IBM Cloud 9 for SCLM for z/OS

Installation Guide

Version 2 Release 1

SC31-8845-01

IBM Cloud 9 for SCLM for z/OS

Installation Guide

Version 2 Release 1

SC31-8845-01

Note
Before using this document, read the general information under "Notices" on page 135.

Second Edition (June 2002)

This edition applies to Version 2 Release 1 of the licensed program IBM Cloud 9 for SCLM for z/OS (program number 5655-G93) and to all subsequent releases and modifications until otherwise indicated in new editions.

Order publications by phone or fax. IBM Software Manufacturing Solutions takes publication orders between 8:30 a.m. and 7:00 p.m. eastern standard time (EST). The phone number is (800) 879-2755. The fax number is (800) 284-4721.

You can also order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

If you would like to make comments about this publication, address them to:

IBM Corporation
Department J46A/G4
555 Bailey Ave
San Jose, CA 95141-1099 U.S.A.

or fax your comments from within the U.S.A.,
to 800-426-7773, or, from ouside the U.S.A., to 408-463-2629.

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:

Title and order number of this book
Page number or topic related to your comment

Contents

Figures

Tables

About This Book

This manual contains the installation procedure for the IBM Cloud 9 for SCLM for z/OS product, which combines standard z/OS installation procedures with Unix System Services (USS) and IBM HTTP server configuration.

There are also sections on enabling the Cloud 9 S-FTP and S-JDK Java Development Kit components.

Who Should Use This Book

This manual is written for systems programmers who will be configuring and administering the Cloud 9 web server. Readers should be familiar with the Unix System Services (USS) environment, Hierarchical File System (HFS) structure, Resource Access Control Facility (RACF) profiles needed to support USS and started tasks (or the equivalent for the installed security product), and the IBM HTTP Server.

It also contains information to be used by the administrator of any SCLM projects that will be using the Java and USS component languages. These administrators also need to be familiar with the USS environment and HFS structures, REXX Script, and the Java Compiler and SCLM project and language definitions.

Where to Find More Information

Where necessary, this book references information in other books, using shortened versions of the book title. For complete titles and order numbers of the books for all products that are part of z/OS, see *z/OS Information Roadmap* (GC28-1727). Direct your request for copies of any IBM publication to your IBM representative or to the IBM branch office serving your locality.

There is also a toll-free customer support number (1-800-879-2755) available Monday through Friday from 6:30 a.m. through 5:00 p.m. Mountain Time. You can use this number to:

- Order or inquire about IBM publications
- Resolve any software manufacturing or delivery concerns
- Activate the program reorder form to provide faster and more convenient ordering of software updates

Hardcopy Publications

Short Title Used in This Book	Title of Publication	Order Number
HTTP Server Guide	IBM HTTP Server for OS/390 HTTP Server Planning, Installing, and Using	SC31-8690-xx
USS Planning	OS/390 UNIX System Services Planning	SC28-1890-xx
USS Messages	OS/390 UNIX System Services Messages and Codes	SC28-1908-xx

Short Title Used in This Book	Title of Publication	Order Number
USS Commands	OS/390 UNIX System Services Command Reference	SC28-1892-xx
SCLM Project Manager's Guide	Interactive System Productivity Facility (ISPF) Software Configuration and Library Manager (SCLM) Developer's and Project Manager's Guide	SC34-4750-xx
SCLM Reference	Interactive System Productivity Facility (ISPF) Software Configuration and Library Manager (SCLM) Reference	SC28-1320-xx
Breeze Installation Guide	IBM Breeze for SCLM for z/OS Installation Guide	SC31-8819-01

Softcopy Publications

The z/OS library is available on the z/OS Collection Kit, SK2T-6700. This softcopy collection contains a set of z/OS and related unlicensed product books. The CD-ROM collection includes the IBM Library Reader, a program that enables customers to read the softcopy books.

Softcopy z/OS publications are also available for web browsing. PDF versions of the z/OS publications for viewing or printing using Adobe Acrobat Reader are available at these URLs:

```
http://www.ibm.com/s390/os390/
http://www.ibm.com/servers/eserver/zseries/zos
```

Select "Library."

IBM Systems Center Publications

IBM systems centers produce redbooks that can be helpful in setting up and using z/OS UNIX System Services. You can order these publications through normal channels, or you can view them with a web browser from this URL:

```
http://www.redbooks.ibm.com
```

These books have not been subjected to any formal review nor have they been checked for technical accuracy, but they represent current product understanding (at the time of their publication) and provide valuable information on a wide range of z/OS topics. You must order them separately. A selected list of these books follows:

Title of Publication	Order Number	Comments
P/390, R/390, S/390 Integrated Server: OS/390 New User's Cookbook	SG24-4757-01	Despite the title, it is oriented towards the systems programmer, and describes considerations for the Unix System Services environment
Debugging UNIX System Services, Lotus Domino, Novell Network Services, and other Applications on OS/390	SG24-5613-00	Provides an overview of the Unix System Services environment along with tips and suggestions for setup and problem analysis.

Title of Publication	Order Number	Comments
OS/390 e-business Infrastructure: IBM HTTP Server 5.1 - Customization and Usage	SG24-5603-00	Provides an overview of web servers in general with specific details for the OS/390 server along with hints and tips for setup and customization
Debugging Unix System Services	SG24-5613-00	
e-business Enablement Cookbook for OS/390 Volumes 1,2, and 3	SG24-5664-00 SG24-5981-00 SG24-5980-00	

Part 1. Basic Cloud 9 Installation

Chapter 1. Installation Overview

This manual contains the installation procedure for the IBM Cloud 9 for SCLM for z/OS product. The procedure is a combination of standard z/OS install procedures, Unix System Services HFS file set up, and IBM HTTP server configuration. Hereinafter, the following names will be used in this manual:

- IBM Cloud 9 for SCLM for z/OS will be called ***Cloud 9***.
- The IBM z/OS HTTP server will be called ***the HTTP server***.
- Unix System Services and HFS will be called ***USS***.
- IBM Breeze for SCLM for z/OS will be called ***Breeze***.

The steps in this manual are organized into five major sections:

- Before you begin
- Create and populate libraries
- Create demo database and product initialization files
- Configure HTTP server components
- Perform Installation Verification Procedures (IVP).

SMP/E Installation

This manual does not cover the implementation aspects of Cloud 9. Rather, it is intended to guide the installer through a successful configuration of the major components.

This manual assumes that the System Modification Program/Extended (SMP/E) installation of Cloud 9 has been completed. The SMP/E instructions for Cloud 9 are in the *IBM Cloud 9 for SCLM for z/OS Program Directory*, GI10–3199.

Separate SCLM Installation

This manual does not cover the implementation and loading of the SCLM product, which is the source of data to the Cloud 9 interface.

Modifying Case Sensitive JCL Members and USS Files

During this installation, you will modify several JCL members and USS files. Certain JCL members and all USS files contain case sensitive values. It is imperative that *before* modifying the JCL and USS members, you issue the CAPS OFF command to ensure that automatic upper casing does not occur. You will be reminded of this case sensitivity issue where appropriate throughout this manual.

Product Components Utilized During Installation

The following JCL, Parameter, and HTML members are modified during the installation process. The names are provided here as an overview of naming standards and component functionality.

JCL Members (located in SCLZJCL and resident on the host) Modified During Installation:

CLZC9SRV JCL for the HTTP server task

CLZJUNIX	JCL to copy members to Cloud 9 rootdir
CLZCHMOD	REXX input to CLZJUNIX to reset permissions on files copied to rootdir in CLZJUNIX
CLZC9J01	JCL to build and initialize Suite Long Name Registry (SLR) database
CLZC9J03	JCL to create and initialize empty SLR database
CLZC9J04	JCL for SLR Backup, Delete, and Define
CLZC9J05	JCL for standalone VSAM index expansion
CLZC9J06	JCL for SLR Installation Verification Procedure (IVP)
CLZC9JS4	JCL to build the CIGINI file
CLZJMIG	JCL shell for Endevor conversion
CLZJIBM	JCL shell for SCLM actions
CLZJDYN	REXX shell for SCLM dynamic allocations

HTTP Server Parameters Modified During Installation and Copied to Cloud 9 rootdir

CLZCNFG	Sample HTTP server **httpd.conf** file
CLZEVARS	Sample HTTP server **httpd.envvars** file

Chapter 2. A Step-by-Step Approach

Table 1. Installation Steps

Before you begin...	
♦	Review system, software, and hardware considerations.
♦	Implement site standards.
CP1.	Verify steps as shown in "CHECKPOINT #1" on page 11.
Create Product Initialization Module	
1.	Allocate and initialize SLR Long Name Registry Database
2.	Set up the CIGINI initialization file.
CP2.	Verify steps as shown in "CHECKPOINT #2" on page 21.
Configure USS and HTTP Server Components	
3.	Modify rules file member CLZHTTPD
4.	Modify environment variable member CLZEVARS
5.	Authorize loadlib, review and customize the Cloud 9 server JCL
CP3.	Verify steps as shown in "CHECKPOINT #3" on page 39.
6.	Create root directories, copy product to HTTP Unix directories, and set USS file permissions.
7.	Ensure that permission authorization is turned on for the SCLM interface module, CLZRSDRV.
Perform Installation Verification Procedures	
8.	Run the Cloud 9 server invocation IVP.
CP4.	Verify steps as shown in "CHECKPOINT #4" on page 46.
9.	Run the Cloud 9 invocation and logon IVP.
10.	Run the profile setup IVP.
11.	Run the Cloud 9 batch and interactive IVPs.
12.	Run the SLR batch IVP.
13.	Set up the Backup, Delete, and Define JCL for the SLR.
CP5.	Verify steps as shown in "CHECKPOINT #5" on page 55.

Chapter 3. Before You Begin

Review Software and Hardware Considerations

In this step you will review the system, software, and hardware requirements for product installation.

System Requirements

To successfully install Cloud 9, the following system requirements must be in place at your installation:

Table 2. System Requirements

z/OS Operating System	Version 2 Release 7 (or higher)
IP address	Numerical IP address of host or named server on host
Port number	1024 or higher*
Product Load Library	Must be an APF authorized load library
Web browser	Microsoft Internet Explorer 5.0 or higher Netscape Navigator 4.7 or higher
*This port number must be higher than 1024, as port numbers lower than this are reserved for internal system services.	

Software Requirements

Cloud 9 requires that SCLM be implemented for at least one project on the z/OS. Contact your systems administrator to ensure that these requirements are in place.

VSAM Exclusion

This product must be excluded from all VSAM buffering products. This should be done on a global basis. Failure to exclude SCLM Suite databases may result in file corruption.

Product Space Requirements

Table 3 outlines the space requirements for the Cloud 9 product software and supporting libraries. Note that the following estimates are based on 3390 track allocations.

Table 3. Space Requirements

Library Type	Total Space Required
Target	131 Tracks on 3390
Distribution	143 Tracks on 3390
HFS	30 Tracks on 3390

Implement Site Standards

Site-Specific Placeholders

The following placeholders represent values that are customer-specific.

- *dvolser*
- *dunit*
- *tdisk*
- *ispfqual*
- *password*
- *WEBJOBNAME*
- *user1*
- *user2*
- *portno*
- *rootdir*
- *ip-address*

These placeholders (see "Cloud 9 Installation Worksheet" on page 9 for definitions) are indicated in this chapter by the use of lowercase italics in the reproduced JCL. Substitute your site-specific values in all installation and implementation JCL. The value for placeholder "password" is provided for you, it is the word ***password***. The value for "password" is case insensitive. Complete the third column on the Placeholder worksheet for easy reference during installation.

ISPF/SCLM Data Set Names

Additionally, identify the data set names for your current Interactive System Productivity Facility (ISPF) data sets as per the "Data Set Worksheet" on page 9. The current ISPF data set names will be needed for the symbolic procedures used in batch.

Cloud 9 Installation Worksheet

Throughout the rest of the Cloud 9 installation process, you will be asked to supply site-specific values for JCL and parameter modifications. The worksheet below enables you to record these values in one place for easy reference.

Table 4. Placeholder Worksheet

Place Holder	Definition	Your Site Value
dvolser	Volume serial number of the disk used to store permanent data sets (if needed).	
dunit	Unit label for permanent disk data sets (usually SYSDA). This specification is limited to 6 characters.	
tdisk	Unit label for temporary disk data sets (usually SYSDA). This specification is limited to 6 characters.	
ispfqual	High-level qualifier for the standard ISPF data sets.	
password		**password**
WEBJOBNAME	Job name of the HTTP server task. Also used in the ***httpd.conf*** file. This must be a value in upper-case.	
user1	User Id 1 for building IVP profile members	
user2	User Id 2 for building IVP profile members	
portno	TCP/IP Port number for Cloud 9 invocation	
rootdir	Root directory for Cloud 9 HTTP Server	Through the SMP/E installation this value is set to **/PathPrefix/usr/lpp/Cloud9/** Refer to "SMP/E Unix Considerations" on page 10 for more information.
ip-address	Internet Protocol address for the HTTP server	

Data Set Worksheet

Cloud 9 relies on SCLM services to perform many of the Cloud 9 request functions. SCLM requires a proper ISPF environment to be established. This means that the Cloud 9 JCL and dynamic allocation routines must have the actual names of the ISPF data set names used at your installation. Use the table following to identify the names of the actual ISPF data sets used at your installation. These are needed during *Step 5C, Modify Batch Shells*, were you will modify the various USS JCL shell files.

Table 5. Data Set Worksheet

DDNAME	ISPF Data Set Examples	Your ISPF Data Set Names
SYSPROC	ISP.SISPCLIB	
ISPMLIB	ISP.SISPMENU	
ISPPLIB	ISP.SISPPENU	
ISPSLIB	ISP.SISPSLIB ISP.SISPENU	

SMP/E Unix Considerations

A portion of the Cloud 9 SMP/E install created and populated Unix directories based on a ***PathPrefix*** variable. If the default values are used by your installation, your **rootdir** value will be equal to the following:

```
/PathPrefix/usr/lpp/Cloud9/
```

Your system administrator can provide more information.

CHECKPOINT #1

At this point the following libraries should be allocated and populated. Using ISPF Option 3.4, verify that these files have been created and contain data.

Table 6. Checkpoint #1

Default Data Set Names	Your Data Set Names	Completed?
CLZ.SCLZDMDB		
CLZ.SCLZHTML		
CLZ.SCLZJCL		
CLZ.SCLZLOAD		
CLZ.SCLZPRF		
CLZ.SCLZCGI		
CLZ.SCLZPDF		

Chapter 4. Create Product Initialization Module

Step 1: Allocate and Initialize an SLR Database

In this step, you will allocate and initialize the Suite Long Name Registry (SLR) long name support database. Long name support helps when moving, viewing, and referencing objects from one platform to another. Many non-host objects have names greater than 8 characters, including the extension, and some of them are case-sensitive.

The SLR database is where the correlation between the distributed platform object name and the standard host OS eight-character name is maintained. It is referenced in the CIGINI file.

The purpose of the file at this point of the install is for the IVP process.

Modify and Submit CLZC9J01

To create this database perform the following tasks:

1. Using ISPF EDIT, access member CLZC9J01 in the SCLZJCL data set.
2. Copy your job card values to the top of the member.
3. Substitute your site-specific values (identified on the "Cloud 9 Installation Worksheet" on page 9) as per the instructions in the comment area of the JCL.
4. Submit the job.

Note: Note that this job should terminate with COND CODE=8 for the delete function the first time and then COND CODE=0. If it does not:

1. Review your job card parameters and the JCL for errors.
2. Resubmit the job.

CLZC9J01 JCL

```
//**(JOBCARD)
//**
//*********************************************************************
//*                                                                   *
//* CLZC9J01 - THE PURPOSE OF THIS JCL IS TO CREATE A SLR DATABASE    *
//*            FOR IVP PURPOSES.                                      *
//*********************************************************************
//*                                                                   *
//* REQUIRED JCL MODIFICATION:                                        *
//*   1) INCLUDE A JOBCARD                                            *
//*   2) CHANGE THE FOLLOWING AS PER THE INSTALLATION WORKSHEET.      *
//*      - VOLUMES(DVOLSER)                                           *
//*                                                                   *
//* THE FOLLOWING MAY NOT REQUIRED FOR SMS INSTALLATIONS:             *
//*   - VOLUMES(DVOLSER)                                              *
//*                                                                   *
//*********************************************************************
//*                                                                   *
//* DO NOT MODIFY THE VSAM PARAMETERS PROVIDED IN THIS JCL. DOING SO *
//* WILL PRODUCE UNEXPECTED RESULTS FROM THE CLOUD9 APPLICATION.      *
//*                                                                   *
//*********************************************************************
//*                                                                   *
//*  STEP 1: ALLOCATE THE SLR VSAM DATABASE AND REPRO THE RECORDS     *
//*          FROM THE INDD01 FILE.                                    *
//*                                                                   *
//*********************************************************************
//STEP1   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//INDD01   DD DSN=CLZ.SCLZDMDB(CLZDEMO),DISP=SHR
//SYSIN   DD *
 DELETE CLZ.SCLZSLR.DATABASE
 DEFINE CLUSTER -
    (NAME('CLZ.SCLZSLR.DATABASE') -
    IMBED SPEED UNIQUE FREESPACE(30 30) -
    VOLUMES(DVOLSER) TRACKS(60 40) -
    SHR(4 3) -
    KEYS(254 0) -
    RECORDSIZE(512 1024)) -
  DATA  (CISZ(16000)) -
  INDEX (CISZ(4096))
 REPRO INFILE(INDD01) OUTDATASET('CLZ.SCLZSLR.DATABASE')
/*
```

Figure 1. CLZC9J01 JCL

Step 2: Set Up the CIGINI Initialization File

In this step you will create the CIGINI member, a file in text format (contains data only, no executable code) that contains various product parameters such as product password, database names, and the product load library name. For test purposes we will create a new version of this file.

Modify and Submit CLZC9JS4

The CIGINI load module must be located in the Cloud 9 steplib or linklist area. Create the CIGINI load module by executing the JCL in member CLZC9JS4 of the SCLZJCL data set. As input to the job, you will need to do the following:

1. Using ISPF EDIT, access member CLZC9JS4 in the SCLZJCL data set.
2. Copy your job card values to the top of the member.

3. Substitute your site-specific values (identified on the "Cloud 9 Installation Worksheet" on page 9) as per the instructions in the comment area of the JCL.

   Note: If the Cloud 9 Java component is to be used in the same SCLM project as Breeze, then the keywords for the Breeze CIGINI generation need to be included here. For more information about the IBM Breeze for SCLM for z/OS product, refer to the *Breeze Installation Guide*, SC31–8819.
4. Update Cloud 9 password.
5. Verify that the SYSLMOD points to the intended execution library.
6. Submit the job.

Note: Note that this job should terminate with COND CODE=0. If it does not:

1. Review your job card parameters and the JCL for errors.
2. Resubmit the job.

CLZC9JS4 JCL and Input

```
//* (JOBCARD)
//* ------------------------------------------------------------------*
//*  NAME:    CLZC9JS4                                                 *
//*  PURPOSE: PARSE, COMPILE AND LINK THE CIGINI MODULE.               *
//*                                                                    *
//* ------------------------------------------------------------------*
//*  TO USE THIS JCL, YOU MUST:                                        *
//*                                                                    *
//*     1.     PERFORM MODIFICATION ON THE CIGINI STATEMENTS.          *
//*            THIS SAMPLE WILL NOT COMPILE AS DELIVERED.              *
//*     2.     INSERT A VALID JOB CARD WITH A VALID CLASS              *
//*     3.     CHANGE THE UNIT=TDISK TO THE APPROPRIATE UNIT           *
//*            NAME FOR TEMPORARY FILES.                               *
//*     4.     MAKE SURE THE SYSLMOD POINTS TO THE INTENDED            *
//*            EXECUTION LIBRARY.                                      *
//*                                                                    *
//* 11NOV2001 RMCC - APAR OW52105 CHANGES FOR BREEZE CO-EXISTENCE      *
//*                                                                    *
//*--------------------------------------------------------------------*
//*                                                                    *
//* STEP 1: PARSE CIGINI SYNTAX.  BUILD INPUT FOR ASSEMBLER.           *
//*                                                                    *
//*--------------------------------------------------------------------*
//PARSE    EXEC PGM=CLZMPILE
//STEPLIB  DD  DSN=CLZ.SCLZLOAD,DISP=SHR  <--- CLOUD 9 LIBRARY
//CIGIN    DD  *
* ------------------------------------------------------------------ *
*   .   COMMON SECTION                                               *
*                                                                    *
*       PLEASE MODIFY TO MEET YOUR NAMING STANDARDS.                 *
*                                                                    *
*     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!   *
*     ! NOTE: THERE ARE TWO PRODUCT LOADLIB STATEMENTS IN         !   *
*     ! THE INPUT. THIS IS BECAUSE, THE CLOUD9 SERVER REQUIRES    !   *
*     ! AN AUTHORISED LOADLIB. IF THE DATASET USED IN THE SERVER  !   *
*     ! JCL IS DIFFERENT THAN THE INSTALL LIBRARY, THE CIGINI     !   *
*     ! WILL HAVE TO BE ASSEMBLED POINTING TO THE AUTHORISED      !   *
*     ! LIBRARY.                                                  !   *
*     !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!   *
*                                                                    *
* ------------------------------------------------------------------ *
DEFINE COMMON SECTION
  PRODUCT LOADLIB     = 'CLZ.SCLZLOAD'
* PRODUCT LOADLIB     = 'CLZ.SCLZLOAD'
  WORK UNIT           = TDISK
  VIO  UNIT           = TDISK
  DO NOT ALLOW ALTERNATE CIGINI FILE
```

Figure 2. CLZC9JS4 JCL and Input (Part 1 of 3)

```
* ------------------------------------------------------------------ *
*   .   BREEZE INPUT TO COMMON SECTION                               *
*                                                                    *
*       THIS IS BREEZE SPECIFIC CIGINI INPUT,                        *
*       PLEASE MODIFY TO MEET YOUR NAMING STANDARDS.                 *
*       IF YOU ARE NOT USING BREEZE THEN REMOVE THESE                *
*       TWO STATEMENTS BELOW.                                        *
*                                                                    *
* ------------------------------------------------------------------ *

  JAVASERVERCONTROL DSNAME = 'BZZ.SBZZJAVA'
                    MEMBER =  BZZ$CNTL

* ------------------------------------------------------------------ *
*   .   CLOUD9 SECTION                                               *
*                                                                    *
*       THIS IS CLOUD 9 SPECIFIC CIGINI INPUT,                       *
*       PLEASE MODIFY TO MEET YOUR NAMING STANDARDS.                 *
*                                                                    *
* ------------------------------------------------------------------ *
DEFINE CLOUD9 SECTION
  PASSWORD = 'PASSWORD'
  SLRVSAM DSNAME = 'CLZ.SCLZSLR.DATABASE'
* ENDEVORBRIDGE

* ------------------------------------------------------------------ *
*   .   BREEZE BRSCLM SECTION                                        *
*                                                                    *
*       THIS IS BREEZE SPECIFIC CIGINI INPUT,                        *
*       PLEASE MODIFY TO MEET YOUR NAMING STANDARDS.                 *
*       IF YOU ARE NOT USING BREEZE THEN REMOVE THIS SECTION.        *
*                                                                    *
* ------------------------------------------------------------------ *
DEFINE BRSCLM  SECTION
  PASSWORD = 'PASSWORD'
  VSAM DSNAME = 'BZZ.SBZZPKG.DATABASE'
/*
//CIGPUNCH DD  DSN=&&TEMP,DISP=(NEW,PASS),
//            UNIT=TDISK,SPACE=(10,10),
//            DCB=(BLKSIZE=3120,LRECL=80,RECFM=FB)
//CIGLOG   DD  SYSOUT=*
//*-------------------------------------------------------------------*
//*                                                                   *
//* STEP 2: ASSEMBLE THE CIGINI INPUT CREATED IN STEP 1.              *
//*                                                                   *
//*  NOTE: CHOOSE THE DESTINATION OF YOUR CIGINI MODULE.              *
//*                                                                   *
//*-------------------------------------------------------------------*
//ASM      EXEC PGM=ASMA90,
//             REGION=3072K,
//             COND=(0,NE),
//             PARM='NODECK,OBJECT,NOTERM,LIST,XREF(SHORT)'
//SYSIN    DD  DSN=&&TEMP,DISP=(OLD,DELETE)
//SYSLIB   DD  DSN=SYS1.MACLIB,DISP=SHR
//SYSLIN   DD  DSN=&&SYSLIN,
//             UNIT=TDISK,SPACE=(TRK,(3,5)),
//             DISP=(NEW,PASS,DELETE),
//             DCB=(RECFM=FB,LRECL=80,BLKSIZE=3200)
//SYSPUNCH DD  DUMMY
//SYSUT1   DD  UNIT=TDISK,SPACE=(TRK,(5,15))
//SYSPRINT DD  SYSOUT=*
```

Figure 2. CLZC9JS4 JCL and Input (Part 2 of 3)

```
//*-------------------------------------------------------------------*
//*                                                                   *
//* STEP 3: LINK EDIT THE CIGINI MODULE                               *
//*                                                                   *
//*  NOTE: CHOOSE THE DESTINATION OF YOUR CIGINI MODULE. IF YOU ARE   *
//*        PLANNING ON USING AN ALTERNATE CIGINI MODULE, YOU MUST     *
//*        FIRST BUILD A CIGINI THAT RESIDES IN A STEPLIB LIBRARY.    *
//*-------------------------------------------------------------------*
//LINK     EXEC PGM=IEWL,
//             REGION=2048K,
//             PARM='LIST,NCAL,XREF,LET,RENT,REUS',
//             COND=(0,NE)
//SYSPRINT DD  SYSOUT=*
//SYSLIN   DD  DSN=&&SYSLIN,
//             DISP=(OLD,DELETE,DELETE)
//SYSLMOD  DD  DSN=CLZ.SCLZLOAD(CIGINI),   <--- LOCATION OF CIGINI MODUL
//             DISP=SHR
//SYSUT1   DD  UNIT=TDISK,SPACE=(TRK,(5,15))
```

Figure 2. CLZC9JS4 JCL and Input (Part 3 of 3)

Define Common Section

This section is always required. The COMMON Section describes parameters required by all products.

Table 7. Define Common Section

Syntax	Purpose	Usage
PRODUCT LOADLIB = *CLZ.SCLZLOAD*	Defines the name of the product load library. Default: None	Required
WORK UNIT = tdisk	Defines DASD unit name for temporary disk files. Default: None	Required
VIO UNIT = tdisk	Defines DASD unit name for temporary disk files in those situations where the product can take advantage of VIO disk access.	Required
JAVASERVERCONTROL DSNAME = 'BZZ.SBZZJAVA' MEMBER = BZZ$CNTL	Defines the dataset and member that contains the Java control datasets required by Breeze. Substitute the dsname parameter with the file name at your location.	Required if Breeze for SCLM is being installed also.

Define Cloud 9 Section

Table 8. Define Cloud 9 Section

Syntax	Purpose	Usage
PASSWORD = password	This required keyword and variable are checked during invocation of the product. For the IBM version of this product, specify PASSWORD=PASSWORD. Default: None	Required
SLRVSAM DSNAME = 'CLZ.SCLZSLR.DATABASE'	This optional keyword and variable is checked when transferring files from and to the browser. The SLR is for supporting long names for distributed types.	Optional.
ENDEVORBRIDGE	This optional keyword is used when the user will be converting from CA-Endevor to SCLM.	Optional.

Define Breeze Section

Table 9. Define Breeze Section

Syntax	Purpose	Usage
PASSWORD = password	This keyword and variable are checked during invocation of the product. For the IBM version of this product, specify PASSWORD = 'PASSWORD'. Default: None	Required if Breeze for SCLM is being installed.
VSAM DSNAME = 'BZZ.SBZZPKG.DATABASE'	This keyword contains the name of the Breeze for SCLM VSAM database. Substitute the file name here with the file name in your location.	Required if Breeze for SCLM is being installed.

Modify and Submit CLZC9TST

This program is an IVP that will test the following:

- Test that the target LOAD library is an APF-AUTHORIZED library.
- Check that access is possible to TCP/IP.
- Check if a REXX runtime library or the REXX alternate library is available.

To run the job perform the following tasks:

1. Using ISPF EDIT, access the member in the SCLZJCL target library.
2. Copy your job card values to the top of the member
3. Substitute your site-specific values (identified on the "Cloud 9 Installation Worksheet", on page 9) as per the instructions in the comment area of the JCL member.
4. Submit the job.

Note: Note that this job should terminate with COND CODE=0. If it does not:

1. Review your job card parameters and the JCL member for errors.

2. Resubmit the job

```
//**(JOBCARD)
//**
//*********************************************************************
//*                                                                   *
//* NAME     - CLZC9TST                                               *
//* PURPOSE  - THE PURPOSE OF THIS JCL IS TO RUN THREE ENVIRONMENTAL  *
//*            DIAGNOSTICS FOR CLOUD 9.                               *
//*                                                                   *
//*            THIS JOB WILL :-                                       *
//*            1.   TEST THAT THE TARGET LOAD LIBRARY IS AN           *
//*                 APF-AUTHORIZED LIBRARY.                           *
//*            2.   CHECK THAT ACCESS IS POSSIBLE TO TCP/IP.          *
//*            3.   CHECK IF A REXX RUNTIME OR THE  REXX ALTERNATE    *
//*                 LIBRARY IS AVAILABLE.                             *
//*                                                                   *
//* 20NOV2001 RMCC - APAR OW52105 IMPROVE COMMENTS                    *
//*********************************************************************
//*                                                                   *
//* REQUIRED JCL MODIFICATION:                                        *
//* 1.   INCLUDE A JOBCARD                                            *
//* 2.   MAKE SURE THAT THE STEPLIB IS EXACTLY THE SAME AS THE ONE    *
//*      USED IN THE CLOUD 9 SERVER JOB.                              *
//*                                                                   *
//*********************************************************************
//*                                                                   *
//*  STEP 1: PERFORM ENVIRONMENTAL TESTS.                             *
//*                                                                   *
//*********************************************************************
//STEP1   EXEC PGM=CLZATEST
//STEPLIB  DD DSN=CLZ.SCLZLOAD,DISP=SHR
//         DD DSN=TCPIP.SEZATCP,DISP=SHR
//* DD DSN=REXX.V1R3M0.SEAGALT,DISP=SHR  <-- REXX ALTERNATE LIBRARY
//CIGRPT  DD SYSOUT=*
```

Figure 3. CLZC9TST JCL and Input

CHECKPOINT #2

At this point the SLR VSAM database should be created and populated and the CIGINI initialization module should be created and stored in the product load library.

Table 10. Checkpoint #2

Task	Completed?
Allocate and initialize the SLR database?	
Build a CIGINI file that points to the demo database?	
Run IVP test program CLZC9TST?	

Chapter 5. Configure USS and HTTP Server Components

Preparation

Before you begin the configuration of the HTTP Server parameters and JCL, it is important to review a few key topics.

HTTP Stand-alone Server

This HTTP server example is delivered as a stand-alone HTTP server. Your installation might already have an HTTP server running, and you might want to merge the Cloud 9 application into the active HTTP configuration. This can be done, but you should not attempt it without the cooperation of the HTTP server administrator.

Sample HTTPD Configuration Files

The CLZHTTPD and CLZEVARS examples are set as default. They are minimally configured for Cloud 9 only. They should not be used as is, they are meant to be reviewed and modified to meet all of your installation's specific settings. Please review these configuration members with your site's HTTP server administrator.

Additional Information

There are two IBM manuals that can be of assistance when configuring your HTTP server:

- *HTTP Server Planning, Installing, and Using*
- *OS/390 e-business Infrastructure: IBM HTTP Server 5.1 — Customization and Usage* (This is an IBM Redbook)

For the publication order numbers for these books, see "Where to Find More Information" on page ix.

Step 3: Modify the CLZHTTPD Configuration Member

In this step you will review and modify the CLZHTTPD member off–loaded from the product tape into the **CLZ.SCLZHTML** file. This member is named the ***rules file*** in HTTP server configuration terminology and is pointed to by the server JCL parameter list. Because many of the members contain case-sensitive values, **please issue the CAPS OFF command** to ensure that automatic upper-casing does not occur.

Rootdir and Portno Values Review

The following shows only the lines that will change in the CLZHTTPD ***rules file*** based on the *rootdir* and *portno*. This member should be reviewed by your HTTP server administrator.

```
 File  Edit  Edit_Settings  Menu  Utilities  Compilers  Test  Help
-------------------------------------------------------------------------------
EDIT       CLZ.SCLZHTML(CLZHTTPD) - 01.00               Columns 00001 00072
Command ===>                                              Scroll ===> CSR
****** ***************************** Top of Data ******************************
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  - 19 Line(s) not Displayed
==CHG> ServerRoot     rootdir/
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  2 Line(s) not Displayed
==CHG> Port      portno
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  7 Line(s) not Displayed
==CHG> Protection WEBJOBNAME {
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  - 10 Line(s) not Displayed
==CHG> PidFile        rootdir/httpd-pid
==CHG> #AccessLog     rootdir/logs/httpd-log
==CHG> #AgentLog      rootdir/logs/agent-log
==CHG> #RefererLog    rootdir/logs/referer-log
==CHG> #ErrorLog      rootdir/logs/httpd-errors
==CHG> #CgiErrorLog   rootdir/logs/cgi-error
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  - 11 Line(s) not Displayed
==CHG> AccessReportRoot  rootdir/reports
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  - 48 Line(s) not Displayed
==CHG> Exec            /cgi-bin/*       rootdir/cgi-bin/*
==CHG> Pass            /html/*          rootdir/html/*
==CHG> Pass            /*               rootdir/*
- - -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  -  193 Line(s) not Displayed
****** **************************** Bottom of Data ****************************
```

Figure 4. CLZHTTPD (httpd.conf) Change Fields Only

ADDTYPE Directives

ADDTYPE directives are used to control the MIME types and file transfer defaults between the browser and the mainframe. Cloud 9 will search for ADDTYPE definitions in the following locations:

- The MVS file or USS file specified by the C9_ADDTYPE_FILE variable defined in httpd.envvars (see "Step 4: Modify the CLZEVARS Configuration Member" on page 25 for information about customising httpd.envvars)
- The USS file used as the RULE_FILE on the Cloud9 web server job (i.e. the file specified with the -r flag on the PARM= statement) or if this is not specified The httpd.conf file in the /etc/ directory.

Consequently, if C9_ADDTYPE_FILE is not specified, then Cloud 9 will search the RULE_FILE, and if the RULE_FILE is not specified on the server JCL, Cloud 9 will use the MIME types defined in the httpd.conf file in /etc/ directory.

Note that the file specified by C9_ADDTYPE_FILE can be a USS file or an MVS file.

The shipped ***rules file*** also contains ADDTYPE directives that control the MIME commands and file transfer defaults between the browser and the mainframe. As the implementation continues, these ADDTYPES might need to be expanded to accommodate additional file types and requirements. At this point, there are no additional modifications required for the ADDTYPE definitions.

Modify and Save CLZHTTPD

1. Review the *rootdir*, *portno*, and *WEBJOBNAME* variables from the Cloud 9 Installation Worksheet.

2. Using ISPF EDIT, access member CLZHTTPD in the CLZ.SCLZHTML data set.
3. Issue the CAPS OFF command to ensure case sensitive values do not become uppercased.
4. Issue the following global commands against the member:
   - X ALL
   - F rootdir ALL
   - F portno ALL
   - F WEBJOBNAME
   - Change rootdir ALL *rootdir* (ensure that the end format of the rootdir is **/rootdir/**)
   - Change portno ALL *portno*
   - Change WEBJOBNAME ALL *JOBNAME*
5. Save the member.

Step 4: Modify the CLZEVARS Configuration Member

In this step you will review and modify the CLZEVARS member that was off–loaded from the installation tape into the **CLZ.SCLZHTML** file. This member is named the ***environment variable file*** in HTTP server configuration terminology and is pointed to by the server JCL parameter list. Because many of the members contain case-sensitive values, **please issue the CAPS OFF command** to ensure that automatic upper casing does not occur.

Review the CLZEVARS Member

The following shows the shipped contents of the CLZEVARS member. This file must be configured by your HTTP server administrator, as many of the parameters are site-specific. Note that:

1. The *rootdir* variable from the Cloud 9 Installation Worksheet is needed.
2. The STEPLIB directive is delivered as STEPLIB=CURRENT. This means the HTTP server–spawned tasks default to the STEPLIB in the Cloud 9 server JCL. If your installation uses STEPLIB=dsn1,dsn2, all of the data sets in the STEPLIB statement must be authorized, as Cloud 9 requires an authorized environment.

```
#--------------------------------------------------------------------
#  Name:      CLZEVARS  (Will be named  rootdir/httpd.envvars in UNIX.)
#  Purpose:   Cloud9 Server Environment variable parameters
#  Usage:     This file is pointed to in the CLZC9SRV JCL.
#--------------------------------------------------------------------
#To customize this file change:
# 1. rootdir as per the Cloud9 installation worksheet.
# 2. If required, include the C9_ADDTYPE_FILE variable and a USS file
#    or MVS file that specify the mime types and translation rules.
#    Cloud9 searches for mime types and translation rules in the
#    following locations:
#       a. MVS file or USS file specified by the C9_ADDTYPE_FILE in
#          httpd.envvars (this member)
#       b. The USS file used as the RULE_FILE on the Cloud9
#          web server job (i.e. the file specified with the -r option)
#       c. The httpd.conf file in the SERVER_ROOT directory.
#    Consequently, if C9_ADDTYPE_FILE is not specified, then Cloud9
#    will search the RULE_FILE, and if the RULE_FILE does not define
#    mime types, then Cloud9 will use the mime types defined in the
#    httpd.conf file in SERVER_ROOT.
#    Note that the file specified by C9_ADDTYPE_FILE can be a USS file
#    or an MVS file.
#
#Various install and configuration paths are currently set /usr/...
#  This and all other parms using /usr/ must be reviewed with
#  the HTTP Server administrator as these set up issues are global
#  in nature versus Cloud 9 specific usage.
#--------------------------------------------------------------------
PATH=/bin:.:/usr/sbin:/usr/lpp/internet/bin:/usr/lpp/internet/sbin:/usr/lpp/ldap
SHELL=/bin/sh
TZ=EST5EDT
LANG=C
LC_ALL=en_US.IBM-1047
NLSPATH=/usr/lib/nls/msg/%L/%N:/usr/lpp/internet/%L/%N:/usr/lpp/ldap/lib/nls/msg
LIBPATH=/usr/lpp/internet/bin:/usr/lpp/internet/sbin:/usr/lpp/ldap/lib:<JAVA_HOM
JAVA_HOME=<JAVA_HOME>
CLASSPATH=.:/usr/lpp/internet/server_root/CAServlet:<JAVA_HOME>/lib/classes.zip
STEPLIB=CURRENT
SERVER_ROOT=rootdir/
#C9_ADDTYPE_FILE=rootdir/c9addtype_filename
```

Figure 5. CLZEVARS (httpd.envvars) Sample Member

Modify and Save CLZEVARS

Now that you have reviewed the considerations and contents of the CLZEVARS file, the following instructions should be used to customize the global variables and local variables.

1. Review the *rootdir* variable from the Cloud 9 Installation Worksheet.
2. Using ISPF EDIT, access member CLZEVARS in the CLZ.SCLZHTML data set you offloaded from the installation tape.
3. Issue the CAPS OFF command to ensure that case sensitive values do not appear in upper case.
4. Perform the following global commands against the member:
   - Change rootdir ALL *rootdir*
5. If you wist to use the C9_ADDTYPE_file mentioned in "ADDTYPE Directives" on page 24, then un-comment the variable and enter a file name that contains the addtype directives.

6. Change any of the other local directory settings as per the HTTP server administrator's direction.
7. Save the member.

Step 5: Customize the Cloud 9 HTTP Server JCL and Supporting Control Files

Step 5(a): Copy Product Load Library into Authorized Library

The Cloud 9 server must run from an authorized library, included in an authorized steplib concatenation. If the product load library is not an authorized data set, then you must copy the product load library into the authorized library for server execution.

Effects on Other JCL

WARNING: . If the authorized library name has changed from the installed library, make sure you review your application JCL members **CLZJIBM**, **CLZJMIG**, and **CLZJDYN** for possible steplib changes.

Step 5(b): Modify CLZC9SRV

The CLZC9SRV member is the recommended JCL for invoking the Cloud 9 HTTP server. It uses many default HTTP settings that may be modified and tailored by your HTTP server administrator. We recommend that you invoke the server "**as is**" for initial installation, and customize it later.

Time-out Parameter

WARNING: This job must not time out. Do not remove the TIME=NOLIMIT parameter on the EXEC statement. This job can also be made a started task.

Security Level for User ID/Password

The job card for the server must contain a user id and password that is at supervisor level for USS.

Modify and Submit CLZC9SRV

To start the Cloud 9 server, perform the following tasks:

1. Using ISPF EDIT, access member CLZC9SRV in the CLZ.SCLZJCL data set you off-loaded from the installation tape.
2. Issue the CAPS OFF command to ensure that case sensitive values do not appear in upper case.
3. Copy a Unix Supervisor-Level job card with password to the top of the member. This jobcard **REQUIRES** a userid and password with enough authority to load the HTTP server application. If the userid authority is not sufficient, then the server task will end with an 'insufficient authority' message on the console.
4. Change Jobname to equal WEBJOBNAME. This should be the value from the "Cloud 9 Installation Worksheet" on page 9 and *must* match the WEBJOBNAME in the CLZHTTDP member or rootdir/httpd.conf file. **A sample job card is provided in the JCL member in Figure 6 on page 28.**
5. Substitute your site-specific values (identified on the "Cloud 9 Installation Worksheet" on page 9) as per the instructions in the comment area of the JCL.
6. Save the member (Do not submit this job).

```
//* (Sample Jobcard)
//*
//*WEBJOBNAME JOB (ACCT#),'COMMENT',CLASS=A,REGION=0M,
//*    MSGCLASS=H,MSGLEVEL=(1,1),USER=XXXX,PASSWORD=XXXXXXX
//*
//*  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
//*  ! This jobcard REQUIRES the userid and password with enough   !
//*  ! authority to load the HTTP application.  If the userid      !
//*  ! authority is not sufficient, then the server task will end  !
//*  ! with an 'insufficient authority' message on the console.    !
//*  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
//*  ! WEBJOBNAME - The value from the placeholder worksheet.      !
//*  !  (The server job name must match the WEBJOBNAME             !
//*  !  in the c9httdp member or rootdir/httpd.conf file.)         !
//*  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
//*
//*-------------------------------------------------------------------
//* THIS IS THE CLOUD 9 FOR SCLM DEFAULT IBM HTTP WEB SERVER JCL
//*-------------------------------------------------------------------
//*
//* INSTRUCTIONS:
//*   1. CHANGE ROOTDIR TO THE VALUE IN THE CLOUD9 WORKSHEET.
//*   2. CHANGE PORTNO TO THE VALUE IN THE CLOUD9 WORKSHEET.
//*   3. CHANGE WEBJOBNAME TO VALUE IN THE CLOUD9 WORKSHEET.
//*   4. BE CAREFUL TO USE THE PROPER CASE WHEN CHANGING VALUES.
//*   5. IF THE CLZ.SCLZLOAD IS NOT AUTHORIZED, THEN
//*      COPY CURRENT CONTENTS OF CLZ.SCLZLOAD INTO EXISTING
//*      AUTHORIZED DATASET OR GET CLZ.SCLZLOAD AUTHORIZED.
//*   6. REVIEW THE NAME OF THE TCPIP LIBRARY FOR SITE
//*      STANDARDS. ACCESS TO THIS LIBRARY IS REQUIRED FOR
//*      CLOUD 9.  YOU WILL NOT NEED TO INCLUDE THE TCPIP LIBRARY
//*      IF IT IS IN THE LINKLIST.
//*   7. REVIEW THE httpd.envvars CONFIGURATION FILE FOR A 'STEPLIB'
//*      STATEMENT. IF THERE IS A STEPLIB= STATEMENT THAT INCLUDES
//*      DATASET NAMES, THEN ENSURE THAT THIS LIST IS AUTHORIZED.
//*   8. AFTER CHANGING THE ROOTDIR AND PORT VALUES, REVIEW THE
//*      EXECUTION PARM. THE PARM STRING SHOULD GO UP THROUGH COL 71
//*      AND THEN CONTINUE IN COL 16 ON THE NEXT LINE.
//*   9. IF YOU ARE A BREEZE USER THEN REVIEW THE NAME OF THE BREEZE
//*      LOAD LIBRARY, AND UNCOMMENT THE LINE LABELED 'BREEZE USERS'.
//*
//*-------------------------------------------------------------------
//*  The parm variable on the EXEC statement is of the format:
//*  (LEPARMS/ICSPARMS).
//*
//*  Refer to the following manuals for more information:
//* 1. HTTP Server Planning,Installing, and Using SC31-8690-02
//* 2. Redbook:OS/390 e-business Infrastructure: IBM HTTP Server 5.1
//*   - Customization and Usage  SG24-5603-00
//*-------------------------------------------------------------------
//CIGWEB  EXEC PGM=IMWHTTPD,TIME=NOLIMIT,
//     ACCT=(ACCT#),
//     PARM=('ENVAR("_CEE_ENVFILE=rootdir/httpd.envvars")/-r rootdir/
//            httpd.conf -B -p portno')
```

Figure 6. CLZC9SRV (Part 1 of 2)

```
//* ------------------------------------------------------------- *
//* This JCL requires an authorized dataset.  Review instruction  *
//*  numbers 5-7 above.                                           *
//* ------------------------------------------------------------- *
//STEPLIB  DD DSN=CLZ.SCLZLOAD,DISP=SHR
//*        DD DSN=BZZ.SBZZLOAD,DISP=SHR   * BREEZE USERS uncomment  *
//*        DD DSN=TCPIP.SEZATCP,DISP=SHR  * Uncomment if not in     *
//*                                       * LINKLIST or LPA         *
//SYSIN    DD DUMMY
//OUTDSC   OUTPUT DEST=HOLD
//SYSPRINT DD SYSOUT=*,OUTPUT=(*.OUTDSC)
//SYSERR   DD SYSOUT=*,OUTPUT=(*.OUTDSC)
//STDOUT   DD SYSOUT=*,OUTPUT=(*.OUTDSC)
//STDERR   DD SYSOUT=*,OUTPUT=(*.OUTDSC)
//SYSOUT   DD SYSOUT=*,OUTPUT=(*.OUTDSC)
//CEEDUMP  DD SYSOUT=*,OUTPUT=(*.OUTDSC)
```

Figure 6. CLZC9SRV (Part 2 of 2)

Step 5(c): Modify Batch Shells

Modify CLZJIBM

1. Using ISPF EDIT, access member CLZJIBM in the installed CLZ.SCLZJCL data set.
2. Skip the step of adding your job card values. The jobcard will be provided from the job card information in your Cloud 9 profile (see "Step 10: Perform Profile Setup" on page 48).
3. Substitute your site-specific values (identified on the "Cloud 9 Installation Worksheet" on page 9) as per the instructions in the comment area of the JCL.
4. Substitute the ISPF data set names for your installation.

   Note: This member needs to use the same names as the customized FLMLIBS skeleton member for SCLM.
5. Save the member.

The following is the CLZJIBM batch JCL member.

```
)DOT
%JOBCARD%
)ENDDOT
//* ------------------------------------------------------------------ *
//* NAME:     CLZJIBM                                                  *
//* PURPOSE:  CLOUD 9 FOR SCLM.                                        *
//*           SCLM BATCH SKELETON.                                     *
//* ------------------------------------------------------------------ *
//*                                                                    *
//* REQUIRED JCL MODIFICATION:                                         *
//*  1) CHANGE THE FOLLOWING AS PER THE INSTALLATION WORKSHEET.        *
//*      - ISPFQUAL                                                    *
//*      - TDISK                                                       *
//*  2) TARGET LIBRARIES CLZ.SCLZLOAD AND CLZ.SCLZCGI        /* C1 */ *
//*     SHOULD BE CHANGED IF THE INSTALLED HIGH LEVEL        /* C1 */ *
//*     QUALIFIER IS NOT 'CLZ.'                              /* C1 */ *
//*                                                                    *
//*  NOTE: BREEZE USERS MUST MODIFY THIS MODULE PER EMBEDDED           *
//*        INSTRUCTION. BREEZE DATASET NAMES MAY ALSO NEED TO BE       *
//*        MODIFIED.                                                   *
//*                                                                    *
//* 22OCT2001 OW51810 - CHANGES MARKED AS /* C1 */                     *
//* Z020402A                                                           *
//*                                                                    *
//*--------------------------------------------------------------------*
//*  RESIDES IN HTTP SERVER AT:                           /* C1 */ *
//*  /ROOTDIR/CLOUD9/JCL/CLZJIBM                                       *
//*--------------------------------------------------------------------*
)IF ACTION=OCOPY
//COPY   EXEC PGM=IKJEFT01
)DOT
%COPYFILES%
)ENDDOT
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
)DOT
%OCOPYSYNTAX%
)ENDDOT
/*
)ENDIF
```

Figure 7. CLZJIBM (Part 1 of 4)

```
)IF ACTION=IEBCOPY
//COPY     EXEC PGM=IEBCOPY
)DOT
%COPYFILES%
)ENDDOT
//SYSIN   DD *
)DOT
%OCOPYSYNTAX%
)ENDDOT
//SYSPRINT  DD  SYSOUT=*
)ENDIF
//*--------------------------------------------------------------------
//GENER  EXEC PGM=IEBGENER
//SYSUT1   DD *
)DOT
%SCLMSYNTAX%
)ENDDOT
//SYSUT2   DD DSN=&&CLIST(TEMPNAME),UNIT=TDISK,
//            SPACE=(TRK,(10,10,2),RLSE),
//            DISP=(NEW,PASS),DCB=(LRECL=80,
//            BLKSIZE=1600,DSORG=PO,RECFM=FB)
//SYSPRINT DD DUMMY
//SYSIN    DD DUMMY
)IF ACTION=DELETE
//DGRPTS   DD DSN=&&DELLIST,DISP=(NEW,PASS),                DELETE
//         SPACE=(TRK,(5,10),RLSE),
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
)ENDIF
)IF ACTION=BUILD
//COPYBULD EXEC PGM=CLZTFILE                                JAVA
//STEPLIB  DD DSN=CLZ.SCLZLOAD,DISP=SHR                        /* C1 */
//SYSIN    DD *                                             JAVA
)DOT
%SCLMSYNTAX%
)ENDDOT
//SYSOUT   DD DSN=&&BSYNTAX,DISP=(NEW,PASS),                JAVA
//            SPACE=(TRK,(10,10),RLSE),UNIT=TDISK,          JAVA
//            DCB=(LRECL=80,BLKSIZE=0,DSORG=PS,RECFM=FB)    JAVA
)ENDIF
```

Figure 7. CLZJIBM (Part 2 of 4)

```
//*****************************************************************
//* BREEZE USERS:  UNCOMMENT LINES WITH "BREEZE USERS"
//*****************************************************************
//TSO    EXEC PGM=IKJEFT01,REGION=4096K,TIME=1439,DYNAMNBR=200
//STEPLIB  DD DSN=CLZ.SCLZLOAD,DISP=SHR
//*        DD DSN=BZZ.SBZZLOAD,DISP=SHR            BREEZE USERS
//SYSTSIN  DD *
  ISPSTART CMD(%TEMPNAME)
//SYSTSPRT DD SYSOUT=(*)
//SYSPROC  DD DSN=&&CLIST,DISP=(OLD,DELETE)
//*        DD DSN=BZZ.SBZZCLIB,DISP=SHR            BREEZE USERS
//*****************************************************************
//* ISPF LIBRARIES
//ISPMLIB  DD DSN=ISPFQUAL.SISPMENU,DISP=SHR
//*        DD DSN=BZZ.SBZZMENU,DISP=SHR            BREEZE USERS
//ISPSLIB  DD DSN=ISPFQUAL.SISPSENU,DISP=SHR
//         DD DSN=ISPFQUAL.SISPSLIB,DISP=SHR
//*        DD DSN=BZZ.SBZZSENU,DISP=SHR            BREEZE USERS
//ISPPLIB  DD DSN=ISPFQUAL.SISPPENU,DISP=SHR
//*        DD DSN=BZZ.SBZZPENU,DISP=SHR            BREEZE USERS
//ISPTLIB  DD UNIT=VIO,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//            DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB)
//         DD DSN=ISPFQUAL.SISPTENU,DISP=SHR
//ISPTABL  DD UNIT=VIO,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//            DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB)
//ISPPROF  DD UNIT=VIO,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//            DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB)
//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
//ZFLMDD   DD *
    ZFLMNLST=FLMNLENU  ZFLMTRMT=ISR3278   ZDATEF=YY/MM/DD
/*
//*CIGLOG  DD SYSOUT=*                             BREEZE USERS
//*CIGLOG0 DD SYSOUT=*                             BREEZE USERS
//*CIGLOG1 DD DSN=&&CIGLOG1,DISP=(NEW,DELETE),     BREEZE USERS
//*           UNIT=TDISK,SPACE=(CYL,(1,1)),        BREEZE USERS/* C1 */
//*           DCB=(LRECL=132,BLKSIZE=0,RECFM=FB)   BREEZE USERS
//*CIGLOG2 DD DSN=&&CIGLOG2,DISP=(NEW,DELETE),     BREEZE USERS
//*           UNIT=TDISK,SPACE=(CYL,(1,1)),        BREEZE USERS/* C1 */
//*           DCB=(LRECL=132,BLKSIZE=0,RECFM=FB)   BREEZE USERS
//*CIGLOG3 DD DSN=&&CIGLOG3,DISP=(NEW,DELETE),     BREEZE USERS
//*           UNIT=TDISK,SPACE=(CYL,(1,1)),        BREEZE USERS/* C1 */
//*           DCB=(LRECL=132,BLKSIZE=0,RECFM=FB)   BREEZE USERS
```

Figure 7. CLZJIBM (Part 3 of 4)

```
//*****************************************************************
//* SCLM OUTPUT FILES
//*****************************************************************
//FLMMSGS  DD SYSOUT=(*)
)IF ACTION=BUILD
//BLDMSGS  DD SYSOUT=*,                                        BUILD
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
//BLDREPT  DD SYSOUT=*,                                        BUILD
//         DCB=(LRECL=80,BLKSIZE=3120,RECFM=FBA)
//BLDLIST  DD SYSOUT=*,                                        BUILD
//         DCB=(LRECL=259,BLKSIZE=3120,RECFM=VB)
//BLDEXIT  DD DSN=&&BLDEXIT,DISP=(NEW,DELETE),                 BUILD
//         SPACE=(TRK,(5,10),RLSE),
//         DCB=(LRECL=160,BLKSIZE=3200,RECFM=FB)
//BSYNTAX  DD DSN=&&BSYNTAX,DISP=(OLD,PASS)                    JAVA
)ENDIF
)IF ACTION=PROMOTE
//PROMMSGS DD SYSOUT=*,                                        PROMOTE
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=FB,DSORG=PS)
//PROMREPT DD SYSOUT=*,                                        PROMOTE
//         DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB,DSORG=PS)
//PROMEXIT DD DSN=&&PROMEXIT,DISP=(NEW,DELETE),                PROMOTE
//         SPACE=(TRK,(5,10),RLSE),
//         DCB=(LRECL=160,BLKSIZE=3200,RECFM=FB)
//COPYERR  DD SYSOUT=*,                                        PROMOTE
//         DCB=(RECFM=FBA,LRECL=133,BLKSIZE=1330)
)ENDIF
)IF ACTION=MIGRATE
//U2LSTS   DD SYSOUT=*,                                        MIGRATE
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
//U2MSGS   DD SYSOUT=*,                                        MIGRATE
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
)ENDIF
)IF ACTION=DELETE
//DGLIST   DD SYSOUT=*,                                        DELETE
//         DCB=(LRECL=137,BLKSIZE=3120,RECFM=VBA)
//DGMSGS   DD SYSOUT=*,                                        DELETE
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
//DGREPT   DD DSN=&&DELLIST,DISP=(MOD,PASS)
//DGEXIT   DD DSN=&&DELEXIT,DISP=(NEW,DELETE),                 DELETE
//         SPACE=(TRK,(5,10),RLSE),
//         DCB=(LRECL=160,BLKSIZE=3200,RECFM=FB)
)ENDIF
)IF ACTION=VERRECOV
//DBUMSGS  DD SYSOUT=*,                                        VERRECOV
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
)ENDIF
//*-------------------------------------------------------------------
)IF ACTION=DELETE
//DELMSGS EXEC PGM=IEBGENER
//SYSUT1   DD DSN=&&DELLIST,DISP=(OLD,PASS)
//SYSUT2   DD SYSOUT=*
//SYSPRINT DD DUMMY
//SYSIN    DD DUMMY
//*
//DELETE  EXEC PGM=IKJEFT01,REGION=4096K,DYNAMNBR=200
//STEPLIB  DD  DSN=CLZ.SCLZLOAD,DISP=SHR
//SYSTSIN  DD  *
  EX 'CLZ.SCLZCGI(CLZTRJDL)'
//SYSTSPRT DD  SYSOUT=*
//DELLIST  DD DSN=&&DELLIST,DISP=(OLD,PASS)
//LISTOUT  DD SYSOUT=*,
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
//UNIXLOC  DD DSN=CLZ.SCLZCGI(CLZTULOC),DISP=SHR
)ENDIF
```

Figure 7. CLZJIBM (Part 4 of 4)

Modify CLZJDYN REXX Shell

1. Using ISPF EDIT, access member CLZJDYN in the SCLZJCL data set you offloaded from the installation tape.
2. Substitute your site-specific values (identified on the "Cloud 9 Installation Worksheet" on page 9) as per the instructions in the comment area of the JCL.
3. Substitute the ISPF data set names for your installation.
4. Save the member.

The following is the CLZJDYN SCLM REXX shell used for dynamic allocation of ISPF libraries for Web based SCLM functions.

```
/* ------------------------------------------------------------- */
/* NAME:     CLZJDYN                                             */
/* PURPOSE:  CLOUD 9 FOR SCLM                                    */
/*           ISPF ALLOCATIONS FOR SCLM WEB BASED FUNCTIONS.      */
/* ------------------------------------------------------------- */
/*                                                               */
/* REQUIRED MODIFICATION:                                        */
/*  1) CHANGE THE FOLLOWING AS PER THE INSTALLATION WORKSHEET.   */
/*      - ISPFQUAL                                               */
/*                                                               */
/*  NOTE: BREEZE USERS MUST MODIFY THIS MODULE PER EMBEDDED      */
/*        INSTRUCTION. BREEZE DATASET NAMES MAY ALSO NEED TO BE  */
/*        MODIFIED.                                              */
/*                                                               */
/* ------------------------------------------------------------- */
/*  RESIDES IN THE HTTP SERVER AT:                               */
/*  ROOTDIR/CLOUD9/JCL/CLZJDYN                                   */
/* ------------------------------------------------------------- */

    ALLOC FI(ISPTLIB) +
        DSN('%TEMPNAME%' +
        'ISPFQUAL.SISPTENU') SHR

    /* COMMENT THE FOLLOWING LINES IF RUNNING BREEZE */
    ALLOC FI(ISPMLIB) DSN('ISPFQUAL.SISPMENU') SHR
    ALLOC FI(ISPSLIB) DSN('ISPFQUAL.SISPSENU') SHR
    ALLOC FI(ISPPLIB) DSN('ISPFQUAL.SISPPENU') SHR

    /* ** UNCOMMENT THE FOLLOWING LINES IF RUNNING BREEZE * */
    /* ALLOC FI(SYSPROC) +                                  */
    /*     DSN('%TEMPNAME%' +                               */
    /*                      'BZZ.SBZZCLIB') SHR REUSE       */
    /* ALLOC FI(ISPMLIB) DSN('BZZ.SBZZMENU' +               */
    /*                      'ISPFQUAL.SISPMENU') SHR REUSE  */
    /* ALLOC FI(ISPSLIB) DSN('BZZ.SBZZSENU' +               */
    /*                      'ISPFQUAL.SISPSENU') SHR REUSE  */
    /* ALLOC FI(ISPPLIB) DSN('BZZ.SBZZPENU' +               */
    /*                      'ISPFQUAL.SISPPENU') SHR REUSE  */
    /* ALLOC FI(CIGLOG) NEW DELETE DSORG(PS) CYLINDERS,+    */
    /*    SPACE(1,1) LRECL(120) BLKSIZE(2400) RECFM(F,B)    */
    /* ALLOC FI(CIGLOG0) NEW DELETE DSORG(PS) CYLINDERS,+   */
    /*    SPACE(1,1) LRECL(120) BLKSIZE(2400) RECFM(F,B)    */
    /* ALLOC FI(CIGLOG1) NEW DELETE DSORG(PS) CYLINDERS,+   */
    /*    SPACE(1,1) LRECL(132) BLKSIZE(13200) RECFM(F,B)   */
```

Figure 8. CLZJDYN (Part 1 of 2)

```
    /* ALLOC FI(CIGLOG2) NEW DELETE DSORG(PS) CYLINDERS,+   */
    /*    SPACE(1,1) LRECL(132) BLKSIZE(13200) RECFM(F,B)   */
    /* ALLOC FI(CIGLOG3) NEW DELETE DSORG(PS) CYLINDERS,+   */
    /*    SPACE(1,1) LRECL(132) BLKSIZE(13200) RECFM(F,B)   */
    /* ************** END OF BREEZE STATEMENTS ************ */

    /* ** UNCOMMENT THE FOLLOWING IF RUNNING JAVA SUPPORT   */
    /* ALLOC FI(SYSEXEC) DSN('CLZ.SCLZCGI') SHR REUSE       */
    /* ALLOC FI(UNIXLOC) DSN('CLZ.SCLZCGI(CLZTULOC)') SHR REUSE*/
    /* ************** END OF JAVA SUPPORT STATEMENTS ****** */

/* ------------------------------------------------------------ */
/* THE FOLLOWING COMMANDS ALLOCATE TEMPORARY ISPF FILES USED    */
/* BY SCLM DURING PROCESSING.                                   */
/* ------------------------------------------------------------ */
    ALLOC FI(ISPTABL) NEW DELETE DSORG(PO) CYLINDERS,+
        SPACE(1,1) DIR(5) LRECL(80) BLKSIZE(19040) RECFM(F,B)
    ALLOC FI(ISPPROF) NEW DELETE DSORG(PO) CYLINDERS,+
        SPACE(1,1) DIR(5) LRECL(80) BLKSIZE(19040) RECFM(F,B)
    ALLOC FI(ISPLOG) NEW DELETE DSORG(PS) CYLINDERS,+
        SPACE(1,1) LRECL(120) BLKSIZE(2400) RECFM(F,B)
    ALLOC FI(ISPCTL1) NEW DELETE DSORG(PS) CYLINDERS,+
        SPACE(1,1) LRECL(80) BLKSIZE(800) RECFM(F,B)
/* ------------------------------------------------------------ */
/* THE FOLLOWING DATASETS ARE USED BY SPECIFIC TRANSLATORS.     */
/* ------------------------------------------------------------ */
    ALLOC FI(SYSPRINT) NEW DELETE DSORG(PS) CYLINDERS,+
        SPACE(1,1) LRECL(120) BLKSIZE(2400) RECFM(F,B)

/* END OF ALLOCATIONS */
```

Figure 8. CLZJDYN (Part 2 of 2)

Modify CLZJMIG

Note: This task is required only if you have enabled CA-Endevor migration in "Step 2: Set Up the CIGINI Initialization File" on page 14.

1. Using ISPF EDIT, access member CLZJMIG in the SCLZJCL data set you offloaded from the installation tape.
2. Skip the step of adding your job card values. The jobcard will be provided from the job card information in your Cloud 9 profile (see "Step 10: Perform Profile Setup" on page 48).
3. Substitute your site-specific values (identified on the "Cloud 9 Installation Worksheet" on page 9) as per the instructions in the comment area of the JCL.
4. Substitute the ISPF data set names for your installation.
5. Review the number of level ddnames allowed for conversion.
6. Save the member.

The following is the CLZJMIG JCL shell used only with the CA-Endevor bridge.

```
)DOT
%JOBCARD%
)ENDDOT
//* ------------------------------------------------------------------ *
//* NAME:     CLZJMIG                                                  *
//* PURPOSE:  CLOUD 9 FOR SCLM                                         *
//*           CONVERT CA-ENDEVOR ELEMENT SOURCE TO SCLM SOURCE.        *
//* ------------------------------------------------------------------ *
//*                                                                    *
//* REQUIRED JCL MODIFICATION:                                         *
//*  1) CHANGE THE FOLLOWING AS PER THE INSTALLATION WORKSHEET.        *
//*      - ISPFQUAL                                                    *
//*      - TDISK                                                       *
//*  2) CHANGE THE XXX.XXX.ENDEVOR DATASET TO THE NAME OF THE ENDEVOR *
//*     AT YOUR INSTALLATION.                                          *
//*                                                                    *
//* ------------------------------------------------------------------ *
//*  RESIDES IN HTTP SERVER AT:                                        *
//*  ROOTDIR/CLOUD9/JCL/CLZJMIG                                        *
//*------------------------------------------------------------------- *
//* THIS JCL WILL CREATE SYNTAX TO CONVERT ENDEVOR ELEMENTS            *
//* INTO SCLM MEMBERS.  HERE IS THE STEPS:                            *
//* STEP0:  COPY THE CONTROL CARDS INTO A TEMPORARY DATASET            *
//* STEP1:  READ THE INPUT FROM CIGTSIN.                               *
//*         IF CVTFLAG=1 THEN                                          *
//*             WRITE RETRIEVE SCL STATEMENTS.                         *
//*             WRITE TSO COMMAND FILE FLMCMD SAVE STATEMENTS.         *
//*             SET RC=6.                                              *
//*         IF CVTFLAG=2 THEN                                          *
//*             WRITE PRINT SUMMARY SCL STATEMENTS.                    *
//*             SET RC=0.                                              *
//* STEP2:  IF STEP1.RC=0 THEN                                         *
//*             CALL ENDEVOR AND PROCESS THE PRINT SCL STATEMENTS.     *
```

Figure 9. CLZJMIG (Part 1 of 3)

```
//* STEP3:  IF STEP1.RC=0 THEN                                        *
//*             READ OUTPUT CREATED FROM STEP2 AND WRITE RETRIEVE     *
//*             LEVEL SCL STATEMENTS.  EACH LEVEL RETRIEVED WILL BE   *
//*             INTO A SEPARATE LEVELS DATASET.  IN TOTAL, 100        *
//*             TEMPORARY DATASETS WILL BE SETUP (ONE FOR EACH        *
//*             POSSIBLE LEVEL. THIS STEP WILL ALSO CREATE            *
//*             A TSO FILE CONTAINING COMMANDS TO                     *
//*             1) COPY THE RETRIEVED LEVEL INTO A TARGET SCLM DATASET*
//*                VIA SOME TYPE OF COPY UTILITY (E.G., OCOPY)        *
//*             2) EXECUTE A FLMCMD SAVE COMMAND                      *
//*             THIS TSO COMMAND FILE WILL BE EXECUTED IN STEP 5.     *
//* STEP4:  CALL ENDEVOR TO EXECUTE THE RETRIEVE SCL CREATED IN STEP 3.
//* STEP5:  CALL TSO (IKJEFT01) AND EXECUTE THE TSO FILE CREATED IN   *
//*         STEP 3.  THIS STEP WILL POPULATE THE TARGET SCLM LOCATION *
//*         WITH ENDEVOR ELEMENTS RETRIEVED IN STEP 4.                *
//*-------------------------------------------------------------------*
//STEP0  EXEC PGM=IEBGENER
//*   CVTFLAG=1  MEANS CONVERT CURRENT ELEMENT LEVEL ONLY
//*   CVTFLAG=2  MEANS CONVERT ALL ELEMENT LEVELS
//SYSUT1    DD  *
)DOT
%SCLMSYNTAX%
)ENDDOT
//SYSUT2    DD  DSN=&&CCARDS,UNIT=TDISK,
//  SPACE=(TRK,(10,10),RLSE),DISP=(NEW,PASS),
//  DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB)
//SYSPRINT DD DUMMY
//SYSIN    DD DUMMY
//*-------------------------------------------------------------------
//STEP1  EXEC PGM=IKJEFT01
//SYSTSIN   DD  *
EXEC 'CLZ.SCLZJCL(CLZSCVT1)'
//SYSTSPRT  DD  SYSOUT=*
//CIGTSIN   DD  DSN=&&CCARDS,DISP=(OLD,PASS)
//SCLOUT    DD  DSN=&&SCL,DISP=(NEW,PASS),
//   UNIT=TDISK,SPACE=(CYL,(1,1)),
//   DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB)
//FLMCMD    DD  DSN=&&CLIST(TEMPNAME),DISP=(NEW,PASS),
//   UNIT=TDISK,SPACE=(CYL,(1,1,10)),
//   DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB)
//*-------------------------------------------------------------------
//*  PROCESS PRINT SUMMARY SCL STATEMENTS
//*-------------------------------------------------------------------
//STEP2  EXEC PGM=NDVRC1,PARM=C1BM3000,DYNAMNBR=1500,REGION=4096K,
//    COND=(0,NE)
//STEPLIB  DD  DSN=XXX.XXX.ENDEVOR,DISP=SHR
//C1MSGS1  DD  SYSOUT=*
//BSTIPT01 DD  DSN=&&SCL,DISP=(OLD,DELETE)
//C1PRINT  DD  DSN=&&PRINT,DISP=(NEW,PASS),
//  UNIT=TDISK,SPACE=(CYL,(1,1)),
//  DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB)
```

Figure 9. CLZJMIG (Part 2 of 3)

```
//*-------------------------------------------------------------------
//*  READ C1PRINT CREATED FROM STEP 2 AND CREATE RETRIEVE LEVEL
//*  STATEMENTS.
//*-------------------------------------------------------------------
//STEP3  EXEC PGM=IKJEFT01,COND=(0,NE)
//SYSTSIN   DD  *
EXEC 'CLZ.SCLZJCL(CLZSCVT2)'
//SYSTSPRT  DD  SYSOUT=*
//CIGTSIN   DD  DSN=&&CCARDS,DISP=(OLD,PASS)
//SCLOUT    DD  DSN=&&SCL,DISP=(NEW,PASS),
//   UNIT=TDISK,SPACE=(CYL,(1,1)),
//   DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB)
//FLMCMD    DD  DSN=&&CLIST(TEMPNAME),DISP=(NEW,PASS),
//   UNIT=TDISK,SPACE=(CYL,(1,1,10)),
//   DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB)
//*-------------------------------------------------------------------
//STEP4  EXEC PGM=NDVRC1,PARM=C1BM3000,DYNAMNBR=1500,REGION=4096K
//STEPLIB  DD  DSN=XXX.XXX.ENDEVOR,DISP=SHR
//C1MSGS1  DD  SYSOUT=*
//BSTIPT01 DD  DSN=&&SCL,DISP=(OLD,DELETE)
//LEVEL00  DD  DSN=&&LEVEL00,DISP=(NEW,PASS),UNIT=TDISK,
//  SPACE=(CYL,(10,10,100)),DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB)
//LEVEL01  DD  DSN=&&LEVEL01,DISP=(NEW,PASS),UNIT=TDISK,
//  SPACE=(CYL,(10,10,100)),DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB)
//LEVEL02  DD  DSN=&&LEVEL02,DISP=(NEW,PASS),UNIT=TDISK,
//  SPACE=(CYL,(10,10,100)),DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB)
//     .
//     .
//     .
//*-------------------------------------------------------------------
//*  PROCESS THE FLMCMDS
//*-------------------------------------------------------------------
//STEP5  EXEC PGM=IKJEFT01
//SYSPROC   DD  DSN=&&CLIST,DISP=(OLD,DELETE)
//SYSTSIN   DD  *
   ISPSTART CMD(%TEMPNAME)
//SYSTSPRT  DD  SYSOUT=*
//ISPMLIB   DD  DSN=ISPFQUAL.SISPMENU,DISP=SHR
//ISPSLIB   DD  DSN=ISPFQUAL.SISPSENU,DISP=SHR
//          DD  DSN=ISPFQUAL.SISPPENU,DISP=SHR
//ISPPLIB   DD  DSN=ISPFQUAL.SISPPENU,DISP=SHR
//ISPTLIB   DD  UNIT=VIO,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//      DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB)
//          DD  DSN=ISPFQUAL.SISPTENU,DISP=SHR
//ISPTABL   DD  UNIT=VIO,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//      DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB)
//ISPPROF   DD  UNIT=VIO,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//      DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB)
//ISPLOG    DD  SYSOUT=*,
//      DCB=(LRECL=120,BLKSIZE=2400,DSORG=PS,RECFM=FB)
//ISPCTL1   DD  UNIT=VIO,DISP=NEW,SPACE=(CYL,(1,1),
//      DCB=(LRECL=80,BLKSIZE=800,DSORG=PS,RECFM=FB)
//ZFLMDD    DD  *
   ZFLMNLST=FLMNLENU ZFLMTRMT=ISR3278 ZDATEF=YY/MM/DD
//FLMMSGS   DD  SYSOUT=*
//* * * * * *  E  N  D     O  F    J  C  L  * * * * * * * * * * * * * *
```

Figure 9. CLZJMIG (Part 3 of 3)

CHECKPOINT #3

At this point the host based modification and configuration work should be complete. Before continuing with the next part of the install which will involve copying files to Unix and performing IVPs, verify that the following has been completed.

Table 11. Checkpoint #3

Task	Completed?
CLZHTTD has been modified?	
CLZEVARS has been modified?	
CLZC9SRV has been reviewed and modified?	
The userid and password on the server JCL has the authority to submit a server task?	
The server jobname is the same as the WEBJOBNAME parameter in the CLZHTTD member?	
The CLZJIBM batch JCL member has been reviewed and modified?	
The CLZJDYN allocation shell has been reviewed and modified?	
The CLZJMIG batch JCL member has been reviewed and modified?	
The steplib in the CLZC9SRV JCL is the same as in CLZJIBM and CLZJMIG?	
Is the steplib in CLZC9SRV JCL authorized?	

Step 6: Create and Populate Additional HFS Cloud 9 Directories

In this step, you will create Cloud 9 Unix Root and product directories and populate them with the Cloud 9 product and configuration files you modified in the previous steps. Because many of the members contain case-sensitive values, **please issue the CAPS OFF command** to ensure that automatic upper casing does not occur.

The following is the REXX exec CLZCHMOD that will be input to the second step of CLZJUNIX.

```
/* ******************************** REXX ************************** */
/* NAME:  CLZCHMOD                                                  */
/* PURPOSE:                                                         */
/* THIS REXX WILL MODIFY THE SECURITY ATTRIBUTES OF THE UNIX BASED   */
/* COMPONENTS COPIED TO USS IN THE CLZJUNIX JCL JOB STREAM.          */
/* MODIFY THE rootdir VARIABLE AS PER THE WORKSHEET VALUES.          */
/* WARNING: THIS MEMBER CONTAINS CASE SENSITIVE INPUT DATA.          */
/* ******************************** REXX ************************** */
 trace all
 call syscalls 'ON'
 address syscall
 CHMOD 'rootdir/cloud9/jcl/CLZJDYN'  755
 CHMOD 'rootdir/cloud9/jcl/CLZJIBM'  755
 CHMOD 'rootdir/cloud9/jcl/CLZJMIG'  755
 CHMOD 'rootdir/httpd.conf'  755
 CHMOD 'rootdir/httpd.envvars' 755
 CHMOD 'rootdir/cloud9/profiles/user1.jpg' 755
 CHMOD 'rootdir/cloud9/profiles/user1.prf' 755
 CHMOD 'rootdir/cloud9/profiles/user2.jpg' 755
 CHMOD 'rootdir/cloud9/profiles/user2.prf' 755
```

Figure 10. CLZCHMOD

Modify CLZCHMOD

1. Using ISPF EDIT, access member CLZCHMOD in the CLZ.SCLZJCL target library.
2. Issue the CAPS OFF command to ensure that case sensitive values do not appear in upper case.

   WARNING : Unix files are case sensitive. Do not change the case on any file names contained in this REXX EXEC.
3. Substitute your site-specific values (identified on the "Cloud 9 Installation Worksheet" on page 9) as per the instructions in the comment area of the JCL.
4. Save the member.

Modify and Submit CLZJUNIX

1. Using ISPF EDIT, access member CLZJUNIX.
2. Issue the CAPS OFF command to ensure that case sensitive values do not appear in upper case.

   WARNING : Unix files are case sensitive. Do not change the case on any file names contained in this JCL.
3. Copy your job card values to the top of the member.
4. Substitute your site-specific values (identified on the "Cloud 9 Installation Worksheet" on page 9) as per the instructions in the comment area of the JCL.
5. Submit the job.

Note: Note that this job should terminate with COND CODE=0. If it does not:

1. Review your job card parameters and the JCL for errors.
2. Resubmit the job.

```
//* (JOBCARD)
//* NAME:     CLZJUNIX
//* PURPOSE:  JCL TO CREATE AND POPULATE THE CLOUD 9 UNIX DIRECTORIES.
//* USAGE:    Make sure profile of 'caps off' prior to modifying this
//*           member.  Unix directory and file names are case sensitive.
//* USAGE:    Set to 'number off' prior to modifying this
//*           member.
//*-------------------------------------------------------------------------
//*            * * *  N O T I C E * * *
//*   THIS PROGRAM IS A PROPRIETARY PRODUCT OF CHICAGO INTERFACE
//*   GROUP, INC. @ COPYRIGHT 2000 CHICAGO INTERFACE GROUP, INC.
//*   ALL RIGHTS RESERVED.
//*-------------------------------------------------------------------------
//*
//* **                                  **
//* ** PRODUCT INSTALLATION/SETUP ISSUES **
//* **                                  **
//* THE FOLLOWING IS A LIST OF MODIFICATIONS REQUIRED DURING PRODUCT
//* INSTALLATION AND INITIAL SETUP:
//*
//* 1) ADD A VALID JOB CARD
//* 2) CHANGE rootdir to the root directory value in your
//*    worksheet.
//* 3) Change USER1 and USER2 to actual userids.  Use Upper Case.
//*    These files are demo profile files for the IVP.
//* 4) DO NOT change the case on the file names. Unix files are
//*    case sensitive.
//* 5) Ensure that the last step points to the dataset that contains
//*    the REXX member CLZCHMOD.
//*
//CMD0     EXEC PGM=IKJEFT01,REGION=4096K,TIME=1439,DYNAMNBR=200
//*--------------------------------------------------------------------
//* TSO OUTPUT FILE
//*--------------------------------------------------------------------
//SYSTSPRT DD SYSOUT=(*)
//*--------------------------------------------------------------------
```

Figure 11. CLZJUNIX (Part 1 of 2)

```
//* TSO INPUT FILE
//*--------------------------------------------------------------------
//SYSTSIN  DD *
 MKDIR 'rootdir/cloud9/jcl'         MODE(7,5,5)
 MKDIR 'rootdir/cloud9/profiles' MODE(7,7,7)
 MKDIR 'rootdir/logs'  MODE(7,7,7)
 MKDIR 'rootdir/reports' MODE(7,7,7)
 OPUT  'CLZ.SCLZJCL(CLZJDYN)' -
        'rootdir/cloud9/jcl/CLZJDYN'
 OPUT  'CLZ.SCLZJCL(CLZJIBM)' -
        'rootdir/cloud9/jcl/CLZJIBM'
 OPUT  'CLZ.SCLZJCL(CLZJMIG)' -
        'rootdir/cloud9/jcl/CLZJMIG'
  OPUT  'CLZ.SCLZHTML(CLZHTTPD)' -
        'rootdir/httpd.conf'
  OPUT  'CLZ.SCLZHTML(CLZEVARS)' -
        'rootdir/httpd.envvars'
  OPUT  'CLZ.SCLZJPG(CLZCIG01)' -
        'rootdir/cloud9/profiles/user1.jpg'
 OPUT  'CLZ.SCLZJPG(CLZCIG02)' -
        'rootdir/cloud9/profiles/user2.jpg'
 OPUT  'CLZ.SCLZPRF(CLZCIG01)' -
        'rootdir/cloud9/profiles/user1.prf'
 OPUT  'CLZ.SCLZPRF(CLZCIG02)' -
        'rootdir/cloud9/profiles/user2.prf'
/*
//CHMOD0  EXEC PGM=IRXJCL,PARM=(CLZCHMOD)
//*-------------------------------------------------------------------
//* REXX STANDARD FILES
//*-------------------------------------------------------------------
//SYSTSPRT DD SYSOUT=(*)
//SYSTSIN  DD DUMMY
//*-------------------------------------------------------------------
//* THE FOLLOWING DATASET MUST CONTAIN THE REXX MEMBER CLZCHMOD
//*-------------------------------------------------------------------
//SYSEXEC  DD DSN=CLZ.SCLZJCL,DISP=SHR
```

Figure 11. CLZJUNIX (Part 2 of 2)

Note: After this job has been submitted and executes successfully, the Cloud 9 USS directories should be populated and ready for testing.

Step 7: Review Authorization Requirements for CLZRSDRV

The module CLZRSDRV is the interface module for invoking authorized, real-time processes in the HTTP server.

To check the attributes of the CLZRSDRV authorized program interface module, use one of the following methods:

Using the ISPF Unix shell (requires that the SYS1.SBPXxxxx libraries are in your logon setup):

1. Access Unix System services
   ```
   tso %ishell
   ```
2. Display the rootdir/cgi-bin directory. When the command line appears, type
   ```
   rootdir/cgi-bin/
   ```

   where *rootdir* is the name of the root directory used by your installation.

3. Issue the attribute 'a' line command for CLZRSDRV.

```
                                      Directory List
                                                                Command===>
_____________________________________________________________________________

/u/ibmdemo/cgi-bin/
 Select one or more files with / or action codes.

   Type  Filename                                                Row 1 of 19
_ Dir   .
_ Dir   ..
_ File  CLZRADDS
_ File  CLZREDRV
_ File  CLZRENDV
_ File  CLZRINDX
_ File  CLZRLMBR
_ File  CLZRLUNX
_ File  CLZRMENU
_ File  CLZRMLST
_ File  CLZRPROF
_ File  CLZRSCLM
_ File  CLZRSCMA
a File  CLZRSDRV
_ File  CLZRSDSF
```

Figure 12. Display of rootdir/cgi-bin Directory

Note that the number of actual members in the CGI-BIN directory is subject to change.

4. From the Edit pull down menu, select Option 1 Mode fields.

```
  .-------------------------------------------------.
/ |   Edit  Help                                    |
S |  .------------------------------. ---------------  |
  |  |    1. Mode fields...         | es               |
  |  |    2. Owning user...         |                  |
_ |  |    3. Owning group...        |                  |
_ |  |    4. User auditing...       |     More:     +  |
_ |  |    5. Auditor auditing...    |                  |
_ |  |    6. File format...         |                  |
_ |  |    7. Extended Attributes... |                  |
_ |  '------------------------------'                  |
a |  Group owner  . . . : SYS1(0)                      |
_ |  Last modified  . . : 10/11/2000 21:59 GMT         |
_ |  Last changed . . . : 10/11/2000 21:59 GMT         |
_ |  Last accessed  . . : 10/11/2000 20:26 GMT         |
  |  Created  . . . . . : 10/11/2000 20:26 GMT         |
_ |  Link count . . . . : 1                            |
_ |  Set UID bit  . . . : 0                            |
_ |                                                    |
_ |                                                    |
_ '----------------------------------------------------'
```

Figure 13. Edit Pull Down — Mode Fields

5. From the "Change the Mode" panel (shown below) ensure that the sticky bit (the access permission setting) is set to 1.

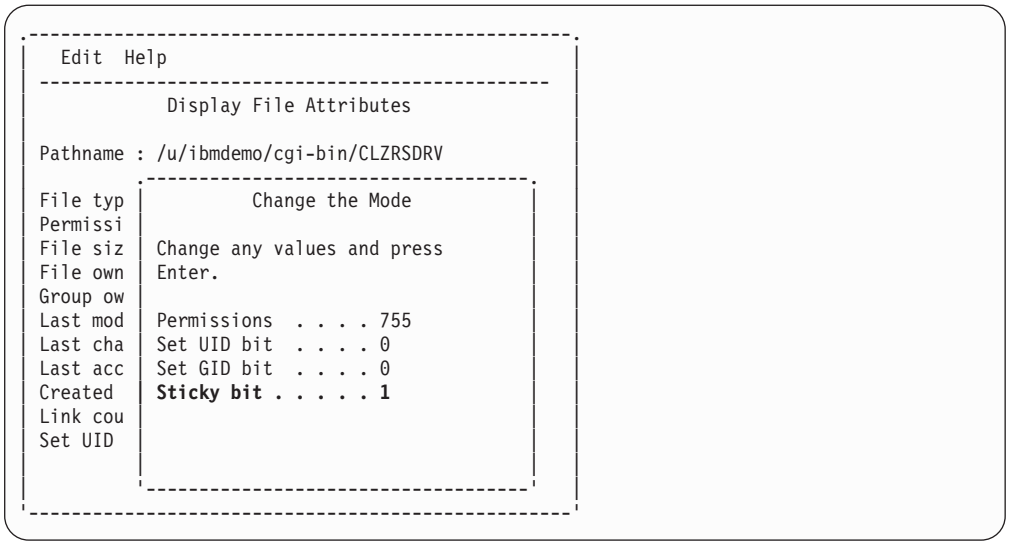

Figure 14. Change the Mode Panel

Using the OMVS Command shell:

1. Access Unix System services using one of the following commands:
   - From TSO Ready:

     ```
     OMVS
     ```
   - From an ISPF panel:

     ```
     TSO OMVS
     ```
   - Telnet to Unix System services: contact your system administrator to find if this function has been enabled on your system and what address:port to use.
2. Change to the rootdir/cgi-bin directory:

   ```
   cd rootdir/cgi-bin
   ```
3. Display the attributes for CLZRSDRV:

   ```
   ls -l CLZRSDRV
   ```

   The first column of output is the attribute bits, which should be:

   ```
   -rwxr-xr-t
   ```

Trouble Shooting

If you encounter any problems with this step, double-check that the following items are in place:

- The real module CLZRSDRV and its required alias C9RSDRV reside in the linklist or steplib.
- A dummy stub entry for CLZRSDRV, with a length of zero, resides in the cgi-bin directory.
- The CLZRSDRV file has the access permission enabled (referred to as ″sticky bit″ in USS terminology).

CA-Endevor Bridge

If you are using the CA-Endevor Bridge CIGINI option, you will also want to perform “Step 7: Review Authorization Requirements for CLZRSDRV” on page 42 for the rootdir/cgi-bin file called CLZREDRV.

Chapter 6. Perform Installation Verification Procedures

Step 8: Cloud 9 Server Installation Verification

To test the Cloud 9 server:

Start the Server

1. Submit the CLZC9SRV job, located in the SCLZJCL data set.
2. View the //SYSPRINT DD and //SYSOUT DD in the job output and verify that it looks like the output below.

```
******************************** TOP OF DATA *******
IMW0234I Starting.. httpd
IMW0235I Server is ready.
******************************** BOTTOM OF DATA *****
******************************** TOP OF DATA ********************
..... This is IBM HTTP Server V5R1M0
..... Built on Feb 17 1999 at 20:10:29.
..... Started at Sat Mar 25 16:17:31 2000
..... Running as "P390", UID:0, GID:0.
******************************** BOTTOM OF DATA ******************
```

Figure 15. SYSPRINT DD AND SYSOUT DD

Shut Down the Server

To quiesce the server job, enter one of the following commands (replace *cloud9–job-name* with the name of your job):

If entered on an MVS console:

`STOP cloud9-job-name`

If entered via a console interface, such as SDSF:

`/STOP cloud9-job-name`

Note: Because the Cloud 9 server uses TCP/IP stack and a cancel does not always clean up storage, it is recommended that the user quiesce the server using the MVS console command method over simply canceling the job. Issuing the console command will allow the Cloud 9 server job to end cleanly.

Restart the Server

1. Re-submit the server JCL (CLZC9SRV) for the next test.

CHECKPOINT #4

At this point, you should have successfully completed the following tasks:

Table 12. Checkpoint #4

Task	Completed?
Submitted the server JCL — CLZC9SRV?	
Reviewed the sysout files showing the port # and verifying that this is port # you expected?	
Issued a Quiesce of the server to test command and clean up?	
Resubmitted the server for the next test?	

Step 9: Invoking and Logging On to Cloud 9

The purpose of this test is to verify that the basic configuration has been performed successfully and that Cloud 9 is accessable through the Web. Most probable failures in this step will be security or configuration.

Execute cloud9.htm

To test installation of the application, execute the cloud9.htm file as follows:

Access Cloud 9 directly from HTTP server directories:

1. On your desktop, launch your browser.
2. Modify the following statement with your IP address and port number and type it in the browser's address window (labeled "Location" in Netscape Navigator and "Address" in Internet Explorer):

   `http://ip-address:portno/cloud9.htm`

   The browser will request the html file directly from HTTP and execute the Cloud 9 application.
3. When the Cloud 9 product is invoked you will be prompted with a log-in panel. Enter your TSO user id and password and click "ok" to begin using Cloud 9.

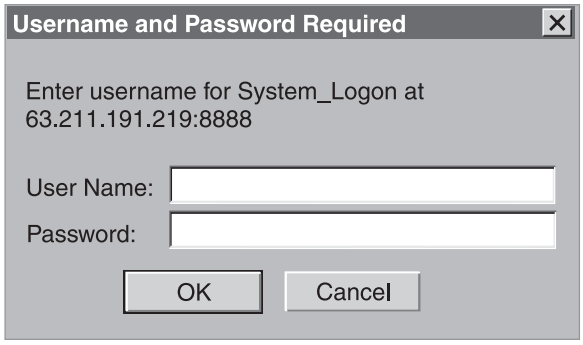

Figure 16. Cloud 9 Logon Prompt

Diagnostics:

If you cannot invoke the Cloud 9 application, use the following list to attempt to determine the problem:

1. Is the Cloud 9 server active? How do you know?
2. Are there any error messages in the SYSOUT queue or on the CONSOLE?
3. From your browser are you using the correct *ip-address:port* combination? How do you know?
4. Are you using a Cloud 9 supported browser? (Netscape 4.7 or Explorer 5.0)
5. Can you access any other HTTP server applications?

Verify that the Unix directories are configured as per Appendix A, "Cloud 9 Unix Directory Structure" on page 129 in this manual.

Step 10: Perform Profile Setup

During the execution of CLZJUNIX a profile and picture for two userids was stored in the Cloud 9 root directory. If you are logged on as one of those userids, then you can view the profile at this time.

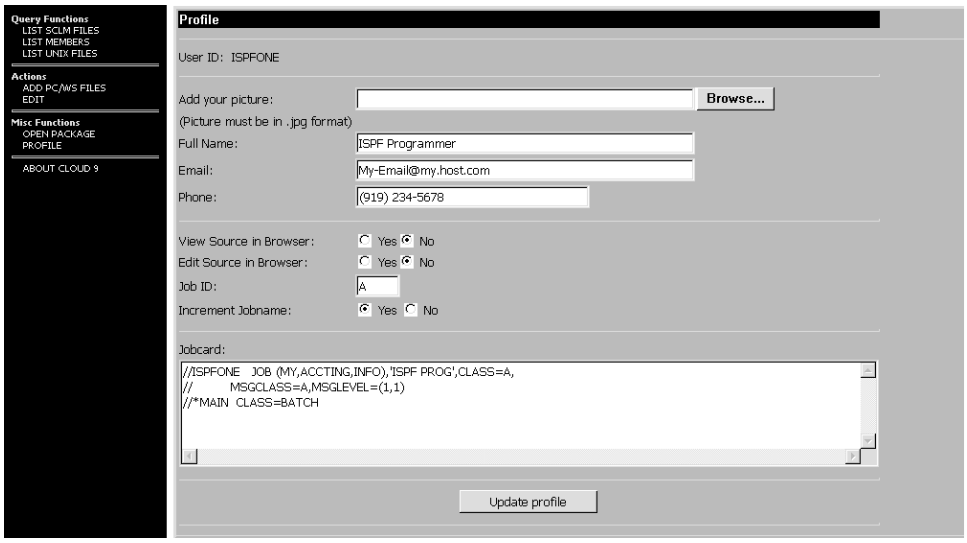

Figure 17. Profile Panel

Select **PROFILE** on the Main Menu to set up your profile at this time with real values and JOB CARD information.

Tip: If this step fails with a message similar to *No data received from host* (the exact wording of the message depends on the web browser being used) it is most likely because the TCPIP.SEZATCP library is inaccessible. Examination of the SYSLOG will show an 806 ABEND for module EZACICnn (where nn depends on your local configuration). Add TCPIP.SEZATCP to the STEPLIB for the Cloud 9 server and restart it. Note that this library, and any other library in the STEPLIB concatenation must be APF authorized.

Step 11: Perform Batch and Interactive IVPs

To test the connection to the host, perform the following steps:

1. Select **List SCLM Files** from the Main Menu.
2. At the Basic Search panel, fill in a valid SCLM project name and optionally other filters.
3. Click Submit .

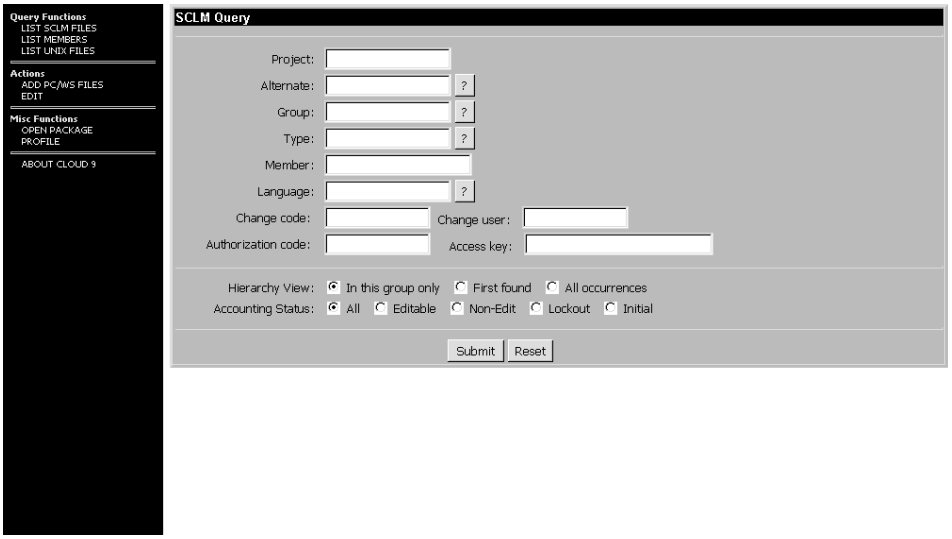

Figure 18. Basic Search Panel

4. Verify that the list returned has the same format as below:

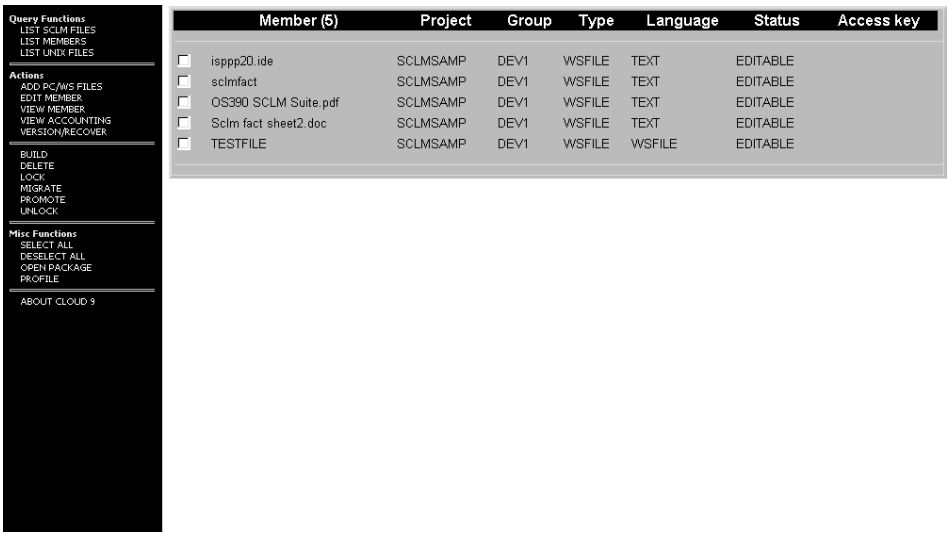

Figure 19. Search List Panel

5. Use this list to perform batch and interactive IVP processes.

Tip: If this step fails (the panel returned has the messages *Failure to call TSO/ISPF/SCLM* and *NULL FILE: SYSTSPRT*) it is most likely because one or more STEPLIB data sets are lacking APF authorization. Verify that authorization is properly specified in the PROG00 member of SYS1.PARMLIB.

Test the Batch Interface:

- Check one of the members on the list
- Select Build action.
- Fill in all options (including selection of batch submission) and click on submit.
- Check the expansion of the JCL in the JES2 Hold queue to validate that CLZJIBM was modified correctly.

- Review the batch JCL that was submitted and ensure CLZJIBM was found and modified correctly.

Exit Cloud 9:

To close your browser, either:

- Select **Close** from the file pull-down menu, Or
- Click on the "X" in the upper right hand corner of the browser window.

Step 12: Perform Batch SLR IVP

Modify and Submit CLZC9J06

Earlier in the installation you created a demo version of the SLR database. At this time execute the SLR IVP to review the CIGINI setup and the demo database display and update. For information about SLR syntax, see Appendix B, "Suite Long Name Registry" on page 131.

1. Using ISPF EDIT, access member CLZC9J06 in the SCLZJCL data set.
2. Add a valid job card.
3. Substitute your site-specific values (identified on the "Cloud 9 Installation Worksheet" on page 9) as per the instructions in the comment area of the JCL.
4. Submit the member.
5. Review output.

The following is the CLZC9J06 member for testing the SLR setup and update.

```
//**(JOBCARD)
//**
//*******************************************************************
//* CLOUD 9 FOR SCLM VERSION OF IVP                                 *
//*******************************************************************
//*                                                                 *
//* CLZC9J06 - THE PURPOSE OF THIS JCL TO RUN THE SLR DATA IVP.     *
//*            STEP 1 WILL PRINT THE CIGINI DEFINITIONS.            *
//*            STEP 2 WILL LIST IVP SLR RULE DEFINITIONS.           *
//*            STEP 3 WILL ADD DATASET AND SCLM TYPE DEFINITIONS    *
//*                   AND THEN LIST ALL RULES IN THE DATABASE.      *
//* NOTE:    - THE SYNTAX PROVIDED IS FOR AN EXAMPLE ONLY.          *
//*            IT IS RECOMMENDED THAT STEP3 SYNTAX BE TAILORED TO   *
//*            ACTUAL LOCAL VALUES.                                 *
//*******************************************************************
//*                                                                 *
//* REQUIRED JCL MODIFICATION:                                      *
//*   1) INCLUDE A JOBCARD                                          *
//*                                                                 *
//*******************************************************************
//*                                                                 *
//*  STEP 1: PRINT THE CIGINI DEFINITIONS.                          *
//*                                                                 *
//*******************************************************************
//STEP1    EXEC PGM=CLZNTINI
//STEPLIB  DD DSN=CLZ.SCLZLOAD,DISP=SHR
//CIGPRINT DD SYSOUT=*
//*******************************************************************
//*                                                                 *
//*  STEP 2: LIST THE CURRENT CONTENTS OF THE SLR DATABASE          *
//*                                                                 *
//*******************************************************************
//STEP2    EXEC PGM=CLZSLR
//STEPLIB DD DSN=CLZ.SCLZLOAD,DISP=SHR
//CIGPUNCH DD SYSOUT=*
//CIGLOG   DD SYSOUT=*
//CIGIN   DD *
 LIST NAME RULES.
/*
//*******************************************************************
//*                                                                 *
//*  STEP 3: ADD DATASET AND TYPE DEFINITIONS TO SLR DATABASE.      *
//*          USE AS IS OR TAILOR WITH LOCAL VALUES.                 *
//*                                                                 *
//*******************************************************************
//STEP3    EXEC PGM=CLZSLR
//STEPLIB DD DSN=CLZ.SCLZLOAD,DISP=SHR
//CIGPUNCH DD SYSOUT=*
//CIGLOG   DD SYSOUT=*
//CIGIN   DD *
 ADD NAME RULE FOR DATASET 'CLZ.SCLZLOAD.PDS1' CASE SENSITIVE.
 ADD NAME RULE FOR DATASET 'CLZ.SCLZLOAD.PDS2' CASE INSENSITIVE.
 ADD NAME RULE FOR SCLM  TYPE HTML     .
 ADD NAME RULE FOR SCLM TYPE JAVA      CASE SENSITIVE .
 ADD NAME RULE FOR SCLM TYPE JAVACLAS  CASE SENSITIVE .
 ADD NAME RULE FOR SCLM TYPE JAR       CASE SENSITIVE .
 ADD NAME RULE FOR SCLM TYPE DOC       CASE INSENSITIVE .
 LIST NAME RULES.
/*
```

Figure 20. CLZC9J06

Step 13: Setup SLR Maintenance JCL

Modify CLZC9J04

At this time the basic install is complete. Take time now to setup a backup, delete, and define JCL stream for production use.

1. Using ISPF EDIT, access member CLZC9J04 in the SCLZJCL data set.
2. Add a valid job card.
3. Substitute your site-specific values (identified on the "Cloud 9 Installation Worksheet" on page 9) as per the instructions in the comment area of the JCL.
4. Savet the member.

The following is the CLZC9J04 member containing the SLR file maintenance JCL.

```
//**(JOBCARD)
//**
//*********************************************************************
//*                                                                   *
//* CLZC9J04 - THE PURPOSE OF THIS JCL IS TO BACKUP, DELETE, AND      *
//*            DEFINE A PRODUCTION SLR DATABASE.                      *
//*                                                                   *
//*********************************************************************
//*                                                                   *
//* REQUIRED JCL MODIFICATION:                                        *
//*   1) INCLUDE A JOBCARD                                            *
//*   2) CHANGE THE FOLLOWING AS PER THE INSTALLATION WORKSHEET.      *
//*      - VOLUMES(DVOLSER)                                           *
//*      - DUNIT                                                      *
//*      - TDISK                                                      *
//*   3) SIZE THE FILES IN STEP2, STEP3, AND STEP4.                   *
//*                                                                   *
//*********************************************************************
//*                                                                   *
//* DO NOT MODIFY THE VSAM PARAMETERS PROVIDED IN THIS JCL. DOING SO  *
//* WILL PRODUCE UNEXPECTED RESULTS FROM THE CLOUD9 APPLICATION.      *
//*                                                                   *
//*********************************************************************
```

Figure 21. CLZC9J04 (Part 1 of 3)

```
//*                                                                   *
//* STEP1: DELETE THE OLD VERSION OF THE BACKUP, IF IT EXISTS.        *
//*        DELETE THE OLD VERSION OF THE SORT FILE. IF IT EXISTS.     *
//* STEP2: CREATE A SEQUENTIAL BACKUP OF THE SLR DATABASE USING       *
//*        STANDARD IDCAMS REPRO SERVICES.                            *
//* STEP3: SORT THE DATA.                                             *
//* STEP4: DELETE, DEFINE, AND REPRO THE SLR DATABASE.                *
//* STEP5: EXPAND VSAM INDEXES ON THE SLR DATABASE.                   *
//*                                                                   *
//*********************************************************************
//*                                                                   *
//* STEP1: DELETE THE OLD VERSION OF THE BACKUP, IF IT EXISTS.        *
//*                                                                   *
//*********************************************************************
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *
      DELETE 'CLZ.SCLZSLR.SLR.SEQ'       PURGE
      DELETE 'CLZ.SCLZSLR.SLR.SORT' PURGE
      IF MAXCC <= 8 THEN DO
        SET MAXCC = 0
        SET LASTCC = 0
      END
//*********************************************************************
//*                                                                   *
//* STEP2: CREATE A SEQUENTIAL BACKUP OF THE SLR DATABASE USING       *
//*        STANDARD IDCAMS REPRO SERVICES.                            *
//*                                                                   *
//*********************************************************************
//STEP2    EXEC PGM=IDCAMS,
//         COND=(0,LT)
//OUTDD02  DD DSN=CLZ.SCLZSLR.SLR.SEQ,DISP=(NEW,CATLG,DELETE),
//         UNIT=DUNIT,SPACE=(CYL,(10,5),RLSE),
//         DCB=(RECFM=VB,LRECL=604,BLKSIZE=6160)
//INDD02   DD DSN=CLZ.SCLZSLR.DATABASE.DATA,DISP=OLD,
//            AMP='BUFNI=10,BUFND=10'
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  REPRO IFILE(INDD02)  OFILE(OUTDD02)
/*
//*********************************************************************
//*                                                                   *
//* STEP3:  SORT THE DATA AND DELETE ALL RECORDS THE QUALIFY FOR A    *
//*         LOGICAL DELETE.                                           *
//*                                                                   *
//*********************************************************************
```

Figure 21. CLZC9J04 (Part 2 of 3)

```
//STEP3    EXEC PGM=SORT,
//         COND=(0,LT)
//SORTIN   DD DSN=CLZ.SCLZSLR.SLR.SEQ,DISP=SHR
//SORTOUT  DD DSN=CLZ.SCLZSLR.SLR.SORT,DISP=(NEW,CATLG,DELETE),
//         UNIT=DUNIT,SPACE=(CYL,(10,5),RLSE),
//         DCB=(RECFM=VB,LRECL=604,BLKSIZE=6160)
//SORTWK01 DD   UNIT=TDISK,SPACE=(CYL,(5,5))
//SORTWK02 DD   UNIT=TDISK,SPACE=(CYL,(5,5))
//SORTWK03 DD   UNIT=TDISK,SPACE=(CYL,(5,5))
//SORTWK04 DD   UNIT=TDISK,SPACE=(CYL,(5,5))
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSIN    DD *
  SORT FIELDS=(5,254,CH,A)
  RECORD TYPE=V,LENGTH=(604,,,254)
  SUM FIELDS=NONE
/*
//*********************************************************************
//*                                                                   *
//*  STEP 4: ALLOCATE THE SLR VSAM DATABASE AND REPRO BACKUP          *
//*                                                                   *
//*********************************************************************
//STEP4   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//INDD01   DD DSN=CLZ.SCLZSLR.SLR.SORT,DISP=SHR
//SYSIN   DD *
 DELETE CLZ.SCLZSLR.DATABASE
 DEFINE CLUSTER -
    (NAME('CLZ.SCLZSLR.DATABASE') -
    IMBED SPEED UNIQUE FREESPACE(30 30) -
    VOLUMES(DVOLSER) TRACKS(60 40) -
    SHR(4 3) -
    KEYS(254 0) -
    RECORDSIZE(512 1024)) -
  DATA  (CISZ(16000)) -
  INDEX (CISZ(4096))
 REPRO INFILE(INDD01) OUTDATASET('CLZ.SCLZSLR.DATABASE')
/*
//*********************************************************************
//*                                                                   *
//*  STEP 5: FORCE A VSAM SPLIT FOR INTEGRITY SUPPORT.                *
//*                                                                   *
//*********************************************************************
//STEP5    EXEC PGM=CLZVSM2L,PARM='CLZ.SCLZSLR.DATABASE'
//STEPLIB DD DSN=CLZ.SCLZLOAD,DISP=SHR
```

Figure 21. CLZC9J04 (Part 3 of 3)

CHECKPOINT #5

At this point, you should have successfully completed the following tasks:

Table 13. Checkpoint #5

Task	Completed?
Modified cloud9.htm and invoked the Cloud 9 application?	
Logon onto the application, passing the security check?	
Viewed demo profile and updated with real data?	
Displayed members and ran SCLM jobs?	
Exited successfully from Cloud 9?	
Set up the backup, delete, and define JCL for the SLR?	

Part 2. S-JDK SCLM Java Development Kit

Chapter 7. Installation Overview for S-JDK

This part of the manual contains the installation procedure for the IBM Cloud 9 for SCLM for z/OS SCLM-Java Development Kit feature. Hereinafter, the following names will be used in this manual:

- IBM Cloud 9 for SCLM for z/OS will be called ***Cloud 9***
- IBM Cloud 9 for SCLM for z/OS S-JDK will be called ***S-JDK***

The steps in this part are organized into four major sections:

- Before you begin
- Customizing translators and translator control files
- Defining the S-JDK inventory, USS, and Cloud 9 parts
- Performing Installation Verification Procedures (IVP)

READ THIS FIRST!

This is not an 'out of the box' solution. You should not use global editing on these files. You must thoroughly understand your JAVA/USS environment before attaching the SCLM translators.

The SCLM part of this solution is standard SCLM. There are translators, types, languages and control files. The JAVA/USS side of the setup may be foreign to SCLM administrators so we recommend that you do the following before moving on to setting up the prototype translators.

Verify the Java/USS Environment

- Verify that you have access to USS environment. Check with your USS Administrator for information on your USS environment.
- Verify which USS directories contain the z/OS Java Compiler and Class Files.
- Obtain sample JCL to run a compile standalone in batch to verify that you have access and security rights in this environment.

Check Other Documents That Can be Useful

There are several Redbooks available from IBM concerning USS and JAVA.

- *Debugging Unix System Services*
- *e-business Enablement Cookbook for OS/390 Volumes 1,2,3*

For the publication order numbers for these books, see "Where to Find More Information" on page ix.

For information regarding USS setup the following may be useful, in particular Chapter 14: *UNIX System Services Planning* manual, GA22-7800-01.

Modifying Case Sensitive S-JDK Files

During this installation, you will modify several JCL members and USS files. Certain JCL members and all USS files contain case sensitive values. It is imperative that *before* modifying the JCL and USS members, you issue the CAPS OFF command to ensure that automatic upper casing does not occur. You will be reminded of this case sensitivity issue where appropriate throughout this manual.

Translators and Translator Control Files

S-JDK Translators

CLZTJTXT Translator for HTML Type

CLZTJBIN Translator for binary Graphic Types

CLZTJAVA Translator for JAVA

CLZTJAR Translator for JAR

S-JDK Translator Control Files

CLZTJCMP

- Input to CLZTJAR Translator
- Compile shell for JAR type

CLZTJMAP

- Input to CLZTJAR Translator
- USS Output Control for JAR type

CLZTJAVC

- Input to CLZTJAVA Translator
- Compile Shell for JAVA type

CLZTUMAP

- Input to CLZTJAVA Translator
- USS Output Control for JAVA type

CLZTCPTH

- Input to JAVA and JAR Translators
- %classpath% Substitution.

CLZTULOC

- Input to all S-JDK Translators
- SCLM to USS mapping rules.

CLZTHTPD

- Input to CLZTJAVA Translator
- ADDTYPE list for Java compiles

Chapter 8. A Step-by-Step Approach

Table 14. S-JDK Installation Steps

Before you begin...	
♦	Review system and software considerations.
Customize translators and translator control files	
Determine Inventory Values and Type Definitions	
1.	Review default S-JDK values and determine actual inventory to use.
2.	Review type definitions.
CP1.	Verify steps as shown in "CHECKPOINT #1 for S-JDK Installation" on page 67.
Define the S-JDK inventory, USS, and Cloud 9 parts	
3.	Review and modify all translators and translator control file members.
CP2.	Verify steps as shown in "CHECKPOINT #2 for S-JDK Installation" on page 81.
4.	Modify and run CLZTALIB,CLZTAVSM, and CLZTPDEF to build S-JDK Project Definitions
5.	Modify and run CLZC9J06 to define S-JDK types to Cloud 9
6.	Modify and run CLZTAUNX to define USS directories
7.	Review CLZJIBM Unix Shell
CP3.	Verify steps as shown in "CHECKPOINT #3 for S-JDK Installation" on page 98.
Perform Installation Verification Procedures (IVP)	
8.	Add Clock2.java and Clockh.html IVP programs
9.	Invoke Clockh.html to display Time of Day java IVP
CP4.	Verify steps as shown in "CHECKPOINT #4 for S-JDK Installation" on page 100.

Chapter 9. Before You Begin

Review Software and Assumptions

In this step you will review the system and software requirements for S-JDK installation.

System Requirements

To successfully install Cloud 9 S-JDK, the following system requirements must be in place at your installation:

Table 15. System Requirements

z/OS Operating System	Version 2 Release 7 (or higher)
SCLM	Standard z/OS
Java/USS	Enabled USS environment
IBM Cloud 9	Cloud 9 installed and configured

Assumptions

Java/USS Environment is already configured.

Users who will be Building and Promoting Java components will need to define to RACF (or other security product) an OMVS segment which includes a UID and home directory. The Userid associated with the Cloud 9 server also needs a UID and home directory. A TCP port needs to be available and it is good practice to reserve the port number. It is recommended that the Root HFS file system is made read only. The Cloud 9 directory can be set up as a separate file system mounted onto the root file system at /usr/lpp/Cloud9.

Chapter 10. Determine Inventory Values and Type Definitions

Step 1. Determine SCLM and USS Inventory Values

The S-JDK is set up with default values. These default values are meant to be modified throughout the various translators, HTML, and JCL members. Before making modifications to these members, please review the worksheets below and fill in your site specific inventory values.

It is important to view the SCLM Inventory and the USS directory structure as extensions to each other. Please review both tables prior to making decisions about the values.

SCLM Inventory Value Worksheet

Table 16. SCLM Inventory Value Worksheet

SCLM Inventory Name	Default Value	Your Values
Project	IBMDEMO	
Alt-project	IBMDEMO	
Group	DEV,QA,REL	
Types	JAVA,JAR,HTML,GRAPHICS	
Languages	JAVA,JAR,EBIZTEXT,EBIZJBIN	

USS Directory Value Worksheet

Table 17. USS Directory Value Worksheet

USS Mapping Locations	Default Value	Your Values
Listings	• /u/ibmdemo/dev/listings • /u/ibmdemo/qa/listings • /u/ibmdemo/rel/listings	
Classes	• /u/ibmdemo/dev/classes • /u/ibmdemo/qa/classes • /u/ibmdemo/rel/classes	
Graphics	• /u/ibmdemo/dev/graphics • /u/ibmdemo/qa/graphics • /u/ibmdemo/rel/graphics	
Html	• /u/ibmdemo/dev/html • /u/ibmdemo/qa/html • /u/ibmdemo/rel/html	
Jar	• /u/ibmdemo/dev/jar • /u/ibmdemo/qa/jar • /u/ibmdemo/rel/jar	

Step 2. Review SCLM and Cloud 9 Type Definitions

This step documents which Types need to be defined to SCLM and Cloud 9. Before moving on to the Translator and Control file modification, you should review all types selected for S-JDK. First, fill in each unique type in Table 18. Then, for each type, review the definition in SCLM and Cloud 9 SLR for existence, consistency, and correctness. If the types aren't defined yet, there is another step for updating the project definition and you can include the type definition process there ("Step 4: Update the Project Definition" on page 81). Note the following matrix is filled in with the default values.

Type Review Matrix

Table 18. Type Review Matrix

Type	Language	Defined to SCLM	Defined to Cloud 9 SLR	Case Sensitive Yes/No	Binary Yes/No	Lrecl	Recfm
Java	Java			Yes	No	256	VB
Jar	Jar			Yes	Yes	256	VB
Html	Ebiztext			Yes	No	256	VB
Graphics	Ebizbin			Yes	Yes	256	VB
Javaclas	Ebizbin			Yes	Yes	256	VB
Javalist	N/A			Yes	No	256	VB

Determining That Types Exist

To determine if a type exists, go to Cloud 9, list SCLM members. For the Group, list the types and verify that the types are there. If they do not exist, then you must define the types and data sets to SCLM.

Determining Cloud 9 Definitions

To determine if the SCLM type is defined to the Cloud 9 SLR (long name registry) run the IVP JCL member CLZC9J06. Review the output from the job, verifying that the SCLM has been defined with the proper attributes. If not, modify this job to define the Types to Cloud 9.

Determining LRECL and File Attributes

To determine the Logical Record Length (LRECL) of the type, go into SCLM Option 3.2 and display the data set information for one of the type datasets. If binary the Lrecl will be 256 and the RECFM = VB.

CHECKPOINT #1 for S-JDK Installation

At this point the following tasks should be completed.

Table 19. Checkpoint #1 for S-JDK Installation

Data Set Names	Completed?
Review all SCLM Inventory Default Values	
Review all USS Directory Default Values	
Determine actual SCLM Inventory and USS Directory Values	
Determine S-JDK Type Matrix	
Document SCLM Types and Cloud 9 definitions.	

Chapter 11. Define the S-JDK Inventory, USS, and Cloud 9 Parts

Step 3. Review and Modify Translators and Control Files

Review and Modify CLZTJAVA Translator

In this step you will review and modify the CLZTJAVA member found in the **CLZ.SCLZJCL** library. These members should be copied to your project definition source and updated to reference the control files listed in "Review and Modify Translator Control Files" on page 75.

Those lines that might need to be modified are identified by being highlighted boldly.

```
*--------------------------------------------------------------------*
*  NAME:     CLZTJAVA                                                *
*  PURPOSE:  S-JDK TRANSLATOR FOR TYPE = JAVA, LANGUAGE = JAVA.      *
*--------------------------------------------------------------------*
*  NOTES: THIS TRANSLATOR DOES NOT REQUIRE CUSTOMIZATION, HOWEVER,   *
*         THE VARIOUS INPUT FILES DO. BOTH CONTROL INPUT AND REXX    *
*         EXECUTABLE CODE ARE READ IN FROM THE PRODUCT LIBRARY       *
*         CLZ.SCLZCGI. YOU DO NOT HAVE TO MODIFY THE REXX EXECS.     *
*--------------------------------------------------------------------*
*         YOU DO HAVE TO REVIEW AND POSSIBLY MODIFY THE FOLLOWING    *
*         CONTROL FILES:                                             *
*          CLZ.SCLZCGI(CLZTCPTH)                                     *
*          CLZ.SCLZCGI(CLZTULOC)                                     *
*          CLZ.SCLZCGI(CLZTUMAP)                                     *
*          CLZ.SCLZCGI(CLZTJAVC)                                     *
*--------------------------------------------------------------------*
         FLMLANGL    LANG=JAVA,VERSION=TEXTV1.0
         FLMTRNSL FUNCTN=PARSE,                                      C
               COMPILE=FLMLPGEN,                                     C
               PORDER=1,                                             C
               OPTIONS=(LANG=T,                                      C
               LISTINFO=@@FLMLIS,                                    C
               LISTSIZE=@@FLMSIZ,                                    C
               SOURCEDD=SOURCE,                                      C
               STATINFO=@@FLMSTP)
         FLMALLOC DDNAME=SOURCE,IOTYPE=A
           FLMCPYLB @@FLMDSN(@@FLMMBR)
```

Figure 22. CLZTJAVA — Build and Promote S-JDK Translator (Part 1 of 2)

```
* ------------------------------------------------------------------ *
*                     BUILD TRANSLATOR                               *
* ------------------------------------------------------------------ *
         FLMTRNSL FUNCTN=BUILD,                                        C
               CALLNAM='COPY PACKAGE MEMBERS TO HFS',                  C
               COMPILE=IRXJCL,                                         C
               PDSDATA=Y,                                              C
               OPTIONS='CLZTRJV1'
         FLMALLOC IOTYPE=A,DDNAME=SYSEXEC
           FLMCPYLB CLZ.SCLZCGI
         FLMALLOC DDNAME=UNIXLOC,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTULOC)
         FLMALLOC DDNAME=HTTPD,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTHTPD)
* ------------------------------------------------------------------ *
*                     BUILD TRANSLATOR                               *
* ------------------------------------------------------------------ *
         FLMTRNSL FUNCTN=BUILD,                                        C
               CALLNAM='INVOKE JAVAC',                                 C
               COMPILE=IRXJCL,                                         C
               OPTIONS='CLZTRJVC @@FLMPRJ @@FLMALT @@FLMGRP @@FLMTYP @@C
               FLMMBR JAVACLAS JAVALIST'
         FLMALLOC IOTYPE=A,DDNAME=SYSEXEC
           FLMCPYLB CLZ.SCLZCGI
         FLMALLOC DDNAME=FILEIN,IOTYPE=S,KEYREF=SINC
         FLMALLOC DDNAME=UNIXMAP,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTUMAP)
         FLMALLOC DDNAME=CLASSPTH,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTCPTH)
         FLMALLOC DDNAME=JCOMPILE,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTJAVC)
         FLMALLOC DDNAME=SYSPRINT,IOTYPE=O,RECFM=FBA,LRECL=121,        C
               RECNUM=2500,PRINT=N
         FLMALLOC DDNAME=CLASSES,DFLTTYP=JAVACLAS,LANG=EBIZBIN,        C
               LRECL=256,BLKSIZE=27998,RECFM=VB,RECNUM=60000,          C
               IOTYPE=P,KEYREF=OUT1
         FLMALLOC DDNAME=JAVALIST,DFLTTYP=JAVALIST,LANG=EBIZTEXT,      C
               LRECL=256,BLKSIZE=27998,RECFM=VB,RECNUM=60000,          C
               IOTYPE=P,KEYREF=OUT2
* ------------------------------------------------------------------ *
*                   PROMOTE TRANSLATOR                               *
* ------------------------------------------------------------------ *
         FLMTRNSL FUNCTN=COPY,                                         C
               CALLNAM='UNIX PROMOTE',                                 C
               COMPILE=IRXJCL,                                         C
               PDSDATA=Y,                                              C
               OPTIONS='CLZTRJVP TEXT @@FLMPRJ @@FLMALT @@FLMGRP @@FLMTC
               YP @@FLMMBR @@FLMTOG'
         FLMALLOC IOTYPE=A,DDNAME=SYSEXEC
           FLMCPYLB CLZ.SCLZCGI
         FLMALLOC DDNAME=FILEIN,IOTYPE=A
           FLMCPYLB @@FLMDSN(@@FLMMBR)
         FLMALLOC DDNAME=UNIXLOC,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTULOC)
* ------------------------------------------------------------------ *
```

Figure 22. CLZTJAVA — Build and Promote S-JDK Translator (Part 2 of 2)

Review and Modify CLZTJAR Translator

In this step you will review and modify the CLZTJAR member found in the **CLZ.SCLZJCL** library delivered with the SMP/E installation of Cloud 9.

Those lines that might need to be modified are identified by being highlighted boldly.

```
*-------------------------------------------------------------------*
*  NAME:     CLZTJAR                                                *
*  PURPOSE:  S-JDK TRANSLATOR FOR TYPE = JAR, LANGUAGE = JAR.       *
*-------------------------------------------------------------------*
*  NOTES: THIS TRANSLATOR DOES NOT REQUIRE CUSTOMIZATION, HOWEVER,  *
*         THE VARIOUS INPUT FILES DO. BOTH CONTROL INPUT AND REXX   *
*         EXECUTABLE CODE ARE READ IN FROM THE PRODUCT LIBRARY      *
*         CLZ.SCLZCGI. YOU DO NOT HAVE TO MODIFY THE REXX EXECS.    *
*-------------------------------------------------------------------*
*         YOU DO HAVE TO REVIEW AND POSSIBLY MODIFY THE FOLLOWING   *
*         CONTROL FILES:                                            *
*          CLZ.SCLZCGI(CLZTJMAP)                                    *
*          CLZ.SCLZCGI(CLZTCPTH)                                    *
*          CLZ.SCLZCGI(CLZTJCMP)                                    *
*          CLZ.SCLZCGI(CLZTULOC)                                    *
*-------------------------------------------------------------------*
         FLMLANGL    LANG=JAR,VERSION=TEXTV1.0
         FLMTRNSL FUNCTN=PARSE,                                     C
               COMPILE=FLMLPGEN,                                    C
               PORDER=1,                                            C
               OPTIONS=(LANG=T,                                     C
               LISTINFO=@@FLMLIS,                                   C
               LISTSIZE=@@FLMSIZ,                                   C
               SOURCEDD=SOURCE,                                     C
               STATINFO=@@FLMSTP)
         FLMALLOC DDNAME=SOURCE,IOTYPE=A
           FLMCPYLB @@FLMDSN(@@FLMMBR)
* ------------------------------------------------------------------ *
*                     BUILD TRANSLATOR                               *
* ------------------------------------------------------------------ *
         FLMTRNSL FUNCTN=BUILD,                                      C
               CALLNAM='INVOKE JAR',                                C
               COMPILE=IRXJCL,                                      C
               OPTIONS='CLZTRJAR @@FLMPRJ @@FLMALT @@FLMGRP @@FLMTYP @@C
               FLMMBR JAR JARLIST'
         FLMALLOC IOTYPE=A,DDNAME=SYSEXEC
           FLMCPYLB CLZ.SCLZCGI
         FLMALLOC DDNAME=FILEIN,IOTYPE=S,KEYREF=SINC
         FLMALLOC DDNAME=JARMAP,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTJMAP)
         FLMALLOC DDNAME=CLASSPTH,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTCPTH)
         FLMALLOC DDNAME=UNIXLOC,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTULOC)
         FLMALLOC DDNAME=JARCOMP,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTJCMP)
         FLMALLOC DDNAME=JARLIST,IOTYPE=O,PRINT=Y,RECNUM=60000,      C
               DFLTTYP=JARLIST,KEYREF=LIST,LRECL=256,LANG=EBIZTEXT
         FLMALLOC DDNAME=JAR,IOTYPE=O,PRINT=Y,RECNUM=60000,          C
               DFLTTYP=JAR,KEYREF=OBJ,LRECL=256,LANG=EBIZBIN
         FLMALLOC DDNAME=SYSPRINT,IOTYPE=O,RECFM=FBA,LRECL=121,      C
               RECNUM=2500,PRINT=N
```

Figure 23. CLZTJAR — Build and Promote JAR S-JDK Translator (Part 1 of 2)

```
* ------------------------------------------------------------------ *
*                   PROMOTE TRANSLATOR                               *
* ------------------------------------------------------------------ *
         FLMTRNSL FUNCTN=COPY,                                        C
               CALLNAM='UNIX PROMOTE',                                C
               COMPILE=IRXJCL,                                        C
               PDSDATA=Y,                                             C
               OPTIONS='CLZTRJVP BINARY @@FLMPRJ @@FLMALT @@FLMGRP @@FLC
               MTYP @@FLMMBR @@FLMTOG'
         FLMALLOC IOTYPE=A,DDNAME=SYSEXEC
           FLMCPYLB CLZ.SCLZCGI
         FLMALLOC DDNAME=FILEIN,IOTYPE=A
           FLMCPYLB @@FLMDSN(@@FLMMBR)
         FLMALLOC DDNAME=UNIXLOC,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTULOC)
* ------------------------------------------------------------------ *
```

Figure 23. CLZTJAR — Build and Promote JAR S-JDK Translator (Part 2 of 2)

Review and Modify CLZTJTXT Translator

In this step you will review and modify the CLZTJTXT member found in the **CLZ.SCLZJCL** library delivered with the SMP/E installation of Cloud 9.

Those lines that might need to be modified are identified by being highlighted boldly.

```
*-------------------------------------------------------------------*
*  NAME:     CLZTJTXT                                               *
*  PURPOSE:  S-JDK TRANSLATOR FOR TYPE=HTML,LANGUAGE=EBIZTEXT       *
*-------------------------------------------------------------------*
*  NOTES: THIS TRANSLATOR DOES NOT REQUIRE CUSTOMIZATION, HOWEVER,  *
*         THE VARIOUS INPUT FILES DO. BOTH CONTROL INPUT AND REXX   *
*         EXECUTABLE CODE ARE READ IN FROM THE PRODUCT LIBRARY      *
*         CLZ.SCLZCGI. YOU DO NOT HAVE TO MODIFY THE REXX EXECS.    *
*-------------------------------------------------------------------*
*         YOU DO HAVE TO REVIEW AND POSSIBLY MODIFY THE FOLLOWING   *
*         CONTROL FILES:                                            *
*          CLZ.SCLZCGI(CLZTULOC)                                    *
*-------------------------------------------------------------------*
         FLMLANGL    LANG=EBIZTEXT,VERSION=TEXTV1.0
         FLMTRNSL FUNCTN=PARSE,                                     C
               COMPILE=FLMLPGEN,                                    C
               PORDER=1,                                            C
               OPTIONS=(LANG=T,                                     C
               LISTINFO=@@FLMLIS,                                   C
               LISTSIZE=@@FLMSIZ,                                   C
               SOURCEDD=SOURCE,                                     C
               STATINFO=@@FLMSTP)
         FLMALLOC DDNAME=SOURCE,IOTYPE=A
           FLMCPYLB @@FLMDSN(@@FLMMBR)
* ------------------------------------------------------------------ *
*                  BUILD TRANSLATOR                                  *
* ------------------------------------------------------------------ *
         FLMTRNSL FUNCTN=BUILD,                                     C
               CALLNAM='UNIX BUILD',                                C
               COMPILE=CLZTRJVP,                                    C
               CALLMETH=TSOLNK,                                     C
               DSNAME=CLZ.SCLZCGI,                                  C
               OPTIONS='TEXT @@FLMPRJ @@FLMALT @@FLMGRP @@FLMTYP @@FLMMC
               BR @@FLMTOG @@FLMBIO'
         FLMALLOC DDNAME=FILEIN,IOTYPE=S,KEYREF=SINC
         FLMALLOC DDNAME=UNIXLOC,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTULOC)
         FLMALLOC DDNAME=WORKFILE,IOTYPE=W,LRECL=32000,RECFM=V
* ------------------------------------------------------------------ *
*                  PROMOTE  TRANSLATOR                               *
* ------------------------------------------------------------------ *
         FLMTRNSL FUNCTN=COPY,                                      C
               CALLNAM='UNIX PROMOTE',                              C
               COMPILE=CLZTRJVP,                                    C
               CALLMETH=TSOLNK,                                     C
               DSNAME=CLZ.SCLZCGI,                                  C
               PDSDATA=Y,                                           C
               OPTIONS='TEXT @@FLMPRJ @@FLMALT @@FLMGRP @@FLMTYP @@FLMMC
               BR @@FLMTOG'
         FLMALLOC DDNAME=FILEIN,IOTYPE=A
           FLMCPYLB @@FLMDSN(@@FLMMBR)
         FLMALLOC DDNAME=UNIXLOC,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTULOC)
         FLMALLOC DDNAME=WORKFILE,IOTYPE=W,LRECL=32000,RECFM=V
* ------------------------------------------------------------------ *
```

Figure 24. CLZTJTXT — Build and Promote Text S-JDK Translator

Review and Modify CLZTJBIN Translator

In this step you will review and modify the CLZTJBIN member found in the **CLZ.SCLZJCL** library delivered with the SMP/E installation of Cloud 9.

Those lines that might need to be modified are identified by being highlighted boldly.

```
*-------------------------------------------------------------------*
*  NAME:     CLZTJBIN                                               *
*  PURPOSE:  S-JDK TRANSLATOR FOR TYPE=GRAPHICS,LANGUAGE=EBIZBIN    *
*-------------------------------------------------------------------*
*  NOTES: THIS TRANSLATOR DOES NOT REQUIRE CUSTOMIZATION, HOWEVER,  *
*         THE VARIOUS INPUT FILES DO. BOTH CONTROL INPUT AND REXX   *
*         EXECUTABLE CODE ARE READ IN FROM THE PRODUCT LIBRARY      *
*         CLZ.SCLZCGI. YOU DO NOT HAVE TO MODIFY THE REXX EXECS.    *
*-------------------------------------------------------------------*
*         YOU DO HAVE TO REVIEW AND POSSIBLY MODIFY THE FOLLOWING   *
*         CONTROL FILES:                                            *
*          CLZ.SCLZCGI(CLZTULOC)                                    *
*-------------------------------------------------------------------*
         FLMLANGL    LANG=EBIZBIN,VERSION=TEXTV1.0
         FLMTRNSL FUNCTN=PARSE,                                     C
               COMPILE=FLMLPGEN,                                    C
               PORDER=1,                                            C
               OPTIONS=(LANG=T,                                     C
               LISTINFO=@@FLMLIS,                                   C
               LISTSIZE=@@FLMSIZ,                                   C
               SOURCEDD=SOURCE,                                     C
               STATINFO=@@FLMSTP)
         FLMALLOC DDNAME=SOURCE,IOTYPE=A
           FLMCPYLB @@FLMDSN(@@FLMMBR)
* ------------------------------------------------------------------ *
*                   BUILD TRANSLATOR                                 *
* ------------------------------------------------------------------ *
         FLMTRNSL FUNCTN=BUILD,                                      C
               CALLNAM='UNIX BUILD',                                C
               COMPILE=CLZTRJVP,                                    C
               CALLMETH=TSOLNK,                                     C
               DSNAME=CLZ.SCLZCGI,                                  C
               OPTIONS='BINARY @@FLMPRJ @@FLMALT @@FLMGRP @@FLMTYP @@FLC
               MMBR @@FLMTOG'
         FLMALLOC DDNAME=FILEIN,IOTYPE=S,KEYREF=SINC
         FLMALLOC DDNAME=UNIXLOC,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTULOC)
         FLMALLOC DDNAME=WORKFILE,IOTYPE=W,LRECL=32000,RECFM=V
* ------------------------------------------------------------------ *
*                   PROMOTE  TRANSLATOR                              *
* ------------------------------------------------------------------ *
         FLMTRNSL FUNCTN=COPY,                                       C
               CALLNAM='UNIX PROMOTE',                              C
               COMPILE=CLZTRJVP,                                    C
               CALLMETH=TSOLNK,                                     C
               DSNAME=CLZ.SCLZCGI,                                  C
               PDSDATA=Y,                                           C
               OPTIONS='BINARY @@FLMPRJ @@FLMALT @@FLMGRP @@FLMTYP @@FLC
               MMBR @@FLMTOG'
         FLMALLOC DDNAME=FILEIN,IOTYPE=A
           FLMCPYLB @@FLMDSN(@@FLMMBR)
         FLMALLOC DDNAME=UNIXLOC,IOTYPE=A
           FLMCPYLB CLZ.SCLZCGI(CLZTULOC)
         FLMALLOC DDNAME=WORKFILE,IOTYPE=W,LRECL=32000,RECFM=V
* ------------------------------------------------------------------ *
```

Figure 25. CLZTJBIN — Build and Promote Binary S-JDK Translator

Review and Modify Translator Control Files

CASE SENSITIVITY ALERT!

The following translator control files contain case sensitive data. To ensure case sensitivity is in place, please issue the 'CAPS OFF' command on the command line of your ISPF session.

Modify CLZTULOC — Common SCLM to USS Life Cycle Mapping Rules

In this step you will review and modify the CLZTULOC member found in the SCLZCGI library.

This member is input to all S-JDK translators. It is used to map the SCLM to USS life cycles locations. This control file is used in the Build, Promote and Delete process.

```
* ------------------------------------------------------------- *
* CLOUD 9 JAVA/USS S-JDK COMPONENT                              *
* ------------------------------------------------------------- *
* NAME:    CLZTULOC                                             *
* PURPOSE: SCLM TO UNIX LIFE CYCLE MAPPING RULES.               *
* REFER:   DIRECTLY REFERENCED IN TRANSLATOR DDNAME UNIXLOC.    *
* ------------------------------------------------------------- *
* prj,alt,grp,typ            unix location     KEEP or DELETE
*                                                on promote  Permissions
IBMDEMO,IBMDEMO,DEV,GRAPHICS /u/ibmdemo/dev/graphics  KEEP   PERM=777
IBMDEMO,IBMDEMO,DEV,HTML     /u/ibmdemo/dev           KEEP   PERM=777
IBMDEMO,IBMDEMO,DEV,HTML     /u/ibmdemo/dev/html      KEEP   PERM=775
IBMDEMO,IBMDEMO,DEV,JAVA     /u/ibmdemo/dev           KEEP   PERM=777
IBMDEMO,IBMDEMO,DEV,JAVACLAS /u/ibmdemo/dev/classes   KEEP   PERM=775
IBMDEMO,IBMDEMO,DEV,JAVALIST /u/ibmdemo/dev/listings  KEEP   PERM=775
*
IBMDEMO,IBMDEMO,QA,GRAPHICS  /u/ibmdemo/qa/graphics   KEEP
IBMDEMO,IBMDEMO,QA,HTML      /u/ibmdemo/qa            KEEP
IBMDEMO,IBMDEMO,QA,HTML      /u/ibmdemo/qa/html       KEEP
IBMDEMO,IBMDEMO,QA,JAVA      /u/ibmdemo/qa            KEEP
IBMDEMO,IBMDEMO,QA,JAVACLAS  /u/ibmdemo/qa/classes    KEEP
IBMDEMO,IBMDEMO,QA,JAVALIST  /u/ibmdemo/qa/listings   KEEP
*
IBMDEMO,IBMDEMO,REL,GRAPHICS /u/ibmdemo/rel/graphics  KEEP
IBMDEMO,IBMDEMO,REL,HTML     /u/ibmdemo/rel           KEEP
IBMDEMO,IBMDEMO,REL,HTML     /u/ibmdemo/rel/html      KEEP
IBMDEMO,IBMDEMO,REL,JAVA     /u/ibmdemo/rel           KEEP
IBMDEMO,IBMDEMO,REL,JAVACLAS /u/ibmdemo/rel/classes   KEEP
IBMDEMO,IBMDEMO,REL,JAVALIST /u/ibmdemo/rel/listings  KEEP
```

Figure 26. CLZTULOC — Common SCLM to USS Life Cycle Map

Position 1
SCLM Life Cycle location

Position 2
Unix System Services Life Cycle location

Position 3
Dispostion of files on promote

Position 4
Permissions to be allocated to files created in the specified directory. This parameter is used to give different Unix permissions to files of different types, and also to files at different levels in the hierarchy.

Modify CLZTCPTH — Common %CLASSPATH% Substitution

In this step you will review and modify the CLZTCPTH member found in the SCLZCGI library.

This member is input to both the Java and Jar compile shells. It is used to fill in the %Classpath% variable in the compiler shells.

```
* ------------------------------------------------------------- *
*  CLOUD 9 JAVA/USS S-JDK COMPONENT                             *
* ------------------------------------------------------------- *
* NAME:    CLZTCPTH                                            *
* PURPOSE: %CLASSPATH% SUBSTITUTION FILE.                      *
* REFER:   DIRECTLY REFERENCED IN TRANSLATOR DDNAME CLASSPTH   *
* ------------------------------------------------------------- *
* prj,alt,gr        classpath and java source concatenation
IBMDEMO,IBMDEMO,DEV /u/ibmdemo/dev/classes
IBMDEMO,IBMDEMO,DEV /u/ibmdemo/dev/java
IBMDEMO,IBMDEMO,DEV /u/ibmdemo/qa/classes
IBMDEMO,IBMDEMO,DEV /u/ibmdemo/qa/java
IBMDEMO,IBMDEMO,DEV /u/ibmdemo/rel/classes
IBMDEMO,IBMDEMO,DEV /u/ibmdemo/rel/java
```

Figure 27. CLZTCPTH — Common %CLASSPATH% Substitution

This is the SCLM hierarchy Classpath allocation control member. It tells the Java compile where to find additional Class files that are within the SCLM hierarchy. The reason the Java source libraries are included in the concatenation is due to the way Java works. If Java come across a Class file it doesn't have in the class directory it will look in the source directory for the Java source. The Java compiler will then compile the source to create the required class file. Including the source directory as part of the classpath allocation enables Java to find the source in cases where the class file is not in the class path.

Note: The actual class path may be more complex than one shown above. For instance the core Java class libraries may reside in directories outside of the standard SCLM /USS life cycle.

Modify CLZTJAVC — Java Compile Shell

In this step you will review and modify the CLZTJAVC member found in the SCLZCGI library.

This member is input to the CLZTJAVA translator for the Java Type. (You might need to review the path location with your USS System Programmer.) The %CLASSPATH% variable is built by Cloud 9 from the values found in the CLZTCPTH member.

```
# ------------------------------------------------------------- #
# CLOUD 9 JAVA/USS S-JDK COMPONENT                              #
# ------------------------------------------------------------- #
# NAME:    CLZTJAVC                                             #
# PURPOSE: JAVA COMPILE SHELL                                   #
# REFER:   DIRECTLY REFERENCED IN THE TRANSLATOR DDNAME JCOMPILE. #
# ------------------------------------------------------------- #
export PATH=/usr/lpp/java/J1.1/bin:$PATH
export CLASSPATH=%CLASSPATH%:$CLASSPATH
cd $1
javac -verbose -d $3 $2
```

Figure 28. CLZTJAVC — Java Compile Shell

Note: The actual class path may be more complex than one shown above. For instance the core Java class libraries may reside in directories outside of the standard SCLM /USS life cycle.

Modify CLZTUMAP — Java Output Mapping Rules

In this step you will review and modify the CLZTUMAP member found in the SCLZCGI library.

This member is the default input to the CLZTJAVA translator for the Java Type. It is used to define USS outputs for Java Compiles.

```
* ------------------------------------------------------------- *
*  CLOUD 9 JAVA/USS S-JDK COMPONENT                             *
* ------------------------------------------------------------- *
* NAME:    CLZTUMAP                                             *
* PURPOSE: MAPPING RULES FOR JAVA COMPILE USS OUTPUT LOCATIONS.   *
* REFER:   DIRECTLY REFERENCED IN TRANSLATOR DDNAME UNIXMAP.    *
* ------------------------------------------------------------- *
*prj,alt,grp             java source         java class             java listing
IBMDEMO,IBMDEMO,DEV,JAVA /u/ibmdemo/dev/java /u/ibmdemo/dev/classes /u/ibmdemo/dev/listings
IBMDEMO,IBMDEMO,QA,JAVA  /u/ibmdemo/qa/java  /u/ibmdemo/qa/classes  /u/ibmdemo/qa/listings
IBMDEMO,IBMDEMO,REL,JAVA /u/ibmdemo/rel/java /u/ibmdemo/rel/classes /u/ibmdemo/rel/listings
*
```

Figure 29. CLZTUMAP — Java Output Mapping Rules

Position 1
SCLM Java Location

Position 2
Unix Source for Javac Compiler

Position 3
Unix Target for Compiler Generated Classes

Position 4
Unix Target for Compiler Generated Listings

Modify CLZTJCMP — Jar Compile Shell

In this step you will review and modify the CLZTJCMP member found in the SCLZCGI library.

This member is the default input to the CLZTJAR translator for the JAR Type.

```
# ------------------------------------------------------------- #
# CLOUD 9 JAVA/USS S-JDK COMPONENT                              #
# ------------------------------------------------------------- #
# NAME:    CLZTJCMP                                             #
# PURPOSE: JAR COMPILE SHELL                                    #
# REFER:   DIRECTLY REFERENCED IN TRANSLATOR DDNAME JARCOMP.    #
# ------------------------------------------------------------- #
export PATH=/usr/lpp/java/J1.1/bin:$PATH
export CLASSPATH=%CLASSPATH%:$CLASSPATH
cd %CURDIR%
jar cfv %FILENAME% \
%SOURCE%
chmod -fR 777 %FILENAME%
jar tvf %FILENAME%
```

Figure 30. CLZTJCMP — Jar Compile Shell

Modify CLZTJMAP — Jar Output Mapping Rules

In this step you will review and modify the CLZTJMAP member found in the SCLZCGI library.

This member is the default input to the CLZTJAR translator for the Jar Type. It is used to define USS output locations for Jar Compiles.

```
* ------------------------------------------------------------- *
*  CLOUD 9 JAVA/USS S-JDK COMPONENT                             *
* ------------------------------------------------------------- *
* NAME:    CLZTJMAP                                             *
* PURPOSE: MAPPING RULES FOR JAR COMPILE USS OUTPUT LOCATIONS.  *
* REFER:   DIRECTLY REFERENCED IN TRANSLATOR DDNAME JARMAP.     *
* ------------------------------------------------------------- *
*prj,alt,grp                jar-cur-dir           jar-exec-loc
jar-listing-loc
IBMDEMO,IBMDEMO,DEV,JAVAMAKE /u/ibmdemo/dev/classes /u/ibmdemo/dev/jar
/u/ibmdemo/dev/listings
```

Figure 31. CLZTJMAP — Jar Output Mapping Rules

Position 1
SCLM location for JAR Files

Position 2
Unix Source for Classes to put into JAR

Position 3
Unix Target for JAR Compile

Modify CLZTHTPD — Addtype list for Java compile

In this step you will review and modify the CLZTHTPD member found in the SCLZCGI target library.

This member contains Addtype definitions, similar to those that can be found in the CLZHTTPD member in the SCLZHTML target library. Addtype definitions are used by the Java compile process to determine if objects included in the Java compile are Binary or Text. This is required by the copy function that is part of the compile process.

```
#--------------------------------------------------------------------
#  Name:      CLZTHTPD
#  Purpose:   Cloud9 Server Rules File
#  Usage:     This file is used by the Java compile process
#--------------------------------------------------------------------
#-------------------------------------------------------------------------------------------------
#
#Non-standard MIME types declared here.  (User style MIME types)
#
#-------------------------------------------------------------------------------------------------
AddType  .asm      text/asm                          ebcdic  1.0 # Assemble Macros
AddType  .doc      binary/doc                        binary  1.0 # Microsoft Word Documents
AddType  .ppt      binary/ppt                        binary  1.0 # Power Point Documents
AddType  .cob      text/cobol                        ebcdic  1.0 # COBOL Source Code
AddType  .cbl      text/cobol                        ebcdic  1.0 # COBOL Source Code
AddType  .cobol    text/cobol                        ebcdic  1.0 # COBOL Source Code
#-------------------------------------------------------------------------------------------------
#
AddType  .cer      application/x-x509-user-cert      ebcdic  0.5 # Browser Certificate
AddType  .der      application/x-x509-ca-cert        binary  1.0 # CA Certificate
AddType  .mime     www/mime                          binary  1.0 # Internal -- MIME is
AddType  .bin      application/octet-stream          binary  1.0 # Uninterpreted binary
AddType  .class    application/octet-stream          binary  1.0 # Java applet or application
AddType  .pdf      application/pdf                   binary  1.0
AddType  .ai       application/postscript            ebcdic  0.5 # Adobe Illustrator
AddType  .PS       application/postscript            ebcdic  0.8 # PostScript
AddType  .eps      application/postscript            ebcdic  0.8
AddType  .ps       application/postscript            ebcdic  0.8
AddType  .rtf      application/x-rtf                 ebcdic  1.0 # RTF
AddType  .csh      application/x-csh                 ebcdic  0.5 # C-shell script
AddType  .latex    application/x-latex               ebcdic  1.0 # LaTeX source
AddType  .cdf      application/x-cdf                 ebcdic  1.0 # Channel Definition Format
AddType  .sh       application/x-sh                  ebcdic  0.5 # Shell-script
AddType  .tcl      application/x-tcl                 ebcdic  0.5 # TCL-script
AddType  .tex      application/x-tex                 ebcdic  1.0 # TeX source
AddType  .t        application/x-troff               ebcdic  0.5 # Troff
AddType  .roff     application/x-troff               ebcdic  0.5
AddType  .tr       application/x-troff               ebcdic  0.5
AddType  .man      application/x-troff-man           ebcdic  0.5 # Troff with man macros
AddType  .me       application/x-troff-me            ebcdic  0.5 # Troff with me macros
AddType  .ms       application/x-troff-ms            ebcdic  0.5 # Troff with ms macros
AddType  .gtar     application/x-gtar                binary  1.0 # Gnu tar
AddType  .shar     application/x-shar                ebcdic  1.0 # Shell archive
AddType  .wrl      x-world/x-vrml                   binary  1.0 # VRML
AddType  .snd      audio/basic                      binary  1.0 # Audio
AddType  .au       audio/basic                      binary  1.0
AddType  .aiff     audio/x-aiff                     binary  1.0
AddType  .aifc     audio/x-aiff                     binary  1.0
AddType  .aif      audio/x-aiff                     binary  1.0
AddType  .wav      audio/x-wav                      binary  1.0 # Windows+ WAVE format
AddType  .bmp      image/bmp                        binary  1.0 # OS/2 bitmap format
AddType  .gif      image/gif                        binary  1.0 # GIF
AddType  .ief      image/ief                        binary  1.0 # Image Exchange fmt
AddType  .jpg      image/jpeg                       binary  1.0 # JPEG
AddType  .JPG      image/jpeg                       binary  1.0
AddType  .JPE      image/jpeg                       binary  1.0
AddType  .jpe      image/jpeg                       binary  1.0
AddType  .JPEG     image/jpeg                       binary  1.0
AddType  .jpeg     image/jpeg                       binary  1.0
AddType  .tif      image/tiff                       binary  1.0 # TIFF
AddType  .tiff     image/tiff                       binary  1.0
AddType  .ras      image/cmu-raster                 binary  1.0
AddType  .pnm      image/x-portable-anymap          binary  1.0 # PBM Anymap format
AddType  .pbm      image/x-portable-bitmap          binary  1.0 # PBM Bitmap format
AddType  .pgm      image/x-portable-graymap         binary  1.0 # PBM Graymap format
AddType  .ppm      image/x-portable-pixmap          binary  1.0 # PBM Pixmap format
AddType  .rgb      image/x-rgb                      binary  1.0
AddType  .xbm      image/x-xbitmap                  ebcdic  1.0 # X bitmap
AddType  .xpm      image/x-xpixmap                  binary  1.0 # X pixmap format
AddType  .xwd      image/x-xwindowdump              binary  1.0 # X window dump (xwd)
AddType  .html     text/html                        ebcdic  1.0 # HTML
AddType  .htm      text/html                        ebcdic  1.0 # HTML on PCs
AddType  .htmls    text/x-ssi-html                  ebcdic  1.0 # Server-side includes
AddType  .shtml    text/x-ssi-html                  ebcdic  1.0 # Server-side includes
```

Figure 32. CLZTHTPD — Cloud9 Server Rules (Part 1 of 2)

```
AddType  .c        text/plain                      ebcdic  0.5 # C source
AddType  .h        text/plain                      ebcdic  0.5 # C headers
AddType  .C        text/plain                      ebcdic  0.5 # C++ source
AddType  .cc       text/plain                      ebcdic  0.5 # C++ source
AddType  .hh       text/plain                      ebcdic  0.5 # C++ headers
AddType  .java     text/plain                      ebcdic  0.5 # Java source
AddType  .js       text/plain                      ebcdic  0.5 # JavaScript source
AddType  .m        text/plain                      ebcdic  0.5 # Objective-C source
AddType  .f90      text/plain                      ebcdic  0.5 # Fortran 90 source
AddType  .txt      text/plain                      ebcdic  0.5 # Plain text
AddType  .bat      text/plain                      ebcdic  0.5 # Plain text
AddType  .css      text/css                        8bit    1.0 # W3C Cascading Style Sheets
AddType  .rtx      text/richtext                   ebcdic  1.0 # MIME Richtext format
AddType  .tsv      text/tab-separated-values       ebcdic  1.0 # Tab-separated values
AddType  .etx      text/x-setext                   ebcdic  0.9 # Struct Enchanced Txt
AddType  .MPG      video/mpeg                      binary  1.0 # MPEG
AddType  .mpg      video/mpeg                      binary  1.0
AddType  .MPE      video/mpeg                      binary  1.0
AddType  .mpe      video/mpeg                      binary  1.0
AddType  .MPEG     video/mpeg                      binary  1.0
AddType  .mpeg     video/mpeg                      binary  1.0
AddType  .qt       video/quicktime                 binary  1.0 # QuickTime
AddType  .mov      video/quicktime                 binary  1.0
AddType  .avi      video/x-msvideo                 binary  1.0 # MS Video for Windows
AddType  .movie    video/x-sgi-movie               binary  1.0 # SGI  moviepalyer
AddType  .zip      multipart/x-zip                 binary  1.0 # PKZIP
AddType  .tar      multipart/x-tar                 binary  1.0 # 4.3BSD tar
AddType  .ustar    multipart/x-ustar               binary  1.0 # POSIX tar
AddType  *.*       www/unknown                     binary  0.2 # Try to guess
AddType  *         www/unknown                     binary  0.2 # Try to guess
AddType  .cxx      text/plain                      ebcdic  0.5 # C++
AddType  .for      text/plain                      ebcdic  0.5 # Fortran
AddType  .mar      text/plain                      ebcdic  0.5 # MACRO
AddType  .log      text/plain                      ebcdic  0.5 # logfiles
AddType  .com      text/plain                      ebcdic  0.5 # scripts
AddType  .sdml     text/plain                      ebcdic  0.5 # SDML
AddType  .list     text/plain                      ebcdic  0.5 # listfiles
AddType  .lst      text/plain                      ebcdic  0.5 # listfiles
AddType  .def      text/plain                      ebcdic  0.5 # definition files
AddType  .conf     text/plain                      ebcdic  0.5 # definition files
AddType  .         text/plain                      ebcdic  0.5 # files with no extension
AddType  .JP932    text/x-DBCS                     binary  1.0 IBM-932  # Japanese DBCS
AddType  .JPeuc    text/x-DBCS                     binary  1.0 IBMeucJP # Japanese DBCS
```

Figure 32. CLZTHTPD — Cloud9 Server Rules (Part 2 of 2)

Position 1
Addtype keyword

Position 2
file extension

Position 3
MIME type and subtype you want to bind to files that match the corresponding suffix pattern.

Position 4
The MIME content encoding to which the data has been converted.

Position 5
An optional indicator of relative value (on a scale of 0.0 to 1.0) for the content type.

Position 6
Description to associate with the document

For more information on the Addtype directives refer to the *HTTP Server Planning, Installing, and Using* manual (SC34-4826-00).

CHECKPOINT #2 for S-JDK Installation

At this point you should have completed the following tasks.

Table 20. Checkpoint #2 for S-JDK Installation

Task	Completed?
Reviewed and modified JAVA translator CLZTJAVA	
Reviewed and modified JAR translator CLZTJAR	
Reviewed and modified TEXT translator CLZTJTXT	
Reviewed and modified GRAPHICS translator CLZTJBIN	
Reviewed and modified the USS to SCLM mapping control file CLZTULOC	
Reviewed and modified Classpath control file CLZTCPTH	
Reviewed and modified JAVA compile shell CLZTJAVC	
Reviewed and modified JAVA Output mapping control file CLZTUMAP	
Reviewed and modified JAR compile shell CLZTJCMP	
Reviewed and modified JAR Output mapping control file CLZTJMAP	

Step 4: Update the Project Definition

To complete the enabling of the S-JDK, the current SCLM Project Definition must be updated with both the S-JDK types and the S-JDK translators. These definitions must be included with the current project definitions that define your existing project or used as a standalone SCLM Project. The following section shows the Type definitions and Translator copy statements required to define the S-JDK defaults. Typically, these types and translators will be included in a common project. The following member shows a stand-alone project definition JCL stream containing the Type and translator definitions.

The following member JCL member, CLZTPDEF, can be found in the CLZ.SCLZJCL library. This is meant as an example. Please review with your SCLM administrator about whether the new types will be an isolated project or part of an existing project.

```
//* (JOBCARD)
//*--------------------------------------------------------------------*
//*  NAME:     CLZTPDEF                                                *
//*  PURPOSE:  S-JDK STANDALONE PROJECT DEFINITION.                    *
//*--------------------------------------------------------------------*
//*  TO USE THIS JCL YOU MUST:                                         *
//*         1) INSERT A VALID JOB CARD.                                *
//*         2) REVIEW THE SCLM VALUES - THEY ARE INITIALLY SET TO      *
//*            THE S-JDK DEFAULT VALUES.                               *
//*         3) REVIEW THE DATASET NAMES - THEY ARE INITIALLY SET TO    *
//*            THE S-JDK DEFAULT VALUES.                               *
//*         4) MODIFY THE DATASET NAMES TO MATCH ACTUAL CHOSEN S-JDK   *
//*            TYPES.                                                  *
//*         5) MODIFY THE SCLM VALUES TO MATCH ACTUAL CHOSEN S-JDK     *
//*            TYPES.                                                  *
//*         6) REVIEW AND MODIFY THE S-JDK TRANSLATORS AND LANGUAGE    *
//*            NAMES TO THE CHOSE TYPES.                               *
//*         7) CHANGE THE UNIT=TDISK TO THE APPROPRIATE UNIT FOR       *
//*            TEMPORARY FILES.                                        *
//COMP   PROC
//*--------------------------------------------------------------------*
//STEP1ASM EXEC PGM=ASMA90,REGION=3072K,COND=(0,NE),
//             PARM='NODECK,OBJECT,NOTERM,LIST,XREF(SHORT)'
//SYSLIB   DD  DSN=CLZ.SCLZJCL,DISP=SHR
//         DD  DSN=ISP.SISPMACS,DISP=SHR
//SYSPUNCH DD  DUMMY
//SYSUT1   DD  UNIT=TDISK,SPACE=(TRK,(5,15))
//SYSPRINT DD  SYSOUT=*
//    PEND
//*--------------------------------------------------------------------*
//LINK   PROC
//STEP2LNK EXEC PGM=IEWL,REGION=2048K,COND=(0,NE),
//             PARM='LIST,XREF,LET,RENT,REUS,CALL'
//SYSPRINT DD  SYSOUT=*
```

Figure 33. Sample IBMDEMO Project Definition (CLZTPDEF) (Part 1 of 3)

```
//SYSLIB   DD  DSN=IBMDEMO.PROJDEFS.OBJLIB,DISP=SHR
//SYSLMOD  DD  DSN=IBMDEMO.PROJDEFS.LOAD,DISP=SHR
//SYSUT1   DD  UNIT=TDISK,SPACE=(TRK,(5,15))
//*-------------------------------------------------------------------*
//    PEND
//COMPILE   EXEC COMP
//SYSIN   DD *
         TITLE '*** PROJECT DEFINITION FOR PROJECT=IBMDEMO ***'
IBMDEMO  FLMABEG
*
*        ************************************************************
*        *  DEFINE THE AUTHORIZATON CODES                           *
*        ************************************************************
GRPDEV   FLMAGRP AC=(DEV)
*
*        ************************************************************
*        *  DEFINE THE TYPES                                        *
*        ************************************************************
COBOL    FLMTYPE ,                    HOST COBOL CODE
DOC      FLMTYPE ,                    WORD FOR WINDOWS DOCUMENTATION
GRAPHICS FLMTYPE ,                    GRAPHICS FILES (JPG, GIF)
HTML     FLMTYPE ,                    HTML AND JAVASCRIPT
JAR      FLMTYPE ,                    JAVA JAR
JAVA     FLMTYPE ,                    JAVA SOURCE
JAVACLAS FLMTYPE ,                    JAVA CLASSES
JAVALIST FLMTYPE ,                    JAVA LISTING
PACKAGES FLMTYPE ,                    PACKAGE ARCHHL
*
*        ************************************************************
*        *  DEFINE THE GROUPS                                       *
*        ************************************************************
DEV      FLMGROUP AC=(DEV),KEY=Y,PROMOTE=QA      DEVELOPMENT
QA       FLMGROUP AC=(DEV),KEY=Y,PROMOTE=REL     QUALITY ASSURANCE
REL      FLMGROUP AC=(DEV),KEY=Y                 PRODUCTION
*
*********************************************************************
*               PROJECT PROJDEFSS
*********************************************************************
        FLMCNTRL ACCT=IBMDEMO.PROJDEFS.ACCT,
               VERS=IBMDEMO.PROJDEFS.VERSION,
               XREF=IBMDEMO.PROJDEFS.XREF,
               LIBID=SCLM
*
*********************************************************************
*                VERSIONING AND AUDITIBILITY                        *
*********************************************************************
        FLMATVER GROUP=DEV,TYPE=COBOL,VERSION=YES
        FLMATVER GROUP=DEV,TYPE=DOC,VERSION=YES
        FLMATVER GROUP=DEV,TYPE=GRAPHICS,VERSION=YES
        FLMATVER GROUP=DEV,TYPE=HTML,VERSION=YES
        FLMATVER GROUP=DEV,TYPE=JAR,VERSION=YES
        FLMATVER GROUP=DEV,TYPE=JAVA,VERSION=YES
        FLMATVER GROUP=DEV,TYPE=JAVACLAS,VERSION=YES
        FLMATVER GROUP=DEV,TYPE=JAVALIST,VERSION=YES
        FLMATVER GROUP=DEV,TYPE=PACKAGES,VERSION=YES
*
        FLMATVER GROUP=QA,TYPE=COBOL,VERSION=YES
        FLMATVER GROUP=QA,TYPE=DOC,VERSION=YES
```

Figure 33. Sample IBMDEMO Project Definition (CLZTPDEF) (Part 2 of 3)

```
        FLMATVER GROUP=QA,TYPE=GRAPHICS,VERSION=YES
        FLMATVER GROUP=QA,TYPE=HTML,VERSION=YES
        FLMATVER GROUP=QA,TYPE=JAR,VERSION=YES
        FLMATVER GROUP=QA,TYPE=JAVA,VERSION=YES
        FLMATVER GROUP=QA,TYPE=JAVACLAS,VERSION=YES
        FLMATVER GROUP=QA,TYPE=JAVALIST,VERSION=YES
        FLMATVER GROUP=QA,TYPE=PACKAGES,VERSION=YES
*
        FLMATVER GROUP=REL,TYPE=COBOL,VERSION=YES
        FLMATVER GROUP=REL,TYPE=DOC,VERSION=YES
        FLMATVER GROUP=REL,TYPE=GRAPHICS,VERSION=YES
        FLMATVER GROUP=REL,TYPE=HTML,VERSION=YES
        FLMATVER GROUP=REL,TYPE=JAR,VERSION=YES
        FLMATVER GROUP=REL,TYPE=JAVA,VERSION=YES
        FLMATVER GROUP=REL,TYPE=JAVACLAS,VERSION=YES
        FLMATVER GROUP=REL,TYPE=JAVALIST,VERSION=YES
        FLMATVER GROUP=REL,TYPE=PACKAGES,VERSION=YES
*
*********************************************************************
*                LANGUAGE DEFINITION TABLES                         *
*********************************************************************
*             TRANSLATOR            LANGUAGE
         COPY  FLM@ARCD           -- ARCHDEF                    --
         COPY  FLM@COB2           -- COBOL                      --
         COPY  CLZTJAR            -- JAR                        --
         COPY  CLZTJAVA           -- JAVA                       --
         COPY  CLZTJBIN           -- BINARY E-BUSINESS OBJECTS  --
         COPY  CLZTJTXT           -- TEXT E-BUSINESS OBJECTS    --
*
*********************************************************************
         FLMAEND
/*
//SYSLIN DD DSN=IBMDEMO.PROJDEFS.OBJLIB(IBMDEMO),DISP=SHR
//LINK   EXEC LINK
//SYSLIN DD *
 INCLUDE SYSLIB(IBMDEMO)
 NAME IBMDEMO(R)
/*
```

Figure 33. Sample IBMDEMO Project Definition (CLZTPDEF) (Part 3 of 3)

Optionally, if you are building a new isolated project for the S-JDK system, then you will need to run two additional jobs: CLZTALIB and CLZTAVSM. These JCL streams will allocate all of the group libraries and the SCLM VSAM files, respectfully.

Step 4a: Allocate New S-JDK Type Data Set

Allocate SCLM Type Files — CLZTALIB

Regardless of whether you build the types into an existing project or as a stand alone project, each type needs a set of libraries. The following JCL stream will allocate these required libraries.

```
//* (JOBCARD)
//*--------------------------------------------------------------------*
//*  NAME:     CLZTALIB                                                *
//*  PURPOSE:  DELETE AND ALLOCATE THE S-JDK BASE AND VERSION FILES.   *
//*--------------------------------------------------------------------*
//*  TO USE THIS JCL YOU MUST:                                         *
//*         1) INSERT A VALID JOB CARD.                                *
//*         2) REVIEW THE DATASET NAMES - THEY ARE DELIVERED           *
//*            MATCHING THE S-JDK DEFAULT VALUES.                      *
//*         3) MODIFY THE DATASET NAMES TO MATCH ACTUAL CHOSEN S-JDK   *
//*            TYPES.                                                  *
//*         4) CHANGE THE UNIT=DUNIT TO THE APPROPRIATE UNIT FOR       *
//*            PERMANENT FILES.                                        *
//*--------------------------------------------------------------------*
//*  DELETE ALL S-JDK TYPE FILES                                       *
//*--------------------------------------------------------------------*
//DELETE  EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN   DD *
 DELETE IBMDEMO.DEV.COBOL
 DELETE IBMDEMO.DEV.DOC
 DELETE IBMDEMO.DEV.GRAPHICS
 DELETE IBMDEMO.DEV.HTML
 DELETE IBMDEMO.DEV.JAR
 DELETE IBMDEMO.DEV.JAVA
 DELETE IBMDEMO.DEV.JAVACLAS
 DELETE IBMDEMO.DEV.JAVALIST
 DELETE IBMDEMO.DEV.JCL
 DELETE IBMDEMO.DEV.PACKAGES
 DELETE IBMDEMO.QA.COBOL
 DELETE IBMDEMO.QA.DOC
 DELETE IBMDEMO.QA.GRAPHICS
 DELETE IBMDEMO.QA.HTML
 DELETE IBMDEMO.QA.JAR
 DELETE IBMDEMO.QA.JAVA
 DELETE IBMDEMO.QA.JAVACLAS
 DELETE IBMDEMO.QA.JAVALIST
 DELETE IBMDEMO.QA.JCL
 DELETE IBMDEMO.QA.PACKAGES
 DELETE IBMDEMO.REL.COBOL
 DELETE IBMDEMO.REL.DOC
 DELETE IBMDEMO.REL.GRAPHICS
 DELETE IBMDEMO.REL.HTML
 DELETE IBMDEMO.REL.JAR
 DELETE IBMDEMO.REL.JAVA
 DELETE IBMDEMO.REL.JAVACLAS
 DELETE IBMDEMO.REL.JAVALIST
 DELETE IBMDEMO.REL.JCL
 DELETE IBMDEMO.REL.PACKAGES
 DELETE IBMDEMO.DEV.COBOL.VERSION
 DELETE IBMDEMO.DEV.DOC.VERSION
 DELETE IBMDEMO.DEV.GRAPHICS.VERSION
 DELETE IBMDEMO.DEV.HTML.VERSION
 DELETE IBMDEMO.DEV.JAR.VERSION
 DELETE IBMDEMO.DEV.JAVA.VERSION
 DELETE IBMDEMO.DEV.JAVACLAS.VERSION
 DELETE IBMDEMO.DEV.JAVALIST.VERSION
 DELETE IBMDEMO.DEV.JCL.VERSION
 DELETE IBMDEMO.DEV.PACKAGES.VERSION
 DELETE IBMDEMO.QA.COBOL.VERSION
 DELETE IBMDEMO.QA.DOC.VERSION
```

Figure 34. CLZTALIB SCLM Type Data Set Allocations (Part 1 of 3)

```
 DELETE IBMDEMO.QA.GRAPHICS.VERSION
 DELETE IBMDEMO.QA.HTML.VERSION
 DELETE IBMDEMO.QA.JAR.VERSION
 DELETE IBMDEMO.QA.JAVA.VERSION
 DELETE IBMDEMO.QA.JAVACLAS.VERSION
 DELETE IBMDEMO.QA.JAVALIST.VERSION
 DELETE IBMDEMO.QA.JCL.VERSION
 DELETE IBMDEMO.QA.PACKAGES.VERSION
 DELETE IBMDEMO.REL.COBOL.VERSION
 DELETE IBMDEMO.REL.DOC.VERSION
 DELETE IBMDEMO.REL.GRAPHICS.VERSION
 DELETE IBMDEMO.REL.HTML.VERSION
 DELETE IBMDEMO.REL.JAR.VERSION
 DELETE IBMDEMO.REL.JAVA.VERSION
 DELETE IBMDEMO.REL.JAVACLAS.VERSION
 DELETE IBMDEMO.REL.JAVALIST.VERSION
 DELETE IBMDEMO.REL.JCL.VERSION
 DELETE IBMDEMO.REL.PACKAGES.VERSION
//*-------------------------------------------------------------------*
//*  PROC TO ALLOCATE BASE FILES                                      *
//*-------------------------------------------------------------------*
//ALLOC     PROC FILE=
//STEP1   EXEC PGM=IEFBR14
//DD01 DD  DSN=&FILE,
//     DISP=(NEW,CATLG,DELETE),
//     UNIT=DUNIT,
//     SPACE=(CYL,(2,10,100)),
//     DCB=(RECFM=VB,LRECL=256,BLKSIZE=0)
//   PEND
//*-------------------------------------------------------------------*
//*  PROC TO ALLOCATE VERSION FILES                                   *
//*-------------------------------------------------------------------*
//VALLOC    PROC FILE=
//STEP2   EXEC PGM=IEFBR14
//DD01 DD  DSN=&FILE,
//     DISP=(NEW,CATLG,DELETE),
//     UNIT=DUNIT,
//     SPACE=(CYL,(2,10,100)),
//     DCB=(RECFM=VB,LRECL=350,BLKSIZE=0)
//   PEND
//*-------------------------------------------------------------------*
//*  NOW DO THE ALLOCATES                                             *
//*-------------------------------------------------------------------*
//FILE01 EXEC ALLOC,FILE=IBMDEMO.DEV.COBOL
//FILE02 EXEC ALLOC,FILE=IBMDEMO.DEV.DOC
//FILE03 EXEC ALLOC,FILE=IBMDEMO.DEV.GRAPHICS
//FILE04 EXEC ALLOC,FILE=IBMDEMO.DEV.HTML
//FILE05 EXEC ALLOC,FILE=IBMDEMO.DEV.JAR
//FILE06 EXEC ALLOC,FILE=IBMDEMO.DEV.JAVA
//FILE07 EXEC ALLOC,FILE=IBMDEMO.DEV.JAVACLAS
//FILE08 EXEC ALLOC,FILE=IBMDEMO.DEV.JAVALIST
//FILE09 EXEC ALLOC,FILE=IBMDEMO.DEV.JCL
//FILE10 EXEC ALLOC,FILE=IBMDEMO.DEV.PACKAGES
//FILE11 EXEC ALLOC,FILE=IBMDEMO.QA.COBOL
//FILE12 EXEC ALLOC,FILE=IBMDEMO.QA.DOC
//FILE13 EXEC ALLOC,FILE=IBMDEMO.QA.GRAPHICS
//FILE14 EXEC ALLOC,FILE=IBMDEMO.QA.HTML
//FILE15 EXEC ALLOC,FILE=IBMDEMO.QA.JAR
//FILE16 EXEC ALLOC,FILE=IBMDEMO.QA.JAVA
//FILE17 EXEC ALLOC,FILE=IBMDEMO.QA.JAVACLAS
//FILE18 EXEC ALLOC,FILE=IBMDEMO.QA.JAVALIST
```

Figure 34. CLZTALIB SCLM Type Data Set Allocations (Part 2 of 3)

```
//FILE19 EXEC ALLOC,FILE=IBMDEMO.QA.JCL
//FILE20 EXEC ALLOC,FILE=IBMDEMO.QA.PACKAGES
//FILE21 EXEC ALLOC,FILE=IBMDEMO.REL.COBOL
//FILE22 EXEC ALLOC,FILE=IBMDEMO.REL.DOC
//FILE23 EXEC ALLOC,FILE=IBMDEMO.REL.GRAPHICS
//FILE24 EXEC ALLOC,FILE=IBMDEMO.REL.HTML
//FILE25 EXEC ALLOC,FILE=IBMDEMO.REL.JAR
//FILE26 EXEC ALLOC,FILE=IBMDEMO.REL.JAVA
//FILE27 EXEC ALLOC,FILE=IBMDEMO.REL.JAVACLAS
//FILE28 EXEC ALLOC,FILE=IBMDEMO.REL.JAVALIST
//FILE29 EXEC ALLOC,FILE=IBMDEMO.REL.JCL
//FILE30 EXEC ALLOC,FILE=IBMDEMO.REL.PACKAGES
//*
//FILE31 EXEC VALLOC,FILE=IBMDEMO.DEV.COBOL.VERSION
//FILE32 EXEC VALLOC,FILE=IBMDEMO.DEV.DOC.VERSION
//FILE33 EXEC VALLOC,FILE=IBMDEMO.DEV.GRAPHICS.VERSION
//FILE34 EXEC VALLOC,FILE=IBMDEMO.DEV.HTML.VERSION
//FILE35 EXEC VALLOC,FILE=IBMDEMO.DEV.JAR.VERSION
//FILE36 EXEC VALLOC,FILE=IBMDEMO.DEV.JAVA.VERSION
//FILE37 EXEC VALLOC,FILE=IBMDEMO.DEV.JAVACLAS.VERSION
//FILE38 EXEC VALLOC,FILE=IBMDEMO.DEV.JAVALIST.VERSION
//FILE39 EXEC VALLOC,FILE=IBMDEMO.DEV.JCL.VERSION
//FILE40 EXEC VALLOC,FILE=IBMDEMO.DEV.PACKAGES.VERSION
//FILE41 EXEC VALLOC,FILE=IBMDEMO.QA.COBOL.VERSION
//FILE42 EXEC VALLOC,FILE=IBMDEMO.QA.DOC.VERSION
//FILE43 EXEC VALLOC,FILE=IBMDEMO.QA.GRAPHICS.VERSION
//FILE44 EXEC VALLOC,FILE=IBMDEMO.QA.HTML.VERSION
//FILE45 EXEC VALLOC,FILE=IBMDEMO.QA.JAR.VERSION
//FILE46 EXEC VALLOC,FILE=IBMDEMO.QA.JAVA.VERSION
//FILE47 EXEC VALLOC,FILE=IBMDEMO.QA.JAVACLAS.VERSION
//FILE48 EXEC VALLOC,FILE=IBMDEMO.QA.JAVALIST.VERSION
//FILE49 EXEC VALLOC,FILE=IBMDEMO.QA.JCL.VERSION
//FILE50 EXEC VALLOC,FILE=IBMDEMO.QA.PACKAGES.VERSION
//FILE51 EXEC VALLOC,FILE=IBMDEMO.REL.COBOL.VERSION
//FILE52 EXEC VALLOC,FILE=IBMDEMO.REL.DOC.VERSION
//FILE53 EXEC VALLOC,FILE=IBMDEMO.REL.GRAPHICS.VERSION
//FILE54 EXEC VALLOC,FILE=IBMDEMO.REL.HTML.VERSION
//FILE55 EXEC VALLOC,FILE=IBMDEMO.REL.JAR.VERSION
//FILE56 EXEC VALLOC,FILE=IBMDEMO.REL.JAVA.VERSION
//FILE57 EXEC VALLOC,FILE=IBMDEMO.REL.JAVACLAS.VERSION
//FILE58 EXEC VALLOC,FILE=IBMDEMO.REL.JAVALIST.VERSION
//FILE59 EXEC VALLOC,FILE=IBMDEMO.REL.JCL.VERSION
//FILE60 EXEC VALLOC,FILE=IBMDEMO.REL.PACKAGES.VERSION
```

Figure 34. CLZTALIB SCLM Type Data Set Allocations (Part 3 of 3)

Step 4b (Optional): Allocate New Project VSAM Files

Allocate Project VSAM Files — CLZTAVSM

If you are building an isolated, stand alone Project for the S-JDK types, there are three VSAM files that need to be allocated. The following JCL stream, found in the CLZ.SCLZJCL library, will allocate these required libraries.

```
//* (JOBCARD)
//*--------------------------------------------------------------------*
//*  NAME:     CLZTAVSM                                                *
//*  PURPOSE:  DELETE AND ALLOCATE THE S-JDK PROJECT VSAM FILES.       *
//*--------------------------------------------------------------------*
//*  TO USE THIS JCL YOU MUST:                                         *
//*         1) INSERT A VALID JOB CARD.                                *
//*         2) REVIEW THE DATASET NAMES - THEY ARE DELIVERED           *
//*            MATCHING THE S-JDK DEFAULT VALUES.                      *
//*         3) MODIFY THE DATASET NAMES TO MATCH ACTUAL CHOSEN S-JDK   *
//*            TYPES.                                                  *
//*         4) CHANGE THE UNIT=TDISK TO THE APPROPRIATE UNIT FOR       *
//*            TEMPORARY FILES.                                        *
//*         5) CHANGE THE "DVOLSER" VARIABLE TO THE APPROPRIATE        *
//*            VOLUME FOR VSAM FILES.                                  *
//*********************************************************************
//*  IBMDEMO.PROJDEFS.JCLLIB(ALLOVSAM)
//*********************************************************************
//ACCT1   EXEC PGM=IDCAMS
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
     DELETE IBMDEMO.PROJDEFS.ACCT
     DEFINE CLUSTER +
        (NAME('IBMDEMO.PROJDEFS.ACCT') +
         CYLINDERS(1 1) +
         VOLUMES(DVOLSER) +
         KEYS(26 0) +
         RECORDSIZE(264 32000) +
         SHAREOPTIONS(4,3) +
         SPEED +
         SPANNED +
         UNIQUE) +
         INDEX(NAME('IBMDEMO.PROJDEFS.ACCT.I') -
         ) +
         DATA(NAME('IBMDEMO.PROJDEFS.ACCT.D') -
         CISZ(2048) +
         FREESPACE(50 50) +
         )
/*
//*********************************************************************
//*
//*  INITIALIZE THE ACCOUNTING FILE
//*
//*********************************************************************
//ACCT2    EXEC PGM=IDCAMS
//INPUT  DD *
                        SCLM ACCOUNTING FILE INITIALIZATION RECORD
/*
//OUTPUT DD DSN=IBMDEMO.PROJDEFS.ACCT,DISP=SHR
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
     REPRO INFILE(INPUT) OUTFILE(OUTPUT)
/*
```

Figure 35. CLZTAVSM — Allocate Project VSAM Files (Part 1 of 3)

```
//*********************************************************************
//*  A JOB STEP IS THEN EXECUTED TO INITIALIZE THE FILE.
//*********************************************************************
//VERS1   EXEC PGM=IDCAMS
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
     DELETE IBMDEMO.PROJDEFS.VERSION
     DEFINE CLUSTER +
        (NAME('IBMDEMO.PROJDEFS.VERSION') +
         CYLINDERS(1 1) +
         VOLUMES(DVOLSER) +
         KEYS(40 0) +
         RECORDSIZE(264 32000) +
         SHAREOPTIONS(4,3) +
         SPEED +
         SPANNED +
         UNIQUE) +
         INDEX(NAME('IBMDEMO.PROJDEFS.VERSION.I') -
         ) +
         DATA(NAME('IBMDEMO.PROJDEFS.VERSION.D') -
         CISZ(2048) +
         FREESPACE(50 50) +
         )
/*
/*
//*********************************************************************
//*  INITIALIZE THE AUDIT CONTROL FILE
//*********************************************************************
//VERS2    EXEC PGM=IDCAMS
//INPUT  DD *
                       SCLM AUDIT CONTROL FILE INITIALIZATION RECORD
/*
//OUTPUT DD DSN=IBMDEMO.PROJDEFS.VERSION,DISP=SHR
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
     REPRO INFILE(INPUT) OUTFILE(OUTPUT)
/*
//*********************************************************************
//*  A JOB STEP IS THEN EXECUTED TO INITIALIZE THE FILE.
//*
//*********************************************************************
//XREF0   EXEC PGM=BZZFPARM,
//  PARM='NEW,SCLM CROSS REFERENCE INITIALIZATION RECORD'
//STEPLIB DD DSN=IBMDEMO.PRODUCT.LOADLIB,DISP=SHR
//CIGPOUT DD DSN=&&PARMFILE,DISP=(NEW,PASS),
//   UNIT=TDISK,DCB=(LRECL=128,BLKSIZE=12800,RECFM=FB),
//   SPACE=(TRK,1)
//CIGLOG  DD SYSOUT=*
//* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
//XREF1   EXEC PGM=IDCAMS
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
     DELETE IBMDEMO.PROJDEFS.XREF
     DEFINE CLUSTER +
        (NAME('IBMDEMO.PROJDEFS.XREF') +
         CYLINDERS(2 1) +
         VOLUMES(DVOLSER) +
         KEYS(128 0) +
         RECORDSIZE(264 32000) +
         SHAREOPTIONS(4,3) +
```

Figure 35. CLZTAVSM — Allocate Project VSAM Files (Part 2 of 3)

```
 SPEED +
         SPANNED +
         UNIQUE) +
         INDEX(NAME('IBMDEMO.PROJDEFS.XREF.I') -
         ) +
         DATA(NAME('IBMDEMO.PROJDEFS.XREF.D') -
         CISZ(2048) +
         FREESPACE(50 50) +
         )
/*
//*********************************************************************
//*  INITIALIZE THE CROSS-REFERENCE FILE
//*********************************************************************
//XREF2    EXEC PGM=IDCAMS
//INPUT  DD DSN=&&PARMFILE,DISP=(OLD,DELETE)
//OUTPUT DD DSN=IBMDEMO.PROJDEFS.XREF,DISP=SHR
//SYSPRINT  DD SYSOUT=*
//SYSIN     DD *
     REPRO INFILE(INPUT) OUTFILE(OUTPUT)
/*
//*
```

Figure 35. CLZTAVSM — Allocate Project VSAM Files (Part 3 of 3)

Step 5: Define S-JDK Types to Cloud 9 SLR

The purpose of this step is to define the S-JDK types to your Cloud 9 SLR file. The example below shows the default S-JDK definitions required to define the S-JDK types to the Cloud 9 SLR.

Modify and Submit CLZC9J06

1. Using ISPF EDIT, access member CLZC9J06 (CLZ.SCLZJCL library). This JCL should already have been modified during the base install and IVP process.
2. Update the member including the following SLR definitions (in BOLD)
3. Submit the job.

Note: Note that this job should terminate with COND CODE=0.

```
//* (JOBCARD)
//* ------------------------------------------------------------------*
//* CLOUD 9 JAVA/S-JDK COMPONENTS.                                     *
//* ------------------------------------------------------------------*
//* ------------------------------------------------------------------*
//*  NAME:    CLZC9J06                                                 *
//*  PURPOSE: DEFINE S-JDK TYPES TO THE SLR DATABASE.                  *
//* ------------------------------------------------------------------*
//*  TO USE THIS JCL, YOU MUST:                                        *
//*         1) INSERT A VALID JOB CARD WITH VALID CLASS AND REGION=0M *
//*         2) MAKE SURE THAT THE STEPLIB POINTS TO YOUR CLOUD9       *
//*            AUTHORIZED DATASET THAT CONTAINS THE CIGINI.           *
//*         3) MODIFY THE TYPE NAMES IF YOU HAVE CHANGED THE DEFAULT  *
//*            SCLM TYPES.                                            *
//* ------------------------------------------------------------------ *
//STEP1    EXEC PGM=CZLSLR
//STEPLIB  DD DSN=CLZ.SCLZLOAD,DISP=SHR
//CIGIN    DD *
ADD NAME RULE FOR SCLM TYPE DOC      CASE SENSITIVE.
ADD NAME RULE FOR SCLM TYPE GRAPHICS CASE SENSITIVE.
ADD NAME RULE FOR SCLM TYPE HTML     CASE SENSITIVE.
ADD NAME RULE FOR SCLM TYPE JAVA     CASE SENSITIVE.
ADD NAME RULE FOR SCLM TYPE JAR      CASE SENSITIVE.
ADD NAME RULE FOR SCLM TYPE JAVACLAS CASE SENSITIVE.
//CIGLOG   DD SYSOUT=*
```

Figure 36. CLZC9J06 — Define S-JDK Types to Cloud 9 SLR

Step 6: Run CLZTAUNX to Build S-JDK USS Directories

The purpose of this step is to define the S-JDK Unix Directories. The member is shown with default values.

Modify and Submit CLZTAUNX

1. Using ISPF EDIT, access member CLZTAUNX (CLZ.SCLZJCL library).
2. Issue the CAPS OFF command to ensure case sensitivity.
3. Copy your job card values to the top of the member. Substitute your site-specific values (identified on the Installation Worksheet) for those values shown below in bold.
4. Submit the job.

Note: Note that this job should terminate with COND CODE=0.

```
//* (JOBCARD)
//*--------------------------------------------------------------------*
//*  NAME:     CLZTAUNX                                                *
//*  PURPOSE:  UNIX ALLOCATION OF S-JDK DIRECTORIES AND PERMISSIONS.   *
//*--------------------------------------------------------------------*
//*  TO USE THIS JCL YOU MUST:                                         *
//*         1) INSERT A VALID JOB CARD.                                *
//*         2) REVIEW THE DIRECTORY NAMES - THEY ARE DELIVERED         *
//*            MATCHING THE S-JDK DEFAULT VALUES.                      *
//*         3) MODIFY THE DIRECTORY NAME AS PER THE SCLM/USS           *
//*            WORKSHEET.                                              *
//*--------------------------------------------------------------------*
//*  UNIX CLEANUP                                                      *
//*--------------------------------------------------------------------*
//UNIX    EXEC PGM=IKJEFT01
//SYSPROC  DD DSN=SYS1.SBPXEXEC,DISP=SHR
//SYSTSIN DD  *

   /* Remove all files in ibmdemo */
   oshell rm -r /u/ibmdemo/dev
   oshell rm -r /u/ibmdemo/qa
   oshell rm -r /u/ibmdemo/rel

   /* Remove all directories in ibmdemo-dev */
   oshell rmdir -p /u/ibmdemo/dev/classes
   oshell rmdir -p /u/ibmdemo/dev/graphics
   oshell rmdir -p /u/ibmdemo/dev/html
   oshell rmdir -p /u/ibmdemo/dev/jar
   oshell rmdir -p /u/ibmdemo/dev/listings

   /* Remove all directories in ibmdemo-qa  */
   oshell rmdir -p /u/ibmdemo/qa/classes
   oshell rmdir -p /u/ibmdemo/qa/graphics
   oshell rmdir -p /u/ibmdemo/qa/html
   oshell rmdir -p /u/ibmdemo/qa/jar
   oshell rmdir -p /u/ibmdemo/qa/listings
```

Figure 37. CLZTAUNX — Create USS S-JDK Directories (Part 1 of 2)

```
/* Remove all directories in ibmdemo-rel */
  oshell rmdir -p /u/ibmdemo/rel/classes
  oshell rmdir -p /u/ibmdemo/rel/graphics
  oshell rmdir -p /u/ibmdemo/rel/html
  oshell rmdir -p /u/ibmdemo/rel/jar
  oshell rmdir -p /u/ibmdemo/rel/listings

  /* Make all directories in ibmdemo-dev */
  oshell mkdir -p /u/ibmdemo/dev/classes
  oshell mkdir -p /u/ibmdemo/dev/graphics
  oshell mkdir -p /u/ibmdemo/dev/html
  oshell mkdir -p /u/ibmdemo/dev/jar
  oshell mkdir -p /u/ibmdemo/dev/listings

  /* Make all directories in ibmdemo-qa  */
  oshell mkdir -p /u/ibmdemo/qa/classes
  oshell mkdir -p /u/ibmdemo/qa/graphics
  oshell mkdir -p /u/ibmdemo/qa/html
  oshell mkdir -p /u/ibmdemo/qa/jar
  oshell mkdir -p /u/ibmdemo/qa/listings

  /* Make all directories in ibmdemo-rel */
  oshell mkdir -p /u/ibmdemo/rel/classes
  oshell mkdir -p /u/ibmdemo/rel/graphics
  oshell mkdir -p /u/ibmdemo/rel/html
  oshell mkdir -p /u/ibmdemo/rel/jar
  oshell mkdir -p /u/ibmdemo/rel/listings

  /* Set permissions for ibmdemo projects */
  oshell chmod -R 777 /u/ibmdemo/dev
  oshell chmod -R 777 /u/ibmdemo/qa
  oshell chmod -R 777 /u/ibmdemo/rel

  /* copy sample files to target location */
 oput 'CLZ.SCLZHTML(CLZHCLCK)' '/u/ibmdemo/dev/clock.html'
 oput 'CLZ.SCLZHTML(CLZJCLCK)' '/u/ibmdemo/dev/Clock2.java'

 oshell chmod 777 '/u/ibmdemo/dev/clock.html'
 oshell chmod 777 '/u/ibmdemo/dev/Clock2.java'

//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//OSHELL   DD SYSOUT=*
```

Figure 37. CLZTAUNX — Create USS S-JDK Directories (Part 2 of 2)

Step 7: Review CLZJIBM Unix Shell — Delete Processing

Understanding SCLM Delete Processing

To perform delete processing, the user must run the delete utility provided for deletion of JAVA/USS compiled and copied objects. This utility is resident in the CLZJIBM JCL shell that comes with the Cloud 9 product.

The following figure contains the CLZJIBM JCL shell as delivered with the product. Note, at the end of the member, there is a Delete Action step that invokes one of the S-JDK REXX utilities. Please review this step for consistency with other customizations that have been performed against the translators and control files.

```
)DOT
%JOBCARD%
)ENDDOT
//* ------------------------------------------------------------------ *
//* NAME:     CLZJIBM                                                  *
//* PURPOSE:  CLOUD 9 FOR SCLM.                                        *
//*           SCLM BATCH SKELETON.                                     *
//* ------------------------------------------------------------------ *
//*                                                                    *
//* REQUIRED JCL MODIFICATION:                                         *
//*  1) CHANGE THE FOLLOWING AS PER THE INSTALLATION WORKSHEET.        *
//*      - ISPFQUAL                                                    *
//*      - TDISK                                                       *
//*  2) TARGET LIBRARIES CLZ.SCLZLOAD AND CLZ.SCLZCGI        /* C1 */ *
//*     SHOULD BE CHANGED IF THE INSTALLED HIGH LEVEL        /* C1 */ *
//*     QUALIFIER IS NOT 'CLZ.'                              /* C1 */ *
//*                                                                    *
//*  NOTE: BREEZE USERS MUST MODIFY THIS MODULE PER EMBEDDED           *
//*        INSTRUCTION. BREEZE DATASET NAMES MAY ALSO NEED TO BE       *
//*        MODIFIED.                                                   *
//*                                                                    *
//* 22OCT2001 OW51810 - CHANGES MARKED AS /* C1 */                     *
//* Z020402A                                                           *
//*                                                                    *
//*--------------------------------------------------------------------*
//*  RESIDES IN HTTP SERVER AT:                           /* C1 */ *
//*  /ROOTDIR/CLOUD9/JCL/CLZJIBM                                       *
//*--------------------------------------------------------------------*
)IF ACTION=OCOPY
//COPY   EXEC PGM=IKJEFT01
)DOT
%COPYFILES%
)ENDDOT
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD *
)DOT
%OCOPYSYNTAX%
)ENDDOT
/*
)ENDIF
)IF ACTION=IEBCOPY
//COPY     EXEC PGM=IEBCOPY
)DOT
%COPYFILES%
)ENDDOT
//SYSIN   DD *
)DOT
%OCOPYSYNTAX%
)ENDDOT
//SYSPRINT  DD  SYSOUT=*
)ENDIF
```

Figure 38. CLZJIBM Unix JCL Shell (Part 1 of 4)

```
//*-------------------------------------------------------------------
//GENER  EXEC PGM=IEBGENER
//SYSUT1   DD *
)DOT
%SCLMSYNTAX%
)ENDDOT
//SYSUT2   DD DSN=&&CLIST(TEMPNAME),UNIT=TDISK,
//            SPACE=(TRK,(10,10,2),RLSE),
//            DISP=(NEW,PASS),DCB=(LRECL=80,
//            BLKSIZE=1600,DSORG=PO,RECFM=FB)
//SYSPRINT DD DUMMY
//SYSIN    DD DUMMY
)IF ACTION=DELETE
//DGRPTS   DD DSN=&&DELLIST,DISP=(NEW,PASS),                DELETE
//         SPACE=(TRK,(5,10),RLSE),
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
)ENDIF
)IF ACTION=BUILD
//COPYBULD EXEC PGM=CLZTFILE                                JAVA
//STEPLIB  DD DSN=CLZ.SCLZLOAD,DISP=SHR                        /* C1 */
//SYSIN    DD *                                             JAVA
)DOT
%SCLMSYNTAX%
)ENDDOT
//SYSOUT   DD DSN=&&BSYNTAX,DISP=(NEW,PASS),                JAVA
//            SPACE=(TRK,(10,10),RLSE),UNIT=TDISK,          JAVA
//            DCB=(LRECL=80,BLKSIZE=0,DSORG=PS,RECFM=FB)    JAVA
)ENDIF
//*****************************************************************
//* BREEZE USERS:  UNCOMMENT LINES WITH "BREEZE USERS"
//*****************************************************************
//TSO    EXEC PGM=IKJEFT01,REGION=4096K,TIME=1439,DYNAMNBR=200
//STEPLIB  DD DSN=CLZ.SCLZLOAD,DISP=SHR
//*        DD DSN=BZZ.SBZZLOAD,DISP=SHR            BREEZE USERS
//SYSTSIN  DD *
  ISPSTART CMD(%TEMPNAME)
//SYSTSPRT DD SYSOUT=(*)
//SYSPROC  DD DSN=&&CLIST,DISP=(OLD,DELETE)
//*        DD DSN=BZZ.SBZZCLIB,DISP=SHR            BREEZE USERS
//*****************************************************************
```

Figure 38. CLZJIBM Unix JCL Shell (Part 2 of 4)

```
//* ISPF LIBRARIES
//ISPMLIB  DD DSN=ISPFQUAL.SISPMENU,DISP=SHR
//*        DD DSN=BZZ.SBZZMENU,DISP=SHR            BREEZE USERS
//ISPSLIB  DD DSN=ISPFQUAL.SISPSENU,DISP=SHR
//         DD DSN=ISPFQUAL.SISPSLIB,DISP=SHR
//*        DD DSN=BZZ.SBZZSENU,DISP=SHR            BREEZE USERS
//ISPPLIB  DD DSN=ISPFQUAL.SISPPENU,DISP=SHR
//*        DD DSN=BZZ.SBZZPENU,DISP=SHR            BREEZE USERS
//ISPTLIB  DD UNIT=VIO,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//            DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB)
//         DD DSN=ISPFQUAL.SISPTENU,DISP=SHR
//ISPTABL  DD UNIT=VIO,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//            DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB)
//ISPPROF  DD UNIT=VIO,DISP=(NEW,PASS),SPACE=(CYL,(1,1,5)),
//            DCB=(LRECL=80,BLKSIZE=19040,DSORG=PO,RECFM=FB)
//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
//ZFLMDD   DD *
    ZFLMNLST=FLMNLENU  ZFLMTRMT=ISR3278   ZDATEF=YY/MM/DD
/*
//*CIGLOG  DD SYSOUT=*                             BREEZE USERS
//*CIGLOG0 DD SYSOUT=*                             BREEZE USERS
//*CIGLOG1 DD DSN=&&CIGLOG1,DISP=(NEW,DELETE),     BREEZE USERS
//*           UNIT=TDISK,SPACE=(CYL,(1,1)),        BREEZE USERS/* C1 */
//*           DCB=(LRECL=132,BLKSIZE=0,RECFM=FB)   BREEZE USERS
//*CIGLOG2 DD DSN=&&CIGLOG2,DISP=(NEW,DELETE),     BREEZE USERS
//*           UNIT=TDISK,SPACE=(CYL,(1,1)),        BREEZE USERS/* C1 */
//*           DCB=(LRECL=132,BLKSIZE=0,RECFM=FB)   BREEZE USERS
//*CIGLOG3 DD DSN=&&CIGLOG3,DISP=(NEW,DELETE),     BREEZE USERS
//*           UNIT=TDISK,SPACE=(CYL,(1,1)),        BREEZE USERS/* C1 */
//*           DCB=(LRECL=132,BLKSIZE=0,RECFM=FB)   BREEZE USERS
//*****************************************************************
```

Figure 38. CLZJIBM Unix JCL Shell (Part 3 of 4)

```
//* SCLM OUTPUT FILES
//*****************************************************************
//FLMMSGS  DD SYSOUT=(*)
)IF ACTION=BUILD
//BLDMSGS  DD SYSOUT=*,                                        BUILD
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
//BLDREPT  DD SYSOUT=*,                                        BUILD
//         DCB=(LRECL=80,BLKSIZE=3120,RECFM=FBA)
//BLDLIST  DD SYSOUT=*,                                        BUILD
//         DCB=(LRECL=259,BLKSIZE=3120,RECFM=VB)
//BLDEXIT  DD DSN=&&BLDEXIT,DISP=(NEW,DELETE),                 BUILD
//         SPACE=(TRK,(5,10),RLSE),
//         DCB=(LRECL=160,BLKSIZE=3200,RECFM=FB)
//BSYNTAX  DD DSN=&&BSYNTAX,DISP=(OLD,PASS)                    JAVA
)ENDIF
)IF ACTION=PROMOTE
//PROMMSGS DD SYSOUT=*,                                        PROMOTE
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=FB,DSORG=PS)
//PROMREPT DD SYSOUT=*,                                        PROMOTE
//         DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB,DSORG=PS)
//PROMEXIT DD DSN=&&PROMEXIT,DISP=(NEW,DELETE),                PROMOTE
//         SPACE=(TRK,(5,10),RLSE),
//         DCB=(LRECL=160,BLKSIZE=3200,RECFM=FB)
//COPYERR  DD SYSOUT=*,                                        PROMOTE
//         DCB=(RECFM=FBA,LRECL=133,BLKSIZE=1330)
)ENDIF
)IF ACTION=MIGRATE
//U2LSTS   DD SYSOUT=*,                                        MIGRATE
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
//U2MSGS   DD SYSOUT=*,                                        MIGRATE
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
)ENDIF
)IF ACTION=DELETE
//DGLIST   DD SYSOUT=*,                                        DELETE
//         DCB=(LRECL=137,BLKSIZE=3120,RECFM=VBA)
//DGMSGS   DD SYSOUT=*,                                        DELETE
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
//DGREPT   DD DSN=&&DELLIST,DISP=(MOD,PASS)
//DGEXIT   DD DSN=&&DELEXIT,DISP=(NEW,DELETE),                 DELETE
//         SPACE=(TRK,(5,10),RLSE),
//         DCB=(LRECL=160,BLKSIZE=3200,RECFM=FB)
)ENDIF
)IF ACTION=VERRECOV
//DBUMSGS  DD SYSOUT=*,                                        VERRECOV
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
)ENDIF
//*-------------------------------------------------------------------
)IF ACTION=DELETE
//DELMSGS EXEC PGM=IEBGENER
//SYSUT1   DD DSN=&&DELLIST,DISP=(OLD,PASS)
//SYSUT2   DD SYSOUT=*
//SYSPRINT DD DUMMY
//SYSIN    DD DUMMY
//*
//DELETE  EXEC PGM=IKJEFT01,REGION=4096K,DYNAMNBR=200
//STEPLIB  DD  DSN=CLZ.SCLZLOAD,DISP=SHR
//SYSTSIN  DD  *
  EX 'CLZ.SCLZCGI(CLZTRJDL)'
//SYSTSPRT DD  SYSOUT=*
//DELLIST  DD DSN=&&DELLIST,DISP=(OLD,PASS)
//LISTOUT  DD SYSOUT=*,
//         DCB=(LRECL=80,BLKSIZE=80,RECFM=F)
//UNIXLOC  DD DSN=CLZ.SCLZCGI(CLZTULOC),DISP=SHR
)ENDIF
```

Figure 38. CLZJIBM Unix JCL Shell (Part 4 of 4)

Java tracing: The tracing facility for the Java process aids with debugging problems and can be helpful to see if the tailoring of the Java control members has been carried out properly. The tracing can be turned on in the following modules, found in the SCLZCGI target library: CLZTRJVC, CLZTRJVP, CLZTRJV1 and CLZTRJDL. Follow the instructions in each member to determine the correct tracing operand. The default is to have tracing turned off.

CHECKPOINT #3 for S-JDK Installation

At this point all SCLM and Cloud 9 definitions should be complete.

Table 21. Checkpoint #3 for S-JDK Installation

Task	Completed?
Update your project definition with JDK Types and Translators (Optionally CLZTPDEF)	
Allocate New SCLM Type Libraries CLZTALIB	
Optionally allocate new SCLM project VSAM, only if creating a new project using CLZTAVSM	
Run CLZC9J06 to define S-JDK types to Cloud 9	
Run CLZTAUNX to define USS directories for S-JDK application	
Modified CLZJIBM Cloud 9 JCL shell	

Chapter 12. Perform Installation Verification Procedures

Step 8: Test the S-JDK Translators

There are two files delivered with the S-JDK for translator verification. One is a JAVA file called CLZJCLCK and one is an HTML invocation file call CLZHCLCK. There are all delivered in the CLZ.SCLZHTML libraries. The CLZTAUNX JCL stream already copied and renamed the files to the JAVA application area as shown below

Table 22. Translator Verification Files

Member	Already saved into USS as the following:
CLZJCLCK	/ibmdemo/dev/Clock2.java
CLZHCLCK	/ibmdemo/dev/clock.html

The following steps will allow you to verify the S-JDK translators.

1. Invoke Cloud 9
2. List Unix Files for the /u/ibmdemo/dev directories
3. Select Clock2.java from list
4. Migrate Clock2.java into Cloud 9
   a. Add as a type JAVA and language JAVA
   b. You should see a short name generated
5. List SCLM files for type JAVA
6. Build the Clock2.java to invoke the translators
7. Review the expansion of the translator
8. Review the population of the USS output life cycle and Java USS Output locations. The following is what you should see in the CLASSES directory:

```
/u/ibmdemo/dev/classes
   Type  Filename
 _ Dir   .
 _ Dir   ..
 _ File  Clock2.class
```

Figure 39. Directory Entry After Java Compile

Java Listing Example

The following is what you should see in the Clock2.java listing file:

```
BROWSE -- /u/test/a/sysj/subj/listings/Clock2.java - Line 00000000 Col 001 080
 Command ===>                                                  Scroll ===> CSR
******************************** Top of Data ********************************
.parsed Clock2.java in 18438 ms.
.loaded /usr/lpp/java/J1.1/bin/../lib/classes.zip(java/applet/Applet.class) in 1
.loaded /usr/lpp/java/J1.1/bin/../lib/classes.zip(java/awt/Panel.class) in 210 m
.loaded /usr/lpp/java/J1.1/bin/../lib/classes.zip(java/awt/Container.class) in 1
.loaded /usr/lpp/java/J1.1/bin/../lib/classes.zip(java/awt/Component.class) in 2
.loaded /usr/lpp/java/J1.1/bin/../lib/classes.zip(java/awt/Component$NativeInLig
.loaded /usr/lpp/java/J1.1/bin/../lib/classes.zip(java/lang/Object.class) in 202
.loaded /usr/lpp/java/J1.1/bin/../lib/classes.zip(java/lang/Runnable.class) in 4
.checking class Clock2.
...more
```

Figure 40. Listing example from USS JAVA Compile

Java SCLM Build Map Example

The following is an example of the component list for Clock2.java.

```
Command ===>                                                  Scroll ===> PAGE
******************************** Top of Data ********************************
                              Build Map Contents
                              ------------------

Keyword  Member                                Type     Last Time Modified Ver
-------- ------------------------------------ -------- ------------------ ---
SINC     CLO00001                             JAVA     01/05/05  20:09:00 1
LIST     CLO00001                             JAVALIST 01/05/07  15:00:21 26
OUT1     CLO00000                             JAVACLAS 01/05/07  15:00:21 13
****************************** Bottom of Data ******************************
```

Figure 41. Build Map List Example

Step 9: Invoking the Compiled Java

Copy the Clock2.class and the clock.html to a known location on a Web Server. For example, clock.html to the HTTP root directory and copy the Clock2.class to HTTP root directory/classes/ directory. You should be able to invoke the compiled Java code by entering the same ip-address and port as Cloud 9, but instead of using Cloud9.htm use the clock.html request.

Enter the following in the location area of your browser:

```
Ip-address:port/clock.html
```

CHECKPOINT #4 for S-JDK Installation

At this point the following tasks should be complete.

Table 23. Checkpoint #4 for S-JDK Installation

Task	Completed?
Copied CLZJCLCK to a Cloud 9 or a USS location as Clock2.java	
Added Clock2.java as a JAVA element into SCLM via Cloud 9	
Reviewed translator output	

Table 23. Checkpoint #4 for S-JDK Installation (continued)

Task	Completed?
Reviewed population of USS Output directories	
Copied Clock2.class to the HTTP /rootdir/classes/ directory	
Invoked *clock.html* from a web server location	

Part 3. S-FTP Remote Build and Deploy

Chapter 13. Installation Overview for S-FTP

This part of the manual contains the installation procedure for the IBM Cloud 9 for SCLM for z/OS SCLM-File Transfer Protocol feature. This feature is a set of Remote Build and Deployment Scripts based in FTP. Hereinafter, the following names will be used in this manual:

- IBM Cloud 9 for SCLM for z/OS will be called ***Cloud 9***
- IBM Cloud 9 for SCLM for z/OS S-FTP will be called ***S-FTP***

The steps in this part are organized into five major sections:

- Software requirements and assumptions
- S-FTP SLR and SCLM defaults and setup
- Customizing translator and REXX script
- Defining the S-FTP translator to your SCLM Project Definition
- Testing the S-FTP translator

Modifying Case Sensitive S-FTP Values

During this installation, you will modify several JCL members and REXX Scripts. Some of these files contain case sensitive values. It is imperative that *before* modifying the files, you issue the CAPS OFF command to ensure that automatic upper casing does not occur. You will be reminded of this case sensitivity issue where appropriate throughout this manual.

Product Components Utilized During Installation

The following JCL members are modified during the installation process. They are located in SCLZJCL and are resident on the host. The names are provided here as an overview of naming standards and component functionality.

JCL Members Modified During Installation:

CLZ@FTP1 Cloud 9 FTP translator

CLZRFTP1 REXX Script

Chapter 14. A Step-by-Step Approach

Table 24. S-FTP Installation Steps

Before you begin...	
♦	Review system, software, and hardware considerations.
Review Default Inventory locations and USS locations	
1.	Review default S-FTP values and determine actual inventory and FTP targets to use.
2.	Review SLR and SCLM definitions for FTP supported Types.
CP1.	Verify steps as shown in “CHECKPOINT #1 for S-FTP Installation” on page 113.
3.	Review and modify the FTP translator and REXX Script.
Create SCLM and Cloud 9 Definitions	
4.	Include CLZ@FTP1 in your SCLM Project Definition.
CP2.	Verify steps as shown in “CHECKPOINT #2 for S-FTP Installation” on page 121.
Perform Installation Verification Procedures	
5.	Add an HTML file and perform a build on the file.

Chapter 15. Before You Begin

Review Software and Hardware Considerations

In this step you will review the system, software, and hardware requirements for S-FTP installation.

System Requirements

To successfully install Cloud 9 S-FTP, the following system requirements must be in place at your installation:

Table 25. System Requirements

z/OS Operating System	Version 2 Release 7 (or higher)
z/OS FTP Server	Standard z/OS
Target FTP Server	Specific to platform
IBM Cloud 9	Cloud 9 installed and configured

Assumptions

The installation administrator understands how to define types to SCLM.

The installation administrator understands how to configure FTP on the z/OS and target FTP platforms. The task of configuring the FTP servers might already be completed at this point. If not, the network system's administrator will have to be contacted to perform the work.

The installation administrator has identified which Types are to be deployed to remote locations, and has defined them in the SCLM Project Definition and the SLR (for cross platform Types).

The installation administrator understands REXX. The CLZRFTP1 REXX Script needs to be modified. While this manual provides guidelines, the modifications are best performed by someone who has REXX experience.

Chapter 16. Review Default Values and Targets

Step 1. Review and Set S-FTP Inventory Values

The S-FTP is delivered with default values.These default values are meant to be modified throughout the translator and REXX script. Prior to making modifications to these members, please review the following worksheets and fill in your site specific Inventory Values. It is important to review and record all possible targets for builds and remote deployment.

SCLM Inventory and REXX Value Worksheet

Table 26. SCLM Inventory and REXX Value Worksheet

SCLM Inventory Name	Default Value	Your Values
SCLM Group	DEV, TEST are used as default group names in CLZRFTP1	
SCLM MAKE Type Referenced	MAKE is referenced as the default 'make' type in CLZRFTP1	
SCLM Binary Types Referenced	GIF, JPG are referenced as examples of types that have binary FTP attributes in CLZRFTP1	
SCLM Language	FTP1 - set in CLZ@FTP1.	
User	Userid is used in CLZRFTP1 in several places.	
Dir	Used in all four of the Set Target Examples in CLZRFTP1. • For USS: '/u/userdir' • For ISP: '/isp/userdir' • For C drive: 'c:\userdir' • For A drive: 'a:\userdir'	
Ip-address	The ip-address is set up as a dummy value of 999.999.999.99 in all four of the Set Target Examples in CLZRFTP1.	
Port	The Port is set to a default of 21 in all four of the Set Target Examples CLZRFTP1.	
Userlog	Userid.ftplog is used at the end of CLZRFTP1. It is the default dataset name for the FTP log file	

FTP Target Platform Worksheet

Table 27. FTP Target Platform Worksheet

Platform Type	Type Deployed	Group Location Deployed	Ip-address/port	Userid/ Password	Target Directory	Target Directories Defined? Yes/No	Target Security Yes/No	FTP server type at target
ISP - NT	HTML	TEST	999.999.999.99/ 21	Userid/ pass	/isp/userdir	Yes	No	Yes
Your Platform Here								

Step 2. Review SCLM and Cloud 9 Type Definitions

Before moving on to the REXX and Translator modification, you should review all Types selected for deployment. First fill in each unique type in Table 28. Then, for each type, review the definition in SCLM and the Cloud 9 SLR. Check for the type's existence, consistency, and correctness.

You should also be sure that the FTP target directories are defined. Use Table 27 to document that the target directories are defined. You may also need to meet with your Network Administrator to review security for the target platforms

Type Review Matrix

Table 28. Type Review Matrix

Type	Language is FTP1	Defined to SCLM	Defined to Cloud 9 SLR	Case Sensitive Yes/No	Binary (lrecl = 256)

Determining That Types Exist

To determine if a type exists, go to Cloud 9, list SCLM members. For the Group, list the types and verify that the types are there. If they do not exist, then you must define the types and data sets to SCLM. The language should not be there yet.

Determining Cloud 9 Definitions

To determine if the SCLM type is defined to the Cloud 9 SLR (long name registry) run the IVP JCL member CLZC9J06. Review the output from the job, verifying that the SCLM has been defined with the proper attributes. If not, modify this job to define the Types to Cloud 9.

Determining LRECL and File Attributes

To determine the LRECL of the type, go into SCLM Option 3.2 and display the dataset information for one of the type datasets. If binary the Lrecl will be 256 and the RECFM = VB.

CHECKPOINT #1 for S-FTP Installation

At this point the following tasks should be completed.

Table 29. Checkpoint #1 for S-FTP Installation

Data Set Names	Completed?
Review all SCLM and FTP Inventory Default Values	
Determine key SCLM target locations	
Determine Target Locations/Type Matrix	
Ensure that SCLM Types are defined to SCLM and Cloud 9, properly.	
Ensure that the FTP target directories are defined.	

Step 3.Review and Modify Translator and REXX Script

Review and Modify CLZ@FTP1

In this step you will review and modify the CLZ@FTP1 member found in the **CLZ.SCLZJCL** file delivered with the SMP/E installation of Cloud 9. There is not a lot of modification for this member, the translator uses default IBM naming standards for all product files.

1. Using ISPF EDIT, access member CLZ@FTP1 in the CLZ.SCLZJCL library.
2. Ensure that the CLZ.SCLZJCL data set is in the SYSLIB concatenation of Project Definition ***JCL***. Either include the data set or copy this member to a library in the **SYSLIB** concatenation.

```
*********************************************************************
* NAME:    CLZ@FTP1                                                 *
* PURPOSE: INVOKE CLOUD 9 FTP REXX SCRIPT TO DEPLOY AND/OR BUILD    *
*          REMOTE OBJECTS. ( SCRIPT NAME IS CLZRFTP1)               *
*********************************************************************
* NOTE:    THIS PROTOTYPE TRANSLATOR REQUIRES CUSTOMIZATION         *
*          BY THE CUSTOMER OR INSTALLER.                            *
*          THE ACTUAL DEPLOYMENT OR BUILD REQUEST IS PERFORMED      *
*          BY THE REXX EXEC CLZRFTP1. THIS REXX EXEC NEEDS TO BE    *
*          CUSTOMIZED TO MEET THE INVENTORY NAMES AND TARGET        *
*          LOCATIONS REQUIRED BY THE CUSTOMER.                      *
*                                                                   *
*********************************************************************
* CHANGE ACTIVITY:                                                  *
*                                                                   *
*                                                                   *
*********************************************************************
         FLMLANGL    LANG=FTP1,VERSION=TEXTV1.0
*********************************************************************
*
* PARSER TRANSLATOR
*
*********************************************************************
          FLMTRNSL   CALLNAM='SCLM TEXT PARSE',                     C
               FUNCTN=PARSE,                                        C
               COMPILE=FLMLPGEN,                                    C
               PORDER=1,                                            C
               OPTIONS=(SOURCEDD=SOURCE,                            C
               STATINFO=@@FLMSTP,                                   C
               LISTINFO=@@FLMLIS,                                   C
               LISTSIZE=@@FLMSIZ,                                   C
               LANG=T)
*          (* SOURCE      *)
           FLMALLOC  IOTYPE=A,DDNAME=SOURCE
            FLMCPYLB @@FLMDSN(@@FLMMBR)
*********************************************************************
*
```

Figure 42. CLZ@FTP1 S-FTP Translator (Part 1 of 3)

```
* BUILD TRANSLATOR
*
*  STEP 1A: GET THE LONGNAME FROM THE SLR
*
*********************************************************************
         FLMTRNSL FUNCTN=BUILD,CALLNAM='STEP1A: CLZFPARM',             X
               COMPILE=CLZFPARM,DSNAME=CLZ.SCLZLOAD,                   X
               OPTIONS='NEW,LIST LONGNAME WHERE SHORTNAME = @@FLMMBR.'
         FLMALLOC DDNAME=CIGLOG,IOTYPE=W
         FLMALLOC DDNAME=CIGPOUT,IOTYPE=W
*                                        COPY CIGPOUT TO CIGIN
         FLMTRNSL FUNCTN=BUILD,CALLNAM='STEP1B: CLZFCOPY',             X
               COMPILE=CLZFCOPY,DSNAME=CLZ.SCLZLOAD,                   X
               OPTIONS='CIGPOUT ,CIGIN   '
         FLMALLOC DDNAME=CIGPOUT,IOTYPE=U
         FLMALLOC DDNAME=CIGIN,IOTYPE=W
         FLMALLOC DDNAME=CIGLOG,IOTYPE=W,PRINT=Y
*                                        COPY CIGPUNCH TO CIGIN
         FLMTRNSL FUNCTN=BUILD,PORDER=0,CALLNAM='STEP1C: CLZSLR',      X
               COMPILE=CLZSLR,DSNAME=CLZ.SCLZLOAD
         FLMALLOC DDNAME=CIGLOG,IOTYPE=W
         FLMALLOC DDNAME=CIGIN,IOTYPE=U
         FLMALLOC DDNAME=CIGPUNCH,IOTYPE=W
*                                        COPY CIGPUNCH TO FTPIN
         FLMTRNSL FUNCTN=BUILD,CALLNAM='STEP2D: CLZFCOPY',             X
               COMPILE=CLZFCOPY,DSNAME=CLZ.SCLZLOAD,                   X
               OPTIONS='CIGPUNCH,FTPIN   '
         FLMALLOC DDNAME=CIGPUNCH,IOTYPE=U
         FLMALLOC DDNAME=FTPIN,IOTYPE=W
         FLMALLOC DDNAME=CIGLOG,IOTYPE=W
*********************************************************************
*
*  STEP 2:  CREATE FTP COMMANDS AND CALL FTP
*
*********************************************************************
         FLMTRNSL FUNCTN=BUILD,CALLNAM='STEP2A: CLZRFTP1',             X
               COMPILE=CLZRFTP1,                                       X
               CALLMETH=TSOLNK,                                        X
               DSNAME=CLZ.SCLZJCL,                                     X
               OPTIONS='MEMBER=@@FLMMBR,PROJECT=@@FLMPRJ,GROUP=@@FLMGRPX
               ,TYPE=@@FLMTYP'
         FLMALLOC DDNAME=FTPIN,IOTYPE=U
         FLMALLOC DDNAME=INPUT,IOTYPE=W            USED BY FTP
         FLMALLOC DDNAME=OUTPUT,IOTYPE=W           USED BY FTP
*********************************************************************
*
* PROMOTE TRANSLATOR
*
*  STEP 1A: GET THE LONGNAME FROM THE SLR
*
*********************************************************************
         FLMTRNSL FUNCTN=COPY,CALLNAM='STEP1A: CLZFPARM',              X
               COMPILE=CLZFPARM,DSNAME=CLZ.SCLZLOAD,                   X
               OPTIONS='NEW,LIST LONGNAME WHERE SHORTNAME = @@FLMMBR.'
```

Figure 42. CLZ@FTP1 S-FTP Translator (Part 2 of 3)

```
FLMALLOC DDNAME=CIGLOG,IOTYPE=W
         FLMALLOC DDNAME=CIGPOUT,IOTYPE=W
*                                        COPY CIGPOUT TO CIGIN
         FLMTRNSL FUNCTN=COPY,CALLNAM='STEP1B: CLZFCOPY',              X
               COMPILE=CLZFCOPY,DSNAME=CLZ.SCLZLOAD,                   X
               OPTIONS='CIGPOUT ,CIGIN   '
         FLMALLOC DDNAME=CIGPOUT,IOTYPE=U
         FLMALLOC DDNAME=CIGIN,IOTYPE=W
         FLMALLOC DDNAME=CIGLOG,IOTYPE=W,PRINT=Y
*                                        COPY CIGPOUT TO CIGIN
         FLMTRNSL FUNCTN=COPY,PORDER=0,CALLNAM='STEP1C: CLZSLR',       X
               COMPILE=CLZSLR,DSNAME=CLZ.SCLZLOAD
         FLMALLOC DDNAME=CIGLOG,IOTYPE=W
         FLMALLOC DDNAME=CIGIN,IOTYPE=U
         FLMALLOC DDNAME=CIGPUNCH,IOTYPE=W
*                                        COPY CIGPUNCH TO FTPIN
         FLMTRNSL FUNCTN=COPY,CALLNAM='STEP2D: CLZFCOPY',              X
               COMPILE=CLZFCOPY,DSNAME=CLZ.SCLZLOAD,                   X
               OPTIONS='CIGPUNCH,FTPIN   '
         FLMALLOC DDNAME=CIGPUNCH,IOTYPE=U
         FLMALLOC DDNAME=FTPIN,IOTYPE=W
         FLMALLOC DDNAME=CIGLOG,IOTYPE=W
*********************************************************************
*
*  STEP 2:  CREATE FTP COMMANDS AND CALL FTP
*
*********************************************************************
         FLMTRNSL FUNCTN=COPY,CALLNAM='STEP2A: CLZRFTP1',              X
               COMPILE=CLZRFTP1,                                       X
               CALLMETH=TSOLNK,                                        X
               DSNAME=CLZ.SCLZJCL,                                     X
               OPTIONS='MEMBER=@@FLMMBR,PROJECT=@@FLMPRJ,GROUP=@@FLMGRPX
               ,TYPE=@@FLMTYP'
         FLMALLOC DDNAME=FTPIN,IOTYPE=U
         FLMALLOC DDNAME=INPUT,IOTYPE=W             USED BY FTP
         FLMALLOC DDNAME=OUTPUT,IOTYPE=W            USED BY FTP
```

Figure 42. CLZ@FTP1 S-FTP Translator (Part 3 of 3)

Review and Modify CLZRFTP1 REXX Script

In this step you will review and modify the CLZRFTP1 REXX Script found in the **CLZ.SCLZJCL** library delivered with the SMP/E installation of Cloud 9.

Note: Issue the CAPS OFF command BEFORE you start to edit this member.

1. Using ISPF EDIT, access member CLZRFTP1 in the CLZ.SCLZJCL library.
2. Issue the CAPS OFF command to ensure case sensitivity.
3. Substitute your site-specific values (identified in Table 27 on page 112)

```
/* rexx */
/*  ------------------------------------ */
/*  CLZRFTP1 - USED WITH CLZ@FTP1 TRANSLATOR*/
/*  ------------------------------------ */
/*  This is a rexx program that invokes FTP */
/*  to ship SCLM    inventory to specified  */
/*  locations out in the network.           */
/*                                          */
/*  This is prototype and must be customized*/
/*  by the user.                            */
/*                                          */
/*  ------------------------------------ */

    TRACE ALL           /* Delete this to eliminate trace */

/*  ------------------------------------- */
/*  The input parameters passed by the caller */
/*  are in the following order:               */
/*                                            */
/*  1:  member=shortname                      */
/*  2:  project=project                       */
/*  3:  group=group                           */
/*  4:  type=type                             */
/*  ------------------------------------- */

    parse arg request
    fx = 0
    true = 1
    false = 0
    ftpIdx=0

    call GetParms
    call GetPCFileName
    call GetTranslationType
/*
   GROUP=DEV
*/
    if (group == 'DEV') then do

       /* -------------------- */
       /* Set Target Location   */
       /* Example of ftp to ISP */
       /* -------------------- */

       /*  ftp -> Surfnet web2.surfnetcorp.com */
        ipaddr='999.999.999.99'; port=21
        user='userid'; password='pass'; dir='/isp/userdir'

       /* ------------------------------------------------------ */
       /* Build FTP commands and invoke FTP.                     */
       /* ------------------------------------------------------ */

        call InvokeFTP
```

Figure 43. CLZRFTP1 REXX Script (Part 1 of 5)

```
      /* ---------------------------------------------------- */
      /* Set Target Location                                  */
      /* Example of ftp to a OS/390 Unix System Services location */
      /* ---------------------------------------------------- */

       /*  ftp -> os/390 */
       ipaddr='999.999.999.99'; port=21
       user='userid'; password='pass'; dir='/u/userdir'

      /* ---------------------------------------------------- */
      /* Build FTP commands and invoke FTP.                   */
      /* ---------------------------------------------------- */

       call InvokeFTP

   end
/*
   GROUP=QA
*/
    if (group == 'QA') then do


      /* ---------------------------------------------------- */
      /* Set Target Location                                  */
      /* Example of ftp to a Windows machine on the network.  */
      /* ---------------------------------------------------- */

       /*  ftp -> CIG demo machine   */
       ipaddr='999.999.999.99'; port=21
       user='userid'; password='pass'; dir='c:\userdir'

      /* ---------------------------------------------------- */
      /* Build FTP commands and invoke FTP.                   */
      /* ---------------------------------------------------- */

       call InvokeFTP


      /* ---------------------------------------------------- */
      /* Set Target Location                                  */
      /* example of ftp to a the A: drive to cut a disk.      */
      /* ---------------------------------------------------- */

      /*  ftp -> Demo Machine A: Drive*/

       ipaddr='999.999.999.99'; port=21
       user='userid'; password='pass'; dir='a:\userdir'

      /* ---------------------------------------------------- */
      /* Build FTP commands and invoke FTP.                   */
      /* ---------------------------------------------------- */

       call InvokeFTP
    end
return
```

Figure 43. CLZRFTP1 REXX Script (Part 2 of 5)

```
/* ---------------------------------------------------------------- */
/*   Get parms                                                      */
/*   There will be four parms.                                      */
/* ---------------------------------------------------------------- */
GetParms:
    do while (request ¬= '')
        parse var request parm','request
        parse var parm var1"="val1
        interpret var1"="'val1'
    end
return

/* ---------------------------------------------------------------- */
/*   Get longname from //FTPIN                                      */
/* ---------------------------------------------------------------- */
GetPCFileName:
    address MVS "EXECIO * DISKR FTPIN (STEM input. FINIS"
    i = input.0 - 1
    pcFileName = input.i
    pcFileName = strip(pcFileName,'B',' ')
/* ---------------------------------- */
/* Create default .exe name for return. */
/* ---------------------------------- */

     if type = 'MAKE' then
     do
      wheredot = lastpos('.',pcFileName)
      pcFileNamexe = substr(pcFileName,1,wheredot)||'exe'
     end

return

/* ---------------------------------------------------------------- */
/*   Get translation type                                           */
/*   The purpose of this routine is to determine the file attribute  */
/*   for the source file of the FTP. This is determined by extension */
/*   type. The first two lines capture the extension for analysis.   */
/*   These lines should not be modified.                            */
/* ---------------------------------------------------------------- */

GetTranslationType:
    fileExtension = substr(pcFileName,lastpos('.',pcFileName)+1)
    fileExtension = translate(fileExtension)    /* upper case */

/* ---------------------------------------------------------------- */
/* Modify the if statements to include your binary types here.       */
/*       GIF    = BINARY                                           */
/*       JPG    = BINARY                                           */
/*       others = ASCII   ( default )                              */
/* ---------------------------------------------------------------- */

    if (fileExtension == 'GIF')       then translationType = 'BINARY'
    else if (fileExtension == 'JPG')  then translationType = 'BINARY'
    else translationType = 'ASCII'

return
```

Figure 43. CLZRFTP1 REXX Script (Part 3 of 5)

```
/* ---------------------------------------------------------------- */
/*   Format FTP commands                                            */
/* ---------------------------------------------------------------- */
InvokeFTP:
    ftpIdx = 0
    ftpIdx=ftpIdx+1; ftp.ftpIdx=ipaddr' 'port
    ftpIdx=ftpIdx+1; ftp.ftpIdx=user
    ftpIdx=ftpIdx+1; ftp.ftpIdx=password
    ftpIdx=ftpIdx+1; ftp.ftpIdx="lcd '"project"."group"."type"'"
    ftpIdx=ftpIdx+1; ftp.ftpIdx="cd "dir

      /* ----------------------------------------  */
      /* sync the source file                     */
      /* ----------------------------------------  */

    ftpIdx=ftpIdx+1; ftp.ftpIdx="put "member" "pcFileName

      /* ----------------------------------------- */
      /* if 'make' type then issue exec MAKE command  */
      /* ----------------------------------------- */

       if (type  == 'MAKE' ) then do

          /* ----------------------------------------  */
          /* execute the make file on the remote machine */
          /* ----------------------------------------  */

          ftpIdx=ftpIdx+1; ftp.ftpIdx="quote site exec " pcFileName

          /* ----------------------------------------  */
          /* reset the host target directory to 'exe'.   */
          /* reset translation type to binary.           */
          /* request a return of the 'exe' to host.      */
          /* ----------------------------------------  */

          ftpIdx=ftpIdx+1; ftp.ftpIdx="lcd '"mbrexe"'"
          ftpIdx=ftpIdx+1; ftp.ftpIdx="binary"

          /* ----------------------------------------  */
          /* assumption: the exec name is same as the    */
          /* .c source.  This will not always be the     */
          /* case.  Per compiler, the rules of output    */
          /* creation and naming standards will need     */
          /* to be examined.                             */
          /* ----------------------------------------  */

          ftpIdx=ftpIdx+1
          ftp.ftpIdx="get "pcFileNamexe member" "mbrexe "replace"

          ftpIdx=ftpIdx+1; ftp.ftpIdx="lcd '"mbrlog"'"
          ftpIdx=ftpIdx+1; ftp.ftpIdx="ASCII"
          ftpIdx=ftpIdx+1

       end

    ftpIdx=ftpIdx+1; ftp.ftpIdx="quit"
    ftp.0 = ftpIdx
```

Figure 43. CLZRFTP1 REXX Script (Part 4 of 5)

```
    /* ------------------------------------------- */
    /* write ftp commands to //input and invoke ftp */
    /* ------------------------------------------- */

    address MVS "EXECIO * DISKW INPUT (STEM ftp. FINIS"

    /* ------------------------------------------- */
    /* Attach FTP and execute commands.            */
    /* ------------------------------------------- */

    address attach FTP       /* here we invoke FTP */

    /* --------------------------- */
    /* read ftp output into an array */
    /* --------------------------- */

    address MVS "EXECIO * DISKR OUTPUT (STEM input. FINIS"

    /* --------------------------- */
    /* copy to a user dataset      */
    /* --------------------------- */

    "alloc dd(MSGFILE) mod reuse da('"USERID()".FTPLOG')"
    "EXECIO * DISKW MSGFILE (STEM input. FINIS"

    "free dd(MSGFILE)"
return
```

Figure 43. CLZRFTP1 REXX Script (Part 5 of 5)

Step 4. Update Your Project Definition

The purpose of this step is to review the requirements for updating your current project definition to include the new CLZ@FTP1 translator. The translator is included in the CLZ.SCLZJCL library. Include this library in your project definition JCL SYSLIB DDNAME so that the compiler can find the language definition member. The following shows the statements required to include the new translator. Include these statements in your current project definition JCL and submit.

```
   ************************************************************************
   *                LANGUAGE DEFINITION TABLES
   ************************************************************************
   *
   *
           COPY  CLZ@FTP1           -- FTP BUILD/DEPLOY           --
```

Figure 44. Copy Statement Example

CHECKPOINT #2 for S-FTP Installation

At this point you should have completed the following tasks.

Table 30. Checkpoint #2 for S-FTP Installation

Task	Completed?
Reviewed and modified translator CLZ@FTP1	
Reviewed and modified REXX script CLZRFTP1	

Table 30. Checkpoint #2 for S-FTP Installation (continued)

Task	Completed?
Updated your project definition include the CLZ@FTP1	

Chapter 17. Test the S-FTP Translator

Step 5. Add an HTML File

To test the S-FTP Translator, perform the following steps:

1. Add a piece of source code defined to the type and language supported. For example, add a type called HTML with a language of FTP1. For information about adding source to SCLM libraries, refer to the *IBM Cloud 9 for SCLM for z/OS User's Guide*.
2. Through Cloud 9, request a build of the source to invoke the **Build CLZ@FTP1** translator. View the batch job output. The following is an example of the BLDMSGS output.

```
********************* TOP OF DATA ***************************
FLM42000 - BUILD PROCESSOR INITIATED - 16:52:33 ON 01/04/30

FLM44500 - >> INVOKING BUILD TRANSLATOR(S) FOR TYPE: HTML MEMBER: WE$SH001
FLM06501 - TRANSLATOR RETURN CODE FROM ===> STEP1A: CIGFPARM     ===> 0
FLM06501 - TRANSLATOR RETURN CODE FROM ===> STEP1B: CIGFCOPY     ===> 0
FLM06501 - TRANSLATOR RETURN CODE FROM ===> STEP1C: C9LSLR       ===> 0
FLM06501 - TRANSLATOR RETURN CODE FROM ===> STEP2D: CIGFCOPY     ===> 0
FLM06501 - TRANSLATOR RETURN CODE FROM ===> STEP2A: IRXJCL       ===> 0
FLM46000 - BUILD PROCESSOR COMPLETED - 16:53:03 ON 01/04/30
******************** BOTTOM OF DATA ***************************
```

Figure 45. BLDMSGS Output

3. The following is an example of the output produced by the S-FTP REXX exec because it has the "trace all" command coded into it. It shows you the progress of the FTP process and where problem areas (if any) will occur. Errors appear as non-zero return codes. Use this output in conjunction with the log in Figure 47 on page 126 to follow the complete process.

```
READY
  ISPSTART CMD(%TEMPNAME)
    3 *-* /*  ---------------------------------------- */
    4 *-* /*  CLZRFTP1 - USED WITH CLZ@FTP1 TRANSLATOR  */
    5 *-* /*  ---------------------------------------- */
    6 *-* /*  This is a rexx program that invokes FTP   */
    7 *-* /*  to ship SCLM    inventory to specified    */
    8 *-* /*  locations out in the network.             */
    9 *-* /*                                           */
   10 *-* /*  This is prototype and must be customized  */
   11 *-* /*  by the user.                              */
   12 *-* /*                                           */
   13 *-* /* ----------------------------------------   */
   16 *-* /* ----------------------------------------  */
   17 *-* /* The input parameters passed by the caller  */
   18 *-* /* are in the following order:                */
   19 *-* /*                                           */
   20 *-* /*  1:  member=shortname                      */
   21 *-* /*  2:  project=project                       */
   22 *-* /*  3:  group=group                           */
   23 *-* /*  4:  type=type                            */
   24 *-* /* ---------------------------------------- */
   26 *-* parse arg request
   27 *-* fx = 0
   28 *-* true = 1
   29 *-* false = 0
   30 *-* ftpIdx=0
   32 *-* call GetParms
  100 *-*  GetParms:
  101 *-*  do while (request = '')
  102 *-*   parse var request parm','request
  103 *-*   parse var parm var1"="val1
  104 *-*   interpret var1"="'val1'
      *-*    MEMBER=val1

(More..)

   38 *-* if (group == 'DEV1')
     *-*  then
     *-*  do
  40 *-*   /* ------------------------------------------------ */
  41 *-*   /* Set Target Location                              */
  42 *-*   /* Example of ftp to a z/OS Unix System Services location   */
  43 *-*   /* ------------------------------------------------------- */
  45 *-*   /*  ftp -> z/OS   */
```

Figure 46. REXX Script Trace Data (Part 1 of 2)

```
46 *-*   ipaddr='999.99.999.99'
    *-*   port=21
 47 *-*   user='?????'
    *-*   password='?????'
    *-*   dir='/u/cig8002/c9demo'
 49 *-*   /* ------------------------------------------------ */
 50 *-*   /* Build FTP commands and invoke FTP.               */
 51 *-*   /* ------------------------------------------------ */
 53 *-*   call InvokeFTP
157 *-*    InvokeFTP:
158 *-*    ftpIdx = 0
 159 *-*    ftpIdx=ftpIdx+1
     *-*    ftp.ftpIdx=ipaddr' 'port
 160 *-*    ftpIdx=ftpIdx+1
     *-*    ftp.ftpIdx=user
 161 *-*    ftpIdx=ftpIdx+1
     *-*    ftp.ftpIdx=password
 162 *-*    ftpIdx=ftpIdx+1
     *-*    ftp.ftpIdx="lcd '"project"."group"."type"'"
 163 *-*    ftpIdx=ftpIdx+1
     *-*    ftp.ftpIdx="cd "dir
 165 *-*    /* ----------------------------------------  */
 166 *-*    /* sync the source file                     */
 167 *-*    /* ----------------------------------------  */
 169 *-*    ftpIdx=ftpIdx+1
     *-*    ftp.ftpIdx="put "member" "pcFileName
 171 *-*    /* ------------------------------------------ */
 172 *-*    /* if 'make' type then issue exec MAKE command  */
 173 *-*    /* ------------------------------------------ */
 175 *-*    if (type  == 'MAKE' )
 209 *-*    ftpIdx=ftpIdx+1
     *-*    ftp.ftpIdx="quit"
 210 *-*    ftp.0 = ftpIdx
 212 *-*    /* ------------------------------------------ */
 213 *-*    /* write ftp commands to //input and invoke ftp */
 214 *-*    /* ------------------------------------------ */
 216 *-*    address MVS "EXECIO * DISKW INPUT (STEM ftp. FINIS"
     >>>      "EXECIO * DISKW INPUT (STEM ftp. FINIS"
 218 *-*    /* ------------------------------------------ */
 219 *-*    /* Attach FTP and execute commands.           */
 220 *-*    /* ------------------------------------------ */
 222 *-*    address attach FTP       /* here we invoke FTP */
     >>>      "FTP"
 224 *-*    /* ---------------------------- */
 225 *-*    /* read ftp output into an array */
 226 *-*    /* ---------------------------- */
 228 *-*    address MVS "EXECIO * DISKR OUTPUT (STEM input. FINIS"
     >>>      "EXECIO * DISKR OUTPUT (STEM input. FINIS"
 230 *-*    /* ---------------------------- */
 231 *-*    /* copy to a user dataset       */
 232 *-*    /* ---------------------------- */
 234 *-*    "alloc dd(MSGFILE) mod reuse da('CIGT.FULL.RBLOG')"
     >>>      "alloc dd(MSGFILE) mod reuse da('CIGT.FULL.RBLOG')"
 235 *-*    "EXECIO * DISKW MSGFILE (STEM input. FINIS"
     >>>      "EXECIO * DISKW MSGFILE (STEM input. FINIS"
 237 *-*    "free dd(MSGFILE)"
     >>>      "free dd(MSGFILE)"
 238 *-*    return
  55 *-*  end
```

Figure 46. REXX Script Trace Data (Part 2 of 2)

4. View the 'userid.ftplog' data set updated in the CLZRFTP1 script. There should be data in the file and it should look roughly like the following. Any real ID's or ip-addresses have been changed to dummy values for security purposes.

```
 EZA1736I FTP
 EZA1450I IBM FTP CS V2R7  1998 282 22:42 UTC
 EZA1466I FTP: using TCPIP
 EZA1456I Connect to ?
 EZA1736I 999.99.999.99 21
 EZA1554I Connecting to:  999.99.999.99 port: 21.
 220-FTPD1 IBM FTP CS V2R7 at P390, 22:04:50 on 2001-04-30.
 220 Connection will close if idle for more than 20 minutes.
 EZA1459I NAME (999.999.999.99:XXXXX):
 EZA1701I >>> USER XXXXX
 331 Send password please.
EZA1789I PASSWORD:
EZA1701I >>> PASS
230 P390C is logged on.  Working directory is "XXXXX.".
EZA1460I Command:
EZA1736I lcd 'CIGDEMO.DEV1.HTML'
EZA2081I Local directory name set to partitioned data set CIGDEMO.DEV1.
EZA1460I Command:
EZA1736I cd /u/cig8002/c9demo
EZA1701I >>> CWD /u/cig8002/c9demo
250 HFS directory /u/cig8002/c9demo is the current working directory
EZA1460I Command:
EZA1736I put WE$SHOUL  'We should ship the web cast this way!.html'
EZA1701I >>> SITE VARrecfm LRECL=256 RECFM=VB BLKSIZE=2564
200 Site command was accepted
EZA1701I >>> PORT 999,99,999,99,4,12
200 Port request OK.
EZA1701I >>> STOR  'We should ship the web cast this way!.html'
125 Storing data set /u/cig8002/c9demo/ We should ship the web cast this
250 Transfer completed successfully.
EZA1617I 52255 bytes transferred in 0.240 seconds.  Transfer rate 217.73
EZA1460I Command:
EZA1736I quit
EZA1701I >>> QUIT
```

Figure 47. User FTP Log Example

Part 4. Appendixes

Appendix A. Cloud 9 Unix Directory Structure

The following charts represent the Unix directory structure and files expected by the Cloud 9 application. The ***rootdir*** value is site specific, all other directory names and file names are not. This includes the case of the file names.

Level 1 — Cloud 9 'rootdir'

/rootdir/

Type	Perm	Changed	(GMT)	Owner	Size	File
_ Dir	755	10/11/2000	21:26	P390C	8192	.
_ Dir	755	09/29/2000	14:56	TCPIP	8192	..
_ Dir	755	10/11/2000	20:26	P390J	8192	cgi-bin
_ Dir	775	10/11/2000	21:25	TCPIP	8192	cloud9
_ File	755	10/11/2000	21:26	TCPIP	312	cloud9.htm
_ File	755	09/28/2000	21:27	P390J	15079	httpd.conf
_ File	755	09/28/2000	21:27	P390J	1249	httpd.envvars
_ File	755	09/28/2000	21:27	P390J	5101	httpd.mvsds
_ File	644	09/28/2000	21:47	P390J	10	httpd-pid
_ Dir	777	09/26/2000	07:26	P390J	8192	logs
_ Dir	777	09/26/2000	07:26	P390J	8192	reports

Level 2 — CGI-BIN Directory

/rootdir/cgi-bin/

Type	Perm	Changed	(GMT)	Owner	Size	File
_ Dir	755	03/19/2001	20:59	TCPIP	8192	.
_ Dir	755	06/08/2001	22:21	TCPIP	8192	..
_ File	755	03/10/2001	17:41	TCPIP	321	CLZRADDS
_ File	755	03/10/2001	17:41	TCPIP	0	CLZREDRV
_ File	755	03/10/2001	17:41	TCPIP	322	CLZRENDV
_ File	755	03/10/2001	17:41	TCPIP	331	CLZRINDX
_ File	755	03/10/2001	17:41	TCPIP	321	CLZRLMBR
_ File	755	03/10/2001	17:41	TCPIP	322	CLZRLUNX
_ File	755	03/10/2001	17:41	TCPIP	330	CLZRMENU
_ File	755	03/10/2001	17:41	TCPIP	322	CLZRMLST
_ File	755	03/10/2001	17:41	TCPIP	321	CLZRPROF
_ File	755	03/10/2001	17:41	TCPIP	321	CLZRSCLD
_ File	755	03/10/2001	17:41	TCPIP	321	CLZRSCLM
_ File	755	03/10/2001	17:41	TCPIP	322	CLZRSCMA
_ File	755	03/10/2001	17:41	TCPIP	0	CLZRSDRV
_ File	755	03/19/2001	21:18	P390S	115	CLZRSDSF
_ File	755	03/19/2001	21:18	P390S	115	CLZRSDSM
_ File	755	03/10/2001	17:41	TCPIP	391	CLZRSPDA
_ File	755	03/10/2001	17:41	TCPIP	322	CLZRULST

Level 2 — cloud9 Directory

/rootdir/cloud9/

Select one or more files with / or action codes.

Type	Perm	Changed	(GMT)	Owner	Size	File
_ Dir	755	10/11/2000	21:25	TCPIP	8192	.
_ Dir	755	10/11/2000	21:26	P390C	8192	..

```
_ File   755  03/10/2001 17:42     TCPIP      210   CLZHMENU.htm
_ File   755  03/19/2001 21:18     TCPIP       99   CLZHSDSB.htm
_ File   755  03/19/2001 21:18     TCPIP      210   CLZHSDSM.htm
_ File   755  03/10/2001 17:42     TCPIP      341   CLZHSPLA.htm
_ Dir    755  10/11/2000 21:21     TCPIP     8192   jcl
_ Dir    755  09/28/2000 21:36     TCPIP     8192   profiles
```

Level 3 — Profiles Directory

/rootdir/cloud9/profiles/

Select one or more files with / or action codes

```
  Type   Perm  Changed    (GMT)     Owner      Size    File

_ Dir    775  10/11/2000 21:28     TCPIP      8192    .
_ Dir    775  10/11/2000 21:25     TCPIP      8192    ..
_ File   755  09/28/2000 21:27     TCPIP     12133    P390C.jpg
_ File   755  09/28/2000 22:20     TCPIP       225    P390C.prf
_ File   600  09/28/2000 21:27     TCPIP     19529    P390K.jpg
_ File   755  10/05/2000 21:30     TCPIP       169    P390K.prf
```

Note the profiles should reflect the user ID's used during the install.

Level 3 — JCL Directory

/rootdir/cloud9/jcl/

Select one or more files with / or action codes.

```
  Type   Perm  Changed    (GMT)     Owner        Size     File

_ Dir    775  10/11/2000 21:21     TCPIP        8192     .
_ Dir    775  10/11/2000 17:33     TCPIP        8192     ..
_ File   755  10/11/2000 21:21     TCPIP        2431     CLZJDYN
_ File   755  10/11/2000 21:22     TCPIP        4791     CLZJIBM
_ File   755  10/11/2000 21:23     TCPIP        7126     CLZJMIG
```

Appendix B. Suite Long Name Registry

The Suite Long Name Registry (SLR) database contains both long name rules and actual data. This is the file where the correlation between the distributed platform object name and the standard z/OS name is maintained. It is referenced in the CIGINI file, in the CLOUD 9 section. The SLR is a standard KSDS VSAM file that needs to be maintained as all VSAM files need to be maintained.

The JCL for CLZSLR

The following is the JCL used to define the SLR long name rules. It is found in the Cloud 9 SCLZJCL library.

```
//**(JOBCARD)
//**
//*********************************************************************
//*                                                                   *
//* CLZC9J09 - THE PURPOSE OF THIS JCL TO RUN THE SLR UTILITY.        *
//*                                                                   *
//* SEE JCL MEMBER CLZC9J06 FOR AN EXAMPLE OF SYNTAX USAGE.           *
//*                                                                   *
//* SEE APPENDIX B, CLOUD 9 FOR SCLM INSTALL GUIDE FOR THE FULL       *
//* SET OF SYNTAX OPTIONS.                                            *
//*********************************************************************
//*                                                                   *
//* REQUIRED JCL MODIFICATION:                                        *
//*   1) INCLUDE A JOBCARD                                            *
//*                                                                   *
//*********************************************************************
//*                                                                   *
//*  STEP 1: ADD DATASET AND TYPE DEFINITIONS TO SLR DATABASE.        *
//*                                                                   *
//*********************************************************************
//STEP1    EXEC PGM=CLZSLR
//STEPLIB DD DSN=CLZ.SCLZLOAD,DISP=SHR
//CIGPUNCH DD SYSOUT=*
//CIGLOG   DD SYSOUT=*
//CIGIN   DD *
 INSERT RULES HERE.
/*
```

Figure 48. CLZC9J09 Utility JCL

The Utility — CLZSLR

The utility program CLZSLR is used for the following three functions, depending on which syntax is used as input:

1. Add/Delete/List Type definitions for SCLM or Data sets types.
2. Add/Delete/List a Short Name based on a given Long Name.
3. Add/Delete/List a Long Name based on a given Short Name.

The Syntax for Defining Types

The following is the syntax used for defining types and their attributes. This task is done during initial setup and installation. It is these rules that determine if a transaction is monitored for a distributed object type.

Data Set Version

```
ADD NAME RULE FOR DATASET 'dataset-name'
case sensitive|case insensitive
.
```

Figure 49. Long Name Rule Syntax for Datasets

SCLM Version

```
ADD NAME RULE FOR SCLM TYPE 'HostSCM-type'
case sensitive|case insensitive
.
```

Figure 50. Long Name Rule Syntax for SCLM

Keywords for Syntax

Table 31. Keywords for Long Name Rule Syntax

Keyword	Description	Notes
ADD|DELETE|LIST NAME RULE FOR SCLM TYPE *HostSCM-type*	These keywords and variable are required. The variable is a 1–8 character HostSCM-type that represents a distributed object type.	Required
Case sensitive | case-insensitive	This is an optional keyword that controls the representation of the distributed object name storage. Use of this parameter should reflect the platform case sensitivity requirements. For example, Unix and Linux are case sensitive, whereas Windows files are not.	Default is case insensitive. Ignored for the Delete and List verbs.

Examples of the Definition Syntax

Data Set Version

```
ADD NAME RULE FOR DATASET 'A.DEFAULT'.
ADD NAME RULE FOR DATASET 'A.CASE.SENSITIVE' CASE SENSITIVE LRECL 256.
ADD NAME RULE FOR DATASET 'A.CASE.INSENSITIVE' CASE INSENSITIVE.
```

Figure 51. Example of Rule Syntax for Data sets

SCLM Version

```
ADD NAME RULE FOR SCLM TYPE XLS .
ADD NAME RULE FOR SCLM TYPE UNIXMAKE CASE SENSITIVE .
ADD NAME RULE FOR SCLM TYPE DOC      CASE INSENSITIVE .
```

Figure 52. Example of Rule Syntax for SCLM

The Syntax for Adding, Deleting, and Listing Entries in the SLR

The following is the syntax used for adding, deleting, or listing entries in the SLR. This task is done during processor/translator execution or during any other utility that the user implements.

Short Name Syntax

```
ADD | DELETE | LIST SHORT NAME WHERE LONGNAME = 'long-name'
  DATASET 'dataset-name' | SCLM TYPE 'sclm-type'.
```

Figure 53. Short Name Syntax

Keywords for Short Name Syntax

Table 32. Keywords for Short Name Syntax

Keyword	Description	Notes
ADD|DELETE|LIST SHORT NAME WHERE LONG NAME = 'long-name'	These keywords and variable are required. The variable is a 1–255 character long name that is translated into a short name for the ADD.	The long name entry must exist for the DELETE and either case can be true for the LIST function. You cannot use wildcards in the name.
DATASET 'dataset-name' | SCLM TYPE 'sclm-type'	This keyword further defines the attributes of the names.	One of these keywords is required for the ADD and DELETE verbs but is optional for the LIST verb.

Examples of Short Name Syntax

```
delete short name where longname = 'Fiscal Year End 2000 Spread Sheets.xls'
  SCLM type xls.
```

Figure 54. Example of Short Name Syntax for SCLM

The following is an example of the output generated by a LIST Short Name Request. The short name appears first, with an asterisk in column 1.

```
* HEL00001
LIST SHORTNAME WHERE LONGNAME =
HELLO STEVE.
```

Figure 55. Example of LIST Short Name Output

Long Name Syntax

```
LIST LONG NAME WHERE SHORTNAME = 'short-name'.
```

Figure 56. Long Name Syntax

Keywords for Long Name Syntax

Table 33. Keywords for Long Name Syntax

Keyword	Description	Notes
LIST LONG NAME WHERE SHORT NAME = 'short-name'	These keywords and variable are required. The variable is a 1–8 character short name.	You cannot use wildcards in the name.

Examples of Long Name Syntax

```
list longname where shortname = 'HEL00001'.
```

Figure 57. Example of Long Name Syntax

The following is an example of the output generated by a LIST Long Name Request. The long name appears first, with an asterisk in column 1.

```
* HELLO STEVE
LIST LONGNAME WHERE SHORTNAME = HEL00001 .
```

Figure 58. Example of LIST Long Name Output

SLR in SCLM Translators

The z/OS SCLM tool is not aware that the short name stored in its repository is actually a long name somewhere else. From a programming object perspective, the short name is a fully-qualified member and normal object being tracked and promoted. As the short name is moved up the inventory maps, the reference to the long name still exists in the SLR. When the user lists against these from the Cloud 9 Browser interface, the long names will appear.

Notices

This information was developed for products and services offered in the U.S.A.

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 give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504–1785, USA.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries in writing to

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, 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 OR 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.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites 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 the IBM Corporation, Department TL3B, 3039 Cornwallis Road, Research Triangle Park, North Carolina, 27709–2195, 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.

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.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

- IBM
- SCLM
- z/OS
- WebSphere

Breeze, Cloud 9, and SOLEI are trademarks of Chicago Interface Group, Incorporated.

Internet Explorer and Windows are trademarks of Microsoft Corporation.

Netscape Navigator is a trademark of Netscape Communications Corporation.

CA-Endevor is a trademark of Computer Associates, Inc.

Other company, product, and service names may be trademarks or service marks of others.

Index

V

W

Program Number: 5655-G93

Printed in U.S.A.

SC31-8845-01