
Chicago Interface Group, Inc.

Cloud 9 for Endeavor Processor Guide



Version 7.0

Chicago Interface Group, Inc
858 West Armitage Avenue #286
Chicago, IL 60614 USA

Phone: (773) 524-0998
Fax: (815) 550-6088
Email: support@cigi.net
Web: www.cigi.net

Cloud 9 version 7.0

Cloud 9 is a trademark of Chicago Interface Group, Inc.
CA-Endevor is a registered trademark of Computer Associates International, Inc.

All rights reserved. © Copyright by Chicago Interface Group, 2007.
Documentation Version April 3, 2006

CONTENTS

Chapter 1: Getting Started	6
Introduction	6
Who should use this manual?	6
How to use this manual?	6
CHAPTER 1: E-FTP WEB DEPLOY AND REMOTE BUILDS	7
Chapter Scope.....	7
Product Components that require modification	8
JCL Members (located in JCL) Modified During Implementation:	8
A Step-by-Step Approach	9
Review Software Requirements and Assumptions	10
<i>System Requirements</i>	10
<i>Assumptions</i>	10
<i>Rexx Knowledge Required</i>	10
<i>FTP Server Requirements</i>	11
Step 1. Determine Endeavor Types and FTP target locations.....	12
FTP Target Platform Worksheet.....	12
Step 2. Review Endeavor and Cloud 9 Type definitions	13
Type Review Matrix	13
<i>Determining Types Exist?</i>	13
<i>Determining Cloud 9 Definitions?</i>	13
<i>Determining LRECL and File Attributes?</i>	14
CHECKPOINT #1.....	14
Step 3. Review and modify processors and REXX scripts	15
Usage Notes:.....	16

<i>Review and Modify EPRCSMAK Processor</i>	17
Usage Notes:	19
<i>Review and modify the NDVRFTP1 REXX Script</i>	19
<i>Review and modify the NDVRMAKE REXX Script</i>	22
Step 4: Add the Processors to Endeavor	26
CHECKPOINT #2	27
Step 5. Test the E-FTP Processor	28
CHAPTER 2: MANAGING JAVA WITH CLOUD 9 AND USS	30
Chapter Scope	30
<i>Modification of Case Sensitive E-JDK Values</i>	31
Product Components that require modification	31
<i>JCL Members (located in JCL) Modified During Implementation:</i>	31
<i>Parameters (located in HTML library) Modified During Implementation:</i> ...	32
A Step-by-Step Approach	33
Review Software and Hardware Considerations	34
<i>System Requirements</i>	34
Before You Begin: Implement Site Standards	35
<i>Site-Specific Placeholders</i>	35
<i>Endeavor Dataset Names</i>	35
Placeholder Worksheet	36
Dataset Worksheet	36
Step 1. Determine CA-Endeavor and USS Inventory Values	37
CA-Endeavor Inventory Value Worksheet.....	37
USS Directory Value Worksheet.....	39
CHECKPOINT #1	40
Step 2. Review and modify all Processors and Control Files	41
<i>E-JDK Processors:</i>	41
<i>E-JDK Processor Control Files:</i>	41
<i>Review and Modify C9JPPDEL Processor</i>	42
<i>Review and Modify C9JPPGEN Processor</i>	43
<i>Review and Modify C9JPPJAV Processor</i>	44
<i>Review and Modify C9JPPJDL Processor</i>	45
<i>Review and Modify C9JPPJMV Processor</i>	47
<i>Modify UNIXLOC – Common Endeavor to USS Life Cycle Mapping Rules</i> ...49	
<i>Special Note on JEXEC and JLIST</i>	49
<i>Modify JCPATH Common %CLASSPATH% Substitution</i>	51
<i>Modify JCOMPILE – JAVA Compile Shell:</i>	51
<i>Modify UNIXMAP Java Output Mapping Rules:</i>	52
<i>Modify JCOMPILR Jar Compile Shell:</i>	53

<i>Modify UNIXMAPR - Jar Output Mapping Rules:</i>	53
CHECKPOINT #2	54
Step 3: Run C9JPJINV to build E-JDK System, Types, and Processors Groups	55
<i>Modify and Submit C9JPJINV</i>	55
Step 4: Run C9JPJADD to add E-JDK Processors	61
<i>Modify and Submit C9JPJADD</i>	61
Step 5: Run C9JPJEE to load the E-JDK control records	63
<i>Modify and Submit C9JPJEE</i>	63
Step 6: Run C9JPJSLR to define E-JDK types to SLR	65
<i>Modify and Submit C9JPJSLR</i>	65
Step 7: Run C9JPJUNX to build E-JDK USS Directories	66
<i>Modify and Submit C9JPJUNX</i>	66
CHECKPOINT #3	68
Step 8. Test the E-JDK Processors	69
<i>Java Listing Example:</i>	69
<i>Java Endeavor Component List Example:</i>	70
Step 9: Invoking the Compiled Java	71
<i>Result of clockj.java Compile:</i>	71
CHECKPOINT #4	72
Step 10: Review the Cloud 9 USS Summary Reports	73
<i>Java Generate Summary Report:</i>	73
<i>Java Move Summary Report:</i>	75

Chapter 1: Getting Started

Introduction

Who should use this manual?

The audience for this manual is the people who will be implementing Endeavor processors for off host applications. Included in this manual are examples of FTP and JAVA/USS processors, rexx scripts, and setup requirements.

How to use this manual?

Each of the prototypes included in this manual include a set of instructions for setting up the Endeavor types and SLR definitions. They also include instructions on which pieces of the prototypes need to be customized. This manual should be used as a starting point for application implementation. None of the processors or prototypes are “drop in” solutions. All will require review and modification.

Chapter 1: E-FTP Web Deploy and Remote Builds

Chapter Scope

This chapter contains the instructions to customize and implement Cloud 9 E-FTP, the FTP based Deployment and Remote Build Scripts. The steps in this chapter are organized into five sections:

- Reviewing software requirements and assumptions.
- Review the E-FTP SLR and Endeavor defaults and setup
- Customizing processors and Rexx Scripts
- Defining the E-FTP processors to your Endeavor configuration.
- Testing the E-FTP Processors

These steps should be followed in the order that they are presented. Once you have successfully completed all of the steps and executed the test procedure in Step 5, you will be ready to use the E-FTP processors for remote builds and deploys.



During this installation, you will modify JCL members and REXX Scripts. Some of these files contain case-sensitive values. It is imperative that prior to modifying the members, you issue the CAPS OFF command to ensure that automatic upper casing of the members does not occur. For your convenience, the icon to the left will be placed in each step where case-sensitive values are an issue.

Product Components that require modification

The following JCL members are modified during the implementation process. The names are provided here as an overview of CIG naming standards and component functionality.

JCL Members (located in JCL) Modified During Implementation:

EPRCSFTP	Cloud 9 FTP Deploy Processor
EPRCSMAK	Cloud 9 Remote Build Processor
NDVRFTP1	REXX Script
NDVRMAKE	REXX Script

A Step-by-Step Approach

1	Review system, software, and hardware considerations.
2	Determine SLR and Endeavor definitions for FTP supported types and update both the SLR and Endeavor.
3	Review and modify the FTP processors and REXX scripts.
4	Add the modified processors to Endeavor.
5	Add an HTML file and perform a move on the file.

Enabling the E-FTP

Review Software Requirements and Assumptions

In this step you will review the system and software requirements for enabling the E-FTP.

System Requirements

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

z/OS 1.1	Version 1.1 or later
OS/390 or z/OS FTP Server	
Target FTP Server	

Assumptions

This chapter does not cover the definitions of types to Endeavor. This topic is covered in the Cloud 9 Administration Guide.

This chapter also does not cover configuring FTP on the z/OS platform or any of the target FTP platforms.

This chapter requires that you identify which types are to be deployed to remote locations. It is assumed that the types have been defined in the project definition and the SLR (for cross platform types). Step 4 conducts a review of both of these concepts.

Rexx Knowledge Required

The NDVRFTP1 and NDVRMAKE Rexx Scripts require modifications. This manual provides guidelines; however, the modifications are best performed by someone who has Rexx experience. This is not a canned, black box modification procedure.

FTP Server Requirements

This manual does not cover installing and configuring the FTP servers on the z/OS or any of the targets. This task is probably already complete. If not, the network system's programmers will have to be contacted to perform this work.

Step 1. Determine Endeavor Types and FTP target locations.

The E-FTP is delivered with no defaults values. It is up to the customer to determine which Endeavor types will be assigned the processors and scripts. Prior to making modifications to the processors and scripts, please review the worksheet below and fill in the site-specific Endeavor and FTP information. It is important to review and record all possible targets for builds and remote deployment.

FTP Target Platform Worksheet

The following is a table to be used for collecting proposed target platforms, types being FTP'd, userid/password for those platforms, and the type of FTP server on the target platform.

Worksheet for Deploy and Build Target Locations

Step 2. Review Endeavor and Cloud 9 Type definitions

Prior to moving on to the Rexx and Processor modification, it is advised that all types selected for deployment be determined. First fill in each unique type in the Type Review Matrix below. Then for each type, review the definition in Endeavor and Cloud 9 SLR for existence, consistency and correctness.

Also, it is recommended that the FTP target directories also be defined. Use the previous “FTP Target Platform Worksheet” 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

Type	Defined to Endeavor	Defined to Cloud 9 SLR	Case Sensitive Yes/No	Binary

Type Review Matrix

Determining Types Exist?

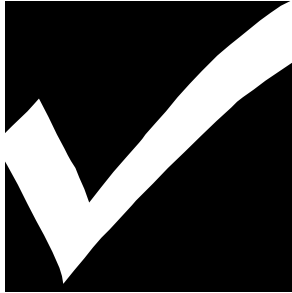
To determine if a type exists, go to Cloud 9 and list Endeavor elements. For the System, list the types and verify that the types are there. If they do not exist, then you must define the types and datasets to Endeavor.

Determining Cloud 9 Definitions?

To determine if the Endeavor type is defined to the Cloud 9 SLR (long name registry) run the IVP JCL member CIGV2IVP. Review the output from the job, verifying that the Endeavor Type 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 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

At this point the following tasks should have been completed.

Tasks	Completed?
Review all Endeavor and FTP Inventory Default Values	
Determine Target Locations/Type Matrix	
Ensure that Endeavor Types are defined to Endeavor and Cloud 9, properly.	
Ensure that the FTP target directories are defined.	

Checkpoint 1

Step 3. Review and modify processors and REXX scripts

The following sample processor can be found in the JCLLIB delivered via email or the product cartridge

1. Using ISPF EDIT, access member EPRCSFTP in the JCLLIB library.
2. There is not a lot of modification for this. The processor uses default CIG naming standards for all product files.

```
//EPRCSFTP PROC LISTLIB='&C1ST..&C1SY..&C1TY..LISTLIB',
//          WRKUNIT=SYSDA
//*****
//*
//*          THIS IS THE PROCESSOR THAT IS CALLED FOR TYPES THAT          *
//*          REQUIRE AN FTP PROCESS TO SEND THE FILE TO ANOTHER PLATFORM *
//*          IT IS DESIGNED AS PROTOTYPE USING THE FTP PROTOCOL          *
//*          CUSTOMIZATION IS REQUIRED FOR EACH CUSTOMER.                  *
//*
//***** ASSUMPTIONS MADE BY THIS PROTOTYPE *****
//*
//*          ELEMENT IS STORED IN REVERSE DELTA OR HAS AN SOL ASSOCIATED TO *
//*          THE ELEMENT TYPE. NO CONWRITE REQUIRED.                       *
//*          REXX PROGRAM BUILDS THE NAME OF THE PDS (IMAGE LIB OR SOL LIB) *
//*          BASED ON VARIABLES PASSED (SYSTEM, STAGE, TYPE) TO NDVRFPT1   *
//*          ELEMENT IS FOUND IN A PDS (FTP COMMANDS ASSUME PDS)          *
//*
//***** GENERAL NOTES *****
//*          THIS PROCESSOR SHOULD BE USED FOR FTP PROCESSOR GROUPS      *
//*
//*          PROCESSOR CALLS :
//*          C9LSLR - CIG SLR UTIL...
//*          1) GET LONG NAME (PC FILENAME) FOR ELEMENT (WRITES TO CIGPUNCH) *
//*          NDVRFPT1 - REXX PROGRAM...
//*          1) GET THE LONG NAME OF ELEMENT
//*          2) FTP FILE TO LONG NAME AT DESIRED LOCATION
//*          3) IF SUCCESSFUL, THE MESSAGES FROM FTP WILL BE TRIMMED TO
//*             DISPLAY MINIMAL MESSAGES, AND STORED IN LISTLIB
//*          4) IF FTP FAILS, THE FULL FTPOUT WILL BE SENT TO PRINT
//*****
//*          INITIALIZE TEMP DATASETS
//*****
//INITDSN EXEC PGM=BC1PDSIN
//C1INIT01 DD DSN=&&SLRDATA,DISP=(,PASS,DELETE),UNIT=&WRKUNIT,
//          SPACE=(TRK,(1,1),RLSE),LRECL=80,RECFM=FB,BLKSIZE=6400
//C1INIT02 DD DSN=&&FTPIN,DISP=(,PASS,DELETE),UNIT=&WRKUNIT,
//          SPACE=(TRK,(1,1),RLSE),LRECL=80,RECFM=FB,BLKSIZE=6400
//C1INIT03 DD DSN=&&FTPOUT,DISP=(,PASS,DELETE),UNIT=&WRKUNIT,
//          SPACE=(TRK,(1,1),RLSE),LRECL=80,RECFM=FB,BLKSIZE=6400
//C1INIT04 DD DSN=&&FTPMSG,DISP=(,PASS,DELETE),UNIT=&WRKUNIT,
//          SPACE=(TRK,(1,1),RLSE),LRECL=133,RECFM=FBA,BLKSIZE=2660
//*****
//*          GET THE LONGNAME FROM SOLEI'S SLR
//*****
//GETSLR EXEC PGM=C9LSLR
//STEPLIB DD DSN=FLHQ1.FLHQ2.LOADLIB,DISP=SHR
//CIGPUNCH DD DSN=&&SLRDATA,DISP=(OLD,PASS,DELETE)
//CIGLOG DD SYSOUT=*
//CIGIN DD *
```

```

LIST LONGNAME WHERE SHORTNAME = &C1ELEMENT .
/*
//*****
//* REXX TO CREATE FTP COMMANDS AND CALL FTP
//*****
//FTPSTEP EXEC PGM=IRXJCL,
// PARM='NDVRFTP1 &C1EL &C1USERID &C1EN &C1ST &C1TY'
//SYSEXEC DD DSN=FLHQ1.FLHQ2.ISRCLIB,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD DUMMY
//FTPIN DD DSN=&&SLRDATA,DISP=(OLD,DELETE,DELETE)
//INPUT DD DSN=&&FTPIN,DISP=(OLD,DELETE,DELETE)
//OUTPUT DD DSN=&&FTPOUT,DISP=(OLD,DELETE,DELETE)
//MSGFILE DD DSN=&&FTPMSG,DISP=(OLD,PASS,DELETE)
//*****
//* WRITE OUT LISTING IF FTP HAS ERRORS
//*****
//IFFTPERR (FTPSTEP.RC > 4) THEN
//LISTPRNT EXEC PGM=CONLIST,PARM='PRINT'
//C1BANNER DD DSN=&&BANNER,DISP=(,PASS,DELETE),
// UNIT=&WRKUNIT,SPACE=(TRK,(1,1)),
// DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6171,DSORG=PS)
//C1PRINT DD SYSOUT=*,
// DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6171,DSORG=PS)
//LIST01 DD DSN=&&FTPOUT,DISP=(OLD,DELETE)
//*****
//IFFTPERR ELSE
//* STORE THE LISTING IF FTP WAS SUCCESSFUL
//*****
//STEPNAME EXEC PGM=CONLIST,PARM='STORE'
//C1BANNER DD DISP=(,PASS,DELETE),
// UNIT=&WRKUNIT,SPACE=(TRK,(1,1)),
// DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6171,DSORG=PS)
//C1LLIBO DD DSN=&LISTLIB,DISP=SHR,FOOTPRNT=CREATE
//LIST01 DD DSN=&&FTPOUT,DISP=(OLD,DELETE)
//IFFTPERR ENDIF

```

EPRCSFTP E-FTP Processor

Usage Notes:

The purpose of this processor is to deploy objects stored in Endeavor out to some target location. For instance, the HTML for a Web site could be stored in Endeavor, but needs to be deployed on NT server. This processor would be used for such a purpose. The REXX exec would be used to logon and perform FTP services on behalf of the processor.

Review and Modify EPRCSMAK Processor

The following sample processor can be found in the Cloud 9 product JCLLIB.

1. Using ISPF EDIT, access member EPRCSMAK in the JCLLIB library.
2. There is not a lot of modification for this. The processor uses default CIG naming standards for all product files.

```
//EPRCSMAK PROC LISTLIB='&C1ST..&C1SY..&C1TY..LISTLIB',
//          WRKUNIT=SYSDA
//*****
//** THIS PROCESSOR IS DESIGNED AS PROTOTYPE USING THE FTP PROTOCOL *
//** AND REMOTE MAKES. CUSTOMIZATION IS REQUIRED FOR EACH CUSTOMER. *
//**
//***** ASSUMPTIONS MADE BY THIS PROTOTYPE *****
//**
//** *ELEMENT IS STORED IN REVERSE DELTA OR HAS AN SOL ASSOCIATED TO *
//** THE ELEMENT TYPE. NO CONWRITE REQUIRED. *
//** *REXX PROGRAM BUILDS THE NAME OF THE PDS (IMAGE LIB OR SOL LIB) *
//** BASED ON VARIABLES PASSED (SYSTEM, STAGE, TYPE) TO NDVRFPT1 *
//** *ELEMENT IS FOUND IN A PDS (FTP COMMANDS ASSUME PDS) *
//** *ASSUMPTION IS THAT THE .EXE FILENAME IS SAME AS .BAT OR *
//** OTHER MAKE TYPE ... (I.E. CLOUD9.BAT CREATES CLOUD9.EXE) *
//**
//***** GENERAL NOTES *****
//**
//** THE PROCESSOR WAS BUILT TO BE USED BY PROCESSOR GROUPS FOR THE *
//** TYPE MAKE. IT CAN BE USED FOR TYPES THAT REQUIRE AN FTP PROCESS*
//** TO SEND THE FILE TO ANOTHER PLATFORM AND THEN PERFORM A MAKE *
//** PROCESS ON THE REMOTE MACHINE(S). THE REXX PROGRAM CHECKS FOR *
//** A TYPE OF MAKE BEFORE ISSUING THE FTP COMMANDS TO EXECUTE THE *
//** MAKE COMMANDS ON THE REMOTE MACHINE *
//**
//** THIS PROCESSOR USES/CALLS... *
//** C9LSLR - CIG UTILITY *
//** DETERMINE LONG NAME (PCFILE NAME) WRITING INFO TO CIGPUNCH DD *
//**
//** NDVRMAKE - REXX PROGRAM *
//** 1) CAPTURE THE LONG NAME OF ELEMENT *
//** 2) FTP FILE TO LONG NAME AT DESIRED LOCATION *
//** 3) PERFORM MAKE ON REMOTE MACHINE *
//** 4) RETURN OUTPUTS (SUCH AS .EXE FILE *
//** 5) STORE OUTPUTS (SUCH AS .EXE FILE) IN PROCESS OUTPUT LIB(S) *
//** 6) IF SUCCESSFUL, MSGS FROM FTP ARE TRIMMED TO MINIMAL MSGS AND *
//** STORED IN LISTLIB *
//** 7) IF FTP FAILS, THE FULL FTPOUT WILL BE SENT TO PRINT *
//*****
//** INITIALIZE TEMP DATASETS
//*****
//INITDSN EXEC PGM=BC1PDSIN
//C1INIT01 DD DSN=&&SLRDATA,DISP=(,PASS,DELETE),UNIT=&WRKUNIT,
//          SPACE=(TRK,(1,1),RLSE),LRECL=80,RECFM=FB,BLKSIZE=6400
//C1INIT02 DD DSN=&&FTPIN,DISP=(,PASS,DELETE),UNIT=&WRKUNIT,
//          SPACE=(TRK,(1,1),RLSE),LRECL=80,RECFM=FB,BLKSIZE=6400
//C1INIT03 DD DSN=&&FTPOUT,DISP=(,PASS,DELETE),UNIT=&WRKUNIT,
//          SPACE=(TRK,(1,1),RLSE),LRECL=80,RECFM=FB,BLKSIZE=6400
//C1INIT04 DD DSN=&&FTPMMSG,DISP=(,PASS,DELETE),UNIT=&WRKUNIT,
//          SPACE=(TRK,(1,1),RLSE),LRECL=133,RECFM=FB,BLKSIZE=2660
//C1INIT05 DD DSN=&&MAKELOG,DISP=(,PASS,DELETE),UNIT=&WRKUNIT,
```

```

//          SPACE=(TRK,(1,1),RLSE),LRECL=133,RECFM=FB,BLKSIZE=2660
//*****
//*        CREATE A LOG FILE THAT CAN BE USED DURING MAKE PROCESS
//*        NDVRMAKE REXX IS EXPECTING THIS DATASET TO BE CREATED ALREADY
//*        IF YOU DETERMINE YOU WANT A PERM WORKFILE OUT THERE AND CHANGE
//*        THE DSN OF THE LOG BE SURE TO ALSO CHANGE IT IN THE REXX NDVRMAKE
//*****
//LOGFILE EXEC PGM=IEFBR14
//C1INIT05 DD DSN=&C1USERID..&C1SY..&C1ST..&C1EL..LOG,
//          UNIT=&WRKUNIT,SPACE=(TRK,(1,1,1),RLSE),DISP=(,CATLG,DELETE),
//          LRECL=133,BLKSIZE=0,RECFM=VB,DSORG=PO,UNIT=&WRKUNIT
//*****
//*        GET THE LONGNAME FROM SOLEI'S SLR
//*****
//GETSLR EXEC PGM=C9LSLR
//STEPLIB DD DSN=FLHQ1.FLHQ2.LOADLIB,DISP=SHR
//CIGPUNCH DD DSN=&&SLRDATA,DISP=(OLD,PASS,DELETE)
//CIGLOG DD SYSOUT=*
//CIGIN DD *
LIST LONGNAME WHERE SHORTNAME = &C1ELEMENT .
/*
//FTPIF IF (GETSLR.RC < 4) THEN
//*****
//*        REXX TO CREATE FTP COMMANDS, CALL FTP AND THEN FORMAT MSGS...
//*****
//FTPSTEP EXEC PGM=IRXJCL,
// PARM='NDVRMAKE &C1EL &C1USERID &C1EN &C1ST &C1TY'
//SYSEXEC DD DSN=FLHQ1.FLHQ2.ISRCLIB,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD DUMMY
//FTPIN DD DSN=&&SLRDATA,DISP=(OLD,DELETE,DELETE)
//INPUT DD DSN=&&FTPIN,DISP=(OLD,DELETE,DELETE)
//OUTPUT DD DSN=&&FTPOUT,DISP=(OLD,DELETE,DELETE)
//MAKELOG DD DSN=&&MAKELOG,DISP=(OLD,DELETE,DELETE)
//*
//MSGFILE DD DSN=&&FTPMSG,DISP=(OLD,PASS,DELETE)
//*****
//*        DELETE THE LOG FILE THAT WAS USED DURING MAKE PROCESS
//*****
//LOGFILE EXEC PGM=IEFBR14
//C1INIT05 DD DSN=&C1USERID..&C1SY..&C1ST..&C1EL..LOG,
//          UNIT=&WRKUNIT,DISP=(OLD,DELETE)
//*****
//*        PRINT OUT LISTING IF FTP/MAKE HAS ERRORS
//*****
//IFTPERR (FTPSTEP.RC > 4) THEN
//LISTPRNT EXEC PGM=CONLIST,PARM='PRINT'
//C1BANNER DD DSN=&&BANNER,DISP=(,PASS,DELETE),
//          UNIT=&WRKUNIT,SPACE=(TRK,(1,1)),
//          DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6171,DSORG=PS)
//C1PRINT DD SYSOUT=*,
//          DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6171,DSORG=PS)
//LIST01 DD DSN=&&FTPMSG,DISP=(OLD,DELETE)
//*****
//IFTPERR ELSE
//*        STORE THE LISTING IF FTP/MAKE WAS SUCCESSFUL
//*****
//STEPNAME EXEC PGM=CONLIST,PARM='STORE'
//C1BANNER DD DISP=(,PASS,DELETE),
//          UNIT=&WRKUNIT,SPACE=(TRK,(1,1)),
//          DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6171,DSORG=PS)
//C1LLIBO DD DSN=&LISTLIB,DISP=SHR,FOOTPRNT=CREATE
//LIST01 DD DSN=&&FTPMSG,DISP=(OLD,DELETE)
//IFTPERR ENDIF
//FTPIF ENDIF

```

Usage Notes:

The purpose of this processor is twofold. You can deploy and optionally build objects at FTP target sites. For instance, the C++ source code may be stored in Endeavor, but needs to be built in a secured QA area on an NT box. This processor would be used to ship the source and MAKE file, and then request that the command file be executed at the FTP target. The script writer should ensure that the make or command file redirects the command output to file that is sent back to z/OS FTP session.

Review and modify the NDVRFTP1 REXX Script

The following REXX script can be found in the Cloud 9 product CGI library delivered. This script is called by the EPRCSFTP processor.



Issue a 'CAPS OFF' command prior to editing this member!

1. Using ISPF EDIT, access member NDVRFTP1 in the ISRCLIB library.
2. Issue a 'CAPS OFF' command on the command line of the edit session to ensure case sensitivity.
3. Substitute your site-specific values as identified on the Installation Worksheet. for those values shown below in bold.

```
/* rexx */
/* ----- */
/* NDVRFTP1 - USED WITH EPRCSFTP PROCESSOR */
/* ----- */
/* This is a rexx program that invokes FTP */
/* to ship endeavor inventory to specified */
/* locations out in the network. */
/* ----- */
/* This is prototype and must be customized */
/* by the user. */
/* ----- */
/*
/* TRACE ALL */
/* member=shortname (&clcl) */
```

```

/* userid=&cluserid    */
/* system=&clsy        */
/* stage=&clst         */
/* type=&clty          */

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

call GetParms
call GetPCFileName
call GetTranslationType

select
  when (stage == 'QA') then do

    /* ftp -> demo machine */
    ipaddr='ip.address1.here'; port=port1
    user='user01'; password='pswd01'; dir='c:\pcdir'
    call InvokeFTP

    /* ftp -> unix system services */
    ipaddr='ip.address2.here'; port=port2
    user='user02'; password='PSWD02'; dir='/u/ussdir'
    call InvokeFTP

    /* ftp -> web site.com */
    ipaddr='ip.address3.here'; port=port3
    user='user03'; password='pswd03'; dir='/nethere/dir'
    call InvokeFTP

    /* ftp -> copy to diskette/A Drive*/
    ipaddr='ip.address1.here'; port=port1
    user='user01'; password='pswd01'; dir='a:\demo'
    call InvokeFTP
  end
  when (stage == 'PROD') then do

    /* ftp -> demo machine */
    ipaddr='ip.address1.here'; port=port1
    user='user01'; password='pswd01'; dir='c:\pcdir'
    call InvokeFTP

    /* ftp -> web site.com */
    ipaddr='ip.address3.here'; port=port3
    user='user03'; password='pswd03'; dir='/nethere/dir'
    call InvokeFTP

    /* ftp -> copy to diskette/A Drive*/
    ipaddr='ip.address1.here'; port=port1
    user='user01'; password='pswd01'; dir='a:\demo'
    call InvokeFTP
  end
  otherwise
end /* select */
return

/* ----- */
/* Get parms */
/* ----- */
GetParms:
parse var request member userid system stage type .
mbrdsn = system||'|'.'||stage||'|'.'||type
/* DSN=&C1SY.&C1ST.&C1ty */
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',' ')
return

/* ----- */
/* Get translation type */
/* GIF = BINARY */
/* HTM = ASCII */
/* HTML = ASCII */
/* JPG = BINARY */
/* ----- */
GetTranslationType:
  fileExtension = substr(pcFileName,lastpos('.',pcFileName)+1)
  fileExtension = translate(fileExtension) /* upper case */
  Select
    when (fileExtension == 'GIF') then
      translationType = 'BINARY'
    when (fileExtension == 'JPG') then
      translationType = 'BINARY'
    when (fileExtension == 'DOC') then
      translationType = 'BINARY'
    when (fileExtension == 'PPT') then
      translationType = 'BINARY'
    when (fileExtension == 'XLS') then
      translationType = 'BINARY'
    otherwise
      translationType = 'ASCII'
  end /* select */
return

/* ----- */
/* 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 "mbrdsn""
  ftpIdx=ftpIdx+1; ftp.ftpIdx="cd "dir
  ftpIdx=ftpIdx+1; ftp.ftpIdx=translationType
  ftpIdx=ftpIdx+1; ftp.ftpIdx="put "member" "pcFileName
  ftpIdx=ftpIdx+1; ftp.ftpIdx="quit"
  ftp.0 = ftpIdx

/* ----- */
/* The following statements will write the formatted FTP */
/* commands to the FTP input file and then invoke FTP. */
/* ----- */

  address MVS "EXECIO * DISKW INPUT (STEM ftp. FINIS"
  fx=fx+1; fmsg.fx='*** FTP STARTED FOR 'ipaddr' port 'port' ***'
  address attach FTP /* here we invoke FTP */

/* ----- */
/* The following statements will echo the FTP output file */
/* called 'OUTPUT' to a rexx stem for processing. */
/* ----- */

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

/* ----- */
/* The following statements will dump the ftp output results */
/* to the dataset specified. */
/* ----- */

```

```
"alloc dd(CONFILE) mod reuse da('some.logdsn.here')"  
address MVS "EXECIO * DISKW CONFILE (STEM input. FINIS"  
"free dd(CONFILE)"  
  
return
```

NDVRFTP1 REXX Script

Review and modify the NDVRMAKE REXX Script

The following REXX script can be found in the Cloud 9 product CGI library. This script is called by the EPRCSMAK processor.



Issue a 'CAPS OFF' command prior to editing this member!

1. Using ISPF EDIT, access member NDVRFTP1 in the ISRCLIB library.
2. Issue a 'CAPS OFF' command on the command line of the edit session to ensure case sensitivity.
3. Substitute your site-specific values (identified on the Installation Worksheet.) for those values shown below in bold.

```
/* rexx */  
/* ----- */  
/* NDVRMAKE - USED WITH EPRCSMAK PROCESSOR */  
/* ----- */  
/* This is a rexx program that invokes FTP */  
/* to ship endeavor inventory to specified */  
/* locations out in the network. It also */  
/* ships a make command and command exec */  
/* request. */  
/* ----- */  
/* This is prototype and must be customized */  
/* by the user. */  
/* ----- */  
/* TRACE ALL */  
/* member=shortname (&clsl) */  
/* userid=&cluserid */  
/* system=&clsy */  
/* stage=&clst */  
/* type=&clty */  
  
parse arg request  
fx = 0
```

```

true = 1
false = 0
ftpIdx=0

call GetParms
call GetPCFileName
call GetTranslationType

/* PROCESS IF stage=QA */
select
  when (stage == 'QA') then do

    /* ftp -> demo machine */
    ipaddr='ip.address1.here'; port=port1
    user='user01'; password='pswd01'; dir='c:\pcdir'
    call InvokeFTP

    /* ftp -> unix system services */
    ipaddr='ip.address2.here'; port=port2
    user='user02'; password='PSWD02'; dir='/u/ussdir'
    call InvokeFTP

    /* ftp -> web site.com */
    ipaddr='ip.address3.here'; port=port3
    user='user03'; password='pswd03'; dir='/nethere/dir'
    call InvokeFTP

    /* ftp -> copy to diskette/A Drive*/
    ipaddr='ip.address1.here'; port=port1
    user='user01'; password='pswd01'; dir='a:\demo'
    call InvokeFTP
  end
  when (stage == 'PROD') then do

    /* ftp -> demo machine */
    ipaddr='ip.address1.here'; port=port1
    user='user01'; password='pswd01'; dir='c:\pcdir'
    call InvokeFTP

    /* ftp -> web site.com */
    ipaddr='ip.address3.here'; port=port3
    user='user03'; password='pswd03'; dir='/nethere/dir'
    call InvokeFTP

    /* ftp -> copy to diskette/A Drive*/
    ipaddr='ip.address1.here'; port=port1
    user='user01'; password='pswd01'; dir='a:\demo'
    call InvokeFTP
  end
  otherwise
end /* select */
return

/* ----- */
/* Get parms and build mainframe dataset names needed for ftp/make */
/* this section may need to change based on your sites' DSN naming */
/* standards and your process */
/* ----- */
GetParms:
  /* parsing &C1EL &C1USERID &C1SY &C1ST &C1TY */
  parse var request member userid system stage type .

  /* build mainframe dataset names : */
  /* mbrdsn=pds where current source is */
  /* mbrexe=pds where exe source will be stored (processor o/p lib */
  /* mbrlog=dataset where log.txt info is stored temporarily */
  mbrdsn = system||'.'||stage||'.'||type /* sol, basedsn or tempdsn */
  mbrexe = system||'.'||stage||'.EXE'
  mbrlog = userid||'.'||system||'.'||stage||'.'||member||'.LOG'

```

```

return

/* ----- */
/*   Get longname from //FTPIN (passed from slr utility)   */
/* ----- */
GetPCFileName:
  address MVS "EXECIO * DISKR FTPIN (STEM input. FINIS"
  i = input.0 - 1
  pcFileName = input.i
  pcFileName = strip(pcFileName,'B',' ')
  if type = 'MAKE' then
  do
    wheredot = lastpos('.',pcFileName)
    pcFileNameexe = substr(pcFileName,1,wheredot)||'exe'
  end
return

/* ----- */
/*   Get translation type   */
/*   GIF   = BINARY   */
/*   HTM   = ASCII   */
/*   HTML  = ASCII   */
/*   JPG   = BINARY   */
/* ----- */
GetTranslationType:
  fileExtension = substr(pcFileName,lastpos('.',pcFileName)+1)
  fileExtension = translate(fileExtension) /* upper case */
  Select
    when (fileExtension == 'GIF') then
      translationType = 'BINARY'
    when (fileExtension == 'JPG') then
      translationType = 'BINARY'
    when (fileExtension == 'DOC') then
      translationType = 'BINARY'
    when (fileExtension == 'PPT') then
      translationType = 'BINARY'
    when (fileExtension == 'XLS') then
      translationType = 'BINARY'
    otherwise
      translationType = 'ASCII'
  end /* select */
return

/* ----- */
/*   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 "mbrdsn""
  ftpIdx=ftpIdx+1; ftp.ftpIdx="cd "dir

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

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

  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 'mbrexex'"
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 "pcFileNameexe member" mbrexex "replace"

ftpIdx=ftpIdx+1; ftp.ftpIdx="lcd 'mbrlog'"
ftpIdx=ftpIdx+1; ftp.ftpIdx="ASCII"
ftpIdx=ftpIdx+1
ftp.ftpIdx="get "log.txt member "replace"

end

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

/* ----- */
/* The following statements will write the formatted FTP */
/* commands to the FTP input file and then invoke FTP.  */
/* ----- */

address MVS "EXECIO * DISKW INPUT (STEM ftp. FINIS"
fx=fx+1; fmsg.fx='*** FTP STARTED FOR 'ipaddr' port 'port' ***'
address attach FTP /* here we invoke FTP */

/* ----- */
/* The following statements will echo the FTP output file */
/* called 'OUTPUT' to a rexx stem for processing.        */
/* ----- */

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

/* ----- */
/* The following statements will dump the ftp output results */
/* to the dataset specified.                                  */
/* ----- */

"alloc dd(CONFILE) mod reuse da('some.logdsn.here')"
address MVS "EXECIO * DISKW CONFILE (STEM input. FINIS"
"free dd(CONFILE)"

return

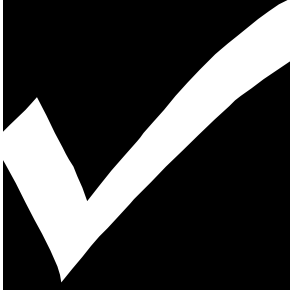
```

NDVRMAKE REXX Script Prototype

Step 4: Add the Processors to Endeavor

The purpose of this step is to remind you to add your processors to Endeavor and attach to the appropriate Endeavor Stage/Types.





CHECKPOINT #2

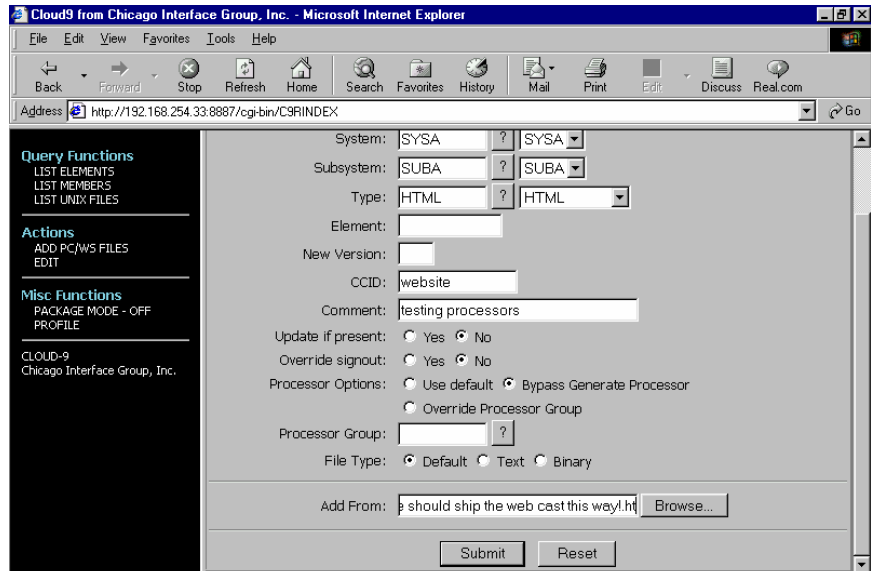
At this point, you should have completed the following tasks:

Task	Completed?
Reviewed and modified processors EPRCSFTP and EPRCSMAK?	
Reviewed and modified REXX script NDVRFTP1 and NDVRMAKE?	
Add processors to Endeavor and assigned to inventory?	

*Checkpoint
t 2*

Step 5. Test the E-FTP Processor

1. Add a piece of source defined to the type and language supported. For example, add a type called HTML. The Cloud 9 screen shot below shows the process of adding in an HTML file into with a language of FTP1.



Example of adding a HTML type

2. Through Cloud 9, request a Generate of the source to invoke the EPRCSFTP processor. View the batch job output. Verify that the object was FTP'd to the desired location.
3. View the 'userid.ftplog' dataset updated in the NDVRFPT1 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 999.999.999.99 21
EZA1554I Connecting to: 999.999.999.99 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
```

User FTP log example

Chapter 2: Managing Java with Cloud 9 and USS

Chapter Scope

This chapter contains the Cloud 9 E-JDK implementation. The steps in this manual are organized into four major sections:

- Before you begin
- Customizing Processors and Processor Control Files
- Defining all E-JDK inventory, USS and Cloud 9 pieces
- Perform Installation Verification Procedures (IVP)

These steps should be followed in the order that they are presented. Once you have successfully completed all of the steps and executed the test procedure in Step 8, you will be ready to use the E-JDK.

Modification of Case Sensitive E-JDK Values



During this installation, you will modify several JCL members and Unix files. Some of these files contain case-sensitive values. It is imperative that prior to modifying the JCL and Unix members, you issue the CAPS OFF command to ensure that automatic upper casing of the Unix members does not occur. For your convenience, the following icon will be placed in each step where case-sensitive Unix values are an issue.

Product Components that require modification

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

JCL Members (located in JCL) Modified During Implementation:

C9JPJADD	JCL to add processors to Endeavor
C9JPJEE	JCL to load EE control records
C9JPJINV	JCL to define E-JDK Inventory locations
C9JPJSLR	JCL to define E-JDK types to Cloud 9
C9JPJUNX	JCL to define E-JDK Unix Directories
C9JPPDEL	Delete Processor for HTML and GRAPHIC Types
C9JPPGEN	Generate Processor for HTML and GRAPHIC Types
C9JPPJAV	Generate Processor for JAVA and JAR Types
C9JPPJDL	Delete Processor for JAVA and JAR Types
C9JPPJMV	Move Processor for JAVA and JAR Types

Parameters (located in HTML library) Modified During Implementation:

JCOMPILE	JAVA compile shell. Review for path.
JCOMPILR	JAR compile shell. Review for path.
UNIXMAP	JAVA USS Output Mapping. Review defaults.
UNIXMAPR	JAR USS Output Mapping. Review defaults.
JCPATH	Parameter substitution for %Classpath%
UNIXLOC	Endevor to USS location mapping.

A Step-by-Step Approach

BEFORE YOU BEGIN...	
◆	Review system, software, and hardware considerations.
◆	Implement site standards.
Review Default Endeavor Inventory Locations and USS locations	
1.	Review default E-JDK values and determine actual inventory to be
2.	Review and modify all processors and processor control file
Perform Endeavor E-JDK Inventory Definitions	
3.	Modify and run C9JPJINV to build E-JDK System, Types, and
4.	Modify and run C9JPJADD to add E-JDK Processors into Endeavor
5.	Modify and run C9JPJEE to load the E-JDK Records for Cloud 9
6.	Modify and run C9JPJSLR to define E-JDK types to Cloud 9
7.	Modify and run C9JPJUNX to define USS directories
Test Pilot using CLOCKJ and CLOCKH application	
8.	Add CLOCKJ Java Type and CLOCKH HTML Type
9.	Invoke clockh.html to display Time of Day – all done!

Enabling the E-JDK

Review Software and Hardware Considerations

In this step you will review the system and software requirements for enabling the E-JDK.

System Requirements

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

z/OS operating system	Version 1.1 or higher
CA-Endevor	Version 3.9 or higher

Before You Begin: Implement Site Standards

Site-Specific Placeholders

The following placeholders represent values that are customer-specific. These are not default values, rather generics that must be customized to meet your site specific values. Although there are many more generics included in the full install of Cloud-9, these are the only ones used in the E-JDK processors and JCL. Please review these generics and be prepared to substitute values for the worksheet on the following page.

<i>Flhq1 and Flhq2</i>	<i>High and 2nd level qualifier for all CIG product datasets.</i>
<i>Qual1 and Qual2</i>	<i>High and 2nd level qualifier used as reference for all non CIG datasets, such as the Endeavor datasets or user files.</i>
<i>Tdisk</i>	<i>Temporary unit value used for allocation of temp files. For example SYSDA.</i>

Endeavor Dataset Names

Additionally, identify the dataset names for your current Endeavor set up as per the Dataset worksheet on the following page. These dataset names will be needed for batch JCL updates.

Remove this worksheet from the manual for easy reference during later installation steps.

Placeholder Worksheet

Place Holder	Definition	Your Site Value
<i>Tdisk</i>	Unit label for temporary disk data sets (For example, SYSDA).	
<i>Flhq1</i>	High-level qualifier for the CIG Product datasets.	
<i>Flhq2</i>	Second-level qualifier for the CIG Product datasets.	
<i>Qual1</i>	High-level qualifier for Endeavor datasets.	
<i>Qual2</i>	Second-level qualifier for Endeavor datasets.	

Site-specific Customization

Dataset Worksheet

DDNAME	Dataset / File Names Examples	Your Dataset / File Names
<i>CA-Endevor LOADLIB</i>	SYS3.ENDEVOR.AUTHLIB	
<i>CA-Endevor CONLIB</i>	SYS3.ENDEVOR.CONLIB	

Endevor Dataset Names

Step 1. Determine CA-Endevor and USS Inventory Values

The E-JDK is set up with default values. These default values are meant to be modified throughout the various processor, HTML and JCL members. Prior to making modifications to these members, please review the worksheets below and fill in the site specific Inventory Values.

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

CA-Endevor Inventory Value Worksheet

Endevor Inventory Name	Default Value	Your Value
Environment	TEST	
System	SYSJ	
Subsystem	SUBJ	
Types	JAVA,JAR,HTML,GRAPHICS	
Stage ID's	A and B	
Stage 1 Processor Load Library	QUAL1.QUAL2.EPLOD1	
Stage 2 Processor Load Library	QUAL1.QUAL2.EPLOD2	
Stage 1 Processor Listing Library	QUAL1.QUAL2.EPLST1	
Stage 2 Processor Listing Library	QUAL1.QUAL2.EPLST2	
Classes Listing Library Name in Processors	QUAL1.QUAL2.ES&C1EN&C1S#..LISTINGS	
Classes EXEC Library Name in	QUAL1.QUAL2.ES&C1EN&C1S#..CLASSES	

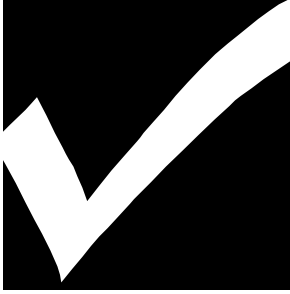
Processors		
Base Library Name In Batch Admin JCL C9JPJINV	QUAL1.QUAL2.EBASE1	
Delta Library Name in Batch Admin	QUAL1.QUAL2.EDELTA1	

CA-Endevor Inventory Value Worksheet

USS Directory Value Worksheet

USS Mapping Locations	Default Values	Your Values
Listings	/u/test/a/sysj/subj/listings /u/test/b/sysj/subj/listings	
Classes	/u/test/a/sysj/subj/classes /u/test/b/sysj/subj/classes	
Graphics	/u/test/a/sysj/subj/graphics /u/test/b/sysj/subj/graphics	
Html	/u/test/a/sysj/subj/html /u/test/b/sysj/subj/html	
Jar	/u/test/a/sysj/subj/jar /u/test/b/sysj/subj/jar	

USS Directory Value Worksheet



CHECKPOINT #1

At this point the following tasks should have been completed.

Tasks	Completed?
Review all CA-Endevor Inventory Default Values	
Review all USS Directory Default Values	
Determine actual CA-Endevor Inventory and USS Directory Values	

Checkpoint 1

Step 2. Review and modify all Processors and Control Files

E-JDK Processors:

C9JPPDEL	Delete Processor for HTML and GRAPHIC Types
C9JPPGEN	Generate Processor for HTML and GRAPHIC Types
C9JPPJAV	Generate Processor for JAVA and JAR Types
C9JPPJDL	Delete Processor for JAVA and JAR Types
C9JPPJMV	Move Processor for JAVA and JAR Types

E-JDK Processor Control Files:

JCOMPILR	Potential Input to C9JPPJAV Processor. Compile shell for JAR type
UNIXMAPR	Potential Input to C9JPPJAV Processor USS Output Control for JAR type
JCOMPILE	Default Input to C9JPPJAV Processor Compile Shell for JAVA type
UNIXMAP	Default Input to C9JPPJAV Processor USS Output Control for JAVA type
JCPATH	Default Input to JAVA and JAR Processors %classpath% Substitution.
UNIXLOC	Default Input to all E-JDK Processors Endevor to USS mapping rules.

Review and Modify C9JPPDEL Processor

The following sample processor can be found in the JCLLIB offloaded from the product cartridge or via email. All “review and modify” values will be highlighted in **BOLD** for easy reading.

```
//* -----*
//* CLOUD 9 JAVA/E-JDK COMPONENTS (V7.0) *
//* -----*
//* NAME: C9JPPDEL *
//* PURPOSE: DELETE PROCESSOR FOR HTML AND GRAPHICS. *
//* -----*
//* TO USE THIS PROCESSOR YOU MUST: *
//* 1) MAKE SURE THAT THE STEPLIB POINTS TO THE CIG PRODUCT *
//* AUTHORIZED DATASET AND INCLUDES THE DATASET THAT *
//* CONTAINS CIGINI AND CIGFEXEC. *
//* 2) CHANGE FLHQ1 AND FLHQ2 AS PER YOUR *
//* INSTALLATION SHEET *
//* 3) CHANGE THE UNIT=TDISK TO THE APPROPRIATE UNIT *
//* NAME. *
//* 4) REVIEW THE DATASET USED IN THE UNIXLOC AND SYSEXEC *
//* DDNAMES AND THE DATASET USED IN THE EXEC STATEMENT *
//* AS INPUT TO THE //SYSTSIN DDNAME. *
//* THESE FILES DEFAULT TO THE PRODUCT LIBRARIES. *
//* MODIFY TO MEET SITE STANDARDS. *
//* -----*
//* C9JPPDEL (DELETE PROCESSOR) FOR HTML AND GRAPHICS *
//* -----*
//C9JPPDEL PROC TBJ=TEXT,
// UNIXLOC=UNIXLOC
//* -----*
//CONWRIT1 EXEC PGM=CONWRITE, PARM='EXPINCL(N)'
//ELMOUT DD DSN=&&SYSIN(&C1ELEMENT), DISP=(NEW,PASS,DELETE),
// UNIT=TDISK, SPACE=(TRK,(10,10,10),RLSE),
// DCB=(LRECL=512,BLKSIZE=26004,RECFM=VB)
//* -----*
//IKJEFT01 EXEC PGM=IKJEFT01
//STEPLIB DD DSN=FLHQ1.FLHQ2.LOADLIB, DISP=SHR
//SYSTSIN DD *
EXEC 'FLHQ1.FLHQ2.CGI(C9JPPDEL)' '&TBJ &C1EN &C1SI &C1SY &C1SU +
&C1TY &C1ELEMENT'
//SYSEXEC DD DSN=FLHQ1.FLHQ2.CGI, DISP=SHR
//SYSTSPRT DD SYSOUT=*
//JFILE DD DSN=&&SYSIN(&C1ELEMENT), DISP=(OLD,DELETE)
//UNIXLOC DD DSN=FLHQ1.FLHQ2.HTML(&UNIXLOC), DISP=SHR
//* -----*
```

C9JPPDEL Graphic and HTML Delete Processor

Review and Modify C9JPPGEN Processor

The following sample processor can be found in the JCLLIB offloaded from the product cartridge or via email. All “review and modify” values will be highlighted in **BOLD** for easy reading.

```
//* -----*
//* CLOUD 9 JAVA/E-JDK COMPONENTS (V7.0) *
//* -----*
//* NAME: C9JPPGEN *
//* PURPOSE: GENERATE PROCESSOR FOR HTML AND GRAPHICS. *
//* -----*
//* TO USE THIS PROCESSOR YOU MUST: *
//* 1) MAKE SURE THAT THE STEPLIB POINTS TO THE CIG PRODUCT *
//* AUTHORIZED DATASET AND INCLUDES THE DATASET THAT *
//* CONTAINS CIGINI AND CIGFEEXEC. *
//* 2) CHANGE FLHQ1 AND FLHQ2 AS PER YOUR *
//* INSTALLATION SHEET *
//* 3) CHANGE THE UNIT=TDISK TO THE APPROPRIATE UNIT *
//* NAME. *
//* 4) REVIEW THE DATASET USED IN THE UNIXLOC AND SYSEXEC *
//* DDNAMES AND THE DATASET USED IN THE EXEC STATEMENT *
//* AS INPUT TO THE //SYSTSIN DDNAME. *
//* THESE FILES DEFAULT TO THE PRODUCT LIBRARIES. *
//* MODIFY TO MEET SITE STANDARDS. *
//* -----*
//* C9JPPGEN (GENERATE HTML AND GRAPHICS) *
//* -----*
//C9JPPGEN PROC TBJ=TEXT,
// UNIXLOC=UNIXLOC
//* -----*
// WRITE ELEMENT TO TEMPORARY FILE
//* -----*
//CONWRITE EXEC PGM=CONWRITE, PARM='EXPINCL(N)'
//ELMOUT DD DSN=&&SYSIN(&C1ELEMENT), DISP=(NEW,PASS,DELETE),
// UNIT=TDISK, SPACE=(TRK,(10,10,10),RLSE),
// DCB=(LRECL=512,BLKSIZE=26004,RECFM=VB)
//* -----*
// COPY ELEMENT TO UNIX LOCATION
//* -----*
//IKJEFT01 EXEC PGM=IKJEFT01
//STEPLIB DD DSN=FLHQ1.FLHQ2.LOADLIB, DISP=SHR
//SYSTSIN DD *
EXEC 'FLHQ1.FLHQ2.CGI(C9JPRGEN)' '&TBJ &C1EN &C1SI &C1SY &C1SU +
&C1TY &C1ELEMENT &C1SENVMT &C1SSTGID &C1SSYSTEM &C1SSUBSYS'
//SYSEXEC DD DSN=FLHQ1.FLHQ2.CGI, DISP=SHR
//SYSTSPRT DD SYSOUT=*
//JFILE DD DSN=&&SYSIN(&C1ELEMENT), DISP=(OLD,DELETE)
//UNIXLOC DD DSN=FLHQ1.FLHQ2.HTML(&UNIXLOC), DISP=SHR
//* -----*
```

C9JPJGEN Graphic and HTML Processor

Review and Modify C9JPPJAV Processor

The following sample processor can be found in the JCLLIB offloaded from the product cartridge or via email. All “review and modify” values will be highlighted in BOLD for easy reading.

```
//* -----*
//* CLOUD 9 JAVA/E-JDK COMPONENTS (V7.0) *
//* -----*
//* NAME: C9JPPJAV *
//* PURPOSE: GENERATE PROCESSOR FOR JAVA AND JAR TYPES. *
//* -----*
//* TO USE THIS PROCESSOR YOU MUST: *
//* 1) MAKE SURE THAT THE STEPLIB POINTS TO THE CIG PRODUCT *
//* AUTHORIZED DATASET AND INCLUDES THE DATASET THAT *
//* CONTAINS CIGINI AND CIGFEXEC. *
//* 2) CHANGE FLHQ1 AND FLHQ2 AS PER YOUR *
//* INSTALLATION SHEET *
//* 3) CHANGE QUAL1 AND QUAL2 AS PER YOUR *
//* INSTALLATION SHEET *
//* 4) CHANGE THE UNIT=TDISK TO THE APPROPRIATE UNIT *
//* NAME. *
//* 5) REVIEW THE DATASETS USED IN THE UNIXMAP, JCPATH, AND *
//* JCOMPILE DDNAMES AND THE DATASET USED IN THE EXEC *
//* STATEMENT AS INPUT TO THE //SYSTSIN DDNAME. *
//* THESE FILES DEFAULT TO THE PRODUCT LIBRARIES. *
//* MODIFY TO MEET SITE STANDARDS. *
//* 6) REVIEW THE LISTING AND CLASSES OUTPUT LIBRARIES *
//* NAMING STANDARDS, MODIFY IF REQUIRED. *
//* (JLIST AND JEXEC DDNAMES) *
//* -----*
//* C9JPPJAV GENERATE PROCESSOR FOR JAVA AND JAR COMPILE. *
//* -----*
//C9JPPJAV PROC JPGM=C9JPRJAV,
// UNIXMAP=UNIXMAP,
// JCPATH=JCPATH,
// JCOMPILE=JCOMPILE,
// UNIXLOC=UNIXLOC
//* -----*
//CONWRITE EXEC PGM=CONWRITE, PARM='EXPINCL(N)'
//ELMOUT DD DSN=&&SYSIN(&C1ELEMENT), DISP=(NEW, PASS, DELETE),
// UNIT=TDISK, SPACE=(TRK, (10, 10, 10)),
// DCB=(LRECL=512, BLKSIZE=26004, RECFM=VB)
//* -----*
//IKJEFT01 EXEC PGM=IKJEFT01
//STEPLIB DD DSN=FLHQ1.FLHQ2.LOADLIB, DISP=SHR
//SYSTSIN DD *
EXEC 'FLHQ1.FLHQ2.CGI(&JPGM)' '&C1EN &C1SI &C1SY &C1SU &C1TY +
&C1ELEMENT'
//SYSTSPRT DD SYSOUT=*
//SYSEXEC DD DSN=FLHQ1.FLHQ2.CGI, DISP=SHR
//JFILE DD DSN=&&SYSIN(&C1ELEMENT), DISP=(OLD, PASS)
//UNIXMAP DD DSN=FLHQ1.FLHQ2.HTML(&UNIXMAP), DISP=SHR
//UNIXLOC DD DSN=FLHQ1.FLHQ2.HTML(&UNIXLOC), DISP=SHR
//JCPATH DD DSN=FLHQ1.FLHQ2.HTML(&JCPATH), DISP=SHR
//JCOMPILE DD DSN=FLHQ1.FLHQ2.HTML(&JCOMPILE), DISP=SHR
//*
//JLIST DD DSN=QUAL1.QUAL2.ES&C1EN&C1S#..LISTINGS,
// DISP=SHR, FOOTPRNT=CREATE, MONITOR=COMPONENTS
//JEXEC DD DSN=FLHQ1.FLHQ2.CLASSES, DISP=(NEW, CATLG, DELETE),
// SPACE=(CYL, (10, 10, 10)), UNIT=TDISK,
// DCB=(LRECL=256, BLKSIZE=27998, RECFM=VB)
//SELECTDD DD DSN=&&SELDD, DISP=(NEW, PASS),
// SPACE=(TRK, (10, 10)), UNIT=TDISK,
// DCB=(LRECL=80, BLKSIZE=3120, RECFM=FB)
//SYSPRINT DD SYSOUT=*
//* -----*
```

```

//*      COPY CLASSES TO TARGET
//* -----*
//CLASSES EXEC PGM=BSTCOPY
//SYSIN DD DSN=&&SELDD,DISP=(OLD,DELETE)
//SYSPRINT DD SYSOUT=*
//CLASSES DD DSN=FLHQ1.FLHQ2.CLASSES,DISP=(OLD,DELETE),
//JEXEC DD DISP=SHR,
// DSN=QUAL1.QUAL2.ES&C1EN&C1S#..CLASSES,
// MONITOR=COMPONENTS
//* -----*
//*      COPY ELEMENT TO UNIX LOCATION
//* -----*
//COPYBACK EXEC PGM=IKJEFT01
//STEPLIB DD DSN=FLHQ1.FLHQ2.LOADLIB,DISP=SHR
//SYSTSIN DD *
EXEC 'FLHQ1.FLHQ2.CGI(C9JPRGEN)' '&TBJ &C1EN &C1SI &C1SY &C1SU +
&C1TY &C1ELEMENT &C1SENVMT &C1SSTGID &C1SSYSTEM &C1SSUBSYS'
//SYSEXEC DD DSN=FLHQ1.FLHQ2.CGI,DISP=SHR
//SYSTSPRT DD SYSOUT=*
//JFILE DD DSN=&&SYSIN(&C1ELEMENT),DISP=(OLD,DELETE)
//UNIXLOC DD DSN=FLHQ1.FLHQ2.HTML(&UNIXLOC),DISP=SHR
//* -----*

```

C9JPPJAV Java/Jar Generate Processor

Review and Modify C9JPPJDL Processor

The following sample processor can be found in the JCLLIB offloaded from the product cartridge or via email. All “review and modify” values will be highlighted in **bold** for easy reading.

```

//* -----*
//* CLOUD 9 JAVA/E-JDK COMPONENTS (V7.0)
//* -----*
//*
//* NAME: C9JPPJDL
//* PURPOSE: DELETE PROCESSOR FOR JAVA AND JAR TYPES.
//* -----*
//* TO USE THIS PROCESSOR YOU MUST:
//* 1) MAKE SURE THAT THE STEPLIB POINTS TO THE CIG PRODUCT
//* AUTHORIZED DATASET AND INCLUDES THE DATASET THAT
//* CONTAINS CIGINI AND CIGFEXEC.
//* 2) CHANGE FLHQ1 AND FLHQ2 AS PER YOUR
//* INSTALLATION SHEET
//* 3) CHANGE THE UNIT=TDISK TO THE APPROPRIATE UNIT
//* NAME.
//* 4) REVIEW THE DATASET USED IN THE //UNIXLOC DDNAME
//* AND THE DATASET USED IN THE EXEC STATEMENT USED
//* AS INPUT TO THE //SYSTSIN DDNAME. BOTH OF
//* THESE FILES DEFAULT TO THE PRODUCT LIBRARIES.
//* MODIFY TO MEET SITE STANDARDS.
//* -----*
//* C9JPPJDL (DELETE PROCESSOR FOR JAVA AND JAR)
//* -----*
//C9JPPJDL PROC TBJ=TEXT,
// UNIXLOC=UNIXLOC
//* -----*
//* GET THE ELEMENT AND SOURCE COMPONENT LIST
//* -----*
//CONWRITE EXEC PGM=CONWRITE,PARM='EXPINCL(N)'
//ELMOUT DD DSN=&&SYSIN(&C1ELEMENT),DISP=(NEW,PASS,DELETE),
// UNIT=TDISK,SPACE=(TRK,(10,10,10),RLSE),

```

```

//      DCB=(LRECL=512,BLKSIZE=26004,RECFM=VB)
//* ----- *
//CONWRIT1 EXEC PGM=CONWRITE
//CONWIN  DD  *
        WRITE ELEMENT &C1ELEMENT
        FROM ENV &C1EN SYSTEM &C1SY SUBSYSTEM &C1SU
        TYPE &C1TY STAGE &C1SI
        TO DDN CMPLST1
        OPTION COMPONENT.
//CMPLST1 DD  DSN=&&CMPLST1,DISP=(NEW,PASS,DELETE),
//      UNIT=TDISK,SPACE=(TRK,(3,5),RLSE),
//      DCB=(LRECL=260,BLKSIZE=6160,RECFM=VB)
//* ----- *
//*      GENERATE COPY STATEMENTS FOR CLASSES AND LISTING
//* ----- *
//READCOMP EXEC PGM=IKJEFT01
//STEPLIB DD  DSN=FLHQ1.FLHQ2.LOADLIB,DISP=SHR
//SYSTSIN DD  *
EXEC 'FLHQ1.FLHQ2.CGI(C9JPRJCL)' 'JEXEC,JLIST'
//SYSEXEC DD  DSN=FLHQ1.FLHQ2.CGI,DISP=SHR
//SYSTSPRT DD  SYSOUT=*
//CMPLST1 DD  DSN=&&CMPLST1,DISP=(OLD,DELETE)
//JEXEC     DD  DUMMY
//JLIST     DD  DUMMY
//CMPOUT   DD  DSN=&&JAVACOMP,DISP=(NEW,PASS),
//          UNIT=TDISK,DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB)
//* ----- *
//IKJEFT01 EXEC PGM=IKJEFT01
//STEPLIB DD  DSN=FLHQ1.FLHQ2.LOADLIB,DISP=SHR
//SYSTSIN DD  *
EXEC 'FLHQ1.FLHQ2.CGI(C9JPRDEL)' '&TBJ &C1EN &C1SI &C1SY +
&C1SU &C1TY &C1ELEMENT'
//SYSEXEC DD  DSN=FLHQ1.FLHQ2.CGI,DISP=SHR
//SYSTSPRT DD  SYSOUT=*
//JCOMPS   DD  DSN=&&JAVACOMP,DISP=(OLD,DELETE)
//UNIXLOC  DD  DSN=FLHQ1.FLHQ2.HTML(&UNIXLOC),DISP=SHR
//* ----- *
//*      DELETE THE SOURCE OUTPUT COMPONENTS
//* ----- *
//DELLIST EXEC PGM=CONDELE,PARM='*COMPONENTS'

```

C9JPPJDL Java/Jar Delete Processor

Review and Modify C9JPPJMV Processor

The following sample processor can be found in the JCLLIB offloaded from the product cartridge or via email. All “review and modify” values will be highlighted in **bold** for easy reading.

```
//* -----*
//* CLOUD 9 JAVA/E-JDK COMPONENTS (V7.0) *
//* -----*
//* NAME: C9JPPJMV *
//* PURPOSE: MOVE PROCESSOR FOR JAVA AND JAR TYPES. *
//* -----*
//* TO USE THIS PROCESSOR YOU MUST: *
//* 1) MAKE SURE THAT THE STEPLIB POINTS TO THE CIG PRODUCT *
//* AUTHORIZED DATASET AND INCLUDES THE DATASET THAT *
//* CONTAINS CIGINI AND CIGFEXEC. *
//* 2) CHANGE FLHQ1 AND FLHQ2 AS PER YOUR *
//* INSTALLATION SHEET *
//* 3) CHANGE THE UNIT=TDISK TO THE APPROPRIATE UNIT *
//* NAME. *
//* 4) REVIEW THE DATASET USED IN THE //UNIXLOC DDNAME *
//* AND THE DATASET USED IN THE EXEC STATEMENT USED *
//* AS INPUT TO THE //SYSTSIN DDNAMES. *
//* THESE FILES DEFAULT TO THE PRODUCT LIBRARIES. *
//* MODIFY TO MEET SITE STANDARDS. *
//* 5) REVIEW THE LISTING AND CLASSES OUTPUT LIBRARIES *
//* NAMING STANDARDS, MODIFY IF REQUIRED. *
//* (SOURCE AND TARGET DDNAMES) *
//* 6) THE DEFAULT DDNAMES FOR THE CLASSES AND LISTING *
//* LIBRARIES ARE JEXEC AND JLIST. THESE NAMES ARE ALSO *
//* USED IN THE ENDEVOR TO UNIX MAPPING FILE C9JPCMAP. *
//* REVIEW THE C9JPCMAP MAPPING FILE TO ENSURE THAT THE *
//* CLASSES AND LISTING MAPPING TYPES MATCH THE DDNAMES *
//* IN THIS PROCESSOR. SEE DOCUMENTATION AND SAMPLES. *
//* -----*
//* C9JPPJMV (MOVE JAVA AND JARS) *
//* -----*
//C9JPPJMV PROC TBJ=TEXT,
// UNIXLOC=UNIXLOC
//* -----*
//* GET THE ELEMENT AND SOURCE COMPONENT LIST *
//* -----*
//CONWRITE EXEC PGM=CONWRITE, PARM='EXPINCL(N)'
//ELMOUT DD DSN=&&SYSIN(&C1ELEMENT), DISP=(NEW, PASS, DELETE),
// UNIT=TDISK, SPACE=(TRK, (10, 10, 10), RLSE),
// DCB=(LRECL=512, BLKSIZE=26004, RECFM=VB)
//* -----*
//CONWRIT1 EXEC PGM=CONWRITE
//CONWIN DD *
WRITE ELEMENT &C1ELEMENT
FROM ENV &C1SENVMT SYSTEM &C1SSYSTEM SUBSYSTEM &C1SSUBSYS
TYPE &C1SELTYPE STAGE &C1SSTGID
TO DDN CMLST1
OPTION COMPONENT.
//CMLST1 DD DSN=&&CMLST1, DISP=(NEW, PASS, DELETE),
// UNIT=TDISK, SPACE=(TRK, (3, 5), RLSE),
// DCB=(LRECL=260, BLKSIZE=6160, RECFM=VB)
//* -----*
//* GENERATE COPY STATEMENTS FOR CLASSES AND LISTING *
//* -----*
//READCOMP EXEC PGM=IKJEFT01
//STEPLIB DD DSN=FLHQ1.FLHQ2.LOADLIB, DISP=SHR
//SYSTSIN DD *
EXEC 'FLHQ1.FLHQ2.CGI(C9JPRJCL)' 'JEXEC, JLIST'
//SYSEXEC DD DSN=FLHQ1.FLHQ2.CGI, DISP=SHR
//SYSTSPRT DD SYSOUT=*
//CMLST1 DD DSN=&&CMLST1, DISP=(OLD, DELETE)
//JEXEC DD DSN=&&JAVACLAS, DISP=(NEW, PASS),
```

```

//      UNIT=TDISK,DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB)
//JLIST  DD  DSN=&&JAVALIST,DISP=(NEW,PASS),
//      UNIT=TDISK,DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB)
//CMPOUT DD  DSN=&&JAVACOMP,DISP=(NEW,PASS),
//      UNIT=TDISK,DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB)
//* ----- *
//* COPY CLASSES TO TARGET
//* ----- *
//CLASSES EXEC PGM=BSTCOPY
//SYSIN  DD  DSN=&&JAVACLAS,DISP=(OLD,DELETE)
//SYSPRINT DD  SYSOUT=*
//SOURCE DD  DISP=SHR,
//      DSN=QUAL1.QUAL2.ES&C1SENVMT&C1SSTGNUM..CLASSES
//JEXEC  DD  DISP=SHR,
//      DSN=QUAL1.QUAL2.ES&C1EN&C1S#..CLASSES,
//      MONITOR=COMPONENTS
//* ----- *
//* COPY LISTING TO TARGET
//* ----- *
//LISTING EXEC PGM=BSTCOPY
//SYSIN  DD  DSN=&&JAVALIST,DISP=(OLD,DELETE)
//SYSPRINT DD  SYSOUT=*
//SOURCE DD  DISP=SHR,
//      DSN=QUAL1.QUAL2.ES&C1SENVMT&C1SSTGNUM..LISTINGS
//JLIST  DD  DISP=SHR,
//      DSN=QUAL1.QUAL2.ES&C1EN&C1S#..LISTINGS,
//      MONITOR=COMPONENTS
//* ----- *
//* COPY CLASSES AND LISTING TO UNIX
//* ----- *
//IKJEFT01 EXEC PGM=IKJEFT01
//STEPLIB DD  DSN=FLHQ1.FLHQ2.LOADLIB,DISP=SHR
//SYSTSIN DD  *
EXEC 'FLHQ1.FLHQ2.CGI(C9JPRGEN)' '&TBJ &C1EN &C1SI &C1SY +
&C1SU &C1TY &C1ELEMENT &C1SENVMT &C1SSTGID &C1SSYSTEM &C1SSUBSYS'
//SYSEXEC DD  DSN=FLHQ1.FLHQ2.CGI,DISP=SHR
//SYSTSPRT DD  SYSOUT=*
//JFILE  DD  DSN=&&SYSIN(&C1ELEMENT),DISP=(OLD,DELETE)
//JCOMPS DD  DSN=&&JAVACOMP,DISP=(OLD,DELETE)
//UNIXLOC DD  DSN=FLHQ1.FLHQ2.HTML(&UNIXLOC),DISP=SHR
//* ----- *
//* REBUILD THE TARGET COMPONENT LIST
//* ----- *
//BC1PMVCL EXEC PGM=BC1PMVCL
//* ----- *

```

Modify UNIXLOC – Common Endeavor to USS Life Cycle Mapping Rules

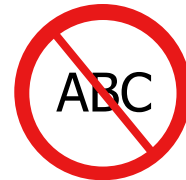
CASE SENSITIVITY ALERT!

The following processor 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.



The following member can be found in the HTML file offloaded from the product cartridge or sent via email. This member is input to all E-JDK processors. It is used to map the Endeavor to USS life cycles locations.

```
*-----*
* CLOUD 9 JAVA/USS E-JDK COMPONENT *
*-----*
* NAME: UNIXLOC *
* PURPOSE: ENDEAVOR TO UNIX LIFE CYCLE MAPPING RULES. *
*-----*
* env, stg, sys, sub, typ          unix location          KEEP or DELETE
*                                  on MOVE
TEST, A, SYSJ, SUBJ, GRAPHICS     /u/test/a/sysj/subj/graphics KEEP
TEST, A, SYSJ, SUBJ, HTML         /u/test/a/sysj/subj/html    KEEP
TEST, A, SYSJ, SUBJ, JAR          /u/test/a/sysj/subj         KEEP
TEST, A, SYSJ, SUBJ, JAREXEC      /u/test/a/sysj/subj/jar     KEEP
TEST, A, SYSJ, SUBJ, JARLIST      /u/test/a/sysj/subj/listings KEEP
TEST, A, SYSJ, SUBJ, JAVA         /u/test/a/sysj/subj         KEEP
TEST, A, SYSJ, SUBJ, JEXEC        /u/test/a/sysj/subj/classes  KEEP
TEST, A, SYSJ, SUBJ, JLIST        /u/test/a/sysj/subj/listings KEEP
*
TEST, B, SYSJ, SUBJ, GRAPHICS     /u/test/b/sysj/subj/graphics KEEP
TEST, B, SYSJ, SUBJ, HTML         /u/test/b/sysj/subj/html    KEEP
```



UNIXLOC – Common Endeavor to USS Life Cycle Map

Special Note on JEXEC and JLIST

The example above shows the defaults shipped with the product. Note that there are Endeavor mapping locations for not only actual types, such as Java and HTML, but there are also mappings for output component libraries such as JEXEC and JLIST. The move processor uses these mappings for connecting the component list ddname to a location in Unix. For example, the default move processor utility C9JPLJCL, uses the input parameter of (JEXEC,JLIST). This tells the utilities that elements in the component list that use this ddname are mapped and should be promoted through the USS life cycle. If you change the processor ddnames of JEXEC and JLIST, you must also change the input

parm and mapping rules in the above member.

Modify JCPATH Common %CLASSPATH% Substitution

The following member can be found in the HTML file offloaded from the product cartridge or sent via email. 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 E-JDK COMPONENT *
* ----- *
* NAME: JCPATH *
* PURPOSE: %CLASSPATH% SUBSTITUTION FILE. *
* ----- *
* env, stg, sys, sub classpath concatenation
TEST, A, SYSJ, SUBJ /u/test/a/sysj/subj/classes
TEST, B, SYSJ, SUBJ /u/test/b/sysj/subj/classes
```



JCPATH – Common %CLASSPATH% Substitution

Modify JCOMPILE – JAVA Compile Shell:

The following member can be found in the HTML file offloaded from the product cartridge or sent via email. This member is the default input to the C9JPPJAV processor for the JAVA Type. You will need to review the path location with the your USS System's Programmer. The %CLASSPATH% variable is built by Cloud 9 from the values found in the JCPATH member.

```
# ----- #
# CLOUD 9 JAVA/USS E-JDK COMPONENT #
# ----- #
# NAME: JCOMPILE #
# PURPOSE: JAVA COMPILE SHELL #
# ----- #
export PATH=/usr/lpp/java/J1.1/bin:$PATH
export CLASSPATH=%CLASSPATH%:$CLASSPATH
cd $1
javac -verbose -d classes $2
```



JCOMPILE Java Compile Shell

Modify UNIXMAP Java Output Mapping Rules:

The following member can be found in the HTML file offloaded from the product cartridge or sent via email. This member is the default input to the C9JPPJAV processor for the JAVA Type and is used to define USS outputs for Java Compiles.

NOTE: EXAMPLE IS SHOWN WRAPPED AROUND.
ACTUAL MEMBER IS TOO LONG TO DISPLAY

```
*-----*
*  CLOUD 9 JAVA/USS E-JDK COMPONENT                *
*-----*
* NAME:      UNIXMAP                               *
* PURPOSE:   MAPPING RULES FOR JAVA COMPILE USS   *
*            OUTPUT LOCATIONS.                    *
*-----*
*env, stg, sys, sub, type      java source      java classes      java listing
TEST, A, SYSJ, SUBJ, JAVA    /u/test/a/sysj/subj /u/test/a/sysj/subj/classes
                               /u/test/a/sysj/subj/listings
```



UNIXMAP – Java Output Mapping Rules

Modify JCOMPILR Jar Compile Shell:

The following member can be found in the HTML file offloaded from the product cartridge or sent via email. This member is the default input to the C9JPPJAV processor for the JAR Type. You will need to review the path statement with your USS Systems Programmer. The %CLASSPATH% variable is built by Cloud 9 from the values found in the JCPATH member.

```
# -----#  
# CLOUD 9 JAVA/USS E-JDK COMPONENT#  
# -----#  
# NAME: JCOMPILR#  
# PURPOSE: JAR COMPILE SHELL#  
# USAGE: USED AS INPUT TO THE C9JPPJAV PROCESSOR#  
# READ FROM STANDARD OS/390 PDS DATASET#  
# -----#  
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%
```



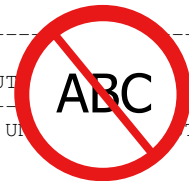
JCOMPILR – JAR Compile Shell

Modify UNIXMAPR - Jar Output Mapping Rules:

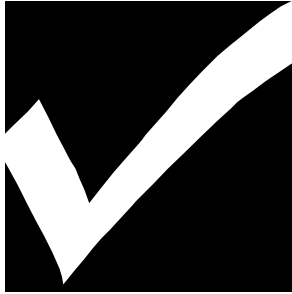
The following member can be found in the HTML file offloaded from the product cartridge or sent via email. This member is the default input to the C9JPPJAV processor for the JAR Type. It is used to define the USS Output locations for Jar compiles.

NOTE: EXAMPLE IS SHOWN WRAPPED AROUND.
ACTUAL MEMBER IS TOO LONG TO DISPLAY.

```
* -----*  
* CLOUD 9 JAVA/USS E-JDK COMPONENT*  
* -----*  
* NAME: UNIXMAPR*  
* PURPOSE: MAPPING RULES FOR JAR COMPILE USS OUTPUT*  
* -----*  
*env, stg, sys, sub, typ UNIX MAKE OUT UI T*  
jar-listing-loc  
TEST,A,SYSJ,SUBJ,JAR /u/test/a/sysj/subj/classes  
/u/test/a/sysj/subj/jar  
u/test/a/sysj/subj/listings
```



UNIXMAPR – JAR Output Mapping Rules



CHECKPOINT #2

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

Task	Completed?
Reviewed and modified processor C9JPPDEL?	
Reviewed and modified processor C9JPPGEN?	
Reviewed and modified processor C9JPPJAV?	
Reviewed and modified processor C9JPPJDL?	
Reviewed and modified processor C9JPPJVM?	
Reviewed and modified control file UNIXLOC?	
Reviewed and modified control file JCOMPILE?	
Reviewed and modified control file JCOMPILR?	
Reviewed and modified control file UNIXMAP?	
Reviewed and modified control file UNIXMAPR?	
Reviewed and modified control file JCPATH?	

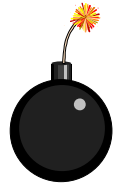
Checkpoint 2

Step 3: Run C9JPJINV to build E-JDK System, Types, and Processors Groups

The purpose of this step is to define all of the CA-Endevor inventory locations required for E-JDK.

Modify and Submit C9JPJINV

1. Using ISPF EDIT, access member C9JPJINV in the JCLLIB you offloaded from the installation tape.
2. Copy your job card values to the top of the member.
3. Substitute your site-specific values (identified on the Installation Worksheet.) for those values shown below in bold.



4. Make sure you change all Endeavor Inventory locations that will not be set to the defaults!

5. Submit the job.

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

```
//* (JOB CARD)
//* -----*
//*      CLOUD 9 JAVA/E-JDK COMPONENTS.
//* -----*
//*
//* NAME:      C9JPJINV
//* PURPOSE:  JCL TO PERFORM BATCH ADMIN DEFINITIONS OF E-JDK TYPES.
//* -----*
//* TO USE THIS JCL, YOU MUST:
//* 1) INSERT A VALID JOB CARD WITH VALID CLASS AND REGION=OM
//* 2) MAKE SURE THAT THE STEPLIB POINTS TO YOUR ENDEVOR
//*    AUTHORIZED DATASET AND INCLUDES THE DATASET THAT
//*    CONTAINS CIGINI AND CIGFEXEC.
//* 3) MAKE SURE THAT THE CONLIB POINTS YOUR ENDEVOR
//*    CONLIB DATASET
//* 4) CHANGE FLHQ1, FLHQ2, QUAL1 AND QUAL2 AS PER YOUR
//*    INSTALLATION SHEET
//* 5) CHANGE THE UNIT=TDISK TO THE APPROPRIATE UNIT
//*    NAME.
//* 6) REVIEW AND MODIFY THE ENDEVOR INVENTORY VALUES
//* -----*
```

```

//*          TO MEET SITE STANDARDS.                                *
//*          6) REVIEW AND MODIFY THE LOAD AND LIST LIBRARY NAMES  *
//*          TO MEET SITE STANDARDS.                                *
//*          7) REVIEW AND MODIFY THE BASE AND DELTA LIBRARY NAMES *
//*          TO MEET SITE STANDARDS.                                *
//* ----- *
//C9JPPDEF EXEC PGM=NDVRC1,PARM='ENBE1000',REGION=0M
//STEPLIB DD DSN=QUAL1.QUAL2.ENDEVOR,DISP=SHR
//CONLIB DD DSN=QUAL1.QUAL2.ENDEVOR,DISP=SHR
//CLMSGSL DD SYSOUT=*
//ENESCLIN DD *
* ----- *
DEFINE SYSTEM SYSJ
TO ENVIRONMENT TEST
DESCRIPTION 'JAVA SYSTEM'
STAGE ONE LOAD LIBRARY 'QUAL1.QUAL2.EPLOD1'
STAGE ONE LIST LIBRARY 'QUAL1.QUAL2.EPLST1'
STAGE TWO LOAD LIBRARY 'QUAL1.QUAL2.EPLOD2'
STAGE TWO LIST LIBRARY 'QUAL1.QUAL2.EPLST2'
.
* ----- *
DEFINE SUBSYSTEM SUBJ
TO ENVIRONMENT TEST SYSTEM SYSJ
DESCRIPTION 'JAVA SUBSYSTEM'
.
* ----- *
DEFINE TYPE GRAPHICS
TO ENVIRONMENT TEST SYSTEM SYSJ STAGE NUMBER 1
DESCRIPTION 'GRAPHIC FILES'
SOURCE ELEMENT LENGTH 256
COMPARE COLUMN 1 TO 256
LANGUAGE TEXT
DEFAULT PROCESSOR GROUP IS 'GRAPHICS'
BASE LIBRARY 'QUAL1.QUAL2.EBASE1'
DELTA LIBRARY 'QUAL1.QUAL2.EDELTA1'
DO NOT COMPRESS BASE
ELEMENT DELTA FORMAT IS REVERSE
.
DEFINE TYPE GRAPHICS
TO ENVIRONMENT TEST SYSTEM SYSJ STAGE NUMBER 2
DESCRIPTION 'GRAPHICS FILES'
SOURCE ELEMENT LENGTH 256
COMPARE COLUMN 1 TO 256
LANGUAGE TEXT
DEFAULT PROCESSOR GROUP IS 'GRAPHICS'
BASE LIBRARY 'QUAL1.QUAL2.EBASE2'
DELTA LIBRARY 'QUAL1.QUAL2.EDELTA2'
DO NOT COMPRESS BASE
ELEMENT DELTA FORMAT IS REVERSE
.
DEFINE TYPE HTML
TO ENVIRONMENT TEST SYSTEM SYSJ STAGE NUMBER 1
DESCRIPTION 'HTML'
SOURCE ELEMENT LENGTH 256
COMPARE COLUMN 1 TO 256
LANGUAGE TEXT
DEFAULT PROCESSOR GROUP IS 'HTML'
BASE LIBRARY 'QUAL1.QUAL2.EBASE1'
DELTA LIBRARY 'QUAL1.QUAL2.EDELTA1'
.
DEFINE TYPE HTML
TO ENVIRONMENT TEST SYSTEM SYSJ STAGE NUMBER 2
DESCRIPTION 'HTML'
SOURCE ELEMENT LENGTH 256
COMPARE COLUMN 1 TO 256
LANGUAGE TEXT
DEFAULT PROCESSOR GROUP IS 'HTML'
BASE LIBRARY 'QUAL1.QUAL2.EBASE2'
DELTA LIBRARY 'QUAL1.QUAL2.EDELTA2'
.

```

```

DEFINE TYPE JAVA
  TO ENVIRONMENT TEST SYSTEM SYSJ STAGE NUMBER 1
  DESCRIPTION 'JAVA SOURCE'
  SOURCE ELEMENT LENGTH 256
  COMPARE COLUMN 1 TO 256
  LANGUAGE TEXT
  DEFAULT PROCESSOR GROUP IS 'JAVA'
  BASE LIBRARY 'QUAL1.QUAL2.EBASE1'
  DELTA LIBRARY 'QUAL1.QUAL2.EDELTA1'
.
DEFINE TYPE JAVA
  TO ENVIRONMENT TEST SYSTEM SYSJ STAGE NUMBER 2
  DESCRIPTION 'JAVA SOURCE'
  SOURCE ELEMENT LENGTH 256
  COMPARE COLUMN 1 TO 256
  LANGUAGE TEXT
  DEFAULT PROCESSOR GROUP IS 'JAVA'
  BASE LIBRARY 'QUAL1.QUAL2.EBASE2'
  DELTA LIBRARY 'QUAL1.QUAL2.EDELTA2'
.
DEFINE TYPE JAR
  TO ENVIRONMENT TEST SYSTEM SYSJ STAGE NUMBER 1
  DESCRIPTION 'JAR SOURCE'
  SOURCE ELEMENT LENGTH 256
  COMPARE COLUMN 1 TO 256
  LANGUAGE TEXT
  DEFAULT PROCESSOR GROUP IS 'JAR'
  BASE LIBRARY 'QUAL1.QUAL2.EBASE1'
  DELTA LIBRARY 'QUAL1.QUAL2.EDELTA1'
.
DEFINE TYPE JAR
  TO ENVIRONMENT TEST SYSTEM SYSJ STAGE NUMBER 2
  DESCRIPTION 'JAR SOURCE'
  SOURCE ELEMENT LENGTH 256
  COMPARE COLUMN 1 TO 256
  LANGUAGE TEXT
  DEFAULT PROCESSOR GROUP IS 'JAR'
  BASE LIBRARY 'QUAL1.QUAL2.EBASE2'
  DELTA LIBRARY 'QUAL1.QUAL2.EDELTA2'
.
DEFINE TYPE PROCESS
  TO ENVIRONMENT TEST SYSTEM SYSJ STAGE NUMBER 1
  DESCRIPTION 'PROCESSORS'
  SOURCE ELEMENT LENGTH 80
  COMPARE COLUMN 1 TO 72
  LANGUAGE CNLPROC
  BASE LIBRARY 'QUAL1.QUAL2.EBASE1'
  DELTA LIBRARY 'QUAL1.QUAL2.EDELTA1'
.
DEFINE TYPE PROCESS
  TO ENVIRONMENT TEST SYSTEM SYSJ STAGE NUMBER 2
  DESCRIPTION 'PROCESSORS'
  SOURCE ELEMENT LENGTH 80
  COMPARE COLUMN 1 TO 72
  LANGUAGE CNLPROC
  BASE LIBRARY 'QUAL1.QUAL2.EBASE2'
  DELTA LIBRARY 'QUAL1.QUAL2.EDELTA2'
.
* - - - - -
DEFINE PROCESSOR GROUP GRAPHICS
  TO ENVIRONMENT TEST SYSTEM SYSJ TYPE GRAPHICS STAGE NUMBER 1
  DESCRIPTION 'GRAPHICS PROCESSOR GROUP'
  GENERATE PROCESSOR C9JPPGEN
  MOVE PROCESSOR C9JPPGEN
  DELETE PROCESSOR C9JPPDEL
.
DEFINE PROCESSOR GROUP GRAPHICS
  TO ENVIRONMENT TEST SYSTEM SYSJ TYPE GRAPHICS STAGE NUMBER 2
  DESCRIPTION 'GRAPHICS PROCESSOR GROUP'

```

```

GENERATE PROCESSOR C9JPPGEN
MOVE PROCESSOR C9JPPGEN
DELETE PROCESSOR C9JPPDEL
.
DEFINE PROCESSOR GROUP HTML
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE HTML STAGE NUMBER 1
DESCRIPTION 'HTML PROCESSOR GROUP'
GENERATE PROCESSOR C9JPPGEN
MOVE PROCESSOR C9JPPGEN
DELETE PROCESSOR C9JPPDEL
.
DEFINE PROCESSOR GROUP HTML
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE HTML STAGE NUMBER 2
DESCRIPTION 'HTML PROCESSOR GROUP'
GENERATE PROCESSOR C9JPPGEN
MOVE PROCESSOR C9JPPGEN
DELETE PROCESSOR C9JPPDEL
.
DEFINE PROCESSOR GROUP JAR
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAR STAGE NUMBER 1
DESCRIPTION 'JAR PROCESSOR GROUP'
GENERATE PROCESSOR C9JPPJAV
MOVE PROCESSOR C9JPPJMV
DELETE PROCESSOR C9JPPJDL
.
DEFINE PROCESSOR GROUP JAR
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAR STAGE NUMBER 2
DESCRIPTION 'JAR PROCESSOR GROUP'
GENERATE PROCESSOR C9JPPJAV
MOVE PROCESSOR C9JPPJMV
DELETE PROCESSOR C9JPPJDL
.
DEFINE PROCESSOR GROUP JAVA
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAVA STAGE NUMBER 1
DESCRIPTION 'JAVA PROCESSOR GROUP'
GENERATE PROCESSOR C9JPPJAV
MOVE PROCESSOR C9JPPJMV
DELETE PROCESSOR C9JPPJDL
.
DEFINE PROCESSOR GROUP JAVA
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAVA STAGE NUMBER 2
DESCRIPTION 'JAVA PROCESSOR GROUP'
GENERATE PROCESSOR C9JPPJAV
MOVE PROCESSOR C9JPPJMV
DELETE PROCESSOR C9JPPJDL
.
DEFINE PROCESSOR GROUP PROCESS
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE PROCESS STAGE NUMBER 1
DESCRIPTION 'PROCESS PROCESSOR'
GENERATE PROCESSOR GPPROCSS
DELETE PROCESSOR DPPROCSS
MOVE PROCESSOR GPPROCSS
.
DEFINE PROCESSOR GROUP PROCESS
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE PROCESS STAGE NUMBER 2
DESCRIPTION 'PROCESS PROCESSOR'
GENERATE PROCESSOR GPPROCSS
DELETE PROCESSOR DPPROCSS
MOVE PROCESSOR GPPROCSS
.
* - - - - -
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE GRAPHICS STAGE NUMBER 1
PROCESSOR GROUP GRAPHICS
PROCESS TYPE = GENERATE SYMBOL TBJ = 'BINARY'
.
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE GRAPHICS STAGE NUMBER 2

```

```

PROCESSOR GROUP GRAPHICS
PROCESS TYPE = GENERATE      SYMBOL TBJ = 'BINARY'
.
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE GRAPHICS      STAGE NUMBER 1
PROCESSOR GROUP GRAPHICS
PROCESS TYPE = MOVE          SYMBOL TBJ = 'BINARY'
.
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE GRAPHICS      STAGE NUMBER 2
PROCESSOR GROUP GRAPHICS
PROCESS TYPE = MOVE          SYMBOL TBJ = 'BINARY'
.
* -----
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE HTML          STAGE NUMBER 1
PROCESSOR GROUP HTML
PROCESS TYPE = GENERATE      SYMBOL TBJ = 'TEXT'
.
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE HTML          STAGE NUMBER 2
PROCESSOR GROUP HTML
PROCESS TYPE = GENERATE      SYMBOL TBJ = 'TEXT'
.
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE HTML          STAGE NUMBER 1
PROCESSOR GROUP HTML
PROCESS TYPE = MOVE          SYMBOL TBJ = 'TEXT'
.
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE HTML          STAGE NUMBER 2
PROCESSOR GROUP HTML
PROCESS TYPE = MOVE          SYMBOL TBJ = 'TEXT'
.
* -----
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAR          STAGE NUMBER 1
PROCESSOR GROUP JAR
PROCESS TYPE = GENERATE      SYMBOL JPGM   = 'C9JPRJAR'
                               SYMBOL JSHELL = 'JCOMPILR'
                               SYMBOL JCMAP  = 'UNIXMAPR'
                               SYMBOL JCPATH = 'JCPATH'
.
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAR          STAGE NUMBER 2
PROCESSOR GROUP JAR
PROCESS TYPE = GENERATE      SYMBOL JPGM   = 'C9JPRJAR'
                               SYMBOL JSHELL = 'JCOMPILR'
                               SYMBOL JCMAP  = 'UNIXMAPR'
                               SYMBOL JCPATH = 'JCPATH'
.
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAR          STAGE NUMBER 1
PROCESSOR GROUP JAR
PROCESS TYPE = MOVE          SYMBOL TBJ = 'JAR'
.
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAR          STAGE NUMBER 2
PROCESSOR GROUP JAR
PROCESS TYPE = MOVE          SYMBOL TBJ = 'JAR'
.
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAR          STAGE NUMBER 1
PROCESSOR GROUP JAR
PROCESS TYPE = DELETE        SYMBOL TBJ = 'JAR'
.
DEFINE PROCESSOR SYMBOL
TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAR          STAGE NUMBER 2
PROCESSOR GROUP JAR
PROCESS TYPE = DELETE        SYMBOL TBJ = 'JAR'

```

```

.
* -----
DEFINE PROCESSOR SYMBOL
  TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAVA          STAGE NUMBER 1
.
  PROCESSOR GROUP JAVA
  PROCESS TYPE = GENERATE      SYMBOL JPGM   = 'C9JPRJAV'
                                SYMBOL JSHELL = 'JCOMPILE'
                                SYMBOL JCMAP  = 'UNIXMAP'
                                SYMBOL JCPATH = 'JCPATH'
.
DEFINE PROCESSOR SYMBOL
  TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAVA          STAGE NUMBER 2
  PROCESSOR GROUP JAVA
  PROCESS TYPE = GENERATE      SYMBOL JPGM   = 'C9JPRJAV'
                                SYMBOL JSHELL = 'JCOMPILE'
                                SYMBOL JCMAP  = 'UNIXMAP'
                                SYMBOL JCPATH = 'JCPATH'
.
DEFINE PROCESSOR SYMBOL
  TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAVA          STAGE NUMBER 1
  PROCESSOR GROUP JAVA
  PROCESS TYPE = MOVE          SYMBOL TBJ   = 'JAVA'
.
DEFINE PROCESSOR SYMBOL
  TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAVA          STAGE NUMBER 2
  PROCESSOR GROUP JAVA
  PROCESS TYPE = MOVE          SYMBOL TBJ   = 'JAVA'
.
DEFINE PROCESSOR SYMBOL
  TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAVA          STAGE NUMBER 1
  PROCESSOR GROUP JAVA
  PROCESS TYPE = DELETE        SYMBOL TBJ   = 'JAVA'
.
DEFINE PROCESSOR SYMBOL
  TO ENVIRONMENT TEST SYSTEM SYSJ TYPE JAVA          STAGE NUMBER 2
  PROCESSOR GROUP JAVA
  PROCESS TYPE = DELETE        SYMBOL TBJ   = 'JAVA'
.
/*

```

Step 4: Run C9JPJADD to add E-JDK Processors

The purpose of this step is to add the modified E-JDK Cloud 9 processors into Endeavor.

Modify and Submit C9JPJADD

1. Using ISPF EDIT, access member C9JPJADD in the JCLLIB you offloaded from the installation tape.
2. Copy your job card values to the top of the member. Substitute your site-specific values (identified on the Implementation Worksheet.) for those values shown below in bold.
3. Submit the job.

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

```
//* (JOB CARD)
//* ----- *
//* CLOUD 9 JAVA/E-JDK COMPONENTS. *
//* ----- *
//* NAME: C9JPJADD *
//* PURPOSE: SAMPLE JCL TO ADD JAVA PROCESSORS INTO ENDEVOR. *
//* ----- *
//* INSTRUCTIONS: *
//* MODIFY THE FOLLOWING SYMBOLICS TO MATCH THE VALUES *
//* IN THE E-JDK WORKSHEET. *
//* 1) FLHQ1 AND FLHQ2 FOR CIG PRODUCT LIBRARIES. *
//* 2) QUAL1 AND QUAL2 FOR ENDEVOR PRODUCT LIBRARIES. *
//* 3) MODIFY ALL ENDEVOR INVENTORY LOCATIONS AS PER *
//* WORKSHEET. *
//* ----- *
//SCL EXEC PGM=NDVRC1,DYNAMNBR=1500,PARM='C1BM3000',REGION=4096K
//STEPLIB DD DSN=FLHQ1.FLHQ2.LOADLIB,DISP=SHR
// DD DSN=QUAL1.QUAL2.ENDEVOR,DISP=SHR
//CONLIB DD DSN=QUAL1.QUAL2.ENDEVOR,DISP=SHR
//BSTIPT01 DD *

* GENERATE PROCESSOR: JAVA, JAR COMPILER
ADD ELEMENT C9JPPJAV
FROM DSN 'FLHQ1.FLHQ2.JCL' MEMBER C9JPPJAV
TO ENV TEST SYSTEM SYSJ SUBSYSTEM SUBJ TYPE PROCESS
OPTIONS OVERRIDE SIGNOUT UPDATE
.

* MOVE PROCESSOR: JAVA, JAR
ADD ELEMENT C9JPPJMV
FROM DSN 'FLHQ1.FLHQ2.JCL' MEMBER C9JPPJMV
TO ENV TEST SYSTEM SYSJ SUBSYSTEM SUBJ TYPE PROCESS
OPTIONS OVERRIDE SIGNOUT UPDATE
.
```

```
* DELETE PROCESSOR: JAVA, JAR
```

```
ADD ELEMENT C9JPPJDL
```

```
FROM DSN 'FLHQ1.FLHQ2.JCL' MEMBER C9JPPJDL  
TO ENV TEST SYSTEM SYSJ SUBSYSTEM SUBJ TYPE PROCESS  
OPTIONS OVERRIDE SIGNOUT UPDATE
```

```
.  
* GENERATE AND MOVE PROCESSOR: HTML, GRAPHICS  
ADD ELEMENT C9JPPGEN
```

```
FROM DSN 'FLHQ1.FLHQ2.JCL' MEMBER C9JPPGEN  
TO ENV TEST SYSTEM SYSJ SUBSYSTEM SUBJ TYPE PROCESS  
OPTIONS OVERRIDE SIGNOUT UPDATE
```

```
.  
* DELETE PROCESSOR: HTML, GRAPHICS
```

```
ADD ELEMENT C9JPPDEL
```

```
FROM DSN 'FLHQ1.FLHQ2.JCL' MEMBER C9JPPDEL  
TO ENV TEST SYSTEM SYSJ SUBSYSTEM SUBJ TYPE PROCESS  
OPTIONS OVERRIDE SIGNOUT UPDATE
```

```
.  
EOJ.
```

```
//C1MSG1 DD SYSOUT=*
```

```
//C1MSG2 DD SYSOUT=*
```

```
//C1PRINT DD SYSOUT=*
```

C9JPJADD – Add E-JDK Processors

Step 5: Run C9JPJEE to load the E-JDK control records

The purpose of this step is to load the required EE control records into Cloud 9.

Modify and Submit C9JPJEE

1. Using ISPF EDIT, access member C9JPJEE in the JCLLIB you offloaded from the installation tape.
2. 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.
3. Submit the job.

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

```
//* (JOB CARD)
/* -----*
/*          CLOUD 9 JAVA/E-JDK COMPONENTS.          *
/* -----*
/*          NAME:      C9JPJEE                        *
/*          PURPOSE:  PERFORM LOAD OF 'EE' CONTROL RECORDS FOR E-JDK TYPES. *
/* -----*
/*          TO USE THIS JCL, YOU MUST:                *
/*          1) INSERT A VALID JOB CARD WITH VALID CLASS AND REGION=OM *
/*          2) MAKE SURE THAT THE STEPLIB POINTS TO YOUR ENDEVOR *
/*             AUTHORIZED DATASETS. *
/*          3) MAKE SURE THAT THE CONLIB POINTS YOUR ENDEVOR *
/*             CONLIB DATASET *
/*          4) CHANGE FLHQ1, FLHQ2, QUAL1 AND QUAL2 AS PER YOUR *
/*             INSTALLATION SHEET *
/*          5) CHANGE THE UNIT=TDISK TO THE APPROPRIATE UNIT *
/*             NAME. *
/*          6) MODIFY THE BUILD SCL REQUEST IN STEP1 TO REFLECT THE *
/*             SCOPE OF YOUR E-JDK ENVIRONMENT. *
/*          7) VERIFY THAT THE FASTLIST DATABASE NAME IS CORRECT *
/*             IN STEP 2. *
/* -----*
/*          STEP1: PERFORM BATCH ADMIN FUNCTIONS AGAINST SELECTED ENVIRONNENTS*
/* -----*
/*          STEP1      EXEC PGM=NDVRC1, PARM='ENBE1000', DYNAMNBR=1500
/*          STEPLIB DD DSN=FLHQ1.FLHQ2.LOADLIB, DISP=SHR
/*                   DD DSN=QUAL1.QUAL2.LOADLIB, DISP=SHR
/*          CONLIB DD DSN=QUAL1.QUAL2.ENDEVOR, DISP=SHR
/*          C1MSG S1 DD SYSOUT=*
/*          C1MSG S2 DD SYSOUT=*
/*          OUTFILE DD DSN=&&BATDATA, DISP=(NEW,PASS),
```

```

//          SPACE=(CYL,(10,10)),UNIT=TDISK,
//          DCB=(LRECL=80,BLKSIZE=31200,RECFM=FB)
//ENESCLIN DD *
BUILD SCL FOR SYSTEM *
FROM ENVIRONMENT 'TEST'
INCLUDE SUBORDINATES
TO DDNAME 'OUTFILE'
.
//* -----*
//*
//* STEP2: PERFORM CONTROL DATA UPDATE TO FASTLIST DATABASE.
//*
//* -----*
//STEP2IF IF (STEP1.RC < 8) THEN
//STEP2 EXEC PGM=CIGELOAD,REGION=0M
//STEPLIB DD DSN=FLHQ1.FLHQ2.LOADLIB,DISP=SHR
//CIGIN DD DSN=&&BATDATA,DISP=(OLD,DELETE)
//CIGLOG DD SYSOUT=*
//CIGPUNCH DD SYSOUT=*
//CIGDB DD DSN=FLHQ1.FLHQ2.FLSTDB,DISP=SHR
//STEP2END ENDIF

```

C9JPJEE – Load EE Control Records

Step 6: Run C9JPJSLR to define E-JDK types to SLR

The purpose of this step is to define the E-JDK types to your Cloud 9 SLR file.

Modify and Submit C9JPJSLR

1. Using ISPF EDIT, access member C9JPJSLR in the JCLLIB you offloaded from the installation tape.
2. 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**.
3. Submit the job.

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

```
//* (JOB CARD)
//* -----*
//*      CLOUD 9 JAVA/E-JDK COMPONENTS.
//* -----*
//*
//* NAME:      C9JPJSLR
//* PURPOSE:  DEFINE E-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) CHANGE FLHQ1 AND FLHQ2 AS PER YOUR
//*          INSTALLATION SHEET
//*      4) MODIFY THE TYPE NAMES IF YOU HAVE CHANGED THE DEFAULT
//*          Endeavor TYPES.
//* -----*
//STEP1      EXEC PGM=C9LSLR
//STEPLIB   DD DSN=FLHQ1.FLHQ2.LOADLIB,DISP=SHR
//CIGIN     DD *
ADD NAME RULE FOR ENDEVOR TYPE GRAPHICS CASE SENSITIVE.
ADD NAME RULE FOR ENDEVOR TYPE HTML CASE SENSITIVE.
ADD NAME RULE FOR ENDEVOR TYPE JAVA CASE SENSITIVE.
ADD NAME RULE FOR ENDEVOR TYPE JAR CASE SENSITIVE.
//CIGLOG    DD SYSOUT=*
```

C9JPJSLR – Define E-JDK Types to Cloud 9 SLR

Step 7: Run C9JPJUNX to build E-JDK USS Directories

The purpose of this step is to define the E-JDK Unix Directories.

Modify and Submit C9JPJUNX

1. Using ISPF EDIT, access member C9JPJUNX in the JCLLIB you offloaded from the installation tape.
2. Issues 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 that this job should terminate with COND CODE=0.

```
/* (jobcard)
/* -----*
/* CLOUD 9 JAVA/E-JDK COMPONENTS. *
/* -----*
/* NAME: C9JPJUNX *
/* PURPOSE: Perform unix setup for e-jdk USS mapping. *
/* -----*
/* TO USE THIS JCL, YOU MUST: *
/* 1) ISSUE CAPS OFF TO MAINTAIN CASE SENSITIVITY. *
/* 2) INSERT A VALID JOB CARD WITH VALID CLASS AND REGION=0M *
/* 3) modify directory values as per worksheet. *
/* -----*
//JAVA1 EXEC PGM=IKJEFT01
//SYSPROC DD DSN=SYS1.SBPXEXEC,DISP=SHR
//SYSTSIN DD *

/* Create all directories for env=test,stg=a */
oshell mkdir -p /u/test/a/sysj/subj
oshell mkdir -p /u/test/a/sysj/subj/classes
oshell mkdir -p /u/test/a/sysj/subj/graphics
oshell mkdir -p /u/test/a/sysj/subj/html
oshell mkdir -p /u/test/a/sysj/subj/jar
oshell mkdir -p /u/test/a/sysj/subj/listings

/* Set permissions for env=test,stg=a */
oshell chmod -R 777 /u/test/a/sysj/subj

/* ----- */

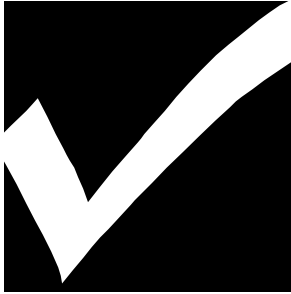
/* Create all directories for env=test,stg=b */
oshell mkdir -p /u/test/b/sysj/subj
```

```
oshell mkdir -p /u/test/b/sysj/subj/classes
oshell mkdir -p /u/test/b/sysj/subj/graphics
oshell mkdir -p /u/test/b/sysj/subj/html
oshell mkdir -p /u/test/b/sysj/subj/jar
oshell mkdir -p /u/test/b/sysj/subj/listings
```

```
/* Set permissions for env=test,stg=b */
oshell chmod -R 777 /u/test/b/sysj/subj
```

```
//SYSTSPRT DD SYSOUT=*
```

C9JPJUNX – Create USS E-JDK Directories



CHECKPOINT #3

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

Task	Completed?
Run C9JPJINV to define the E-JDK inventory locations to Endeavor	
Run C9JPJADD to add E-JDK Processors into Endeavor	
Run C9JPJEE to load the E-JDK Records for Cloud 9	
Run C9JPJSLR to define E-JDK types to Cloud 9	
Run C9JPJUNX to define USS directories	

Checkpoint 3

Step 8. Test the E-JDK Processors

There are two files delivered with the E-JDK for processor verification. One is a JAVA file called and one is an HTML invocation file call CLOCKH. There are all delivered in the flhq1.flhq2.HTML PDS file.

Member	Save to Desk Top As
CLOCKJ	Clock2.java
CLOCKH	Clockh.html

The following steps will allow you to verify the E-JDK processors.

1. Copy the CLOCKJ member to your desk top as Clock2.java
2. Invoke Cloud 9
3. Add Clock2.java into Cloud 9 using the 'ADD PC/WS File'
 - a. Add as a type JAVA
 - b. You should see a short name generated
4. Make sure to add the member with a GENERATE option.
5. Check the return codes of the Endeavor Add Job.
6. Review the expansion of the processor.
7. Review the component list for the Java Element.
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/test/a/sysj/subj/classes
Type Filename
_ Dir .
_ Dir ..
_ File Clock2.class
```

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
Java Endeavor Component List Example: /usr/lpp/java/J1.1/bin/./lib/classes
                                       .zip(java/lang/Runnable.class) in 4
                                       .checking class Clock2.
                                       ...more

```

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

```

PRINT ELEMENT: CLO00001      COMPONENT BROWSE

*****
*****
**
** COMPONENT BROWSE
**
** ENVIRONMENT: TEST        SYSTEM: SYSJ        SUBSYSTEM: SUBJ        **
** ELEMENT: CLO00001      TYPE: JAVA        STAGE: C9STAGE1        **
**
*****

----- COMPONENT LEVEL INFORMATION -----

VV.LL SYNC USER      DATE      TIME  STMTS CCID      COMMENT
-----
01.00      SYS01      01MAY01 19:03      13

----- ELEMENT INFORMATION -----

+00      VV.LL DATE      TIME  SYSTEM  SUBSYS  ELEMENT  TYPE  GROUP  STG S
+00      01.00 01MAY01 18:07 SYSJ    SUBJ    CLO00001 JAVA  JAVA   1

----- PROCESSOR INFORMATION -----

+00      VV.LL DATE      TIME  SYSTEM  SUBSYS  ELEMENT  TYPE  GROUP  STG S
+00      01.06 12APR01 15:14 SYSJ    SUBJ    C9JPPJAV PROCESS 1

----- SYMBOL INFORMATION -----

+00      DEFINED      SYMBOL      VALUE
+00      PROCESSOR   JCMAP       UNIXMAP
+00      PROCESSOR   JCPATH      JCPATH
+00      PROCESSOR   JPGM        C9JPRJAV
+00      PROCESSOR   JSHELL      JCOMPILE

----- OUTPUT COMPONENTS -----

STEP: IKJEFT01 DD=JEXEC      VOL=CIGV04 DSN=CIG.EJDK.ESTEST1.CLASSES

+00      MEMBER      VV.LL DATE      TIME  SYSTEM  SUBSYS  ELEMENT  TYPE  STG
+00      CLO00002    01.00 01MAY01 19:03 SYSJ    SUBJ    CLO00001 JAVA   1

```

```
STEP: IKJEFT01 DD=JLIST VOL=CIGV04 DSN=CIG.EJDK.ESTEST1.LISTINGS
```

```
+00 MEMBER VV.LL DATE TIME SYSTEM SUBSYS ELEMENT TYPE STG  
CLO00001 01.00 01MAY01 19:03 SYSJ SUBJ CLO00001 JAVA 1
```

Step 9: Invoking the Compiled Java

Copy the Java Classes Library and the clockh.html to a known location on a Web Server. For example, clockh.html to the WebSphere root directory and copy the Clock2.class to WebSphere rootdirectory/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 clockh.html request.

Enter the following in the location area of your browser:

Ip-address:port/clockh.html

Result of clockj.java Compile:

The result should be as follows:


A Clock (1.1) - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss Real.com

Address http://999.99.999.99:port#/clock.html Go

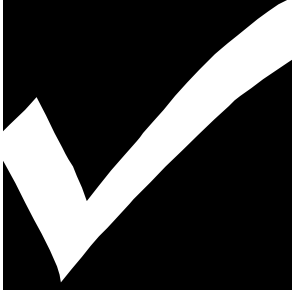
A Clock (1.1)



Mon Apr 16 16:48:31 2001

This is a demo of the clock

Clock2.class Display



CHECKPOINT #4

At this point you should have completed the following tasks:

Task	Completed?
Copied CLOCKJ to the Cloud 9 to a USS or Desk top location as Clock2.java?	
Added Clock2.java as a JAVA element into Endeavor via Cloud 9?	
Reviewed processor output?	
Reviewed population of USS Output directories?	
Invoked clockh.html from a web server location?	

Checkpoint 4

Step 10: Review the Cloud 9 USS Summary Reports

Java Generate Summary Report:

Using the SDSF Viewer or native SDSF, view the SYSTSPRT sysout output. The following is an example of the report created during a Generate Action:

```
-----  
Cloud 9 USS Processor Summary Report.  
Chicago Interface Group, Inc. All rights reserved.  
-----  
  
-----  
19:20:42 Starting the Cloud 9 USS JAVA/JAR Compile Process.  
19:20:42 Executing Cloud 9 USS Processor Utility C9JPLJAV.  
-----  
User = P390C Date = 15 Jul 2001  
Input Data = TEST A SYSJ SUBJ JAVA CLO00000  
-----  
19:20:42 in Initialization  
  
-----  
19:20:42 in MapLocation  
  
-----  
State of symbolic variables at start of MapLocation.  
-----  
ndvrEnv = TEST  
ndvrStg = A  
ndvrSys = SYSJ  
ndvrSub = SUBJ  
ndvrTyp = JAVA  
-----  
Match found in JCMAP file for inventory: TEST A SYSJ SUBJ JAVA  
Unix Source File = /u/test/a/sysj/subj  
Unix Class File = /u/test/a/sysj/subj/classes  
Unix List File = /u/test/a/sysj/subj/listings  
-----  
19:20:42 in PathLocation  
  
-----  
19:20:43 in GetLongname  
Performing short to long name translation.  
Short name = CLO00000  
Long name = Clock2.java  
sourceFileName = '/u/test/a/sysj/subj/Clock2.java'  
listFileName = '/u/test/a/sysj/subj/listings/Clock2.java'  
-----  
19:20:44 in FormatUnixMakeFile  
  
-----  
19:20:45 in CopyUnixMakeToTarget  
  
Return code from ALLOCATE Function = 0  
Allocation of FILEOUT complete:  
'/u/test/a/sysj/subj/jshell.JAVA.make'
```

```
Return code from OCOPY function = 0
Copying the Unix Shell file from JWORK ddname.
Copy complete to target: '/u/test/a/sysj/subj/jshell.JAVA.make'
Permissions set to 777: '/u/test/a/sysj/subj/jshell.JAVA.make'
```

```
-----
19:20:53 in CopySourceToTarget
```

```
Return code from ALLOCATE Function = 0
Allocation of FILEOUT complete: '/u/test/a/sysj/subj/Clock2.java'
```

```
Return code from OCOPY function = 0
Copying source from the JFILE ddname in processor.
Copy complete to target: '/u/test/a/sysj/subj/Clock2.java'
Permissions set to 777: '/u/test/a/sysj/subj/Clock2.java'
```

```
-----
19:21:01 in CompileJava
```

```
Return code from ALLOCATE Function = 0
Allocation      of          STDOUT          complete:
'/u/test/a/sysj/subj/listings/Clock2.java'
```

```
Return code from USS JAVA COMPILE function = 0
```

```
-----
19:24:54 in ReadListing
Permissions set to 777: '/u/test/a/sysj/subj/listings/Clock2.java'
```

```
Return code from ALLOCATE Function = 0
Allocation      of          STDOUT          complete:
'/u/test/a/sysj/subj/listings/Clock2.java'
```

```
Return code from OCOPY function = 0
Copy from: '/u/test/a/sysj/subj/listings/Clock2.java'
Copy complete to target: JWORK
```

```
-----
19:25:09 in WriteListing
```

```
Return code from ALLOCATE Function = 0
Allocation      of          JLIST          complete:
CIGT.STEVE.ESTEST1.LISTINGS(CLO00000)
```

```
-----
19:25:15 in GetInputComponentsNames
```

```
-----
19:25:15 in GetOutputComponentsNames
```

```
-----
19:25:15 in CopyClassesToTarget
```

```
Return code from ALLOCATE Function = 0
Allocation      of          I001          complete:
/u/test/a/sysj/subj/classes/Clock2.class
```

```
Return code from OCOPY function = 0
Copying source from: /u/test/a/sysj/subj/classes/Clock2.class
Copy complete to target: JWORK
```

```
Performing long to short name lookup.
Long name = Clock2.class
1Short Name = CLO00001
```

```
Return code from ALLOCATE Function = 0
```

```
Allocation of JEXEC complete: CIGT.STEVE.ESTEST1.CLASSES(CLO00001)
Return code from MVS Copy function = 0
Copied source from the JWORK ddname.
To MVS target: CIGT.STEVE.ESTEST1.CLASSES(CLO00001)
copying SN=CLO00001 LN=Clock2.class rc=0
```

```
-----
19:25:56 Ending the Cloud 9 USS JAVA/JAR Compile Process.
-----
```

```
READY
```

Clock2.java Generate Action Summary Report

Java Move Summary Report:

The following is an example of the report created during a Move Action:

```
-----
Cloud 9 USS Processor Summary Report.
Chicago Interface Group, Inc. All rights reserved.
-----
```

```
-----
19:26:54 Starting Cloud 9 USS Java Class Extraction.
19:26:54 Executing Cloud 9 USS Processor Utility C9JPLJCL.
-----
```

```
User = P390C Date = 15 Jul 2001
DDNames for Java Classes and Java Listings: JEXEC,JLIST
-----
```

```
19:26:54 in Initialization
-----
```

```
19:26:54 in ProcessComponentLists
-----
```

```
Processing the CMLST1 conwrite component data.
Building SELECT for component: CLO00001
Building SELECT for component: CLO00000
-----
```

```
19:26:54 Ending Cloud 9 USS Java Class Extraction.
-----
```

```
-----
Cloud 9 USS Processor Summary Report.
Chicago Interface Group, Inc. All rights reserved.
-----
```

```
-----
19:27:04 Starting the Cloud 9 USS Map and Copy Process.
19:27:04 Executing Cloud 9 USS Processor Utility C9JPLGEN.
-----
```

```
User = P390C Date = 15 Jul 2001
Input Data = JAVA TEST B SYSJ SUBJ JAVA CLO00000 TEST A SYSJ SUBJ
-----
```

19:27:04 in Initialization

19:27:04 in ReadJMAP

19:27:05 in FindMatchingRowInJMAP

State of symbolic variables at start of FindMatchingRowInJmap.

ndvrEnv = TEST
ndvrStg = A
ndvrSys = SYSJ
ndvrSub = SUBJ
ndvrTyp = JAVA
ndvrLoc2Env = TEST
ndvrLoc2Stg = B
ndvrLoc2Sys = SYSJ
ndvrLoc2Sub = SUBJ

Source Match Found:
Unix Source = /u/test/a/sysj/subj
Target Match Found:
Unix Target = /u/test/b/sysj/subj

19:27:05 in GetLongname

Performing short to long name translation.
Short name = CL000000
Long name = Clock2.java

19:27:06 in CopySourceToTarget

Return code from ALLOCATE Function = 0
Allocation of FILEOUT complete: '/u/test/b/sysj/subj/Clock2.java'
Return code from OCOPY function = 0
Copying source from the JFILE ddname.
Copy complete to target: '/u/test/b/sysj/subj/Clock2.java'
Permissions set to 775: '/u/test/b/sysj/subj/Clock2.java'

119:27:13 in DeleteUnixSource

Determining if any Unix files need to be deleted.
Source Action for this source location: KEEP

19:27:13 in ProcessComponentLists

19:27:14 in FindMatchingRowInJMAP

State of symbolic variables at start of FindMatchingRowInJmap.

ndvrEnv = TEST
ndvrStg = A
ndvrSys = SYSJ
ndvrSub = SUBJ
ndvrTyp = JEXEC
ndvrLoc2Env = TEST
ndvrLoc2Stg = B
ndvrLoc2Sys = SYSJ
ndvrLoc2Sub = SUBJ

Source Match Found:
Unix Source = /u/test/a/sysj/subj/classes
Target Match Found:

Unix Target = /u/test/b/sysj/subj/classes

19:27:14 in GetLongname

Performing short to long name translation.
Short name = CLO00001
Long name = Clock2.class

Return code from ALLOCATE Function = 0
Allocation for JFILE complete:
CIGT.STEVE.ESTEST1.CLASSES(CLO00001)

19:27:16 in CopySourceToTarget

Return code from ALLOCATE Function = 0
Allocation of FILEOUT complete:
'/u/test/b/sysj/subj/classes/Clock2.class'
Return code from OCOPY function = 0
Copying source from the JFILE ddname.
Copy complete to target:
'/u/test/b/sysj/subj/classes/Clock2.class'
Permissions set to 775: '/u/test/b/sysj/subj/classes/Clock2.class'

19:27:22 in DeleteUnixSource

Determining if any Unix files need to be deleted.
Source Action for this source location: KEEP

19:27:22 in FindMatchingRowInJMAP

1

State of symbolic variables at start of FindMatchingRowInJmap.

ndvrEnv = TEST
ndvrStg = A
ndvrSys = SYSJ
ndvrSub = SUBJ
ndvrTyp = JLIST
ndvrLoc2Env = TEST
ndvrLoc2Stg = B
ndvrLoc2Sys = SYSJ
ndvrLoc2Sub = SUBJ

Source Match Found:
Unix Source = /u/test/a/sysj/subj/listings
Target Match Found:
Unix Target = /u/test/b/sysj/subj/listings

19:27:22 in GetLongname

Performing short to long name translation.
Short name = CLO00000
Long name = Clock2.java

Return code from ALLOCATE Function = 0
Allocation for JFILE complete:
CIGT.STEVE.ESTEST1.LISTINGS(CLO00000)

19:27:24 in CopySourceToTarget

Return code from ALLOCATE Function = 0
Allocation of FILEOUT complete:
'/u/test/b/sysj/subj/listings/Clock2.java'

```
Return code from OCOPY function = 0
Copying source from the JFILE ddname.
Copy           complete           to           target:
'/u/test/b/sysj/subj/listings/Clock2.java'
Permissions set to 775: '/u/test/b/sysj/subj/listings/Clock2.java'
```

```
-----
19:27:31 in DeleteUnixSource
-----
```

```
Determining if any Unix files need to be deleted.
Source Action for this source location: KEEP
```

```
-----
19:27:31 Ending the Cloud 9 USS Map and Copy Process.
-----
```

Clock2.java Move Action Summary Report

Add E-JDK Processors	62	EPRCSFTP	8
Batch Admin	38, 60	EPRCSMAK8 , 17, 18, 22, 27	
C9JPCJRC	41, 53, 54, 59	Example	28
C9JPCJRM	41, 53, 54, 59	FTP server	12
C9JPCJVC	41, 51, 54, 60, 70	JAR31, 32, 37, 41, 49, 53,	
C9JPCJVM ...	32, 41, 52, 54,	57, 58, 59, 61, 62, 65	
	60, 70	JAR Compile Shell	53
C9JPCJVP	41, 51, 53, 54,	JAR Output Mapping Rules	
	59, 60, 70	53
C9JPCMAP ...	32, 41, 49, 54	JAVA . 31, 32, 37, 41, 49, 51,	
C9JPJADD	31, 33, 61, 62, 68	52, 53, 55, 56, 57, 58,	
C9JPJEE ..	31, 33, 63, 64, 68	60, 61, 62, 63, 65, 66,	
C9JPJINV	31, 33, 38, 55, 60,	69, 70, 71, 72	
	68	Java Compile Shell	51
C9JPJSLR	31, 33, 65, 68	<i>Java Output Mapping Rules</i>	
C9JPJUNX	31, 33, 66, 67, 68	52
C9JPPDEL	31, 41, 42, 54,	JCL 7, 8, 31, 35, 37, 38, 55,	
	57, 58, 62	61, 62, 63, 65, 66	
C9JPPGEN ...	31, 41, 43, 54,	JEXEC	49, 70
	57, 58, 62	JLIST	49, 71
C9JPPJAV	31, 41, 44, 45,	<i>Life Cycle Mapping Rules</i> . 49	
	51, 52, 53, 54, 58, 61, 70	Load EE Control Records . 64	
C9JPPJDL	31, 41, 45, 46,	NDVRFTP1 8, 10, 15, 16,	
	54, 58, 62	17, 19, 22, 27, 28	
C9JPPJMV ...	31, 41, 47, 58,	NDVRMAKE .. 8, 10, 17, 18,	
	61	22, 25, 27	
Case Sensitive	13	processor .. 33, 37, 42, 43, 44,	
Classpath	32, 51	45, 47, 49, 51, 52, 53,	
Create USS E-JDK		54, 69, 72	
Directories.....	67	REXX Script	8, 19, 22
Define E-JDK Types to		Target Platform	12
Cloud 9 SLR	65	Unix.....	31, 66
E-FTP translator.....	7		
