF services in FORTRAN programs (continued)

Model Name	Description	Member Name
LMPUT	Write a record	ISREMFLE
LMQUERY	Provide data set information	ISREMFLF
LMRENAME	Rename a library	ISREMFLG
MEMLIST	Displays Option 3.1 member list	ISREMFLH
Miscellaneous		
BRIF	Browse interface service	ISREMFB4
BROWSE	Browse service (MVS)	ISREMFB1
BROWSE	Browse service (VM)	ISREMFB2
CONTROL	Control service	ISREMFM2
DSINFO	Returns data set information	ISREMFME
EDIF	Edit interface service	ISREMFE4
EDIREC	Edit recovery for EDIF	ISREMFM9
EDIT	Edit service (MVS)	ISREMFE1
EDIT	Edit service (VM)	ISREMFE2
EDREC	Edit recovery services	ISREMFM7
GETMSG	Get message service	ISREMFM6
LIBDEF	LIBDEF service	ISREMFM8
LIST	Write to list data set	ISREMFMA
LOG	Write message or log data set	ISREMFM5
QBASELIB	Query base library information	ISREMFMC
QLIBDEF	Query LIBDEF library information	ISREMFMD
SELECT	Select service	ISREMFM1
VIIF	View Interface service	ISREMFE6
Table functions (general)		
TBCLOSE	Table close	ISREMFZ5
TBCREATE	Table create	ISREMFZ1
TBEND	Table end	ISREMFZ6
TBERASE	Table erase	ISREMFZ7
TBOPEN	Table open	ISREMFZ2
TBQUERY	Table query	ISREMFZ3
TBSAVE	Table save	ISREMFZ4
TBSTATS	Table statistics	ISREMFZ8
Table functions (row)		
TBADD	Table row add	ISREMFR1

Table 66. DM and PDF services in FORTRAN programs (continued)

Model Name	Description	Member Name
TBBOTTOM	Table row pointer to bottom	ISREMFRA
TBDELETE	Table delete	ISREMFR2
TBEXIST	Table exist	ISREMFR6
TBGET	Table get	ISREMFR3
TBMOD	Table modify	ISREMFR5
TBPUT	Table put	ISREMFR4
TBSARG	Table search parameter	ISREMFR7
TBSCAN	Table scan	ISREMFR8
TBSKIP	Table skip	ISREMFRB
TBSORT	Table sort	ISREMFRD
TBTOP	Table top	ISREMFR9
TBVCLEAR	Table variable clear	ISREMFRC
Variables		
VCOPY	Copy variable	ISREMFV5
VDEFINE	Variable define	ISREMFV3
VDELETE	Variable delete	ISREMFV4
VERASE	Variable erase	ISREMFV8
VGET	Variable get	ISREMFV1
VMASK	Variable mask	ISREMBV9
VPUT	Variable put	ISREMFV2
VREPLACE	Variable replace	ISREMFV6
VRESET	Variable reset	ISREMFV7

Message format

Table 67. Message format

Model Name	Description	Member Name
MSGS	Message member selection	ISREMMSG

Panel formats and statements

Table 68. Panel formats and statements

Model Name	Qualifier	Description	Member Name
Panel formats			
ACTION		Panel with action bar	ISREMMF6
ENTRY		Data entry panel	ISREMMF1

Table 68. Panel formats and statements (continued)

Model Name	**Qualifier**	**Description**	**Member Name**
HELPSCR		Help panel with scrollable area	ISREMMF8
MULTIPLE	SELECT2	Double-column selection panel	ISREMSE1
MULTIPLE	ENTRY2	Double-column entry panel	ISREMSE2
SCROLL		Panel with scrollable area	ISREMMF7
SELECTION		Choice panel	ISREMMF3
SELECTION	CUA	Choice panel	ISREMMF9
TBDISPL		Table display table	ISREMMF4
TUTORIAL		Help panel	ISREMMF5
Panel statements			
ABC		Action bar	ISREMMSE
AREA		AREA section header	ISREMMPI
AREA	DYNAMIC	Dynamic area attribute	ISREMMA1
AREA	GRAPHIC	Graphic area attribute	ISREMMA2
AREA	SCRL	Scrollable area attribute	ISREMMA3
ASSIGN	SIMPLE	Simple assignment statement	ISREMAS1
ASSIGN	TRANS	Trans assignment statement	ISREMAS2
ASSIGN	TRUNC	Trunc assignment statement	ISREMAS3
ASSIGN	TRANSTRU	Nested translate truncate statement	ISREMAS4
ASSIGN	PFKEY	Function key built-in function	ISREMAS5
ASSIGN	LVLINE	Last visible line built-in function	ISREMAS6
ATTR		Attribute section header	ISREMMS2
ATTRIB		New attribute character definition	ISREMMS3
ATTRIBA		New attrib char definition for area	ISREMMSB
BODY		Body section header	ISREMMS4
CCSID		CCSID section header	ISREMMPA
CONTROL	CURSOR	Control first cursor placement	ISREMCN1
CONTROL	HELP	Establish a tutorial panel	ISREMCN2
CONTROL	MSG	Identify message to be displayed	ISREMCN3
CONTROL	RESP	Show user response to panel	ISREMCN4
CONTROL	TRAIL	Contain remainder from TRUNC function	ISREMCN5
CONTROL	ALARM	Shows the alarm is to be sounded	ISREMCN6
CONTROL	ATTR	Override field attr by field name	ISREMCN7
CONTROL	ATTRCHAR	Override field attr by character	ISREMCN8
CONTROL	AUTOSEL	Control table display row selection	ISREMCN9

Table 68. Panel formats and statements (continued)

Model Name	Qualifier	Description	Member Name
CONTROL	CSRPOS	Shows position of cursor in field	ISREMCNA
CONTROL	CSRROW	Row where cursor is positioned	ISREMCNB
CONTROL	PFKEY	Function key pressed by user	ISREMCND
CONTROL	ZVARS	Define names of placeholder fields	ISREMCNC
CUAATTR		CUA attributes	ISREMMSJ
END		END section header	ISREMMPP
HELP		HELP section	ISREMMPM
IF		IF statement	ISREMMS6
INIT		INIT section header	ISREMMPJ
KEYLIST		Keylist specification	ISREMMSF
MODEL		Model section header	ISREMMS7
PANEXIT		Panel language exit	ISREMMSD
PDC		Action bar pull-down	ISREMMSH
PNTS		Point-and-shoot section	ISREMMPN
PROC		PROC section header	ISREMMPL
REFRESH		Retrieve variables before redisplay	ISREMMSA
REINIT		REINIT section header	ISREMMPK
SC	ATTR	TYPE(SC) attribute	ISREMMA5
VEDIT		Validate available	ISREMMSI
VERIFY	ALPHA	Alphabetic or special characters	ISREMVE1
VERIFY	ALPHAB	Alphabetic characters	ISREMVEE
VERIFY	BIT	Binary characters	ISREMVE2
VERIFY	DSNAME	TSO data set name	ISREMVE3
VERIFY	DSNAMEF	TSO data set name with filters	ISREMVEL
VERIFY	DSNAMEFM	TSO data set name with filter member name only	ISREMVEO
VERIFY	DSNAMEPQ	TSO data set name' (adds missing end parenthesis and quote)	ISREMVEx
VERIFY	DSNAMEQ	TSO data set name' (adds missing end quote)	ISREMVEJ
VERIFY	ENUM	Extended numeric	ISREMVED
VERIFY	FILEID	CMS file ID	ISREMVE4
VERIFY	IPADDR4	IP Version 4 address	ISREMVEV
VERIFY	HEX	Hexadecimal characters	ISREMVE5
VERIFY	IDATE	International date	ISREMVEP

Table 68. Panel formats and statements (continued)

Model Name	Qualifier	Description	Member Name
VERIFY	INCLUDE	Specify list of types	ISREMVEF
VERIFY	ITIME	International time	ISREMVET
VERIFY	JDATE	Julian date	ISREMVER
VERIFY	JSTD	Julian standard date	ISREMVES
VERIFY	LEN	Length of data stored in variable	ISREMVEC
VERIFY	LIST	List of valid values	ISREMVE6
VERIFY	LISTV	Specify list of values	ISREMVEG
VERIFY	LISTVX	Specify list of excluded values	ISREMVEI
VERIFY	LISTX	Specify list of excluded values	ISREMVEH
VERIFY	NAME	Data set member name	ISREMVE7
VERIFY	NAMEF	Data set member name with filters	ISREMVEM
VERIFY	NONBLANK	Verify nonblank field	ISREMVE8
VERIFY	NUM	Numeric characters	ISREMVE9
VERIFY	PICT	Mixed characters matching picture	ISREMVEA
VERIFY	PICTCN	Constants and mixed characters matching picture	ISREMVEK
VERIFY	RANGE	Numeric value within specified limits	ISREMVEB
VERIFY	STDDATE	Standard date	ISREMVEQ
VERIFY	STDTIME	Standard time	ISREMVEU
VGET		Variable get statement	ISREMMSC
VPUT		Variable put statement	ISREMMS9

DM and PDF services in PL/I programs

Table 69. DM and PDF services in PL/I programs

Model Name	Description	Member Name
Display		
ADDPOP	Display pop-up window	ISREMPD5
DISPLAY	Display option	ISREMPD1
PQUERY	Get panel information	ISREMPD4
REMPOP	Remove pop-up window	ISREMPD6
SETMSG	Set message display	ISREMPD3
TBDISPL	Table display information	ISREMPD2
File tailoring		
FTCLOSE	End file tailoring	ISREMPF3
FTERASE	File tailor erase	ISREMPF4

Table 69. DM and PDF services in PL/I programs (continued)

Model Name	Description	Member Name
FTINCL	File tailor include skeleton	ISREMPF2
FTOPEN	File tailor open	ISREMPF1
Graphics		
GRERROR	Graphics error block service	ISREMPS3
GRINIT	Graphics initialization	ISREMPS1
GRTERM	Graphics completion service	ISREMPS2
Library access		
DIRLIST	Displays z/OS UNIX directory list	ISREMPLD
LMCLOSE	Close a data set	ISREMPL1
LMCOMP	Compress a data set	ISREMPLS
LMCOPY	Copy a data set	ISREMPLQ
LMDDISP	Data set list	ISREMPLZ
LMDFREE	Release a data set list	ISREMPLW
LMDINIT	Establish a data set ID	ISREMPLU
LMDLIST	Obtain a list of data sets	ISREMPLV
LMERASE	Erase a data set or library	ISREMPL2
LMFREE	Release a data set	ISREMPL3
LMGET	Read a record	ISREMPL4
LMINIT	Establish a data ID	ISREMPL5
LMMADD	Add a member	ISREMPL6
LMMDEL	Delete a member	ISREMPL7
LMMDISP	Display member list	ISREMPLO
LMMFIND	Find a member	ISREMPL8
LMMLIST	Create a member list	ISREMPL9
LMMOVE	Move a data set or member	ISREMPLP
LMMREN	Rename a member	ISREMPLA
LMMREP	Replace a member	ISREMPLB
LMMSTATS	Set member statistics	ISREMPLR
LMOPEN	Open a data set	ISREMPLC
LMPRINT	Write member to list data set	ISREMPLT
LMPUT	Write a record	ISREMPLE
LMQUERY	Provide data set information	ISREMPLF
LMRENAME	Rename a library	ISREMPLG
MEMLIST	Displays Option 3.1 member list	ISREMPLH

Table 69. DM and PDF services in PL/I programs (continued)

Model Name	Description	Member Name
Miscellaneous		
BRIF	Browse interface service	ISREMPB4
BROWSE	Browse service (MVS)	ISREMPB1
BROWSE	Browse service (VM)	ISREMPB2
CONTROL	Control service	ISREMPM2
DSINFO	Returns data set information	ISREMPME
EDIF	Edit interface service	ISREMPE4
EDIREC	EDIF recovery service	ISREMPM9
EDIT	Edit service (MVS)	ISREMPE1
EDIT	Edit service (VM)	ISREMPE2
EDREC	Edit recovery services	ISREMPM7
GETMSG	Get message service	ISREMPM6
LIBDEF	LIBDEF service	ISREMPM8
LIST	Write to list data set	ISREMPMA
LOG	Write message or log data set	ISREMPM5
QBASELIB	Query base library information	ISREMPMC
QLIBDEF	Query LIBDEF library information	ISREMPMD
SELECT	SELECT service	ISREMPM1
VIIF	View Interface service	ISREMPE6
Table functions (general)		
TBCLOSE	Table close	ISREMPZ5
TBCREATE	Table create	ISREMPZ1
TBEND	Table end	ISREMPZ6
TBERASE	Table erase	ISREMPZ7
TBOPEN	Table open	ISREMPZ2
TBQUERY	Table query	ISREMPZ3
TBSAVE	Table save	ISREMPZ4
TBSTATS	Table statistics	ISREMPZ8
Table functions (row)		
TBADD	Table row add	ISREMPR1
TBBOTTOM	Table row pointer to bottom	ISREMPRA
TBDELETE	Table delete	ISREMPR2
TBEXIST	Table exist	ISREMPR6
TBGET	Table get	ISREMPR3

Table 69. DM and PDF services in PL/I programs (continued)

Model Name	Description	Member Name
TBMOD	Table modify	ISREMPR5
TBPUT	Table put	ISREMPR4
TBSARG	Table search parameter	ISREMPR7
TBSCAN	Table scan	ISREMPR8
TBSKIP	Table skip	ISREMPRB
TBSORT	Table sort	ISREMPRD
TBTOP	Table top	ISREMPR9
TBVCLEAR	Table variable clear	ISREMPRC
Variables		
VCOPY	Copy variable	ISREMPV5
VDEFINE	Variable define	ISREMPV3
VDELETE	Variable delete	ISREMPV4
VERASE	Variable erase	ISREMPV8
VGET	Variable get	ISREMPV1
VMASK	Variable mask	ISREMPV9
VPUT	Variable put	ISREMPV2
VREPLACE	Variable replace	ISREMPV6
VRESET	Variable reset	ISREMPV7

File tailoring control statements

Table 70. File tailoring control statements

Model Name	Description	Member Name
BLANK	Create blank lines	ISREMSK8
CM	Define comment statement	ISREMSK1
DEFAULT	Change control character defaults	ISREMSK2
DOT	Define DO group for table row	ISREMSK3
IM	Imbed specified data set skeleton	ISREMSK4
SEL	Conditional processing definition	ISREMSK5
SET	Set dialog variable value	ISREMSK6
TB	Set tab stop position	ISREMSK7

DM and PDF services in Pascal programs

Table 71. DM and PDF services in Pascal programs

Model Name	Description	Member Name
Display		
ADDPOP	Display pop-up window	ISREMQD5
DISPLAY	Display option	ISREMQD1
PQUERY	Get panel information	ISREMQD4
REMPOP	Remove pop-up window	ISREMQD6
SETMSG	Set message display	ISREMQD3
TBDISPL	Table display information	ISREMQD2
File tailoring		
FTCLOSE	End file tailoring	ISREMQF3
FTERASE	File tailor erase	ISREMQF4
FTINCL	File tailor include skeleton	ISREMQF2
FTOPEN	File tailor open	ISREMQF1
Graphics		
GRERROR	Graphics error block service	ISREMQS3
GRINIT	Graphics initialization	ISREMQS1
GRTERM	Graphics completion service	ISREMQS2
Library access		
DIRLIST	Displays z/OS UNIX directory list	ISREMQLD
LMCLOSE	Close a data set	ISREMQL1
LMCOMP	Compress a data set	ISREMQLS
LMCOPY	Copy a data set	ISREMQLQ
LMDDISP	Data set list	ISREMQLZ
LMDFREE	Release a data set list	ISREMQLW
LMDINIT	Establish a data set ID	ISREMQLU
LMDLIST	Obtain a list of data sets	ISREMQLV
LMERASE	Erase a data set or library	ISREMQL2
LMFREE	Release a data set	ISREMQL3
LMGET	Read a record	ISREMQL4
LMINIT	Establish a data ID	ISREMQL5
LMMADD	Add a member	ISREMQL6
LMMDEL	Delete a member	ISREMQL7
LMMDISP	Display member list	ISREMQLO
LMMFIND	Find a member	ISREMQL8

Table 71. DM and PDF services in Pascal programs (continued)

Model Name	**Description**	**Member Name**
LMMLIST	Create a member list	ISREMQL9
LMMOVE	Move a data set or member	ISREMQLP
LMMREN	Rename a member	ISREMQLA
LMMREP	Replace a member	ISREMQLB
LMMSTATS	Set member statistics	ISREMQLR
LMOPEN	Open a data set	ISREMQLC
LMPRINT	Write member to list data set	ISREMQLT
LMPUT	Write a record	ISREMQLE
LMQUERY	Provide data set information	ISREMQLF
LMRENAME	Rename a library	ISREMQLG
MEMLIST	Displays Option 3.1 member list	ISREMQLH
Miscellaneous		
BRIF	Browse interface service	ISREMQB4
BROWSE	Browse service (MVS)	ISREMQB1
BROWSE	Browse service (VM)	ISREMQB2
CONTROL	Control service	ISREMQM2
DSINFO	Returns data set information	ISREMQME
EDIF	Edit interface service	ISREMQE4
EDIREC	Edit recovery for EDIF	ISREMQM9
EDIT	Edit service (MVS)	ISREMQE1
EDIT	Edit service (VM)	ISREMQE2
EDREC	Edit recovery services	ISREMQM7
GETMSG	Get message service	ISREMQM6
LIBDEF	LIBDEF service	ISREMQM8
LIST	Write to list data set	ISREMQMA
LOG	Write message or log data set	ISREMQM5
QBASELIB	Query base library information	ISREMQMC
QLIBDEF	Query LIBDEF library information	ISREMQMD
SELECT	Select service	ISREMQM1
VIIF	View Interface service	ISREMQE6
Pascal definitions		
PASDEFS	Pascal Definitions	ISREMQPD
Table functions (general)		
TBCLOSE	Table close	ISREMQZ5

Table 71. DM and PDF services in Pascal programs (continued)

Model Name	**Description**	**Member Name**
TBCREATE	Table create	ISREMQZ1
TBEND	Table end	ISREMQZ6
TBERASE	Table erase	ISREMQZ7
TBOPEN	Table open	ISREMQZ2
TBQUERY	Table query	ISREMQZ3
TBSAVE	Table save	ISREMQZ4
TBSTATS	Table statistics	ISREMQZ8
Table functions (row)		
TBADD	Table row add	ISREMQR1
TBBOTTOM	Table row pointer to bottom	ISREMQRA
TBDELETE	Table delete	ISREMQR2
TBEXIST	Table exist	ISREMQR6
TBGET	Table get	ISREMQR3
TBMOD	Table modify	ISREMQR5
TBPUT	Table put	ISREMQR4
TBSARG	Table search parameter	ISREMQR7
TBSCAN	Table scan	ISREMQR8
TBSKIP	Table skip	ISREMQRB
TBSORT	Table sort	ISREMQRD
TBTOP	Table top	ISREMQR9
TBVCLEAR	Table variable clear	ISREMQRC
Variables		
VCOPY	Copy variable	ISREMQV5
VDEFINE	Variable define	ISREMQV3
VDELETE	Variable delete	ISREMQV4
VERASE	Variable erase	ISREMQV8
VGET	Variable get	ISREMQV1
VMASK	Variable mask	ISREMQV9
VPUT	Variable put	ISREMQV2
VREPLACE	Variable replace	ISREMQV6
VRESET	Variable reset	ISREMQV7

DM and PDF services in TSO/REXX commands

Table 72. DM and PDF services in TSO/REXX commands

Model Name	Description	Member Name
Display		
ADDPOP	Display pop-up window	ISREMRD5
DISPLAY	Display option	ISREMRD1
PQUERY	Get panel information	ISREMRD4
REMPOP	Remove pop-up window	ISREMRD6
SETMSG	Set message display	ISREMRD3
TBDISPL	Table display information	ISREMRD2
File tailoring		
FTCLOSE	End file tailoring	ISREMRF3
FTERASE	File tailor erase	ISREMRF4
FTINCL	File tailor include skeleton	ISREMRF2
FTOPEN	File tailor open	ISREMRF1
Library access		
DIRLIST	Displays z/OS UNIX directory list	ISREMRLD
LMCLOSE	Close a data set	ISREMRL1
LMCOMP	Compress a data set	ISREMRLU
LMCOPY	Copy a data set	ISREMRLQ
LMDDISP	Data set list	ISREMRLZ
LMDFREE	Release a data set list	ISREMRLW
LMDINIT	Establish a data set ID	ISREMRLX
LMDLIST	Obtain a list of data sets	ISREMRLV
LMERASE	Erase a data set or library	ISREMRL2
LMFREE	Release a data set	ISREMRL3
LMGET	Read a record	ISREMRL4
LMINIT	Establish a data ID	ISREMRL5
LMMADD	Add a member	ISREMRL6
LMMDEL	Delete a member	ISREMRL7
LMMDISP	Display member list	ISREMRLO
LMMFIND	Find a member	ISREMRL8
LMMLIST	Create a member list	ISREMRL9
LMMOVE	Move a data set or member	ISREMRLP
LMMREN	Rename a member	ISREMRLA
LMMREP	Replace a member	ISREMRLB

Table 72. DM and PDF services in TSO/REXX commands (continued)

Model Name	Description	Member Name
LMMSTATS	Set member statistics	ISREMRLR
LMOPEN	Open a data set	ISREMRLC
LMPRINT	Write member to list data set	ISREMRLT
LMPUT	Write a record	ISREMRLE
LMQUERY	Provide data set information	ISREMRLF
LMRENAME	Rename a library	ISREMRLG
MEMLIST	Displays Option 3.1 member list	ISREMRLH
Miscellaneous		
BROWSE	Browse service	ISREMRM3
CONTROL	Control service	ISREMRM2
DSINFO	Returns data set information	ISREMRME
EDIT	Edit service	ISREMRM4
EDREC	Edit recovery services	ISREMRM7
GETMSG	Get message service	ISREMRM6
LIBDEF	LIBDEF service	ISREMRM8
LIST	Write to list data set	ISREMRMA
LOG	Write message or log data set	ISREMRM5
QBASELIB	Query base library information	ISREMRMC
QLIBDEF	Query LIBDEF library information	ISREMRMD
SELECT	SELECT service	ISREMRM1
Table functions (general)		
TBCLOSE	Table close	ISREMRG5
TBCREATE	Table create	ISREMRG1
TBEND	Table end	ISREMRG6
TBERASE	Table erase	ISREMRG7
TBOPEN	Table open	ISREMRG2
TBQUERY	Table query	ISREMRG3
TBSAVE	Table save	ISREMRG4
TBSTATS	Table statistics	ISREMRG8
Table functions (row)		
TBADD	Table row add	ISREMRR1
TBBOTTOM	Table row pointer to bottom	ISREMRRA
TBDELETE	Table delete	ISREMRR2
TBEXIST	Table exist	ISREMRR6

Table 72. DM and PDF services in TSO/REXX commands (continued)

Model Name	Description	Member Name
TBGET	Table get	ISREMRR3
TBMOD	Table modify	ISREMRR5
TBPUT	Table put	ISREMRR4
TBSARG	Table search parameter	ISREMRR7
TBSCAN	Table scan	ISREMRR8
TBSKIP	Table skip	ISREMRRB
TBSORT	Table sort	ISREMRRD
TBTOP	Table top	ISREMRR9
TBVCLEAR	Table variable clear	ISREMRRC
Variables		
VERASE	Variable erase	ISREMRV8
VGET	Variable get	ISREMRV1
VPUT	Variable put	ISREMRV2

SCLM architecture definition formats

Table 73. SCLM architecture definition formats

Model Name	Description	Member Name
Architecture definition formats		
CC	Compilation Control	ISREMHAC
LEC	Linkage Editor Control	ISREMHAL
HL	High Level Definition	ISREMHAH
GENERIC	Special processing control	ISREMHAG

SCLM project definition macros and templates

Table 74. SCLM project definition macros and templates

Model Name	Description	Member Name
Macros		
FLMABEG	Define project name	ISREMGAB
FLMAEND	End project definition	ISREMGAE
FLMAGRP	Define authcode group	ISREMGAG
FLMALLOC	Define ddname	ISREMGAL
FLMALTC	Define alternate control set	ISREMGAC
FLMATVER	Enable audit tracking and versioning	ISREMGAV
FLMCMPLB	Name compool library	ISREMGCM

Table 74. SCLM project definition macros and templates (continued)

Model Name	Description	Member Name
FLMCNTRL	Specify project controls	ISREMGCN
FLMCPYLB	Name data set for allocation	ISREMGCP
FLMEXLIB	Define external library name	ISREMGEX
FLMGROUP	Define group of libraries	ISREMGGR
FLMLANGL	Define a language	ISREMGLA
FLMSYSLB	Name system library	ISREMGSY
FLMTRNSL	Define language translator	ISREMGTR
FLMTYPE	Define a library type	ISREMGTY
Templates		
PROJDEF	Project definition	ISREMGT1
LANGUAGE	Language definition	ISREMGT2

ISPF Dialog Tag Language models

Table 75. Dialog Tag Language models

Model Name	Description	Member Name
Panel formats		
ACTION BAR	Action bar panel	ISREMDP2
ENTRY	Data entry panel	ISREMDP1
SELECTION	Choice panel	ISREMDP3
TABLE DISPLAY	Scrollable list	ISREMDP5
TUTORIAL	Help/Tutorial panel	ISREMDP4
Command table format		
COMMAND TABLE	Command table application	ISREMDC1

DM and PDF services in C/370 programs

Table 76. DM and PDF services in C/370 programs

Model Name	Description	Member Name
Display		
ADDPOP	Display pop-up window	ISREMWD5
DISPLAY	Display option	ISREMWD1
PQUERY	Get panel information	ISREMWD4
REMPOP	Remove pop-up window	ISREMWD6
SETMSG	Set message display	ISREMWD3
TBDISPL	Table display information	ISREMWD2

Table 76. DM and PDF services in C/370 programs (continued)

Model Name	Description	Member Name
File tailoring		
FTCLOSE	End file tailoring	ISREMWF3
FTERASE	File tailor erase	ISREMWF4
FTINCL	File tailor include skeleton	ISREMWF2
FTOPEN	File tailor open	ISREMWF1
Graphics		
GRERROR	Graphics error block service	ISREMWS3
GRINIT	Graphics initialization	ISREMWS1
GRTERM	Graphics completion service	ISREMWS2
Library access		
DIRLIST	Displays z/OS UNIX directory list	ISREMWLD
LMCLOSE	Close a data set	ISREMWL1
LMCOMP	Compress a data set	ISREMWL0
LMCOPY	Copy a data set	ISREMWLY
LMDDISP	Data set list	ISREMWLZ
LMDFREE	Release a data set list	ISREMWLW
LMDINIT	Establish a data set ID	ISREMWLU
LMDLIST	Obtain a list of data sets	ISREMWLV
LMERASE	Erase a data set or library	ISREMWL2
LMFREE	Release a data set	ISREMWL3
LMGET	Read a record	ISREMWL4
LMINIT	Establish a data ID	ISREMWL5
LMMADD	Add a member	ISREMWL6
LMMDEL	Delete a member	ISREMWL7
LMMDISP	Display member list	ISREMWLL
LMMFIND	Find a member	ISREMWL8
LMMLIST	Create a member list	ISREMWL9
LMMOVE	Move a data set or member	ISREMWLM
LMMREN	Rename a member	ISREMWL0
LMMREP	Replace a member	ISREMWLB
LMMSTATS	Set member statistics	ISREMWLP
LMOPEN	Open a data set	ISREMWLC
LMPRINT	Write member to list data set	ISREMWLQ
LMPUT	Write a record	ISREMWLE

Table 76. DM and PDF services in C/370 programs (continued)

Model Name	Description	Member Name
LMQUERY	Provide data set information	ISREMWLF
LMRENAME	Rename a library	ISREMWLG
MEMLIST	Displays Option 3.1 member list	ISREMWLH
Miscellaneous		
BRIF	Browse interface service	ISREMWM9
BROWSE	Browse service	ISREMWM3
CONTROL	Control service	ISREMWM2
DSINFO	Returns data set information	ISREMWME
EDIF	Edit interface service	ISREMWMA
EDIREC	Edit recovery for EDIF	ISREMWMB
EDIT	Edit service	ISREMWM4
EDREC	Edit recovery services	ISREMWM7
GETMSG	Get message service	ISREMWM6
LIBDEF	LIBDEF service	ISREMWM8
LIST	Write to list data set	ISREMWMC
LOG	Write message or log data set	ISREMWM5
QBASELIB	Query base library information	ISREMWQ2
QLIBDEF	Query LIBDEF library information	ISREMWQ1
SELECT	Select service	ISREMWM1
VIIF	View Interface service	ISREMWMF
Table functions (general)		
TBCLOSE	Table close	ISREMWG5
TBCREATE	Table create	ISREMWG1
TBEND	Table end	ISREMWG6
TBERASE	Table erase	ISREMWG7
TBOPEN	Table open	ISREMWG2
TBQUERY	Table query	ISREMWG3
TBSAVE	Table save	ISREMWG4
TBSTATS	Table statistics	ISREMWG8
Table functions (row)		
TBADD	Table row add	ISREMWR1
TBBOTTOM	Table row pointer to bottom	ISREMWRA
TBDELETE	Table delete	ISREMWR2
TBEXIST	Table exist	ISREMWR6

Table 76. DM and PDF services in C/370 programs (continued)

Model Name	**Description**	**Member Name**
TBGET	Table get	ISREMWR3
TBMOD	Table modify	ISREMWR5
TBPUT	Table put	ISREMWR4
TBSARG	Table search parameter	ISREMWR7
TBSCAN	Table scan	ISREMWR8
TBSKIP	Table skip	ISREMWRB
TBSORT	Table sort	ISREMWRD
TBTOP	Table top	ISREMWR9
TBVCLEAR	Table variable clear	ISREMWRC
Variables		
VCOPY	Copy variable	ISREMWV5
VDEFINE	Variable define	ISREMWV3
VDELETE	Variable delete	ISREMWV4
VERASE	Variable erase	ISREMWV8
VGET	Variable get	ISREMWV1
VMASK	Variable mask	ISREMWV9
VPUT	Variable put	ISREMWV2
VREPLACE	Variable replace	ISREMWV6
VRESET	Variable reset	ISREMWV7

Dialog Tag Language models

Table 77. Dialog Tag Language models

Model Name	**Description**	**Member Name**
AB	Action bar	ISREMDAB
AB (example)	Action bar with pull-down choice	ISREMDAG
ABC	Action bar choice	ISREMDA2
ACTION	Action	ISREMDAC
AREA	Area	ISREMDAR
ASSIGNI	Assignment List Item	ISREMDAI
ASSIGNL	Assignment list	ISREMDAL
ATTENTION	Attention	ISREMDAN
ATTR	Attribute	ISREMDAT
BOTINST	Bottom instruction	ISREMDBI
CAUTION	Caution	ISREMDCU

Table 77. Dialog Tag Language models (continued)

Model Name	Description	Member Name
CHDIV	Choice divider	ISREMDCD
AB	Action bar	ISREMDAB
AB	Action bar	ISREMDAB
CHECKI	Validity Check Item	ISREMDCI
CHECKL	Check validity list	ISREMDCL
CHOFLD	Choice data field	ISREMDCF
CHOICE	Selection choice	ISREMDCH
CMD	Command definition	ISREMDCM
CMDACT	Command action	ISREMDCC
CMDAREA	Command area	ISREMDCA
CMDTBL	Command table	ISREMDCT
CMDTBL example	Command table example	ISREMDC1
COMMENT	Comment	ISREMDCN
COMPOPT	Compiler options	ISREMDCO
COPYR	Copyright	ISREMDCR
DA	Dynamic Area	ISREMDDA
DD	Definition description	ISREMDDD
DDHD	Definition description header	ISREMDD1
DIVIDER	Area divider	ISREMDDI
DL	Definition list	ISREMDDL
DOCTYPE	Document type	ISREMDTP
DT	Definition term	ISREMDDT
DTACOL	Data column	ISREMDDC
DTAFLD	Data field	ISREMDDF
DTAFLDD	Data field description	ISREMDDX
DTHD	Definition term header	ISREMDD2
ENTITY	Entity	ISREMDEN
FIG	Figure	ISREMDFI
FIGCAP	Figure caption	ISREMDFC
GA	Graphic area	ISREMDGA
GRPHDR	Group header	ISREMDGH
HELP	Help panel	ISREMDHE
HELPDEF	Help default	ISREMDHD
HP	Highlighted phrase	ISREMDHP

Table 77. Dialog Tag Language models (continued)

Model Name	Description	Member Name
H1	Heading 1	ISREMDH1
Hn	Heading (H2, H3, H4)	ISREMDHN
INFO	Information region	ISREMDIN
KEYI	Key item	ISREMDKI
KEYL	Key list	ISREMDKL
LI	List item	ISREMDLI
LINES	Lines	ISREMDLN
LIT	Literal	ISREMDLT
LP	List part	ISREMDLP
LSTCOL	List column	ISREMDLC
LSTFLD	List field	ISREMDLF
LSTGRP	List group	ISREMDLG
LSTVAR	List variable	ISREMDLV
M	Mnemonic	ISREMDMN
MSG	Message	ISREMDMS
MSGMBR	Message member	ISREMDMM
NOTE	Note	ISREMDNO
NOTEL	Note list	ISREMDNL
NT	Note	ISREMDNT
OL	Ordered list	ISREMDOL
Panel example	Panel example	ISREMDP1
Panel example	Panel example	ISREMDP2
Panel example	Panel example	ISREMDP3
Panel example	Panel example	ISREMDP4
Panel example	Panel example	ISREMDP5
PANDEF	Panel default	ISREMDPB
PANEL	Panel	ISREMDPA
PARML	Parameter list	ISREMDPL
PD	Parameter description	ISREMDPD
PDC	Pull-down choice	ISREMDA3
PG	Paragraph	ISREMDPG
PNLINST	Panel instruction	ISREMDPI
PS	Point-and-shoot	ISREMDPS
PT	Parameter term	ISREMDPT

Table 77. Dialog Tag Language models (continued)

Model Name	Description	Member Name
Region	Region	ISREMDRE
RP	Reference phrase	ISREMDRP
SF	Selection field	ISREMDSF
SL	Simple list	ISREMDSL
SOURCE	Source	ISREMDSO
T	Truncation	ISREMDTR
TOPINST	Top instruction	ISREMDTI
UL	Unordered list	ISREMDUL
VARCLASS	Variable class	ISREMDVC
VARDCL	Variable declaration	ISREMDVD
VARLIST	Variable list	ISREMDVL
VARSUB	Variable substitution	ISREMDVS
WARNING	Warning	ISREMDWA
XLATI	Translate item	ISREMDXI
XLATL	Translate list	ISREMDXL
XMP	Example	ISREMDXM

Chapter 13. Programming interface macros for customers

This topic describes the macros ISPF and SCLM provide as programming interfaces.

Attention: Do not use any ISPF macros other than those described in this topic as programming interfaces.

DM component general-use programming macros

```
ISPCCSID   ISPMXDEF
ISPMDAD    ISPMXED
ISPMEPT    ISPMXEND
ISPMTCM    ISPMXLST
ISPMXDD    ISPDSTAT
```

See Chapter 5, “Customizing DM,” on page 105 for further information about DM component macros.

SCLM general-use programming macros

```
FLMABEG    FLMGROUP
FLMAEND    FLMINCLS
FLMAGRP    FLMLANGL
FLMALLOC   FLMSYSLB
FLMALTC    FLMTRNSL
FLMATVER   FLMTSEXT
FLMCNTRL   FLMTYPE
FLMCPYLB
```

See the SCLM Macros chapter in *z/OS ISPF Software Configuration and Library Manager Guide and Reference* for more information.

Chapter 14. ISPF data set descriptions

These figures list the target and distribution libraries (data sets) used by ISPF, and their contents.

Table 78. Target data set descriptions

Library	Description
SISPALIB	APL2 workspace library
SISPCLIB	CLIST library
SISPEXEC	REXX exec library
SISPGENP	Uppercase English Language-specific panel source library
SISPGENU	English Language-specific panel source library
SISPGJPN	Japanese Language-specific panel source library
SISPGMLI	Non-language-specific panel source library
SISPHELP	Help library
SISPLOAD	Linklist load library
SISPLPA	LPA load library
SISPMACS	ISPF Macro library
SISPMENP	Uppercase English Language-specific message library
SISPMENU	English Language-specific message library
SISPMJPN	Japanese Language-specific message library
SISPPENP	Uppercase English Language-specific panel library
SISPPENU	English Language-specific panel library
SISPPJPN	Japanese Language-specific panel library
SISPSAMP	Sample library
SISPSENP	Uppercase English Language-specific skeleton library
SISPSENU	English Language-specific skeleton library
SISPSJPN	Japanese Language-specific skeleton library
SISPSLIB	Non-language-specific skeleton library
SISPTENP	Uppercase English Language-specific table library
SISPTENU	English Language-specific table library
SISPTJPN	Japanese Language-specific table library

Table 79. Distribution data set descriptions

Library	Description
AISPALIB	APL2 workspace library
AISPCLIB	CLIST library
AISPEXEC	REXX exec library

Table 79. Distribution data set descriptions (continued)

Library	Description
AISPGENP	Uppercase English Language-specific panel source library
AISPGENU	English Language-specific panel source library
AISPGJPN	Japanese Language-specific panel source library
AISPGMLI	Non-language-specific panel source library
AISPHELP	Help library
AISPMACS	ISPF Macro library
AISPMENP	Uppercase English Language-specific message library
AISPMENU	English Language-specific message library
AISPMJPN	Japanese Language-specific message library
AISPMOD1	Executable modules library
AISPPENP	Uppercase English Language-specific panel library
AISPPENU	English Language-specific panel library
AISPPJPN	Japanese Language-specific panel library
AISPSAMP	Sample library
AISPSENP	Uppercase English Language-specific skeleton library
AISPSENU	English Language-specific skeleton library
AISPSJPN	Japanese Language-specific skeleton library
AISPSLIB	Non-language-specific skeleton library
AISPTENP	Uppercase English Language-specific table library
AISPTENU	English Language-specific table library
AISPTJPN	Japanese Language-specific table library

Appendix A. ISPF enqueue processing for data integrity

Note: The enqueue processing performed by ISPF as described in this topic does not apply (that is, ISPF takes no action) when the data set is being accessed with the VSAM utility.

This topic describes the enqueue processing ISPF performs to maintain the integrity of data sets, PDS members, member queues, and the SCLM VSAM database.

RESERVE processing for partitioned data set extended

ISPF uses RESERVE to protect data sets residing on a shared volume. However, PDSEs are designed to be shared in a sysplex, so Configuration Table keyword PDSE_RESERVE_PROCESSING can be used to disable the use of RESERVE.

Assuming a PDSE is being processed and ISPF services are used to access the PDSE then:

PDSE_RESERVE_PROCESSING=ON (default)
RESERVE issued prior to STOW and released afterwards. LMOPEN with ENQ(SHRW) issues a RESERVE at LMOPEN and OPTION(OUTPUT). If batch job issues RESERVE before an interactive user, then the interactive user waits on SAVE until the batch dialog releases the RESERVE. Another batch job will also wait. See the description of LMCLOSE in *z/OS ISPF Services Guide*.

PDSE_RESERVE_PROCESSING=OFF
No RESERVE is issued. DFSMS preserves the integrity of the PDSE. Interactive users and batch jobs can access the PDSE simultaneously. ISPF dialogs continue to use Member name enqueue to serialize access to individual members. In environments where there may be more than one Configuration Table, the dialog that issues a RESERVE will only wait for another dialog that also issues RESERVE. See chapter 3.8.9.3.3 "Choosing Volumes for PDSEs in a Sysplex" in *z/OS DFSMS Using Data Sets*.

Serializing with non-ISPF TSO and BATCH

ISPF relies on MVS allocation to serialize access to resources with concurrent batch or non-ISPF TSO users. ISPF uses dynamic allocation and allocates partitioned data sets with DISP=SHR before any ISPF generated ENQUEUE. To ensure integrity when batch or TSO users are not using ISPF services, if you are updating a data set, you must allocate the data set with DISP=OLD.

Note: ISPF's allocation with DISP=SHR causes MVS allocation to issue a shared ENQUEUE on Qname SYSDSN as follows:

```
    ENQ SYSDSN,dsname,S,44,SYSTEM
```

Non-ISPF TSO and Batch allocation with DISP=OLD causes MVS allocation to issue an exclusive ENQUEUE on Qname SYSDSN as follows:

```
    ENQ SYSDSN,dsname,E,44,SYSTEM
```

ISPF also issues ENQ, DEQ, and RESERVE macro instructions to serialize access to resources among multiple ISPF users. It is possible for the LMMOVE service or option 3.3 Move function to be used such that each of 2 users is holding a RESERVE (therefore an exclusive ENQ is held) that is being waited on by the other user. This occurs when the user 1 is moving from data set A to data set B, while at the same time, user 2 is moving from data set B to data set A. Each user holds a reserve on the output data set for the move and is requesting a reserve on the input for purposes of deleting the moved member. If this situation occurs, cancel one user off TSO, log back on, and reissue the move request.

ISPF data set integrity enqueue

For sequential and PDS data sets, ISPF does a reserve to protect an entire data set on DASD that you are updating through edit, utilities (such as move, copy, delete, and rename), or services (such as TBSAVE, FTCLOSE, FTERASE, EDIT, LMCOPY, LMMOVE, and LMOPEN depending on whether the option is INPUT or OUTPUT). For PDSE data sets, whether the reserve is done depends on the configuration table setting as discussed in "RESERVE processing for partitioned data set extended" on page 315.

ISPF issues this macro to protect the entire partitioned data set:

```
RESERVE SPFEDIT,dsname,E,44,SYSTEMS
```

When writing to a data set on a DASD (COPY/MOVE) that has a RECFM of "U", ISPF serializes with the linkage editor using this macro to protect the entire partitioned data set:

On a shared volume:

```
     RESERVE SYSIEWLP,dsname,E,44,SYSTEMS
```

On a volume that is not shared:

```
     ENQ SYSIEWLP,dsname,E,44,SYSTEM
```

Member name enqueue

To restrict concurrent use of a member of a partitioned data set while still allowing ISPF users to concurrently use different members of the same data set (such as through EDIT, Table Processing, File Tailoring, and library access services), ISPF issues this ENQ macro for the member:

```
      ENQ SPFEDIT,rname,E,52,SYSTEMS
```

where

rname
: the data set name, length of 44, padded with blanks, followed by the member name, length of 8, padded with blanks

Note: When a member or a member generation in a PDSE version 2 data set that is configured for member generations is edited, enqueue processing restricts access to the member and all generations of the member.

z/OS UNIX file Enqueue

To restrict concurrent use of a z/OS UNIX file, ISPF edit issues this ENQ macro for the file:

```
ENQ SPFEDIT,rname,E,12,SYSTEMS  (z/OS UNIX in sysplex; OextSysplexActv = '1'b)
```

or

```
ENQ SPFEDIT,rname,E,12,SYSTEM   (z/OS UNIX not in sysplex)
```

where

rname

Is 12 bytes in length, made up of three 4-byte values:

- A binary integer, which is the inode number of the file.
- A binary integer, which is the device number of the file.
- Four flag bytes (0-3).

  Bit 2 (of bits 0-7) of byte 3 is the Sysplex indicator. It can have one of these values:

1
: z/OS UNIX is in a sysplex (OextSysplexActv is set ON).

0
: z/OS UNIX is not in a sysplex (OextSysplexActv is set OFF).

This enqueue is compatible with the enqueue issued by the z/OS UNIX OEDIT command.

SCLM VSAM enqueue

To restrict the concurrent use of the SCLM VSAM data base, ISPF issues this ENQ macro for the VSAM data set:

```
    ENQ SLMVSAM,vsam-dsn,E,44,SYSTEMS
```

where

vsam-dsn
: The VSAM data set name, padded to 44 characters with blanks.

Appendix B. Accessibility

Accessible publications for this product are offered through IBM Documentation for z/OS (www.ibm.com/docs/en/zos).

If you experience difficulty with the accessibility of any z/OS documentation see How to Send Feedback to IBM to leave documentation feedback.

Notices

This information was developed for products and services that are offered in the USA or elsewhere.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
United States of America

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

This information could include missing, incorrect, or broken hyperlinks. Hyperlinks are maintained in only the HTML plug-in output for IBM Documentation. Use of hyperlinks in other output formats of this information is at your own risk.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Site Counsel
2455 South Road

Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or

reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user's name, email address, phone number, or other personally identifiable information for purposes of enhanced user usability and single sign-on configuration. These cookies can be disabled, but disabling them will also eliminate the functionality they enable.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at ibm.com®/privacy and IBM's Online Privacy Statement at ibm.com/privacy/details in the section entitled "Cookies, Web Beacons and Other Technologies," and the "IBM Software Products and Software-as-a-Service Privacy Statement" at ibm.com/software/info/product-privacy.

Policy for unsupported hardware

Various z/OS elements, such as DFSMSdfp, JES2, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

Minimum supported hardware

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: IBM Lifecycle Support for z/OS (www.ibm.com/software/support/systemsz/lifecycle)
- For information about currently-supported IBM hardware, contact your IBM representative.

Programming Interface Information

This publication primarily documents information that is NOT intended to be used as Programming Interfaces of ISPF.

This publication also documents intended Programming Interfaces that allow the customer to write programs to obtain the services of ISPF. This information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

```
+---------------------Programming Interface information----------------------+

+------------------End of Programming Interface information------------------+
```

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at Copyright and Trademark information (www.ibm.com/legal/copytrade.shtml).

Index

Product Number: 5655-ZOS

GC19-3623-70