



# Maintain an ILE COBOL application using Remote System Explorer



---

# Contents

## Maintain an ILE COBOL application using Remote System Explorer. . . . . 1

Introduction . . . . .	1
Starting the product and the Remote System Explorer	2
Starting the product . . . . .	2
Opening the Remote System Explorer perspective	5
Module summary. . . . .	6
Configuring a connection to an iSeries system and connecting to an iSeries. . . . .	7
Configuring a connection to an iSeries system . . . . .	7
Connecting to an iSeries system . . . . .	9
Viewing and accessing objects in the Remote System Explorer . . . . .	13
Opening a second source member and multiple views . . . . .	18
Lesson 2.5: Displaying an outline of a source member . . . . .	20
Module summary . . . . .	21
Editing source . . . . .	22
Introducing the editor . . . . .	23
Changing default editor settings . . . . .	24
Entering SEU commands . . . . .	26
Requesting undo and redo operations . . . . .	27
Invoking language-sensitive help . . . . .	28
Finding and replacing text . . . . .	34
Filtering lines by string . . . . .	35
Filtering lines by type . . . . .	36
Searching multiple files . . . . .	38
Comparing file differences from the Remote Systems view. . . . .	40
Checking syntax. . . . .	44
Understanding your program code with the Application Diagram . . . . .	47
Module summary . . . . .	49
Verifying and compiling source. . . . .	50
Verifying the source . . . . .	51
Compiling source remotely . . . . .	53
Submitting iSeries commands in the iSeries Table view. . . . .	56
Running commands and programs . . . . .	59
Module summary . . . . .	61
Debugging a program . . . . .	61
Introducing the Integrated iSeries Debugger . . . . .	62
Starting a Debug session using a service entry point . . . . .	63
Setting breakpoints . . . . .	66
Monitoring variables . . . . .	68
Stepping into a program . . . . .	71
Listing call stack entries . . . . .	73
Setting breakpoints in PAYROLLD. . . . .	73
Removing a breakpoint in PAYROLLD . . . . .	75
Monitoring variables in PAYROLLD . . . . .	76

Adding a memory monitor . . . . .	78
Setting Watch breakpoints . . . . .	79
Terminate a debug session . . . . .	81
Starting the Integrated Debugger using the Debug action . . . . .	82
Debugging a Job. . . . .	88
Module summary . . . . .	90
Customizing the Remote System Explorer . . . . .	91
More about the Remote System Explorer . . . . .	92
Customizing the perspective. . . . .	92
Saving the perspective . . . . .	104
Resetting the perspective . . . . .	106
Expanding files and folders . . . . .	107
Introducing filters . . . . .	110
Creating a library filter . . . . .	111
Creating an object filter . . . . .	115
Showing Filter Pools . . . . .	118
Sharing filter pools . . . . .	121
Creating a user action . . . . .	123
Creating user actions for jobs . . . . .	130
Customizing compile commands for iSeries Objects . . . . .	132
Using Run configurations . . . . .	137
Module summary . . . . .	142
Designing screens and reports. . . . .	143
Opening a DDS member in the Remote Systems view . . . . .	144
Viewing the DDS tree . . . . .	145
Selecting the DDS object. . . . .	146
Designing the DDS screen . . . . .	148
Creating groups from existing records . . . . .	149
Creating new screens. . . . .	151
Adding fields to the subfile record . . . . .	153
Switching between multiple records. . . . .	156
Adding field error handling . . . . .	159
Accessing field properties . . . . .	160
Adding new keywords . . . . .	162
Verifying the source changes . . . . .	164
Switching between designing and editing the screen . . . . .	166
Compiling your source changes . . . . .	167
Creating a report and closing the Designer . . . . .	167
Module summary . . . . .	172
Introducing the product and Remote System Explorer (optional) . . . . .	173
Introducing Development Studio and Development Studio Client. . . . .	174
Introducing iSeries Application Development Tools . . . . .	175
Summary. . . . .	176
Notices . . . . .	177



---

# Maintain an ILE COBOL application using Remote System Explorer

This tutorial teaches you how to maintain a payroll application written in ILE COBOL 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 iSeries™ system
- Edit, verify, compile and debug a payroll application
- Customize Remote System Explorer
- Design screens and reports

## Time required

3 hours



---

## Introduction

### Learning objectives

- Start the product and open the Remote System Explorer perspective
- Use tools and views in this perspective to connect to an iSeries system
- Edit, verify, compile and debug a payroll application
- Customize Remote System Explorer
- Design screens and reports

### Time required

This tutorial should take approximately 180 minutes to finish. If you explore other concepts related to this tutorial, it could take longer to complete.

### Skill level

Introductory

### Audience

iSeries programmer

### System requirements

- IBM® WebSphere® Development Studio Client for iSeries, V7 and all software updates through the IBM Installation Manager.
- i5/OS® V5R3 or V5R4

## 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
- Restore the RSELABXX savefile (RSELABXX.savf) on an iSeries system:
  - Import the RSELABXX project into your workspace.
  - Open the Project Explorer view by clicking **Window** → **Show View** → **Other** → **General** → **Project Explorer**.
  - Expand the **rselabxx** project.
  - Right-click the **rselabxx.savf** savefile, and select **Restore on iSeries**.

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 Screen design and RSE introduction and you can complete the modules on Debug, Customizing, Screen Design and RSE Introduction in any order after the module on Verify and compile. Each module contains several lessons that must be completed in order for the tutorial to work properly.

## Expected results

Upon completion of this tutorial you will know how to edit, compile and debug an iSeries 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 iSeries application development tasks

### Time required

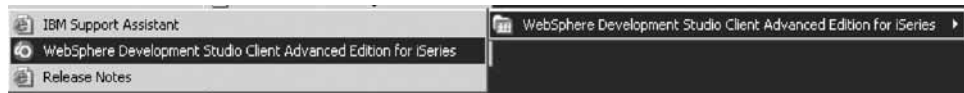
This module should take approximately 10 minutes to complete.

## Starting the product

If you want to know more about the product before you get started you can read “Introducing the product and Remote System Explorer (optional)” on page 173.

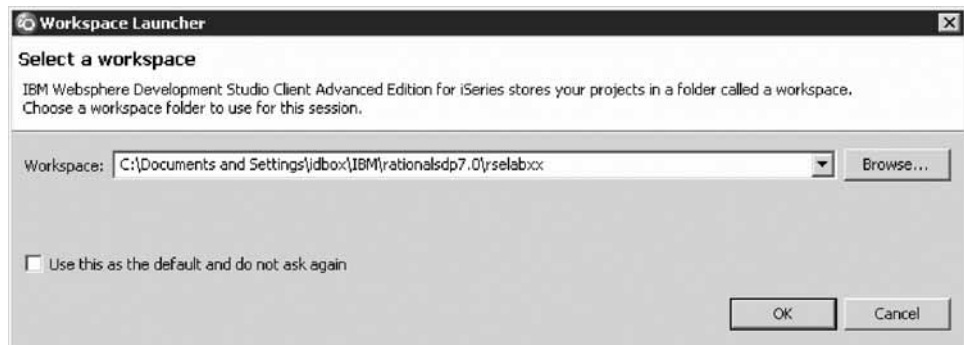
First you must start the product. Follow these steps to start the product:

1. Click **Start** on the task bar of your desktop.
2. Select **Programs > IBM Software Development Platform> IBM WebSphere Development Studio Client for iSeries> WebSphere Development Studio Client for iSeries**



If you are working with the Advanced Edition of the product you will see the words Advanced Edition in the product name.

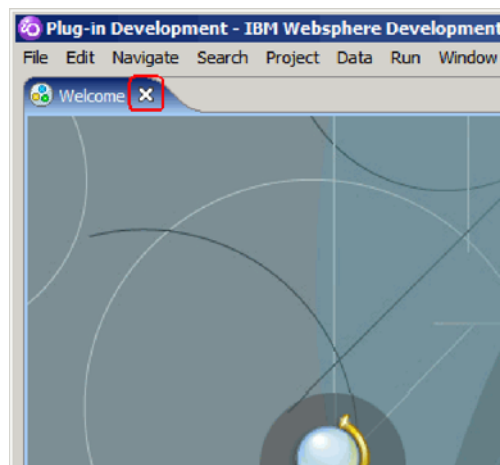
A dialog will appear. Here you specify the directory of the workspace where your projects and other resources such as folders, subfolders and files that you are developing in the workbench will reside.



3. (Optional) Change the field in this dialog and use a unique directory name, for example, RSELABxx (where xx is a unique number).
4. Click **OK** to open the workbench.



5. 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.

**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.

### Lesson checkpoint

You learned the following:

- About workspaces
- About the workbench
- How to start the product

## Opening the Remote System Explorer perspective

Now you are ready to open the Remote System Explorer (RSE) perspective.

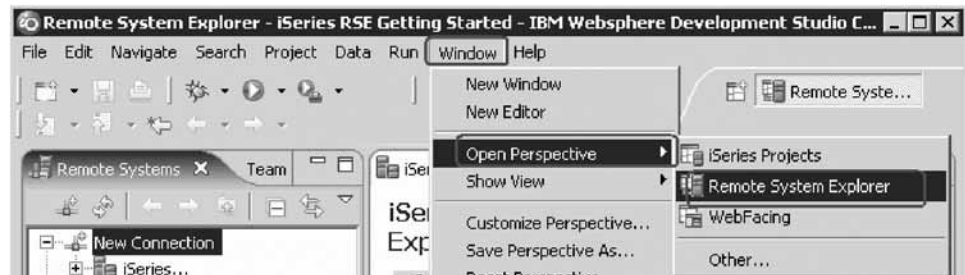
1. Check for the name of the perspective.



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 an iSeries programmer to 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 iSeries 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 iSeries programmer tools and use the views from the workbench.

You have opened the Remote System Explorer perspective.

### Lesson checkpoint

You learned the following:

- About perspectives
- About the RSE perspective
- How to open the RSE perspective

## Module summary

You have learned how to start the product, and open the RSE perspective.

### Lessons learned

By completing this module, you learned about the following concepts and tasks:

- About workspaces
- About the workbench

- About perspectives
- About the RSE perspective
- How to start the product
- How to access unique tools and views targeted towards iSeries application development tasks

### **Assessment**

- What is a workspace?
- What is the workbench?
- What are perspectives?
- What is the RSE perspective?
- How do you start the product?
- How do you open the RSE perspective?

---

## **Configuring a connection to an iSeries system and connecting to an iSeries**

This module teaches you how to create a connection to an iSeries 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, iSeries Table view, and the Outline view.

### **Learning objectives**

- Create a connection to an iSeries system
- Connect to an iSeries system
- 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 iSeries Table view
- Customize the columns in the iSeries Table view
- Open a member for edit from the iSeries Table view or the Remote Systems view
- Maximize the editor window
- 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

### **Time required**

This module should take approximately 10 minutes to complete.

## **Configuring a connection to an iSeries system**

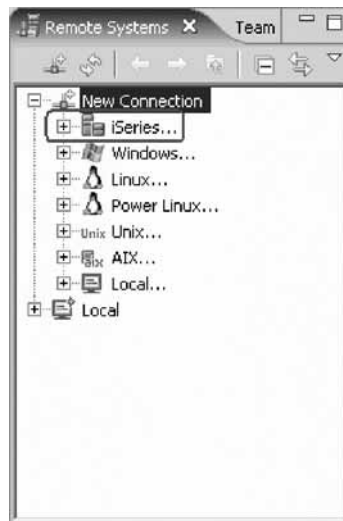
When you first open the Remote System Explorer, you are not connected to any system except your local hard drive on your workstation. To connect to a remote iSeries system, you need to define a connection. When you define a connection, you specify the name or IP address of the remote system and you give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

The first time you connect to an iSeries system, you need to specify a profile. All connections, filters, and filter pools belong to profiles. 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.

1. In the Remote Systems view, **New Connection** is automatically expanded to show the various remote systems types you can connect to through the Remote System Explorer.



Click the plus sign + beside **iSeries** to configure a connection to an iSeries system.

The Name personal profile page opens.



2. Click **Next** to accept the default value. The profile defaults to the name of the workstation. Your profile will be different from the one shown here.

The Remote iSeries System Connection page opens.

On this second page you specify the information for your connection. The cursor on this page is positioned in the **Host Name** field.

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

You have configured a connection to an iSeries system.

### Lesson checkpoint

You learned the following:

- About configuring a connection
- How to configure a connection to an iSeries system

## Connecting to an iSeries system

After you configure a connection to an iSeries system, you can easily connect and expand your new connection to reveal your 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.

### iSeries Objects

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

### iSeries Commands

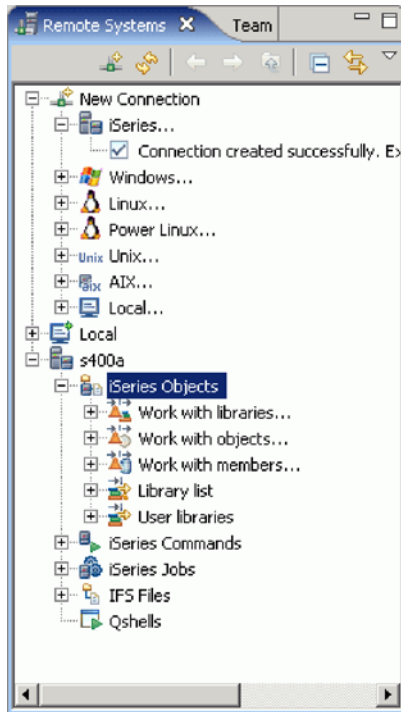
Contains predefined commands and allows you to create command sets each of which contain one or more often used commands. When run, all commands in a command set are sent to the remote system and executed, and the results are logged in the iSeries Commands log.

### iSeries 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 iSeries system.



To connect to an iSeries system:

1. In the Remote Systems view, your new connection is expanded to reveal your subsystems. The **iSeries 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.

Notice the first three entries under the **iSeries 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)

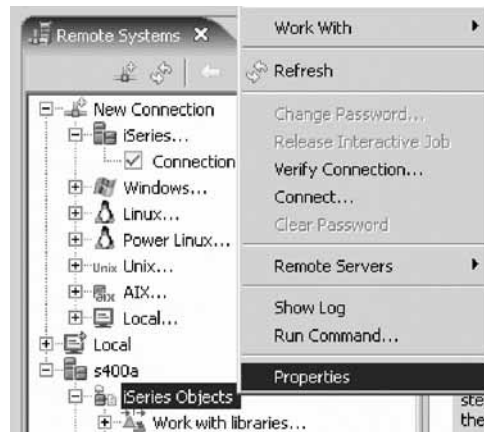
In addition there are entries for working with library lists and user libraries:

- **Library list** (to simulate PDMs 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 iSeries server.)

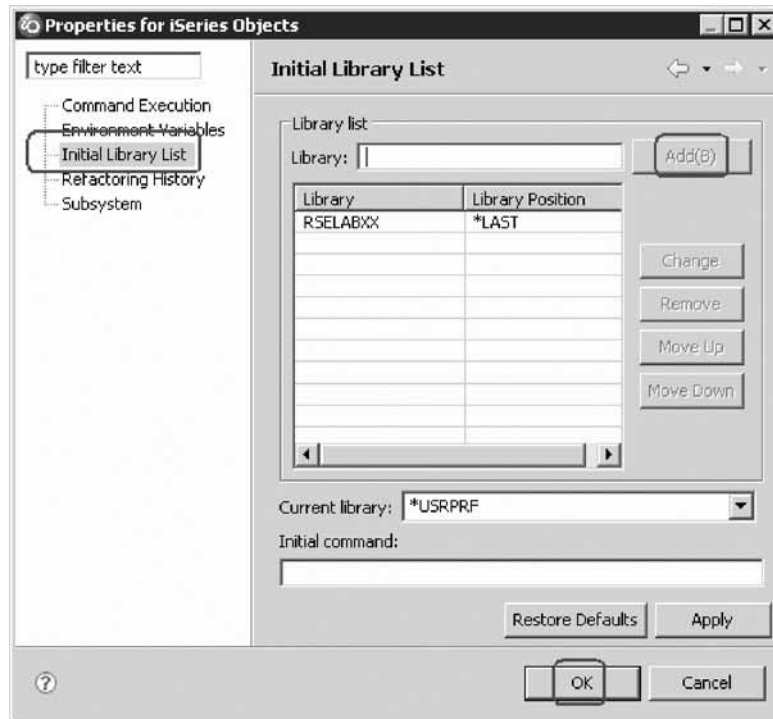
You also have more entries to work with under the connection itself and you can see from these entries that Remote System Explorer goes well beyond PDM! It allows you to explore iSeries jobs and commands and the IFS file system.

2. 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 **iSeries Objects** and click **Properties** on the pop-up menu.



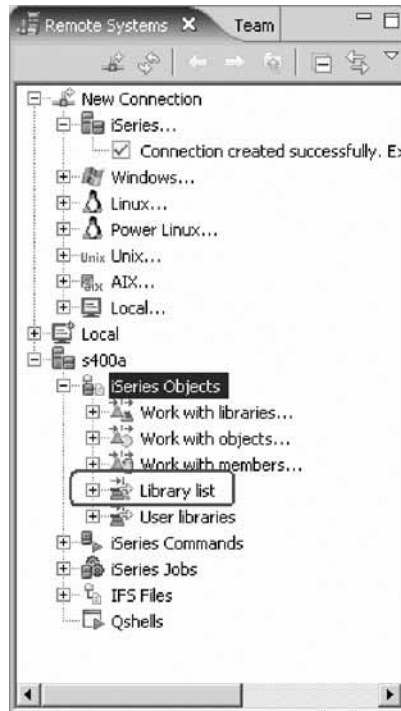
- b. Select **Initial Library List** on the left pane.
- c. Type RSELABxx where XX is a unique number in the **Library** field and click **Add**.



- d. Click **OK**.  
This will add the library RSELABxx to your library list every time you use this connection.

**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 iSeries Objects subsystem. These changes are only valid until you disconnect.

3. Expand the **Library list** folder.



Now the connection will be activated and you will be prompted for a user ID and password.



4. Enter your user ID and password.
5. Select the **Save user ID** check box.
6. Select the **Save password** check box.
7. Click **OK**.

As you know, you can use the properties of any of the subsystems to set connection information such as adding a library to the library list.

Back in the workbench 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 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 iSeries system and used the Remote Systems view to view libraries in the library list.

### Lesson checkpoint

You learned the following:

- About subsystems
- About the iSeries Objects subsystem
- How to connect to an iSeries system

## Viewing and accessing objects in the Remote System Explorer

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

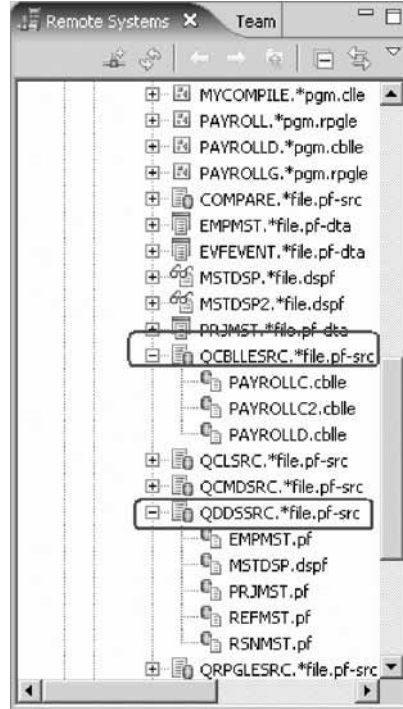
To view and access an object:

1. Expand library RSELABxx.

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. Drill-down through the files in the Remote Systems view until you find QDDSSRC source file and then expand it.

3. Scroll up through the files in the Remote Systems view until you find QCBLESRC source file and expand it as well.

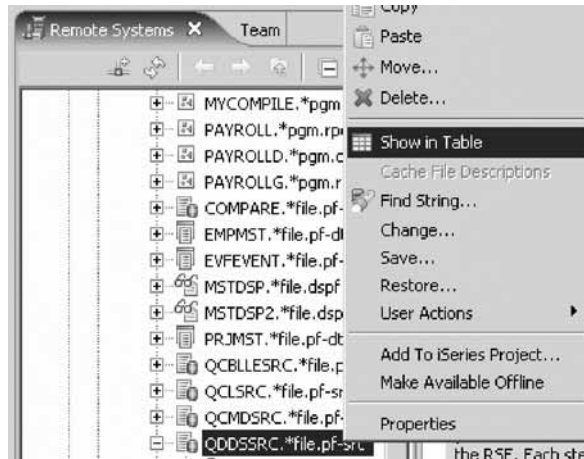


Now you can see and access the members in these two source files. For each member you can right-click and select from a number of actions. The exact list of actions depends on whether the member is a data file or source file and whether you select one or multiple members. For a COBOL source member, the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

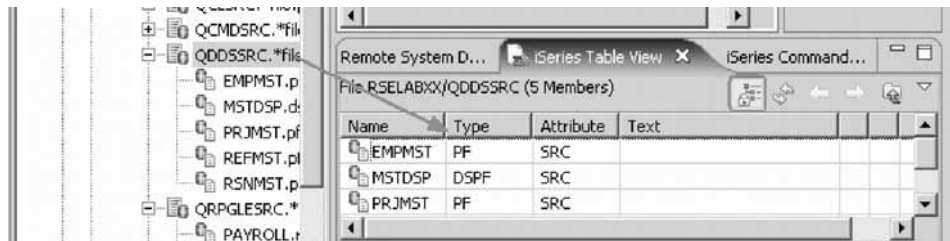
- open with
- browse with
- verify
- compile


Before you go ahead and work with these members, let's see the members in the iSeries 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.


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



The iSeries 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.



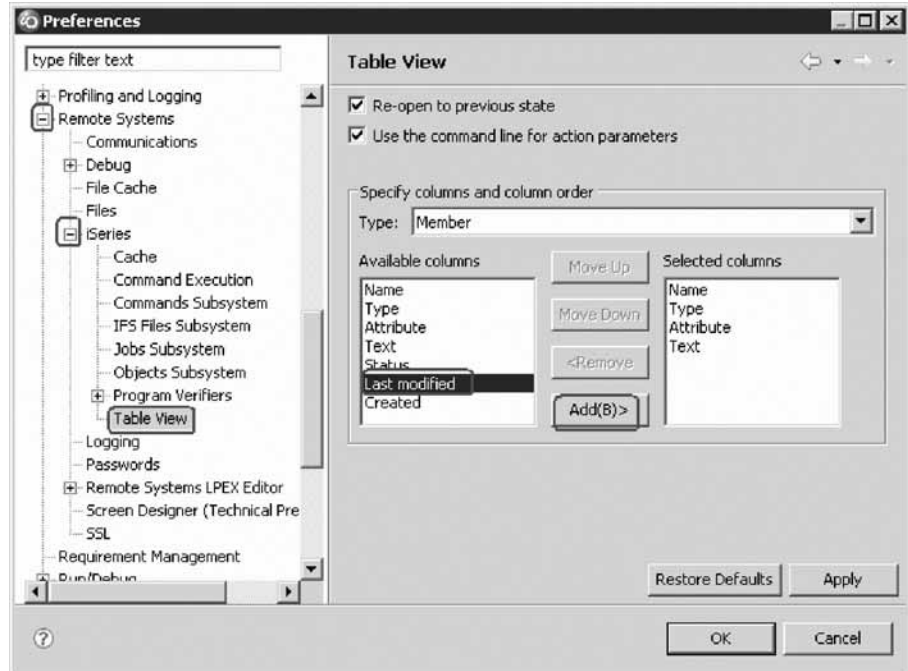
The top of the iSeries Table view contains a lock icon  that controls the correlation between the Remote Systems view and the iSeries Table view. If the lock is disabled then whenever you click an object or library in the Remote Systems view, the associated contents of that item automatically populate the iSeries 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 iSeries 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.

5. In the iSeries 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.

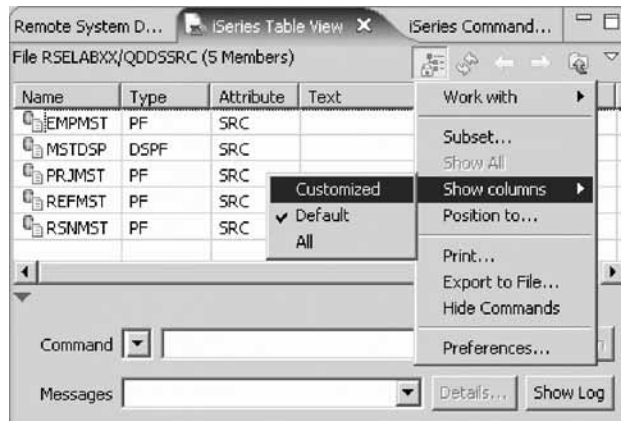
This means now the table will automatically be updated when a different object is selected in the Remote Systems view. This is a shortcut to open the pop-up menu for an object in the Remote Systems view and to select Show in Table.

You can also modify which specific columns you want to see in the iSeries Table view.

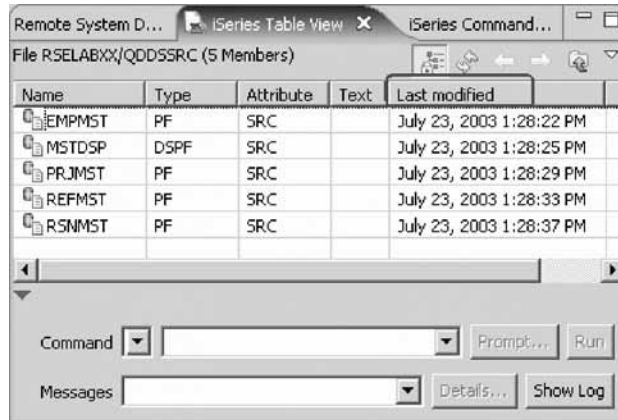
6. To modify the iSeries Table properties:
  - a. Click **Window > Preferences** from the workbench menu.  
The Preferences Window opens.



- b. In the left pane of the Preferences window, expand **Remote Systems**.
- c. Expand **iSeries** under Remote Systems.
- d. Click **Table View** under iSeries.
- 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 iSeries Table view. Click the down arrow on the iSeries Table view title bar.

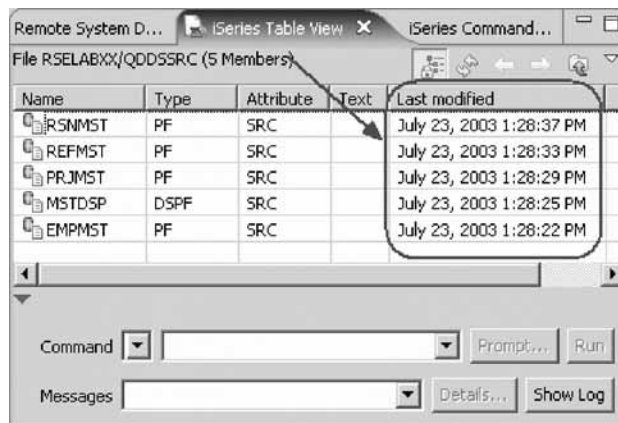


- i. Click **Show columns > Customized** in the pop-up menu. Now you'll see the extra column that you've added.



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

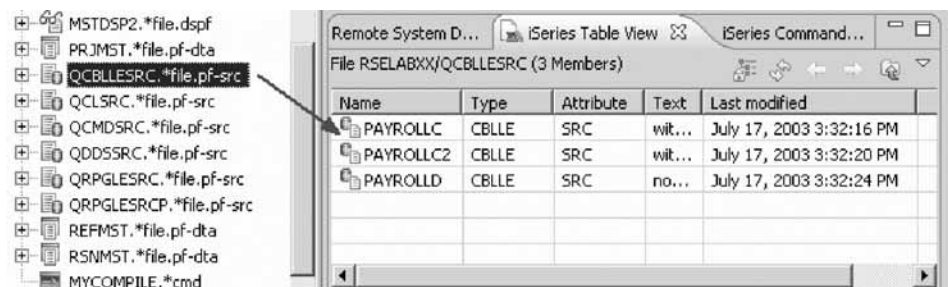
- j. To sort the objects in ascending order by Last modified, click on the heading.



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

7. In the Remote Systems view, select QCBLLESRC.

The table shows the members in QCBLLESRC.



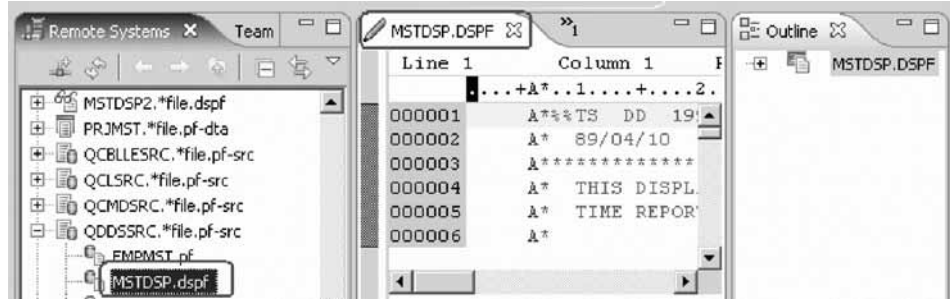
Now you are ready to use the Remote Systems LPEX Editor to edit the member MSTDSP found in QDDSSRC.

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 iSeries Table view.

The Remote Systems LPEX Editor opens. It is built right into the workbench, with rich editing functions and is iSeries 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.

### Lesson checkpoint

You learned the following:

- About member actions
- About the iSeries Table view

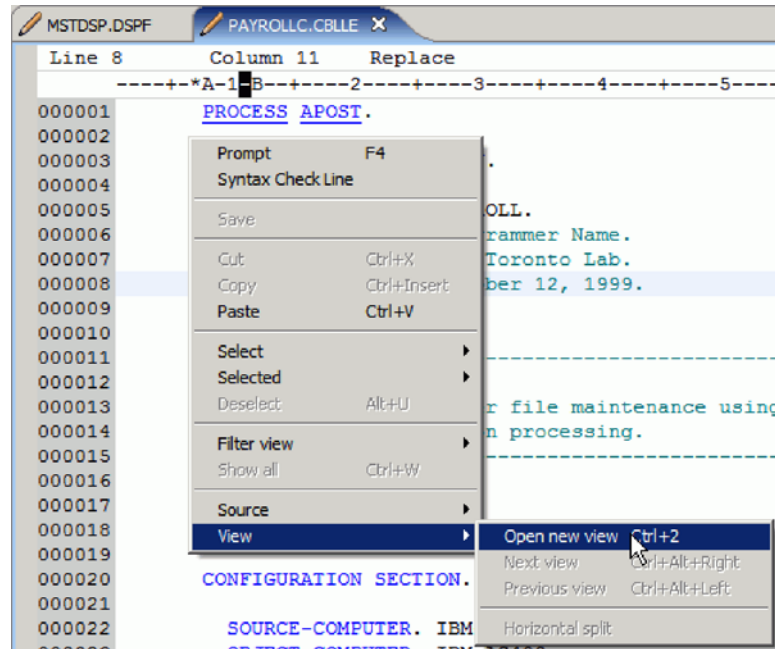
## Opening a second source member and multiple views

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

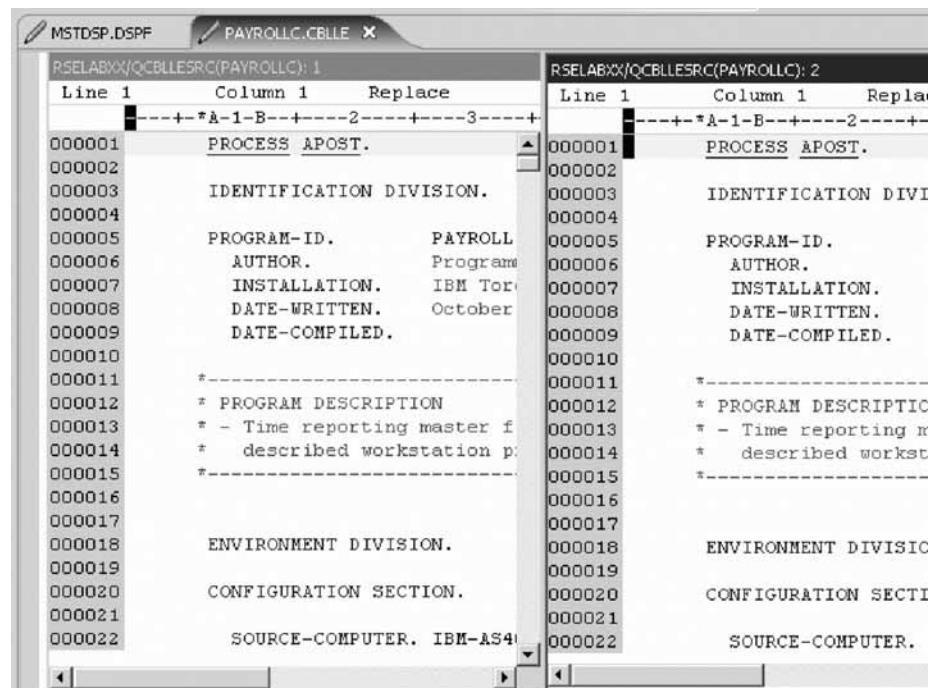
To open a second source member:

- 1.
2. In the Remote Systems view, double-click member **PAYROLLC** in the **QCBLLSRC** source file.
3. Click on each tab to switch from one edit session to another edit session.
4.
  - a.
  - b. Double-click the **PAYROLLC** tab in the editor to maximize the Editor window.
  - c. 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.



- d. 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.

- e. 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.

```

RSELABXX/QCBLLESRC(PAYROLL): 1
Line 1      Column 1      Replace
-----+*A-1-B--+-----2-----3-----4-----5-----6-
000001      PROCESS  APOST.
000002
000003      IDENTIFICATION DIVISION.
000004
000005      PROGRAM-ID.          PAYROLL.
000006      AUTHOR.             Programmer Name.
000007      INSTALLATION.       IBM Toronto Lab.
000008      DATE-WRITTEN.       October 12, 1999.

RSELABXX/QCBLLESRC(PAYROLL): 2
Line 1      Column 1      Replace
-----+*A-1-B--+-----2-----3-----4-----5-----6-
000001      PROCESS  APOST.
000002
000003      IDENTIFICATION DIVISION.
000004
000005      PROGRAM-ID.          PAYROLL.
000006      AUTHOR.             Programmer Name.
000007      INSTALLATION.       IBM Toronto Lab.
000008      DATE-WRITTEN.       October 12, 1999.

```

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

## Lesson checkpoint

You learned the following:

- About multiple views
- How open a second source member
- How to open multiple views

## Lesson 2.5: Displaying an outline of a source member

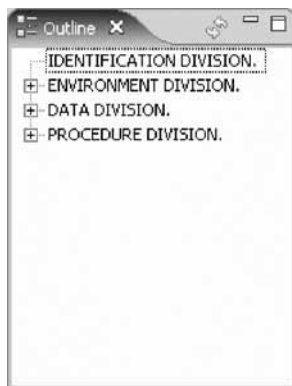
The Outline view acts as an excellent resource when you want to edit RPG, COBOL and DDS source in the Remote Systems LPEX editor. The Outline view displays a structural outline of items defined in the file that you currently have open in the Remote Systems LPEX editor window. With the editor active,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

1. Look at the Outline view to the right of the editor window. If you have closed the Outline view, you can reset the perspective by selecting **Window > Reset perspective** from the workbench menu or **Show view > Other** then expand Basic and click Outline in the Show view dialog.

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

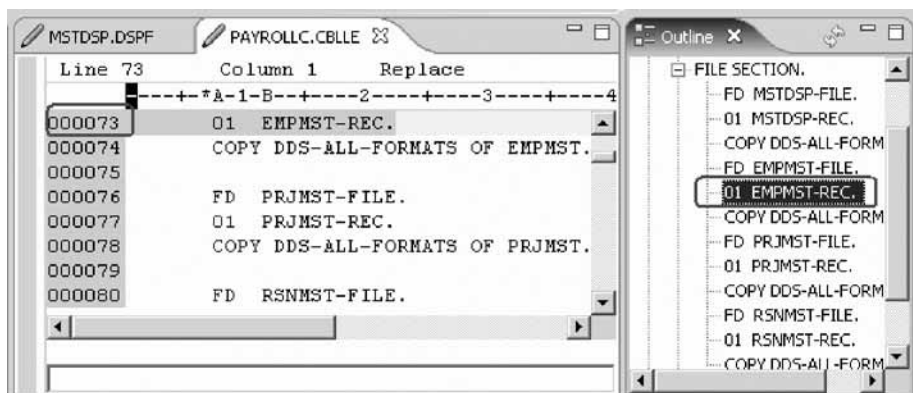




Now you want to see more details of your source member.

2. Expand ENVIRONMENT DIVISION.
3. Expand the CONFIGURATION SECTION.
4. Expand DATA DIVISION.
5. Expand FILE SECTION.
6. Double-click on any of the entries in the Outline view.

This will position the source editor accordingly.



7. Stay in the **PAYROLLC** tab to get the PAYROLLC Editor window in focus for the next exercise.

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

### Lesson checkpoint

You learned the following:

- About the Outline view
- How to display an outline of a source member

## Module summary

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

## Lessons learned

- Create a connection to an iSeries system
- Connect to an iSeries system
- 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 iSeries Table view
- Customize the columns in the iSeries Table view
- Open a member for edit from the iSeries Table view or the Remote Systems view
- Maximize the editor window
- Open another member for edit
- Open multiple views of the same source member
- Switch from one edit session to another edit session
- Display a structural outline of items defined in a source member

## Assessment

- How do you connect to a remote iSeries system?
- What are subsystems?
- What is the iSeries Objects Subsystem?
- How do you add a library to your library list?
- How do you find files in your library?
- What is the iSeries Table view?
- What does the lock icon do in the iSeries Table view?
- How do you view members in the iSeries Table view?
- How do you open a member of edit?
- How do you maximize the Editor window?
- What is the Outline view?
- How do you display an outline of a source member?

---

## Editing source

This module teaches you how to edit ILE COBOL source member PAYROLLC, 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
- 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 view to understand your program code

### **Time required**

This module should take approximately 45 minutes to complete.

## **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 iSeries 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

You have learned about the features of the editor.

### **Lesson checkpoint**

You learned the following:

- About the LPEX Editor

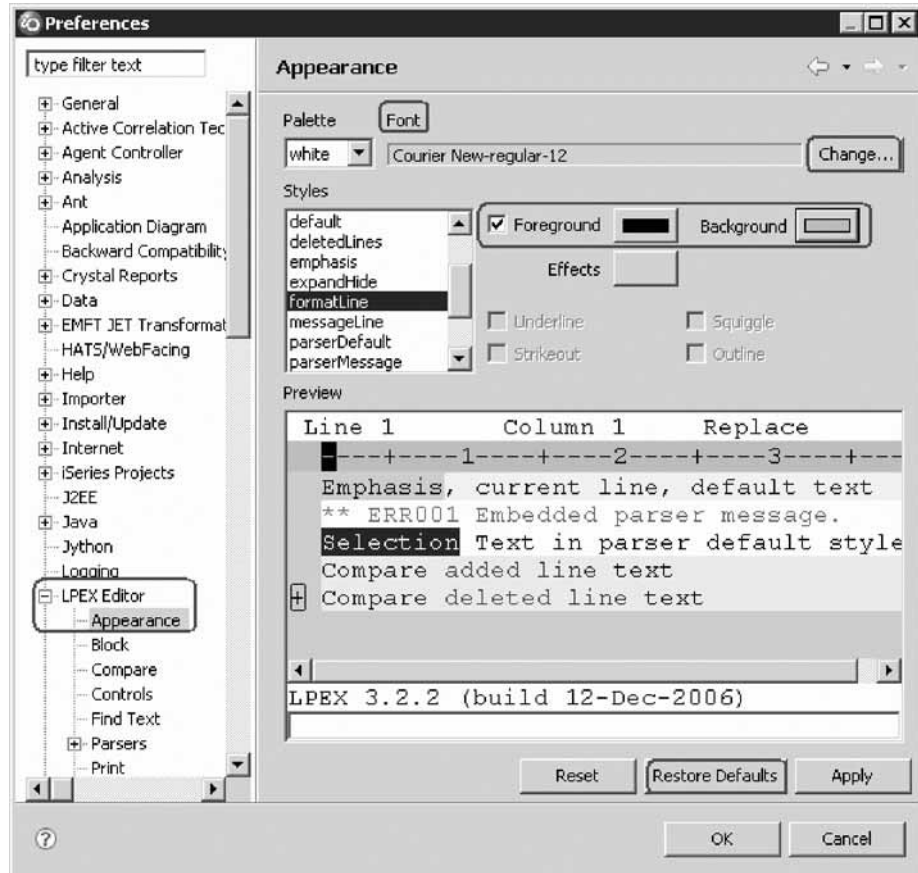
## Changing default editor settings

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 Appearance and User Key Actions.

- 1.
2. To change the editor appearance:
  - a. In the left pane of the Preferences window, expand **LPEX Editor**.
  - b. Select **Appearance** under LPEX Editor.
  - c. Select **formatLine** under the **Styles** list.
  - 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.

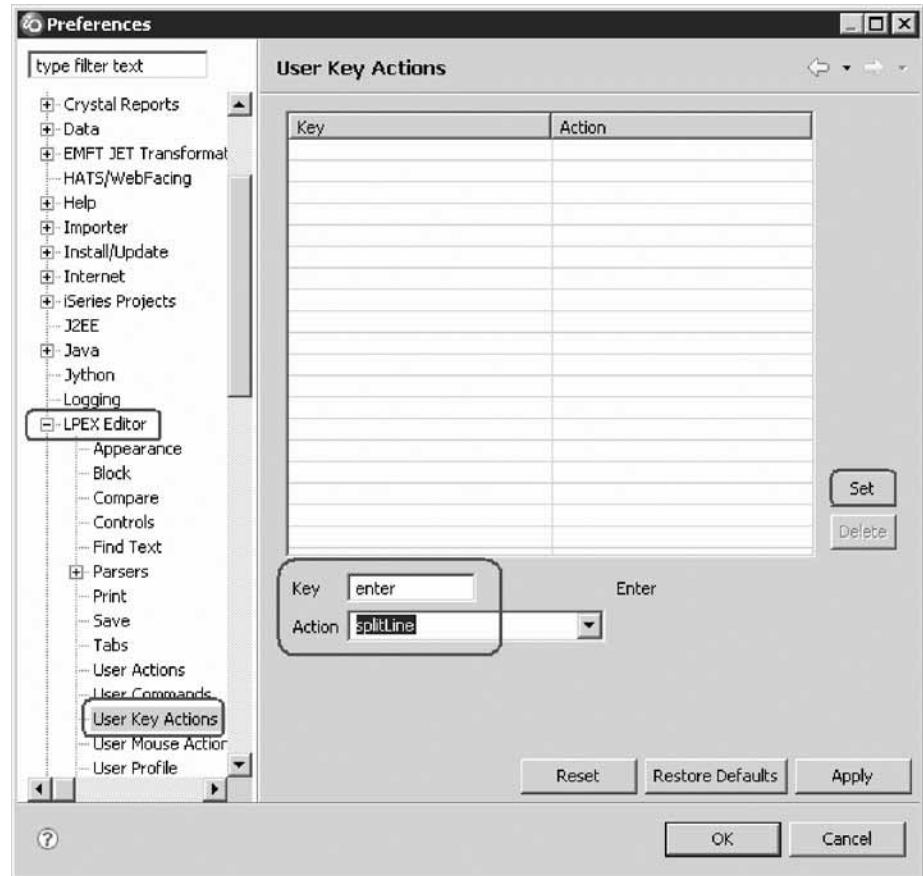


- g. Select **currentLine** under the **Styles** list.  
This option highlights the line that the cursor is on. The option applies to all source files opened in the editor area.
  - h. Change the **Background** color to light yellow.
  - i. If you don't like the changes you made, you can click **Restore Defaults** to return to the original settings.
3. To autosave while editing source:

To enable or disable autosave while editing source in the Remote Systems LPEX Editor, you need to change an editor preference. You use the **Window > Preferences > Remote Systems > Remote Systems LPEX Editor** and click the check box **Autosave**.

The default value for the minutes is set to 5. You can specify a value between 1 and 60 minutes.

4. To modify the default behavior of the Enter key:
  - a. Expand **LPEX Editor** if not already expanded then select **User Key Actions**.



- b. 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.

- c. Type splitLine in the **Action** field.
    - d. Click **Set**.
    - e. Click **OK** on the Preferences window.
    - f. Return to the Editor window.

Next let's see the results of column sensitive editing.

5. To see the results of splitLine:
  - a. 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.

## Lesson checkpoint

You learned the following:

- How to change some of the default editor settings

## 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:

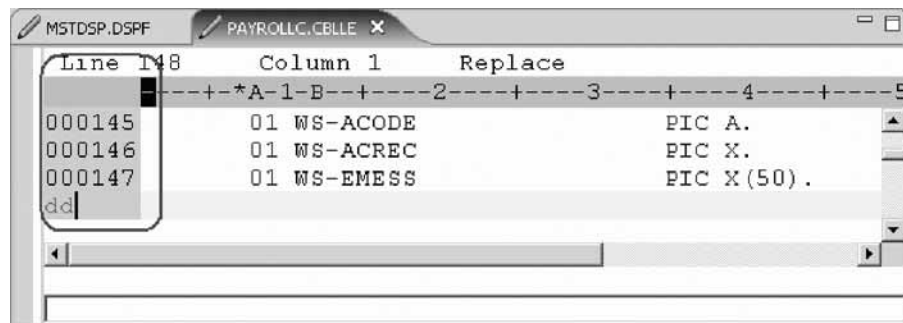
- ispf
- seu
- xedit

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.

## Lesson checkpoint

You learned the following:

- About editor profiles
- How to use SEU prefix area commands

## 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.

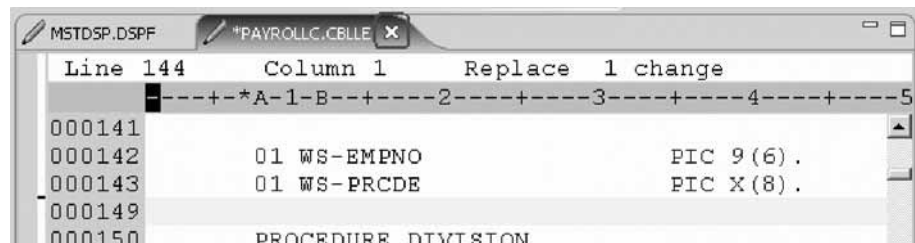
To undo and redo edit changes:

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.

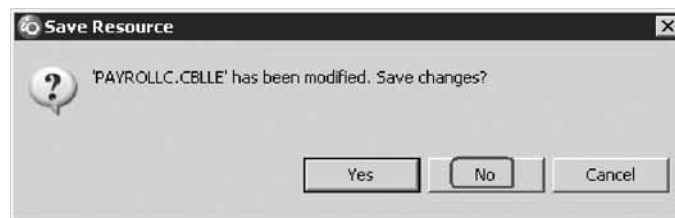
At this point you will reload the source from the iSeries to make sure that it is back in its original form.

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

**Tip:** You can also click the X on the **PAYROLLC** 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 **PAYROLLC** member in the **QCBLLESRC** file.

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

### Lesson checkpoint

You learned the following:

- About undo and redo operations
- How to undo and redo editing changes

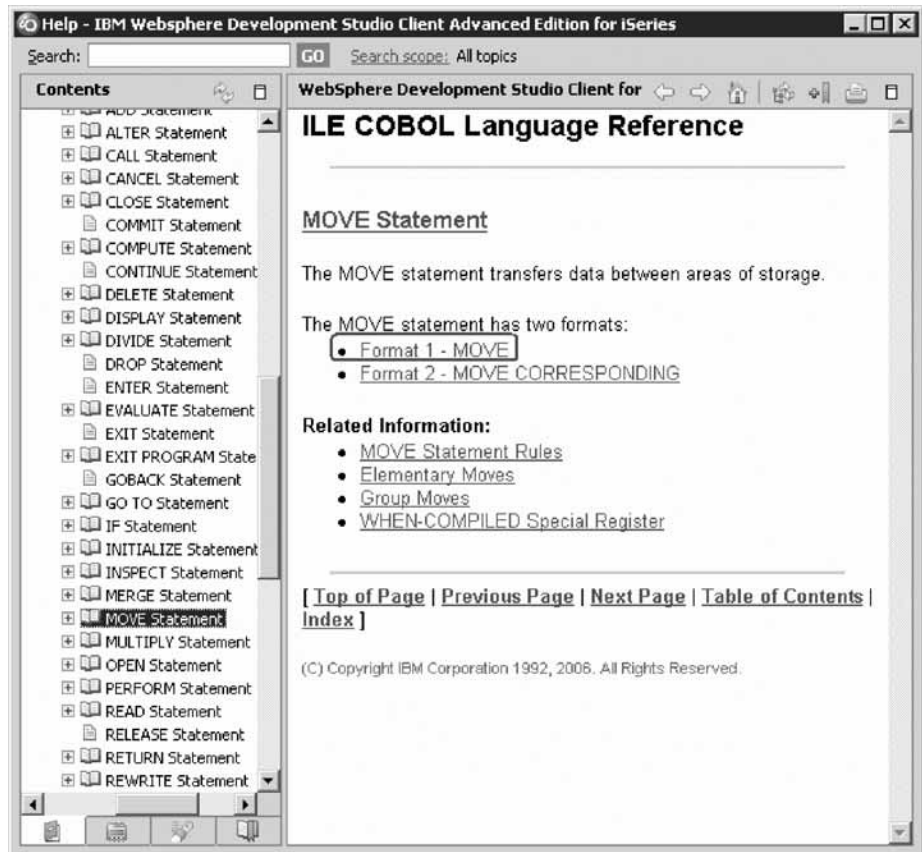
## Invoking language-sensitive help

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

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.

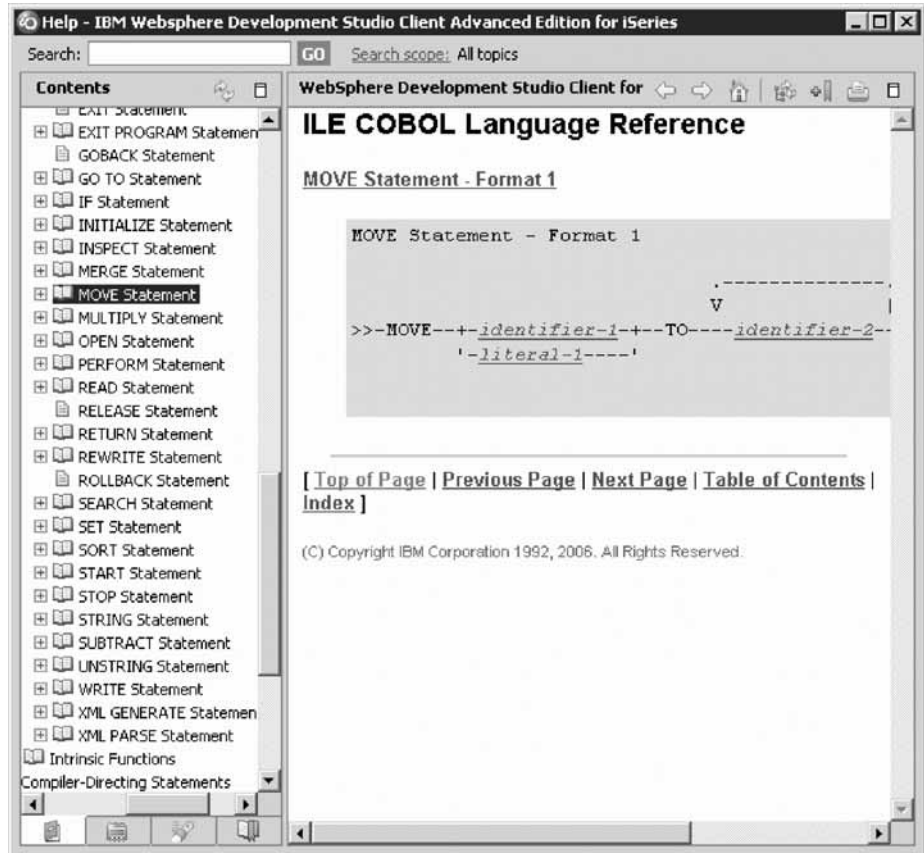
1. To access language sensitive help:
  - a. Position the cursor over the word MOVE in line 231 of the ILE COBOL source.
  - b. Press **F1**. Language-sensitive help for the MOVE operation code appears in a Help window.

Text marked in blue in the Help window contains the link to detailed information about the topic in blue.

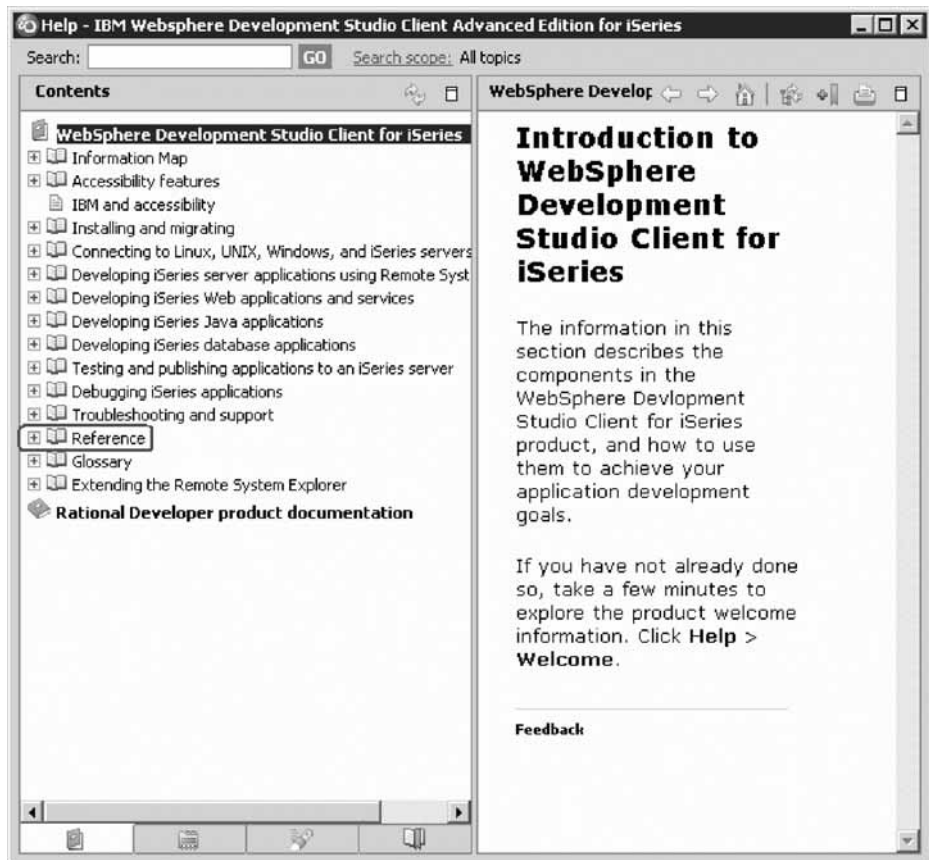


- c. Click the link Format 1- MOVE.  
The Help page for Format 1- MOVE is displayed.

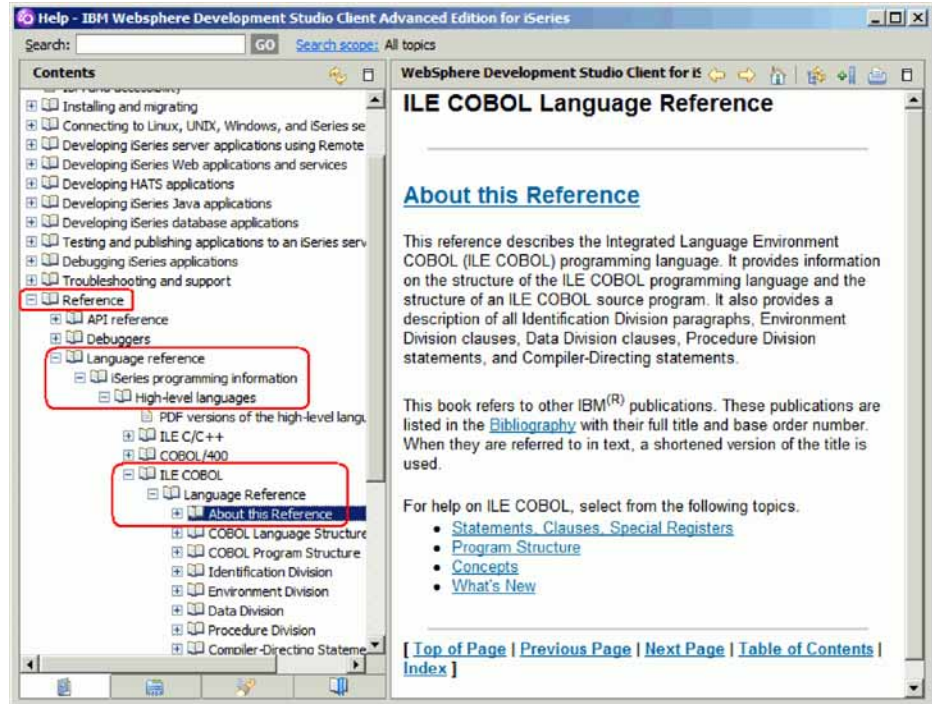




- d. Explore the Help window to see what else is available.
- e. Minimize the Help window.
- f. Select **Help > Help Contents** on the workbench menu and expand WebSphere Development Studio Client for iSeries book to see a list of all help that is available in the product.



- g. In the left pane of the Help window, click **Reference**.
- h. Expand **Language Reference**.
- i. Expand **iSeries programming information**.
- j. Expand **High-level languages**.
- k. Expand **ILE COBOL**.
- l. Expand **Language Reference**.



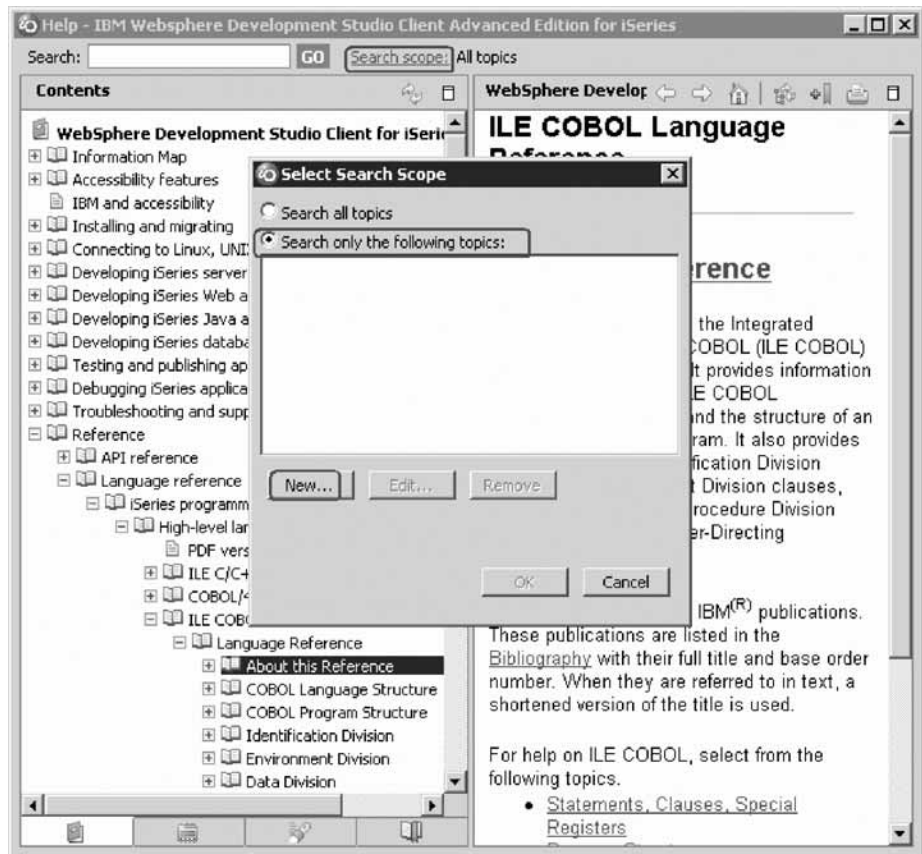
Having the latest version of the manuals at your fingertips will make it easier to find programming information. There is also the option to search the help by specifying a search string. By default, the complete help will be searched.

2. To limit the search scope:

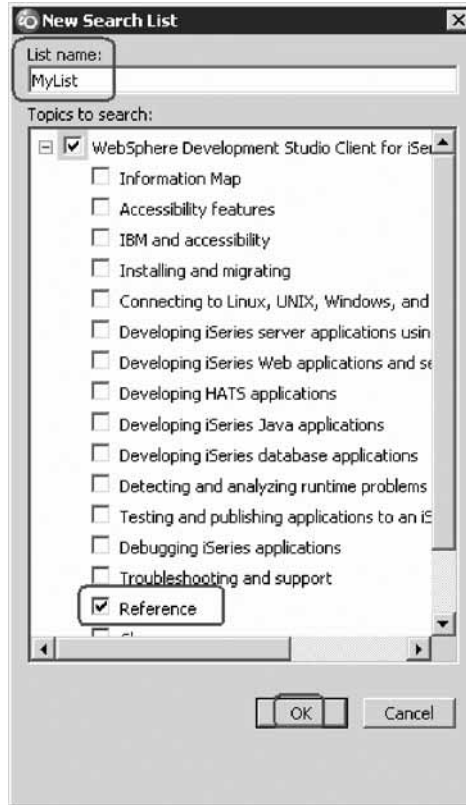
**Tip:** If you are using WDSC Lite, you can limit the search scope, however, you must choose another topic instead of the Reference topic shown.

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 **WebSphere Development Studio Client for iSeries**.
- f. Select the **Reference** check box.
- g. Click **OK** on the New Search List dialog.  
The Select Search Scope dialog reopens again with MyList selected in the topic list.

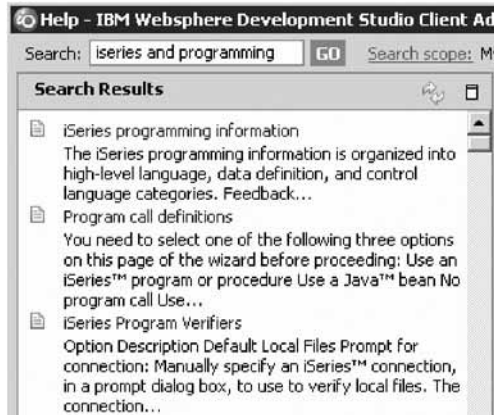


- h. Click **OK** on the Select Search Scope dialog.
- i. In the **Search** field, type iSeries and programming for example.



- j. Searching requires a help index and it takes approximately 10 minutes to create the index in your workspace. If you want to complete the search query, click **GO**.

The search results display.



You have accessed language sensitive help.

## Lesson checkpoint

You learned the following:

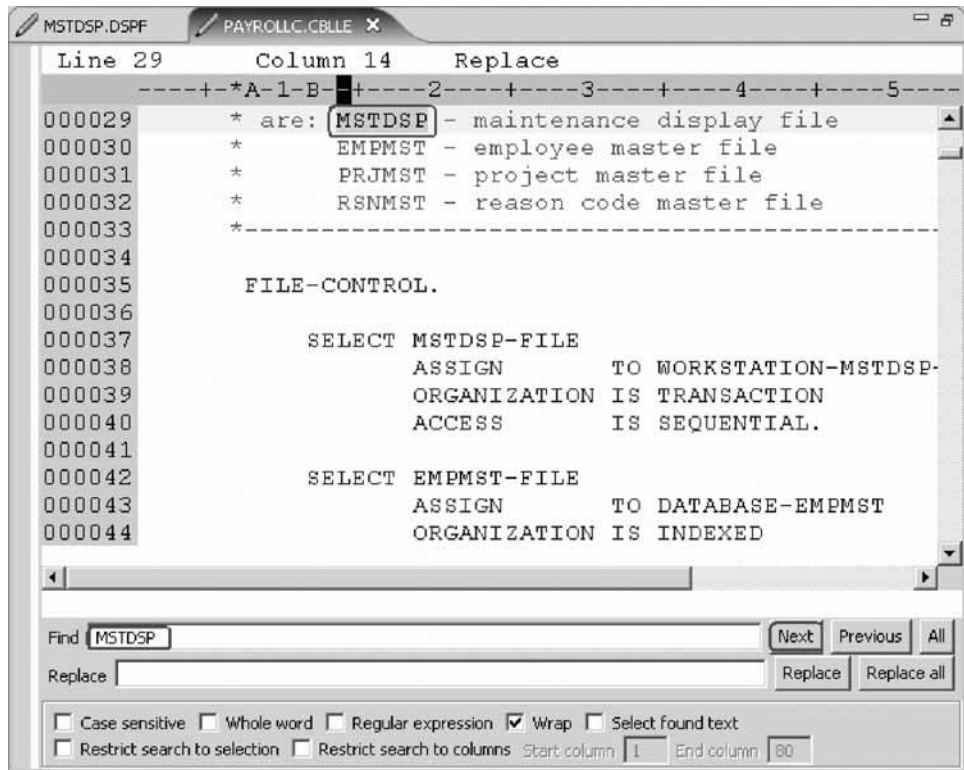
- About language sensitive help
- How to access language sensitive help

## 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. 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, etc. You want to find the first occurrence of MSTDSP.

3. In the **Find** field, enter MSTDSP to find the start of a file section.
4. Make sure the **Replace** field is blank.

You would use this field for text replacement.

5. Click **Next** to go to the next location of MSTDSP in the file. The Editor moves the active line to line 37, which contains the first MSTDSP phrase 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.

## Lesson checkpoint

You learned the following:

- About the Find and Replace window
- How to find text in source

## 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 EMPNO in the Editor window.
2. Select **Edit > Selected > Filter Selection** from the workbench menu.

```

MSTDSP.DSPF  PAYROLLC.CBLLE X
Line 142      Column 14      Replace
-----+*A-1-B-+-----2-----3-----4-----5--
000142      01 WS-EMPNO                      PIC 9(6) .
000302                      MOVE EMPNO                      TO WS-EMPNO.
000307                      MOVE WS-EMPNO TO EMPNO OF RCEMP
000320                      MOVE WS-EMPNO                      TO EMPNO OF EM
000353                      MOVE WS-EMPNO                      TO EMPNO OF RCEMP.

```

3. Move the cursor down a few lines to line 302.
4. Expand line 302. This expands the section up to the next instance of EMPNO.

```

MSTDSP.DSPF  PAYROLLC.CBLLE X
Line 302      Column 1      Replace
-----+*A-1-B-+-----2-----3-----4-----5--
000142      01 WS-EMPNO                      PIC 9(6) .
000302      MOVE EMPNO                      TO WS-EMPNO.
000303      MOVE ACODE OF EMPSEL-I TO WS-ACODE.
000304
000305      IF IND-OFF (IND-MAINT) AND IND-OFF (IND-EC
000306      SET IND-OFF (IND-NOT-FOUND) TO TRUE
000307      MOVE WS-EMPNO TO EMPNO OF RCEMP
000320      MOVE WS-EMPNO                      TO EMPNO OF EM
000353      MOVE WS-EMPNO                      TO EMPNO OF RCEMP.

```

Now you want to show the entire source again.

5. 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.

### Lesson checkpoint

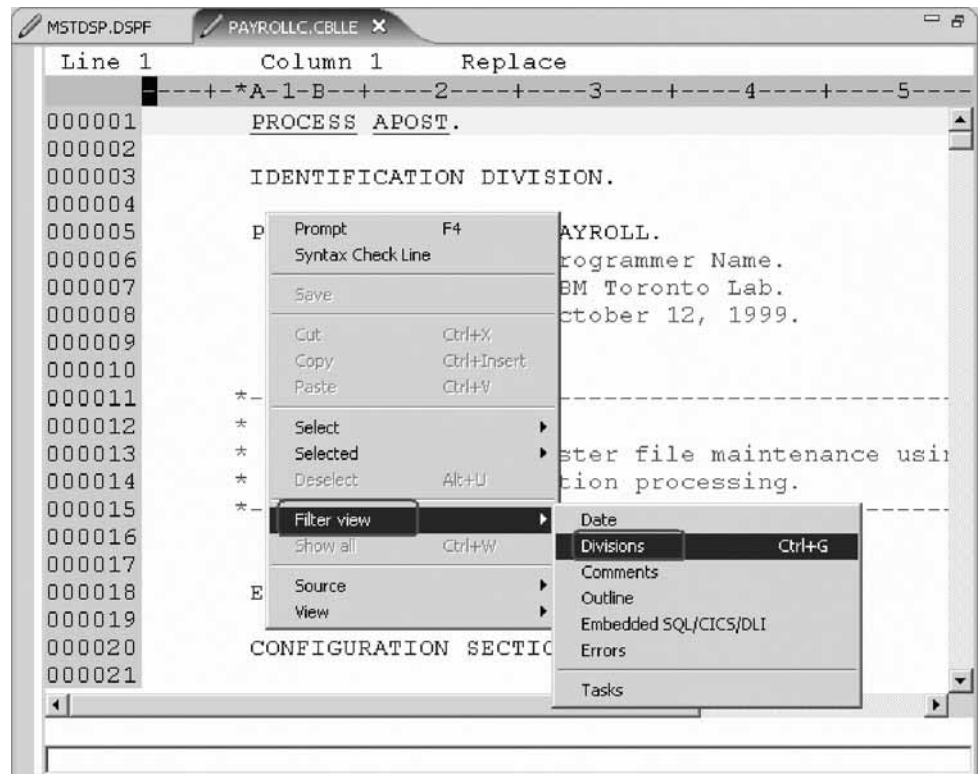
You learned the following:

- About filtering source
- How to filter source

### Filtering lines by type

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

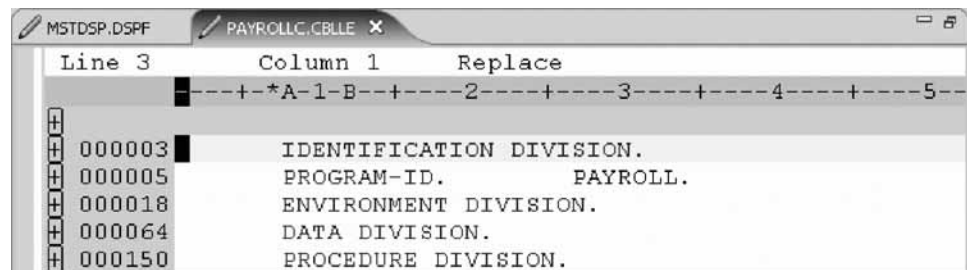




To filter lines by type:

1. Right-click in the Editor window with the PAYROLLC program.
2. Click **Filter view** > **Divisions** on the pop-up menu.

All source lines with divisions are displayed allowing you to move quickly and easily to the desired division in your file.



3. Move your cursor to the line with the DATA DIVISION (line 64).
4. Expand the division to show all lines in this division. Now you could work with the source inside this division.
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.

### Lesson checkpoint

You learned the following:

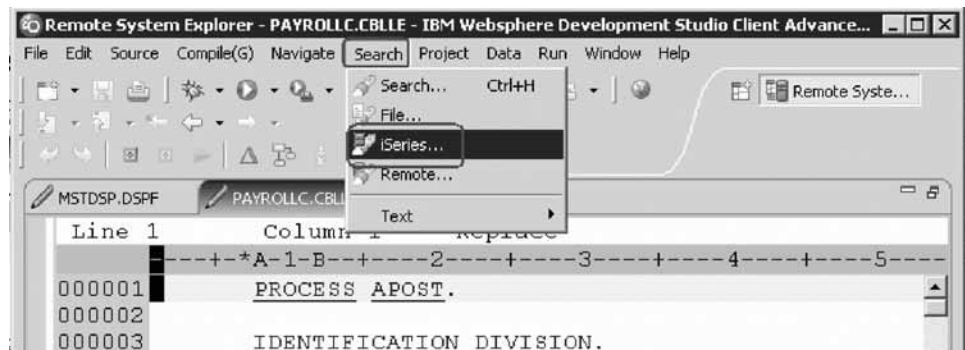
- About filter source by line type
- How to filter source by line type

## 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.

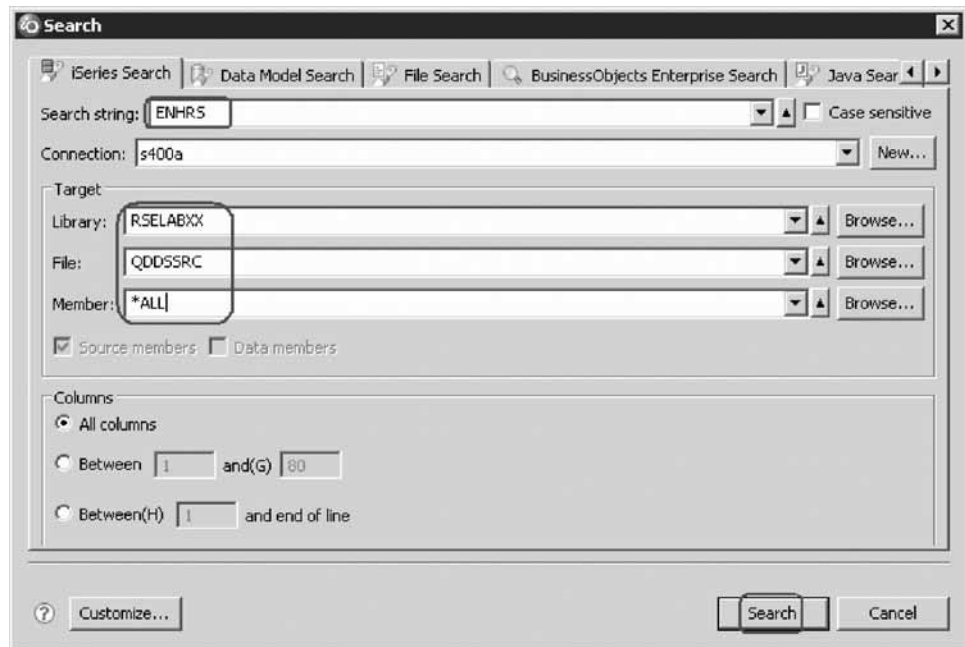
To search multiple files:

1. Click **Search > iSeries** from the workbench menu.



The Search window opens.

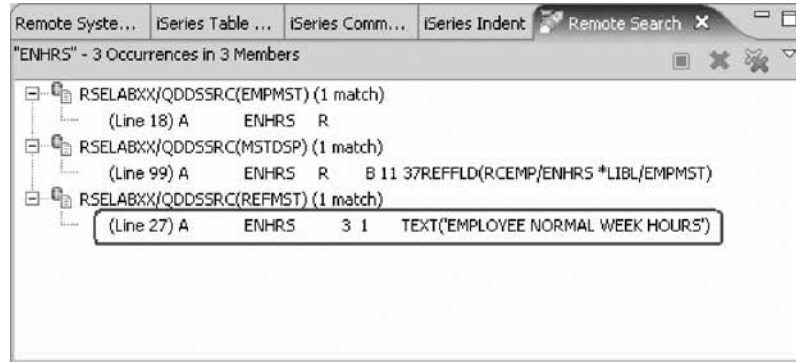
2. In the **Search string** field, type ENHRS.



The **Connection** field should contain your iSeries 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**.

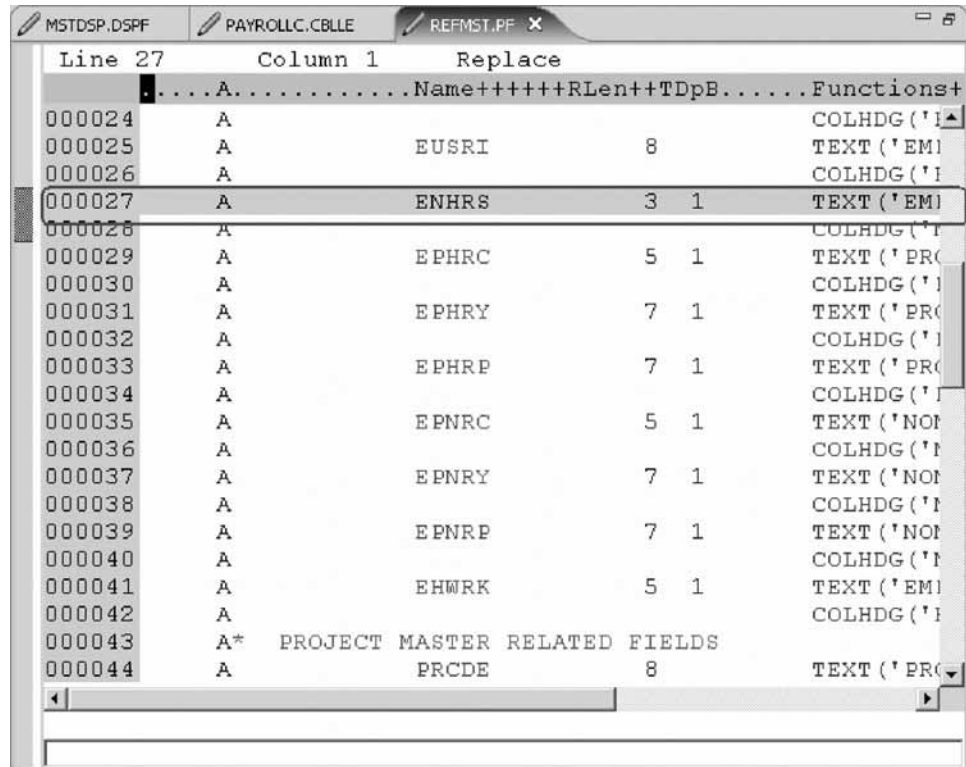
The Multi-File Search window lists all the lines in all the files that reference ENHRS.



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

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.

### Lesson checkpoint

You learned the following:

- About search
- How to search through members in a source physical file

## Comparing file differences 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. There are two ways to do a compare: use the Compare utility in the workbench or use the Compare utility in the CODE tool. The compare in the CODE tool is more intuitive but requires you to start the CODE Editor outside of the workbench.

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.

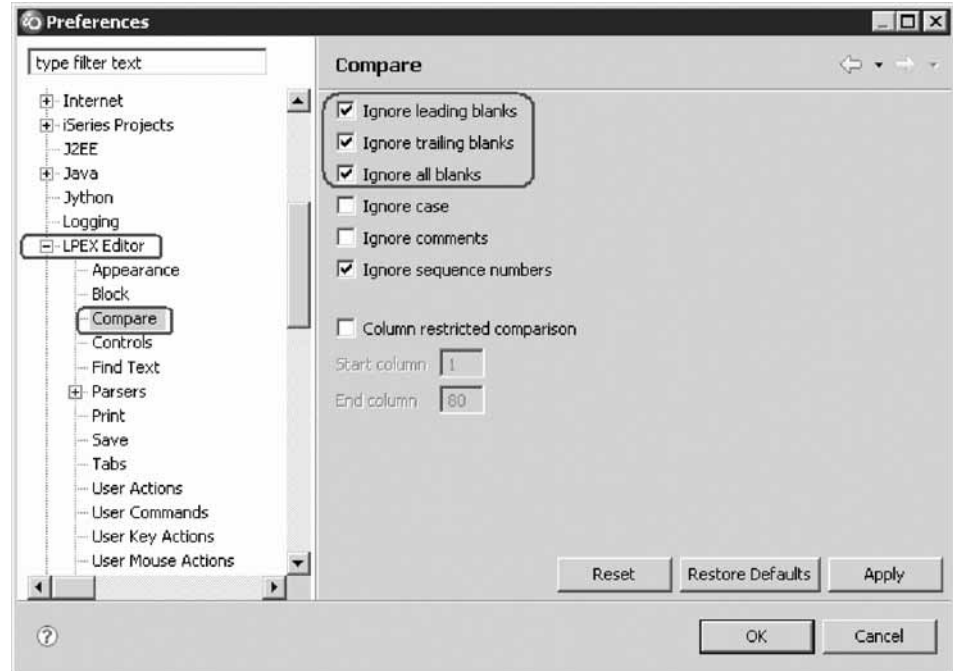
After a comparison is carried out, the Compare Editor opens in the editor area. In the compare Editor, you can browse through all the differences and copy highlighted differences between the compared resources. You can save changes to resources that are made in the comparison editor.

Using the compare utility in CODE you can also view the differences between two files by comparing them. You enter a name of a file to compare against the file in the CODE Editor view. You can type the name of a file, or you can select one from the list of files already open in the editor. If you type the name of the file that is not already open in the editor, it is loaded into the editor. If no file is specified, the current file is compared against a new, untitled file. The current file appears on the left side of the Compare view, and the specified file on the right. You use the Compare menu to view the next and previous mismatch and to select options such as ignore case, font, protect view and show mismatches only.

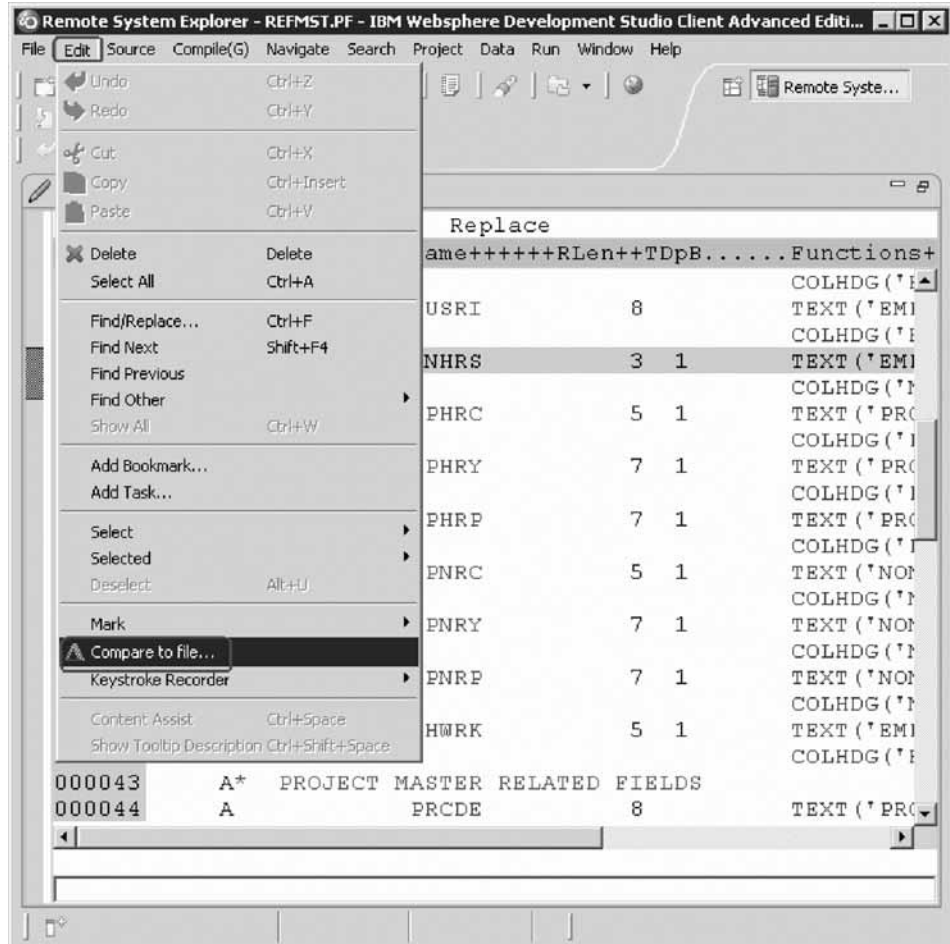
**Tip:** Make sure all lines show in the source before starting the Compare tool.

To compare files in the workbench:

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 blanks** check boxes are selected.
4. Click **OK** in the Preferences window.
5. Back in the Editor window of the PAYROLLC member double-click the **PAYROLLC** tab.
6. Click **Edit > Compare to file** on the workbench menu.



The Compare window opens.

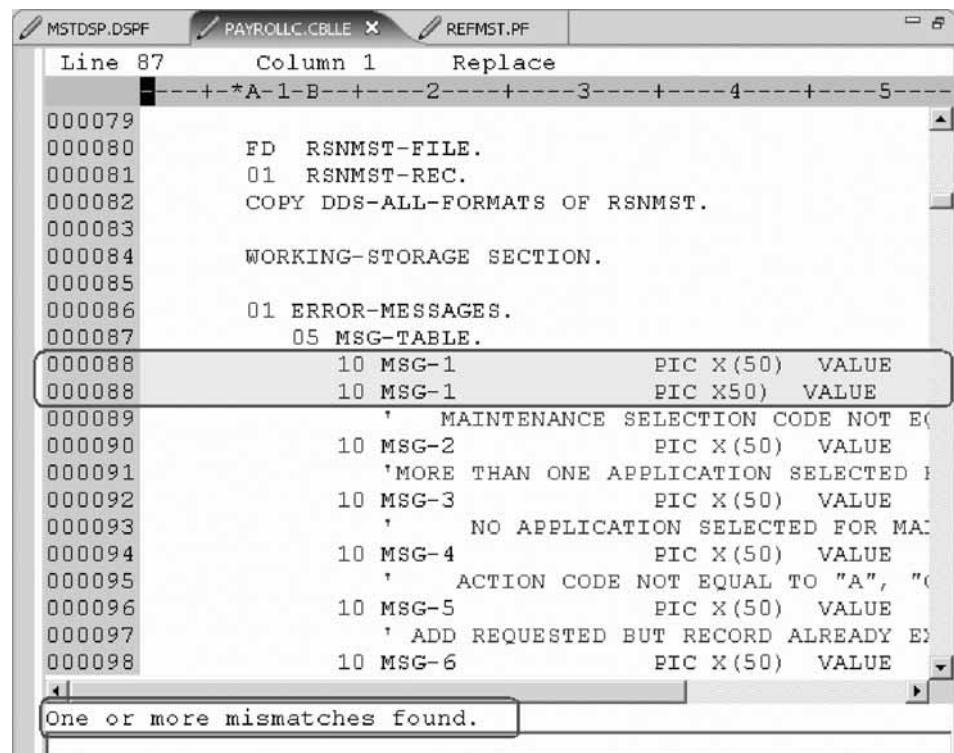
7. Expand your connection.
8. Expand \*LIBL.
9. Expand RSELABxx.
10. Expand QCBLESRC.
11. Select member PAYROLLC2
12. Click **OK**



The editor now will show the differences of these two members PAYROLLC and PAYROLLC2 .

You can move from mismatch to mismatch by using the Compare menu under the Edit menu.

Mismatches in PAYROLLC and PAYROLLC2 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.

## Lesson checkpoint

You learned the following:

- About compare
- How to compare files

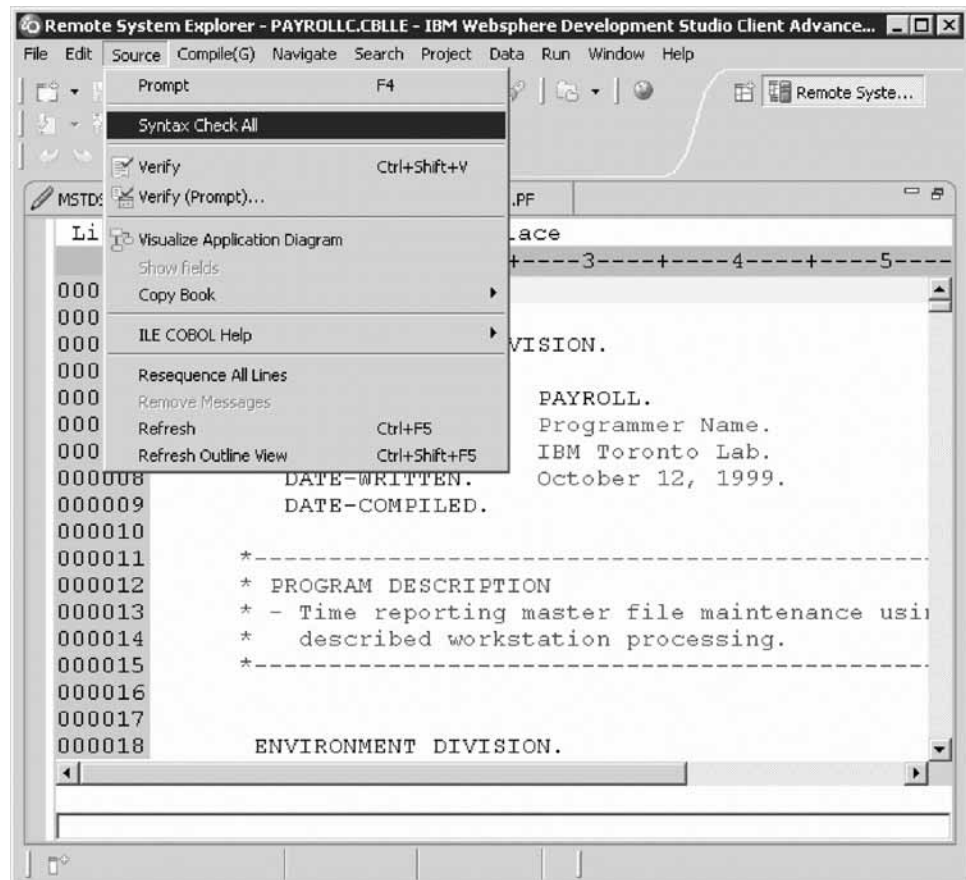
## Checking syntax

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. Click the PAYROLLC Editor window, click **Source** and then click **Syntax Check All** on the workbench menu.



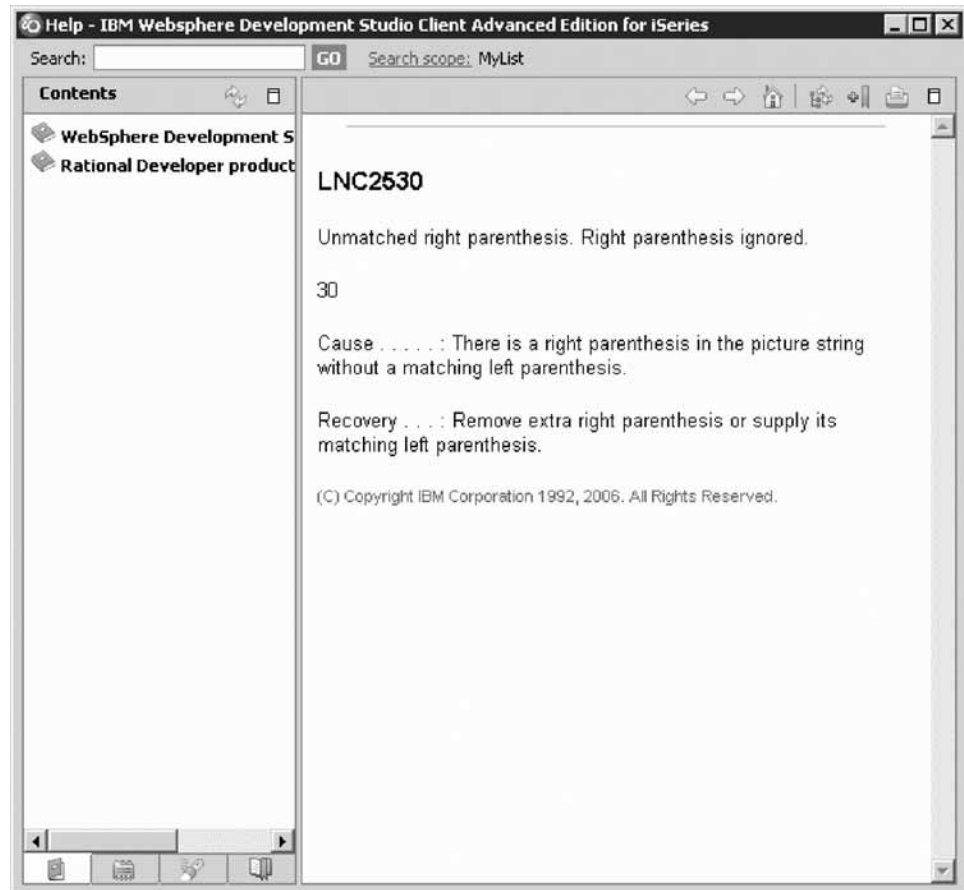
Error messages appear to draw attention to the errors.



```

Line 88      Column 1      Replace
-----+---+*A-1-B--+----2-----3-----4-----5-----
000080      FD RSNMST-FILE.
000081      01 RSNMST-REC.
000082      COPY DDS-ALL-FORMATS OF RSNMST.
000083
000084      WORKING-STORAGE SECTION.
000085
000086      01 ERROR-MESSAGES.
000087      05 MSG-TABLE.
000088      10 MSG-1      PIC X(50) VALUE
LNC2530E Unmatched right parenthesis. Right parenthesis
LNC1248E PICTURE string 'X50' does not describe value.
000089      *      MAINTENANCE SELECTION CODE NOT EQUAL TO
000090      10 MSG-2      PIC X(50) VALUE
000091      *MORE THAN ONE APPLICATION SELECTED FOR MAINTENANCE
000092      10 MSG-3      PIC X(50) VALUE
000093      *      NO APPLICATION SELECTED FOR MAINTENANCE
000094      10 MSG-4      PIC X(50) VALUE
000095      *      ACTION CODE NOT EQUAL TO "A", "C", "S", "D", "E"
  
```

2. Move the cursor onto the pink error message.
3. Press F1.

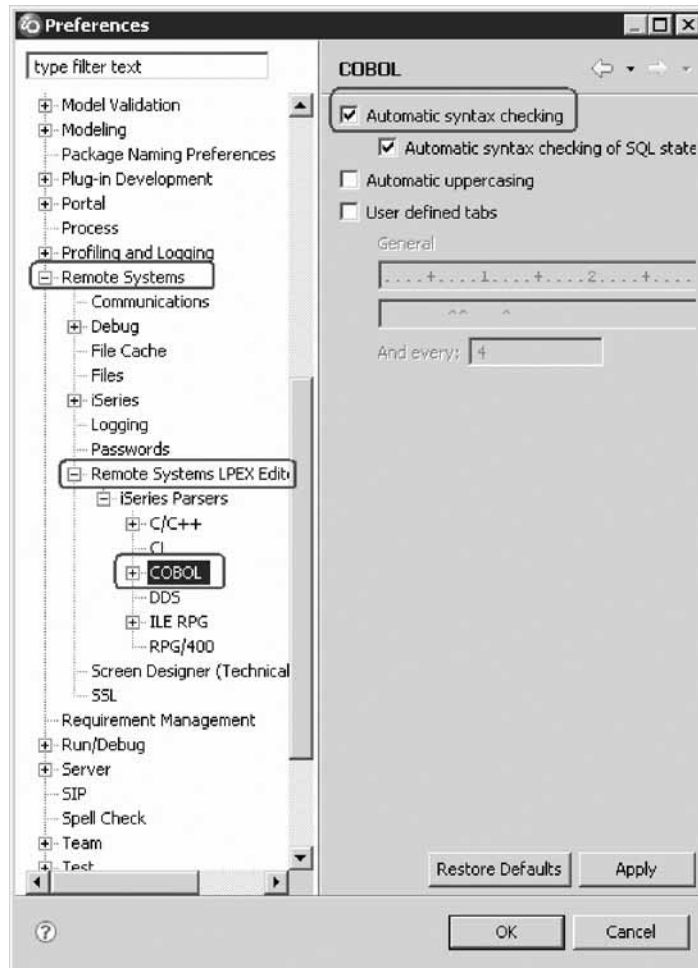


This opens a window with additional help for the error.

4. Minimize the Help window.
5. Add the required left parenthesis to correct the error.
6. 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, iSeries Parsers**. Now, select the language you want to change the settings for in the left pane of the Preferences window, select or deselect the **Automatic syntax checking** check box and then click **OK**.



In this lesson you'll use the syntax checking feature to check your source.

### Lesson checkpoint

You learned the following:

- About syntax checking
- How to syntax check your source
- How to set syntax checking preferences

## Understanding your program code with the Application Diagram

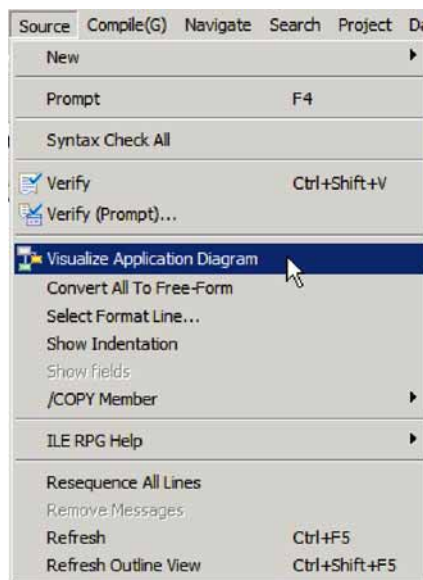
**Important:** This lesson only applies to WebSphere Development Studio Client for iSeries Advanced Edition.

In this lesson, you learn about the Application Diagram view and how to switch to it, how to view the program topology, and using the viewer to locate a subroutine.

The Application Diagram provides a graphical view of the different resources in an i5/OS native application and their relationships to each other. There are two different diagrams that you can look at in the Application Diagram view: a Source Call Diagram and a Program Structure Diagram. The Source Call Diagram takes ILE COBOL 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.

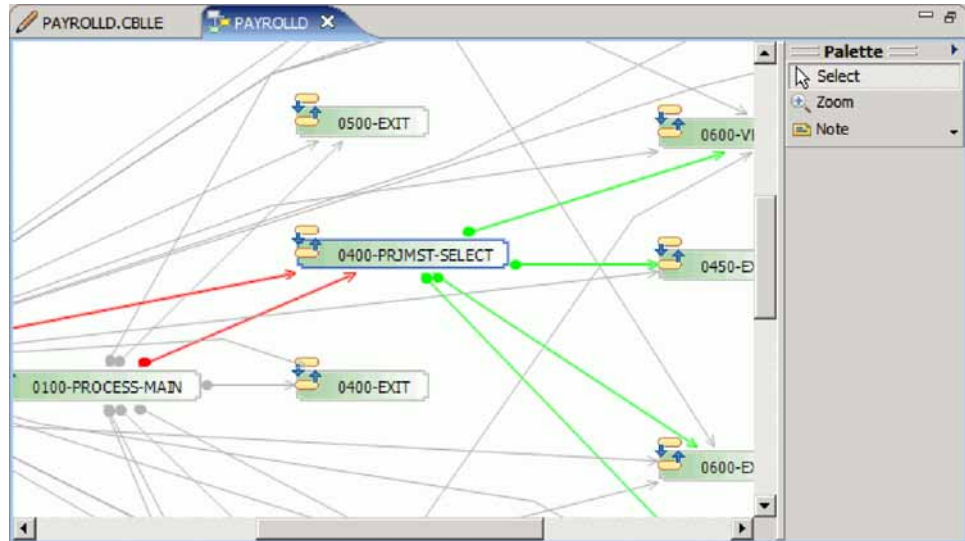
To learn more about how to use the Application Diagram, see [Launching the Application Diagram](#) and [Using the Application Diagram](#).

1. Open the **PAYROLLD.cbll** source member in the **QCBLLSRC** source file.
2. Click **Source** → **Visualize Application Diagram**



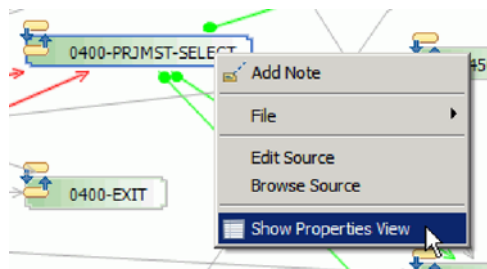
The Application Diagram opens.

3. Select the subroutine **0400-PRJMST-SELECT**.



Notice that the box is now highlighted and all relationship arrows are highlighted. Arrows pointing away from the subroutine point to subroutines that 0400-PRJMST-SELECT calls and arrows pointing to the subroutine 0400-PRJMST-SELECT indicated calls to 0400-PRJMST-SELECT.

4. Right-click the subroutine 0400-PRJMST-SELECT 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 0400-PRJMST-SELECT calls and the lines where it is called.

Called by	Name	Location	Start line	End line	Type	Language
	0450-PRJMST-MAINT	PAYROLLD.CBLLE	439	481	PROCEDURE	COBOL
	0450-EXIT	PAYROLLD.CBLLE	482	485	PROCEDURE	COBOL
	0600-VERIFY-ACTION-C...	PAYROLLD.CBLLE	586	608	PROCEDURE	COBOL
	0600-EXIT	PAYROLLD.CBLLE	609	612	PROCEDURE	COBOL

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 0400-PRJMST-SELECT subroutine.

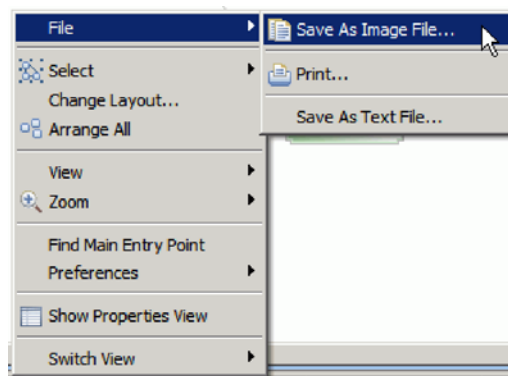
6. In the **Calls** tab, double-click the 0450-EXIT subroutine.  
This takes you into the source to the call to subroutine 0450-EXIT.

```

PAYROLLD.CBLLE X PAYROLLD
Line 482 Column 1 Replace
-----*A-1-B-----2-----3-----4-----+
000474                                END-REWRITE
000475                                WHEN 'C'
000476                                REWRITE PRJMST-REC
000477                                INVALID KEY SET IND-
000478                                END-REWRITE
000479                                END-EVALUATE
000480                                END-IF.
000481
000482 0450-EXIT.
000483                                EXIT.
000484
000485
000486                                0500-RSNMST-SELECT.
000487
000488 *-----*
000489 * Reason code maintenance routine. *

```

7. Return to the Application Diagram view by selecting its tab. You can also save the application diagram as a text or image file.
8. Right-click inside the Application Diagram view, and select **File** → **Save as Image File** or **File** → **Save as Text File**



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

## Lesson checkpoint

You learned the following:

- The basics of the Application Diagram view
- How to switch to the Application Diagram view
- How to navigate through the application topology
- How to save the application diagram as an image or text file

### Related information

Launching the Application Diagram

Using the Application Diagram

## Module summary

In this module, you learned how to use 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
- 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

## Assessment

- What does column sensitive editing do?
- Where do you define LPEX Editor settings?
- How do you configure the LPEX Editor to adopt the keyboard and command parameters of many popular editors?
- What operation must you use to undo a set of changes made to a file?
- How do you cancel the effects of changes of an undo operation?
- What do you use instead of entering or changing code directly in the Editor window?
- What window do you use to search for an item in the current source?
- How do you filter source?
- How do you search multiple source files?
- How do you compare files?
- How do you syntax check source files?

---

## 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 iSeries Error List appears. The iSeries 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 iSeries 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 iSeries Error list to locate each error in the source
- Save your source

- Re-verify source
- Change compile preferences
- Invoke the compile command
- Change the current library using the Command field in the iSeries Table view
- Start an interactive connection
- Invoke the payroll program

### Time required

This module should take approximately 20 minutes to complete.

## 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 iSeries, 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 the iSeries. Think of the host cycles you'll save. It is especially handy when you are writing code but you are disconnected from an iSeries. You can do this because Remote System Explorer ported the parsing and checking code from the iSeries system compilers to the workstation. The iSeries 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. 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.)

You will need to open the COBOL source file named PAYROLLC2 to complete this verify step.

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

ID	Message	S..	L..	Location	Con...
LNC1904	Program PAYROLL syntax ...	4..	1	RSELABXX...	s400a
LNC1463	'EMPNO' is not unique in thi...	3..	3..	RSELABXX...	s400a
LNC1326	'PRCD OF PRJSEL-I' not de...	3..	4..	RSELABXX...	s400a
LNC1329	Subscript value '14' exce...	3..	6..	RSELABXX...	s400a
LNC0407	AT END phrase missing fro...	2..	1..	RSELABXX...	s400a
LNC0407	AT END phrase missing fro...	2..	2..	RSELABXX...	s400a
LNC0407	AT END phrase missing fro...	2..	3..	RSELABXX...	s400a
LNC0407	AT END phrase missing fro...	2..	3..	RSELABXX...	s400a
LNC0407	AT END phrase missing fro...	2..	4..	RSELABXX...	s400a
LNC0407	AT END phrase missing fro...	2..	4..	RSELABXX...	s400a
LNC0407	AT END phrase missing fro...	2..	5..	RSELABXX...	s400a

The error list shows you:


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

- The connection name
2. To fix an error in your source go to the error list:
    - a. Double-click the error LNC1463. You are automatically brought back into the Editor window to the line where the error occurred. The error is caused by EMPNO not being uniquely defined. Go to line 302 where EMPNO is defined. The variable EMPNO should be EMPNO OF EMPSEL-I.
    - b. Make the change. The next error is LNC1326.
    - c. Double-click LNC1326. Fix it in the editor.
    - d. PRCD should really be PRCDE. Make the appropriate change on line 402.
    - e. Double-click LNC1329. This error is because the array was declared with 4 elements. Go to line 606.
    - f. Change the index of the array from 14 to 4.

The next errors are LNC0407. You can ignore these errors as they are severity 20.

All severity 30 errors and above are now fixed. You can filter out different severities by using the filter menu.



- g. Click the arrow in the iSeries Error List title bar.
  - h. Click **Show Severity** on the pop-up menu.
  - i. Clear the severities you don't want to see in the list (Warning for example).
- Now before you loose any of your changes, it's a good idea to save them. Make sure the member is selected. You then verify the source again to make sure that all the errors are fixed.
3. 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 iSeries.

    - d. Verify your source again.



ID	Message	Severity	Line	Location	Con...
LNC0407	AT END phrase missing fro...	20	184	RSELABXX...	s400a
LNC0407	AT END phrase missing fro...	20	299	RSELABXX...	s400a
LNC0407	AT END phrase missing fro...	20	348	RSELABXX...	s400a
LNC0407	AT END phrase missing fro...	20	399	RSELABXX...	s400a
LNC0407	AT END phrase missing fro...	20	448	RSELABXX...	s400a
LNC0407	AT END phrase missing fro...	20	499	RSELABXX...	s400a
LNC0407	AT END phrase missing fro...	20	548	RSELABXX...	s400a

Everything should be ok. You should see only severity 20 messages. You are ready to compile the program.

You have verified your source and fixed any errors.

### Lesson checkpoint

You learned the following:

- About program verifier
- How to verify programs

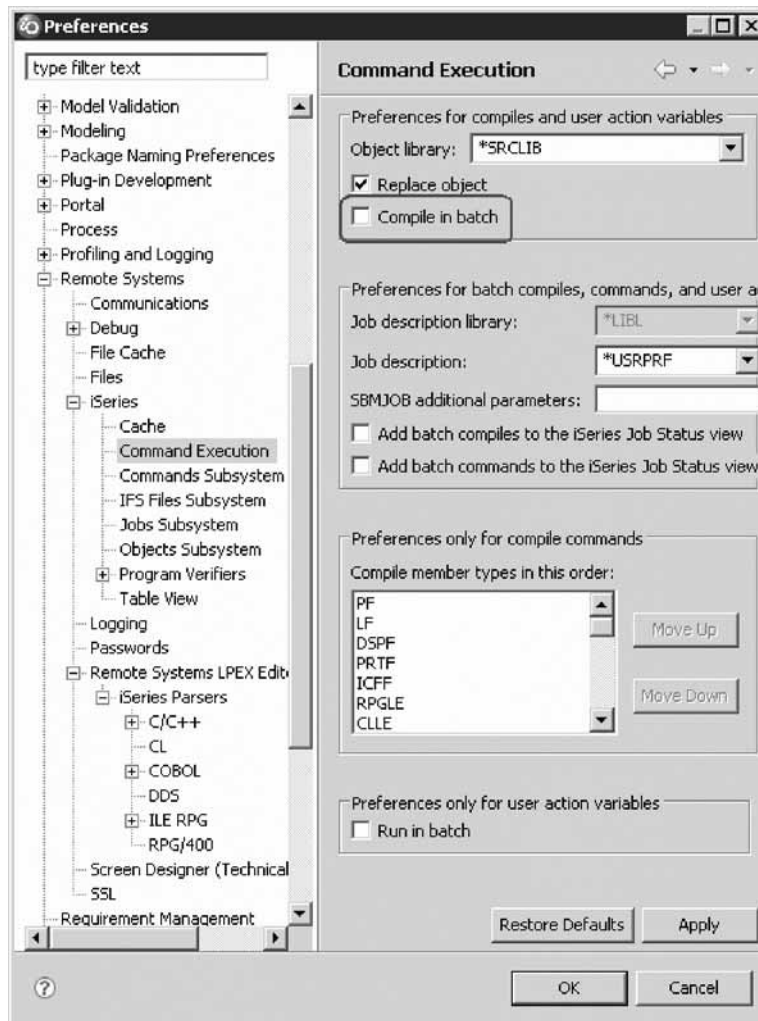
## Compiling source remotely

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 iSeries 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 iSeries 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 iSeries Error List view, which behaves just as it did when you did a program 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.

1. To change the preferences to run the compile interactive:
  - a. Click **Window > Preferences** from the workbench menu.



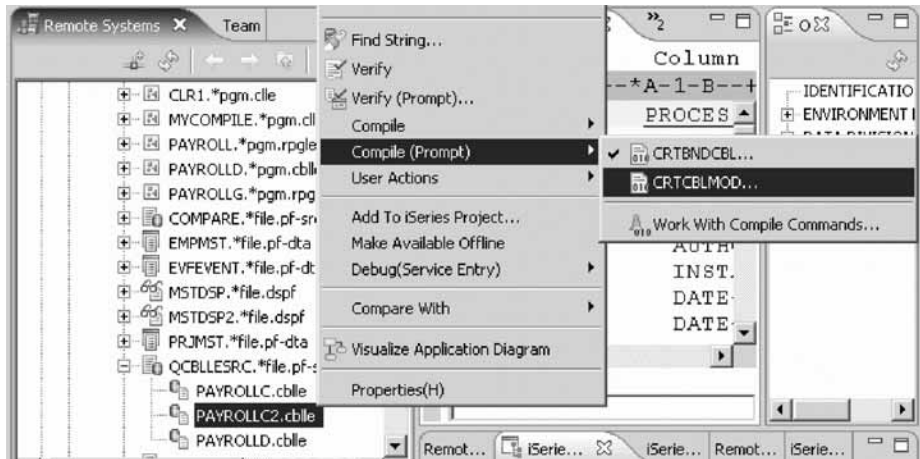
- b. In the left pane of the Preferences window, expand **Remote Systems**.
- c. Expand **iSeries** under Remote Systems.
- d. Click **Command Execution** under iSeries.
- e. In the right pane of the Preferences window, clear the **Compile in batch** check box.
- f. Click **OK** to return to the Remote System Explorer perspective.

## 2. Invoking the compile command

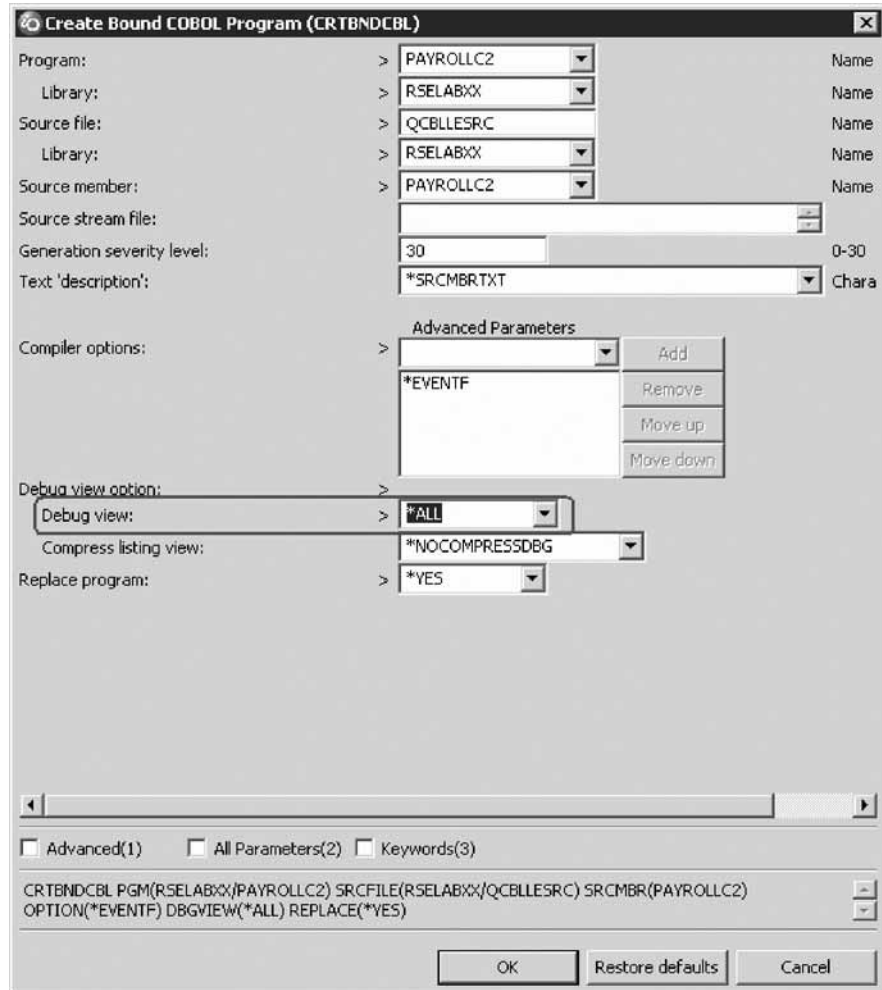
You will now use the prompt for the CRTBNDCBL command to specify your compile parameters. All entry fields pertaining to names are already filled in with the correct information.

To compile source:

- a. Right-click the PAYROLLC2 member in QCBLLESRC.



- b. Click **Compile (Prompt) > CRTBNDCBL** on the pop-up menu. The Create Bound COBOL Program (CRTBNDCBL) dialog opens.
- c. In the **Debug view list**, select the **\*ALL** parameter.

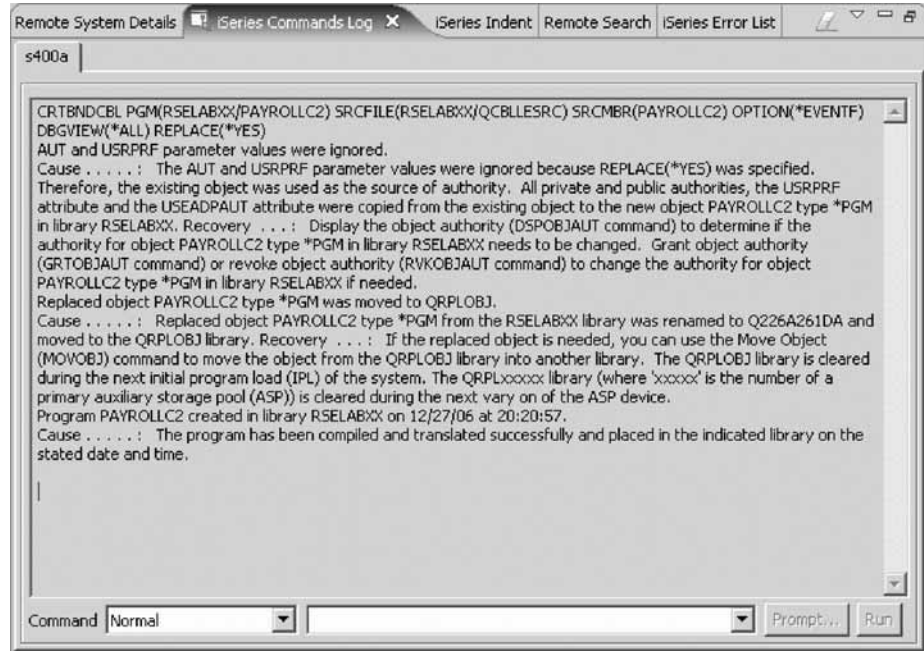


If you want to see the other parameters available, click **Advanced**.

- d. 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.

If you are not sure that the compile was successful, you can check the iSeries Commands log.



- e. Click the **iSeries Commands log** tab from the view 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, checked for a successful compile.

## Lesson checkpoint

You learned the following:

- About compile preferences
- How to invoke the compile command and check for a successful compile

## Submitting iSeries commands in the iSeries Table view

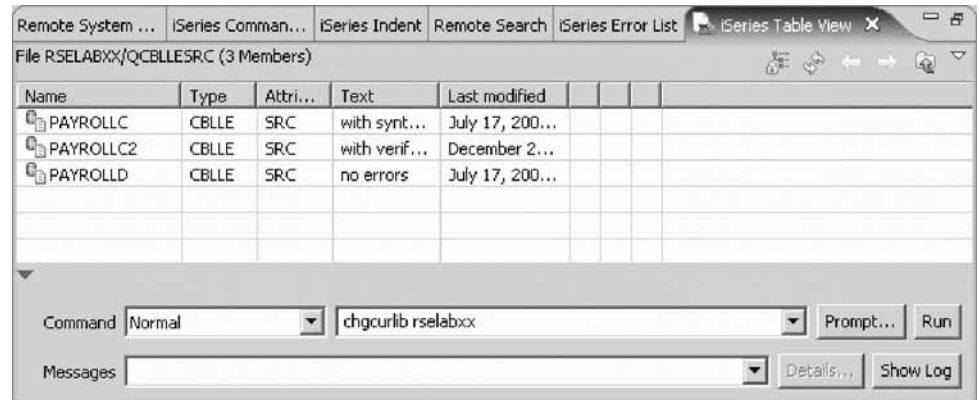
You can use the iSeries Table view inside the Remote System Explorer to submit commands to the iSeries. You can run commands from the **Commands** field beneath the iSeries Table view, and view messages in the **Messages** field. After you populate the table, you can enter a command and click either **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 iSeries 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 iSeries 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 iSeries 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 **iSeries Table View** tab from the views at the bottom of the workbench.
2. In the **Command** field type, CHGCURLIB RSELABxxfor example.

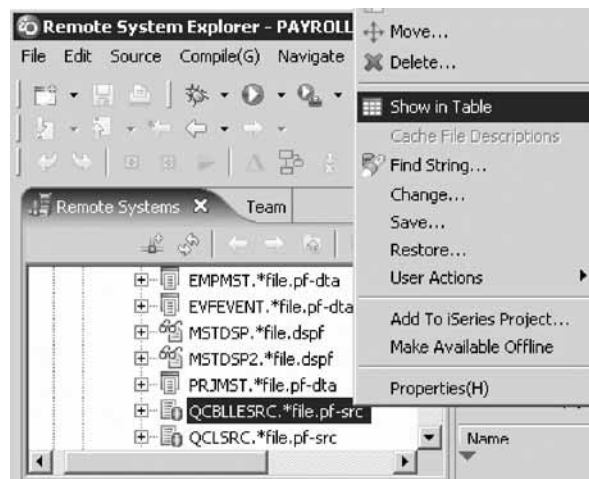
**Tip:** Use a library that is on your iSeries system.

3. Click **Run**.

If you haven't used the iSeries Table view to show iSeries objects in this view you will see an error message because the iSeries Table view is not linked to an active connection.

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

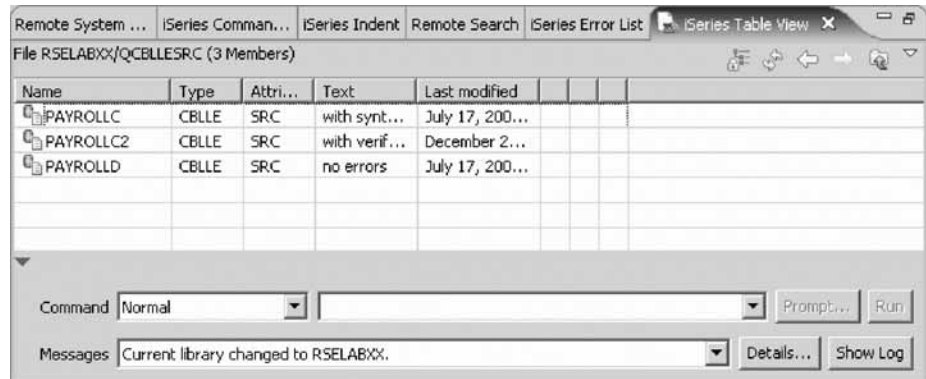
- a. In the Remote Systems view, right-click QCBLLESRC.



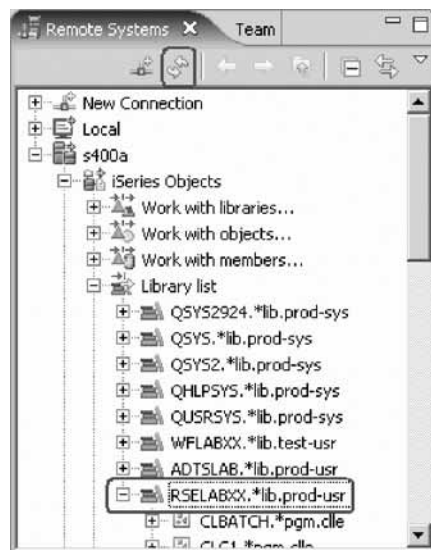
- b. Click **Show in Table** on the pop-up menu. The iSeries Table view is now populated with the member in the selected source file.

c. Run the CHGCURLIB command again.

The command will run on the iSeries and after completion you will see the completion message on the bottom of the iSeries Table view.



4. Back in the workbench in the Remote Systems view, right-click **Library list** and click **Refresh**.



5. You will see a small green asterisk beside the RSELABxx library to indicate it as your current library.

6. You can also connect to other than iSeries 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 iSeries Table view.

### Lesson checkpoint

You learned the following:

- About submitting commands in the iSeries Table view
- How to submit a CL command in the iSeries Table view

## Running commands and programs

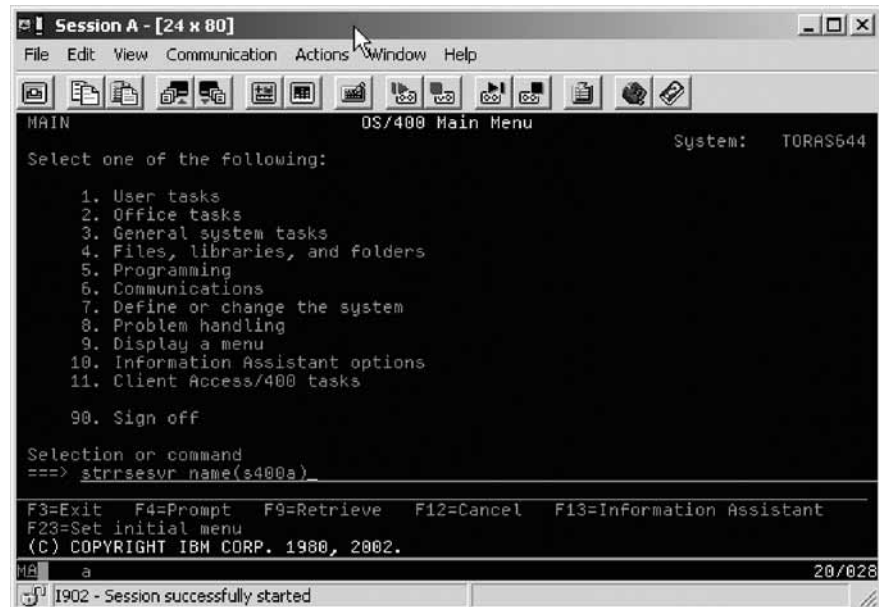
As you know, you can run programs and commands from the Remote Systems view or the iSeries Table view in three ways:

1. In the Remote System Explorer communications server job. (Your current method)
2. In a batch job.
3. In an interactive job (to test 5250 applications).
4. In a server job

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, 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 debug session creates a new server job and this way keeps the RSE communications server job free for other tasks.

1. To start an interactive connection:
  - a. Start a 5250-emulation session.
  - b. Sign-on to the iSeries with your User ID and password.

**Tip:** Instead of the **Enter** key, you may have to use the **Ctrl** key in your 5250-emulation session.



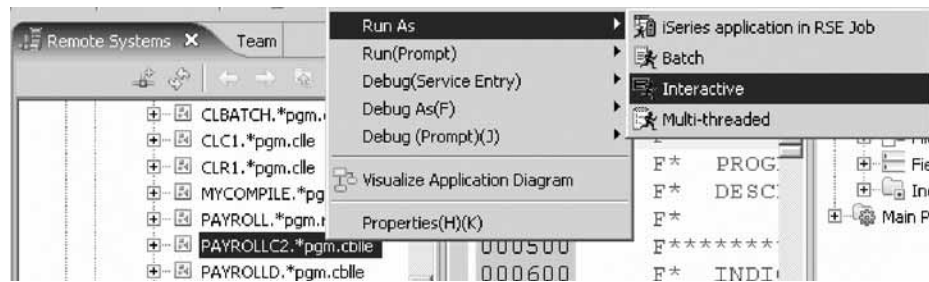
```
Session A - [24 x 80]
File Edit View Communication Actions Window Help
MAIN DS/400 Main Menu System: TORAS644
Select one of the following:
1. User tasks
2. Office tasks
3. General system tasks
4. Files, libraries, and folders
5. Programming
6. Communications
7. Define or change the system
8. Problem handling
9. Display a menu
10. Information Assistant options
11. Client Access/400 tasks
90. Sign off
Selection or command
==> strrsesvr name(s400a)
F3=Exit F4=Prompt F9=Retrieve F12=Cancel F13=Information Assistant
F23=Set initial menu
(C) COPYRIGHT IBM CORP. 1980, 2002.
MA a 20/028
I902 - Session successfully started
```

- 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.

Return to the workbench.

2. To run the program:



- a. In the Remote Systems view, locate the PAYROLLC2 program that you created.
- b. Right-click the PAYROLLC2 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.
- n. To get control of the interactive job, right-click **iSeries 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 iSeries Table view.



## Lesson checkpoint

You learned the following:

- About run commands
- How to run programs

## Module summary

In this module you learned how to verify and compile RPG in the Remote Systems LPEX Editor. When errors were found by either the verify or the compile step, the iSeries Error List appears. The iSeries 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 iSeries Error List and the RPG program that you created.

### Lessons learned

- Check for semantic errors on your workstation
- Start the Program Verifier tool
- Use the iSeries 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 iSeries Table view
- Start an interactive connection
- Invoke the payroll program

### Assessment

- What tool checks for semantic (compile) errors on your workstation?
- What view identifies each error from a compile?
- How do you sort the entries in the iSeries Error List view?
- What can you do in the iSeries Table view?
- What are the different ways to run a program?
- How do you verify source?
- How do you compile source?

---

## Debugging a program

This module teaches you how to debug a CL and ILE COBOL 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.

### Learning objectives

- Introduce the debug features
- Start a debug session using a service entry point

- Add a breakpoint
- Add a conditional breakpoint
- Edit a breakpoint
- Monitor a variable through the Monitors view
- Step into your payroll program
- Show a Listing view
- List the call stack entries in the Debug view
- View all breakpoints
- Remove a breakpoint
- Monitor memory
- Set a Watch breakpoint
- Close the debugger
- Invoke the debugger from the Launch Configurations window

### **Time required**

This module should take approximately 60 minutes to complete.

## **Introducing the Integrated iSeries Debugger**

The Integrated iSeries Debugger is a source-level debugger that enables you to debug and test an application that is running on an iSeries 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 the iSeries system.
- 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 execution.
- 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 iSeries system than 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<sup>®</sup> and ILE RPG, COBOL and ILE COBOL, C, C++ and CL.

Now you know the basic features of the debugger.

## Lesson checkpoint

You learned the following:

- About the Debugger

## Starting a Debug session using a service entry point

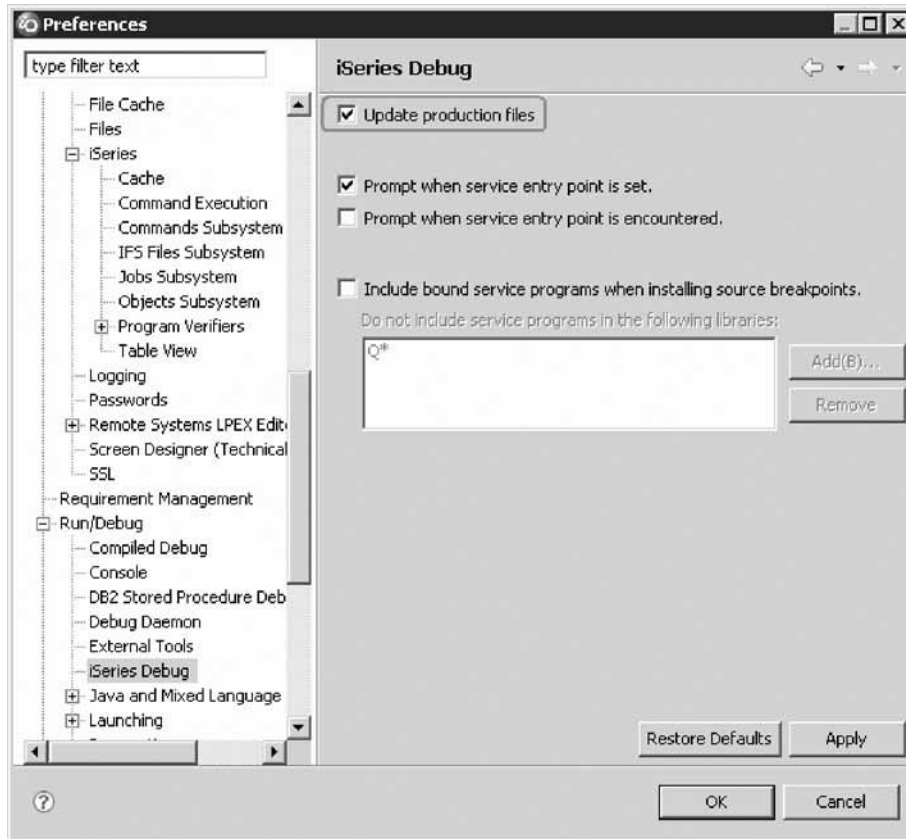
You will be working with the COBOL program PAYROLLD.

**Note:** PAYROLLD is the same COBOL program as PAYROLLC2 but without compile errors. You are using it instead of PAYROLLC2 in this lesson, to accommodate anyone who decided to skip right to this exercise without completing the lessons in “Verifying and compiling source” on page 50.

To make the lesson interesting you will use CL program CLC1 to call PAYROLLD and you will pass one parameter to CLC1.

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.

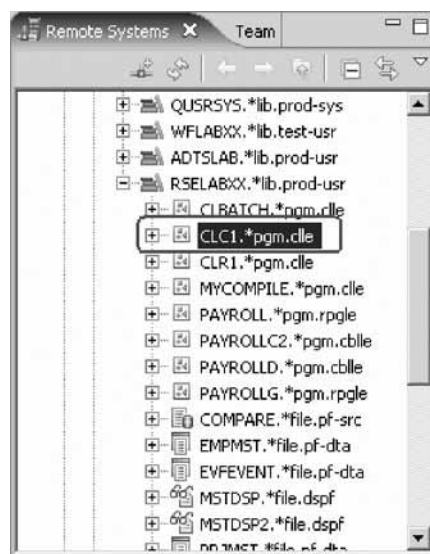
**Tip:** To use a service entry point to start a debug session for your application and to allow updating of files in production libraries while debugging, the **Update Production Files** must be checked in the preferences of iSeries Debug (**Window > Preferences** then expand **Run/Debug** and select **iSeries Debug**).



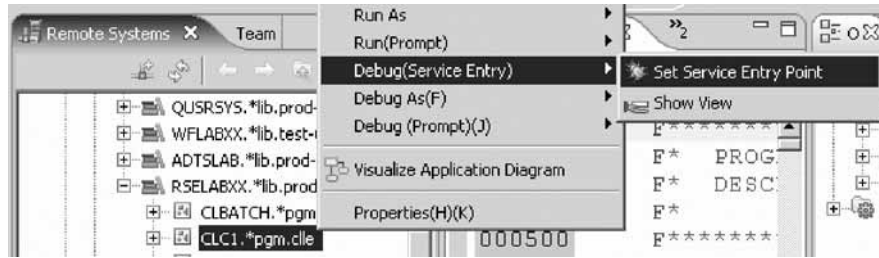
Since you are using test libraries for the exercises, you don't have to check this iSeries Debug preference.

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

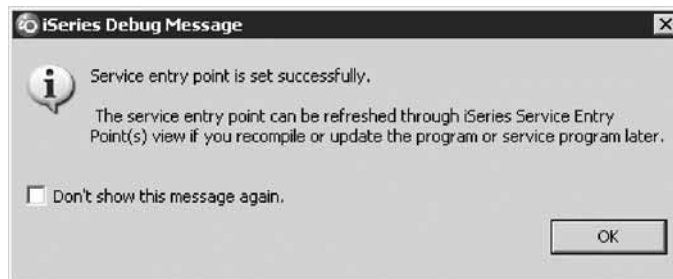


2. Expand library RSELABxx, if it isn't expanded already.
3. Right-click program CLC1 in library RSELABxx.

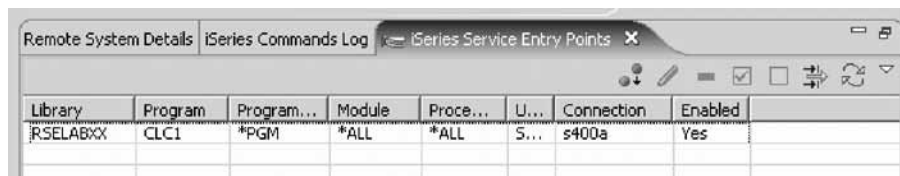


4. Click **Debug (Service Entry) > Set Service Entry Point** on the pop-up menu to set a service entry point.

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



The Service Entry Points view is automatically added to the notebook at the bottom 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 **iSeries Objects** and click **Release Interactive Job** on the pop-up menu.

6. Click anywhere in the workbench to give it focus.
7. On the command line of the 5250 screen, add the library RSELABxx to the library list and invoke the CLC1 program: `ADDLIBLE RSELABxx` and then `CALL PGM(RSELABxx/CLC1) PARM('XX')`

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.

You have started a debug session with a service entry point.

## Lesson checkpoint

You learned the following:

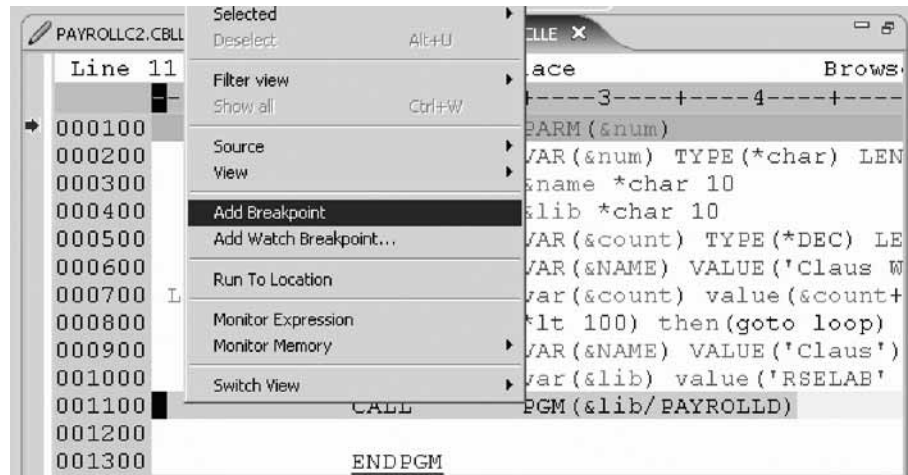
- About service entry points
- How to start a debug session with a service entry point

## 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.

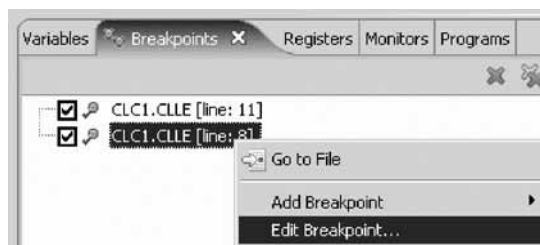


3. Click **Add breakpoint** on the pop-up menu.

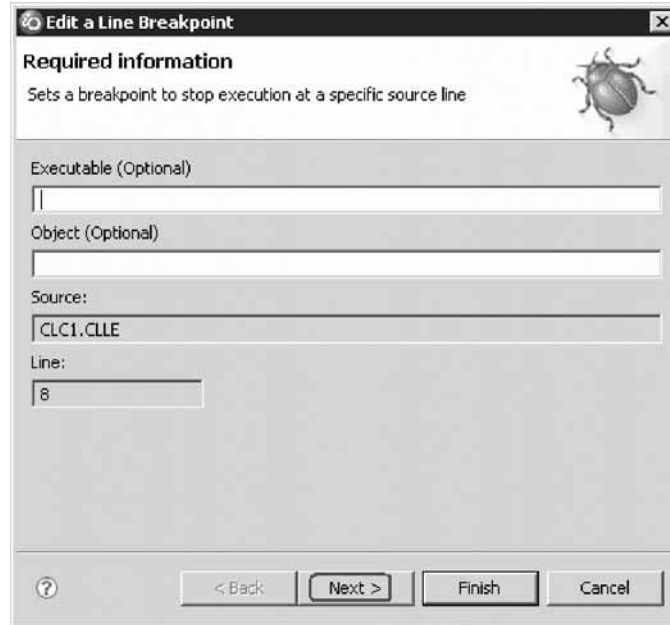
A dot with a checkmark in the prefix area indicates that a breakpoint has been set for that line. The prefix area is the small grey margin to the left of the source lines.

Now you add a conditional breakpoint to stop in the loop when it loops the 99th time.

4. **Adding a conditional breakpoint**
  - a. Select line 8.
  - b. Right-click on line 8.
  - c. Click the **Breakpoints** tab in the upper right pane of the Debug perspective. The Breakpoints view opens.

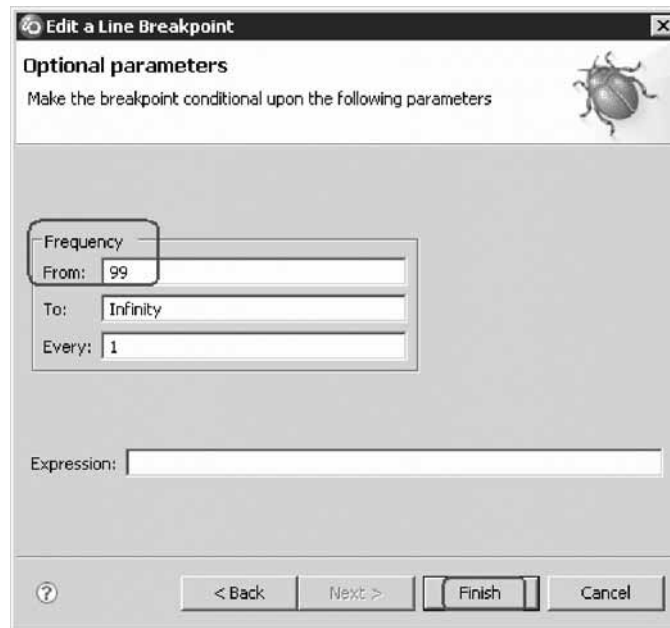


- d. Right-click anywhere within the **Breakpoints** view.
- e. Click **Add Breakpoint** → **Line** on the pop-up menu. The Add a Line Breakpoint window opens.



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

- f. 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.

- g. Under **Frequency** in the **From** field, type 99.
- h. Click **Finish**.

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

## Lesson checkpoint

You learned the following:

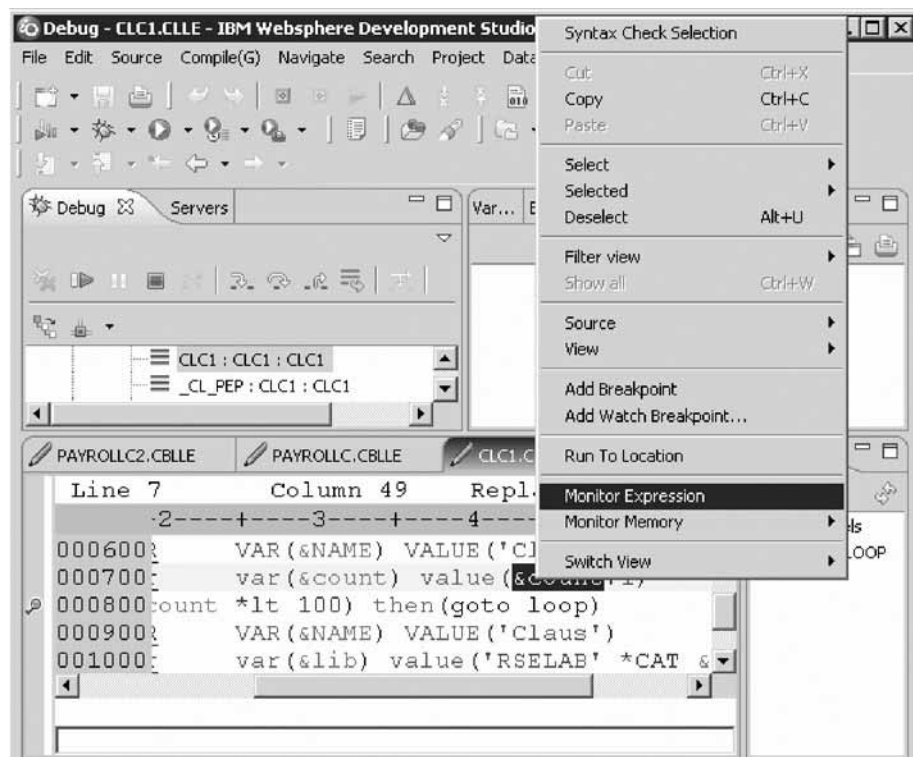
- About breakpoints
- How to add a breakpoint and a conditional breakpoint

## Monitoring variables

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

To monitor a variable:

1. In the Source view, double-click the variable `&count`.



2. Right-click `&count`.
3. Click **Monitor Expression** on the pop-up menu.

The Monitors view opens.

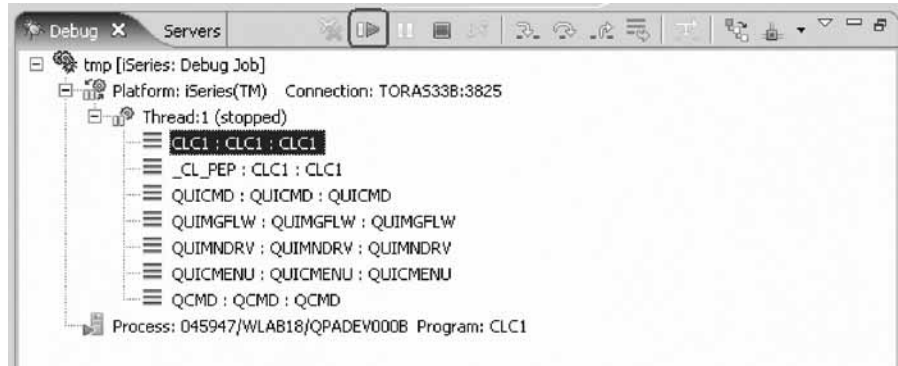



The variable appears in the Monitors view. Its current value is zero.

**Tip:** If you quickly want to see the value of a variable without adding it to the Monitor, leaving the mouse pointer on a variable for a second or so will display its value in a pop-up window.

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



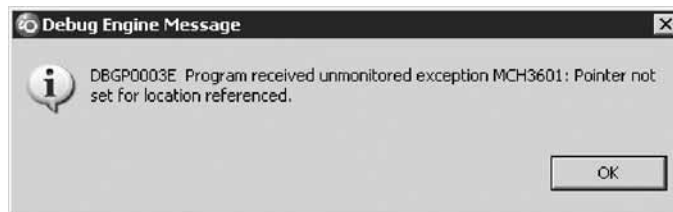



4. Click the **Resume**  icon from the Debug toolbar.  
The program starts running and stops at the breakpoint at line 8. (Be patient, the Debugger has to stop 98 times but because of the condition continues to run until the 99th time.) Notice in the Monitors view, that *&count* now has the value 99.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

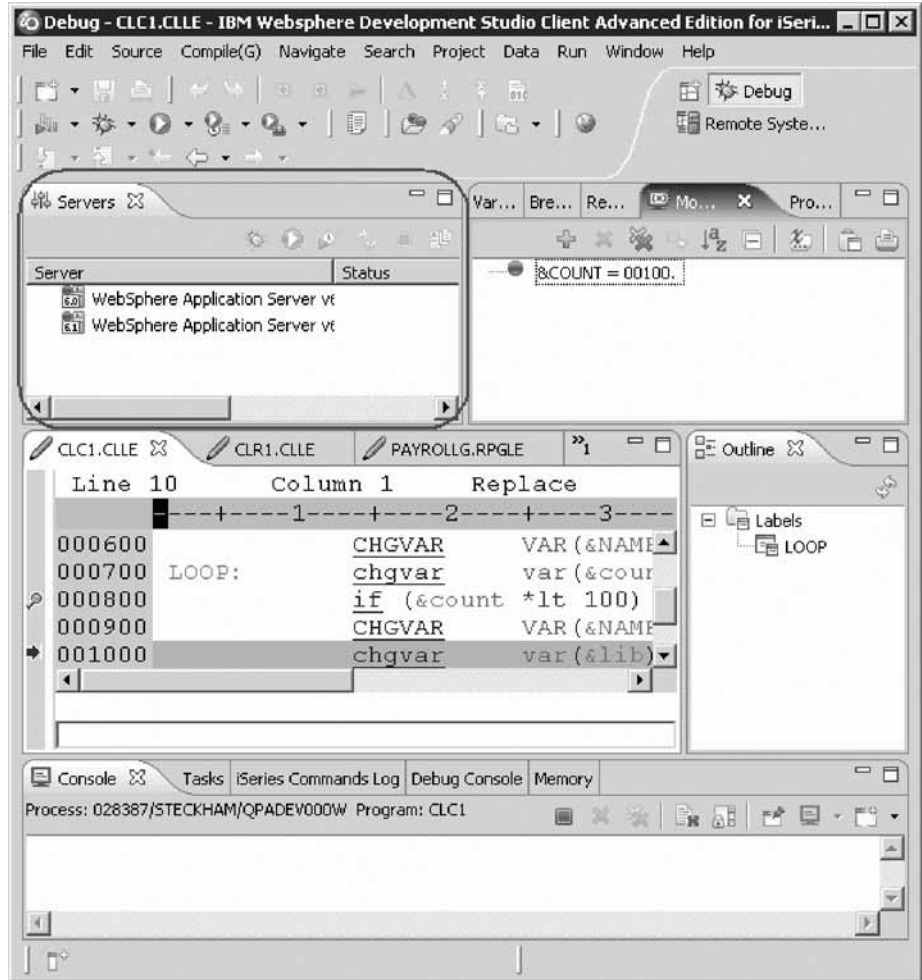
**If you do not see the error message below, go to “Stepping into a program” on page 71.**

#### **Error Handling**

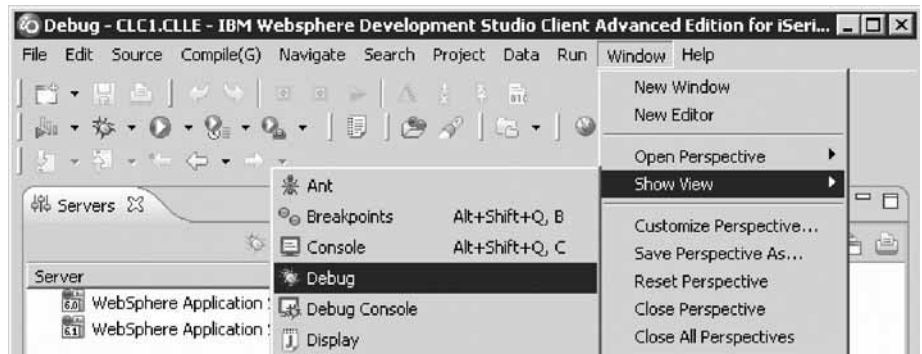
If you forget to add the parameter to the CALL program command when you call the program, you will see this error message.



- a. Click **OK**.
- b. 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.

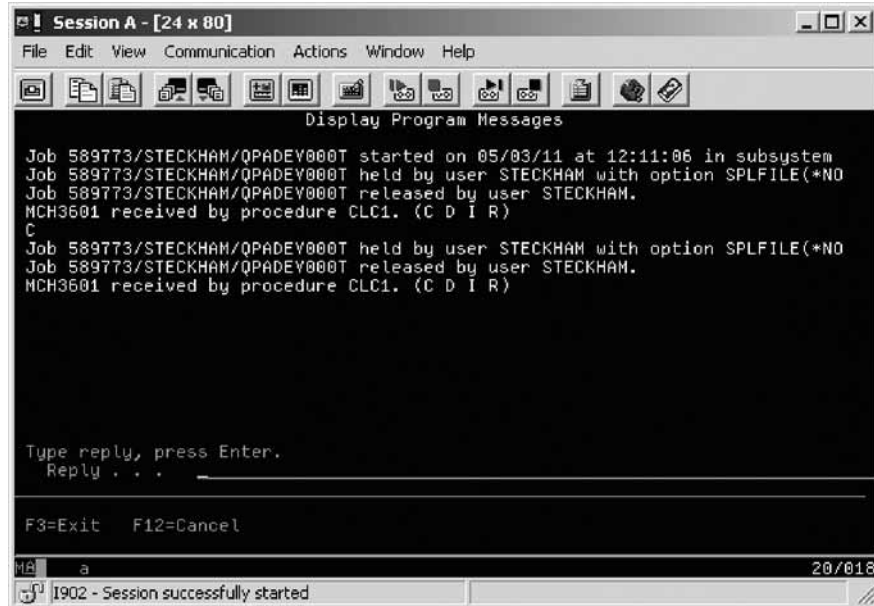


Click **Window > Show View > Debug**.



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

- c. Go to your 5250 emulator and press **Enter** until the program messages complete.



- d. In the workbench, click the **Remove all terminated launches** icon on the Debug toolbar to clean up the Debug view.



- e. From a 5250 command line, call the CLC1 program with the parameter XX.  
CALL PGM(RSELABxx/CLC1) PARM('XX')

You have monitored the variable *&count*.

## Lesson checkpoint

You learned the following:

- About the Monitor Expression view
- How to monitor a variable
- How to run a program

## 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.

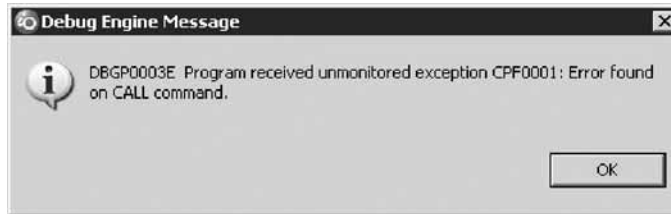
You are going to step into the PAYROLLD program.

To step into a program:

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

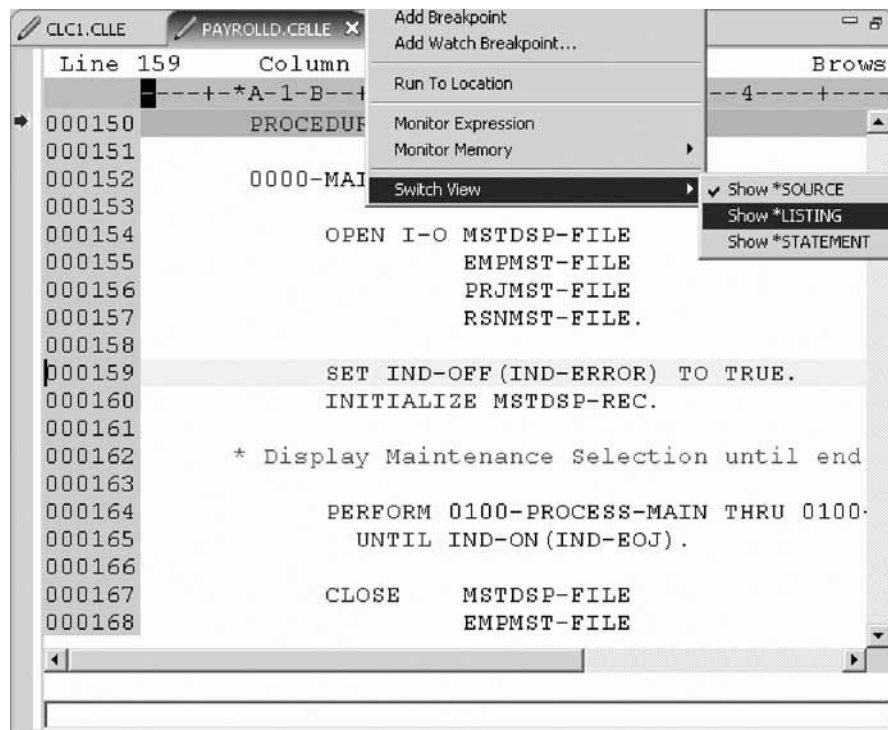


The source of PAYROLLD is displayed. Depending on the option you used to compile the program (\*SRCDBG or \*LSTDBG for RPG, or \*SOURCE, \*LIST, or \*ALL for ILE RPG), this window displays either the Source or Listing View. If you specified an incorrect parameter for the CALL program command, or your 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 68.

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 PAYROLLD 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 PAYROLLD program, switched the view from source to listing and back to source

## Lesson checkpoint

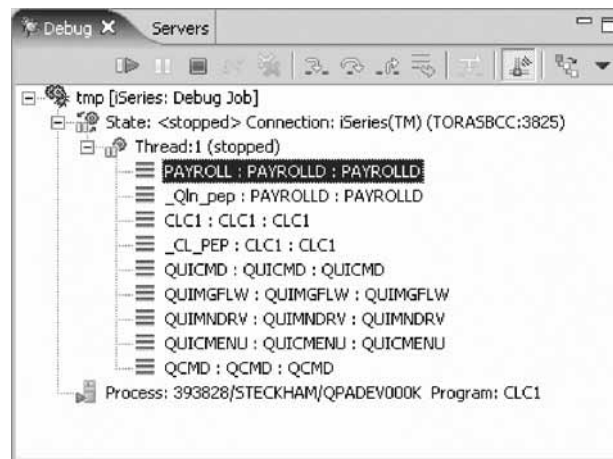
You learned the following:

- About step commands
- How to step into a program

## 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.

## Lesson checkpoint

You learned the following:

- About call stacks
- How to display source from entries

## Setting breakpoints in PAYROLLD

Now you add some breakpoints in PAYROLLD.

To add breakpoints:

1. Select PAYROLLD in Thread1.
2. In the source view (also called the iSeries default editor) scroll to line 201.
3. Double-click the prefix area of line 201.

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

- Repeat the above step for line 206.

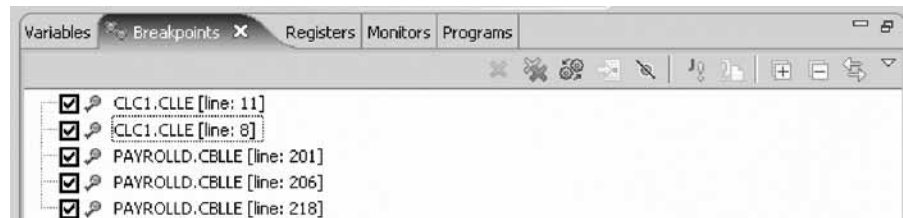
The screenshot shows a code editor window with two tabs: 'CLC1.CLLE' and 'PAYROLLD.CBLLE'. The code is displayed in a table-like format with columns for 'Line', 'Column 1', 'Replace', and 'Browse'. A breakpoint icon (a small circle with a vertical line) is placed in the 'Column 1' area of line 218. The code includes several conditional blocks and performance statements.

```

Line 218      Column 1      Replace      Browse
-----+---+*A-1-B-+-----2-----+-----3-----+-----4-----+-----
000200          IF IND-OFF (IND-ERROR)
000201          IF (EMPAPL OF SELECT-I = 'X')
000202              INITIALIZE EMPSEL-O
000203              PERFORM 0300-EMPMST-SELECT T
000204                  UNTIL IND-ON (IND-MAINT) OR
000205          ELSE
000206          IF (PRJAPL OF SELECT-I = 'X')
000207              INITIALIZE PRJSEL-O
000208              PERFORM 0400-PRJMST-SELE
000209                  UNTIL IND-ON (IND-MAINT)
000210          ELSE
000211              INITIALIZE RSNSEL-O
000212              PERFORM 0500-RSNMST-SELE
000213                  UNTIL IND-ON (IND-MAINT)
000214          END-IF
000215          END-IF
000216          END-IF.
000217
000218          IF IND-ON (IND-MAINT)
  
```

- Repeat the above step for line 218.

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.

You have added several breakpoints to PAYROLLD.

### Lesson checkpoint

You learned the following:

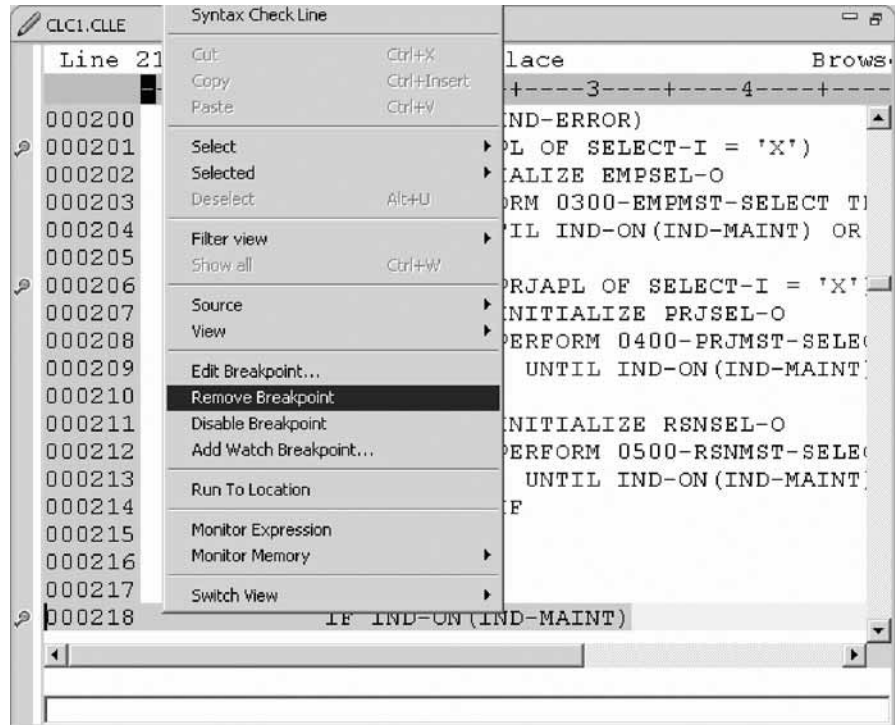
- How to add breakpoints

## Removing a breakpoint in PAYROLLD

It is also easy to remove breakpoints from the Source view.

To remove a breakpoint:

1. Right-click the prefix area of line 206.
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. Now you are ready to run the PAYROLLD program.

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

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



4. Type an X beside the **Project Master Maintenance** option.
5. Press **Enter** in the emulation session. The program runs to the breakpoint at line 201.

You have removed a breakpoint from PAYROLLD.

### Lesson checkpoint

You learned the following:

- How to remove a breakpoint

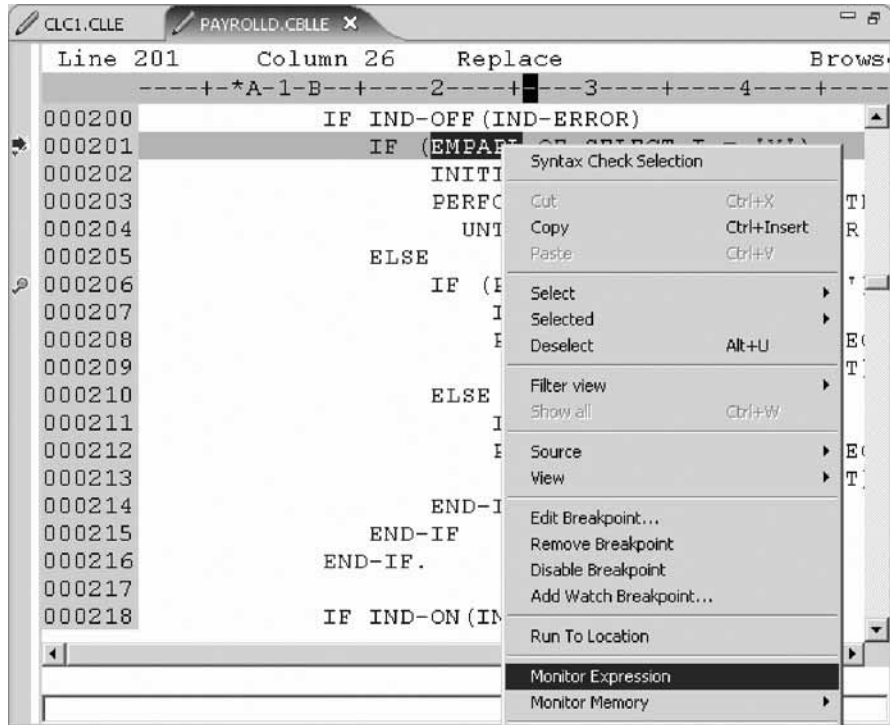
## Monitoring variables in PAYROLLD

Now lets monitor variables and change them in PAYROLLD.

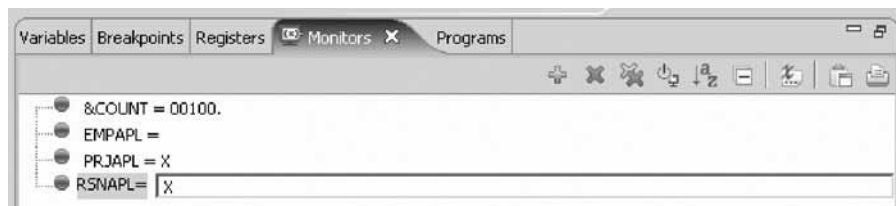
To monitor variables:

1. In the source view, double-click the variable EMPAPL on line 201.
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 206 and RSNAPL on line 244 to the monitor. Variable PRJAPL equals X because you did select the **Project Master Maintenance** option.
6. In the Monitors view, double-click the variable RSNAPL. 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.

You have monitored several variables in PAYROLLD.

### Lesson checkpoint

You learned the following:

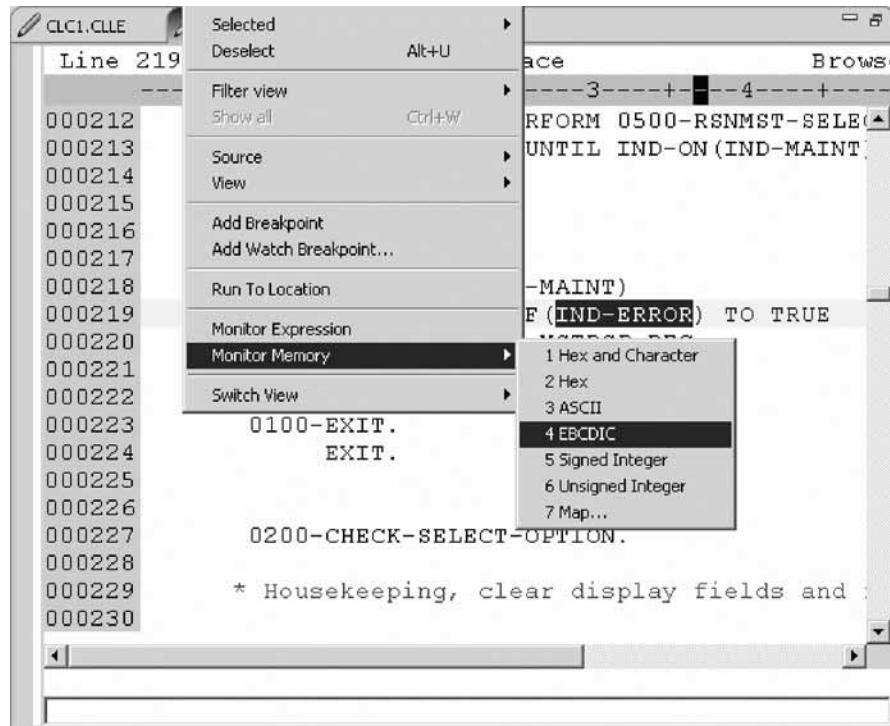
- About monitoring variables
- How to monitor variables

## Adding a memory monitor

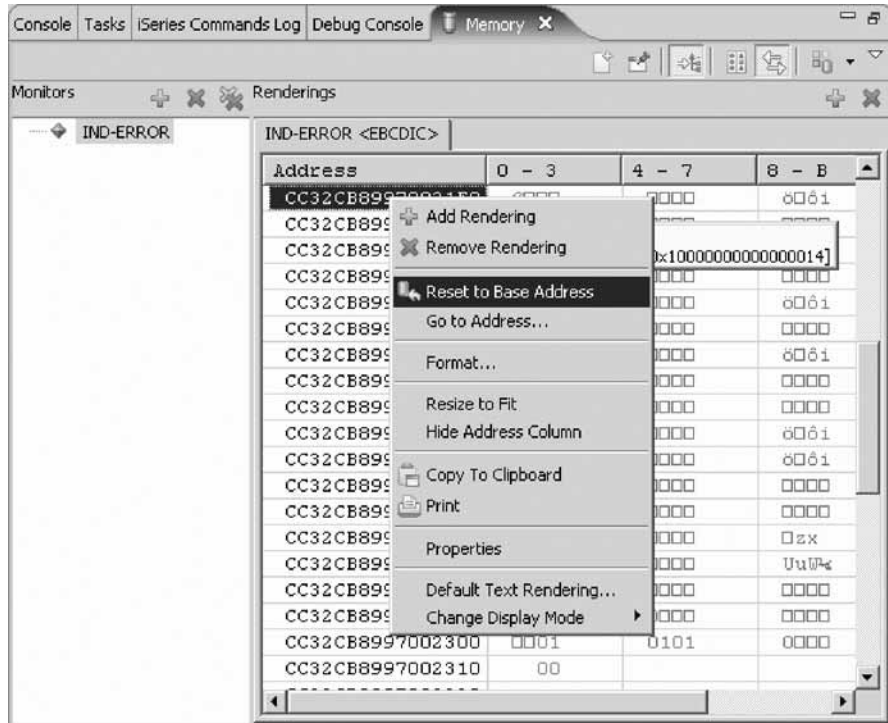
Adding a memory monitor for a variable allows you to view the memory starting with the address where the variable is located. The memory can be displayed in different formats, for example hexadecimal and text.

To add a memory monitor:

1. In the Source view, double-click the variable `IND-ERROR` in line 219.
2. Right-click and select **Monitor Memory** > **EBCDIC** on the pop-up menu.



This will open the Memory view in the notebook 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 **Monitor Memory > Hex**. 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 IND-ERROR.

### Lesson checkpoint

You learned the following:

- About memory monitors
- How to add a memory monitor

## 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:

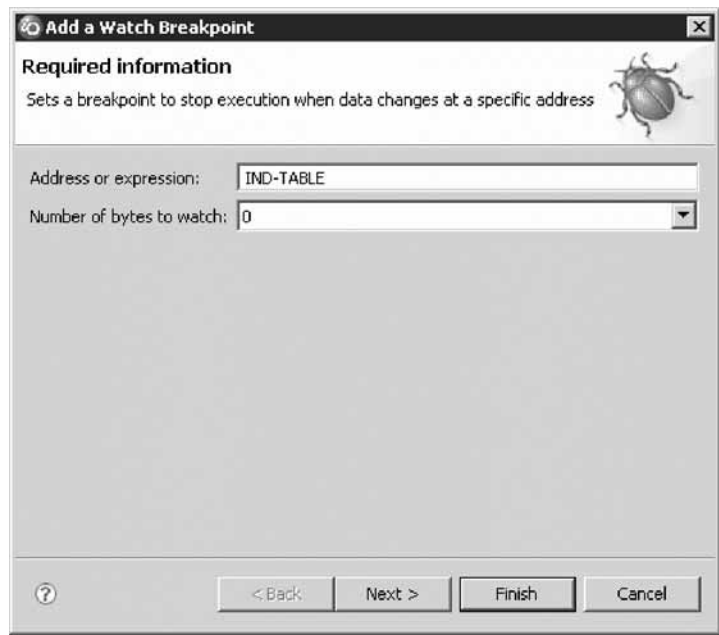
1. Go to the **Line number** field at the bottom of the source area. In this field enter 118 to go to that line.

```

Line 219      Column 37      Replace      Brows
-----+--*A-1-B--+-----2-----+-----3-----+-----4-----+-----
000212                                PERFORM 0500-RSNMST-SELE
000213                                UNTIL IND-ON (IND-MAINT
000214                                END-IF
000215                                END-IF
000216                                END-IF.
000217
000218                                IF IND-ON (IND-MAINT)
000219                                SET IND-OFF (IND-ERROR) TO TRUE
000220                                INITIALIZE MSTDSP-REC
000221                                END-IF.
000222
000223                                0100-EXIT.
000224                                EXIT.
000225
000226
000227                                0200-CHECK-SELECT-OPTION.
000228
000229                                * Housekeeping, clear display fields and
000230

```

2. Double-click variable IND-TABLE 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 IND-TABLE.  
 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.



6. In the 5250 emulation session, type 123 for **Project Code** and D (for delete) in the **Action Code** field.
7. Press **Enter**. A message is displayed indicating that the variable IND-TABLE has changed.



8. Click **OK**.
9. In the Breakpoints view, right-click the Watch breakpoint and click **Disable** on the pop-up menu.

You have added a Watch breakpoint for the variable IND-TABLE and run the program to see the notification that the variable has changed.

### Lesson checkpoint

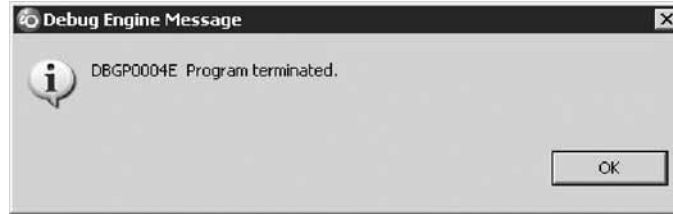
You learned the following:

- About Watch breakpoints
- How to set a watch breakpoint and see the results

### 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**.

### Lesson checkpoint

You learned the following:

- How to terminate a debug session

## Starting the Integrated Debugger using the Debug action

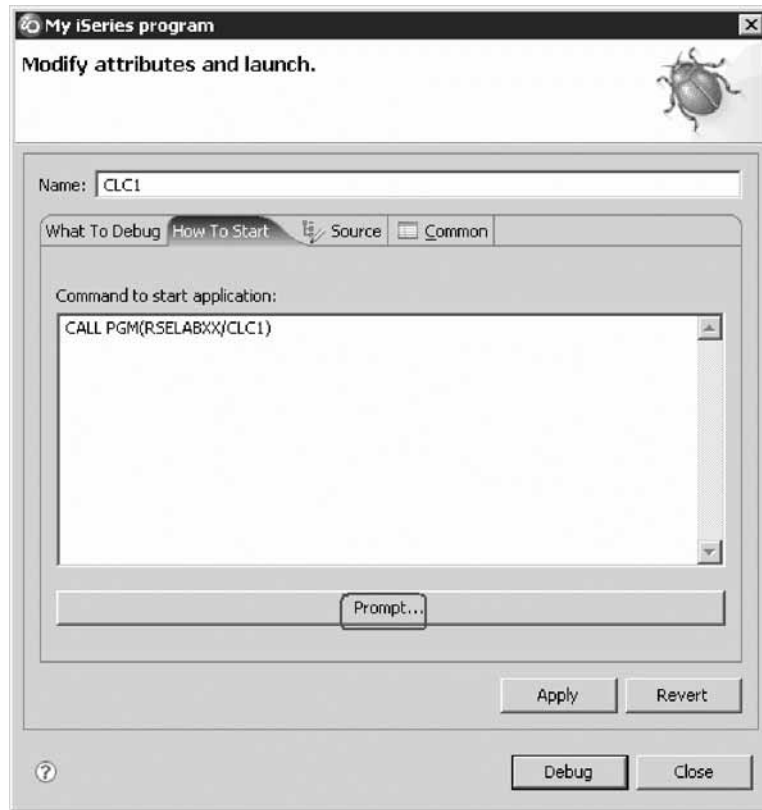
You will be working with the COBOL program PAYROLLD.

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.

CLC1 requires a parameter.

To start the debugger:

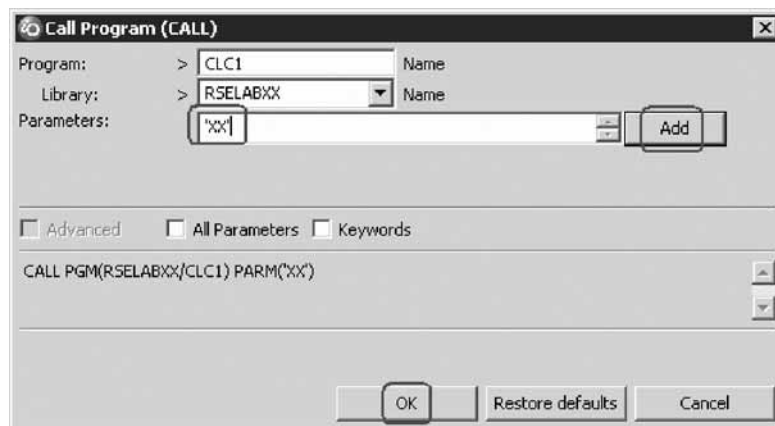
1. In the **Name** field, type the program name CLC1. This gives your debug configuration a unique name so you can use it again when you debug this program.
2. You can leave the **Step into** and **Terminate debug session on program completion** check boxes selected and **Update production files** check box deselected, since you are working with a test library.
3. Click the **How To Start** tab.



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

4. Click **Prompt**.

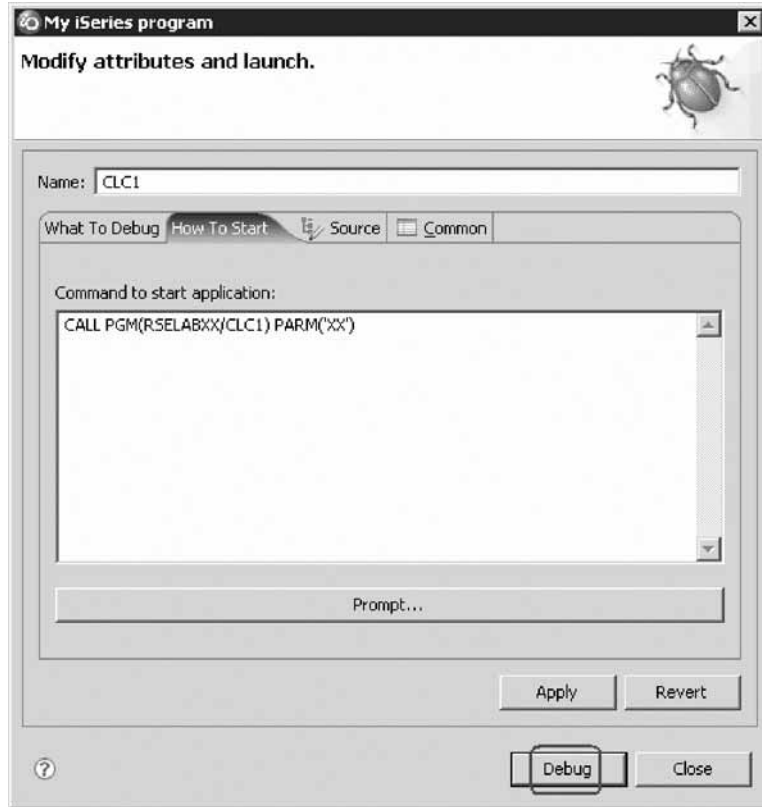
The Call Program (CALL) window opens.



5. In the **Parameters** field, type 'XX' where 'XX' is your workstation number.

6. Click **OK**.

The complete start command for the program appears.



7. Click **Debug**.

The Debug perspective opens.

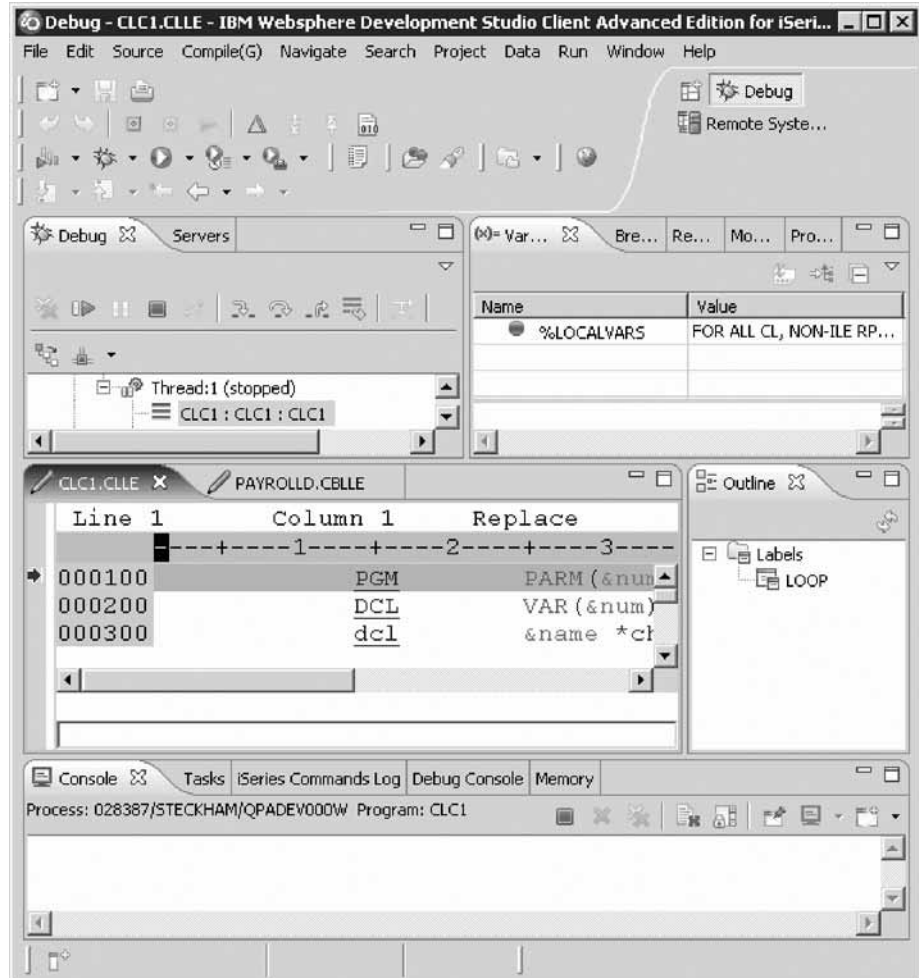
If not, you may see this error message.



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.

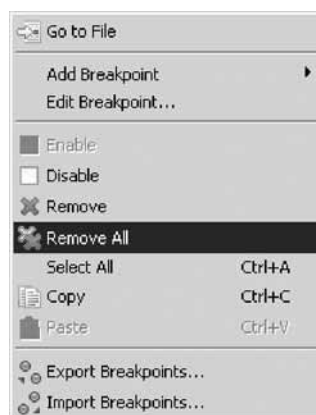
Now that the program is active on the iSeries and stopped at the first executable statement, the debugger displays the source.






You have started an interactive debug session.

- 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.



- Click **Resume**.

PAYROLLD 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, click **Suspend** 

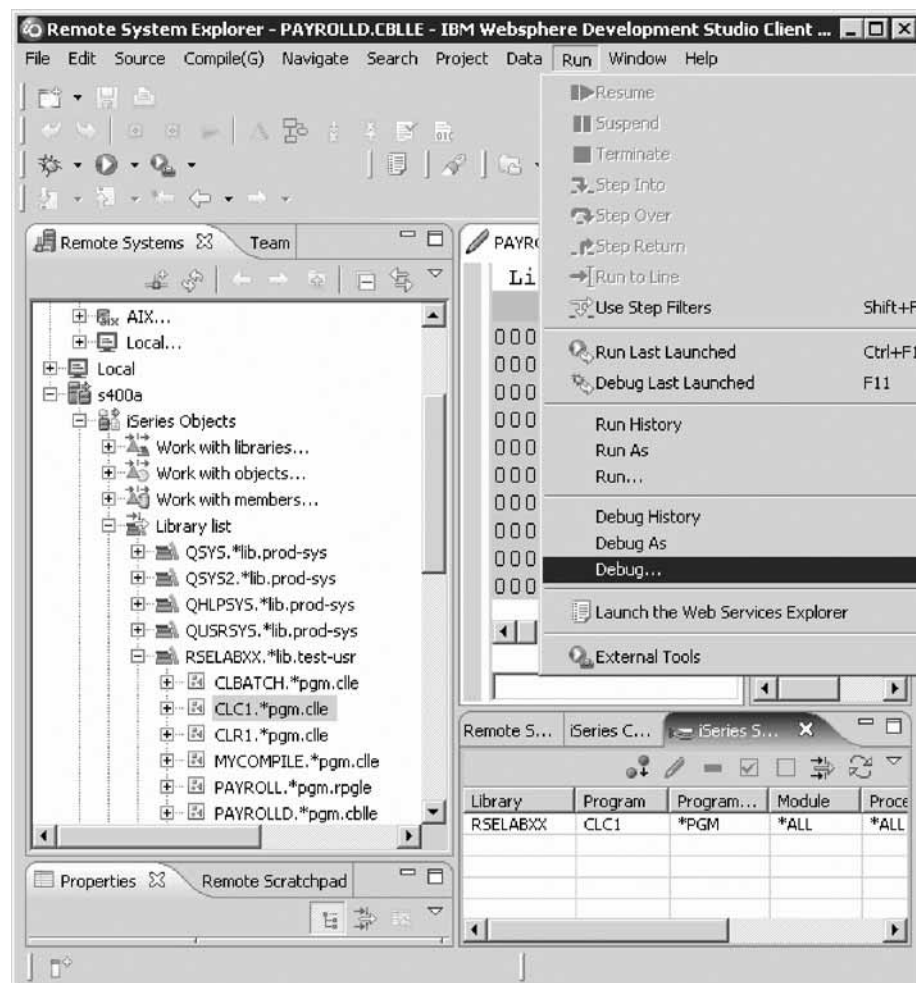
You can now set breakpoints and use all the Debug features.

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

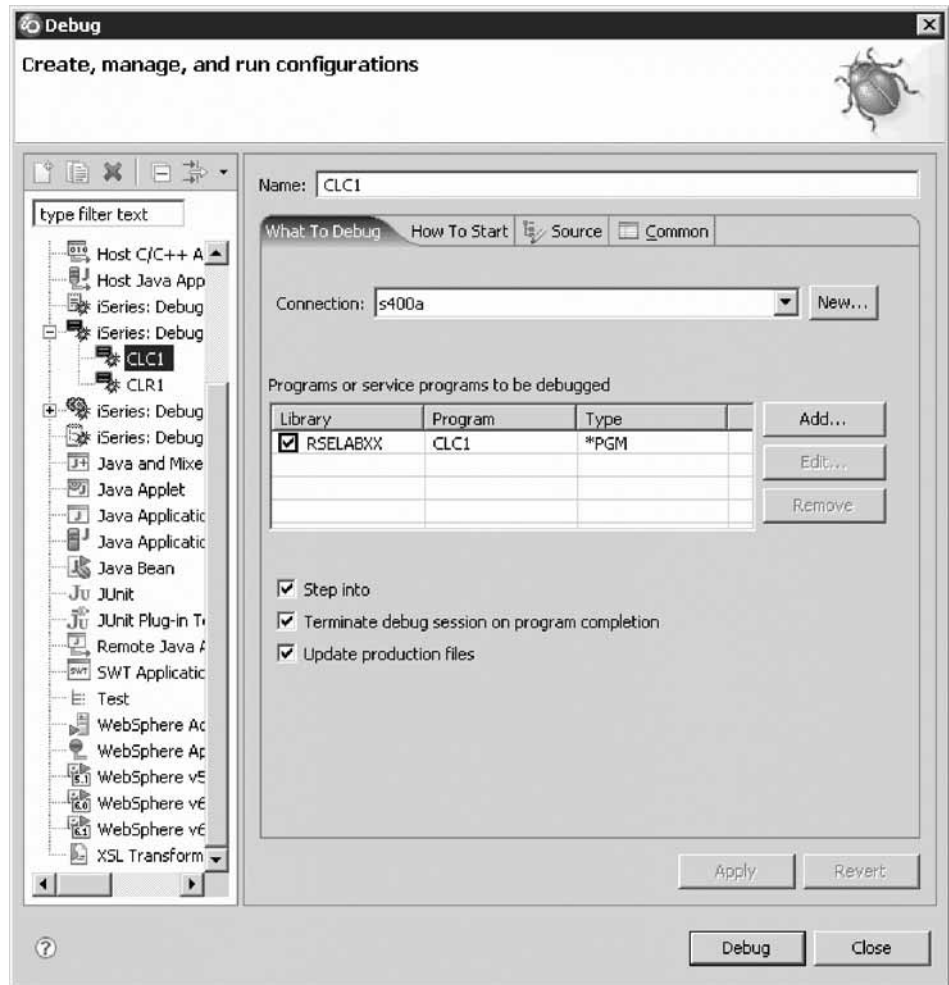
- Click **Terminate** to end the Debug session.  
 The Debug session is terminated, but this does not end the program.
- Switch to the 5250 emulation session and press **F3** to end the program.

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

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



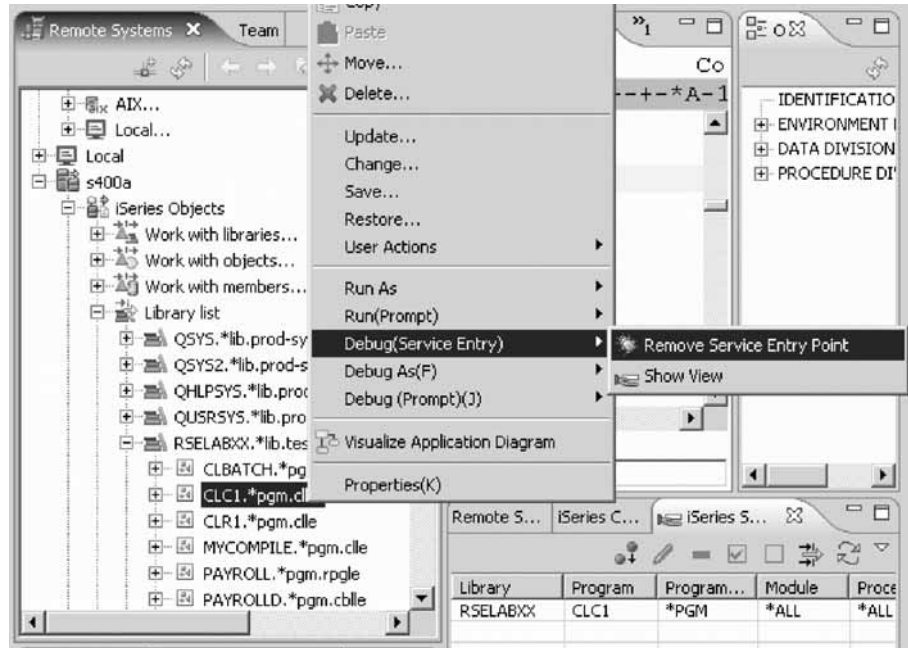
The Debug Launch Configurations window opens.



Here you can see the CLC1 configuration that you just created. This is your saved configuration to debug CLC1 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.

13. To remove the service entry point and close the launch configuration:
  - a. In the Remote System Explorer perspective, expand library RSELABxx, if it isn't expanded already.
  - b. Right-click program CLC1 in library RSELABxx and 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.

### Lesson checkpoint

You learned the following:

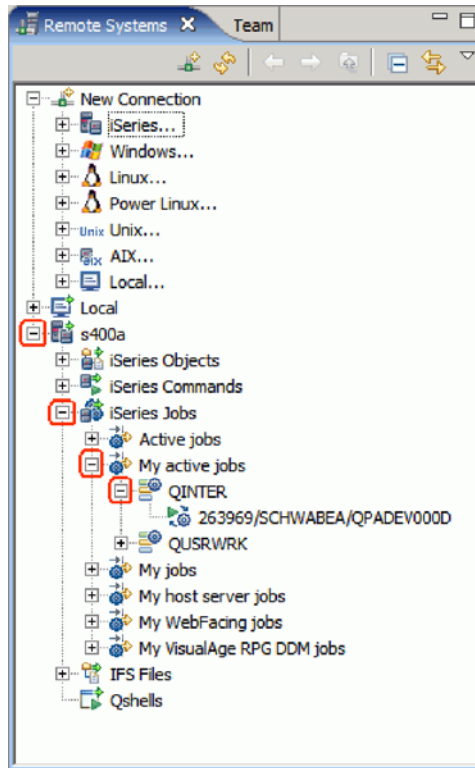
- About other ways to start the debugger
- How start the debugger using a debug action
- How to remove a service entry point

## Debugging a Job

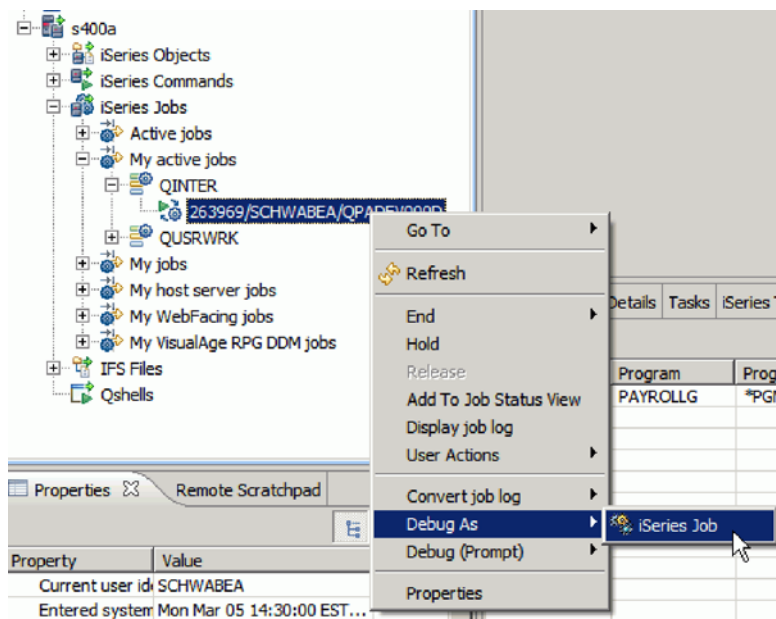
In addition to being able to debug a program, you can also debug a job.

To debug a job:

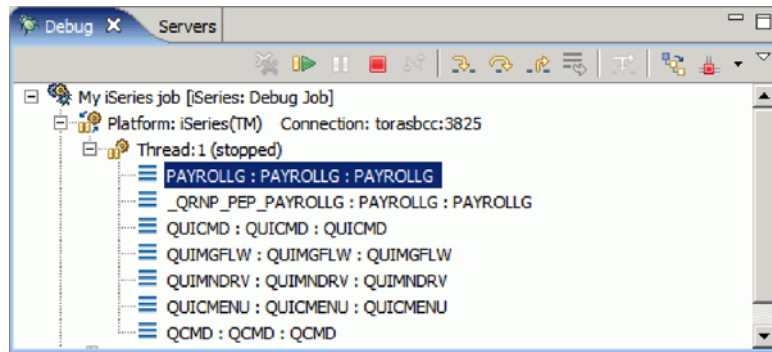
1. Open the Remote System Explorer perspective by selecting **Window** → **Open Perspective** → **Remote System Explorer**
2. Under your active server connection, s400a, expand **iSeries Jobs** → **My Active Jobs** → **QINTER**.



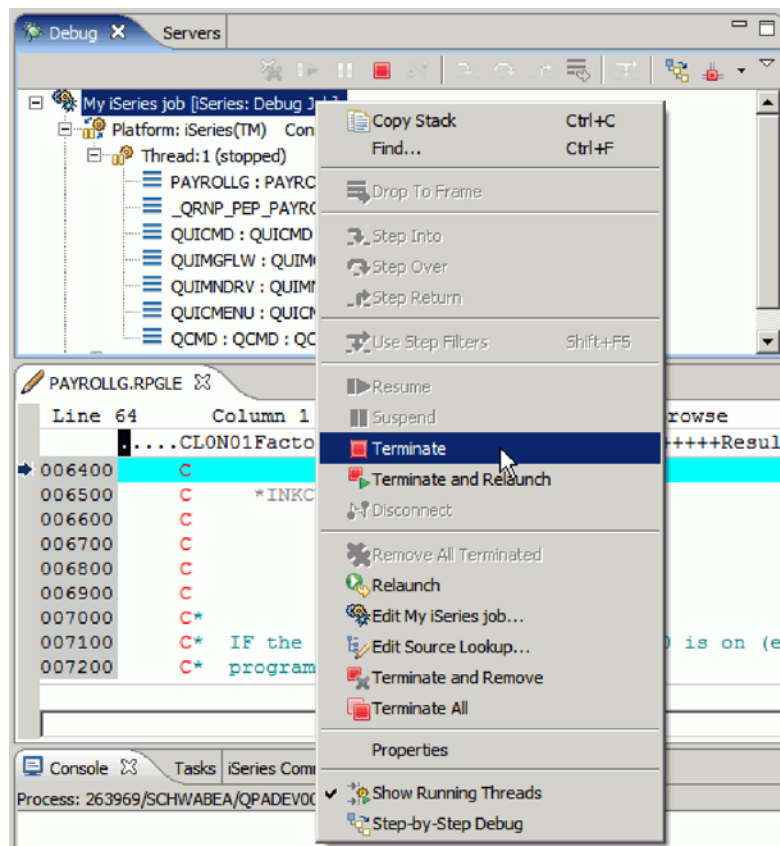
3. Right-click the active job under QINTER, and select **Debug As** → **iSeries Job**.



The debug session begins. From here you can set breakpoints, monitor variables, and memory in the same way that you did with a program.



4. Terminate the debug session by right-clicking **My iSeries job** and selecting **Terminate**.



The debug session is terminated.

### Lesson checkpoint

You learned the following:

- How to debug an iSeries Job

### Module summary

In this module, you learned how to debug a program using the Integrated iSeries 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 Launch Configurations window

## Assessment

- How do you start a debug session using service entry points?
- Where can you only set breakpoints?
- How do you set a breakpoint?
- What is the Monitors view?
- How do you change variables and indicators?
- How do you step over or into a program?
- What actions can you perform on breakpoints?
- What is the Memory Monitor?
- What type of breakpoint provides a notification to a user when a variable changes.

---

## Customizing the Remote System Explorer

This module teaches you how to use the Remote System Explorer perspective to work with the iSeries objects that you used in the previous modules. You will learn how easy it is to define filters, perform actions and define your own actions. In short, you'll see how Remote System Explorer can organize and integrate your work and make that work easier.

This module also teaches you how to move, re-size or close existing views. You learn how to open other views that you want to add to the perspective. You then save the customized layout as a new perspective.

### Learning objectives

- Know the features of Remote System Explorer
- Move, dock, rearrange, resize, hide, close, reopen and add views
- Save and reset a customized perspective
- Create a filter to show specific iSeries libraries
- Change the filter to add more iSeries libraries
- Create a filter to show all the source files in a library
- Access members to edit from your filter

- Create a user action that copies a source file with data to a new source file in the same library
- Specify user action parameters
- Specify a restriction on a user action
- Try the user action
- Create a user action for iSeries jobs
- Create a user action for IFS folders and files
- Create your own compile command
- Edit an existing compile command
- Using run configurations

### **Time required**

This module should take approximately 60 minutes to complete.

## **More about the Remote System Explorer**

The Remote System Explorer is replacing PDM (Program Development Manager) on the workstation. It currently doesn't have all the function of PDM but will over time eventually be a full replacement for PDM.

Remote System Explorer allows you to:

1. Simplify your work by giving you quick access to lists of iSeries libraries, objects, members, IFS files, UNIX<sup>®</sup> files, and local files.
2. Use the context-sensitive pop-up menus on these lists to perform actions such as start the Remote Systems LPEX Editor, CODE Designer, or Integrated Debugger or other common iSeries actions.
3. Use the Work with User Actions option to create and manage your own user-defined actions and have them appear in the pop-up menus.
4. Use the command support to increase your productivity by allowing you to enter and repeat iSeries or local commands without switching to an emulator session.

You have read the list of Remote System Explorer capabilities.

### **Lesson checkpoint**

You learned the following:

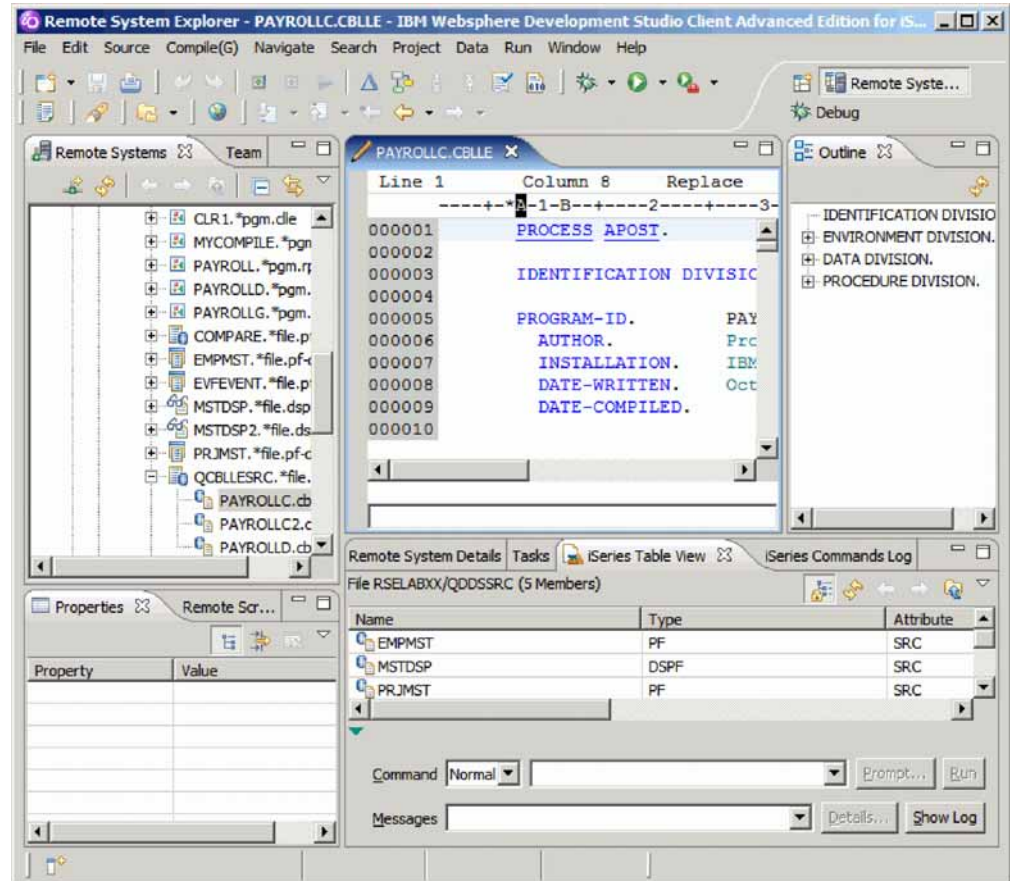
- About Remote System Explorer

## **Customizing the perspective**

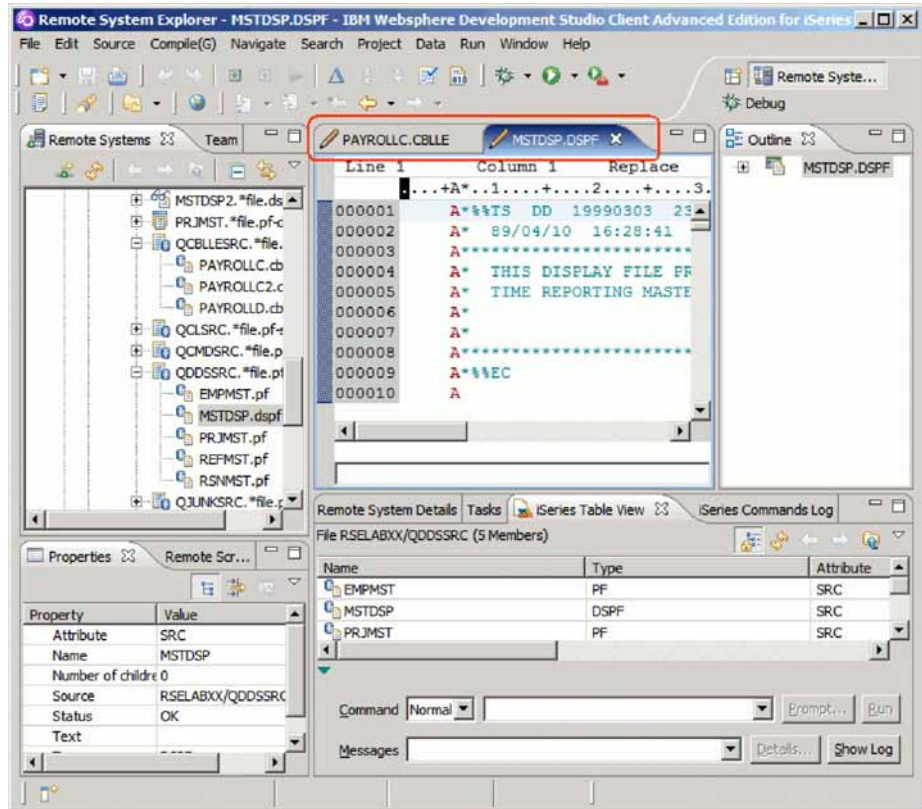
Perspectives can be modified to fit your work style. You can move, resize, or close existing views. You can open other views that you want to add to the perspective.

This is the familiar default appearance of the Remote System Explorer perspective.



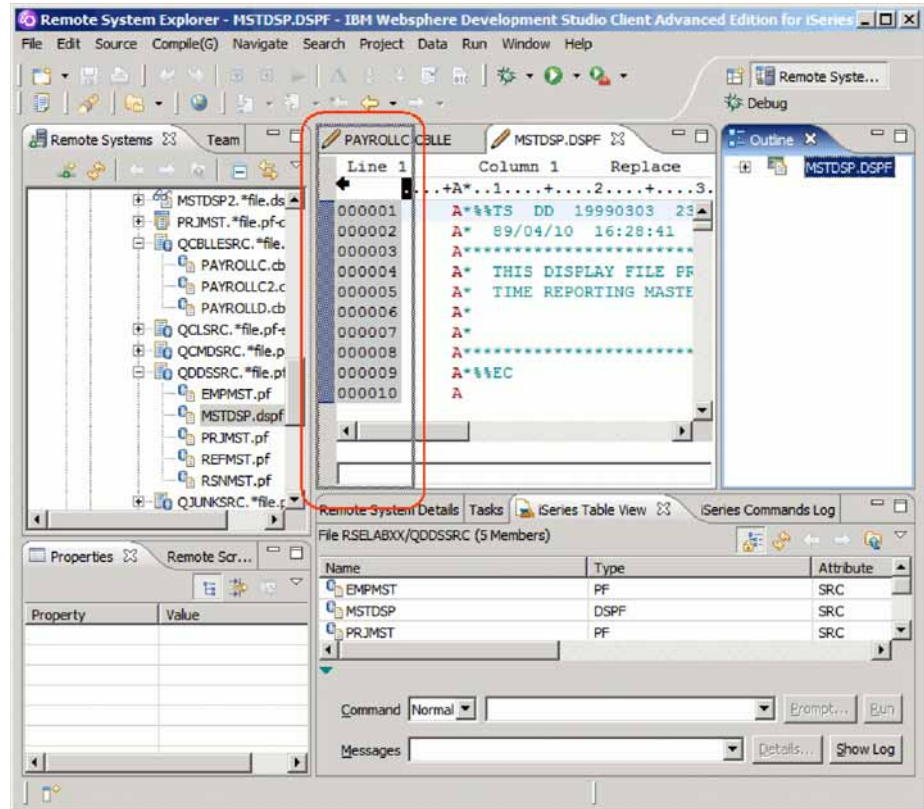


1. To move the Outline view:
  - a. From the Remote Systems view double-click member MSTDSP in the QDDSSRC source file.  
The Remote Systems LPEX Editor opens.
  - b. In the Remote Systems view, double-click member PAYROLLC in the QCBLESRC source file.  
This member will be loaded into the editor as well.  
Your perspective will look something like:

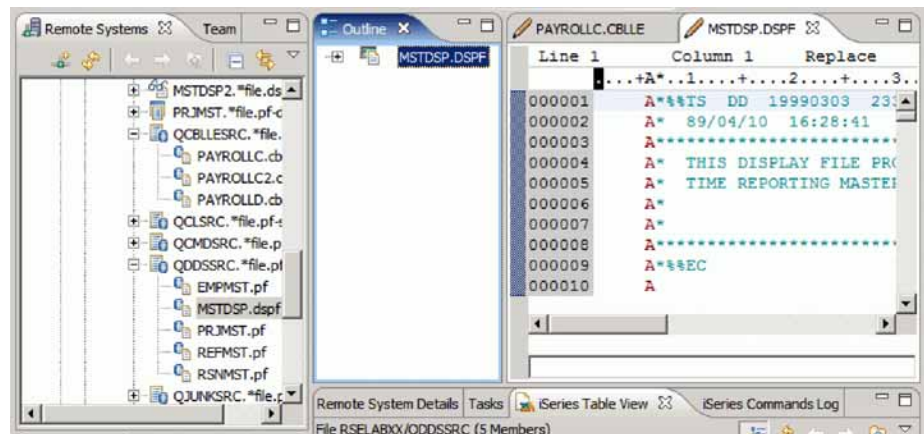


Notice the two tabs in the Editor window. Now let's customize the perspective.

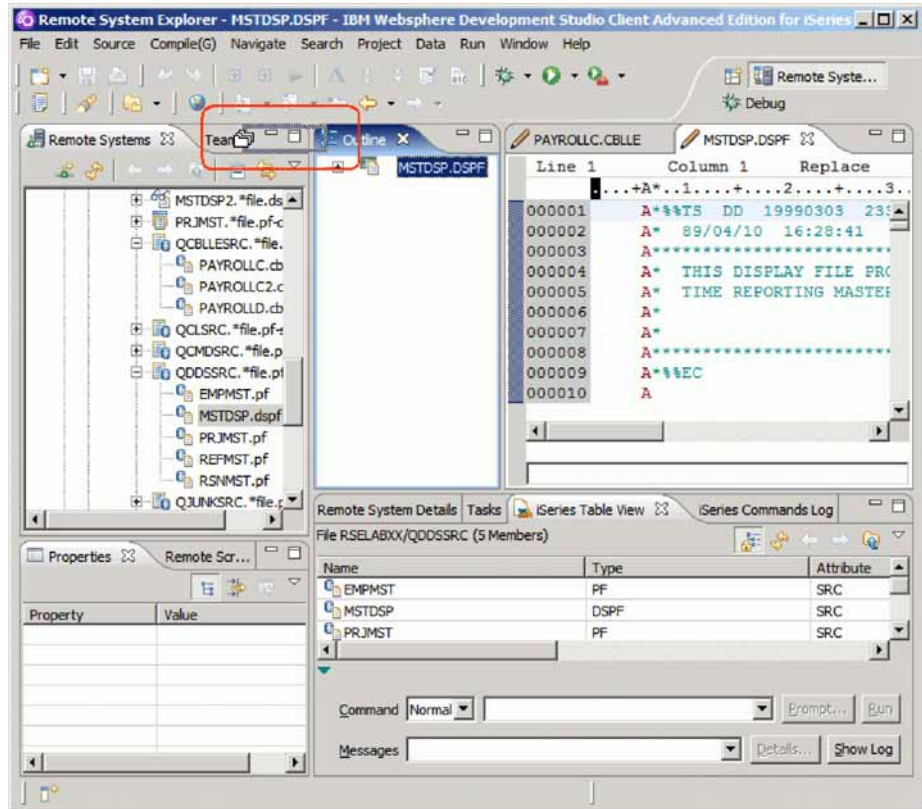
- c. Click the title bar of the Outline view in the workbench window and drag the view across the Workbench window. Do not release the left mouse button yet.
- d. While still dragging the view around on top of the workbench window, note that the various drop cursors appear. These drop cursors indicate where the view will dock in relation to the view or editor area underneath the cursor when you release your mouse button. To see the drop cursor change, drag the view over the left, right, top, or bottom border of another view or editor.



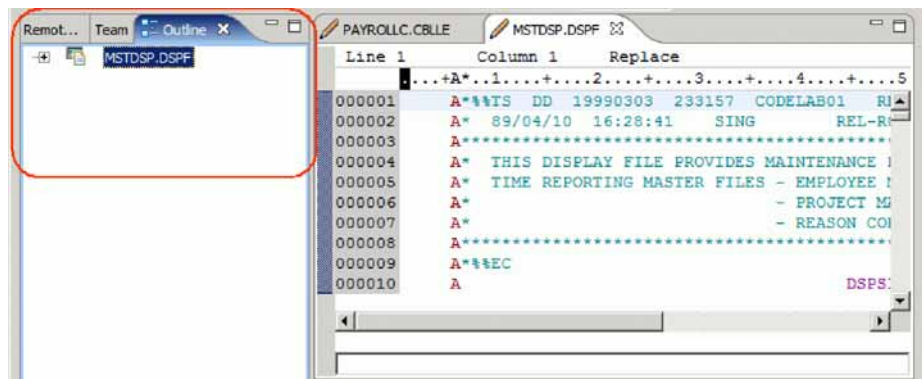
- e. Dock the view in any position in the Workbench window, and view the results of this action.



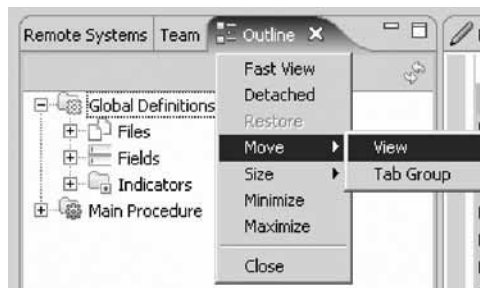
- f. Click and drag the view's title bar to re-dock the view in another position in the Workbench window. Observe the results of this action.
- g. Finally, drag the Outline view over the Remote Systems view. You will see a stack cursor.



- h. When you release the mouse button the Outline view will be stacked with the Remote Systems view into a tabbed notebook.



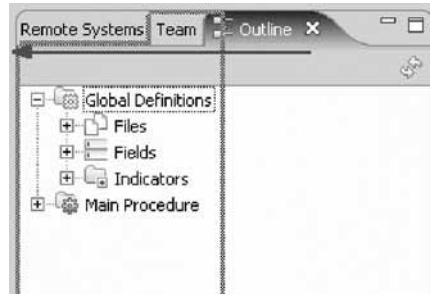
You can also move a view by using the pop-up menu for the view. Left-click on the icon at the left of the views title bar, or right-click anywhere else in the view's title bar.



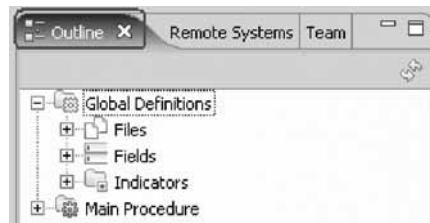
**Tip:** A group of stacked views can be dragged using the empty space to the right of the view tabs.

You can rearrange the order of views in the tabbed notebook.

- i. Click on the Outline view tab and drag it to be in front of the Remote Systems view tab.

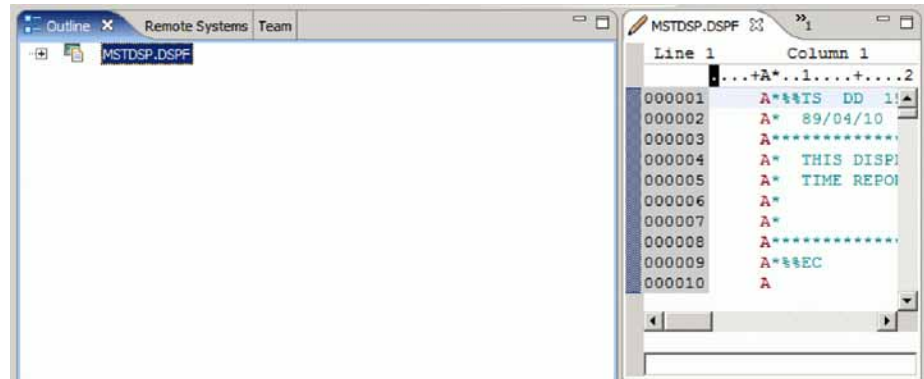


- j. Release the mouse button when the Outline view tab is in the desired location. The view that you selected is now moved.

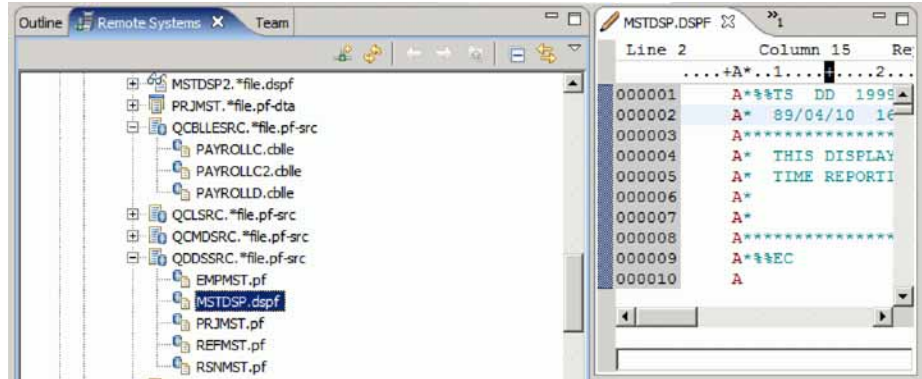


- 2. To resize the Outline view:

- a. To resize the Outline view, select the right border and drag the mouse pointer to the right to increase the size of the Outline view. Notice that the Editor and iSeries Commands Log got smaller.



- b. Click on any of the other tabbed views such as Remote Systems view and you will see this view has the same size as the Outline view.

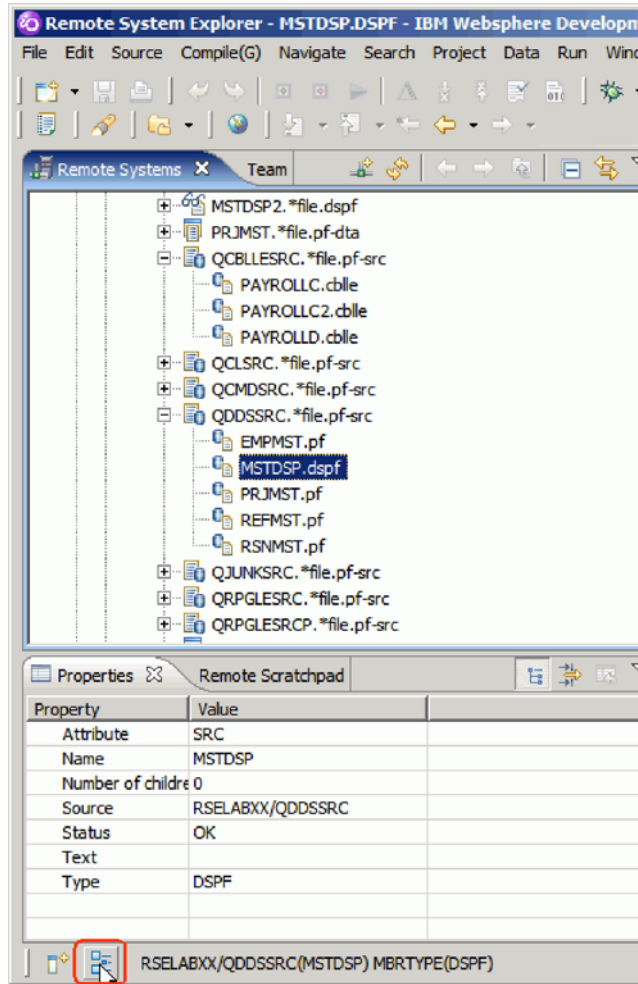


3. To hide a view:

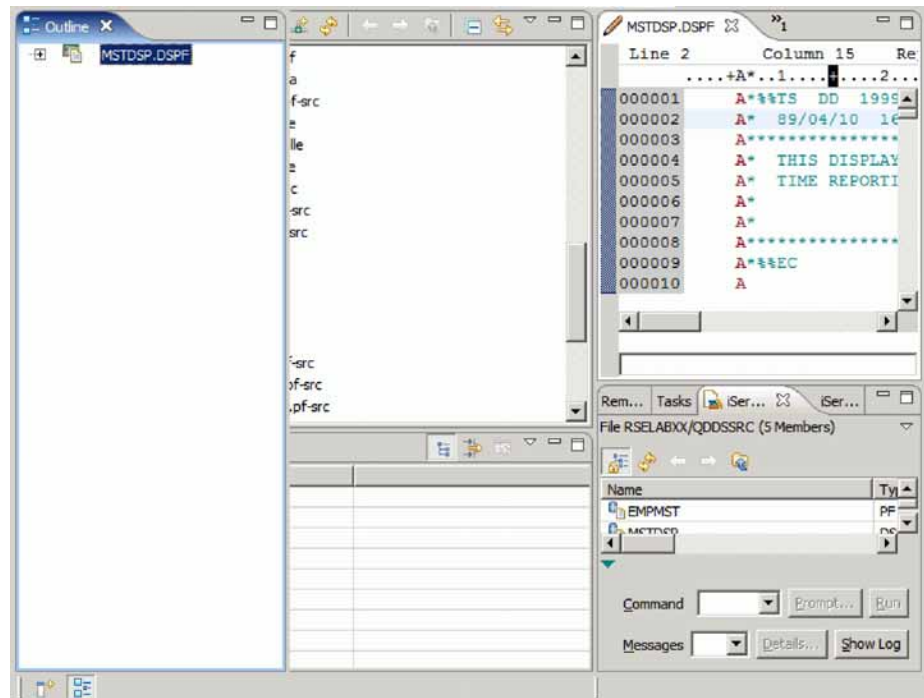
You can create fast views to quickly open and close frequently used views. They work like other views except they do not take up space in your Workbench window.

To create a fast view:

- a. You can dock views on the shortcut bar at the bottom of the workbench. Click the title bar of the Outline view.
- b. Hold the mouse button down.
- c. Drag the view to the shortcut bar at the bottom left of the window and release the mouse button. A toolbar button for that view that you dragged now appears on the shortcut bar.

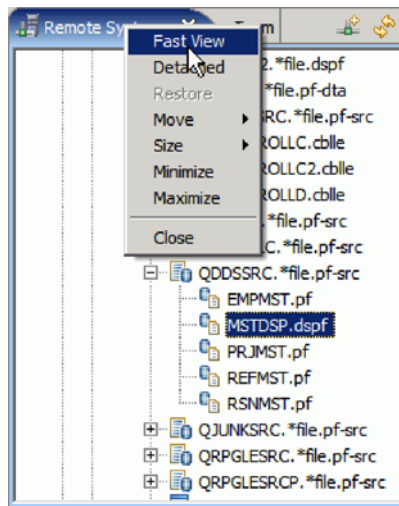


d. Click the toolbar button on the shortcut bar to look at the view.



- e. Click somewhere else outside the view to hide the view again.

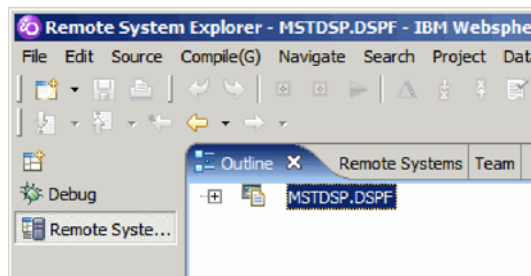
**Tip:** You can also create and restore fast views by selecting **Fast View** from the context menu of the view's title bar.



- f. Right-click the toolbar button on the shortcut bar and click **Fast View** on the pop-up menu to deselect it. This will show the view again for the next lesson.
4. To dock a perspective: By default, the shortcuts for the open perspectives are displayed at the top right of the workbench. You can dock these shortcuts somewhere else in the workbench.
    - a. To dock the shortcuts, right-click in the top right area of the workbench and select **Dock On > Left**.



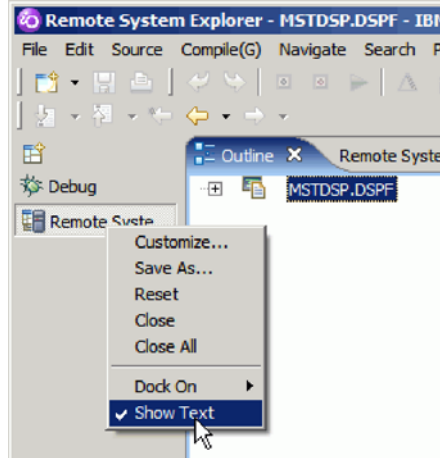
The view is docked on the left of the workbench.



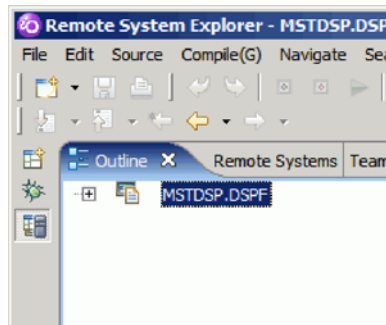
Once you are familiar with the short cut icons for the open perspectives, you can remove the text to save space.

- b. Right-click the shortcut for an open perspective and deselect **Show Text** on the pop-up menu.





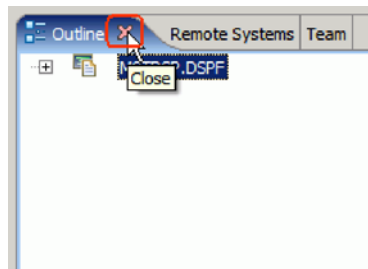
The text for all shortcuts for the open perspectives disappears.



5. To close, reopen and add other views:

You can remove views, reopen views and add new views to a perspective.

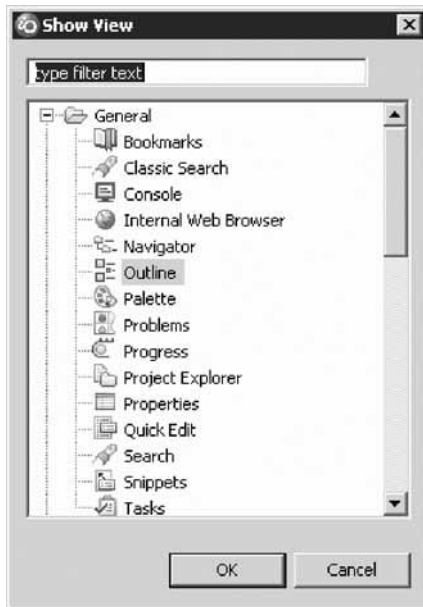
a. To remove the Outline view from the Remote System Explorer perspective, click the Close icon in the top right-hand corner of the tab.



b. Reopen the Outline view by clicking **Window > Show View > Other**.

The Show Views dialog opens.

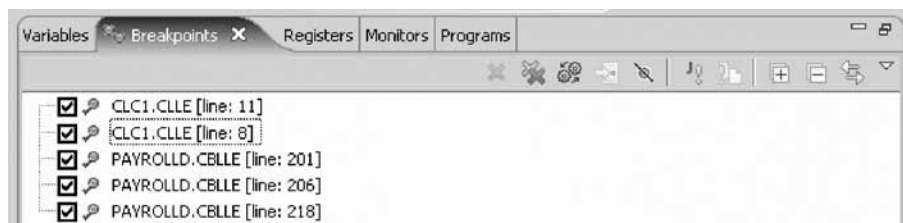
c. Expand **General** and select **Outline**.



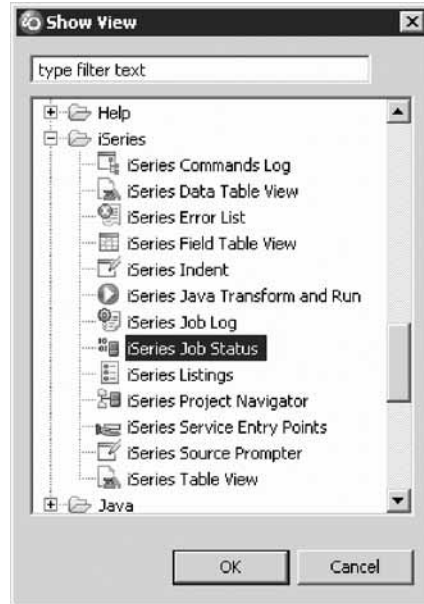
- d. Click **OK**. The Outline view opens in the workbench at the location where it resided last.

You can add views to the perspective.

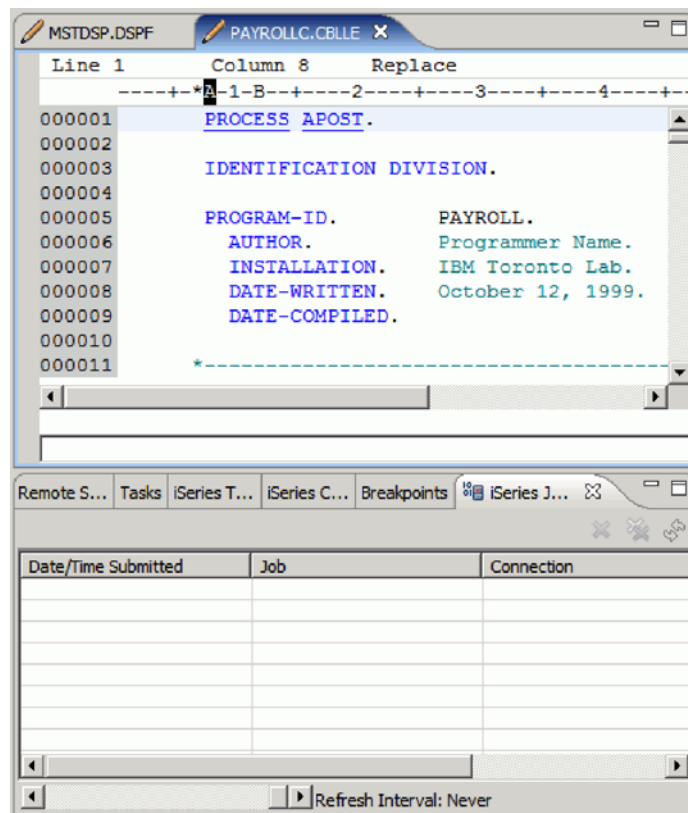
- e. To add the Breakpoints view, click **Window > Show View > Other**. Expand **Debug** and select **Breakpoints**



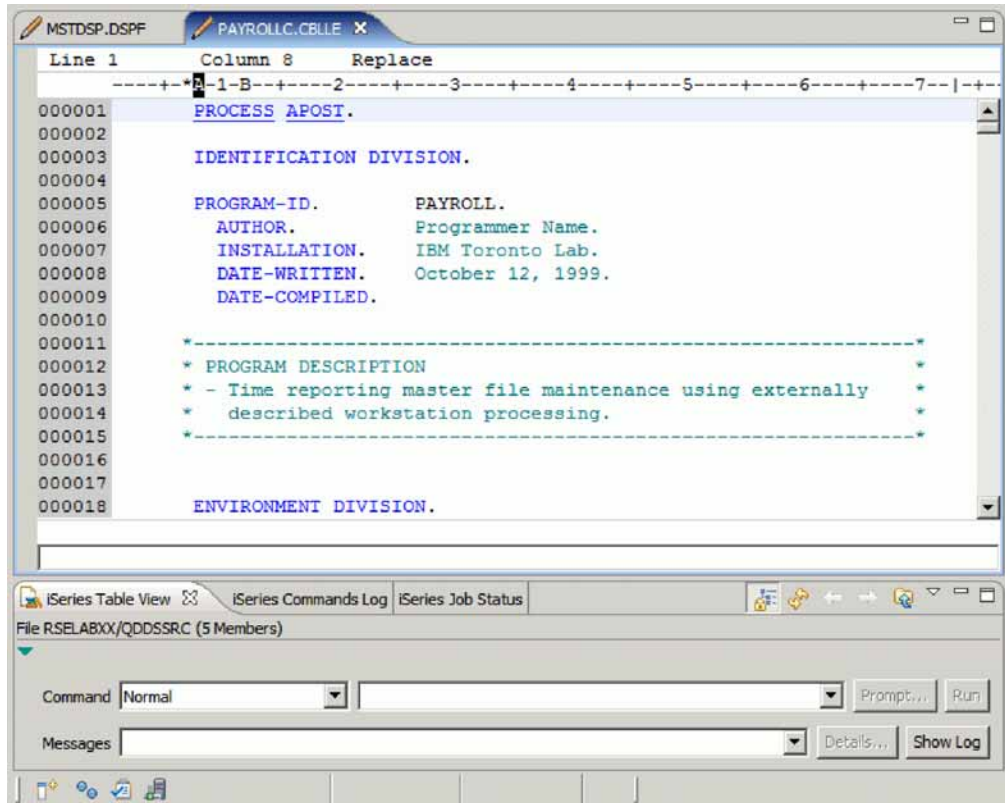
- f. To add more iSeries views, click **Window > Show View > Other**.
- g. In the Show View dialog, expand **iSeries** and choose a view from the list of views, for example iSeries Job Status.



- h. Click **OK**.  
The iSeries Job Status is added to the perspective.



Now you know how to add, move, hide, and close views. Manipulating the RSE perspective allows you to work with a highly flexible, and customized workbench. For example, you could change the perspective so that the editor takes up most of the space, the iSeries Table view resides below the editor and all other views are either closed or moved to the bottom as fast views, as shown here:



You have customized the Remote System Explorer perspective.

## Lesson checkpoint

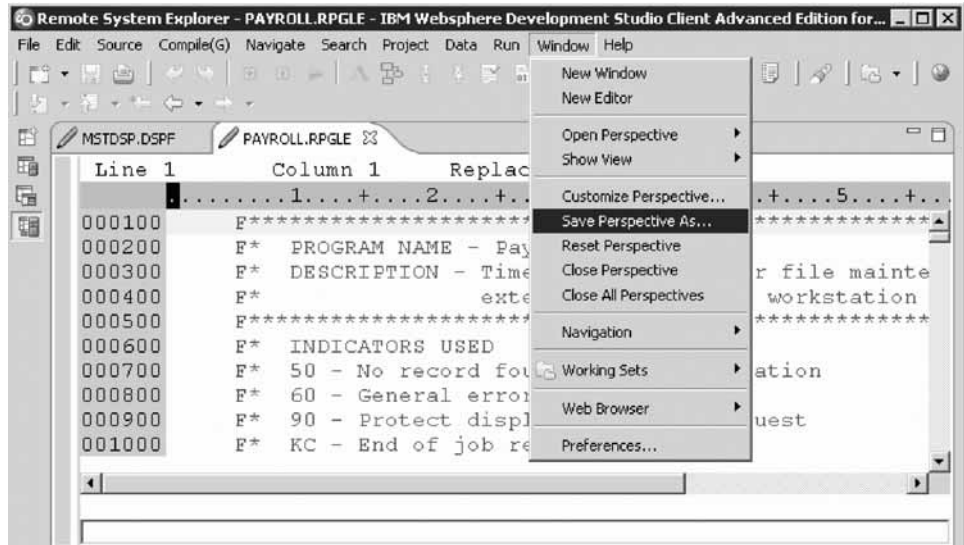
You learned the following:

- About customizing the Remote System Explorer
- How to customize the Remote System Explorer

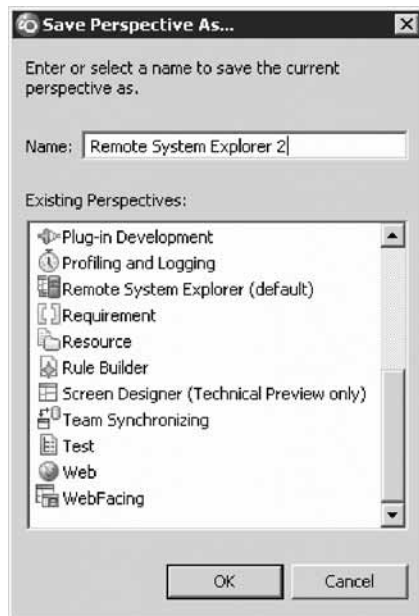
## Saving the perspective

If you have modified a perspective by adding, deleting, or moving (docking) views, you can save your changes for future use.

1. Click **Window > Save Perspective As**.

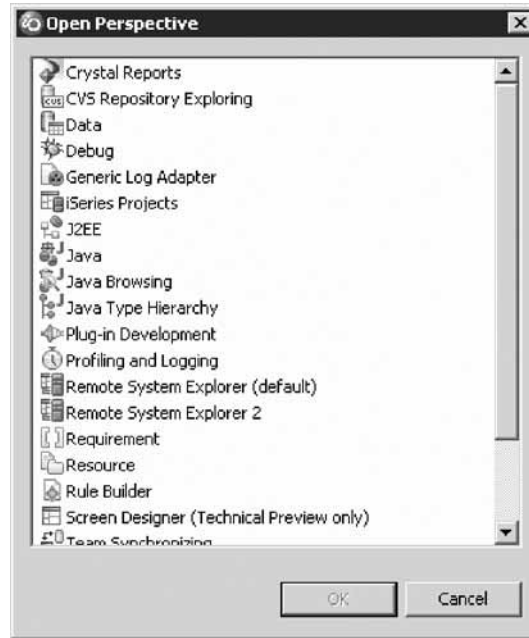


2. Type a new name for the perspective into the **Name** field.



3. Click **OK**.

The name of the new perspective is added to the Select Perspective window **Window > Open Perspective** menu then select **Other**.



You have saved the perspective.

**Tip:** You can also make the new perspective the default by selecting **Window > Preferences**, expanding **Workbench** and then clicking **Perspectives**. You then select the new perspective and make it the default by clicking **Make Default**. The next time you open the workbench, this will be your default perspective.

### Lesson checkpoint

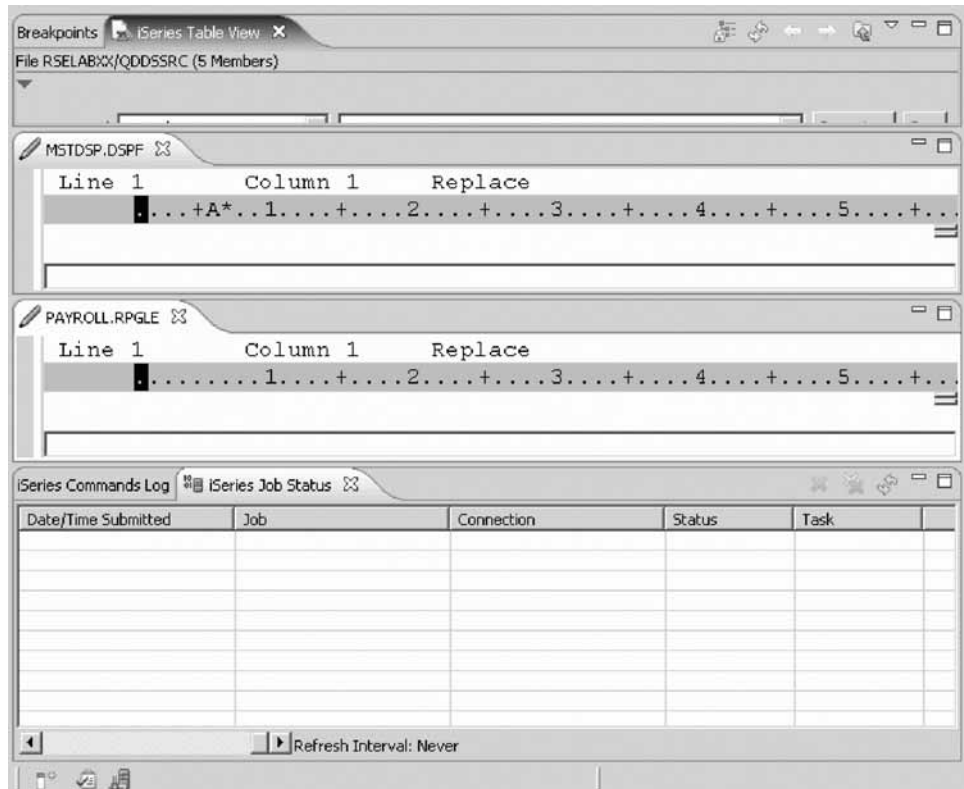
You learned the following:

- About saving a perspective
- How to save a perspective

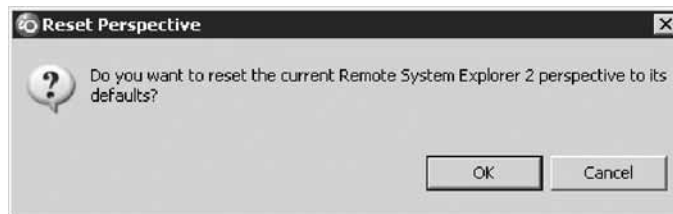
### Resetting the perspective

If you have modified a perspective and don't like the changes that you have made you can reset the perspective to its original layout.

1. Move some views around in the current perspective.



2. Click **Window > Reset Perspective**.  
The following message dialog appears.



3. Click **OK**.  
The perspective returns to its original layout.
4. Click **Window > Open Perspective** and select **Remote System Explorer**. This opens the default Remote System Explorer perspective, ready for the next exercise.

You have reset the perspective.

### Lesson checkpoint

You learned the following:

- About resetting the perspective
- How to reset the perspective

## Expanding files and folders

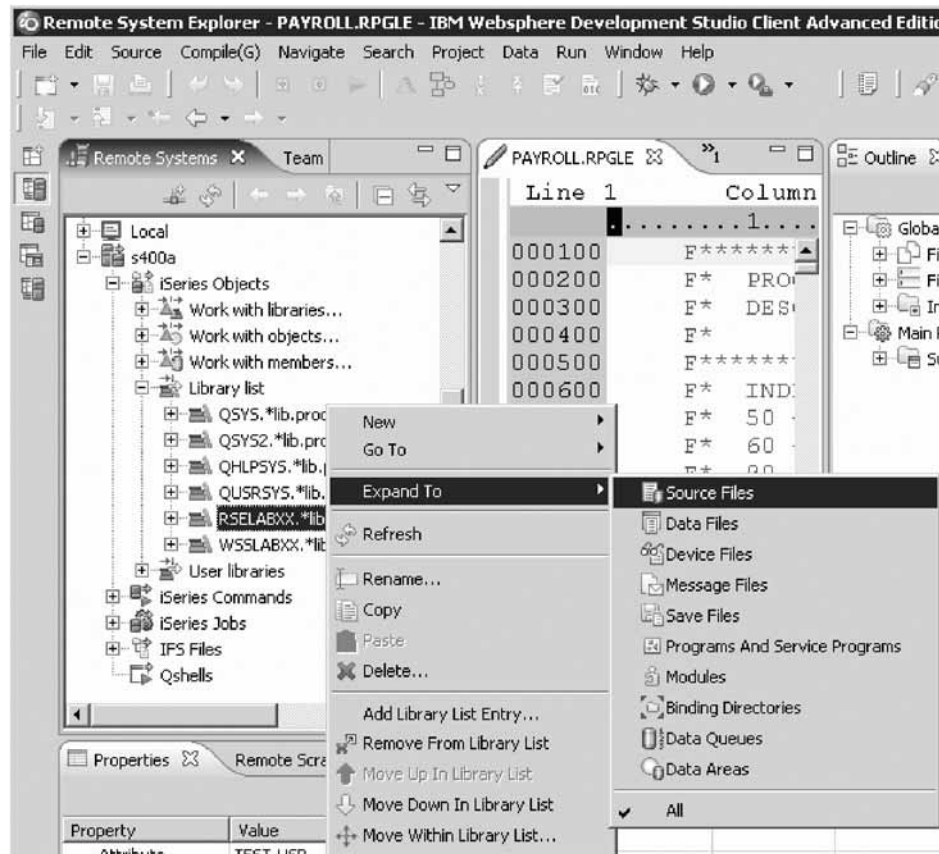
Typically you start using the Remote System Explorer by just expanding libraries, then objects, then source files and then members. You can also expand the Home

directory to see folders in \home in IFS. But sometimes this produces lists that are too big. You really want to keep lists small, to a few hundred at most.

One very quick way to reduce the amount of items in a list is to use the Expand To object for libraries. It allows you to expand a library to see only objects of a particular type. This subsetting remains in effect forever, even when you expand with the plus sign, until you subsequently choose All or any other expand-to criteria.

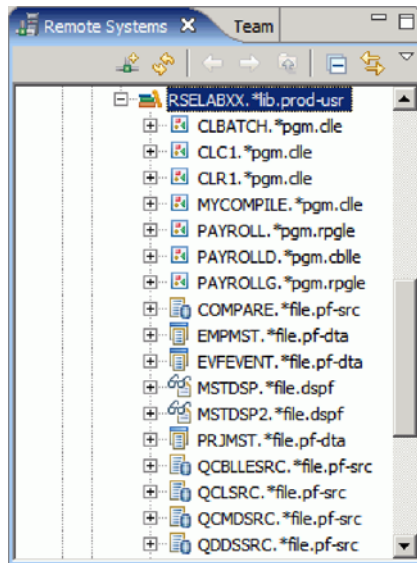
To expand files and folders:

1. In the Remote Systems view, right-click library RSELABxx.
2. Click **Expand To > Source Files** from the pop-up menu.

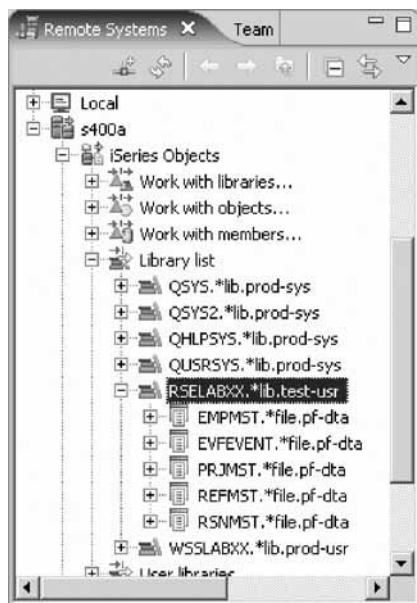


All the source files display.

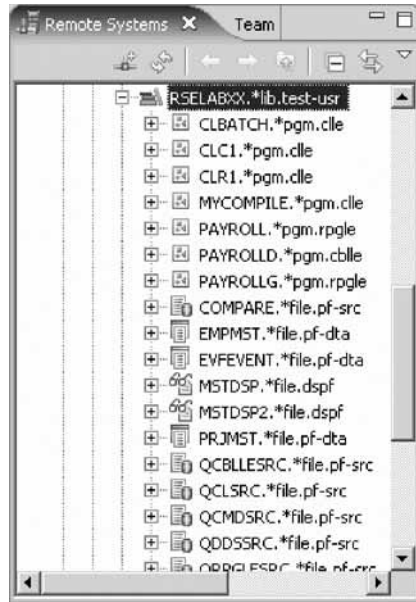




3. Right-click RSELABxx and click **Expand To > Data Files** on the pop-up menu. All the data files display.



4. Right-click RSELABxx and click **Expand To > All** on the pop-up menu. All the files display.



You have learned how to expand source files, data files and all files in the RSELABxx library.

### Lesson checkpoint

You learned the following:

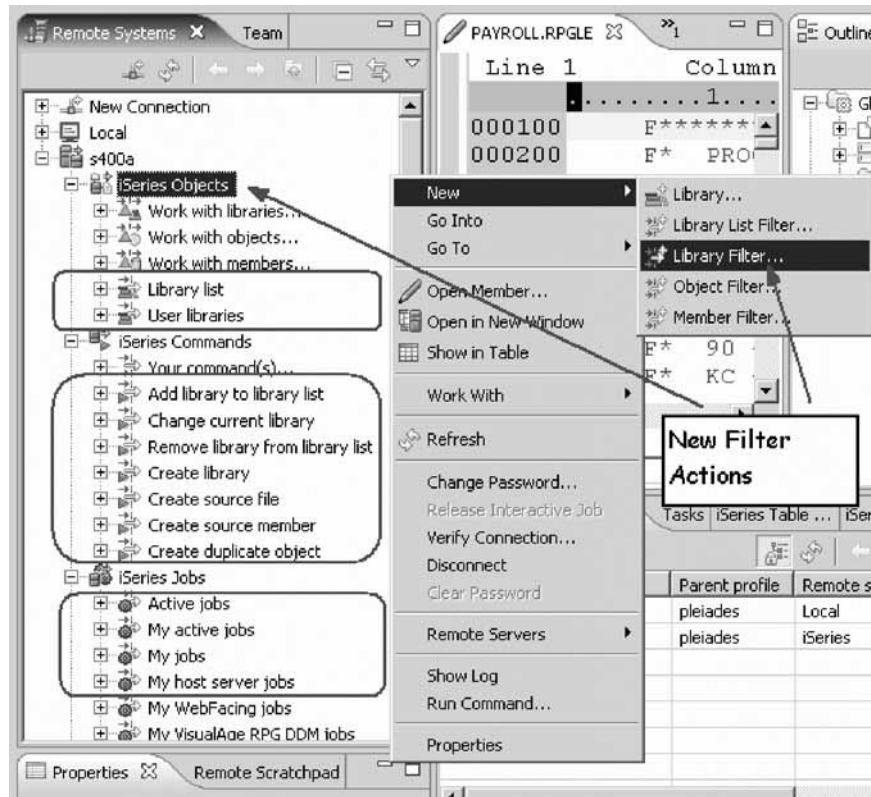
- About expanding files
- How to expand files

## Introducing filters

Eventually you will find the need to see a subset list. That is what filters offer and the Remote System Explorer has extensive filter support. On each subsystem you can create filters. In the iSeries Objects subsystem you can create a library filter, an object filter and a member filter. In the iSeries Commands subsystem you can create a command set filter. In the iSeries Jobs subsystem you can create a job filter. In the IFS Files subsystem you can create a filter.

You can also use the Work with libraries, Work with objects and Work with members prompts under iSeries Objects to create filters.

There are several predefined filters as shown below under the Remote Systems view.



You have learned what predefined filters exist in each subsystem and how to create a filter.

### Lesson checkpoint

You learned the following:

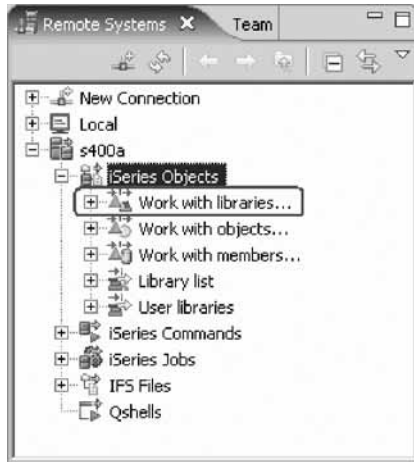
- About predefined filters
- How to create a filter

## Creating a library filter

In the Remote System Explorer perspective, you now need to get to the iSeries objects you want to work with.

In the previous modules you have worked with the Library list. Now you will create your own library filter. Library filters list a set of libraries from your iSeries system in the Remote Systems view. But first let's understand what filters are all about.

Filters allow you to easily organize elements within your system. You use the filter function to list iSeries native file system objects (such as libraries, objects, or members).



To create a library filter:

1. In the Remote Systems view expand the connection that connects to your iSeries system if its not already expanded.
2. Expand **iSeries Objects** if its not already expanded.
3. Expand **Work with Libraries**. (You can also right-click **iSeries Objects** and click **New > Library Filter** on the pop-up menu).

Expanding Work with libraries corresponds to the WRKLIBPDM command, plus creates the filter in the Remote Systems view.

The Create a new iSeries library filter page opens:



You are going to create a filter to specify the libraries you want to work with, so they will show in iSeries Objects. You want to create a filter that shows all libraries on the iSeries with the name **RSExxxxxxx** and **VARxxxxxxx**, xxx being any character.

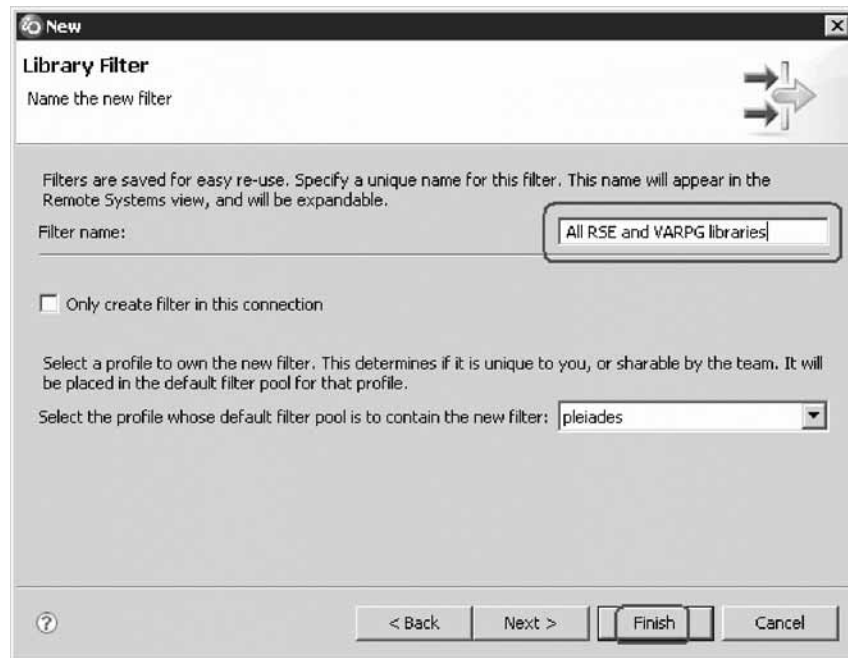
**Note:** You may need to select different libraries that appear on your system if libraries with the above names do not exist.

You specify the first filter string that selects the libraries starting with RSE.

4. Type RSE\* into the **Library** field, using the \* wild card character.

5. Click **Next**.

The Name the new filter page opens.



**Tip:** You can choose between creating the filter for all connections or for this specific one only.

6. In the **Filter name** field, type All RSE and VARPG libraries.

You give your filters a name because the Remote System Explorer saves them for future use, unlike PDM, which does not save filters.

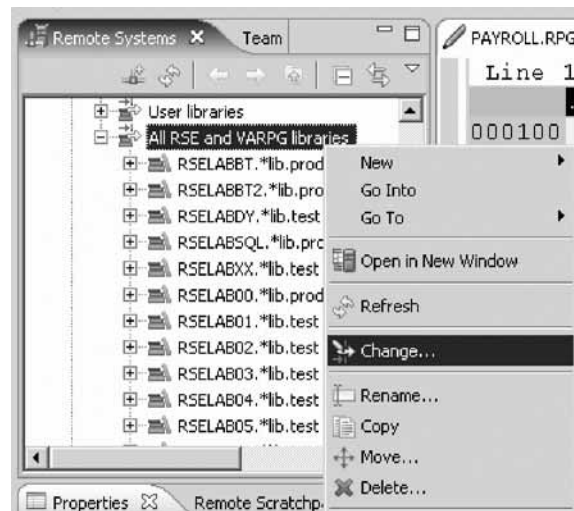
7. Click **Finish**.

Back in the Remote Systems view under iSeries Objects you will see the new filter. Expand it to see the list of all RSE\* libraries.

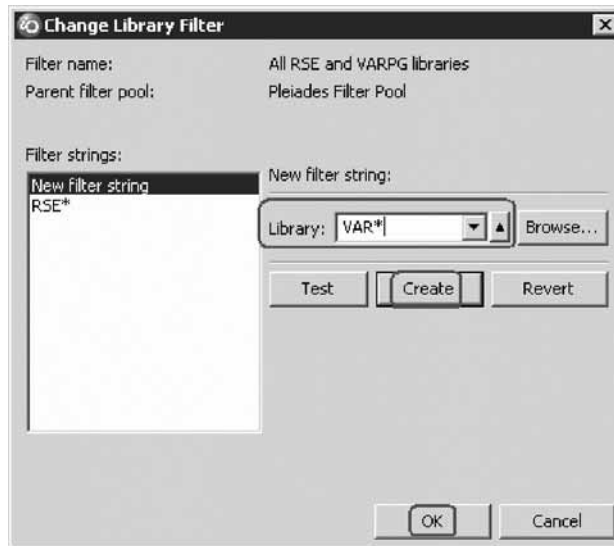
Now you need to add the VARPG libraries.

8. To change the library filter:

- a. Right-click the filter **All RSE and VARPG libraries** and click **Change**.



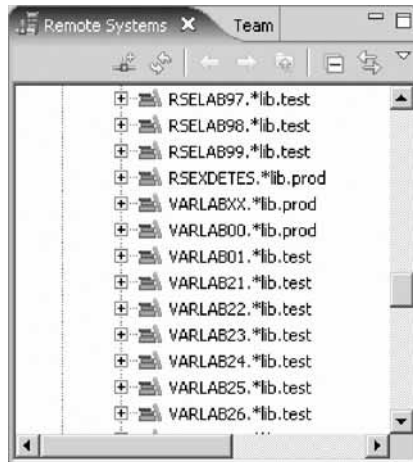
The **Change Library Filter** window opens.



- b. Select **New filter string** from the **Filter strings** list.
- c. In the **Library** field, type VAR\*.
- d. Click **Create**.
- e. Click **OK**.

The **VAR\*** filter string is added to the list.

You are now back in the Remote Systems view.



You will see the list expanded to include your filter. Now you can work with the libraries directly and can drill down to the object you want to work with.

You have created a filter to show a specific iSeries library and changed that filter to add more iSeries libraries.

### Lesson checkpoint

You learned the following:

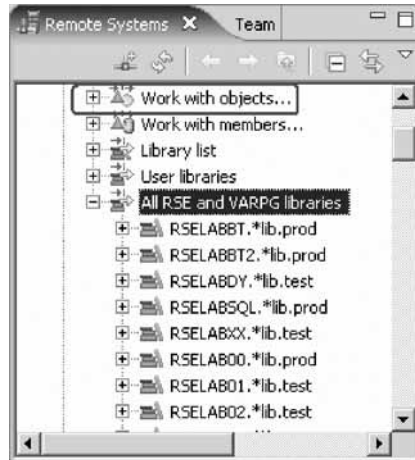
- About library filters
- How to create and change a library filter

## Creating an object filter

Now create an object filter. Object filters list a set of objects from your iSeries host in the Remote Systems view.

To create an object filter:

1. In the Remote Systems view, expand your connection and then expand **iSeries Objects** if not already expanded.

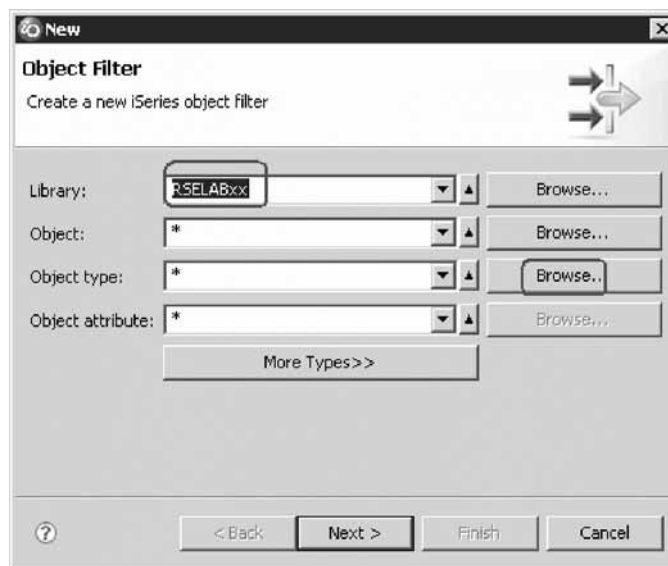


2. Expand **Work with objects**. You can also right-click **iSeries Objects** and click **New > Object filter** on the pop-up menu.

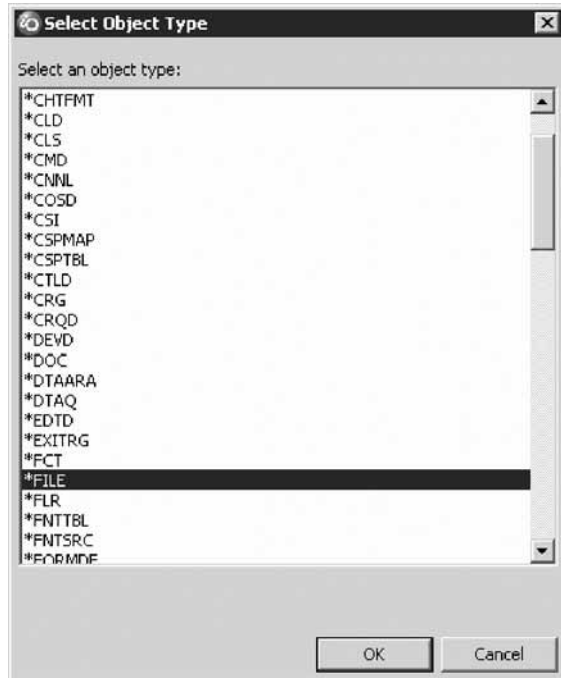
**Tip:** Expanding Work with objects corresponds to the WRKOBJPDM command.

The Create a new iSeries object filter page opens:

Now create a filter to show all your source files in your RSELABxx library.

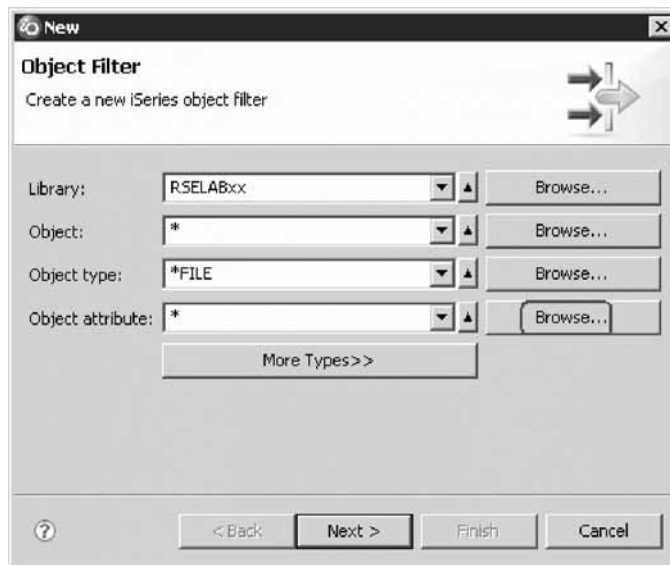


3. In the **Library** field, type RSELABxx.
4. Click **Browse** beside the **Object type** field.  
The Select Object Type window opens.



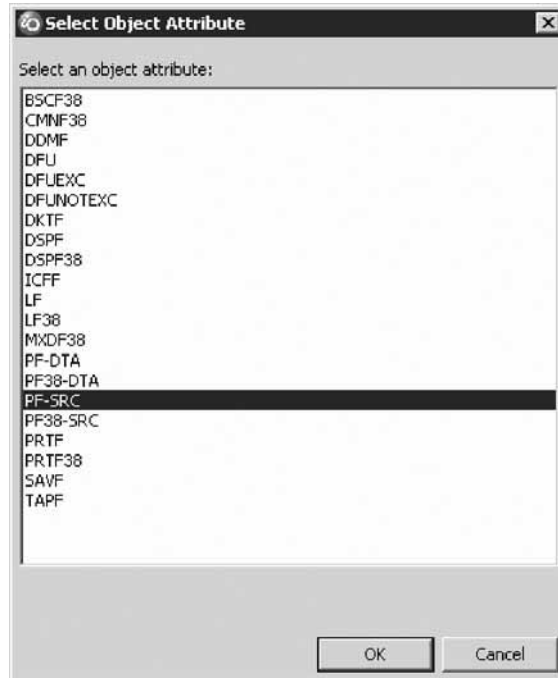
5. Select **\*FILE** under the **Select an object type** list.
6. Click **OK**.

The Create a new iSeries object filter page displays with the object type updated.

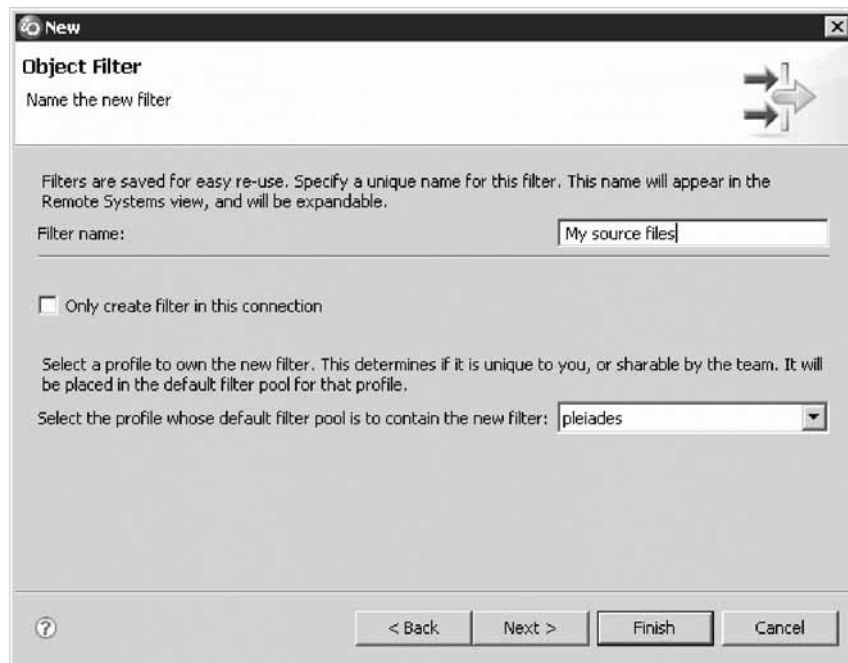


7. Click **Browse** beside the **Object attribute** field.  
The Select Object Attribute window opens.
8. Select **PF-SRC** from the **Select an object attribute** list.
9. Click **OK**.

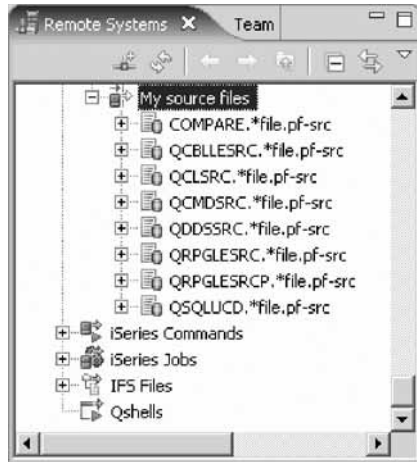




10. Click **Next**.  
The Name the new filter page opens.



11. In the **Filter name** field, type My source files.
12. Click **Finish**.  
The new object filter displays in the Remote Systems view under iSeries Objects:



**Note:** If you end up with too many filters, you can create filter pools. They allow you to group filters. You will learn about filter pools later.

Now you know how to create filters and tailor your development environment. Filters can also be specified for non iSeries servers and your local system.

Now you can work with the objects you have in your Remote Systems view like you worked in PDM with a subset of libraries, objects, or members.

Let's assume you want to edit the member PAYROLL in the source file QRPGLSRC using this object filter.

13. To edit a member from your own object filter:
  - a. Expand QCBLLSRC.
  - b. Right-click member PAYROLLC.
  - c. Click **Open With > Remote Systems LPEX Editor** on the pop-up menu. This will download the source member and open the editor with this member. After you have edited the member you could save it and then compile it from the Remote Systems view by using the pop-up menu options on this member. You can also create your own actions in addition to the default actions. You will learn about creating user actions later.

You have created a filter to show all the source files in your library and accessed members to edit from your filter.

## Lesson checkpoint

You learned the following:

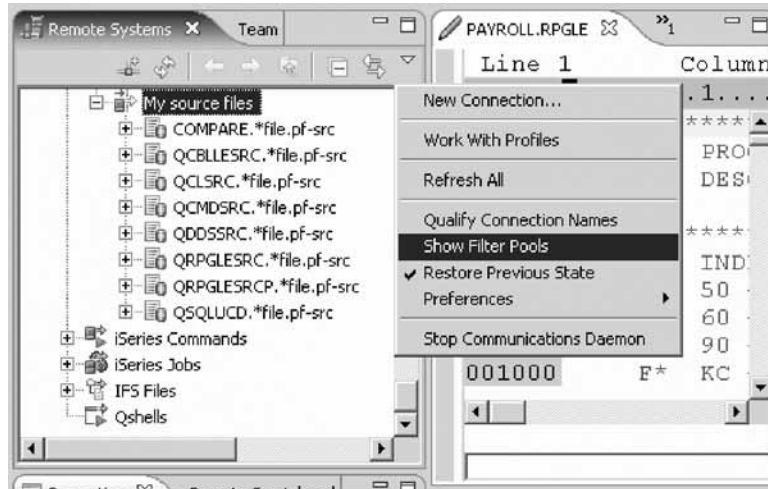
- About object filters
- How to create an object filter

## Showing Filter Pools

If you have been using the Remote System Explorer for some time, your workspace might contain too many filters to navigate easily. Or, you might just want to keep groups of filters separate if, for example, you need to represent two distinct iSeries environments in the Remote System Explorer, regardless of how many filters you have. In either case, you can group filters into filter pools. Without filter pools, all of your filters appear together in all connections. When

you create filter pools, however, any filter you create within that filter pool is distinct to that connection, and will not appear in any other connections.

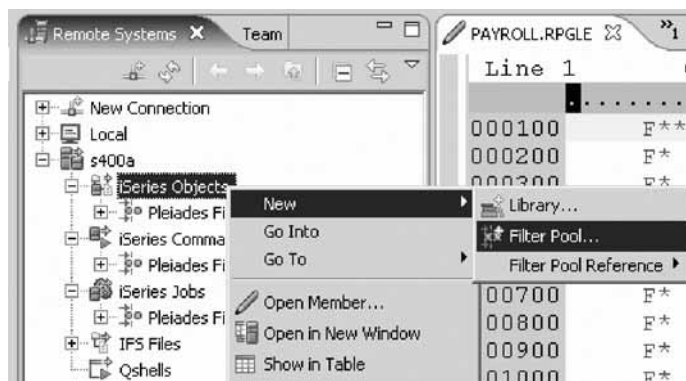
1. Create a connection to the same host. (Expand New Connection then iSeries). Give your new connection the name s400b.
2. To illustrate the use of filter pools, Click the menu button on the toolbar for the Remote Systems view, and select **Show Filter Pools**.



3. Under iSeries Objects you can now see your filters listed under Connection name Filter Pool.



4. Right-click iSeries Objects and select **New > Filter Pool**.

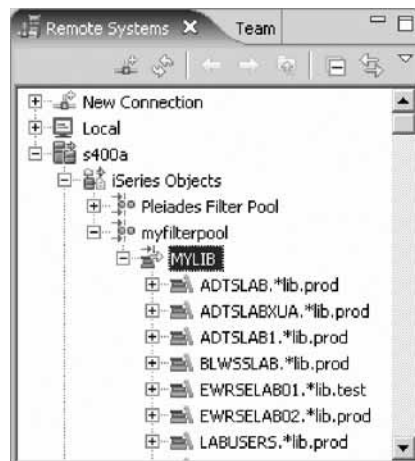


5. Enter a pool name and click **Finish**. (You do not need to change your profile selection.) Your new filter pool displays underneath your connection.



The filter pool is added only to the connection from which it was created.

6. Right-click your new filter pool and select **New > Library filter**.
7. Complete the wizards as you did before. Use \*LAB\* as the generic library name for the library filter. Give your filter any name you like. Click **Finish**.
8. When you are finished, you can see your new library filter displayed underneath the new filter pool.



If you decide not to work with filter pools anymore, click the menu button on the toolbar for the Remote Systems view, and select **Show Filter Pools** again to clear the check mark.

For each filter pool, you can right-click and select from a number of actions. For example, you can rename, copy, move or delete a filter pool.

You have learned how to group filters into filter pools and how to create a new filter pool.

## Lesson checkpoint

You learned the following:

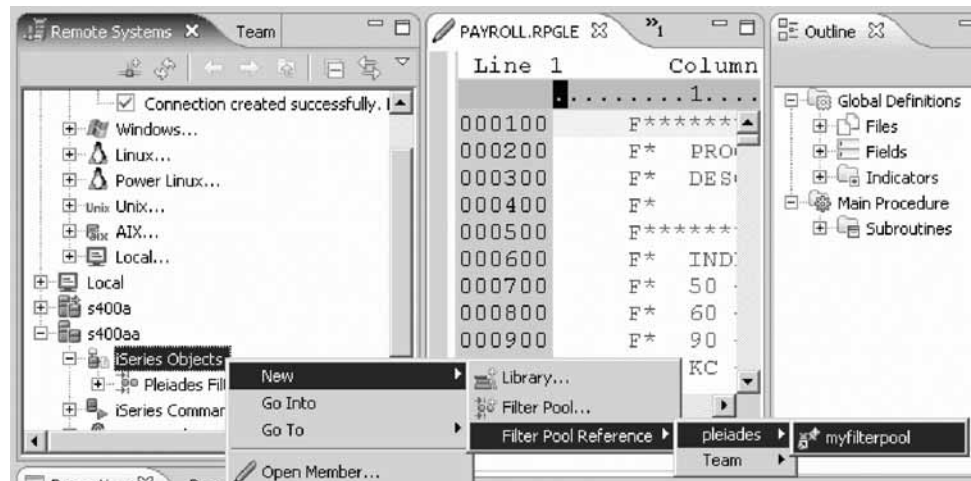
- About filter pools
- How to create a new filter pool

## Sharing filter pools

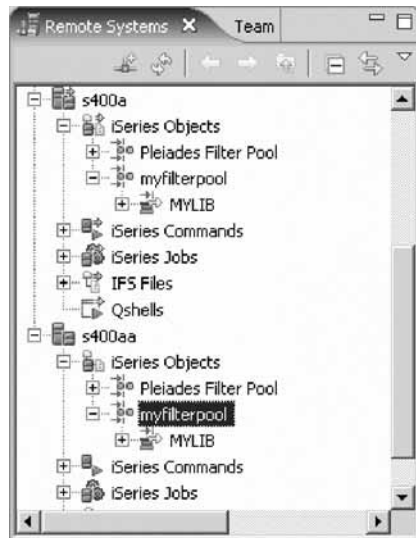
You can share filter pools among many connections through the use of a filter pool reference. A filter pool reference is a mechanism that displays a filter pool from one connection in any other connection, so that when you make a change to the original filter pool, your change is reflected in your filter pool reference. Before you create a filter pool reference, ensure that you have already completed the following: You have defined more than one connection to the same iSeries server  
You have defined more than one filter pool  
You have enabled Show Filter Pools from the Remote Systems view toolbar .

To use filter pool references:

1. Make sure you have another connection established.
2. In the Remote Systems view, expand the connection where you want to display a filter pool that exists in another connection.
3. Right-click iSeries Objects and select **New > Filter Pool Reference > your profile > pool name**.

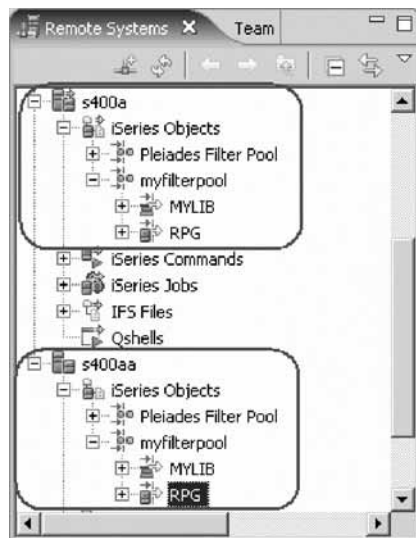


4. Look under iSeries Objects again and you will see the filter pool reference.

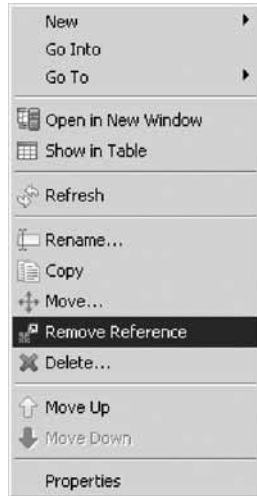


Next, you make a change to the filter pool in order to see that change also occur in the filter pool reference.

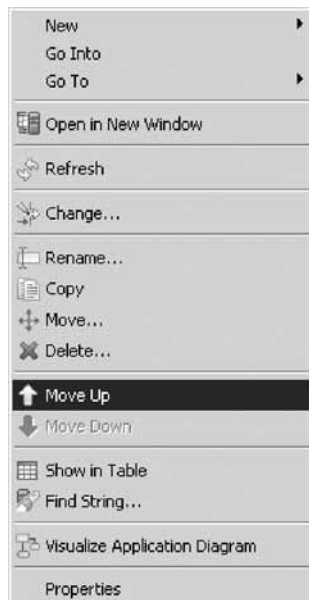
5. Add new object filter called RPG. Right-click your new filter pool and select **New > Object filter**. Complete the wizards. When you are finished you will see the referenced filter is available in both connections.



6. To delete a filter pool reference, right-click it and select **Remove reference**.



7. You can also move your filter pools up and down with the pop-up menu.



You have learned how to share filters.

### Lesson checkpoint

You learned the following:

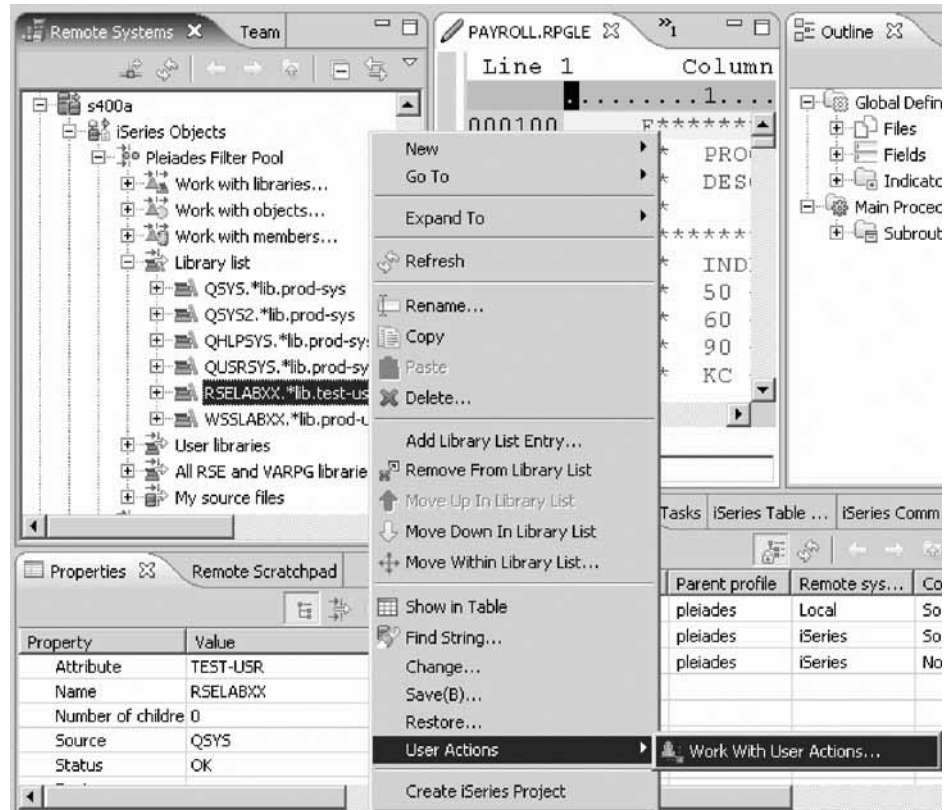
- About sharing filters
- How to share filters

### Creating a user action

In PDM you can create user actions in addition to using the pre-supplied system actions. In Remote System Explorer you can do the same. You define user actions through the Work With User Actions window. User actions can be defined for iSeries libraries, objects, members and jobs as well as folders and files in any remote UNIX, Windows, Linux, Local, or IFS system.

To open the Work with User Actions wizard:

1. Expand your iSeries connection and expand **iSeries Objects** if not already expanded.



2. Expand the Library list filter if not already expanded.
3. Right-click RSELABxx.
4. Click **User Actions > Work with User Actions** on the pop-up menu. The Work with User Actions window opens.

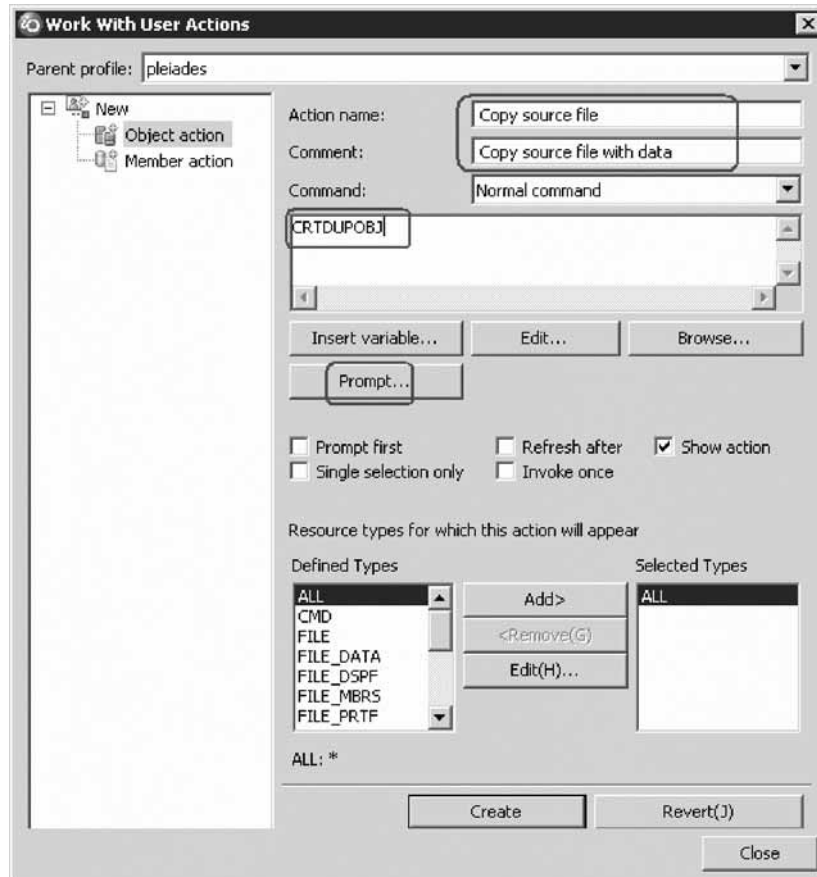


5. In the right pane of the Work with User Actions window, expand **New** in the list, if it is not expanded already.

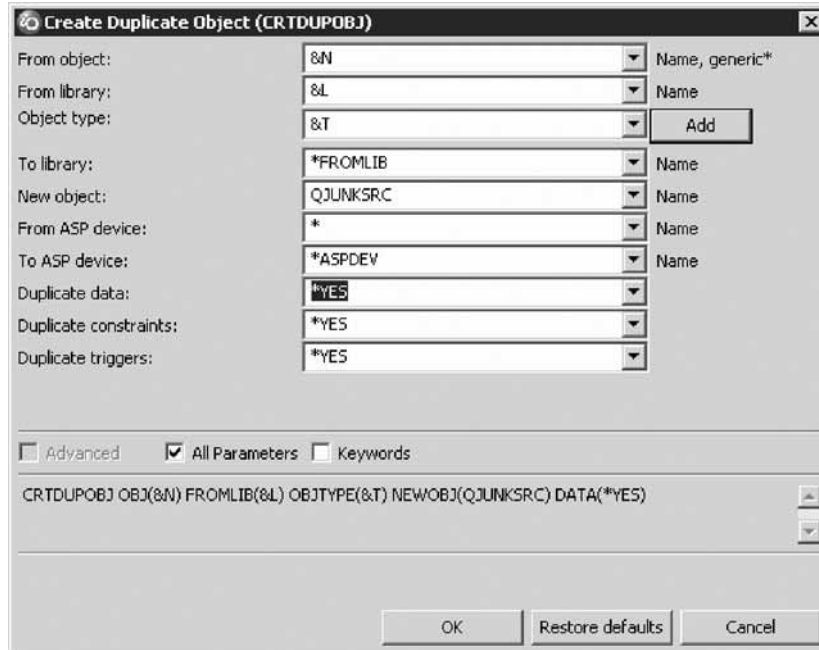


6. Select **Object action**.

You want to create a user action that copies a source file with data to a new source file called QJUNKSRC in the same library.



7. In the **Action name** field, type Copy source file for the user action name.
8. In the **Comment** field, type Copy source files with data.
9. In the **Command** field, type CRTDUPOBJ for the command to execute.
10. Click **Prompt** to open the command prompter for this command.



This is the command you will be running:

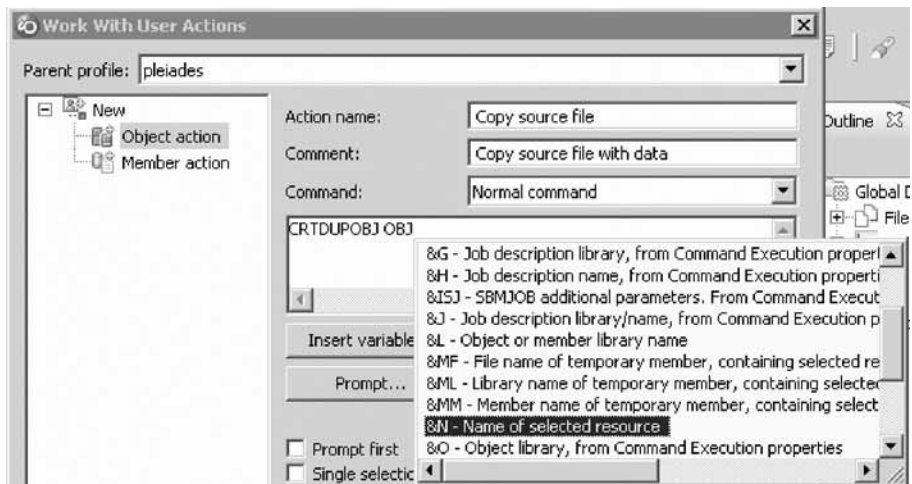
```
CRTDUPOBJ OBJ(&N) FROMLIB(&L) OBJTYPE(&T) NEWOBJ(QJUNKSRC) DATA(*YES)
```

11. To specify user action parameters:

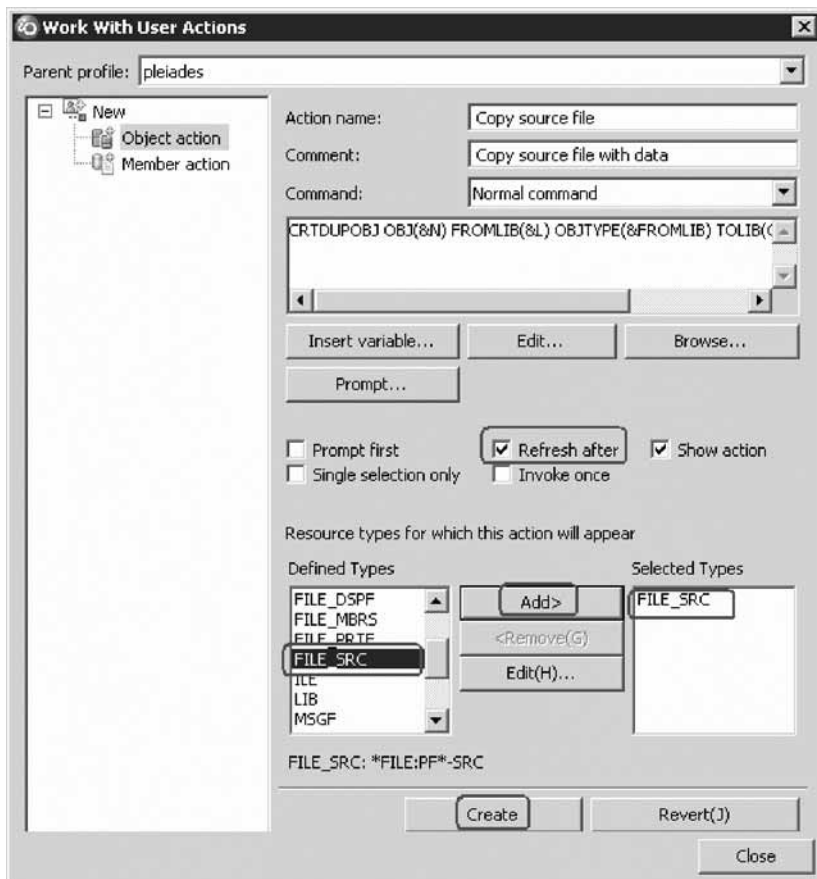
- a. In the **From Object** field, type &N to indicate to use the name of the selected object in the Remote Systems view.
- b. In the **From Library** field, type &L to pick up the library name from the selected object.
- c. In the **Object Type** field, type &T to pick up the object type from the selected object.
- d. In the **New Object** field, type QJUNKSRC.
- e. Select the **All parameters** check box to see the additional Duplicate data parameter.  
Now the Duplicate data parameter is also shown on the prompt window.
- f. Select **\*YES** from the **Duplicate data** list.
- g. Click **OK**.

You return to the Work with User Actions window.

**Tip:** You can use the **Prompt** button to enter the variables or you can type the command directly and when you type &, you see a pop-up selection list, or you can use **Ctrl+Space** or press the **Insert Variables** button. From the list, you can then double click to insert the selected variable, at the cursor position.



- h. Select the **Refresh after** check box, so that the Remote Systems view gets refreshed after the action has been run.



**Tip:** Clicking the **Insert variable** push button displays a list of valid replacement variables with the explanation of what they do.

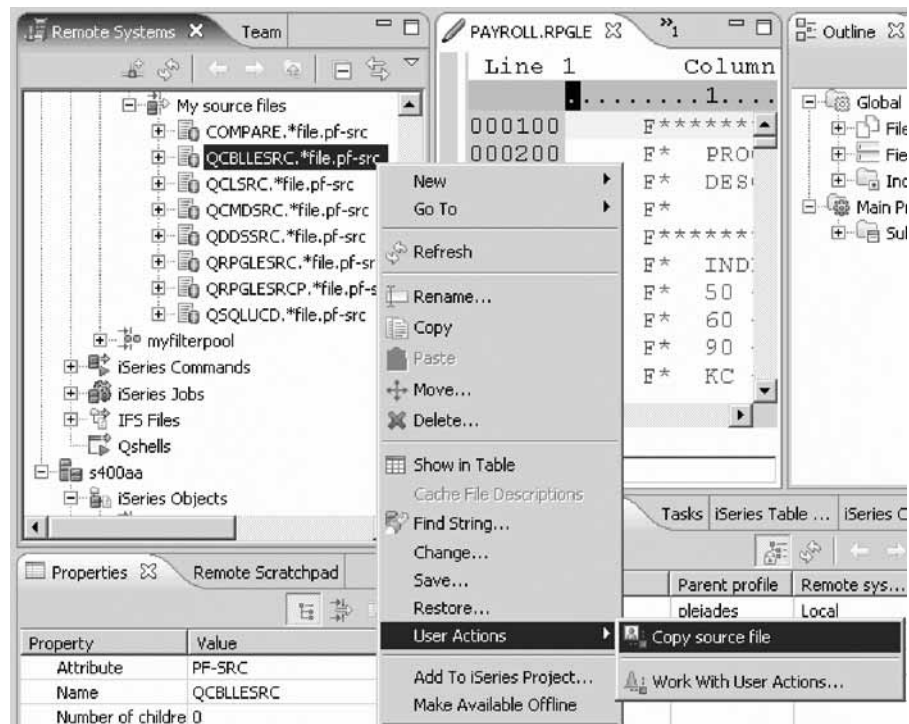
This user action is only valid for Source physical files. You need to specify this restriction so this user action will only show in pop-up menus when you right-click on a source physical file.

12. To specify a restriction on a user action:
  - a. Under the **Defined Types** list box, click **FILE\_SRC**.

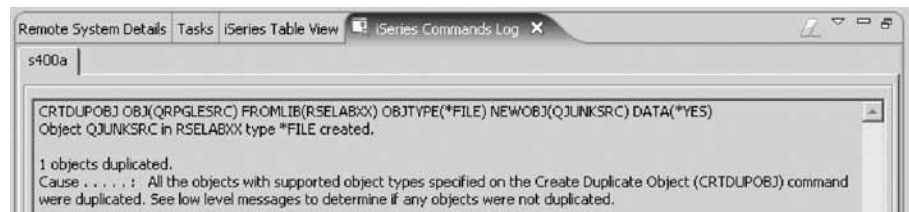
- b. Click **Add** beside the **Defined Types** list box.  
FILE\_SRC is now one of the selected types. Actually since you only selected this one it is the only one.
- c. Click **Create** then **Close**.  
Now, only when you right-click on a source file, will this user action appear on the pop-up menu. For any other object type it will not appear. Back in the workbench and the Remote System view, give it a try.

**Tip:** Remember to close all the source members if you opened any earlier.

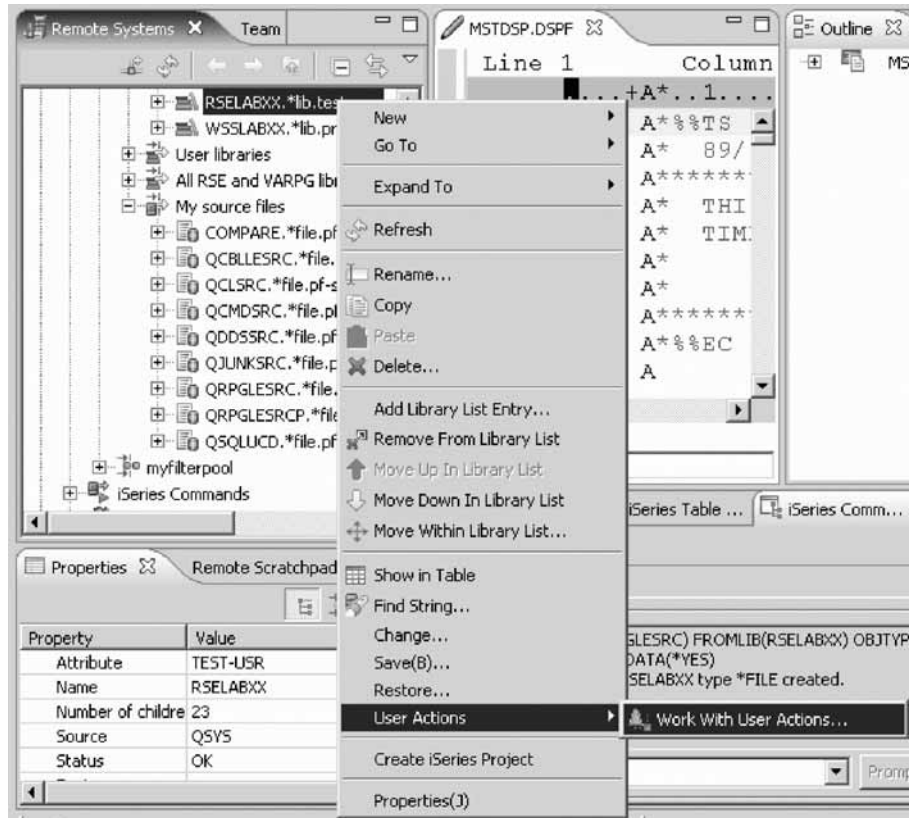
- 13. To try a user action:
  - a. Locate your filter **My source files**.



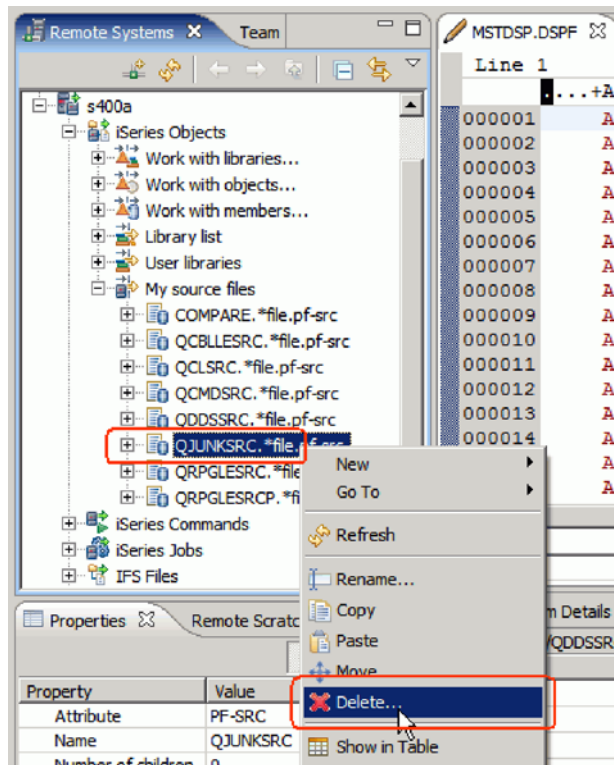
- b. Expand the filter **My source files**, if it is not already expanded.
- c. Right-click the **QCBLLESRC** file.
- d. Click **User Actions > Copy source file** on the pop-up menu.  
The file gets duplicated and the list gets refreshed. Your new source file will show in the list. You can check the messages of the CL commands you are running in the RSE communications server job by looking at the iSeries Commands log view in the bottom right of the workbench.



- e. Try other objects such as **\*pgm** or **\*lib**. Notice that the action that you just created is not there.



- f. To delete the source file QJUNKSRC that you just created, right-click QJUNKSRC.



- g. Click **Delete** on the pop-up menu.

The Delete Confirmation dialog opens.

h. Click **Delete**.

You have created a user action that copies a source file with data to a new source file, specified user action parameters, specified restrictions on the user action and tried the user action.

## Lesson checkpoint

You learned the following:

- About user actions
- How to create a user action

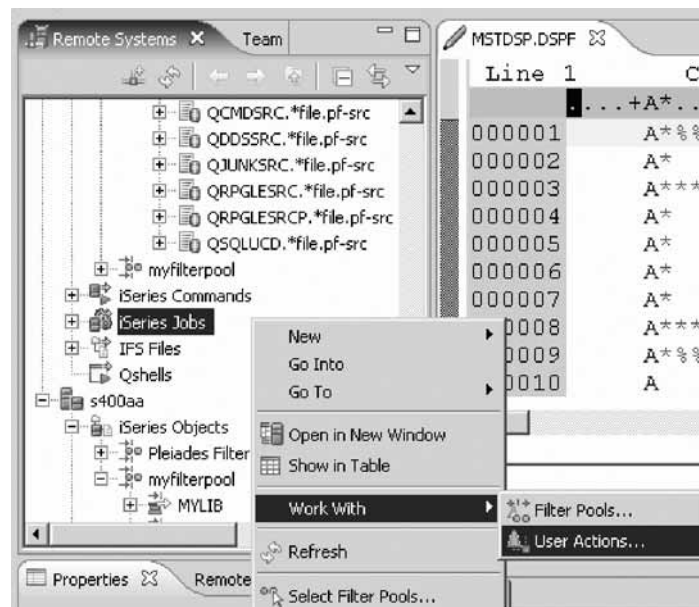
## Creating user actions for jobs

You can also create user actions for Jobs, which will appear in the User Actions popup menu for jobs in the iSeries Jobs subsystem under a connection. The substitution variables include variables for the selected job's number, user and name.

To create a job action:

1. In the Remote Systems view, expand your iSeries connection, if not already, right-click iSeries Jobs, and select **Work with > User actions**.

You can also right-click on a file in one of your filters, and select **User Actions > Work With User Actions**.

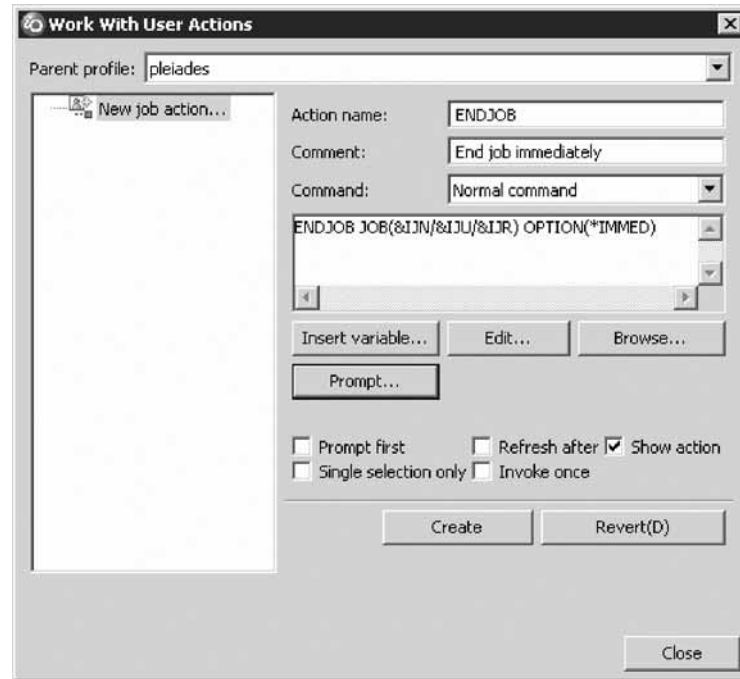


2. Select **New Job action**.
3. Type the text to display in the **Action name** field. For example, ENDJOB. This is a brief label for the action.
4. Type a longer, more descriptive text description for the action in the **Comment** field. For example, End Job Immediately.
5. Type the actual workstation or iSeries command string to run when a user selects this action. For example, ENDJOB JOB(&IJR/&IJU/&IJN) OPTION(\*IMMED).

This command can use action substitution variables when you run the action. These variables are used when defining the command string to run for a particular action. Substitution variables keep you from having to explicitly code command parameter values.

6. Click **Insert variable** to view and select valid variables.

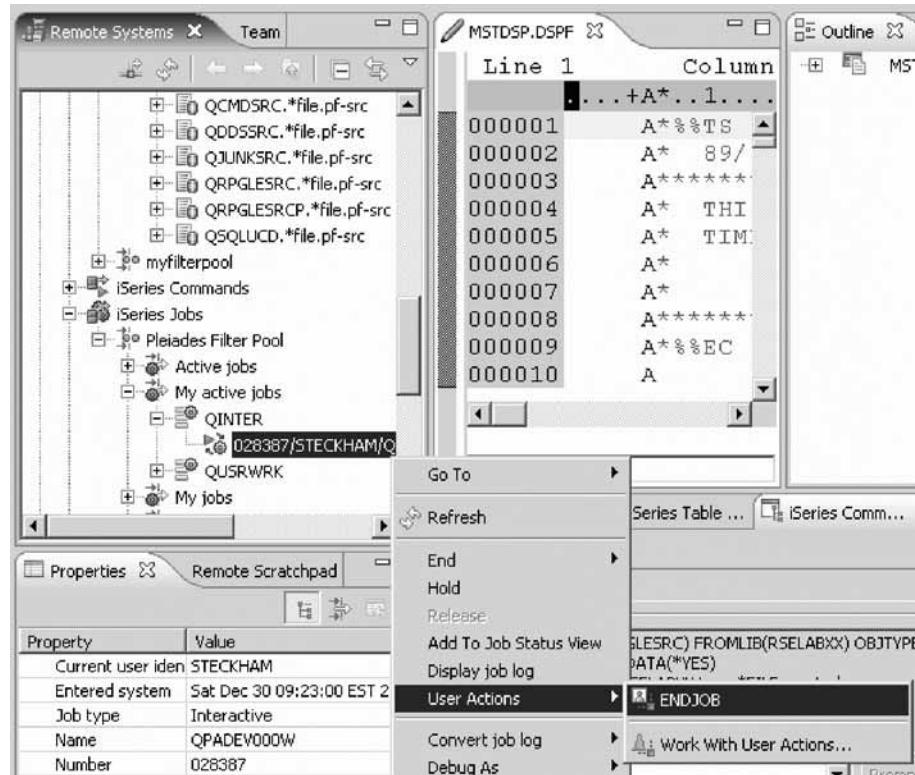
Here is your completed job action:



7. Click **Create** then **Close**.

Now let's try this job action that you just created.

8. Make sure you have a 5250 session. Expand iSeries Jobs. Expand My active jobs. Expand QINTER.
9. Right-click the 5250 job and then select **User actions > ENDJOB** from the pop-up menu.



10. Switch to a 5250 session to verify that the job has ended.

**Tip:** Similar to user actions for objects and jobs, you can also create user actions for IFS.

You have created a user job action.

## Lesson checkpoint

You learned the following:

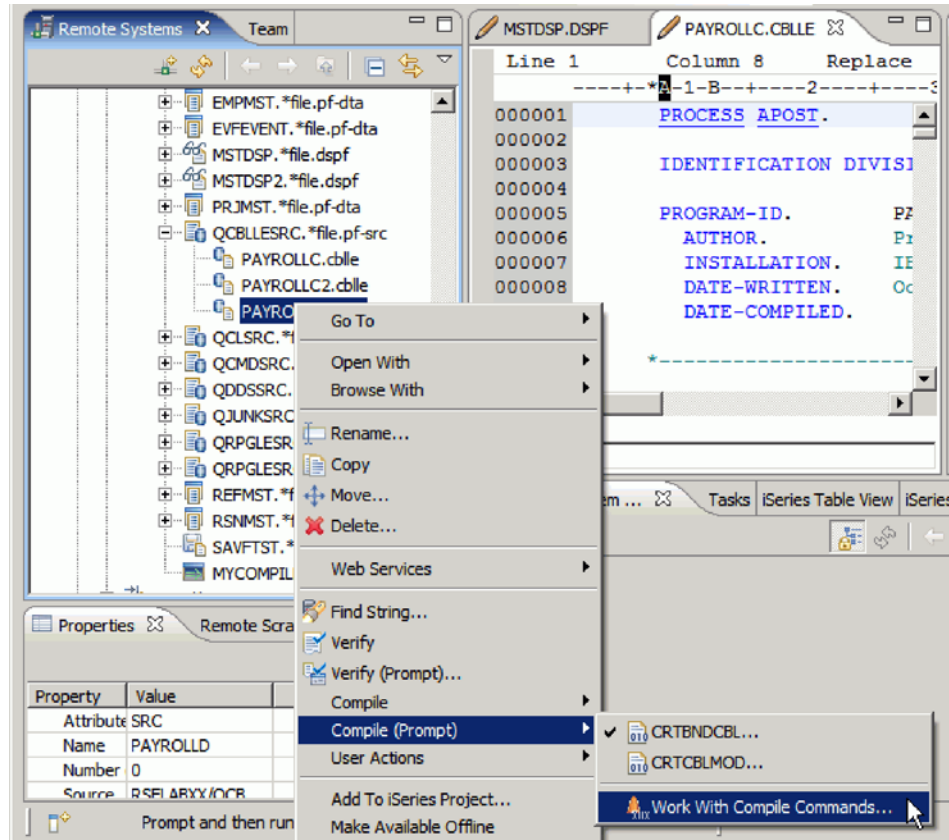
- About user job actions
- How to create a user job action

## Customizing compile commands for iSeries Objects

In addition to user actions, there is specific support for creating compile commands too. You use the Work with Compile Commands window from the iSeries Objects subsystem under an iSeries connection to change IBM or vendor supplied compile commands or your own compile commands.

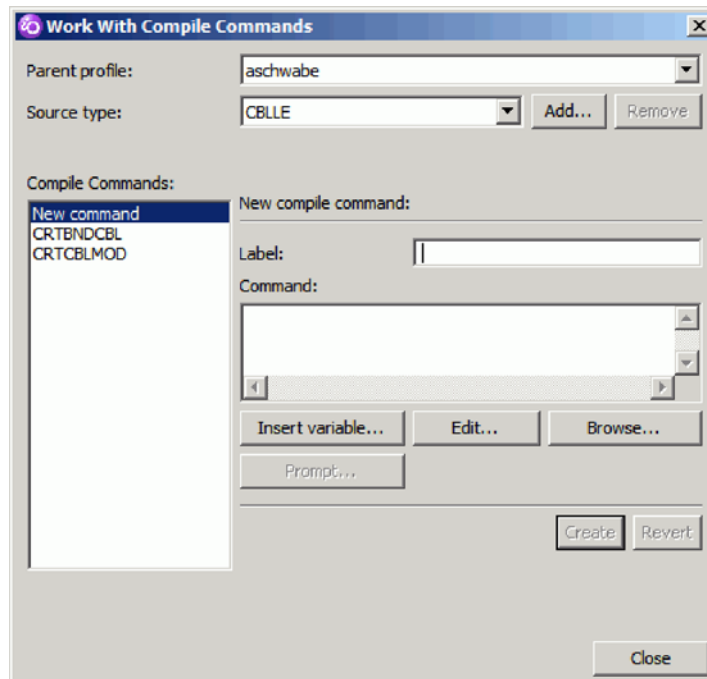
To create your own command:





1. In the Remote Systems view, expand RSELABxx, expand QCBLLSRC and right-click PAYROLLD.cbllc.
2. Click **Compile (Prompt)** > **Work with Compile Commands** in the pop-up menu.

The Work with Compile Commands dialog opens.



**Tip:** You can also work with compile commands from the Compile option (**Compile > Work with Compile Commands**).

3. **New command** is already selected for you in the list of commands.

**Tip:** To edit an existing command, first find it by selecting the member type it applies to (or add a new member type if necessary) at the top of the Work with Compile Commands dialog, and select the command in the list of commands on the left. Edit the command and apply the changes. You can also right-click on a command to delete it, copy and paste it or re-order it. You cannot delete IBM-supplied commands, but after editing them, you can restore them to their shipped value.

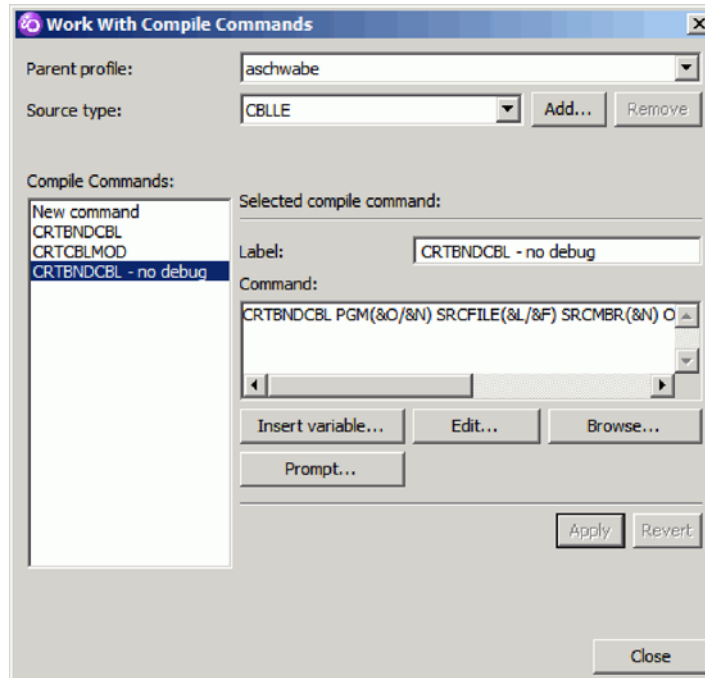
4. In the **Label** field, type CRTBNDCBL - no debug command.
5. In the **Command** field, type CRTBNDCBL command.
6. Click **Prompt**.

The Create Bound COBOL Program (CRTBNDCBL) dialog opens.

Program: > &N Name  
Library: > &O Name  
Source file: > &F Name  
Library: > &L Name  
Source member: > &N Name  
Source stream file: [ ]  
Generation severity level: 30 0-30  
Text 'description': \*SRCMBRTXT Character value...  
Compiler options: > [ ] Add  
\*EVENTF Remove  
Move up  
Move down  
Debug view option: > [ ]  
Debug view: > \*SOURCE  
Compress listing view: > \*NOCOMPRESSDBG  
Replace program: > \*YES  
 Advanced(1)  All Parameters(2)  Keywords(3)  
CRTBNDCBL PGM(&O/&N) SRCFILE(&L/&F) SRCMBR(&N) OPTION(\*EVENTF) DBGVIEW(\*SOURCE) REPLACE(\*YES)  
OK Restore defaults Cancel

7. Change the **Debugging Views** option to **\*NONE**.
8. Click **OK**.

The Work with Compile Commands displays.



9. Click **Create** to create this new command.
10. Click **Close**.
11. Right-click PAYROLLG.rpgle.

The new command is added to the list of available compile commands for members of the type specified in this command. The checkmark appears beside the last used compile command for the selected member's type.

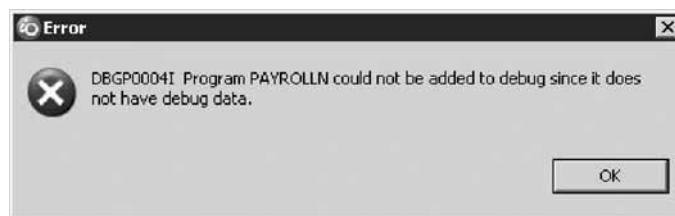
12. Click **Compile > CRTBNDRPG - no debug** on the pop-up menu and change the program name to PAYROLLN.

The member is compiled and the program object is created.

Any errors produced by the compile are displayed in the Error List window, where you can double-click to open the editor and position it at the error.

13. Right-click the program PAYROLLN and click **Debug As > Batch**. If you don't see the program in the list, click the **Refresh** icon in the Remote Systems view.

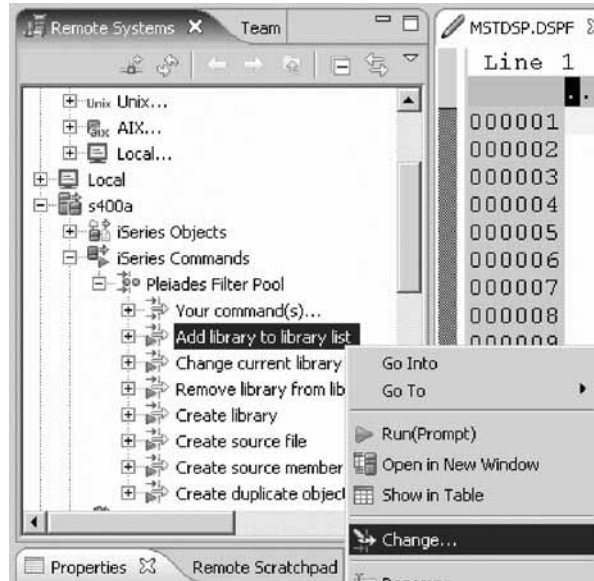
An error message displays indicating that PAYROLLN cannot be added to debug since it does not have debug data.



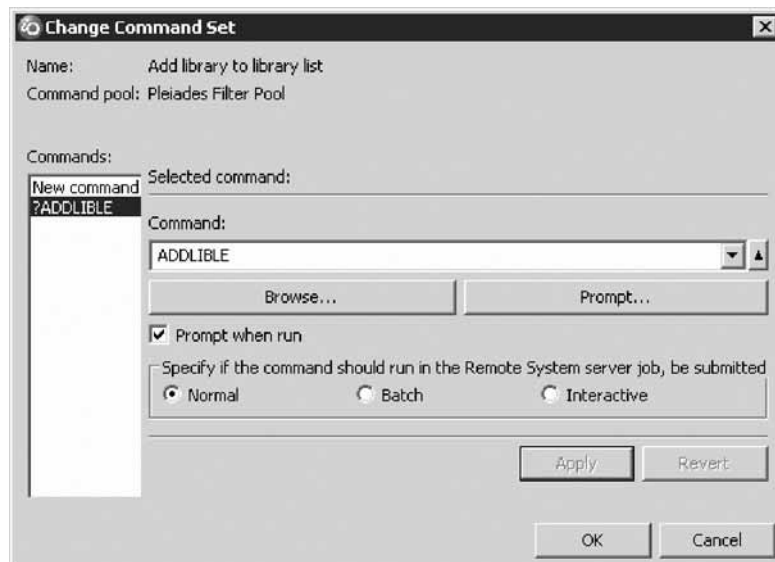
14. **Using predefined commands**

For some of the frequently used commands, the Commands subsystem provides you with a number of predefined command sets. You can use these command sets or create new ones of your own. For example, to run the ADDLIBL command set:

- a. In the Remote Systems view, expand the iSeries Commands subsystem.
- b. Right-click **Add library to library list** and click **Change**.



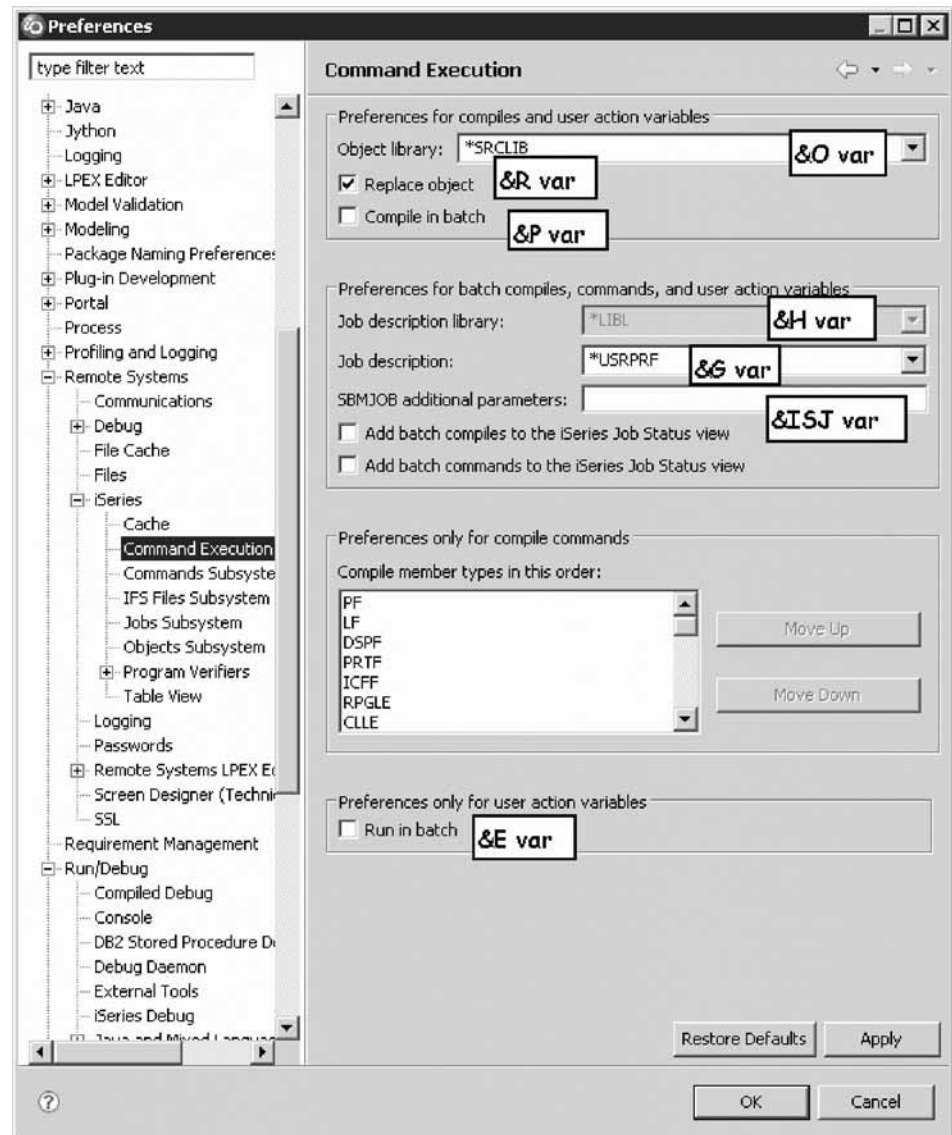
The Change Command Set window opens.



From here you can now modify the existing command or create a new one.

c. Click **Cancel**.

**Tip:** There is a Preferences dialog that has many preferences which effect substitution variables for user actions and compile commands. Click **Window > Preferences** and then expand **Remote Systems**, then expand **iSeries** and select **Command Execution**



You have created a new CRTBNDRPG command and looked at predefined command sets.

## Lesson checkpoint

You learned the following:

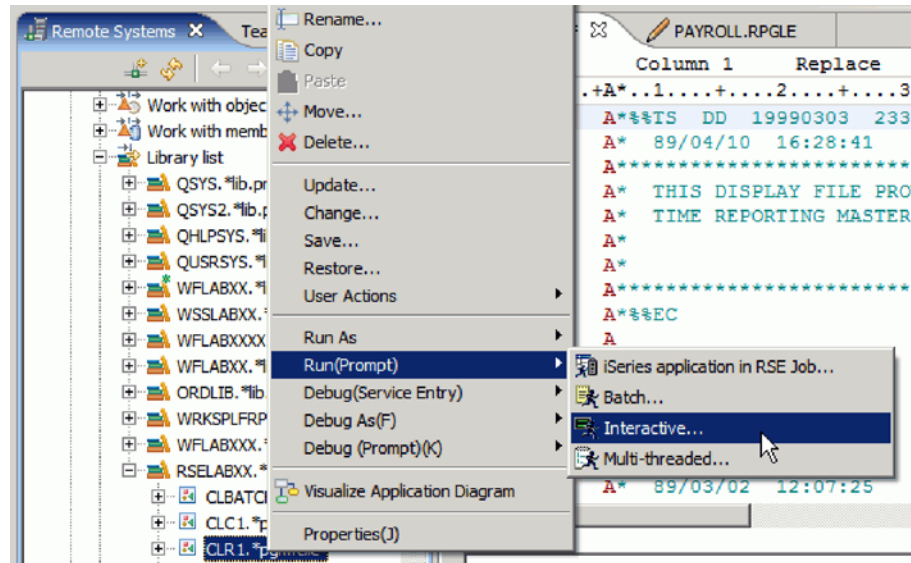
- About predefined command sets
- How to create a new command

## Using Run configurations

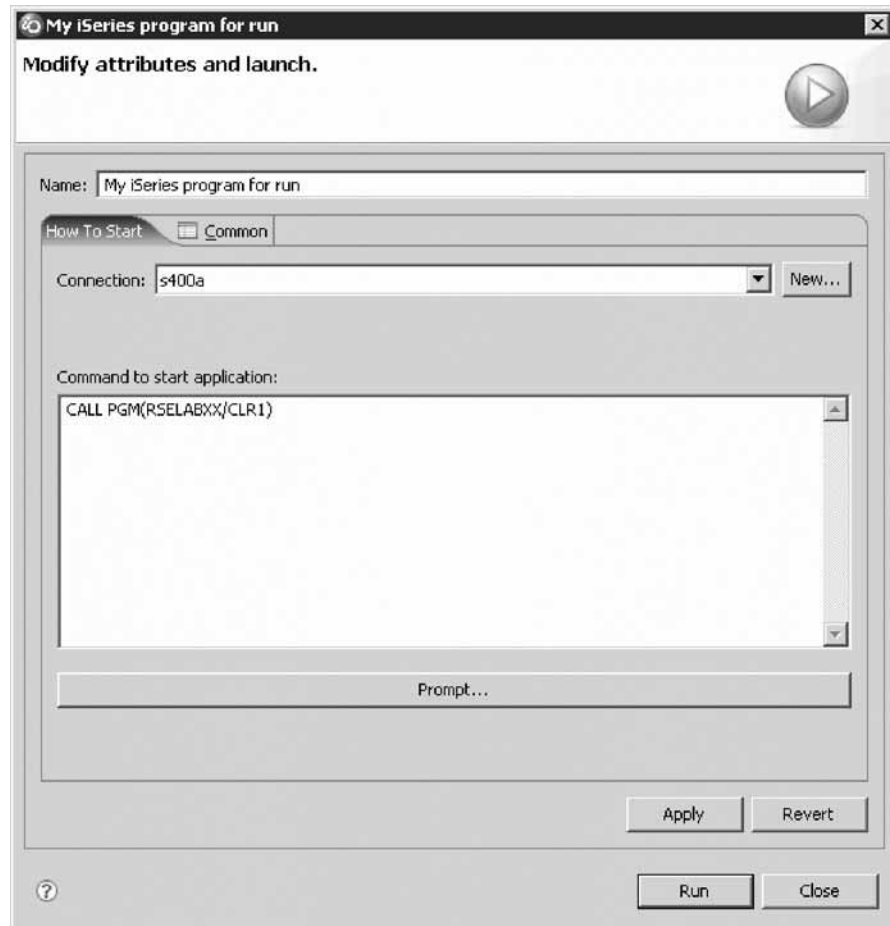
Run configurations are for powerful re-use. If you want to run a program that takes a number of parameters, or is not straightforward to launch, you can predefine this information into a named configuration. Once created, the configuration appears in the configuration list, and can be selected from there. Every configuration defined can be accessed from the pull down menu of the Run tool bar button through the Run option. Clicking the Run tool bar button will run the previous configuration again.

To change an existing run configuration:

1. Right-click CLR1 then **Run (Prompt)** > **Interactive** on the pop-up menu.

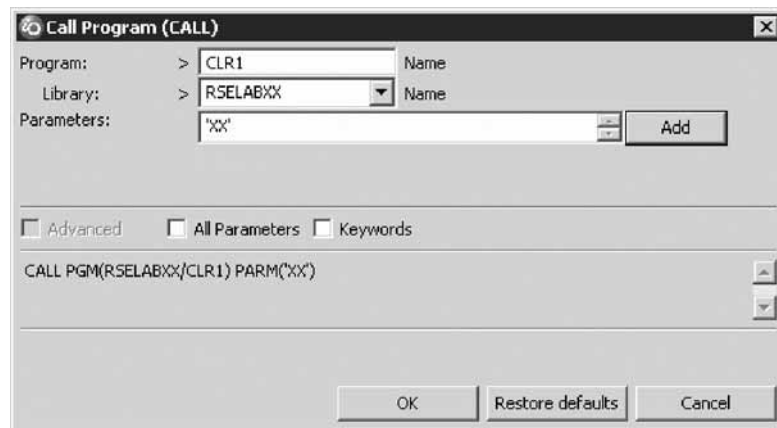


The Run configurations window opens.



**My iSeries program for run** is the default name assigned to a configuration created on the fly when you select **Run** from the pop-up menu of a program. To save a configuration for later use, you would change this default name.

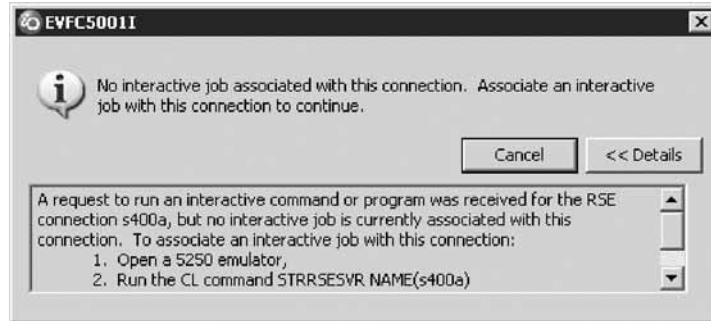
2. Change the name of the configuration to CLR1Run.
3. Click **Prompt**.  
The Call Program (CALL) window opens.



4. In the **Parameters** field, type 'XX', where XX is your workstation number.
5. Click **OK**.  
The complete start command for the program appears.

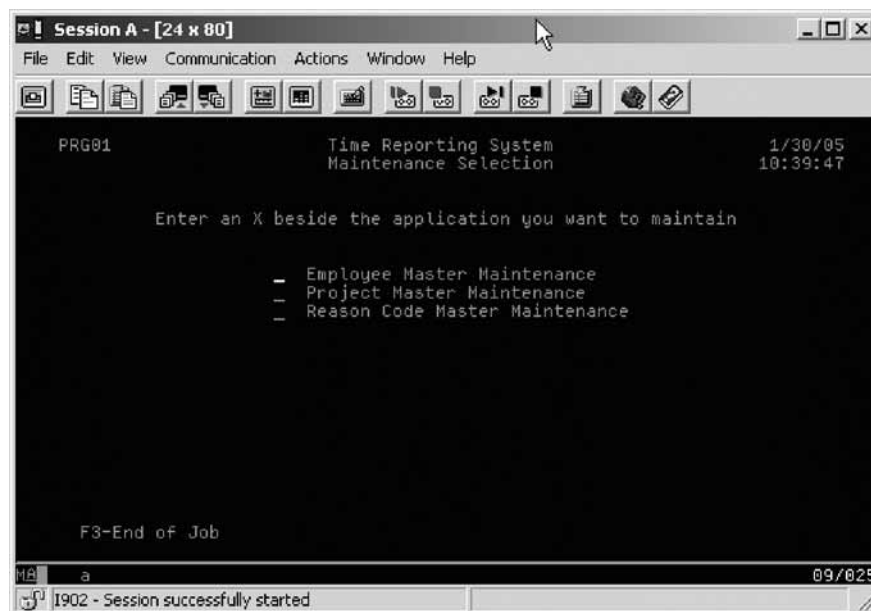


6. Click **Run**.  
The program runs.  
If not, you may see this error message.



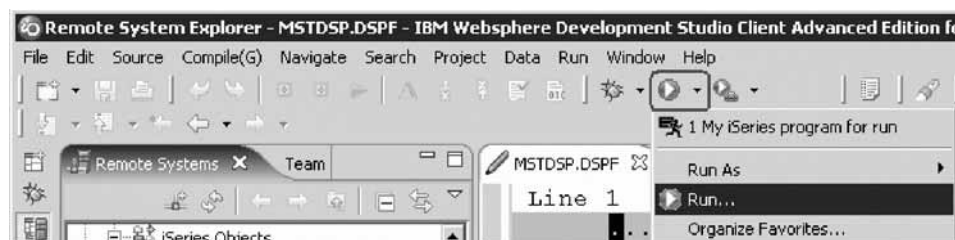
The interactive connection has been shut down in the meantime. Go to your 5250 emulator and restart the interactive connection following the instructions in the message. You don't have to cancel the message. It will be removed as soon as the connection between the interactive connection and the interactive session has been established.

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



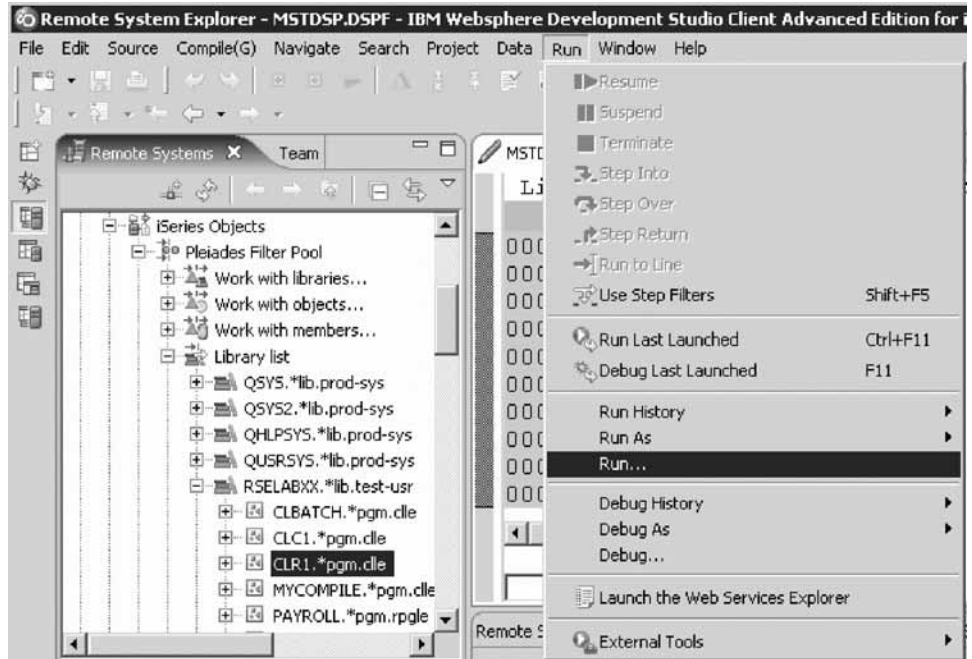
7. Press F3 to end the program.

**Tip:** You can edit, delete and create run configurations by clicking the arrow beside the icon on the workbench toolbar and selecting from the list.



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





The Run Launch Configurations window opens.



Here you can see the CLR1Run configuration that you just created. This is your saved configuration to run CLR1 as an interactive application. Notice the list of configurations you can choose.

You have created and saved a run configuration.

### Lesson checkpoint

You learned the following:

- About run configurations
- How to create a run configuration

### Module summary

In this module, you learned how to work with some of the many features of the Remote System Explorer perspective.

### Lessons learned

- Know the features of Remote System Explorer
- Move, dock, rearrange, resize, hide, close, reopen and add views
- Save and reset a customized perspective

- Create a filter to show specific iSeries libraries
- Change the filter to add more iSeries libraries
- Create a filter to show all the source files in a library
- Access members to edit from your filter
- Create a user action that copies a source file with data to a new source file in the same library
- Specify user action parameters
- Specify a restriction on a user action
- Try the user action
- Create a user action for iSeries jobs
- Create your own compile command
- Using run configurations

### **Assessment**

- What is the Remote System Explorer?
- What filters list a set of libraries?
- What is the purpose of a filter?
- Can you create filters for all connections or a specific connection?
- Can you give filters a specific name for future use?
- What filters list a set of objects?
- What is the purpose of a filter pool?
- Can you define filters for non iSeries servers and your local system?
- What is the purpose of a user action?
- Can you specify a restriction on a user action?

---

## **Designing screens and reports**

This module teaches you about the various aspects of the CODE Designer while modifying a display file to add a screen and creating a simple printer file. You will step through each part of the CODE Designer tool interface and update some DDS as well. In the workbench, in the Remote System Explorer perspective you will use the connection that you used in the module before.

### **Learning objectives**

- Open a DDS member for edit with CODE Designer
- Show file-level keywords and record keywords
- View the details of records, record-level keywords and field-level keywords
- View the design of the payroll application main menu
- Create a group from an existing record format
- Create a new group and add a subfile record and a subfile control record
- Add columns to the subfile record
- Add fields to the subfile control record
- Copy existing fields
- Set indicators to handle field errors
- View and update record and field properties
- View keywords and the properties of a keyword
- Insert a keyword

- View help for a keyword
- Check there are no semantic errors in the DDS source
- View help for an error
- Launch the editor in read mode from the error list
- Launch the editor in write mode to fix the error
- Find a keyword in the source
- Save source changes
- Compile your source changes
- Create a printer file report
- Close the Designer

### Time required

This module should take approximately 30 minutes to complete.

## Opening a DDS member in the Remote Systems view

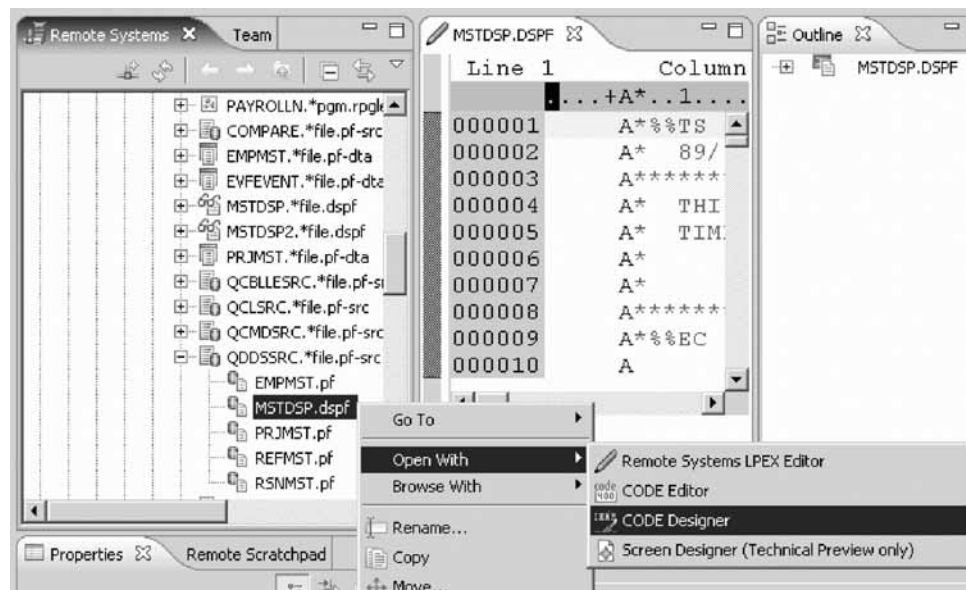
Using an editor to create and maintain DDS source for your display, printer and physical files can be a frustrating and difficult task. What would be great is a graphical design tool that let's you design your screens and reports visually and then generate the DDS source for you. Well, that's exactly what CODE Designer does for you.

CODE Designer helps the novice DDS programmer create screens, reports and databases quickly and easily without worrying about the details of the DDS language, while at the same time letting the expert DDS programmer get access to all the features and power of the language.

**Tip:** Make sure MSTDSP is closed from the previous LPEX Editor lessons.

To open a DDS file member from the Remote Systems view:

1. Expand the Library list filter if it is not already expanded.
2. Expand the QDDSSRC file in library RSELABxx.



3. Right-click the MSTDSP member.
4. Click **Open with > CODE Designer** on the pop-up menu.  
The member MSTDSP will be downloaded to the workstation and loaded into CODE Designer. CODE Designer is a separate tool not integrated into the workbench.

You have opened the DDS source member MSTDSP in the Remote Systems view using CODE Designer.

### Lesson checkpoint

You learned the following:

- About CODE Designer
- How to open a source member in CODE Designer

## Viewing the DDS tree

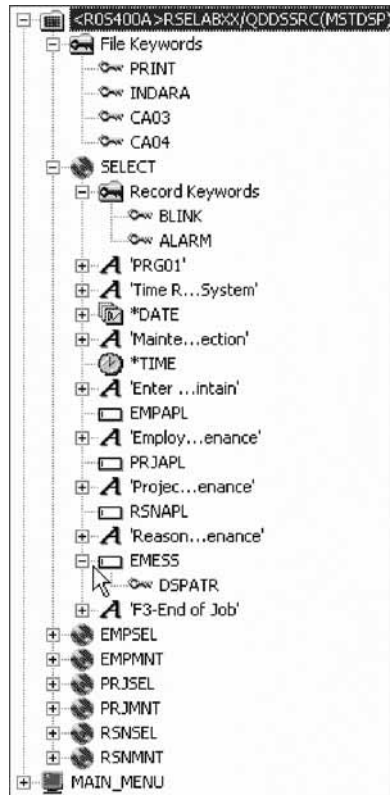
What you are looking at now is basically an Explorer view of the DDS. The DDS tree view on the left-hand side of the Designer displays the DDS source in its file, record, field, and keyword hierarchy. It is a familiar and intuitive way to see the overall structure of the DDS source and to navigate through it quickly. Don't worry if you're not a DDS expert, everything will be explained to you.

The DDS tree shows groups of records, which represent the screens or reports you are designing, as peers of the file in the tree hierarchy.

In this view, you can create groups, and copy or move keys, keywords, fields, and records. If any DDS object contains an error, the icon representing it displays a red X.

To show file-level keywords and the record SELECT in the DDS tree:

1. Expand the <Servername>RSELABxx/QDDSSRC(MSTDSP) folder.
2. Expand the File Keywords folder.
3. Expand the SELECT record.
4. Expand the Record Keywords folder.
5. Expand the EMESS field.



The DDS tree now shows you a summary of the file-level keywords and of the record SELECT.

You have seen the file-level keywords and the record SELECT in the DDS tree.

### Lesson checkpoint

You learned the following:

- About file-level keywords
- How to view file-level keywords

## Selecting the DDS object

In the upper right-hand side of the Designer is the Workbook with several different tabbed pages. The Workbook is the area of the CODE Designer where you design display files, printer or physical files. You can view this notebook on the top right-hand side of the CODE Designer window. The top page is called Details and it provides a detailed view of the DDS objects selected by the DDS tree. You can view this page in either details mode or list mode by clicking View > List from the CODE Designer menu.

In the Details page columns display information about the selected DDS object. You can use this page to display for example, details of all the fields in the record SELECT or keywords and conditions of a field or record.

The Listing page is a listing of the source statements generated by the Program Verifier.

In the bottom right-hand side of the Designer is the Utility notebook. This notebook contains several pages: Selected DDS, Web Settings, Comments and Error List. The Selected DDS page in the notebook shows the actual DDS source for the currently selected item.

**Tip:** The Web Settings page allows you to specify attributes that are used by the WebFacing tool.

To work with the DDS record SELECT:

1. In the DDS tree click the SELECT record.

The Details page lists all the fields in the record SELECT and summarizes some of their properties. The Selected DDS page shows the DDS for the SELECT record.

Field	Positi...	Len...	Type	Shift	Usa...	Sample
'PRG01'	2, 5	5	Text const...			PRG01
'Time R...System'	2, 30	21	Text const...			Time Reporting Syste
'DATE'	2, 70	6	Date const...			MM/DD/YY
'Mainte...ection'	3, 30	21	Text const...			Maintenance Selectic
'TIME'	3, 70	6	Time const...			HH:MM:SS
'Enter ...intain'	6, 14	54	Text const...			Enter an X beside the
EMPAPL	9, 25	1	Alphanumeric	A - Alphanum...	Both	B
'Employ...enance'	9, 28	27	Text const...			Employee Master Mai
PRJAPL	10, 25	1	Alphanumeric	A - Alphanum...	Both	B
'Projec...enance'	10, 28	26	Text const...			Project Master Mainte
RSNAPL	11, 25	1	Alphanumeric	A - Alphanum...	Both	B

2. In the DDS tree click Record keywords immediately below SELECT.

The Details page shows the current record-level keywords. The Selected DDS page still shows the DDS for the SELECT record.

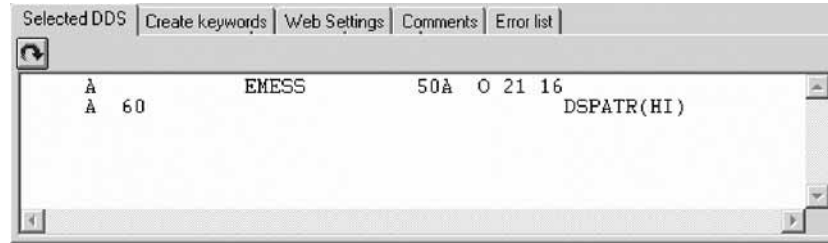
Keyword	Conditioni...
BLINK	
ALARM	60

Selected DDS	Create keywords	Web Settings	Comments	Error list
<pre> A* 89/03/02 12:07:25 DOUG REL- A* THE SELECT FORMAT ALLOWS SELECTION OF A* MASTER FILE THE OPERATOR WANTS TO MAIN A      R SELECT A A      BL] A 60      AL] A      2 5'PF A      2 30'Tj </pre>				

3. In the DDS tree click the EMESS field.

The Details page shows its field-level keywords. The Selected DDS page now shows the DDS for the EMESS field.



Even this relatively small and simple DDS source member demonstrates how much easier it is to use the Designer to navigate through your DDS source. The syntax is being interpreted in intuitive graphical ways making it an ideal tool for learning DDS. But to get orders of magnitude improvement in your productivity what you really need is to work with your screens and reports in a WYSIWYG fashion, completely independent of the DDS required to make things appear the way they do. You need the Design page.

You have seen the details of the record SELECT, the record-level keywords and the field-level keywords.

### Lesson checkpoint

You learned the following:

- About viewing the details of a record
- How to view the details of a record

## Designing the DDS screen

You will spend most of your time creating, updating, and designing your DDS screens and reports in the Design page. The Design pages allow you to design your screens or reports visually using an intuitive graphical user interface. The Design page shows the DDS source as it would appear on either a screen (for display files) or a printed page (for printer files). It allows you to design your application's screens or reports by laying out records and fields in a graphical user interface.

On the Design page, you can easily create, edit, resize, and move DDS objects graphically. You can create new records, fields, and constants directly on the Design page by using the palette push buttons to the left of the Design area or from the pop-up menus. The toolbar above the Design area allows quick access to many of the editing features as well as information about the currently selected object.

Click the **MAIN-MENU** tab in the workbook.





In order to understand where MAIN\_MENU came from, you need to understand the concept of a group. A group is simply a collection of one or more DDS records that represent how a screen or report would be assembled at runtime. It allows you at development time to work with screens or reports as they would appear when they get assembled by your programs at runtime. To work with groups in CODE Designer you need to tell CODE Designer which record formats make up a screen. In this case this has been done for the screen you see in the Design page. A group MAIN\_MENU has been created for you and CODE Designer has saved this information in the DDS source in comment lines. Any groups that you create are persisted as comment lines in the DDS so you can re-use these groups in subsequent CODE Designer sessions.

The groups you create will appear in the tree view as well as in the workbook as a Design page tab for each group defined, to allow quick access to each group of records.

You have seen the concept of a group, specifically the MAIN\_MENU group.

### Lesson checkpoint

You learned the following:

- About groups
- How to see a group in CODE Designer

## Creating groups from existing records

If you are working with existing DDS, you will want to create groups that will correspond to how the records are being used. In this example you will create a group for the next screen, where the user selects which employee in the payroll database to maintain. The screen is made up of the record format EMPSEL.

To create a new group:

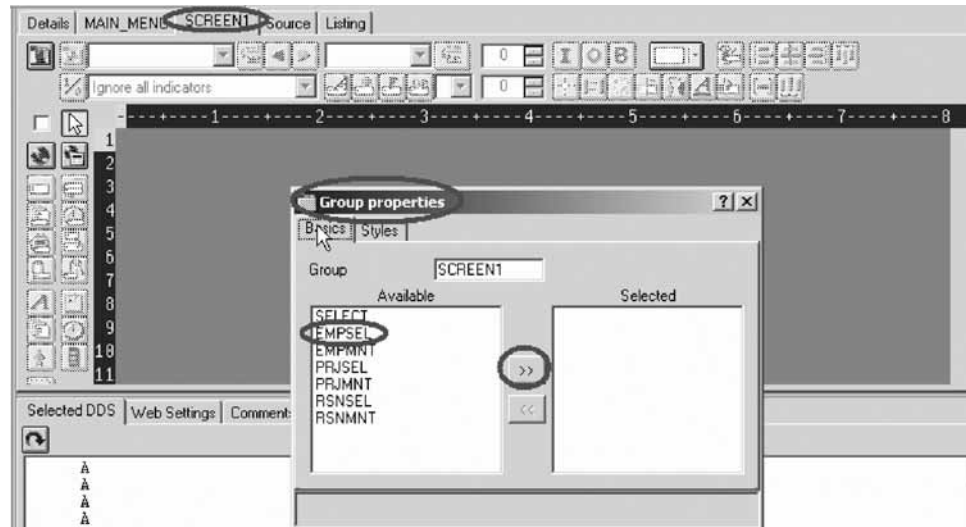
1. Scroll to the bottom of the DDS tree and expand the MAIN\_MENU group.  
The SELECT record appears as the only record in this group.



2. Right-click the MAIN-MENU group.

3. Click **Insert group** on the pop-up menu.

A Group Properties notebook opens and a blank Design page for the group SCREEN1 also opens.



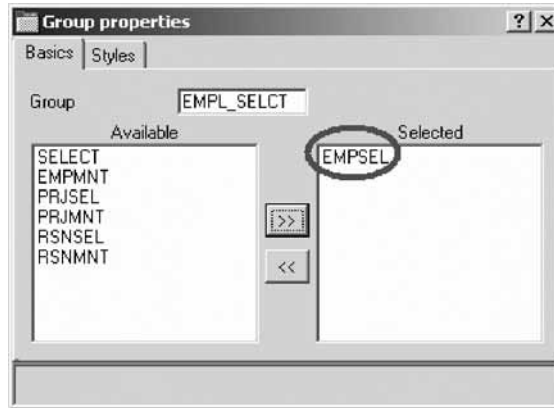
The Properties notebook lets you view and update the properties of the currently selected DDS object. You can open this notebook from any view, pop-up menu, or menu of the CODE Designer. The Properties notebook is modeless. When you change an object's properties, the selected object changes immediately.

4. In the Group Properties notebook, select the EMPSEL record from the **Available** list and click the **>>** button.

For simplicity this is the only record you will add for now. The Design page now shows you what the record EMPSEL looks like.



5. In the group field, type `EMPL_SELCT` over `SCREEN1` to rename the group.



6. Close the Group Properties notebook. Click the **X** in the top right corner of the Group Properties notebook.

You have finished creating a group. You could now work in the Design page with the record formats contained in this group. Instead you'll create a new record format.

It appears that this is one of those unusable applications where you have to know the employee number ahead of time instead of being able to browse what is in the database. What we really need is a subfile. But aren't those difficult to code, you ask? Not with CODE Designer.

You have inserted a new group using the existing record EMPSEL.

### Lesson checkpoint

You learned the following:

- About inserting groups
- How to insert a new group

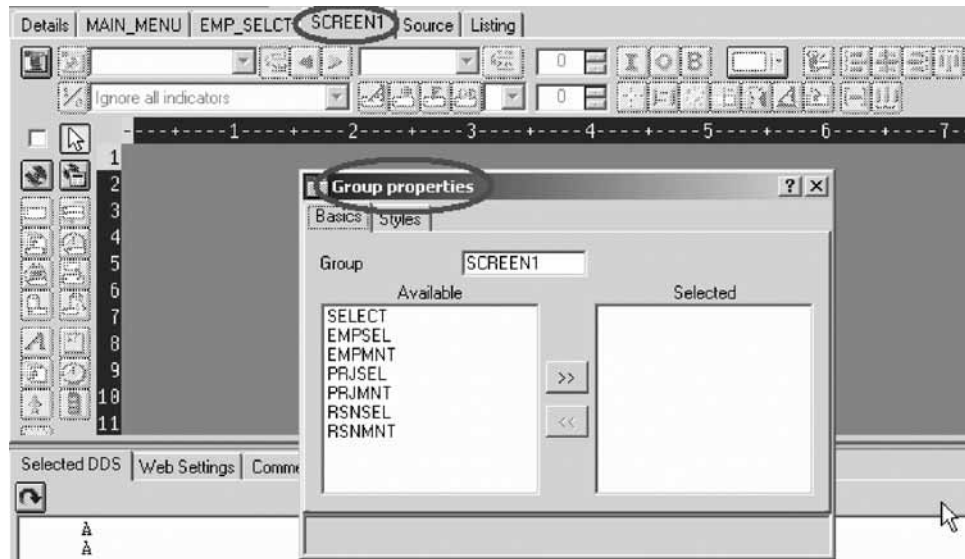
## Creating new screens

To create a new record screen on the Design page you need to create a group that will create an empty page you can work with.

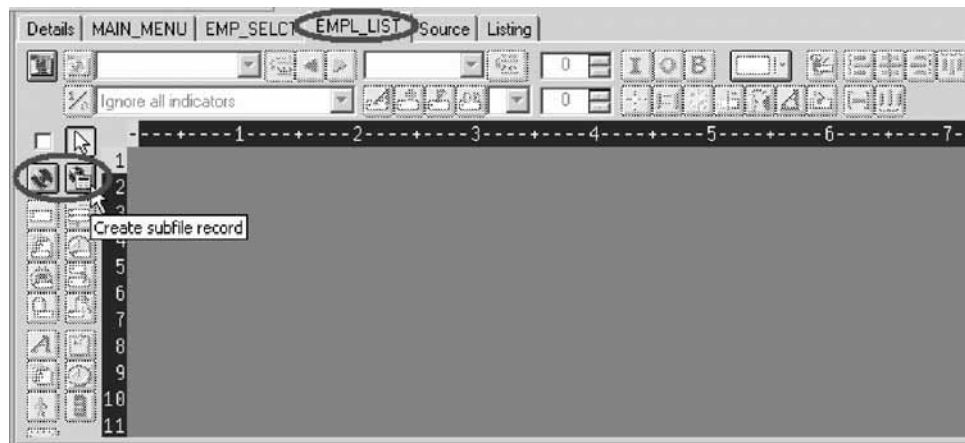
To create a new group:

1. Right-click the new EMPL\_SELECT group in the DDS tree.
2. Click **Insert group** on the pop-up menu.

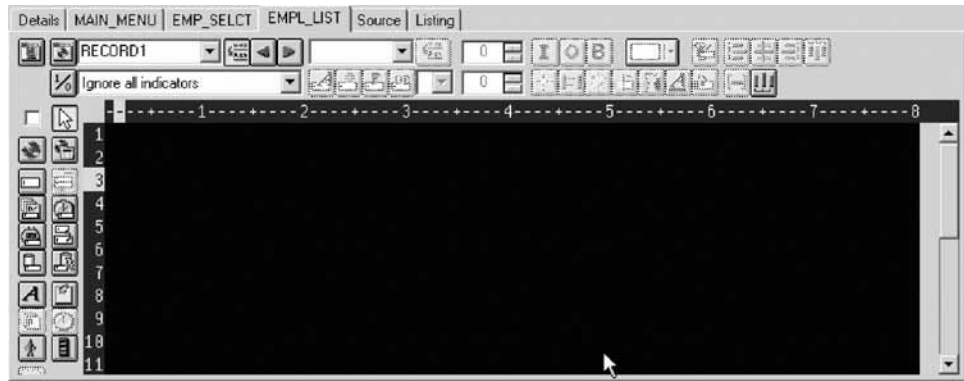
A Group Properties notebook appears and a blank Design page for the group SCREEN1 also appears.



3. Rename the group to EMPL\_LIST and close the Group Properties notebook.
4. You can create things on the Design page by selecting the appropriate tool from the palette on the left-hand side and then click on the Design page where you want it to be created. Right now, most things are disabled in the palette because there is no record in which to create fields. The only two buttons available are **Create standard record** and **Create subfile record**. If you leave the mouse over a button for a second or two, flyover help will appear describing the indicated button.



5. Click the **Create subfile** record button and then click in the dark gray area. A subfile and a subfile control record pair are created.



You have created a new record screen.

### Lesson checkpoint


You learned the following:

- About new screen records
- How to create a new screen record

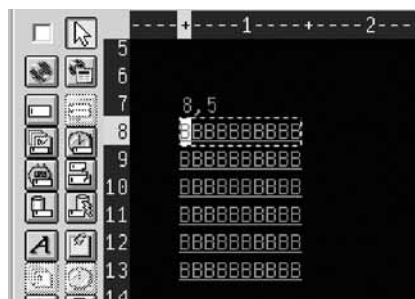
### Adding fields to the subfile record

Now you add some columns to the subfile using the Design page. The subfile should be positioned on row eight. You use the cursor to specify the location of the part you want to put on the screen, in this case your subfile.

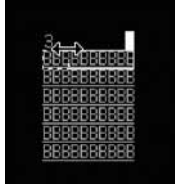
To add fields on the subfile:

1. Click the **Create named field** button  and then **click** somewhere on **row 8**. Six fields appear in a vertical column. This is because the subfile you created, currently specified a SFLPAGE (visible list size) of six.
2. Click the top field and **hold** the mouse button down and **move** it to **row 8, column 5**.

Note the current row and column appear just above the field as you move it.




3. Move the cursor over to the **right edge** of the field. It turns into a double-headed arrow. Hold down the mouse button and **move** it to the **left**. The size of the field will be reduced. The current size will appear just above the field. When the size is **3**, let go of the mouse button.



The toolbar at the top of the Design page is a very convenient place to monitor and manipulate the currently selected parts.

4. Rename the record from RECORD1 to EMPLSTSFL and the field from FIELD1 to OPCODE by typing over the text in each list.



5. Click the Color palette  button and select pink to change the color of the field.
6. Click the **I** button to change the usage of the field to input.



Now you will create an additional column in the subfile.


7. To create an additional output column:
  - a. Position the cursor at **row 8, column 9**.

**Tip:** The bottom right of the CODE Designer window shows the current cursor position .

If you can't see the field with the cursor position on your screen, click the **Maximize** button in the top right corner of the screen. You can use the cursor keys or the mouse to move the cursor.

- b. If you are creating a long field with an exact length, the SDA syntax can be easier. Type: `+0(30)` and then press the **back arrow** (not Backspace!) to select the text you created.

Notice from the Selected DDS page that you have created a text constant containing `+0(30)`.

- c. Click the **Convert string to field**  button on the toolbar or press **F11** to convert the SDA syntax into a character output field of length 30.



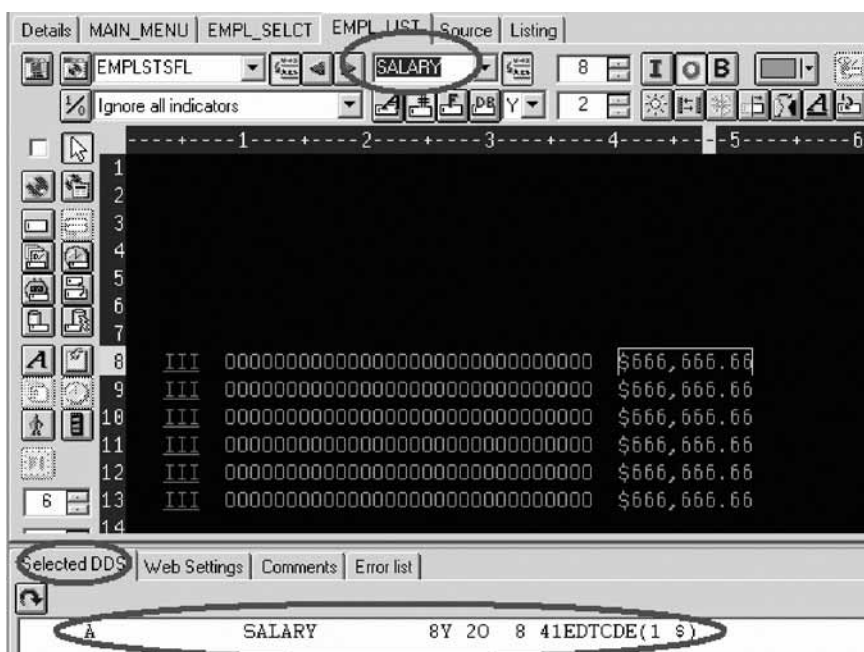
- d. Rename the new field to ENAME using the toolbar. This will show the name of the employee.




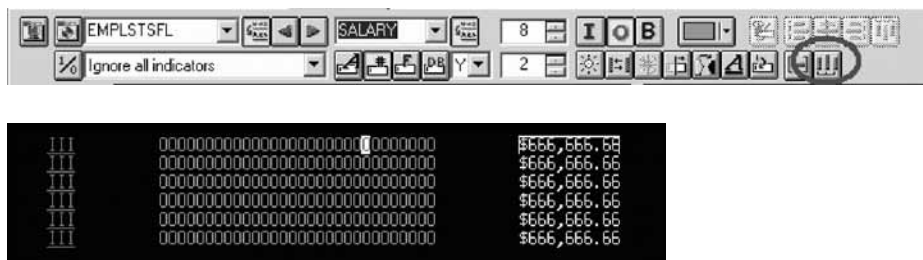
- e. Position the cursor to **8, 41**.
- f. Now you will add a field for the employee's salary. Type \$666,666.66 and then press the back arrow.

Now wouldn't it be better if we could just tell the Designer what we wanted the number to look like and then have Designer generate all the cryptic EDTCDEs to make it happen?

- g. Press **F11** to convert this field into an output numeric field with comma delimiters, two decimal positions, a currency symbol and no sign. Look at the Selected DDS page to see what was generated for you. Impressive!

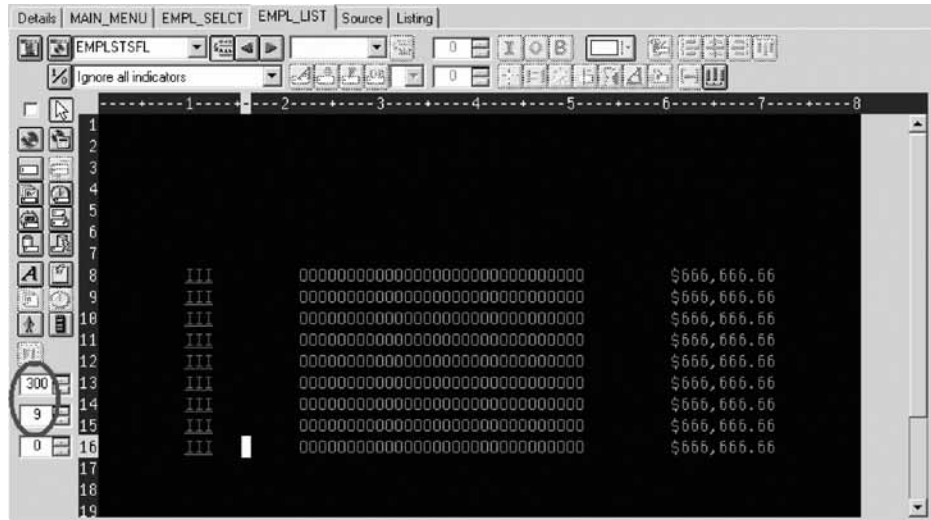


- h. Rename the field to SALARY and change its color to yellow, using the toolbar.
- i. The subfile seems compacted to the left. It would be better to space it out evenly. Just select a field and click the space horizontally  button on the far right side of the toolbar. The other alignment buttons will align fields, left, right, center and top.



Just below the palette on the left there are three spin buttons. The top one, **Subfile size**, specifies the total number of entries in the list that will be filled in by the application. The second one, **Subfile page size**, is how many entries appear on the screen.

- j. In the **Subfile size** field, type 300.
- k. In the **Subfile page size** field, type 9.
- l. Click in the Design page.



The Design page is updated accordingly.

You have added some columns to the new subfile record screen.

### Lesson checkpoint

You learned the following:

- About adding fields to a subfile record
- How to add fields to a subfile record

## Switching between multiple records

Now let's fix up the subfile control record. The group you created contains 2 records. You can verify this by looking at the record list in the toolbar:

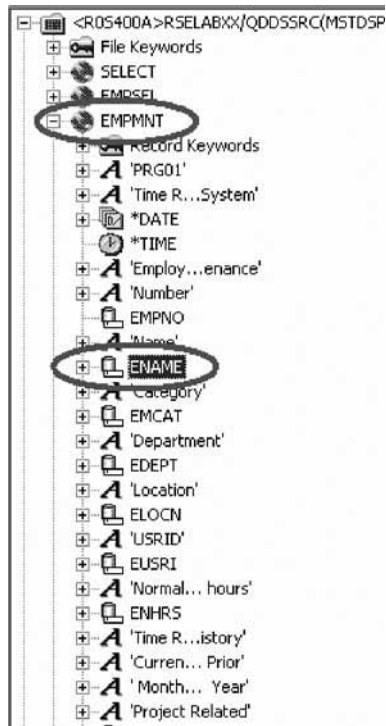


1. Change the current record by selecting RECORD1CTL from the record list or click next record  or press **Alt+End**.

The fields in the subfile still appear so that column heading can be lined up, but they appear at half-intensity so that they can be distinguished from the fields of the current record.

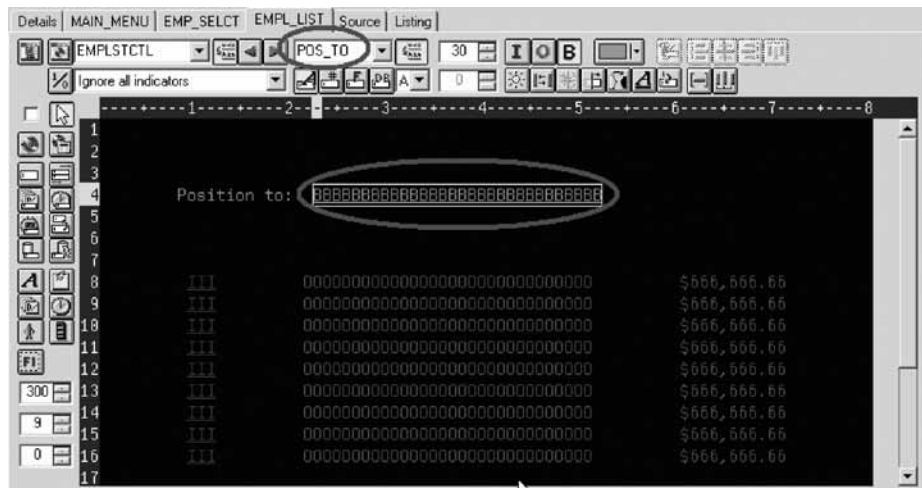






(The pop-up menu or Edit menu shows the Copy menu item as well).

- c. Position the cursor to 4, 23 and press **Ctrl+V**. Now that was easy!
- d. Click the field and change the name from ENAME to POS\_TO.



You have fixed up the subfile records to add an employee name field with a position to entry opposite this field.

### Lesson checkpoint

You learned the following:


- How to fix up the subfile records to add an employee name field

## Adding field error handling

Let's put in some error handling for the 'position to employee name' field. If the employee name is not found in the database, the program will set on indicator 60.

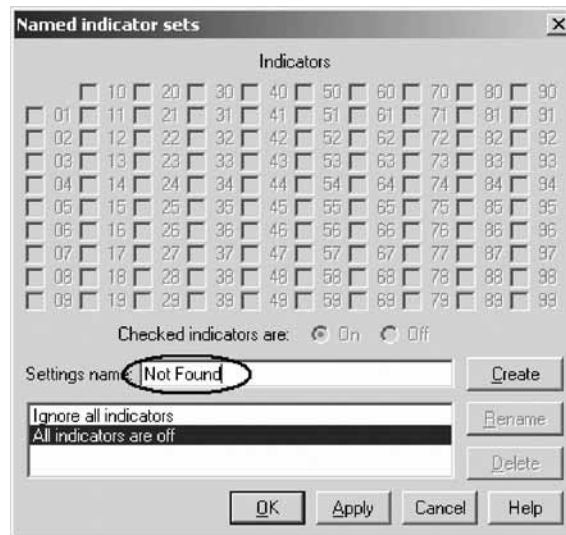
In this case the field should turn red, reverse image and position the cursor to it. Now wouldn't it be better if you had something easier to remember than some arbitrary number from 1 to 99.

To set indicators:

1. Click the Change named indicator sets  button on the Design page toolbar (or press F7.)

The Named indicator sets window opens.

2. In the **Settings name** field, type: Not Found.
3. Click **Create**.



4. Select the check box next to **60** and click **OK**.

The Not found indicator set is now in effect. The Design area is shown as if indicator 60 was on and all other indicators were off. The Design page toolbar shows the current indicator set in the **Select named indicator set** list on the bottom left.

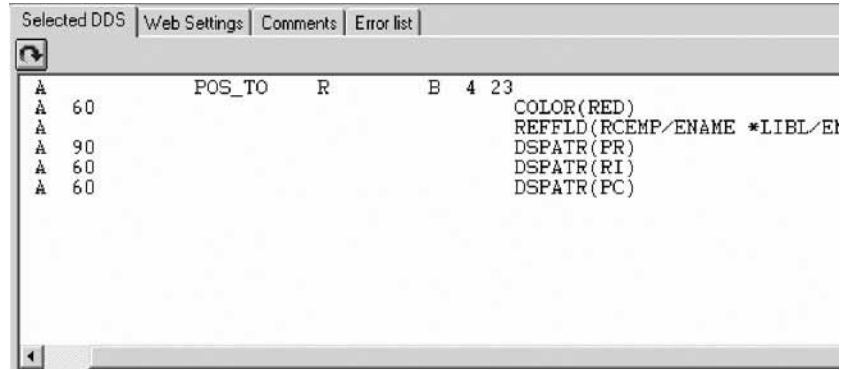


5. Now select the **POS\_TO** field.
6. On the toolbar, select the color **red** and the display attributes **reverse image** and **position cursor**. (The set of toolbar buttons representing the current display attributes is found just below the Color button).

The toolbar should look as follows:

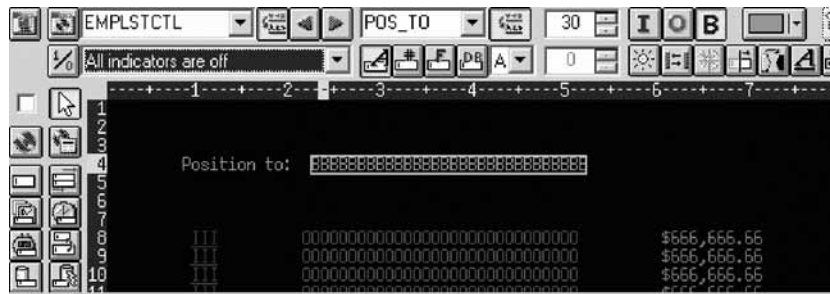
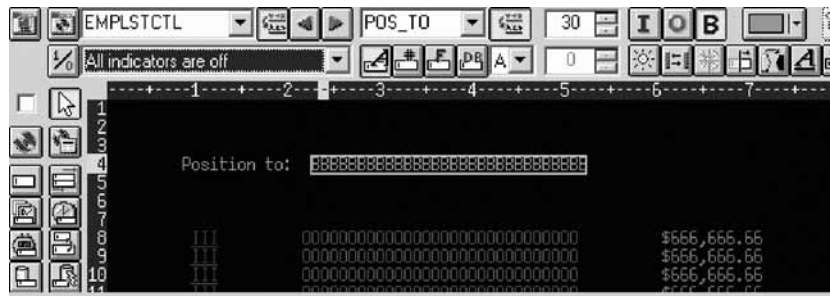


7. Examine the DDS generated in the Selected DDS page.



Notice that all the new keywords were created with a condition of 60. (The DSPATR(PR) was pasted with the field originally).

- Now let's try it out! From the **Select named indicator sets** list, select **All indicators are off**.



- From the **Select named indicator sets** list, select **Not found**.  
The field appears red and reverse imaged.

You have added error handling to the position to employee name field.

### Lesson checkpoint

You learned the following:

- How to add error handling

## Accessing field properties

Second to the direct manipulation and the toolbar on the Design page, the easiest and quickest ways of getting access to the properties of a field, record, or entire file is the Properties notebook.

The Properties notebook lets you view and update the properties of the currently selected DDS object. You can open this notebook from any view, pop-up menu, or menu of the CODE Designer.

The Properties notebook is modeless. When you change an object's properties, the selected object changes immediately.

You can get to a Properties notebook from the **Selected** menu, by pressing **F4**, or double-click on anything in the DDS tree or the Details page or Design page.

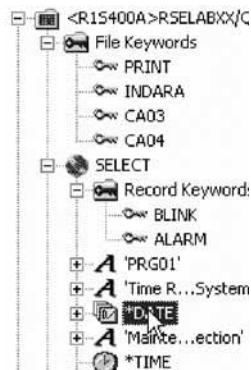
To open the Properties notebook:

1. In the DDS tree, click the record SELECT and press **F4** to see the Record properties.

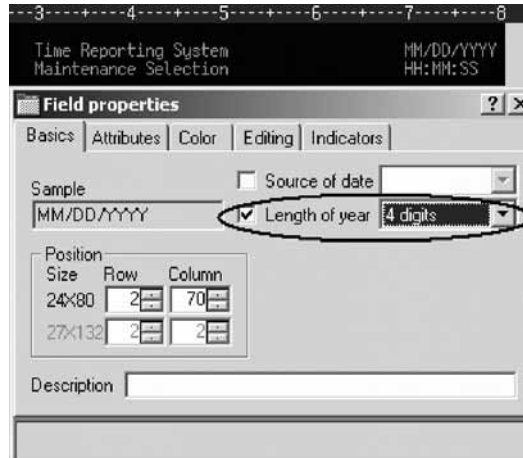


As you select different items, the Properties notebook will continuously update itself to show you the properties of the selected item.

2. Click the **\*DATE** field in the SELECT record. (You may have to move the Properties notebook out of the way.) This field has a different set of pages describing its properties.

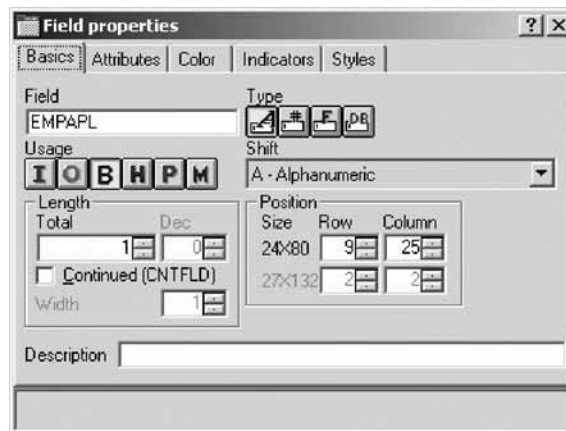


3. Change the year from 2 to 4 digits. Select the **Length of year** check box.
4. Select **4** digits from the list.



Notice how the sample is updated on the Properties notebook.

- To test the Design page, click the **MAIN\_MENU** tab in the workbook and look at the upper right corner of the screen. The year now has 4 digits.
- Click the EMPAPL field in the SELECT record. On the Field properties notebook click the **Basics** tab. On this page you can change the field's name, usage, length, type, and screen position. The other pages give you quick access to other properties of this field.



You have seen the record properties for the record SELECT and changed the length of year to 4 digits.

## Lesson checkpoint

You learned the following:

- About the record properties for the record SELECT
- How to change the length of year to 4 digits

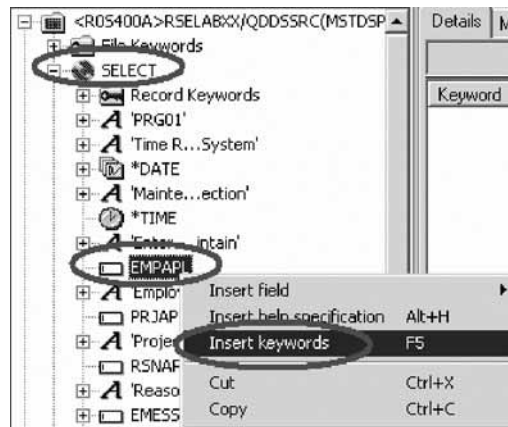
## Adding new keywords

CODE Designer helps you manage the visual aspects of your displays and reports. But you also need to access the full power of DDS. You need to access keywords.

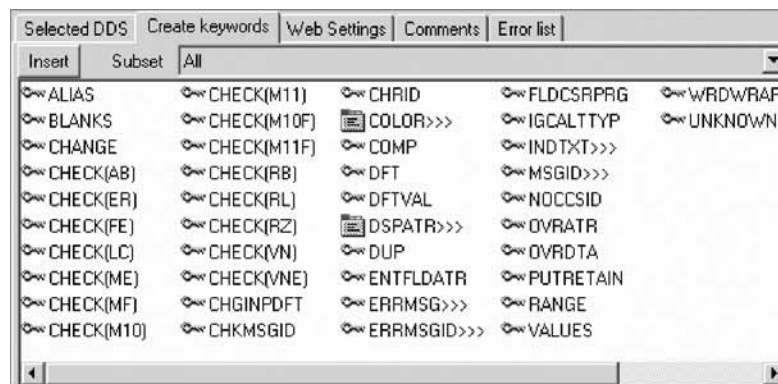
To add keywords:

1. Click the EMPAPL field in the DDS tree.

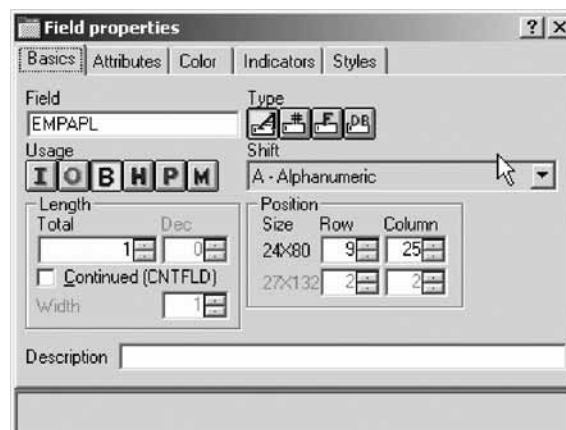
- Press **F5** or right-click and click **Insert keywords** on the pop-up menu.



You see the Details page for the EMPAPL field and the **Create keywords** tab is added to the Utility notebook. This page shows you the subset of keywords that are allowed for the selected file, record or field and it takes into account the field's type, usage, shift and what record it is in. It is very powerful to know exactly what your options are. This information cannot be quickly ascertained from the Reference manual.



- With the EMPAPL Properties notebook at the **Basics** page, click the numeric field  button to change the field to numeric type.



Notice that the list of keywords in the Create keywords page has changed.

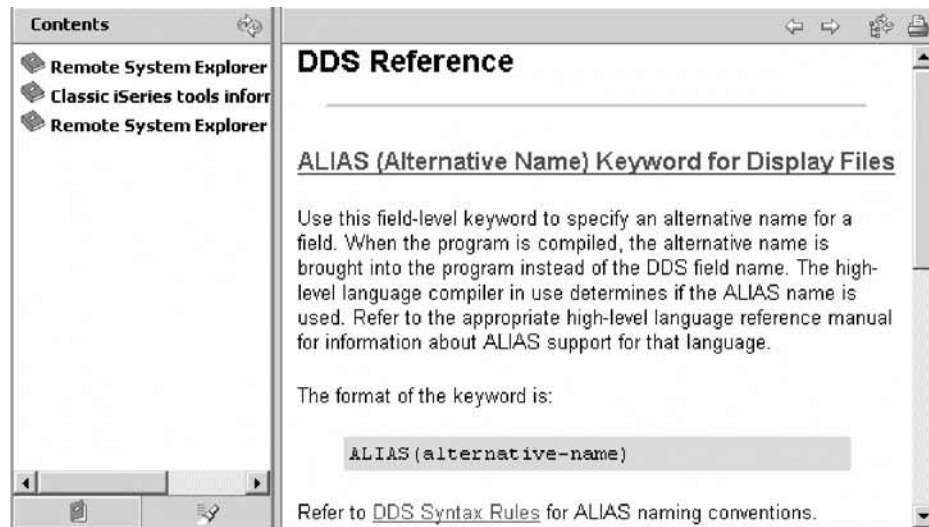
- Click the  button to change the field back to alphanumeric.

Notice that the list of keywords in the Create keywords page has changed again.

5. Click the **ALIAS** keyword and press **F1**.

The DDS Reference help for the ALIAS keyword appears.

**Tip:** CODE Designer has lots of on-line help. Press F1 anywhere you want to see help for an item, icon or notebook. You will see help relevant to what you are currently trying to do. From the Help menu you have quick access to the DDS Language Reference as well as several other useful sources of information.



6. Minimize the Help window.
7. In the Create keywords page, double-click the INDTXT keyword. (You may have to scroll to the right to find it).  
The keyword is created with default values which can be changed when you want.
8. Double-click the INDTXT keyword again.  
The keyword is created with the same default values creating a conflict.

Keyword	Parame...
EDTCDE	3
INDTXT	01 '?'
INDTXT	01 '?'

9. Close the Keyword Properties dialog.

You have seen the help on the ALIAS keyword and added INDTXT keywords.

### Lesson checkpoint

You learned the following:

- About DDS keywords
- How to see help on a DDS keyword


## Verifying the source changes

You have just added a new record and some new fields to your DDS source. Everything that the CODE Designer adds to your DDS source is certain to have the



correct syntax. Now you need to make sure that there are no semantic errors. You just introduced one in the last exercise by creating two INDTXT keywords describing the same indicator.

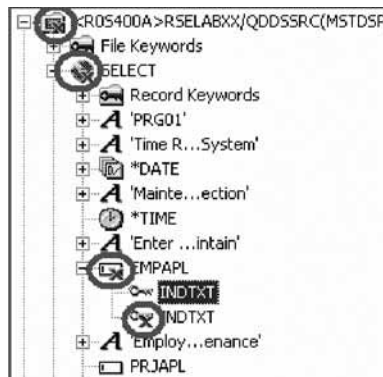
To verify your source:

1. Click **Tools > Verify file** (or click the verify  button on the main toolbar) on the CODE Designer menu.



The DDS source is checked using the same verifier that the CODE Editor or LPEX Editor uses. A message appears on the status line at the bottom of the Designer stating that the verify process completed with errors.

2. In the DDS tree, there is a trail of red X's leading to the problem.



The file icon has a red X, as does the SELECT record, the EMPAPL field and finally the second INDTXT keyword.

3. Click the **MAIN\_MENU** tab in the workbook.

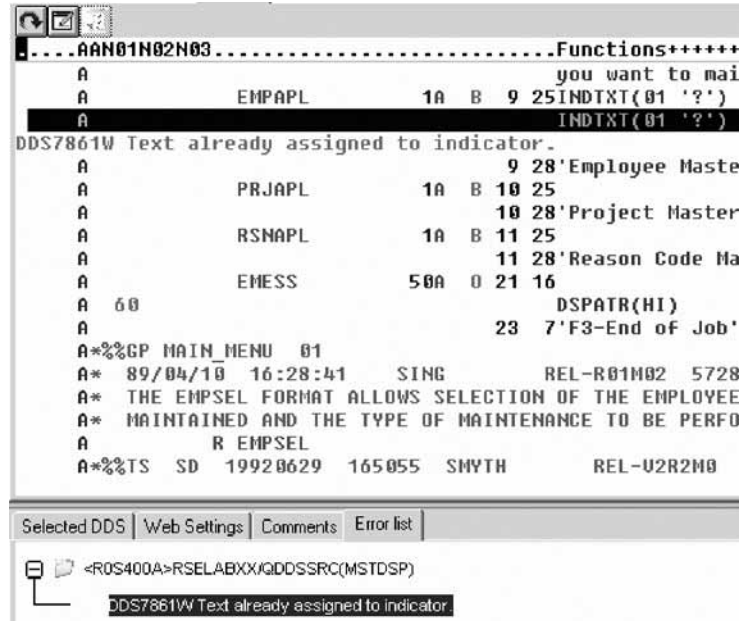
The EMPAPL field is highlighted in red.

4. Click the **Listing** tab in the workbook.

This page shows you the listing generated by the most recent program verify. A warning message is buried somewhere in the listing but it's not easy to find.

5. If there are problems, they will show up in the Error list page in the Utility notebook. It behaves exactly like the Error list in the CODE Editor or LPEX Editor. Click the **Error list** tab.
6. Double-click the warning **DDS7861** in the Error list. (Press **F1** to see detailed help on the message).

The Source page appears and the cursor is placed exactly where the error is in the source. The Source page is a tokenized read-only view of the current state of the DDS source. Read-only? Wouldn't it be great if you could just clear the error right here. There are some things that are just plain faster in the editor and many others that are faster in the visual environment. It would be great to switch between the two modes at the push of a button. Well, let's just do that.



You have verified your DDS source and have identified an error in the source.


### Lesson checkpoint

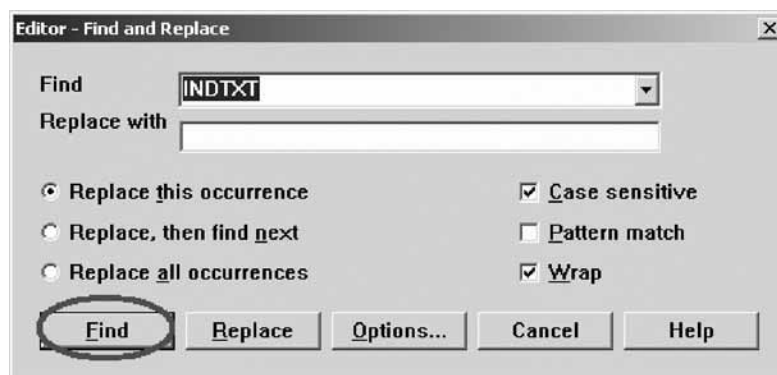
You learned the following:

- About program verifier
- How to verify DDS source

## Switching between designing and editing the screen

To switch between the Design mode and the Edit mode:

1. Click the Edit DDS source  button or click **File > Edit DDS source** from the Editor menu.  
You now have access to the full power of the editor.
2. Explore the **Edit** and **View** menu items.
3. Press **Ctrl-F** to open the Find/Replace window.
4. In the **Find** field, type **INDTXT** and click **Find**.



5. Press **Ctrl-N** to find the next occurrence.
6. Delete the second **INDTXT** line. Type **D** in the number column and press **Enter**.

You have edited the source to fix the error by switching to the editor.

## Lesson checkpoint


You learned the following:

- About switching between design and edit mode
- How to edit the source to fix an error

## Compiling your source changes

Now you will compile the source on the iSeries just as you did in the Remote Systems LPEX Editor.

To compile your source:

1. Click **File** > **Save** from the Designer menu to save your source to the iSeries.
2. Click **Tools** > **Compile** from the Designer menu and then click **No prompt** (or click the compile  button on the CODE Designer toolbar).
3. A message indicates when the compile is complete. Click **OK** in the Message dialog. If you re-compile and run the payroll program, you will see the 4 digit year change you made to the opening screen of the program.

**Tip:** You will also see the message CPD7886W Field overlaps another field with no conditions specified. You can ignore this message.

You have compiled your source changes.

## Lesson checkpoint



You learned the following:

- How to compile source changes





## Creating a report and closing the Designer

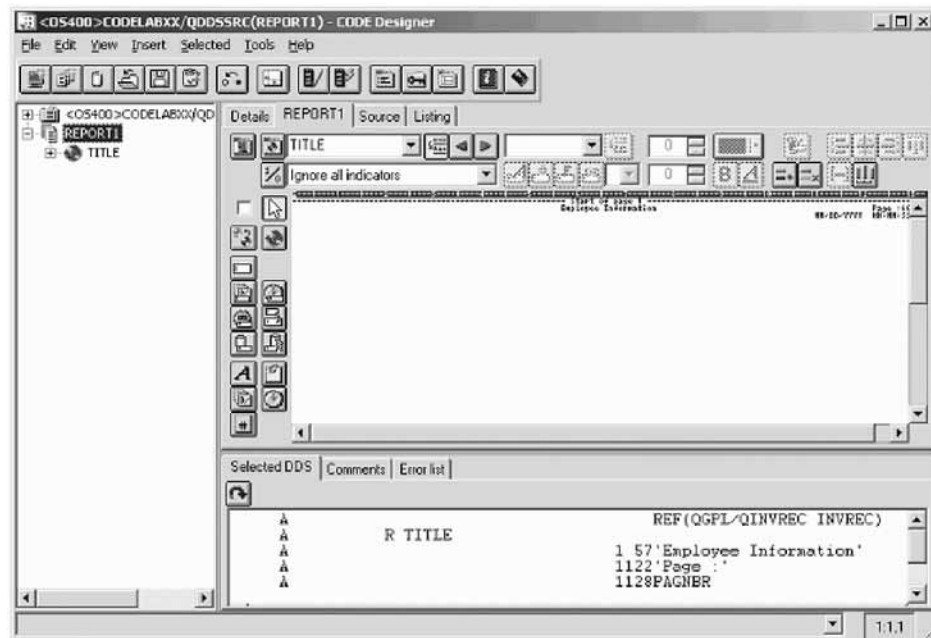
Now you create a simple printer file as well as explore checkpoints, which can be used for all DDS created or maintained with CODE Designer. You will begin by looking at the physical file specification for the database file that the printer file you will create will use in CODE Designer.

To create a printer file:

1. In CODE Designer open the member REFMST in file QDDSSRC in library RSELLABxx.
2. Take some time to explore the fields and information for this physical file. You may want to refer back to this information as you work through the exercise. Now that you are familiar with the file REFMST, its time to begin creating a printer file.
3. In CODE Designer, click the Create new PRTF file button  on the toolbar. An initial printer file is created and the design page in the workbook is blank.
4. Click the Create absolute record button  at the left of the design page and then click on the first line of the Design page.

The Design page will now change to a white background and you will also see the text “Start of page 1” at the top of the Design page.

5. Press F4 to bring up the record properties. Change the record name to Title. This will help you when you create additional records for this printer file.
6. Close the properties dialog.  
You will see your change both in the tree view and in the design page of the workbook.
7. Now you will add some fields. On the first line, enter the text Employee Information. To easily center the text, select the field by clicking on the text with the left mouse button, then click on the Center horizontally button  .
8. On line 1, column 122, enter the text Page.  
**Tip:** You can always use your mouse to drag and drop a field wherever you'd like it. And when you do this, you will see the current position of your field next to your pointer. You could also use the properties dialog for the field to position a field to your desired location.
9. Select the Create a page number constant button  and point to the empty space after the text field you added in the previous step (about column 128). The page number constant will be added to your Title record.
10. On line 2, column 110, add a date field by selecting the Create a date constant button  and then clicking on the appropriate column. Bring up the properties for the date constant (F4) and change the date to a 4 digit year. You will see your changes appear immediately in the Design page.
11. On line 2, column 122, add a time field by selecting the Create a time constant button  .  
Your report should look something like this.




At this point it is a good time to save your source.

12. Press Ctrl-S. The Save As dialog should appear.
13. Save your changes so far to "<>RSELABxx/QDDSSRC(REPORT1)"  
You will now see a dialog indicating that the source is being saved. You will also see a dialog indicating that a checkpoint is being saved. Because this is

the first time you save this source, a checkpoint is created which you could revert to if you would like to. We will manually create a checkpoint later in this exercise.

14. To add another record:

The record that you have added so far doesn't really do much except set up the header for our report. You will now add a couple more records so that this report design makes much more sense.

- a. Click on the Create relative record button  and click somewhere after line 3.

The record will be added and you will see it in your DDS tree (if the group is expanded).

- b. Press F4 to bring up the properties dialog. Change the record name to COLHEAD.

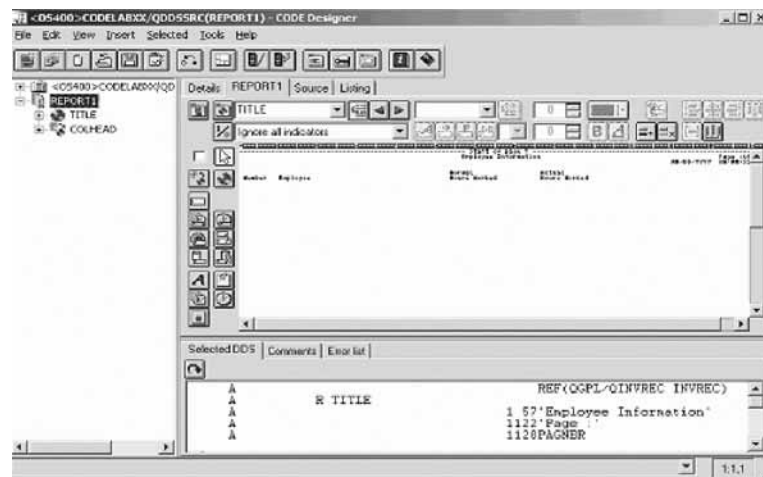
You will use this record to design the column headers for the report.

- c. Close the properties dialog.

You should see your change in the design page in the record name field at the top in the toolbar section of the design page. You will also see a new field at the left toolbar with a numeric spin box. You will use that later.

- d. In row 5, column 2 add the text Number.
- e. In row 5, column 12, add the text Employee.
- f. In row 4, column 54, add the text Normal.
- g. In row 5, column 54, add the text Hours worked.
- h. In row 4, column 77, add the text Actual.
- i. In row 5, column 77, add the text Hours worked.

Your design page should look something like this:



15. To create a manual checkpoint and revert to a checkpoint:

You've done quite a bit of work at this point. Rather than just save your source, it would be a good idea to create a checkpoint so that you can go back to exactly this point in your design.

- a. To create a manual checkpoint, click on the Design page and press Ctrl-M.
- b. Name your checkpoint COLHEAD record. Click **OK**.

Your checkpoint will now be created. To see what happens when you revert to a checkpoint, try to revert to the checkpoint that was created when you first saved your source. To do this, select File -> Revert to checkpoint.

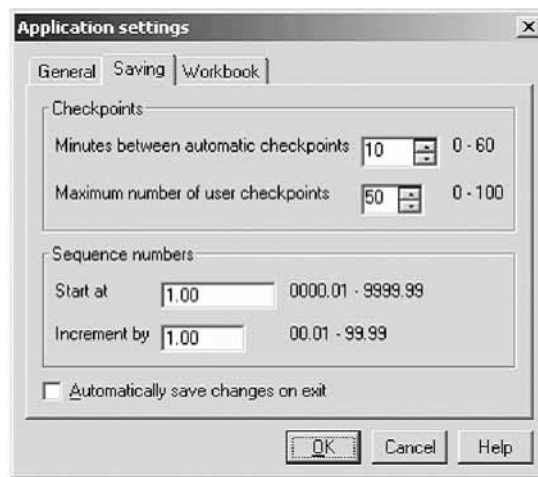
- c. The checkpoint dialog will appear. Select the **Initial load** checkpoint, and click the **Revert** button.
- d. Go to the report Design page.  
You will now see none of the record you just previously added. Saving checkpoints is a good idea so that you can go back to a design that you liked.
- e. Now that you have explored checkpoints, revert back to the checkpoint you created in step 2.

16. To change automatic checkpoints:

Much like autosave in LPEX Editor, CODE Designer has automatic checkpoints which are triggered by elapsed minutes. You can also configure the maximum number of user checkpoints.

- a. Select **Tools > Settings**.

The Application Settings notebook appears.



- b. Select the **Saving** notebook tab.

You will now see the default settings for checkpoints.


- c. If you like, change the number of minutes between automatic checkpoints.

17. To add another record:

Now you will add the record which will be the important part of the report.

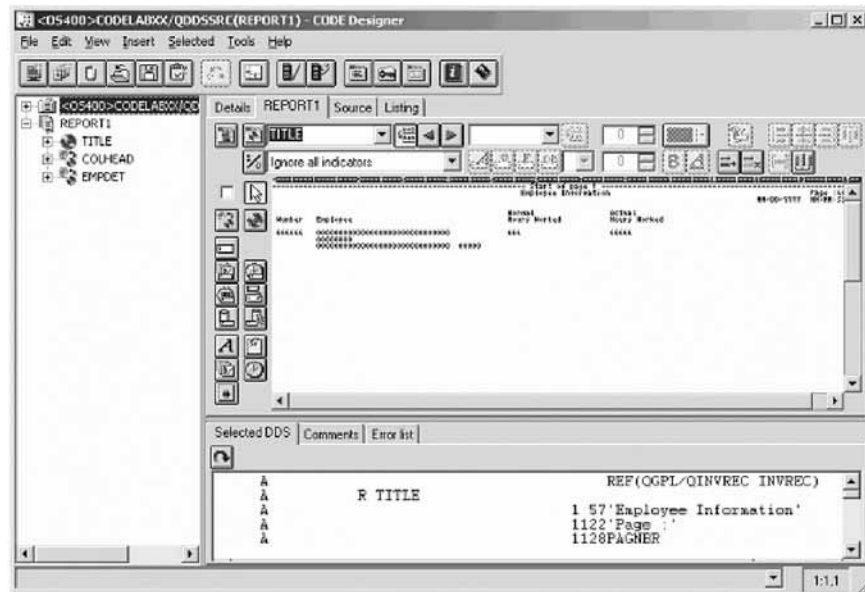
- a. Add a new relative record on line 5 called EMPDET.

This record will contain the employee detail information.

- b. In row 7, column 2 add a database reference field by selecting the Create database reference field button  from the toolbar at the left, and then clicking on the appropriate location in the Design page.
- c. Press F4 to change the settings for the field. Enter RSELABxx as the library, REFMST as the file name and RCREF as the record name. Select EMPNO from the combobox for the field name.
- d. Keep the properties dialog open, but move it out of your way to add more fields to the Design page.
- e. Add another reference field to row 7, column 12, with the following information: library RSELABxx, REFMST as file name, RCREF as record name and ENAME as field name.

- f. Add another reference field to row 8, column 12, with the following information: library RSELABxx, REFMST as file name, RCREF as record name and EUSRI as field name. For proper spacing, select SPACEA and leave the spinbox at 1.
- g. Add another reference field to row 9, column 12, with the following information: library RSELABxx, REFMST as file name, RCREF as record name and ELOCN as field name.
- h. Add another reference field to row 9, column 44, with the following information: library RSELABxx, REFMST as file name, RCREF as record name and EDEPT as field name. Ensure that the relative column checkbox is selected.
- i. Add another reference field to row 7, column 54, with the following information: library RSELABxx, REFMST as file name, RCREF as record name and ENHRS as field name.
- j. Add another reference field to row 7, column 77, with the following information: library RSELABxx, REFMST as file name, RCREF as record name and EHWRK as field name.

Your design page should look something like this:



- k. You may want to create another checkpoint now and save your source.
18. To increase the number of sample records:
- Sometimes it is hard to get a good idea of how a report will really look like. To get a better idea, you can increase the number of sample records.
- a. Select the record EMPDET from the combo box on the toolbar in the design page.
  - b. On the left of the design page at the bottom of the palette, you will see a numeric spinbox, which is set at 0. Increase the number here to see what your report might look like with many records.  
At this point you should ensure that you save your source and create the printer file for this report.
  - c. Press **Ctrl-S** to save.  
Your changes will be save to the host.
  - d. Click the compile button to compile with the default options.

19. To close the Designer:
  - a. Click **File > Exit** from the Designer menu.

You have now successfully created a printer file. You should be familiar with using checkpoints and you should be able to create your own report layouts using CODE Designer.

### **Lesson checkpoint**

You learned the following:

- About checkpoints
- How to create a printer report

## **Module summary**

In this module, you learned how to design screens and reports using the CODE Designer.

### **Lessons learned**

- Open a DDS member for edit with CODE Designer
- Show file-level keywords and record-level keywords
- View the details of records, record-level keywords and field-level keywords
- View the design of the payroll application main menu
- Create a group from an existing record format
- Create a new group and add a subfile record and a subfile control record
- Add columns to the subfile record
- Add fields to the subfile control record
- Copy existing fields
- Set indicators to handle field errors
- View and update record and field properties
- View keywords and the properties of a keyword
- Insert a keyword
- View help for a keyword
- Check there are no semantic errors in the DDS source
- View help for an error
- Launch the editor in read mode from the error list
- Launch the editor in write mode to fix the error
- Find a keyword in the source
- Save source changes
- Compile your source changes
- Create a printer file report
- Close the Designer

### **Assessment**

- What is CODE Designer?
- What is the utility notebook?
- What is the Design page?
- What is the Properties notebook?



- What is a group?
- What can you do in the utilities notebook?
- What can you do on the Design page?
- Why do you want to group records?
- What verifier is used to check DDS source?
- Can you switch between Design mode and Edit mode in CODE Designer?
- How do you open a DDS member for edit with CODE Designer?
- How do you show file-level keywords and record-level keywords?
- How do you view the details of records, record-level keywords and field-level keywords?
- How do you view the design of an application main menu
- How do you create a group from an existing record format
- How do you create a new group and add a subfile record and a subfile control record
- How do you add columns to the subfile record
- How do you add fields to the subfile control record
- How do you copy existing fields
- How do you set indicators to handle field errors
- How do you view and update record and field properties
- How do you view keywords and the properties of a keyword
- How do you view and insert a keyword
- How do you check there are no semantic errors in the DDS source
- How do you launch the editor in read mode from the error list
- How do you launch the editor in write mode to fix the error
- How do you find a keyword in the source
- How do you save and compile source changes
- How do you create a printer file report
- How do you close the Designer

---

## Introducing the product and Remote System Explorer (optional)

This module teaches you about IBM WebSphere Development Studio for iSeries and its relationship to IBM WebSphere Development Studio Client for iSeries. You learn which product makes up the host components and which product makes up the workstation components. You recognize the iSeries application development tools included with Development Studio Client for iSeries programmers. You then are introduced to Remote System Explorer the launching point for iSeries application development tools.

### Learning objectives

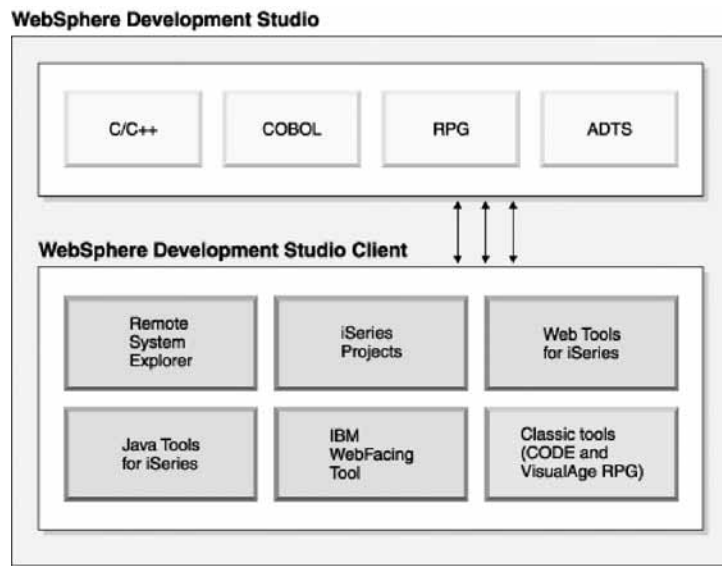
- Know the goals of the product
- Know the editions of the product
- Identify the host tools and the client tools
- List and describe the iSeries application development tools

### Time required

This module should take approximately 15 minutes to complete.

## Introducing Development Studio and Development Studio Client

Development Studio Client is the ideal set of workstation development tools for creating, testing, deploying and maintaining traditional and e-business applications for your iSeries server. Development Studio Client is included with the compiler-based server product, WebSphere Development Studio. The following diagram illustrates the interaction between host and client tools:



Development Studio Client is designed to help you:

1. Develop and maintain iSeries applications using the Remote System Explorer.
2. Develop Web-enabled front-ends to iSeries business logic.
3. Create GUI front-ends to iSeries business logic.

Both Development Studio Client and Development Studio Client Advanced Edition are built on the Rational® Software Development Platform. This platform offers a fresh look and feel for the Eclipse workbench, and helps make it easier for you to build, integrate, and extend your applications. The Rational Software Development Platform offers a tutorials gallery and a samples gallery, to help you get up and running with the product as quickly as possible. The platform also offers user roles, which you can select from bottom-right corner of the Welcome view, (and from **Window** → **Preferences** → **Workbench** → **Capabilities**) that customize and simplify the user interface according to your programming role. These are just some of the new features provided in the IBM Software Development Platform.

The product comes in two editions for iSeries programmers. Both editions of the product are packaged with an additional base Rational product:

- WebSphere Development Studio Client for iSeries inherits and extends the robust, easy-to-use IBM Rational Software Delivery Platform (RSDP) and a subset of IBM Rational Application Developer for WebSphere Software to deliver an integrated development environment (IDE) with tools for developing Web, Web services, client/server, and i5/OS server applications using programming languages like RPG, COBOL, CL, and Java.

- WebSphere Development Studio Client Advanced Edition for iSeries contains all of the development tools included in WebSphere Development Studio Client, plus it inherits and builds on additional premium Web, Enterprise Java Bean (EJB), and J2EE development capabilities from IBM Rational Application Developer. In addition, it provides specific advanced System i™ tools including Navigation support, ClearCase® integration, and a Eclipse-based Screen designer technology preview.

You have reviewed the goals of the product and the product editions.

## Introducing iSeries Application Development Tools

Now, you know what the two flavors are of Development Studio Client and why you would want to use each one. Next let's look at those next generation iSeries server application development tools. What are they and what do they do?

### Remote System Explorer

You can manage your development tasks in the Remote System Explorer. This is an enhanced and more flexible workstation version of Program Development Manager (PDM). You can create and manage development projects on your iSeries system from your Windows-based workstation with the Remote System Explorer and iSeries projects. With these tools, you can connect to an iSeries remote host, view iSeries libraries, files, and members. You can also launch the host compilers, the workstation editor, a program verifier and various debuggers all from the Remote System Explorer. This tool also supports other system types, such as UNIX(R), Linux, and Windows.

### LPEX Editor

Your program editing tasks are simplified with the Remote Systems LPEX Editor. This is a powerful language-sensitive editor that you can customize. Token highlighting of source makes the various program elements stand out. It has SEU-like specification prompts for RPG and DDS to help enter column-sensitive fields. Local syntax checking and semantic verification for your RPG, COBOL and DDS source makes sure it will compile without errors on an iSeries system. If there are verification errors, an Error List lets you locate and resolve problems quickly. On-line programming guides, language references, and context-sensitive help make finding the information you need just a keystroke away.

### Shells and commands in the Remote Commands view

You can use the Remote Commands view to run and interact with commands and command shells on universal systems. A universal system includes Windows, Linux, and UNIX system types. Specifically, you use the view to:

- Run commands in a command shell
- Display and interpret the output of a program
- Enter input to a program
- Display and manage different commands and shells from the same view. Multiple commands can be run in a single shell (one command at a time per shell), multiple shells may be run on a single system, and multiple systems may be running shells.

Whenever a command shell is launched or a command is run from the Remote System Explorer, the Remote Commands view displays the output and provides a way to work with that output.

### **Program Verifier**

One of the most powerful and unique features of the Remote System Explorer is the Program Verifier. Before you compile your code on an iSeries system, you can ensure 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 the iSeries. Think of the host cycles you'll save. It is especially handy when you are writing code but you are disconnected from an iSeries system. You can do this because Remote System Explorer ported the parsing and checking code from the iSeries host compilers to the workstation. The Error List window lists the errors that are found and their severity, display the error messages directly in the source and helps you to navigate between the errors.

### **iSeries Debugger**

With the Integrated iSeries Debugger you can debug an application that is running on an iSeries system. It provides an interactive graphical user interface that makes it easy to debug and test your iSeries programs. It is fully integrated into the workbench. You can also set breakpoints before running the debugger, by inserting breakpoints directly in your source while editing. The Integrated iSeries Debugger client user interface also enables you to control program execution. For example, you can run your program, set line, watch, and service entry point breakpoints, step through program instructions, examine variables, and examine the call stack. You can also debug multiple applications, which may be written in different languages, from a single debug window. Each session you debug is listed separately in the Debug view.

### **CODE Designer**

Using an editor to create and maintain DDS source for your display and printer files can be a frustrating and difficult task. What would be great is a graphical design tool that lets you design your screens and reports visually and then generate the DDS source for you. Well, that's exactly what the CODE Designer does for you.

The CODE Designer interface was designed to help the novice DDS programmer create screens, reports and databases quickly and easily without worrying about the details of the DDS language, while at the same time letting the expert DDS programmer get access to all the features and power of the language. CODE Designer is not fully integrated into the workbench, but you can launch it as a separate tool from the workbench.

---

## **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 iSeries system and edit, verify, compile and debug the payroll application.

## Lessons learned

If you have completed all of the modules, you should now be able to:

- Start the product and open the Remote System Explorer perspective
- Create a connection to an iSeries and select iSeries objects from this connection
- Use the Remote Systems LPEX Editor to edit source
- Verify and compile source in the Remote Systems LPEX Editor
- Debug your interactive payroll application from the workstation
- Customize the Remote System Explorer
- Modify a display file
- Create a printer file
- Recognize the product features and packaging

## Additional resources

### More information

For more information on the product and the Remote System Explorer, see <http://ibm.com/software/awdtools/iseries>.

---

## Notices

© Copyright IBM Corporation 1992, 2007. All rights reserved.

U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

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 or withdrawal without notice, and represent goals and objectives only.

## **Copyright license**

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

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

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 1992, 2007. All rights reserved

## **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
- i5/OS
- iSeries
- Rational
- System i
- WebSphere

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

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

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

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

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

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