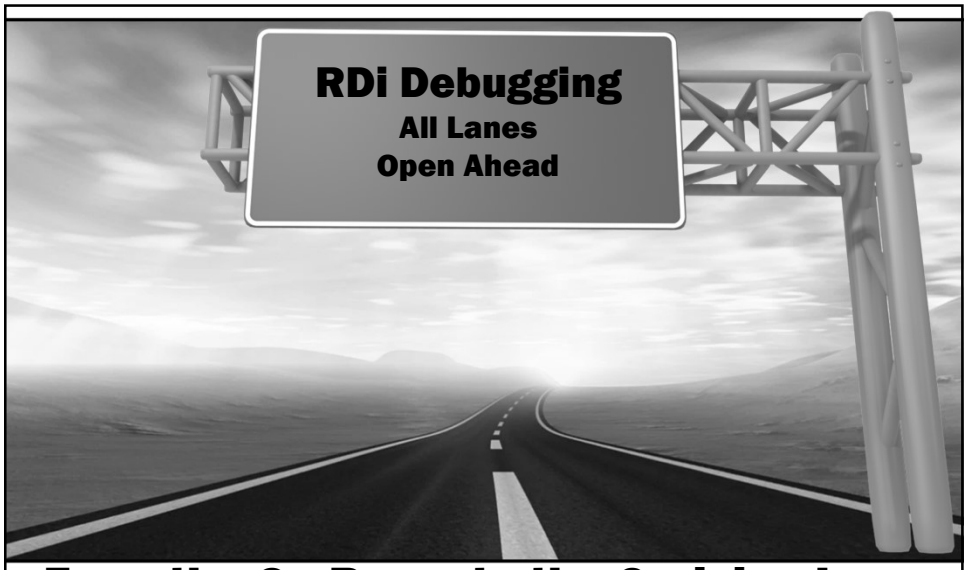


From the On Ramp to the Cruising Lane

Charles Guarino @charlieguarino

Central Park Data Systems

0

About The Speaker

Charles Guarino believes in the “power” of IBM Power Systems. His career reflects his dedication and interest in bringing the platform and its solutions to others and as a result has been recognized as an IBM Champion.

He is a member of COMMON’s Speaker Excellence Hall of Fame and a proud recipient of the Al Barsa Memorial Scholarship Award. Additionally, he participates on COMMON’s Strategic Education Team. Other professional endeavors have included the roles of President and monthly Q&A host of the Long Island System User’s Group LISUG (www.lisug.org)

Today, along with the team at Central Park Data Systems, he is serving individuals and companies on a worldwide basis though his consulting work and award-winning speaking engagements.

Charles is a true people person and can often be found at conferences sharing his expertise on RDi and other IBM i topics.

Charles can be reached at cguarino@centralparkdata.com.
LinkedIn - http://www.linkedin.com/in/guarinocharles
Twitter - @charlieguarino

1

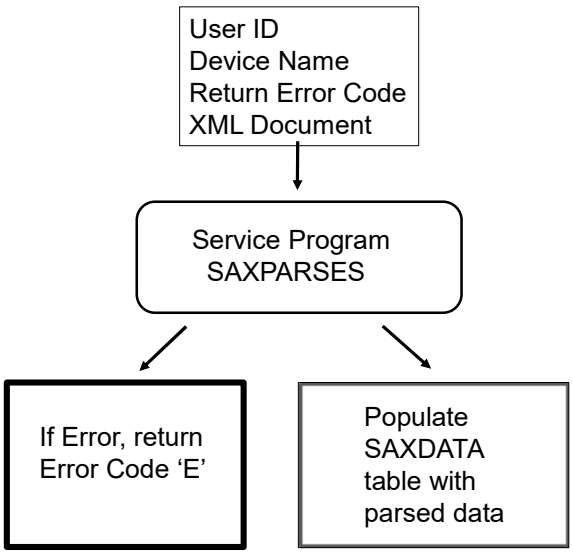
Copyright Central Park Data Systems Inc

What We'll Cover …

- **Let's Get Started**
- **Preparing to Submit**
- **The Debugger Comes to Life**
- **Setting a Service Entry Point**
- **RDi Debugging Hacks**
- **Wrap-up**

2

Program Flow



3

Copyright Central Park Data Systems Inc

How an XML SAX Parser Works

<Attribute Name="Dealer">10912</Attribute>

Start Element	“Attribute”
Attribute Name	“Name”
Attribute Characters	“Dealer”
XML Characters	“10912”
End Element	“Attribute”

4

XML Data Passed to Service Program

```
----+----1----+----2----+----3----+----4----+----5----+----6---
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE XMLFileIn>
<!--  XML ORDER EXPORT FILE V1.0 -->
<XMLFileIn>
<OrderV2>
<Attribute Name="Dealer">10912</Attribute>
<Attribute Name="Homeowner">000002</Attribute>
<Attribute Name="ProjectName">New Construction 529</Attribute>
<Attribute Name="CreatedBy">11714</Attribute>
<Attribute Name="CustPONumber">Verbal</Attribute>
<Attribute Name="JobNumber"/>
<Attribute Name="PriceMultiplier">.83</Attribute>
<Attribute Name="SalesTaxPercent">8.65</Attribute>
<Attribute Name="SalesTaxAmount">0.00</Attribute>
<CustomerInfo>
<Attribute Name="Name"/>
<Attribute Name="CustomerID"/>
<Billing>
<Attribute Name="Contact"/>
<Attribute Name="Address1"/>
<Attribute Name="Address2"/>
<Attribute Name="City"/>
<Attribute Name="State"/>
<Attribute Name="ZipCode"/>
<Attribute Name="Phone"/>
```

5

Copyright Central Park Data Systems Inc

Parsed Data in Table SAXDATA

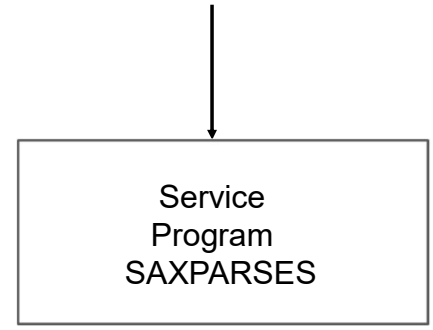
```
1 select * from xmldebug/saxdata  order by dsequence;
```

DDEVICE	DUSERID	DPARSEDATE	DDATATYPE	DDATA	DSEQU
*NONE	CHARLIE	2020-02-16	Start_Element	XMLFileIn	... 5466
*NONE	CHARLIE	2020-02-16	Start_Element	OrderV2	... 5467
*NONE	CHARLIE	2020-02-16	Start_Element	Attribute	... 5468
*NONE	CHARLIE	2020-02-16	Attr_Name	Name	... 5469
*NONE	CHARLIE	2020-02-16	Attr_Chars	Dealer	... 5470
*NONE	CHARLIE	2020-02-16	XML_Chars	10912	...5471
*NONE	CHARLIE	2020-02-16	End_Element	Attribute	... 5472
*NONE	CHARLIE	2020-02-16	Start_Element	Attribute	... 5473
*NONE	CHARLIE	2020-02-16	Attr_Name	Name	... 5474
*NONE	CHARLIE	2020-02-16	Attr_Chars	Homeowner	... 5475
*NONE	CHARLIE	2020-02-16	XML_Chars	000002	...5476
*NONE	CHARLIE	2020-02-16	End_Element	Attribute	... 5477
*NONE	CHARLIE	2020-02-16	Start_Element	Attribute	... 5478
*NONE	CHARLIE	2020-02-16	Attr_Name	Name	... 5479
*NONE	CHARLIE	2020-02-16	Attr_Chars	ProjectName	... 5480
*NONE	CHARLIE	2020-02-16	XML_Chars	New Construction 529	... 5481
*NONE	CHARLIE	2020-02-16	End_Element	Attribute	... 5482
*NONE	CHARLIE	2020-02-16	Start_Element	Attribute	... 5483
*NONE	CHARLIE	2020-02-16	Attr_Name	Name	... 5484
*NONE	CHARLIE	2020-02-16	Attr_Chars	CreatedBy	...5485
*NONE	CHARLIE	2020-02-16	XML_Chars	11714	...5486

6

Testing the Service Program

Program XMLDEBUG reads table XMLTRANDTA which contains data to simulate calls to the web service



7

Copyright Central Park Data Systems Inc

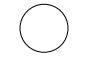
```
CREATE TABLE XMLDEBUG.XMLTRANDTA (
      CUSERID CHAR(10) CCSID 37 NOT NULL DEFAULT '' ,
      CPARSESTAT CHAR(1) CCSID 37 NOT NULL DEFAULT '' ,
      CPARSEDATE  DATE  NOT NULL  DEFAULT,
      CCMP DECIMAL(2, 0) NOT NULL DEFAULT 0 ,
      CCUSNO DECIMAL(7, 0) NOT NULL DEFAULT 0 ,
      CCSNAM CHAR(30) CCSID 37 NOT NULL DEFAULT '' ,
      CADDR1 CHAR(30) CCSID 37 NOT NULL DEFAULT '' ,
      CADDR2 CHAR(30) CCSID 37 NOT NULL DEFAULT '' ,
      CADDR3 CHAR(30) CCSID 37 NOT NULL DEFAULT '' ,
      CYTDSL DECIMAL(7, 0) NOT NULL DEFAULT 0 ,
      CYTDSLA DECIMAL(7, 0) NOT NULL DEFAULT 0 ,
      CYTDSLB DECIMAL(7, 0) NOT NULL DEFAULT 0 ,
      CYTDSLC DECIMAL(7, 0) NOT NULL DEFAULT 0 ,
      CXMLDATA CHAR(24000) CCSID 37 NOT NULL DEFAULT '' )

      RCDFMT XMLTRANDTR   ;

LABEL ON TABLE XMLDEBUG.XMLTRANDTA
      IS 'Customer XML Transaction File' ;
```

8

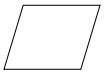
```
CREATE TABLE XMLDEBUG.SAXDATA (
      DDEVICE CHAR(10) CCSID 37 NOT NULL DEFAULT '' ,
      DUSERID CHAR(10) CCSID 37 NOT NULL DEFAULT '' ,
      DPARSEDATE  DATE,
      DDATATYPE CHAR(20) CCSID 37 NOT NULL DEFAULT '' ,
      DDATA CHAR(256) CCSID 37 NOT NULL DEFAULT '' ,
      DSEQUENCE  INTEGER
       GENERATED ALWAYS AS IDENTITY
         (START WITH 1  INCREMENT BY 1
           CYCLE)  )

      RCDFMT SAXDATAR   ;

LABEL ON TABLE XMLDEBUG.SAXDATA
      IS 'Parsed XML Transaction Table' ;
```

9

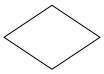
Program we will be debugging



Start program



Read a record from file XMLTRANDTA



If %EOF, leave program loop and exit program

Call XML parsing service program.
If XML is good, SAX parse to table SAXDATA.
If error occurs, return error code 'E'

Update XMLTRANDTA with parsing status

Read more records from file XMLTRANDTA

10

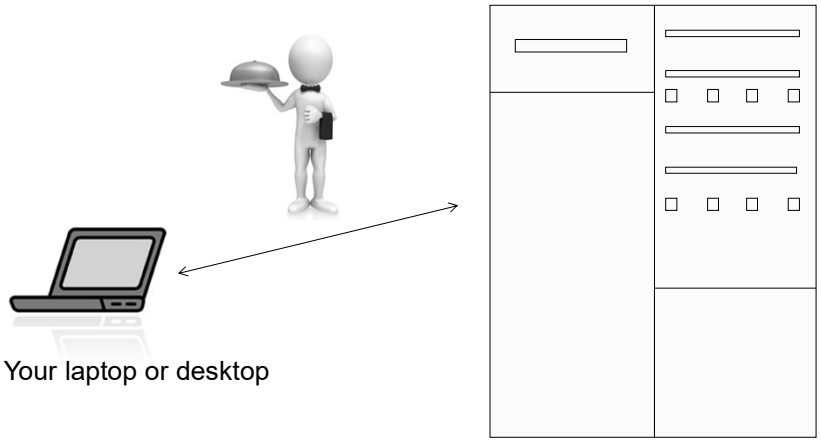
Program we will be debugging (cont.)

```
          if %parms = *zero;
            device = '*NONE';
          else;
            device = DeviceIn;
          endif;

// Begin loop

read xmltrandta;
dow not %eof (xmltrandta);

  parsingstatus = *blank;

SaxParser(cuserid : device : ParsingStatus : cxmldata);

// Update every processed row with parsing status and date

  cparsestat = parsingstatus;
  cparsedate = currentdate;
  update xmltrandtr;

read xmltrandta;
enddo;

*inlr = *on;
```

11

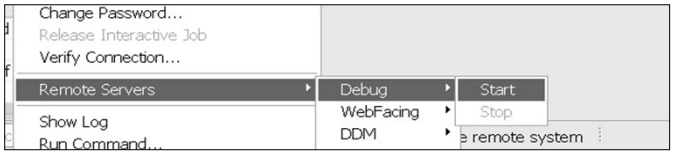
Introducing the Debug Server

- You enter debugging commands in RDi
  - The commands execute on the Remote System

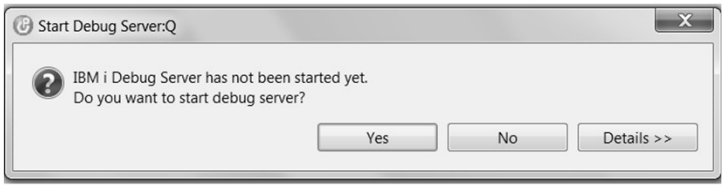


12

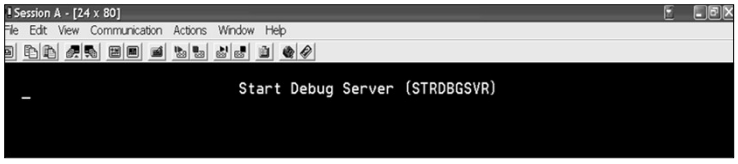
Starting the Debug Server (3 different ways!)



- OR -



- OR -



13

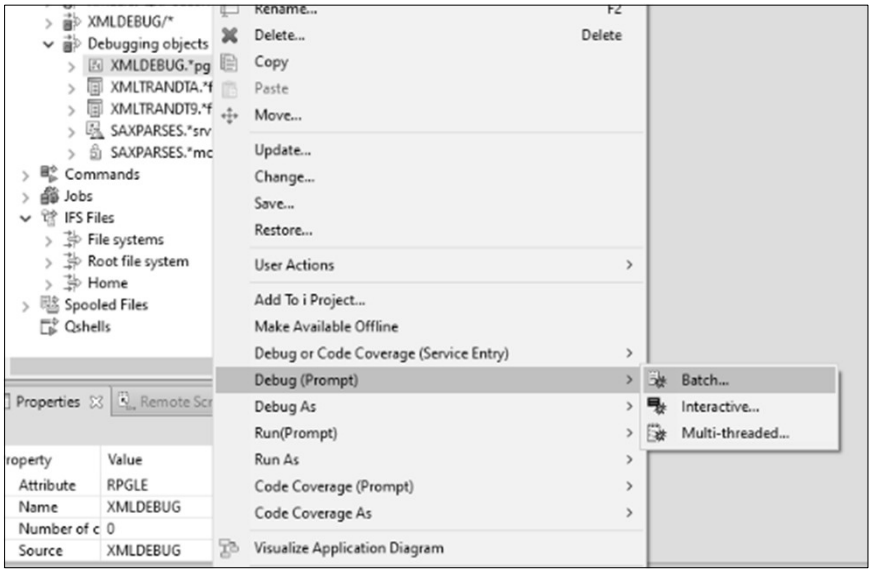
Copyright Central Park Data Systems Inc

What We'll Cover …

- Let's Get Started
- Preparing to Submit
- The Debugger Comes to Life
- Setting a Service Entry Point
- RDi Debugging Hacks
- Wrap-up

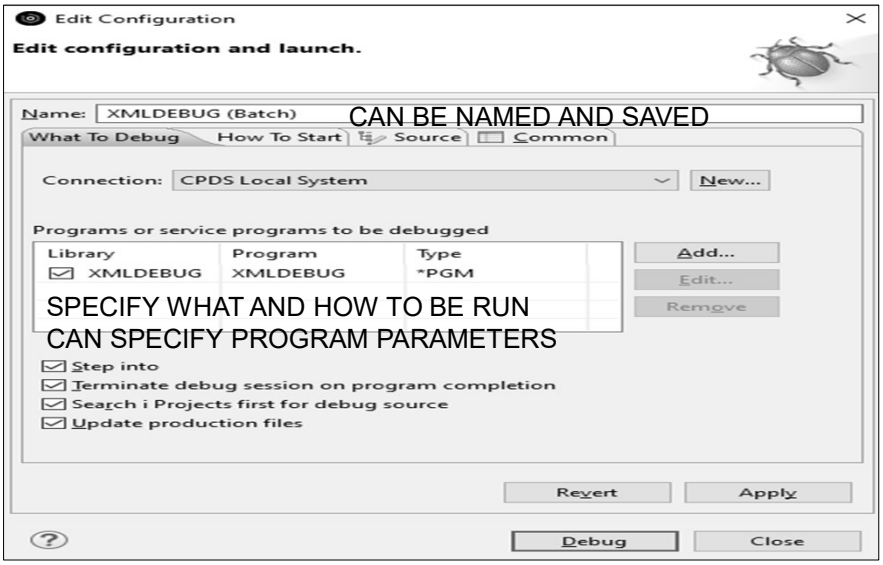
14

Calling and debugging with a prompt



15

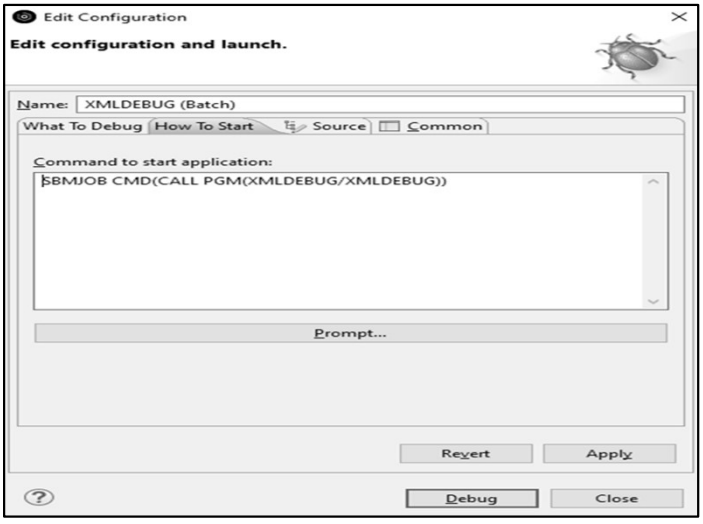
Debug configurations



16

Debug configurations (cont.)

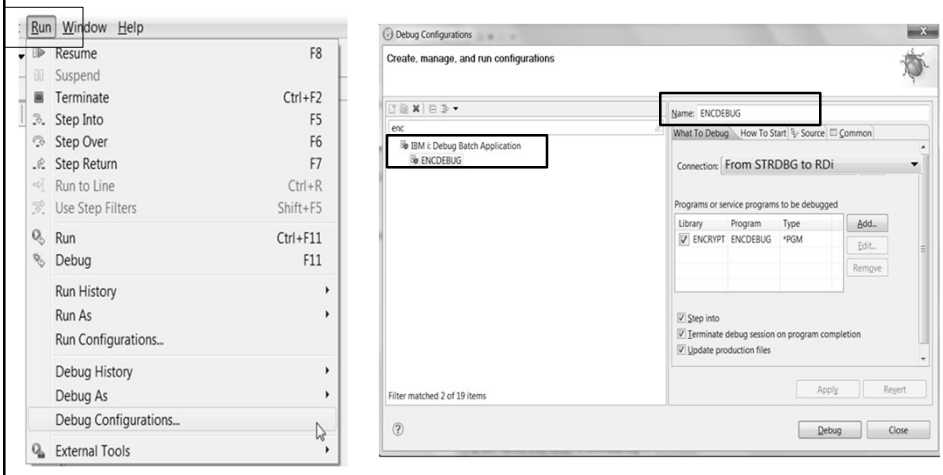
- **Click on "How to Start" for additional parameters**



17

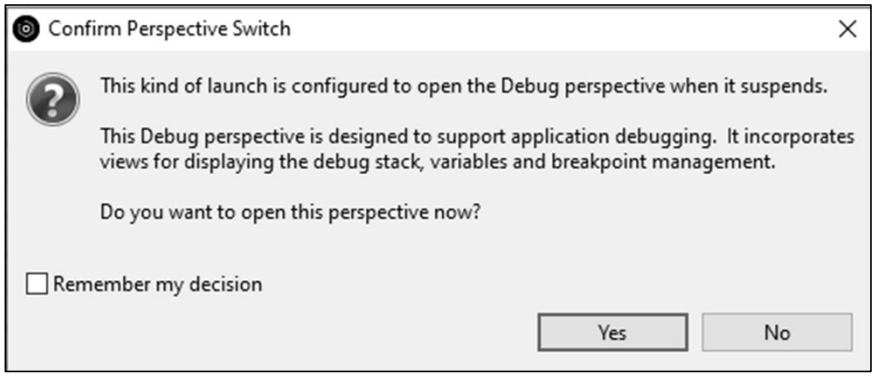
Launching an existing configuration

- Very useful is you will debugging programs multiple times
- The configuration will remember all of your settings



18

The first time you launch the debugger



19

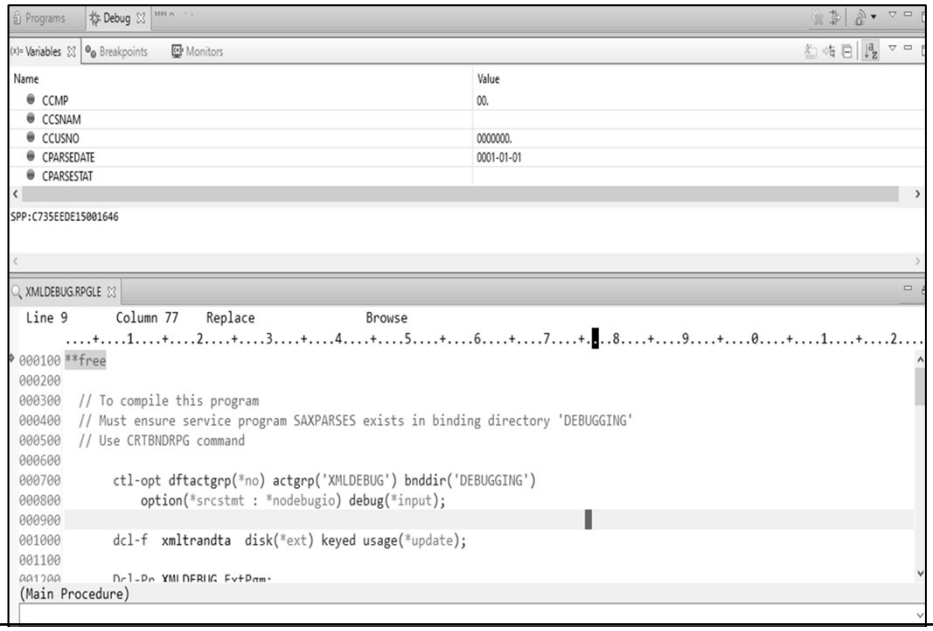
Copyright Central Park Data Systems Inc

What We'll Cover …

- Let's Get Started
- Preparing to Submit
- The Debugger Comes to Life
- Setting a Service Entry Point
- RDi Debugging Hacks
- Wrap-up

20

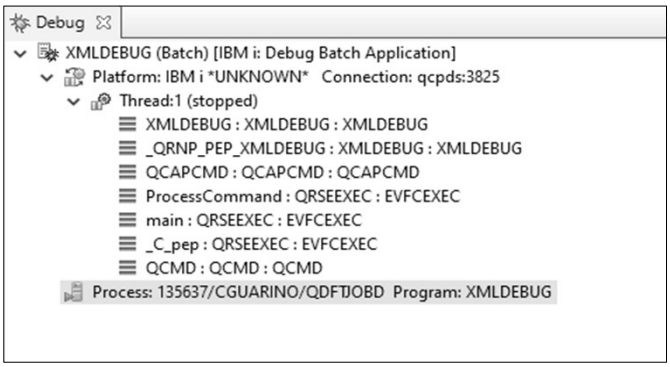
Welcome to the DEBUG perspective



21

Introducing the DEBUG view

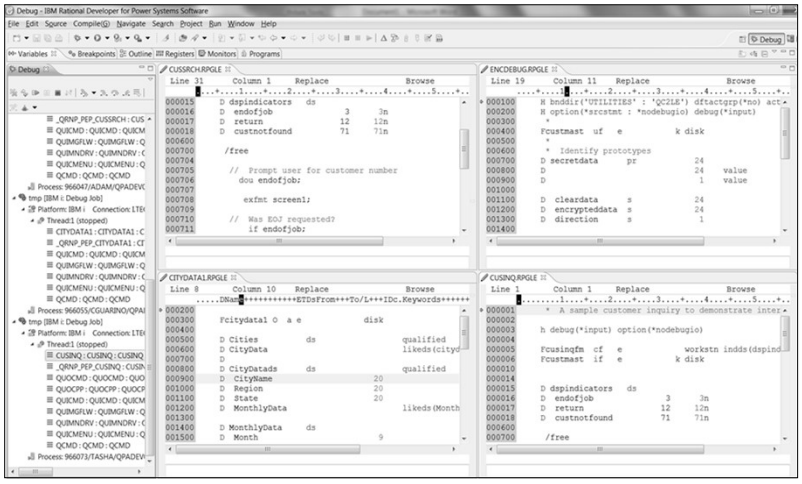
- This is the call stack and communication area between RDi and the IBM i
- Can be used to debug multiple jobs at the same time
  - Simply click on the job you want to debug



22

Debugging multiple jobs at once

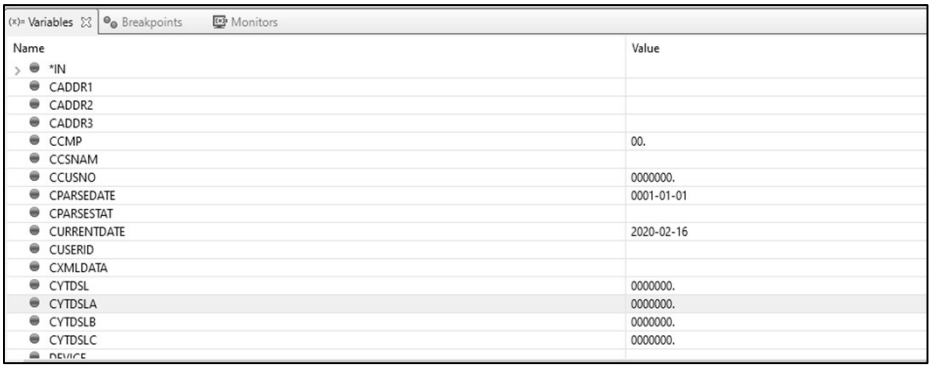
- When debugging multiple jobs at once keep the debug view open
  - Makes it easier to keep track of current job being debugged



23

Introducing the VARIABLES view

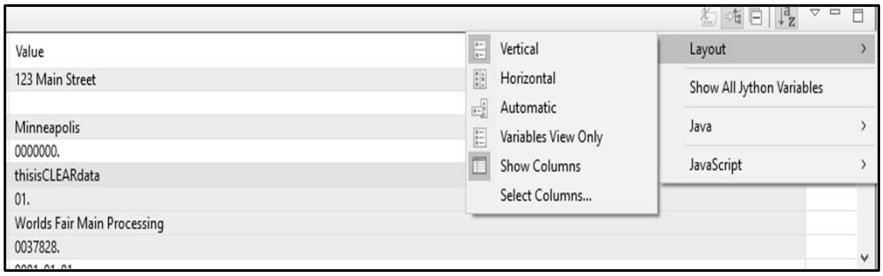
- All program variables are displayed and updated in real time
  - Each variable will change color when its value changes
- This view is customizable using the drop-down menu
- Right click to change view and add to monitors view
- Values can be changed by simply over-typing



24

Introducing the VARIABLES view – drop down menu

- All program variables are displayed and updated in real time
  - Each variable will change color when its value changes
- This view is customizable using the drop-down menu
- Right click to change view and add to monitors view
- Values can be changed by simply over-typing



25

Copyright Central Park Data Systems Inc

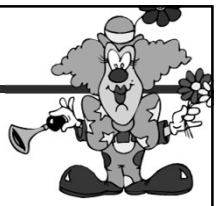
Green screen equivalent to variables view!

- **Type the debug command EVAL %LOCALVARS to see all variables!**

```
                          Evaluate Expression

Previous debug expressions

  *IN(98) = '0'
  *IN(99) = '0'
  CADDR1 = '18 Ridgepage Road             '
  CADDR2 = 'Ohio                          '
  CADDR3 = '                              '
  CAVGSAL = 0045678.
  CCLRDATA = '1234567890             '
  CCMP = 01.
  CCSNAM = 'John's Hardware Store         '
  CCUSNO = 0000145.
  CDTLSPM = '2009-12-22'
  CDTLSSL = '2009-12-01'
  CENCDATA = 'Sî▒▒B▒ÛWg8¦ûP ▒▒gÒ▒à▒▒+'
  CLEARDATA = '1234567890                 '
                                                                    More...
Debug . . .   eval %localvars_
F3=Exit   F9=Retrieve   F12=Cancel   F16=Repeat find   F19=Left   F20=Right
F21=Command entry       F23=Display output
```

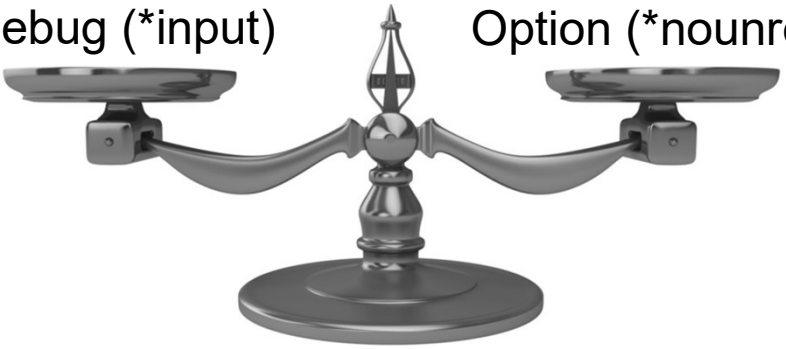
26

Unreferenced Field Fun Fact!



RNF7031

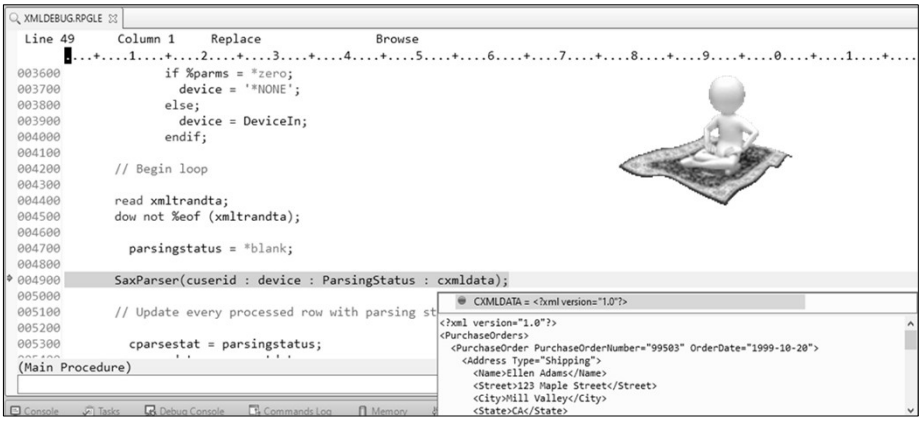
Debug (*input) Option (*nounref)



27

Field hovering

- Position the cursor over a field and its value appears.
- Much easier than typing “ev cleardata” or pressing F11!



28

**** NEW for RDi V9.6.0.7 !!! *****

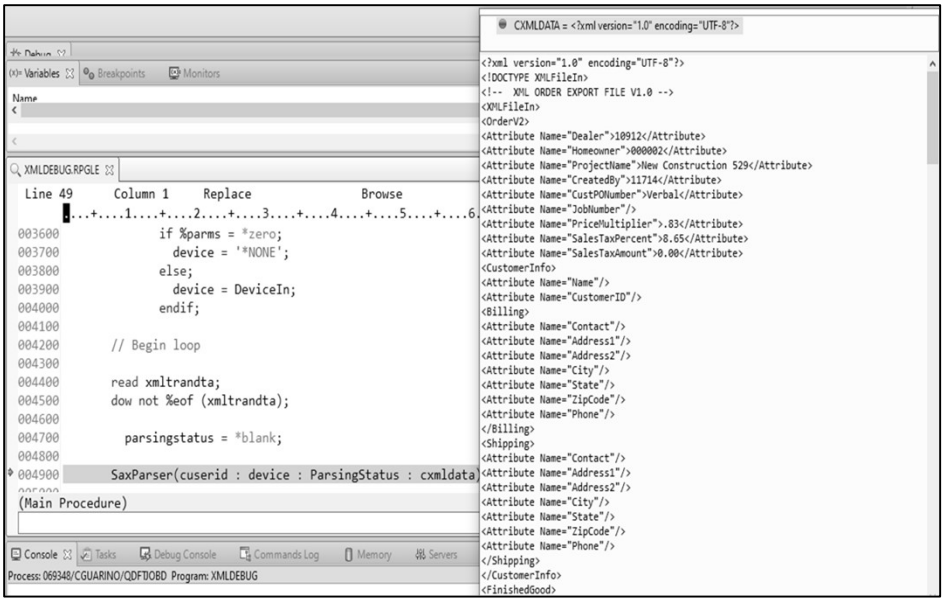
- Can now display large strings in hover, variables and monitor views
- PTF SI71314 for IBM i 7.4
- PTF SI71302 for IBM i 7.3

https://www-01.ibm.com/support/docview.wss?uid=nas3SI71314

29

Copyright Central Park Data Systems Inc

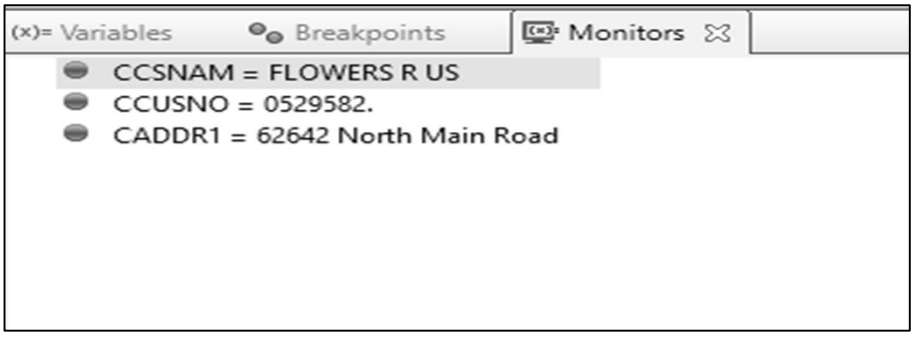
Large String Hover Display in Action



30

Introducing the MONITORS view

- **You decide which variables will appear in this view**
- **Useful when watching a specific set of fields**
- **You can right click on a field to switch between character or hexadecimal view**

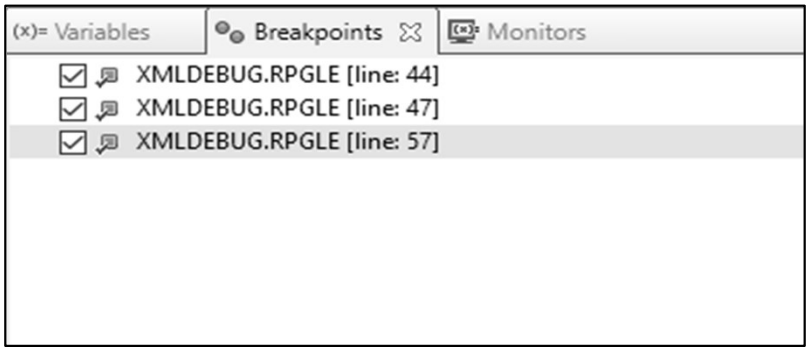


31

Copyright Central Park Data Systems Inc

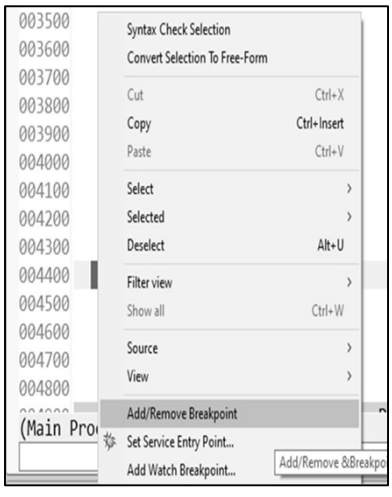
Introducing the BREAKPOINTS view

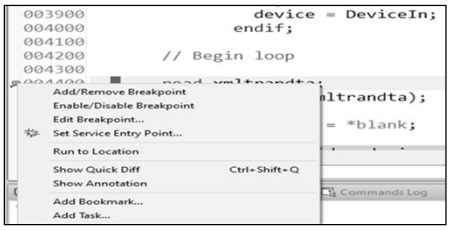
- Breakpoints can be set at the source level or at runtime
- Breakpoints can be conditional or unconditional
- Can also be disabled so you don't have to delete them
- Watch breakpoints are set at runtime – here we're watching the variable named "direction"

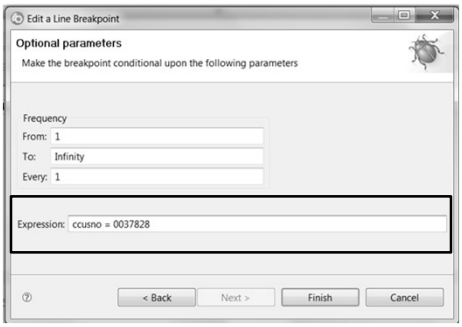


32

Adding a breakpoint

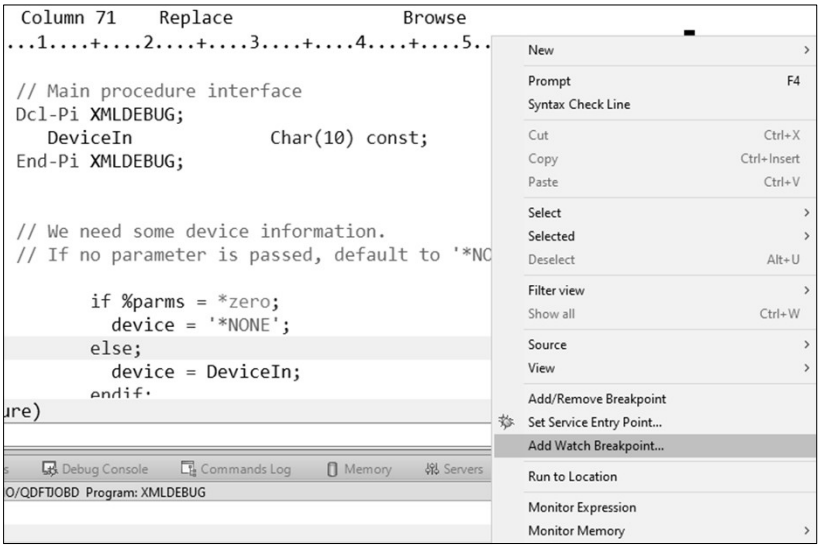






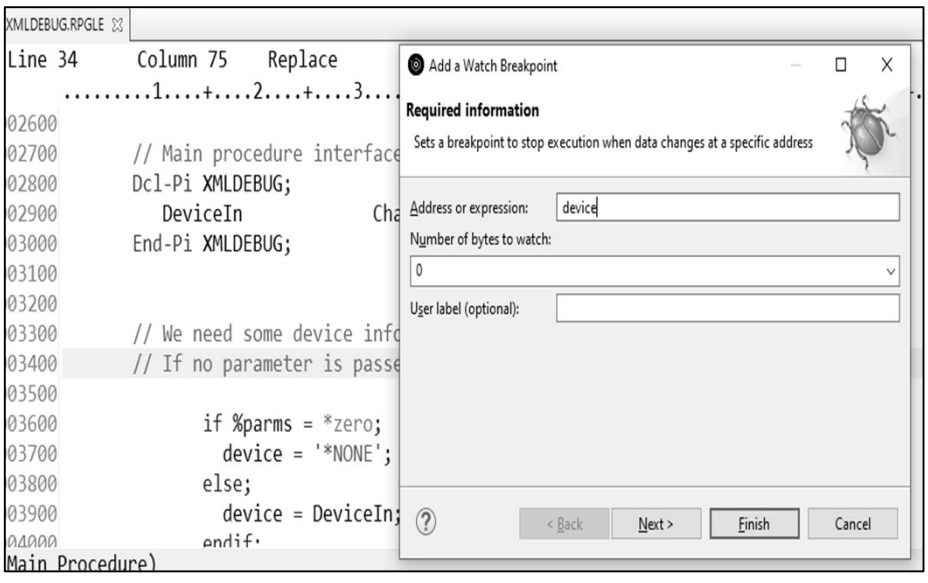
33

Adding a watch breakpoint



34

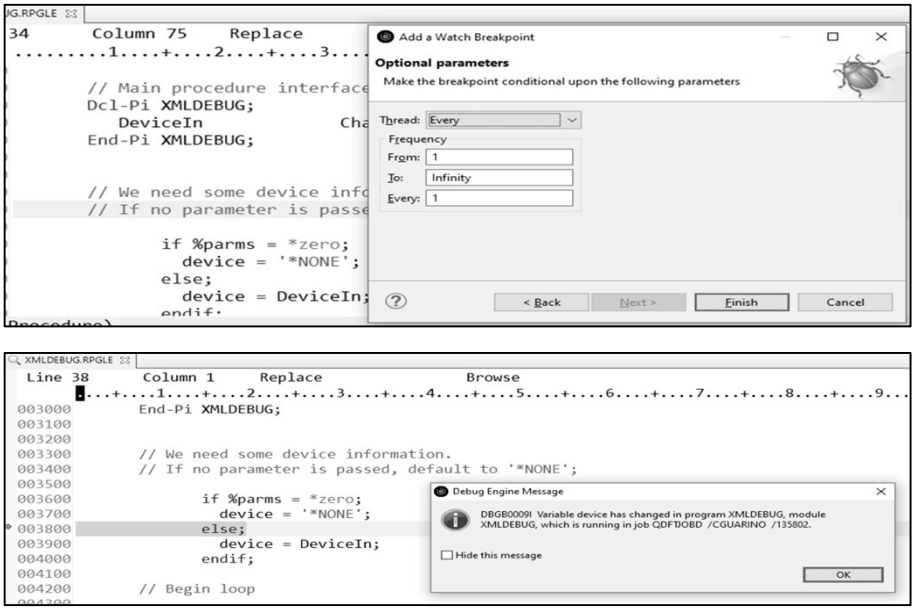
Watching for changes in the field “device”



35

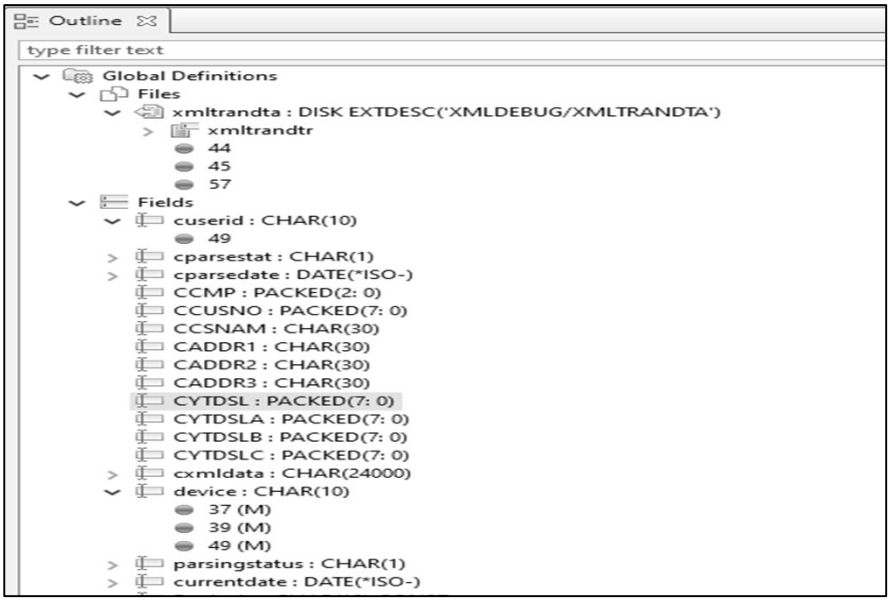
Copyright Central Park Data Systems Inc

Watching for changes in the field “device” (cont.)



36

Introducing the OUTLINE view

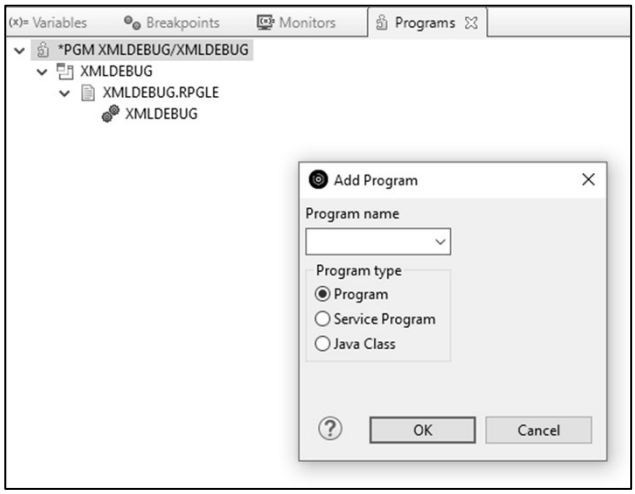


37

Copyright Central Park Data Systems Inc

Introducing the PROGRAMS view

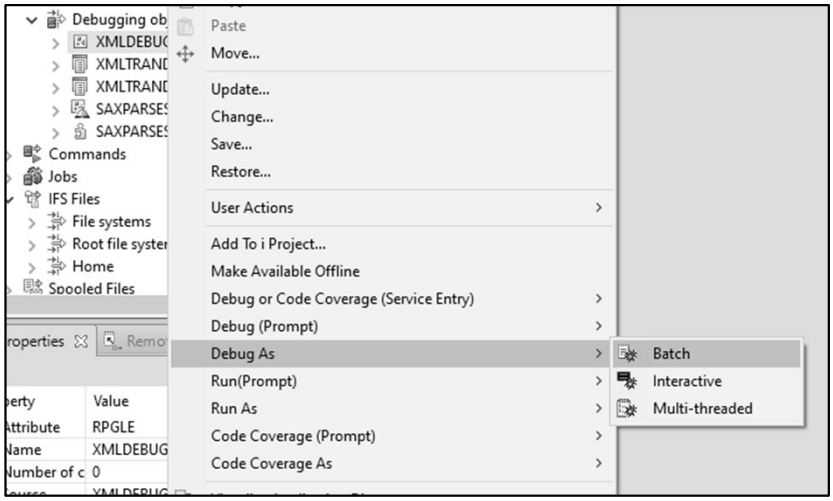
- **Functionally similar to pressing F14 from DSPMODSRC screen**



38

Submitting without any parameters

- **RDi will use the job description associated with the log in profile**



39

Copyright Central Park Data Systems Inc

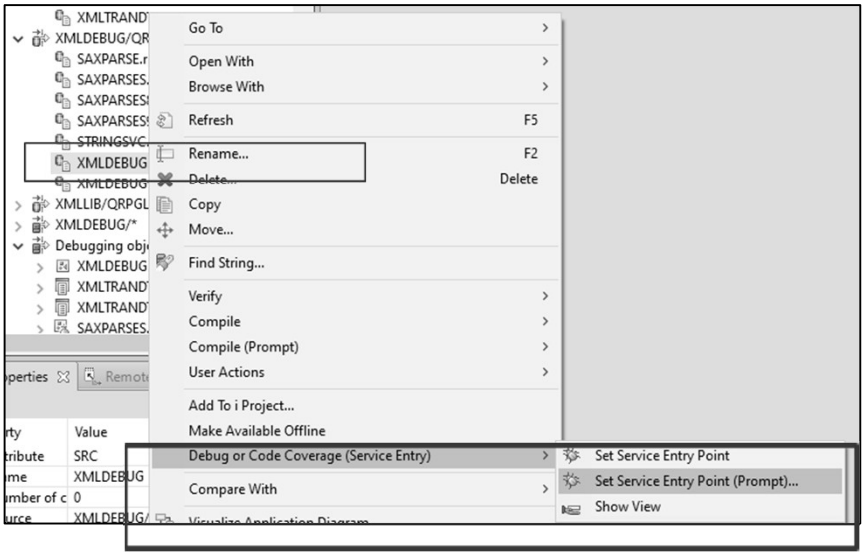
What We'll Cover …

- Let's Get Started
- Preparing to Submit
- The Debugger Comes to Life
- Setting a Service Entry Point
- RDi Debugging Hacks
- Wrap-up

40

Setting a Service Entry Point

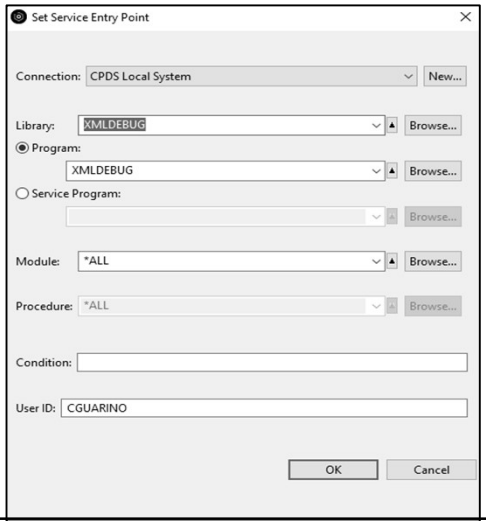
- Right click on any source member



41

Setting a Service Entry Point from a source member (cont.)

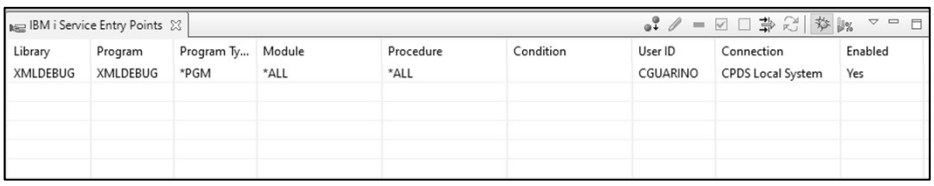
- You will have an opportunity to change any of these values
- This is a HUGE improvement over service jobs and STRSRVJOB



42

Setting a Service Entry Point from a source member (cont.)

- Once the SEP has been set you will receive this confirmation
- You will see your parameters in the SEP view in the RSE

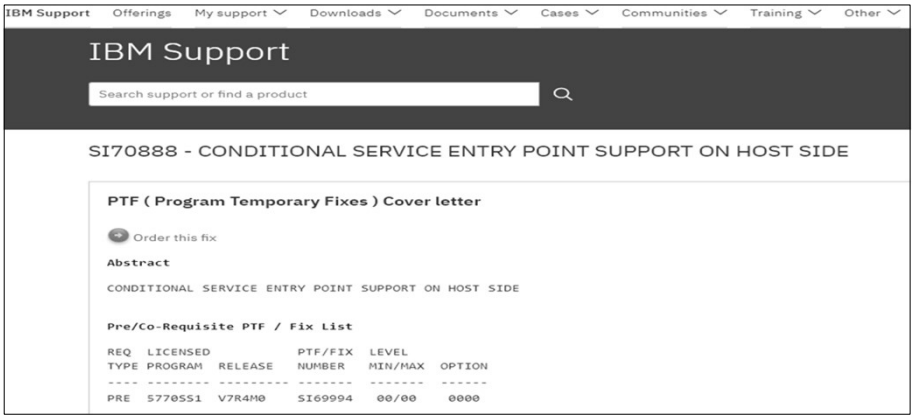


43

**** NEW for RDi V9.6.0.7 !!! *****

SEPs now allow conditional entries for breakpoints

- **PTF SI70888 for IBM i 7.4**
- **PTF SI70844 for IBM i 7.3**



https://www-01.ibm.com/support/docview.wss?uid=nas3SI70888

44

SEP with conditions test cases

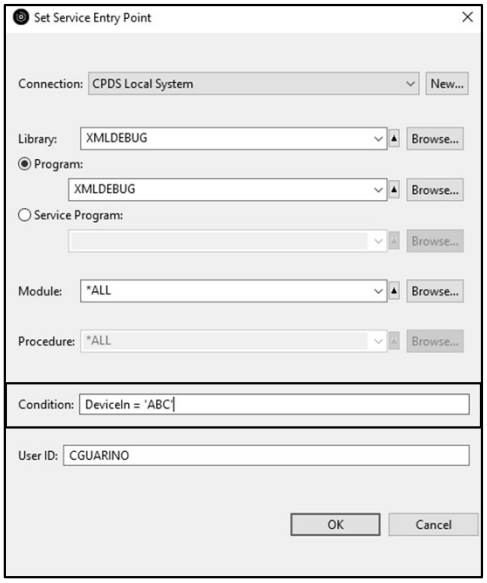
TESTDEBUG.RPGLE

```
 Line 28      Column 3     Replace
      ...+....1....+....2....+....3....+....4....+....5....+..
000700         ctl-opt dftactgrp(*no) actgrp('TESTDEBUG')
000800             option(*srcstmt : *nodebugio) debug(*input);
000900
001200         Dcl-Pr TESTDEBUG ExtPgm;
001300             Parm1             Char(1);
001301             Parm2             Char(1);
001302             Parm3             Char(1);
001303             Parm4             Char(1);
001400         End-Pr TESTDEBUG;
001500
002700         // Main procedure interface
002800         Dcl-Pi TESTDEBUG;
002801             Parm1             Char(1);
002802             Parm2             Char(1);
002803             Parm3             Char(1);
002804             Parm4             Char(1);
003000         End-Pi TESTDEBUG;
003100
003200
003501                       parm1 = parm1;
003503                       parm2 = parm2;
003505                       parm3 = parm3;
003507                       parm4 = parm4;
005900
006000         *inlr = *on;
006100         return;
```

45

Setting a Service Entry Point from a source member (cont.)

```
// Main procedure interface
Dcl-Pi XMLDEBUG;
  DeviceIn              Char(10) const;
End-Pi XMLDEBUG;
```



46

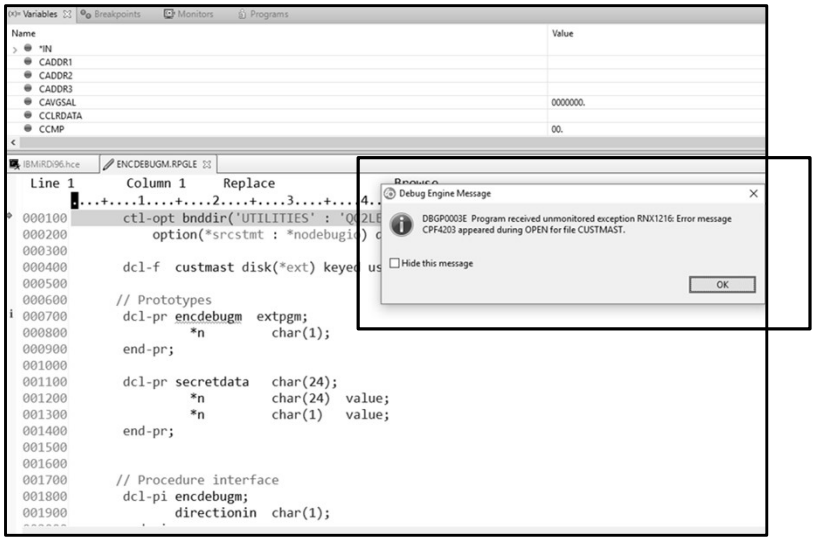
What We'll Cover …

- Let's Get Started
- Preparing to Submit
- The Debugger Comes to Life
- Setting a Service Entry Point
- RDi Debugging Hacks
- Wrap-up

47

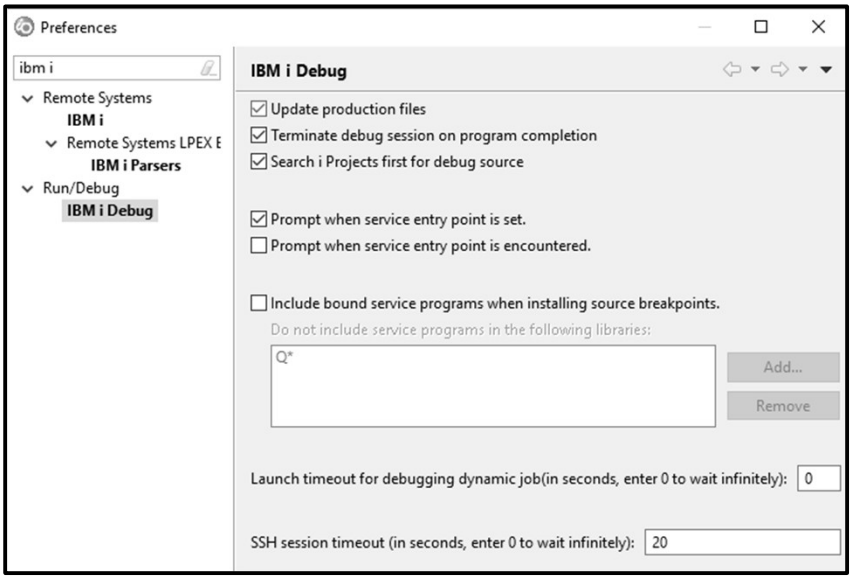
Problem: Update production files set to *NO* when using a Service Entry Point

- Received error CPF4203 when starting to debug program



48

Solution: Select Update production files set in IBM i Debug

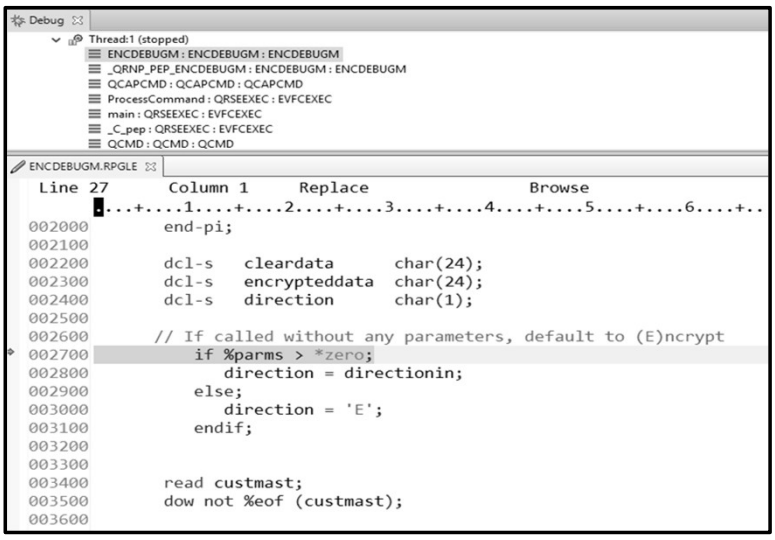


49

Copyright Central Park Data Systems Inc

Problem: Debug view keeps "popping up" every time you step through the code

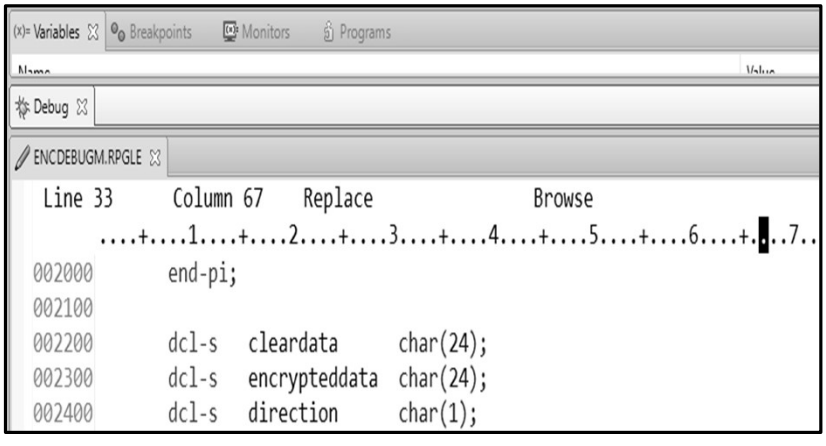
- Reason: Debug view must be open to retain communication with the debug servicer



50

Solution: Move the Debug view to an out of sight area

- Better yet, move it to a second monitor!

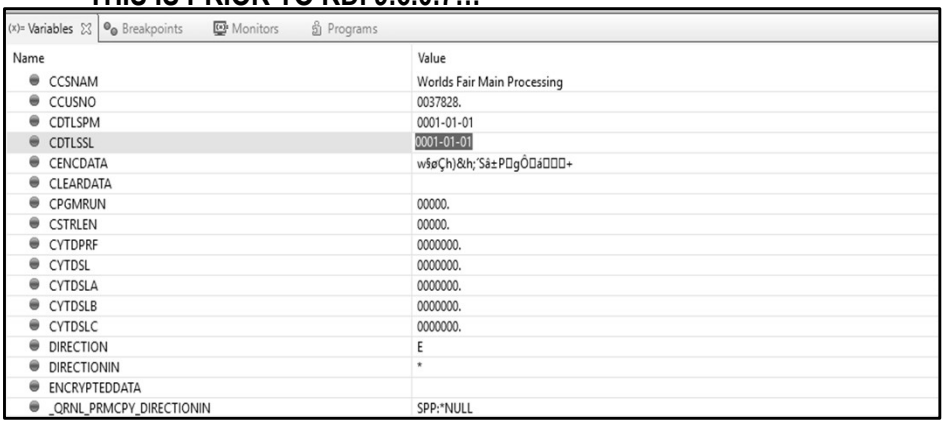


51

Copyright Central Park Data Systems Inc

Problem: Variables view doesn't show very large values

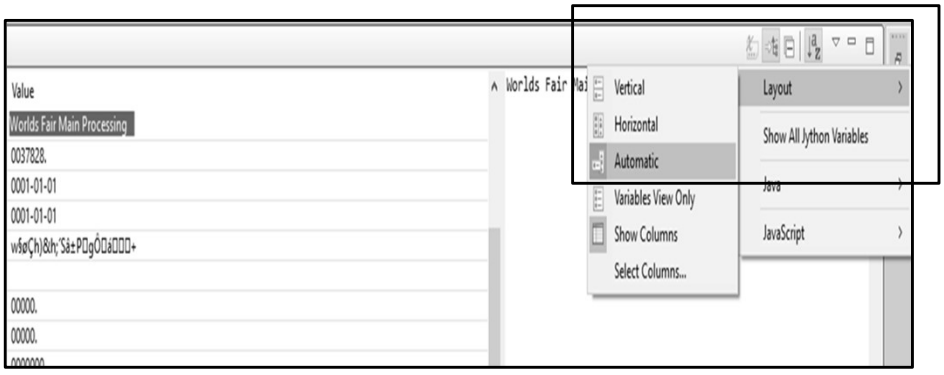
- Debug engine has a limit of 1024 characters
- THIS IS PRIOR TO RDi 9.6.0.7!!!



52

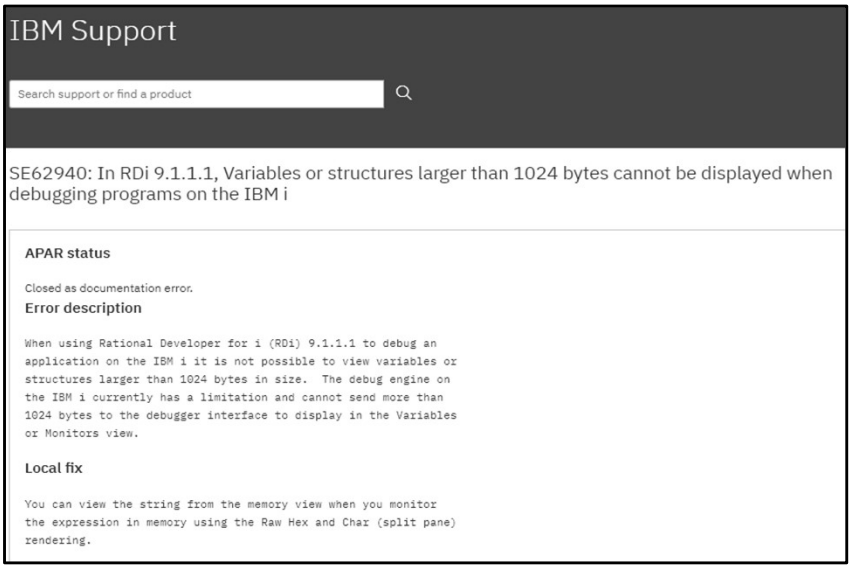
Solution 1: Change the layout of the debug view

- The upside down triangle reveals the drop down menu



53

Solution 2: Official IBM response

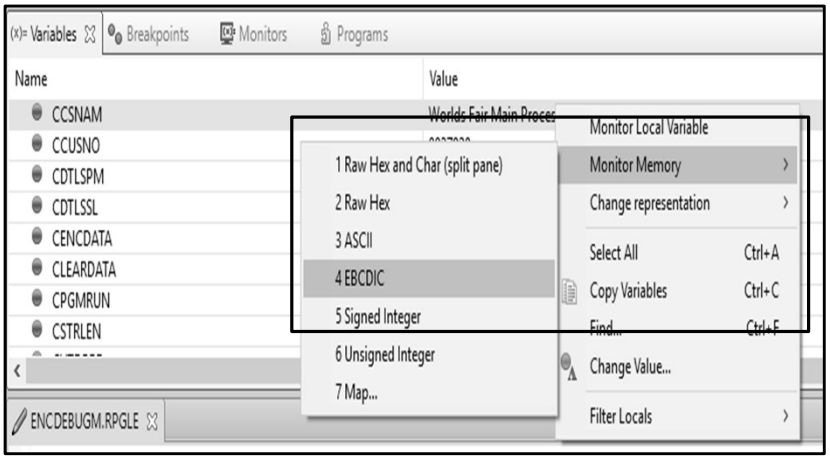


http://www-01.ibm.com/support/docview.wss?uid=swg1SE62940

54

Solution 2: Monitor Memory of the variable

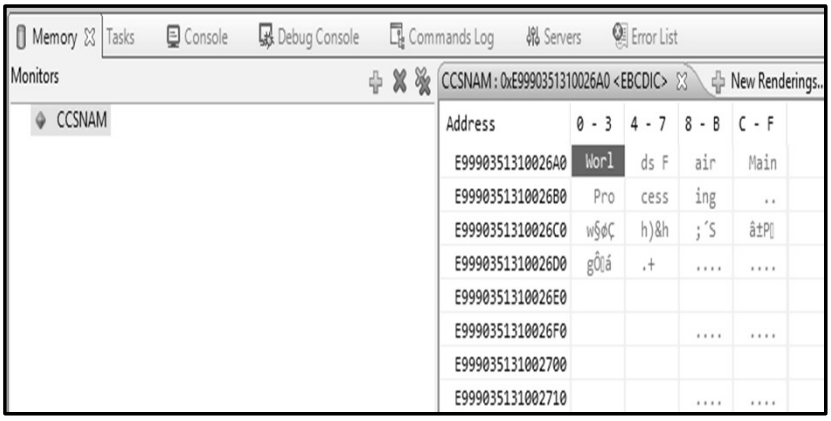
- **Choose the EBCDIC option**



55

Solution 2: Monitor Memory of the variable

- **Can then display up to first 4096 bytes of the variable**



56

Solution 3: Use STRDBG (Wait, what?)

```
                          Display Module Source

Program:   ENCDEBUGM      Library:   ENCRYPT        Module:   ENCDEBUGM
    31            endif;
    32
    33
    34         read custmast;
    35         dow not %eof (custmast);
    36
    37
    38          if direction = 'E';
    39           encrypteddata = secretdata(cclrdata:direction);
    40                 cencdata = encrypteddata;
    41          else;
    42           cleardata = secretdata(cencdata:direction);
    43                 cclrdata = cleardata;
    44          endif;
    45
                                                                    More...
Debug . . .   eval ccsnam :c 1000_
```

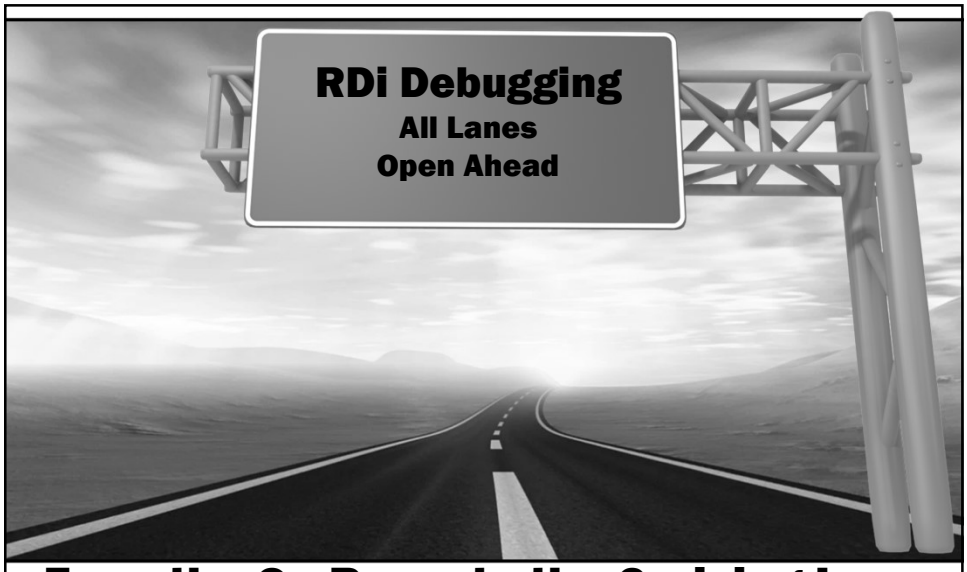
57

Copyright Central Park Data Systems Inc

What We'll Cover …

- Let's Get Started
- Preparing to Submit
- The Debugger Comes to Life
- Setting a Service Entry Point
- RDi Debugging Hacks
- Wrap-up

58



From the On Ramp to the Cruising Lane

Charles Guarino @charlieguarino

THANK YOU !!!!

59