



About The Speaker

With an IT career spanning over 30 years, Charles Guarino has been a consultant for most of them. Since 1995 he has been founder and President of Central Park Data Systems, Inc., a New York area based IBM midrange consulting company. In addition to being a professional speaker, he is a frequent contributor of technical and strategic articles and webcasts for the IT community. He is a proud member of COMMON's Speaker Excellence Hall of Fame and also Long Island Software and Technology Network's Twenty Top Techies of 2009. Charles currently serves as a member of COMMON's Strategic Education Team (SET) and is also Immediate Past President and monthly Q&A host of LISUG, a Long Island IBM i User's Group www.lisug.org.

Charles can be reached at cguarino@centralparkdata.com.

LinkedIn - <http://www.linkedin.com/in/guarinocharles>

Twitter - @charlieguarino

What We'll Cover ...

- The Path to Application Development Tools Modernization
- Crash Course on RSE and LPEX
- Out Of This World Debugging
- Hot Off The Press!
- Wrap-up

Who are you?

20

10 * 2

4 * 5

1 * 20

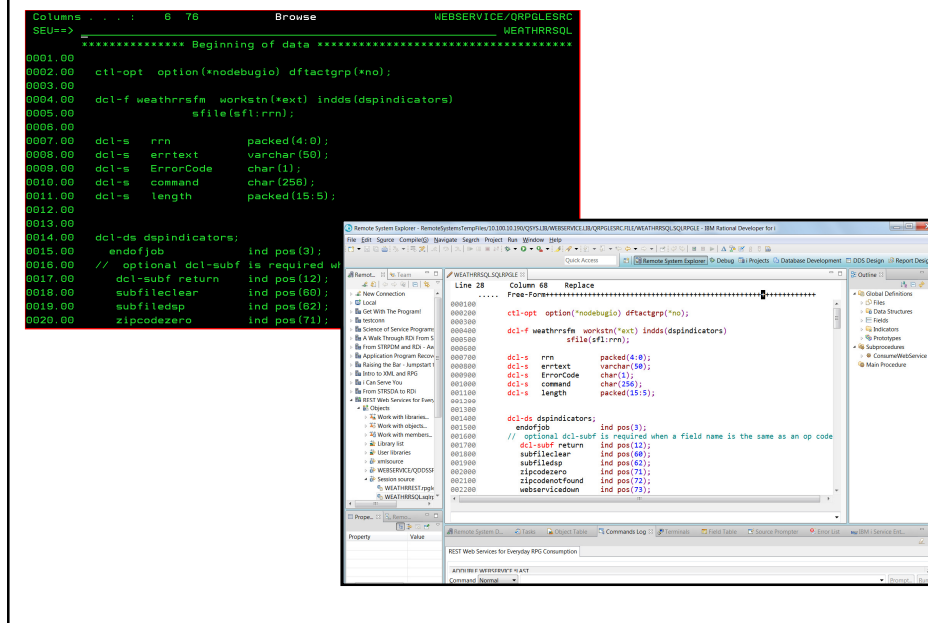
Scary Myths



Application Development Tools Modernization Myth #1

"I can develop faster using what I already know"

CLI versus IDE



Application Development Tools Modernization Myth #2

“LPEX is just another editor, just like SEU”

SEU vs LPEX – It's Not Even A Fair Fight

SEU features

"Full Screen" Source code editing

Can prompt fixed format specs

Left side SEU commands ("D", "P", "CC" etc.)

- Supports RPG free form enhancements
- Tokenization (colorized)
- Copy and Paste
- Edit multiple members at once
- Hyperlinks
- Windows functions (CTRL-Z, CTRL-Y, Up/Down)
- Member and source code filtering
- Data Structure and Procedure Wizards
- Split screen view of source code editor
- Automatic update of Outline view
- Source Code Verification
- Content Assist and source code prompting
- Set breakpoints while editing source code
- Automatic closure of control blocks
- Show block nesting
- Change font size
- Customize screen for maximum productivity
- Display help text for prompted statements
- Control block indentation and auto complete



Application Development Tools Modernization Myth #3

"The new tooling is difficult to learn"

Meet the Perspectives...

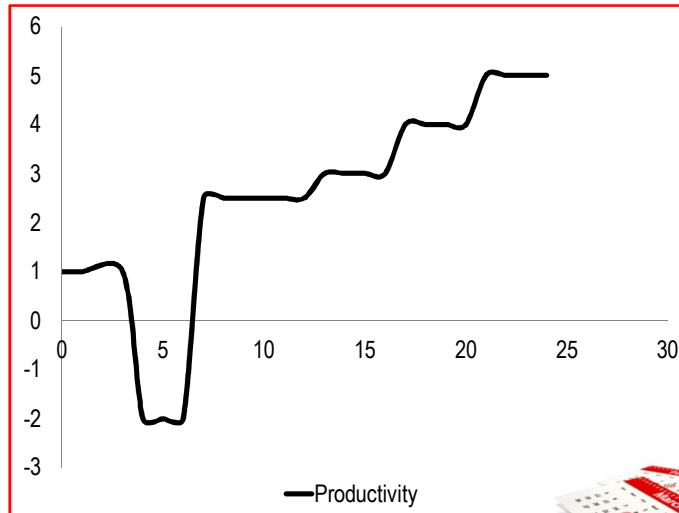
The collage consists of four overlapping screenshots of the RDI IDE interface:

- Remote Systems Explorer (RSE):** A screenshot showing a tree view of a remote system's file structure on the left and a code editor on the right.
- Debug:** A screenshot showing a code editor with a 'Debug' toolbar at the top and a 'Debug Console' at the bottom.
- Graphical Screen and Report Designer:** A screenshot showing a graphical design view with various UI components like buttons and text fields.
- General Code Editor:** A screenshot showing a code editor with a 'Line 18' indicator and a 'Column 18' indicator.

Application Development Tools Modernization Myth #4

"I'm too busy and it will take too long to learn RDi"

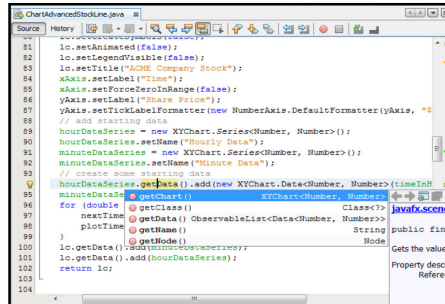
Productivity Gains Over Time



RDi Truths

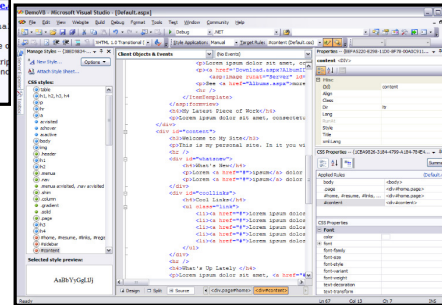


Application Development Tools Modernization Truth #1



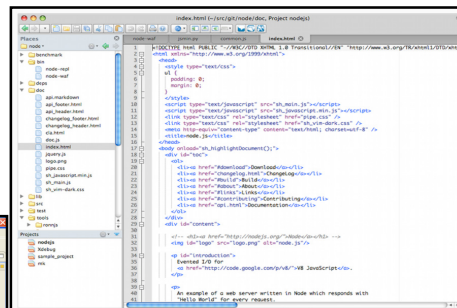
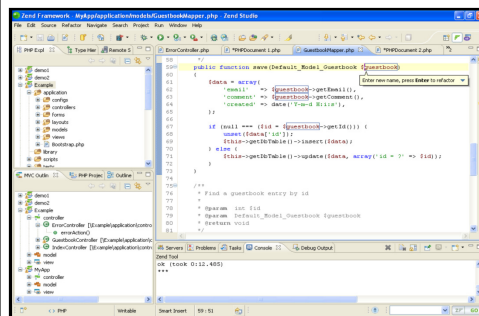
NetBeans IDE for Java
www.netbeans.org

Visual Studio
www.visualstudio.com



Application Development Tools Modernization Truth #1

Komodo Edit
<http://www.activestate.com/komodo-edit>



Zend Studio for PHP
www.zend.com

Application Development Tools Modernization Truth #2



What We'll Cover ...

- The Path to Application Development Tools Modernization
- Crash Course on RSE and LPEX
- Out Of This World Debugging
- Hot Off The Press!
- Wrap-up

RSE and LPEX – A Winning Combination

The screenshot displays the SAP IDE interface with the Remote System Explorer (RSE) and the Local Program Editor (LPEX) open. The RSE pane on the left shows a tree view of the remote system 'CITYDATAA8RPGLE'. The LPEX pane on the right shows the code for the 'CITYDATAA8RPGLE' program, which is a data transfer program using the 'CITYDATA' data source. The code includes a 'dcl-f' statement for the output file, a 'dcl-ds' statement for the data source, and a 'dcl-s' statement for the data source. The code is as follows:

```

Line 1      Column 1      Replace
1...+...1...+...2...+...3...+...4...+...5...+...6...+...7...+...8...
000100
000200
000300      dcl-f citydata1 disk("ext"   usage("output" rename(citydata1:citydata1);
000400
000500      dcl-ds cities qualified;
000600          CityData      likeds(CityDataDS) dim(2);
000700      end-ds;
000800
000900      dcl-ds CityDataDS qualified;
001000          CityName      char(20);
001100          Region        char(20);
001200          State          char(20);
001300          MonthlyData    likeds(MonthlyData) dim(3);
001400      end-ds;
001500
001600      dcl-s MonthlyData;
001700          Month          char(9);
001800          Low            char(3);
001900          High           char(3);
002000      end-s;
002100
002200      dcl-s cities1      char(100);
002300

```

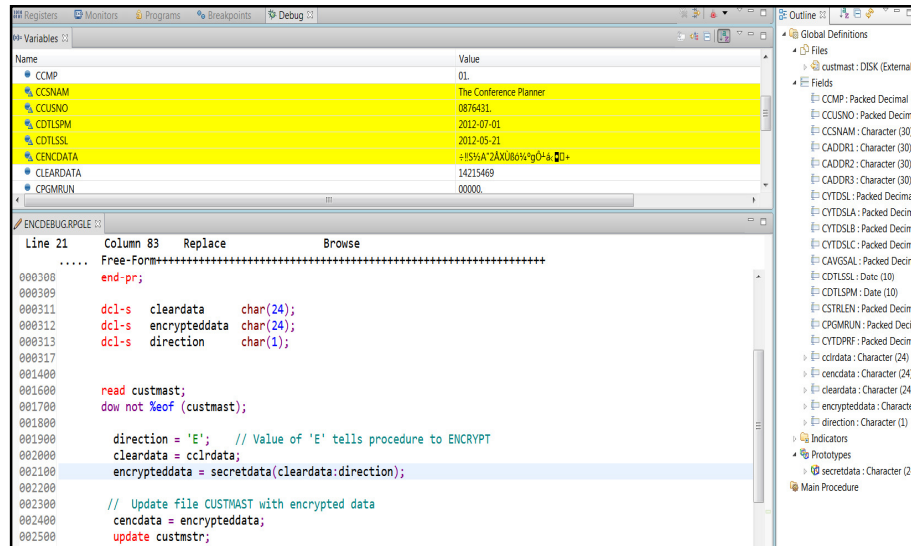
The Properties pane at the bottom left shows the properties of the 'Remote System Details' table. The table has columns for Name, Record, Type, Length, Text, and Alias. The table is currently empty.

What We'll Cover ...

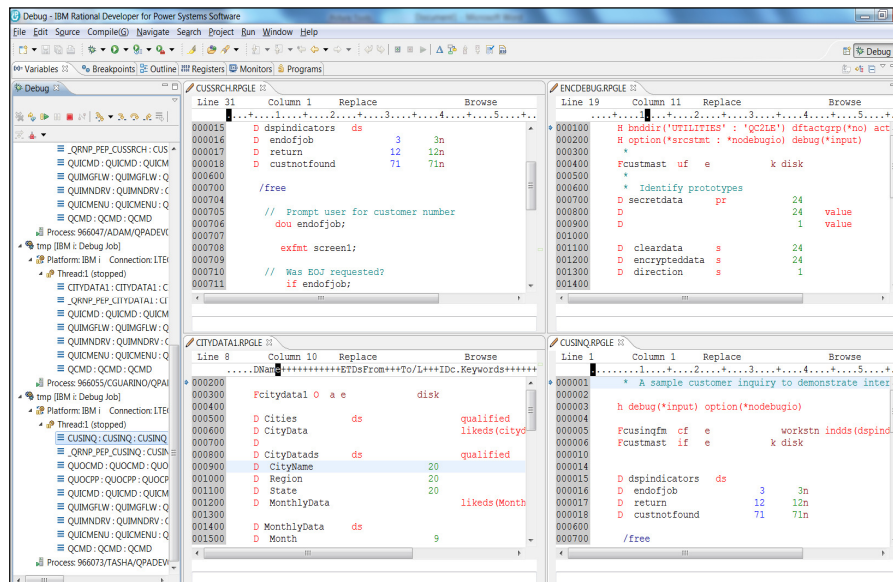
- The Path to Application Development Tools Modernization
- Crash Course on RSE and LPEX
- Out Of This World Debugging
- Hot Off The Press!
- Wrap-up

- The Path to Application Development Tools Modernization
- Crash Course on RSE and LPEX
- Out Of This World Debugging
- Hot Off The Press!
- Wrap-up

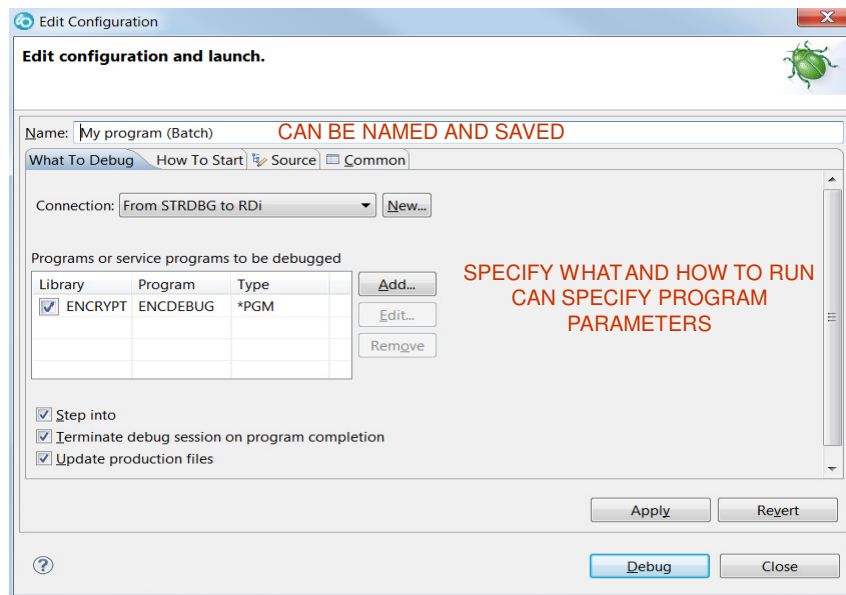
Debugging free form RPG using RDi



Can debug multiple jobs at the same time!



Debug configurations



What We'll Cover ...

- The Path to Application Development Tools Modernization
- Crash Course on RSE and LPEX
- Out Of This World Debugging
- Hot Off The Press!
- Wrap-up

Version 9.1 is Announced and Available Shortly



Passport Advantage

May 23 – Electronic Availability

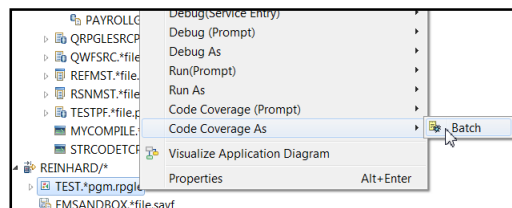
June 6 – Physical Availability

Advanced Administration System (AAS)

June 13

RD9.1 Line level Code Coverage Analysis Capability




- Code coverage can be launched on any program or service program that can be debugged – independent of language



- You can see exactly which lines were covered and not.
- This can be used to determine the effectiveness of automated or manual tests
- Can help focus additional testing on code paths that have not been executed
- Can aid in detecting duplicate tests
- Can help recognize dead code

Code Coverage Report


- After running code coverage, a report is shown as an editor.
- You can drill down through programs, modules and procedures and see the coverage statistics for each

Code Coverage Report (RBC4NNF1C_2014_04_23_123357_0632)   

Code Coverage Report (Line)



Code coverage report (analyzed at Apr 23, 2014 12:34:22 PM, generated at Apr 29, 2014 10:55:30 AM)

Element	Coverage	Covered	Total
*PGM MKWAN/RBC4NNF1C	88%	617	699
RBC4NNF1C	88%	617	699
RBC4NNF1C.RPGLE	88%	617	699
GRHPRCCMP10()	0%	0	6
UCSPRCCMP8()	0%	0	6
INDPRC1()	0%	0	6
CHRPCCMP12()	0%	0	6
PUT_TC_MSG()	0%	0	13
finish_tc()	78%	7	9
chk_result()	89%	8	9
show_error()	90%	18	20
RBC4NNF1C()	91%	406	444
chk_subres()	92%	24	26
DATPRCISO()	100%	3	3
DATPRCUSA()	100%	3	3
DATPRCEUR()	100%	3	3
DATPRCYMD()	100%	3	3

Report 

Coverage annotated in the editor

- Drilling down from the report, the editor will be opened on the related member with green and red annotations showing which lines were covered.

Code Coverage Report (RBC4NNF1C_2014_04_23_115425_0227)  RBC4NNF1C.RPGLE 

```

Line 1490   Column 50   Replace   Browse
.....PName.....B.....Keyword.....Comments.....
P finish_tc   b         export
D finish_tc   pi
D tc          liked(tc_t)
/free

// If all variations passed, set the test case status to successful
if tc.succ_v = tc.run;
  RptTest (tc.testname: tc.succ_v: tc.fail_v: SUCCESS);
return;
endif;

// Otherwise, write a summary line and close the output file
VarFail ('*CLOSE': tc.succ_v);
RptTest (tc.testname: tc.succ_v: tc.fail_v: FAILURE);

close qprint;

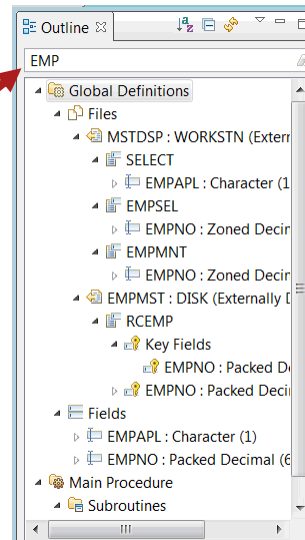
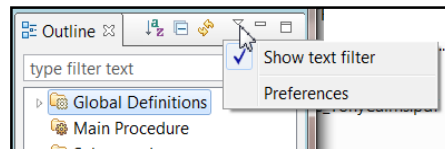
/end-free
P finish_tc   e

P put_tc_msg  b         export
D put_tc_msg  pi
D msgPtr     *         value options(*string)
D MAX LINE   c         52

```

RDi 9.1 - Filterable outline view

- The following 2 RFEs were implemented using an optional filter field at the top of the outline view that subsets the view as you type, quickly zeroing in on the definition you had in mind.
- RFE 10188 Search within Outline VIEW - 28 votes
- RFE 24677 Outline View: Filtering - 20 votes
- As you can see in the screen cap below, after type EMP, only those definitions containing EMP are visible in the outline view



Extend IBM i member filters to be able to use member text

- CAAC AD0150 – IBM is listening!
- With 10 characters, names are too restrictive for filtering

IBM Toronto: File REINHARD/QSRC (54 Members)

Name	Type	Attribute	Text
CBLTESTLE	CBLL	SRC	PAYROLL
CBLTEST2	CBL	SRC	Accounts receivable
CCSIDPF	PF	SRC	Accounts receivable
CCSIDPMR	DSPF	SRC	PAYROLL
CLP401F1	RPGLE	SRC	

New Member Filter

Create a new IBM i member filter

Library: REINHARD

File: QSRC

Member filter: *

Member text: payroll

Member type: Enter partial text of the items to be subset. Do not need *

More Types >>

REINHARD/QSRC(*)-Payroll

CBLTESTLE.cbll

CCSIDPMR.dspf

What We'll Cover ...

- The Path to Application Development Tools Modernization
- Crash Course on RSE and LPEX
- Out Of This World Debugging
- Hot Off The Press!
- Wrap-up

COMMON Europe Enterprise Modernization Tour

