IDz Workbench – Debugging z/OS Assembler Applications

- **Jon Sayles, Rational System z Products - jsayles@us.ibm.com**

DevOps/Modernization

Updated July, 2019

IBM Trademarks and Copyrights

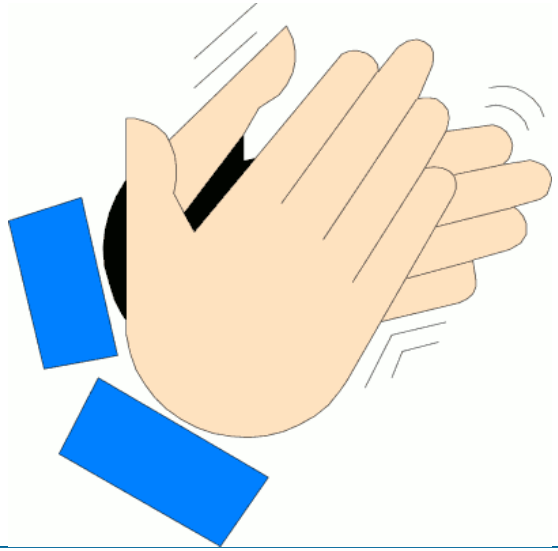
The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. Nothing contained in these materials 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 the applicable license agreement governing the use of IBM software. References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates.

This information is based on current IBM product plans and strategy, which are subject to change by IBM without notice. Product release dates and/or capabilities referenced in these materials may change at any time at IBM’s sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way.

IBM, the IBM logo, the on-demand business logo, Rational, the Rational logo, and other IBM Rational products and services are trademarks or registered trademarks of the International Business Machines Corporation, in the United States, other countries or both. Other company, product, or service names may be trademarks or service marks of others.

Course Contributing Authors

- **Thanks to the following individuals, for assisting with this course:**
  - **Larry England/IBM**
  - **Russ Courtney/IBM**
  - **Doug Stout/IBM**



Course Overview

- **Audience**
  - This course is designed for application developers who have learned or programmed in Assembler, and who need to do z/OS Traditional Development and Maintenance as well as build leading-edge applications using Assembler and Rational Developer for System z.

- **Prerequisites**
  - This course assumes that the student has a basic understanding and knowledge of software computing technologies, and general data processing terms, concepts and vocabulary, as well as a working knowledge of Assembler and z/OS.
  - Knowledge of SQL (Structured Query Language) is assumed for database access is assumed as well.
  - Basic PC and mouse-driven development skills, terms and concepts are also assumed.

UNIT

The IDz Workbench

Topics:

- **Debugging z/OS Assembler Batch Applications**
- Debugging z/OS Assembler Online Applications
- Appendix

Topic Considerations

Note: In this topic you will learn how to debug a Assembler program running on a z/OS mainframe. The screen captures all describe connecting to a public z/OS machine that IBM makes available – during classes.

If you are taking this course through standard IBM services delivery you should be able to use the properties (I/P address, port#s, etc.), logon IDs and passwords that your instructor provides you with.

But you may also be taking this course standalone – and in that case, you will need to speak to your company's Systems Programming staff to learn how to connect and logon.

It goes without saying that the actual file names in the screen captures of mainframe libraries and datasets will vary. So you should focus on the process and steps and "how to" – and don't be perplexed at differences in screen captures.

You also may be using your company's Source Control Management system – to do things like builds, compiles, etc. In that case much of the remote functionality in IDz will be customized and tailored to your company's unique and idiosyncratic procedures and protocols.

Topic Objectives

After completing this unit, you should be able to:

- Describe the concept of source code debugging
  - List the run-times that Debug Tool supports
  - List the steps in preparing a program for debugging
  - Debug a mainframe batch job
    - Describe the run/step/animate options
    - List PF-Keys associated with them
    - Set/unset/inspect conditional and unconditional break-points
    - Set "watch" break-points that halt execution when a value in a variable changes
    - Show how to access the LPEX editor functionality during debugging (such as Perform Hierarchy)
    - Be able to Jump to any given line, and run to a line
    - Show how to change variable values dynamically during debug
    - Show how to set different levels of variable display
    - Monitor specific variables you are interested in
  - Debug a CICS online transaction
    - Discuss the Debug Option setup and configuration requirements for Online Debugging
      - DTCN Profile/View
      - DTCN Transaction
    - Launch a CICS transaction that invokes Debug Tool

Debugging Overview

Face facts: **No one gets it right the first time.**

- Not at the level of production business logic

That's why IBM invented source-level application debuggers, so that you can:

- View program execution, line-by-line
- Verify the value of a variable – during program execution
- Stop and start program execution, and analyze results at the speed that our procedural understanding of the application's execution flow can handle

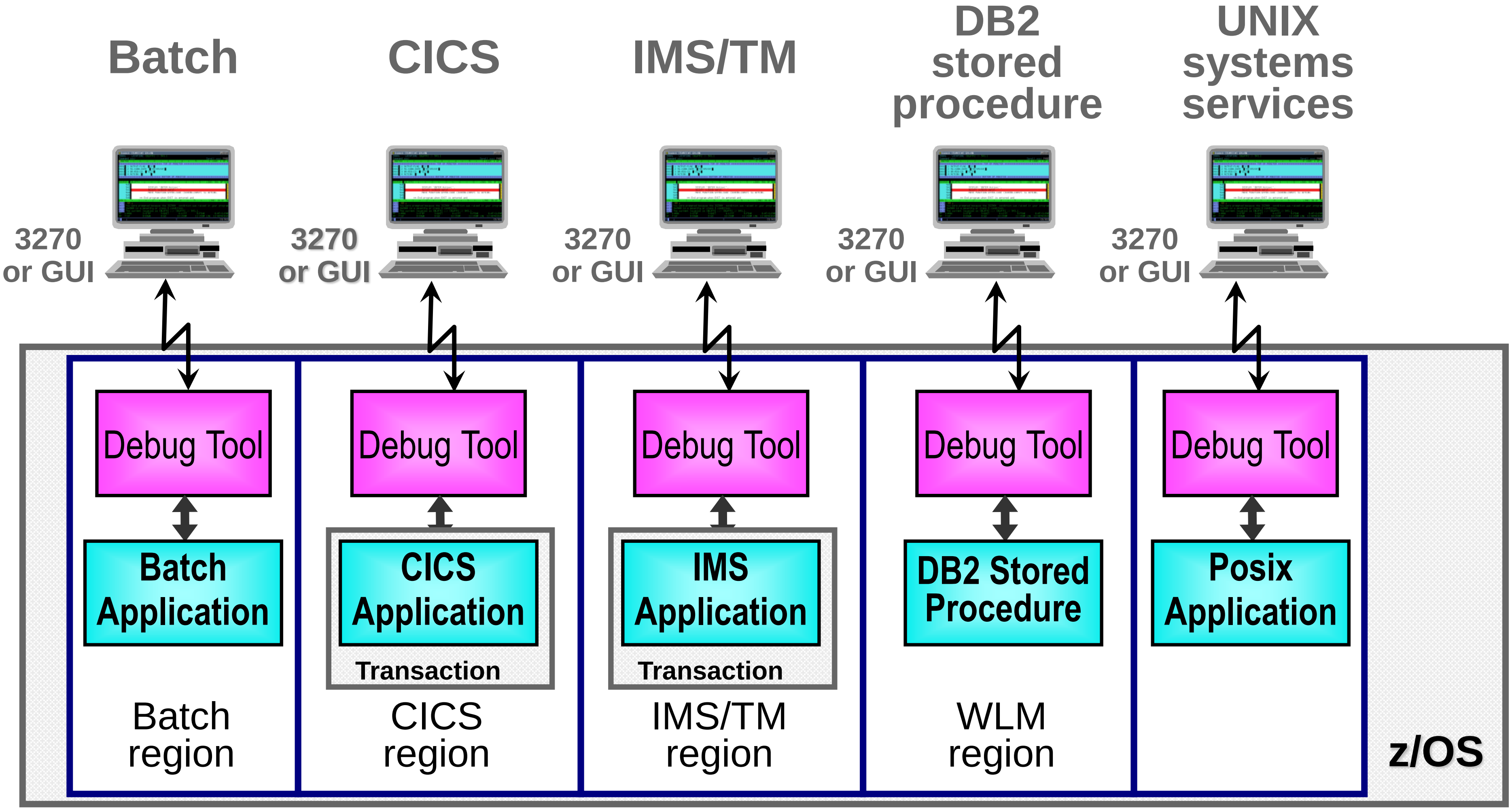
Enter Source-Level Debuggers

- Specifically: **IBM Debug Tool/PD Tools Family**
  - Green-screen (TSO-based) **or IDz/Workstation-based** interface to z/OS-based debugging engines
  - Debug:
    - Online (CICS, or IMS TM)
    - Batch
    - Multiple languages (Assembler, PL/I, COBOL, Java, etc.)
  - Seamless debugging of mixed-language/cross-platform applications
  - Interactive, source-level debugging in IDz with program running on z/OS
  - Display, monitor and alter program variables
  - Set standard types of breakpoints
  - View data in Hex (EBCDIC) or string values
  - Multiple configurable views
  - Ability to make adjustments to the program while debugging

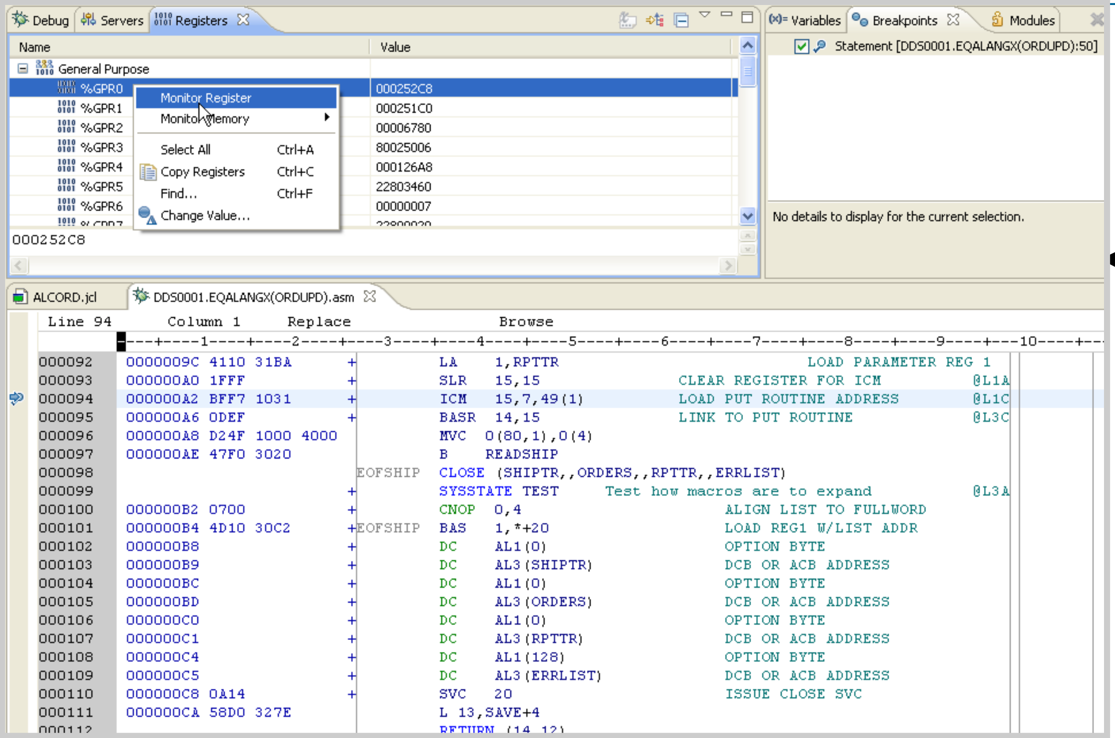
- Debug Tool product web-site: http://www-01.ibm.com/software/awdtools/debugtool/

Debug Tool - Application Environments

One debugging engine, with support for many environments:



IDz Interfacing with Debug Tool



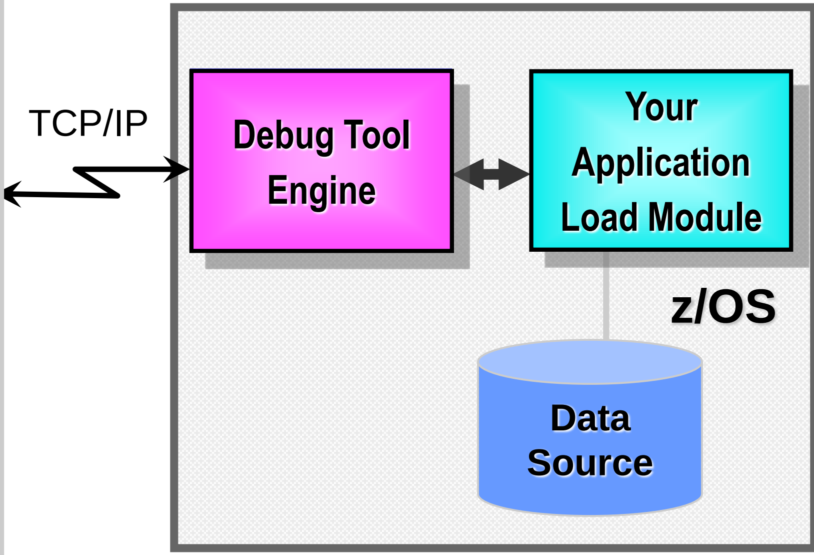
Name	Value
General Purpose	
%GPR0	000252C8
%GPR1	000251C0
%GPR2	00006780
%GPR3	80025006
%GPR4	000126A8
%GPR5	22803460
%GPR6	00000007

000252C8

☑ Statement [DDS0001.EQALANGX(ORDUPD):50]

No details to display for the current selection.

```
 Line 94      Column 1      Replace                 Browse
          ---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----9----+---10----+--
000092    0000009C 4110 31BA      +         LA     1,RPTTR                         LOAD PARAMETER REG 1
000093    000000A0 1FFF           +         SLR    15,15                CLEAR REGISTER FOR ICM         @L1A
000094    000000A2 BFF7 1031      +         ICM    15,7,49(1)           LOAD PUT ROUTINE ADDRESS       @L1C
000095    000000A6 0DEF           +         BASR   14,15                LINK TO PUT ROUTINE            @L3C
000096    000000A8 D24F 1000 4000           MVC   0(80,1),0(4)
000097    000000AE 47F0 3020                B     READSHIP
000098                             EOFSHIP  CLOSE (SHIPTR,,ORDERS,,RPTTR,,ERRLIST)
000099                            +         SYSSTATE TEST     Test how macros are to expand        @L3A
000100    000000B2 0700           +         CNOP   0,4                         ALIGN LIST TO FULLWORD
000101    000000B4 4D10 30C2      +EOFSHIP  BAS    1,*+20                      LOAD REG1 W/LIST ADDR
000102    000000B8                +         DC     AL1(0)                      OPTION BYTE
000103    000000B9                +         DC     AL3(SHIPTR)                 DCB OR ACB ADDRESS
000104    000000BC                +         DC     AL1(0)                      OPTION BYTE
000105    000000BD                +         DC     AL3(ORDERS)                 DCB OR ACB ADDRESS
000106    000000C0                +         DC     AL1(0)                      OPTION BYTE
000107    000000C1                +         DC     AL3(RPTTR)                  DCB OR ACB ADDRESS
000108    000000C4                +         DC     AL1(128)                    OPTION BYTE
000109    000000C5                +         DC     AL3(ERRLIST)                DCB OR ACB ADDRESS
000110    000000C8 0A14           +         SVC    20                          ISSUE CLOSE SVC
000111    000000CA 58D0 327E                L 13,SAVE+4
000112                                      RETURN (14,12)
```



The IDz remote debugger

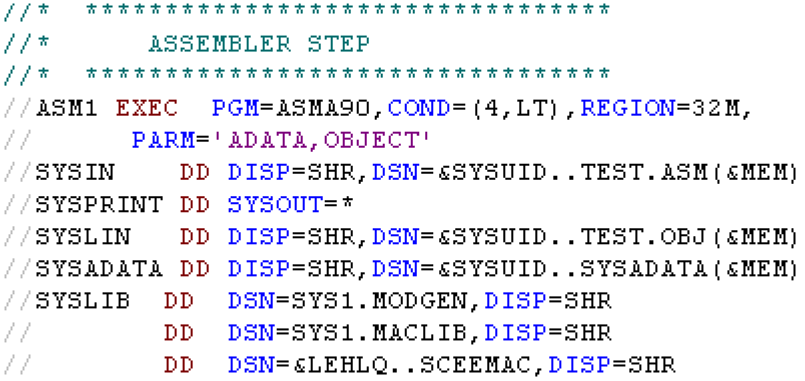
- Client software that is installed with IDz on your workstation
- Communicates with the Debug Tool engine on the mainframe
  - Note that Debug Tool must be installed on z/OS in order for you to do the labs in this unit

Steps for **Batch** Application Debug Session

1. Ensure that your compile proc has the necessary TEST parameter, and Compile/Link options and DD cards to create a debug-ready load module
2. Discover workstation TCP/IP parameters:
   - IP Address
   - Listener port#
3. Enter TCP/IP address of workstation in run JCL for Debug Tool DD statement, and Submit the JCL
4. Load the Assembler source code
5. Debug the application

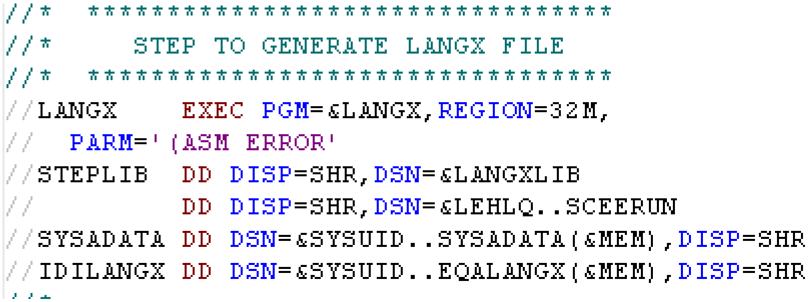
Compile JCL Requirements for Using Debug Tool for Assembler

- To debug Assembler programs, you will need additional datasets and steps:



```
//*   *********************************
//*       ASSEMBLER STEP
//*   *********************************
//ASM1 EXEC  PGM=ASMA90,COND=(4,LT),REGION=32M,
//         PARM='ADATA,OBJECT'
//SYSIN     DD DISP=SHR,DSN=&SYSUID..TEST.ASM(&MEM)
//SYSPRINT DD SYSOUT=*
//SYSLIN    DD DISP=SHR,DSN=&SYSUID..TEST.OBJ(&MEM)
//SYSADATA DD DISP=SHR,DSN=&SYSUID..SYSADATA(&MEM)
//SYSLIB   DD  DSN=SYS1.MODGEN,DISP=SHR
//         DD  DSN=SYS1.MACLIB,DISP=SHR
//         DD  DSN=&LEHLQ..SCEEMAC,DISP=SHR
```

- **SYSADATA**



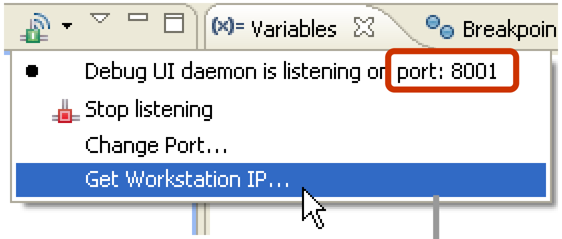
```
//*   *********************************
//*      STEP TO GENERATE LANGX FILE
//*   *********************************
//LANGX     EXEC PGM=&LANGX,REGION=32M,
//   PARM='(ASM ERROR'
//STEPLIB   DD DISP=SHR,DSN=&LANGXLIB
//          DD DISP=SHR,DSN=&LEHLQ..SCEERUN
//SYSADATA DD DSN=&SYSUID..SYSADATA(&MEM),DISP=SHR
//IDILANGX DD DSN=&SYSUID..EQALANGX(&MEM),DISP=SHR
//*
```

- **EQALANGX**
  - Step creates Debug symbolics

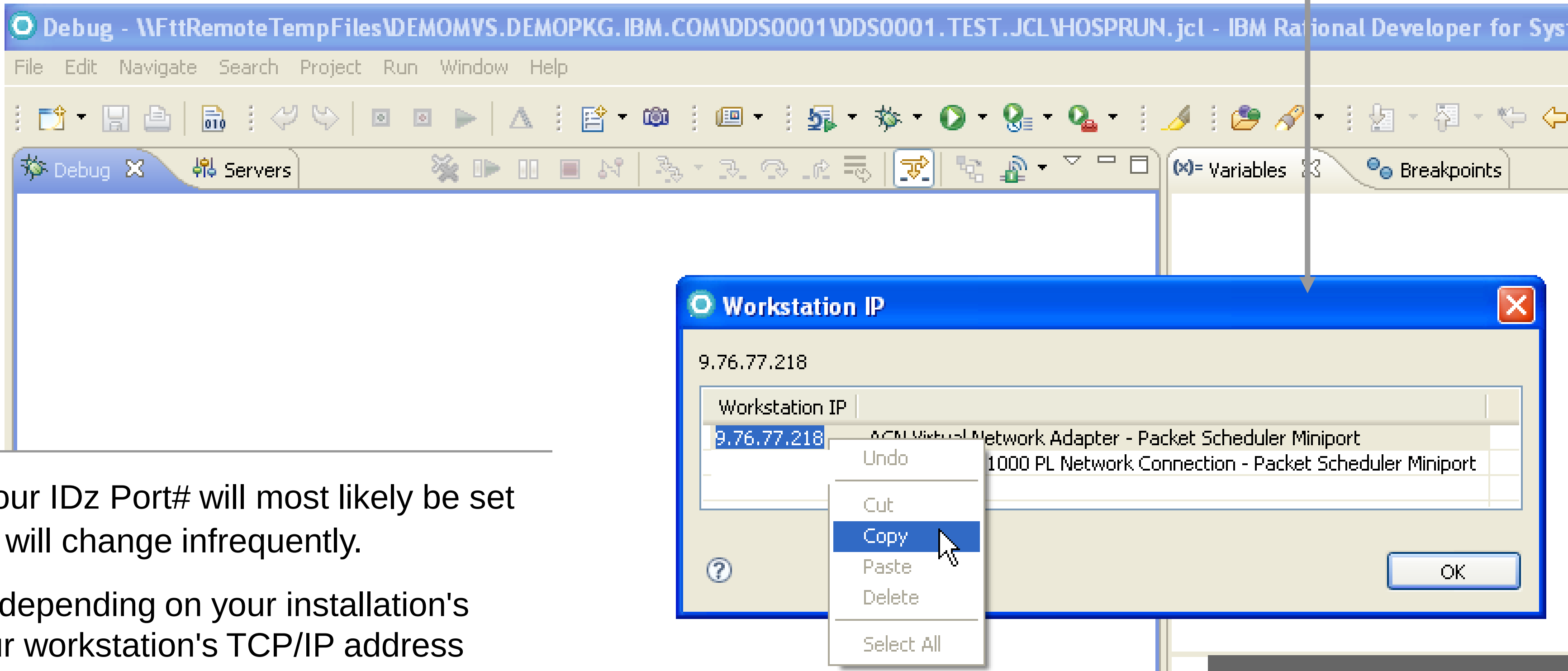
- See the **Debug Tool vxx Users Guide** – Preparing an Assembler program – for more information on these datasets.
- Sample Assembler JCL is in the slide notes

2. Discover TCP/IP address and IDz Port

- Open the **Debug Perspective**
  - Click the small downward pointing triangle next to the debug-daemon icon
    - **Note the Port#**
    - Select: **Get Workstation IP…**
    - **Copy the IP address**



Debug UI daemon is listening on port: 8001



Debug - \\FttRemoteTempFiles\DEMOMVS.DEMOPKG.IBM.COM\DDS0001\DDS0001.TEST.JCL\HOSPRUN.jcl - IBM Rational Developer for Sys

Workstation IP

9.76.77.218

Workstation IP	
9.76.77.218	ACN Virtual Network Adapter - Packet Scheduler Miniport
	1000 PL Network Connection - Packet Scheduler Miniport

Note: Your IDz Port# will most likely be set once, and will change infrequently.

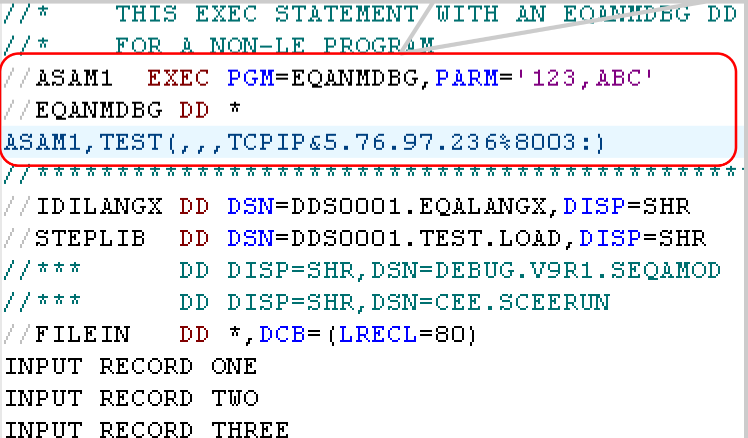
However, depending on your installation's setup, your workstation's TCP/IP address could change - often

See Notes

3. Submit the JCL for Assembler Debugging

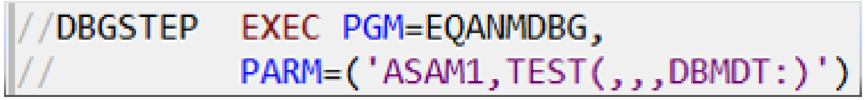
- Configure your application to start Debug Tool by including a specific DD card in the run JCL – that includes your workstation's current Port# and TCP/IP address
  - This is an example of JCL to run a batch job
  - The EQANMDBG DD statement is the easiest way to start the Debug Tool for batch applications

```
//EQANMDBG  DD  *
PGM TEST(,,,TCPIP&5.76.97.236%8003:)
```



```
//*     THIS EXEC STATEMENT WITH AN EQANMDBG DD
//*     FOR A NON-LE PROGRAM
//ASAM1   EXEC PGM=EQANMDBG,PARM='123,ABC'
//EQANMDBG DD *
ASAM1,TEST(,,,TCPIP&5.76.97.236%8003:)
//*****************************************
//IDILANGX DD DSN=DDS0001.EQALANGX,DISP=SHR
//STEPLIB  DD DSN=DDS0001.TEST.LOAD,DISP=SHR
//***         DD DISP=SHR,DSN=DEBUG.V9R1.SEQAMOD
//***         DD DISP=SHR,DSN=CEE.SCEERUN
//FILEIN   DD *,DCB=(LRECL=80)
INPUT RECORD ONE
INPUT RECORD TWO
INPUT RECORD THREE
```

Note: You may be able to use DBMDT instead of the TCPIP Parms



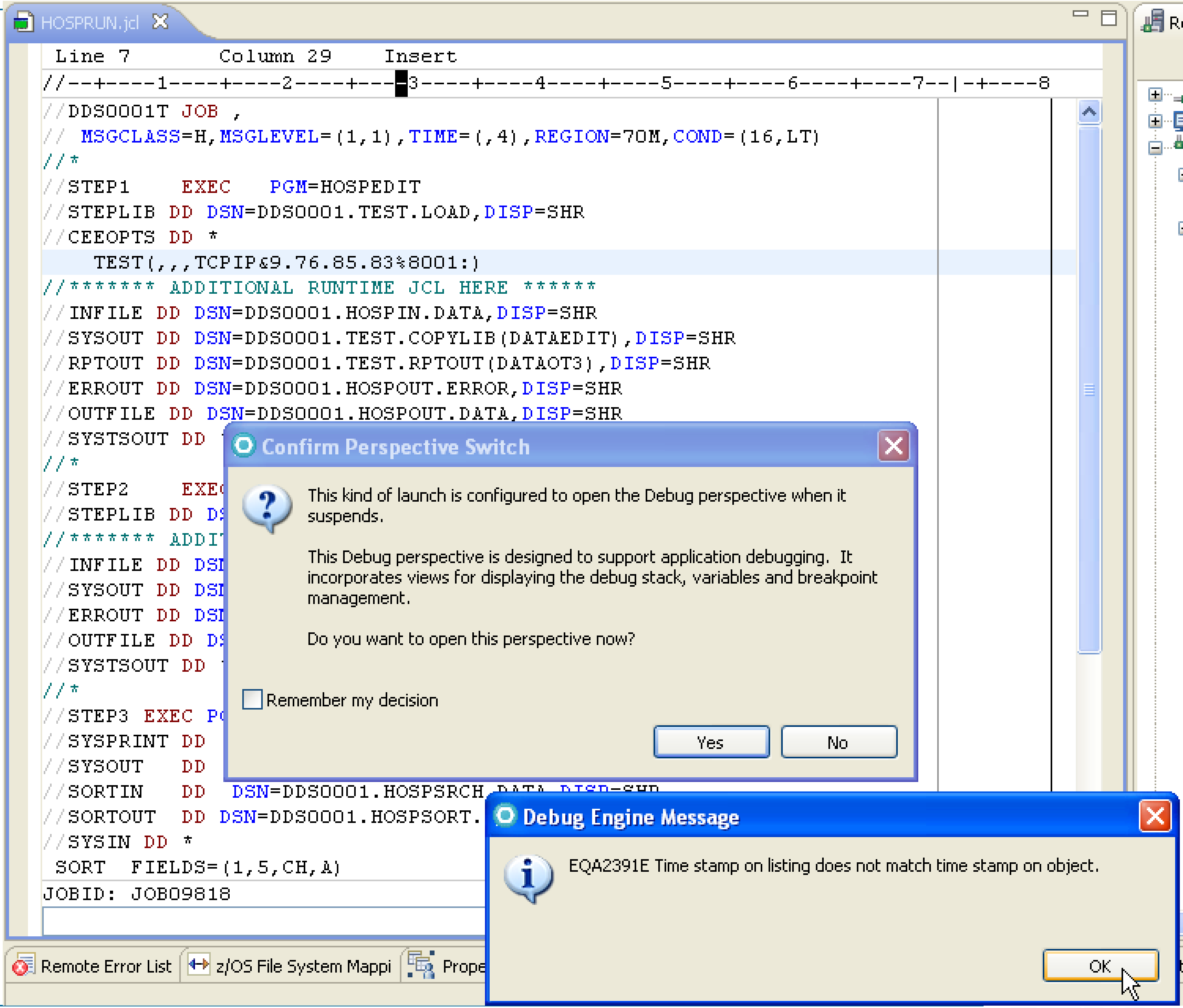
```
//DBGSTEP   EXEC  PGM=EQANMDBG,
//                PARM=('ASAM1,TEST(,,,DBMDT:)')
```

3a. Debug Tool - Prompts

- Debug Tool will interface with IDz and throw the Confirm Perspective Switch prompt
  - Click Yes

Additionally, **if** your mainframe source code is out sync with the Load Module you'll get an informational prompt.

This typically means you need to check your compile listings for syntax errors that caused the link edit step not to execute because of condition codes



```
 Line 7        Column 29     Insert
//--+----1----+----2----+----3----+----4----+----5----+----6----+----7--|-+----8
//DDS0001T JOB ,
//  MSGCLASS=H,MSGLEVEL=(1,1),TIME=(,4),REGION=70M,COND=(16,LT)
//*
//STEP1    EXEC   PGM=HOSPEDIT
//STEPLIB DD DSN=DDS0001.TEST.LOAD,DISP=SHR
//CEEOPTS DD *
  TEST(,,,TCPIP&9.76.85.83%8001:)
//******* ADDITIONAL RUNTIME JCL HERE ******
//INFILE DD DSN=DDS0001.HOSPIN.DATA,DISP=SHR
//SYSOUT DD DSN=DDS0001.TEST.COPYLIB(DATAEDIT),DISP=SHR
//RPTOUT DD DSN=DDS0001.TEST.RPTOUT(DATAOT3),DISP=SHR
//ERROUT DD DSN=DDS0001.HOSPOUT.ERROR,DISP=SHR
//OUTFILE DD DSN=DDS0001.HOSPOUT.DATA,DISP=SHR
//SYSTSOUT DD
//*
//STEP2    EXE
//STEPLIB DD DS
//******* ADDI
//INFILE DD DS
//SYSOUT DD DS
//ERROUT DD DS
//OUTFILE DD DS
//SYSTSOUT DD
//*
//STEP3 EXEC PG
//SYSPRINT DD
//SYSOUT   DD
//SORTIN   DD  DSN=DDS0001.HOSPSRCH.DATA,DISP=SHR
//SORTOUT  DD DSN=DDS0001.HOSPSORT.
//SYSIN DD *
 SORT  FIELDS=(1,5,CH,A)
JOBID: JOB09818
```

Confirm Perspective Switch

This kind of launch is configured to open the Debug perspective when it suspends.

This Debug perspective is designed to support application debugging. It incorporates views for displaying the debug stack, variables and breakpoint management.

Do you want to open this perspective now?

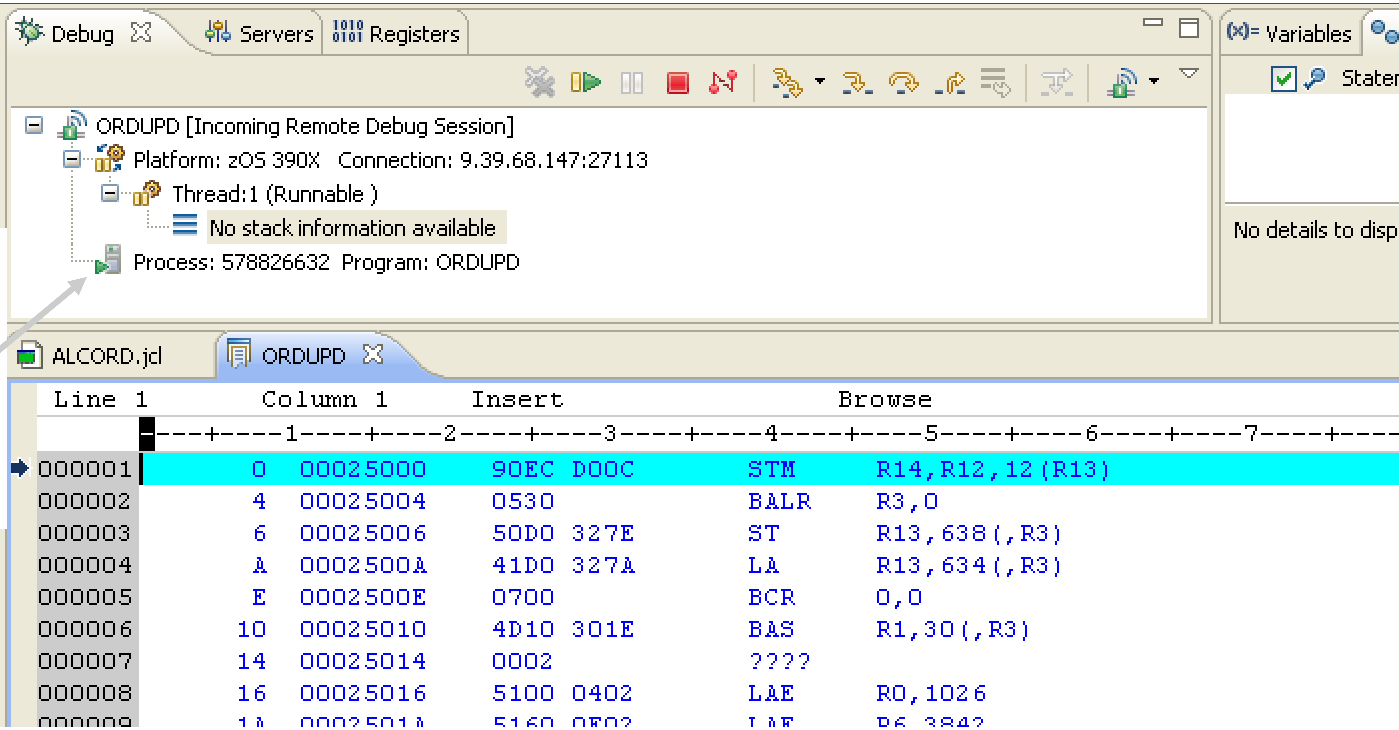
□ Remember my decision

Debug Engine Message

EQA2391E Time stamp on listing does not match time stamp on object.

3b. Debug Tool Connects to IDz

- **Debug Perspective** is launched in IDz



```
ORDUPD [Incoming Remote Debug Session]
  Platform: zOS 390X   Connection: 9.39.68.147:27113
    Thread:1 (Runnable )
      No stack information available
  Process: 578826632  Program: ORDUPD
```

No details to disp

```
Line 1          Column 1       Insert                     Browse
      ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----
000001        0   00025000     90EC D00C         STM      R14,R12,12(R13)
000002        4   00025004     0530              BALR     R3,0
000003        6   00025006     50D0 327E         ST       R13,638(,R3)
000004        A   0002500A     41D0 327A         LA       R13,634(,R3)
000005        E   0002500E     0700              BCR      0,0
000006       10   00025010     4D10 301E         BAS      R1,30(,R3)
000007       14   00025014     0002              ????
000008       16   00025016     5100 0402         LAE      R0,1026
000009       1A   0002501A     5160 0F02         LAE      R6,3842
```

- Program source is copied down from z/OS to your IDz workstation
- Execution is on z/OS

Note: Initially the Assembled instruction set is loaded into Debug Tool.

You will want to load and utilize "debug data" source – the LANGX file output from Assemble in your testing.

This is achieved via the LDD XXXX command (next slide)

4. Load Debug Data (LDD Command)

- Before you can debug an assembler program, you must:
  - Define the compilation unit (CU) as an assembler CU
  - Load the debug data for the compilation unit.
    - This can only be done for a compilation unit that is currently known to Debug Tool as a "disassembly CU".
- Use the **LOADDEBUGDATA** command
  - abbreviated as **LDD**

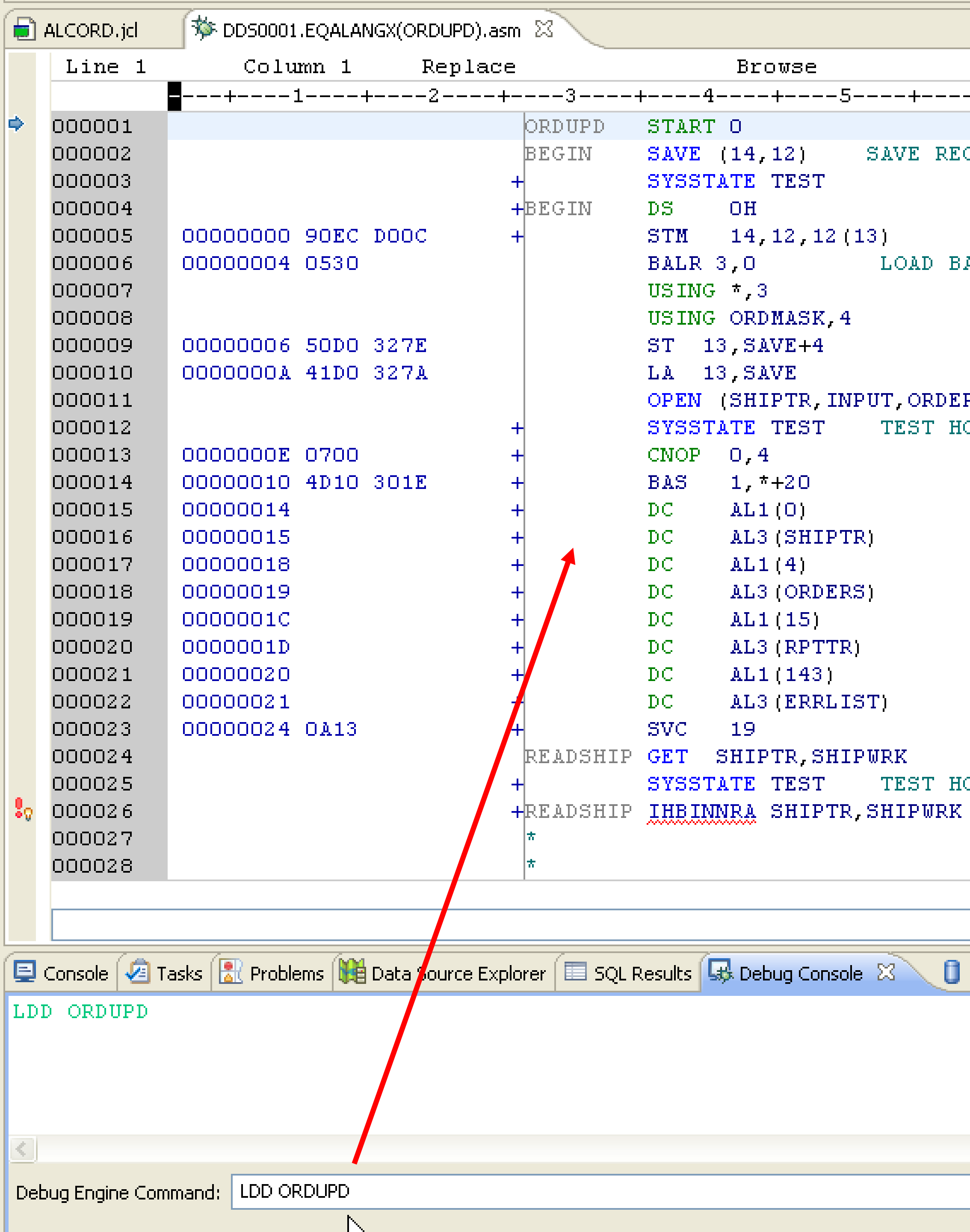
… to define a disassembly CU as an assembler CU and to cause the debug data for this CU to be loaded.

- See the Debug Tool Users Guide for additional details on this command

Steps:

- From the Debug Console view
- Enter the Debug Command:
  - LDD <modulename>

Note: Debug Commands are <u>not</u> case-sensitive

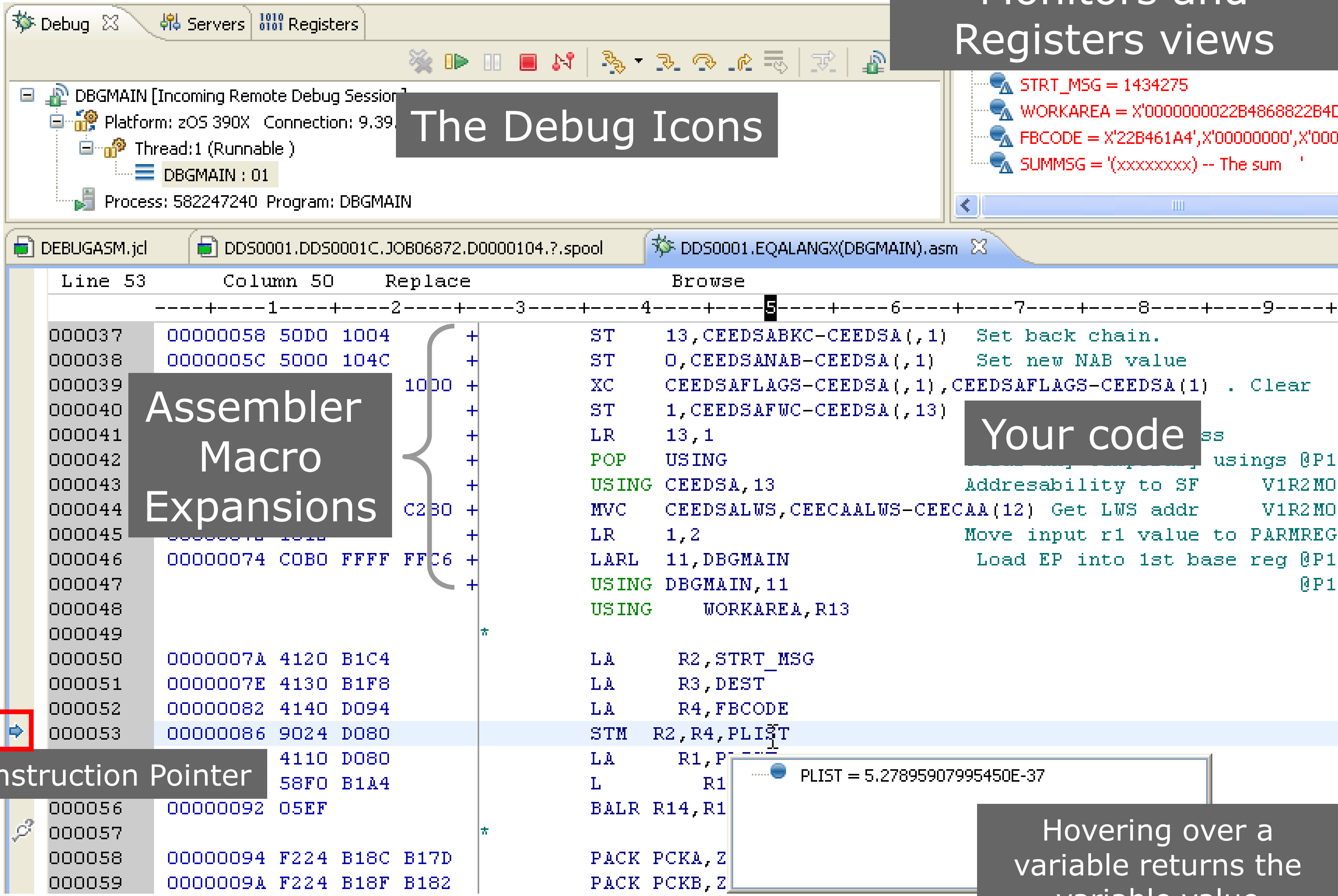


```
 Line 1          Column 1      Replace                Browse
        ----+----1----+----2----+----3----+----4----+----5----+----
                                 ORDUPD   START 0
                                 BEGIN    SAVE  (14,12)       SAVE REG
                                +         SYSSTATE TEST
                                +BEGIN    DS    0H
 00000000 90EC D00C             +         STM   14,12,12(13)
 00000004 0530                            BALR  3,0            LOAD BA
                                          USING *,3
                                          USING ORDMASK,4
 00000006 50D0 327E                       ST    13,SAVE+4
 0000000A 41D0 327A                       LA    13,SAVE
                                          OPEN  (SHIPTR,INPUT,ORDER
                                +         SYSSTATE TEST        TEST HO
 0000000E 0700                  +         CNOP  0,4
 00000010 4D10 301E             +         BAS   1,*+20
 00000014                       +         DC    AL1(0)
 00000015                       +         DC    AL3(SHIPTR)
 00000018                       +         DC    AL1(4)
 00000019                       +         DC    AL3(ORDERS)
 0000001C                       +         DC    AL1(15)
 0000001D                       +         DC    AL3(RPTTR)
 00000020                       +         DC    AL1(143)
 00000021                       +         DC    AL3(ERRLIST)
 00000024 0A13                  +         SVC   19
                                 READSHIP GET   SHIPTR,SHIPWRK
                                +         SYSSTATE TEST        TEST HO
                                +READSHIP IHBINNRA SHIPTR,SHIPWRK
                                 *
                                 *
```

```
LDD ORDUPD
```

Debug Engine Command: LDD ORDUPD

5. The Debug Perspective and Views


Debug
Servers
Registers
DEBUGASM.jcl
DDS0001.DDS0001C.JOB06872.D0000104.?.spool
DDS0001.EQALANGX(DBGMAIN).asm

```
DBGMAIN [Incoming Remote Debug Session]
  Platform: zOS 390X   Connection: 9.39.
    Thread:1 (Runnable )
      DBGMAIN : 01
  Process: 582247240  Program: DBGMAIN
```

```
STRT_MSG = 1434275
WORKAREA = X'0000000022B4868822B4D
FBCODE = X'22B461A4',X'00000000',X'000
SUMMSG = '(xxxxxxxx) -- The sum   '
```

```
 Line 53          Column 50     Replace                  Browse
        ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----9----+
000037  00000058 50D0 1004      +        ST    13,CEEDSABKC-CEEDSA(,1)  Set back chain.
000038  0000005C 5000 104C      +        ST    0,CEEDSANAB-CEEDSA(,1)   Set new NAB value
000039                     1000 +        XC    CEEDSAFLAGS-CEEDSA(,1),CEEDSAFLAGS-CEEDSA(1) . Clear
000040                          +        ST    1,CEEDSAFWC-CEEDSA(,13)
000041                          +        LR    13,1                               ss
000042                          +        POP   USING                              usings @P1
000043                          +        USING CEEDSA,13             Addresability to SF      V1R2M0
000044                     C230 +        MVC   CEEDSALWS,CEECAALWS-CEECAA(12) Get LWS addr   V1R2M0
000045                          +        LR    1,2                   Move input r1 value to PARMREG
000046  00000074 C0B0 FFFF FFC6 +        LARL  11,DBGMAIN             Load EP into 1st base reg @P1
000047                          +        USING DBGMAIN,11                                      @P1
000048                                   USING    WORKAREA,R13
000049                          *
000050  0000007A 4120 B1C4               LA      R2,STRT_MSG
000051  0000007E 4130 B1F8               LA      R3,DEST
000052  00000082 4140 D094               LA      R4,FBCODE
000053  00000086 9024 D080               STM   R2,R4,PLIST
                 4110 D080               LA      R1,P
                 58F0 B1A4               L          R1
000056  00000092 05EF                    BALR R14,R1
000057                          *
000058  00000094 F224 B18C B17D          PACK PCKA,Z
000059  0000009A F224 B18F B182          PACK PCKB,Z
```

```
PLIST = 5.27895907995450E-37
```

Breakpoints Monitors and Registers views

The Debug Icons

Assembler Macro Expansions

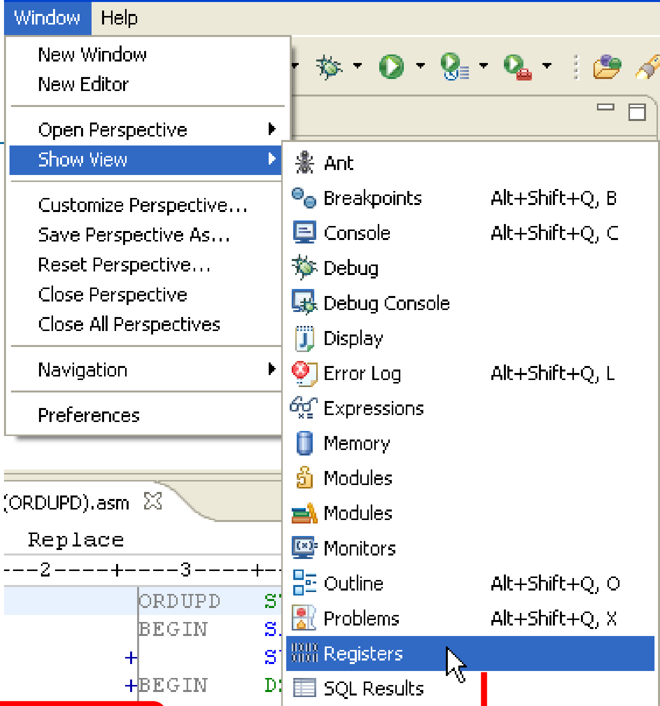
Your code

Current Instruction Pointer

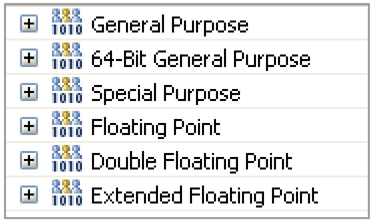
Hovering over a variable returns the variable value

Displaying/Manipulating the Registers

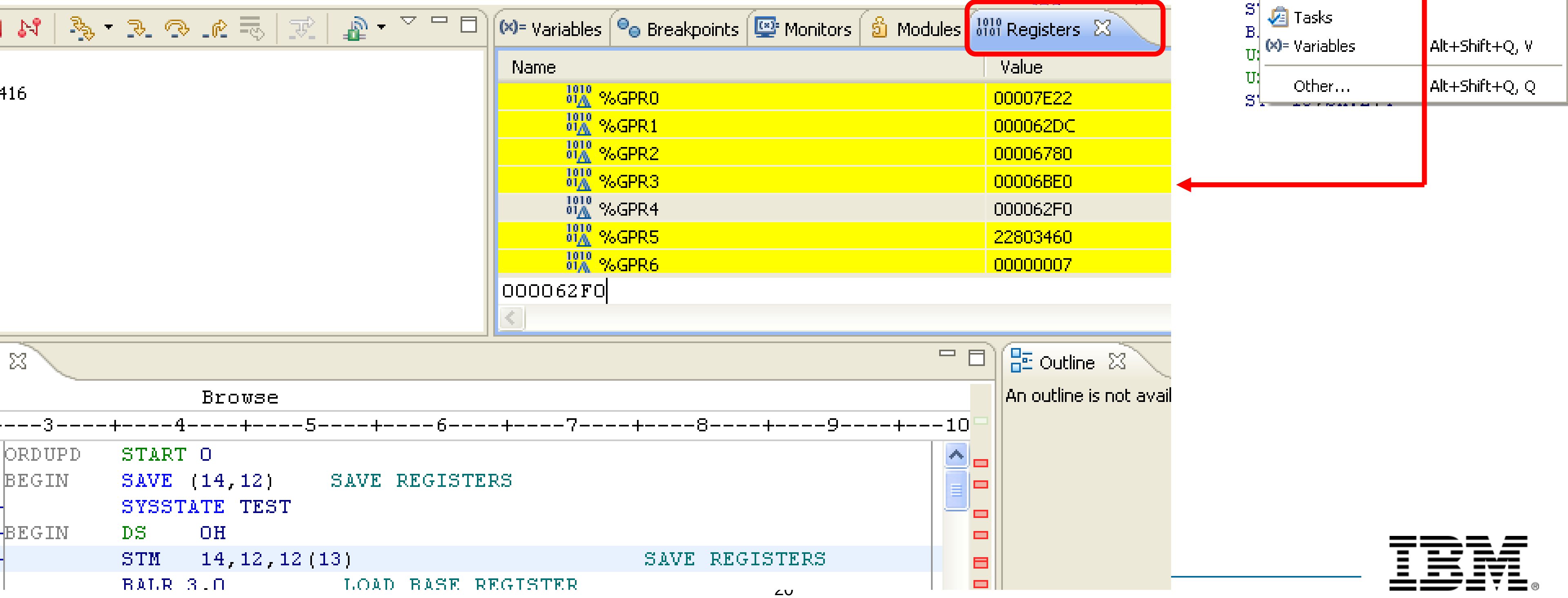
Enable from: **Window > Show View > Registers**



Opens the **Registers** view to a scrollable, editable register list with access to the following:



- General Purpose
- 64-Bit General Purpose
- Special Purpose
- Floating Point
- Double Floating Point
- Extended Floating Point

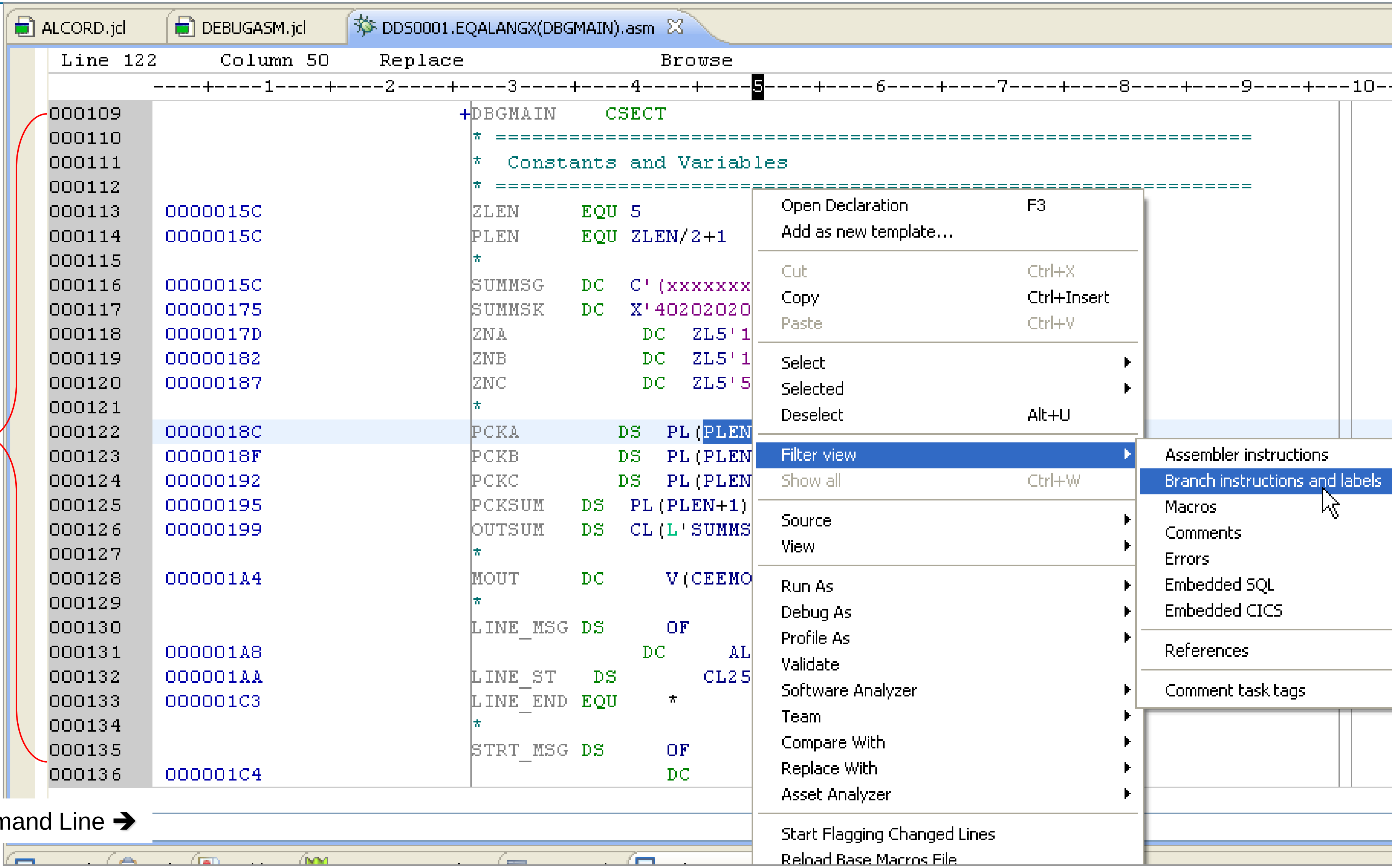


Name	Value
%GPR0	00007E22
%GPR1	000062DC
%GPR2	00006780
%GPR3	00006BE0
%GPR4	000062F0
%GPR5	22803460
%GPR6	00000007

000062F0

```
----3----+----4----+----5----+----6----+----7----+----8----+----9----+---10
ORDUPD   START 0
BEGIN    SAVE  (14,12)     SAVE REGISTERS
         SYSSTATE TEST
BEGIN    DS    0H
         STM   14,12,12(13)                     SAVE REGISTERS
         BALR  3,0          LOAD BASE REGISTER
```

Debug and IDz's LPEX Editor Functionality



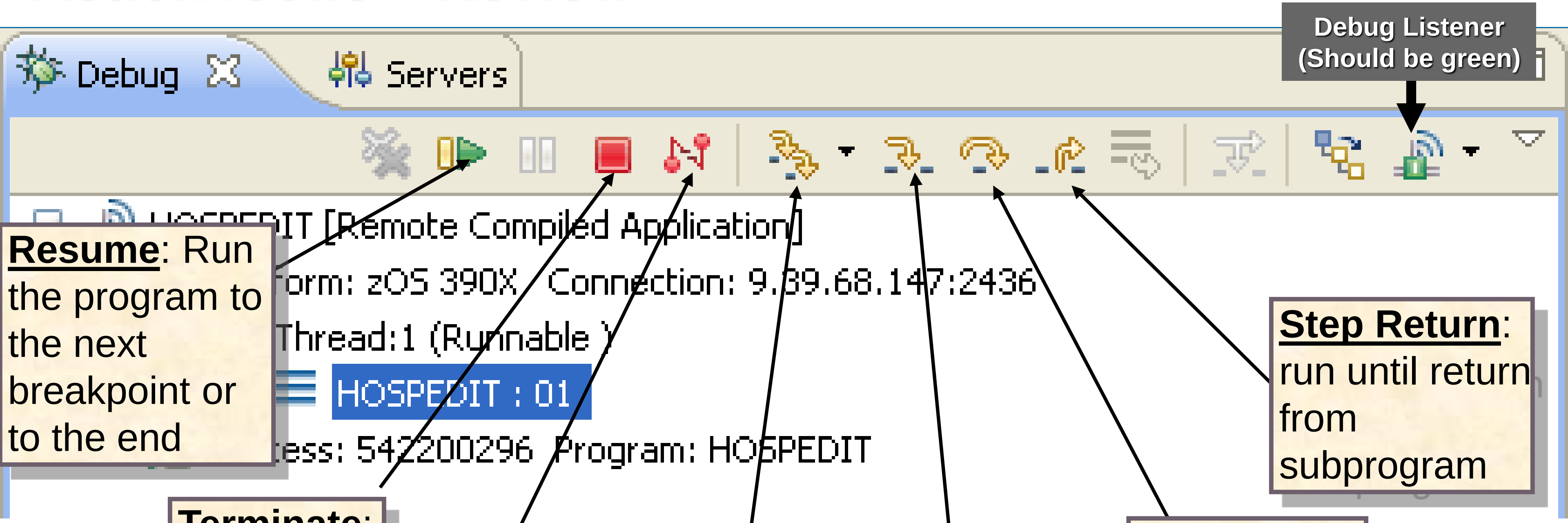
```
 Line 122      Column 50     Replace                      Browse
          ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----9----+----10--
000109                                    +DBGMAIN    CSECT
000110                                     * ==================================================================
000111                                     *  Constants and Variables
000112                                     * ==================================================================
000113    0000015C                         ZLEN      EQU 5
000114    0000015C                         PLEN      EQU ZLEN/2+1
000115                                     *
000116    0000015C                         SUMMSG    DC   C'(xxxxxxx
000117    00000175                         SUMMSK    DC   X'40202020
000118    0000017D                         ZNA            DC   ZL5'1
000119    00000182                         ZNB            DC   ZL5'1
000120    00000187                         ZNC            DC   ZL5'5
000121                                     *
000122    0000018C                         PCKA         DS   PL(PLEN
000123    0000018F                         PCKB         DS   PL(PLEN
000124    00000192                         PCKC         DS   PL(PLEN
000125    00000195                         PCKSUM    DS   PL(PLEN+1)
000126    00000199                         OUTSUM    DS   CL(L'SUMMS
000127                                     *
000128    000001A4                         MOUT      DC      V(CEEMO
000129                                     *
000130                                     LINE_MSG DS      OF
000131    000001A8                                        DC      AL
000132    000001AA                         LINE_ST   DS         CL25
000133    000001C3                         LINE_END EQU      *
000134                                     *
000135                                     STRT_MSG DS      OF
000136    000001C4                                           DC
```

Prefix Area

Command Line ➔

All of the LPEX editing features work under Debug Tool

Action Icons – Review



HOSPEDIT [Remote Compiled Application]

form: zOS 390X Connection: 9.39.68.147:2436

Thread:1 (Runnable)

HOSPEDIT : 01

ess: 542200296 Program: HOSPEDIT

Debug Listener (Should be green)

Resume: Run the program to the next breakpoint or to the end

Terminate: End the program

Disconnect: from the debug engine

Animated Step Continuous source-level debugging without user interaction

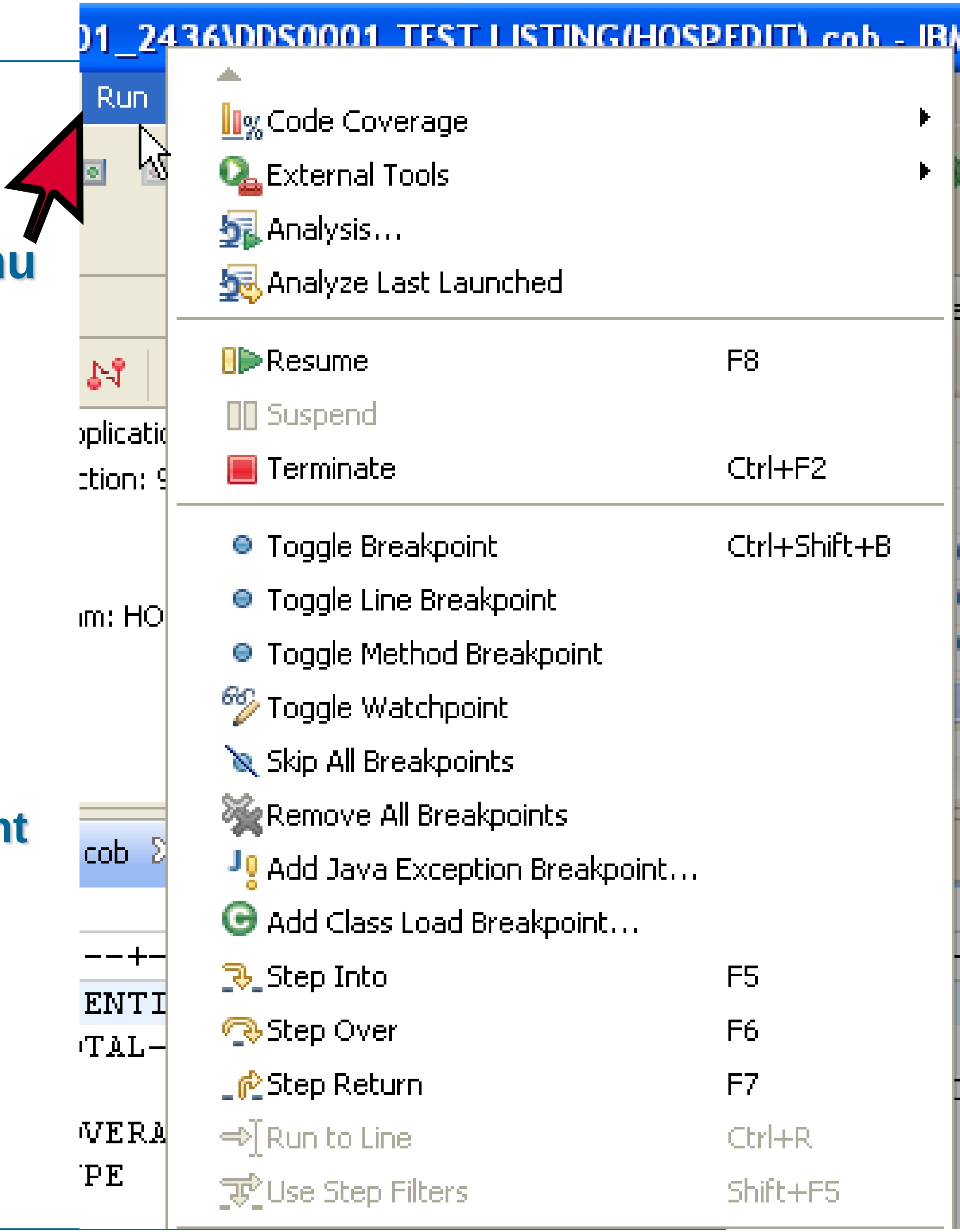
Step: run one statement

Step Over: run one statement, but step over a CALL

Step Return: run until return from subprogram

Run Menu

- Shows same + additional debugging functionality as icons on toolbar
  - However, not all Run menu functionality enabled for Assembler/PL1
- Also shows hot-keys
  - Your PC's function keys
- Context-sensitive:
  - Options are grayed in current debug session if not applicable

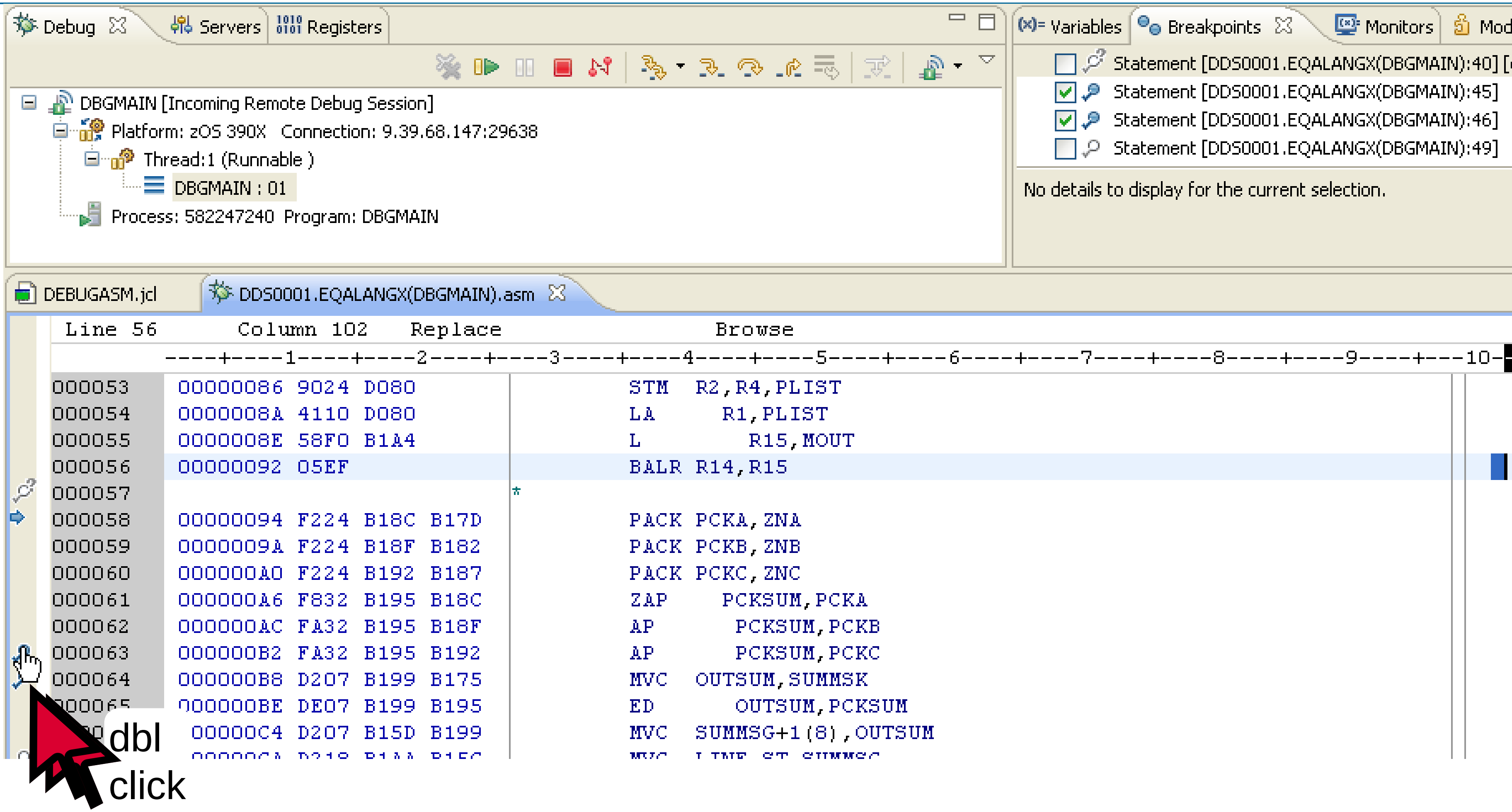


Statement Breakpoints – Review

- A statement breakpoint will stop the program when it reaches a statement:
  - It stops ***before*** the statement runs
- A breakpoint can optionally be made conditional
  - A simple condition may be specified such as:
    - **`VAR1 > 999`**

      …or…
    - **`VAR2 = 'ABC'`**
- A breakpoint can be based on a frequency:
  - Stop the Nth time a statement runs

Set a Statement Breakpoint – Review



```
DBGMAIN [Incoming Remote Debug Session]
  Platform: zOS 390X   Connection: 9.39.68.147:29638
    Thread:1 (Runnable )
      DBGMAIN : 01
  Process: 582247240  Program: DBGMAIN
```

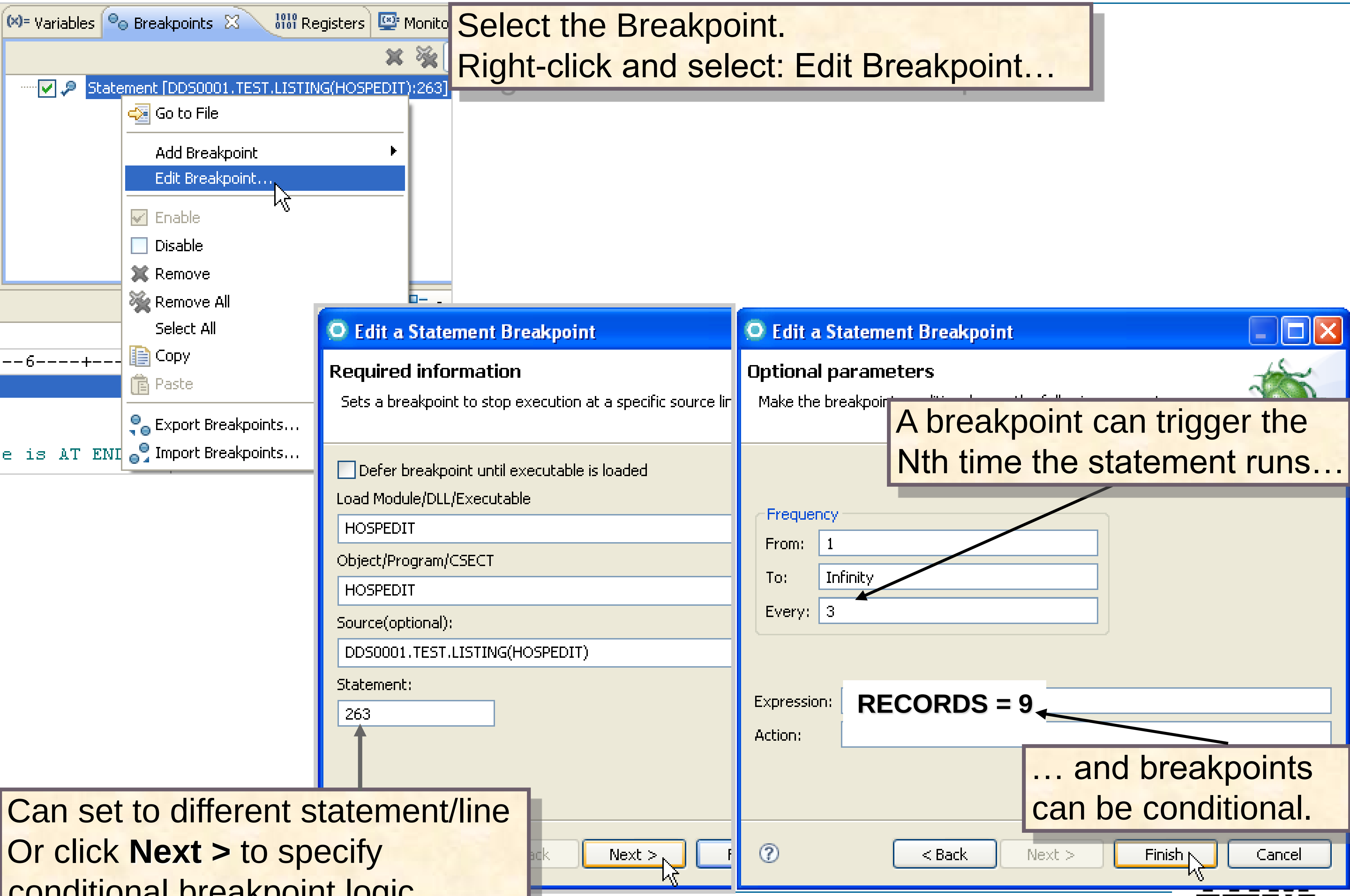
- □ Statement [DDS0001.EQALANGX(DBGMAIN):40]
- ☑ Statement [DDS0001.EQALANGX(DBGMAIN):45]
- ☑ Statement [DDS0001.EQALANGX(DBGMAIN):46]
- □ Statement [DDS0001.EQALANGX(DBGMAIN):49]

No details to display for the current selection.

```
 Line 56       Column 102    Replace                  Browse
        ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----9----+---10-
000053  00000086 9024 D080                  STM  R2,R4,PLIST
000054  0000008A 4110 D080                  LA     R1,PLIST
000055  0000008E 58F0 B1A4                  L        R15,MOUT
000056  00000092 05EF                       BALR R14,R15
000057                         *
000058  00000094 F224 B18C B17D             PACK PCKA,ZNA
000059  0000009A F224 B18F B182             PACK PCKB,ZNB
000060  000000A0 F224 B192 B187             PACK PCKC,ZNC
000061  000000A6 F832 B195 B18C             ZAP    PCKSUM,PCKA
000062  000000AC FA32 B195 B18F             AP       PCKSUM,PCKB
000063  000000B2 FA32 B195 B192             AP       PCKSUM,PCKC
000064  000000B8 D207 B199 B175             MVC  OUTSUM,SUMMSK
000065  000000BE DE07 B199 B195             ED       OUTSUM,PCKSUM
000066   000000C4 D207 B15D B199            MVC  SUMMSG+1(8),OUTSUM
```

Set a statement breakpoint by double-clicking in the gray area next to a statement

Set/Edit Conditional Statement Breakpoints



☑ Statement [DDS0001.TEST.LISTING(HOSPEDIT):263]

Select the Breakpoint.
Right-click and select: Edit Breakpoint…

Edit a Statement Breakpoint

Required information

Sets a breakpoint to stop execution at a specific source lir

□ Defer breakpoint until executable is loaded

Load Module/DLL/Executable: HOSPEDIT

Object/Program/CSECT: HOSPEDIT

Source(optional): DDS0001.TEST.LISTING(HOSPEDIT)

Statement: 263

Can set to different statement/line
Or click **Next >** to specify
conditional breakpoint logic

Edit a Statement Breakpoint

Optional parameters

Make the breakpoir

A breakpoint can trigger the
Nth time the statement runs…

Frequency

From: 1

To: Infinity

Every: 3

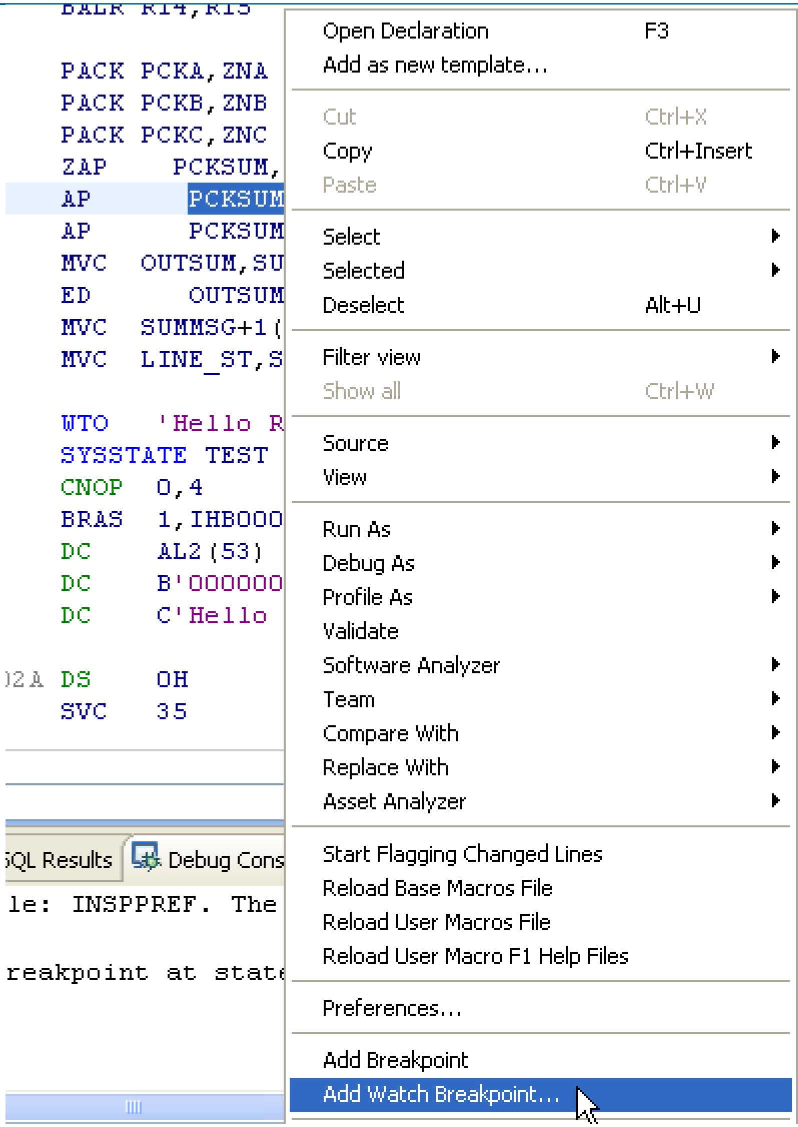
Expression: **RECORDS = 9**

Action:

… and breakpoints
can be conditional.

Watch Monitor Breakpoints

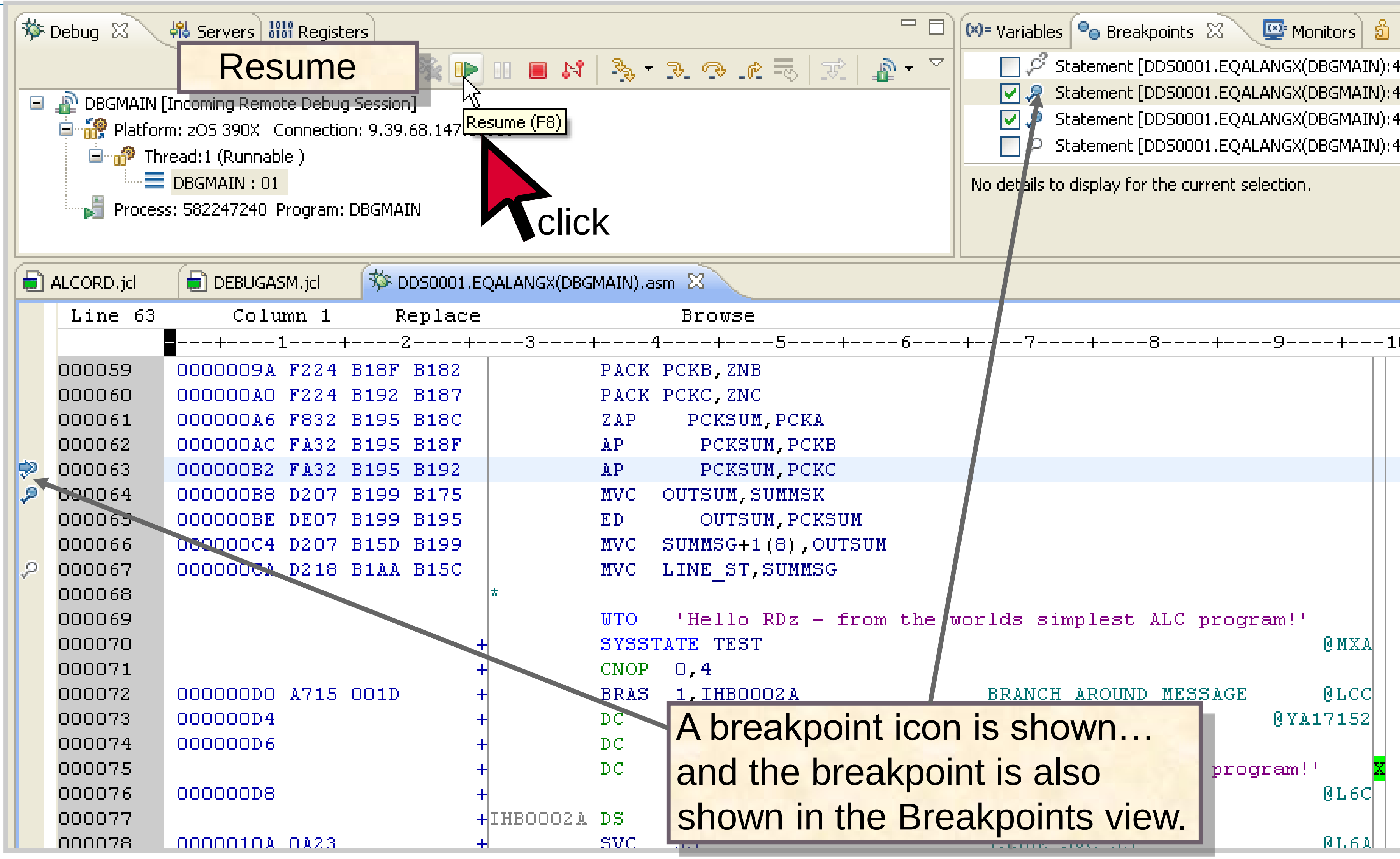
- Can have breakpoints occur conditionally, when:
  - The value in a field changes
  - Some portion (# of bytes) of a field changes
  - A simple condition tests true for the value in the field
- Steps:
  - Select a variable
  - Right-click, and select: Add Watch Breakpoint…
  - Select Number of bytes to watch – or add a simple condition
    - Specify Auto to test for all bytes



```
     BALR R14,R15

     PACK PCKA,ZNA
     PACK PCKB,ZNB
     PACK PCKC,ZNC
     ZAP    PCKSUM,
     AP      PCKSUM
     AP      PCKSUM
     MVC  OUTSUM,SU
     ED      OUTSUM
     MVC  SUMMSG+1(
     MVC  LINE_ST,S

     WTO    'Hello R
     SYSSTATE TEST
     CNOP  0,4
     BRAS  1,IHB000
     DC    AL2(53)
     DC    B'000000
     DC    C'Hello

)2A  DS    0H
     SVC   35
```

```
le: INSPPREF. The

reakpoint at state
```

Run (F8) to a Statement Breakpoint



Resume

click

```
DBGMAIN [Incoming Remote Debug Session]
  Platform: zOS 390X   Connection: 9.39.68.147
    Thread:1 (Runnable )
      DBGMAIN : 01
  Process: 582247240  Program: DBGMAIN
```

```
□ Statement [DDS0001.EQALANGX(DBGMAIN):4
☑ Statement [DDS0001.EQALANGX(DBGMAIN):4
☑ Statement [DDS0001.EQALANGX(DBGMAIN):4
□ Statement [DDS0001.EQALANGX(DBGMAIN):4
```

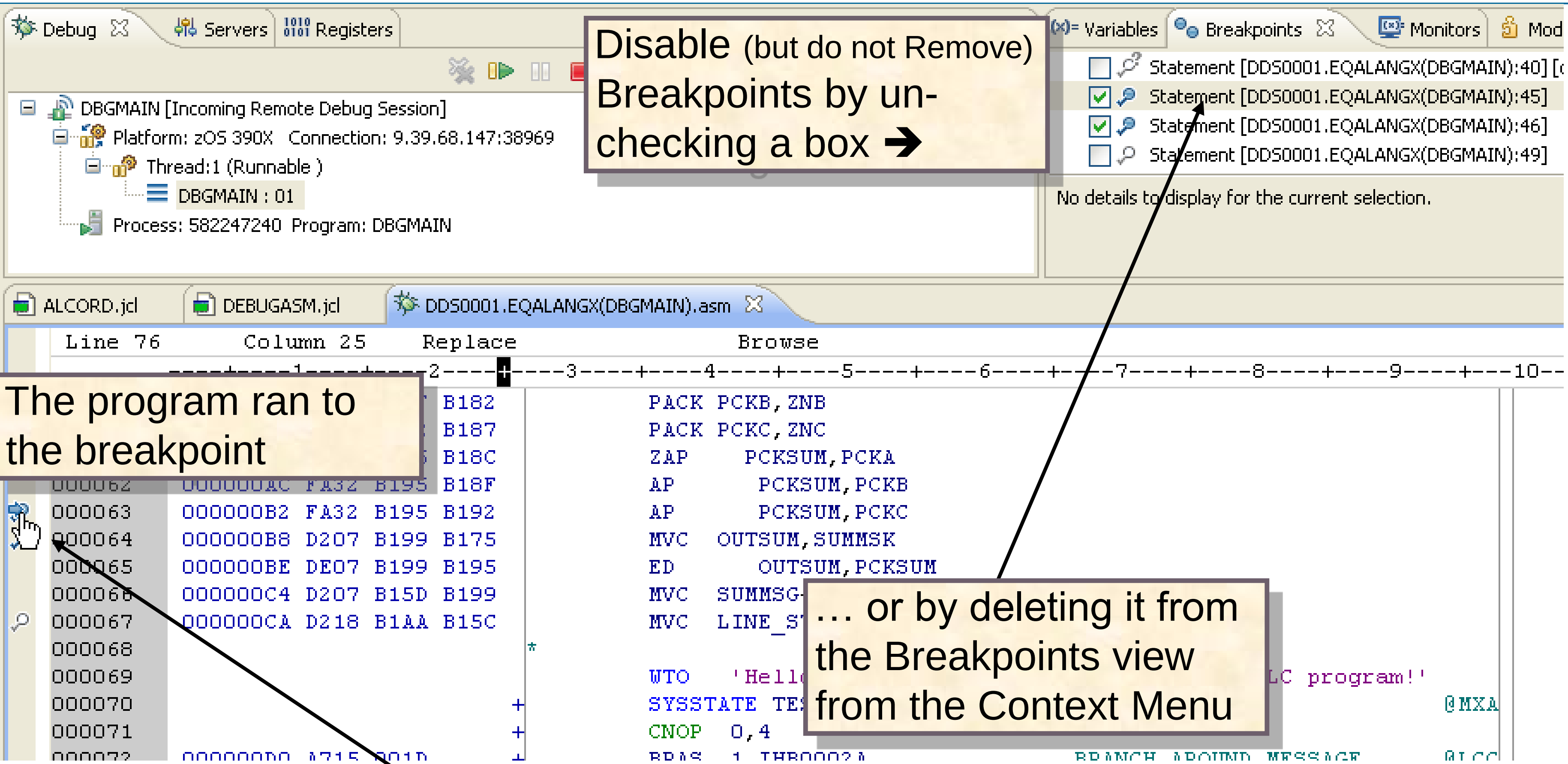
No details to display for the current selection.

```
 Line 63      Column 1      Replace               Browse
          ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----9----+---1
000059    0000009A F224 B18F B182              PACK PCKB,ZNB
000060    000000A0 F224 B192 B187              PACK PCKC,ZNC
000061    000000A6 F832 B195 B18C              ZAP    PCKSUM,PCKA
000062    000000AC FA32 B195 B18F              AP      PCKSUM,PCKB
000063    000000B2 FA32 B195 B192              AP      PCKSUM,PCKC
000064    000000B8 D207 B199 B175              MVC  OUTSUM,SUMMSK
000065    000000BE DE07 B199 B195              ED      OUTSUM,PCKSUM
000066    000000C4 D207 B15D B199              MVC  SUMMSG+1(8),OUTSUM
000067    000000CA D218 B1AA B15C              MVC  LINE_ST,SUMMSG
000068                                *
000069                                         WTO   'Hello RDz - from the worlds simplest ALC program!'
000070                               +         SYSSTATE TEST                                               @MXA
000071                               +         CNOP  0,4
000072    000000D0 A715 001D         +         BRAS  1,IHB0002A                   BRANCH AROUND MESSAGE      @LCC
000073    000000D4                   +         DC                                                         @YA17152
000074    000000D6                   +         DC
000075                               +         DC                                  program!'
000076    000000D8                   +                                                                     @L6C
000077                               +IHB0002A DS
000078    0000010A 0A23              +         SVC                                                         @L6A
```

A breakpoint icon is shown… and the breakpoint is also shown in the Breakpoints view.

See Slide Notes

Breakpoint Options – 1 of 2



DBGMAIN [Incoming Remote Debug Session]
Platform: zOS 390X Connection: 9.39.68.147:38969
Thread:1 (Runnable)
DBGMAIN : 01
Process: 582247240 Program: DBGMAIN

□ Statement [DDS0001.EQALANGX(DBGMAIN):40] [
☑ Statement [DDS0001.EQALANGX(DBGMAIN):45]
☑ Statement [DDS0001.EQALANGX(DBGMAIN):46]
□ Statement [DDS0001.EQALANGX(DBGMAIN):49]

No details to display for the current selection.

```
 Line 76       Column 25     Replace               Browse
 ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----9----+---10--
                           B182              PACK PCKB,ZNB
                           B187              PACK PCKC,ZNC
                           B18C              ZAP    PCKSUM,PCKA
 000062    000000AC FA32 B195 B18F              AP       PCKSUM,PCKB
 000063    000000B2 FA32 B195 B192              AP       PCKSUM,PCKC
 000064    000000B8 D207 B199 B175              MVC   OUTSUM,SUMMSK
 000065    000000BE DE07 B199 B195              ED       OUTSUM,PCKSUM
 000066    000000C4 D207 B15D B199              MVC   SUMMSG-
 000067    000000CA D218 B1AA B15C              MVC   LINE_S
 000068                                 *
 000069                                          WTO   'Hell                      LC program!'
 000070                               +          SYSSTATE TE                                          @MXA
 000071                               +          CNOP  0,4
 000072    000000D0 A715 001D         +          BRAS  1,IHB0002A           BRANCH AROUND MESSAGE      @LCC
```

Disable (but do not Remove) Breakpoints by un-checking a box ➔

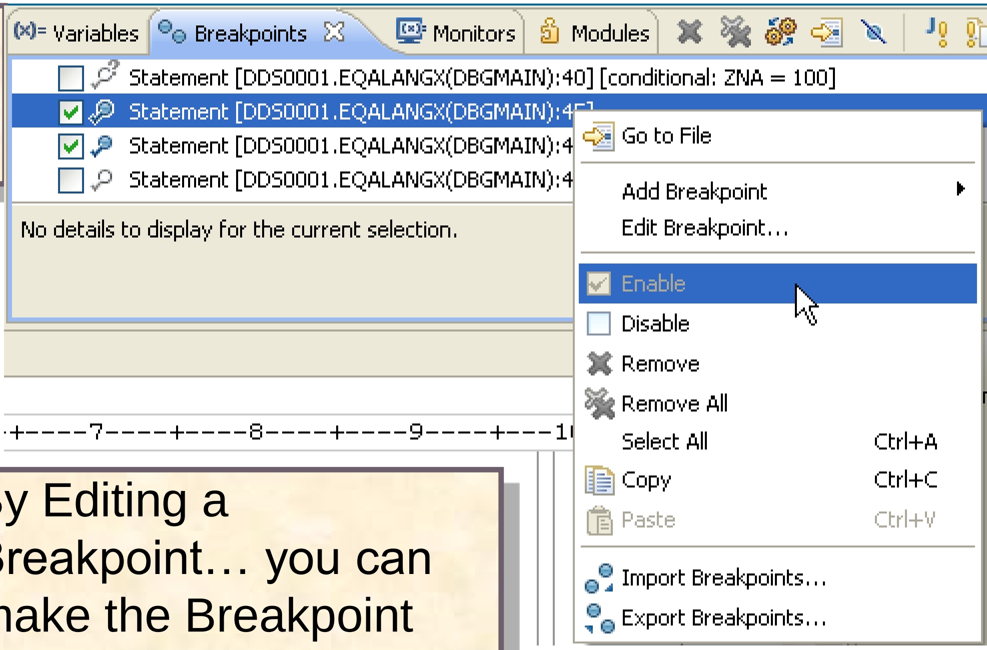
The program ran to the breakpoint

… or by deleting it from the Breakpoints view from the Context Menu

You can remove the breakpoint by double clicking again here…

Breakpoint Options – 2 of 2

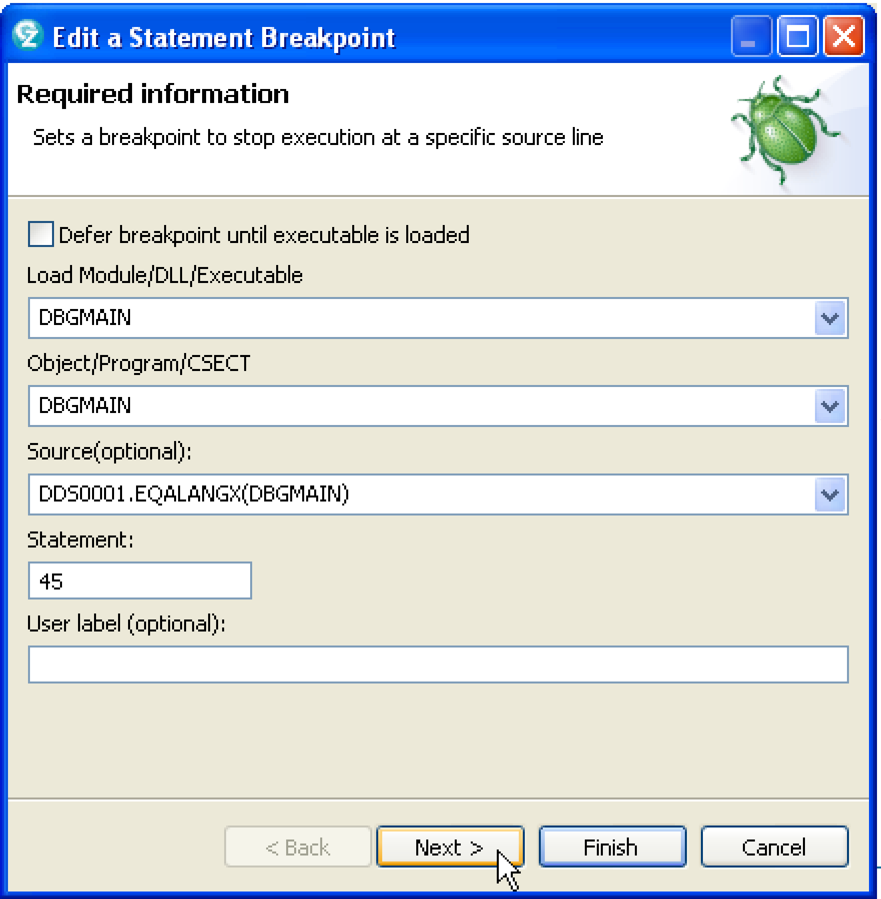
Disable (but do not Remove) Breakpoints by un-checking a box ➔



☐ Statement [DDS0001.EQALANGX(DBGMAIN):40] [conditional: ZNA = 100]
☑ Statement [DDS0001.EQALANGX(DBGMAIN):45]
☑ Statement [DDS0001.EQALANGX(DBGMAIN):4
☐ Statement [DDS0001.EQALANGX(DBGMAIN):4

No details to display for the current selection.

By Editing a Breakpoint… you can make the Breakpoint conditional (prior topic)



Edit a Statement Breakpoint

Required information

Sets a breakpoint to stop execution at a specific source line

☐ Defer breakpoint until executable is loaded

Load Module/DLL/Executable: DBGMAIN

Object/Program/CSECT: DBGMAIN

Source(optional): DDS0001.EQALANGX(DBGMAIN)

Statement: 45

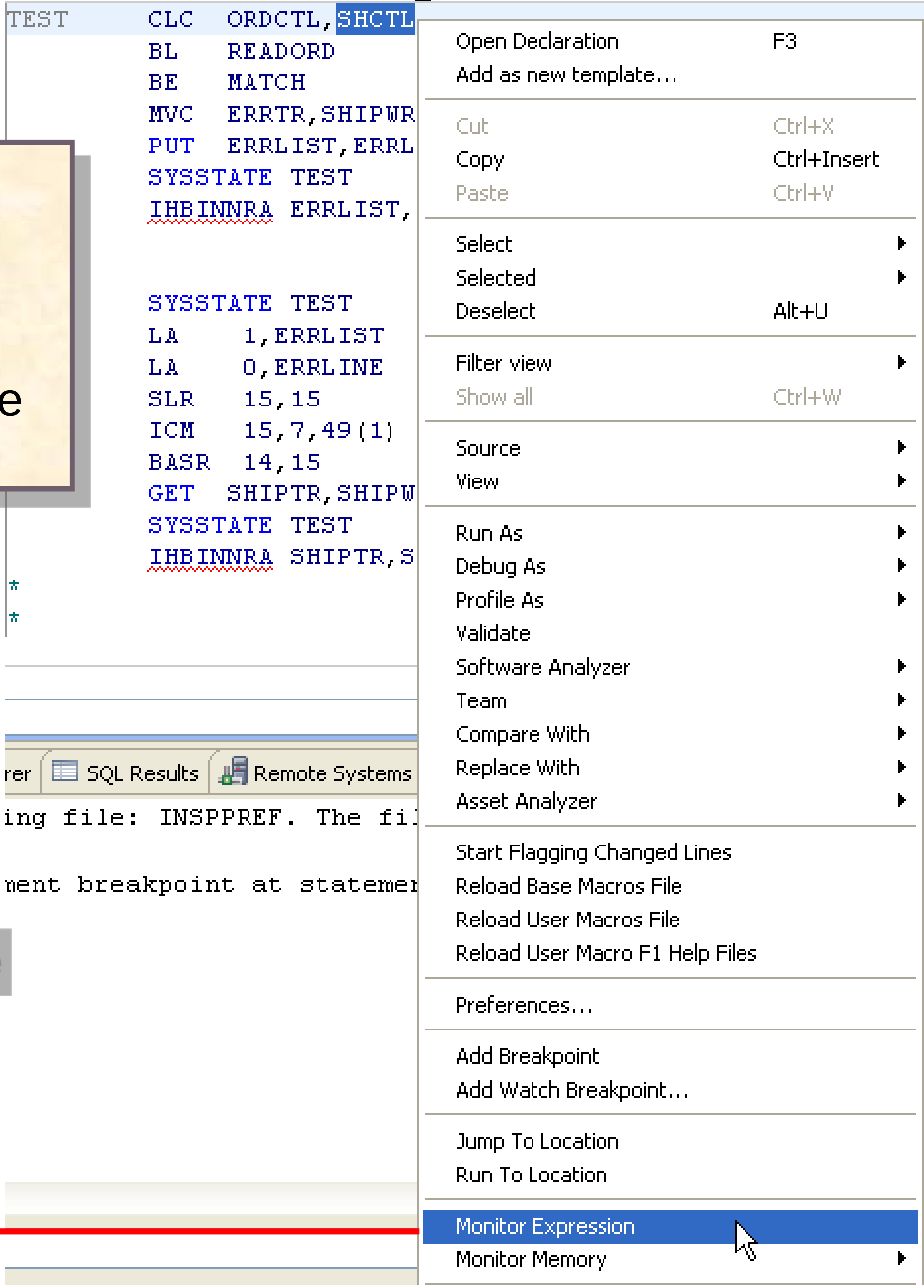
User label (optional):

Monitoring Variable Values

Besides hovering over a variable, you can:

1. Double-click and select any variable

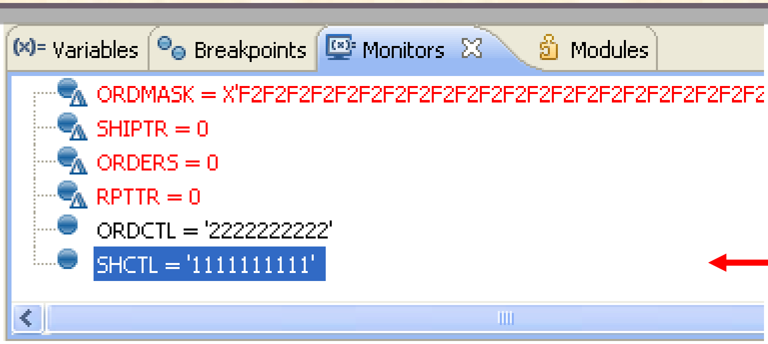
2. Right-click and monitor the variable value throughout your debug session



```
TEST      CLC   ORDCTL,SHCTL
          BL    READORD
          BE    MATCH
          MVC   ERRTR,SHIPWR
          PUT   ERRLIST,ERRL
          SYSSTATE TEST
          IHBINNRA ERRLIST,

          SYSSTATE TEST
          LA     1,ERRLIST
          LA     0,ERRLINE
          SLR    15,15
          ICM    15,7,49(1)
          BASR   14,15
          GET   SHIPTR,SHIPW
          SYSSTATE TEST
          IHBINNRA SHIPTR,S
*
*
```

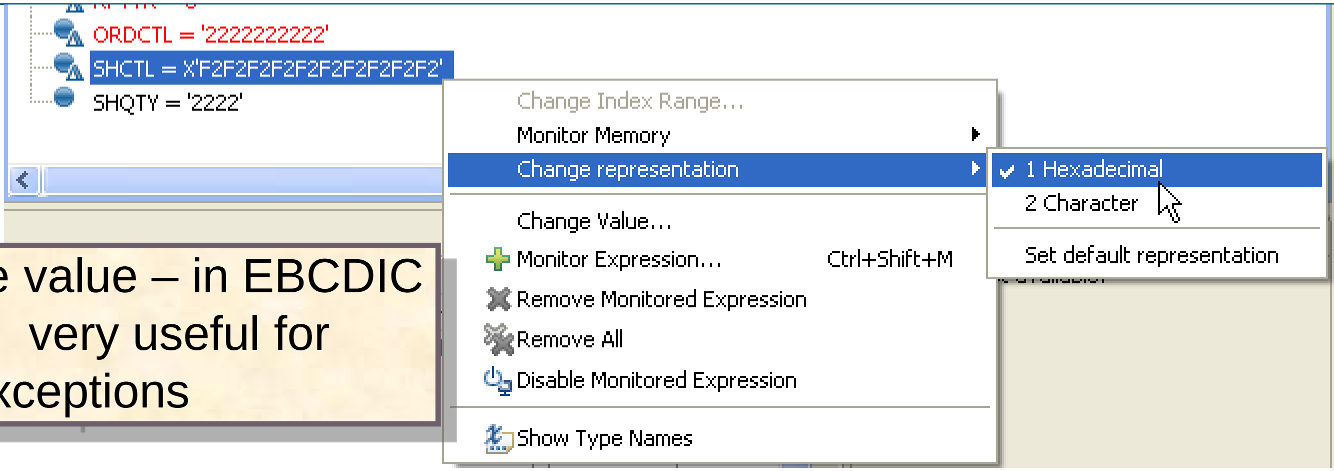
```
ing file: INSPPREF. The fi

ment breakpoint at statemen
```

The Monitors view shows the variable's value



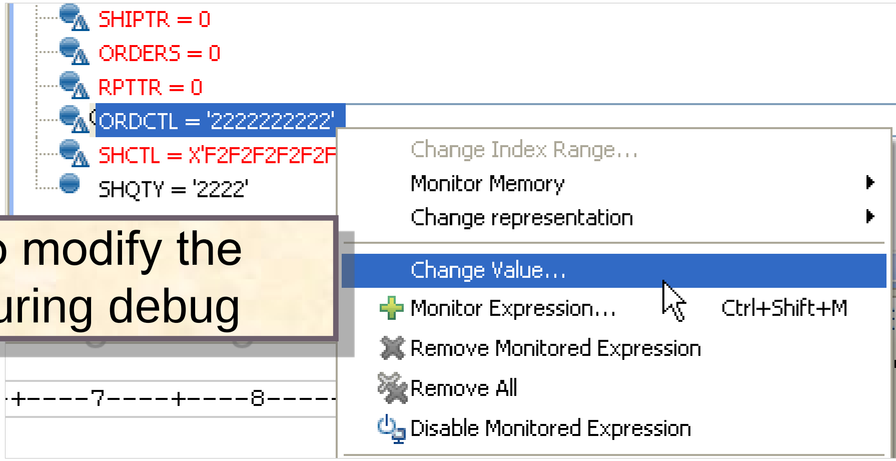
```
ORDMASK = X'F2F2F2F2F2F2F2F2F2F2F2F2F2F2F2F2F2F2F2F2F2F2F2
SHIPTR = 0
ORDERS = 0
RPTTR = 0
ORDCTL = '2222222222'
SHCTL = '1111111111'
```

Monitors View – Options



```
ORDCTL = '2222222222'
SHCTL = X'F2F2F2F2F2F2F2F2F2F2'
SHQTY = '2222'
```

Monitored variable value – in EBCDIC internal display ➔ very useful for debugging data exceptions



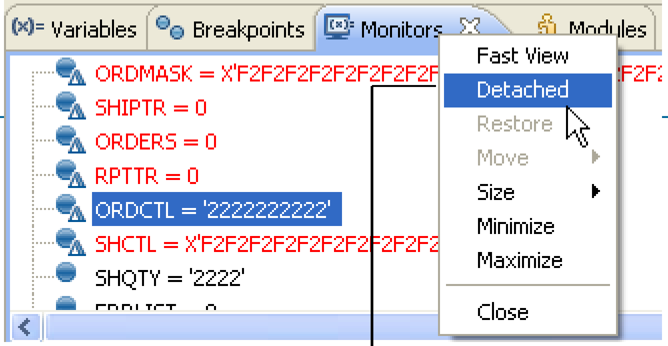
```
SHIPTR = 0
ORDERS = 0
RPTTR = 0
ORDCTL = '2222222222'
SHCTL = X'F2F2F2F2F2F
SHQTY = '2222'
+----7----+----8----
```

Change Index Range…
Monitor Memory
Change representation
Change Value…
Monitor Expression… Ctrl+Shift+M
Remove Monitored Expression
Remove All
Disable Monitored Expression

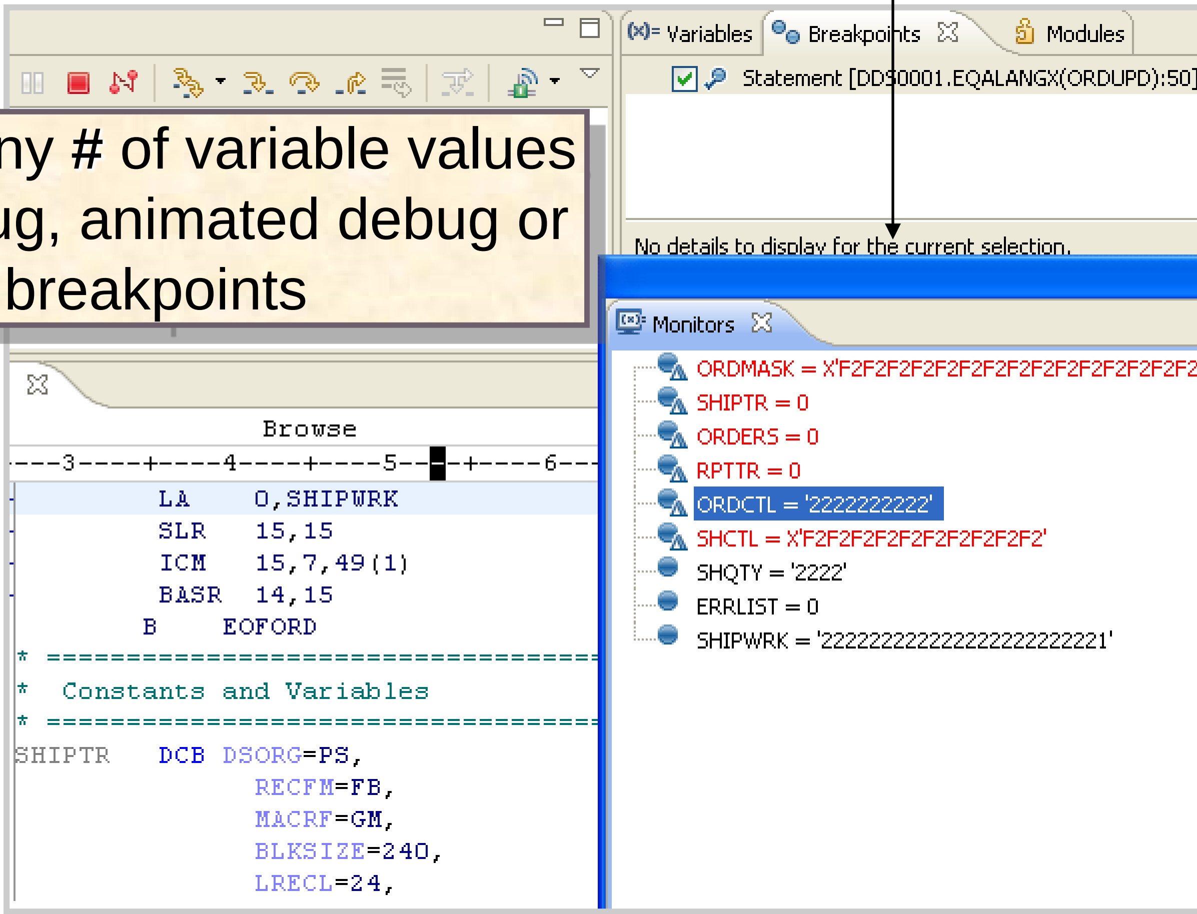
Change Value… allows you to modify the variable's value on the fly – during debug

Detach the Monitors View

A useful Best Practice…



```
ORDMASK = X'F2F2F2F2F2F2F2F2F
SHIPTR = 0
ORDERS = 0
RPTTR = 0
ORDCTL = '2222222222'
SHCTL = X'F2F2F2F2F2F2F2F2F2F2F2
SHQTY = '2222'
```



Statement [DDS0001.EQALANGX(ORDUPD):50]

No details to display for the current selection.

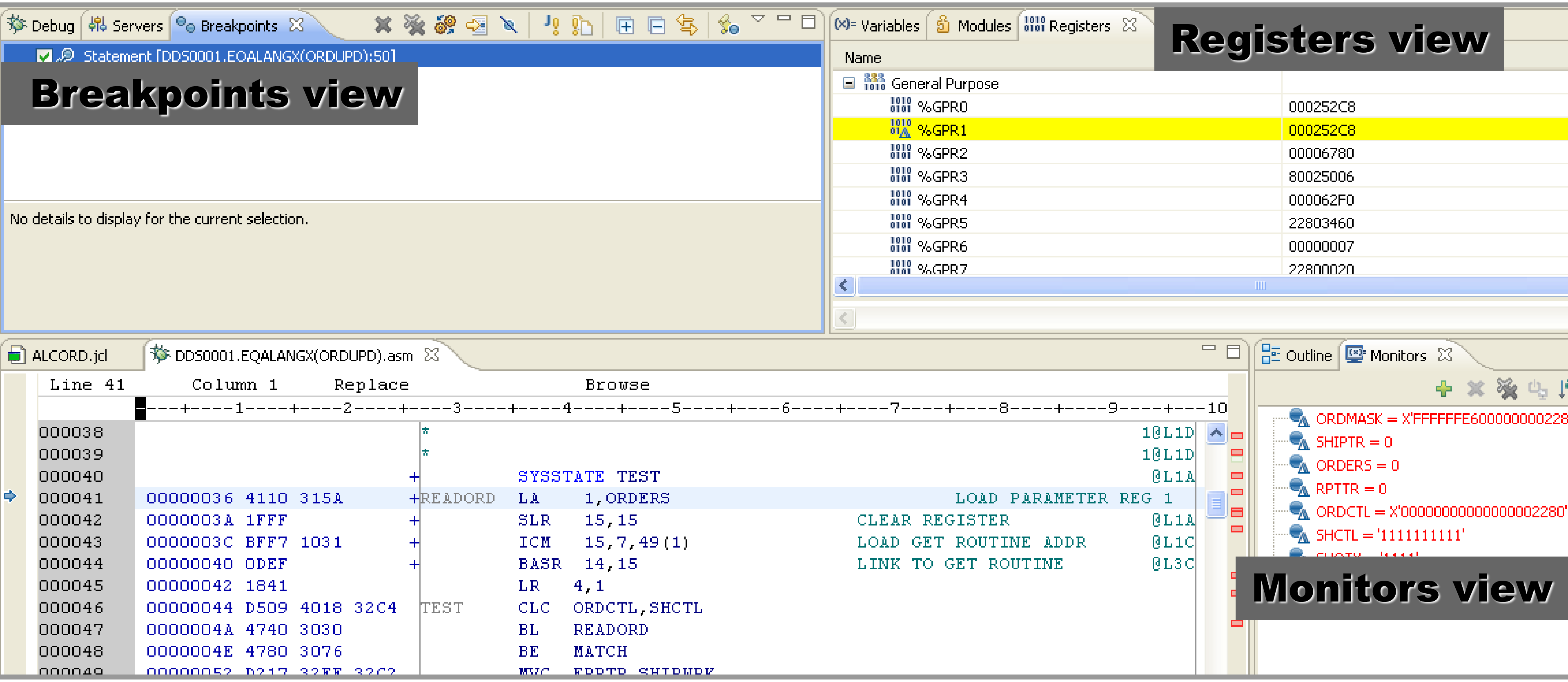
Can view any # of variable values during debug, animated debug or Resume to breakpoints

```
                              Browse
----3----+----4----+----5----+----6---
         LA    0,SHIPWRK
         SLR   15,15
         ICM   15,7,49(1)
         BASR  14,15
        B     EOFORD
* =====================================
*  Constants and Variables
* =====================================
SHIPTR   DCB DSORG=PS,
               RECFM=FB,
               MACRF=GM,
               BLKSIZE=240,
               LRECL=24,
```

```
ORDMASK = X'F2F2F2F2F2F2F2F2F2F2F2F2F2F2F2
SHIPTR = 0
ORDERS = 0
RPTTR = 0
ORDCTL = '2222222222'
SHCTL = X'F2F2F2F2F2F2F2F2F2F2'
SHQTY = '2222'
ERRLIST = 0
SHIPWRK = '222222222222222222222221'
```

Making Optimal use of Screen Real Estate

- Some of the Debug Perspective views are not enabled for Assembler programs: Variables, Outline, etc.
- Along with detaching views, consider moving the useful Assembler views "front-and-center" to maximize your screen real estate – adding to your ability to see as much useful information at a glance



☑ Statement [DDS0001.EQALANGX(ORDUPD):50]

No details to display for the current selection.

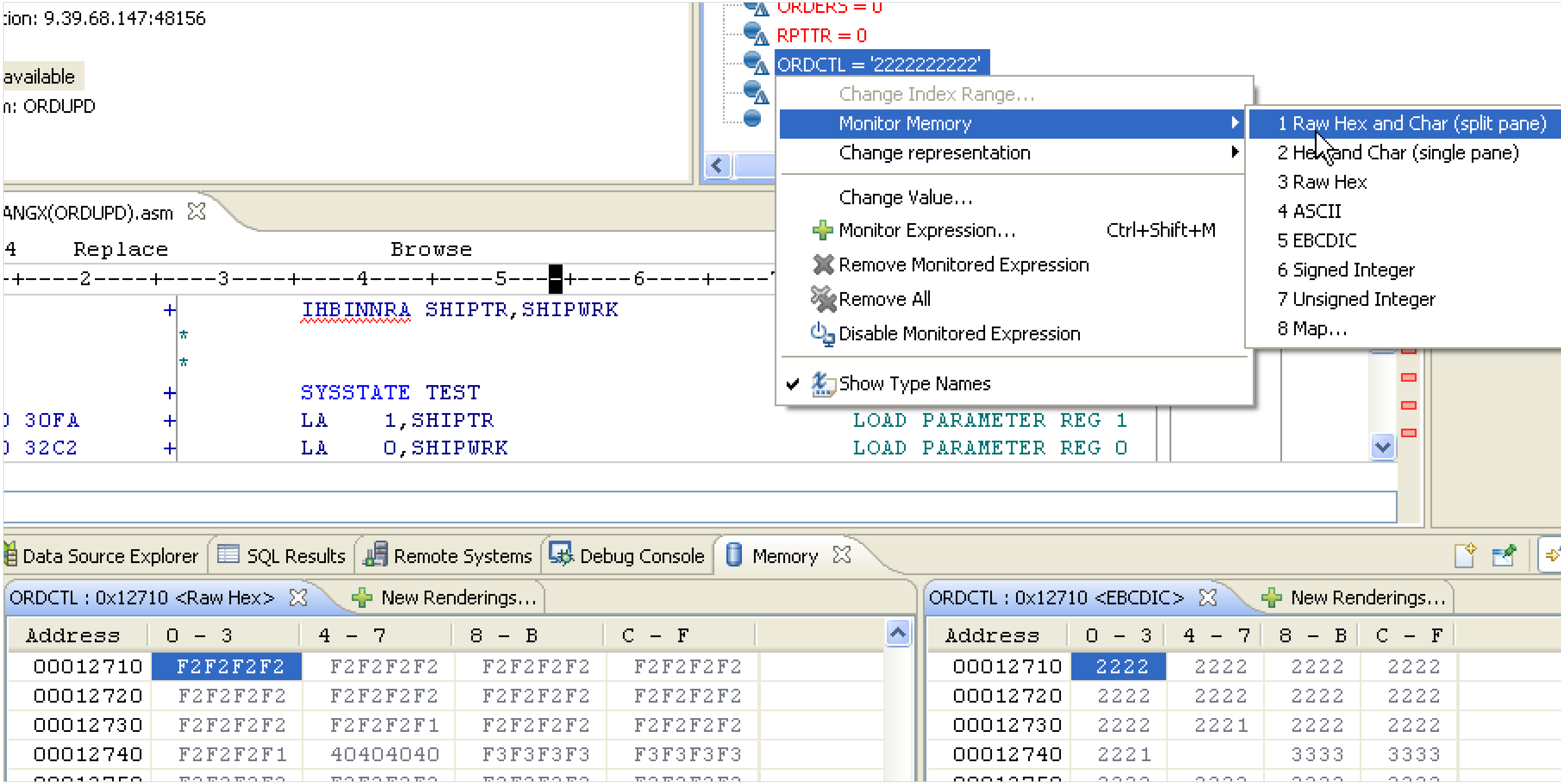
Name	
⊟ General Purpose	
%GPR0	000252C8
%GPR1	000252C8
%GPR2	00006780
%GPR3	80025006
%GPR4	000062F0
%GPR5	22803460
%GPR6	00000007
%GPR7	22800020

```
 Line 41       Column 1      Replace                 Browse
 ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----9----+---10
000038                              *                                                      1@L1D
000039                              *                                                      1@L1D
000040                             +          SYSSTATE TEST                                 @L1A
000041 00000036 4110 315A          +READORD   LA    1,ORDERS                   LOAD PARAMETER REG 1
000042 0000003A 1FFF               +          SLR   15,15             CLEAR REGISTER            @L1A
000043 0000003C BFF7 1031          +          ICM   15,7,49(1)        LOAD GET ROUTINE ADDR     @L1C
000044 00000040 0DEF               +          BASR  14,15             LINK TO GET ROUTINE       @L3C
000045 00000042 1841                          LR   4,1
000046 00000044 D509 4018 32C4      TEST      CLC  ORDCTL,SHCTL
000047 0000004A 4740 3030                     BL   READORD
000048 0000004E 4780 3076                     BE   MATCH
```

- ORDMASK = X'FFFFFFE600000000228
- SHIPTR = 0
- ORDERS = 0
- RPTTR = 0
- ORDCTL = X'00000000000000002280'
- SHCTL = '111111111'

Monitor Memory

- Monitor Memory
  - The memory content can be shown (or “rendered”) in several different formats:
    - Raw HEX, EDBCDIC or ASCII
    - Tree structure using customized XML mappings.

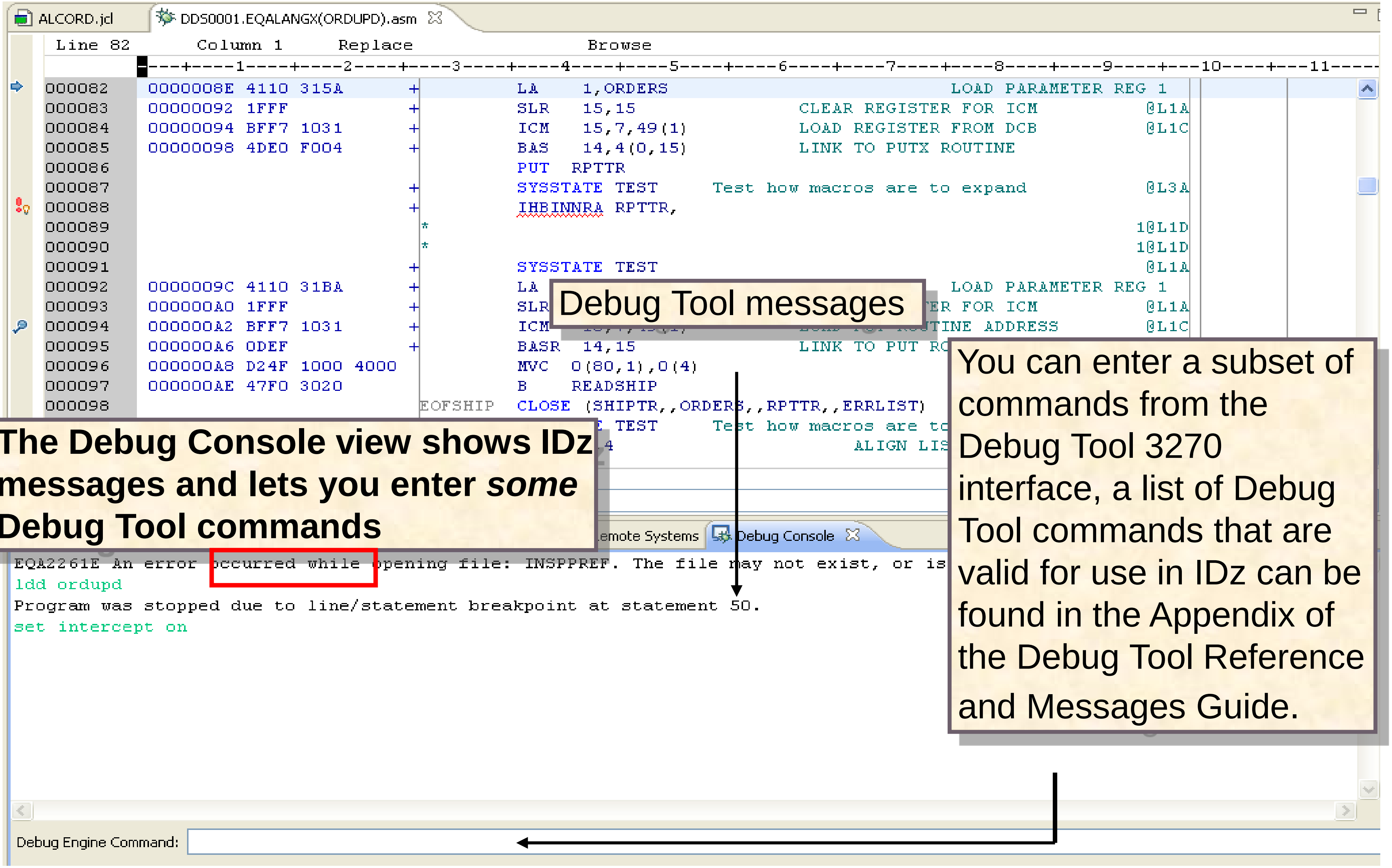


```
4      Replace                     Browse
-+----2----+----3----+----4----+----5----+----6----+----
               +          IHBINNRA SHIPTR,SHIPWRK
                *
                *
               +          SYSSTATE TEST
) 30FA         +          LA     1,SHIPTR                LOAD PARAMETER REG 1
) 32C2         +          LA     0,SHIPWRK               LOAD PARAMETER REG 0
```

Address	0 - 3	4 - 7	8 - B	C - F
00012710	F2F2F2F2	F2F2F2F2	F2F2F2F2	F2F2F2F2
00012720	F2F2F2F2	F2F2F2F2	F2F2F2F2	F2F2F2F2
00012730	F2F2F2F2	F2F2F2F1	F2F2F2F2	F2F2F2F2
00012740	F2F2F2F1	40404040	F3F3F3F3	F3F3F3F3

Address	0 - 3	4 - 7	8 - B	C - F
00012710	2222	2222	2222	2222
00012720	2222	2222	2222	2222
00012730	2222	2221	2222	2222
00012740	2221		3333	3333

The Debug Console View



```
 Line 82       Column 1      Replace            Browse
         ----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----9----+---10----+---11----
000082  0000008E 4110 315A       +        LA    1,ORDERS                       LOAD PARAMETER REG 1
000083  00000092 1FFF            +        SLR   15,15           CLEAR REGISTER FOR ICM            @L1A
000084  00000094 BFF7 1031       +        ICM   15,7,49(1)      LOAD REGISTER FROM DCB            @L1C
000085  00000098 4DE0 F004       +        BAS   14,4(0,15)      LINK TO PUTX ROUTINE
000086                                    PUT   RPTTR
000087                           +        SYSSTATE TEST     Test how macros are to expand         @L3A
000088                           +        IHBINNRA RPTTR,
000089                            *                                                              1@L1D
000090                            *                                                              1@L1D
000091                           +        SYSSTATE TEST                                           @L1A
000092  0000009C 4110 31BA       +        LA                                   LOAD PARAMETER REG 1
000093  000000A0 1FFF            +        SLR                              ER FOR ICM             @L1A
000094  000000A2 BFF7 1031       +        ICM                              TINE ADDRESS           @L1C
000095  000000A6 0DEF            +        BASR  14,15           LINK TO PUT RO
000096  000000A8 D24F 1000 4000           MVC   0(80,1),0(4)
000097  000000AE 47F0 3020                B     READSHIP
000098                           EOFSHIP  CLOSE (SHIPTR,,ORDERS,,RPTTR,,ERRLIST)
                                                TEST     Test how macros are to
                                          4                    ALIGN LIS
```

```
EQA2261E An error occurred while opening file: INSPPREF. The file may not exist, or is
ldd ordupd
Program was stopped due to line/statement breakpoint at statement 50.
set intercept on
```

Debug Tool messages

The Debug Console view shows IDz messages and lets you enter *some* Debug Tool commands

You can enter a subset of commands from the Debug Tool 3270 interface, a list of Debug Tool commands that are valid for use in IDz can be found in the Appendix of the Debug Tool Reference and Messages Guide.

Place your cursor in the Command area and press Ctrl+Spacebar – to see a list of available commands

Debug Console Commands – Tracing Statement Execution

This is another very popular command:

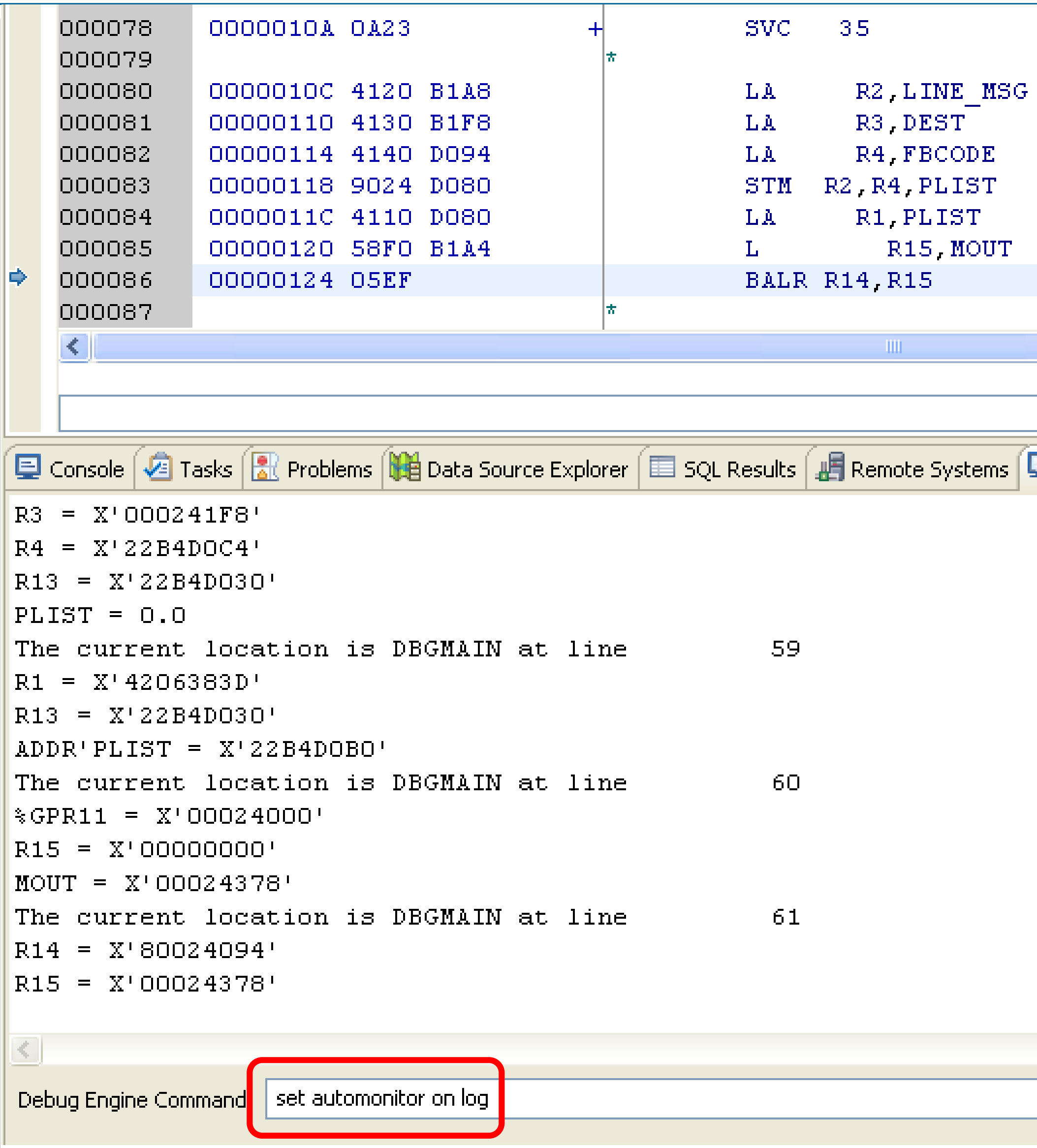
SET AUTOMONITOR ON LOG

It forces Debug Tool to track each statement as it's executed and write it to the Debug Console

Using this technique you can copy and paste your program's dynamic execution and trace forward and backward through any portion of your code

You can also copy all of the statements to hard-copy :

1. **Right-click**
2. **Select Export History**
3. **Specify a file – preferably an RTF or MS-Word doc, as formatting will be retained**



```
000078   0000010A 0A23          +        SVC    35
000079                           *
000080   0000010C 4120 B1A8               LA     R2,LINE_MSG
000081   00000110 4130 B1F8               LA     R3,DEST
000082   00000114 4140 D094               LA     R4,FBCODE
000083   00000118 9024 D080               STM  R2,R4,PLIST
000084   0000011C 4110 D080               LA     R1,PLIST
000085   00000120 58F0 B1A4               L        R15,MOUT
000086   00000124 05EF                    BALR R14,R15
000087                           *
```

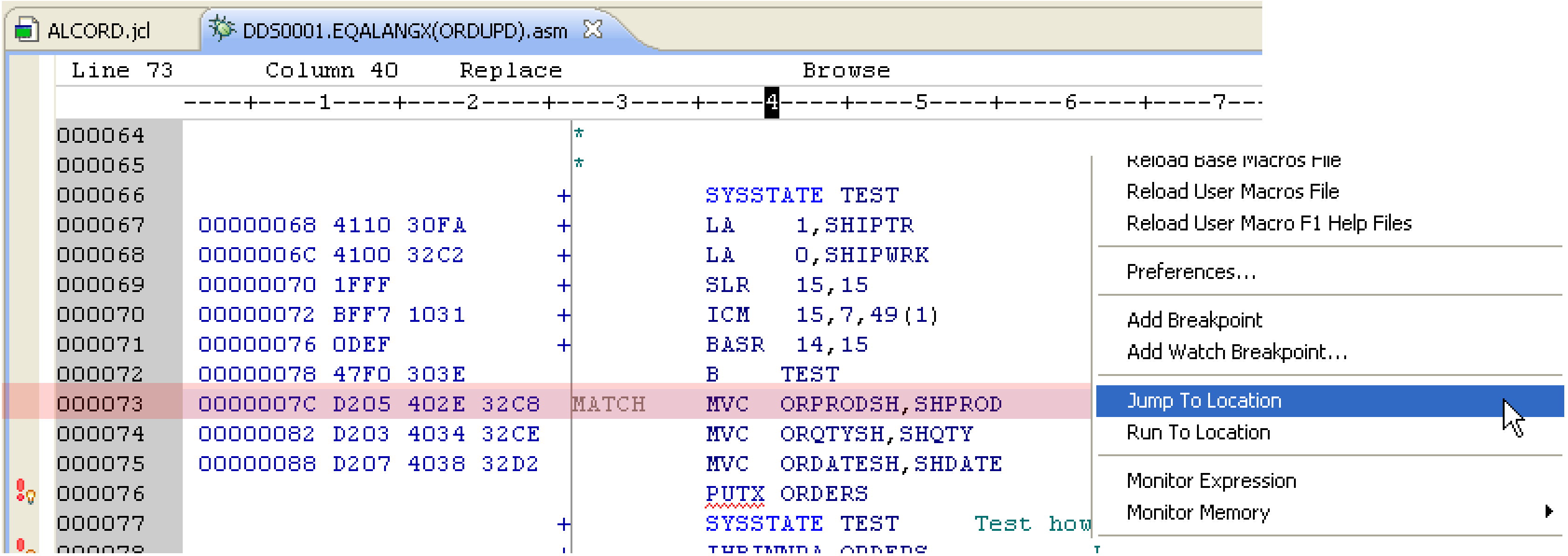
```
R3  = X'000241F8'
R4  = X'22B4D0C4'
R13  = X'22B4D030'
PLIST = 0.0
The current location is DBGMAIN at line          59
R1  = X'4206383D'
R13  = X'22B4D030'
ADDR'PLIST = X'22B4D0B0'
The current location is DBGMAIN at line          60
%GPR11 = X'00024000'
R15  = X'00000000'
MOUT  = X'00024378'
The current location is DBGMAIN at line          61
R14  = X'80024094'
R15  = X'00024378'
```

Debug Engine Command: set automonitor on log

Debug Option – Jump to / Run To

- **Jump to Location** - skip over sections of code to avoid executing certain statements or move to a position where certain statements can be executed again. Useful:
  - To avoid called programs or I/OS to a not available dataset
  - Or to iteratively execute some statements of interest

- **Run to Location** - executes all statements between the current location and the run-to location.

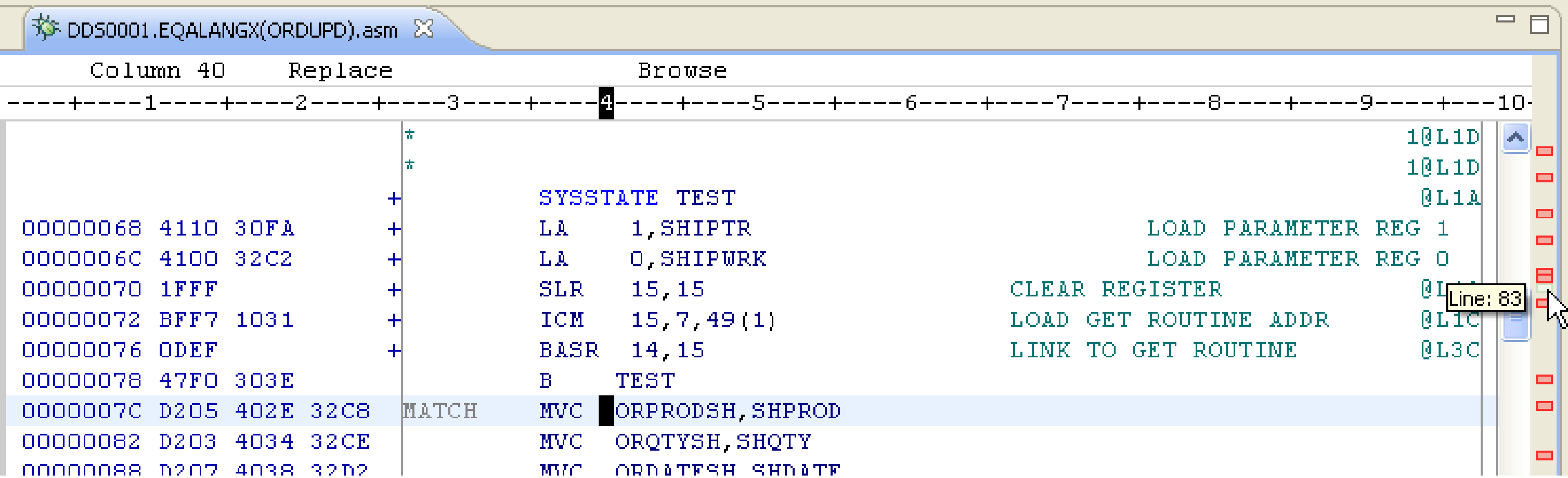


```
 Line 73        Column 40    Replace                      Browse
         ----+----1----+----2----+----3----+----4----+----5----+----6----+----7--
000064                             *
000065                             *
000066                            +         SYSSTATE TEST
000067   00000068 4110 30FA       +         LA     1,SHIPTR
000068   0000006C 4100 32C2       +         LA     0,SHIPWRK
000069   00000070 1FFF            +         SLR    15,15
000070   00000072 BFF7 1031       +         ICM    15,7,49(1)
000071   00000076 0DEF            +         BASR   14,15
000072   00000078 47F0 303E                 B    TEST
000073   0000007C D205 402E 32C8  MATCH     MVC  ORPRODSH,SHPROD
000074   00000082 D203 4034 32CE            MVC  ORQTYSH,SHQTY
000075   00000088 D207 4038 32D2            MVC  ORDATESH,SHDATE
000076                                      PUTX ORDERS
000077                            +         SYSSTATE TEST      Test how
```

Context Menu

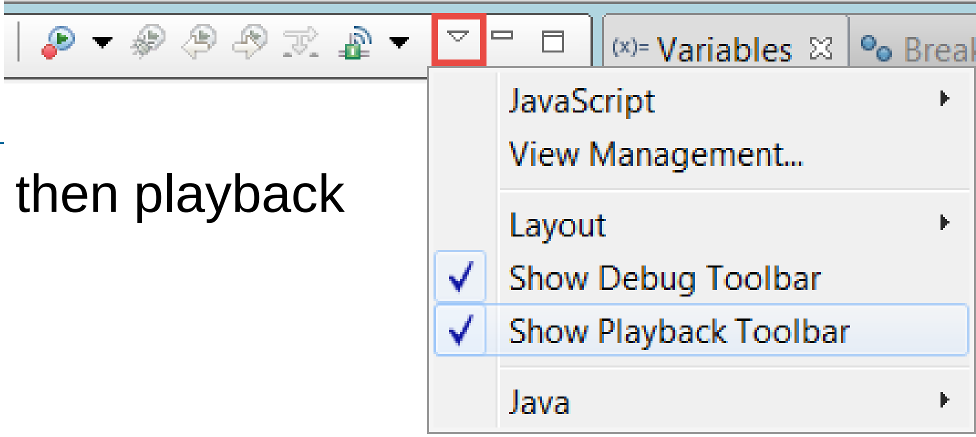
How to return from anywhere in your program to the Current Instruction

- To get back to the Current Instruction Pointer (the "next sequential instruction") – if you've navigated away within the source:
  - **Click the small blue rectangle in the right-hand margin of your source code**



```
    Column 40      Replace                        Browse
----+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----9----+---10-
                                    *                                                         1@L1D
                                    *                                                         1@L1D
                                   +           SYSSTATE TEST                                   @L1A
00000068 4110 30FA                 +           LA     1,SHIPTR                  LOAD PARAMETER REG 1
0000006C 4100 32C2                 +           LA     0,SHIPWRK                 LOAD PARAMETER REG 0
00000070 1FFF                      +           SLR    15,15            CLEAR REGISTER              @L
00000072 BFF7 1031                 +           ICM    15,7,49(1)       LOAD GET ROUTINE ADDR       @L1C
00000076 0DEF                      +           BASR   14,15            LINK TO GET ROUTINE         @L3C
00000078 47F0 303E                             B     TEST
0000007C D205 402E 32C8 MATCH      MVC   ORPRODSH,SHPROD
00000082 D203 4034 32CE            MVC   ORQTYSH,SHQTY
00000088 D207 4038 32D2            MVC   ORDATESH,SHDATE
```

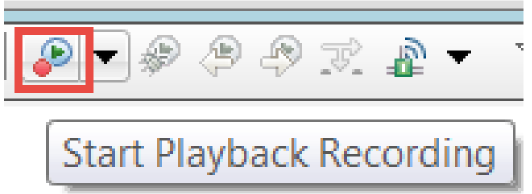
Record and Playback



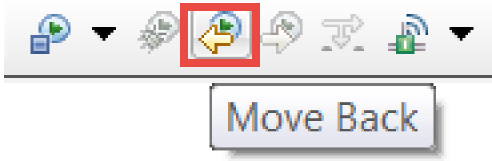
- Debug Tool allows you to record and then playback recorded statements during Debug

Steps:

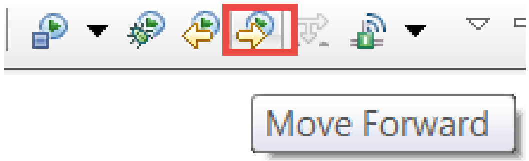
- From the Debug toolbar
  - Click the white downward-pointing triangle, and select:
    ✓ **Show Playback Toolbar**
  - From the Playback toolbar, click the green-go button, to start playback recording
    - All of your statements are being recorded from that point until you:
      - Stop recording
      - End the Debug Session (ABEND or normal EOJ)



  - If your program pauses (Breakpoint, etc.) you can backtrack through the recorded statements by pressing the **Move Back** icon on the toolbar

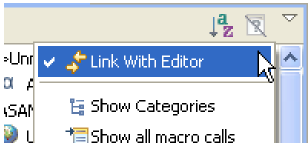


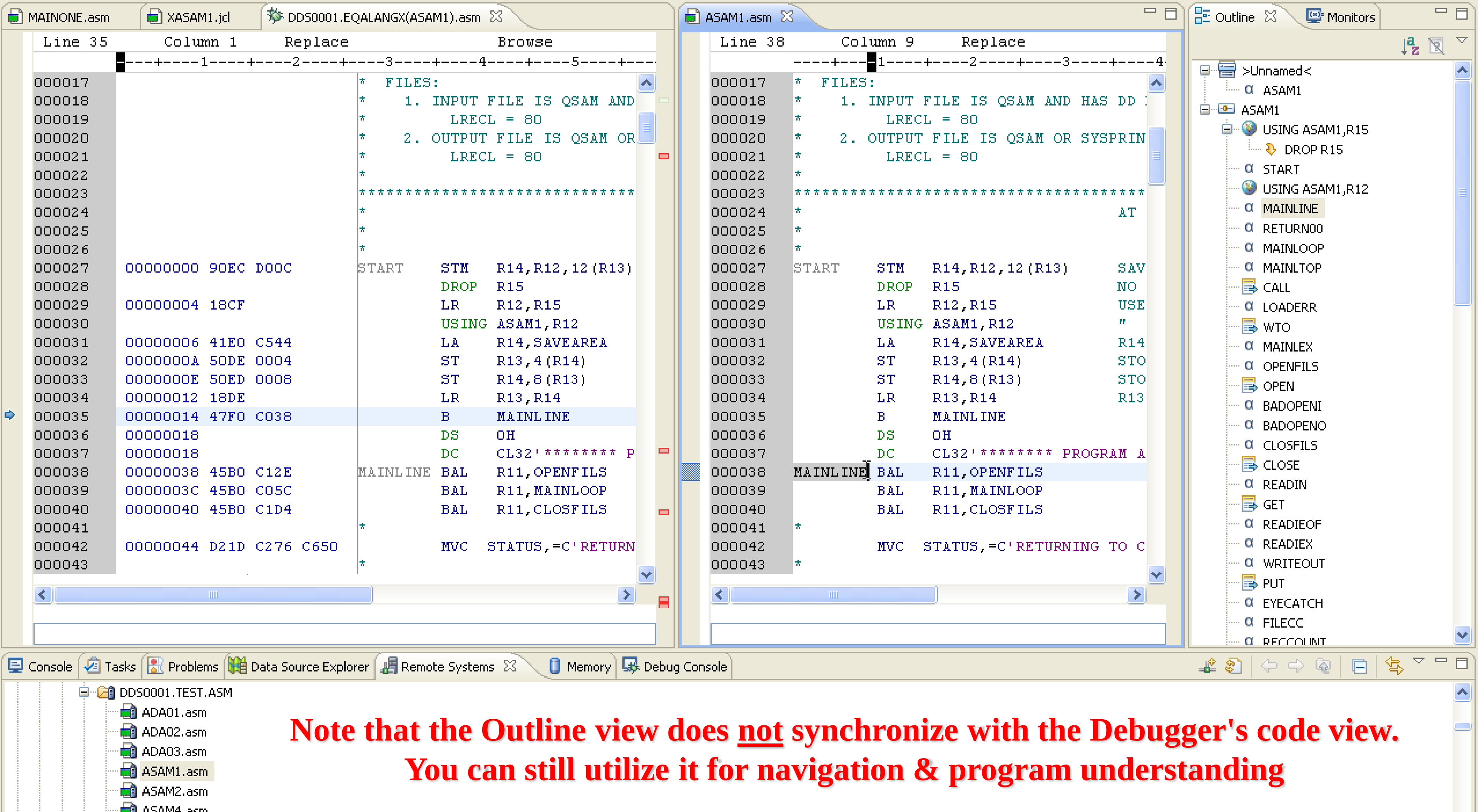
  - You can also play the recorded statements forward, by clicking **Move Forward** on the toolbar



Utilizing the Outline View

- To enable the Outline View during your Debugging session:
  - From Remote Systems – open the program
  - Manipulate the View size/window proportion, and ensure that the Outline view synchronizes with the source file editor





```
 Line 35       Column 1       Replace                Browse
           ----+----1----+----2----+----3----+----4----+----5----+---
000017                                *  FILES:
000018                                *    1. INPUT FILE IS QSAM AND
000019                                *         LRECL = 80
000020                                *    2. OUTPUT FILE IS QSAM OR
000021                                *         LRECL = 80
000022                                *
000023                                ******************************
000024                                *
000025                                *
000026                                *
000027 00000000 90EC D00C             START     STM   R14,R12,12(R13)
000028                                          DROP  R15
000029 00000004 18CF                            LR    R12,R15
000030                                          USING ASAM1,R12
000031 00000006 41E0 C544                       LA    R14,SAVEAREA
000032 0000000A 50DE 0004                       ST    R13,4(R14)
000033 0000000E 50ED 0008                       ST    R14,8(R13)
000034 00000012 18DE                            LR    R13,R14
000035 00000014 47F0 C038                       B     MAINLINE
000036 00000018                                 DS    0H
000037 00000018                                 DC    CL32'******** P
000038 00000038 45B0 C12E             MAINLINE  BAL   R11,OPENFILS
000039 0000003C 45B0 C05C                       BAL   R11,MAINLOOP
000040 00000040 45B0 C1D4                       BAL   R11,CLOSFILS
000041                                *
000042 00000044 D21D C276 C650                  MVC  STATUS,=C'RETURN
000043                                *
```

```
 Line 38       Column 9       Replace
           ----+---1----+----2----+----3----+----4
000017 *  FILES:
000018 *    1. INPUT FILE IS QSAM AND HAS DD
000019 *         LRECL = 80
000020 *    2. OUTPUT FILE IS QSAM OR SYSPRIN
000021 *         LRECL = 80
000022 *
000023 **************************************
000024 *                                    AT
000025 *
000026 *
000027 START     STM   R14,R12,12(R13)      SAV
000028           DROP  R15                  NO
000029           LR    R12,R15              USE
000030           USING ASAM1,R12            "
000031           LA    R14,SAVEAREA         R14
000032           ST    R13,4(R14)           STO
000033           ST    R14,8(R13)           STO
000034           LR    R13,R14              R13
000035           B     MAINLINE
000036           DS    0H
000037           DC    CL32'******** PROGRAM A
000038 MAINLINE  BAL   R11,OPENFILS
000039           BAL   R11,MAINLOOP
000040           BAL   R11,CLOSFILS
000041 *
000042           MVC  STATUS,=C'RETURNING TO C
000043 *
```

Note that the Outline view does not synchronize with the Debugger's code view. You can still utilize it for navigation & program understanding

Handling program abends

- Debug Tool can receive control back from the system after an abend occurs
  - The program will be stopped at the abending statement

- You can:
  - Allow the application to abend and terminate
    - Capture abend info with a product such as Fault Analyzer
    - Terminate the application and prevent further processing
  - Or continue running the program

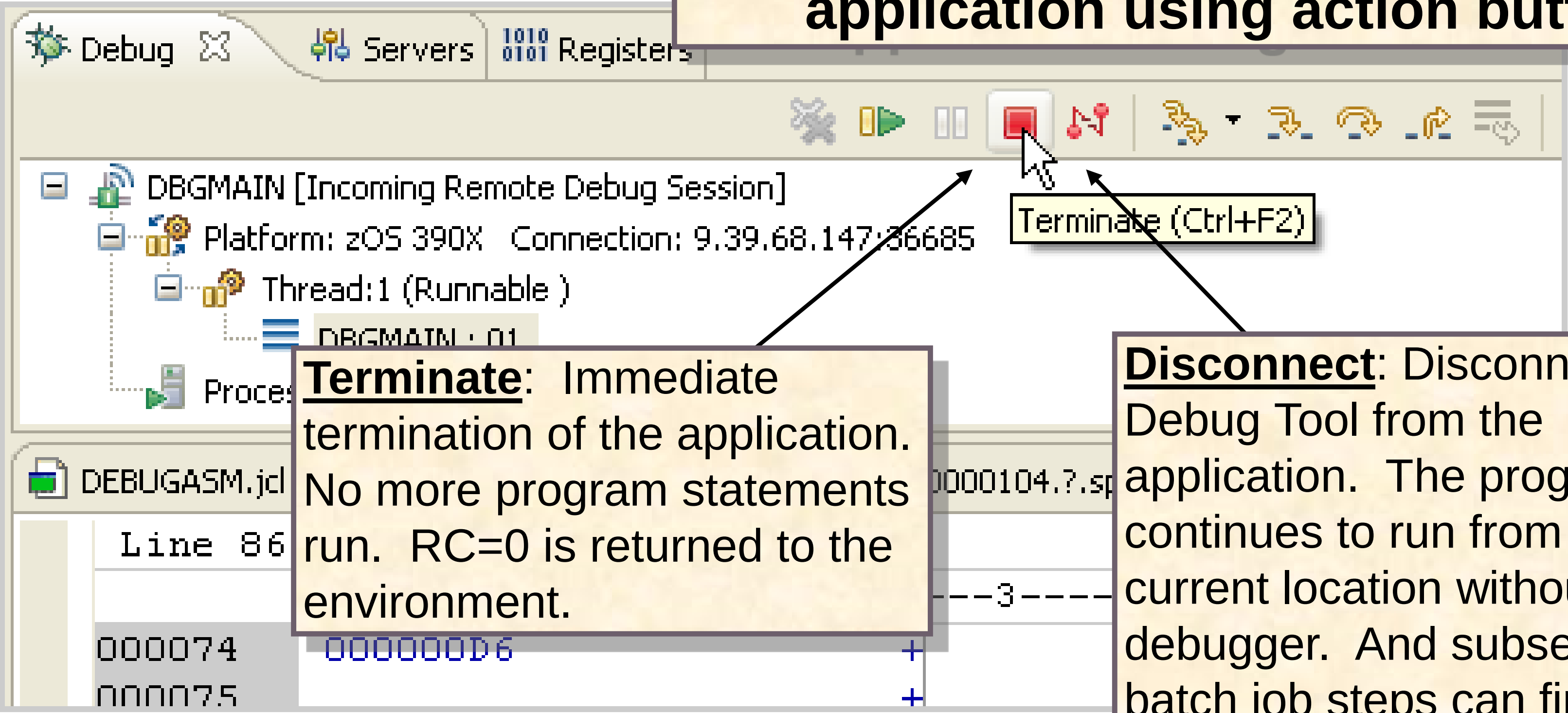
- Usage note:
  - The **LE TRAP(ON)** option must be active

Terminating the application

- There are several options for terminating your application:
  - **Remain in the debugger, and RESUME until the program runs to completion**
    - The program will terminate normally or with an abend
    - The return code is controlled by the program
  - **Disconnect the debugger, and allow the program to run to completion**
    - The program will terminate normally or with an abend
    - The return code is controlled by the program

Termination action buttons

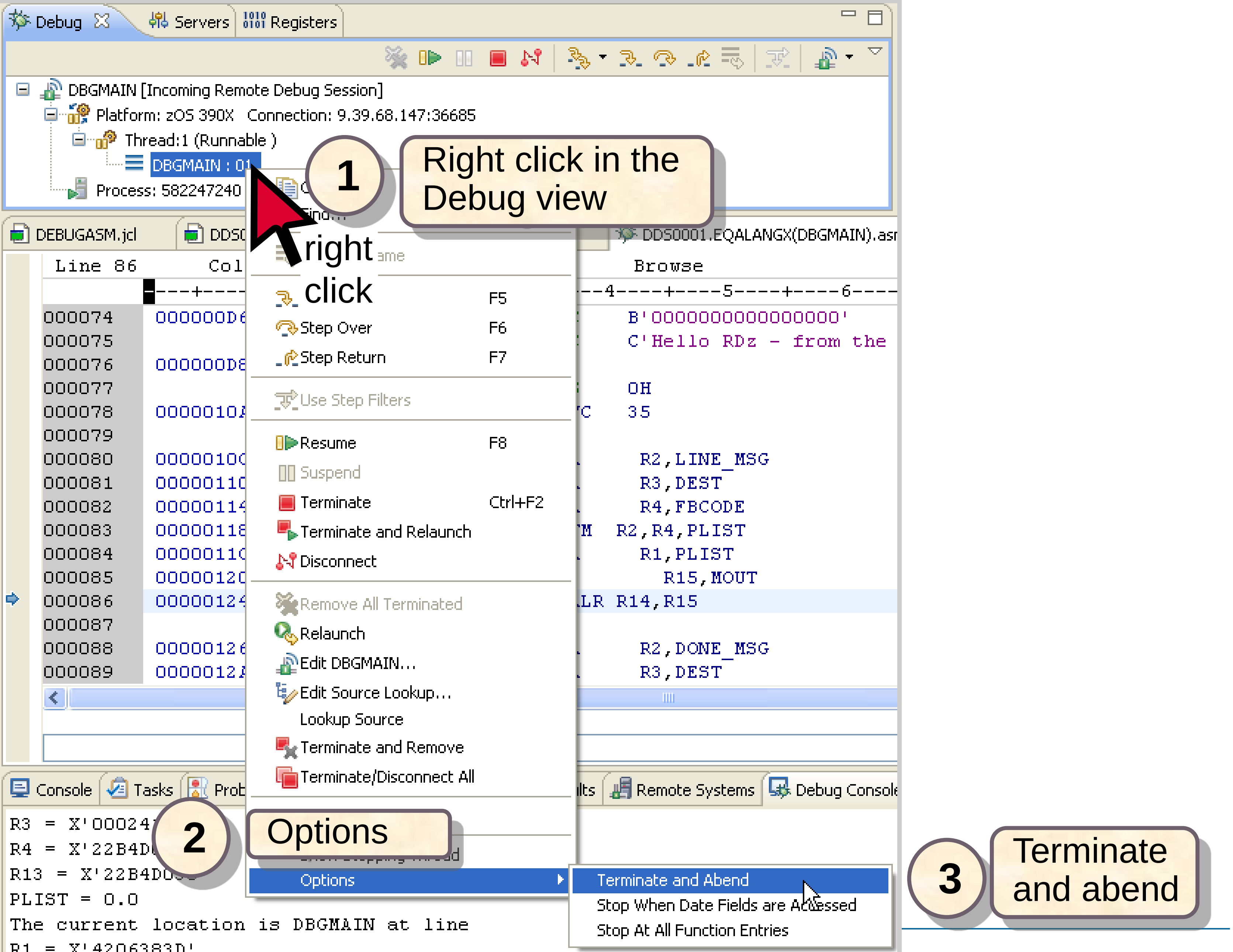
You can immediately terminate the application using action buttons



<u>Terminate</u>: Immediate termination of the application. No more program statements run. RC=0 is returned to the environment.

<u>Disconnect</u>: Disconnect Debug Tool from the application. The program continues to run from the current location without the debugger. And subsequent batch job steps can finish as well.

Force an immediate termination with abend



```
Line 86      Col                  Browse
        ---+----                 --4----+----5----+----6----
000074  000000D6                    B'0000000000000000'
000075                              C'Hello RDz - from the
000076  000000D8
000077                              OH
000078  0000010A               C    35
000079
000080  0000010C                     R2,LINE_MSG
000081  0000011C                     R3,DEST
000082  00000114                     R4,FBCODE
000083  00000118               M    R2,R4,PLIST
000084  0000011C                     R1,PLIST
000085  00000120                        R15,MOUT
000086  00000124               LR   R14,R15
000087
000088  00000126                     R2,DONE_MSG
000089  0000012A                     R3,DEST
```

```
R3 = X'00024
R4 = X'22B4D
R13 = X'22B4D0
PLIST = 0.0
The current location is DBGMAIN at line
R1 = X'4206383D'
```

1 Right click in the Debug view

2 Options

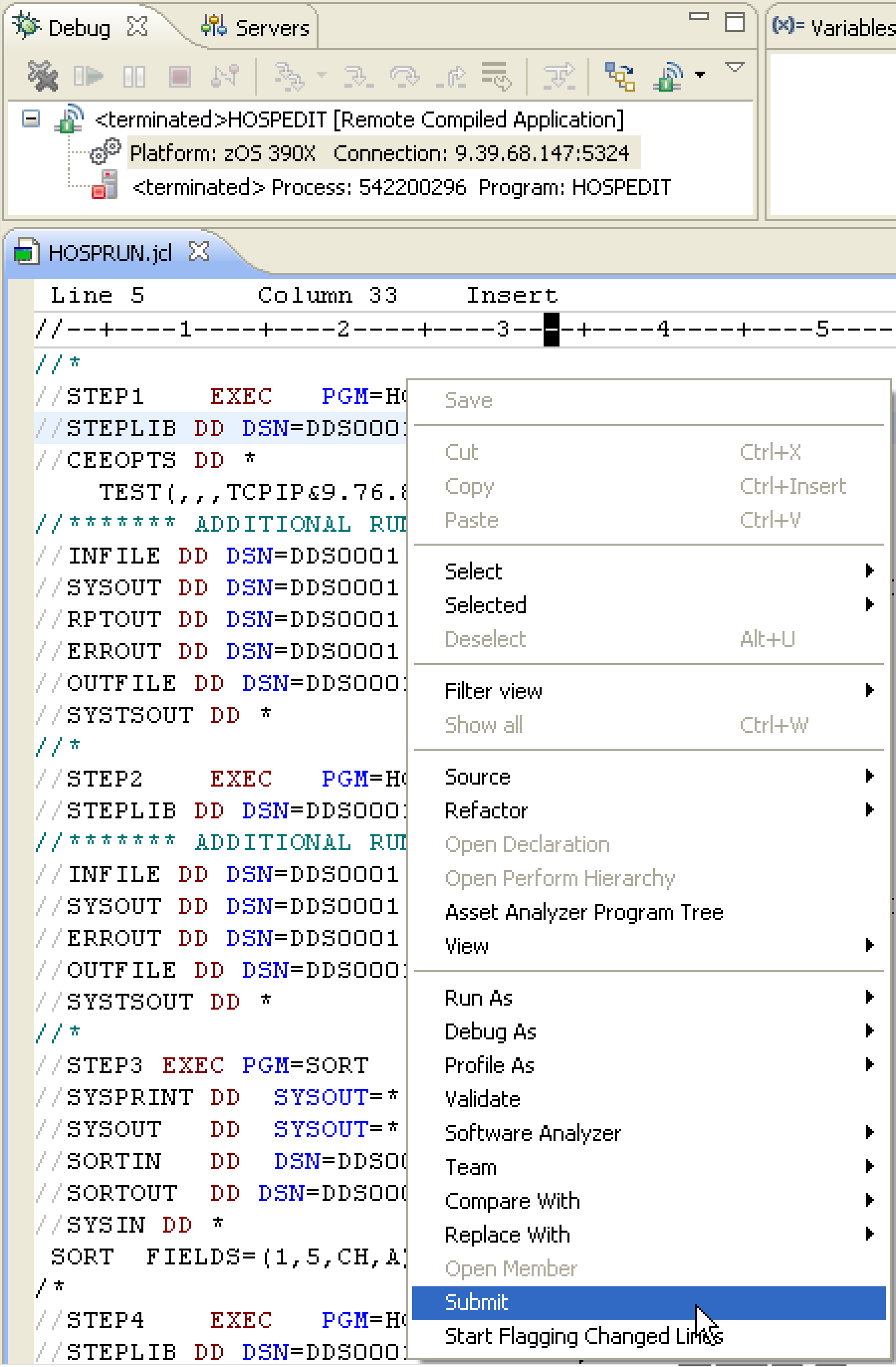
3 Terminate and abend

Restart Your Debugging Session

For batch debugging

- If your submitted JCL is still in the code (Content) area
  - No need to return to the z/OS Projects perspective
- **Right-click**
- Select: **Submit**

✎Note that F11 (or Debug from the Run menu) does <u>NOT</u> work – as it did with Local Assembler debugging



<terminated>HOSPEDIT [Remote Compiled Application]
Platform: zOS 390X   Connection: 9.39.68.147:5324
<terminated> Process: 542200296 Program: HOSPEDIT

```
 Line 5          Column 33      Insert
//--+----1----+----2----+----3----+----4----+----5----
//*
//STEP1     EXEC    PGM=H
//STEPLIB DD DSN=DDS000
//CEEOPTS DD *
   TEST(,,,TCPIP&9.76.
//******* ADDITIONAL RU
//INFILE DD DSN=DDS0001
//SYSOUT DD DSN=DDS0001
//RPTOUT DD DSN=DDS0001
//ERROUT DD DSN=DDS0001
//OUTFILE DD DSN=DDS000
//SYSTSOUT DD *
//*
//STEP2     EXEC    PGM=H
//STEPLIB DD DSN=DDS000
//******* ADDITIONAL RU
//INFILE DD DSN=DDS0001
//SYSOUT DD DSN=DDS0001
//ERROUT DD DSN=DDS0001
//OUTFILE DD DSN=DDS000
//SYSTSOUT DD *
//*
//STEP3 EXEC PGM=SORT
//SYSPRINT DD  SYSOUT=*
//SYSOUT   DD  SYSOUT=*
//SORTIN   DD  DSN=DDS0
//SORTOUT  DD DSN=DDS00
//SYSIN DD *
 SORT  FIELDS=(1,5,CH,A)
/*
//STEP4     EXEC    PGM=H
//STEPLIB DD DSN=DDS000
```

Summary

Having completed this unit, you should now be able to:

- Describe where the debug engines are located
- Show how to set the workbench preferences for running and debugging
- Show how to invoke the debugger for local programs
- Describe the views of the Debug perspective
- Demonstrate how to set breakpoints in Assembler code
- Explain how to set up the Assembler compile options for remote debugging
- Show how to debug a remote batch Assembler program

UNIT

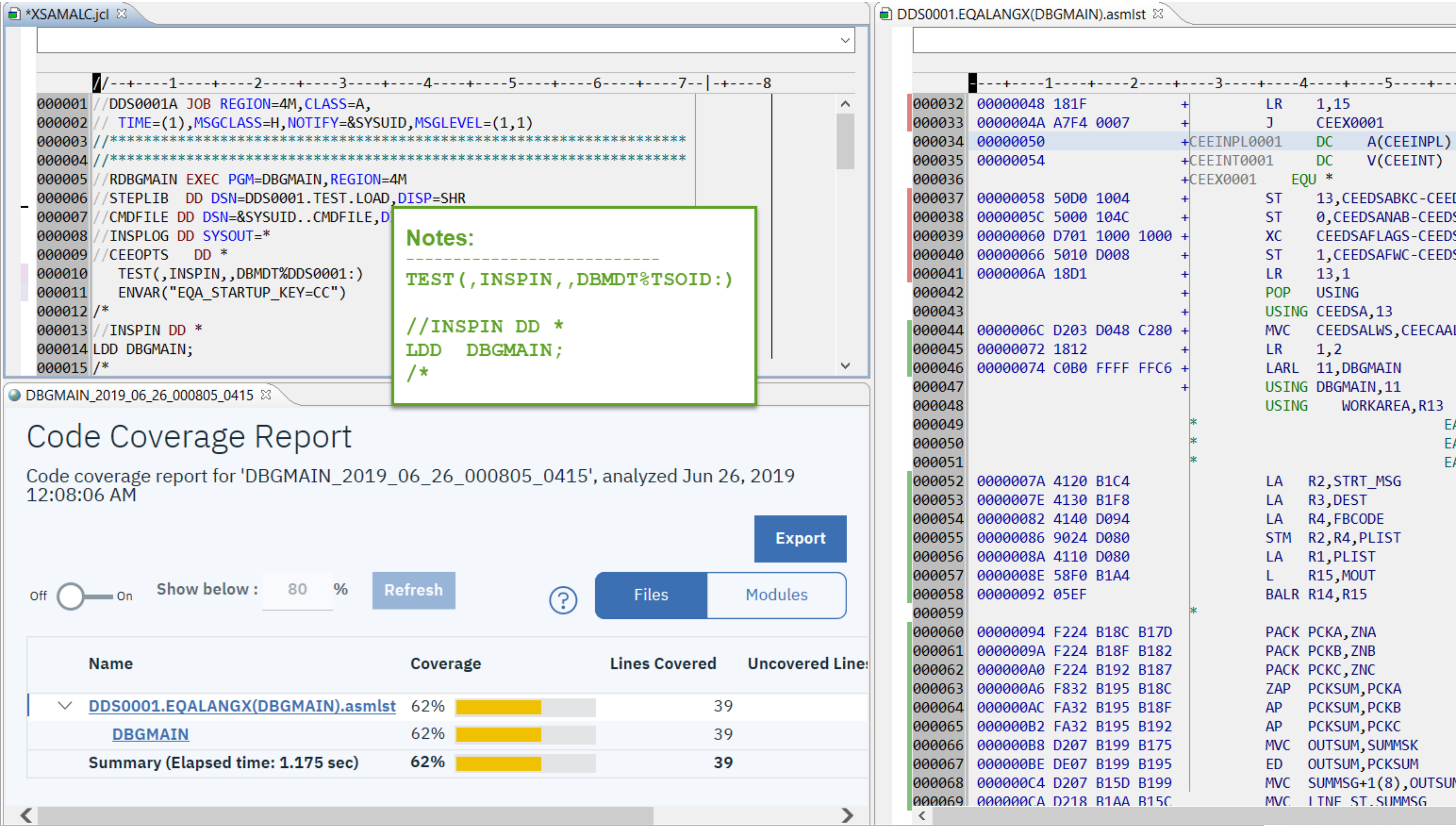
The IDz Workbench

Topics:

- Debugging z/OS Assembler Batch Applications
- **Code Coverage - for Assembler Programs**
- Appendix

Code Coverage with Assembler - LE Assembler

- You can run Code Coverage with Both LE & Non-LE Assembler ("BAL")
- No changes are needed to the Load Module (Assemble) process. But you will need to modify the JCL slightly - example here of an LE-Assembler program and its Code Coverage



*XSAMALC.jcl

```
//DDS0001A JOB REGION=4M,CLASS=A,
//  TIME=(1),MSGCLASS=H,NOTIFY=&SYSUID,MSGLEVEL=(1,1)
//*******************************************************************
//*******************************************************************
//RDBGMAIN EXEC PGM=DBGMAIN,REGION=4M
//STEPLIB  DD DSN=DDS0001.TEST.LOAD,DISP=SHR
//CMDFILE DD DSN=&SYSUID..CMDFILE,D
//INSPLOG DD SYSOUT=*
//CEEOPTS   DD *
  TEST(,INSPIN,,DBMDT%DDS0001:)
  ENVAR("EQA_STARTUP_KEY=CC")
/*
//INSPIN DD *
LDD DBGMAIN;
/*
```

Notes:

```
---------------------------
TEST(,INSPIN,,DBMDT%TSOID:)

//INSPIN DD *
LDD  DBGMAIN;
/*
```

DBGMAIN_2019_06_26_000805_0415

Code Coverage Report

Code coverage report for 'DBGMAIN_2019_06_26_000805_0415', analyzed Jun 26, 2019 12:08:06 AM

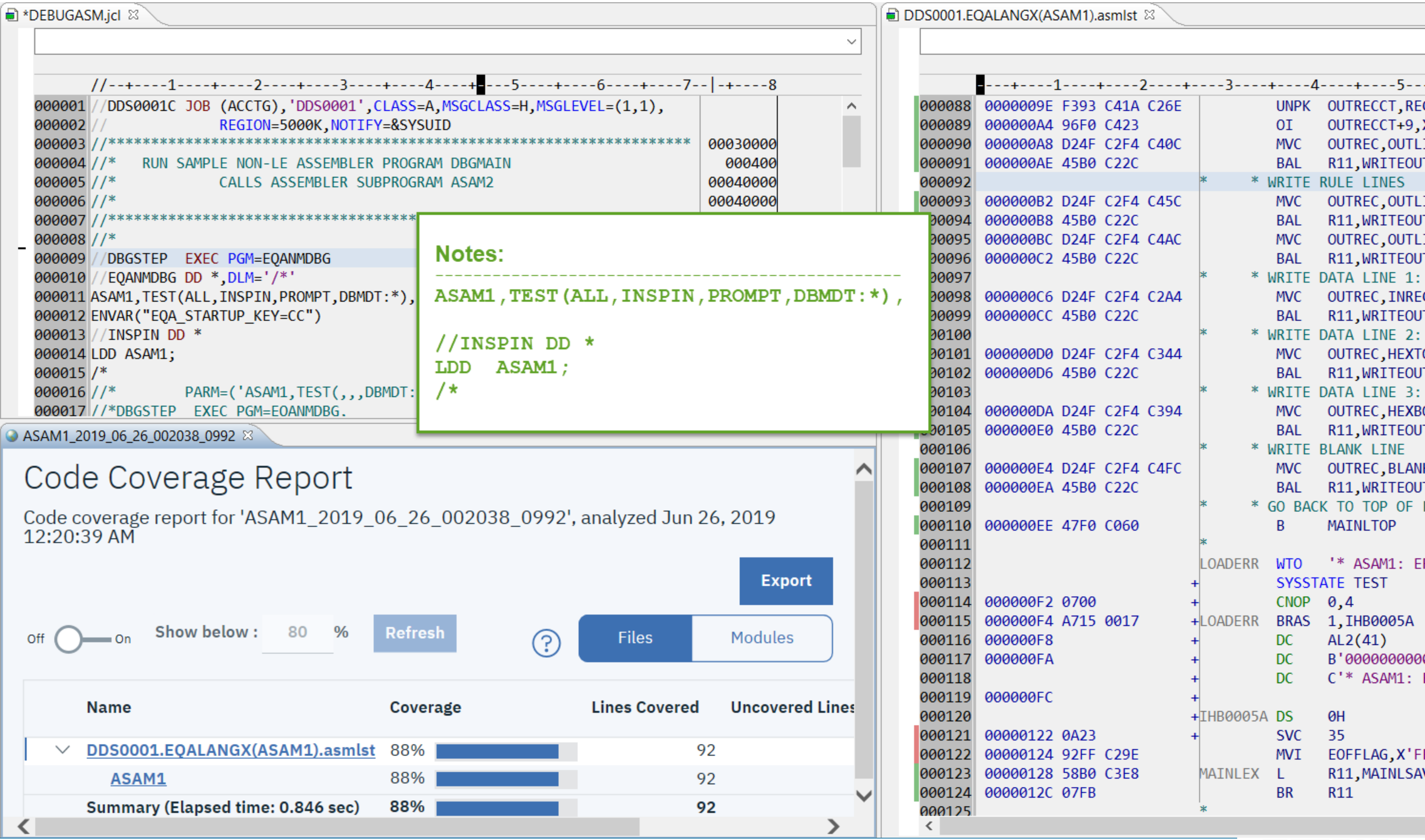
Name	Coverage	Lines Covered	Uncovered Line
DDS0001.EQALANGX(DBGMAIN).asmlst	62%	39	
DBGMAIN	62%	39	
Summary (Elapsed time: 1.175 sec)	**62%**	**39**	

DDS0001.EQALANGX(DBGMAIN).asmlst

```
00000048 181F           +           LR    1,15
0000004A A7F4 0007      +           J     CEEX0001
00000050                +CEEINPL0001     DC    A(CEEINPL)
00000054                +CEEINT0001      DC    V(CEEINT)
                        +CEEX0001    EQU *
00000058 50D0 1004      +           ST    13,CEEDSABKC-CEED
0000005C 5000 104C      +           ST    0,CEEDSANAB-CEEDS
00000060 D701 1000 1000 +           XC    CEEDSAFLAGS-CEEDS
00000066 5010 D008      +           ST    1,CEEDSAFWC-CEEDS
0000006A 18D1           +           LR    13,1
                        +           POP   USING
                        +           USING CEEDSA,13
0000006C D203 D048 C280 +           MVC   CEEDSALWS,CEECAAL
00000072 1812           +           LR    1,2
00000074 C0B0 FFFF FFC6 +           LARL  11,DBGMAIN
                        +           USING DBGMAIN,11
                                    USING    WORKAREA,R13
                        *                                   EA
                        *                                   EA
                        *                                   EA
0000007A 4120 B1C4                  LA   R2,STRT_MSG
0000007E 4130 B1F8                  LA   R3,DEST
00000082 4140 D094                  LA   R4,FBCODE
00000086 9024 D080                  STM  R2,R4,PLIST
0000008A 4110 D080                  LA   R1,PLIST
0000008E 58F0 B1A4                  L    R15,MOUT
00000092 05EF                       BALR R14,R15
                        *
00000094 F224 B18C B17D             PACK PCKA,ZNA
0000009A F224 B18F B182             PACK PCKB,ZNB
000000A0 F224 B192 B187             PACK PCKC,ZNC
000000A6 F832 B195 B18C             ZAP  PCKSUM,PCKA
000000AC FA32 B195 B18F             AP   PCKSUM,PCKB
000000B2 FA32 B195 B192             AP   PCKSUM,PCKC
000000B8 D207 B199 B175             MVC  OUTSUM,SUMMSK
000000BE DE07 B199 B195             ED   OUTSUM,PCKSUM
000000C4 D207 B15D B199             MVC  SUMMSG+1(8),OUTSUM
000000CA D218 B1AA B15C             MVC  LINE_ST,SUMMSG
```

Code Coverage with Assembler - Non-LE Assembler

- Note changes to the TEST statement - and the addition of the LDD statements, which identify the Debug Data. Note; Thanks to Francisco Anaya/IBM for the syntax examples



*DEBUGASM.jcl

```
//--+----1----+----2----+----3----+----4----+----5----+----6----+----7--|-+----8
//DDS0001C JOB (ACCTG),'DDS0001',CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1),
//             REGION=5000K,NOTIFY=&SYSUID
//*******************************************************************  00030000
//*   RUN SAMPLE NON-LE ASSEMBLER PROGRAM DBGMAIN                        000400
//*             CALLS ASSEMBLER SUBPROGRAM ASAM2                       00040000
//*                                                                    00040000
//*********************************
//*
//DBGSTEP  EXEC PGM=EQANMDBG
//EQANMDBG DD *,DLM='/*'
ASAM1,TEST(ALL,INSPIN,PROMPT,DBMDT:*),
ENVAR("EQA_STARTUP_KEY=CC")
//INSPIN DD *
LDD ASAM1;
/*
//*        PARM=('ASAM1,TEST(,,,DBMDT:
//*DBGSTEP  EXEC PGM=EQANMDBG.
```

Notes:

```
---------------------------------------------
ASAM1,TEST(ALL,INSPIN,PROMPT,DBMDT:*),

//INSPIN DD *
LDD  ASAM1;
/*
```

DDS0001.EQALANGX(ASAM1).asmlst

```
---+----1----+----2----+----3----+----4----+----5--
0000009E F393 C41A C26E             UNPK  OUTRECCT,REC
000000A4 96F0 C423                  OI    OUTRECCT+9,X
000000A8 D24F C2F4 C40C             MVC   OUTREC,OUTLI
000000AE 45B0 C22C                  BAL   R11,WRITEOUT
                              *    * WRITE RULE LINES
000000B2 D24F C2F4 C45C             MVC   OUTREC,OUTLI
000000B8 45B0 C22C                  BAL   R11,WRITEOUT
000000BC D24F C2F4 C4AC             MVC   OUTREC,OUTLI
000000C2 45B0 C22C                  BAL   R11,WRITEOUT
                              *    * WRITE DATA LINE 1:
000000C6 D24F C2F4 C2A4             MVC   OUTREC,INREC
000000CC 45B0 C22C                  BAL   R11,WRITEOUT
                              *    * WRITE DATA LINE 2:
000000D0 D24F C2F4 C344             MVC   OUTREC,HEXTO
000000D6 45B0 C22C                  BAL   R11,WRITEOUT
                              *    * WRITE DATA LINE 3:
000000DA D24F C2F4 C394             MVC   OUTREC,HEXBO
000000E0 45B0 C22C                  BAL   R11,WRITEOUT
                              *    * WRITE BLANK LINE
000000E4 D24F C2F4 C4FC             MVC   OUTREC,BLANK
000000EA 45B0 C22C                  BAL   R11,WRITEOUT
                              *    * GO BACK TO TOP OF M
000000EE 47F0 C060                  B     MAINLTOP
                              *
                              LOADERR  WTO   '* ASAM1: EF
                             +          SYSSTATE TEST
000000F2 0700                +          CNOP  0,4
000000F4 A715 0017           +LOADERR   BRAS  1,IHB0005A
000000F8                     +          DC    AL2(41)
000000FA                     +          DC    B'0000000000
                             +          DC    C'* ASAM1: E
000000FC                     +
                             +IHB0005A DS    0H
00000122 0A23                +          SVC   35
00000124 92FF C29E                      MVI   EOFFLAG,X'FF
00000128 58B0 C3E8             MAINLEX  L     R11,MAINLSAV
0000012C 07FB                           BR    R11
                              *
```

ASAM1_2019_06_26_002038_0992

Code Coverage Report

Code coverage report for 'ASAM1_2019_06_26_002038_0992', analyzed Jun 26, 2019 12:20:39 AM

Show below : 80 %

Name	Coverage	Lines Covered	Uncovered Lines
DDS0001.EQALANGX(ASAM1).asmlst	88%	92	
ASAM1	88%	92	
Summary (Elapsed time: 0.846 sec)	**88%**	**92**	

UNIT

The IDz Workbench

Topics:

- Debugging z/OS Assembler Batch Applications
- **Debugging z/OS Assembler Online Applications**
- Appendix

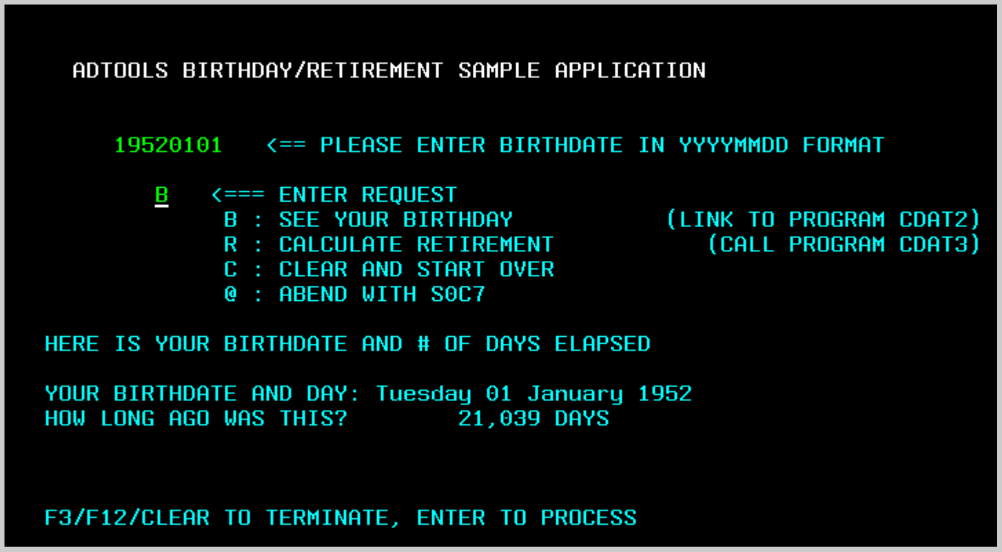
Topic Objectives

After completing this unit, you should be able to:

- Using the Problem Determination Tools, Debug Option and IDz:
  - Debug a mainframe online transaction
- Describe the online transaction features for configuring your 3270 sessions with Debug Option
- Debug a CICS 3270 Application

Online Debugging Overview

- Guess what?
  No one gets it right the first time coding online programs either ☺



```
  ADTOOLS BIRTHDAY/RETIREMENT SAMPLE APPLICATION

     19520101   <== PLEASE ENTER BIRTHDATE IN YYYYMMDD FORMAT

        B   <=== ENTER REQUEST
             B : SEE YOUR BIRTHDAY           (LINK TO PROGRAM CDAT2)
             R : CALCULATE RETIREMENT           (CALL PROGRAM CDAT3)
             C : CLEAR AND START OVER
             @ : ABEND WITH S0C7

HERE IS YOUR BIRTHDATE AND # OF DAYS ELAPSED

YOUR BIRTHDATE AND DAY: Tuesday 01 January 1952
HOW LONG AGO WAS THIS?        21,039 DAYS


F3/F12/CLEAR TO TERMINATE, ENTER TO PROCESS
```

- Lucky for you:
  - Debug tool handles:
    - CICS 3270 online transactions
    - IMS TM online transactions
  - Without any different debugging techniques
    - The only difference from batch is the debug setup procedure for the online environment

Steps for **Online** (CICS) Application Debug Session

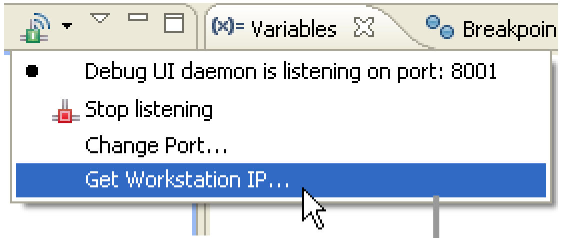
- Ensure that your compile proc has the necessary TEST parameter, and Compile/Link to create load module – and that your CICS application is setup for Debug Option testing
- Discover workstation TCP/IP parameters:
  - IP Address
  - Listener port#
- Access and login to your CICS region – Green Screen
- Use the DTCN view, or execute the DTCN transaction and specify:
  - Terminal ID
  - Transaction code and programs – to put under Debug control
  - User-ID
  - TCP/IP parameters:
    - IP Address
    - Port#
  - Save the DTCN transaction specification
- Debug your CICS application

Discover TCP/IP address and IDz Port - Review

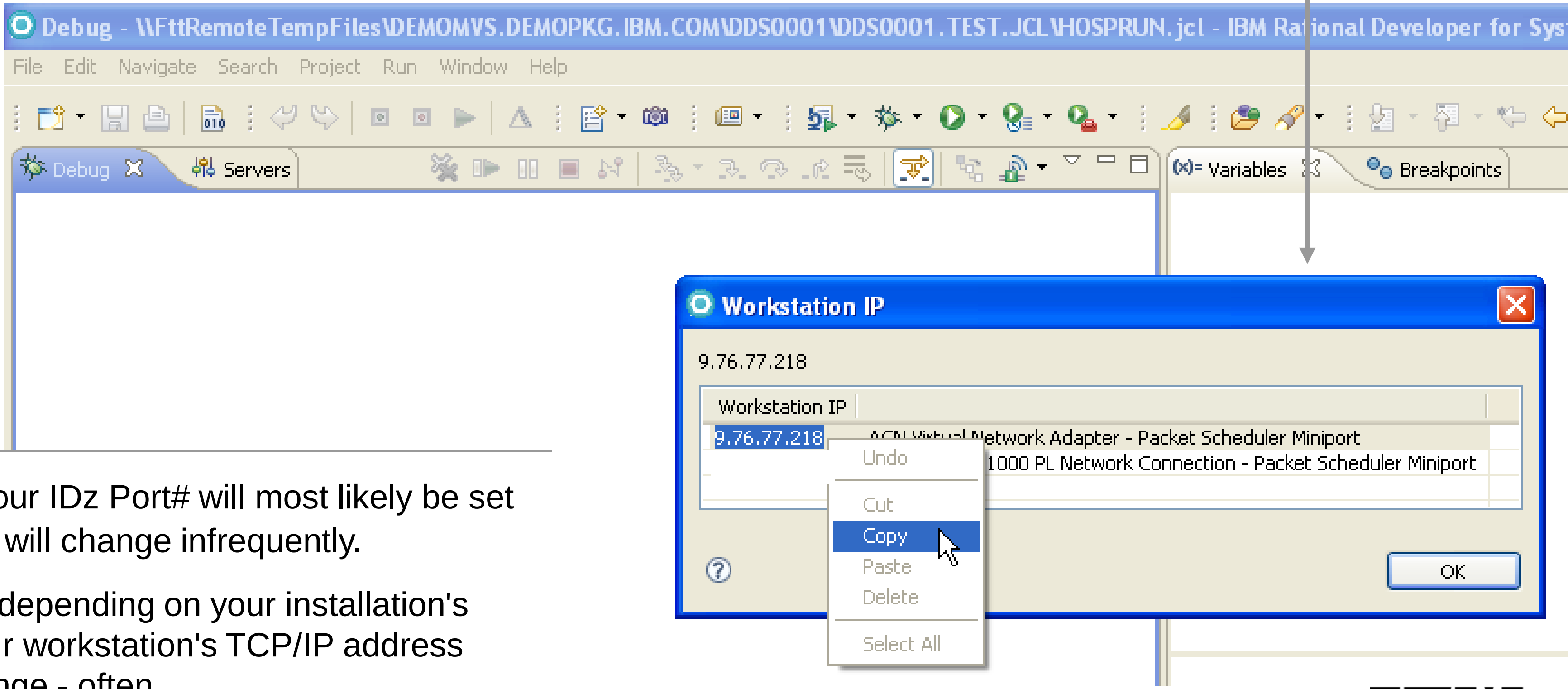
- Open the Debug Perspective

Click the small downward pointing triangle next to the debug-daemon icon

- **Note the Port#**
- Select: **Get Workstation IP…**
- **Copy the IP address**
- **Either paste the IP address into Notepad, or write it down**



Debug UI daemon is listening on port: 8001



Debug - \\FttRemoteTempFiles\DEMOMVS.DEMOPKG.IBM.COM\DDS0001\DDS0001.TEST.JCL\HOSPRUN.jcl - IBM Rational Developer for Sys

Workstation IP

9.76.77.218

Workstation IP	
9.76.77.218	ACN Virtual Network Adapter - Packet Scheduler Miniport
	1000 PL Network Connection - Packet Scheduler Miniport

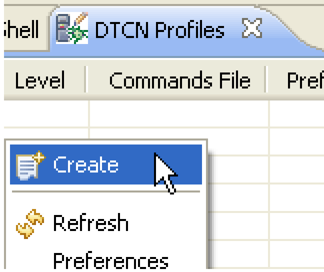
Note: Your IDz Port# will most likely be set once, and will change infrequently.

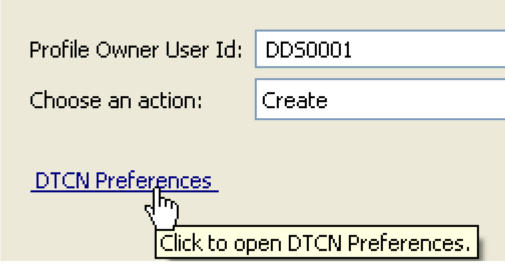
However, depending on your installation's setup, your workstation's TCP/IP address could change - often

Setup the DTCN Parameters Using the DTCN View

If you are using IDz v7.6.1 or higher, you can utilize an IDz view to setup your DTCN CICS Debug properties.

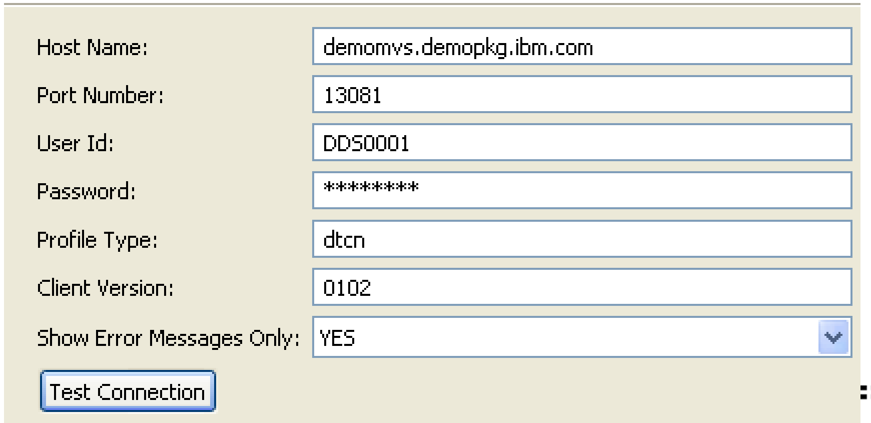
- Steps:
  - From Window > Show View > Other type: DTCN and select DTCN Profiles
  - Right-click inside the new, empty view and select: **Create**
  - From the **DTCN profiles** window:
    - Enter your User ID
    - Click DTCN Preferences
  - From **DTCN preferences** specify:
    - Host Name/IP Address
    - CICS DTCN **transaction port**
    - CICS login credentials:
      - User ID
      - Password
    - Other fields as shown ➔
    - Click Test Connection
    - Click OK to check your work





Profile Owner User Id: DDS0001

Choose an action: Create



Host Name: demomvs.demopkg.ibm.com

Port Number: 13081

User Id: DDS0001

Password: *********

Profile Type: dtcn

Client Version: 0102

Show Error Messages Only: YES

Setup the DTCN Parameters Using the DTCN View – continued

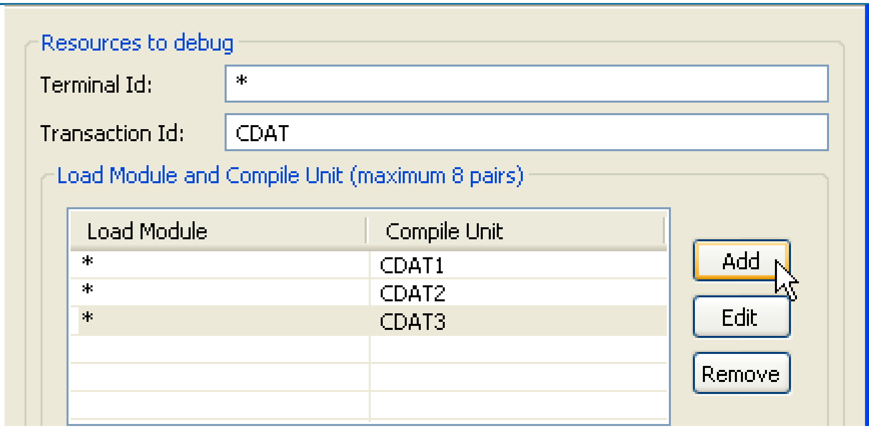
From **DTCN profiles** click **Next >**

From **DTCN pattern matching** specify :

- Terminal ID: *
- Transaction ID (Trancode) ➔
- Click Add, and specify the Compile Units

  (Load Module names)

Click **Next >**



Resources to debug

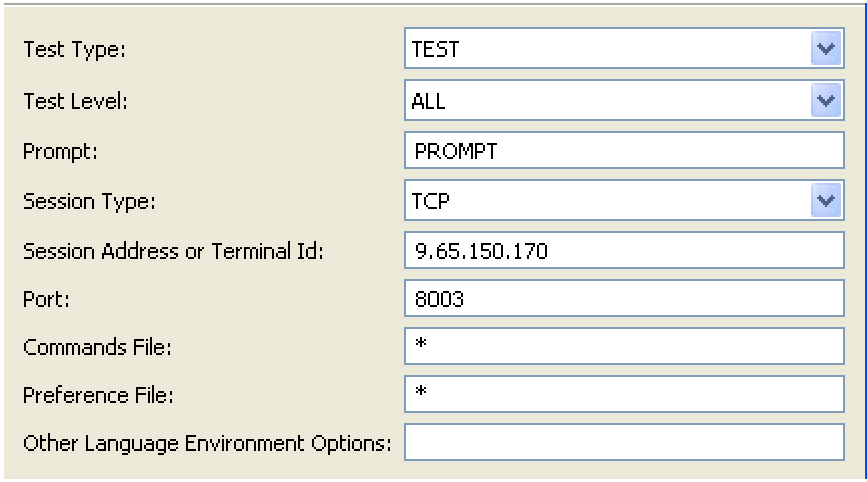
Terminal Id:	*
Transaction Id:	CDAT

Load Module and Compile Unit (maximum 8 pairs)

Load Module	Compile Unit
*	CDAT1
*	CDAT2
*	CDAT3

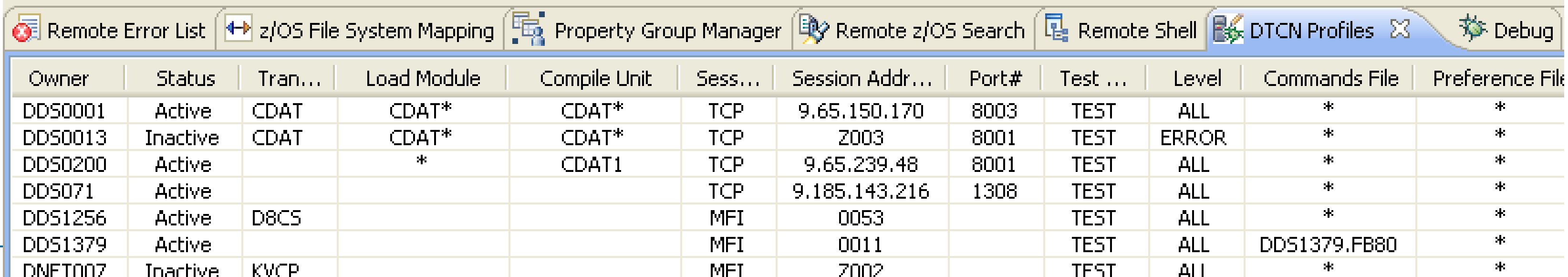
From **DTCN TEST run-time** specify:

- Fields as shown ➔
- Session Address (your workstation I/P address)
- Port (your listener Debug Tool listener port)
- Other fields – as shown ➔
- Click Finish



Test Type:	TEST
Test Level:	ALL
Prompt:	PROMPT
Session Type:	TCP
Session Address or Terminal Id:	9.65.150.170
Port:	8003
Commands File:	*
Preference File:	*
Other Language Environment Options:	

DTCN Profiles will be populated with entries for all users connecting into that CICS region

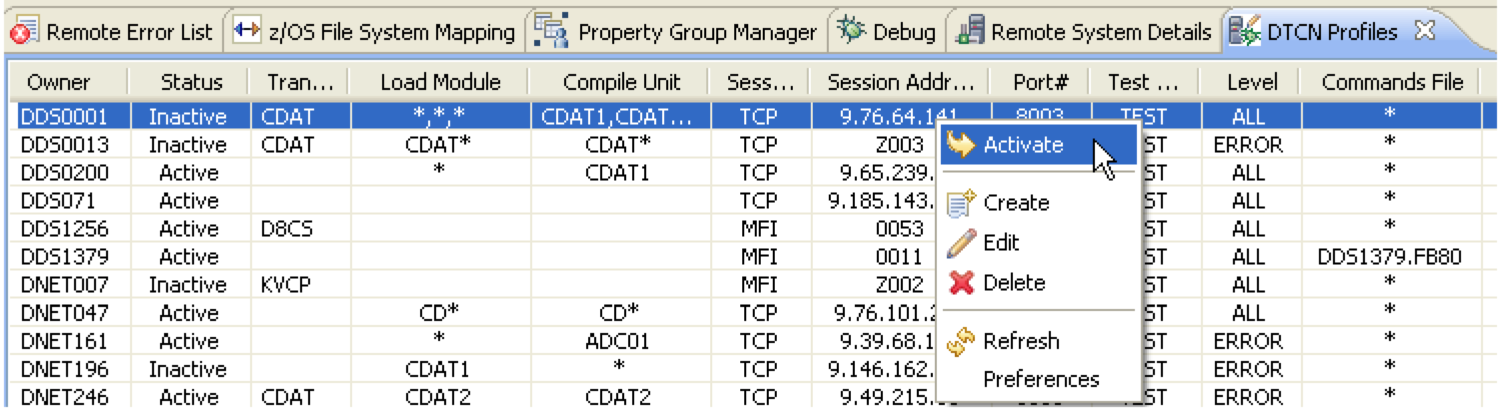


Owner	Status	Tran...	Load Module	Compile Unit	Sess...	Session Addr...	Port#	Test ...	Level	Commands File	Preference File
DDS0001	Active	CDAT	CDAT*	CDAT*	TCP	9.65.150.170	8003	TEST	ALL	*	*
DDS0013	Inactive	CDAT	CDAT*	CDAT*	TCP	Z003	8001	TEST	ERROR	*	*
DDS0200	Active		*	CDAT1	TCP	9.65.239.48	8001	TEST	ALL	*	*
DDS071	Active				TCP	9.185.143.216	1308	TEST	ALL	*	*
DDS1256	Active	D8CS			MFI	0053		TEST	ALL	*	*
DDS1379	Active				MFI	0011		TEST	ALL	DDS1379.FB80	*
DNET007	Inactive	KVCP			MFI	Z002		TEST	ALL	*	*

Using the DTCN View

Once you have setup the DTCN View, you can:

- **Activate the profile**
  - This modifies the CICS System Tables dynamically, through the Debug Tool facilities – and allows you to debug CICS transactions



Owner	Status	Tran...	Load Module	Compile Unit	Sess...	Session Addr...	Port#	Test ...	Level	Commands File
DDS0001	Inactive	CDAT	*,*,*	CDAT1,CDAT...	TCP	9.76.64.141	8003	TEST	ALL	*
DDS0013	Inactive	CDAT	CDAT*	CDAT*	TCP	Z003		ST	ERROR	*
DDS0200	Active		*	CDAT1	TCP	9.65.239.		ST	ALL	*
DDS071	Active				TCP	9.185.143.		ST	ALL	*
DDS1256	Active	D8CS			MFI	0053		ST	ALL	*
DDS1379	Active				MFI	0011		ST	ALL	DDS1379.FB80
DNET007	Inactive	KVCP			MFI	Z002		ST	ALL	*
DNET047	Active		CD*	CD*	TCP	9.76.101.		ST	ALL	*
DNET161	Active		*	ADC01	TCP	9.39.68.1		ST	ERROR	*
DNET196	Inactive		CDAT1	*	TCP	9.146.162.		ST	ERROR	*
DNET246	Active	CDAT	CDAT2	CDAT2	TCP	9.49.215.		ST	ERROR	*

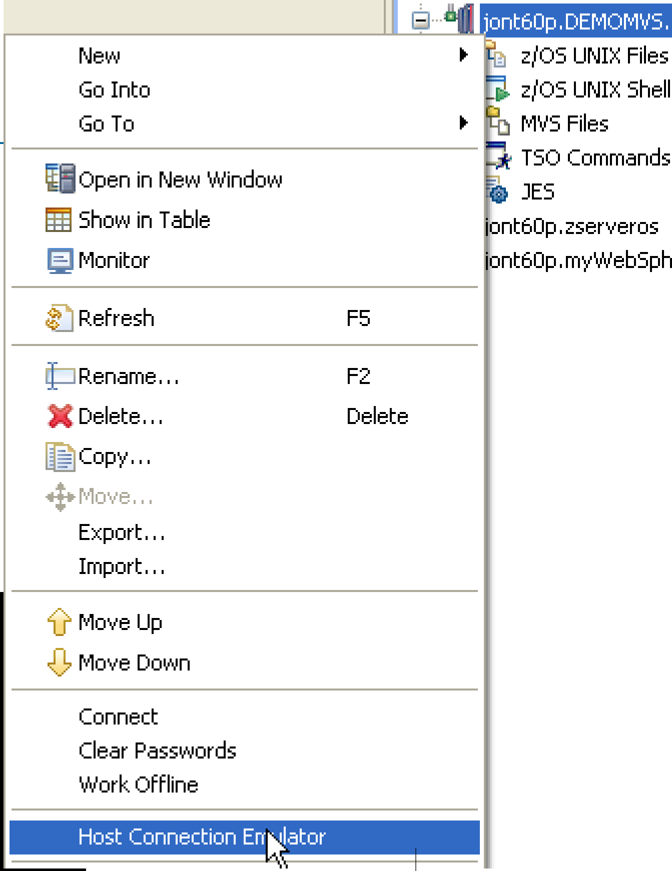
- Other options include:
  - **Edit the profile** – and change your I/P address
    - After you reboot your machine
  - Delete the profile
  - Create a new profile
  - Refresh the display of DTCN entries in the region

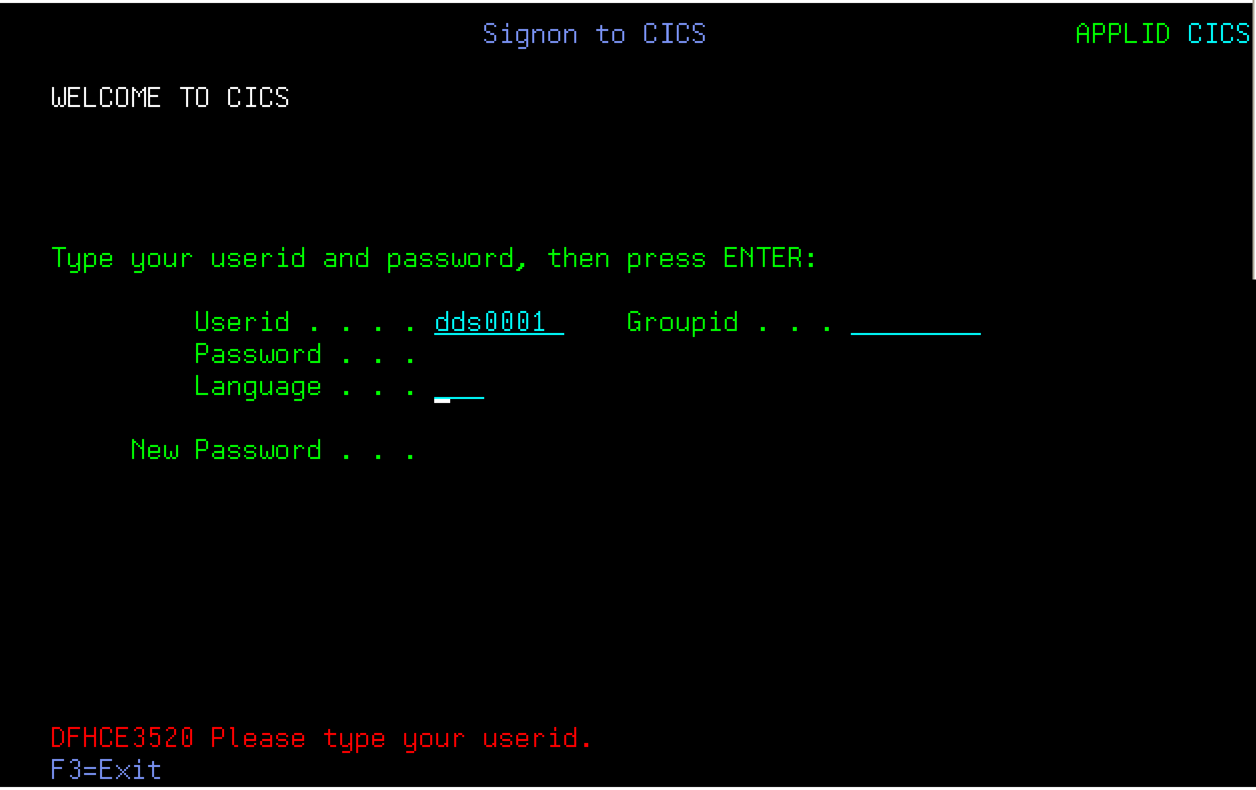
Note: In order to debug CICS programs you will have to launch a 3270 emulation session (next slides) to kick off the transaction

3. Login to your CICS Region

From Remote Systems Explorer:

- Right-click
- Select: Host Connection Emulator
- Select your CICS application
- Enter your Userid and Password and sign in

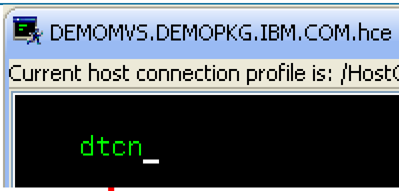




```
                    Signon to CICS                    APPLID CICS

 WELCOME TO CICS




 Type your userid and password, then press ENTER:

          Userid . . . . dds0001      Groupid . . . ________
          Password . . .
          Language . . . ___

      New Password . . .








 DFHCE3520 Please type your userid.
 F3=Exit
```

3. Setup the Debug Option Parameters using DTCN Transaction – 1 of 2

If you did NOT use the DTCN view to enter your DTCN properties you can do so using a CICS Transaction (green screen)



DEMOMVS.DEMOPKG.IBM.COM.hce

Current host connection profile is: /Host(

```
dtcn_
```

From CICS (after signing in):

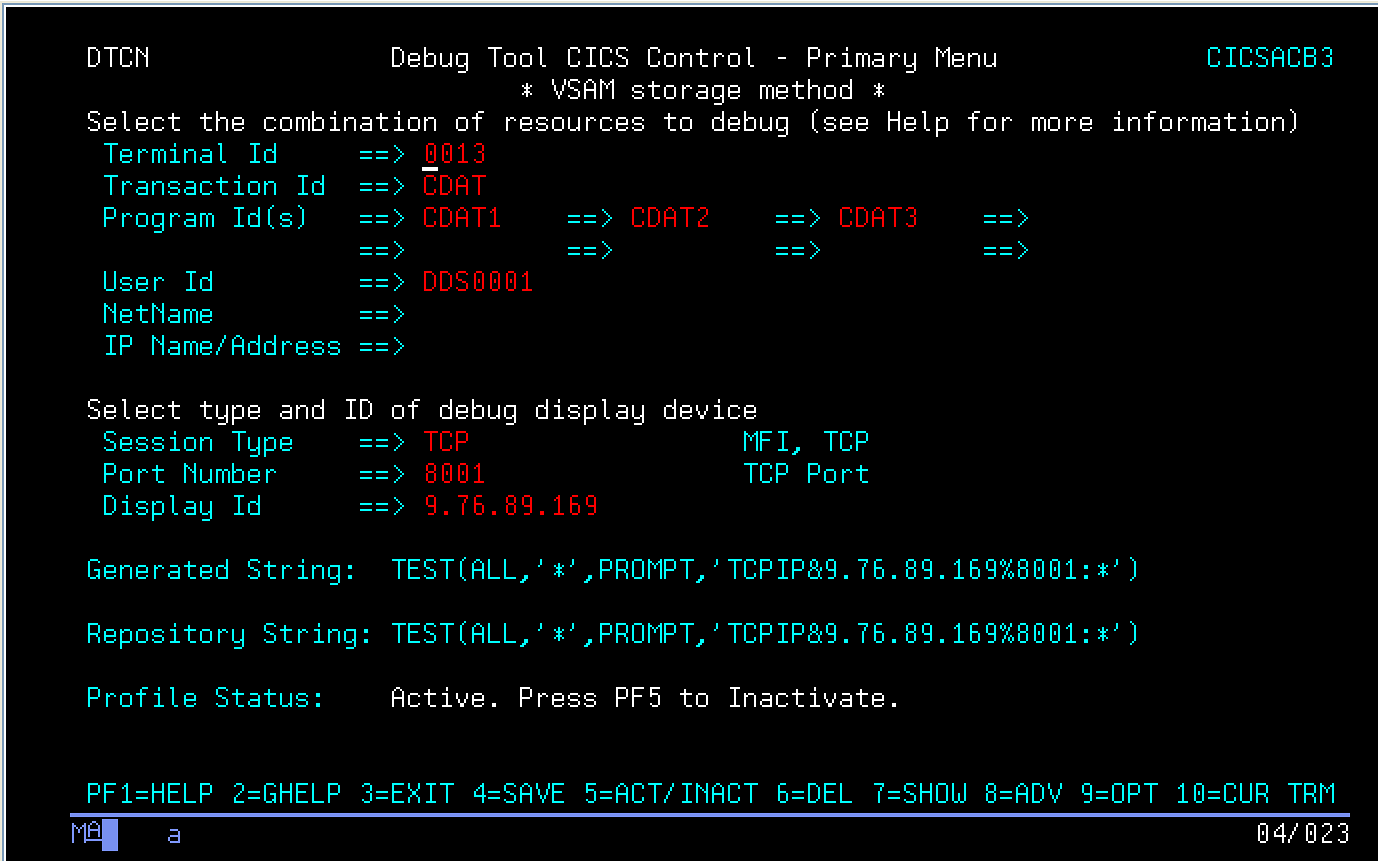
- Clear the screen, Enter: **DTCN** – and press ↵**Enter**

From the DTCN screen

- **Press F10** – this will fill in the Terminal Id for your workstation

Note that you can also type an asterisk: *****

…as the Terminal Id



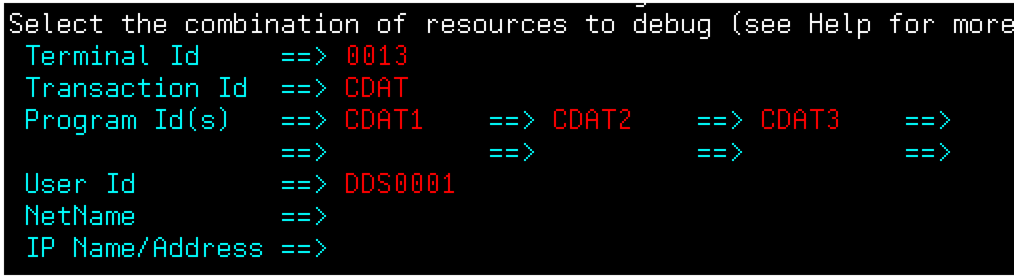
```
DTCN               Debug Tool CICS Control - Primary Menu             CICSACB3
                           * VSAM storage method *
Select the combination of resources to debug (see Help for more information)
 Terminal Id     ==> 0013
 Transaction Id  ==> CDAT
 Program Id(s)   ==> CDAT1    ==> CDAT2    ==> CDAT3    ==>
                 ==>          ==>          ==>          ==>
 User Id         ==> DDS0001
 NetName         ==>
 IP Name/Address ==>

Select type and ID of debug display device
 Session Type    ==> TCP                 MFI, TCP
 Port Number     ==> 8001                TCP Port
 Display Id      ==> 9.76.89.169

Generated String:  TEST(ALL,'*',PROMPT,'TCPIP&9.76.89.169%8001:*')

Repository String: TEST(ALL,'*',PROMPT,'TCPIP&9.76.89.169%8001:*')

Profile Status:    Active. Press PF5 to Inactivate.


PF1=HELP 2=GHELP 3=EXIT 4=SAVE 5=ACT/INACT 6=DEL 7=SHOW 8=ADV 9=OPT 10=CUR TRM
MA    a                                                                 04/023
```

Note: You would only use the DTCN transaction to specify your Debug Option properties if you could not use the DTCN view (prior slides)

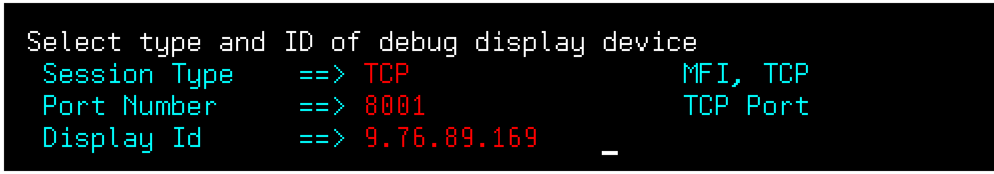
3. Setup the Debug Option Parameters using DTCN Transaction – 2 of 2

DTCN transaction data entry screen

- **Enter the Tran-code**
  - **Transaction ID**
- **Enter up to eight specific Program Id(s) you wish to debug through …or…**
- **Enter wildcard text for the Program Id(s)**
  - Ex. CD*
- Enter your User-ID
- Session Type: **TCP**
- **Port Number:** from your Debugger look-up
- **Display ID:** Your TCP/IP address, from your Debugger look-up (note that you can not paste into this 3270, screen)



```
Select the combination of resources to debug (see Help for more
 Terminal Id     ==> 0013
 Transaction Id  ==> CDAT
 Program Id(s)   ==> CDAT1     ==> CDAT2     ==> CDAT3     ==>
                 ==>           ==>           ==>           ==>
 User Id         ==> DDS0001
 NetName         ==>
 IP Name/Address ==>
```



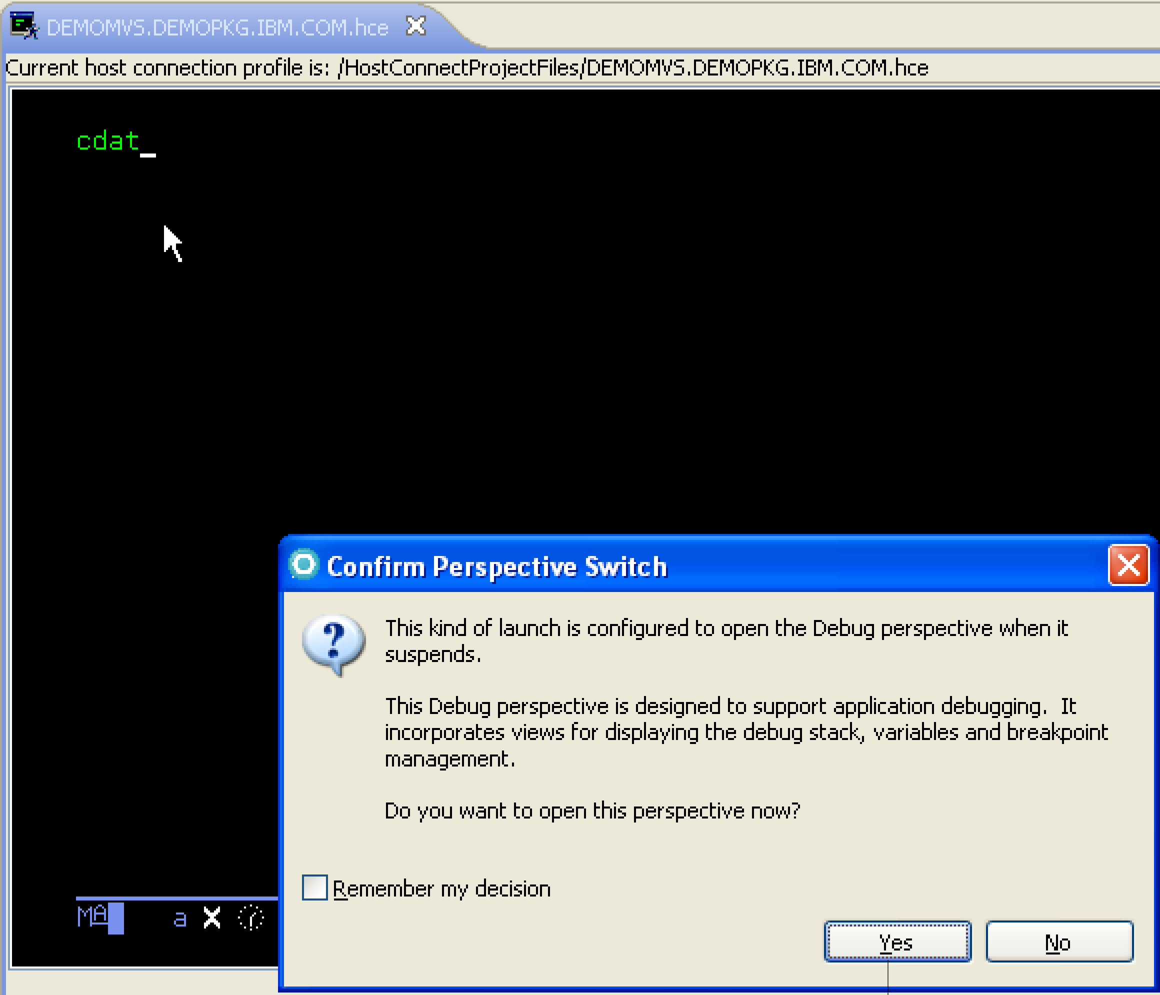
```
Select type and ID of debug display device
 Session Type    ==> TCP                  MFI, TCP
 Port Number     ==> 8001                 TCP Port
 Display Id      ==> 9.76.89.169   _
```

- **Press F4** to save your debug profile
- **Press F3** to clear the screen

4. Start Debugging

From the CICS region

- **Enter the Tran-code**
- **Press ↵Enter**
- **Click: Yes at the Confirm Perspective Switch**



DEMOMVS.DEMOPKG.IBM.COM.hce

Current host connection profile is: /HostConnectProjectFiles/DEMOMVS.DEMOPKG.IBM.COM.hce

```
cdat_
```

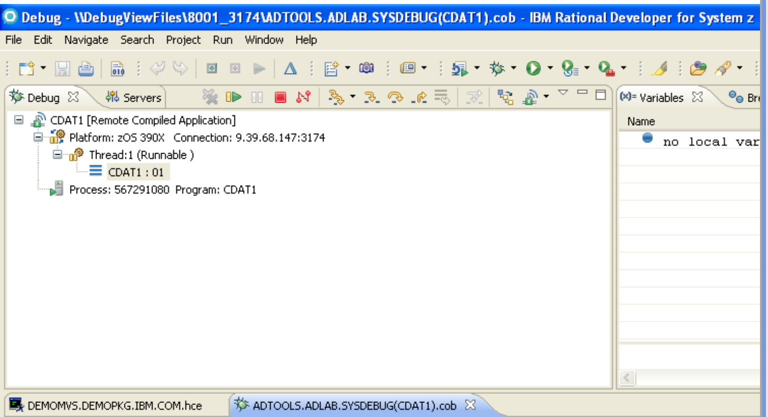
Confirm Perspective Switch

This kind of launch is configured to open the Debug perspective when it suspends.

This Debug perspective is designed to support application debugging. It incorporates views for displaying the debug stack, variables and breakpoint management.

Do you want to open this perspective now?

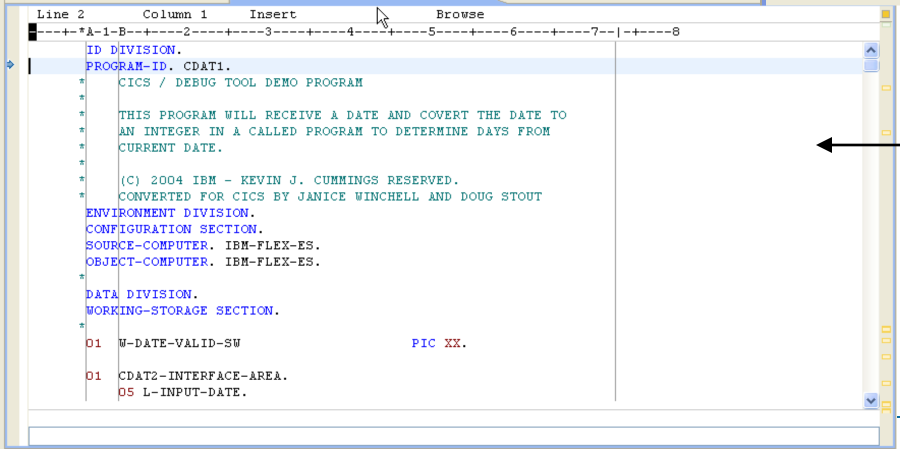
☐ Remember my decision



Debug - \\DebugViewFiles\8001_3174\ADTOOLS.ADLAB.SYSDEBUG(CDAT1).cob - IBM Rational Developer for System z

CDAT1 [Remote Compiled Application]
Platform: zOS 390X Connection: 9.39.68.147:3174
Thread:1 (Runnable)
CDAT1 : 01
Process: 567291080 Program: CDAT1

Name
no local var

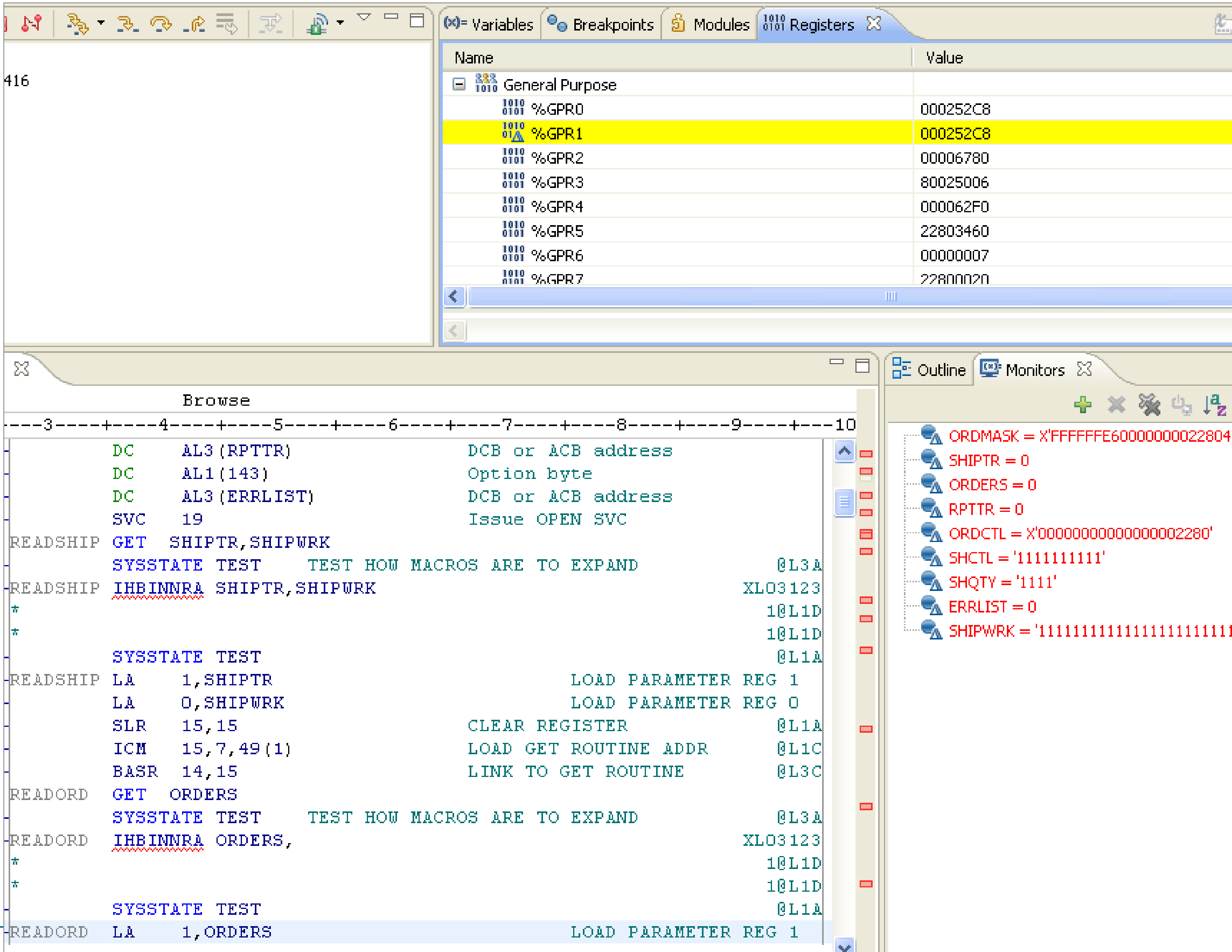


DEMOMVS.DEMOPKG.IBM.COM.hce
ADTOOLS.ADLAB.SYSDEBUG(CDAT1).cob

```
 Line 2      Column 1     Insert          Browse
----+-*A-1-B--+----2----+----3----+----4----+----5----+----6----+----7--|-+----8
       ID DIVISION.
       PROGRAM-ID. CDAT1.
      *    CICS / DEBUG TOOL DEMO PROGRAM
      *
      *    THIS PROGRAM WILL RECEIVE A DATE AND COVERT THE DATE TO
      *    AN INTEGER IN A CALLED PROGRAM TO DETERMINE DAYS FROM
      *    CURRENT DATE.
      *
      *    (C) 2004 IBM - KEVIN J. CUMMINGS RESERVED.
      *    CONVERTED FOR CICS BY JANICE WINCHELL AND DOUG STOUT
       ENVIRONMENT DIVISION.
       CONFIGURATION SECTION.
       SOURCE-COMPUTER. IBM-FLEX-ES.
       OBJECT-COMPUTER. IBM-FLEX-ES.
      *
       DATA DIVISION.
       WORKING-STORAGE SECTION.
      *
       01  W-DATE-VALID-SW                    PIC XX.

       01  CDAT2-INTERFACE-AREA.
           05 L-INPUT-DATE.
```

4. Start Debugging

Debug as previously learned in the batch/remote and Local debug units.



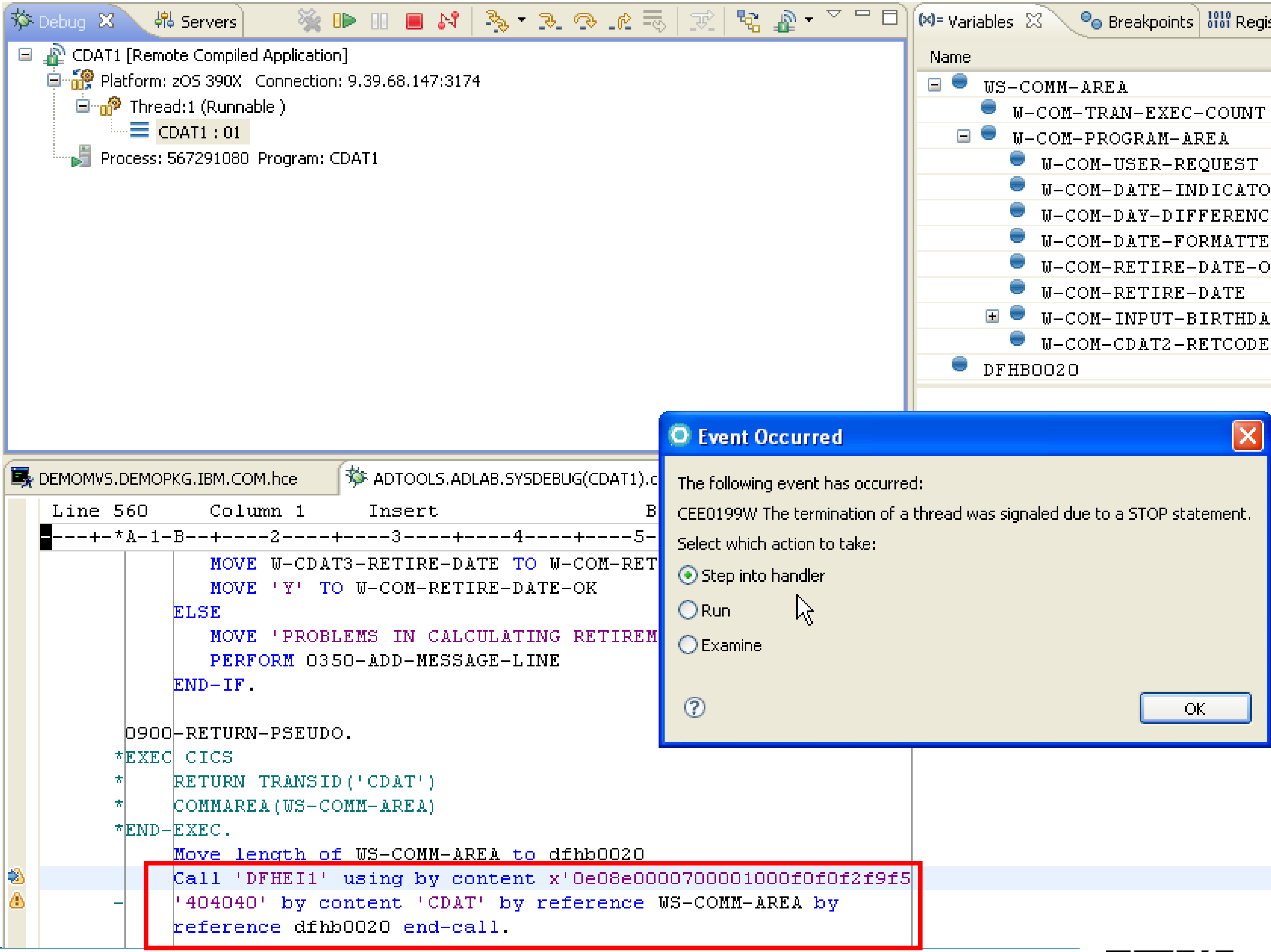
Name	Value
General Purpose	
%GPR0	000252C8
%GPR1	000252C8
%GPR2	00006780
%GPR3	80025006
%GPR4	000062F0
%GPR5	22803460
%GPR6	00000007
%GPR7	22800020

```
----3----+----4----+----5----+----6----+----7----+----8----+----9----+----10
         DC    AL3(RPTTR)              DCB or ACB address
         DC    AL1(143)                Option byte
         DC    AL3(ERRLIST)            DCB or ACB address
         SVC   19                      Issue OPEN SVC
READSHIP GET  SHIPTR,SHIPWRK
         SYSSTATE TEST    TEST HOW MACROS ARE TO EXPAND             @L3A
READSHIP IHBINNRA SHIPTR,SHIPWRK                                 XL03123
*                                                                 1@L1D
*                                                                 1@L1D
         SYSSTATE TEST                                             @L1A
READSHIP LA    1,SHIPTR                         LOAD PARAMETER REG 1
         LA    0,SHIPWRK                        LOAD PARAMETER REG 0
         SLR   15,15                   CLEAR REGISTER              @L1A
         ICM   15,7,49(1)              LOAD GET ROUTINE ADDR       @L1C
         BASR  14,15                   LINK TO GET ROUTINE         @L3C
READORD  GET  ORDERS
         SYSSTATE TEST    TEST HOW MACROS ARE TO EXPAND             @L3A
READORD  IHBINNRA ORDERS,                                        XL03123
*                                                                 1@L1D
*                                                                 1@L1D
         SYSSTATE TEST                                             @L1A
READORD  LA    1,ORDERS                         LOAD PARAMETER REG 1
```

```
ORDMASK = X'FFFFFFE60000000000022804
SHIPTR = 0
ORDERS = 0
RPTTR = 0
ORDCTL = X'000000000000000002280'
SHCTL = '1111111111'
SHQTY = '1111'
ERRLIST = 0
SHIPWRK = '11111111111111111111111
```

What Happens for Calls and Screen-IO? – 1 of 2

You will be prompted, and presented with debug-run-time options



CDAT1 [Remote Compiled Application]
Platform: zOS 390X   Connection: 9.39.68.147:3174
Thread:1 (Runnable)
CDAT1 : 01
Process: 567291080 Program: CDAT1

WS-COMM-AREA
W-COM-TRAN-EXEC-COUNT
W-COM-PROGRAM-AREA
W-COM-USER-REQUEST
W-COM-DATE-INDICATO
W-COM-DAY-DIFFERENC
W-COM-DATE-FORMATTE
W-COM-RETIRE-DATE-O
W-COM-RETIRE-DATE
W-COM-INPUT-BIRTHDA
W-COM-CDAT2-RETCODE
DFHB0020

Event Occurred

The following event has occurred:

CEE0199W The termination of a thread was signaled due to a STOP statement.

Select which action to take:

● Step into handler
○ Run
○ Examine

DEMOMVS.DEMOPKG.IBM.COM.hce

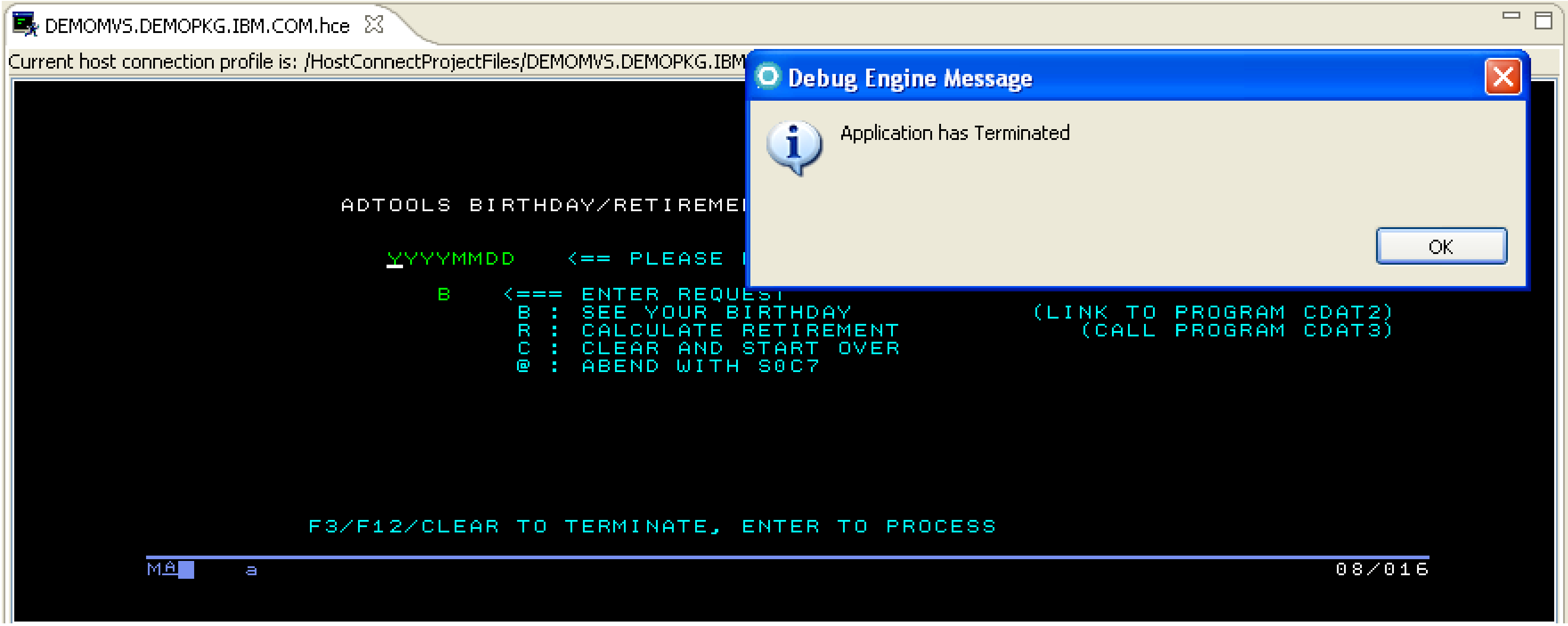
ADTOOLS.ADLAB.SYSDEBUG(CDAT1).c

```
 Line 560      Column 1      Insert      B
----+-*A-1-B--+----2----+----3----+----4----+----5-
           MOVE W-CDAT3-RETIRE-DATE TO W-COM-RET
           MOVE 'Y' TO W-COM-RETIRE-DATE-OK
       ELSE
           MOVE 'PROBLEMS IN CALCULATING RETIREM
           PERFORM 0350-ADD-MESSAGE-LINE
       END-IF.

 0900-RETURN-PSEUDO.
*EXEC CICS
*      RETURN TRANSID('CDAT')
*      COMMAREA(WS-COMM-AREA)
*END-EXEC.
       Move length of WS-COMM-AREA to dfhb0020
       Call 'DFHEI1' using by content x'0e08e0000700001000f0f0f2f9f5
-      '404040' by content 'CDAT' by reference WS-COMM-AREA by
       reference dfhb0020 end-call.
```

What Happens for Calls and Screen-IO? – 2 of 2

If your current transaction ends, and a BMS or 3270 screen is sent:

- You will be notified (prompted) by the debug engine
- If a screen is sent, the 3270 will display in the content area



Current host connection profile is: /HostConnectProjectFiles/DEMOMVS.DEMOPKG.IBM

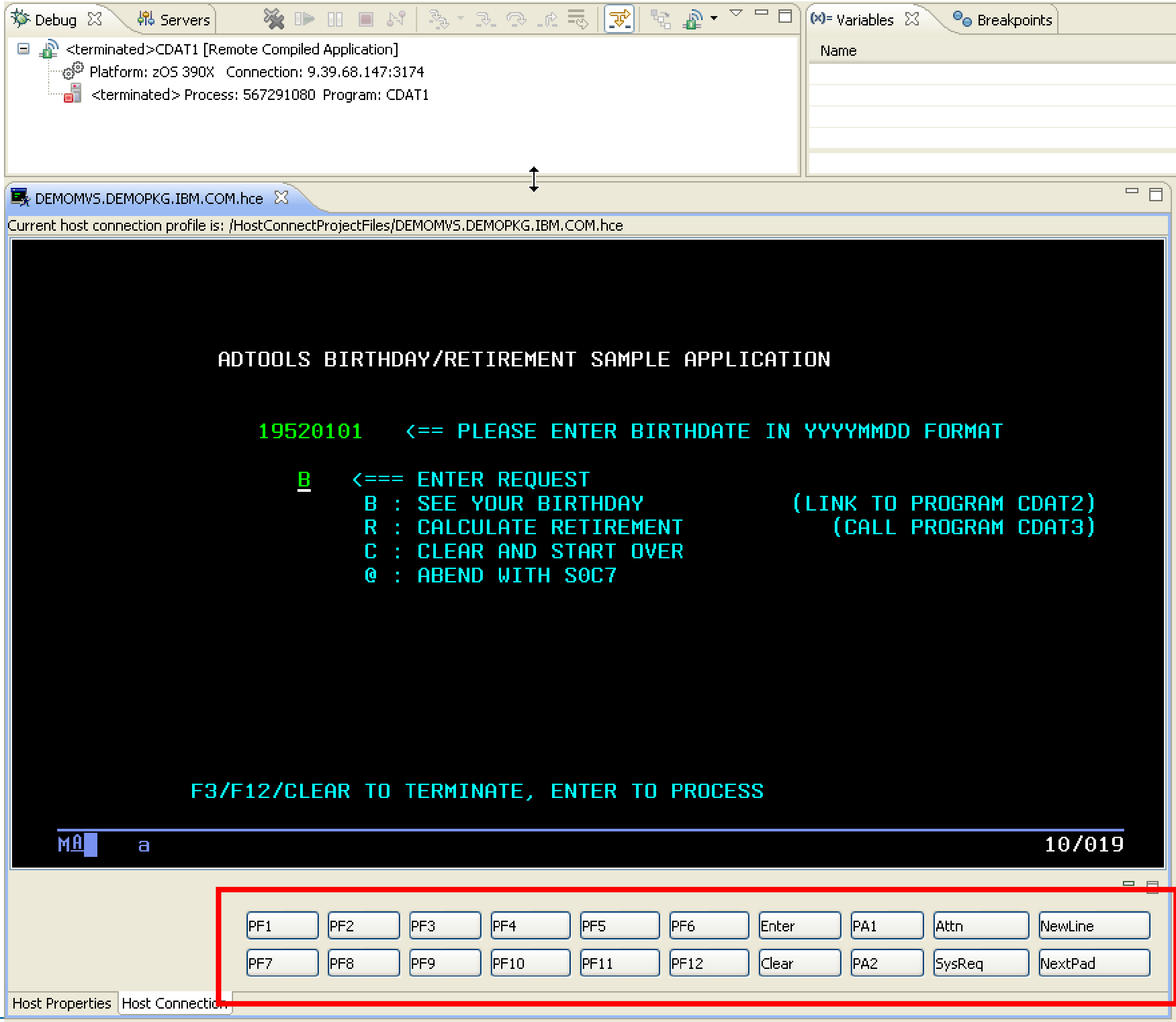
Debug Engine Message

Application has Terminated

```
ADTOOLS BIRTHDAY/RETIREME

YYYYMMDD    <== PLEASE 

B    <=== ENTER REQUEST
     B : SEE YOUR BIRTHDAY                  (LINK TO PROGRAM CDAT2)
     R : CALCULATE RETIREMENT                 (CALL PROGRAM CDAT3)
     C : CLEAR AND START OVER
     @ : ABEND WITH S0C7

F3/F12/CLEAR TO TERMINATE, ENTER TO PROCESS

MA      a                                                    08/016
```

What About PF-Keys and Other Data Entry?

- You can resize the screen portion of the debugger
- And use the PF-Key emulation options in the Host Connection



<terminated>CDAT1 [Remote Compiled Application]
Platform: zOS 390X Connection: 9.39.68.147:3174
<terminated> Process: 567291080 Program: CDAT1

DEMOMVS.DEMOPKG.IBM.COM.hce

Current host connection profile is: /HostConnectProjectFiles/DEMOMVS.DEMOPKG.IBM.COM.hce

```
          ADTOOLS BIRTHDAY/RETIREMENT SAMPLE APPLICATION

            19520101   <== PLEASE ENTER BIRTHDATE IN YYYYMMDD FORMAT

                B   <=== ENTER REQUEST
                     B : SEE YOUR BIRTHDAY          (LINK TO PROGRAM CDAT2)
                     R : CALCULATE RETIREMENT         (CALL PROGRAM CDAT3)
                     C : CLEAR AND START OVER
                     @ : ABEND WITH S0C7



         F3/F12/CLEAR TO TERMINATE, ENTER TO PROCESS

MA      a                                                          10/019
```

Topic Objectives

After having completed this unit, you now should be able to:

- Using the Problem Determination Tools, Debug Option and IDz:
  - Debug a mainframe online transaction
- Describe the online transaction features for configuring your 3270 sessions with Debug Option
- Debug a CICS 3270 Application