

Developing Dialog Manager Applications in z/OS

The following terms that may appear in these course materials are trademarks or registered trademarks:

Trademarks of the International Business Machines Corporation:

AIX, BookManager, CICS, DB2, DRDA, DS8000, ESCON, FICON, HiperSockets, IBM, ibm.com, IMS, Language Environment, MQSeries, MVS, NetView, OS/400, POWER7, PR/SM, Processor Resource / Systems Manager, OS/390, OS/400, Parallel Sysplex, QMF, RACF, Redbooks, RMF, RS/6000, SOMobjects, S/390, System z, System z9, System z10, VisualAge, VTAM, WebSphere, z/OS, z/VM, z/VSE, z/Architecture, zEnterprise, zSeries, z9, z10

Trademarks of Microsoft Corp.: Microsoft, Windows

Trademarks of Micro Focus Corp.: Micro Focus

Trademark of American National Standards Institute: ANSI

Trademarks of America Online, Inc.: America Online, AOL

Trademarks of Quercus Systems: Personal REXX, REXXTERM

Trademark of Chicago-Soft, Ltd: MVS/QuickRef

Trademark of Phoenix Software International: (E)JES

Trademark of Triangle Systems: IOF

Trademark of Syncsort Corp.: SyncSort

Trademark of CA: Endeavor

Trademark of Serena Software International: ChangeMan

Registered Trademarks of Institute of Electrical and Electronic Engineers: IEEE, POSIX

Registered Trademarks of Corel Corporation: Corel, CorelDRAW, Corel VENTURA

Registered Trademark of Oracle Corporation: Oracle

Registered Trademark of The Open Group: UNIX

Trademarks of Sun Microsystems, Inc.: Java, EmbeddedJava, Enterprise JavaBeans, EJB, Java Naming and Directory Interface, JavaBeans, JavaOS, JavaScript, JavaServer, JavaServerPages, JSP, JDBC, JDK, JVM, J2EE, Sun Microsystems, 100% Pure Java

Registered Trademark of Linus Torvalds: LINUX

Registered Trademark of Unicode, Inc.: Unicode

Trademarks held on behalf of World Wide Web Consortium: W3C, XHTML, XSL, WebFonts

Trademark of Object Management Group: CORBA

Trademarks of Apple Computer: QuickTime, Safari

Trademarks of Adobe Systems, Inc.: Macromedia, PDF, Shockwave, Flash

Trademark of The Eclipse Foundation: Eclipse

Developing Dialog Manager Applications in z/OS - Course Objectives

Upon successful completion of this course, the student, with the aid of the appropriate reference materials, should be able to:

1. Design and write applications using Dialog Manager services for the TSO environment, using REXX or CLIST as the programming language
2. Design panels and use panel language to display, accept, and process data placed in dialog variables; preprocess panels to improve performance
3. Provide for diagnostic and help messages for use when requested or when the user makes an error
4. Use menus to structure an application in a manner useful for the user
5. Use the dialog test and trace services to debug an application in development
6. Provide access to the PDF BROWSE and EDIT services, where appropriate in an application
7. Create, process, and display ISPF tables, including the ISPF Table Utility
8. Use ISPF Library Access services (formerly Library Management services)
9. Use the ISPF File Tailoring services
10. Use Pop-up windows for error or other processing
11. Decide whether to code an application in a procedures language or a compiled language
12. Create user-oriented commands using the Commands Table capability of ISPF
13. Create action bars with pull-down choices
14. Create keylists for use with multiple panels.

Developing Dialog Manager Applications in z/OS - Topical Outline

Day One

Introduction to Dialog Manager

Dialog components

Dialog variables

Panel definitions

Data set requirements

Invoking Dialog Manager services from a CLIST or Exec

Invoking Dialog Manager services from a program

Dialog Testing

Computer Exercise: Setting Up for Dialog Manager 31

Panel Definitions

The LIBDEF Service

User libraries

Defining panels

Headers and sections in a panel definition

Panel design

Attribute characters

Panel definition - the)BODY section

The)END section

Panel layout concerns

DISPLAY services

Computer Exercise: Defining Panels 74

Dialog Variables and pools

Applications

Variable Pools

Function pools for execs and CLISTs

The shared pool

The application profile pool

Z variables

The system profile pool

Variable pools relationships

System variables

Variable services: VGET, VPUT, VERASE

Brief TSO Review

Brief REXX Review

Brief CLIST Review

Developing Dialog Manager Applications in z/OS - Topical Outline, p.2.

Common notes

Running Dialogs from DSLIST

Computer Exercise: The Case Study, Backup and Restore Functions 151

Day Two

Panel processing and messages

Test and Trace modes

Snapshot - quick review

Panel processing statements

Panel processing built-in functions

Control variables

Messages

Message format

Message processing

Message services

Computer Exercise: The Case Study, Panel processing and Messages 214

Basic Library Access services

Placeholder variables

DATAIDs

LMINIT

LMOPEN

LMGET

LMPUT

LMCLOSE

LMFREE

Computer Exercise: The Case Study, Part 2, Sequential I/O in a dialog 238

Pop-up windows

Windows

Primary and active windows

ADDPOP service

Window frames

Defining panels with windows

Window fit

REMPOP service

Interacting with pop-ups

Messages and windows

Computer Exercise: The Case Study, Part 2, Third Stage: Windows 257

Developing Dialog Manager Applications in z/OS - Topical Outline, p.3.

Scrollable fields

- Design issues
- Implementing scrollable fields
- The)FIELD section
- Scrollable Fields: An Example

Day Three

Menus and Debugging

- Command Processing
- Jump function processing
- Menus
- The SELECT service
- Syntax for TRANS and TRUNC in a menu
- Handling lower level requests
- Primary option menus
- Master application menus
- Menus, panels, and SELECT
- Dialog Test tracing services
- Computer Exercise: The Case Study, Add Menus 311

Some new services and tutorials

- Edit Models
- The CONTROL service
- Browse, Edit, and View services
- Browse, Edit, and View: Working with z/OS UNIX files
- Edit recovery interface
- Tutorials
- Computer Exercise: The Case Study: Add Tutorials Support 343

ISPF Tables

- Table types
- Tables and keys
- Defining tables - TBCREATE
- Row variables
- Extension variables
- Working with tables
- Working with rows
- Computer Exercise: The Case Study: Add a Table of Courses 403

Developing Dialog Manager Applications in z/OS - Topical Outline, p.4.

Day Four

Table Display services

Panels for table displays

The)ATTR section for table display panels

The)BODY section for table display panels

The)MODEL section for table display panels

The TBDISPL service

Processing selected rows

Table display variables

TBSARG and TBSCAN

Dialog Test and tables

Computer Exercise: The Case Study: Process Rows in Courses Table 447

The ISPF Table Utility

Introduction to the Table Utility

Table List Formats

Editing and Browsing Tables

Re-Structuring The Table Display

Sorting Tables

Exporting and Importing Tables

Table Utility Options

Computer Exercise: Using the Table Utility 468

File Tailoring services

The file tailoring process

Skeletons

File tailoring services - FTOPEN, FTINCL, FTCLOSE, FTERASE

Computer Exercise: The Case Study: Using File Tailoring 517

More Library Access Services

LMCOPY, LMMOVE, LMPRINT, LMRENAME, LMERASE

Library access services to work with true libraries: LMCOMP, LMMFIND, LMMREN, LMMREP, LMMADD, LMMDEL, LMMSTATS, LMMLIST, LMMDISP, MEMLIST

Library access services to work with lists of data sets: LMDINIT, LMDFREE, LMDLIST, LMDDISP

DIRLIST - Display a z/OS UNIX Directory List

Developing Dialog Manager Applications in z/OS - Topical Outline, p.5.

Day Five

Miscellaneous Topics

- Panel preprocessing
- Dialog Test: the DTEST command
- Obtaining data set information: QLIBDEF, QBASELIB, DSINFO
- Dialog Manager and commands
- Command tables
- Creating command tables
- Using command tables
- Computer Exercise: Create a User Command 599

Introduction to Common User Access (CUA)

- The CUA standard
- CUA panel formats
- Using Action Bars
- Working with pull-down menus
- Creating action bars using panel language
- Computer Exercise: Adding Action Bars 619

Keylists

- Keylists
- Dialog Tag Language (DTL)
- Defining keylists
- The ISPDTLC utility
- Using keylists
- Computer Exercise: Creating a Keylist 636

Final Topics

- Using Compiled Languages for Dialogs
- Tradeoffs
- Installing an ISPF Application
- Using ISPSTART
- Read-Only Profile Pool Extensions

Optional Exercise:

- The Case Study: Table to Sequential File [and Back] 646

Section Preview

Introduction To Dialog Manager

- ◆ Dialog components
- ◆ Dialog variables
- ◆ Panel definitions
- ◆ Data set requirements
- ◆ Invoking Dialog Manager services from a CLIST or Exec
- ◆ Invoking Dialog Manager services from a program
- ◆ Dialog testing
- ◆ Setting Up for Dialog Manager (Machine Exercise)

Dialog Manager

- Dialog Manager (ISPF) is an application development and execution tool that provides a number of services relating to displaying panels and messages, processing data, and so on**

- Typically, a dialog is driven by a program written in CLIST, REXX, or a compiled language, such as COBOL, PL/I, Assembler, C, etc.**

- In any language, requests for Dialog Manager services are made by invocations of the ISPEXEC routine**

- The Dialog Manager product is supported under z/OS TSO, OS/390 TSO, z/VM/CMS, and z/VSE**

Dialog

- A dialog is an interaction between a person and a computer system
 - ◆ Assisted by one or more functions written in CLIST, REXX, or some compiled language (or some combination of these)
 - ✗ CLIST functions are only supported under TSO
 - ✗ REXX functions are supported under TSO and CMS
 - ✗ Program functions are supported in all environments
 - ◆ And that:
 - ✗ Runs under the Dialog Manager
 - ✗ Uses Dialog Manager services

Dialog Components

- The most commonly used components of a dialog are:

Panels

- ◆ Definitions of what a screen should look like, as well as some elementary processing of input commands and data
- ◆ Created by using a text editor

Functions

- ◆ Provide the bulk of logic in an application
- ◆ May be written in CLIST, REXX, APL2, PL/I, COBOL, Assembler, Pascal, FORTRAN, C

Variable pools

- ◆ Allow communication between panels, functions, and other Dialog Manager facilities

Dialog Manager services

- ◆ Support routines for invoking panels, functions, and other Dialog Manager facilities
- ◆ Invoked by ISPEXEC commands (CLISTs and REXX execs) or CALL to ISPEXEC or ISPLINK (compiled programs)

A Dialog and Its Environment

A dialog itself may either

- ◆ Stand by itself under the Dialog Manager (as a turnkey system)
- ◆ Or it may be added to the standard list of applications in use by an installation

✗ For example, a dialog may be added as a choice on the ISPF/PDF Primary Option Menu

Dialog Structures

- The structure of a dialog is described in terms of a hierarchy of functions and panels
 - ◆ Begin with display of a panel or execution of a function (CLIST, exec, or program) that ultimately displays a panel
 - ◆ User responds to a panel by entering data or commands
 - ✗ Pressing a PF key is the same as entering the data or command string assigned to the key and pressing <ENTER>
 - ◆ The dialog examines the user data or command and decides what to do next ...

Dialog Structures, 2

Possible dialog actions on return from the display of a panel

- X** Retry the panel display until valid information is gathered (possibly issuing an error message)
- X** Process information gathered as appropriate
- X** Handle user-defined commands
- X** Issue TSO or CMS commands
- X** Request ISPF services
- X** Invoke a subsequent panel or function
- X** Repeat this panel or function, in a loop
- X** Return to the previous panel or function in the hierarchy
- X** Take a side trip to a tutorial or HELP screen (then return)
- X** Terminate the dialog

Dialog Variables

Dialog variable names

- ◆ 1 to 8 alphanumeric or national characters
(A - Z, 0 - 9, \$, #, @)
- ◆ First character of name must not be numeric
- ◆ APL2 names may not contain \$, #, or @
- ◆ FORTRAN names may only be 6 characters maximum
- ◆ Dialog Manager system dialog variable names all begin with the letter Z (so do not begin your own dialog variable names with a Z)

Dialog variable values

- ◆ Are always considered to be only character strings
- ✗ Provisions exist for converting formats when placed into, or retrieved from, program functions
- ◆ Zero to 32K bytes long

Panel Definitions

- Panel definitions may be 80 to 160 characters wide

- Resulting display may not be wider than the screen being used

- Most common to edit and store panel definitions in libraries as 80-byte records
 - ◆ No sequence numbers (NUM OFF in ISPF/PDF editor)

Sample Panel Definition

```
)BODY
%----- Customer Information -----
%COMMAND ==> _ZCMD
+
%
%Customer Number: &custno
+
+   Change request%==> _CHGREQ  + (New, Update, Examine, Delete)
+
+   Customer name%==> _CUSTNAM          +
+
+   Mailing address:
+     Line 1   %==> _ADDR1              +
+     Line 2   %==> _ADDR2              +
+     Line 3   %==> _ADDR3              +
+     City     %==> _CITY                +
+     State    %==> _ST+
+     ZIP      %==> _MAILCODE          +
+
+   Telephone numbers:
+     Main switchboard   %==> _SWITCHBD   +
+     Toll free no.     %==> _TOLLFREE    +
+
)END
```

Notes On The Panel Definition

- "+" , "%" , and "_" are examples of attribute characters

- Each attribute character takes a position on the screen, even though the character itself does not display

- Input variable names immediately follow an underscore (_)
 - ◆ The value that can be entered goes from the underscore to the plus sign (+)

 - ◆ Input variable names do not show on the display

- All other items on this screen are called "text" fields
 - ◆ Text fields may contain dialog variable names, preceded by an ampersand (&), in which case the current value in that variable will be displayed on the screen at the location shown

Resulting Display

- Assuming the current value in the variable CUSTNO is "DD87052", the display the user would see from the previous definition would look like this:

```
----- Customer Information -----  
COMMAND ==>  
  
Customer Number: DD87052  
  
Change request ==>_      (New, Update, Examine, Delete)  
  
Customer name ==>  
  
Mailing address:  
  Line 1  ==>  
  Line 2  ==>  
  Line 3  ==>  
  City    ==>  
  State   ==>  
  ZIP     ==>  
  
Telephone numbers:  
  Main switchboard  ==>  
  Toll free no.    ==>
```

Data Set Requirements

- For the Dialog Manager to find your panels, functions, messages, and so on, you must allocate certain DDnames:

<u>DDname</u>	<u>Description</u>
ISPPLIB	Panel definition library
ISPSLIB	Skeleton library
ISPTLIB	Input table library
ISPTABL	Output table library
ISPMLIB	Messages library
ISPFIL	File tailoring output file
ISPLLIB	Link library (program function load modules)
ISPPROF	User profile tables
ISPILIB	Image library (GIFs, when running in GUI mode)
SYSPROC	CLIST library
SYSEXEC	REXX exec library

- These DDnames must be allocated prior to invoking ISPF
- Usually there must be a concatenated list to include the installation libraries and your library for each type
- This allocation may be part of your logon procedure, or it may be done in a CLIST or a REXX exec
- Alternatively, a Dialog Manager service, LIBDEF, can be used for dynamic allocation (except for ISPPROF and any SYSPROC or SYSEXEC files) after ISPF invocation, from a dialog function

Invoking Dialog Manager Services From a CLIST or Exec

- ❑ From a CLIST or REXX exec, you request Dialog Manager services using the ISPEXEC command:

ISPEXEC *command-name* *parameters*

- ❑ On completion of the service, the Dialog Manager places a return code value in the CLIST variable &LASTCC or the REXX variable RC

Convention is:

- ◆ "0" means service was completed successfully
- ◆ Other values may mean errors, or they may just be informative
- ◆ Possible values are documented as part of the description of each service

Invoking Dialog Manager Services From a CLIST or Exec, 2

- ❑ For example, to request allocation of your panel library, code:

```
ISPEXEC LIBDEF ISPPLIB DATASET ID(panel-lib-name)
```

- ◆ This places your panel library ahead of the system libraries allocated to ISPPLIB for your session

- ❑ Then, to request a panel display, issue:

```
ISPEXEC DISPLAY PANEL(panel-name)
```

The Dialog Manager

- ◆ Searches the dataset(s) pointed to by ISPPLIB
- ◆ Displays the panel (if the panel cannot be found, Dialog Manager terminates the request with a non-zero return code)

The user then keys in data or commands and presses <ENTER>

- ◆ Any data entered into input variables are stored into the appropriate dialog variables

Control returns to the next statement in your CLIST or exec

Notes For Dialogs Written In REXX

- Before you make your first request for ISPF services from an exec, you may issue

ADDRESS ISPEXEC

- ◆ In which case you may omit ISPEXEC on your subsequent Dialog Manager requests:

DISPLAY PANEL(MYPAN1)

- ◆ Also, then, commands directed to other host environments must be explicitly ADDRESSed to the name of the intended host environment:

ADDRESS TSO ALLOCATE ...

- Host statements with special characters (especially parentheses) need to be bounded by [single or double] quotes

- ◆ But make sure variables to be substituted are left un-quoted:

"DISPLAY PANEL("VARX")"

- ◆ Here, VARX will have its value substituted before this request is passed to the Dialog Manager, while the rest of the string is simply passed on as is

- ISPEXEC statements are case insensitive, even in quotes

- In this course, we follow the convention used in the IBM manuals: omit ADDRESS, include ISPEXEC, and do not generally code quotes (except when needed for accuracy)

Invoking Dialog Manager Services From a Program

- If you write a function in a compiled language, you simply issue a **CALL** to the **ISPLINK** routine (the examples assume the appropriate variables have been initialized):

Assembler

```
CALL  ISPLINK, (LIBDEF, ISPPLIB, DATASET, LIBNAME), VL
```

COBOL

```
CALL  'ISPLINK' USING LIBDEF ISPPLIB DATASET LIBNAME
```

PL/I

```
CALL  ISPLINK ('LIBDEF', 'ISPPLIB', 'DATASET',  
              '('libname')');
```

C

```
ISPLINK (LIBDEF, ISPPLIB, DATASET, LIBNAME);
```

Invoking Dialog Manager Services From a Program, continued

- Or, you can CALL ISPEXEC, using this format:

```
CALL ISPEXEC (buf-len,buffer)
```

or, in C:

```
rc = ispexec(&buf_len,buffer);
```

X Where buf-len is a fullword binary integer containing the length of the buffer

X And buffer contains the name of the service and its parameters, as if the command had been issued from a CLIST or exec

- Since FORTRAN only allows six character module names, you must use the synonyms ISPLNK and ISPEX for ISPLINK and ISPEXEC, respectively

- On completion of the service, the standard return code value is returned, using the linkages expected by the CALLing language

Programming Notes

- CALLs to Dialog Manager services from program functions must pass parameters in a predetermined positional sequence**

- For teaching purposes, we do not always list all possible parameters for a service call, and sometimes we may list parameters out of sequence**

- When in doubt, check the "ISPF Dialog Developer's Guide and Reference" manual**

- Especially note the use of apostrophes for languages that allow literals in CALL parameter lists, and how to indicate that positional parameters are omitted**

- Check the Appendix to these materials for some simple examples of calling ISPF services from compiled programs and, at the end, some sources of information**

Some Other Dialog Manager Services

- ❑ **Aside from displaying panels and invoking functions, some of the other services available from the Dialog Manager are:**
 - ◆ **Support for messages and tutorials**
 - ◆ **Create, display, and modify data in ISPF tables**
 - ◆ **Facilities for creating tailored JCL, program code, or data, based on pre-coded "skeleton" JCL, program code, or data and the current values in dialog variables**
 - ◆ **Interfaces to library access routines**
 - ◆ **Interfaces to ISPF/PDF Browse, View, and Edit services**
 - ◆ **Interfaces to command tables, to build your own commands**
 - ◆ **Support for Double Byte Character Set (DBCS) data, and other international requirements (punctuation for numeric values, date formats, and so on)**

- ❑ **ISPF also has the ability to run in "GUI mode"**
 - ◆ **This means dialogs can run on a workstation using the facilities normally associated with GUI interfaces (check boxes, drop down lists, push buttons, etc.)**
 - ◆ **However, we only discuss this ability tangentially in this course - it would be a distraction from our main goals, and this capability is no longer being enhanced**

Dialog Testing

- Dialogs have many pieces to them that all need to fit together for a dialog to work properly

- ISPF/PDF has provided a facility for testing the individual pieces of a dialog as they are written, and for debugging errors in existing dialogs

- ISPF/PDF option 7 is usually Dialog Test
 - ◆ This is a Primary Option Menu
 - ✗ Which means once you are in it, you can not "jump" out of it to some other option outside of Dialog Test:
 - ✗ For example, if you specify ==> =3.4 on a panel under option 7, you will be sent to Dialog Test option 3 suboption 4 (an error) instead of to PDF option 3.4

 - ✗ When you are in a suboption of Dialog Test, entering =X on the command line takes you to the standard Primary Option Menu, while "RETURN" takes you to the Dialog Test Primary Option Menu

- If you will be using Dialog Test, make sure your LOG default is not "2" (Delete)
 - ◆ Dialog Test writes trace and debugging type information to the log, and you may want to put data out there too

Dialog Test Primary Option Menu

☐ Here is a typical menu for Dialog Test:

```
Menu Utilities Compilers Options Status Help
----- Dialog Test -----
  Menu Utilities View Help
-----
0 |                                     Primary Option Panel
0 | Option ==>
1 |
2 | 1 Functions          Invoke dialog functions/selection panel _____
3 | 2 Panels             Display panels
4 | 3 Variables          Display/set variable information
5 | 4 Tables             Display/modify table information
6 | 5 Log                Browse ISPF log
7 | 6 Dialog Services   Invoke dialog services
8 | 7 Traces             Specify trace definitions
9 | 8 Breakpoints        Specify breakpoint definitions
1 | T Tutorial           Display information about Dialog Test
1 | X Exit              Terminate dialog testing
```

select →

Dialog Test Option 6: Requesting Dialog Services

- ❑ Filling in the command field here, you do not need to prefix it with "ISPEXEC"

```
Menu List Mode Functions Utilities Help
-----
Invoke Dialog Service

Command ==>

Enter dialog service and its parameters:
==>

Place cursor on choice and press enter to retrieve command.

=>
=>
=>
=>
=>
=>
=>
=>
=>
=>
```

- ❑ For example, you might enter:

```
==> LIBDEF ISPLLIB DATASET ID('panel-library-name')
```

Dialog Test Option 2: Display A Panel

- ❑ You can request a panel be displayed, even if the panel is not in the context of a dialog:

```

Menu Utilities Compilers Options Status Help
----- Dialog Test -----
| Menu Save Utilities Help |
|-----|
|           Display Panel   |
| Command ==>              |
| Panel name . . . . .     |
| Message id . . . . .     | (Optional)
| Cursor field . . . . .   | (Optional)
| Cursor position . . . . .| (Optional)
| Message pop-up field . . | (Optional)
|
| Enter "/" to select option|
|   Display in window      |
|-----|
| Enter X to Terminate using log/list defaults

```

- ◆ Just key in your panel name, and the panel will be displayed
- ◆ If you have an error, you will get a diagnostic message
- ◆ If you have no logic associated with this panel, you exit with an "END" command (PF3)

Computer Exercise: Setting Up for Dialog Manager

This exercise is intended to help you set up for subsequent Dialog Manager work in the class. There will be more setup work later, also, but this will give you a good start.

Step 1: Setting Up Libraries

First, you need to run A810STRT, a supplied REXX exec that runs a dialog that will prompt you for the high level qualifier (HLQ) you want to use for your data set names; the exec uses a default of your TSO id, and that is usually fine; it also asks if you intend to code your labs in CLIST or REXX; then it creates data sets and copies members you will need.

From ISPF option 6, on the command line enter:

```
===> ex '_____ .train.library(a810strt)' exec
```

The files created are:

userid.TR.PANELS	Panel Definitions
userid.TR.MESSAGES	Message Definitions
userid.TR.PEOPLE	data file used in later labs
userid.TR.TABLES	for table handling labs
userid.TR.EXEC	REXX Functions (if using REXX for exercises)
OR	
userid.TR.CLIST	CLIST Functions (if using CLIST for exercises)

Step 2: Creating a Panel Definition

In your panel library, create a member called SAMPL01 based on the definition on page 16 of the student handout.

Step 3: Testing a Panel Definition

Use Dialog Test to allocate your panel library ahead of the system panel libraries. Then display your panel definition using option 2 of Dialog Test.

Computer Exercise: Setting Up for Dialog Manager, p.2.

Optional Step: Start a Dialog

In your EXEC or CLIST library, create a function called DIALOG01 that only contains two commands:

- 1) A LIBDEF request to put your panel library ahead of the system panel library
- 2) A request to display your panel (Hint: see page 21)

Under option 6 (TSO, not under Dialog Test), execute your procedure.

Section Preview

Panel Definitions

- ◆ LIBDEF service
- ◆ User libraries
- ◆ Defining panels
- ◆ Headers and sections in a panel definition
- ◆ Panel design
- ◆ Attribute characters
- ◆ Panel definition - the)BODY section
- ◆ The)END section
- ◆ Panel layout concerns
- ◆ Display services
- ◆ Defining panels (Machine Exercise)

LIBDEF Service

- ❑ The "LIBDEF" ISPF service allows you to specify application level data sets to be concatenated ahead of the ISPF system level data sets

Syntax

```
ISPEXEC LIBDEF lib-ddname  
  
[ID(dsname-list)|ID(ddname)]  
  
[DATASET|LIBRARY]  
  
[COND|UNCOND|STACK|STKADD]
```

Where

<u>lib-ddname</u>	<u>Used for application level</u>
ISPMLIB	Message library
ISPPLIB	Panel library
ISPSLIB	Skeleton library
ISPTLIB	Table input library
ISPTABL	Table output library
ISPFIL	File tailoring output library
ISPLLIB	Load module library
ISPILIB	Image library (for ISPF GUI mode)
xxxxxxx	Generic library (may be used for table input, table output, or file tailoring output)

Other LIBDEF Parameters

DATASET

- ◆ Indicates the "ID" field contains a list of data set names

LIBRARY

- ◆ Indicates the "ID" field contains the DDname of a previously ALLOCATED file

COND

- ◆ Do the LIBDEF only if there is not already some active file of this type

UNCOND

- ◆ Do the LIBDEF unconditionally (replace any pre-existing LIBDEF for this lib-ddname); this is the default

STACK

- ◆ Save the current LIBDEF setting in a stack, then do the LIBDEF unconditionally

STKADD

- ◆ Concatenate the new library of this type ahead of the current library (ies); only valid if DATASET also specified

If DATASET and LIBRARY are omitted, or if ID has a null value, ISPF is to remove any data set already associated with this lib-ddname

- ◆ If the previous LIBDEF value was STACKed, then it is restored

Return Codes From LIBDEF

- 0 — Successful completion
- 4 — Attempt to remove library, but library did not already exist
- 8 — COND specified, and application library was already established; current LIBDEF was not established
- 12 — Invalid *lib-ddname* specified
- 16 — Invalid contents of ID parameter
- 20 — Severe error

LIBDEF: Special Cases

- ❑ Two additional, special parameters apply to the link (load module) library, ISPLLIB, only:

"EXCLDATA" instead of "DATASET"

"EXCLLIBR" instead of "LIBRARY"

- ❑ When either of these is specified, the search for ISPF program and command load modules will only consider the ISPLLIB dataset(s), ignoring any user and ISPF system libraries

- ◆ Although the Link Pack Area and Linklist data sets will still be searched for programs invoked dynamically by these programs (and the ISPLLIB will not be searched)

- ❑ To find out the status of LIBDEFs, one can issue the ISPLIBD command from any command line

- ◆ This brings up a panel displaying the current status of all LIBDEFs

User Libraries

- When you establish application level libraries for your dialog or application, individual users may wish to have their private libraries searched ahead of any LIBDEFed libraries
 - ◆ Or you may want each user to have their own private library of one or more types, to tailor the application even more for each individual user

- This can be accomplished by using TSO "ALLOCATE" commands prior to starting ISPF
 - ◆ Could be in the TSO logon procedure, for example

 - ◆ Or in a CLIST or exec you invoke prior to starting ISPF

- A different set of DDnames must be used for these ALLOCATE commands:

<u>DDname</u>	<u>For user level</u>
ISPMUSR	Message library
ISPPUSR	Panel library
ISPSUSR	Skeleton library
ISPTUSR	Table library
ISPTABU	Table output library
ISPFILU	File tailoring output library
ISPLUSR	Load module library
ISPIUSR	Image library

User Libraries, 2

- ❑ Allocation of these DDnames prior to starting ISPF, and the presence or absence of a LIBDEF service request, interact to determine the order of library search:
 - ◆ If a user library has been ALLOCATED, with no LIBDEF, the user library allocation is ignored!
 - ◆ If a user library is allocated, and a LIBDEF request is made for the same library type, the search order is
 - X User library
 - X LIBDEF library
 - X System library
 - ◆ If no user library is allocated, LIBDEF causes library search to use the LIBDEF library before the system library

Panel Types

Selection Panel

- ◆ A MENU; Select from a list of options

Parameter Entry Panel

- ◆ A Fill-in-the-blanks screen; for example, specify a dataset name for browse or edit

Scrollable Data Display

- ◆ Table Format (Data arranged in rows and columns)

X For example, a member selection list for members in a library

- ◆ Dataset Format (Displays one row per record)

X For example, PDF edit and browse

Information-only Panels

- ◆ Full screen of text, such as tutorial or help panels

Defining Panels

- A panel is defined by coding a series of lines in a member of a library used to store panel definitions: a panel library

- These lines defining a panel are written in what is called "Panel Language" or "Panel Definition Language" in the literature
 - ◆ There is an alternative language for defining panels called "Dialog Tag Language" or "DTL"; DTL is not covered in this course
 - ✗ Except for a short section near the end needed for creating keylists

- A panel definition can have up to 17 kinds sections

- Each section is designated by a header of a right parenthesis, ")", followed immediately by the name of the section

- The right parenthesis must be in column 1 ...

Headers and Sections in a Panel Definition

<u>Header</u>	<u>Designates Section</u>
)INEXIT	*Identify an exit to process panel source on the fly
)CCSID	*Coded Character Set IDentifier - foreign language support
)PANEL	Panel information - any keylist id; forces CUA mode
)ATTR	Attribute - Define attribute characters for display (these attribute characters may then be used in the BODY section)
)ABC	Action Bar Choice - defines action bar components
)ABCINIT	Action Bar Choice Initialization - required if ")ABC" specified
)ABCPROC	Action Bar Choice Processing - panel processing for action bar
)BODY	Panel Body - The representation of the screen appearance
)MODEL	Model Lines - For table displays
)AREA	*Scrollable Area Definition - for non-table scrollable areas
)INIT	Initialization - Pre-DISPLAY processing
)REINIT	Reinitialization - Processing prior to reshowning a panel
)PROC	Process - Describes how to process any input variables entered on the panel
)FIELD	Scrollable field section; define a field as scrollable
)HELP	*Help section - used for field-level help
)PNTS	*Point and shoot - declarations for all point and shoot fields
)LIST	*List - specifies list to build on panel
)END	End - Signifies the end of the panel definition

Headers and Sections in a Panel Definition — Notes

- Only the "BODY" and "END" sections are required

- Sections with asterisks (*) at head of their description are not covered in this course

- If a section from)INEXIT to)PROC is included in a panel definition, it must appear in the relative order listed on the previous page
 - ◆ Section types)FIELD,)HELP,)LIST, and)PNTS may occur in any order as long as they appear after sections)CCSID through)PROC

 - ◆ And)END must always be the last section in a panel definition

Panel Design

- There are four types of panels supported by ISPF, and the type of panel implies the necessary sections needed in a panel definition

- Also, there are some conventions about panel layout we should follow, in the interest of providing a consistent, familiar environment for the user

- At this point, we are looking at defining basic parameter entry / data display (information only) panels

- All panel definitions require)BODY and)END sections, and many use the)ATTR section, so these sections are our first concern ...

Attribute Characters

- ❑ Each field on a screen must be assigned a collection of attributes that indicate data types (Input, Output, Text), display characteristics (Intensity, Color, Highlighting) and so on

- ❑ We do this by assigning collections of attributes to specific characters
 - ◆ And then placing these attribute characters in the panel body, thus assigning attributes to the following text or variable fields

- ❑ The characters to use may be any alphanumeric or special character or two-digit hex number, except the ampersand (&) and the following hex values: 00, 0E, 0F, 40
 - ◆ Up to 127 attribute characters may be defined for a single panel
 - ◆ Recommend: Do not use characters that could conflict with panel text
 - ✗ For example, if you will be using "===>" for arrows, do not use the equals sign ("=") or greater than sign (">") for attribute characters

Attribute Characters, 2

Sample Attribute Section

)ATTR

\$ TYPE(INPUT) INTENS(LOW) CAPS(ON)

@ TYPE(OUTPUT) CAPS(OFF) COLOR(PINK)

- An attribute character applies to all following data (even on subsequent lines) until another attribute character is encountered (or end of screen)

- Each attribute character occupies a location (character position) on the screen (although the character will not display)

- The most useful attribute options are discussed on the following pages

Programming Note

◆ All panel language statements, variable names, keywords, and keyword values can be entered in uppercase or lowercase

✗ Literal values are stored as entered; other elements are translated to uppercase before processing

TYPE(TEXT|INPUT|OUTPUT)

- ❑ Field TYPE basic attributes are

TEXT

- ◆ Display on screen exactly as shown in panel definition; except, substitute values for symbolic variables
- ✗ A symbolic variable is a dialog variable name preceded by an ampersand (*e.g.*: &CUSTNO)

INPUT

- ◆ Single variable; current value is displayed; user may type over

OUTPUT

- ◆ Single variable; current value is displayed; user may not type over

- ❑ Additional TYPE attributes are used for Dynamic Areas and for CUA panel elements; these topics are not discussed in this course

TYPE(TEXT|INPUT|OUTPUT), 2

- ❑ User may not type over or change TEXT or OUTPUT fields (we say the field is protected)

- ❑ On a panel definition, a field defined with an attribute character indicating a TEXT field will display exactly as coded; symbolic variables will have their values substituted, with trailing blanks stripped

- ❑ A field defined with an attribute character indicating an INPUT field must be immediately followed by a dialog variable name, with no intervening ampersand (e.g.: _EMPID if "_" is an INPUT attribute character)
 - ◆ No text may be included for an INPUT field type; ISPF initializes the display with the current value in the variable, and the user may overwrite or enter new data

- ❑ A field defined with an attribute character indicating an OUTPUT field must be immediately followed by a dialog variable name, with no intervening ampersand
 - ◆ No text may be included for an OUTPUT field type; ISPF initializes the display with the current value in the variable, but the user may not overwrite or enter new data

INTENS(HIGH|LOW|NON)

Field INTENSITY attributes are

HIGH

- ◆ Display with high intensity (if supported on terminal)

LOW

- ◆ Display with low intensity (if supported on terminal)

NON

- ◆ Non-display; do not display this field

Dynamic Attribute Specification

- ❑ Many of these field attributes may be specified using symbolic variables

For Example:

INTENS(&XYCH)

- ❑ Then dynamically changing the value in the symbolic variable changes the characteristics of the attribute character:

&XYCH = LOW

&XYCH = NON

- ❑ In this instance, you may selectively display or not display certain lines on a panel, depending on the circumstances

- ◆ For example, display some text only if a particular option is requested
- ◆ Or display some text only for novices

- ❑ Exception: TYPE(TEXT) may not be specified as a result of symbolic variable substitution

Dynamic Attribute Specification, Continued

- The actual changing of the symbolic variable values may be done in the **"INIT"** section of the panel definition (as shown on the previous page)

- Or it may be done in the function (EXEC, CLIST, or program) prior to displaying the panel
 - ◆ In REXX: `xych = low`

 - ◆ In CLIST: `SET &XYCH = LOW`

CAPS(ON|OFF|IN|OUT)

- Field CAPS attributes are

ON

- ◆ Translate data to upper case before display and (if an INPUT field) when entered

OFF

- ◆ Display and accept data as is (do not translate to upper case)

IN

- ◆ Display as found in variables, but force to upper case when data entered

OUT

- ◆ Display all variables in upper case; accept input data as is

- For a CLIST function, CONTROL ASIS must be specified for "OFF", "IN", or "OUT" to take effect

- Default is CAPS(ON) for most situations

HILITE(USCORE|BLINK|REVERSE)

- Field HILITE attributes are

USCORE

- ◆ Underscore the data (display data as underlined)

BLINK

- ◆ Display data blinking

REVERSE

- ◆ Display data in reverse video

- These are useful for drawing attention to a field

- ◆ Especially when a user enters bad data, you can dynamically alter the attribute for that field to have one of these attributes to emphasize which field is in error

- The default is to have no highlighting

- If terminal does not support extended highlighting, this parameter is ignored (or forces high intensity in some cases)

PAD(NULLS|USER|char)
or
PADC(NULLS|USER|char)

Field padding attributes ...

- ◆ **Are optional; but if one is specified, can not specify the other**
- ◆ **Are not valid for TEXT fields**
- ◆ **Specify what character to initialize a field with when first displayed**

PADC character is used conditionally: only if field is initially blank; default is PAD(USER) for INPUT fields, PAD(" ") (blank) for OUTPUT fields

NULLS

- ◆ **Null padding character**

USER

- ◆ **Allow user to specify PAD character (using PDF option 0.1)**

char

- ◆ **Enter a literal character (e.g.: PAD("***"))**

SKIP(ON|OFF)

Field SKIP attributes are valid only for TEXT and OUTPUT fields (protected fields)

The possible values are

ON

- ◆ Cursor skips over field when a character is entered into the last position of the preceding INPUT field

OFF

- ◆ Cursor does not automatically skip over this field

SKIP(ON) is user-friendly for screens that have lots of TEXT and OUTPUT fields interspersed with INPUT fields

COLOR(value)

- Field COLOR attributes are chosen from
 - ◆ WHITE|RED|BLUE|GREEN|PINK|YELLOW|TURQ

- If a terminal does not support color, the color value is used to imply a default intensity
 - ◆ BLUE, GREEN or TURQUOISE imply LOW intensity
 - ◆ WHITE, RED, PINK or YELLOW imply HIGH intensity
 - ◆ Depending on terminal type, if INTENSITY is explicitly specified, COLOR may be ignored

- If a terminal does support color, and COLOR is not specified, colors are chosen implicitly based on field TYPE and INTENSITY:

<u>FIELD TYPE</u>	<u>INTENSITY</u>	<u>COLOR</u>
TEXT/OUTPUT	HIGH	WHITE
TEXT/OUTPUT	LOW	BLUE
INPUT	HIGH	RED
INPUT	LOW	GREEN

Options Supported But Not Discussed Here:

- Light Pen / Cursor Select Key feature
- Text justification
- Numeric only input
- Disallow jumping from input fields
- Double Byte Character Set (DBCS) parameters
- Dynamic Areas
- Graphic Areas
- CUA Panel Elements
- GUI mode elements

- Rules for defaults and mutually exclusive combinations are spelled out in the Dialog Developer's Guide And Reference manual

Default Attribute Characters

- ❑ If you omit the)ATTR section, ISPF provides three default attribute characters implicitly:

<u>Character</u>	<u>Attributes</u>
%	TEXT, HIGH INTENSITY
+	TEXT, LOW INTENSITY
-	INPUT, HIGH INTENSITY

- ❑ You can override what characters should play these roles by specifying the DEFAULT parameter in the)ATTR header:

```
)ATTR  DEFAULT($@#)
```

- ◆ The three characters in the parentheses must be adjacent, no intervening spaces or commas

- ❑ You may override default characters and specify additional attribute characters:

```
)ATTR  DEFAULT(?$#)  
      %    TYPE(INPUT)    COLOR(GREEN)  
      @    TYPE(TEXT)     COLOR(PINK)  
      /    TYPE(OUTPUT)   COLOR(BLUE)
```

Standard Panel Format

- ❑ The first three lines of most panels have special reserved functions, leaving the rest of the screen available for specific application use

LINE 1	Title of panel	Short Message
LINE 2	Command / option line	
LINE 3	Long Message	

- ◆ Short Message area is used to pass information about a request (status, error, etc.)

- ◆ Long Message area is normally used for headers, data, etc.

✗ But if an error message appears in the Short Message area, issuing a HELP command or pressing the HELP PF key will cause further explanation to display in the Long Message area; a Long Message may cover multiple lines, being up to 512 characters long

- ❑ Note that as of ISPF Version 4, the standard style has been supplemented with the CUA (Common User Access) style, which we discuss briefly later in the course

Panel Definition - The BODY Section

- ❑ The header for the BODY section may contain some keyword parameters:

**)BODY [CMD(*field-name*)] [SMSG(*field-name*)] [ASIS]
 [LMSG(*field-name*)] [WIDTH(*nn*)] [EXPAND(*xy*)]**

- ❑ Note that the)BODY statement cannot extend past one line

CMD / SMSG / LMSG Parameters:

- ◆ Normally, the first INPUT field on a panel is treated as the command field, the place where the user may enter a command
- ◆ However, you may designate any INPUT field on the panel as the command field; do this by specifying the name of a variable that follows an INPUT attribute character in the BODY in the “CMD(*field-name*)” parameter on the header
- ◆ Similarly, the short message field is normally the last 26 characters of line 1, but you can change that by specifying “SMSG(*field-name*)”; the specified field must be TYPE(OUTPUT)
- ◆ Also, the long message field is, by default, the third line of a panel, but you can change it by coding “LMSG(*field-name*)” in the header; this field must also be TYPE(OUTPUT)

The BODY Section, continued

ASIS:

- ◆ Means: do not let the user override the placement of the command and long message lines using PDF option 0

WIDTH:

- ◆ Specify the character width (from 80 to 160) of the panel (it must not be greater than the screen width of the terminal being used)
 - ◆ If the width is greater than 80, this parameter must be specified
- If you specify CMD as “CMD()”, no command field will be used
- ◆ For a table display panel you must have a command field
- You may code the DEFAULT parameter (to change the default attribute characters) on the)BODY header if you do not do so on the)ATTR header

The BODY Section, 3

EXPAND:

- ◆ You can request ISPF to repeat a character as many times as it will fit on a line
- ◆ Do this by identifying the starting and ending delimiter characters in the EXPAND parameter of the BODY header:

```
)BODY      EXPAND(//)
```

- ◆ Note the starting and ending delimiter characters can be the same
- ◆ Now, whenever a single character is bounded by the two characters, ISPF repeats the bounded character to fill out the line:

```
+--/-- &TITLE --/--  
%COMMAND===>_ZCMD //  +SCROLL%===>_SCRL+  
+  
%CUSTOMER NAME: _CUSTNAME      //      +
```

- See the Dialog Developer's Guide and Reference manual for pointers and concerns with this facility

BODY Section Contents

- ❑ After the ")BODY" header line, code lines that describe the screen image, using the attribute characters you have previously defined

Remember

- ◆ Only "TEXT" TYPE fields can use ampersands (&) to request symbolic substitution
- ◆ "INPUT" and "OUTPUT" TYPE fields simply have variable names immediately following their respective attribute characters

Also

- ◆ A dialog variable may only appear once in a panel body in a non-TEXT field (a field with an attribute of INPUT or OUTPUT)
 - ✗ A dialog variable may appear any number of times in a panel body in a field with the TEXT attribute

The END Section

- A panel definition must always terminate with an)END statement

- Any lines following this statement are ignored

- Blank lines and comments may appear anywhere in the following sections

```
)ATTR  
)INIT  
)REINIT  
)PROC
```

- Comments are of the form: `/* comments */`
 - ◆ A comment must be the last (rightmost) element on a line

- Note that since any lines placed after the)END statement are ignored, this is a good place for comments documenting a panel definition

Another Sample Panel Definition

```
)ATTR
  $ COLOR(RED) TYPE(TEXT)
  * COLOR(BLUE) TYPE(INPUT)
)BODY EXPAND(//)
%--/--/ VSAM Data Set CISIZE Selection /-/--
%Command ==> _ZCMD
%
+Fill in Data Set information:
+
$      Record size          %==>*recsz+
$      Max CISIZE           %==>*maxci +
$      No. Records          %==>*norecs +
$      Organization         %==>*org + (KSDS,RRDS,ESDS,LSDS)
+
+For KSDS, specify free space, if known:
+
$      CI FREESPACE         %==>*cifspc+
$      CA FREESPACE         %==>*cafspc+
)END
/* panel to extract parameters for calculations */
/* Copyright (C) 2000 by Steven H. Comstock */
```

◆ Resulting display:

```
----- VSAM Data Set CISIZE Selection -----
Command ==>

Fill in Data Set information:

    Record size          ==>
    Max CISIZE           ==>
    No. Records          ==>
    Organization         ==>          (KSDS,RRDS,ESDS,LSDS)

For KSDS, specify free space, if known:

    CI FREESPACE         ==>
    CA FREESPACE         ==>
```

Panel Layout Concerns - Line 1

- Line 1 should contain a title and any critical information
 - ◆ For example, consider ISPF edit, which identifies the dataset [and member] being edited

- If possible, let the right hand 26 characters of line 1 be used for a Short Message area

Panel Layout Concerns - Command Line

- ❑ Although the user can change the location of the command line (and, sometimes, the Long Message line), you should plan for the command line to be on line 2
 - ◆ If the command line is near the bottom, on entering split screen mode, the user may not be able to get to the command line
 - ◆ The command field may not be longer than 255 characters (other fields may be longer than 255 characters)

- ❑ The command field may have any name, but “ZCMD” is a system variable provided for this purpose and seems to be as good a choice as any; implications:
 - ◆ If you use “ZCMD”, default initial cursor placement is in the first INPUT field following the command field, if the command field is blank (and in the command field if it is non-blank)
 - ◆ If you use some other name for the command field, the default initial cursor placement is in the command field
 - ◆ In any case, the value of certain dialog variables and other influences may be at work here, so trial and error is sometimes called for

Panel Layout Concerns - Line 3

- Generally leave line 3 (the default Long Message line) blank

- Never let the Long Message area be on the same line as an INPUT field

Panel Layout Concerns - General

- ❑ **Keep panel “look and feel” simple and consistent**
 - ◆ **If too many fields, break panel into two successive panels**
 - ◆ **Group related fields under a common heading**
 - ◆ **Align fields vertically on a panel, whenever possible**
 - ◆ **Establish visual “signals” to clue the user in, even subconsciously, to the kinds of fields on the panel**
 - ✗ For example, using arrows (==>) to indicate input fields and colons (:) to indicate protected data
 - ◆ **Consider keeping in the same style as ISPF/PDF**
 - ✗ In any case, at least be consistent across an application
 - ◆ **Use color and highlighting sparingly**
 - ✗ Be consistent: use the same colors / highlighting for similar types of information

More Panel Layout Concerns

- ❑ Realize that if the user is requesting “PFSHOW”, two to four lines at the bottom will be taken up with the PFKEY display

- ❑ Split Screen Impacts
 - ◆ Do not use the last line on a panel (may not see in split screen mode)

 - ◆ Put most important fields near the top (may need to re-split the screen to see, otherwise)

Display Services

- ❑ To understand the basic handling of a DISPLAY request, remember that a panel definition may contain none, some, or all of the following sections:

)INIT - to perform pre-display processing

)REINIT - to perform pre-redisplay processing

)PROC - to perform post-display processing

- ❑ Panel processing includes

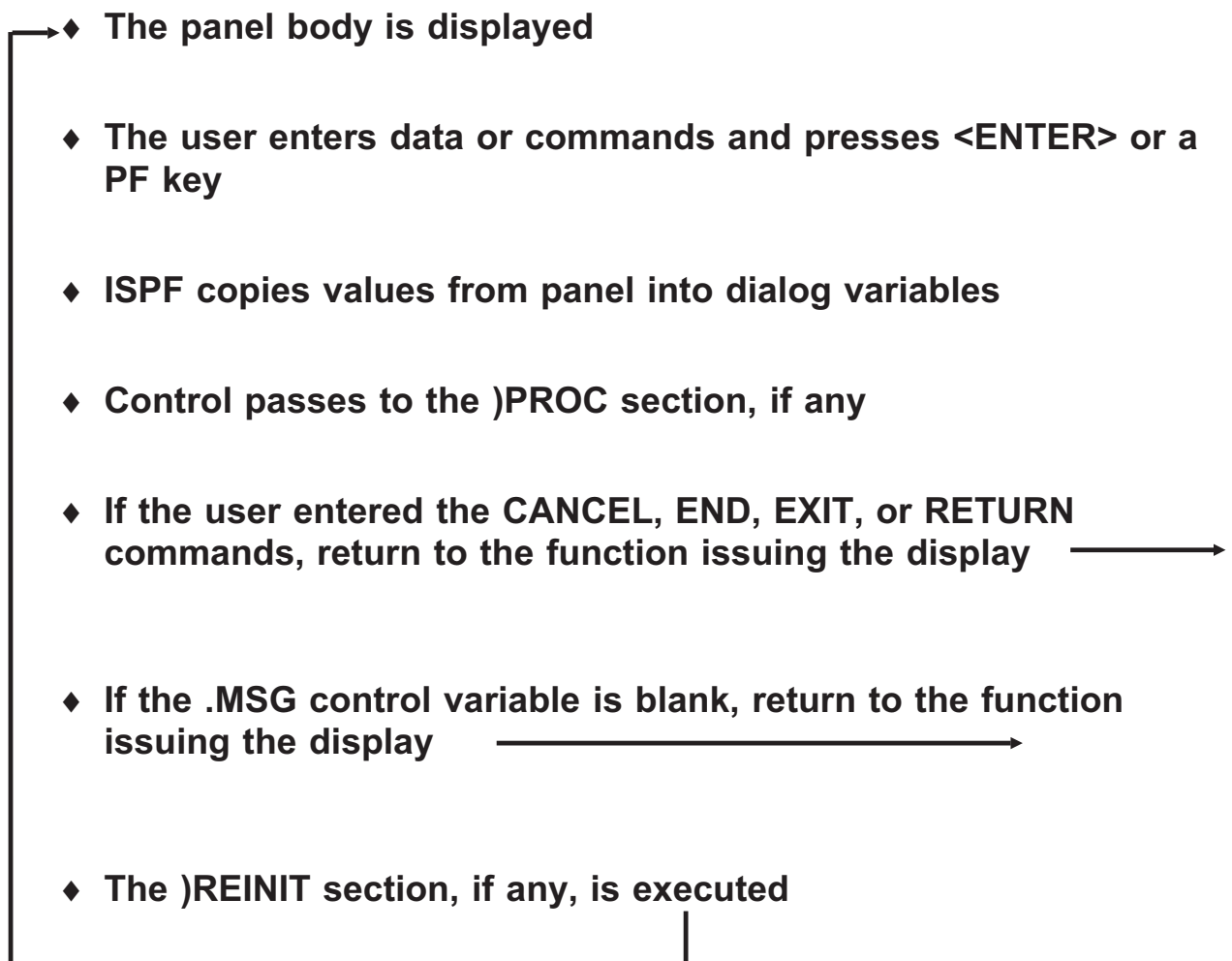
- ◆ Examining variables, changing variable values, and validating values the user may have entered
- ◆ If data is not valid, setting the .MSG panel control variable to indicate what error message to issue

✗ If .MSG is not blank, the panel will be redisplayed without returning to your dialog function

Display Services, continued

□ The general processing of a DISPLAY is as follows:

- ◆ The named panel is found in the panel library (ISPPLIB)
- ◆ The)INIT section, if any, is executed
- ◆ ISPF fills variable values into the panel body



Request Panel Display

- ❑ To display a panel, simply issue the ISPEXEC request DISPLAY:

ISPEXEC DISPLAY [PANEL(panel-name)]

- ❑ If the PANEL parameter is omitted, the)REINIT section, if any, is processed and the current panel is redisplayed

- ❑ After the DISPLAY, the return code is one of these values:

- 0 - Normal completion
- 4 - Unknown command entered
- 8 - User entered END or RETURN command
- 12 - Panel could not be found
- 16 - Error in storing or processing variables
- 20 - Severe error

Two other optional parameters on the DISPLAY request

- ◆ **CURSOR**(*field_name*) - where cursor should appear on display
- ◆ **CSRPOS**(*position*) - position where cursor should appear within the field where it appears

Computer Exercise: Defining Panels

REXX VERSION

1. Code two panel definitions, called SAMPLX2 and SAMPLX3, as described on the following page. SAMPLX2 is to display some REXX variables using different attributes, as well as to provide a field for user data entry. SAMPLX3 will display the user-supplied data as well as some status information.

2. After you have coded the panel definitions, code an EXEC that performs as follows:

- Put TIME() into UTIME1
- Put DATE() into UDATE
- Put USERID() into USER
- PARSE SOURCE into SRCE
(Parse source SRCE)
- PARSE VERSION into VER
(Parse version ver)
- Put LINESIZE() into UWTERM
- Issue a LIBDEF request to put your panel library in the list of panel libraries
- Put TIME() into UTIME2
- Request a DISPLAY of SAMPLX2
- Put RC into USERCC
- Put TIME() into UTIME3
- Put USERIN into USERIN2
(Parse upper var USERIN USERIN2)
- Request a DISPLAY of SAMPLX3

3. Run this dialog from option 6 (COMMAND) of ISPF.

4. [optional] issue the ISPLIBD command and examine the output.

REXX VERSION

Description of panel SAMPLX2

This panel should display various REXX variables using different attribute characters, as follows:

<u>Variable</u>	<u>Attributes</u>
SRCE	Output, low intensity, red
UWTERM	Output, high intensity, red
USER	Output, low intensity, green
UDATE	Output, high intensity, green

These fields should be labelled by text fields (white).

Also display a text field (blue) containing the text:

TIME AT ENTRY TO EXEC: &UTIME1, TIME NOW: &UTIME2
REXX VERSION AND RELEASE: &VER

Ask the user to enter a string of text, including upper and lower case characters.

Save this string in the field USERIN. Remember to define an attribute character with TYPE(INPUT) and CAPS(OFF) for defining the input variable for this string.

Description of panel SAMPLX3

This panel should display the following variables:

USERCC
UTIME1
UTIME2
UTIME3
USERIN as is
USERIN with Caps on (use variable USERIN2)

NOTES

1. Include the panel name on each panel.
2. Experiment with different EXPAND characters.
3. Experiment with whatever HILITE and COLOR options your terminal supports.

CLIST VERSION

1. Code two panel definitions, called SAMPL02 and SAMPL03, as described on the following page. SAMPL02 is to display some TSO CLIST variables using different attributes, as well as to provide a field for user data entry. SAMPL03 will display the user-supplied data as well as some status information.

2. After you have coded the panel definitions, code a CLIST that performs as follows:

- Establish CONTROL ASIS
- SET &SYSTIME into &UTIME1
- SET &SYSTSOE into &VER
- SET &SYSLTERM into &ULTERM
- SET &SYSWTERM into &UWTERM
- Issue a LIBDEF request to put your panel library in the list of panel libraries
- SET &SYSTIME into &UTIME2
- Request a DISPLAY of SAMPL02
- SET &LASTCC into &USERCC
- SET &SYSTIME into &UTIME3
- SET &USERIN into &USERIN2
- Request a DISPLAY of SAMPL03

3. Run this dialog from option 6 (TSO) of ISPF.

4. [optional] issue the ISPLIBD command and examine the output

CLIST VERSION

Description of panel SAMPL02

This panel should display various TSO CLIST variables using different attribute characters, as follows:

<u>Variable</u>	<u>Attributes</u>
&ULTERM	Output, low intensity, red
&UWTERM	Output, high intensity, red
&SYSUID	Output, low intensity, green
&SYSPROC	Output, high intensity, green

These fields should be labelled by text fields (white).

Also display a text field (blue) containing the text:

```
TIME AT ENTRY TO CLIST: &UTIME1, TIME NOW: &UTIME2  
TSO VERSION AND RELEASE: &VER
```

Prompt the user to enter a string of text, including upper and lower case characters. Save this string in the field &USERIN. Remember to define an attribute character with TYPE(INPUT) and CAPS(OFF) for defining the input variable for this string.

Description of panel SAMPL03

This panel should display the following variables:

```
&USERCC  
&UTIME1  
&UTIME2  
&UTIME3  
&USERIN as is  
&USERIN with Caps on (use variable &USERIN2).
```

NOTES

1. Include the panel name on each panel.
2. Experiment with different EXPAND characters.
3. Experiment with whatever HILITE and COLOR options, your terminal supports.

This page intentionally left almost blank.

Index

Special characters

%	17
&	196
&LASTCC Control Variable	20, 132
&MAXCC Control Variable	132
&STR Built-in Function	136
&SUBSTR Built-in Function	133, 446
&SYSCSUBSTR Built-in Function	133
&SYSDSN Built-in Function	134
&SYSINDEX Built-in Function	135
&SYSNSUB Built-in Function	446
)	41
)ABC	42-43
)ABC - Action Bar Choice	609-618
)ABCINIT	42-43
)ABCINIT Section	615
)ABCPROC	42-43
)ABCPROC Section	616
)AREA	42-43
)ATTR	42-58, 65, 159, 414
)BODY	42-44, 60-63, 65-70, 159, 220-221, 248, 254, 413, 415
)CCSID	42-43
)END	42-44, 64, 159
)FIELD	42, 264-268
)HELP	42-43
)init	65
)INIT	42-43, 71-73, 160-200, 206, 208, 220-221, 420
)LIST	42
)MODEL	42-43, 360, 412-413, 416-418
)PANEL	42-43
)PANEL Section	622, 635
)PNTS	42
)PROC	42-43, 65, 71-73, 160-200, 206-208, 275-277, 295, 420, 423-424
)REINIT	42-43, 65, 71-73, 160-200, 206-208, 277, 420, 423-424
*ENDREXX Statement	162
*REXX Statement	162
¬	179-180
.ALARM Control Variable	174, 202-203, 209

.ATTR Control Variable	175
.ATTRCHAR Control Variable	175
.AUTOSEL Control Variable	420-421
.CSRPOS Control Variable	176
.CSRROW Control Variable	420-421
.CURSOR Control Variable	176
.HELP Control Variable	176, 202-203, 209, 338
.MSG Control Variable	177, 206-207, 423
.PFKEY Control Variable.	177
.RESP Control Variable	177
.TRAIL Control Variable	164-165, 178
.ZVARS Control Variable.	220-221
_	17
.	196
+	17
<<	179
=	161, 173, 179-180, 198-199
>>	179

A

Access Method Services.	114-116
Action Bar	42, 602-607
ACTION entry in PDC	611-618
Actions - ISPF command.	603-606
ADDPOP	279
ADDPOP Service	243, 245-247
ADDRESS Statement	22
Alarm (on terminal).	174
ALLOCATE.	38, 106, 226
ALTLIB Command	109-113
AMS	-See Access Method Services
AND	196
APL2	10, 14
Application ID	80-81, 84-86, 223, 279, 592
Application Profile Pool	-See Profile Pool
Applications	80-82, 84-86, 89-91, 278-279, 640-641
Assembler	10, 23
Assignment Statement.	161, 173
Assignment Statement (Panel Language)	198-199
Attribute Characters	17, 45-58, 175, 414

B

Blank Lines	65
BROWSE Service	325-329, 333
Built-in Functions	119, 121

C

C	10
Cancel - ISPF command	604, 606
CANCEL Command	271, 612
CAPS Attribute	46, 52
CASE Logic Structure	140-144
Clauses	122-123
CLIST	8-10, 12, 19-21, 83, 108-113, 132-144, 159
CLISTs	148-150
CMS	-See Conversational Monitor System
COBOL	10, 23
COLOR Attribute	46, 56
COLOR Command	612
Command / option line	602
Command Line	60, 67
Command Suppression	318, 323
Command Tables	80, 270, 592-598
Commands	591-598
Comments	65
Common User Access	602-607
CONTROL Service	318-324
Control Variables	132, 164-165, 173-179, 202-203, 206-207, 209, 220-221, 420-421, 423
Conversational Monitor System	8-9, 13
Copy	522-523
CRP	-See Current Row Pointer
CUA - Common User Access	57, 59, 601, 608-618
CUA panel format	602-604, 606
CUAATTR Comand	612
Current Row Pointer	358-359, 372, 374-381, 385-389, 392, 395, 421-422, 425, 435, 439-440
Cursor Location	176, 548-549
Cursor Select	57

D

Data set information	584-590
DATAID	222-237, 333, 520, 522, 524, 526, 529, 532-533, 535-548, 551-563
Dataset Name	120, 193
DBCS.	-See Double Byte Character Support
DELETE Command	116
Dialog Manager	-See Interactive System Productivity Facility
Dialog Tag Language	-See DTL
Dialog Testing	27-30, 158-159, 296-310, 442, 583
DIRLIST Service	573-578
DISPLAY Service	21, 30, 71-73, 159, 206, 208, 255, 272-273, 295
DO Groups	146-147
DO Instruction	124-126
DO Statement	138-139
DOCTYPE - DTL declaration	627
Double Byte Character Support	57, 133, 195
DSINFO service	584, 589-590
DSLISL.	148-150
DSLISL Command	612
Dslist-id.	564-572
DSNAME.	193
DTL - Dialog Tag Language	623-635
Dynamic Areas.	57
Dynamic Attribute Specification	50-51

E

Edit Models.	316-317
Edit Profile	80
Edit Recovery	330-332
EDIT Service	325-333
END	124
End - ISPF command	606
END Command	271, 612
EPDF Command	612
Error Handling	318-319
EXCLDATA.	36-37
EXCLLIBR	36-37
EXECs	9-10, 12, 19-21, 83, 108-113, 148-150
Exit - ISPF command	606

EXIT Command	271, 612
EXIT Statement	161
EXIT Statement (Panel Language)	184, 198-199
EXPAND	60, 62
EXPAND comand	268
Explicit Chaining	290
Expressions	122
Extension Variables	361, 375, 378-379, 381, 386, 388-391, 416, 433-435, 439-440

F

Field Padding	54
FIELD statement	264-268
File id's	192
File Tailoring	357
File Tailoring Services	470-516
FKA Command.	612
FORTTRAN	10, 14, 24
FREE Command	107
FTCLOSE Service	472-473, 512
FTERASE Service	472, 513
FTINCL Service	472-473, 511
FTOPEN Service.	472-473, 510
Full Screen Mode	281-282
Function keys	602
Function Pool	81-84, 90-93, 159
Functions.	10, 298-306

G

Generic DDnames	357
Generic libraries	34
GETMSG Service	209-210, 256
Getting around	605-607
GO TO Statement	146-147
GOTO Statement	161
GOTO Statement (Panel Language).	182-183, 198-199
Graphic Areas	57
GUI - Graphical User Interface.	57

H

Haystack	121
HELP	59, 203, 334-340
HELP Command	612
HELP Panel	176, 341-342
HILITE Attribute	53
Host Environment	118

I

IF Instruction	122-123
IF Statement	137
IF Statement (Panel Language)	161, 179-181, 196, 198-199
Index (Tutorial)	335, 341-342
Index panel (Tutorial)	337
Input Fields	17, 46-49, 63
Installing Applications	640-641
INTENS Attribute	46, 49-51, 56
Interactive System Productivity Facility	8-17, 148-150
ISPCMDS	592
ISPDTLC Command	612, 634
ISPEX	24
ISPEXEC	10, 20-23, 73, 94-97, 103-104, 159, 206, 209-213, 222-224, 227, 229, 232, 234-235, 245, 250-253, 256, 278, 294-295, 318-324, 326, 331-333, 351, 363, 365, 368-371, 375, 377-378, 380, 382-384, 387-388, 390, 393-395, 397, 421, 433, 435, 439, 510-513, 522, 524, 526-529, 532-533, 535-537, 539, 541-542, 544, 548, 552-553, 555-563, 565-567, 571, 573-578
ISPF	602-607, -See Integrated System Productivity Facility
ISPF Commands	
ACTIONS	603-606
CANCEL	604, 606, 612
COLOR	612
CUAATTR	612
DSLIST	612
END	606, 612
EPDF	612
EXIT	606, 612
EXPAND	268

FKA	612
HELP	612
ISPDTLC	612
ISPLIBD	613
ISPPREP	613
ISRROUTE	613
KEYLIST	613
KEYS.	613
LIST	613
LOG	613
PANELID	614
PFSHOW	614
PRINT	614
PRINTL.	614
RESIZE.	253
TSO	614
ISPF GUI mode	34
ISPF Table Utility.	450-461, 463-467
ISPF/PDF	148-150, 331-332
ISPFILU	19, 34, 471-472, 512-513
ISPFILU	38
ISPILIB	19, 34
ISPIUSR	38
ISPLIBD Command	613
ISPLINK	10, 23
ISPLLIB	19, 34, 36-37
ISPLNK.	24
ISPLUSR.	38
ISPMLIB	19, 34
ISPMUSR	38
ISPPLIB	19, 21, 34
ISPPREP Command.	580-590, 613
ISPPROF.	19
ISPPUSR.	38
ISPSLIB	19, 34, 471-472
ISPSTART	274, 291, 642
ISPSUSR.	38
ISPTABL	19, 34, 349, 366-367, 370
ISPTABU.	38
ISPTLIB	19, 34, 349, 354, 363
ISPTUSR.	38
ISPTUTOR.	334
ISRROUTE Comand.	613

J

Jump Function 272-273

K

Keyboard Locking 318, 321
Keyed Tables 350, 358-360, 380, 382, 387-388, 390-391
KEYI tag 630-631
KEYL tag 629
KEYLIST Command 613
Keylists 621-622
KEYS Command 613

L

Labels 146-147
LENGTH Function 163, 170
Levels (of Dataset Name) 148-150
LIBDEF Service 21, 23, 29, 34-39, 159, 357
Library Access Services 222, 224-237, 520-572
Library Management Services 223, See also Library Access Services
Light Pen 57
Line Mode 281-282
Link Pack Area 36-37
Linklist 37
LIST Command 613
LIST Service 213
LMCLOSE Service 234, 520
LMCOMP Service 531-532
LMCOPY Service 520, 522-523
LMDDISP Service 564, 571-572
LMDFREE Service 564, 566
LMDINIT Service 564-565
LMDLIST Service 564, 567-570
LMERASE Service 521, 528
LMFREE Service 235, 520
LMGET Service 229-231, 520
LMINIT Service 223-226, 333, 520

LMMADD Service	531, 537-538
LMMDEL Service	531, 541
LMMDISP Service	531, 534, 547-558
LMMFIND Service	531, 533-535
LMMLIST Service	531, 544-546
LMMOVE Service	520, 524-525
LMMREN Service	531, 536
LMMREP Service	531, 539-540
LMMSTATS Service	531, 542-543
LMOPEN Service	227-228, 520
LMPRINT Service	520, 526
LMPUT Service	232-233, 520
LMQUERY Service.	529
LMRENAME Service.	521, 527
LOG Command	613
Log File.	212
LOG Service	212
Long Message	59-60, 68, 202-204, 209, 212, 254, 319, 602
Loops.	146-147

M

Master Application Menus	289
Member Name	120
MEMLIST Service	559-563
Menus	40, 273-291, 293-295
Message id's	166, 177, 201-207, 209-212
Messages	59-60, 66, 68, 166, 201-207, 209-212, 254-256, 319, 342, 422, 602
Mnemonics.	605
Model Sets	417-419
Models	-See Edit Models

N

Needle	121
Nondisplay Mode.	318, 322
NOP Instruction	122
Null	122
Numeric Only Fields	57

O

OR	196
OTHERWISE.	127-129
Output Fields	46-49, 63, 414

P

PAD, PADC Attributes	54
Padding Characters	54
Panel Language	580-590
Panel styles	602-604, 606-607
PANELID Command	614
Panels	10, 12-13, 15-18, 21, 40-73, 160-200, 208, 295, 602-607
PANEXIT Statement	162
Pascal	10
PDC - Pull Down Choice	610-618
PF Keys	12, 169, 177
PFK Function.	163, 169, 198-199
PFSHOW Command.	70, 419, 614
PL/I	10, 23
Placeholder Variables	220-221
Pools	-See Variable Pools
Pop-up Windows	-See Windows, ADDPOP, REMPOP
Pop-ups	251-253
POS Built-in Function	121
Post-test	125, 138
Preprocess Utility	581-582
Preprocessed Panels	580-590
Pre-test.	125, 138
Primary Option Menu	27, 288, 339
PRINT Comand	614
PRINTL Command.	614
Profile Pool	80-82, 84-97, 103-104, 159, 643
Protected Fields	48
Pull-down menu	603-606

Q

QBASELIB service	584, 587-588
QLIBDEF service.	584-586
QTABOPEN Service	392-394

R

RC Special Variable	20, 118
REFRESH Statement	161, 185-186, 198-199
REMPop	279
REMPop Service	243-244, 250
REPRO Command	114-115
RESIZE Command.	253
RESTORE Logical Screen.	324
Return Codes	118
RETURN Command	271
Review	159
REXX.	8-10, 12, 108, 118-119, 121-129
Rowid.	359, 384, 421

S

SAVE Logical Screen	324
Scrollable fields	260-268
Scrolling	413, 415, 419, 423
SELECT Instruction	127-129
SELECT Service	274-289, 292, 294-295
SELECT Statement	140-144
Selection Panels	40, 274-289, 291, 293
Separator line	602
SETMSG Service	211, 256
Settings.	602
Shared Pool	81-82, 84, 88-97, 103-104, 159
Short Message.	59-60, 66, 202-204, 209, 212, 254, 319, 602
Skeletons.	471-472, 474-516
SKIP Attribute	55
Sorted Tables	371-373, 375, 388, 390
Split Screen	70, 90, 318, 320, 419

Strings	119, 121
SUBSTR Built-in Function	119, 444
Summary	159
Symbolic Variables	132
SYSDSN TSO/E Function	120
SYSEXEC	19, 108
SYSPROC	19, 108, 110
System Profile Pool	89, 91
System Symbols	99-102, 162-163, 192, 197, 506
System Variables	92-93, 98
SYSUEXEC	108, 110
SYSUPROC	108, 110

T

Tab Characters	477
Table Display	40
Table Display Services	412-432, 441-443
Table of Contents (Tutorial)	335, 341-342
Table Utility	-See ISPF Table Utility
Tables	348-402, 412-446, 516
TBADD Service	358, 360, 372-376, 390
TBBOTTOM Service	359, 374, 378-379
TBCLOSE Service	352-354, 359, 362, 368, 372
TBCREATE Service	351-357, 360, 362, 364
TBDELETE Service	374, 387
TBDISPL Service	360, 374, 412-413, 419-432, 437
TBEND Service	352, 354, 359, 362, 369
TBERASE Service	362, 370
TBEXIST Service	374, 382
TBGET Service	359, 374, 380-381
TBMOD Service	360, 372-374, 390-391
TBOPEN Service	354, 362-364, 372
TBPUT Service	360, 372-374, 388-389
TBQUERY Service	392, 395-396
TBSARG Service	359, 374, 416, 432-439
TBSAVE Service	352-353, 362, 365-368, 372
TBSCAN Service	359, 374, 432, 434, 437, 439-440
TBSKIP Service	359, 374, 384-386
TBSORT Service	358, 362, 371-373, 375
TBSTATS Service	392, 397-400
TBTOP Service	359, 374, 377

TBVCLEAR Service	374, 383
TEST, TESTX	158
Text Fields	17, 47-48, 63
Text Justification	57
Time Sharing Option	8-9, 13, 106-116, 120, 159
TOG Statement	161, 187-188, 198-199
TRACE, TRACEX	158
Tracing	158, 296-310
TRANS Function	163, 166-168, 195, 198-199, 206, 283-284, 295
TRUNC Function	163-165, 168, 178, 195, 198-199, 283-284, 295
TSO	-See Time Sharing Option
TSO Command	614
TSO/E Functions	120
Tutorial	203
Tutorial Panel	40, 176, 334-342
TYPE Attribute	46-48

U

UNTIL	124-126
UPPER Function	163, 171
User Libraries	38-39

V

Variable Pools	10, 80-91, 94-97, 103-104, 159, 197, 279, 291, 298-300, 303
Variables	14, 17, 21, 48, 80-104, 159, 220-221, 298-300, 303, 360-361, 374, 380, 382, 390, 444-446
VEDIT Statement	162
VER Statement.	162, 189-195, 198-199, 206
VERASE	88, 103-104, 159
VGET.	84, 86, 90-91, 94-95, 98-102, 159
VGET (Panel Language).	162, 197-199
VIEW Service	325-330, 333
VPUT.	84, 86, 90-91, 96-98, 159
VPUT (Panel Language).	162, 197-199
VSAM Files.	114
VSYM (panel language)	197
VSYM Function	163, 172
VSYM Statement (panel language)	162

W

WHEN	127-129
WHILE	124-126
WINDOW Command	253
Windows	242-256, 280

Z

Z Variables	87, 89, 92-93, 98
z/OS UNIX files	329
ZCMD	67, 275-276, 336, 548-549
ZCONT	337
ZCTACT	593
ZCTDESC	593
ZCTTRUNC	593
ZCTVERB	593
ZDATE	83, 92
ZDAY	92
ZDSN	550
ZERRALRM	319
ZERRHM	319
ZERRLM	319
ZERRMSG	319
ZERRSM	319
ZHINDEX	339, 341
ZHTOP	339, 341
ZIND	337
ZKEYS	92
ZLCDATE	534
ZLCNORC	534
ZLINORC	534
ZLLCMD	548-550
ZLLIB	534
ZLMDATE	534
ZLMEMBER	550
ZLMNORC	534
ZLMOD	534
ZLMTIME	534
ZLMTOP	550
ZLUDATA	548-550

ZUSER	534
ZLVERS	534
ZMLCOLS	550
ZMLCR	550
ZMLTR	550
ZMONTH	92
ZPANELID	92
ZPFLnn	92
ZPFnn	92
ZPREFIX	92
ZPRIM	288
ZPROFAPP	643
ZSCROLLD	415
ZSEL	275-277, 283, 287, 294-295, 336
ZSTDYEAR	92
ZTDMARK	431
ZTDMSG	431
ZTDROWS	431
ZTDSELS	425-428, 431
ZTDTOP	425, 429, 431
ZTDVROWS	431
ZTEMPF	472, 510
ZTEMPN	472, 510
ZTIME	83, 92
ZUP	340
ZUSER	92
ZWINTTL	247
ZYEAR	92

This page intentionally left almost blank.