



This is the tutorial for IBM Debug Tool for z/OS[®], one of the IBM zSeries[®] problem determination tools.



Scenarios for starting the debugger for LE batch programs

- Trigger the debugger with an LE TEST option in JCL, and
 1. Display the debugger on a graphical user interface
 2. Display the debugger on a TIM (Debug Tool terminal interface manager) terminal through a session manager
 3. Display the debugger on a dedicated TIM terminal
 4. Display the debugger on a dedicated non-TIM terminal
- Trigger the debugger with the LE 'user exit data set' facility, and
 5. Display the debugger on a graphical user interface
 6. Display the debugger on a TIM terminal through a session manager
 7. Display the debugger on a dedicated TIM terminal
 8. Display the debugger on a dedicated non-TIM terminal

Running and debugging an LE batch program under TSO

- Use the 'Debug Tool setup file' online panels to run the program and display the debugger on the TSO terminal

In this section, you will see a scenario for starting Debug Tool for a batch application. In this scenario the application runs as a batch job, and a TEST option is coded in the JCL that will trigger the debugger when the job runs. The debugger displays on a Debug Tool terminal interface manager terminal session that you access through a session manager. You can skip this section if you do not plan to use Debug Tool this way on your system.

Debug in batch using a TEST option in JCL and a dedicated Debug Tool TIM terminal



- **Description**
 - A TEST(...) option is coded in the application's run-time JCL to trigger the debugger
 - The debugger displays on a Debug Tool TIM (terminal interface manager) terminal session that is accessed through a session manager
- **This method can be used:**
 - To debug LE programs running in batch jobs
 - including programs that access IMS™, DB2®, or other types of databases
 - and non-LE programs that run in the call chain under an LE program
 - If the Debug Tool TIM feature is installed on your system
- **When not to use this method:**
 - If TIM sessions are not installed, use a dedicated non-TIM terminal instead
 - On older versions of z/OS this method may not work for IMS programs. In that case, use the Debug Tool 'user exit data set' facility instead.

3

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

This is a commonly used, simple method to start the debugger. The debugger is displayed on a Debug Tool terminal interface manager terminal. You set a trigger, to start the debugger when the application runs, by coding a Language Environment (LE) TEST option in the program's run-time JCL.

You can use this method to debug batch LE programs, including programs that access DB2, IMS, and other types of databases. Even non-LE subroutines can be debugged using this method, as long as there is at least one LE program higher in the call chain. This method assumes that the application is running in a batch job, and that the Debug Tool terminal interface manager is installed on your system.

This is generally the best method when a TIM terminal is used. However, you cannot use this method if TIM terminal sessions have not been defined on your system's network. If not, consider displaying the debugger on a non-TIM terminal or a GUI debugger instead. Also, if you are running on an older version of z/OS (1.7 or earlier), a TEST option may not trigger the debugger for IMS batch programs. In that case, the 'user exit data set' method could be used instead.

Select Debug Tool from the session manager menu



```
Session Manager          Software University Main Menu          17/12/2010 10:28:16
LU  TCP00055          DNET074
Sel  Description          APPLID  Status  ACB          LOGMODE  NODE
-----
PF1  TSO session 1          TSOA    ACTIVE  ISZ004      D4A32792
PF2  TSO session 2          TSOB    AVAIL
PF3  CICS AOR1 - CICS A    CICSACB1 AVAIL
PF4  CICS AOR2 - CICS B    CICSACB2 AVAIL
PF5  CICS AOR3 - CICS C    CICSACB3 AVAIL
PF6  CICS AOR4 - CICS D    CICSACB4 AVAIL
PF7  CICS AOR5 - CICS E    CICSACB5 AVAIL
S PF8  DEBUG TOOL          EQASESSM AVAIL
PF9  IMS A                  IMSACB   UNAVAIL
PF10 IMS B                  IMSACB2  AVAIL
PF23 Disc from Session Manager  CMD
PF24 Logoff from Session Manager  CMD
810 System Management Menu      AVAIL

HARDCOPY = /H
====> |

ESC: Menu @@  Bwd <<  Fwd >>  Cut *c  Paste *p  Push />  Enter
```

Your session manager will look different than this one, and will have different menu options

Note: If Debug Tool is not already defined on your menu, you may be able to add a session definition that connects to application EQASESSM

4 IBM Debug Tool for z/OS tutorial © 2012 IBM Corporation

In this scenario, terminal session manager software is used to switch between TSO and a Debug Tool session. A terminal session manager may be installed on your system to give users an easy way to access different applications or systems from a single terminal. There are different terminal session manager products available. This example uses one called IBM session manager. There may be a different one on your system, so your session manager will probably look different than the one shown here, and the menu options will be different. Even so, most session managers provide a menu of systems that can be selected, or a special "hotkey" you can use to switch between sessions.

In this example the user is already logged on to TSO. If your session manager shows the debugger as one of the options, select it. If your menu does not have an option for the debugger, depending on your authorizations, you may be able to define a new session for it. If you define your own session, try connecting to an APPL with a name of EQASESSM, which is the default network name for the Debug Tool application. If your session manager does not give you a way to connect to Debug Tool, consider contacting your help desk or the Debug Tool installer so that the session manager can be configured correctly.

Log on to the Debug Tool TIM session



```
DEBUG TOOL TERMINAL INTERFACE MANAGER
Terminal TRMDT007

EQAY001I Session manager escape ==>
EQAY001I Enter your userid: TSS16
EQAY001I Password: █
```

Log on to Debug Tool with your User ID and password

Enter

PF3=EXIT

5 IBM Debug Tool for z/OS tutorial © 2012 IBM Corporation

That opened a terminal session for the debugger. Notice the title "Debug Tool terminal interface manager" on the Debug Tool terminal. That indicates that it is a TIM terminal. If your Debug Tool terminal does not have this title, then you either have not correctly connected to it, or you have a non-TIM terminal session. If the TIM feature is not installed on your system, take one of the tutorial sections that describes how to debug using a non-TIM terminal, instead of this section.

Next, log on to with your ID and password. Typically, this is the same ID and password you use to log onto TSO. Do not be confused by thinking that you are logging onto TSO twice. You are not logging onto TSO here, you are logging on to Debug Tool. The ID and password are typed in, and enter is pressed.

The Debug Tool TIM terminal is ready



```
DEBUG TOOL TERMINAL INTERFACE MANAGER

EQAY001I Session manager escape ==> █

EQAY001I Terminal TRMDT007 connected for user TSS16
EQAY001I Ready for Debug Tool

PF3=EXIT PF10=Edit LE options data set PF12=LOGOFF
```

Hot key to your session manager

6 IBM Debug Tool for z/OS tutorial © 2012 IBM Corporation


At this point your Debug Tool terminal is ready to receive a debugging session. Next the TSO terminal session will be selected. Typically, there is a special "hot key" you can press to jump to the session manager menu, such as a specific PA key or a key combination. Press whatever hot key or key combination you use to re-display your session manager menu.

Select the TSO session



```
Session Manager          Software University Main Menu          17/12/2010 09:03:19
LU  TCP00055                DNET074
Sel  Description                APPLID  Status  ACB      LOGMODE  NODE
-----
S PF1  TSO session 1            TSOA    ACTIVE  ISZ004   D4A32792
- PF2  TSO session 2            TSOB    AVAIL
- PF3  CICS AOR1 - CICS A      CICSACB1 AVAIL
- PF4  CICS AOR2 - CICS B      CICSACB2 AVAIL
- PF5  CICS AOR3 - CICS C      CICSACB3 AVAIL
- PF6  CICS AOR4 - CICS D      CICSACB4 AVAIL
- PF7  CICS AOR5 - CICS E      CICSACB5 AVAIL
- PF8  DEBUG TOOL              EQASESSM ACTIVE
- PF9  IMS A                    IMSACB  UNAVAIL
- PF10 IMS B                    IMSACB2 AVAIL
- PF23 Disc from Session Manager  CMD
- PF24 Logoff from Session Manager  CMD
- 810 System Management Menu      AVAIL

HARDCOPY = /H

====> _____ 
ESC: Menu @@  Bwd <<  Fwd >>  Cut *c  Paste *p  Push />
```

Select the TSO session.

Open the JCL that will be used to run the batch program

```
Menu Utilities Compilers Options Status Help
OS/390 Primary Option Menu
Option ==> 2
0 Settings Terminal and user parameters User ID . . : TSS16
1 View Display source data or listings Time. . . : 09:53
2 Edit Create or change source data Terminal. : 3278A
3 Utilities Perform utility functions Screen. . : 1
4 Foreground Interactive language processing Language. : ENGLISH
5 Batch Submit job for language processing Appl ID . : ISR
6 Command Enter TSO or Workstation commands TSO logon : TPROC02
7 Dialog Test Perform dialog testing TSO prefix: TSS16
8 LM Facility Library administrator functions System ID : STLABF6
9 IBM Products IBM program development products MVS acct. : *
10 SCLM SW Configuration Library Manager Release . : ISPF 6.1
11 Workplace ISPF Object/Action Workplace
12 OS/390 System OS/390 system programmer applications
13 OS/390 User OS/390 user applications
DB DataBase Interactive Database Functions
G SWS/STL Display SWS/STL Selection Panel
Enter X to Terminate using log/list defaults
```



8

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

In TSO, open the JCL that runs the application you want to debug. In this example, the ISPF editor is selected.

Open the JCL that will run the batch program



Menu RefList RefMode Utilities Workstation Help

Command ==> _____

ISPF Library:

Project . . . TSS16

Group . . . ADLAB

Type . . . JCL

Member . . . XSAM (Blank or pattern for member selection list)

Other Partitioned, Sequential or VSAM Data Set, or z/OS UNIX file:

Name . . . _____ +

Volume Serial _____ (If not cataloged)

Workstation File:

File Name . . . _____

Options

Initial Macro . . . _____ - Confirm Cancel/Move/Replace

Profile Name . . . _____ - Mixed Mode

Format Name . . . _____ - Edit on Workstation

Data Set Password . . . _____ - Preserve VB record length

Record Length . . . _____

Enter

Open the JCL that will run the application in batch

9 IBM Debug Tool for z/OS tutorial © 2012 IBM Corporation

Open the JCL in the editor.

This is the JCL before adding a TEST option



```
File Edit Edit_Settings Menu Utilities Compilers Test Help
EDIT      TSS16.ADLAB.JCL (XSAM) - 01.00          Columns 00001 00072
Command ==> |                                     Scroll ==> CSR
***** Top of Data *****
000001 //TSS16D JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000002 //      MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000003 //*
000004 //PRINT1 EXEC PGM=IDCAMS
000005 //SYSPRINT DD SYSOUT=*
000006 //FILE DD DSN=&SYSUID..ADLAB.FILES(CUST2FA),DISP=SHR
000007 //SYSIN DD *
000008 PRINT INFILE(FILE) COUNT(1)
000009 //*
000010 //RUNSAM1 EXEC PGM=SAM1,REGION=4M
000011 //***** DD'S FOR DEBUG TOOL *****
000012 //** //CEEOPPTS DD *
000013 //** TEST(,,VTAM%USERID:)
000014 //** //INSPLOG DD SYSOUT=*
000015 //** //EQADDEBUG DD DSN=&SYSUID..ADLAB.SYSDEBUG,DISP=SHR
000016 //** //      DD DSN=&SYSUID..ADLAB.EQALANGX,DISP=SHR
000017 //** //INSPREF DD DSN=&SYSUID..ADLAB.DTPREF,DISP=SHR
000018 //*****
000019 //STEPLIB DD DISP=SHR,DSN=&SYSUID..ADLAB.LOAD
```

10

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

This is JCL that is used to run the application program in this example. It is shown here before any changes have been made.

Code a TEST option in a CEEOPTS DD or as a PARM on the EXEC statement



```
File Edit Edit_Settings Menu Utilities Comp
EDIT      TSS16.ADLAB.JCL (XSAM) - 01.00          Columns 00001 00072
Command ==> |                                     Scroll ==> CSR
***** Top of Data *****
000001 //TSS16D JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000002 //          MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000003 //*
000004 //PRINT1 EXEC PGM=IDCAMS
000005 //SYSPRINT DD SYSOUT=*
000006 //FILE DD DSN=&SYSUID..ADLAB.FILES(CUST2FA),DISP=SHR
000007 //SYSIN DD *
000008 PRINT INFILE(FILE) COUNT(1)
000009 //*
000010 //RUNSAM1 EXEC PGM=SRM1,PARM='/TEST(,,VTAM%TSS16:)',REGION=4M
000011 //***** DD'S FOR DEBUG TOOL *****
000012 //CEEOPTS DD *
000013 TEST(,,VTAM%TSS16:)
000014 //** //INSPLG DD SYSOUT=*
000015 //** //EQADDEB DD DSN=&SYSUID
000016 //** //          DD DSN=&SYSUID
000017 //** //INSPREF DD DSN=&SYSUID
000018 //*****
000019 //STEPLIB DD DISP=SHR,DSN=&SYSU
```

In the JCL, code a TEST option

VTAM%userid: denotes a TIM terminal

Code a TEST option:
- on the EXEC statement, or
- in a CEEOPTS DD statement
(But not both)

11

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

To set a trigger for the debugger, code a Language Environment TEST option in the JCL. There are two ways to do it. Depending on how the JCL is coded, you may be able to code a TEST option directly on the EXEC statement that runs the program. Or you can add a special DD statement called CEEOPTS, and code a TEST option as input data in the CEEOPTS DD. Although both methods are shown in this example, only code one or the other.

Notice how the TEST option is coded. After the third comma, it has "VTAM%user ID:". The VTAM® keyword specifies that the Debug Tool terminal interface manager will be used. The user ID is coded after the % sign. This is the user ID that was used to log onto the Debug Tool TIM terminal, and it will be used by Debug Tool to locate the terminal.

Submit the JCL to run the batch job



```
File Edit Edit_Settings Menu Utilities Compilers Test Help
EDIT TSS16.ADLAB.JCL (XSAM) - 01.16 Columns 00001 00072
Command ==> SUBMIT Scroll ==> CSR
***** Top of Data *****
000001 //TSS16D JOB (ACCTG),'IBM TOOLS WORKSHOP',REGION=4M,CLASS=A,
000002 // MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
000003 //*
000004 //PRINT1 EXEC PGM=IDCAMS
000005 //SYSPRINT DD SYSOUT=*
000006 //FILE DD DSN=&SYSUID..ADLAB.FILES(CUST2FA),DISP=SHR
000007 //SYSIN DD *
000008 PRINT INFILE(FILE) COUNT(1)
000009 //*
000010 //RUNSAM1 EXEC PGM=SAM1,REGION=4M
000011 //***** DD'S FOR DEBUG TOOL *****
000012 //CEEOPDS DD *
000013 TEST(,,VTAM%TSS16:)
000014 //** //INSPLDG DD SYSOUT=*
000015 //** //EQADEBUG DD DSN=&SYSUID..ADLAB.SYSDEBUG,DISP=SHR
000016 //** // DD DSN=&SYSUID..ADLAB.EQUALANGX,DISP=SHR
000017 //** //INSPREF DD DSN=&SYSUID..ADLAB.DTPREF,DISP=SHR
000018 //*****
000019 //STEPLIB DD DISP=SHR,DSN=&SYSUID..ADLAB.LOAD
```

Submit the job

Enter

12 IBM Debug Tool for z/OS tutorial © 2012 IBM Corporation

The job is ready to run. The JCL is submitted to batch.

```

File Edit Edit_Settings Menu Utilities Compilers Test Help
EDIT      TSS16.ADLAB.JCL (XSAM) - 01.16          Columns 00001 00072
Command ==> |                                     Scroll ==> CSR
***** Top of Data *****
000001 //TSS16D JOB (ACCTG), 'IBM TOOLS WORKSHOP', REGION=4M, CLASS=A,
000002 //          MSGCLASS=H, NOTIFY=&SYSUID, MSGLEVEL=(1,1)
000003 //*
000004 //PRINT1 EXEC PGM=IDCAMS
000005 //SYSPRINT DD SYSOUT=*
000006 //FILE DD DSN=&SYSUID..ADLAB.FILES(CU
000007 //SYSIN DD *
000008 PRINT INFILE(FILE) COUNT(1)
000009 //*
000010 //RUNSAM1 EXEC PGM=SAM1, REGION=4M
000011 //***** DD'S FOR DEBUG TOOL *****
000012 //CEEOPPTS DD *
000013 TEST(,,VTAM%TSS16:)
000014 //* //INSPLOG DD SYSOUT=*
000015 //* //EQADEBUG DD DSN=&SYSUID..ADL
IKJ56250I JOB TSS16D (JOB09299) SUBMITTED
***
    
```

When the step runs, the TEST option is processed, and the debugger displays on the Debug Tool terminal

Select the Debug Tool terminal

Hot key to your session manager

When a step runs that has a TEST option specified, Language Environment receives the TEST option and starts Debug Tool. The debugger is displayed in the TIM terminal session. Press whatever hot key or key combination is needed to display the session manager.

Select the Debug Tool session



```
Session Manager                Software University Main Menu                17/12/2010 12:01:15
LU  TCP00055                    DNET074
Sel  Description                APPLID  Status  ACB      LOGMODE  NODE
-----
- PF1 TSO session 1              TSOA    ACTIVE  ISZ004   D4A32792
- PF2 TSO session 2              TSOB    AVAIL
- PF3 CICS AOR1 - CICS A         CICSACB1 AVAIL
- PF4 CICS AOR2 - CICS B         CICSACB2 AVAIL
- PF5 CICS AOR3 - CICS C         CICSACB3 AVAIL
- PF6 CICS AOR4 - CICS D         CICSACB4 AVAIL
- PF7 CICS AOR5 - CICS E         CICSACB5 AVAIL
- PF8 DEBUG TOOL                EQASESSM ACTIVE
- PF9 IMS A                      IMSACB  UNAVAIL
- PF10 IMS B                     IMSACB2 AVAIL
- PF23 Disc from Session Manager  CMD
- PF24 Logoff from Session Manager CMD
- 810 System Management Menu      AVAIL

HARDCOPY = /H
====> |
ESC: Menu @@  Bwd <<  Fwd >>  Cut *c  Paste *p  Push />
```



14

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

Select the Debug Tool TIM terminal.

The debugger automatically displays on the TIM terminal

```

COBOL  LOCATION: SAM1 initialization
Command ==>
Scroll ==> PAGE
MONITOR +-----1-----2-----3-----4-----5-----6- LINE: 0 OF 0
***** TOP OF MONITOR *****
***** BOTTOM OF MONITOR *****

SOURCE: SAM1 +-----1-----2-----3-----4-----5----- LINE: 1 OF 467
1 *****
2 * PROGRAM: SAM1
3 * Sample program for the ENTERPRISE COBOL Compiler
4 *
5 * AUTHOR : Doug Stout
6 * IBM PD TOOLS

LOG 0-----1-----2-----3-----4-----5----- LINE: 32 OF 34
0032 IBM Debug Tool Version 10 Release 1 Mod 0
0033 12/31/2009 00:44:02 AM
0034 5655-V50: Copyright IBM Corp. 1992, 2009
PF 1:? 2:STEP 3:QUIT 4:LIST 5:FIND 6:AT/CLEAR
PF 7:UP 8:DOWN 9:GO 10:ZOOM 11:ZOOM LOG 12:RETRIEVE
    
```

The batch program can be controlled from this Debug Tool session

And you can select your TSO session to continue working in TSO

The batch job is running on the host z/OS system, and the debugger is communicating with the Debug Tool terminal. You can control the program from the Debug Tool terminal, and you can also select your TSO session to continue working in TSO at the same time. Be aware that the TSO session is no longer needed. You could log off from TSO and continue to debug, because Debug Tool is controlling the application that is running in its own batch address space.

Choosing between the CEEOPTS DD or EXEC parm



- There are two ways to code a TEST option in JCL to start Debug Tool:
 - in a CEEOPTS DD, or
 - on the EXEC statement

- Comparing the two methods:
 - Both are easy, but CEEOPTS can be easier because:
 - it is simpler to cut and paste into JCL without making a syntax mistake
 - It can be used with a JCL PROC without changing the PROC
 - It will trigger the debugger when the first LE program runs, even if it is a subprogram
 - but a TEST option on the EXEC statement will *only* trigger if the *main* program is an LE program

There are two ways to code a TEST option in JCL to start Debug Tool. On the EXEC statement, or with a CEEOPTS DD statement. For most people, the CEEOPTS DD method is a little easier. Here is why.

When you use the CEEOPTS method, it can be easier to “cut and paste” a TEST option into JCL from other JCL where you have coded it before. Also, if a cataloged JCL PROC is used, you may not be able to code a parm on the EXEC statement, because it is embedded in the PROC. With a CEEOPTS DD, it is easy to code it in your JCL to make it take effect in any step you want in your PROC. You will see an example of that in a minute.

Finally, when you use CEEOPTS, Debug Tool will trigger when the first LE program is called, even if it is a subprogram. That is an advantage over the other method, because when you code a TEST option on the EXEC statement, Debug Tool will only trigger if the main program is an LE program.

A CEEOPTS DD with a TEST(...) option can be used to trigger the debugger



- When LE initializes, it looks for a CEEOPTS DD statement
 - If CEEOPTS is present in the JCL step, LE reads it to retrieve run-time options
 - If a TEST(...) option is present, LE starts the debugger
- Considerations
 - LE version 1.7 or later is required (shipped with z/OS V1.7)
 - LE will *not* read a CEEOPTS DD :
 - In older versions of z/OS:
 - with IMS applications (batch or IMS/TM)
 - when the LE Library Routine Retention facility (LRR) is used

When you run an LE program, which includes programs compiled with compilers such as Enterprise COBOL and Enterprise PL/I, Language Environment initializes when the program starts. When LE initializes, as part of its normal processing, it looks for a CEEOPTS DD statement. If it finds one, it reads it to retrieve run-time options. When you pass a TEST option to LE, it will start Debug Tool.

But there are a couple of considerations and restrictions. First, you must be at LE version 1.7 or later. Since LE is shipped with z/OS, typically that means that the z/OS operating system must be at version 1.7 or later.

LE will not read a CEEOPTS DD statement with IMS applications on older versions of z/OS. So, if you want to debug an IMS batch application and you are running on an older system, you cannot use a TEST option in JCL to do it. In that case, use the 'Debug Tool user exit data set facility' to trigger the debugger.

- Example:

```
//STEP5 EXEC PGM=MYPROG
//CEEOPTS DD *
TEST(,,,VTAM%userid:)
```

- where *userid* = your user ID
- *VTAM%user-id:* directs the debugger to use the terminal interface manager. It will connect to the TIM terminal where *userid* is logged on.

- You can use a file instead of in-stream data:

```
//CEEOPTS DD DSN=MY.CEEOPTS.FILE,DISP=SHR
```

- When executing a JCL PROC, use the *stepname.CEEOPTS* syntax to name the step you want to debug:

```
//RUNPROC EXEC PROC99
//ASTEP.CEEOPTS DD *
TEST(,,,VTAM%userid:)
```

Here are examples of coding CEEOPTS DD statements in your JCL. In the first example, CEEOPTS is coded as one of the DD statements in a step. The TEST option can be coded as in-stream data as is shown.

The second example shows that you can have your TEST option coded in a file if you find that more convenient.

The third example shows how you can code a CEEOPTS DD statement when the JCL executes a PROC. In this case, notice that the EXEC statement runs a PROC named PROC99. The “ASTEP.CEEOPTS” DD statement will add the CEEOPTS DD to a proc step named ASTEP. When it is done this way, you do not have to make any changes to the PROC itself.

A TEST(...) option on the EXEC statement can be used to trigger the debugger

- When LE initializes, it examines the PARM string on the EXEC statement for LE run-time options
 - A slash character (/) in the parm string signals that LE options are present
 - If an LE TEST(...) option is present, LE starts the debugger
- For COBOL programs, code LE options after the last slash

COBOL

```
//STEP5 EXEC PGM=COBPROG,  
// PARM= '/TEST( , , ,VTAM%userid: )'
```

- For non-COBOL LE programs, code LE options before the first slash

PL/I, C/C++, LE Assembler

```
//STEP5 EXEC PGM=MYPROG,  
// PARM= 'TEST(ALL, , ,VTAM%userid: ) /'
```

There are two ways to code a TEST option in JCL. Either use a CEEOPTS DD statement as you have already seen, or code a TEST option on the EXEC statement.

Here is how to code a TEST option on the EXEC statement. When LE initializes, it examines the PARM string on the EXEC statement for LE run-time options and looks for a slash. For COBOL programs, everything after the last slash is taken as LE options. For non-COBOL LE programs, everything before the first slash is taken as LE options. In either case, you must code a slash at the appropriate location in your PARM string.

Here is an important restriction. LE will examine the PARM string from the EXEC statement only if the main program is an LE program. That means that you cannot use this method to start Debug Tool if the main program is a non-LE program.

The slash (/) character is a separator between program parameters and LE options

- You can code program parameters together with a TEST option
- These examples pass the string 'ABC,1234' to the program
 - LE options and the slash (/) are stripped off, and only the remaining string is passed to the application program

- For a COBOL program, **code LE options after the last slash:**

```
//STEP5 EXEC PGM=COBPROG,
// PARM= 'ABC,1234/TEST(,,,VTAM%userid:)'
```

- For a PL/I, C/C++, or LE assembler program, **code LE options before the first slash:**

```
PL/I, C/C++, LE Assembler:
//STEP5 EXEC PGM=PLIPROG,
// PARM= 'TEST(ALL,,,VTAM%userid:)/ABC,1234'
```

When you code a TEST option on the EXEC statement, you can still code parameters for your program. Here are examples that pass the data string "ABC,1234" to the application program as a parameter.

Remember that for COBOL programs, all parameters coded after the last slash are taken as LE options. So to pass parameters to the program, code them all before the last slash.

For non-COBOL LE programs, all parameters coded before the first slash are taken as LE options. So to pass parameters to your program, code them all after the first slash.

In either case, LE strips its options and the slash out of the parm string before control is passed to the application program. So the LE options are not received by your program.

EXEC parm considerations



- z/OS limits the JCL parameter string to 100 bytes
 - Adding the TEST option can make the total length too long
- If the parm string becomes too long, you can:
 - Use a different method to trigger Debug Tool, or
 - Truncate the user portion of the parm string, and then add the missing data manually with the debugger after the program starts
- If the parm string becomes too long to fit on a line, it can be continued to the next line using this syntax
 - Example:

```
//RUNSAM1 EXEC PGM=SAM1,  
//  PARM=( 'THIS,IS AN,EXAMPLE,OF,A,VERY,LO',  
//    'NG,PARAMETER/TEST(,,,VTAM%USRID01:)',  
//      REGION=4M
```

z/OS limits JCL parameters to a total of 100 bytes. It is possible to run into a situation where adding a TEST option will make the total length exceed 100 bytes.

If the parameter string becomes too long, you have a couple of options. First, you can use a different method to start Debug Tool. Either use a CEEOPTS DD statement, or use the 'Debug Tool user exit data set' facility. Another option is to truncate the user portion of the string, and then add the missing data manually using the debugger after the program starts.

If the string becomes too long to fit on one line, it can be continued to the next line. An example is shown for the syntax used to code a long parameter string that spans multiple lines.

Coding a TEST option with a batch DB2 application



- Do not code a TEST option on the EXEC statement in DB2 batch jobs if the JCL executes program IKJEFTxx as in this example
- A TEST option will not trigger the debugger with program IKJEFTxx, because it is not an LE program
 - Instead, use a CEEOPTS DD or the 'Debug Tool user exit data set' facility

- or optionally, the TEST option can be coded in the DB2 RUN parms:

```
//*          RUN DB2 PROGRAM
//STEP6 EXEC PGM=IKJEFT01,DYNAMNBR=20,COND=(4,LT)
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD *
  DSN SYSTEM(DSNA)
  RUN  PROGRAM(PHONEP01) PLAN(PHONEP01) -
      PARS('/TEST(,,VTAM%USRID99:)' ) -
      LIB('DNET603.ADLAB.LOAD')
  END
//*
```

COBOL example
(slash before TEST)

Do not code a TEST option on the EXEC statement when running a DB2 batch application that executes program IKJEFT01. The debugger will not be triggered, since that program is not an LE program. Instead, consider using a CEEOPTS DD. Or optionally, a TEST option can be coded in the run parms in SYSTSIN as in the example.

Coding a TEST option with a batch IMS application



- Do not code a TEST option on the EXEC statement for IMS batch jobs if the JCL executes program DFSRRCxx as in this example
- A TEST option will not trigger the debugger with program DFSRRCxx, because it is not an LE program
 - Instead, use a CEEOPTS DD or the 'Debug Tool user exit data set' facility

- Example of batch IMS JCL:

```
//*  
//*          RUN IMS PROGRAM  
//STEP6 EXEC PGM=DFSRRC00,  
//          PARM=( 'DLI,IMSPROG,TDIMSP,200,,,,,,,,,N' )  
//DFSRESLB DD DSN=IMS.RESLIB,DISP=SHR  
//IMS      DD DSN=FMN.PSBLIB,DISP=SHR  
//CEEOPTS DD *  
TEST(,,VTAM%USRID99:)  
.  
.  
.
```

23

IBM Debug Tool for z/OS tutorial

© 2012 IBM Corporation

Also, do not code a TEST option on the EXEC statement when running an IMS batch application that executes program DFSRRC00. The debugger will not be triggered, since that program is not an LE program. Instead, consider using a CEEOPTS DD statement or the 'Debug Tool user exit data set' facility.

That is the end of this section, which described starting the debugger for a program running in batch, triggering the debugger by coding a TEST option in JCL, and displaying the debugger on Debug Tool TIM terminal.

Feedback



Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send email feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_DTv12s06bDebuggingBatchSMTimTest.ppt

This module is also available in PDF format at:
[../DTv12s06bDebuggingBatchSMTimTest.pdf](http://DTv12s06bDebuggingBatchSMTimTest.pdf)

You can help improve the quality of IBM Education Assistant content by providing feedback.



Trademarks, copyrights, and disclaimers

IBM, the IBM logo, ibm.com, DB2, IMS, VTAM, z/OS, and zSeries are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "[Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)" at <http://www.ibm.com/legal/copytrade.shtml>

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

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.

© Copyright International Business Machines Corporation 2012. All rights reserved.