IBM Rational Developer for i

Maintain an ILE RPG application using Remote System Explorer Edit, Compile, and Debug

Open Lab 430247
Using RSE 450153

Rational Developer for i V7.5

Maintain an ILE RPG application using Remote System Explorer

Introduction

This tutorial teaches you how to maintain a payroll application written in ILE RPG using the Remote System Explorer.

Learning objectives

- Start the product and open the Remote System Explorer perspective
- Use tools and views in this perspective to connect to an IBM® i system
- Edit, verify, compile and debug a payroll application
- Use the Application Diagram tool

Skill level

Introductory

Audience

- IBM i developer

System requirements

- IBM Rational® Developer for i, V7.5 and all software updates through the IBM Installation Manager.
- IBM i V5R3, V5R4, or V6R1

Prerequisites

- Basic Microsoft Windows operations such as working with the desktop and basic mouse operations such as opening folders and performing drag-and-drop operations
- It will also help if you understand DDS and ILE RPG.

This tutorial is divided into a number of modules, each with its own learning objectives. You can choose to skip the modules on Edit and Compile. You can go directly to the Debug module if you are only interested in that part. Each module contains several lessons that must be completed in order for the tutorial to work as shown in this script.

Expected results

Upon completion of this tutorial you will know how to edit, compile and debug an IBM i application from the Remote System Explorer. You will also know how to customize the Remote System Explorer.

Conventions used in this tutorial

- **Bold** font for user interface controls
- `Mono-spaced` font for user input and code blocks
- *Italic* font for variable names and glossary terms.

Starting the product and the Remote System Explorer

This module teaches you about the workbench, the workspace, a perspective and specifically the Remote System Explorer perspective.

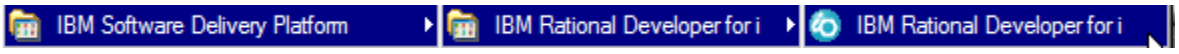
Learning objectives

- Start the product
- Set the default workspace
- Access unique tools and views targeted towards IBM i application development tasks

Starting the product

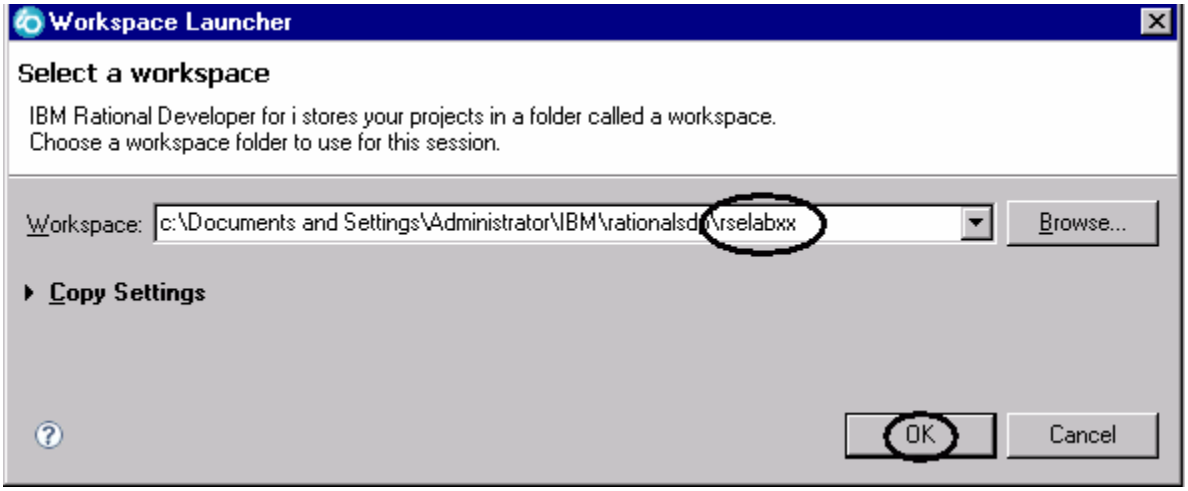
First you must start the product. Follow these steps to start the product:

1. Click **Start** on the task bar of your desktop.

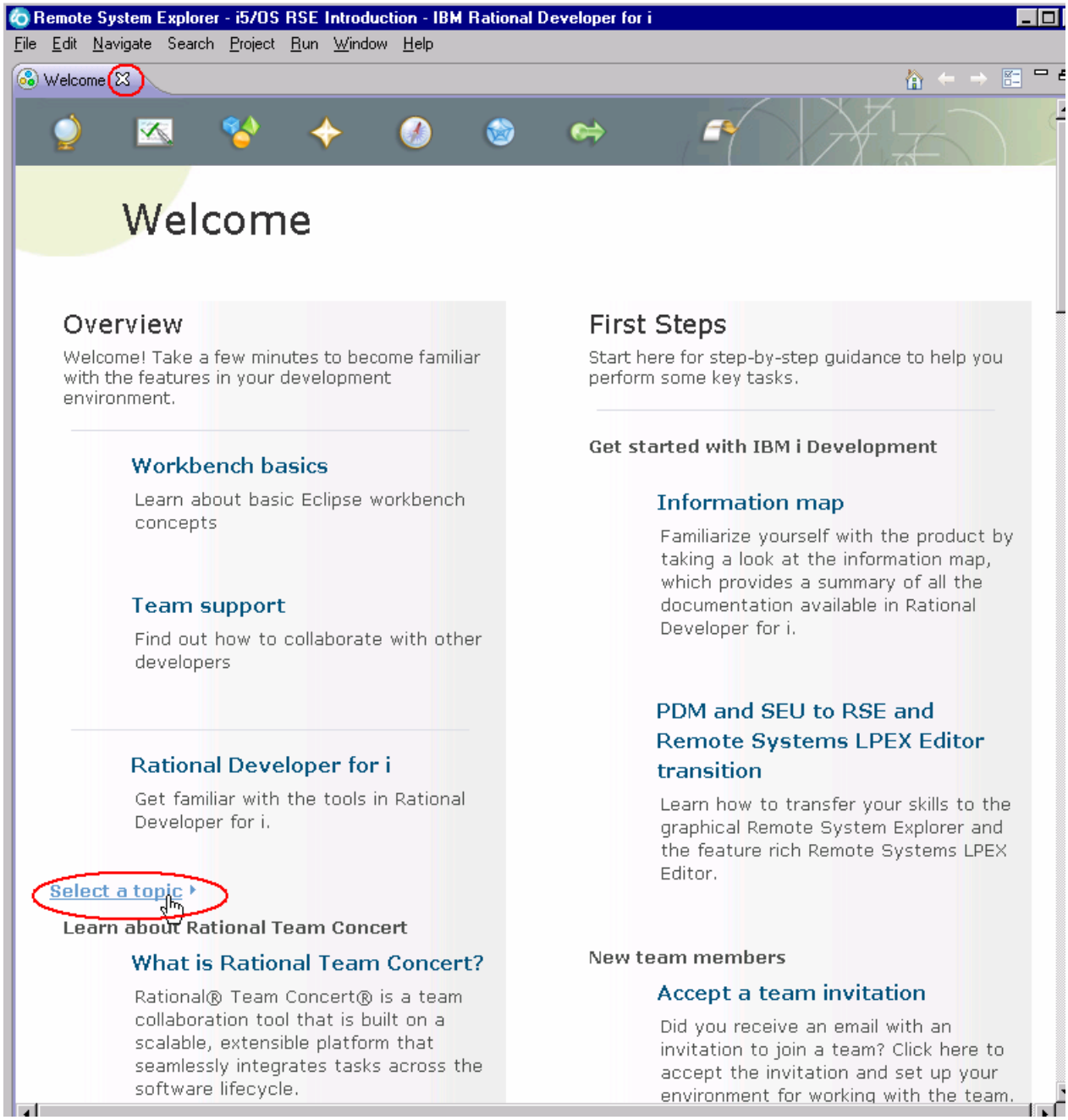


2. Select **Programs > IBM Software Delivery Platform > IBM Rational Developer for i > IBM Rational Developer for i**

If you are working with the RDI SOA version of the product you will see slightly different product names.
A dialog will appear. Here you specify the directory of the workspace where your projects and other resources such as folders, subfolders and files that you are developing in the workbench will reside.



3. (Optional) Change the field in this dialog and use a unique directory name, for example, RSELABxx (where **xx** is a unique number).
   Don't worry about the directory path it might vary from one workstation to the other.
4. Click **OK** to open the workbench.

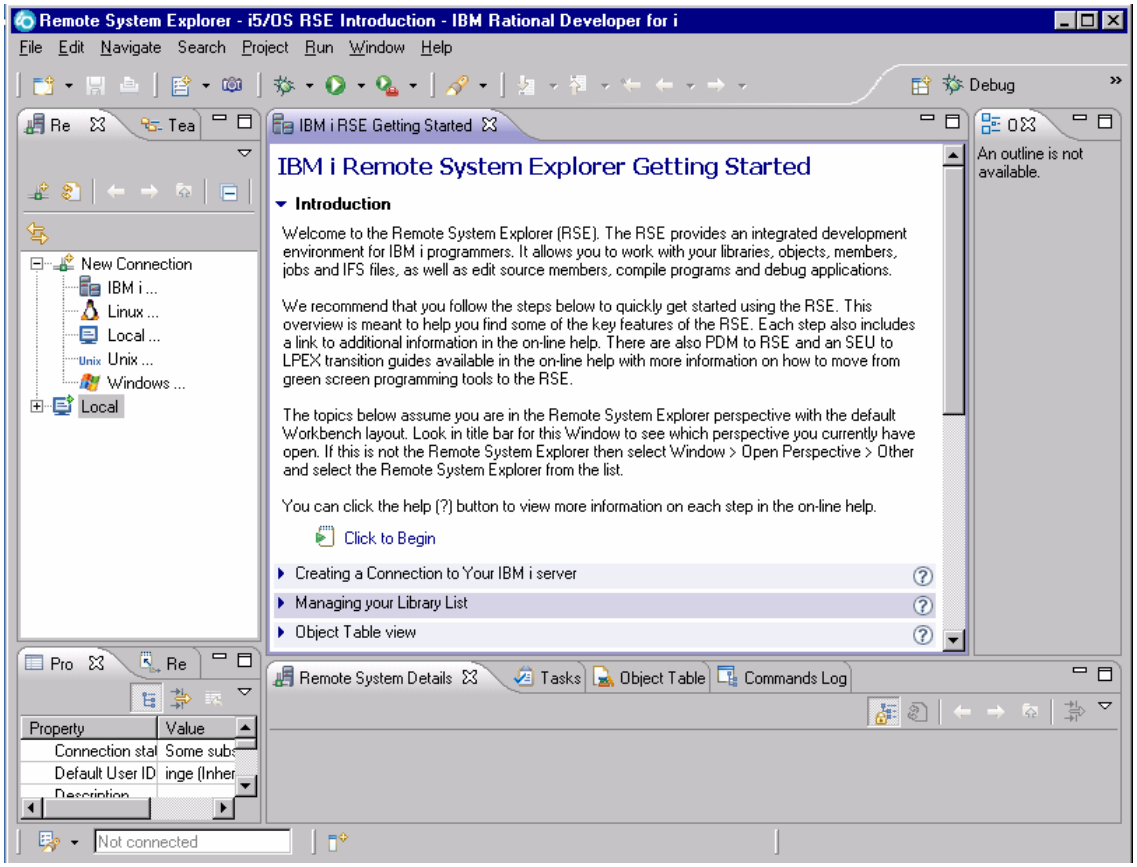


5. Click **Select a topic** to view a list of available information. Select any of the entries and explore the topic. When you are done, click the X next to the Welcome tab to close the Welcome page.

   Closing the Welcome page will take you to the Remote System Explorer perspective.

   Notice that each image on the Welcome page will lead you to additional information about the product.

   Tip: To open the Welcome page again, select **Help > Welcome**.
6. Click the maximize button to maximize the workbench.

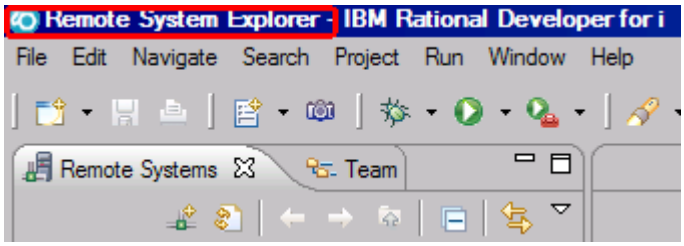


You have started the product and opened the workbench. The workbench refers to the desktop development environment. The workbench aims to achieve seamless tool integration and controlled openness by providing a common paradigm for the creation, management, and navigation of workbench resources. Each workbench window contains one or more views and an editor.

Working with the Remote System Explorer perspective

In the Remote System Explorer (RSE) perspective,

1. If you are not sure which perspective is open at the moment, you can check for the name of the perspective in the workbench title bar.

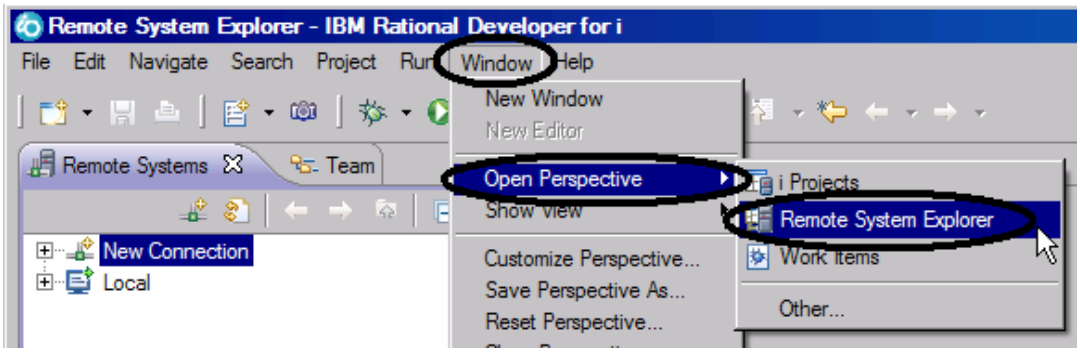


What is a perspective?

A perspective defines the initial set and layout of views in the workbench window. Within the window, each perspective shares the same set of editors. Each perspective provides a set of capabilities aimed at accomplishing a specific type of task or working with specific types of resources. For example, the Java™ perspective combines views that you would commonly use while editing Java source files, while the Debug perspective contains views that you would use while debugging a program. Perspectives contain views and editors and control what appears in certain menus and tool bars.

If you see a different perspective, not the **Remote System Explorer** open in the workbench or no perspective:

2. Click **Window > Open Perspective > Remote System Explorer** from the workbench menu.



The Remote System Explorer perspective opens.

You work in the Remote System Explorer perspective in the workbench. This perspective is for IBM i developers. You can display the connections that you have already configured, create a new connection, connect to and disconnect from the connections that you have defined, work with IBM i files, commands, jobs, and integrated file system files.

This perspective will be active when you start the product with a new workspace. If you had used the workspace before then, the workbench would come up with the perspective that you last opened. You will learn more about the Remote System Explorer perspective in the coming exercises as this is where you launch the IBM i developer tools and use the views from the workbench.

Configuring a connection to IBM i and connecting to a server

This module teaches you how to create a connection to an IBM i server, find a library in your library list, select objects from a library and finally open a member in the Remote Systems LPEX Editor. You also learn about several views such as the Remote Systems view, IBM i Table view, and the Outline view.

Learning objectives

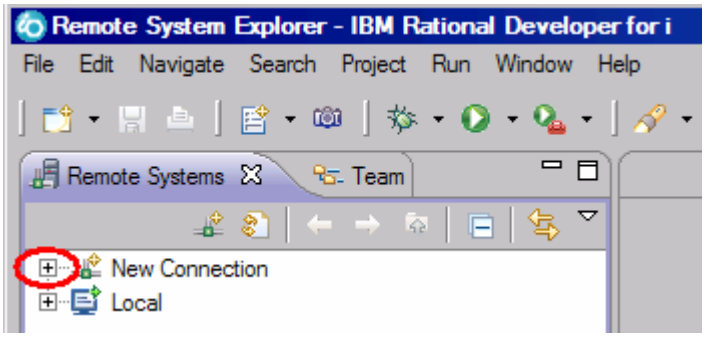
- Create a connection to an IBM i server
- Connect to an IBM i server
- Add a library to your library list
- View libraries in your job's library list from the Remote Systems view
- Find a source physical file in your library
- View members in a source physical file using the Table view
- Customize the columns in the Table view
- Open a member for edit from the Table view or the Remote Systems view
- Maximize the editor space
- Open another member for edit
- Switch from one edit session to another edit session
- Open multiple views of the same source member
- Display a structural outline of items defined in a source member

Configuring a connection to an IBM i server

When you first open the Remote System Explorer, you are not connected to any system except your local hard drive on your workstation. To connect to a remote system, you need to define a connection. When you define a connection, you specify the name or IP address of the remote system and you give your connection a unique name that acts as a label in your workspace so that you can easily connect and disconnect. When you connect to the remote system, the workbench prompts you for your user ID and password on that host.

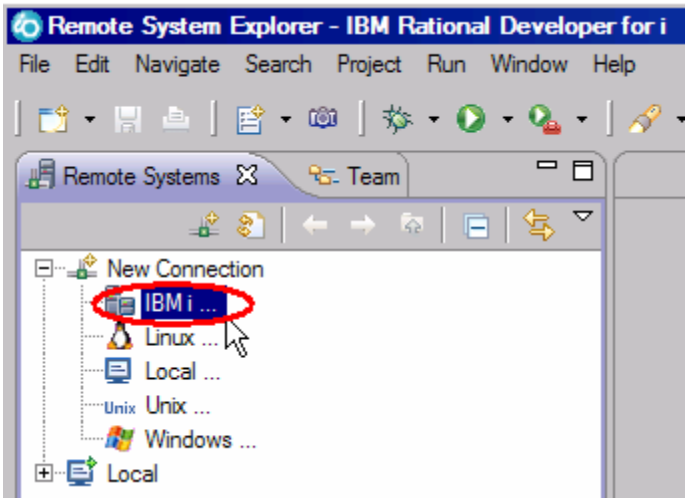
All connections, filters, and filter pools belong to a parent profile. Filters are described in a later lesson. Profiles are discussed when you create your first connection.

Remember you have already opened the Remote System Explorer perspective in the previous module.
In the Remote Systems view,

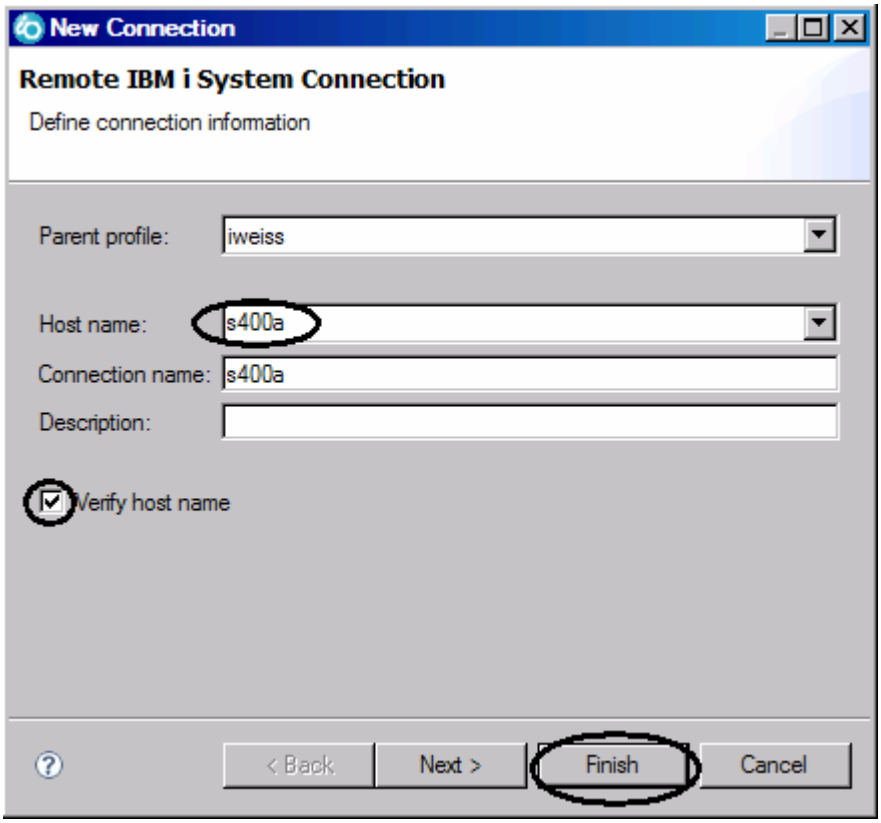


1. Click the plus sign + to expand **New Connection** if it is not already expanded to show the various remote systems types you can connect to through the Remote System Explorer.

To connect to an IBM i remote server



2. Double-click **IBM i** to configure a connection to a remote system.
The Remote IBM i System Connection page opens.



Here you specify the information for your connection. The **Parent profile** defaults to the name of the workstation. Your profile will be different from the one shown here.

The cursor on this page is positioned in the **Host name** field.

1. In the **Host name** field, type the IP address or the name of your host system.

The Connection name is automatically filled with the host name. Leave it this way. This name displays in your Remote Systems view and must be unique to the profile.

2. Leave the **Parent profile** default value. You don't need to change it.
3. Leave the **Verify host name** check box selected.
4. Click **Finish** to define your system.

Connecting to an IBM i system

After you configure a connection to an IBM i system, you can easily connect and expand your new connection to show the subsystems. Subsystems are pre-defined filters grouping the various types of remote resources that can be explored in the remote system. There are four subsystems.

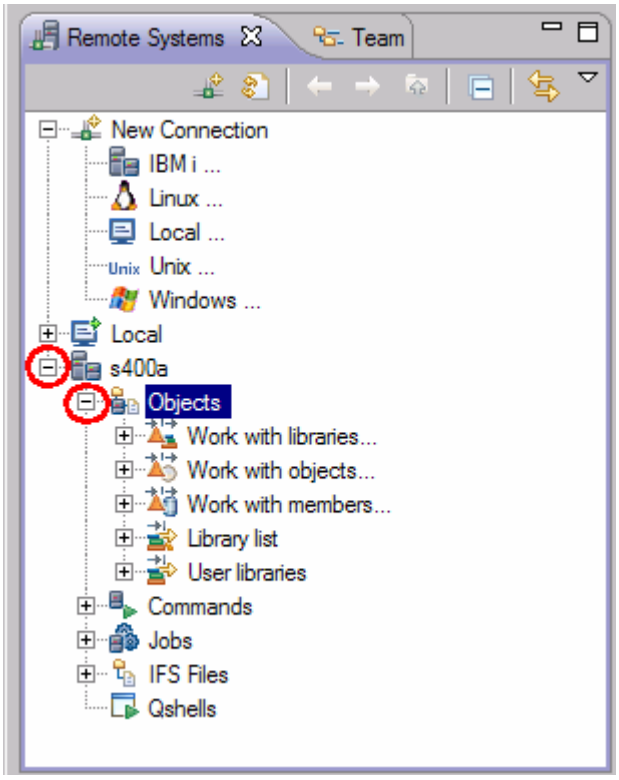
Objects
A PDM-like group, allowing access to libraries, objects and members.

Commands
Contains predefined commands and allows you to create command sets each of which contain one or more often used commands. When run, all commands in a command set are sent to the remote system and executed, and the results are displayed in the Commands log view.

Jobs
Allow you to see various jobs, subset by job attributes, and to perform a number of operations on those jobs.

IFS Files
Allow you to explore folders and files in the Integrated File System of the remote IBM i system.

To connect to the **s400a** system:



1. Click on the plus sign + to expand your s400a connection.

In the Remote Systems view, your new connection is expanded to reveal your subsystems. The **Objects** subsystem is the subsystem you will use most often! It is very similar to PDM, in that it allows you to access objects in the QSYS file system, and perform actions on those objects.

2. Click on the plus sign + to expand the **Objects** subsystem.

Notice the first three entries under the **Objects** subsystem are named after the PDM options, because they have similar capabilities

- **Work with libraries** (similar to WRKLIBPDM)
- **Work with objects** (similar to WRKOBJPDM)
- **Work with members** (similar to WRKMBRPDM)

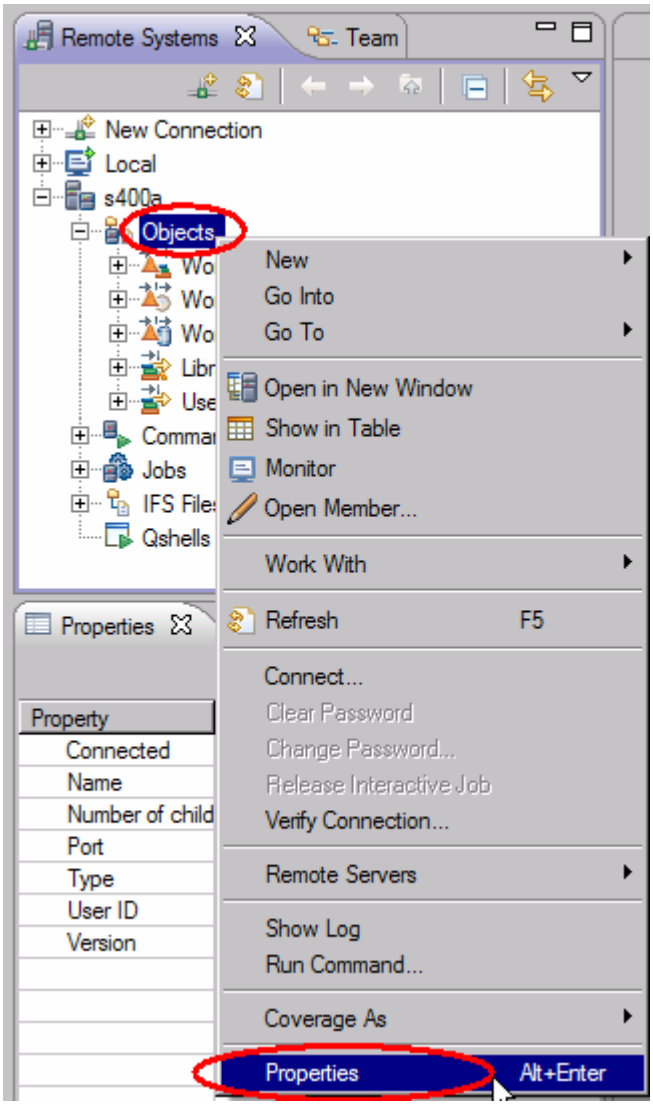
In addition there are entries for working with library lists and user libraries:

– **Library list** (to simulate PDM's WRKLIBPDM you can start with the pre-defined Library list filter, that when expanded lists all libraries in your library list.)

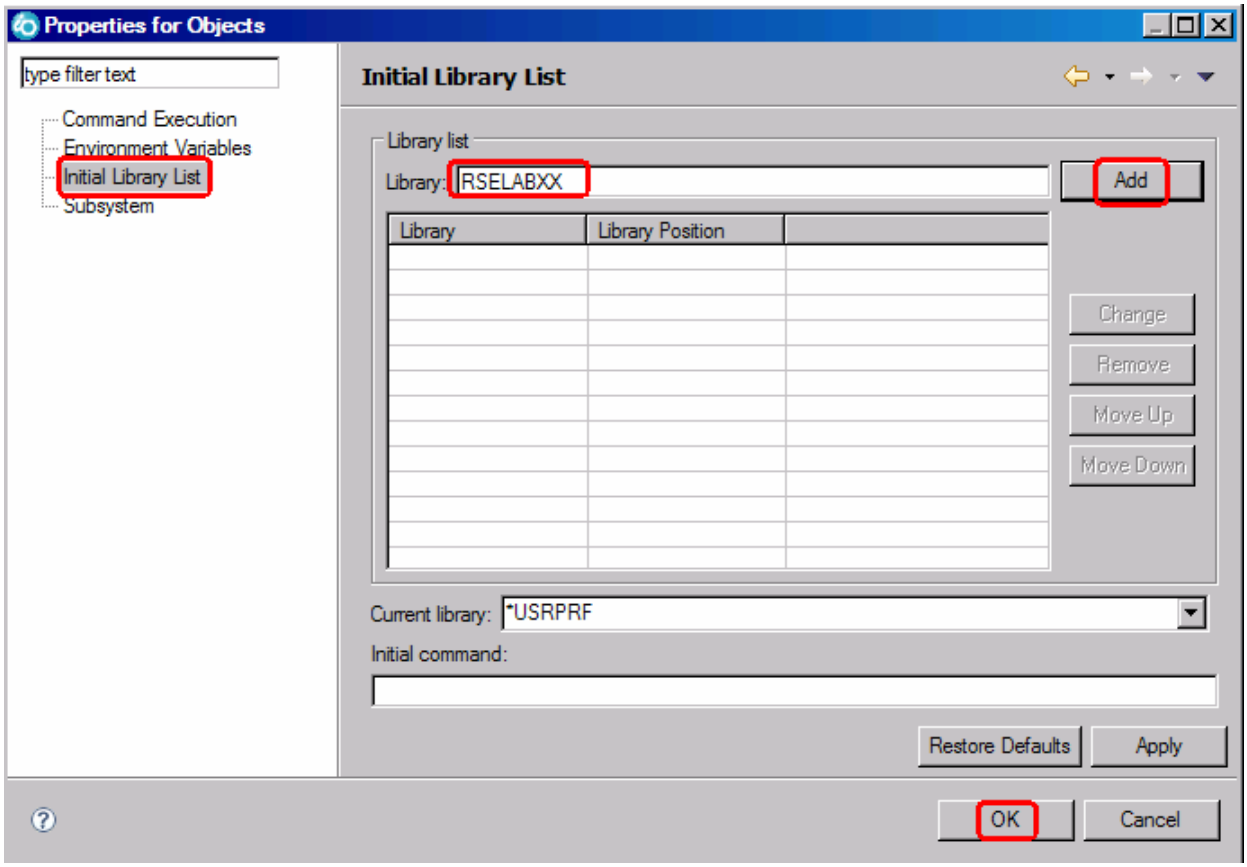
– **User libraries** (allows you to work with all user libraries you can access on that server.)

You also have more entries to work with under the connection itself and you can see from these entries that Remote System Explorer goes well beyond PDM! It allows you to explore IBM i jobs and commands and the IFS file system.

Now let's work with a library in your library list and add the library that you'll be using in this tutorial:



a. Right-click **Objects** and click **Properties** on the pop-up menu. The **Properties for Objects** dialog displays.



b. Select **Initial Library List** on the left pane.

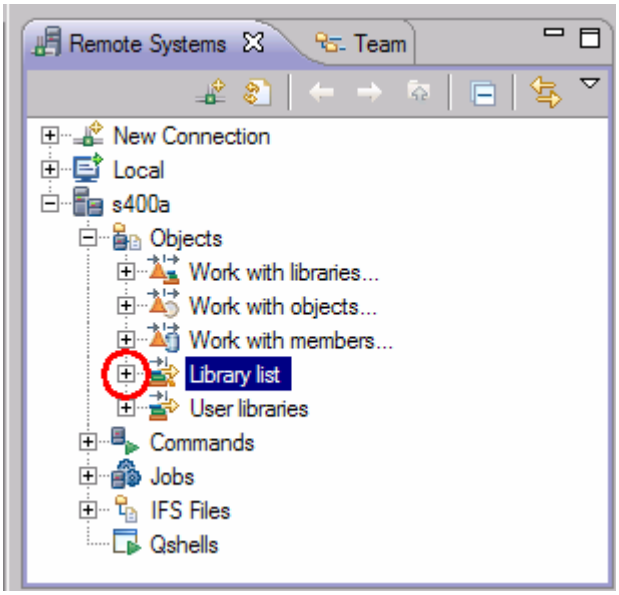
c. In the **Library** field, type RSELABXX where **XX** is your team number.

d. Click **Add.**

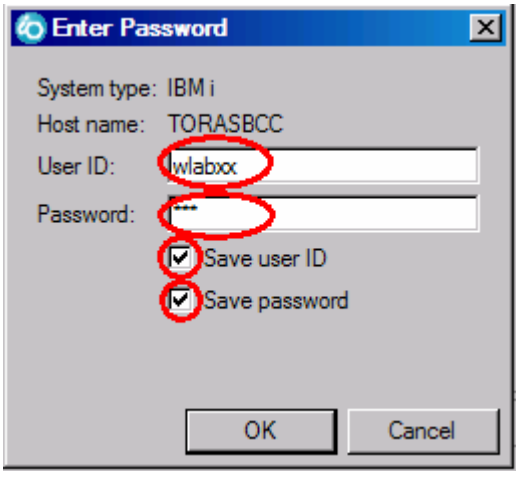
e. Click **OK.**

This will add the library RSELAB**xx** to your library list every time you open this connection. You can use the properties of any of the subsystems to set connection information such as adding a library to the library list.

Tip: You can also change your library list using the pop-up menu items **Add Library List Entry** or **Change Current Library** on the **Library list** folder in the Objects subsystem. These changes are only valid until you disconnect.

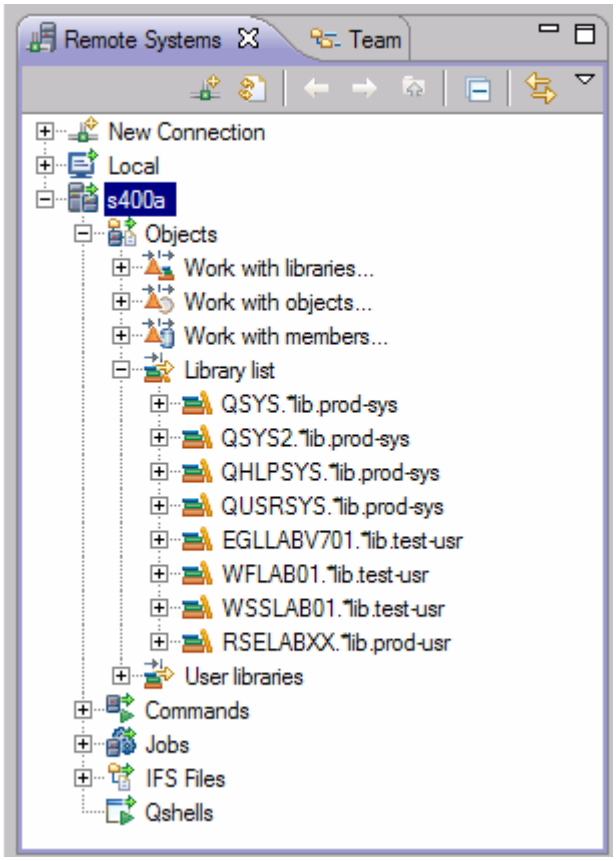


1. Expand the **Library list** folder.
2. Now the connection will be activated and you will be prompted for user ID and password. By default, the user ID field contains the user name that you used to log on to the workstation.



3. Enter your user ID and password. Remember to enter your team number instead of **XX**.
4. Select the **Save user ID** check box.
5. Select the **Save password** check box.
6. Click **OK.**

Back in the Remote Systems view, you will see the libraries in your job's library list.



Notice that the s400a folder now has a small green arrow in the icon to indicate that it is an active connection.
For each library, you can right-click and select from a number of actions. For example, there is an action to create a new source file within the selected library. Common actions like delete, move, copy, etc. are valid for all kinds of objects.

You have connected to an IBM i system and used the Remote Systems view to view libraries in the library list.

Viewing and accessing objects in the Remote System Explorer

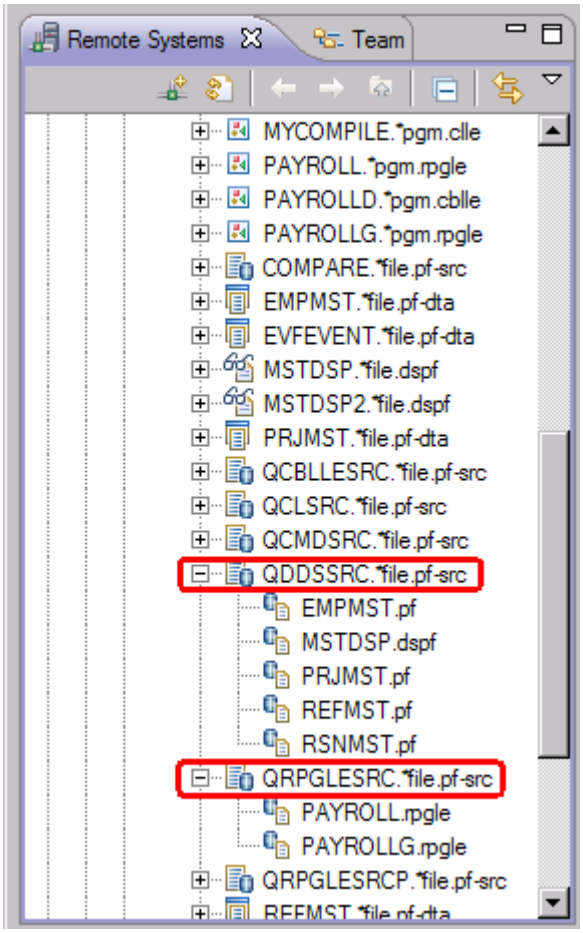
Now you are ready to view and access objects in your library RSELABxx.

To view and access an object:

1. Expand library RSELABxx by clicking the **plus sign +** beside it.

You will see all objects in this library appear in the Remote Systems view. For each object you can right-click and select from a number of actions. The list of actions depends on the object selected and whether you selected one or multiple objects. For example, for a source file the pop-up menu has an action to create a new member within the selected file.

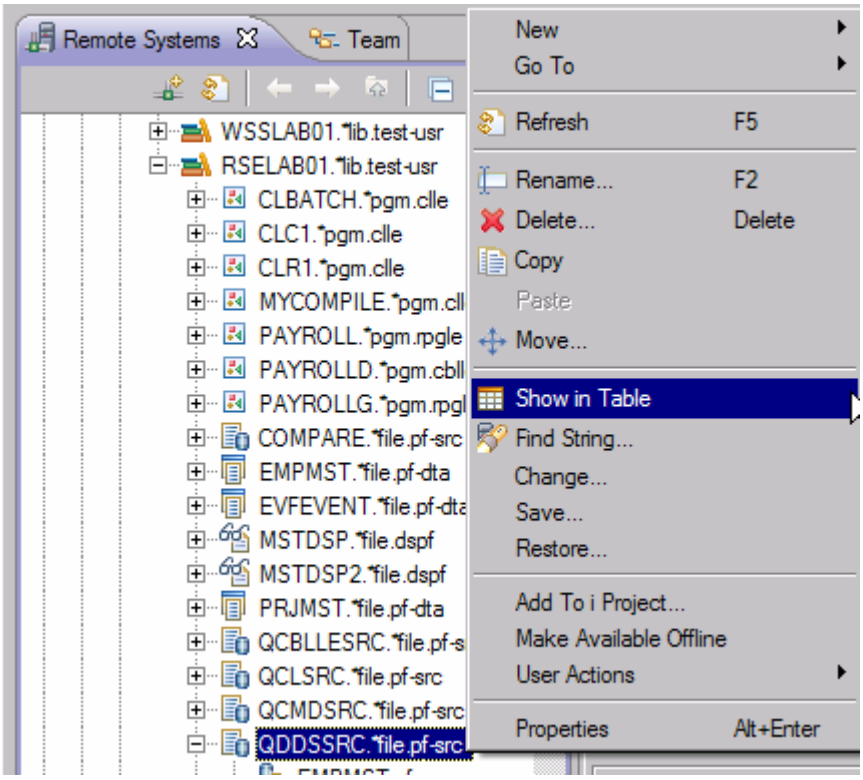
2. Scroll-down through the files in the Remote Systems view until you find QDDSSRC source file and expand it. Still in the Remote Systems view, locate QRPGLESRC source file and expand it as well.



Now you can see and access the members in these two source files. For each member you can right-click and select from a number of actions. The exact list of actions depends on whether the member is a data file or source file and whether you select one or multiple members. For a RPG source member, the pop-up menu actions include:

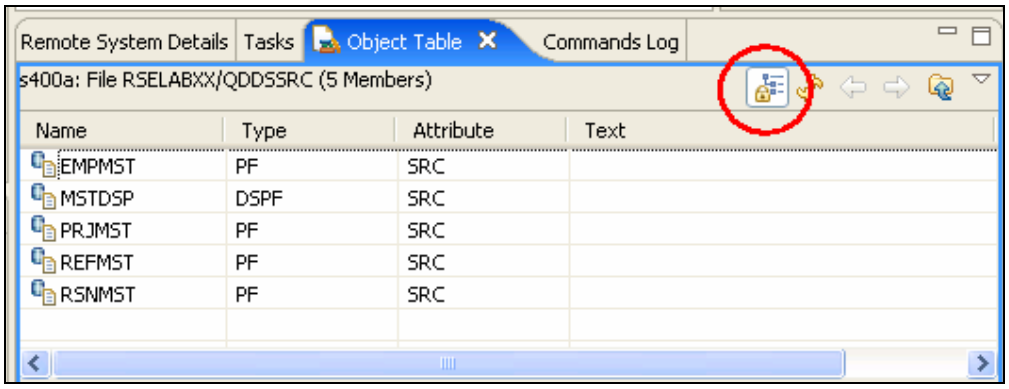
- open with
- browse with
- verify
- compile

Before you go ahead and work with these members, let's see the members in the Object Table view as well because that is similar to the view you are used to from PDM. You use this view to display a list of items, for example members or objects, in a table format similar to PDM. You can also perform actions against these items such as editing and compiling.



3. Right-click the QDDSSRC file and then click **Show in Table** on the pop-up menu.

The Object Table view takes the selected object in the Remote Systems view as input, and displays the contents in the table. For source physical files, this step displays the members inside, their names, types, attributes, and text descriptions.



The top of the Object Table view contains a lock icon that controls the correlation between the Remote Systems view and the Object Table view. If the lock is disabled then whenever you click an object or library in the Remote Systems view, the associated contents of that item automatically populate the Object Table view. If the lock is enabled then when you click on various items in the Remote Systems view, this view does not change the content of the Object Table view. To enable or disable the lock, you can click it once to change its state.
You can click on the columns heading to sort the view by column.

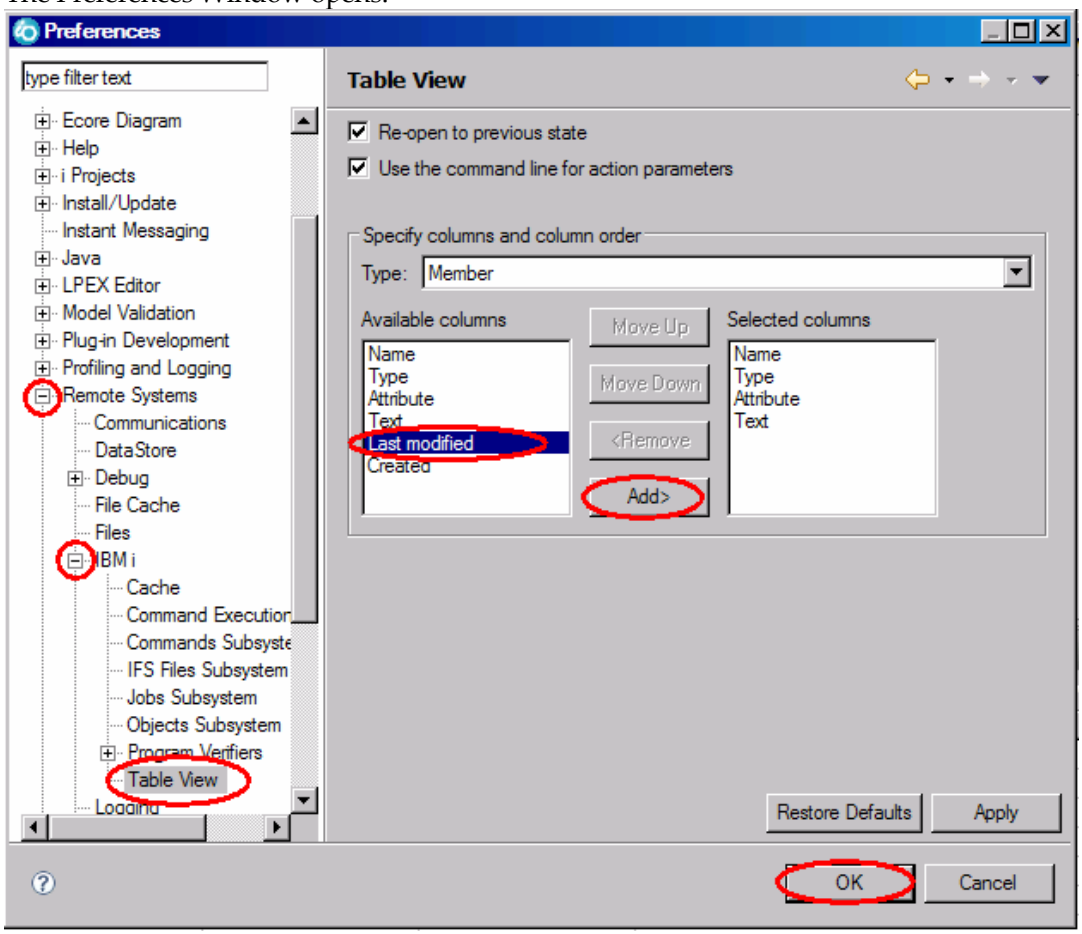
1. In the Object Table view toolbar make sure the lock/unlock button is in the unlock position. Leave the mouse pointer over the tool button for a second or two to display the flyover help. That way you can check if the view is locked or unlocked.
This means now the table will automatically be updated when a different object is selected in the

Remote Systems view. This is a shortcut to open the pop-up menu for an object in the Remote Systems view and to select Show in Table.
You can also modify which specific columns you want to see in the Object Table view.

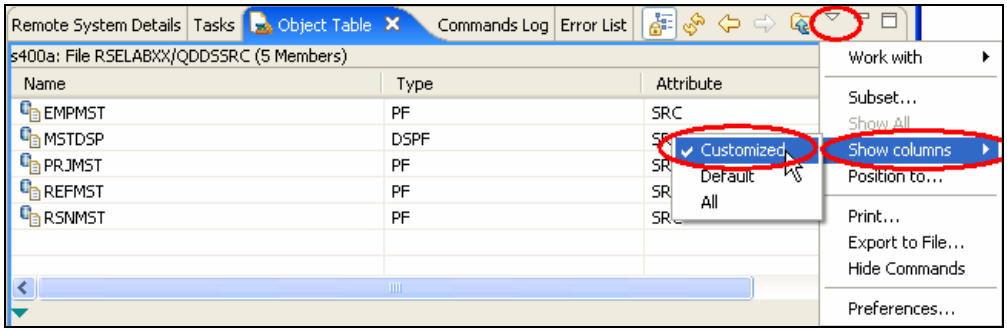
To modify the Object Table properties:

a. Click **Window > Preferences** from the workbench menu.

The Preferences Window opens.

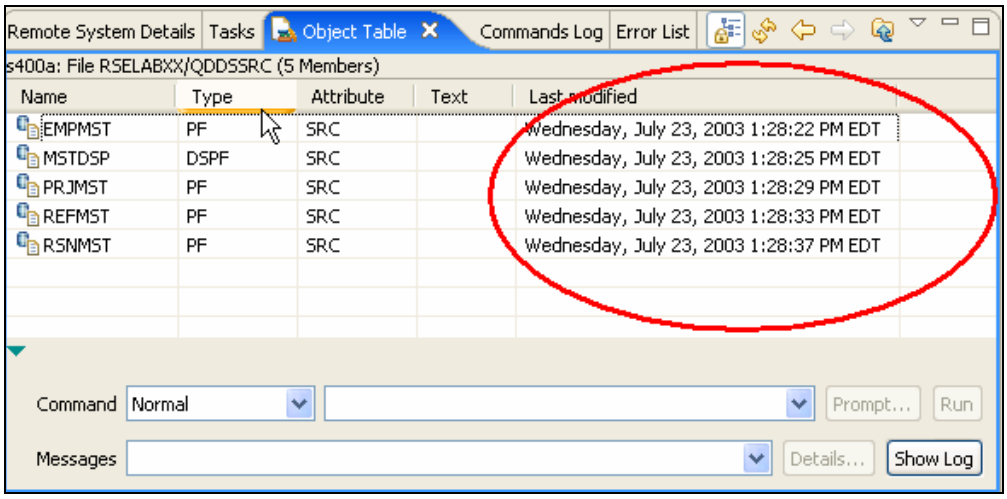


b. In the left pane of the Preferences window, expand **Remote Systems**.
c. Expand **IBM i** under Remote Systems.
d. Click **Table View** under IBM i.
e. In the right pane of the Preferences window, select **Last modified** in the **Available columns** list
f. Click the **Add** button.
g. Click **OK**.
h. Now, let's update the Object Table view. Click the down arrow on the Object Table view title bar



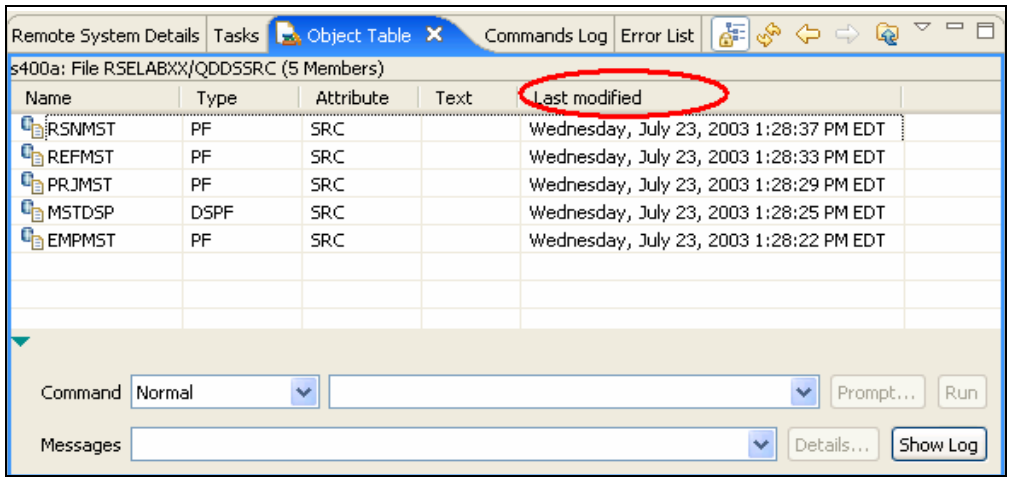
i. Click **Show columns > Customized** in the pop-up menu.

Now you'll see the extra column that you've added.



You can also sort the objects in the object Table view by column.

j. To sort the objects in ascending order by Last modified, click on the heading.

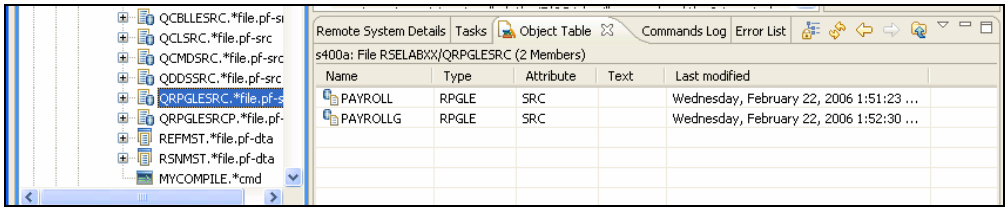


k. If you click the heading the second time, it will sort it in descending order.

7. Back in the Remote Systems view,

Select QRPGLESRC.

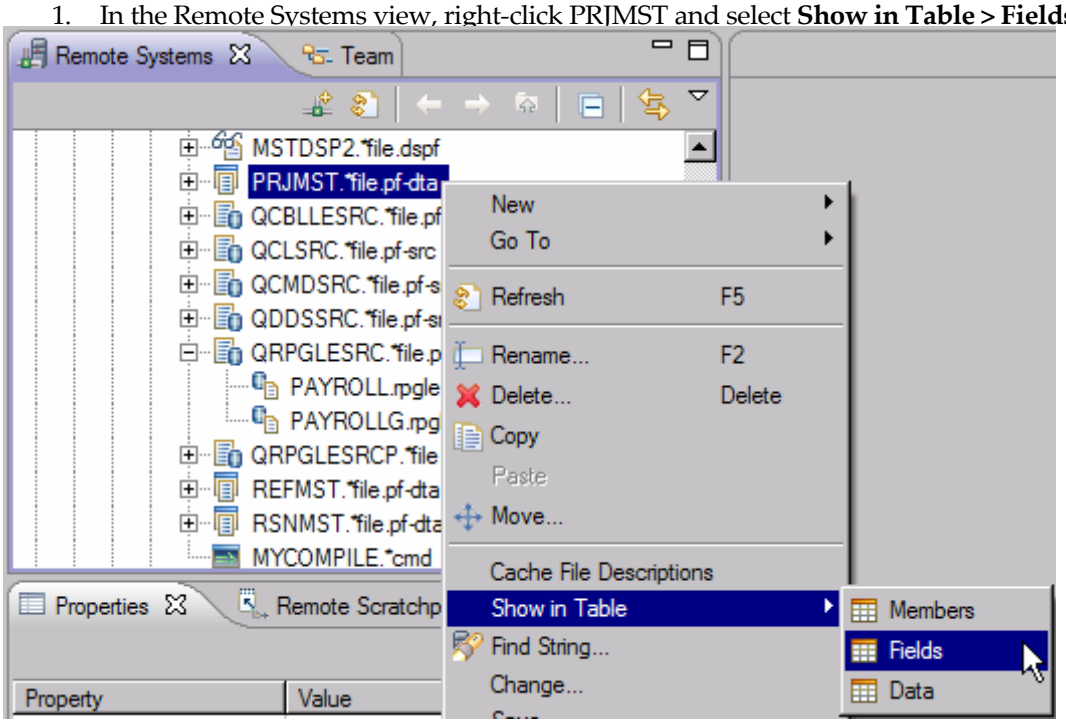
The table shows the members in QRPGLESRC.



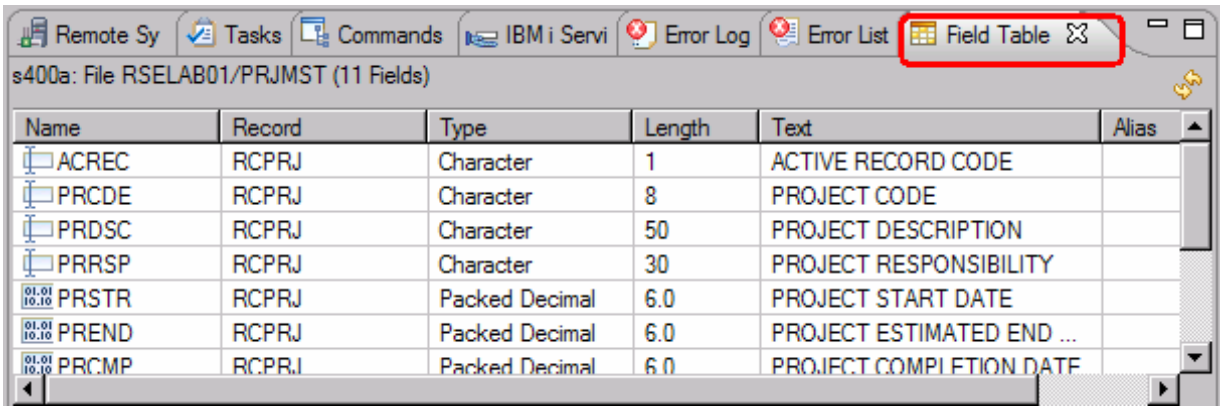
Besides the Object Table view, there are also a Field Table view and a Data Table view. For a selected data file, the Field Table view shows which fields are defined in the file and their properties. The Data Table view displays the data contained in the file.

To display the other table views:

1. In the Remote Systems view, right-click PRJMST and select **Show in Table > Fields**.



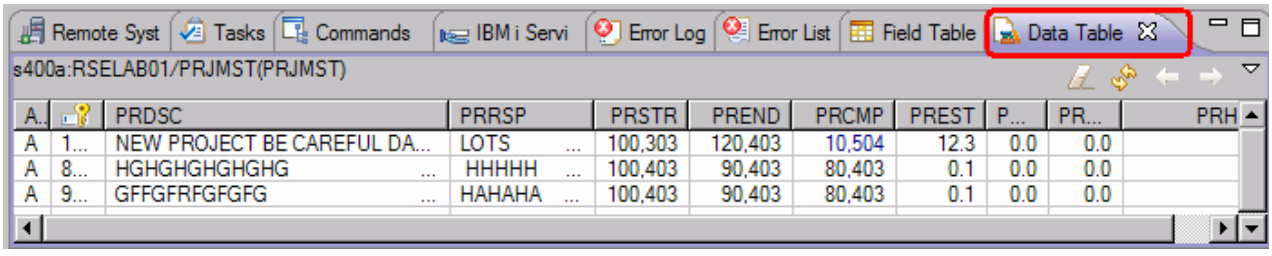
The Field Table view opens.



To display the Data Table view:

1. In the Remote Systems view, right-click PRJMST again, select **Show in Table > Data**.

The Data Table view opens.



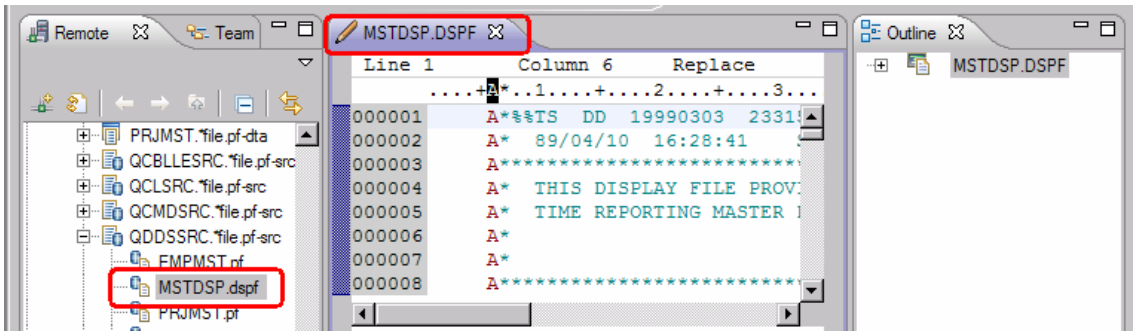
Now you are ready to use the Remote Systems LPEX Editor to edit the member MSTDSP found in QDDSSRC.

8. From the Remote Systems view double-click member MSTDSP in the QDDSSRC source file.

You can do this in the Remote Systems view or in the Object Table view.

The Remote Systems LPEX Editor opens. It is built right into the workbench, with rich editing functions and is IBM i aware! It is a superset of SEU! The syntax checker is ported from SEU, and the reference manuals are built-in and F1 cursor sensitive.

9. Double-click the **MSTDSP** tab to maximize the Editor window.
10. Double-click the **MSTDSP** tab again to return the view to its original size.



You have viewed and accessed objects in the RSELABxx library.

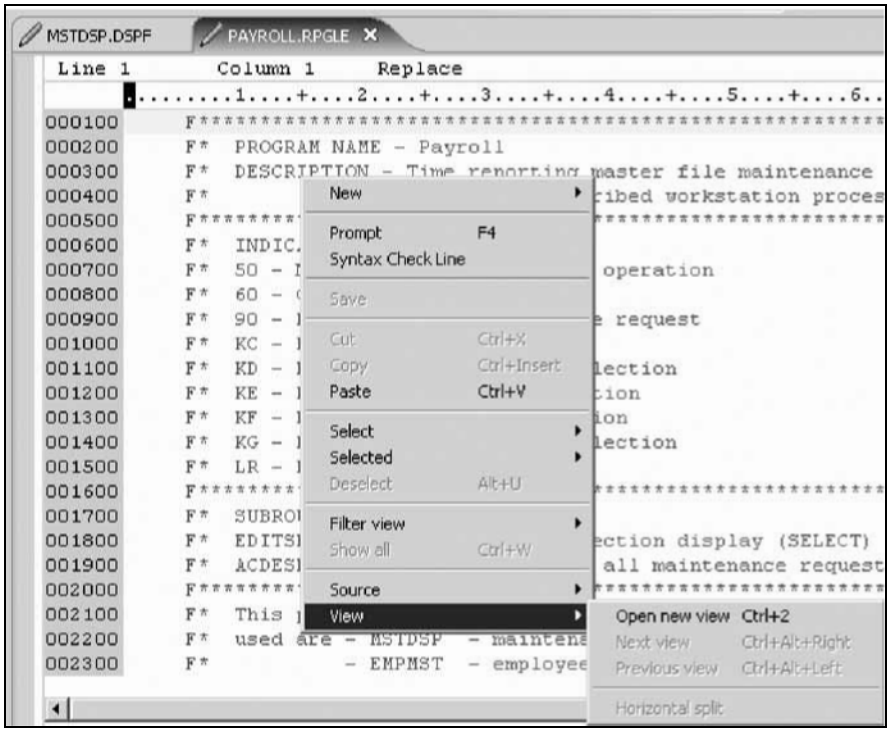
Opening a second source member and multiple views

Next let's open a second member in the editor.

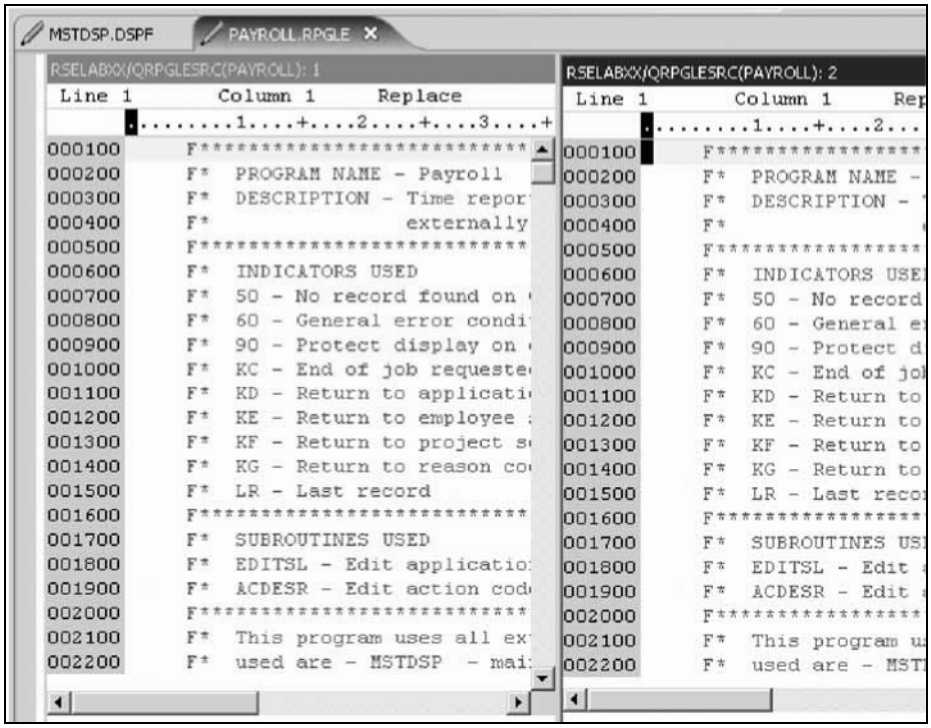
To open a second source member:

1. In the Remote Systems view, double-click member PAYROLL in the QRPGLESRC source file.
2. Click on each tab to switch from one edit session to another edit session.
3. You can open multiple views of the same source member while editing in the Remote Systems LPEX editor. To open multiple views of your RPG source:
   a. Double-click the **PAYROLL** tab in the editor to maximize the Editor window.
   b. Right-click this source in the Editor view and click **View > Open new view**.

Tip: You can open a maximum of five views of the same source.



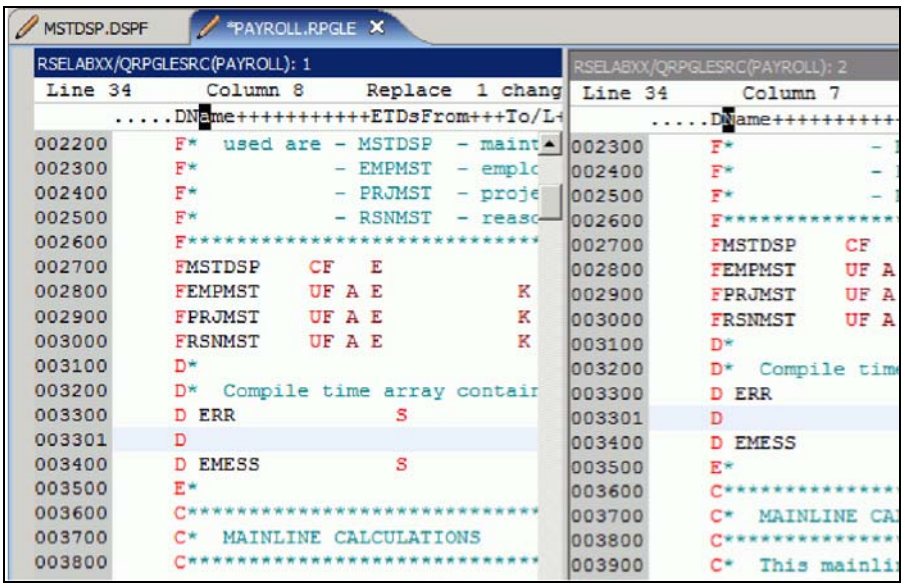
c. Right-click a source view and select **View > Next view** or **View > Previous view** to navigate among the views.



Note: Any changes made in one of the views will automatically update all other views of the same source.

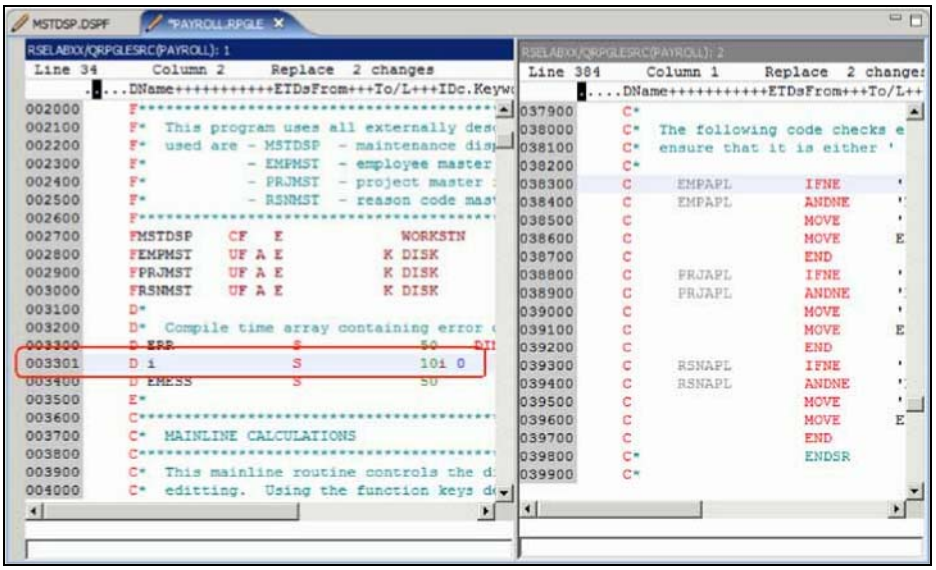
d. Scroll down and place the cursor at line 33 in the left view of the source.

e. Press Enter.



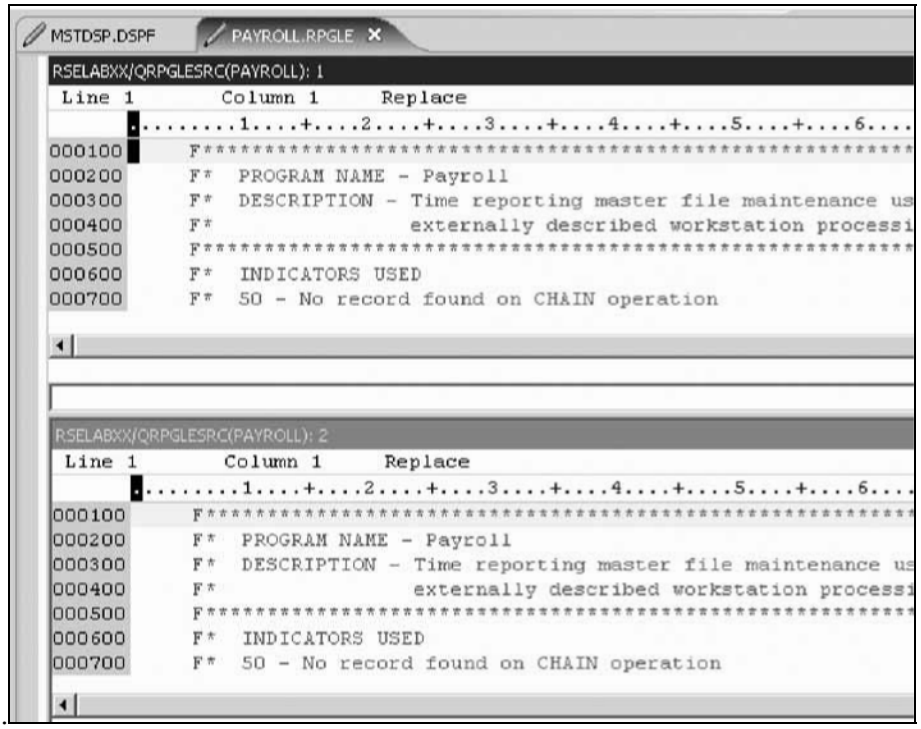
A new line is inserted in both views and since the previous line is a D-spec, the new line is also marked as a D-spec.

f. Define a new variable, i as shown in the screenshot below, with length 10:



Now, in the view on the right, you can start using the new variable while being able to maintain your view on the left on the definition of the variable.

g. Right-click a source view and select **View > Horizontal split** to change from a vertical split of the views to a horizontal split of the views.



h. Right-click the source view that you want to close (RSELABXX/QRPGLESRC(PAYROLL): 2)

h. Select **View > Close view**

Tip: This option is not available on the first view.

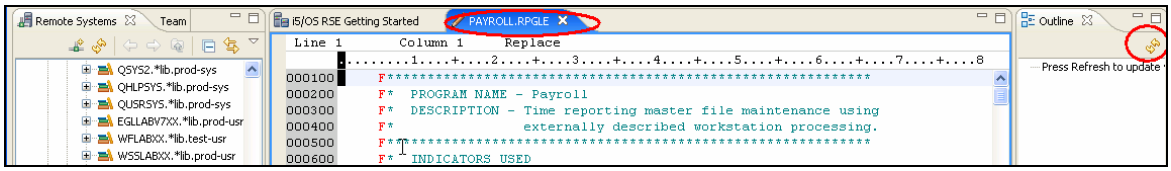
i. **Double**-click the **Payroll** tab to return the Editor window to its original size.

You have opened another member for edit and seen multiple views of a member.

Displaying an outline of a source member

The Outline view acts as an excellent resource when you want to edit RPG, COBOL and DDS source in the Remote Systems LPEX editor. The Outline view displays a structural outline of items defined in the file that you currently have open in the Remote Systems LPEX editor window. With the editor active, you can expand the file structure in the Outline view, and click various elements in the view to jump to that location in the source itself.

To see an Outline view of your RPG source:



1. Click the **PAYROLL** tab in the editor and click **Refresh** on the Outline view toolbar.

Tip: If you have closed the Outline view, you can reset the perspective by selecting **Window > Reset perspective** from the workbench menu or **Window > Show view > Other** then expand **General** and click **Outline** in the Show view dialog.

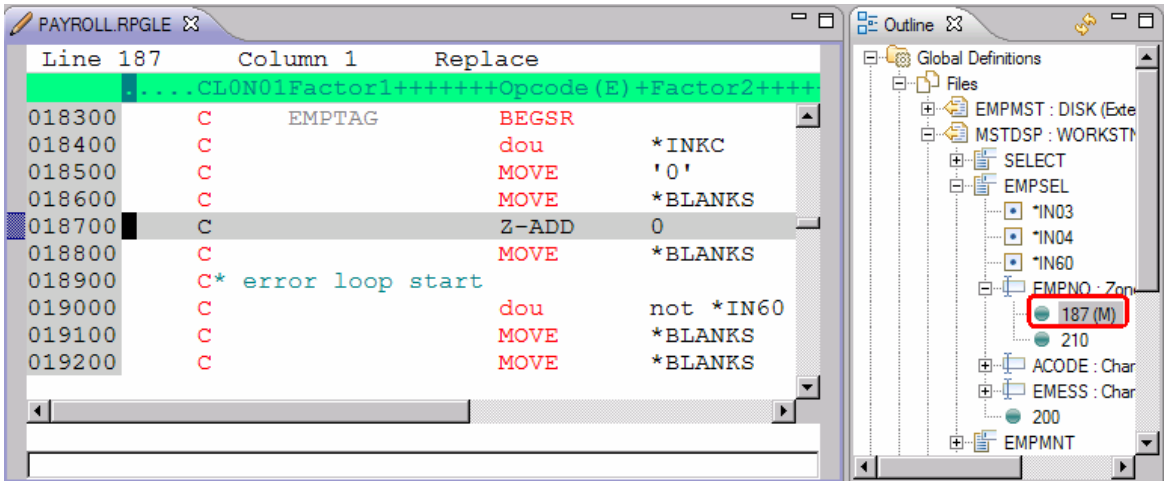
The Outline view contains your source program in a tree view without the lines containing logic.

Now you want to see more details of your source member.

2. Expand Files.
3. Expand the MSTDSP workstation file.
4. Expand the EMPSEL record format.
5. Expand EMPNO.
6. Double-click on any line number in the Outline view.

This will position the source editor accordingly.

Tip: Double-clicking a field or a variable in the Outline view will position the editor to its definition.



7. Click the **PAYROLL** tab to get the PAYROLL editor window in focus for the next lesson.

You have displayed an outline of a source member while editing RPG or DDS sources.

Module summary

You have learned how to create and configure a connection to an IBM i system and access libraries, files and members in that system. You have also learned about several views such as the Remote Systems view, Object Table view and the Outline view.

Editing source

This module teaches you how to edit ILE RPG source member PAYROLL, which should already be open, and learn about some of the Remote Systems LPEX Editor's language features.

Learning objectives

- Change the default settings of the LPEX Editor Parsers
- Change the color settings and font used by the Editor
- Change the default behavior of the Enter key
- Use SEU commands to edit source
- Undo and redo source changes
- View language sensitive help for the MOVE operation code
- View a list of all help contents
- Limit the search of help to specific documents
- Search the help
- Request a prompt for a specification line
- Display context sensitive help for any field in the IBM i Source Prompter
- View the beginning and ending of constructs in your source
- Use the Find and Replace window to search for an item in your source
- Filter or subset your source
- Filter lines based on line type
- Search through members in a source physical file
- Compare different versions of a program and identify the differences
- Syntax check source by line
- View help on syntax errors
- Use the Application Diagram to understand your program code

Introducing the editor

Your program editing tasks are simplified with the Remote Systems LPEX Editor. The editor can access source files on your workstation or your IBM i system directly. When a compilation results in errors, you can jump from the compiler messages to an editor containing the source. The editor opens with the cursor positioned at the offending source statements so that you can correct them.

Here is a list of some of the basic editor features that you would expect in a workstation editor:

- Cut, copy, and paste
- Block marking of lines, characters, or rectangles with copy, move, and delete operations
- Powerful find and replace function
- Unlimited undo and redo

In addition there are a few more functions that you may not have seen in a workstation editor:

- Token highlighting where different language constructs are highlighted using different colors to help identify them in a program
- SEU-like format-line rulers to show the purpose of each column for column-sensitive languages like RPG and DDS. These rulers can automatically update themselves to reflect the current specification.
- SEU-like specification prompting for CL, RPG, and DDS

- Sequence numbers, which allow SEU-style commands in the prefix area
- Intelligent tabbing between columns for column-sensitive languages
- Automatic uppercasing for languages that expect uppercase
- Settings for column-sensitive languages that simplify text insertions and deletions
- On-line language reference

Changing default editor settings

The LPEX Editor has predefined settings, but also has an associated preferences page containing settings that you can modify. The name of the category is LPEX Editor and it appears in the left pane of the Preferences window.

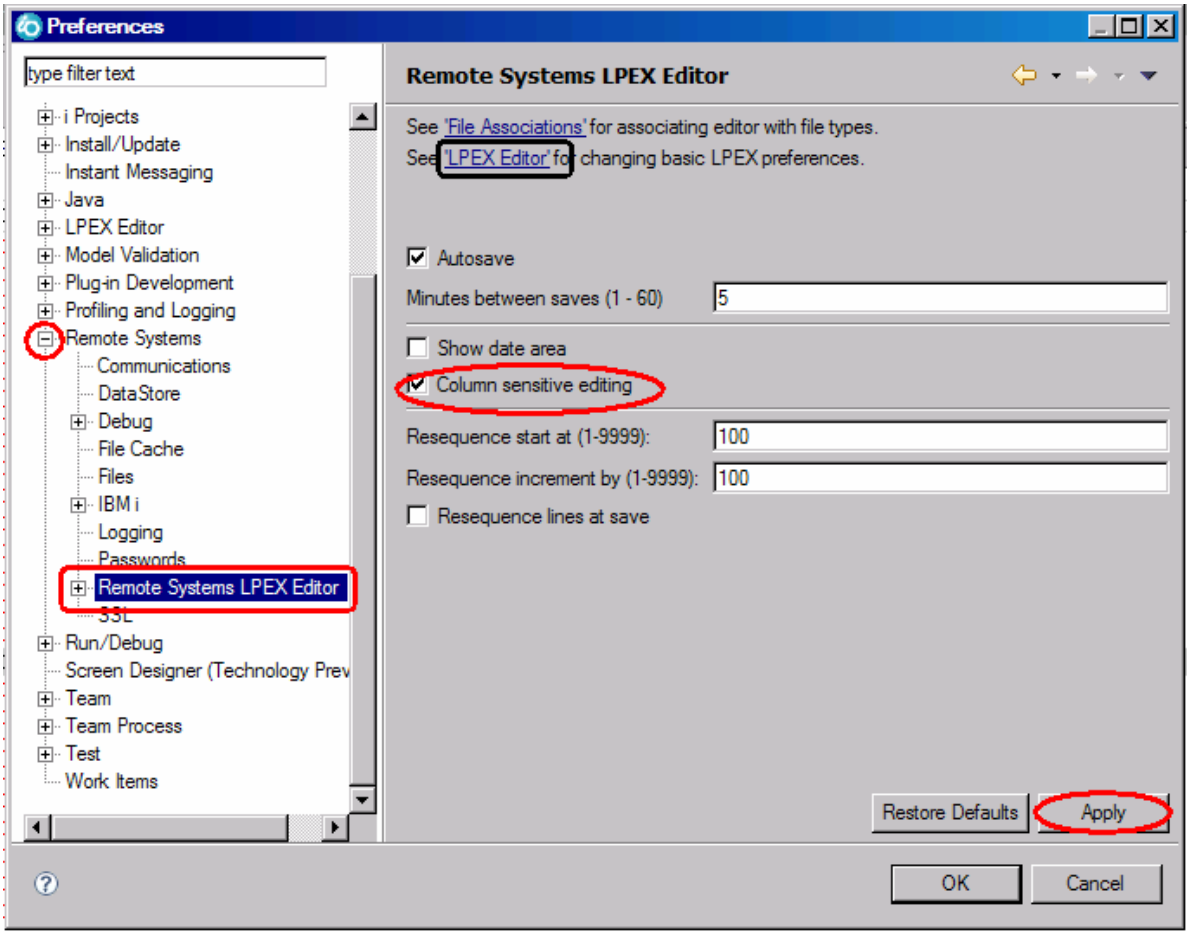
You will change the default settings of LPEX Editor Parsers, Appearance and User Key Actions.

LPEX provides special support for insertion and deletion in column-sensitive languages.

When column-sensitive editing is selected, each column is considered as a separate entry space. For example, in an RPG source member, if you are inserting into or deleting characters from a string that is in the Factor 2 entry, the Result field entry does not move. The default editor preference is that column-sensitive editing is off. You can switch this support on by going to the workbench preferences window.

To set column sensitive editing:

a. Click **Window > Preferences** from the workbench menu.
The Preferences window opens



b. In the left pane of the Preferences window, expand **Remote Systems**.

c. Select **Remote Systems LPEX Editor**.

The right pane allows you to set preferences for this feature.

d. In the right pane of the Preferences window, select the **Column sensitive editing** check box. When selected, each column is considered as a separate entry space.

e. Click **Apply**.

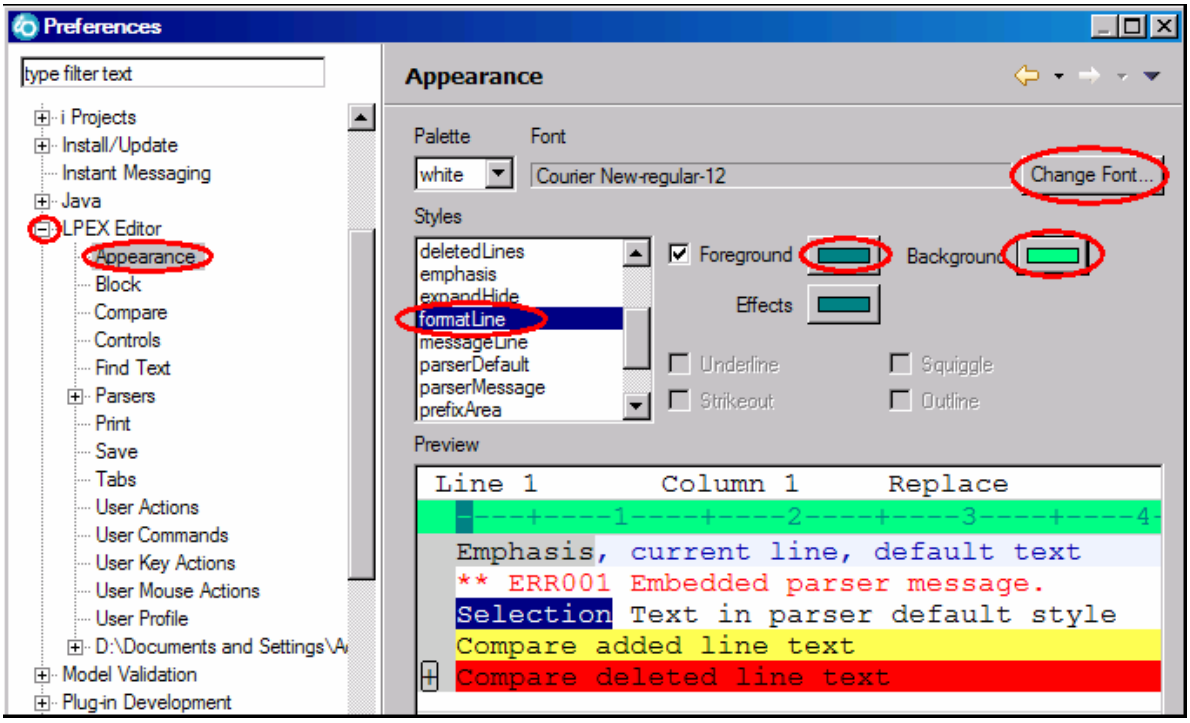
Autosave while editing source:

To enable or disable autosave while editing source in the Remote Systems LPEXEditor, select or de-select the Autosave check box.

By default, autosave is enabled and the value for the minutes is set to 5. You can specify a value between 1and 60 minutes

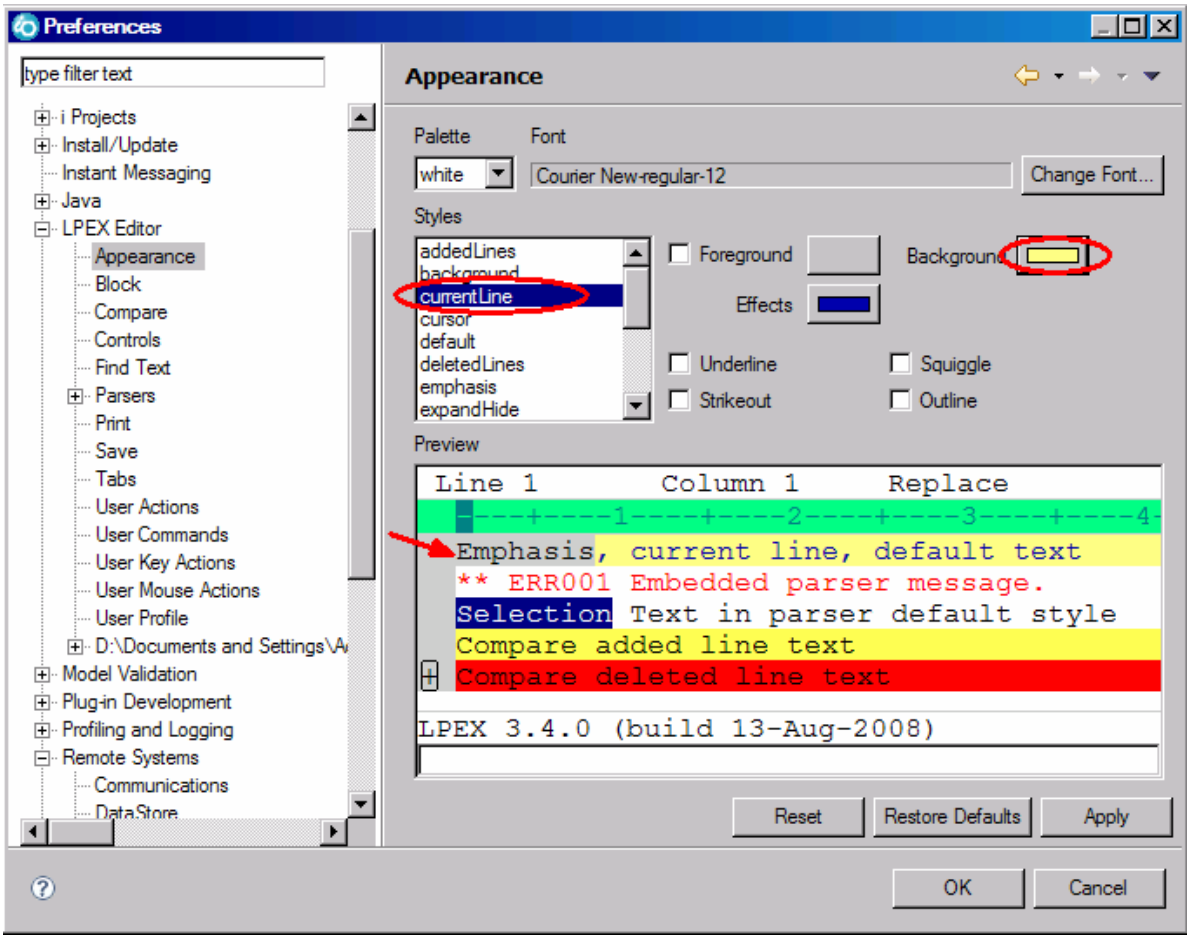
Other interesting preference settings are located under LPEX Editor. You can use the link at the top of the page to quickly jump to the LPEX Editor preferences. Appearance allows you to modify color settings and font used by theEditor.

To change the editor appearance:



a. In the left pane of the Preferences window, expand **LPEX Editor**.
b. Select **Appearance** under LPEX Editor
c. In the right pane under the **Styles** list, select **formatLine**.
d. Change the **Foreground** color to dark green.
e. Change Font to 12.
f. Change **Background** color to light green.

Notice, how your changes are reflected in the sample edit view.



To change the current line appearance to make it more visible when editing:

g. Select **currentLine** under the **Styles** list.

This option highlights the line that the cursor is on. The option applies to all source files opened in the editor area.

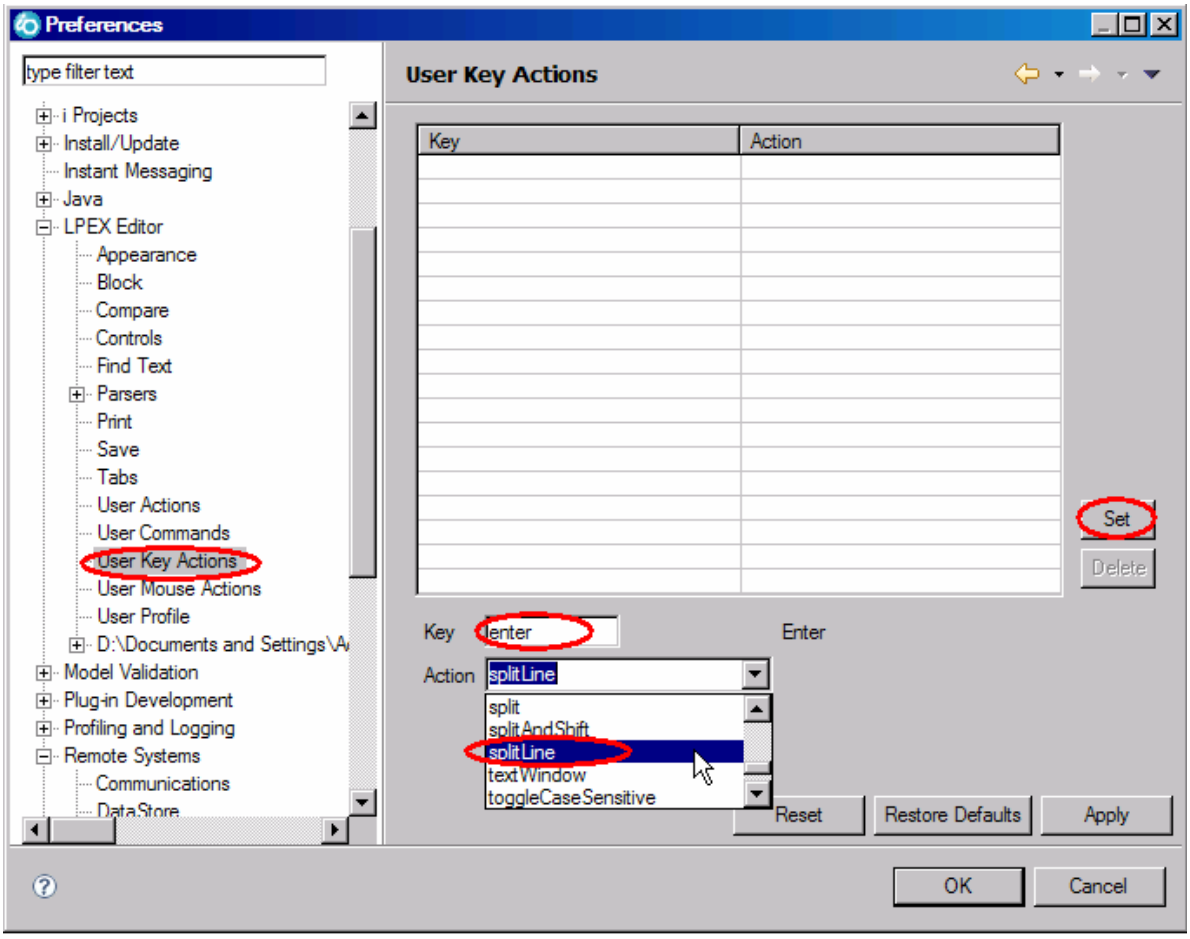
h. Change the **Background** color to light yellow.

i. If you don't like the changes you made, you can click **Restore Defaults** to return to the original settings.

To modify the default behavior of the Enter key:

a. Expand **LPEX Editor** if not already expanded

b. Select **User Key Actions**.



The LPEX Editor has a predefined behavior for the Enter key to always add a new line and not split the line when the Enter key is pressed. To change the default behavior for the Enter key to split the line, use the following instructions.

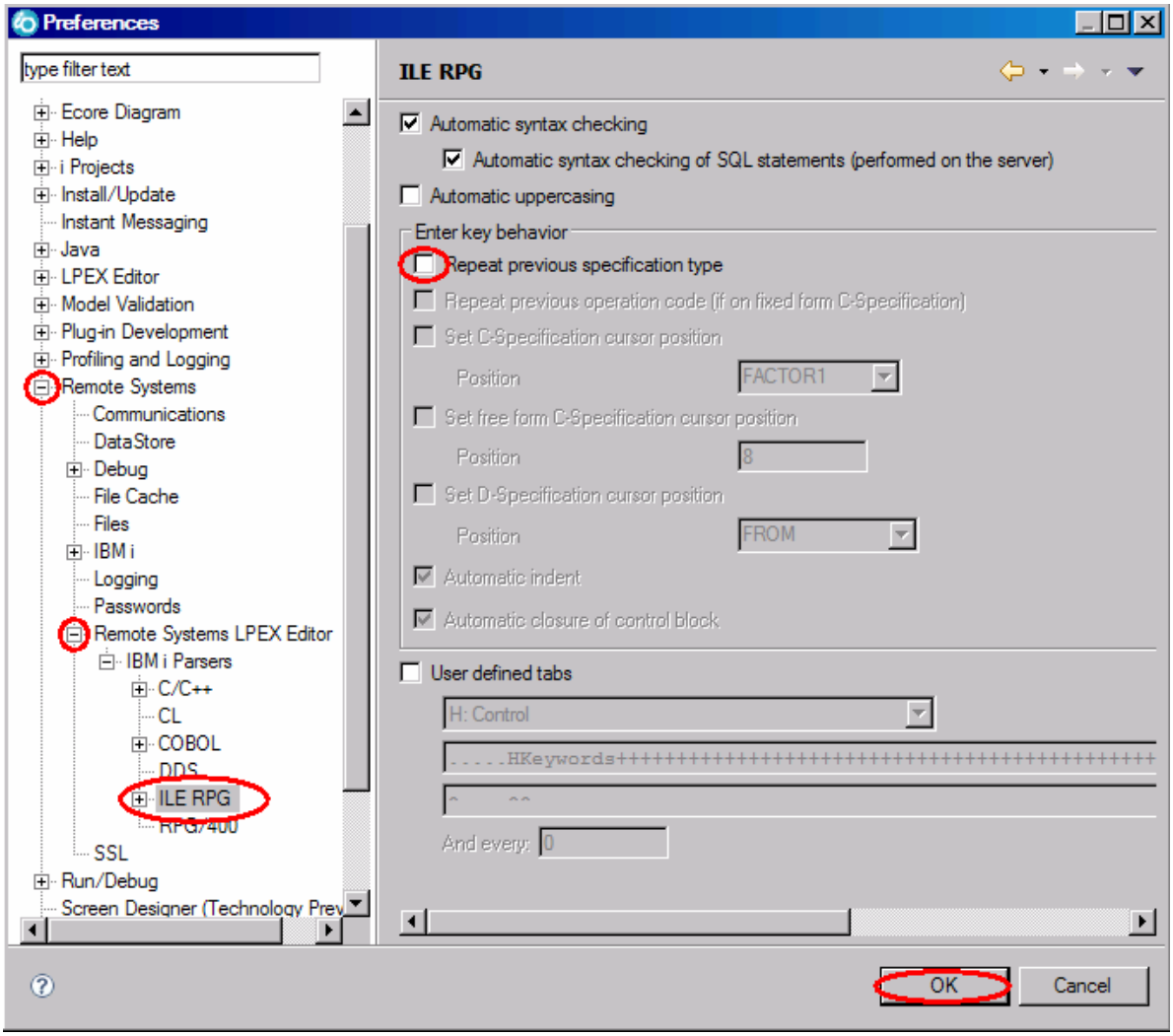
a. Type **`enter`** in the Key field.

Tip: The Key and Action fields are case sensitive. Make sure that the values typed in the Key and Action fields are exactly as shown above.

b. Type **`splitLine`** in the **Action** field. **Tip:** Use the drop down list to select the value

c. Click **Set**.

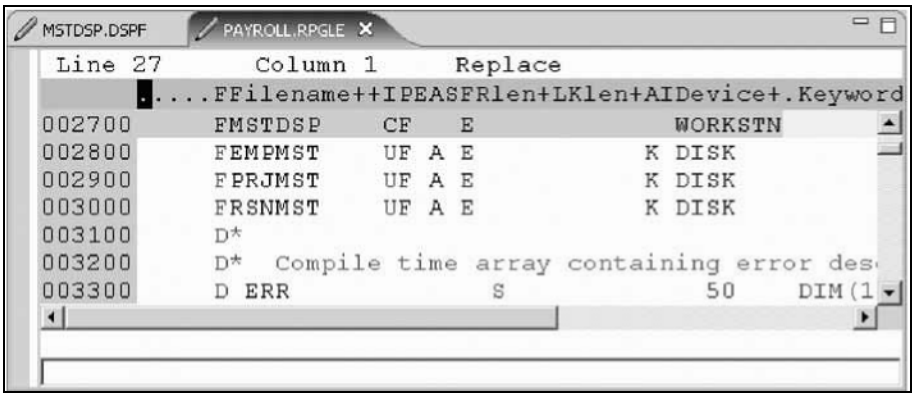
We go through one other preferences entry that users like to change. Adding the previous specification type to a new line when editing RPG source is not liked by all users. Here is how to change this behavior:



1. Expand **Remote Systems**
2. Expand **Remote Systems LPEX Editor**, and then **IBM i Parsers**
3. Select **ILE RPG**
4. Clear the **Repeat previous specification type** check box
5. Click **OK** on the Preferences window
6. Return to the Editor window.

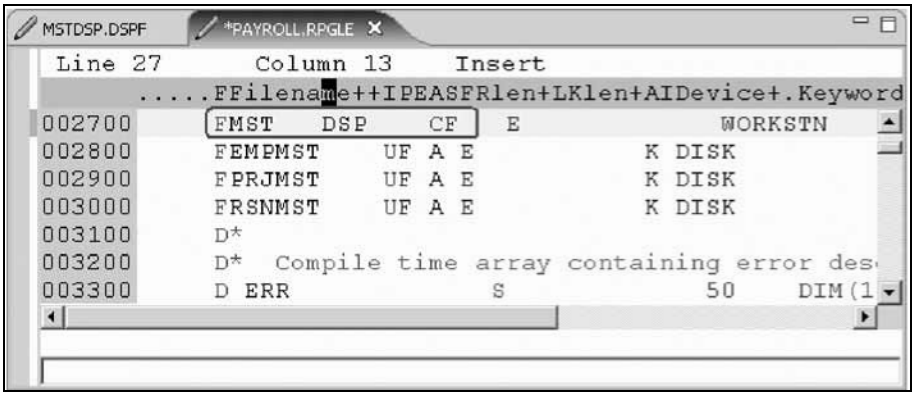
Next let's see the results of the customization.

Notice that the line the cursor is on is now highlighted in the color you selected for it in the preference setting for highlight current line.



1. Move the cursor to line 27, column 10
2. Make sure the editor is in **Insert** mode. If the status area shows **Replace** press the **Insert** key.
3. Press the spacebar 3 times.

Notice that only the filename is shifted but none of the other columns to the right are effected.



4. Press the **Backspace** 3 times. Once again the filename is shifted but no other columns are effected.

Next let's see the results of splitLine

5. Place the cursor somewhere on a line and press **Enter**. The text to the right of the cursor is moved to the next line.

You have changed some of the default editor settings and seen the results of the changes.

Entering SEU commands

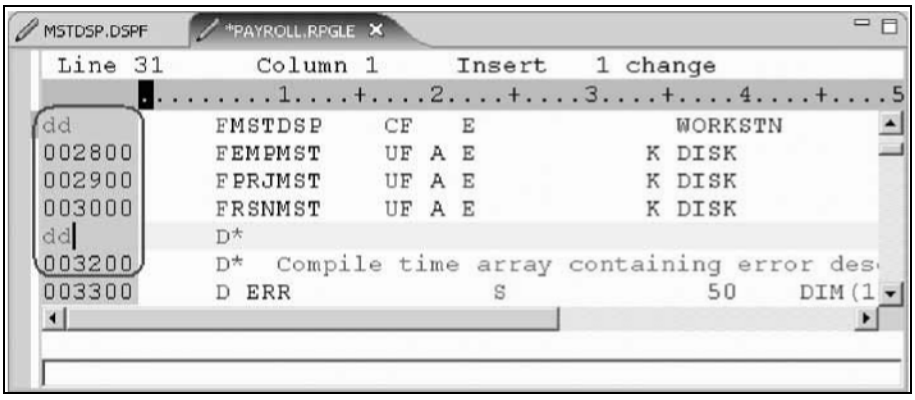
You can configure the LPEX Editor to adopt the keyboard and command personalities of many popular editors. Most editor profiles differ only in the keys and commands used to perform various editor tasks. Some base editor profiles, listed below, also add prefix information and a command area at the start of each line:

- ispf
- seu
- xedit

The editor recognizes prefix commands used by these editor profiles. Depending on which profile you are using, you can enter SEU, XEDIT, or ISPF commands when the prefix area is

active.

If you are an SEU expert you will appreciate the ability to use SEU commands. To enter SEU commands:



1. Move the cursor into the gray sequence number area to the left of the edit area
2. On any sequence number type `dd`.
3. Go down a few lines and type `dd` again and press **Enter**.

Notice that the lines have been deleted.

4. Now type `i5` in the sequence number area.
5. Make sure the cursor is within the sequence number area.
6. Press **Enter**.

Five new lines are inserted.

You have learned how to use SEU commands in the editor.

Requesting undo and redo operations

The editor records each set of changes you make to a file in the Editor window. The number of changes made since the last file save is displayed on the status line. If you want to undo a set of changes made to a file you use the Undo operation. You can also cancel the effects of an Undo operation by using the Redo operation.

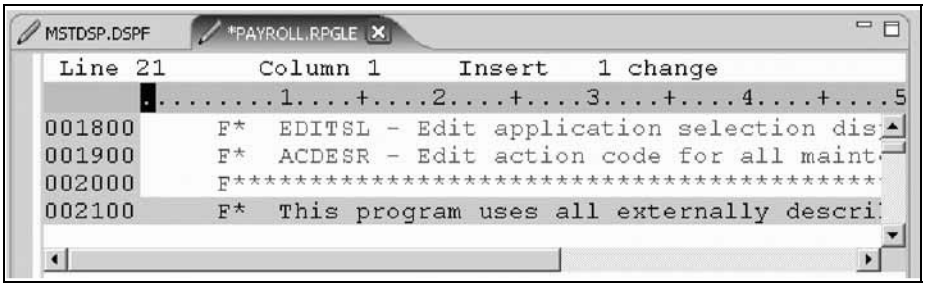
Now you are going to undo some of the changes you just made to the file. Then you will cancel the Undo operation by using the Redo operation. Finally you will reload the source so that it is back to its original content.

1. Click **Edit > Undo** from the workbench menu. Notice that the 5 new lines disappear.
2. Press **Ctrl+Z** to undo the last change. Notice that the deleted lines reappear.
3. Click **Edit > Redo** from the workbench menu. Notice that the lines are deleted again.

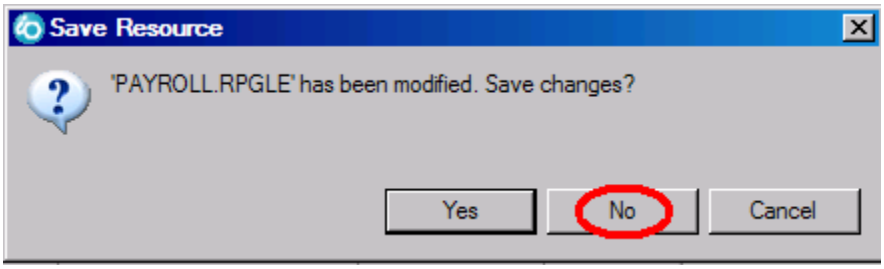
At this point you will reload the source from the IBM i server to make sure that it isback in its original form.

4. Click **File > Close** on the workbench menu.

Tip: You can also click the X on the **PAYROLL** tab.



A Save Resource dialog opens asking if you want to save the latest changes.



5. Click **No**
6. Go back to the workbench to the Remote Systems view and open the PAYROLL member in the QRPGLESRC file.

You have learned how to undo and redo changes that you made to a file.

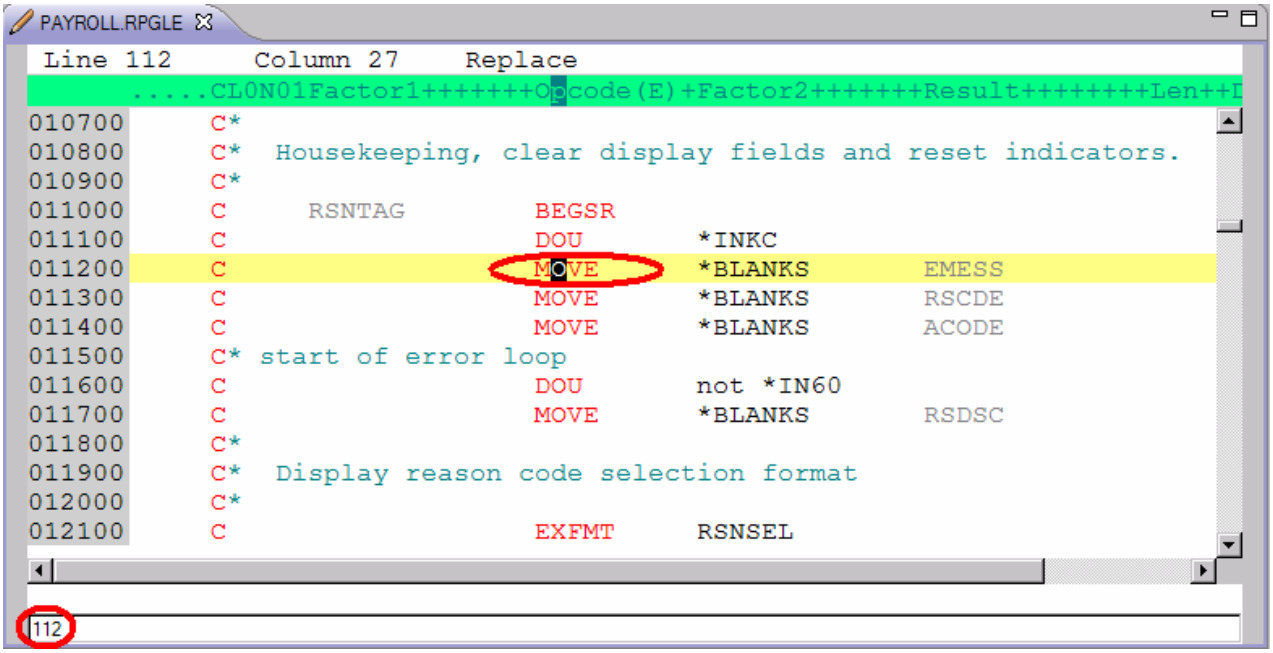
Invoking language-sensitive help

Inside the editor, there is cursor-sensitive language-reference help available.

This help is invaluable if you cannot remember the order of fields in an RPG specification or the allowed values for a variable field. This help is available from the LPEX Editor window.

To receive language sensitive help, press F1 in an Editor window. If the cursor is on an operation code, you receive help for that operation code; otherwise, you receive help for the current specification.

To access language sensitive help

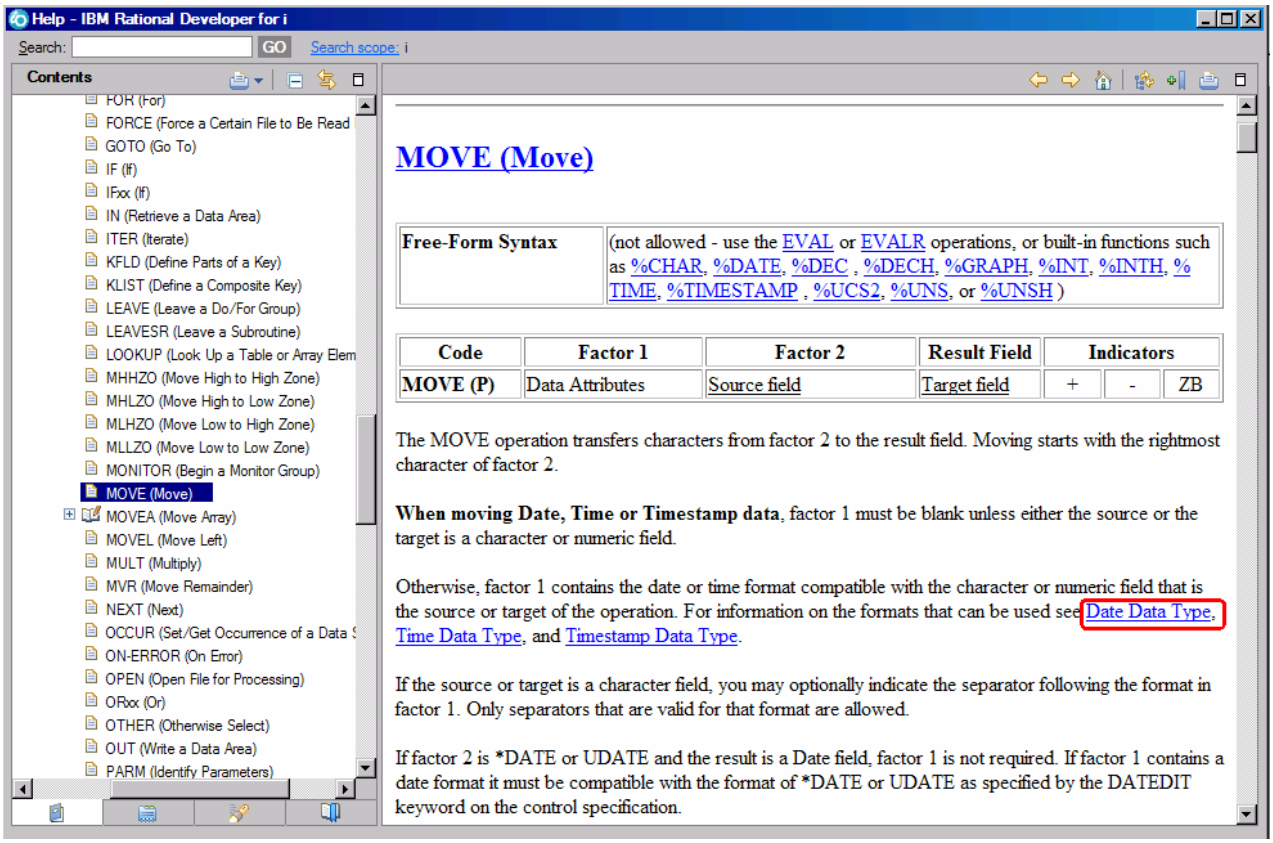


a. Position the cursor over MOVE operation code in line 112 of the ILE RPG source.

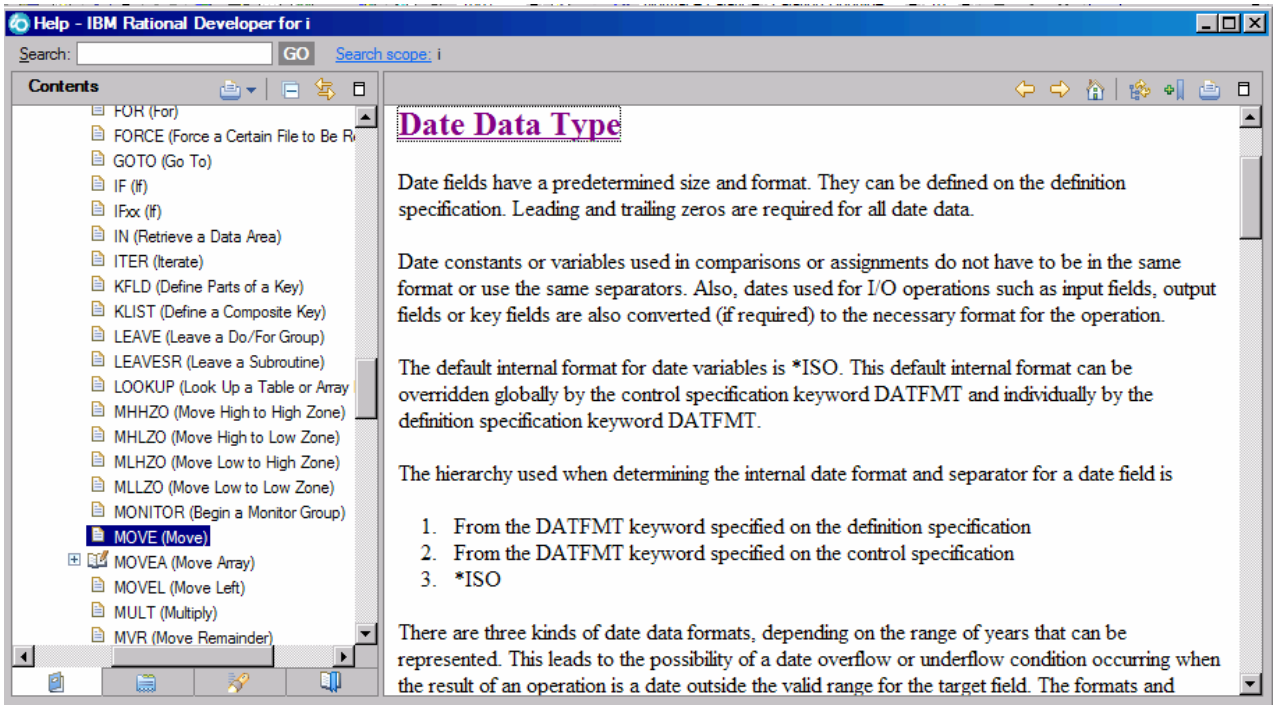
Tip: To jump to a specific line, type the number into the editor's command entry line and press Enter.

b. Press **F1**.

Language-sensitive help for the MOVE operation code appears in a Help window.
Text marked in blue in the Help window contains the link to detailed information about the topic in blue.

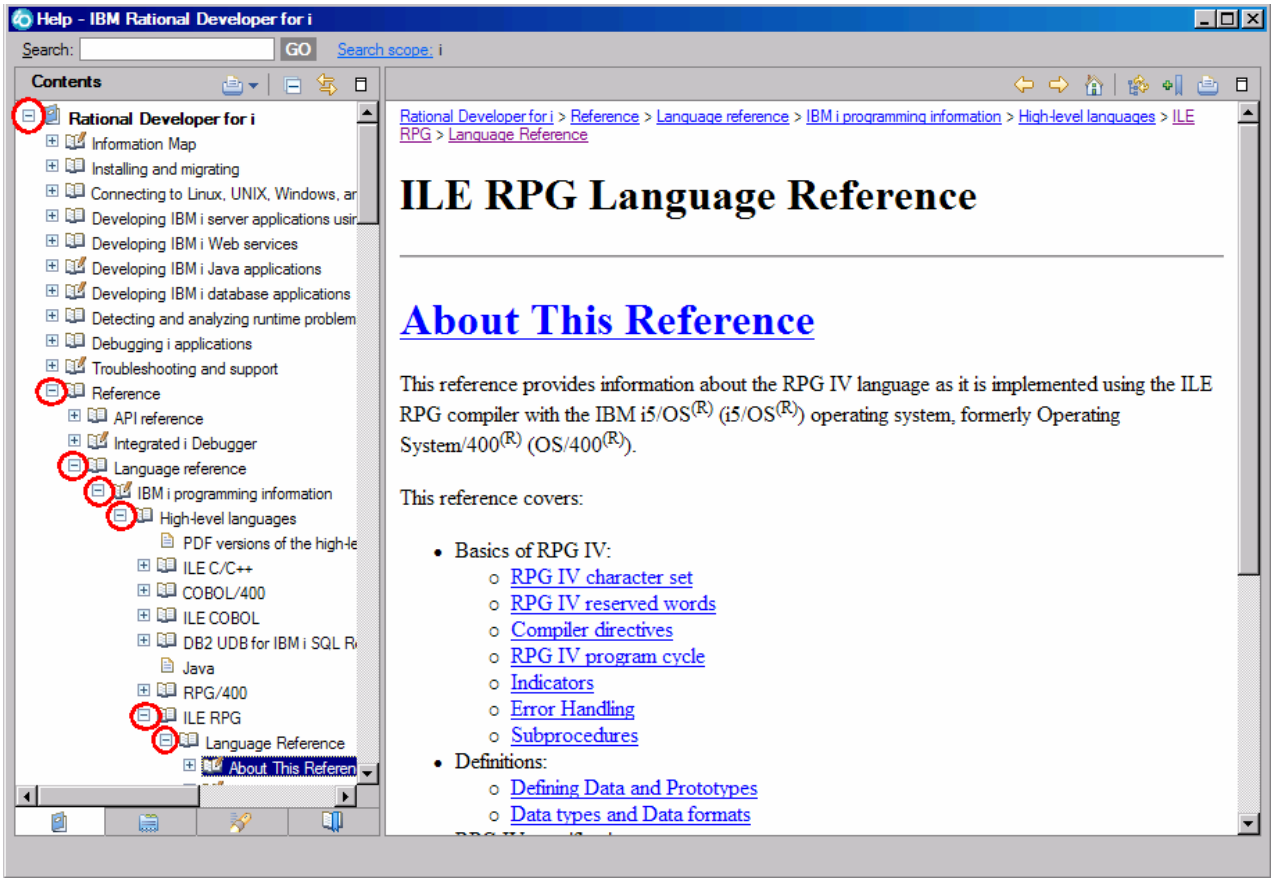


c. Click the link **Date Data Type**.
The Help page for **Date Data Type** is displayed



d. Explore the Help window to see what else is available.
e. Close the Help window.
f. Select **Help > Help Contents** on the workbench menu and expand Rational Developer for i book to see a list of all help that is available in the product.

To locate the language reference information for ILE RPG:



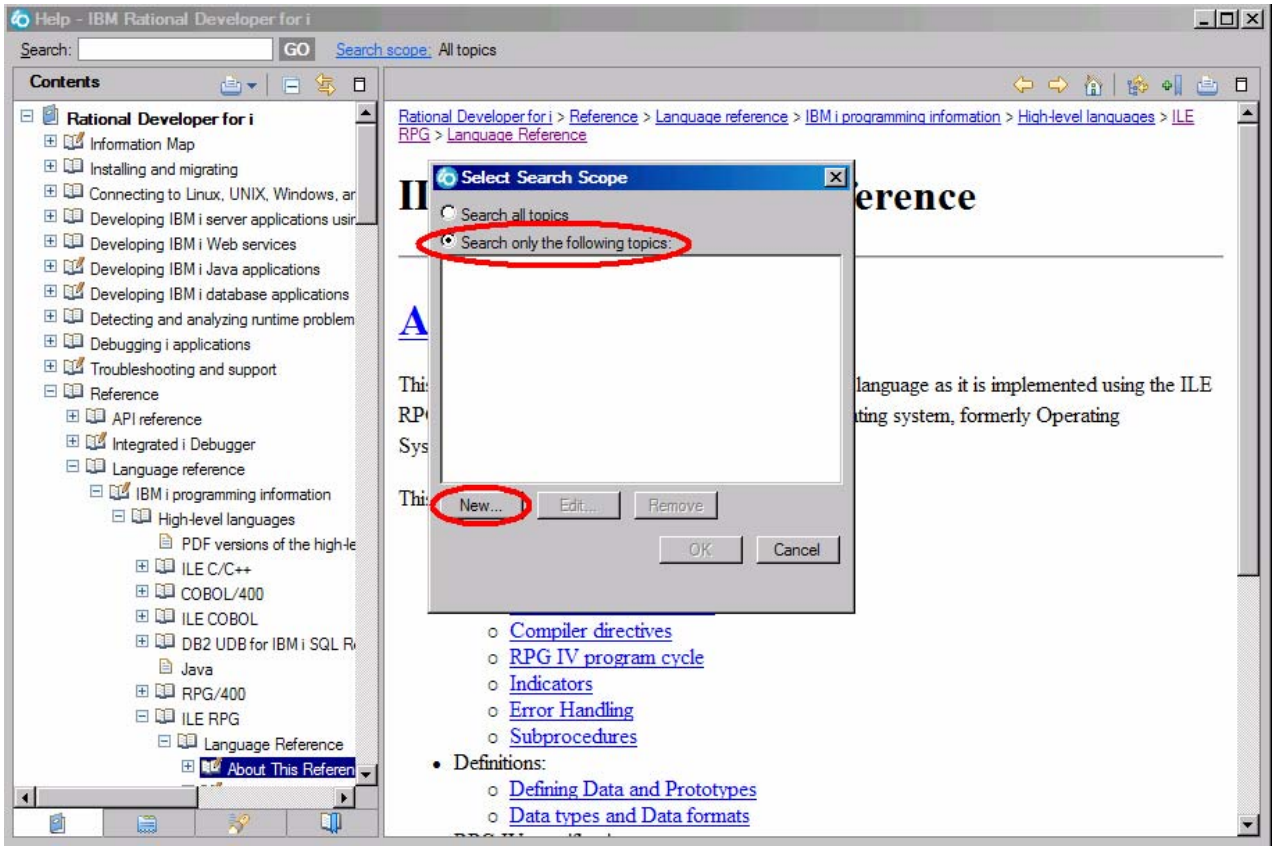
In the left pane of the Help window

g. Expand **Rational Developer for i**
h. Expand **Reference**.
i. Expand **Language reference**.
j. Expand **IBM i programming information**.
k. Expand **High-level languages**.
l. Expand **ILE RPG**.
m. Expand **Language Reference**.

Having the latest version of the manuals at your fingertips will make it easier to find programming information. There is also the option to search the help by specifying a search string. By default, the complete help will be searched.

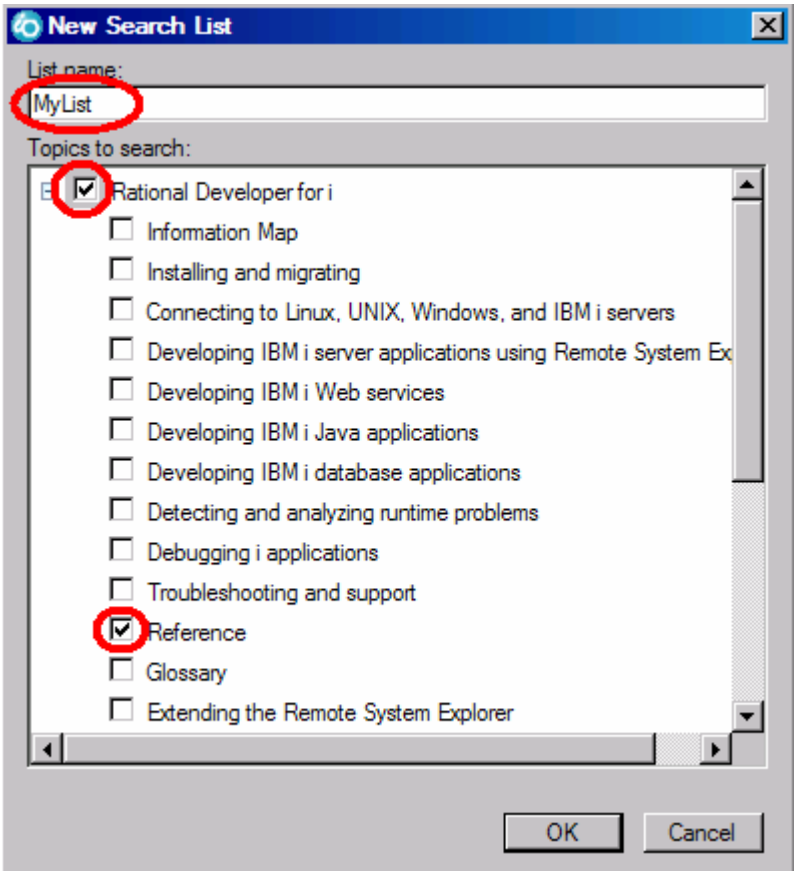
Limiting the search scope:

Tip: To limit the search to specific documents:



a. Click **Search scope**. The Select Search Scope dialog opens
b. Select **Search only the following topics** radio button
c. Click **New**.

The New Search List dialog opens.



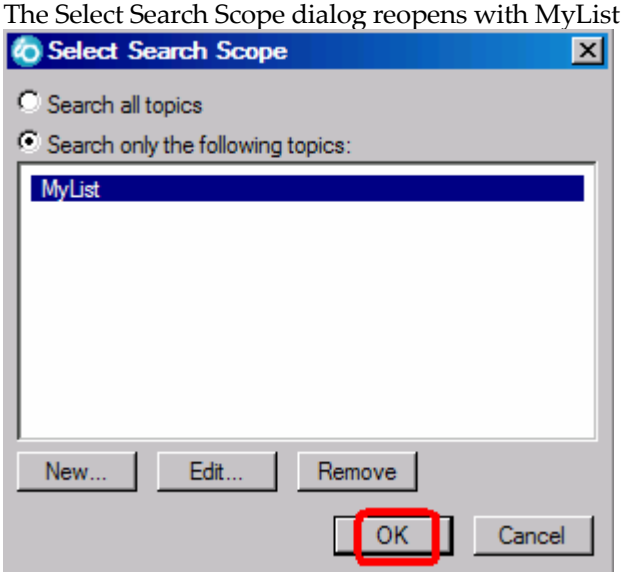
d. In the **List name** field, type `MyList` for example.

e. Expand **Rational Developer for System i** .

f. Select the **Reference** check box.

g. Click **OK** on the New Search List dialog.

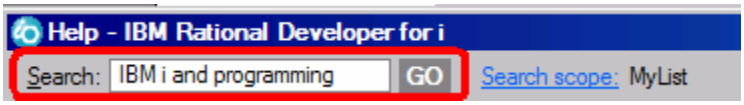
The Select Search Scope dialog reopens with MyList selected in the topic list.



h. Click **OK** on the **Select Search Scope** dialog.

i. In the **Search** field, type **`IBM i and programming`** for example

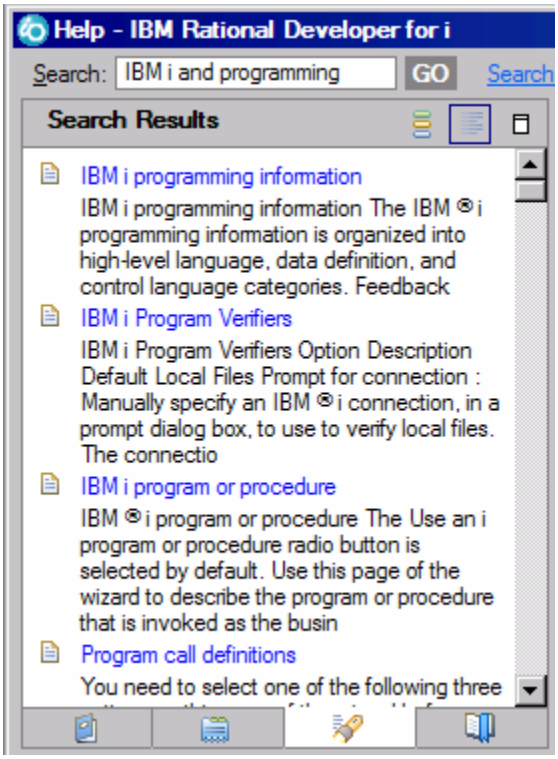
.



Searching requires a help index and it takes a bit of time to create the index in your workspace.
If you want to complete the search query

1. Click **GO**.

The search results display.



You have accessed language sensitive help

.

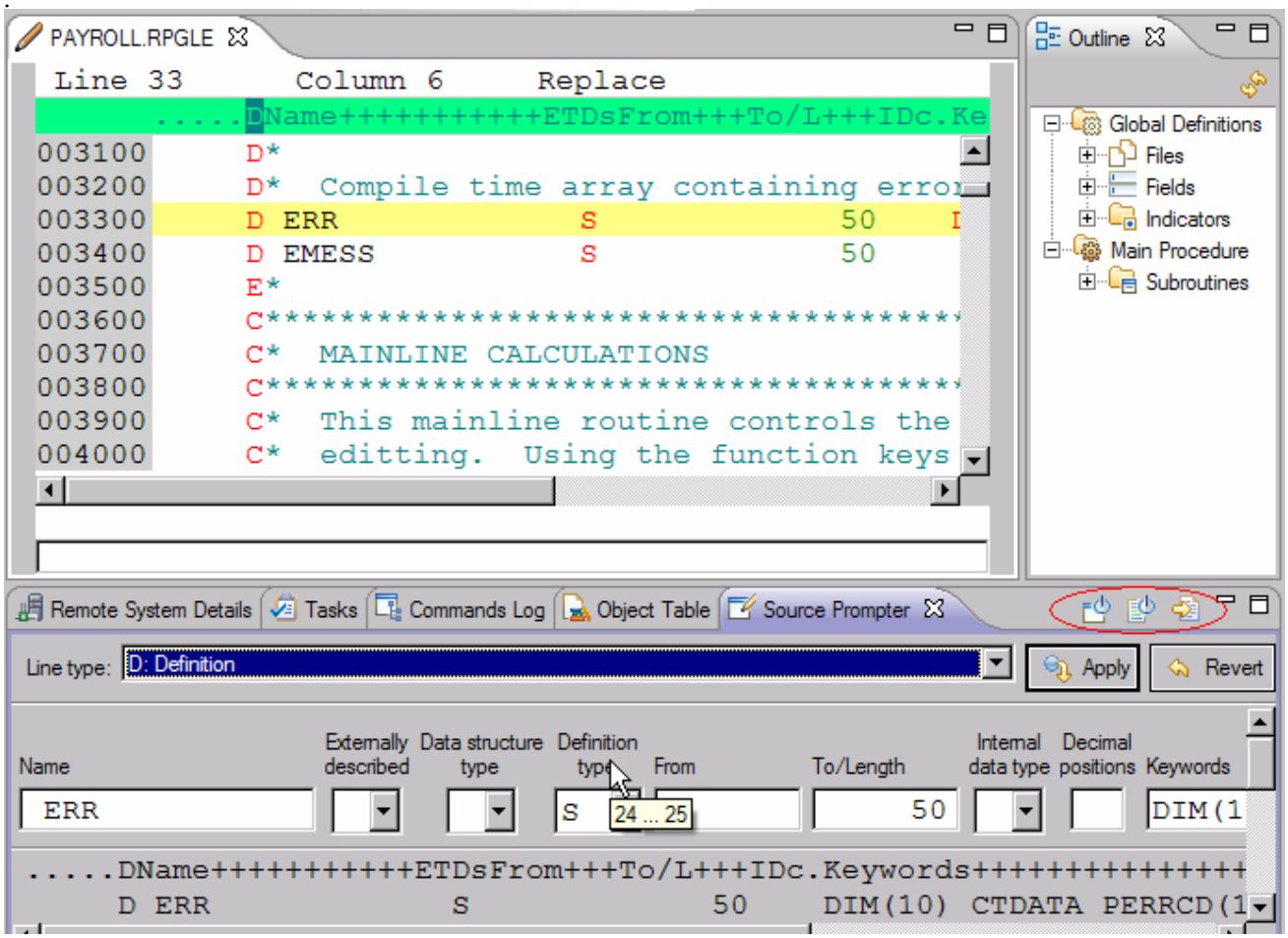
Prompting language specifications

Now back to editing the source code, let's look at prompting and more.

Instead of entering or changing code directly in the Editor window, you can use prompts. When you request a prompt for a specification line, a window appears where you can enter or change that line using entry fields.

To access prompts:
a. Return to the workbench.
b. In the Editor window move your cursor to the D-spec on line 33.
c. Press **F4** (You can also click **Source** from the workbench menu and then click **Prompt**.)
You see the Source Prompter at the bottom of the workbench. The Source Prompter shows the specification line broken down into its individual fields

.



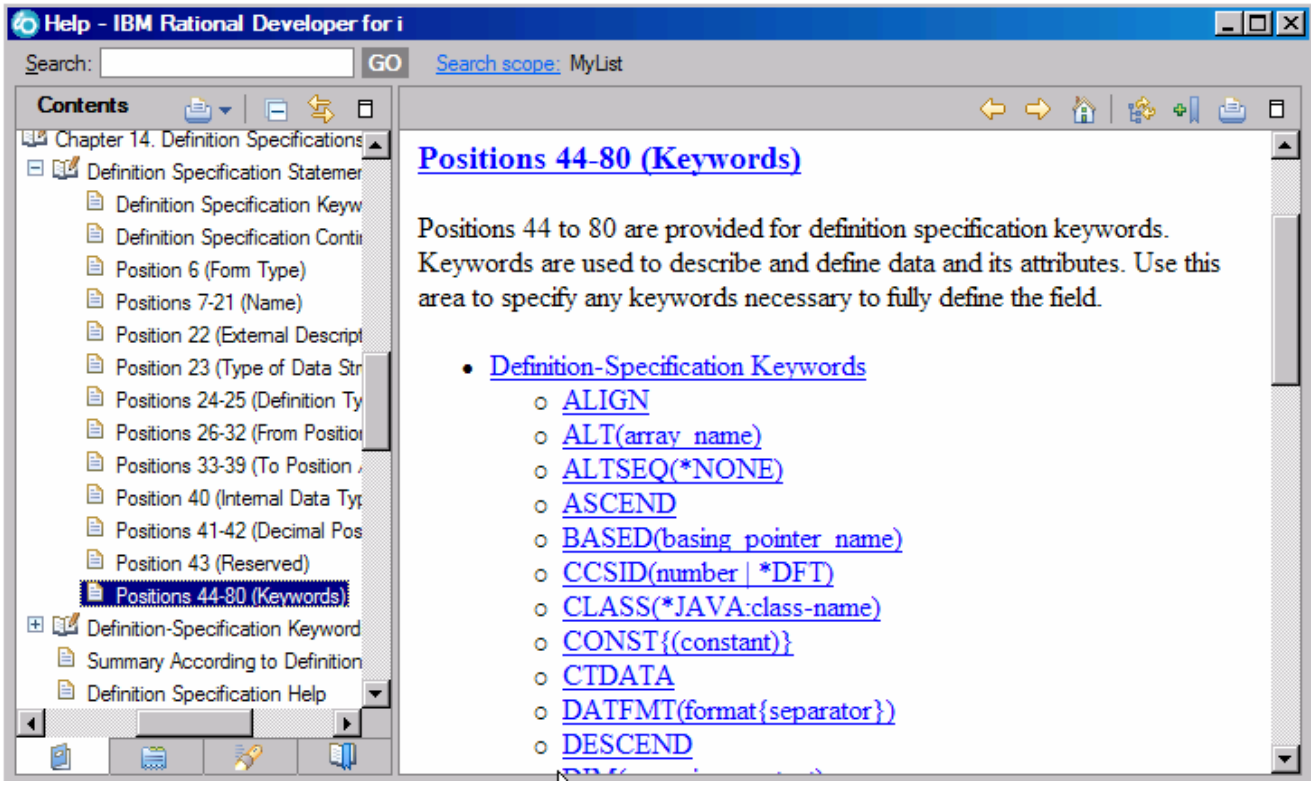
On the Source Prompter toolbar you can use the three push buttons to:

- Disable source prompt view,
- Disable syntax checking,
- Change to insert mode.

To display context sensitive help for any field in the Source Prompter:

a. Tab to the **Keywords** field.
b. Press **F1** to see help for this field.

The Help window with help for the D- spec keywords opens. If it doesn't appear automatically, you might have to bring it to the foreground by clicking on its icon on the Windows taskbar.



You will see words in the help that appear in a different color than the regular text. These are help links, and they show that there is additional help available on that word or phrase.

c. Click on any link to see specific help for that item.

d. Close the Help window.

Prompting is easy, let's look at more good stuff.

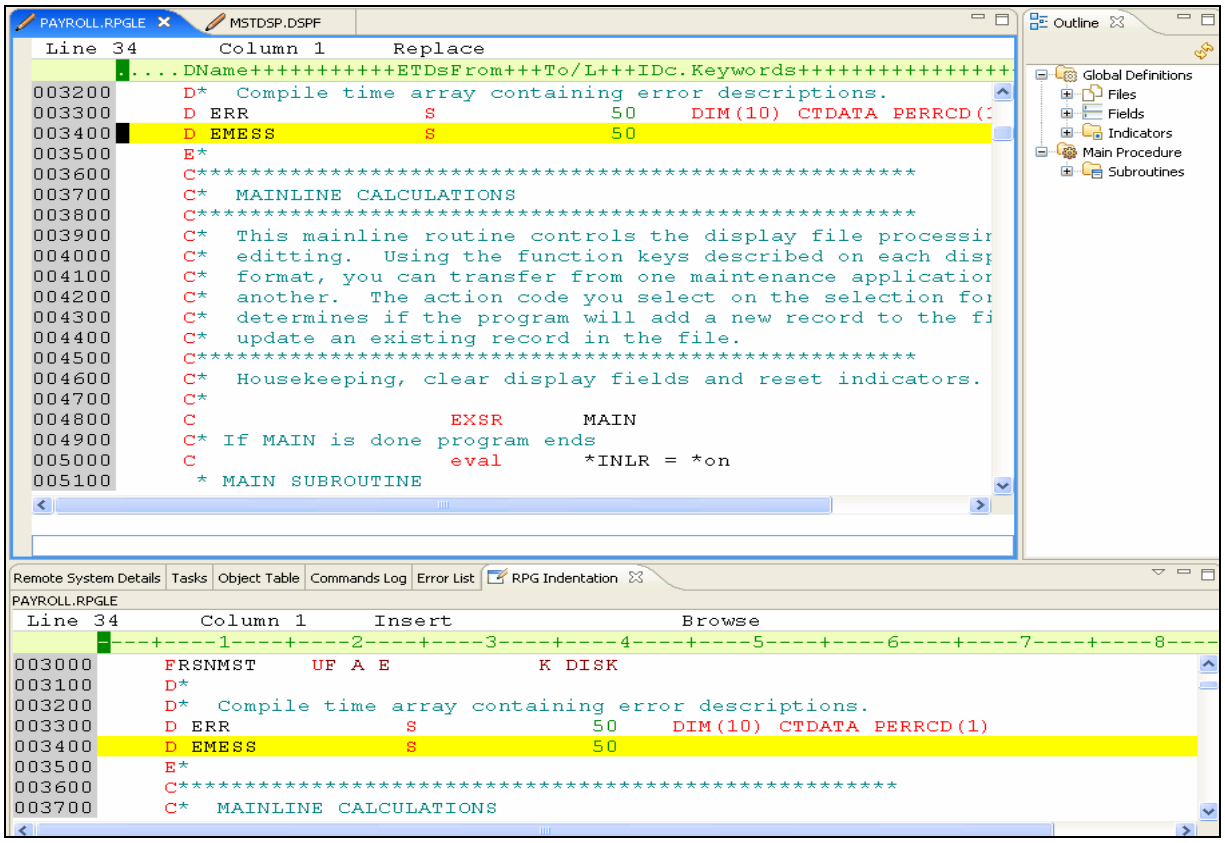
Indenting source

When editing ILE RPG source, it can be difficult to determine the beginning and ending of constructs. The RPG Indentation view allows you to see your source with constructs in an indented mode.

To indent source:

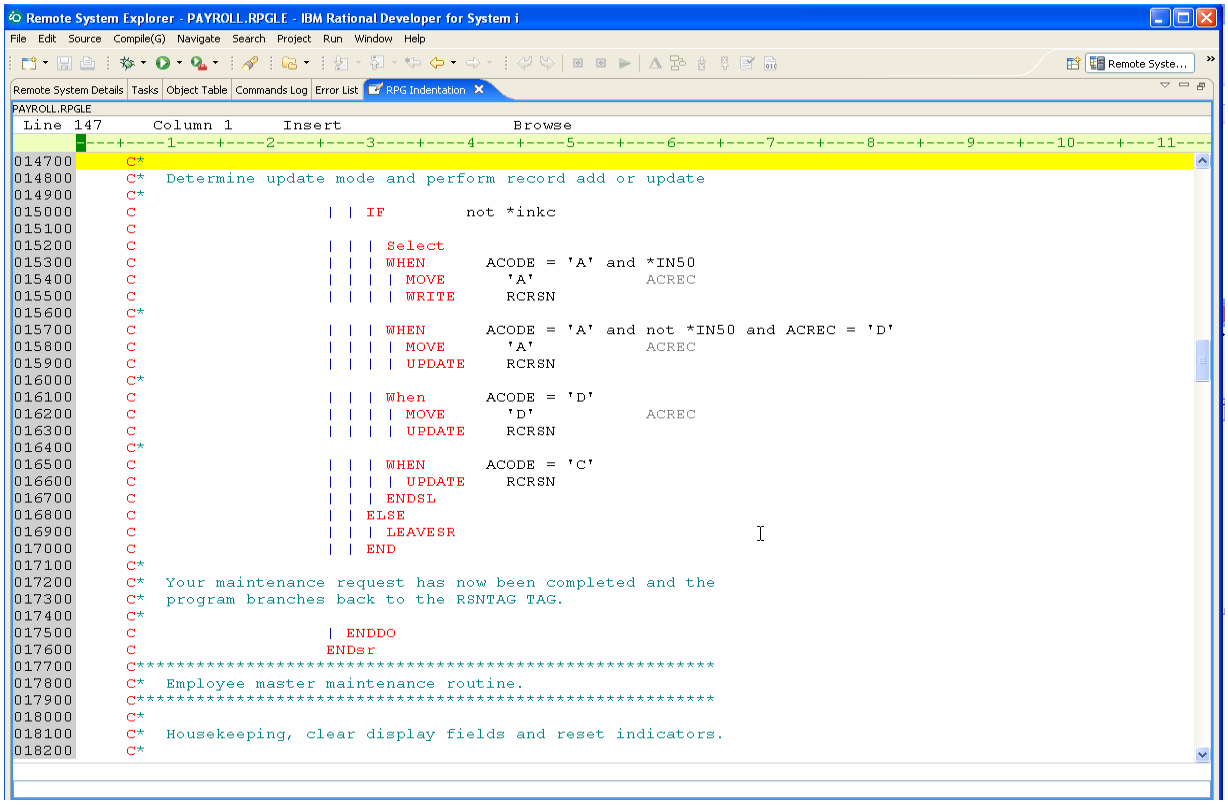
1. Click on the editor tab for the PAYROLL member to make it active
2. Click **Source > Show Indentation** on the workbench menu.

You see the RPG Indentation view below the edit dialog



You can display the RPG Indentation view as a full view

3. Double-click **RPG Indentation** tab



4. Go to line 150.

The line shown on the status bar is the cursor position. In the Indent view you see some nested conditions with indented lines. As you will notice this helps to recognize the beginning and ending of these conditions.

Tip: The RPG Indentation view is Browse mode only and cannot be edited.

5. Click the **X** in the top right corner of the Indentation view to close it.
6. Double-click any of the tabs to show the editor again.

You have used the RPG Indentation view to see the beginning and ending of constructs.

Finding and replacing text

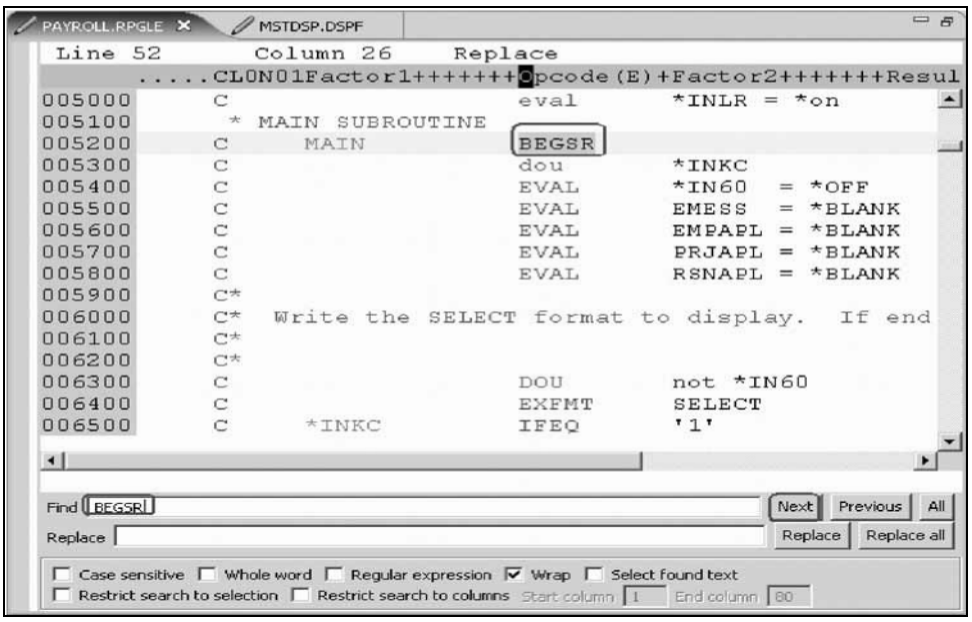
The LPEX Editor also has a powerful find and replace text feature. You use the Find and Replace window to search for an item. You can search for a word, a partial word, or a sequence of such. You can also enter a pattern you want to match, provided that the pattern follows the rules of regular expression. You can replace the found search item. If the entered text or pattern is found, the cursor moves to either the next or previous occurrence of the search item, according to your chosen search direction, and replaces the found text according to your selections.

To find and replace text:

1. Click anywhere in the editor to give it focus then press **Ctrl+Home** to go to the top of the file.

Tip: When you press **Ctrl+Home** to go to the top of a file or **Ctrl+End** to go to the bottom of a file, a quick mark is set at your cursor position. This allows you to return to that line by pressing **Alt+Q**. **Ctrl+Q** will set a quick mark.

2. Click **Edit > Find/Replace** from the workbench menu or press **Ctrl+F**. The Find/Replace window opens at the bottom of the Editor window



At the bottom of this window, you will notice that you have some options to select from, for example, search only in certain columns. You want to find the first occurrence of BEGSR.

3. In the **Find** field, enter BEGSR to find the start of a subroutine.

4. Make sure the **Replace** field is blank.

You would use this field for text replacement.

The Editor moves the active line to line 52, which contains the first `BEGSR` phrase in the file.

5. Click **Next** to go to the next location of `BEGSR` in the file
6. Click in the Editor window to close the Find/Replace window.

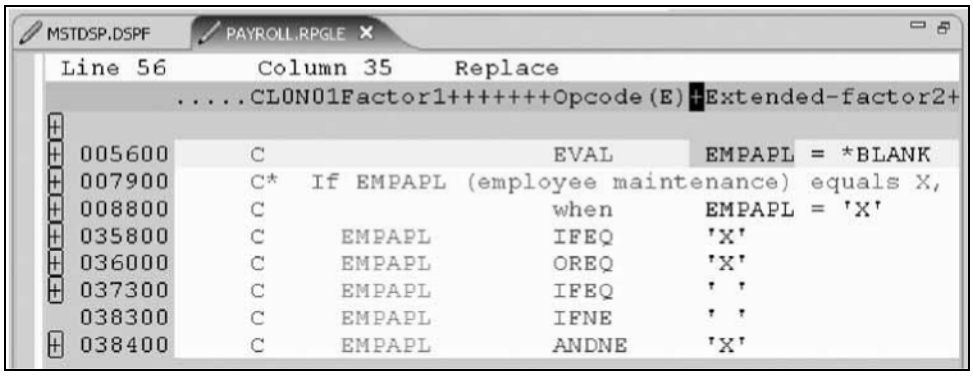
You have searched for an item in your source using the Find/Replace window.

Filtering lines by string

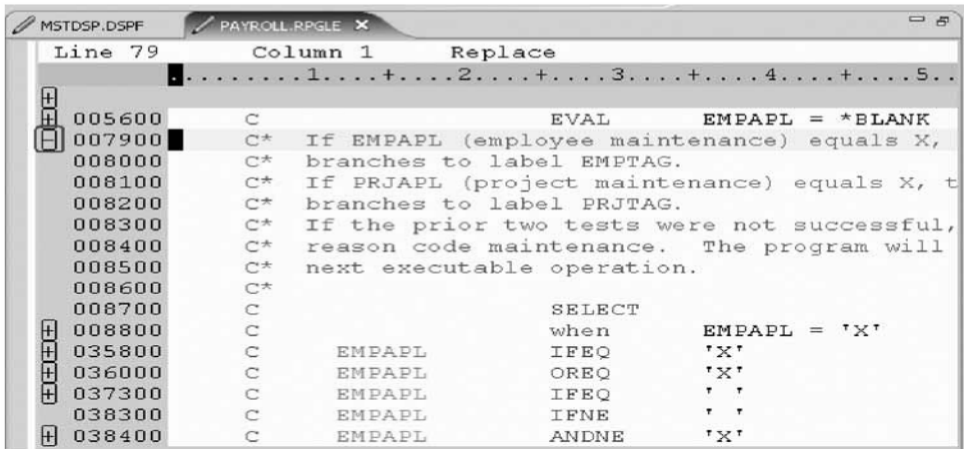
The editor allows you to filter or subset your source so that you see only lines containing a given string. Filtering lines makes it quick and easy to find lines without having to scroll through your source.

To filter source by string:

1. Double-click the variable EMPAPL in the Editor window
2. Select **Edit > Selected > Filter Selection** from the workbench menu.



4. Move the cursor down a few lines to line 79.
5. Expand line 79. This expands the section up to the next instance of EMPAPL.



Now you want to show the entire source again.

6. Click **Edit > Show all** from the workbench menu or press **Ctrl+W**.

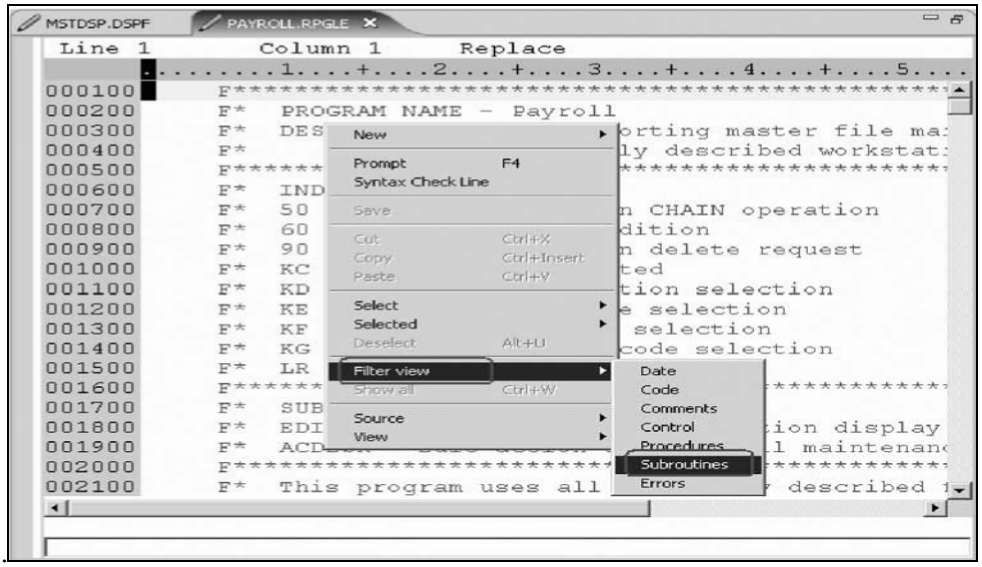
Your cursor is still positioned on the same line that you moved the cursor to, even though all lines

are now showing.

You have filtered your source so that you see only lines containing a given string.

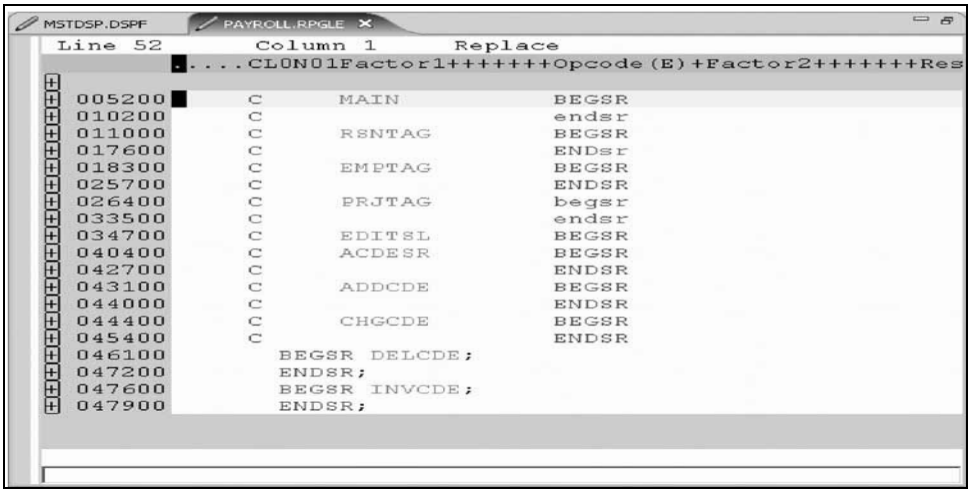
Filtering lines by type

To help you navigate quickly through your ILE RPG source the editor allows you to filter lines based on the line type. Imagine you want to see where all the subroutines are defined in your source.



To filter lines by type:

1. Right-click in the Editor window with the PAYROLL program.
2. Click **Filter view > Subroutines** on the pop-up menu.
   All source lines with BEGSR or ENDSR are displayed allowing you to move quickly and easily to the desired subroutine in your file.



3. Move your cursor to the line with the subroutine declaration CHGCDE (line 444).
4. Expand the declaration to show all lines in this subroutine. Now you could work with the source inside this subroutine.

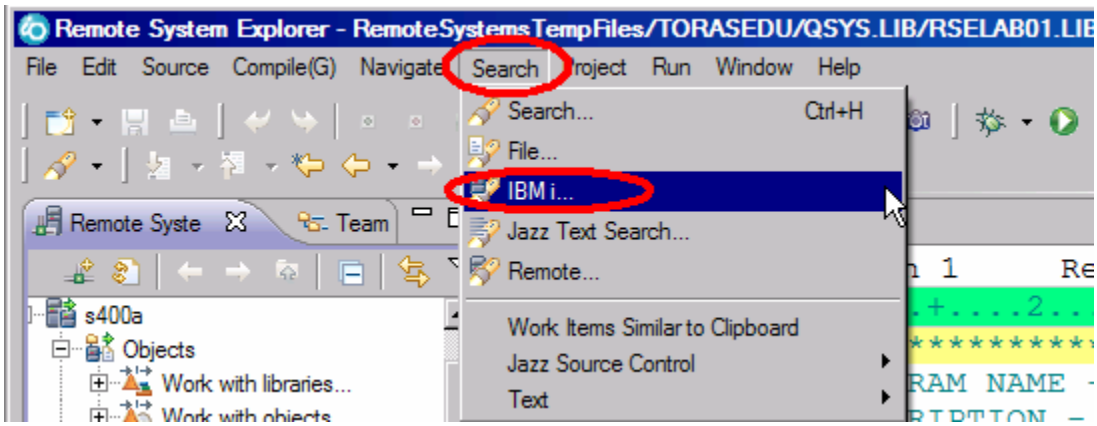
5. Right-click in the Editor window and click **Show all** on the pop-up menu.

You have filtered lines in your source by line type.

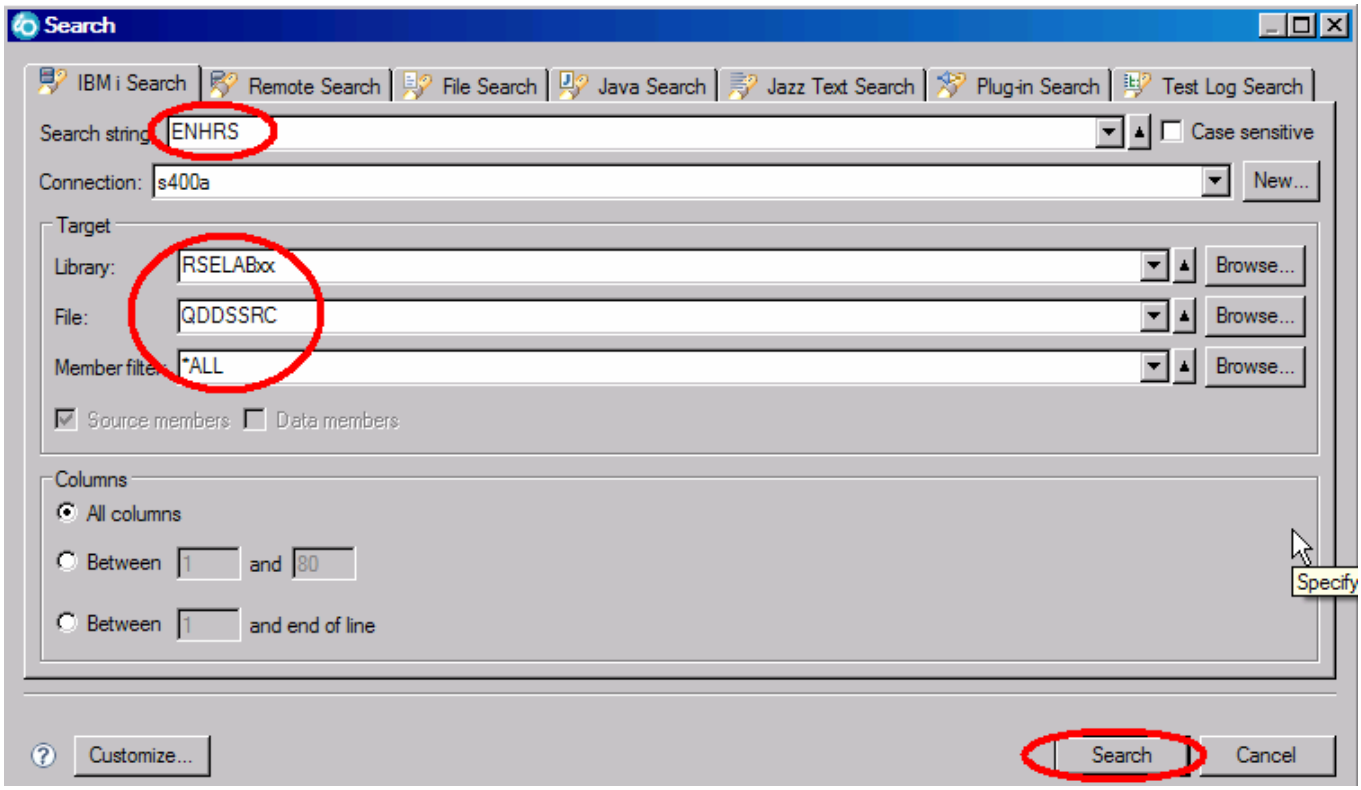
Searching multiple files

If you would like to search through the members in a source physical file or through the files in a local directory, you can use the Search tool. The Multi-File Search utility allows you to search for a particular string of text in many members on the host. This function can also be used on local files.

To search multiple files:



1. Click **Search > IBM i** from the workbench menu.
The Search window opens

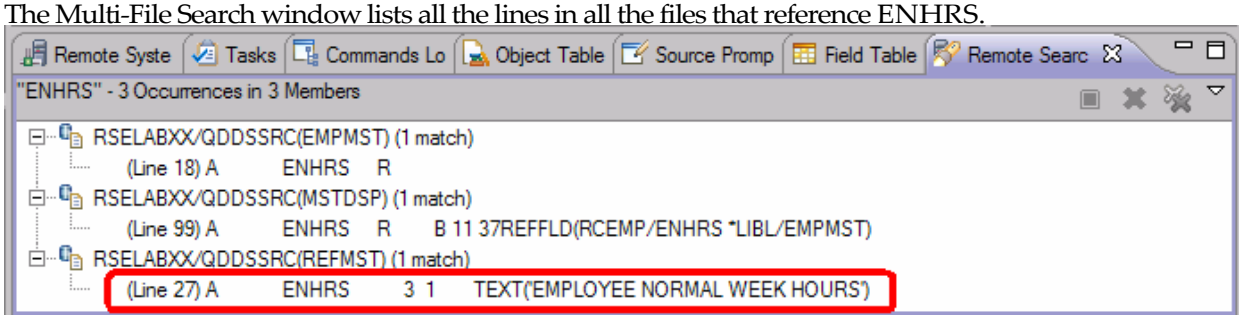


2. In the **Search string** field, type ENHRS.

The **Connection** field should contain your IBM i server name, otherwise enter it there.

3. Under **Target** in the **Library** field, type RSELABxx.
4. Under **Target** in the **File** field, type QDDSSRC to search all members in this source physical file.
5. Under **Target** in the **Member** field, select ***ALL**.
6. Click **Search**.

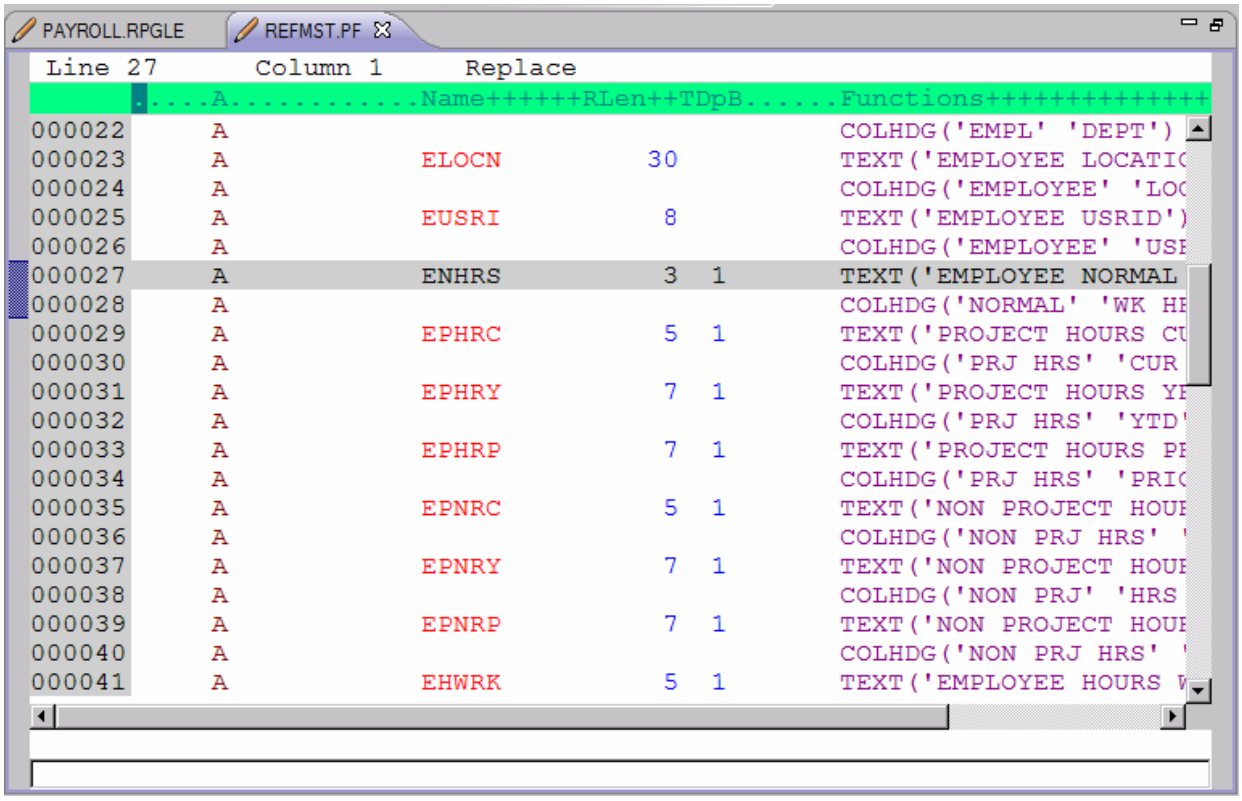
The Multi-File Search window lists all the lines in all the files that reference ENHRS.



7. Double-click the last line in the list at

```
A            ENHRS 3 1          TEXT('EMPLOYEE NORMAL WEEK HOURS')
```

The member REFMST is automatically loaded into the editor and the cursor is placed on the correct line.



8. Click the X in the **REFMST** tab to close the REFMST file.

You have searched through members in a source physical file.

Comparing files from the Remote Systems view

If your product undergoes many changes, you will find the Compare utility useful. It allows you to compare different versions of a program and find the differences.

Using the compare utility in the workbench you can view the differences between two files by comparing them. You can compare different files, and you can compare versions in the workbench with versions in the repository or with the local edit history.

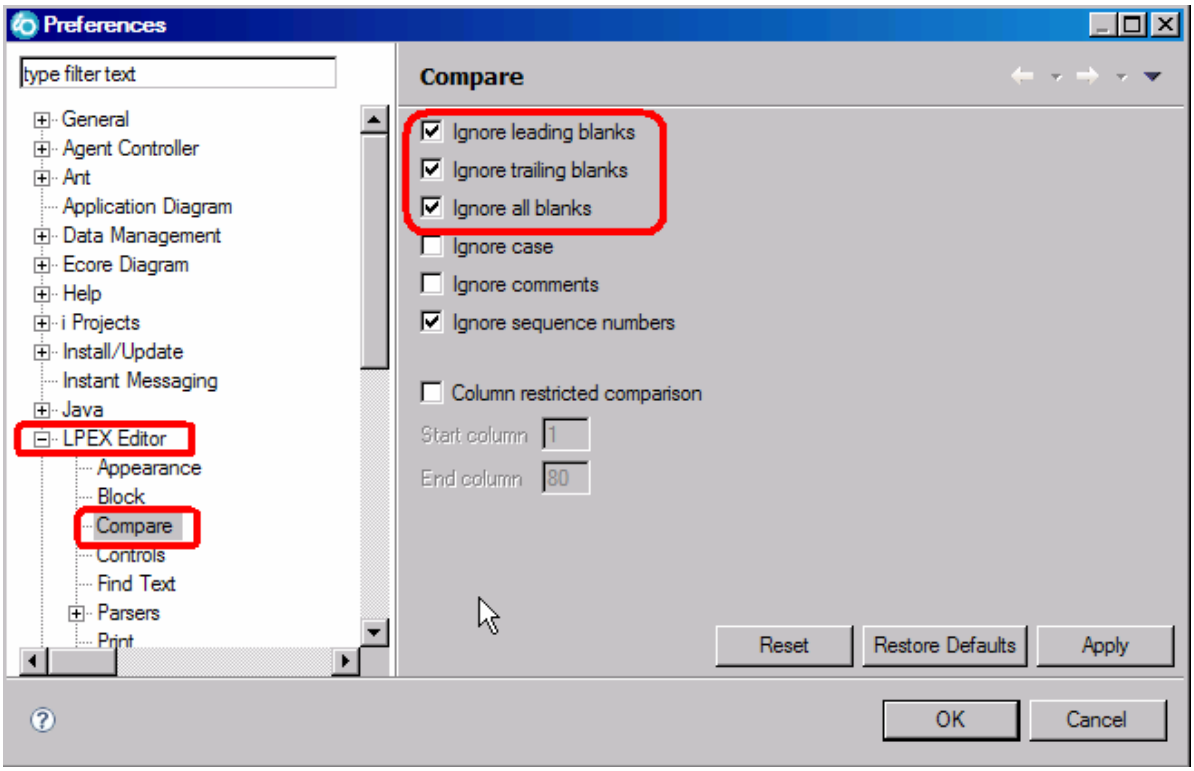
After a comparison is carried out, the Compare Editor opens in the editor area. In the compare Editor, you can browse through all the differences and copy highlighted differences between the compared resources. You can save changes to resources that are made in the comparison editor.

Tip: Make sure all lines show in the source before starting the Compare tool.

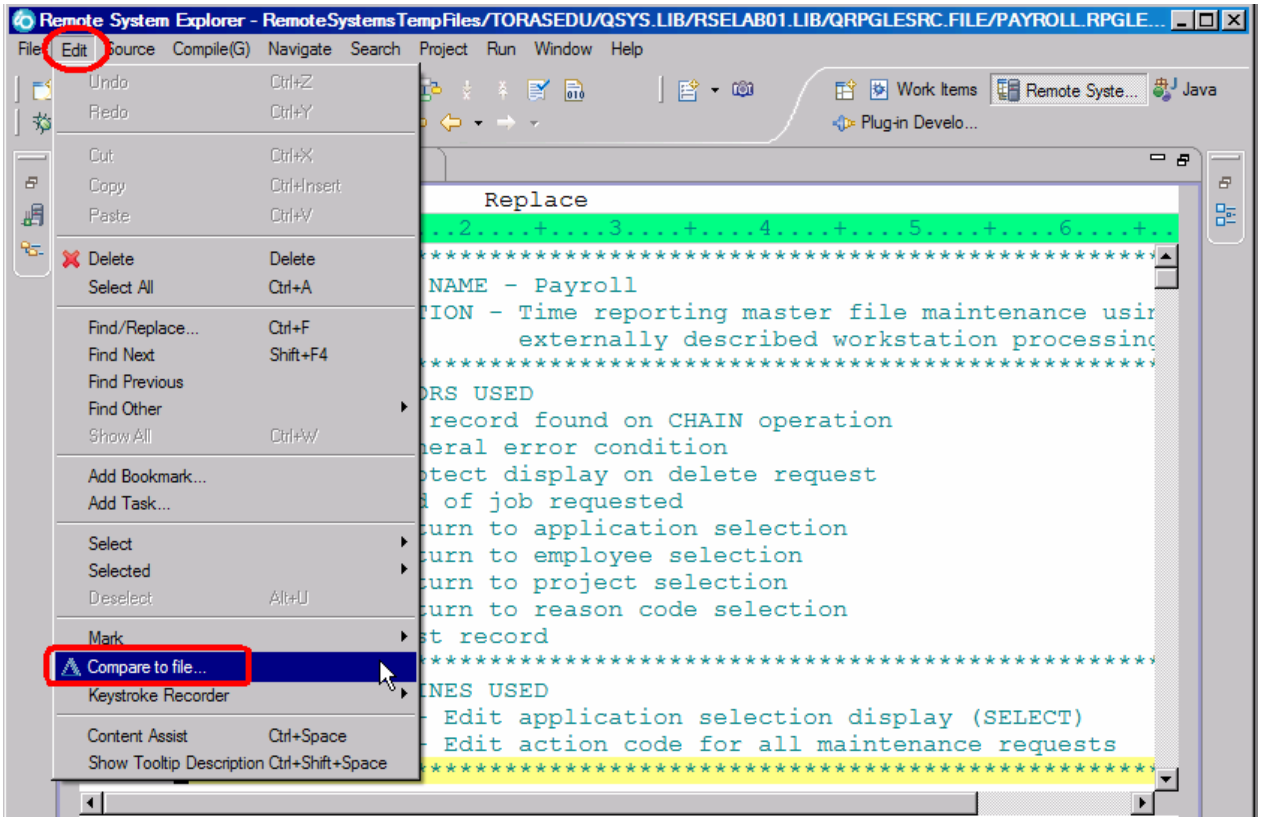
To compare files in the workbench:

First let's setup some preferences to get the best results from the Compare tool

1. Click **Window > Preferences** from the workbench menu. The Preferences window opens.



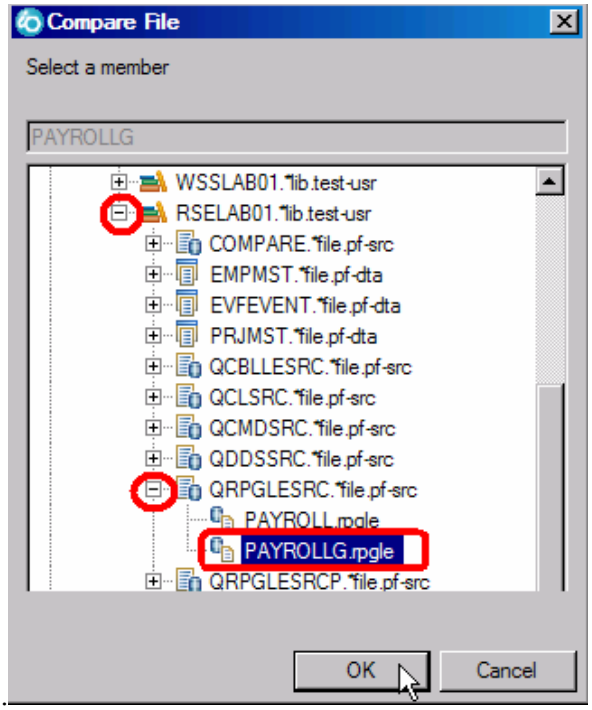
2. In the left pane of the Preferences window, expand **LPEX Editor**.
3. Click **Compare** under **LPEX Editor**. In the right pane of the Preferences window make sure that the **Ignore all blanks** check box is selected.
4. Click **OK** in the Preferences window.
5. Back in the Editor window of the PAYROLL member double-click the **PAYROLL** tab.



6. Click **Edit > Compare to file** on the workbench menu.

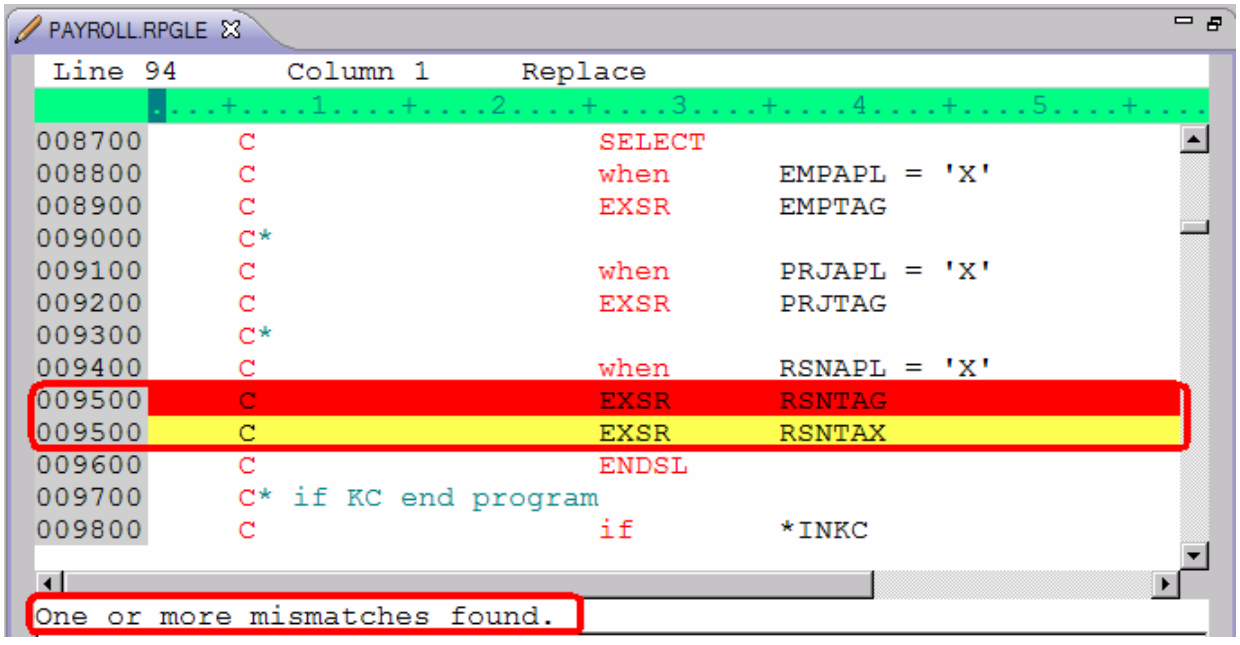
The Compare window opens.

7. Expand your connection.
8. Expand *LIBL.
9. Expand RSELABxx.
10. Expand QRPGLESRC.
11. Select member PAYROLLG.
12. Click **OK.**



The editor now will show the differences of these two members PAYROLL and PAYROLLG

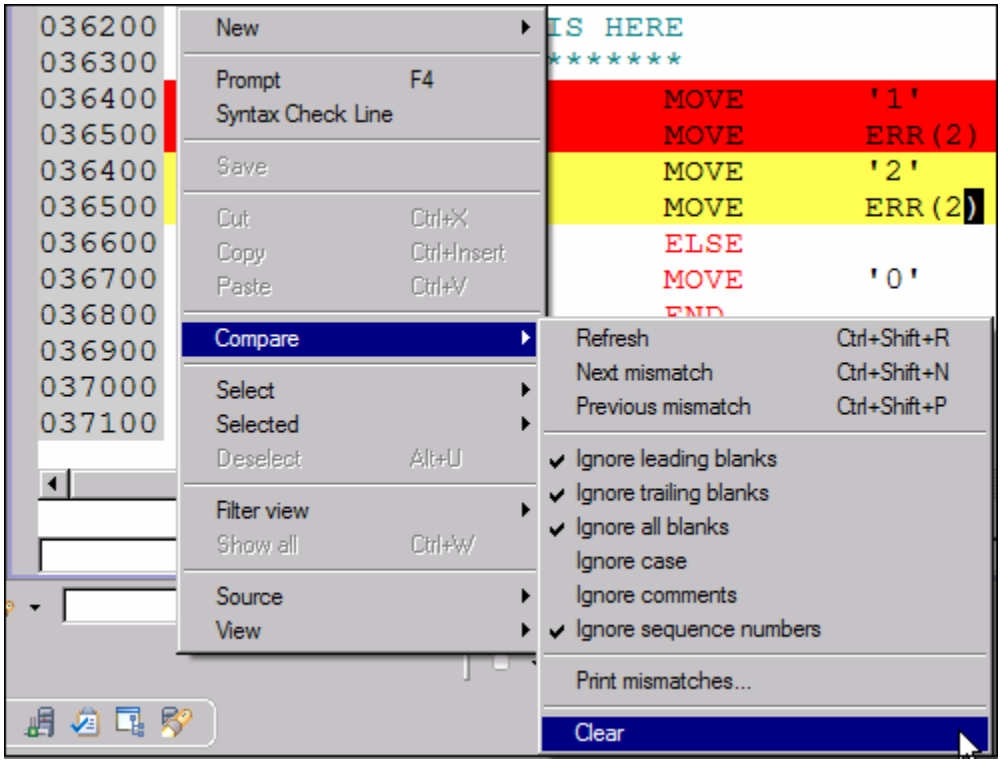
You can move from mismatch to mismatch by right-clicking the source and selecting **Compare -> Next Mismatch**, or by using Ctrl+Shift+N.
Mismatches in PAYROLL and PAYROLLG are highlighted in different colors so that you know where the mismatched lines are in each file.



13. Click **Ctrl+Shift+N** to find the next mismatch.

Next, end the compare session

14. Right-click the source and select **Compare > Clear**.



You have compared different versions of the program and found the differences.

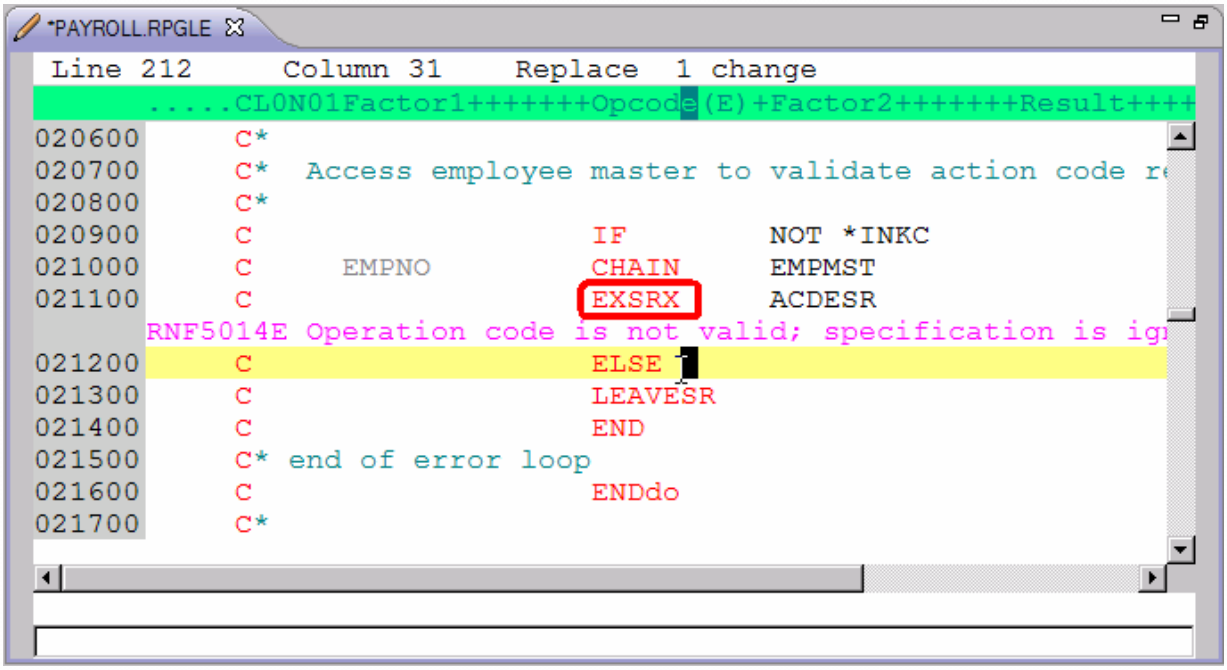
Checking syntax

One of the powerful features that the LPEX Editor shares with SEU is its ability to syntax check your source. Syntax checking can be done either when the cursor leaves each line of source or all at once on either the currently selected source or on the entire source member.

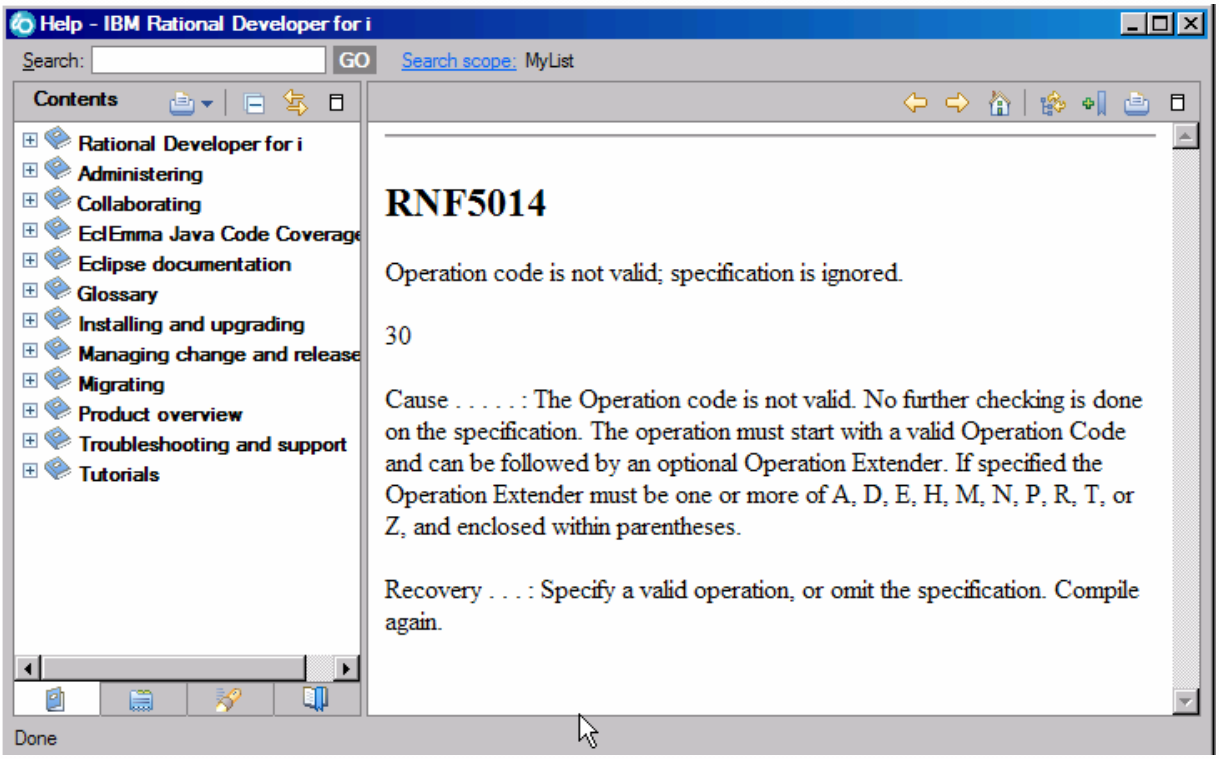
Now you will create a syntax error and watch for the prompt to correct it. To syntax check the file:

1. In the PAYROLL Editor window move the cursor to line 211 which contains EXSR ACDESR. Type the line number in the sequence number column, or scroll down.
2. Append an `X` to the `EXSR` op-code to make it `EXSRX`.
3. Move the cursor off of the line.

An error message appears to draw attention to the error.



4. Move the cursor onto the pink error message.
5. Press **F1.**



This opens a window with second level help for the error.

6. Close the Help window.
7. Change EXSRX to EXSR to correct the error. (Similarly, you can use **Edit > Undo** to correct this).
8. Move the cursor off the line you just fixed.
   The error message is automatically removed from the editor.

Tip: You can toggle automatic syntax checking. Click **Window > Preferences** from the workbench

menu and then expand **Remote Systems > Remote Systems LPEX Editor > IBM i Parsers**. Now, select the language you want to change the settings for in the left pane ofthe Preferences window, select or deselect the **Automatic syntax checking** check box and then click **OK**.

Tip: You can syntax check the whole source member currently in the editor by clicking **Source > Syntax Check All**.

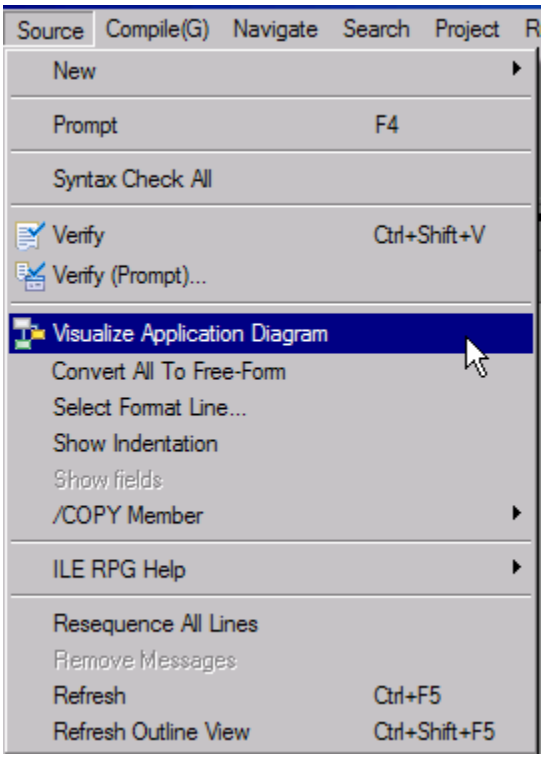
Understanding your program code with the Application Diagram

In this lesson, you learn about the Application Diagram and, how to use it to view the program topology, locate a subroutine and open the editor with the source for this subroutine.

The Application Diagram provides a graphical view of the different resources in an IBM i native application and their relationships to each other. There are two different diagrams that you can look at in the Application Diagram view:

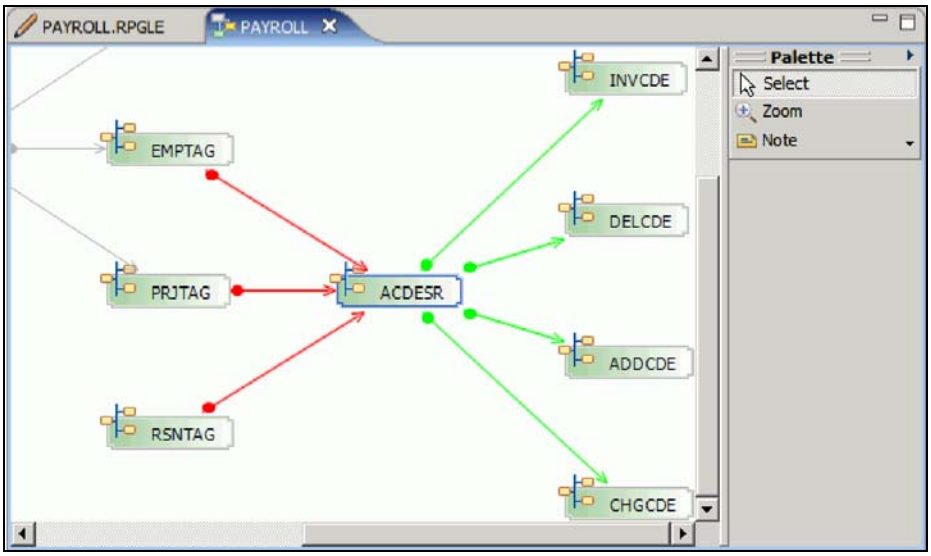
- Source Call Diagram
- Program Structure Diagram.

The Source Call Diagram takes ILE RPG source as input and displays a call graph showing subroutine and procedure calls. The Program Structure Diagram takes program and service program objects as input and displays the binding relationships between them as well as the modules bound into each program and service program.

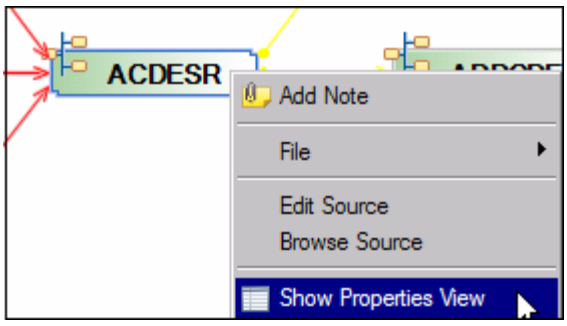


1. Open the **PAYROLLG.rpgle** source member in the **QRPGLESRC** source file.
2. On the workbench menu, click **Source > Visualize Application Diagram.**

The Application Diagram opens.

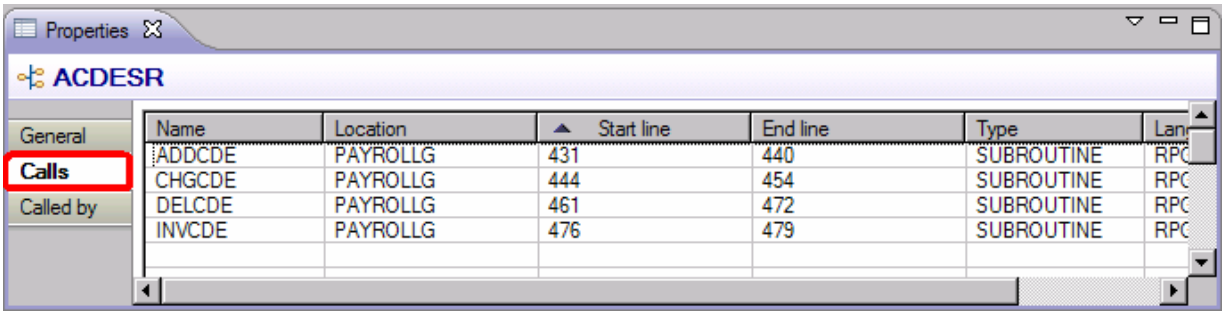


3. Select the subroutine ACDESR.
.Notice that the box is now highlighted and all relationship arrows are highlighted. Arrows pointing away from the subroutine point to subroutines that ACDESR calls and arrows pointing to the subroutine ACDESR indicate calls to ACDESR.



4. Right-click the subroutine ACDESR and select **Show Properties View**. The Properties view opens, and displays information about the currently selected subroutine.

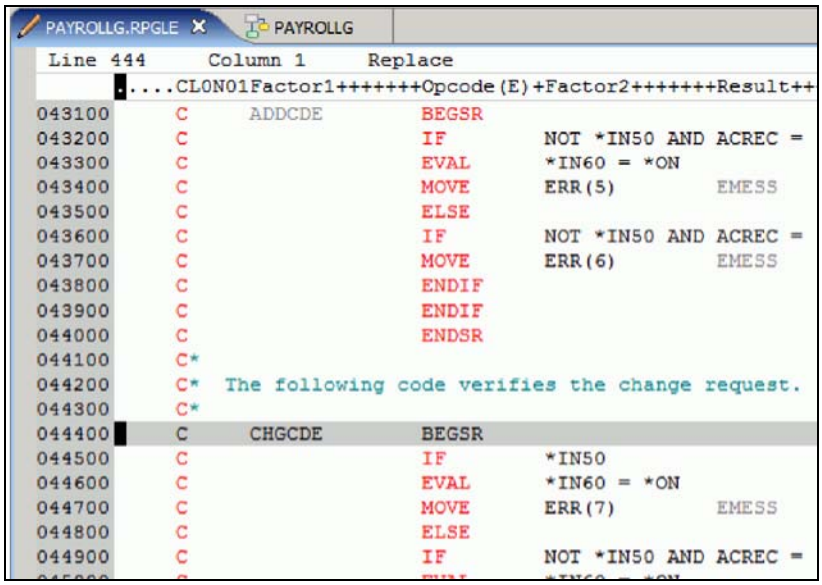
5. Select the **Calls** tab to view a list of the respective subroutines which the subroutine ACDESR calls and the lines where the subroutines are located.



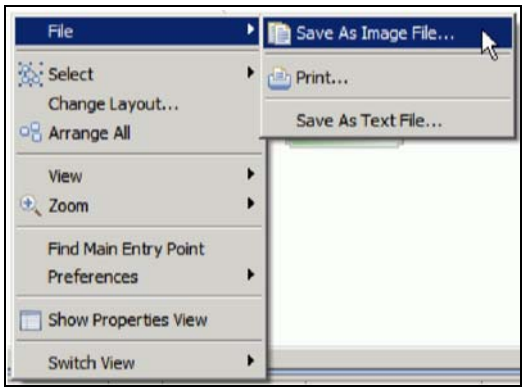
6. The list of subroutines can be used to quickly jump to the specified line in the source. Likewise, the **Called by** tab shows the list of subroutines which call the ACDESR subroutine.

7. In the **Calls** tab, double-click the CHGCDE subroutine.

This opens the editor with the member that contains the subroutine CHGCDE.



5. Return to the Application Diagram view by selecting its tab.
You can also save the application diagram as a text or image file.



6. Right-click inside the Application Diagram view, to save the diagram as a text file.
7. Right-click the white space outside the diagram boarder.
8. Select **File > Save As Image File** or **File > Save as Text File**

Depending on your selection, the application diagram will be either saved as an image file, or a text file in the directory you specify.

Module summary

In this module, you learned how to use the Application Diagram and the LPEX editor's different features.

Lessons learned

- Change the default settings of the LPEX Editor Parsers
- Change the color settings and font used by the Editor
- Change the default behavior of the Enter key
- Use SEU commands to edit source
- Undo and redo source changes

- View language sensitive help for the MOVE operation code
- View a list of all help contents
- Limit the search of help to specific documents
- Search the help
- Request a prompt for a specification line
- Display context sensitive help for any field in the IBM i Source Prompter
- Use the RPG Indentation view to see the beginning and ending of constructs in your source
- Use the Find and Replace window to search for an item in your source
- Filter or subset your source
- Filter lines based on line type
- Search through members in a source physical file
- Compare different versions of a program and identify the differences
- Syntax check source by line
- Use the Application Diagram to display a program outline.

Verifying and compiling source

This module teaches you how to verify and compile RPG in the Remote Systems LPEX Editor. When errors are found by either the verify or the compile step, the Error List appears. The Error List is a powerful tool that manages errors found by verify and compile utilities. You will become familiar with these tools, the various capabilities of the Error List and the RPG program that you have created.

Learning objectives

- Check for semantic errors on your workstation
- Start the Program Verifier tool
- Use the Error List to locate each error in the source
- Use content-assist to fix an error
- Save your source
- Re-verify source
- Change compile preferences
- Invoke the compile command
- Change the current library using the Command field in the Object Table view
- Start an interactive connection
- Invoke the Payroll program

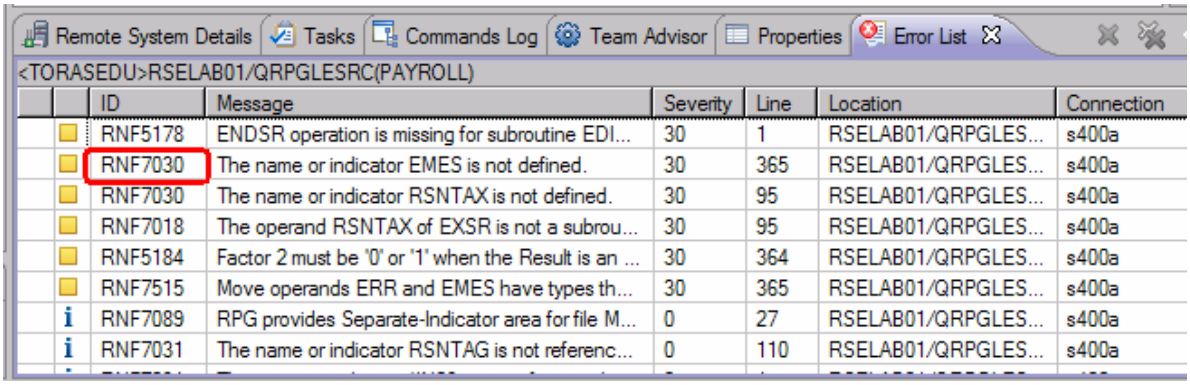
Verifying the source

Now you get to play with one of the most powerful and unique features of the Remote System Explorer – the Program Verifier. Before you compile your code on an IBM i system, you can make certain that there are no errors by invoking the Program Verifier. The verifier checks for semantic (compile) errors on your workstation so that you can guarantee a clean compile on your IBM i system. Think of the host cycles you'll save. It is especially handy when you are writing code but you are disconnected from a server. You can do this because Remote System Explorer ported the parsing and checking code from the IBM i compilers to the workstation. The Error List view lists the errors that are found and their severity, displays the error messages directly within the source and helps you to navigate between the errors.

To invoke the verifier:

1. Open member PAYROLL in QRPGLESRC in the editor if it is not already open.
2. With the focus on the editor, click **Source > Verify** from the workbench menu. (Similarly, you can also use the pop-up menu for the source member in the Remote Systems View or the Verify tool button — you need the source in the editor for the button to appear.)

After a moment the verifier will display an Error List below the Editor window.



	ID	Message	Severity	Line	Location	Connection
	RNF5178	ENDSR operation is missing for subroutine EDI...	30	1	RSELAB01/QRPGLES...	s400a
	RNF7030	The name or indicator EMES is not defined.	30	365	RSELAB01/QRPGLES...	s400a
	RNF7030	The name or indicator RSNTAX is not defined.	30	95	RSELAB01/QRPGLES...	s400a
	RNF7018	The operand RSNTAX of EXSR is not a subrou...	30	95	RSELAB01/QRPGLES...	s400a
	RNF5184	Factor 2 must be '0' or '1' when the Result is an ...	30	364	RSELAB01/QRPGLES...	s400a
	RNF7515	Move operands ERR and EMES have types th...	30	365	RSELAB01/QRPGLES...	s400a
i	RNF7089	RPG provides Separate-Indicator area for file M...	0	27	RSELAB01/QRPGLES...	s400a
i	RNF7031	The name or indicator RSNTAG is not referenc...	0	110	RSELAB01/QRPGLES...	s400a

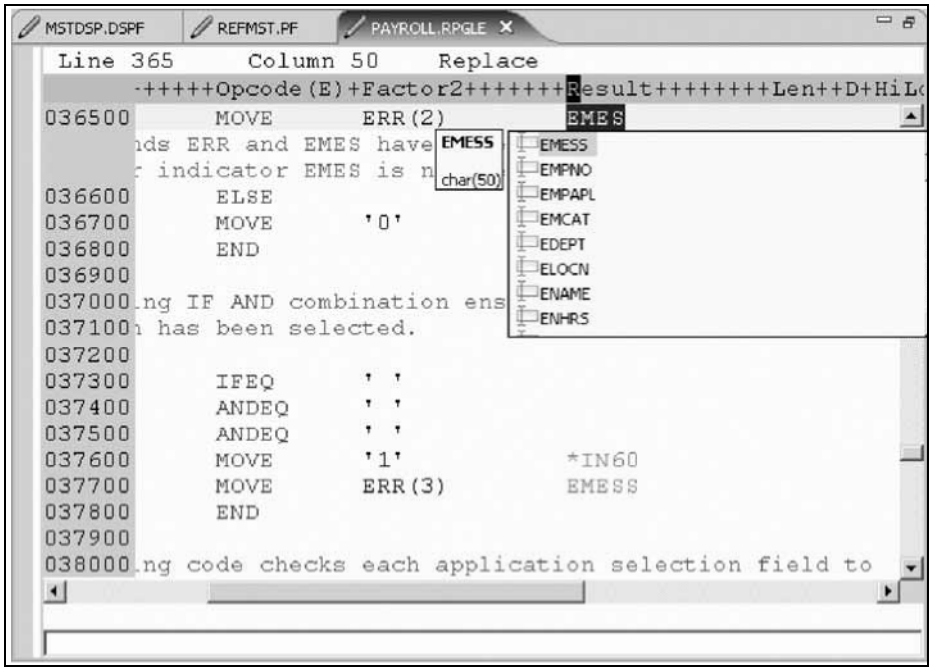
The error list shows you:

- The error message itself
- The severity
- The line number
- The source location
- The connection name

Next you will fix the errors in your source.

1. Double-click the error RNF7030.

You are automatically brought back into the Editor window to the line where the error occurred. The error on line 365 is a typo. The variable EMES should be EMESS. One good way of finding the correct name is the content-assist tool.



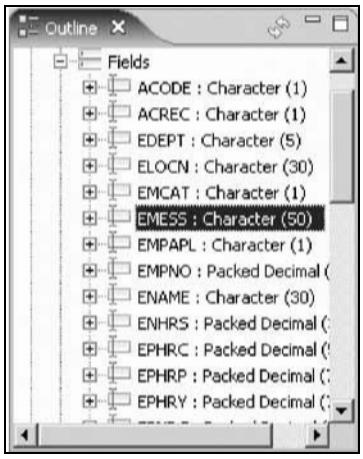
Tip: The Outline view must be populated to use content-assist.

2. Select the misspelled variable and press **CRTL+spacebar.**

If the variable starts correctly the selection presented contains the correct name.

3. Double-click the variable EMESS in the list to correct the variable name.

Another way to find the variable name is to use the Outline view and see what variables are declared.



The next error is a RNF7030 as well.

4. Double-click RNF7030. Fix it in the editor.
5. RSNTAX should really be RSNTAG. Make the appropriate change.
6. Go to the next error RNF5184.

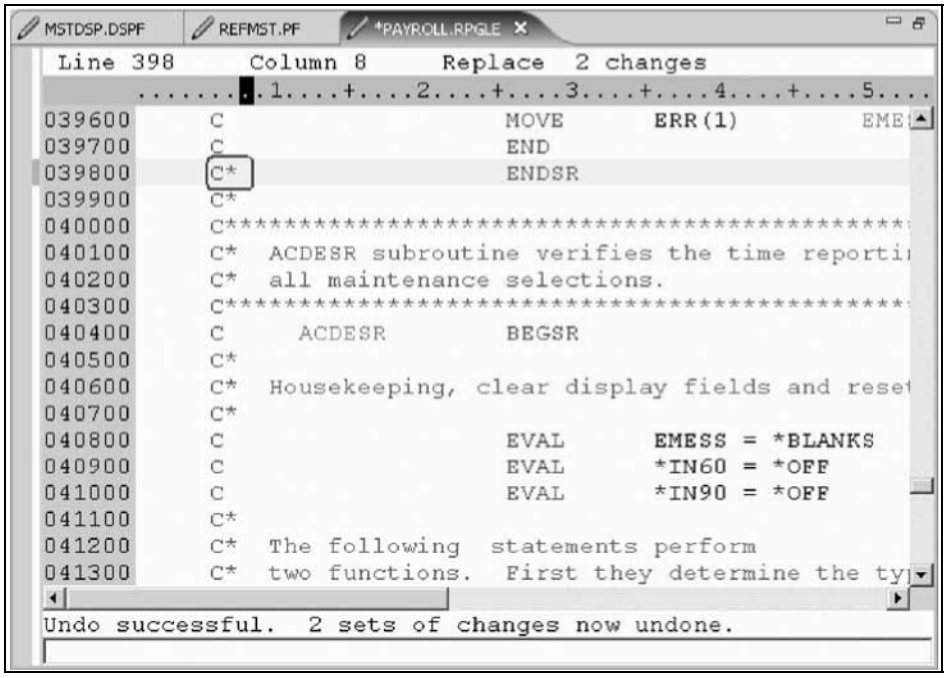
The next error is actually RNF7018 but it is related to the one you just fixed and can be ignored. It shows the same line number, which is an indication that both errors are related.

7. Double-click RNF5184.
8. Fix this by replacing the 2 with a 1.

Error RNF7515 is related to the first error you fixed. It has the same line number. You can ignore it. The only serious remaining error is RFN5178. This error is caused by a missing ENDSR.

Tip: The verifier could not determine where the ENDSR is missing so the line number reported is 1, for this reason you can't just double-click on the error message. You need to investigate where the missing statement belongs. You can use the Indent view to determine where the ENDSR is missing

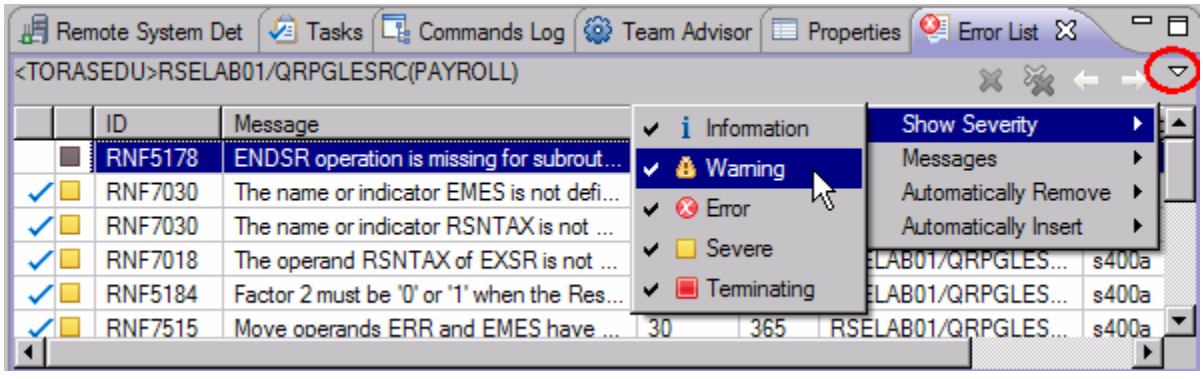
9. Move to line 398 in the editor.



10. Remove the asterisk (*) from the C-spec.

You can use the **Tab** key to quickly move to the appropriate column. All the non-informational errors are now fixed.

You can filter out messages according to these severities by using the filter menu.



11. Click the arrow in the Error List title bar.
12. Click **Show Severity** on the pop-up menu.
13. Clear the severities you don't want to see in the list (**Warning** for example)
14. You can save the member using one of these ways:
    a. Click **File > Save** from the workbench menu.

b. Click the Save icon  in the workbench toolbar

c. Press **Ctrl+ S**.

Changes are uploaded to the IBM i system.

15. Verify your source again

Everything should be ok. You should see only severity 0 messages. You are ready to compile the program.

You have verified your source and fixed any errors.

Compiling source remotely

The remote compile capability is part of the Remote System Explorer. It gives you a workstation interface to submit requests to compile, bind, or build objects on the IBM i host. It allows for easy access to all the compile options available for all the supported CRTxxx commands.

If you used the local program verifier, then your host compiles should be successful -- no wasted IBM i cycles. However, if there are errors, the host compiler will send the error information back to the workstation and they will be loaded into the Error List view, which behaves just as it did when you did a local verify.

The default for compiling programs is to submit the compile to the batch job queue.
Here in this exercise you can run the compile interactive.

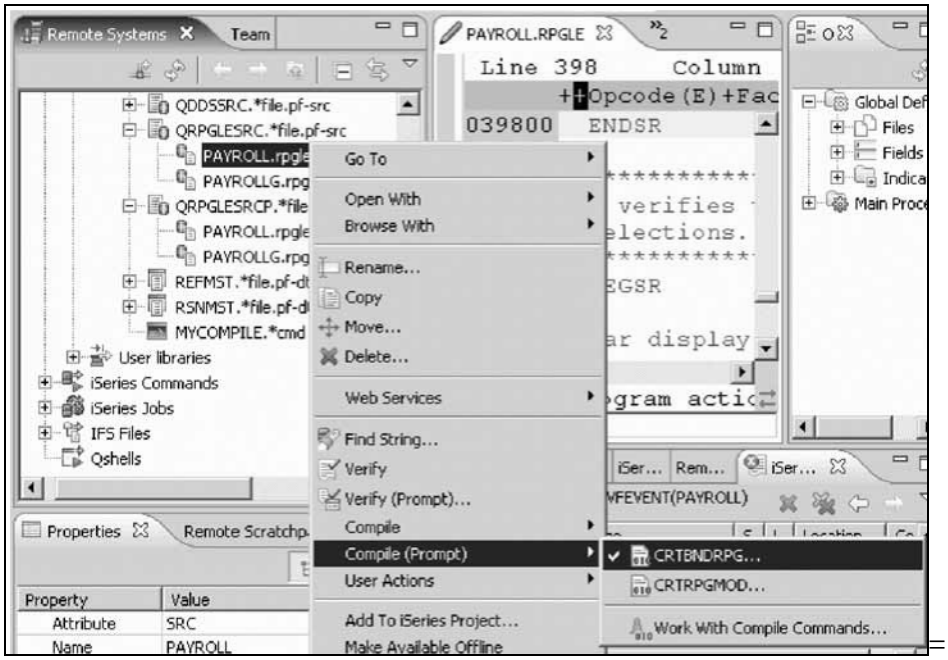
To change the preferences to run the compile interactive:

1. Click **Window > Preferences** from the workbench menu
2. In the left pane of the Preferences window, expand **Remote Systems**
3. Expand **IBM i** under Remote Systems
4. Click **Command Execution** under IBM i.
5. In the right pane of the Preferences window, clear the **Compile in batch** check box.
6. Click **OK** to return to the Remote System Explorer perspective.

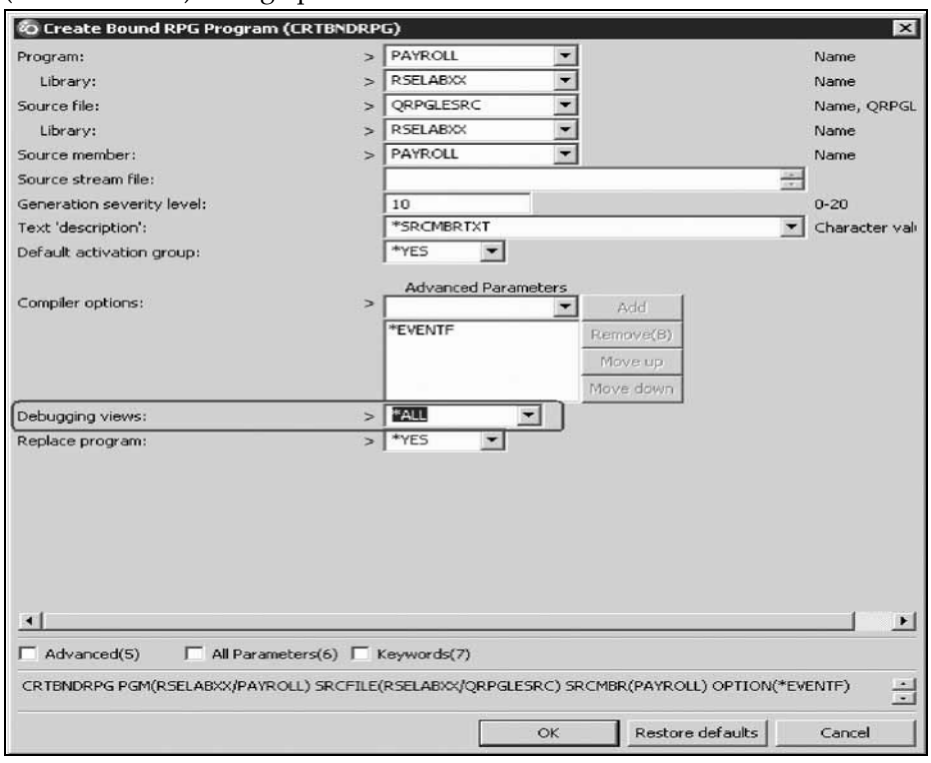
You will now use the prompt for the CRTBNDRPG command to specify your compile parameters. All entry fields pertaining to names are already filled in with the correct information.

To compile source:

1. Right-click the PAYROLL member in QRPGLESRC.



2. Click **Compile (Prompt) > CRTBNDRPG** on the pop-up menu. The Create Bound RPG Program (CRTBNDRPG) dialog opens.

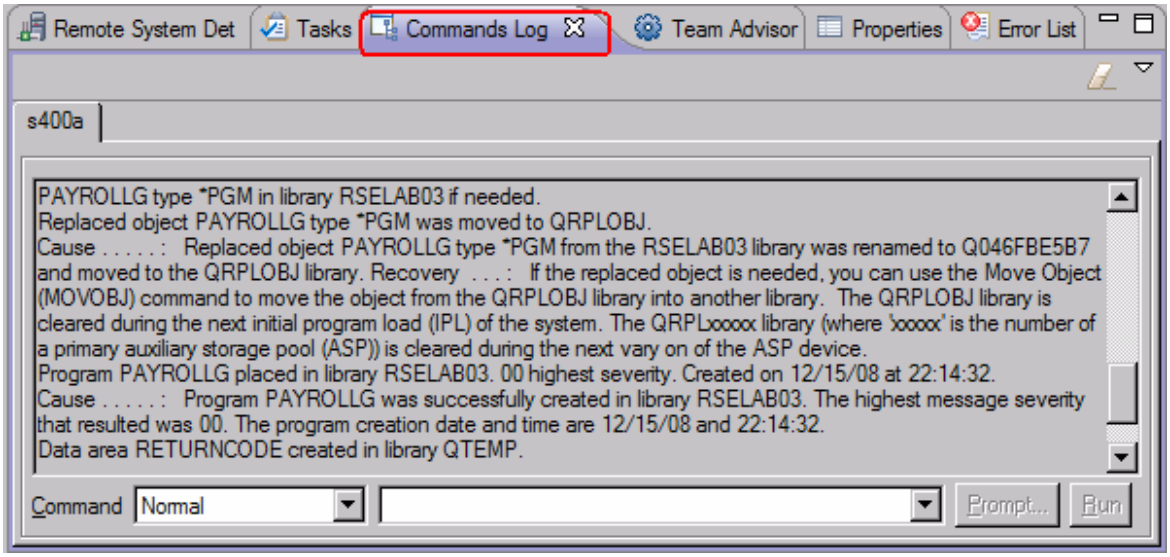


3. In the **Debugging views** drop down list, select the ***ALL** parameter

If you want to see the other parameters available, click **Advanced**.

4. Click **OK** when you are finished.

The progress bar on the workbench (bottom right corner) will indicate that the compile runs.
Then the error list will be shown, with no errors, just information messages.
You can also check the Commands Log view to verify that your program was created.



5. Click the **Commands Log** tab at the bottom of the workbench. This log shows a list of all commands run on the remote system and the messages returned for each command.

You have set compile preferences, invoked the compile command, and checked for a successful compile.

Submitting IBM i commands in the Object Table view

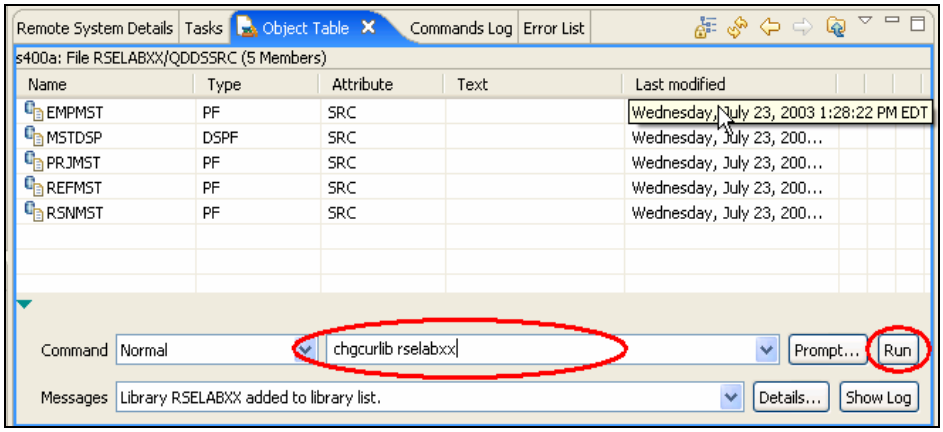
You can use the Object Table view inside the Remote System Explorer to submit commands to the IBM i system. You can run commands from the **Commands** field beneath the Object Table view, and view messages in the **Messages** field. After you populate the table, you can enter a command and click either **Prompt** to specify parameters and then **Run** or just click **Run**. When you run a command, the **Messages** field is populated with the messages from the command. When you select a message, the **Details** button is enabled. When you click this button, the message and its help is displayed.

Also note that besides the Object Table view, you can also use the Remote Systems view to run commands and programs. Which one you choose depends on your personal preference. In the Object Table view, you can see the properties of all items at the same time; they are displayed as rows across the table.

In the Remote Systems view, you have greater ease of navigation; you can work from your Library list in the Objects subsystem, and you can see the contents of many items before selecting the one you want to run.

In the **Commands** field, you select where you want to run the command. The choices are Normal, which means that the command will run in the RSE communication server job, Batch or Interactive.

To change the library list:



1. Click the **Object Table** view tab from the views at the bottom of the workbench.
2. In the **Command** field type, `CHGCURLIB RSELABxx` for example.

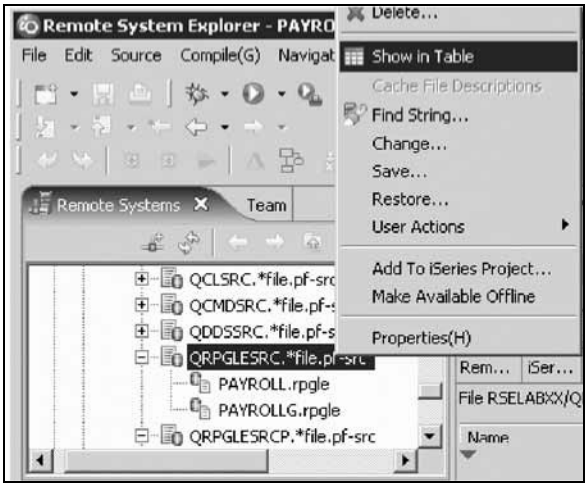
Tip: Use a library that is on your IBM i system.

3. Click **Run**.

If you haven't used the Object Table view to show IBM i objects, you will see an error message because the Table view is not linked to an active connection.

If you see this message, click **OK**.

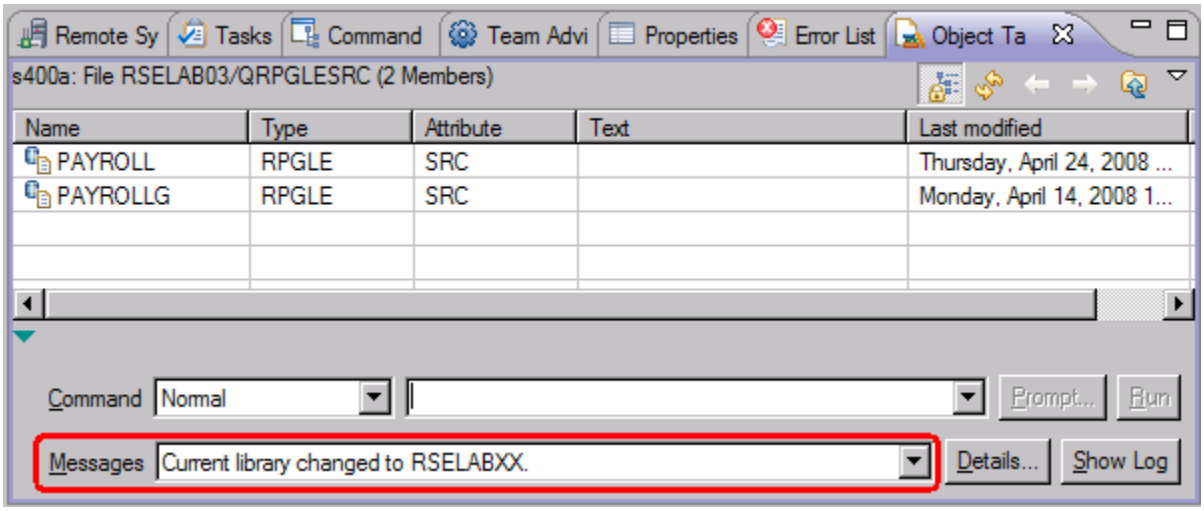
a. In the Remote Systems view, right-click QRPGLESRC.



b. Click **Show in Table** on the pop-up menu. The Table view is now populated with the member in the selected source file.

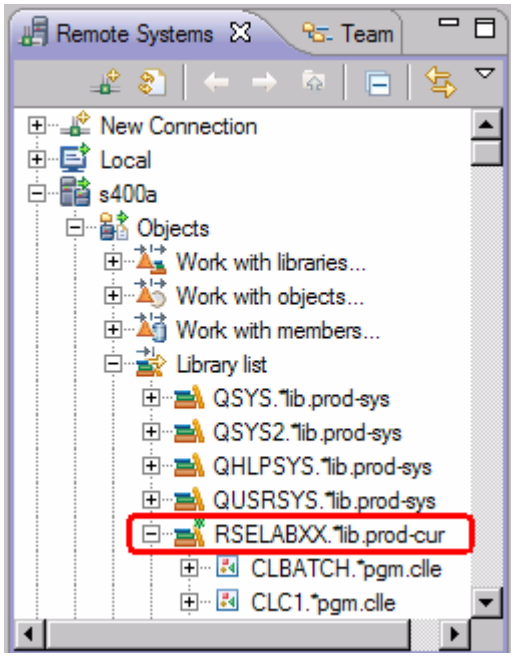
c. Run the CHGCURLIB command again.

The command will run on the IBM i system and after completion you will see the completion message on the bottom of the Object Table view.



Back in the workbench in the Remote Systems view

4. Right-click **Library list**
5. Click **Refresh**.



You will see a small green asterisk beside the RSELABxx library to indicate it as your current library.
You can also connect to other than IBM i systems with the Remote System Explorer and launch commands for these systems as well, for example, your local system, or Linux®.

You have submitted a command to change the current library in the command line of the Object Table view.

Running commands and programs

As you know, you can run programs and commands from the Remote Systems view or the Table view in the following ways:

1. In the Remote System Explorer communications server job. (Your current method)
2. In a batch job.
3. In an interactive job (for 5250 applications).
4. In a server job (Multi-threaded action in the menu)

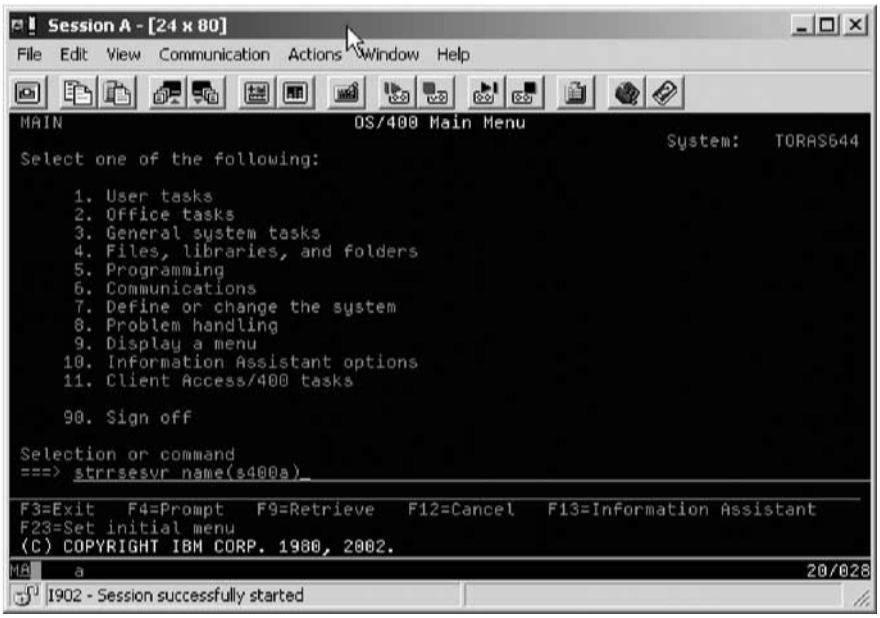
Using the first option lets you run the program in the same job as the communications server. With batch and interactive jobs, you cannot monitor the status as easily, however, you do not tie up your communications server and you are notified when the program command ends. Batch jobs work as you would expect and do not require any initial setup. Running an application as multi-threaded creates a BCI job and runs the application in that job. Interactive programs require a 5250 emulator, and you need to first run a STRRSESVR connectionName command to associate the emulator with a particular connection in the Remote System Explorer communications server. A multi-threaded run creates a new server job and this way keeps the RSE communications server job free for other tasks.

To start an interactive connection:

a. Start a 5250-emulation session.

b. Sign-on to the server with your User ID and password.

Tip: Instead of the **Enter** key, you may have to use the **Ctrl** key in your5250-emulation session.



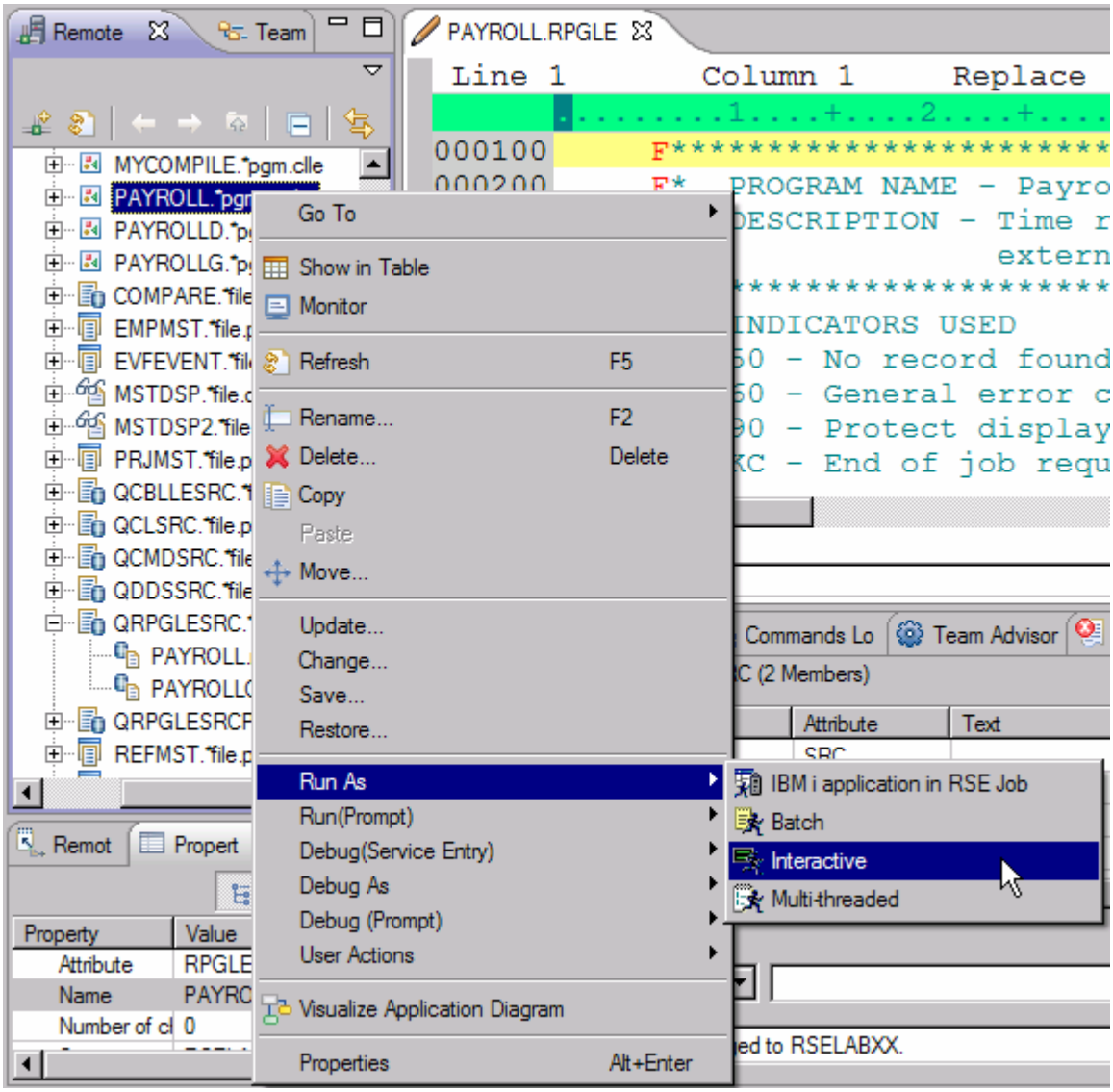
c. In the command line, type the command `STRRSESVR connectionName`

d. Press **Enter**. The connectionName parameter is the name of your connection defined in the Remote Systems view. This associates the interactive job with the Remote System Explorer communications server.

Now you are ready to run the program that you just compiled.

e. Return to the workbench.

To run the program:



a. In the Remote Systems view, locate the PAYROLL program that you created.
b. Right-click the PAYROLL program.
c. Click **Run As > Interactive** on the pop-up menu.
d. Switch to your 5250-emulation session.
   You will see the payroll program menu.

e. Type **X** beside **Employee Master Maintenance**.

f. Press **Enter**.

g. Type 234 for the Employee Number.

h. Type A for the Action Code to add employee **234**.

i. Press **Enter**.

j. Type any information you like about the employee.

k. Press **Enter**.

l. Play in the application as much as you like.

m. Press **F3** to end the applications.

To get control of the interactive job:

n. Right-click **Objects** and click**Release Interactive Job** on the pop-up menu.

You can also choose to disconnect a session. You would right-click the connection and click **Disconnect** on the pop-up menu.

You ran programs and commands from the Remote Systems view or the Objects Table view.

Module summary

In this module you learned how to verify and compile RPG from the Remote Systems LPEX Editor. When errors are found by either the verify or the compile step, the Error List appears. Error List is a powerful tool that manages errors found by verify and compile utilities. You became familiar with these tools, the various capabilities of the Error List and the RPG program that you created. You also learned how to associate an interactive session with your IBM i connection, run your program, and release the interactive job.

Debugging a program

This module teaches you how to debug a CL and ILE RPG program. You will learn how to start the debugger, set breakpoints, monitor variables, run, and step into a program, view the call stack in the Debug view, remove a breakpoint, add a memory monitor, and set Watch breakpoints and all from the Debug perspective.

Introducing the Integrated IBM i Debugger

The Integrated IBM i Debugger is a source-level debugger that enables you to debug and test an application that is running on an IBM i system. It provides a functionally rich interactive graphical interface that allows you to:

- View source code or compiler listings, while the program is running on an IBM i host.
- Set, change, delete, enable and disable line breakpoints in the application program. You can easily manage all your breakpoints using the Breakpoints view.
- Set Watch breakpoints to make the program stop whenever a specified variable changes.
- View the call stack of your program in the Debug view. As you debug, the call stack gets updated dynamically. You can view the source of any debug program by clicking on its call stack entry.
- Step through your code one line at a time.
- Step return, step into or step over program calls and ILE procedure calls.
- Suspend program execution and get control back to the debug session.
- Display a variable and its value in the Monitors view. The value can easily be changed to see the effect on the program's flow.
- Locate procedure calls in a large program quickly and easily using the Modules/ Programs view.
- Debug multi-threaded applications, maintaining separate stacks for each thread with the ability to enable and disable any individual thread.
- Load source from the workstation or a different IBM i system than the one the program runs on – useful if you don't want the source code on a production machine.
- Debug client/server and distributed applications.

The Debugger supports RPG/400® and ILE RPG, COBOL and ILE COBOL, C, C++ and CL.

Now that you know the basic features of the debugger, let's try them out.

Starting a Debug session using a service entry point

You will be working with the ILE RPG program PAYROLLG.

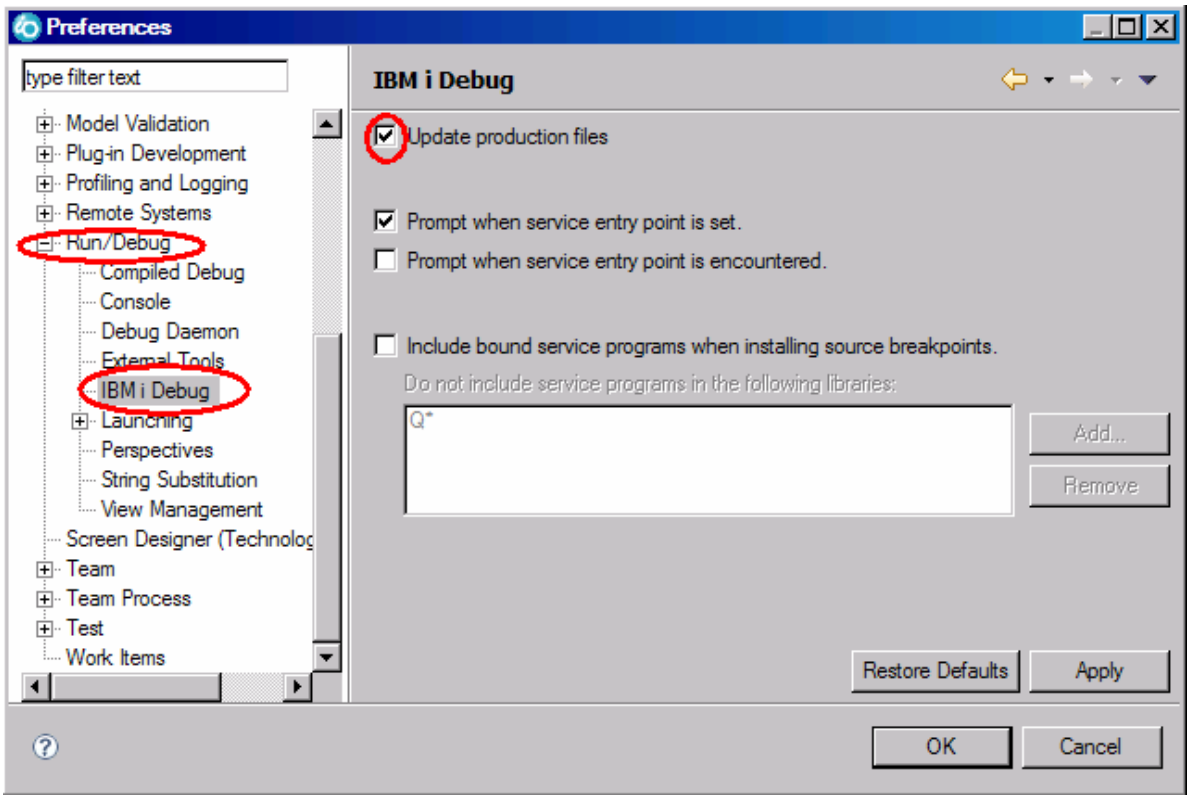
Note: PAYROLLG is the same RPG program as PAYROLL but without compile errors. You are using it instead of PAYROLL in this lesson, to accommodate anyone who decided to skip right to this lesson without completing the editor lessons in "Editing source" on page 25.

To make the lesson interesting you will use CL program **CLR1** to call PAYROLLG and you will pass one parameter to CLR1.

In this lesson, you will use a service entry point to start a debug session for your application. The service entry point feature is designed to allow easy debugging of applications that invoke business logic written in RPG, COBOL, CL, or even C or C++. The service entry point is a special kind of entry breakpoint that can be set directly from the Remote System Explorer. It is

triggered when the first line of a specified procedure is executed in a job that is not under debug. Service entry points allow you to gain control of your job at that point. A new debug session gets started and execution is stopped at that location.

Tip: To use a service entry point to start a debug session for your application and to allow updating of files in production libraries while debugging, the **Update Production Files** must be checked in the preferences of IBM i Debug.

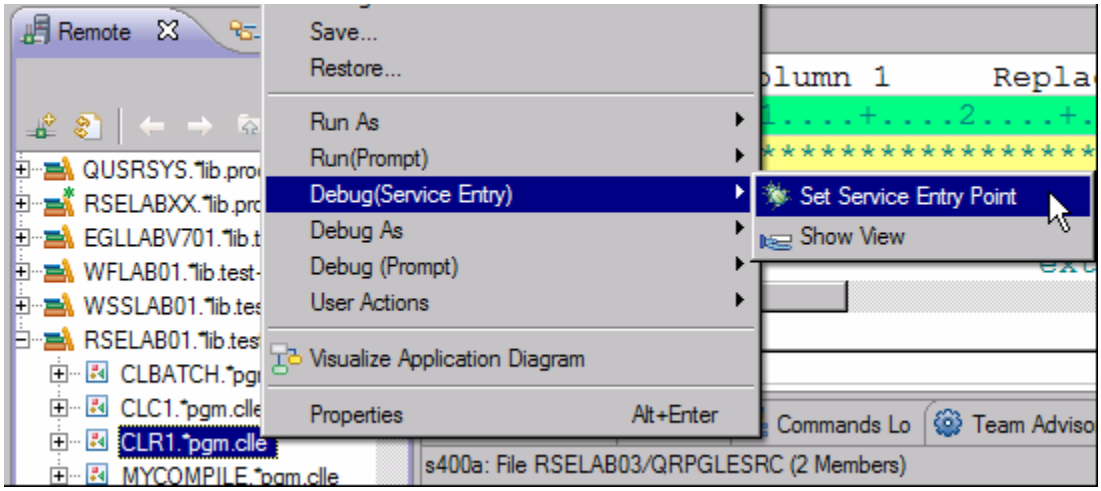


(**Window >Preferences** then expand **Run/Debug** and select **IBM i Debug**)

Since you are using test libraries for the exercises, you don't have to check this IBM i Debug preference.

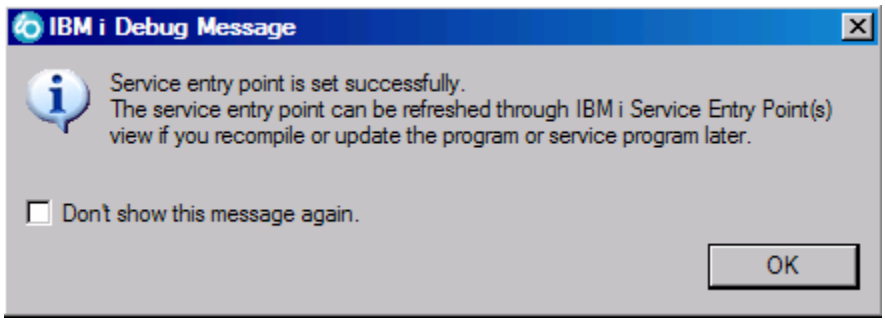
To start a debug session using a service entry breakpoint:

1. In the Remote Systems view expand the **Library list** filter, if it isn't expanded already.
1. Expand library RSELABxx, if it isn't expanded already.
2. Right-click program CLR1 in library RSELABxx.

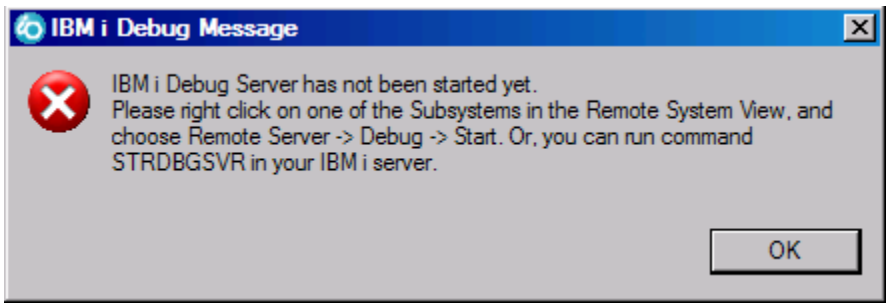


2. Click **Debug (Service Entry) > Set Service Entry Point** on the pop-up menu to set a service entry point.

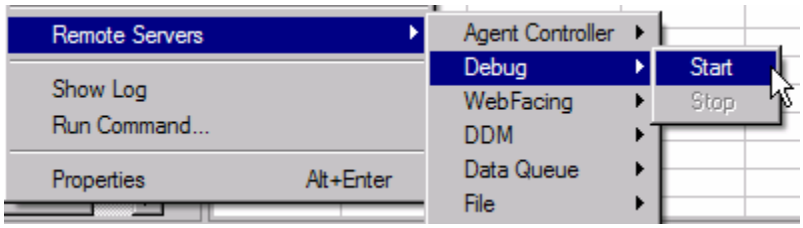
A message displays indicating the service entry point was successfully set.



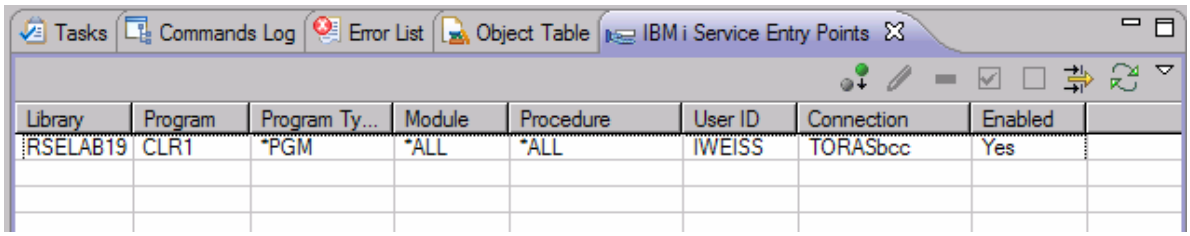
If you get an error message instead,



indicationg that the Debug Server has not been started yet, right-click Objects in the Remote Systems view and select **Remote Servers > Debug > Start** and perform step 2 again.



The Service Entry Points view is automatically added to the stacked views at the lower right. It lists all the service entry points. You use this view to delete, activate, de-activate, modify and refresh service entry points.



5. Switch to the 5250 emulation session.

Tip: If your 5250 session is still associated with the RSE server job, you need to release the interactive session. To do so, in the Remote Systems view, right-click **Objects** and click **Release Interactive Job** on the pop-up menu.

6. On the command line of the 5250 screen, add the library RSELABxx to the library list and invoke the program CLR1:
   `ADDLIBLE RSELABxx` and then
   `CALL PGM(RSELABxx/CLR1) PARM( 'XX')` (where **XX** is your team number)

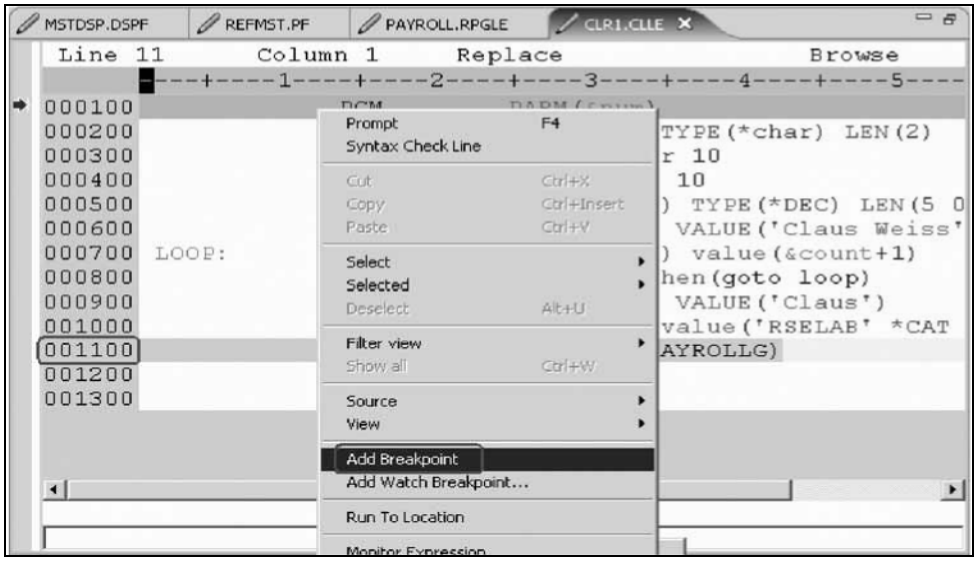
As soon as the program enters the system, the service entry point is hit and the debug session is started on the workstation and the perspective displays with the CLR1 source code in the editor. The Debug perspective gives you access to all available debugger features. Let's look at some of them.

Click anywhere in the workbench to give it focus.

Setting breakpoints

You can only set breakpoints at executable lines. One way to set a breakpoint is to right-click on the line in the Source view.

To set a breakpoint:



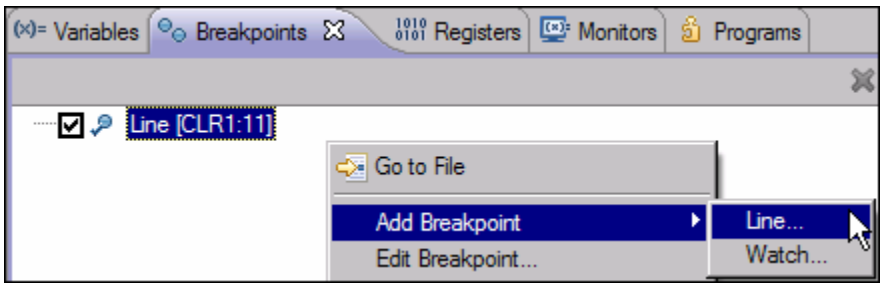
1. Position the cursor on line 11.
2. Right-click anywhere on line 11.
3. Click **Add breakpoint** on the pop-up menu.

A dot with a checkmark in the prefix area indicates that a breakpoint has been set for that line. The prefix area is the small grey margin to the left of the source lines.
Now you add a conditional breakpoint to stop in the loop when it loops the 99th time.

Adding a conditional breakpoint

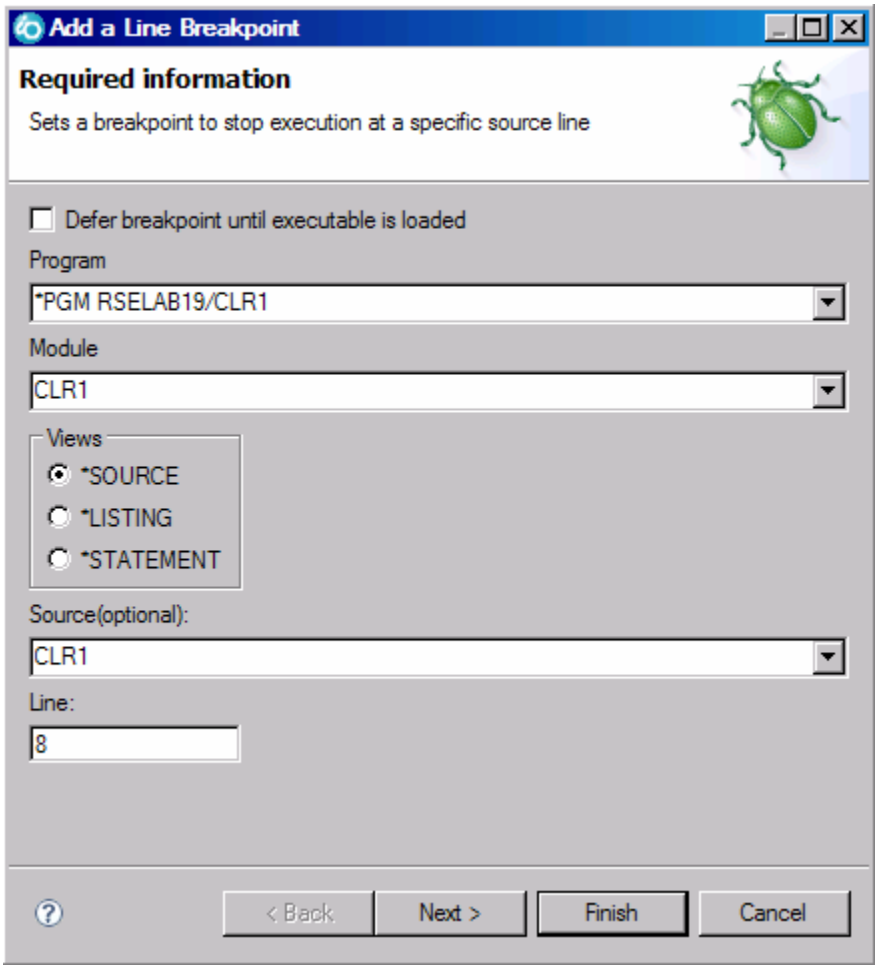
1. Select line 8.
2. Click the **Breakpoints** tab in the upper right pane of the Debug perspective.

The Breakpoints view opens.



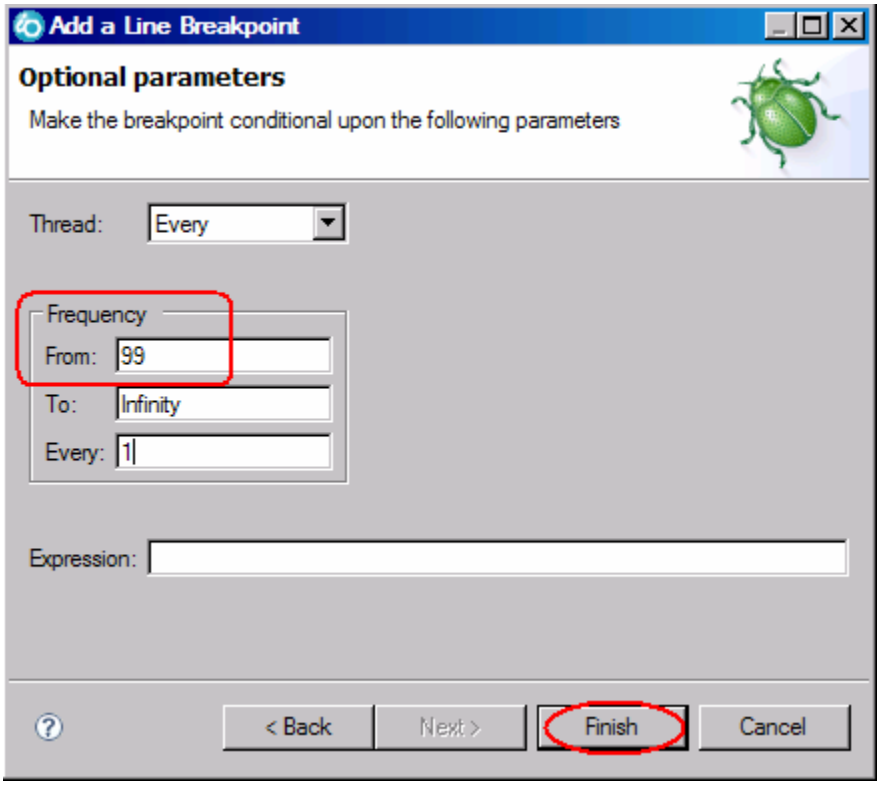
3. Right-click anywhere within the **Breakpoints** view.
4. Click **Add Breakpoint > Line** on the pop-up menu.

The Add a Line Breakpoint window opens.



Tip: You can select an existing breakpoint by right-clicking it and selecting **Edit Breakpoint**.

5. Click **Next**.



You only want to stop in the loop when it executes for the 99th time or more. You can do that by setting the **From** field of the **Frequency** group to 99.

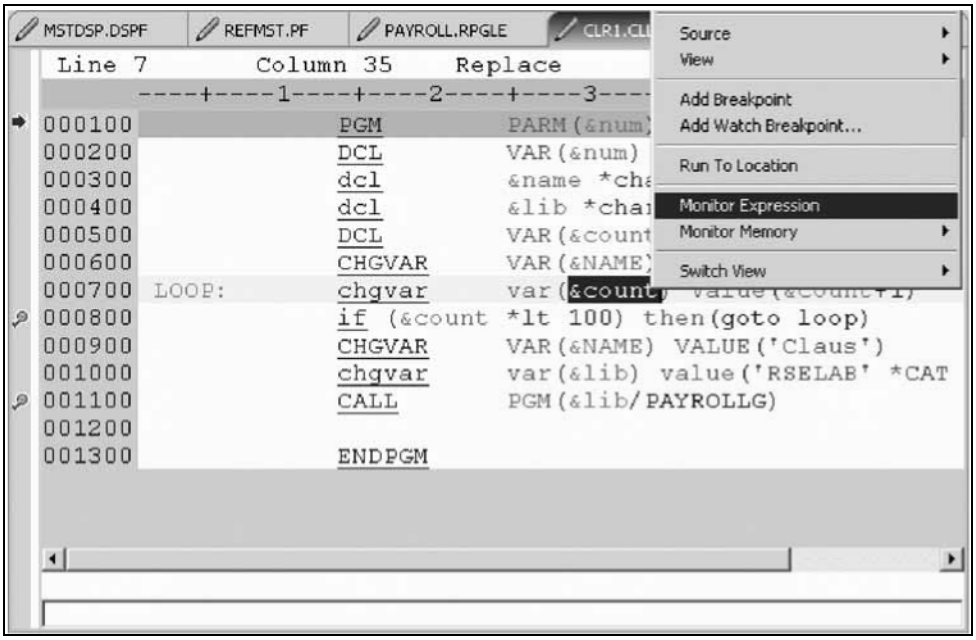
6. Under **Frequency** in the **From** field, type `99`.
7. Click **Finish**.

You have added a breakpoint including a conditional breakpoint to your debug session.

Monitoring variables

You can monitor variables in the Monitors view. Now you will monitor the variable &count.

To monitor a variable:

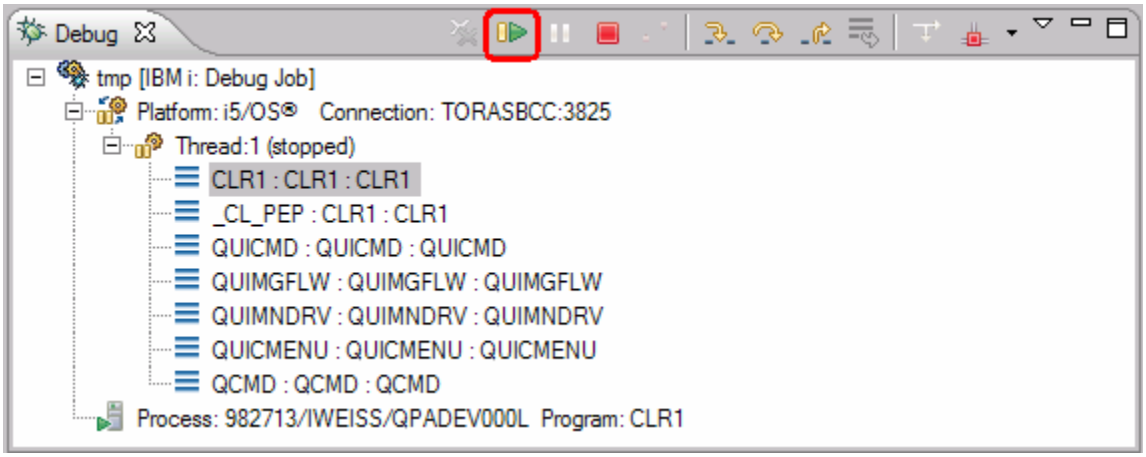


2. In the Source view, double-click the variable *&count*.
3. Right-click *&count*.
4. Click **Monitor Expression** on the pop-up menu.

The Monitors view opens.

The variable appears in the Monitors view. Its current value is zero.

Now that some breakpoints and a monitor are set, you can start to run the application.



4. Click the **Resume** icon from the Debug toolbar.

   The program starts running and stops at the breakpoint at line 8. (Be patient, the Debugger has to stop 98 times but because of the condition continues to run until the 99th time.) Notice in the Monitors view, that *&count* now has the value 99.

5. Click the **Resume** icon again.

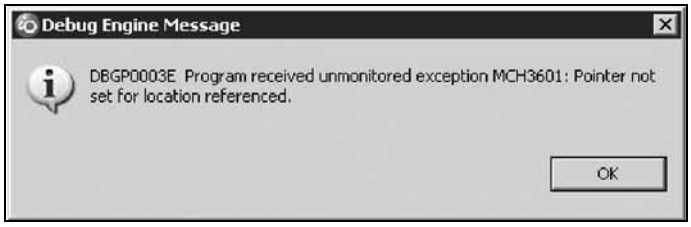
   The program stops at the breakpoint at line 8 again and *&count* now has the value 100.

6. Click the **Resume** icon once more so that the program runs to the breakpoint at line 11.

7. **If you do not see the error message below, go to** "Stepping into a program" on page 88.

Error Handling

If you forget to add the parameter to the CALL program command when you call the program, you will see this error message.

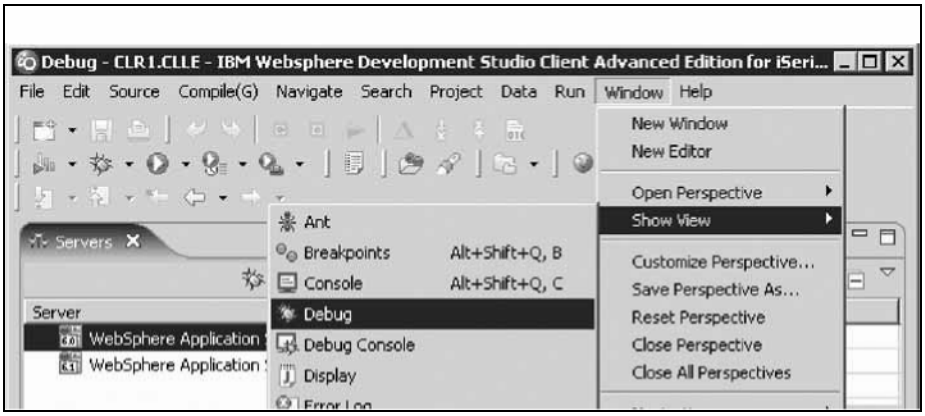


1. Click **OK**

2. Click the **Terminate** icon on the Debug toolbar.

The debug session terminates on the workstation but the exception waits for input from the 5250 emulation session.
If you closed the Debug view by mistake, you will need to re-open the Debug view and then terminate the debug session on the workstation.

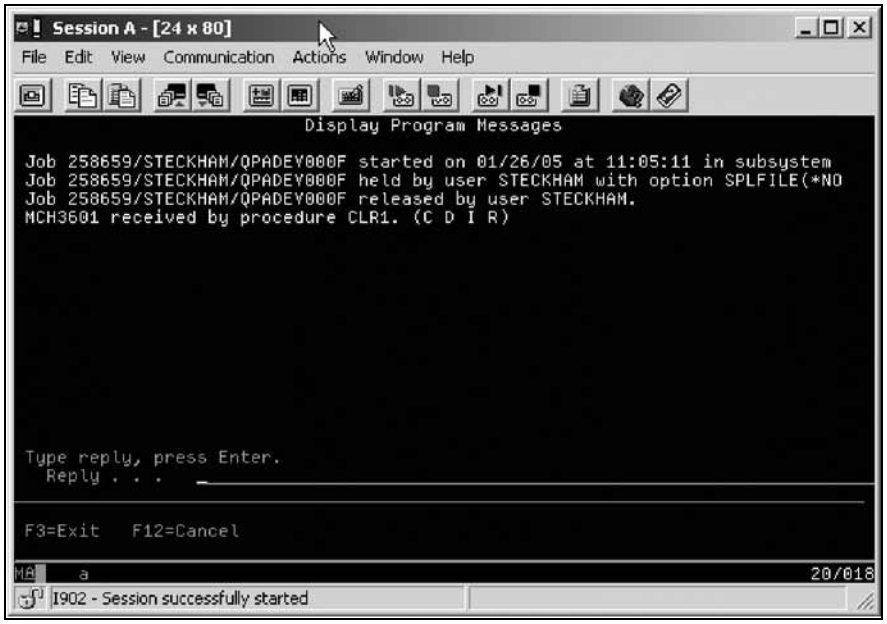
Here are the steps to re-open the debug view:



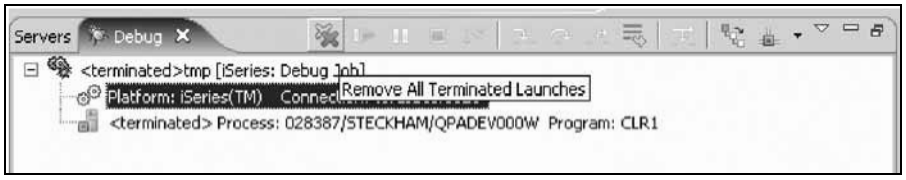
1. Click **Window > Show View > Debug**.

Now terminate the Debug session if you haven't done so already.

1. Go to your 5250 emulator



2. Enter C for cancel and press **Enter** until the program messages complete.
In the workbench,



3. Click the **Remove all terminated launches** icon on the Debug toolbar to clean up the Debug view.

To restart the program and start the debug session again:

4. On the 5250 command line, call the CLR1 program with the parameter XX.

`CALL PGM(RSELABxx/CLR1) PARM('XX').` **XX** being your team number

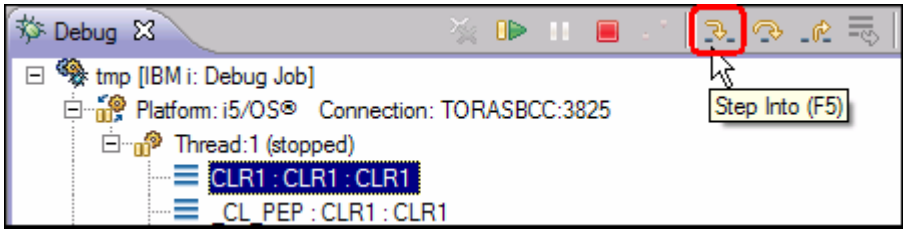
Stepping into a program

The Debugger allows you to step over a program call or step into it. When you step over a program call, the called program runs and the Debugger stops at the next executable statement in the calling program.

You are going to step into the PAYROLLG program.

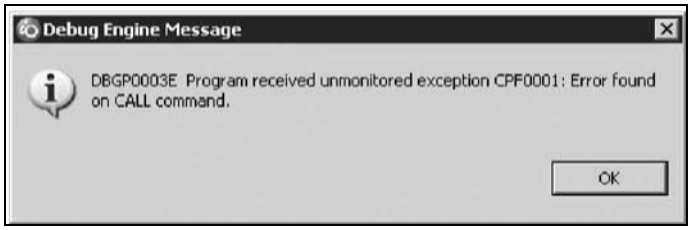
To step into a program:

1. Click the **Step into** icon on the Debug toolbar.



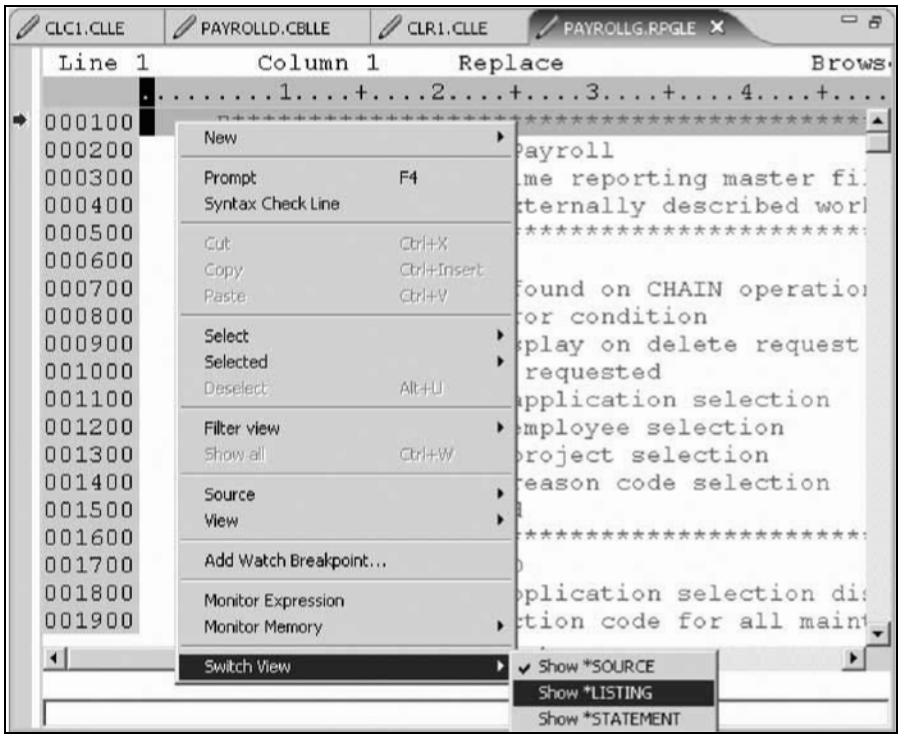
The source of PAYROLLG is displayed. Depending on the option you used to compile the program (*SRCDBG or *LSTDBG for RPG, or *SOURCE, *LIST, or *ALL for ILE RPG), this window displays either the Source or Listing View.

If you specified an incorrect parameter for the CALL program command, or your library list does not include RSELABxx, you will see this error message.



Make sure your library list is correct and complete the same steps as covered in the section called **Error Handling** in "Monitoring variables" on page 87.

2. Right-click anywhere in the Source view and click **Switch view > Show *LISTING** on the pop-up menu.



3. Page down in the source and take a look at the expanded file descriptions.

You don't have any /Copy member in your PAYROLLG program but these would also be shown in a Listing view. Switch back to the Source view.

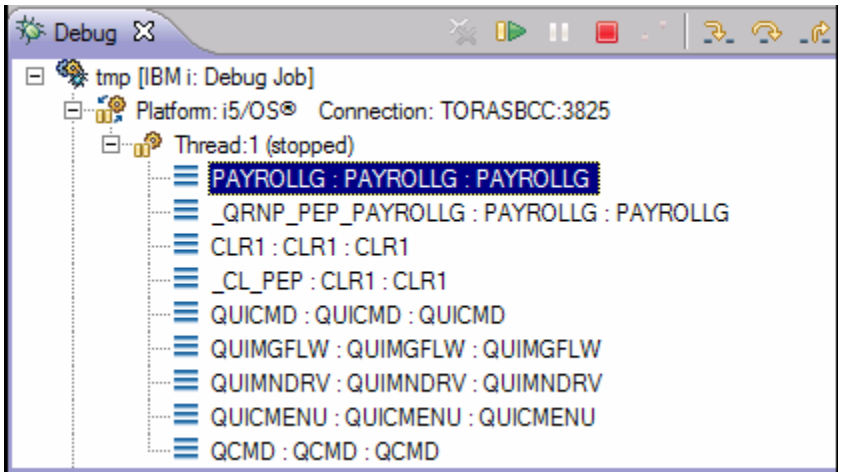
4. Right-click anywhere in the Source view.
5. Click **Switch view > Show *SOURCE** on the pop-up menu.

You have stepped into PAYROLLG program, switched the view from source to listing and back to source.

Listing call stack entries

The Debug view in the upper left pane, lists all call stack entries. It contains a tree view for each thread. The thread can be expanded to show every program, module, and procedure that is on the stack at the current execution point. If you double-click on a stack entry you will display the corresponding source if it is available. Otherwise the message No Debug data available appears in the Source view.

In the Debug view, expand the stack entry of Thread1 if it is not expanded already.



The stack entry allows you to work with and switch between different programs and/or ILE modules.

You have viewed the call stack entries of your program.

Setting breakpoints in PAYROLLG

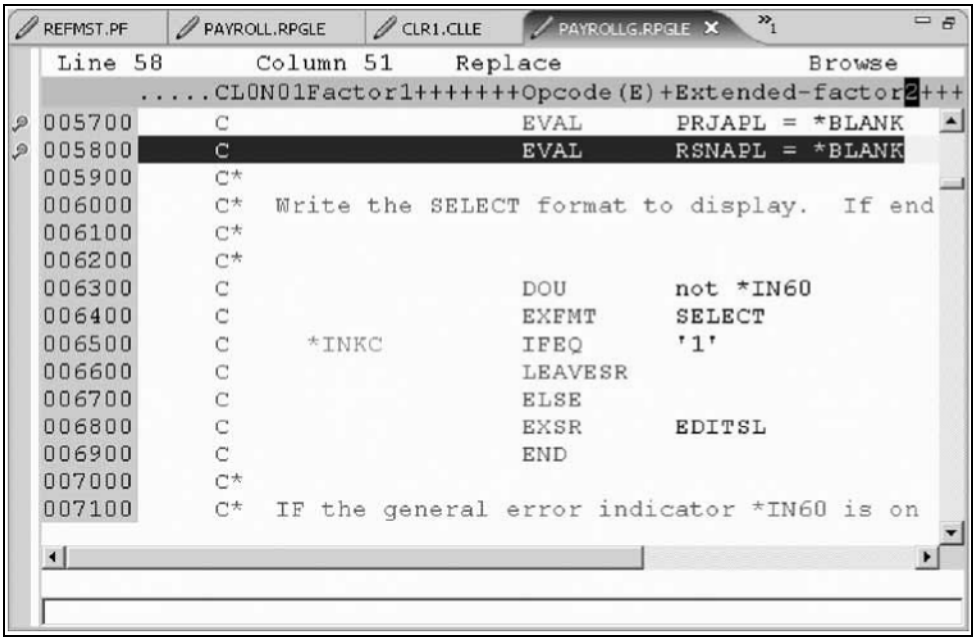
Now you add some breakpoints in PAYROLLG.

To add breakpoints:

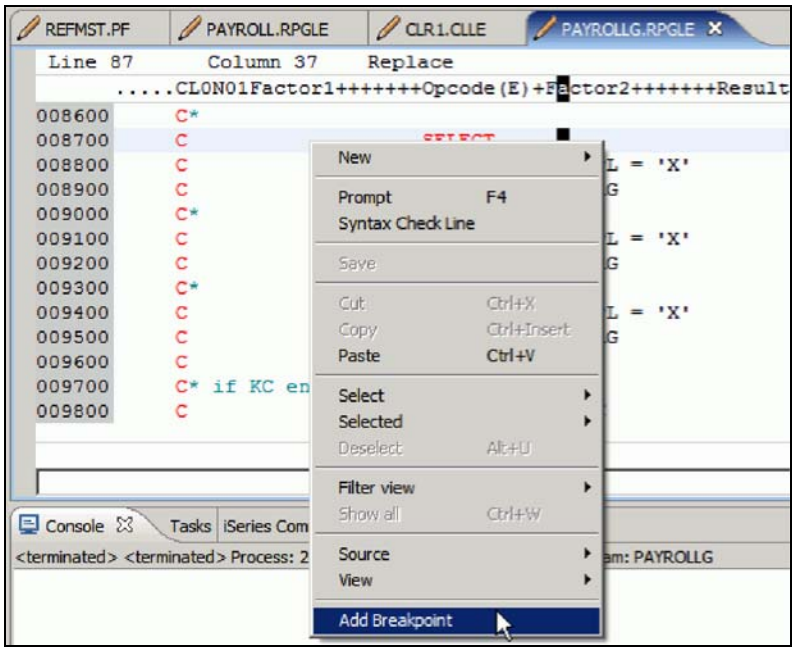
1. Select PAYROLLG in Thread1.
2. In the source view scroll to line 57.
3. Double-click the prefix area of line 57.

A breakpoint icon is added to the prefix area of this line to indicate that a breakpoint is set.

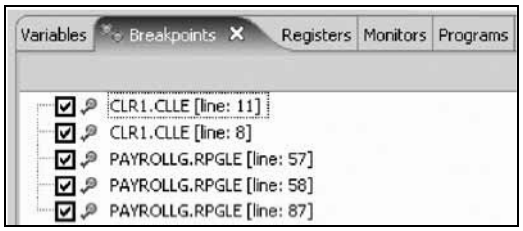
4. Repeat the above step for line 58



5. Right-click in the prefix area of line 87 and click **Add Breakpoint** on the pop-up menu.



To view all breakpoints, select the **Breakpoints** tab from the top left pane.



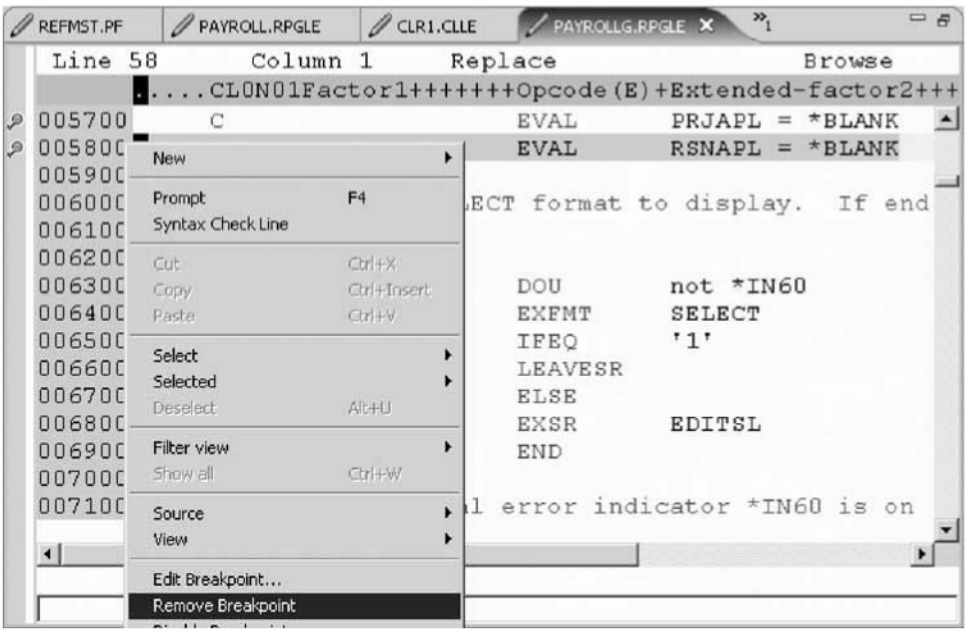
This view shows all breakpoints currently set in your Debug session. This is a convenient place to work with breakpoints. You can remove, disable/enable, add, or edit a breakpoint. These tasks are available from the pop-up menu when you right-click in the view area. Double-click any entry to show the source where the breakpoint is set.

Removing a breakpoint in PAYROLLG

It is also easy to remove breakpoints.

To remove a breakpoint:

1. Right-click the prefix area of line 58.
2. Click **Remove Breakpoint** on the pop-up menu.



The icon is removed from the prefix area indicating that no breakpoint is set on that line. The breakpoint is also removed from the list in the Breakpoints view.

Tip: Double-clicking on a breakpoint in the prefix area will also remove that breakpoint.

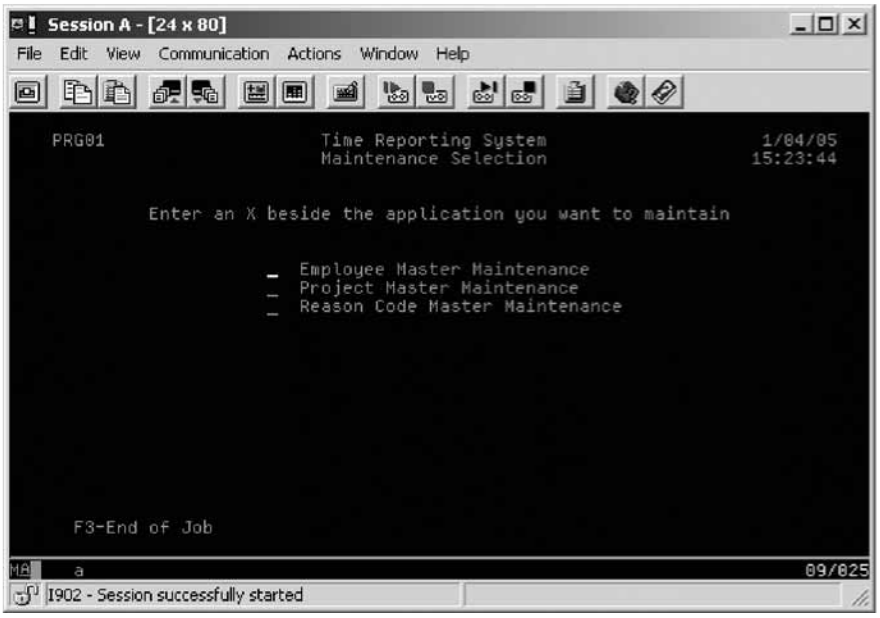
Now you are ready to run the PAYROLLG program.

3. Click the **Resume** icon from the Debug toolbar.

The program starts running and runs to the breakpoint at line 57.

4. Click the **Resume** icon again.

The program waits for input from the 5250-emulation session.



5. Type an X beside the **Project Master Maintenance** option.
6. Press **Enter** in the emulation session. The program runs to the breakpoint at line 87.

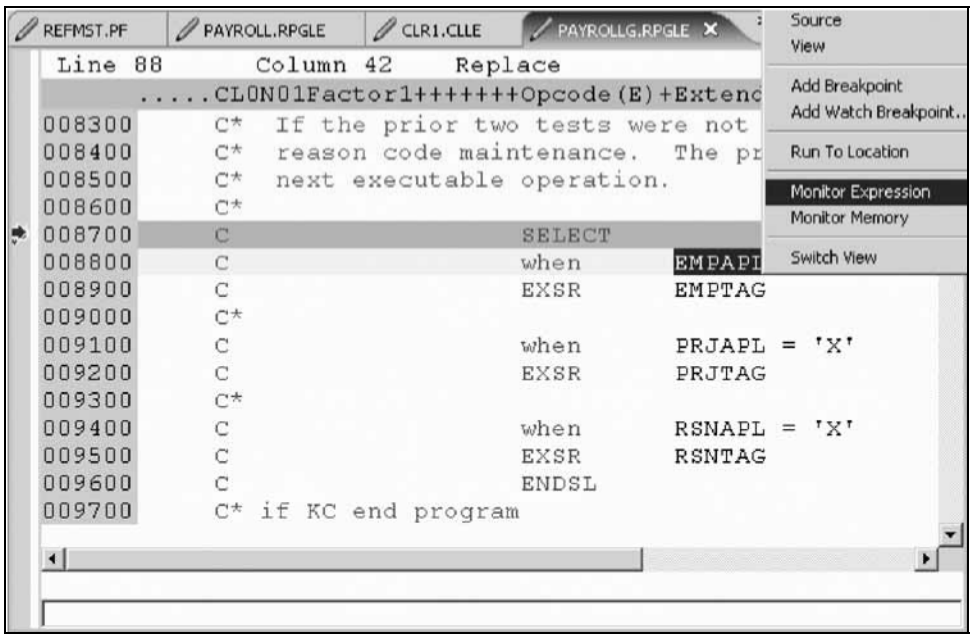
You have removed a breakpoint from PAYROLLG and started to run the program.

Monitoring variables in PAYROLLG

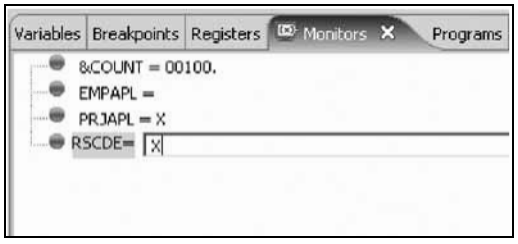
Now let's monitor variables and change them in PAYROLLG.

To monitor variables:

1. In the source view, double-click the variable EMPAPL on line 88.
2. Right-click the variable.
3. Click **Monitor Expression** on the pop-up menu.



4. Click the **Monitors** tab in the upper right pane. The variable appears in the **Monitors** view. Its value is blank because you did not select the **Employee Master Maintenance** option.
5. In the same way add the variables PRJAPL on line 91 and RSCDE on line 113 to the monitor. Variable PRJAPL equals X because you did select the **ProjectMaster Maintenance** option.
6. In the Monitors view, double-click the variable RSCDE. The value changes into an entry field.
7. In the entry field, type in the new value X for the variable.



8. Press **Enter**.

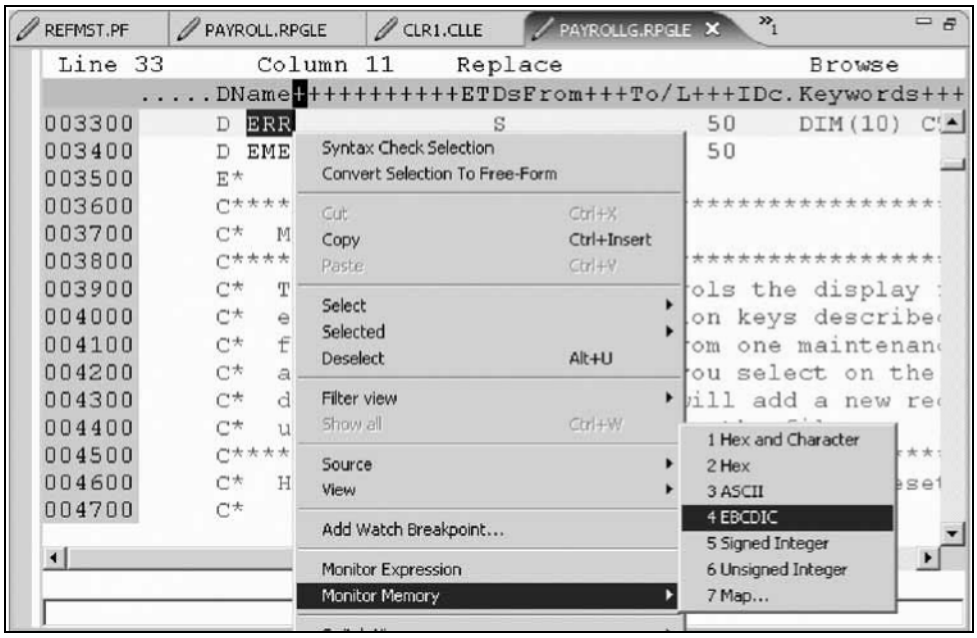
The variable is successfully changed.

Adding a memory monitor

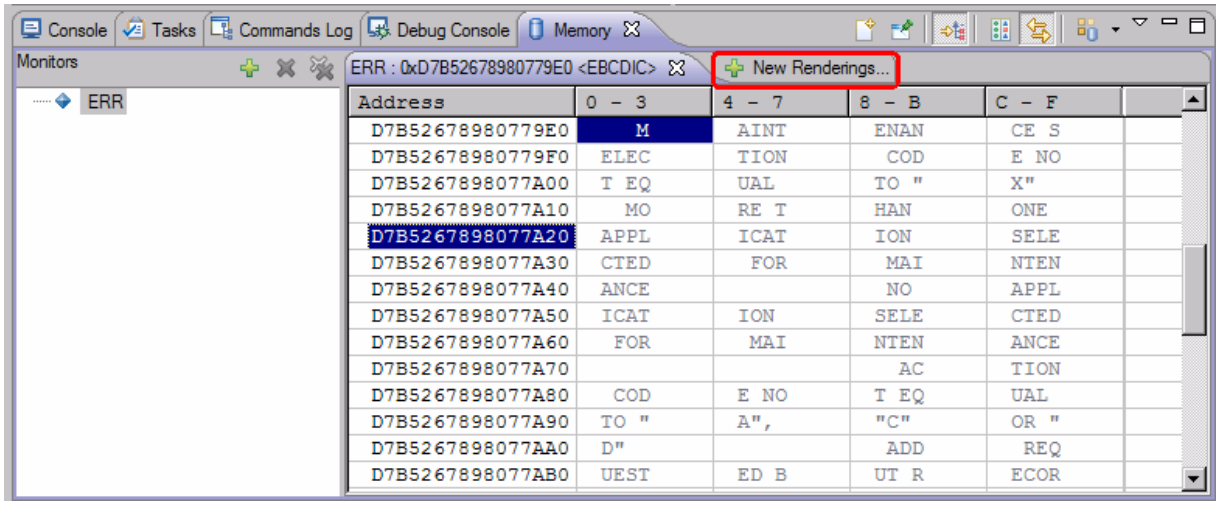
Adding a memory monitor for a variable allows you to view the memory starting with the address where the variable is located. The memory can be displayed in different formats, for example hexadecimal and text.

To add a memory monitor:

1. In the Source view, double-click the variable ERR in line 33.
2. Right-click and select **Monitor Memory > EBCDIC** on the pop-up menu.

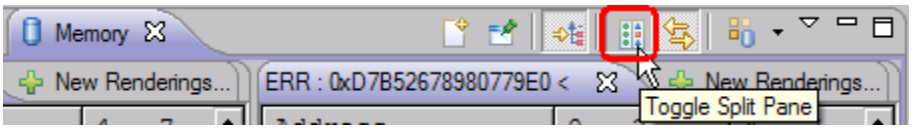


This will open the Memory view in the pane at the bottom of the perspective. The tab shows the name of the variable.



3. Use the scroll bar on the right of the Memory view to scroll down. You can see the current content of the memory.

4. Right-click in the view area.
5. Click **Reset to Base Address** on the pop-up menu to return to the starting address.
6. To get the hex content of the memory starting with the selected variable, click the tab **New Renderings** and select **Hex** for example. A new page with the hex values is added to the Memory view.
7. Click the Toggle Split Pane icon to display the character values as well.



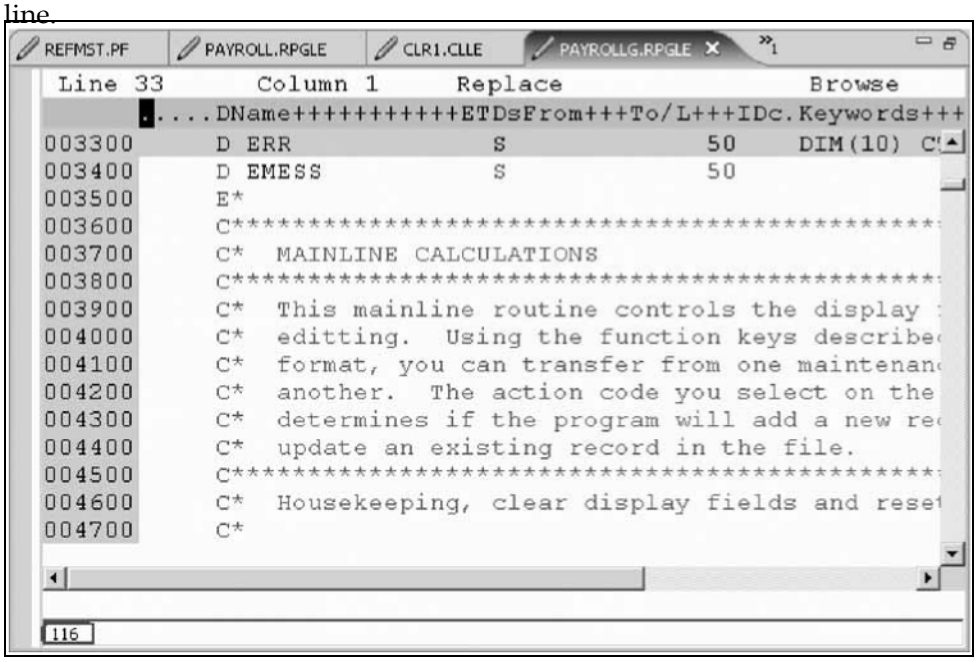
You have added a memory monitor for the variable ERR.

Setting Watch breakpoints

A Watch breakpoint provides a notification to the user when a variable changes. It will suspend the execution of the program until an action is taken.

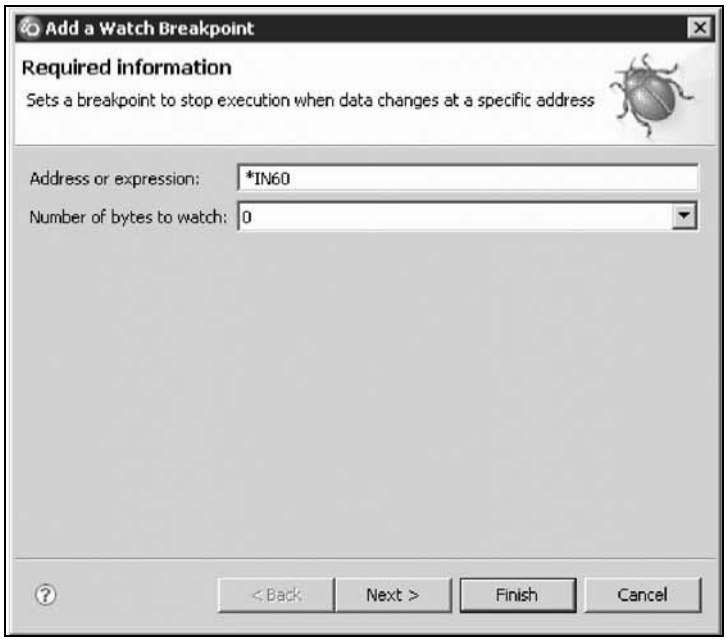
To set a Watch breakpoint:

1. Go to the **Line number** field at the bottom of the source area. In this field enter `116` to go to that line.



2. Double-click variable **`*IN60`** to highlight it.
3. Right-click and click **Add Watch Breakpoint** on the pop-up menu

The Add a Watch Breakpoint window opens. The **Expression** field is pre-filled with the highlighted variable **`*IN60`**.
By default the **Number of bytes to watch** field is set to zero, which means the variable will be watched in its defined length.



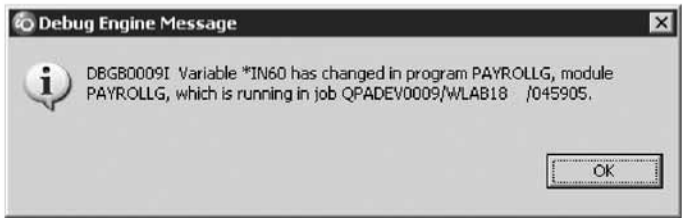
4. Click **Finish**. The Watch breakpoint is now set.
5. Click the **Resume** button on the Debug toolbar.

The application waits for input from the 5250-emulation session.

In the 5250 emulation session, type:

6. `123` for **Project Code** and `D` (for delete) in the **Action Code** field.
7. Press **Enter**.

A message is displayed indicating that the variable ***IN60** has changed.



8. Click **OK**. The program stops at line **465**. This line is located immediately after the statement which

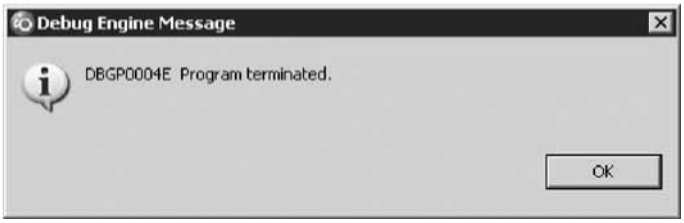
caused the variable ***IN60** to change.

You have added a Watch breakpoint for the variable *IN60 and run the program to see the notification that the variable has changed.

Terminate a debug session

To close the debugger:

1. Click the **Resume** icon on the Debug toolbar. The application waits for input from the 5250 emulation session.
2. Switch to the 5250 emulation session.
3. Press **F3** to end the program. A message **Program terminated** appears:



4. Click **OK**.

Starting the Integrated Debugger using the Debug action

Besides using service entry points to start a debug session, you can start the Debugger in several ways: directly from the pop-up menu of a program or service program in the Remote Systems view, or from a Launch Configurations window. Starting directly from the Remote Systems view without prompt doesn't allow you to specify parameters to be passed to the program. The Launch Configurations window allows you to modify how the program is invoked and to specify parameters.

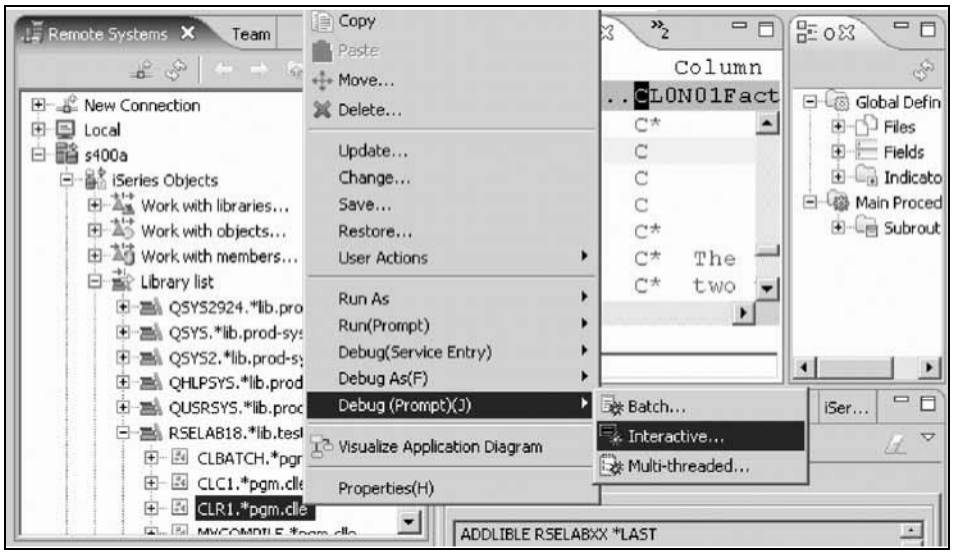
CLR1 requires a parameter.

Before starting the debugger you need to associate a 5250 emulation session with the connection in the same way you did for running the program.

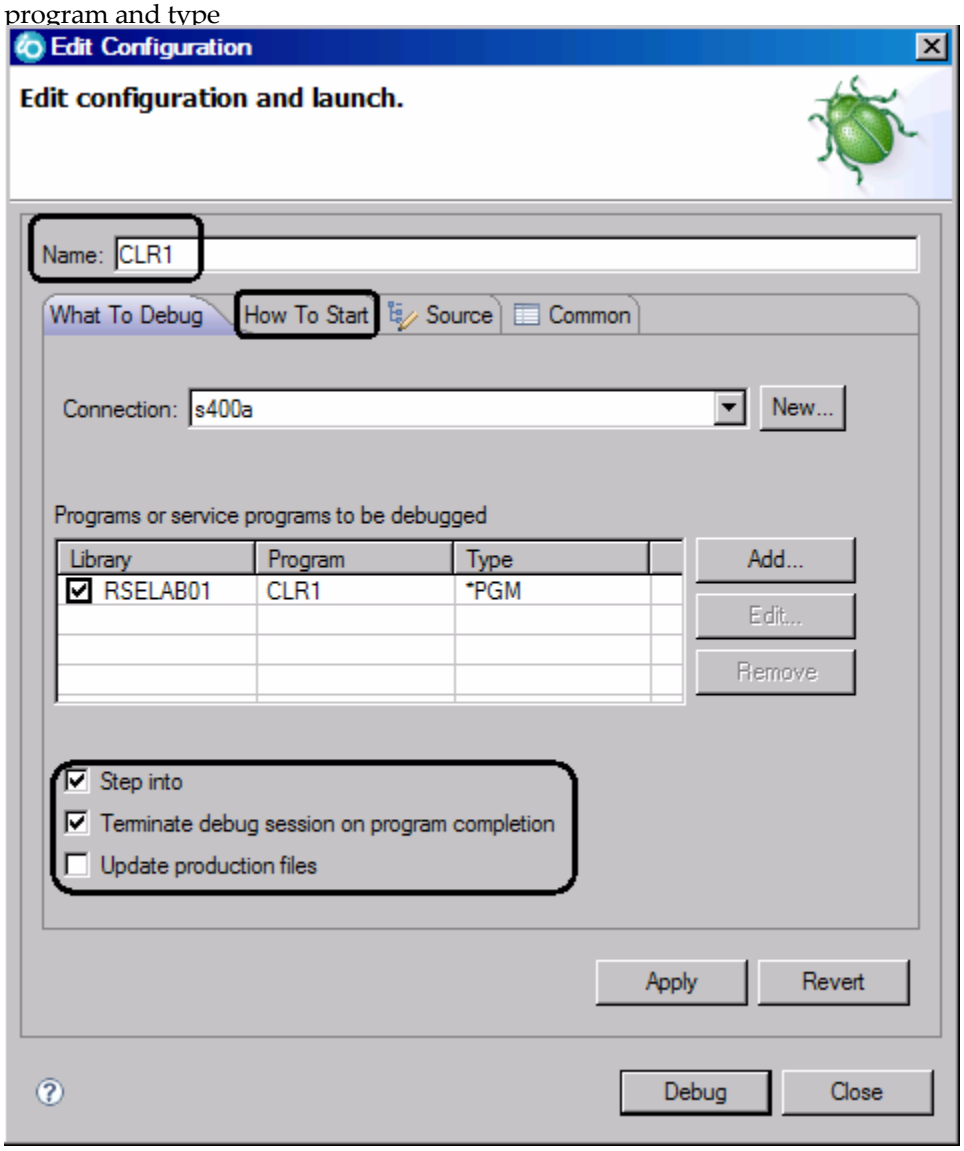
1. In the command line of the 5250 emulation session, type the command `STRRSESVR connectionName`
2. Press **Enter**. The connectionName parameter is the name of your connection defined in the Remote Systems view. This associates the interactive job with the Remote System Explorer communications server.
3. Return to the workbench.

To start the debugger:

1. Select program CLR1 in the Remote Systems view.
2. Right-click and select **Debug (Prompt) > Interactive** from the pop-up menu.



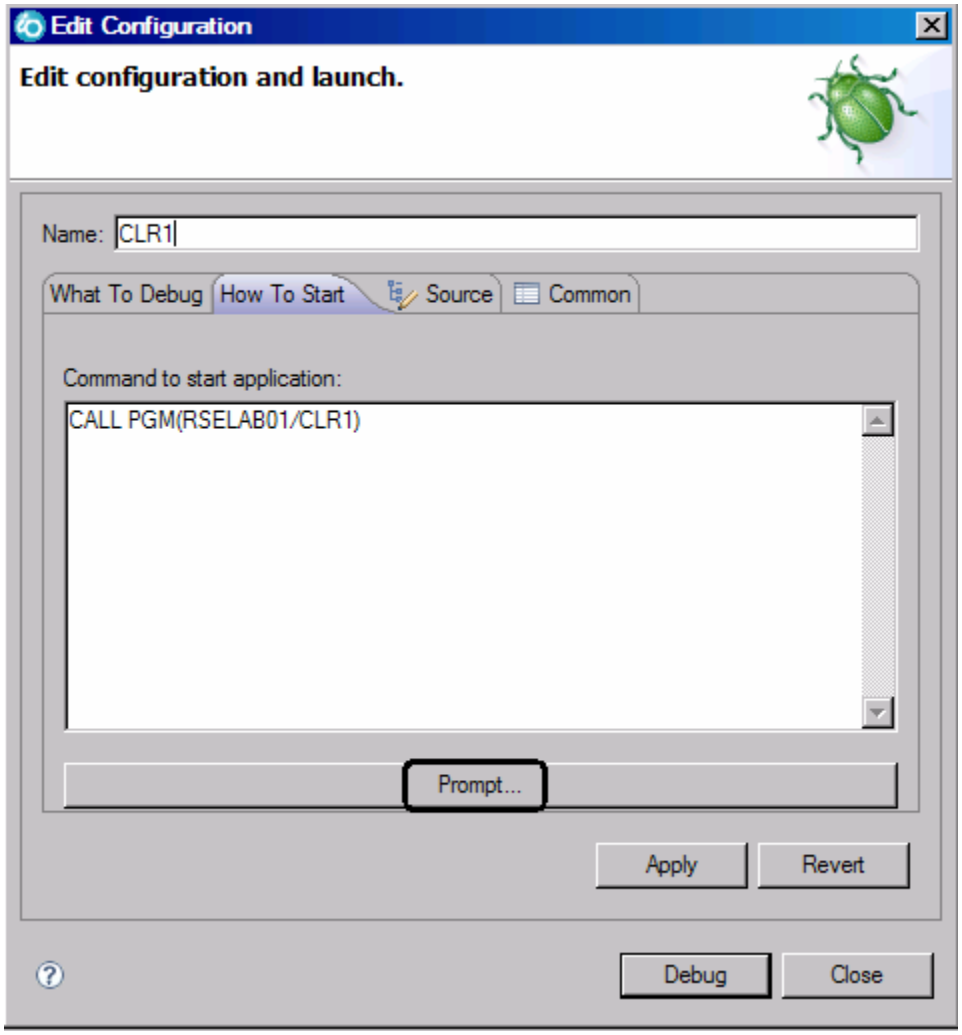
The debugger prompt window opens for the selected program. The connection is prefilled with the name of the current connection. The program to be debugged is also prefilled with the library, program and type



3. In the **Name** field, type the program name `CLR1`. This gives your debug configuration a unique name so you can use it again when you debug this program.

Note: If you invoke the debugger even without prompting, a launch configuration with the default configuration named **My program (Interactive)** will be reused.

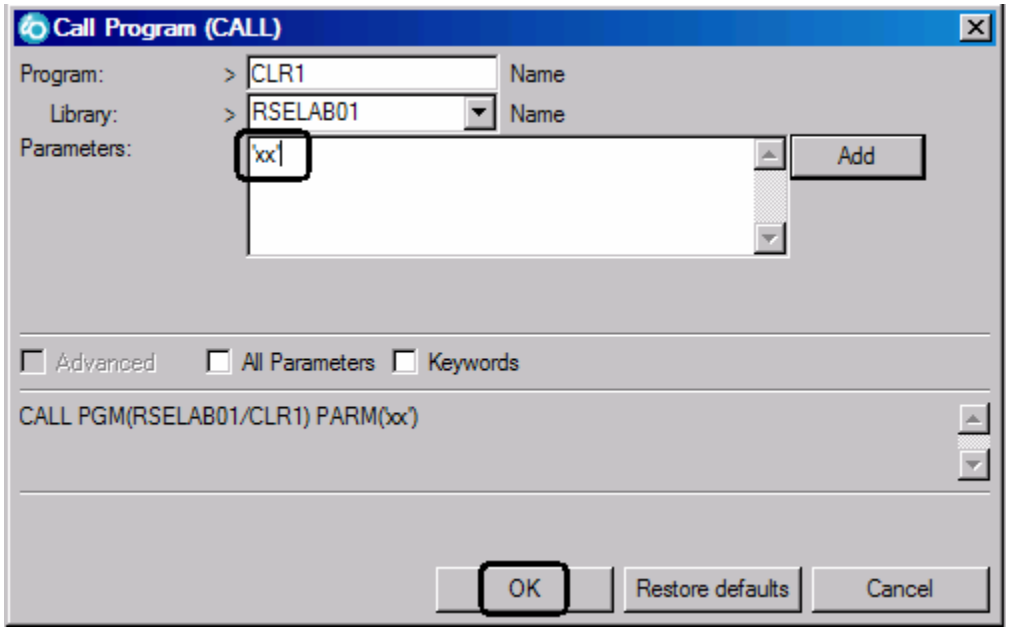
4. You can leave the **Step into** and **Terminate debug session on program completion** check boxes selected and **Update production files** check box deselected, since you are working with a test library.
5. Click the **How To Start** tab.



By default, the page contains a call for the program specified in the **What To Debug** tab.

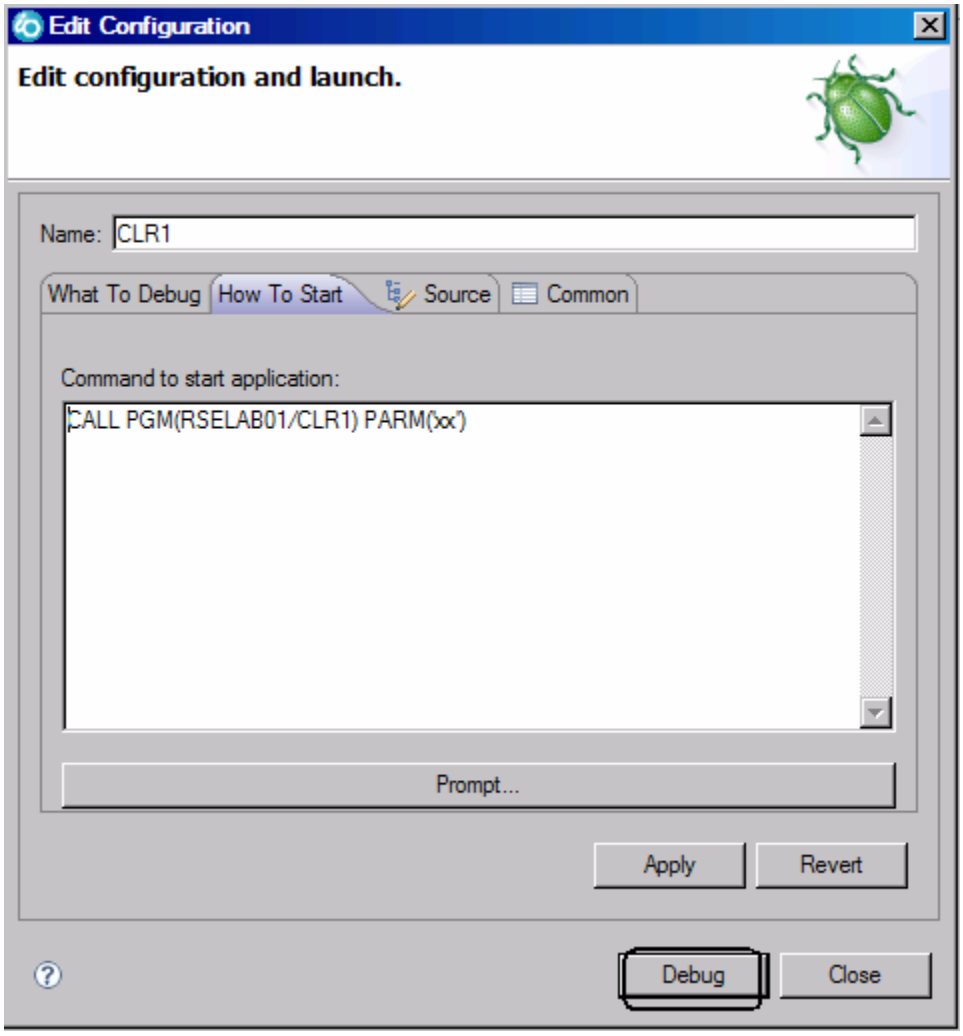
6. Click **Prompt**.

The Call Program (CALL) window opens.



7. In the **Parameters** field, type ' XX' where 'XX' is your workstation number.
8. Click **OK**.

The complete start command for the program appears



9. Click **Debug**.

The Debug perspective opens.

If not, you may see this message.

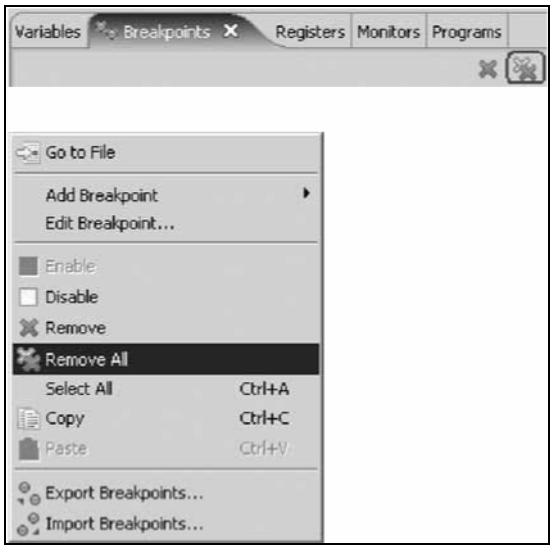
The interactive connection has been shut down in the meantime. Go to your 5250 emulator and restart the interactive connection following the instructions in the message. You don't have to cancel the message. It will be removed as soon as the connection between the Remote System Explorer communications server and the interactive session has been established. The Debug perspective is displayed in the workbench.

Now that the program is active on IBM i and stopped at the first executable statement, the debugger displays the source and the debug functions are available.

You have started an interactive debug session.

10. Remove all breakpoints. In the Breakpoints view, click the Remove All Breakpoints button in the

toolbar or select **Remove All** from the pop-up menu.



11. Click **Resume**

PAYROLLG is called and waits for input from the 5250 session.

Only Terminate and Suspend buttons are available on the Debug view toolbar.

To get control back to the Debug session:

12. Click **Suspend**

You can now set breakpoints and use all the Debug features.

Tip: Suspend is a valuable feature to debug a looping program.

13. Click **Terminate** to end the Debug session.

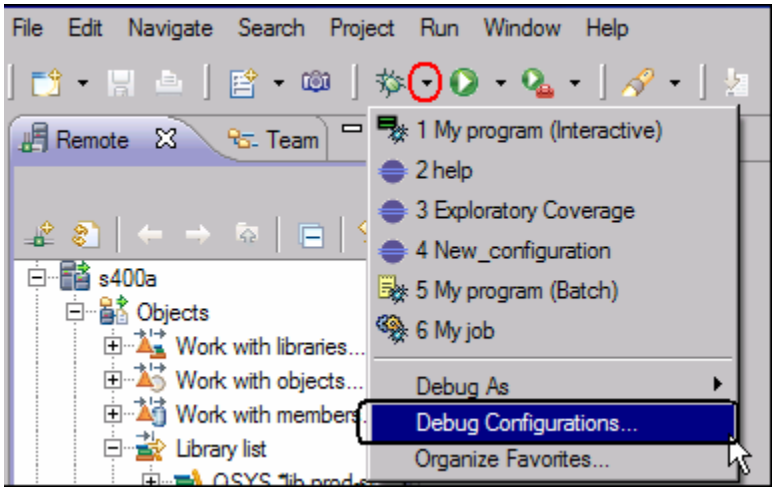
The Debug session is terminated, but this does not end the program.

14. Leave the program running, we will use it again a little later.

15. Switch to the Remote System Explorer Perspective

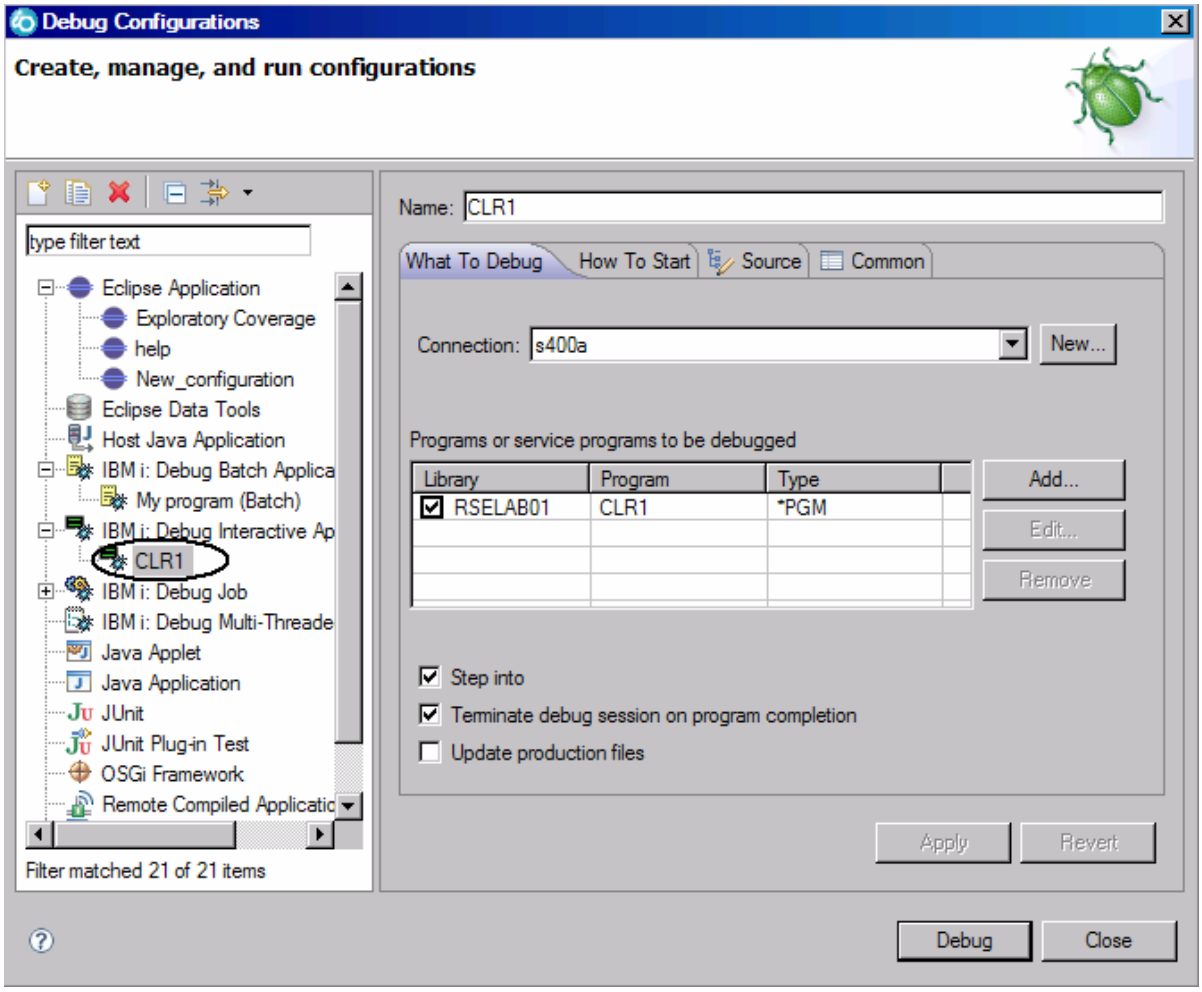
Window > Open Perspective > Remote System Explorer

Tip: You can edit, delete and create debug configurations by clicking the arrow beside the **Debug** icon on the workbench toolbar and selecting **Debug Configurations f**rom the list.



You can also click **Run** on the workbench menu and select **Debug Configurations**.

The Debug Configurations window opens

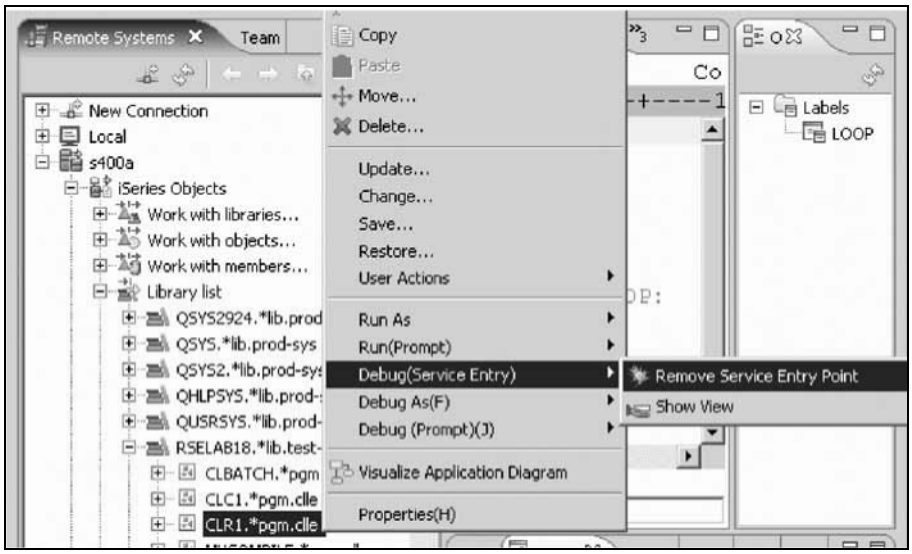


Here you can see the CLR1 configuration that you just created. This is your saved configuration to debug CLR1 as an interactive application. You could now modify this configuration to use a different parameter, copy this configuration, or create a new one. Notice the list of configurations you can choose from.

You are now ready to remove the service entry point you created earlier and close the debug perspective.

To remove the service entry point

16. Click Close in the Debug configuration window, if it is still open.
17. In the Remote System Explorer perspective, expand library RSELABxx, if it isn't expanded already.



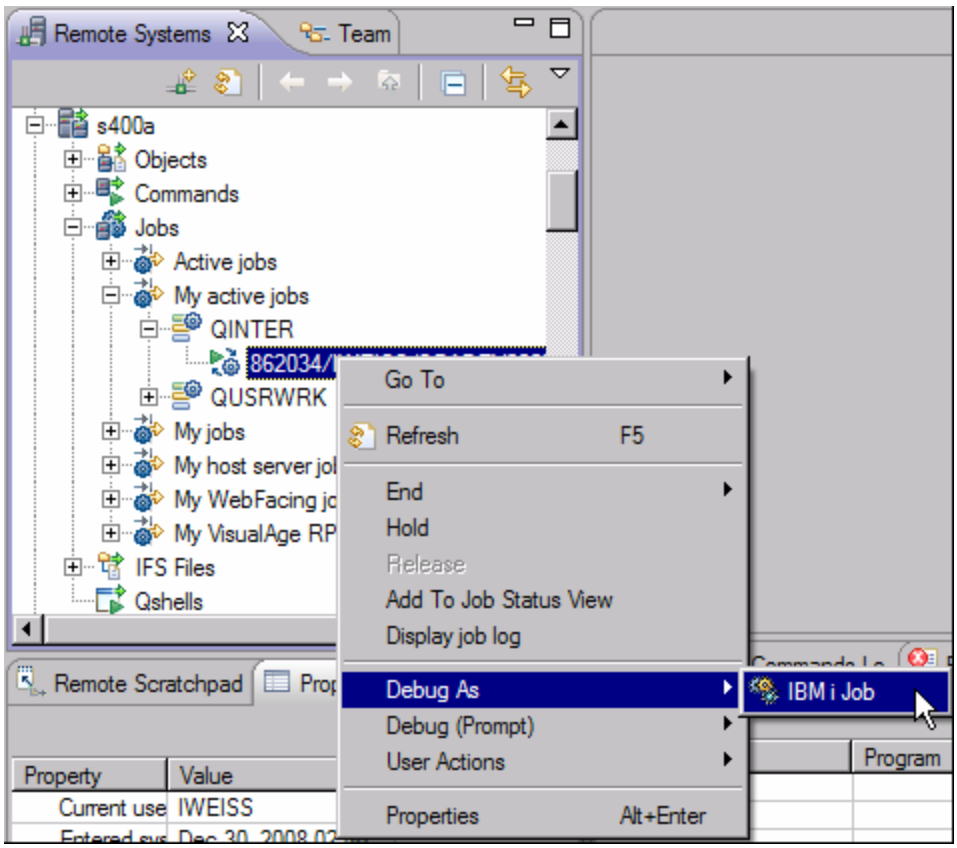
18. Right-click program CLR1 in library RSELABxx
19. Click **Debug (Service Entry) > Remove Service Entry Point**.

The service entry point is removed.

You have started the debugger using a debug action, and removed a service entry point.

Debugging a Job

In addition to being able to debug a program, you can also debug a job. To debug a job:

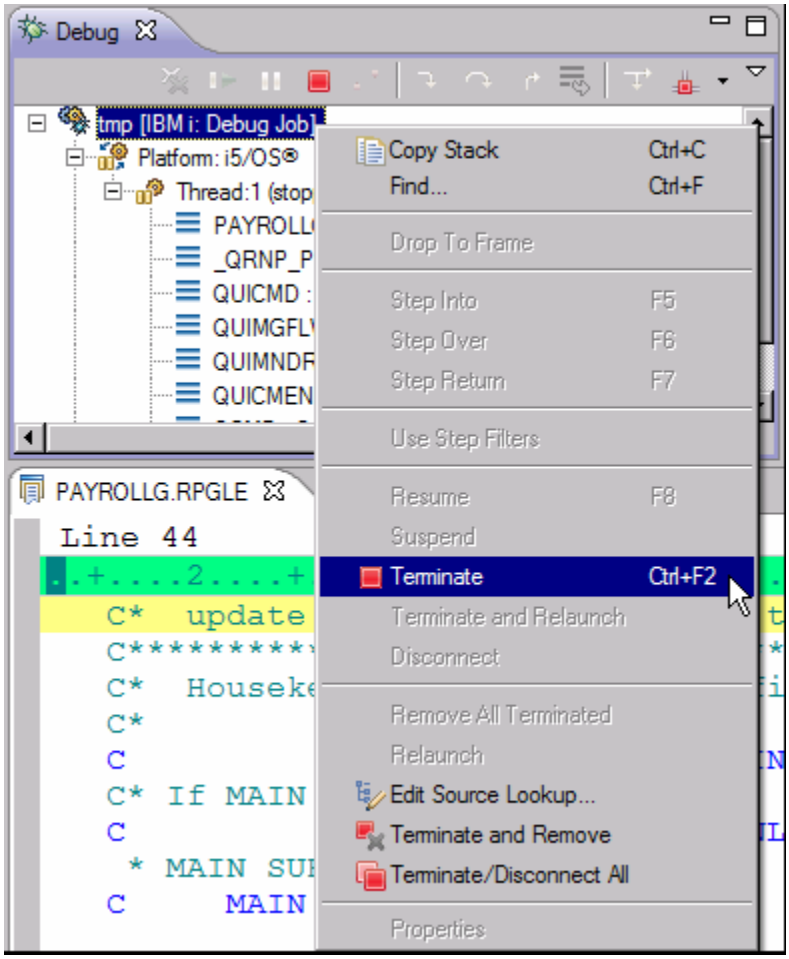


In the Remote System Explorer perspective, under your active server connection, **s400a**,

1. expand **Jobs > My ActiveJobs > QINTER**

2. Right-click the active job under QINTER, and select **Debug As > IBM i Job**.

The debug session begins and connects you to the running application.
Now you can set breakpoints, monitor variables and memory in the same way you did before .



1. Terminate the debug session by right-clicking the job in the Debug view and selecting **Terminate** from the pop-up menu.

The debug session is terminated

Module summary

In this module, you learned how to debug a program using the Integrated IBM i Debugger.

Lessons learned

- Start a debug session using service entry points
- Add a breakpoint
- Add a conditional breakpoint
- Edit a breakpoint
- Monitor a variable in the Monitors view
- Step into your payroll program
- Show a Listing view

- Display source from call stack entries
- View all breakpoints
- Remove a breakpoint
- Monitor memory
- Set a Watch breakpoint
- Close the debugger
- Invoke the debugger from a Debug Configurations window.

Summary

This tutorial has taught you how to maintain a payroll application using the Remote System Explorer. You learned how to start the product and open the Remote System Explorer perspective and how to use tools and views in this perspective to connect to an IBM i system and edit, verify, compile and debug the payroll application

Congratulations you finished the RSE lab
We hope you enjoyed working with the tools in
Rational Developer for i (RDI)

Additional resources

Visit the RPG Café at:
http://www-949.ibm.com/software/rational/cafe/community/rpg

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this documentation in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this documentation. The furnishing of this documentation does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation Licensing
2-31 Roppongi 3-chome, Minato-ku Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OR CONDITIONS OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Intellectual Property Dept. for WebSphere Software IBM Corporation
3600 Steeles Ave. East
Markham, Ontario
Canada L3R 9Z7

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this documentation and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change orwithdrawal without notice, and represent goals and objectives only.

Copyright license

This information contains sample application programs in source language, whichillustrates programming techniques on various operating platforms. You may copy,modify, and distribute these sample programs in any form without payment toIBM, for the purposes of developing, using, marketing or distributing applicationprograms conforming to the application programming interface for the operatingplatform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee orimply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, mustinclude a copyright notice as follows:

Trademarks and service marks

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

- IBM
- iSeries
- Rational
- RPG/400
- System i

Intel® and Pentium® are trademarks of Intel Corporation in the United States, othercountries, or both.

Microsoft, Windows, Windows NT® and the Windows logo are trademarks ofMicrosoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in theUnited States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and othercountries.

Linux is a registered trademark of Linus Torvalds in the United States, othercountries, or both.

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