

# Leveling the Playing Field - Rational Developer for System Z (RDz) Amplifies Skills and Attracts New Talent to Z

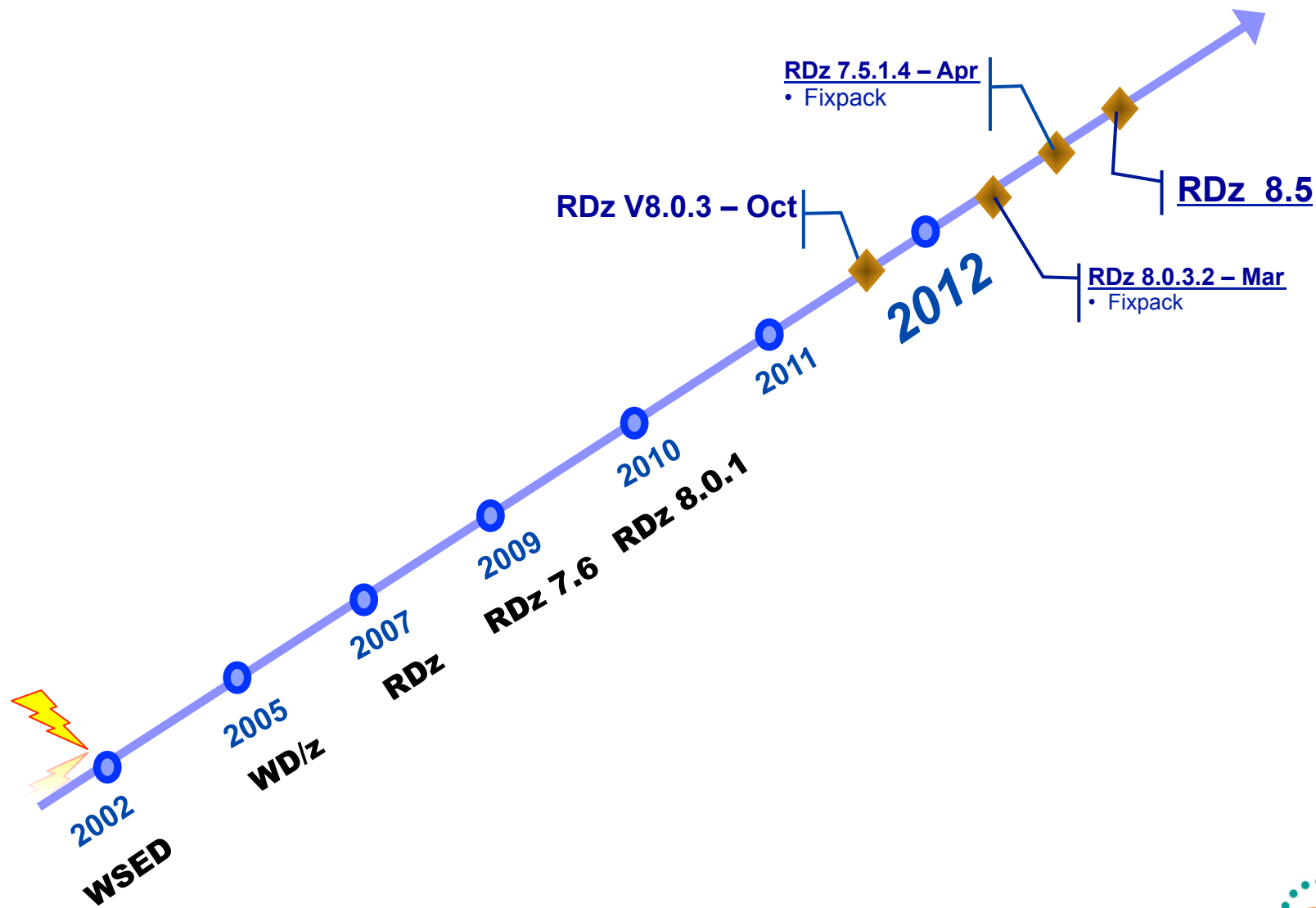
Venkatuday Balabhadrapatruni  
August 9, 2012



# Disclaimer

© Copyright IBM Corporation 2012. All rights reserved. The information contained in these materials is confidential and 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. 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, Rational, the Rational logo, Telelogic, the Telelogic logo, and other IBM products and services are 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.

# Rational Developer for System z - v.Next





# Deeper dive into RDz v8.5 ....

Resource	Parent filter poc	Parent filter	Number of filter	Connection-priv
Retrieved Jobs	CN-ctfmvs08-c	Not applicable	1	Yes
My Jobs	Dana-Boudreau	Not applicable	1	No



# Rational Developer for System z Roadmap Themes



- Performance and Scalability

- *Push to client*
- *Search*

- Productivity

- Languages

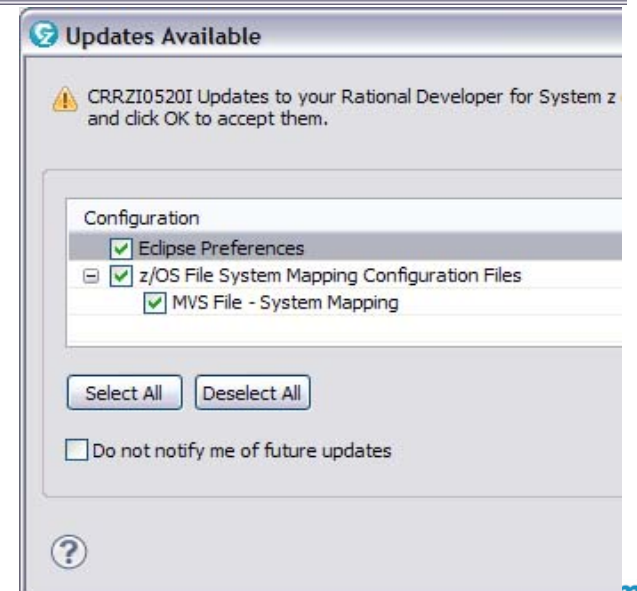
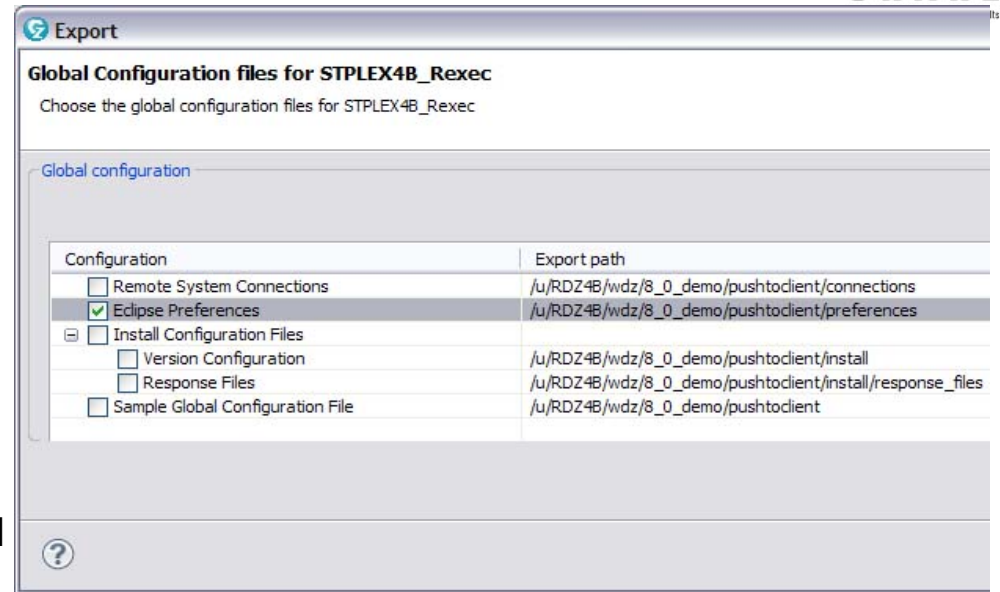
- Integration

- Advancing technologies

# RDz 8.0.x Push to Client - Configuration Files



- An Export Wizard – to help one person – preferably an Admin to configure various settings on RDz client and upload it to a central location.
- Upon connecting to the host, a user is notified if they have incoming changes to their configuration files.
- If the product version on the client is less than a specified target version, the client will be updated to the target version.
- The target version is associated with an Installation Manager response file on the host.
- Client workbench is updated via Installation Manager update
- RDz 8.0.3 – Added Group based configuration



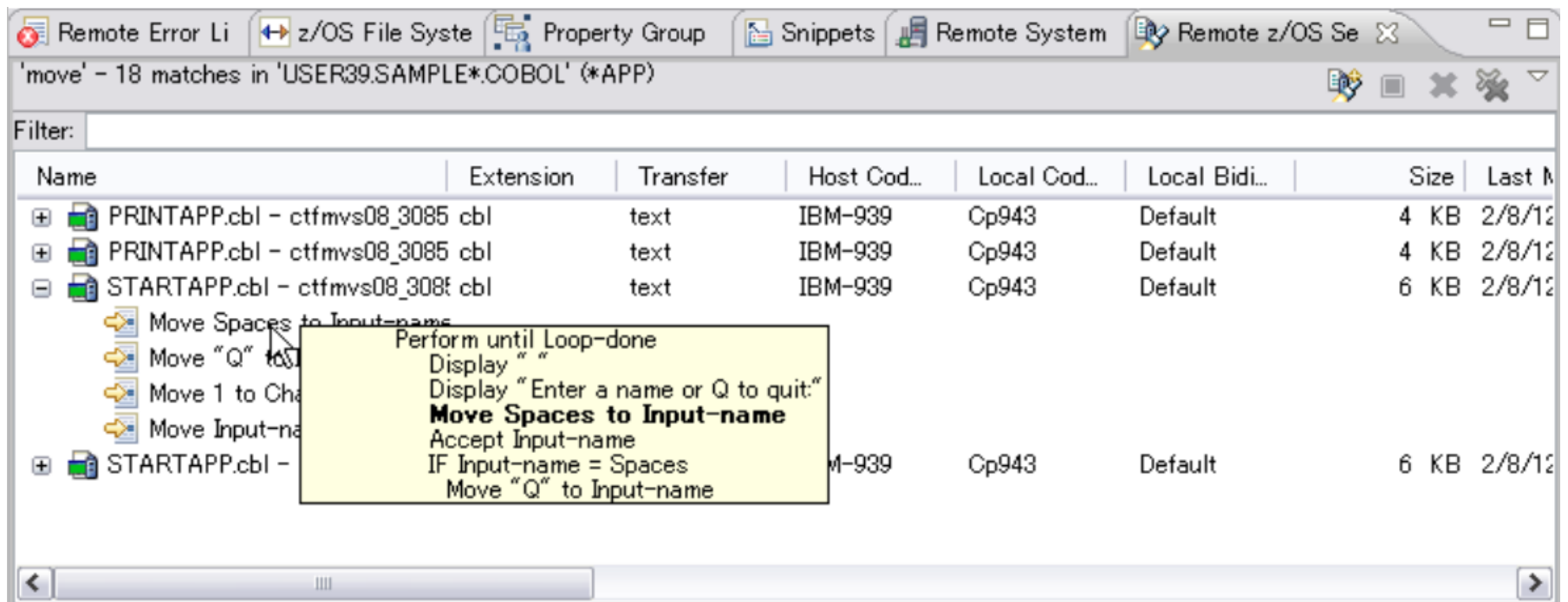
## Push to client - 8.5

### *Enhanced Push to Client Software Analyzer rules behavior*

- Provide a "merge" capability so that the users changes do not get overridden when Code Rule configs are re-downloaded
  - *Host setting*

## z/OS Remote Search View enhancements – 8.5

- Provide an ability to "preview" +/- a few lines of code before opening
- Search view is enhanced to show partial host file contents as tooltip
- Highlighting current line in the search result contents preview
- Controlled by Preferences (next slide)



'move' - 18 matches in 'USER39.SAMPLE\*.COBOL' (\*APP)

Filter:

Name	Extension	Transfer	Host Cod...	Local Cod...	Local Bidi...	Size	Last M
PRINTAPP.cbl - ctfmvs08_3085	cbl	text	IBM-939	Cp943	Default	4 KB	2/8/12
PRINTAPP.cbl - ctfmvs08_3085	cbl	text	IBM-939	Cp943	Default	4 KB	2/8/12
STARTAPP.cbl - ctfmvs08_308f	cbl	text	IBM-939	Cp943	Default	6 KB	2/8/12
STARTAPP.cbl -			IBM-939	Cp943	Default	6 KB	2/8/12

Tooltip content:

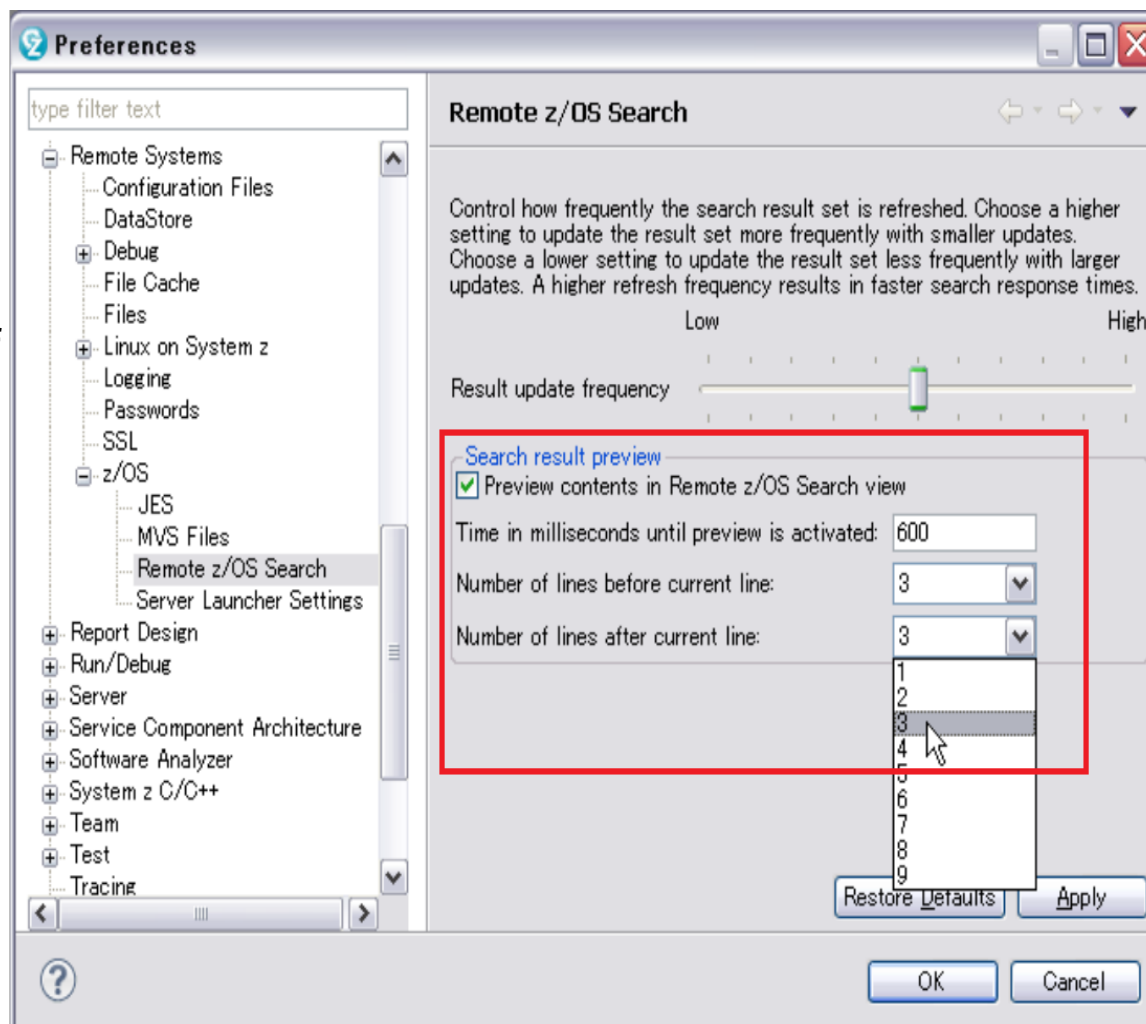
```

Perform until Loop-done
Display " "
Display "Enter a name or Q to quit:"
Move Spaces to Input-name
Accept Input-name
IF Input-name = Spaces
Move "Q" to Input-name
    
```



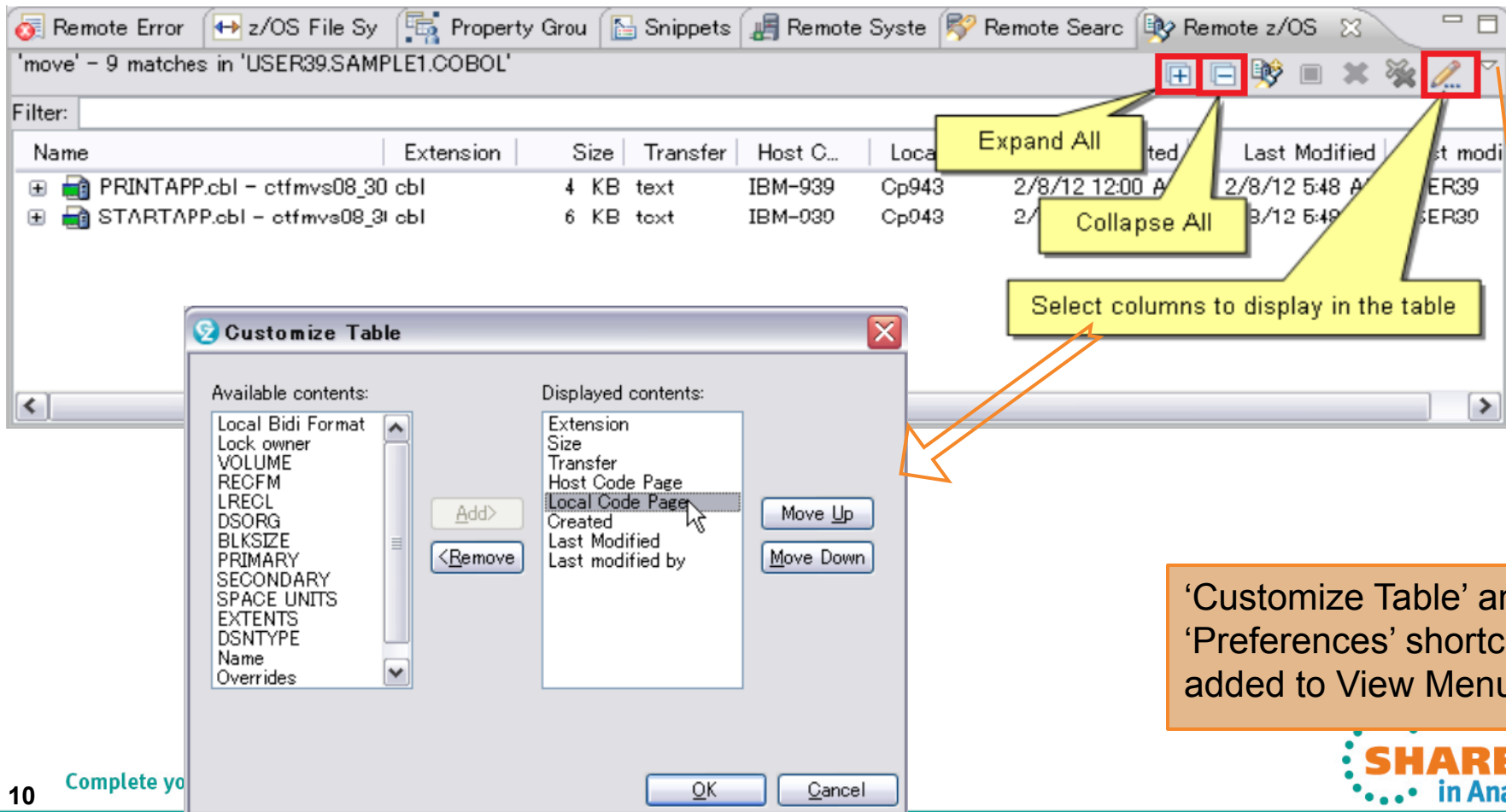
## z/OS Remote Search View enhancements – 8.5

- Remote z/OS Search preference
  - ▶ Three input controls are configurable
  - ▶ Inputs are disabled if preview checkbox is unchecked



## z/OS Remote Search Usability Improvements – 8.5

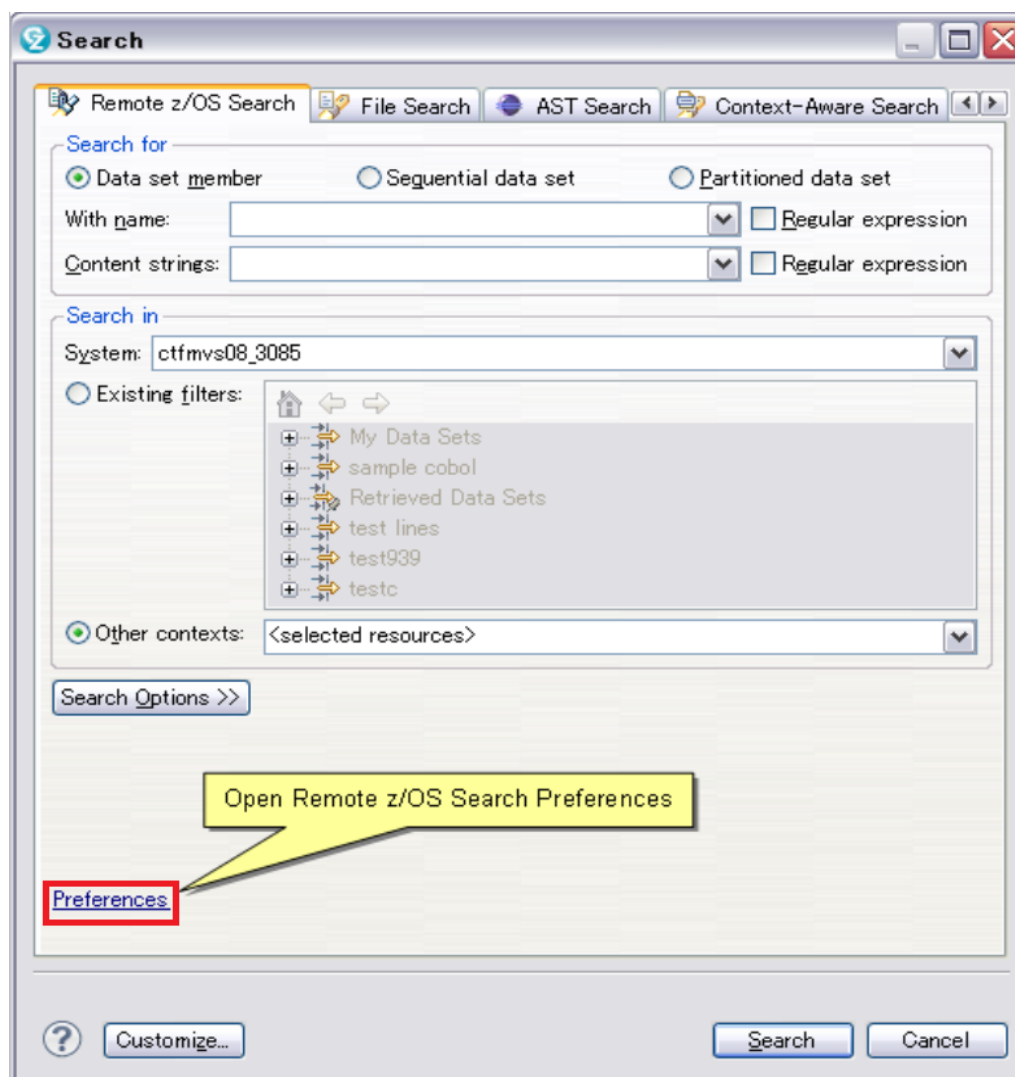
- ‘Expand All’ and ‘Collapse All’ toolbar button
- Capability to customize table columns – order, show/hide, width



The screenshot shows the 'Remote z/OS' window with a search result table. The table has columns for Name, Extension, Size, Transfer, Host C..., Local..., Date, and Last Modified. Two callouts point to the 'Expand All' (+) and 'Collapse All' (-) buttons in the toolbar. A third callout points to the 'Customize Table' dialog box, which is open in the foreground. The dialog has two panes: 'Available contents' and 'Displayed contents'. The 'Available contents' list includes items like 'Local Bidi Format', 'Lock owner', 'VOLUME', 'RECFM', 'LRECL', 'DSORG', 'BLKSIZE', 'PRIMARY', 'SECONDARY', 'SPACE UNITS', 'EXTENTS', 'DSNTYPE', 'Name', and 'Overrides'. The 'Displayed contents' list includes 'Extension', 'Size', 'Transfer', 'Host Code Page', 'Local Code Page', 'Created', 'Last Modified', and 'Last modified by'. The 'Local Code Page' item is selected in the 'Displayed contents' list. An orange callout box points to the 'Customize Table' dialog and the 'View' menu, stating: '‘Customize Table’ and ‘Preferences’ shortcut added to View Menu'. At the bottom left, the text '10 Complete yo' is partially visible.

## z/OS Search Dialog Usability – 8.5

- Easy access to “z/OS Search Preferences” within the dialog



# Rational Developer for System z Roadmap Themes

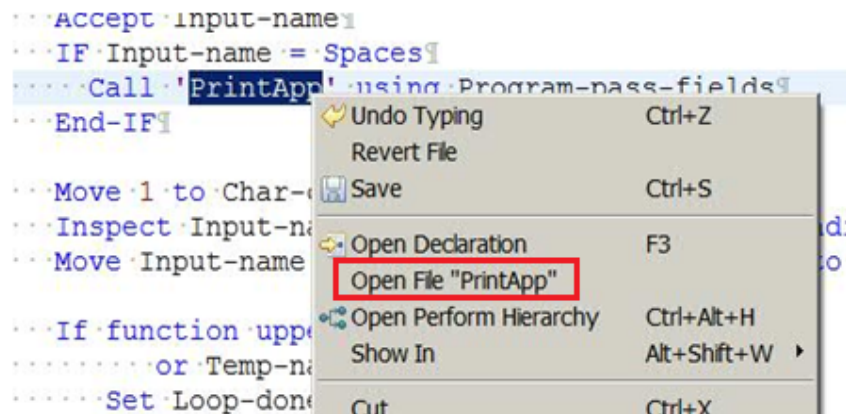


- Performance and Scalability
  - Productivity
  - Languages
  - Integration
  - Advancing technologies
- *Editor Enhancements*
  - Easy and quicker access to what you need

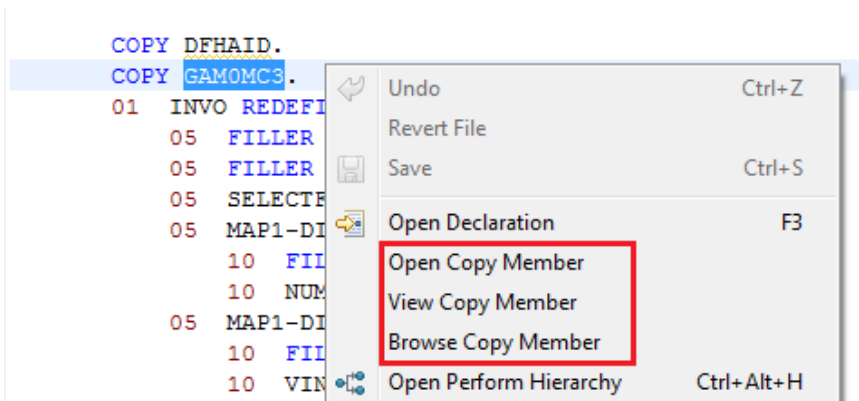
# New features for System z LPEX, COBOL, and PL/I Editors



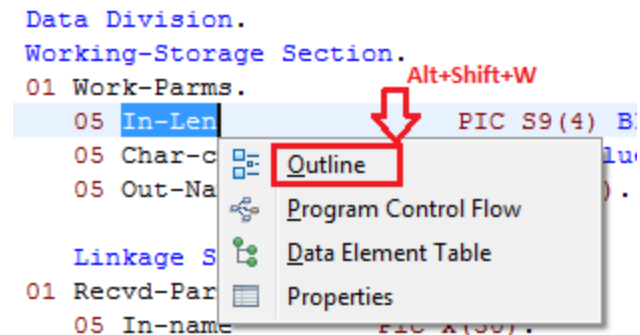
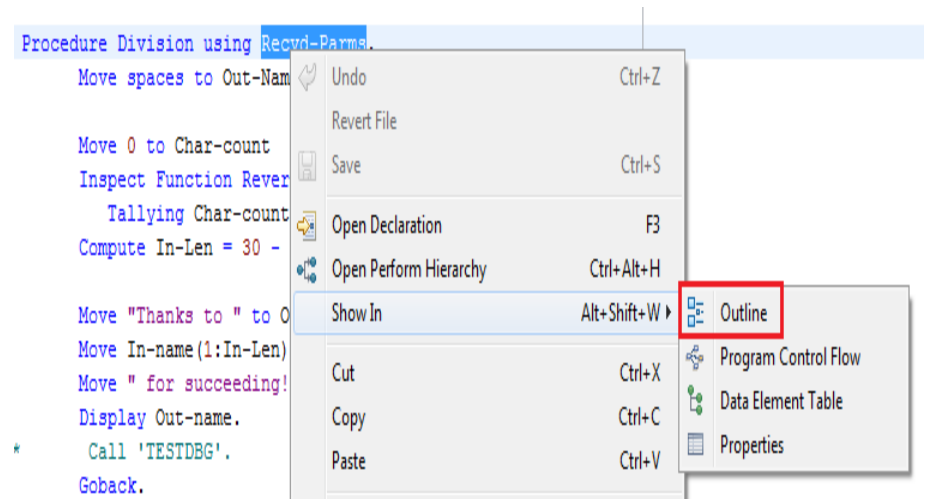
- Provide "Open Called Program" action



- Hyper linking support for Open/ Browse/ View copybooks/include files



- Show In > Outline action to COBOL and PL/I Editor



# New features for COBOL, and PL/I Editors

- Mark “Write occurrences” capability to the supported EXEC statements (EXEC CICS, EXEC DLI, EXEC SQL)

- Occurrences within EXEC statements known to be “writes” are highlighted with a BROWN background
- All “read” statements will continue to be highlighted with a GREY background

- \* PHRASE and NEWPHRASE are read-only data areas;
- \* ESMREASON and ESMRESP are write

```
EXEC CICS
CHANGE PHRASE (data-area) PHRASELEN (data-value)
        NEWPHRASE (data-area) NEWPHRASELEN (data-value)
        USERID (data-value)
        ESMREASON (data-area) ESMRESP (data-area)
END-EXEC.
```

```
/* INTO is write, LENGTH is read */
```

```
EXEC DLI STATISTICS
USING PCB (expression)
        INTO (areal)
        LENGTH (areal)
        VSAM
        FORMATTED
;
```

```
* INTO :hv1:ind1, :hv2:ind2 are WRITE, :hv3 is READ
```

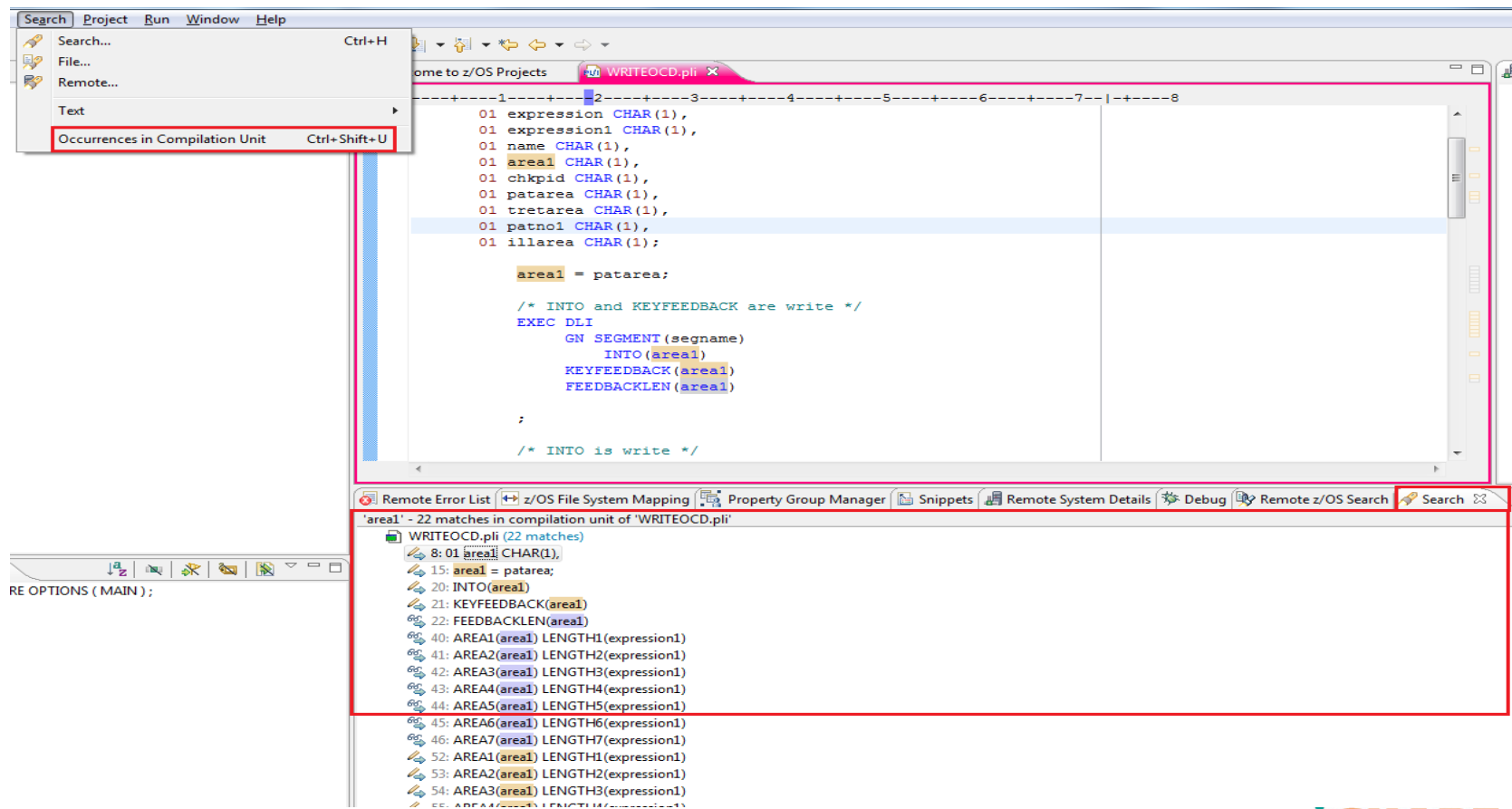
```
EXEC SQL
        FETCH ABSOLUTE :hv3 CURSOR1 INTO :hv1:ind1, :hv2:ind2
END-EXEC.
```

```
* all READ
```

```
EXEC SQL
FREE LOCATOR :hv1, :HV2, :HV3
END-EXEC.
```

## Search for Occurrences Action – RDz 8.5

- Once a variable is selected the user triggers the “Find Occurrences” action using the Menu under search or keyboard shortcut “Ctrl+Shift+U”
- The occurrences are shown in the “Search results” page



The screenshot displays the RDz 8.5 IDE interface. The main editor window shows the source code of 'WRITEOCD.pli'. A search menu is open, highlighting the 'Occurrences in Compilation Unit' option with the keyboard shortcut 'Ctrl+Shift+U'. The search results pane at the bottom of the IDE shows the following matches for the variable 'areal':

```

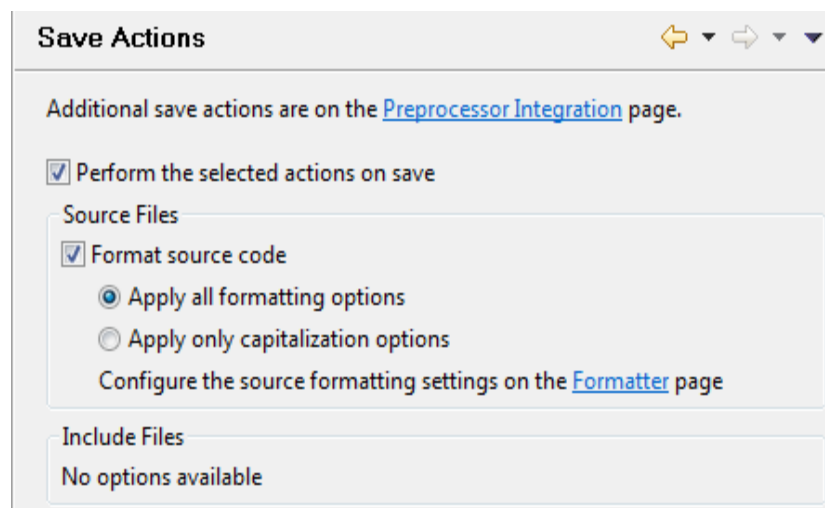
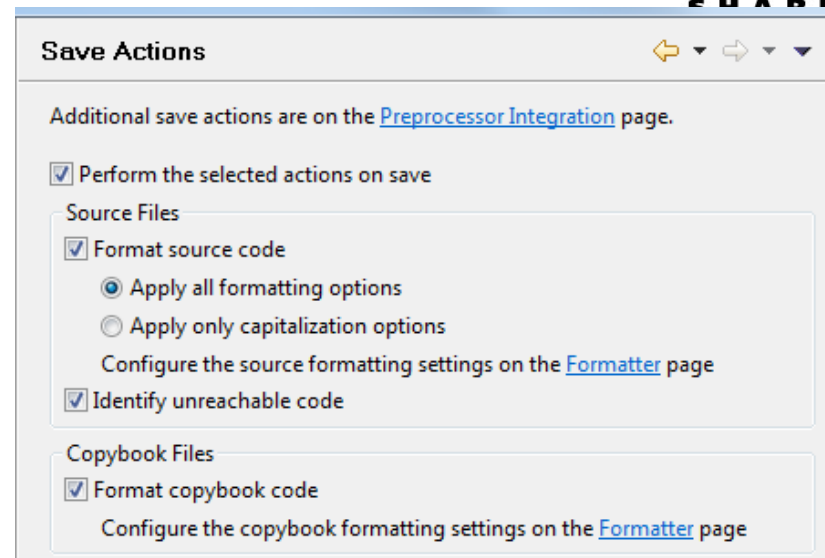
WRITEOCD.pli (22 matches)
8: 01 areal CHAR(1),
15: areal = patarea;
20: INTO(areal)
21: KEYFEEDBACK(areal)
22: FEEDBACKLEN(areal)
40: AREA1(areal) LENGTH1(expression1)
41: AREA2(areal) LENGTH2(expression1)
42: AREA3(areal) LENGTH3(expression1)
43: AREA4(areal) LENGTH4(expression1)
44: AREA5(areal) LENGTH5(expression1)
45: AREA6(areal) LENGTH6(expression1)
46: AREA7(areal) LENGTH7(expression1)
52: AREA1(areal) LENGTH1(expression1)
53: AREA2(areal) LENGTH2(expression1)
54: AREA3(areal) LENGTH3(expression1)

```



## “Save Actions” in COBOL and PL/I Editors - 8.5

- Enable actions before and after a file is saved
- A new preference page added under the **COBOL** (*top*) and **PL/I** (*bottom*) Editor preference category called “Save Actions”
- Source files have an option to Format source code, plus a sub-option to only apply the capitalization
- When an editor is saved, the save actions will be run in this order:
  - **Formatting**
  - **Save the file**
  - **Identify Preprocessor Statements**
  - **Identify Unreachable Code**
- **PL/I Save Actions will not have**
  - **Identify Unreachable Code**
  - **Formatting of Include Files**





# New features for COBOL, and PL/I Editors - 8.5

- Improve Control Statement Visualization and Navigation

Some examples of Annotations provided

## COBOL Editor

PERFORM... END PERFORM

```
PERFORM VARYING UnitsCnt FROM 0 BY 1 UNTIL
    MOVE HundredsCnt TO PrnHunds
    MOVE TensCnt     TO PrnTens
END-PERFORM
```

UNSTRING ... ON OVERFLOW

UNSTRING ... NOT ON OVERFLOW

```
UNSTRING AAAAA delimited by all 'a' into
    AAAAA
ON OVERFLOW
    DISPLAY
    AAAAA "TRAITEMENT DE LA DATA APPLICATIVE PC"
NOT ON OVERFLOW
    DISPLAY
    AAAAA "TRAITEMENT DE LA DATA APPLICATIVE PC"
END-UNSTRING
```

## PL/I Editor

BEGIN BLOCKS

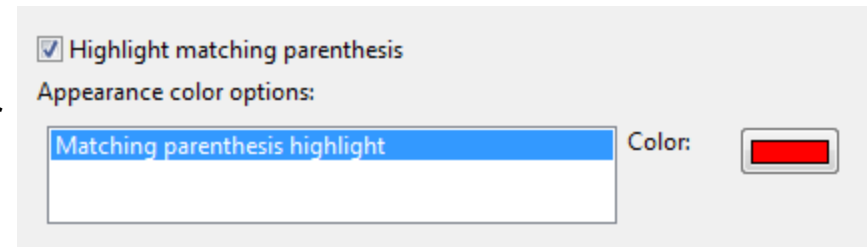
```
begin;
    display( 'tower size must be numeric' );
    goto recover;
end;
```

IF STATEMENTS

```
if run_Again = 'y' | run_Again = 'Y' then
    goto recover;
else;
```

# New features for COBOL, and PL/I Editors

- Improve Control Statement Visualization and Navigation
  - Showing matching Parenthesis and Brackets
  - New preference pages will be added under **COBOL** and **PL/I > Editor** categories
  - Add preference to set the color used in the matching parenthesis annotation
  - Placing the caret to the right of an open or closed parenthesis will annotate the matching parenthesis



```
num2 = MOD( num2 , num3 );
do while( eof = '0'b );
```

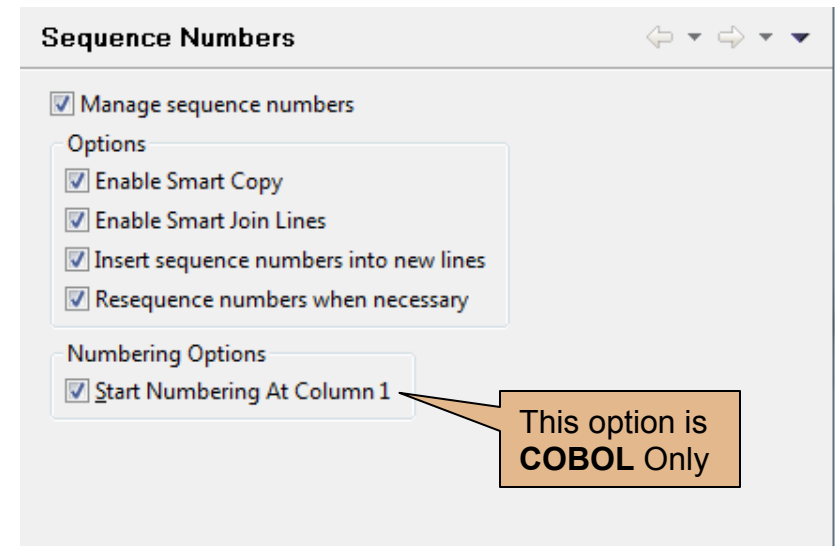
```
if theAnswer = 'q' | theAnswer = 'Q'
then Leave;
else Display( 'Hello ' | theAnswer );
```

# New features for COBOL, and PL/I Editors - 8.5

- Sequence Number Support

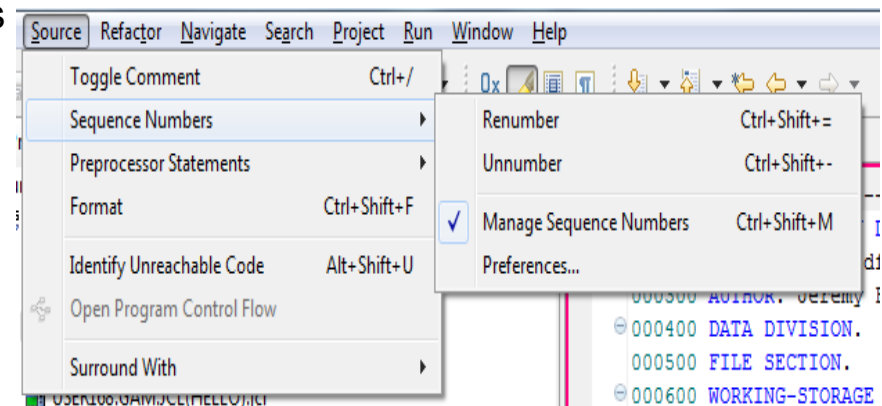
New preference pages will be added under **COBOL** and **PL/I** Editor categories called “**Sequence Numbers**”.

- Manage sequence numbers: Enables sequence numbers support and other “Options”
- Start Numbering at column 1 (available for COBOL only). Affects the Renumber and Unnumber actions



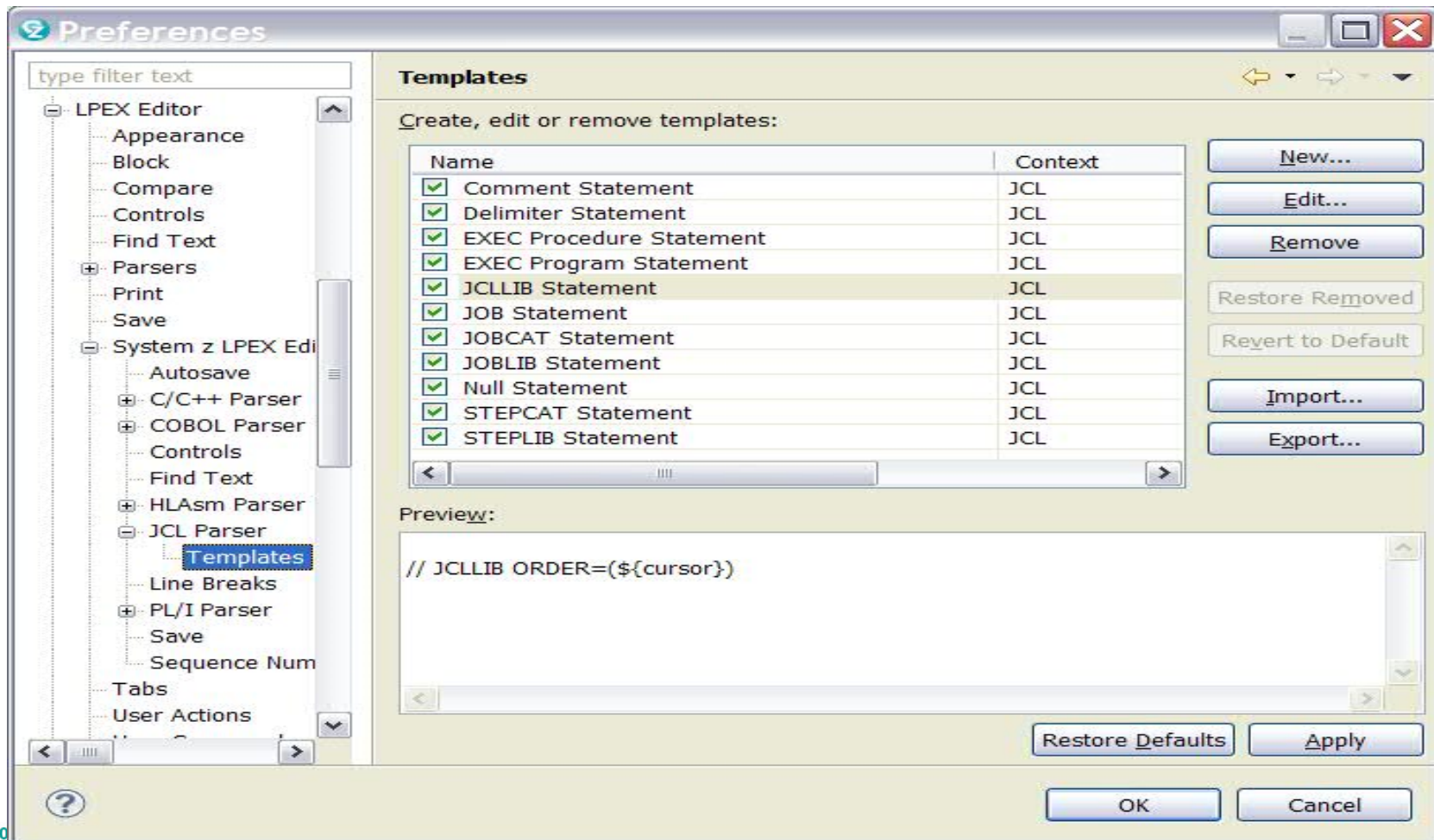
Sequence Number submenu has been updated

- An action “Manage Sequence Numbers” toggles the preference, and has a key binding for quick access.
- Action that jumps to the Sequence Number preference



# JCL Template Support – RDz 8.5

- › Templates are provided for standard JCL statements, and users can create their own Templates
- › When editing .jcl file using “Ctrl+Space” in the editor will trigger a pop list allowing the user to select the template to insert into the editor contents



# Bringing PL/I Capabilities up to par with COBOL in System z LPEX Editor and PL/I Editor – RDz 8.5



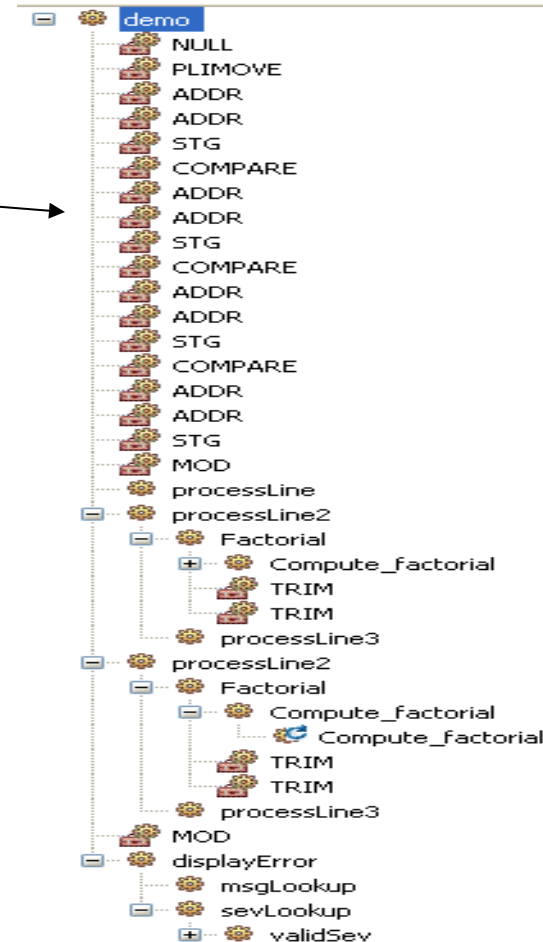
## Adding support for

LPEX & PL/I

- **Call Hierarchy (perform hierarchy equivalent of PLI )**
- Improve PL/I Outline View
- Implement Go to Next/Previous Element
- PL/I rename

PL/I only

- ▶ Write occurrences for PL/I
- ▶ Indentation formatting

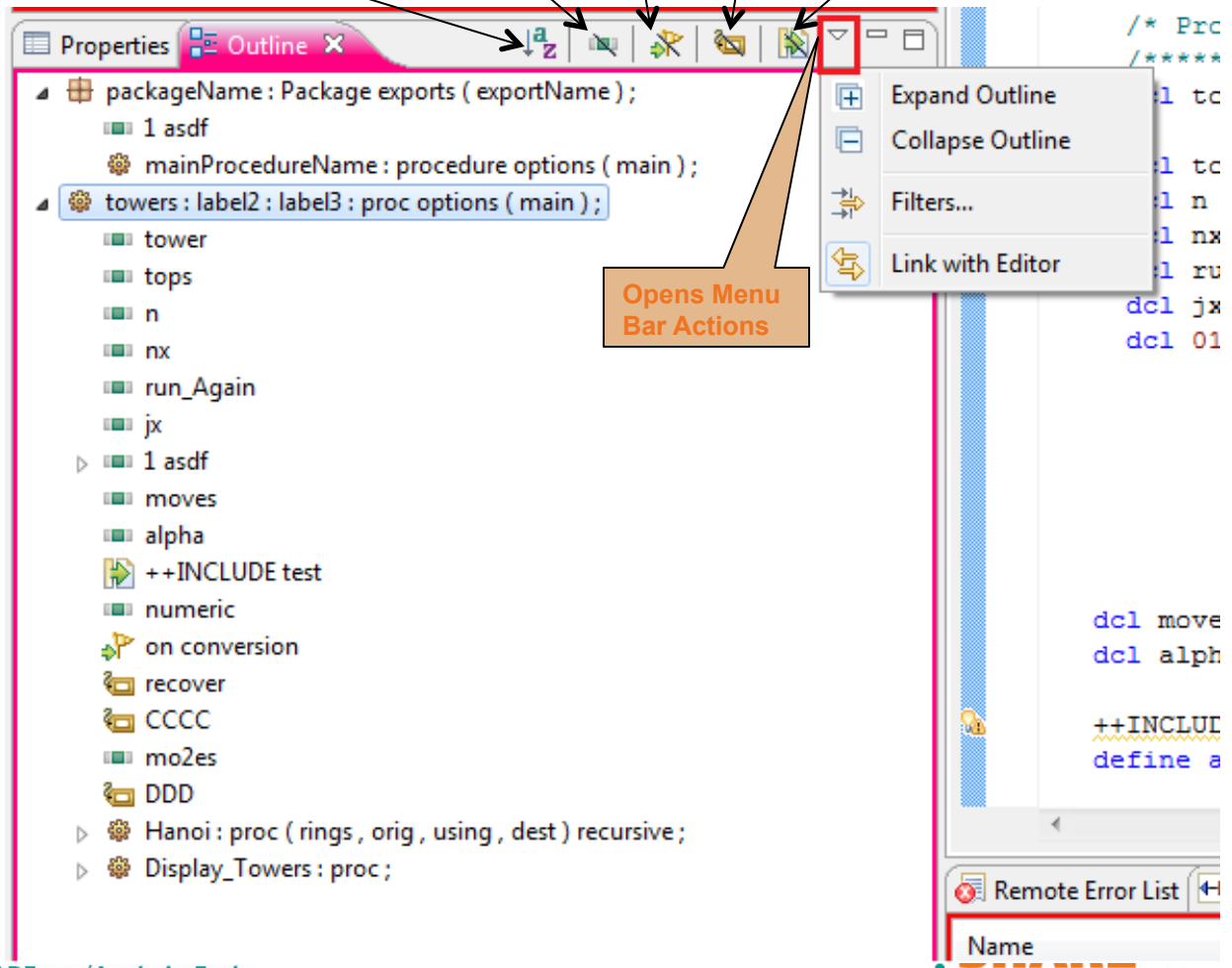


# Improve PL/I Outline View - 8.5

- Items displayed on the PL/I outline view
- Packages
- Procedures
- Data Items
- Labels
- On Units
- Include statements

**Tool bar actions**

Sorting
Hide Data Elements
Hide on-units
Hide labels
Hide Includes



Opens Menu Bar Actions

```

packageName : Package exports ( exportName );
  1 asdf
  mainProcedureName : procedure options ( main );
  towers : label2 : label3 : proc options ( main );
    tower
    tops
    n
    nx
    run_Again
    jx
    1 asdf
    moves
    alpha
    ++INCLUDE test
    numeric
    on conversion
    recover
    CCCC
    mo2es
    DDD
    Hanoi : proc ( rings , orig , using , dest ) recursive ;
    Display_Towers : proc ;
  
```

```

/* Proc
/*****
1 tc
1 tc
1 n
1 nx
1 ru
dcl jx
dcl 01

dcl move
dcl alph

++INCLU
define a
  
```

Remote Error List

Name

# Write Occurrences for PL/I - 8.5

- Show the occurrences of writes and reads of variables in a PL/I program

## In the Editor

- User highlights a variable in the PL/I editor, pressing the “**Toggle Mark Occurrences**” button on the tool bar to toggles Mark Occurrences on.

## In the Search View

- Upon selecting “Occurrences in Compilation Unit” Search view is filled with lines containing reads and writes for a selected PL/I variable

The screenshot displays the PL/I editor interface. The top toolbar contains a button labeled "WRITES are Gold" (highlighted in yellow) and another labeled "READS are Grey" (highlighted in yellow). The editor window shows the following code:

```
if asdf.gewr.bbb > 1 then
  call hanoi( rings-3, orig, dest, using );
else:
  tops(dest) = tops(dest) - 1;
  tower(dest,tops(dest)) = tower(orig,tops(orig));
  tower(orig,tops(orig)) = '';
  tops(orig) = tops(orig) + 1;

moves = moves + 1;
call display_Towers;

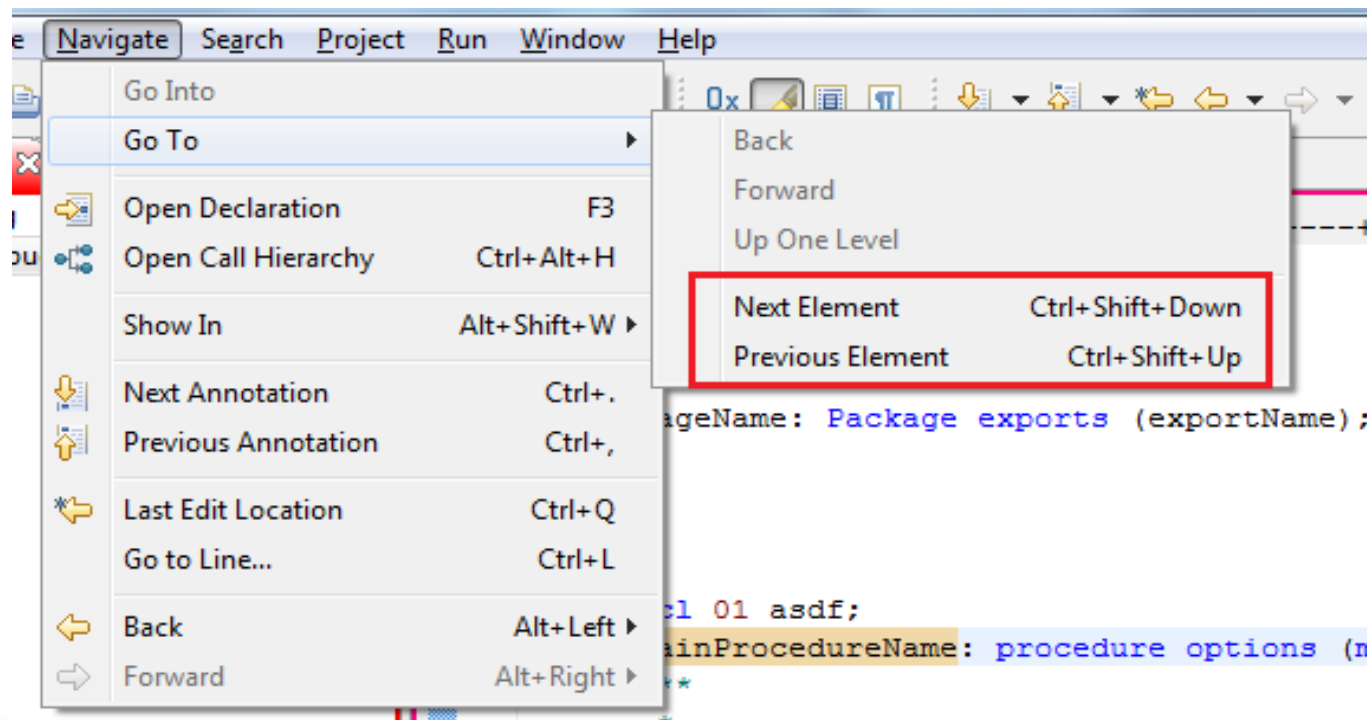
if rings > 1 then
  call Hanoi(
```

The search view at the bottom shows 16 matches for the variable 'tops' in the file 'TOWERS.pli'. The matches are listed as follows:

- 56: dcl tops(3) fixed bin(31);
- 103: tops(1) = 1;
- 107: tops(2) = hbound(tower,2) + 1;
- 113: tops(2) = hbound(tower,2) + 1;
- 117: tops(2) = hbound(tower,2) + 1;
- 121: tops(2) = hbound(tower,2) + 1;
- 125: tops(2) = hbound(tower,2) + 1;
- 128: tops(2) = hbound(tower,2) + 1;
- 135: tops(3) = hbound(tower,2) + 1;
- 184: tops(dest) = tops(dest) - 1;
- 184: tops(dest) = tops(dest) - 1;
- 185: tower(dest,tops(dest)) = tower(orig,tops(orig));
- 185: tower(dest,tops(dest)) = tower(orig,tops(orig));
- 186: tower(orig,tops(orig)) = '';
- 187: tops(orig) = tops(orig) + 1;
- 187: tops(orig) = tops(orig) + 1;

# Implement Go to Next/Previous Element – PL/I

- **Go To Next Element:** Moves the caret to the next element of the program
- **Go To Previous Element:** Moves the caret to the previous element of the program
- Bound to Ctrl+Shift+Down and Ctrl+Shift+Up respectively (Can be changed in Preferences > Keys)

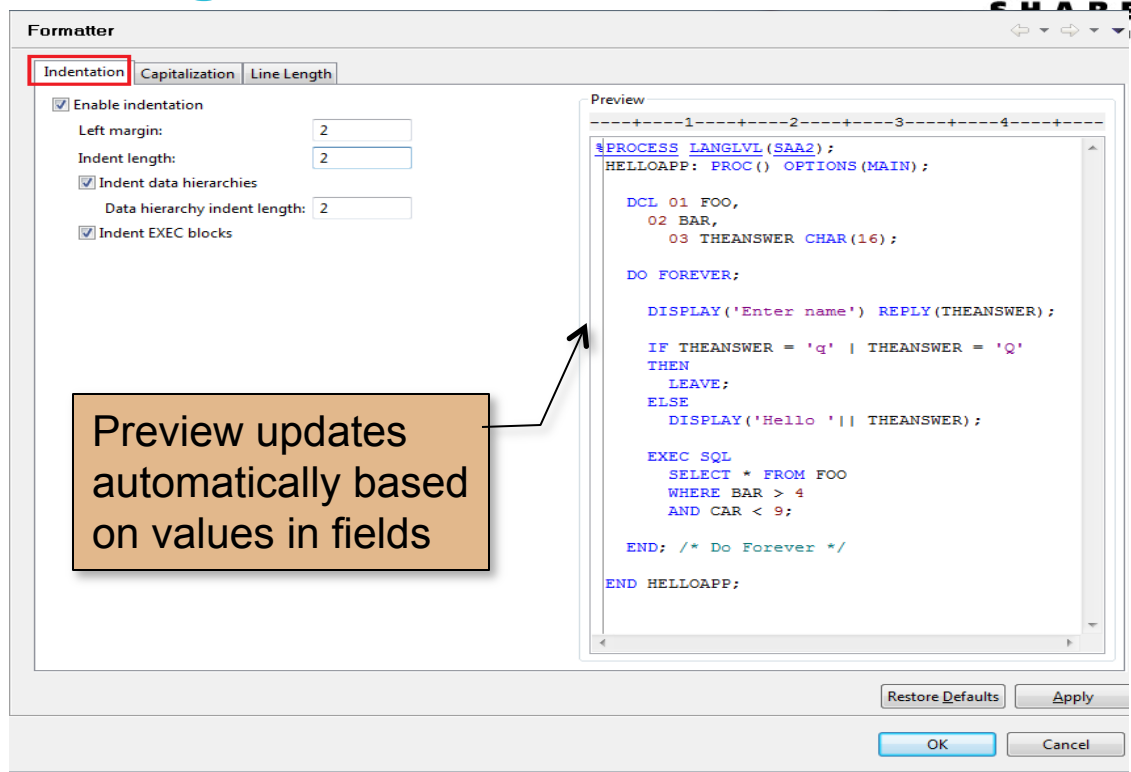




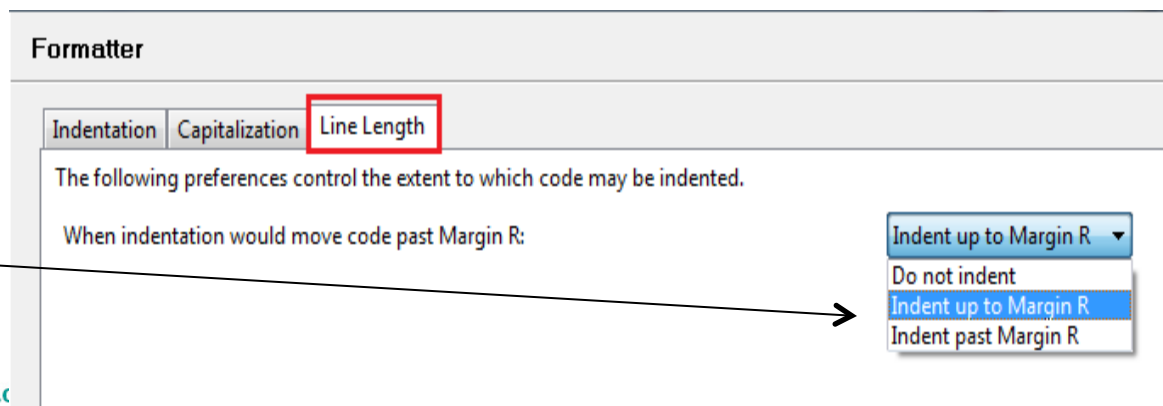
# PL/I Indentation Formatting

PL/I > Editor > Formatter preference page

**Indentation** - "Enable indentation" check box will enable/disable the alteration of indentation of PL/I formatter. It is enabled by default



**Line Length** - 3 behaviors the Added that user may select for when indenting a line moves the code past Margin R



# PL/I Rename Refactoring

- Adding context menu items for Refactor and Rename in PL/I Editor

```

using,
dest ) recursive;

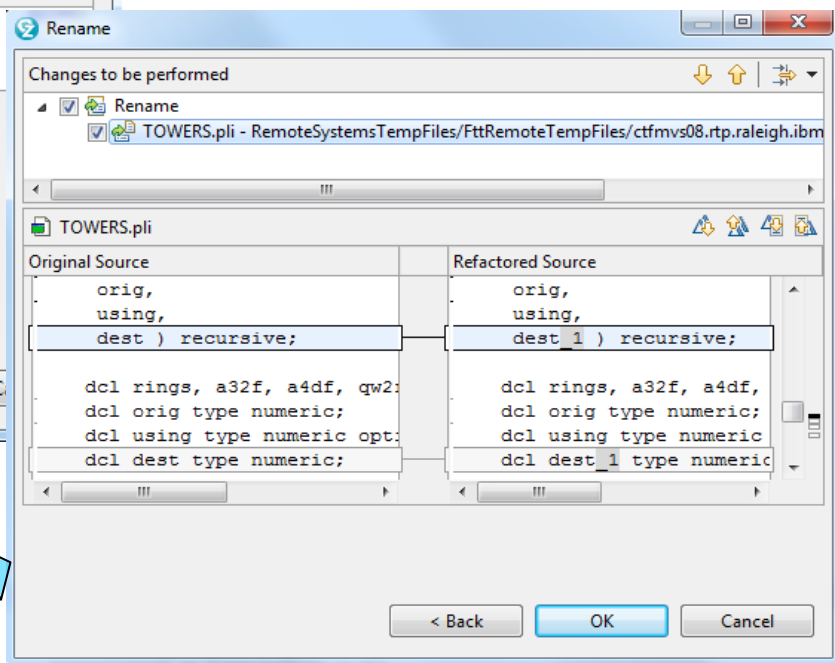
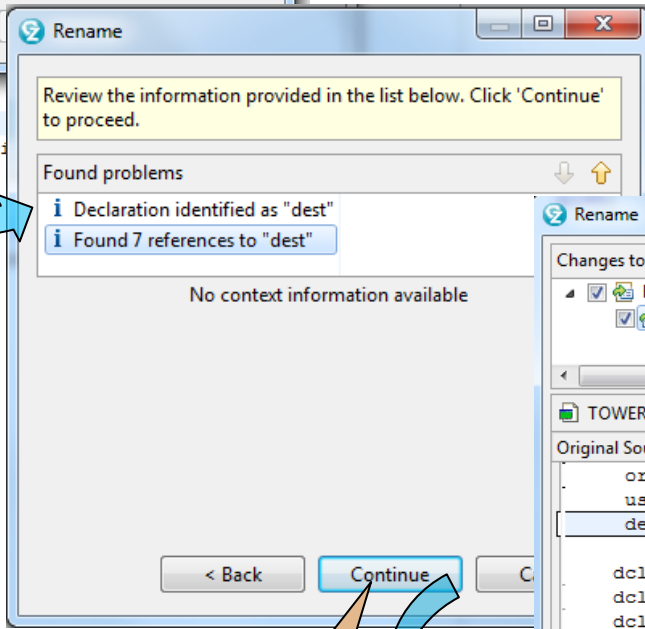
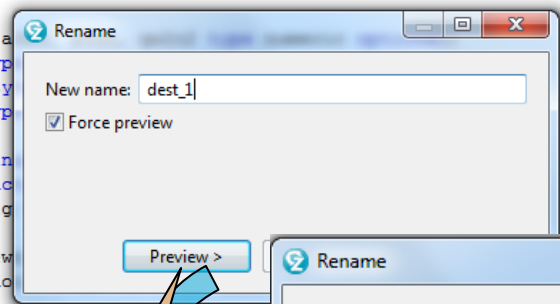
dcl rings, a
dcl orig typ
dcl using ty
dcl dest typ

define ordin
define struc
    02 orig

if asdf.qew
    call Hand
else;

tops(dest) = tops(dest) - 1;
tower(dest, tops(dest)) = tower(orig

```

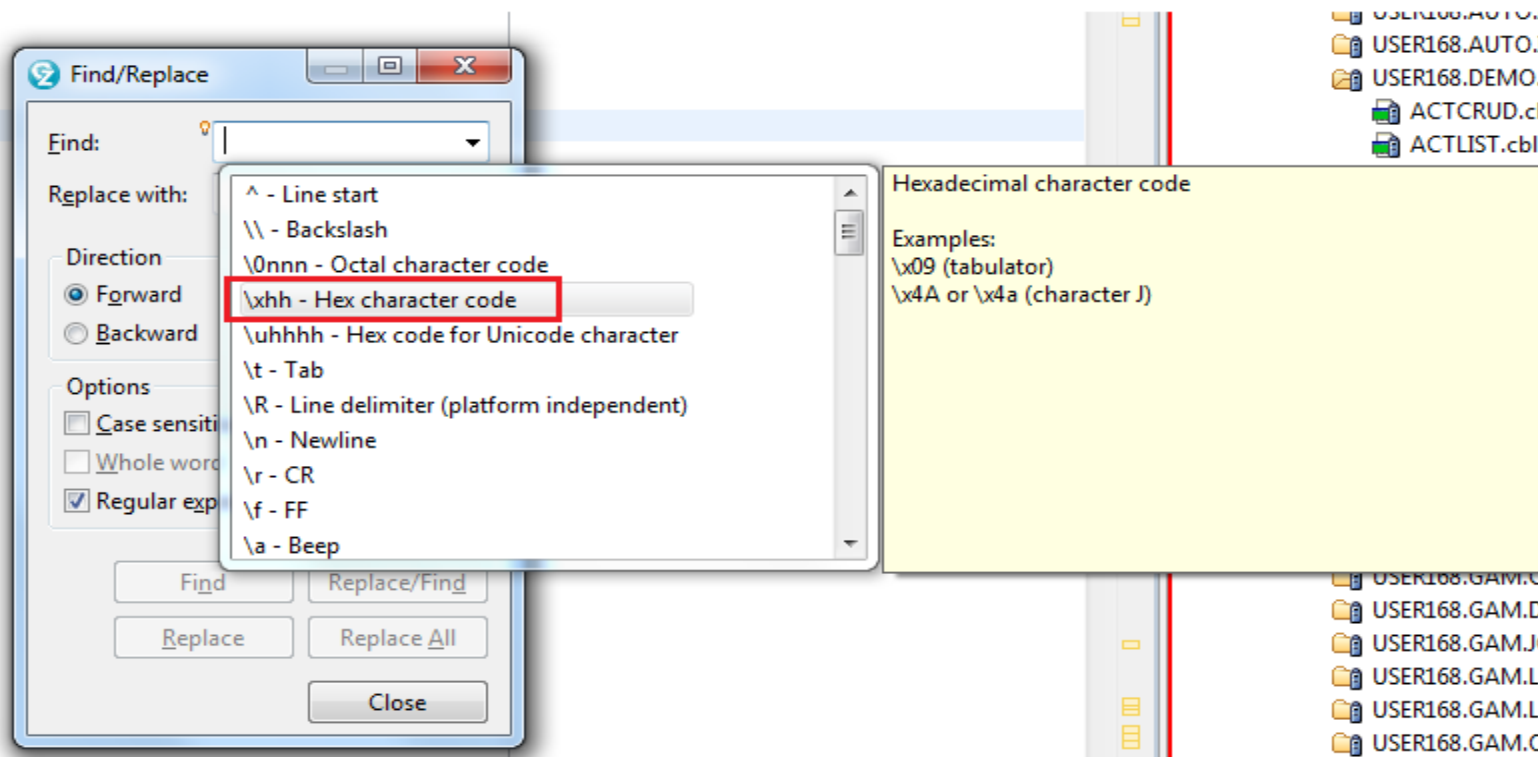


Preview opens  
Rename problems  
view

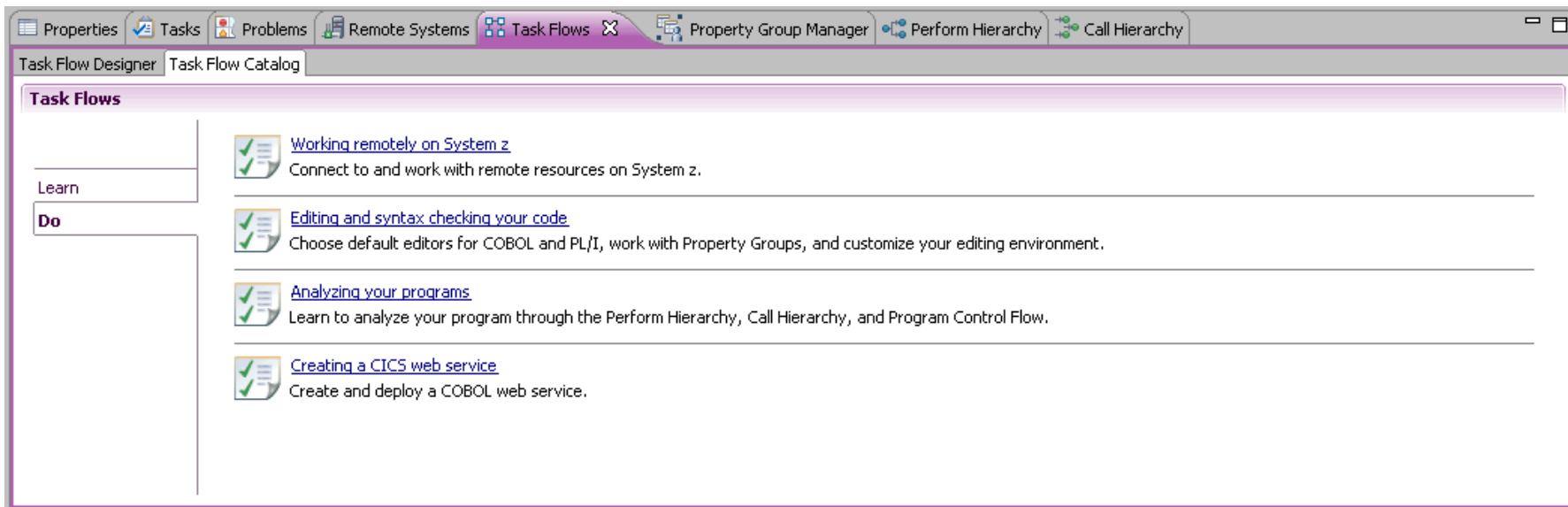
Continue opens  
Compare Editor  
view

## Find for HEX Characters in COBOL & PL/I Editors

- Find text using the hexadecimal values based on the file's remote codepage
- Override the behavior of the existing escape sequence “\xhh” to always be interpreted as being based on the remote codepage values.
- This behavior will work on both the Find text field and the Replace text field.



# Misc Usability Enhancement ... Simplify out-of-box learning



- Providing “cheat sheets” to guide new users in learning and exploring core RDz functions
  - 4 “Do” scenarios to guide users through tasks like data set access, edit, program analysis, and web service generation
  - Additional task-based scenario documentation under “Learn”

# Cheat sheet in action



**New Filter**

Filter

Create a new filter

Filter string:

< Back Next > Finish Cancel

**Defining a filter**

**Background**

Every z/OS connection has a default *My Data Sets* filter under *MVS Files*. This filter contains all remote assets that begin with your high-level qualifier. You can create additional filters.

**Instructions**

- 1) You are now looking at the first page of the *New Filter* wizard. To create a filter, enter a valid filter string that begins with a valid high-level qualifier (HLQ). Click **Next**.
- 2) Specify a name for this filter. This name appears in the *Remote Systems* view to represent this filter. Click **Finish**.

**Result**

The new filter you created is listed in the *Remote Systems* view. Expand it to view the matching artifacts.

Next: [Customizing Remote Systems view](#)

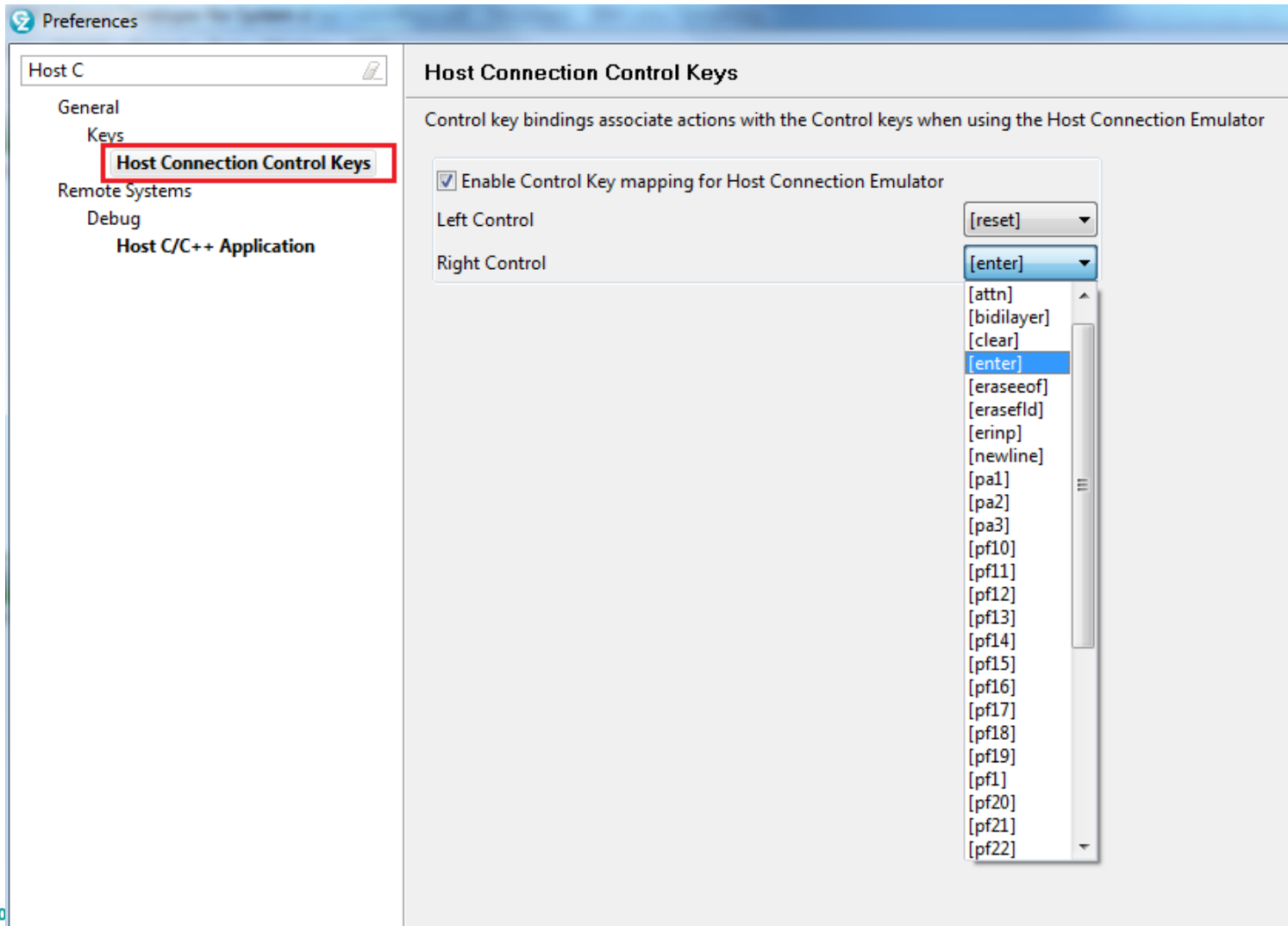
Working remotely on System z: In progress

- Connecting to z/OS
  - Before you begin
  - Connecting to z/OS
- Working with remote assets
  - Retrieving data sets
  - Finding a member
  - Searching on z/OS
  - Defining a filter
- Customizing your experience
  - Customizing Remote Systems view
  - Customizing Remote z/OS Search

1. User clicks on a step to launch the cheatsheet pop-up
2. Cheatsheet guides user to complete that task.

# Mapping Left “Ctrl” key in Emulator

Added a Supplemental page within RDz “Preferences” for mapping Control Keys



# Rational Developer for System z Roadmap Themes



- Performance and Scalability

- Productivity

- Languages

- *COBOL and PL/I*
- *BMS/MFS*
- *System z Data Editor - QSAM*

- Integration

- Advancing technologies

# Enable Find/Replace – System z Data Editor - 8.5



- Provided for ASCII or HEX string search
- Found word will be highlight with light gray background color
- When invalid values are entered, error messages will be shown in the bottom of the new dialog

Field	Data
26130AKleiner,	Rick



# Rational Developer for System z Roadmap Themes

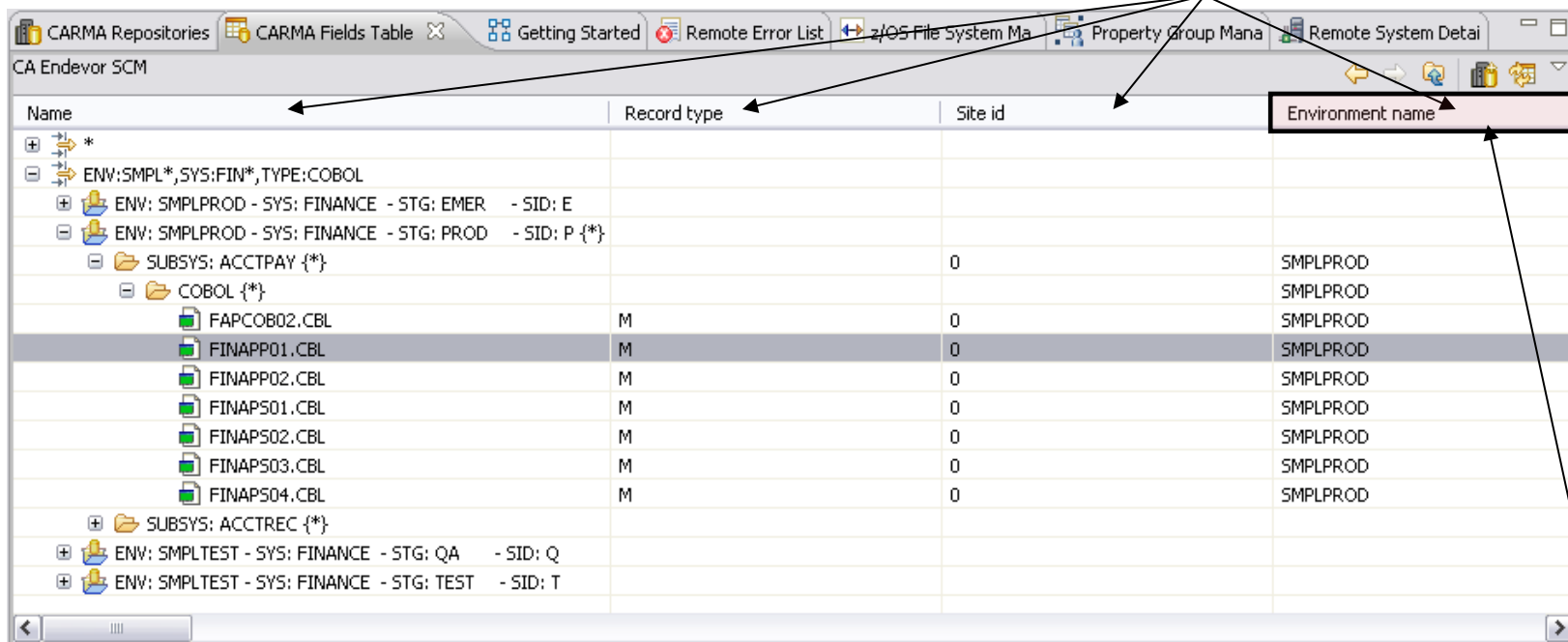


- Performance and Scalability
- Productivity
- Languages
- Integration
  - Endeavor
  - Code Coverage
  - Code Rules
  - RTC EE
  - DB2/IMS/CICS
  - Data Studio
- Advancing technologies

## Endevor Table View - 8.5

- Display your elements or packages in a table

Click on column headers to sort items by attribute.



The screenshot shows the Endevor Table View interface with the following table:

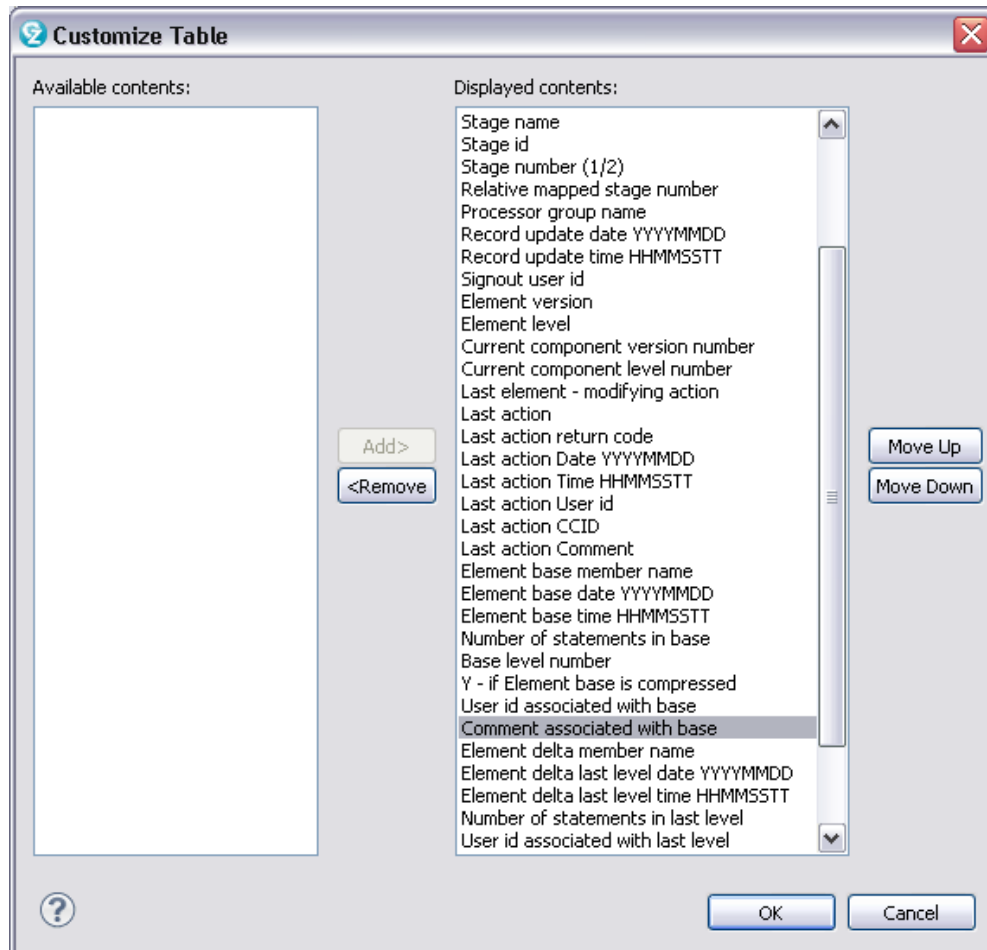
Name	Record type	Site id	Environment name
* (Folder icon)			
ENV: SMPL*,SYS:FIN*,TYPE:COBOL (Folder icon)			
ENV: SMPLPROD - SYS: FINANCE - STG: EMER - SID: E (Folder icon)			
ENV: SMPLPROD - SYS: FINANCE - STG: PROD - SID: P {*} (Folder icon)			
SUBSYS: ACCTPAY {*} (Folder icon)		0	SMPLPROD
COBOL {*} (Folder icon)			SMPLPROD
FAPCOB02.CBL (File icon)	M	0	SMPLPROD
FINAPP01.CBL (File icon)	M	0	SMPLPROD
FINAPP02.CBL (File icon)	M	0	SMPLPROD
FINAPS01.CBL (File icon)	M	0	SMPLPROD
FINAPS02.CBL (File icon)	M	0	SMPLPROD
FINAPS03.CBL (File icon)	M	0	SMPLPROD
FINAPS04.CBL (File icon)	M	0	SMPLPROD
SUBSYS: ACCTREC {*} (Folder icon)			
ENV: SMPLTEST - SYS: FINANCE - STG: QA - SID: Q (Folder icon)			
ENV: SMPLTEST - SYS: FINANCE - STG: TEST - SID: T (Folder icon)			

Drag and drop to re-order columns.

## Endevor Table View - 8.5

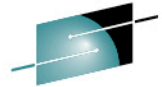
- Display your elements or packages in a table

Customize order and appearance of columns using the Customize Table panel.



Element id	Environment name
	SMPLPROD
	SMPLPROD
	SMPLPROD
	SMPLPROD
	SMPLPROD
	SMPLPROD
	SMPLPROD
	SMPLPROD
	SMPLPROD
	SMPLPROD

# Integration with z/OS Projects - 8.5



- UPDATE element

RDz automatically selects entry stage according to map

Property	Value
Attribute	
BLKSIZE	32720
DSNTYPE	DATA_LIBRA
DSORG	PO
EXTENTS	1
LRECL	80
PRIMARY	301
RECFM	FB
SECONDAR	100
SPACE UNI	BLOCK
VOLUME	CSPL00
Info	
Created	Fri Apr 20 00
Last Modifie	Fri Apr 20 15
Last modifie	USER91

**Replace Element**

Specify the following parameters to perform Replace Element on FINAPP02.

Element name \* FINAPP02  
Environment name \* SMPLTEST  
System name \* FINANCE  
Subsystem name \* ACCTPAY  
Type name \* COBOL  
Change control id DEMO  
Comment DEMO  
New version (1-99)  
Update if present   
Delete input source   
Override signout   
Bypass generate processor   
Processor group name CLENBL  
Submit as batch job

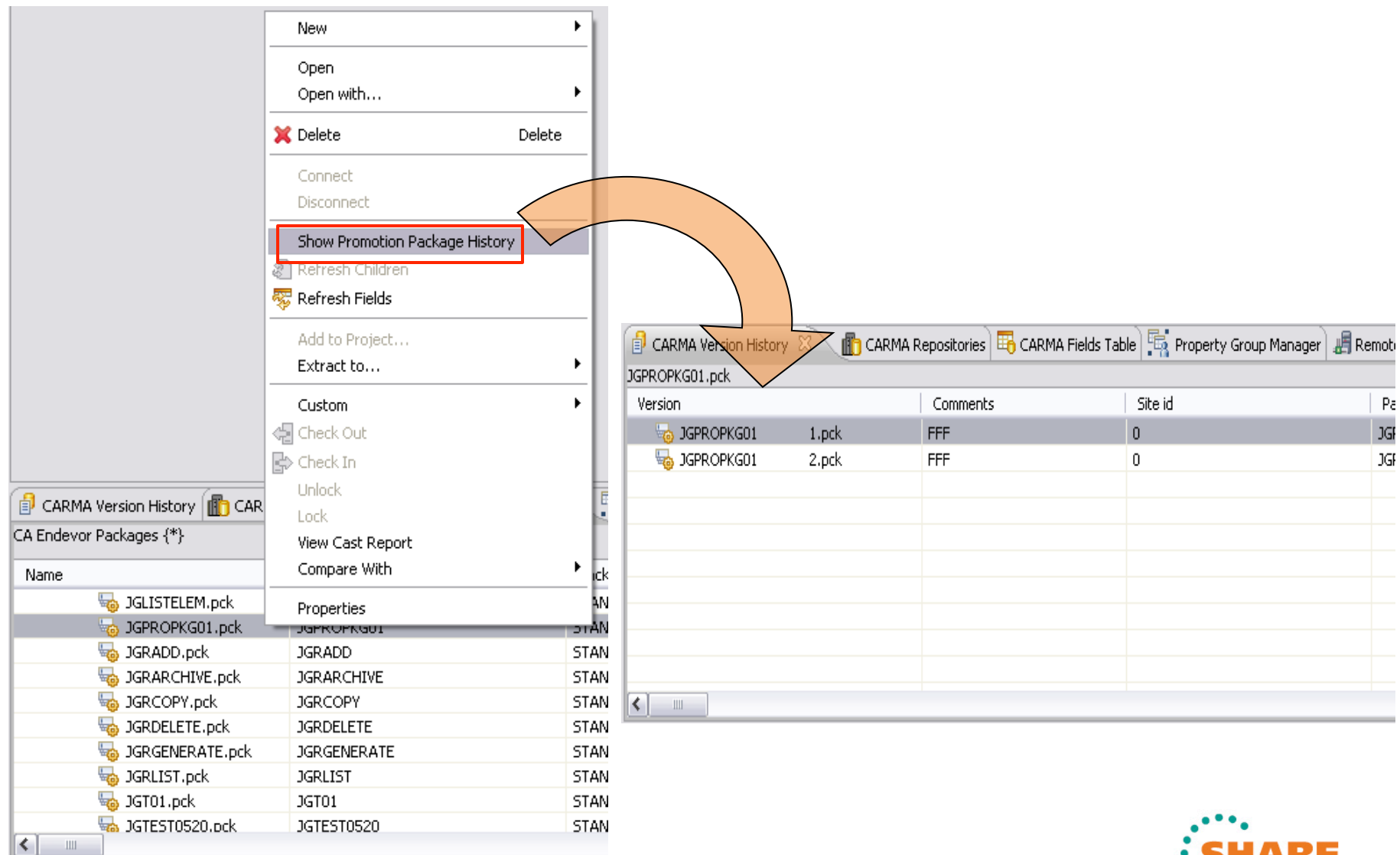
\* - required parameter

Remember the entered values for later  
 Continue to prompt me for the remaining members on this task

OK Cancel

Labels updated

# Show Promotion Package History - 8.5



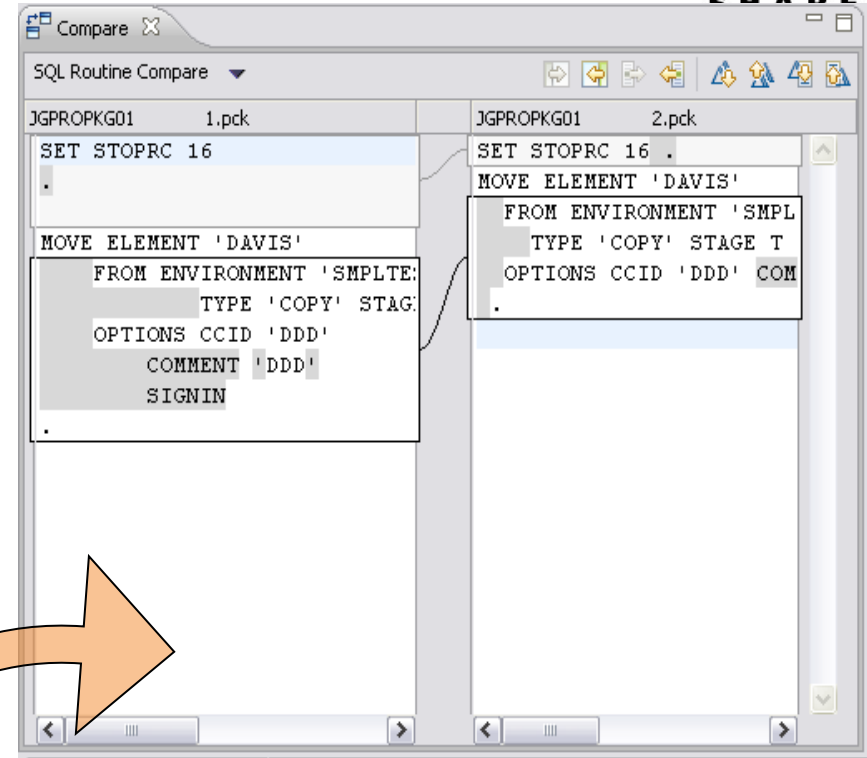
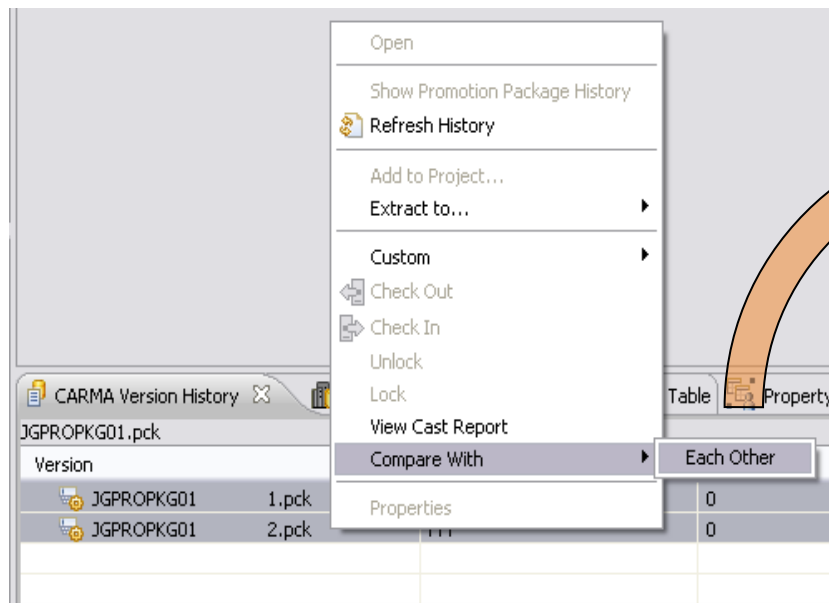
The screenshot shows a software interface with a file explorer on the left and a table on the right. A context menu is open over the file explorer, with the 'Show Promotion Package History' option highlighted in red. An orange arrow points from this menu option to the 'CARMA Version History' window. The table in the window displays the following data:

Version	Comments	Site id	Pe
JGPROPKG01 1.pck	FFF	0	JG
JGPROPKG01 2.pck	FFF	0	JG

# Compare Package History - 8.5

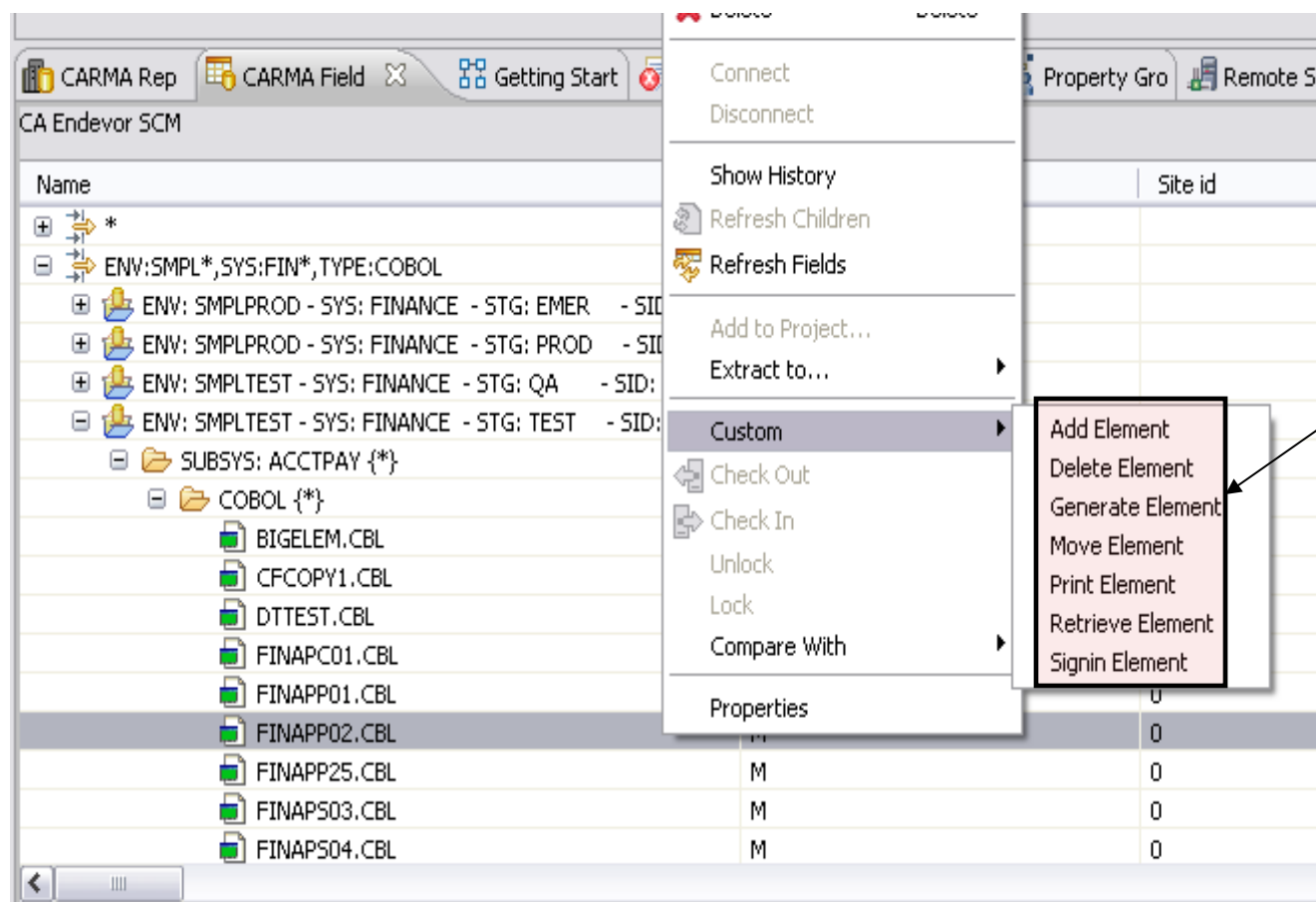


Compare versions in package history



## Element Actions Update - 8.5

- List of element actions in the Custom submenu refined

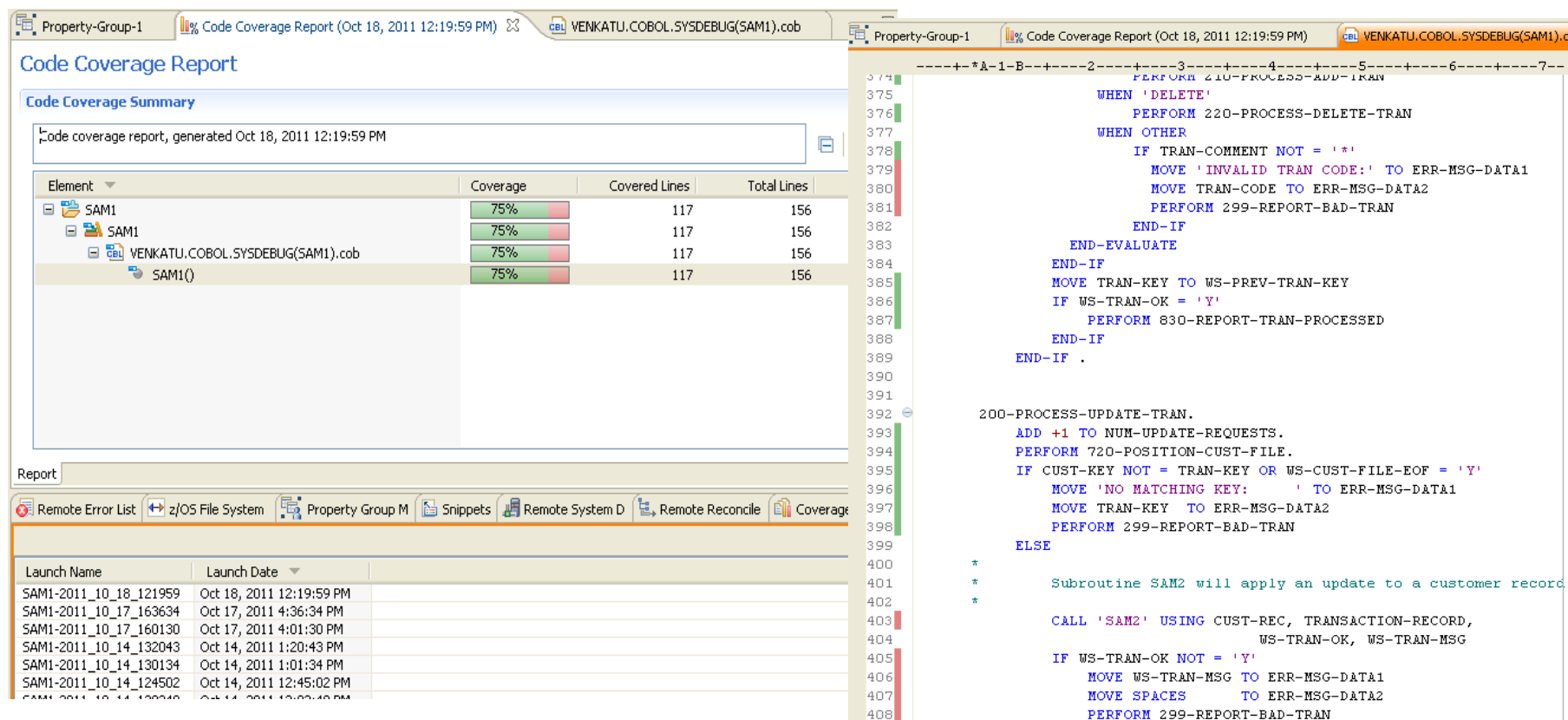


Package actions moved exclusively to Packages repository.

Batch support for all element actions

# Code Coverage

- RDz 803 introduced the ability to generate Code Coverage reports
- From the report, the source view can be launched to see the colored annotations that indicate the coverage (or not )



**Code Coverage Report**

Code coverage report, generated Oct 18, 2011 12:19:59 PM

Element	Coverage	Covered Lines	Total Lines
SAM1	75%	117	156
SAM1	75%	117	156
VENKATU.COBOI.SYSDEBUG(SAM1).cob	75%	117	156
SAM1()	75%	117	156

Report

Launch Name	Launch Date
SAM1-2011_10_18_121959	Oct 18, 2011 12:19:59 PM
SAM1-2011_10_17_163634	Oct 17, 2011 4:36:34 PM
SAM1-2011_10_17_160130	Oct 17, 2011 4:01:30 PM
SAM1-2011_10_14_132043	Oct 14, 2011 1:20:43 PM
SAM1-2011_10_14_130134	Oct 14, 2011 1:01:34 PM
SAM1-2011_10_14_124502	Oct 14, 2011 12:45:02 PM
SAM1-2011_10_14_123338	Oct 14, 2011 12:33:38 PM

```

374 -----+*A-1-B-----2-----3-----4-----5-----6-----7-----|
375         PERFORM 210-PROCESS-ADD-TRAN
376         WHEN 'DELETE'
377         PERFORM 220-PROCESS-DELETE-TRAN
378         WHEN OTHER
379         IF TRAN-COMMENT NOT = '*'
380         MOVE 'INVALID TRAN CODE:' TO ERR-MSG-DATA1
381         MOVE TRAN-CODE TO ERR-MSG-DATA2
382         PERFORM 299-REPORT-BAD-TRAN
383         END-IF
384         END-EVALUATE
385     END-IF
386     MOVE TRAN-KEY TO WS-PREV-TRAN-KEY
387     IF WS-TRAN-OK = 'Y'
388         PERFORM 830-REPORT-TRAN-PROCESSED
389     END-IF
390
391
392
393 200-PROCESS-UPDATE-TRAN.
394     ADD +1 TO NUM-UPDATE-REQUESTS.
395     PERFORM 720-POSITION-CUST-FILE.
396     IF CUST-KEY NOT = TRAN-KEY OR WS-CUST-FILE-EOF = 'Y'
397     MOVE 'NO MATCHING KEY:' TO ERR-MSG-DATA1
398     MOVE TRAN-KEY TO ERR-MSG-DATA2
399     PERFORM 299-REPORT-BAD-TRAN
400     ELSE
401     *
402     *     Subroutine SAM2 will apply an update to a customer record
403     *
404     CALL 'SAM2' USING CUST-REC, TRANSACTION-RECORD,
405         WS-TRAN-OK, WS-TRAN-MSG
406     IF WS-TRAN-OK NOT = 'Y'
407     MOVE WS-TRAN-MSG TO ERR-MSG-DATA1
408     MOVE SPACES TO ERR-MSG-DATA2
409     PERFORM 299-REPORT-BAD-TRAN
410
  
```



# Code Review use cases

Implemented in RDz v8.0.3

1. IDE code review support for COBOL
  - Defining/selecting COBOL rule sets
  - Running analysis and generating reports

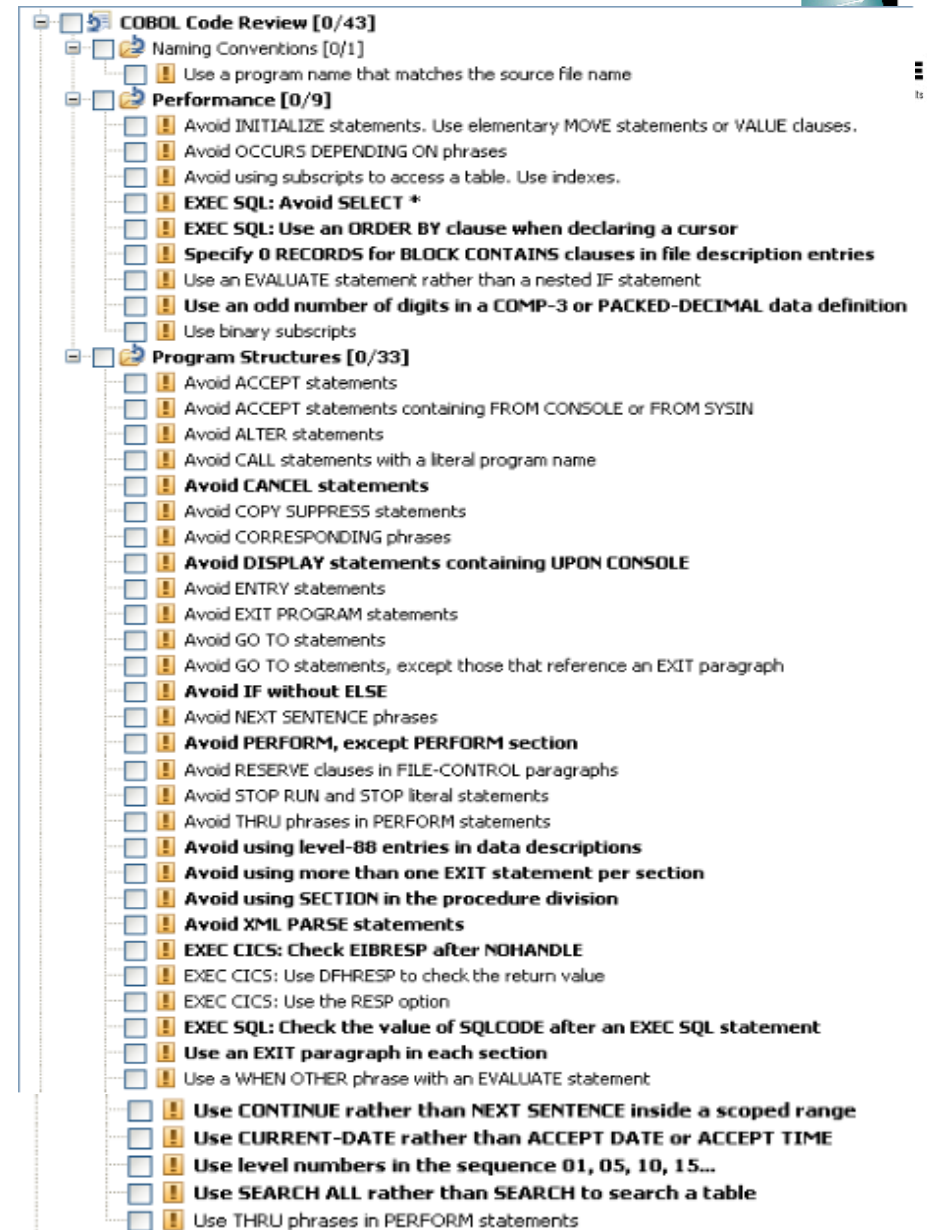
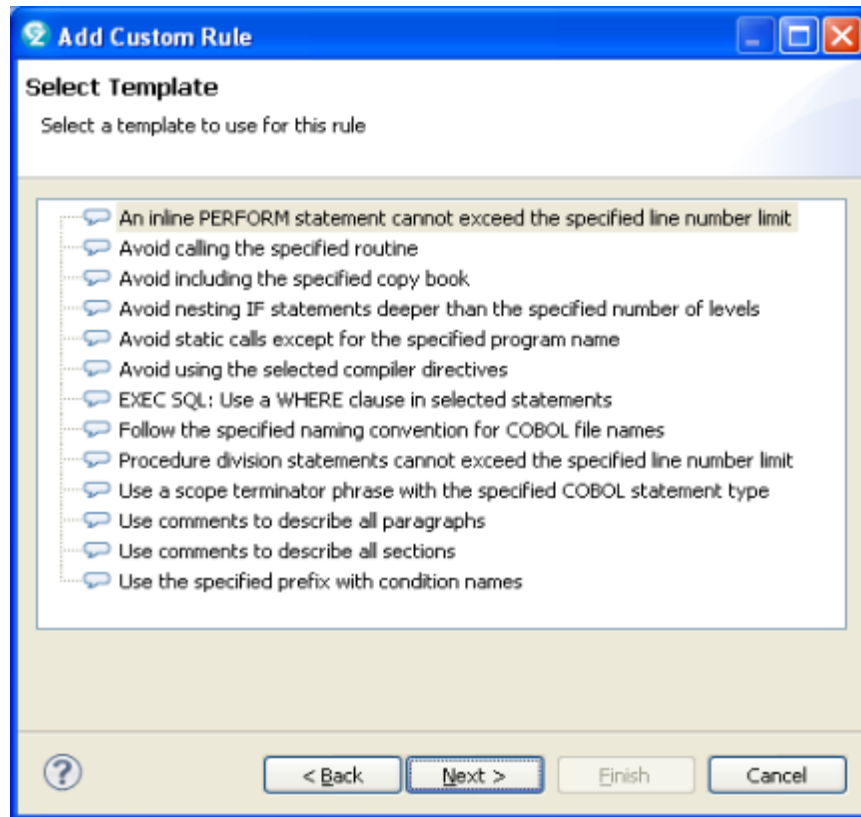
RDz Next enhancements:

- 1a. Additional COBOL rules requested by customers
2. IDE code review support for PL/I – analysis engine and set of rules and templates
3. COBOL Custom rules support

Use cases under consideration for future RDz releases:

- 3a. PL/I Custom rules support
4. Batch support - Run COBOL/PLI code review from batch environment including from SCM's build. Ability to run local or server-side
5. Code Review options – run against source code updates only, run on “file save”, review “real-time”.
6. SCM Process support - Run COBOL/PLI rules analysis as pre-condition to SCM source code delivery
7. Support for “quickfixes”
8. Reports - Analysis data collection and reports via graphs, dashboards, build scripts, etc.

# COBOL rules



# PL/I rules and templates in 8.5



- PL/I Code Review [0/21]
  - Naming Conventions [0/1]
    - Use a main procedure name that matches the source file name
  - Performance [0/10]
    - Avoid assigning or comparing two entry variables having different parameter lists
    - Avoid assigning or comparing two entry variables having different RETURNS attributes
    - Avoid declaring multiple data items by factoring attributes
    - Avoid FIXED DECIMAL declarations with even precision
    - Avoid implicit declarations
    - Declare a single variable in an IF, WHILE, UNTIL, or WHEN clause as BIT(1) NONVARYING
    - EXEC SQL: Avoid SELECT \*
    - EXEC SQL: Use unqualified table names
    - Use a loop control variable that is FIXED BINARY, FLOAT, ORDINAL, HANDLE, POINTER, or OFFSET.
    - Use INITIAL with STATIC data declarations
  - Program Structures [0/10]
    - Avoid assigning a value to an INONLY or NONASSIGNABLE parameter
    - Avoid GO TO statements that specify a label in another block
    - Avoid GO TO statements that specify a label variable
    - Avoid most GO TO statements
    - Avoid STOP statements
    - EXEC CICS: Use DFHRESP to check the return value
    - EXEC CICS: Use the RESP option
    - Initialize locator variables used for based storage
    - Use a SELECT statement rather than ELSE-IF
    - Use dot qualified references to structure members that are not level 1

**Add Custom Rule**

**Select Template**  
Select a template to use for this rule

- Avoid data item names longer than the specified number of characters
- Avoid nesting DO, BEGIN, PROCEDURE, and IF statements deeper than the specified level
- Avoid using the specified include file
- Follow the specified convention for PL/I file names
- Procedure statements cannot exceed the specified line number limit
- Use DO-END within a THEN, ELSE, WHEN, or OTHERWISE unit

# COBOL Custom rules support



- Most likely you will have coding standards not covered in our list of pre-built and custom templates; therefore, you will need to add your own set of custom rules to the selection lists.
- Process to build your own custom COBOL rule:
  - **Use RDz wizard** (Eclipse PDE new plugin project template) to generate plugin for custom COBOL rules
    - Creates the java plugin project
    - Creates new category(s) to hold all your domain specific rules
    - Adds rules to the categories
    - Creates java class templates for each of your custom rules
  - Rule developer **fills in the template** with java code to implement their custom rule
    - Using RDz published **COBOL Application Model API**
  - **Package your plugin** as P2 update site and **install in the RDz** Eclipse environment using Eclipse Software Updater

# [RTC Integration] Remote Include Library Support

## What it does



- Allows local source files to reference include files on remote systems
  - Remote **libraries** are searched after local libraries
  - COBOL and PLI
  - SYSLIB and named libraries
  
- Supported functions
  - Open / browse / view include file
  - Real time syntax check
  
- Non-supported functions
  - Other languages
  - Functions provided by external tools
    - Syntax check
    - Show dependencies
    - Project build
    - Preprocessors

# Remote Include Library Support – What it looks like



The screenshot displays a COBOL development environment with three main components:

- Code Editor:** Shows a COBOL program with the following code:

```
Line 49      Column 32      Insert
-----+*A-1-B--+-----2-----+-----3-----+-----4-----+-----5-----+-----6-----+-----7--|+-----8
Procedure Division.
Copy myfile Of mylib.
Initialize Program-pass-fields
              Program-other-fields
              Program-flags.
Perform until Loop-done
              Display " "
```
- Local COBOL Settings Dialog:** The 'Copy Libraries' tab is active, showing a 'Remote' connection named 'stplex4b-4035'. The table below lists the configured libraries:

Library Name	Library Path
SYSLIB	LONGWEL.COBOL.COPYLIB
MYLIB	LONGWEL.COBOL.MYLIB
- Remote Systems Tree View:** Shows a directory structure for 'stplex4b-4035' containing 'z/OS UNIX Files', 'z/OS UNIX Shells', and 'MVS Files'. Under 'MVS Files', there is a 'Retrieved Data Sets' folder containing 'My Data Sets', which includes several COBOL libraries and source files, including 'LONGWEL.COBOL.COPYLIB' and 'LONGWEL.COBOL.MYLIB'. The file 'MYFILE.cpy' is highlighted.

Red arrows indicate the mapping from the 'Copy Libraries' dialog to the remote system tree view, showing how the local 'Remote' connection is linked to the specific remote system and its file structure.

# Data Studio 3.1.1 – Bundled and Shell Shares with RDz 8.5



# Simplified Debug of application that run in IMS and DB2 – 8.5



- Provide user interface consistent with the way other applications are debugged in eclipse (instead of using GREEN screen) similar to the way the support was added for CICS.
- Load module (or DB2 catalog) not requiring to include IP address of the workstation
  - IBM Debug Tool provides a customized version of the Language Environment (LE) user exit, which returns a TEST runtime option when called by the LE initialization logic.
- Have the ability to manage multiple debug profiles on the client even though server supports one profile.

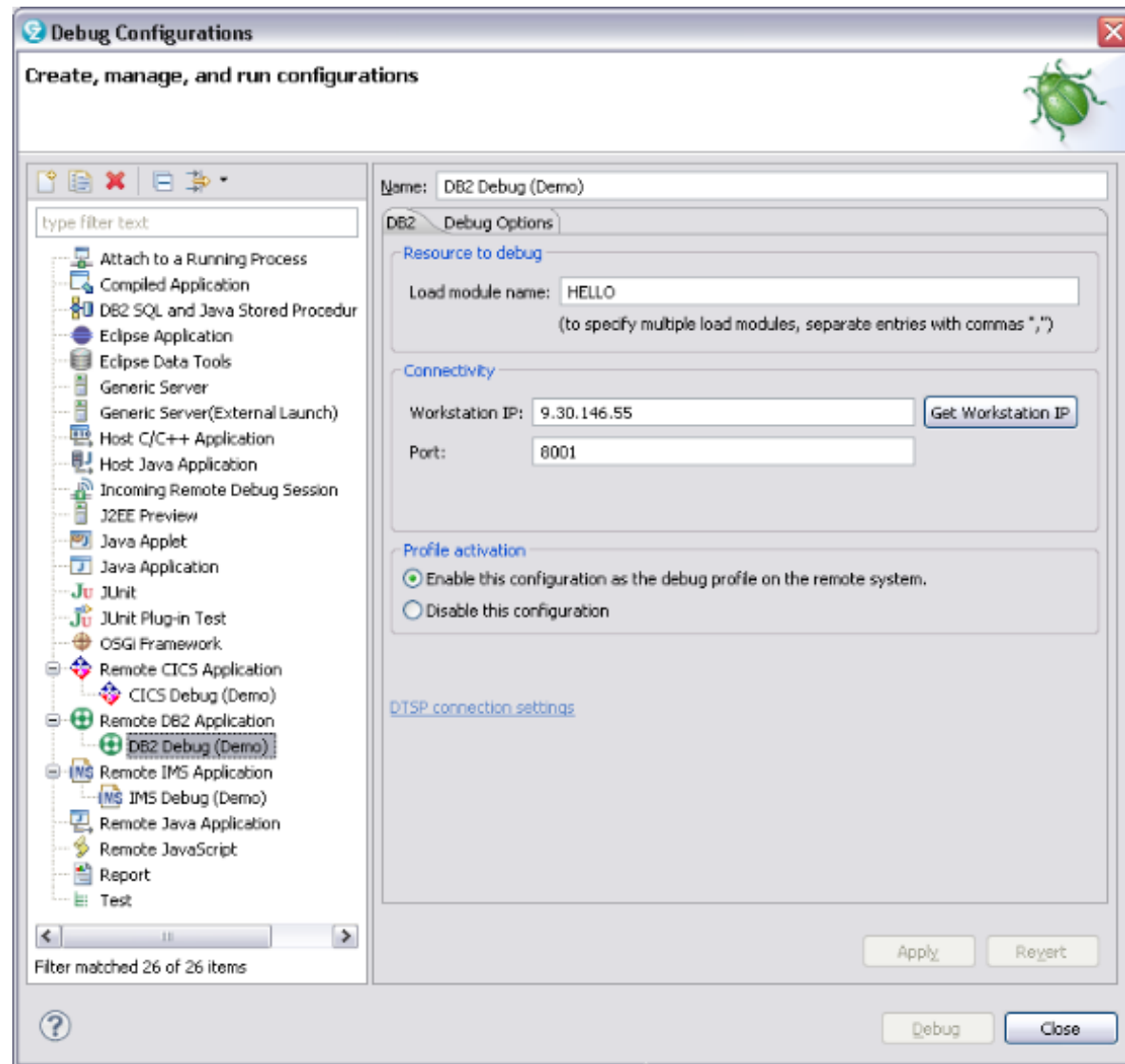
```
----- Debug Tool - Edit TEST Run-time Option Data Set -----
Command ==> _____
Enter test program names: (* is a valid wild card, by itself or as the last
character of a name)
Name 1: _____ Name 2: _____ Name 3: _____ Name 4: _____
Name 5: _____ Name 6: _____ Name 7: _____ Name 8: _____
Enter IMS identifiers: (only valid for IMS user exit)
IMS Subsystem ID: IMS3 IMS Transaction ID: IRAN3
Select Test Options:
Test Option ==> TEST Test/Notest
Test Level ==> ALL All/Error/None
Commands File ==> * *, DDname, or Data Set Name
Prompt Level ==> PROMPT Prompt, NoPrompt, ;, *, command
Preference File ==> * *, DDname, or Data Set Name
EQAQPTS File ==> _____ Data Set Name or blank
Select (/) a session type and provide parameters:
- Full-screen mode
Network name ==> _____ Blank or Dedicated terminal Network name
LU name ==> _____ Blank or Dedicated terminal LU name
- Full-screen mode using the Debug Tool Terminal Interface Manager
User ID ==> _____ User ID
^ Remote debug mode
Address ==> 9.30.60.90
Port ==> 8001
Other run-time options: _____
```

Providing Debug Criteria in Debug Tool User Exit Data set (Green screen way)



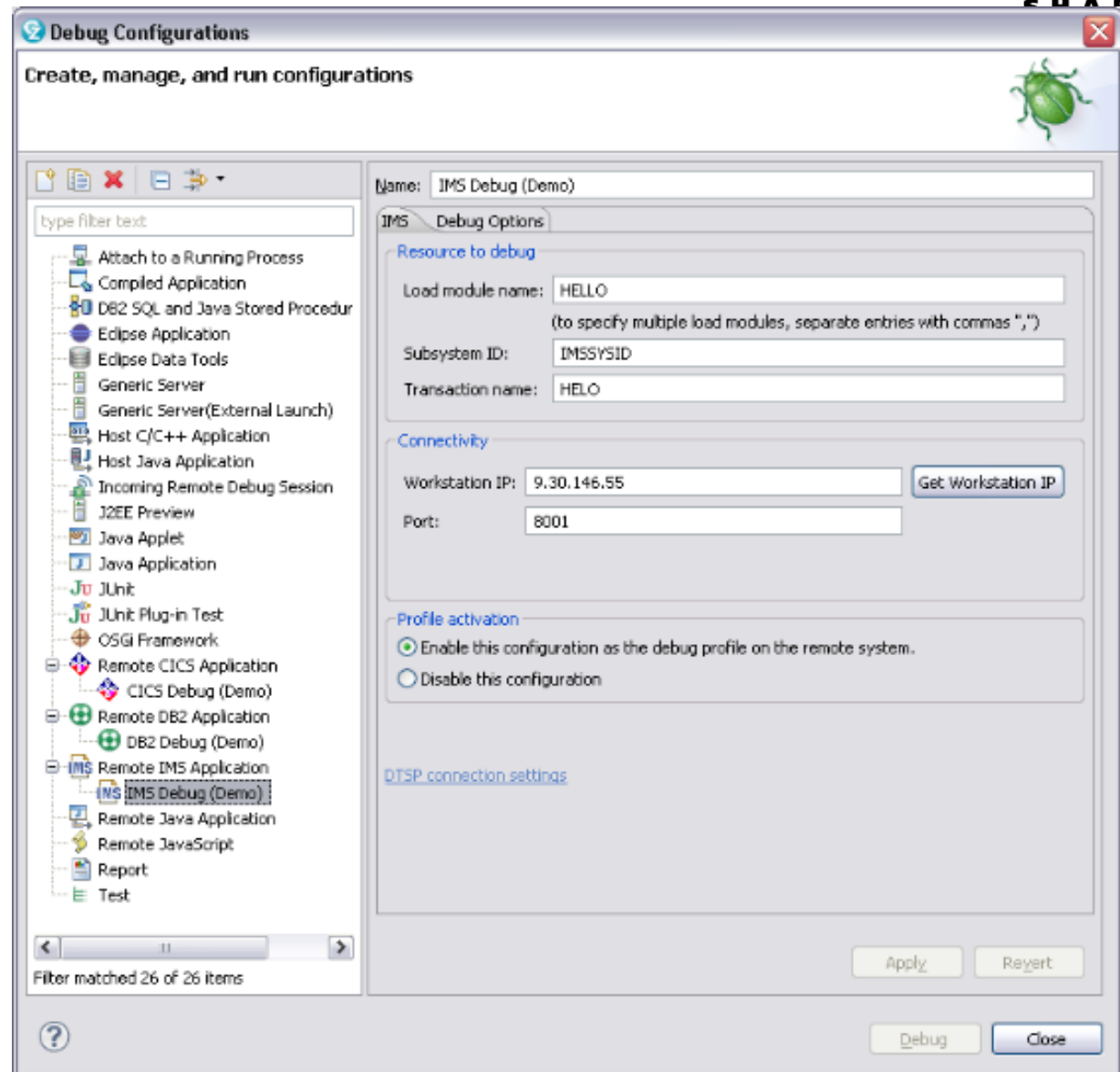
# RDz way in 8.5

## DB2 Debug Configuration UI



# RDz way in 8.5

## IMS Debug Configuration UI



# Rational Developer for System z Roadmap Themes



- Performance and Scalability
- Productivity
- Languages
- Integration

## • Advancing technologies

- IBM z/OS Automated Unit Testing Framework (zUnit)
- Program Analysis

# RDz 8.5 – IBM z/OS Automated Unit Testing Framework (zUnit)



- Unit testing framework for COBOL and PLI
- Similar to JUNIT for Java
- RDz 8.5 will have
  - Test runner infrastructure
  - Wizards to create COBOL and/or PLI test cases
  - Build and run the test cases
  - Display the test execution results along with trace back information



# zUnit Test config viewer



The screenshot displays the 'zUnit Test Runner Configuration' dialog box. It is divided into several sections:

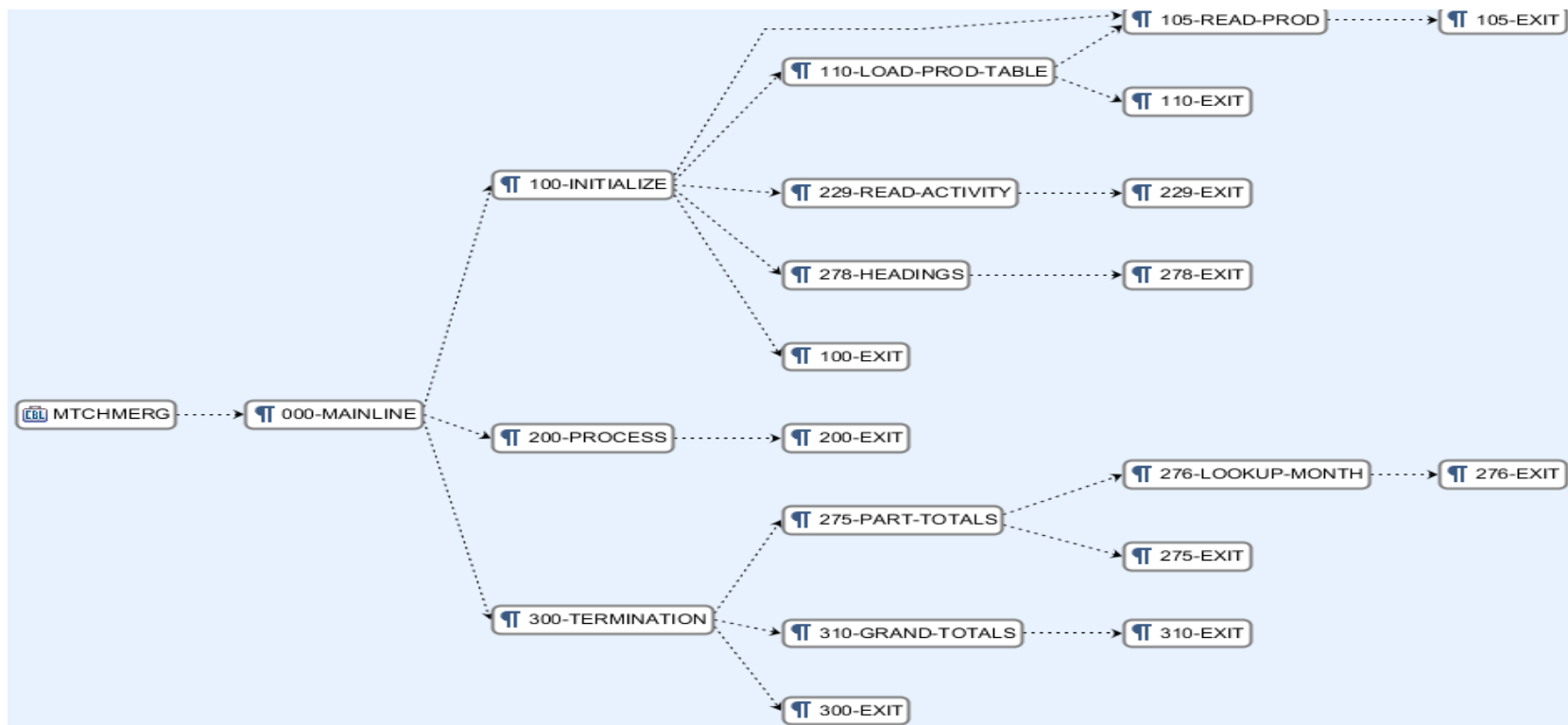
- Test runner configuration:** A tree view showing a 'zUnit Runner' containing several test cases: CBLTESTC, GETRESCD, PLITESTC, REQUESTD, and STKZCHAN. Buttons for 'Add...', 'Remove', 'Up', and 'Down' are visible.
- Settings and actions:** A section for 'Runner continuation settings (optional)' with checkboxes for:
  - Continue if Test failed:
  - Continue if error in Test:
  - Continue if Test case failed:
  - Continue if error in Test case:
- Remote Systems:** A tree view on the right showing a list of MVS Files, including 'Retrieved Data Sets' and 'My Data Sets (USER28.\*)' with various file names like USER28.CICS.COBOL, USER28.CICS01.JCL, etc.
- Bottom Panel:** A table titled 'Subsystem JES' showing file system mappings.
 

Resource	Parent filter pool	Parent filter	Number of filter strings	Connection-private
Retrieved Jobs	CN-mvs040.rtp.raleigh.ibm.com-com.ibm...	Not applicable	1	Yes
My Jobs	rdzdevtemp1.com.ibm.zos.jes	Not applicable	1	No

# RDz 8.5 - Program Analysis



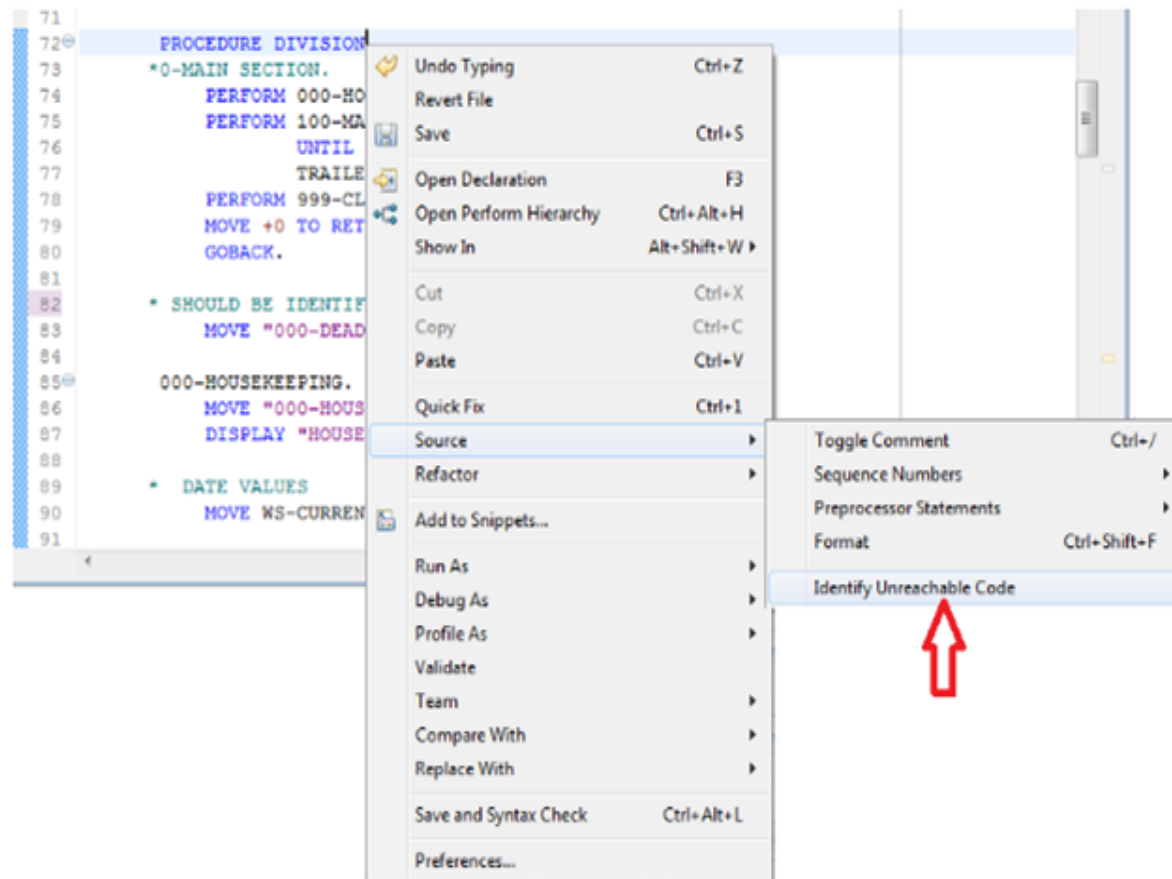
- Scoped to a Program and its “direct” dependencies ( copy books, static calls )
- Available out of the box with RDz 8.5, no additional dependencies or installs
- Features in RDz 8.5 (for COBOL only)
  - **Program Control flow diagram**
  - Identify Unreachable Code
  - Data Element Table



# Identify Unreachable Code

- An editor action to help identify unreachable code
  - ▶ “Dead code” - lines of source code which cannot be reached in a full control-flow analysis.

In fact, generates Program Control Flow Diagram internally, unreachable code is all left over paths





# Data Element Table

- A table representation of the user-defined data items and symbols in a program
  - ▶ Hyperlinks in the table are integrated with the editor allowing easy access to the declaration of the data items.
- Generated by showing the “symbol table” generated when RDz real-time syntax check parses the program

Data Element Table X

Showing data elements from WARDRPT.cbl      Search:

Data Item Name:	Data Type:	Level:	Top-Level Item:	Declaration:	Initial Value:	Line ...	Reference count:	Full Declaration:
PATLISTEST-S-ID	Data	10	PATIENT-MASTER-REC	PIC X(08)		378	0	10 PATLISTEST-S-ID PIC X
PATMSTR	File Descriptor	0	PATMSTR			116	4	FD PATMSTR RECORD CO.
PATMSTR-FOUND	Data	88	FILE-STATUS-CODES			134	1	88 PATMSTR-FOUND VALUE "0
PATMSTR-KEY	Data	5	PATMSTR	PIC X(06)		120	2	05 PATMSTR-KEY PIC X(06).
PATMSTR-REC	Data	1	PATMSTR			119	1	01 PATMSTR-REC.
PATMSTR-STATUS	Data	5	FILE-STATUS-CODES	PIC X(2)		133	3	05 PATMSTR-STATUS PIC X
PATPERSN	File Descriptor	0	PATPERSN			123	4	FD PATPERSN RECORD CO
PATPERSN-FOUND	Data	88	FILE-STATUS-CODES			136	1	88 PATPERSN-FOUND VALUE "
PATPERSN-KEY	Data	5	PATPERSN	PIC X(06)		127	2	05 PATPERSN-KEY PIC X(06).
PATPERSN-REC	Data	1	PATPERSN			126	2	01 PATPERSN-REC.
PATPERSN-STATUS	Data	5	FILE-STATUS-CODES	PIC X(2)		135	3	05 PATPERSN-STATUS PIC
PATSRCH	File Descriptor	0	PATSRCH			98	4	FD PATSRCH RECORDING
PAYMENT-METHOD-TYPE	Data	5	PATIENT-PERSONAL-...	PIC X(02)		313	0	05 PAYMENT-METHOD-TYPE
PEDIATRICS	Data	88	INPATIENT-DAILY-REC			157	0	88 PEDIATRICS VALUE "1010".

