

IBM WebSphere Development Studio Client for iSeries

Introduction to Remote System Explorer

Student Exercises

For WDSv V5 GA

Course Code SL19

Claus Weiss/Inge Weiss and the iSeries team

weiss@ca.ibm.com

IBM Toronto Laboratory

3/11/2003 8:18:32 AM

homepage:

<http://www.ibm.com/software/ad/iseries>

newsgroup:

**[news://news.software.ibm.com/ibm.software.web
sphere.code400](news://news.software.ibm.com/ibm.software.web
sphere.code400)**

First Edition March 2003

This edition applies to Version 5.0, GA1 of WebSphere Development Studio Client and to all subsequent releases and modifications until otherwise indicated in new editions.

IBM welcomes your comments. You can send comments to:

IBM Canada Ltd. Laboratory
Information Development
B3/KB7/8200/MKM
8200 Warden Avenue
Markham, Ontario, Canada, L6G 1C7

You can also send your comments electronically to IBM. See "How to send your comments".

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2001, 2002. All rights reserved.

US Government Users Restricted Rights – Use, duplication restricted by GSA ADP Schedule Contract with IBM Corp.

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services or features discussed in this document 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 the 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 in this document. The furnishing of the document does not give you any license to these patents. You can send license inquiries, in writing, to:

Director of Licensing
Intellectual Property & Licensing
International Business Machines Corporation
North Castle Crive, MD – NC119
Armonk, New York 10504-1785
U.S.A

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 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 contain 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:

IBM Canada Ltd. Laboratory
Information Development
B3/KB7/8200/MKM
8200 Warden Avenue
Markham, Ontario
Canada L6G 1C7

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 information 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.

Programming Interface Information

This publication is intended to help you to create and manage applications and user interfaces on the workstation, in a client/server environment. It contains examples of data and reports that are used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses that are used by an actual business enterprise is entirely coincidental. This publication documents General-Use Programming Interface and Associated Information provided by IBM WebSphere Development Studio Client for iSeries.

Trademarks and Service Marks

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

Application System/400
AS/400
AS/400e
DB2/400
IBM
iSeries
Integrated Language Environment
OS/400
RPG/400
VisualAge
WebSphere

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

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

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

Overall Lab Guide

The objective of the Remote System Explorer (RSE) Introduction is to explore some of the basic features of the tools that make up RSE, the RSE tree view, LPEX Editor, CODE Designer, and Debugger as well as experience how RSE is integrated into the WebSphere Development Studio Client (WDS) workbench. The example of a small payroll application is used to walk you through the edit, compile, debug cycle. At the end of the lab, the student should know how to:

- Create connections to iSeries servers.
- Navigate in the RSE tree view
- Open source in the Editor, make changes, and use different ways to maneuver through the source.
- Invoke the Program Verifier and work with the Error list.
- Start a host compile using the remote compile capabilities.
- Use the DDS tree, the Design Page and the Properties Notebook.
- Invoke the debugger, set breakpoints, display and change variables, and control program execution.
- Create different filters in the Remote Systems Explorer, invoke actions and create new user actions.

The exercises for each tool, Editor, Designer, Debugger, and Remote Systems Explorer should be worked on in sequence. You can switch to a different tool though, without completing all exercises for the previous one.

Note: *The pictures in these labs show a similar application being built. Some of the names and icons may be different from the environment you are working with.*

Prerequisites

Although not required, familiarity with the RPG language is an asset.

Also, it is helpful if the student is familiar with basic MS Windows operations such as working with the desktop and basic mouse operations such as opening folders and performing drag-and-drop operations.

Remote System Explorer hands on Lab

The Steps in this Lab

- Start WDS and use RSE perspective to connect to your iSeries server
- Open a source member in the LPEX source editor
- Work with this source using editor features
- Verify the source with the built-in program verifier
- Compile the source by using the remote command capabilities in RSE
- Run the application from the RSE environment
- Use CODE designer to change a DDS display file source member
- Create a display file directly from CODE Designer
- Use the integrated Debugger, set breakpoints, display and change variables, control program execution
- Create different filters in RSE
- Create a user action in RSE

Table of content

Notices	4
Programming Interface Information	5
Trademarks and Service Marks	5
Overall Lab Guide	6
Prerequisites	6
The Steps in this Lab	7
EXERCISE 1: Introduction to WebSphere Development	
Studio	11
Installing WebSphere Development Studio Client	12
Using the WebSphere Studio workbench to invoke the CODE Editor	12
Create a Connection	15
Selecting an iSeries object in the RSE perspective	17
Opening a second source member	24
Using the outline view	24
Basic Editor Features	26
Column Sensitive Editing	27
SEU Commands	29
Undo and redo	29
Using Language-Sensitive Help	30
Using Prompts	32
Indenting Source	34
Find and Replace	36
Filtering Lines	37
Using Filters	38
Comparing Files	40
This compare file section is work in progress	40
Syntax Checking	40
Verifying Your Source	42
Remote compile	45
The iSeries table view with Command entry field	48
Interactive Connection to the iSeries	50
Running the PAYROLL Program	52
Open a DDS display file member using the WebSphere workbench	55
The DDS Tree	56
The Details Page	57
The Design Page	59
Creating Groups from Existing Records	60
Creating New Screens	62
Field Creation and Design Page Toolbar	63

Switching between Multiple Records	65
Copy and Paste	66
Working with Indicators	67
The Properties Notebook.....	69
Adding New Keywords	71
Verifying Your Source.....	73
Switching between Design mode and Edit mode.....	74
Compiling Your Source	75
Closing CODE Designer	75
Starting the Distributed Debugger.....	76
Setting Breakpoints	81
Monitoring Variables.....	83
Running a Program	85
Stepping into a Program	85
The Call Stack	87
Setting Breakpoints in PAYROLLG.....	87
Removing a Breakpoint	88
Running the Program.....	88
Monitoring Variables in PAYROLLG.....	89
Adding a Storage Monitor.....	90
Watch Breakpoints	92
Stepping Through the Program.....	94
Closing the Debug Session.....	94
Creating a library filter	96
Creating an object filter	99
Creating a user action.....	104
Running commands from the RSE.....	110

EXERCISE 1: Introduction to WebSphere Development Studio

The **IBM WebSphere Development Studio for iSeries** product is a suite of e-business enabling technologies including host and workstation components :

Host Components

- ILE RPG
- ILE COBOL
- ILE C
- ILE C++
- Application Development ToolSet (ADTS) includes PDM, SEU, SDA, RLU

Workstation Components (WebSphere Development Studio Client, WDSclient)
WebSphere Studio Site Developer

+ iSeries Plugins

- Remote Systems Explorer
- WebFacing

Classic Tools

- VisualAge RPG
- CODE

The **CoOperative Development Environment**, better known as **CODE**, is a set of integrated development tools that allow you to: create, edit, compile, and maintain your source code; debug programs using a PC connected to an iSeries. In this Lab you will only use the CODE Designer tool, all other tools used are workbench tools

The Development Studio workbench includes the following tools features for the iSeries programmer:

- Remote System Explorer (RSE)

An enhanced and more flexible workstation version of the Program Development Manager (PDM). It ties all iSeries objects together and allows you to quickly access all the objects on iSeries and to effectively manage and organize your development projects.

- LPEX Editor

A powerful language-sensitive editor that is easy to 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 cleanly the first time on an iSeries. 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.

- **iSeries command interface to manage iSeries objects**
An interface that allows you to submit requests to the iSeries to invoke actions like compile, bind, or build objects on the host.
- **CODE Designer**
A tool with a rich graphical user interface that makes designing or maintaining display file screens, printer file reports, and physical file databases easy and fun.
- **iSeries Debugger**
A source-level debugger that allows you to debug an application running on a host iSeries from your workstation. It provides an interactive graphical interface that makes it easy to debug and test your host programs. It is fully integrated in the workbench.

In this session, you will learn some of the basic features and functionality of each of these tools. We are confident that Development Studio Client will save you lots of time and effort in your day-to-day programming tasks. It will make you a more efficient and effective programmer. At the same time, it will save cycles on your iSeries. Now let's spend a couple of hours playing and see if you agree.

Installing WebSphere Development Studio Client

The WebSphere Development Studio Client for iSeries (WDS*c*) product consists of two parts:

1. The integrated tools in the Eclipse based workbench
2. The Classic tools CODE and VARPG

The installation is done on the workstation from:

1. a local CD drive
2. a LAN drive (assumes that an installable image has been set up on the LAN)
3. an iSeries (assumes that the workstation files have been copied to the iSeries IFS).

This workstation install uses the Windows Installer.

The minimum hardware requirements for WebSphere Development Studio are an Intel® Pentium III processor or faster with 512 MB of memory, a SVGA 800 x 600 monitor, CD-ROM drive, and a mouse or pointing device. An install of **WDS*c*** uses about 1.6GB of disk space.

Using the WebSphere Studio workbench to invoke the CODE Editor

In this Lab you will use the WebSphere Studio workbench to access iSeries objects, and then work with RSE tools from the workbench.

Invoking WebSphere Development Studio client (WDSclient)

To start Development Studio Client:

- Click the **Start** button on the task bar of your desktop
- Select **Start** → **Programs** → **IBM WebSphere Studio** → **IBM WebSphere Development Studio Client (Advanced)**

Note: If the **Advanced Version** of the product is not installed on your workstation then you will not see the word **advanced** in the Start menu

A dialog will appear

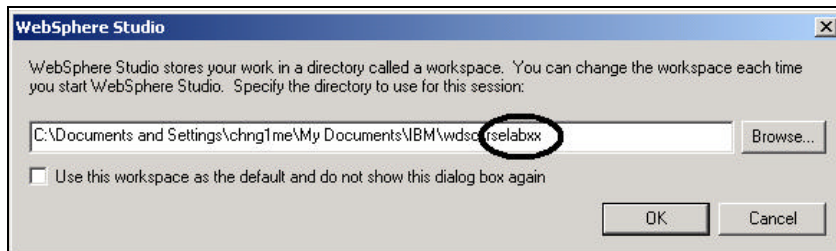


Figure 1: WebSphere Studio dialog for specifying workspace directory name

- Change the **Entry field** in this dialog and use directory name **rselabXX** (where XX is your team number)
 - If your team 01 use directory name rselab01 as shown in Figure 1

After a few moments of loading, the workbench appears

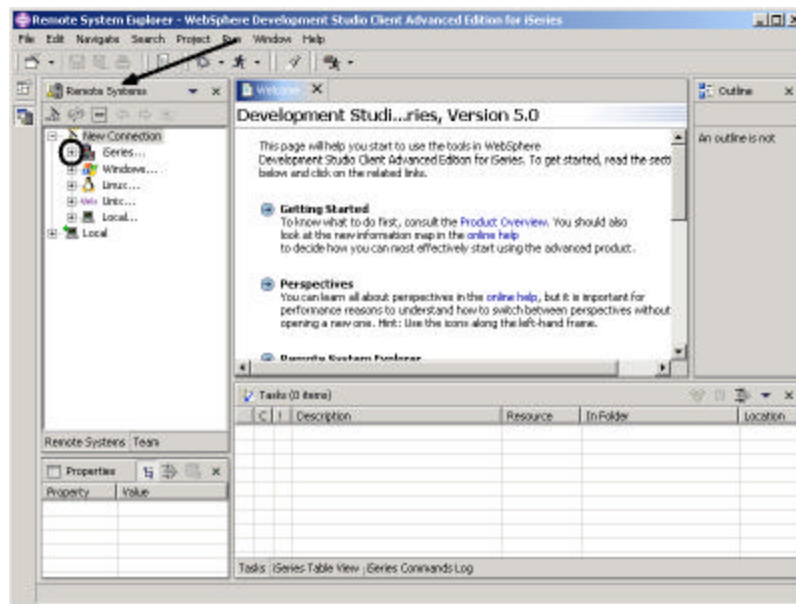


Figure 2: Workbench with Remote System Explorer

You will be working with the **Remote Systems Explorer (RSE)** perspective in the workbench. This perspective will be the active perspective when you start Development Studio Client with a fresh workspace, if you had used the workspace before then the workbench would come up with the image that it was last exited from.

A perspective is a specific arrangement of views and tools in the workbench, depending on what role the workbench user has, he/she will use a different perspective. A web developer will use the **web perspective**, a Java developer will use the **Java perspective**, an iSeries developer will use the **Remote Systems Explorer perspective**.

- Check for the name of the perspective, the arrow in Figure 2 indicates where to look for the perspective name.

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

- Select the **Window** option on the workbench Menu bar
- Select the **Perspective** option on the sub menu
- Select **Remote System Explorer**

Now you are ready to create a connection

Create a Connection

Go thru the following steps to create a connection to the iSeries server you will work with in this Lab.

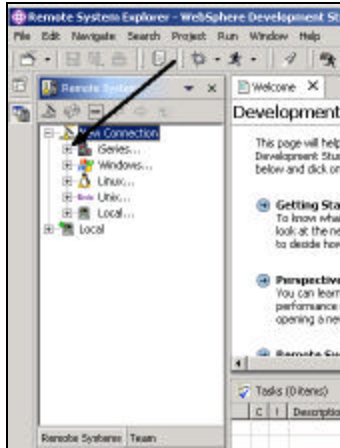


Figure 3: Create a new connection

- Locate the *New Connection* node.

If it is not expanded already

- Expand the *New connection* node by clicking on the + sign beside it
- Locate the *iSeries.....* node
- Expand the *iSeries* node by clicking on the + sign beside it

The new connection wizard will appear:

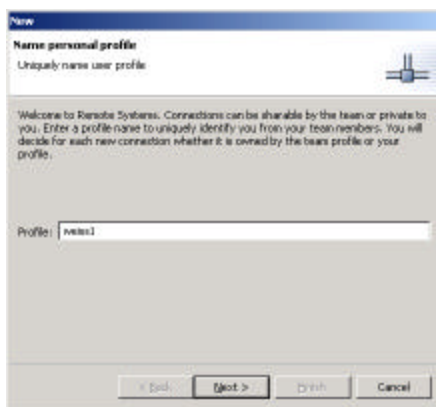


Figure 4: New connection wizard

Leave the *Profile* information as is:

- Click the *Next>* push button

The second page of the *New connection* wizard will show:

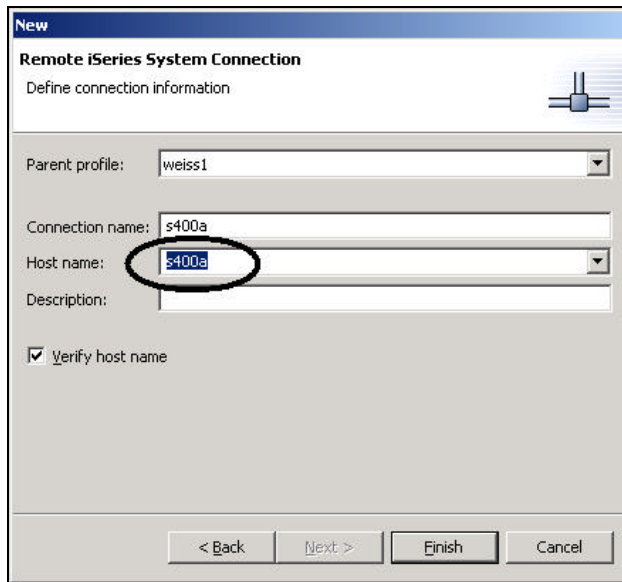


Figure 5: The create connection dialog

The cursor in the **Define Connection Information** dialog should be positioned in the **Host Name** entry field

The name of the host name you are using is **s400a** (or what your instructor tells you)

If a different host name is used in your Lab, the instructor will let you know

- Enter **s400a**, or the name given to you by the instructor in the **Host name** entry field

The **Connection name** will be automatically filled with the host name, leave it the way.

- Leave the **Parent profile** as is, don't change it.
- Leave the **Verify host name** check box checked
- Click the **Finish** push button.

Selecting an iSeries object in the RSE perspective

You will notice that the tree view in the Remote System view is now updated with this new connection. There are several new nodes listed under the new connection node.

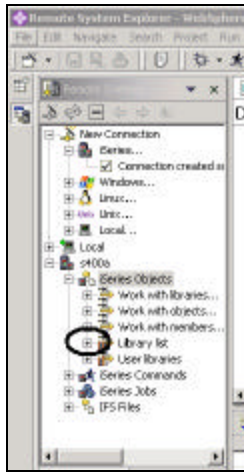


Figure 6: RSE tree view with iSeries connection

You notice the first three nodes in the tree view under the iSeries objects node are named after the PDM options, because they have similar capabilities

1. Work with libraries
2. Work with objects
3. Work with members

In addition you have a node for working with library lists and user libraries

4. Library list
5. User libraries

You also have more nodes to work with under the connection node itself

1. iSeries Commands
2. iSeries job
3. IFS Files

I want you to work with a library in your library list:

- Expand the **Library list** node, by clicking on the + sign beside it.

Now the connection will be activated and you will be prompted for userid and password



Figure 7: Authentication dialog

- Enter Userid **WDSCLABxx** (xx being your team number)
- Enter password **WDSCLABxx** (being your team number)
- Check the check box **Permanently change user id**
- Check the check box **Save password**
- Click the **OK** push button

The **Userid** has been setup, so that your group/team library has been added to the library list automatically. Back in the workbench in the RSE perspective you will see the libraries in your job's library list.

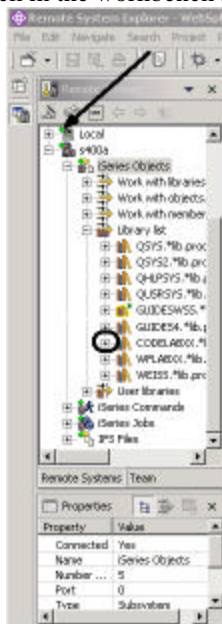


Figure 8: RSE with libraries in library list

Notice that the s400a node now got a small green arrow in the icon to indicate it is an active connection

Also your **CODELABxx** library should have a small green star that indicates it is the current library

In Figure 8 the library **GUIDESWSS** is the current library, if you have sharp eyes you can spot the star in the icon.

Now to proceed in your exercise:

- Expand library **CODELABxx**, (**xx** being your group number)

You will see all objects in this library appear in the tree view.

- Locate the **QDDSSRC** source file and expand it.
- Locate the **QRPGLESRC** source file and expand it as well.

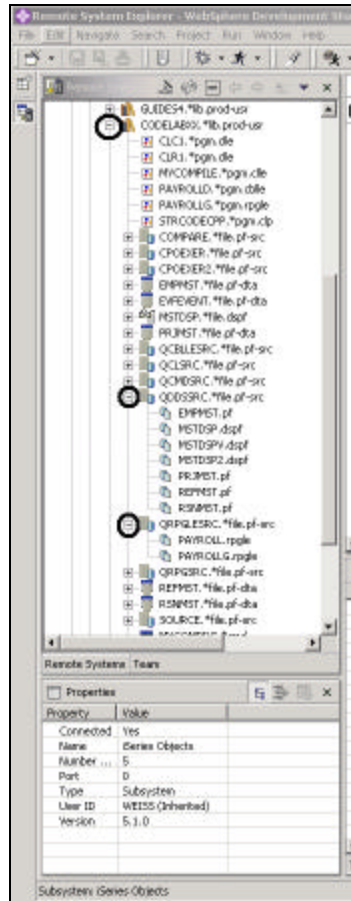


Figure 9: RSE tree view with expanded source files

Now you can see and access the **members** in these 2 source files.

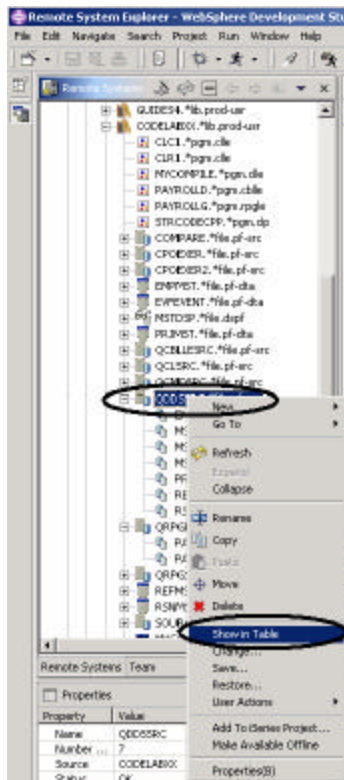


Figure 10: Select table view for QDDSSRC file

Before you go ahead and work with these members, I suggest you have a look at the table view as well because that is more similar to the view you are used to from PDM:

- Right mouse click on the **QDDSSRC** file
- Select the **Show in Table** option in the pop up menu

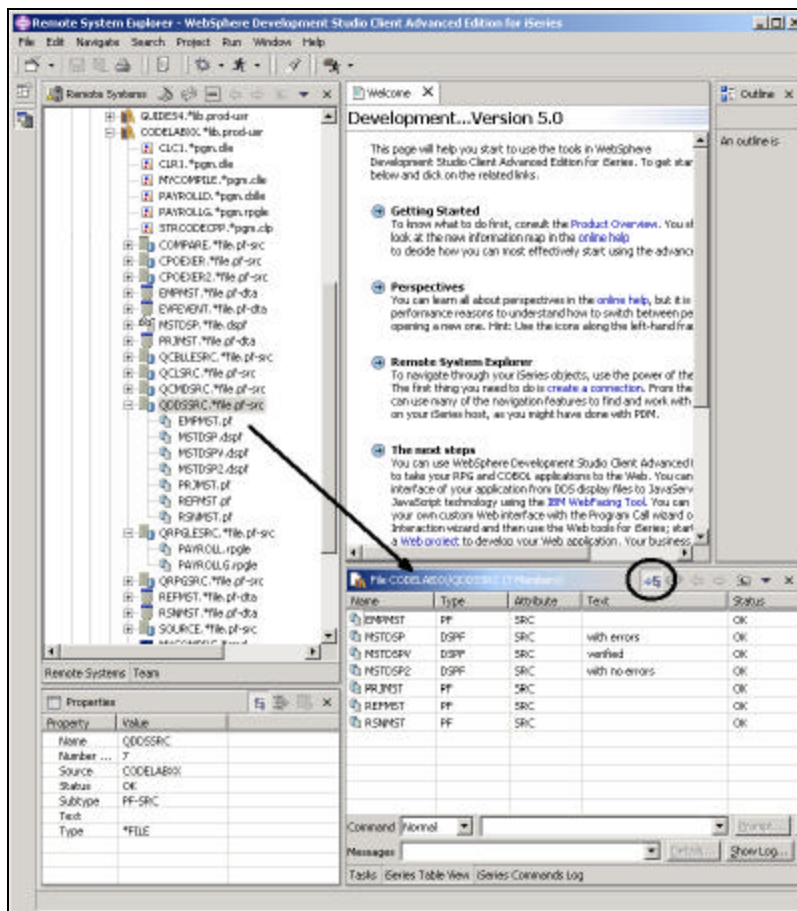



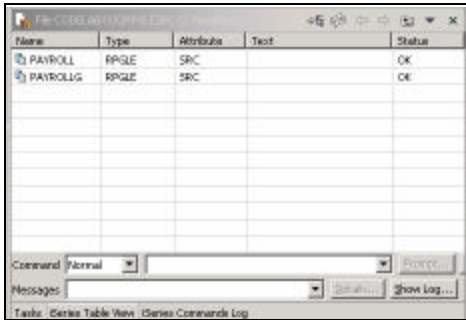
Figure 11: Table view with lock/unlock button

In the table view as shown in Figure 11:

Tip: You can sort the view by column, by clicking on the column's heading.

You can also unlock this table view, so it switches to the selected object in the tree view, as you select objects in the RSE tree view.

- In the table view toolbar click the lock/unlock button , since it is in lock position by default it is now unlocked from the current QDDSSRC object. Meaning it is not locked to QDDSSRC.
- In the tree view, select QRPGLSRC, now the table should show the members in QRPGLSRC



Name	Type	Attribute	Text	Status
PAYROLL	SPGL	SRC		OK
PAYROLLG	SPGL	SRC		OE

Figure 12: Table view for QRPGLSRC

To edit a source member MSTDSP, go to the tree view

- Select QDDSSRC
- Double click on member **MSTDSP** in the QDDSSRC source file.

Tip: You can do this in the tree view or in the table view

The source editor will open

Tip: If you want to start the CODE editor instead, you can right mouse click on the source member and select the **Open with CODE editor** menu option

The screenshot shows the Remote System Explorer interface. The main window is titled 'Source Editor' and displays the following code:

```
Row 1 Col 1 Replace
000100 A**STS DD 20020613 153333 QUSER REL-95.1 iSeries VDT
000200 A* This is the DSFF use for the CODE Introduction 2 day course
000300 A* 5763CL2 (C)Copyright 1998
000400 A*****
000500 A* THIS DISPLAY FILE PROVIDES MAINTENANCE FORMATS FOR ALL THE
000600 A* TIME REPORTING MASTER FILES - EMPLOYEE MASTER
000700 A* - PROJECT MASTER
000800 A* - REASON CODE MASTER
000900 A*****
001000 A*Created by BATHIER
001100 A**MFD with errors
001200 A**REC
001300 A DEPSIZ(24 00 *DS3)
001400 A REF(OGPL-QINVREC INVREC)
001500 A PRINT
001600 A INDATA
001700 A CA03(03 'end of job')
001800 A CA04(04 'return to maintenance sele-
001900 A ction')
002000 A*
002100 A* THE SELECT FORMAT ALLOWS SELECTION OF THE TIME REPORTING
002200 A* MASTER FILE THE OPERATOR WANTS TO MAINTAIN
002300 A R SELECT
002400 A**STS DD 20020613 153333 QUSER REL-95.1 iSeries VDT
002500 A**
002600 A BLINK
002700 A ALARM
002800 A 2 5 'PRG01'
002900 A 2 30 'Time Reporting System'
003000 A 2 70DATE(ATT)
003100 A EDTCDE(Y)
003200 A 3 30 'Maintenance Selection'
003300 A 3 70TIME
003400 A 6 14 'Enter an X beside the application -
003500 A you want to maint-ain'
003600 A 9 28 'Employee Master Maintenance'
003700 A PRJAFL 1A B 10 25
003800 A 10 28 'Project Master Maintenance'
003900 A RSNAPL 1A B 11 25
004000 A 11 28 'Reason Code Master Maintenance'
004100 A EMESS 50A O 21 16
004200 A 23 7 'F3-End of Job'
004300 A EMPAPL 1 B 9 25INDXT(01 '7')
004500 A**MGP MAINPANEL 01
004600 A**
004700 A* THE EMPSEL FORMAT ALLOWS SELECTION OF THE EMPLOYEE TO BE
004800 A* MAINTAINED AND THE TYPE OF MAINTENANCE TO BE PERFORMED
004900 A R EMPSEL
005000 A**STS SD 19920629 165055 SNYTH REL-V2R2M0 S738-PU1
005100 A**
005200 A BLINK
005300 A ALARM
005400 A 2 5 'PRG01'
005500 A 2 30 'Time Reporting System'
```

Figure 13: Source editor with DDS member

You can maximize the editor window by double clicking on its window heading, as shown by the arrow in Figure 13. You can get it back to its original size, by double clicking on the heading again.

Opening a second source member

We want you to open a second member in the editor, so back in the RSE perspective

- Double click on member **PAYROLL** in the QRPGLSRC source file.

This member will be loaded into the editor as well.

Your editor window will look something like this, notice the two window bars with both editor window open, switch from one editor session to the next by clicking on the window bars.

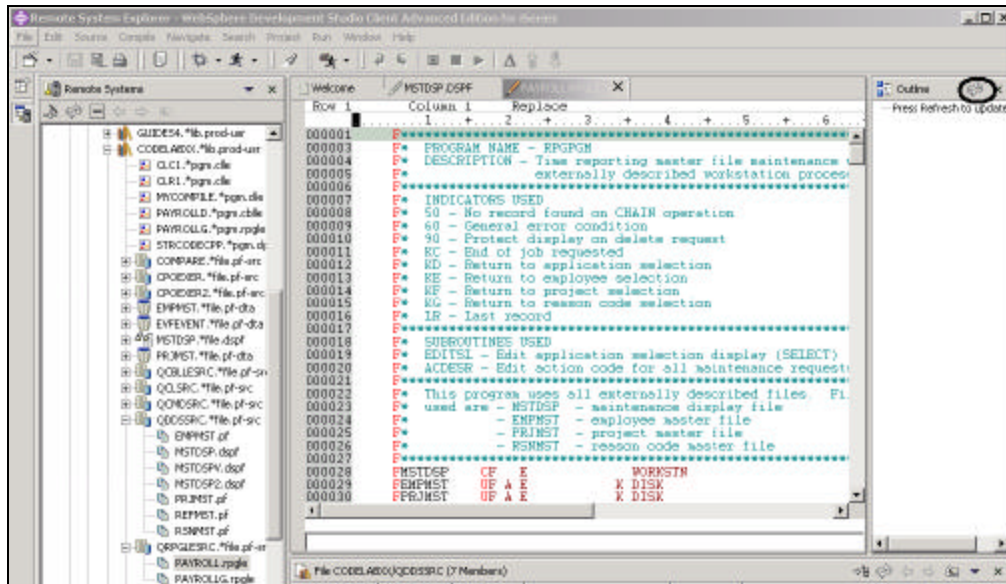


Figure 14: Editor with RPG source member and Outline view frame

Using the outline view

To see an outline view of your RPG source,

- Click on the **Refresh** button on the Outline view toolbar, as shown by the circle in Figure 14

You can see, the Outline view contains your source program in a tree view without the lines containing logic.

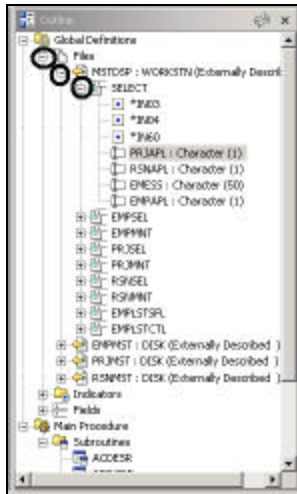


Figure 15: Outline view with file and record format expanded

To get some more details of your source member:

- Expand the **Files** node
- Expand the **MSTDSP** workstation file
- Expand the **SELECT** record format

Double clicking on any of the icons in the outline view will position the source editor accordingly.

Now if you want to switch back to a different member loaded in the source editor just click on its window heading and the source editor window with the correct member comes into focus

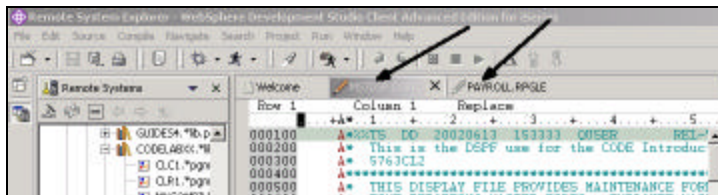


Figure 16: Click on the editor window heading to give focus to the edit session

To switch back to the DDS source member:

- Click on the Window heading to get **MSTDSP** editor window in focus

Make sure you are back at the **RPG member** for the next exercise.

Now you are ready to exploit the editor's basic features.

Move on to the next section:

Editor Basic Features

EXERCISE 2: Working with the Source Editor

Basic Editor Features

The LPEX Editor has all the basic functions that you would expect in any serious editor:

- Cut, copy, and paste
- Block marking of lines, characters, or rectangles with copy, move, and delete operations.
- Powerful find and replace functionality.
- Unlimited undo and redo.
- Automatic backup and recovery.

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

- Token highlighting -- different language constructs are highlighted using different colors and fonts 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 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.
- For column-sensitive languages there is a command that simplifies text insertions and deletions.
- On-line language reference help.

Now let's take a minute to try a few of these features.

Column Sensitive Editing

Working with the RPG source member PAYROLL, which should be already open, try some of these features by following the instructions in this guide. You might want to maximize the editor window during this exercise.

LPEX provides special support for insertion and deletion in column -sensitive languages. You can switch this support on by going to the workbench preferences dialog:

- Select the **Window** option on the workbench menu bar.
- Select the **Preferences** option on the pull down menu.

The Preferences dialog appears

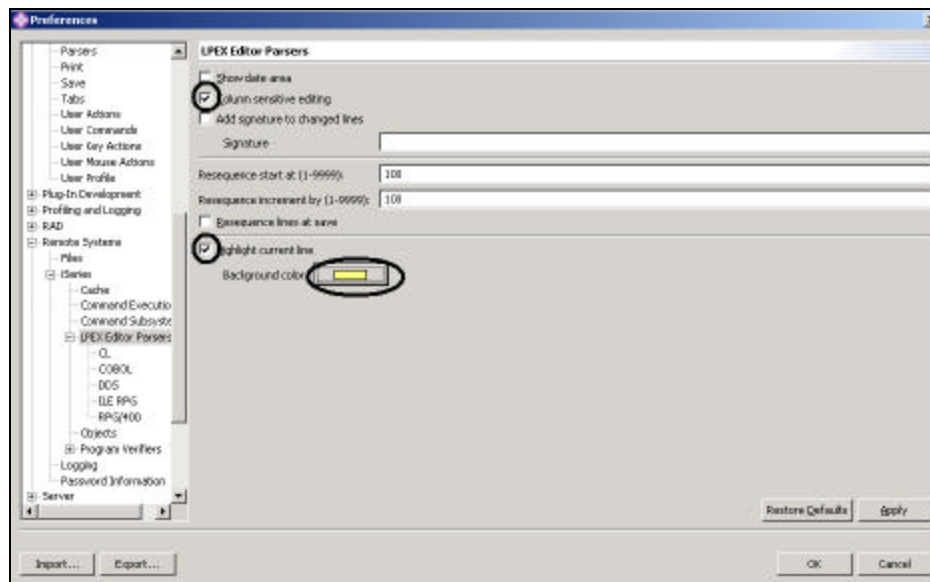


Figure 17: Preferences dialog

In the tree view in the left pane:

- Expand the **Remote Systems** node
- Click on the **LPEX Editor Parsers** node

The right pane of the dialog will allow you to set preferences for this feature:

- Select the **Column sensitive editing** check box
- Select the **Highlight current line** checkbox
- Click on the **Background Color** push button
- Select a **light yellow** from the color palette
- Click the **OK** push button on the color chooser dialog

Tip: Other interesting preference settings are located under the LPEX Editor Parser node, when you expand this node you see preference settings for the individual language environments.

- Click the Ok push button on the Preferences dialog

Now let's see what this does.

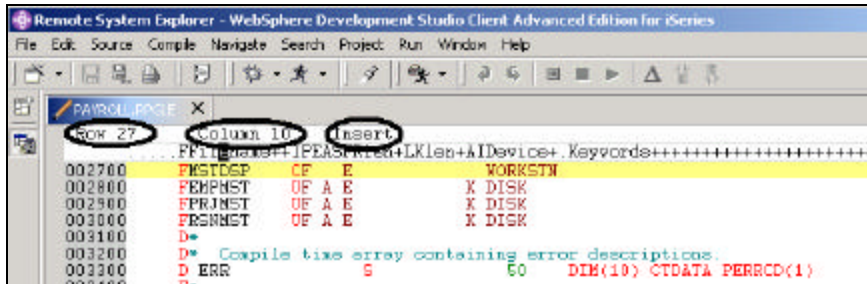


Figure 18: Editor in Insert mode with highlighted line 27

In the editor window:

- Move the cursor to **line 27, column 10**.
- Make sure the editor is in **Insert** mode.

If the status area says '**Replace**':

- Press the **Insert** key.
- Press the **spacebar** 3 times.

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

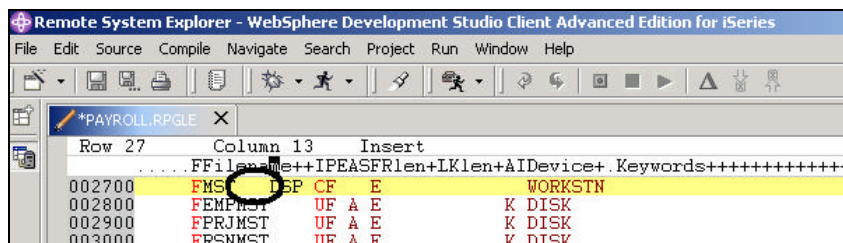


Figure 19: Inserted blanks in file name, none of the other columns has been shifted

- Press the **backspace** 3 times.
- Once again the filename is affected but no other columns are touched.

SEU Commands

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

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

Notice that the lines have been deleted.

- Now type *i5* in the sequence number area.

Make sure the cursor is within the sequence number area

- Press **Enter**.

Five new lines are inserted.

Undo and redo

Now you are going to undo some of the changes you just made to the file.

- Select the **Edit** option from the workbench menu bar
- Select the **Undo** option from the pull down menu

Notice that the 5 new lines disappear.

- Do another undo action by pressing **Ctrl+Z**.

Notice that the deleted lines reappear.

- Select the **Edit** option from the workbench menu bar
- Select the **Redo** option from the pull down 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. To do this:

- Select the **File** option on the workbench menu bar
- Select the **Close** option from the pull down menu

A dialog will appear asking you to save the latest changes.

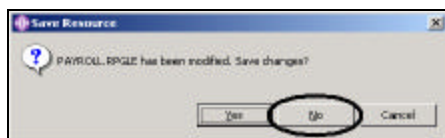


Figure 20: Confirmation dialog

- Click the **No** push button.
- Go back to the workbench to the *RSE perspective* and reload the **PAYROLL** member in the **QRPGLESRC** file

Using Language-Sensitive Help

Inside the editor, there is cursor-sensitive language-reference help available. This help is invaluable if you cannot remember the order of fields in an RPG specification or the possible values for a variable field. This help is available from the LPEX Editor window.

- Position the cursor over the word **MOVE** in line **56** of the ILE RPG source.
- Press **F1**.

Language-sensitive help for the MOVE operation code appears in a help window.

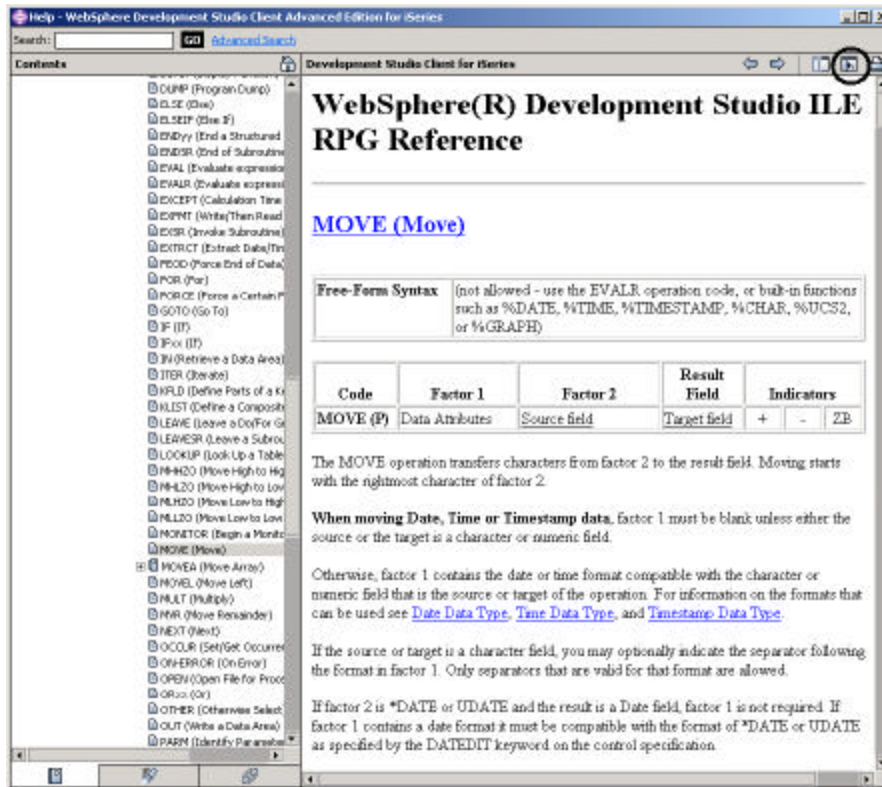


Figure 21: Help for MOVE and synchronize button

- Click on the **Synchronize Navigation** button on the Help window toolbar. This will synchronize the topics in left overview pane to the help topic you are viewing in the main help pane.
- Play around in the help window to see what else is available.
- Minimize the Help window.
- Select the **Help** menu option on the workbench menu bar, to see what help is available.
- Select the **Help contents** menu option from the menu pull down.

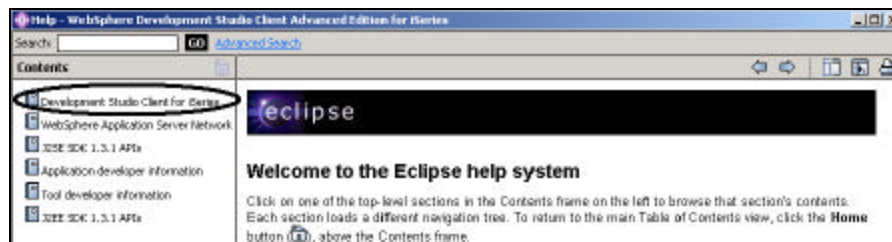


Figure 22: Main Help window

- Select **Development Studio Client for iSeries** from the topic list in the left pane

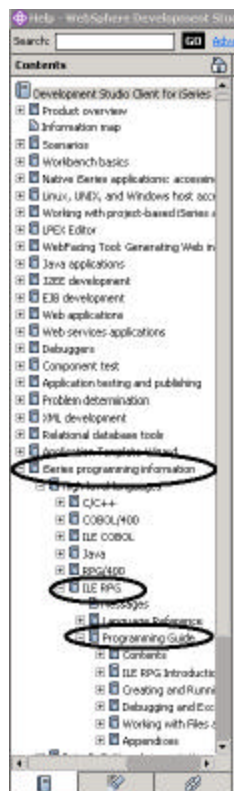


Figure 23: Finding the RPG IV manuals

- Expand the **iSeries programming information** node
- Expand the **ILE RPG** node
- Expand the **Programming Guide** node

Here you find the table of content and can select the topic your interested in

Having the latest version of the manuals at your fingertips will make your life easier

Using Prompts

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

- Move your cursor to *the D-spec* on line 33.
- Select the **Edit** option from the workbench menu bar
- Select **Prompt** (or press F4).

The editor window will shrink back and allow you to see the prompt window at the bottom of the workbench. The prompt window shows the specification line broken down into its individual fields.

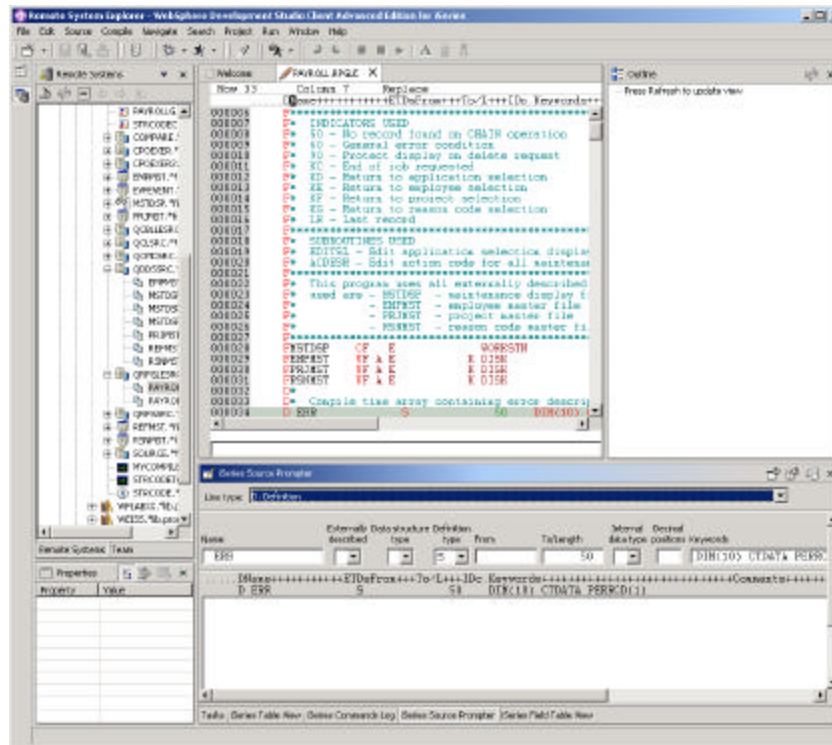


Figure 24: Prompt window

To display context sensitive help for any field in the prompt window,

- Move the cursor to the **Keywords** field using the Tab key.
- Press **F1** to see help for this field.

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

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

- Click on any *link* to see specific help for that item.
- Minimize the *Help* window.

To get to the prompt window from the maximized editor window without bringing the size of the editor window down, I suggest you park the prompt window on the left task bar as a Fast view.

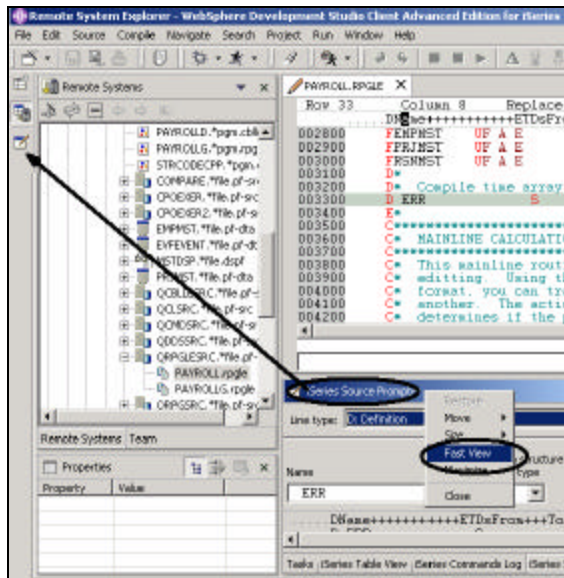


Figure 25: Prompter window moved to fast view on taskbar

- Right click on the prompt window heading
- Select the *Fast view* option from the pop up menu

You will see the prompt window icon on the task bar on the left hand side of the workbench

- Maximize the editor window
- Select a line in the editor window and press **F4** to prompt for a line

The prompt window will be placed on top of the editor window

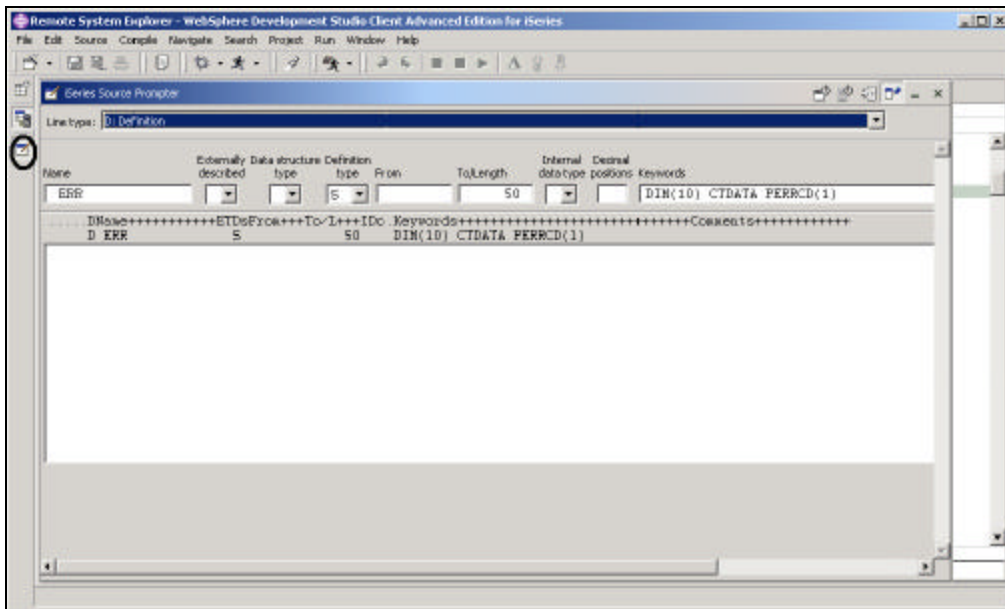


Figure 26: Prompt window on top of editor window

- Click on **Prompter icon** on the task bar to minimize the prompt window

Note: On the prompt window toolbar you can use the three push buttons to: disable prompt view, disable syntax checking, and toggle insert mode

Indenting Source

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

- Select the **Source** option from the workbench menu bar
- Select the **Show Indentation** option from the pull down menu

The indented view is displayed at the bottom of the workbench.

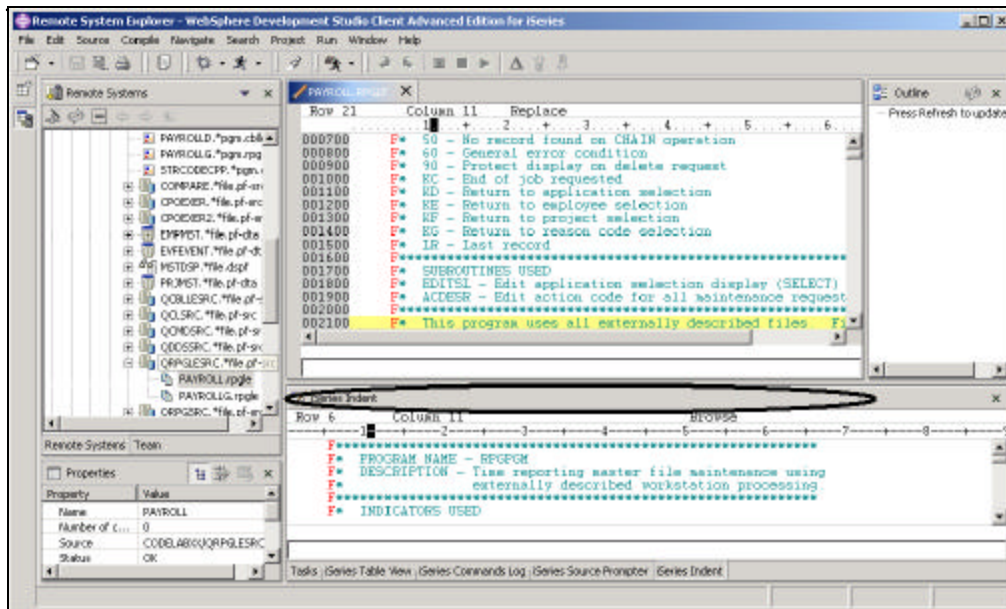


Figure 27: Indent view at bottom of workbench

To get the indented view displayed as full view,

- Double click on its **window heading**.

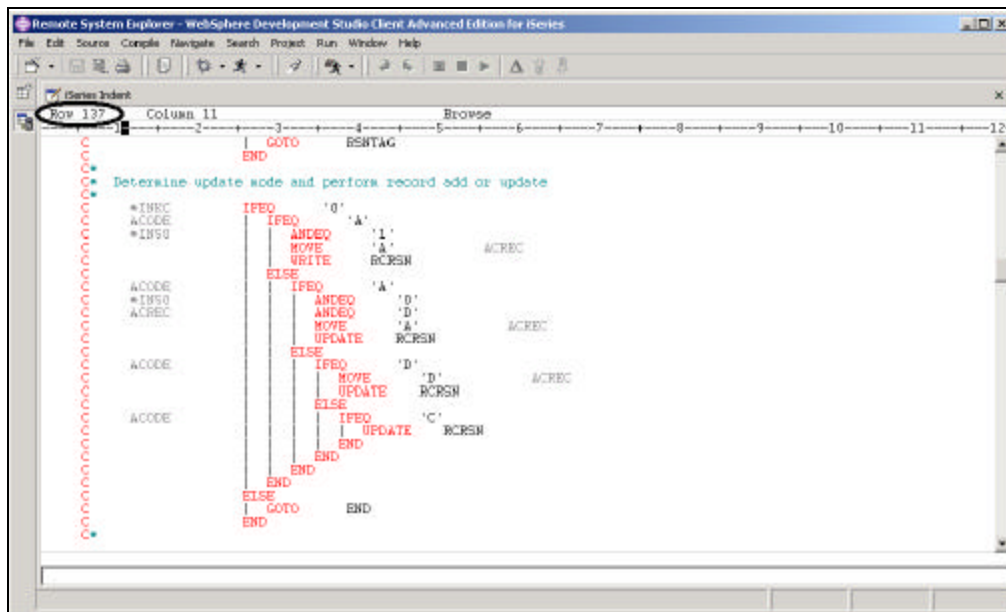


Figure 28: Maximized Indent view

- Scroll down to **line 137**

In this area you see some nested conditions with indented lines, as you will notice this helps to recognize the beginning and ending of these conditions.

Tip: You can move this view as a Fast view to the left task bar as well, if you plan to use this view frequently.

Note: The **INDENTED** view is Browse mode only and cannot be edited.

- Switch back to the editor window with the *PAYROLL* program

Find and Replace

The LPEX editor also has a powerful find and replace text feature.

- Press **Ctrl+Home** to go to the top of the file.
- Select the *Edit* option from the workbench menu bar
- Select the *Find/Replace* option from the pull down menu (or **press Ctrl+F**).

The Find /Replace dialog appears at the bottom of the editor window.

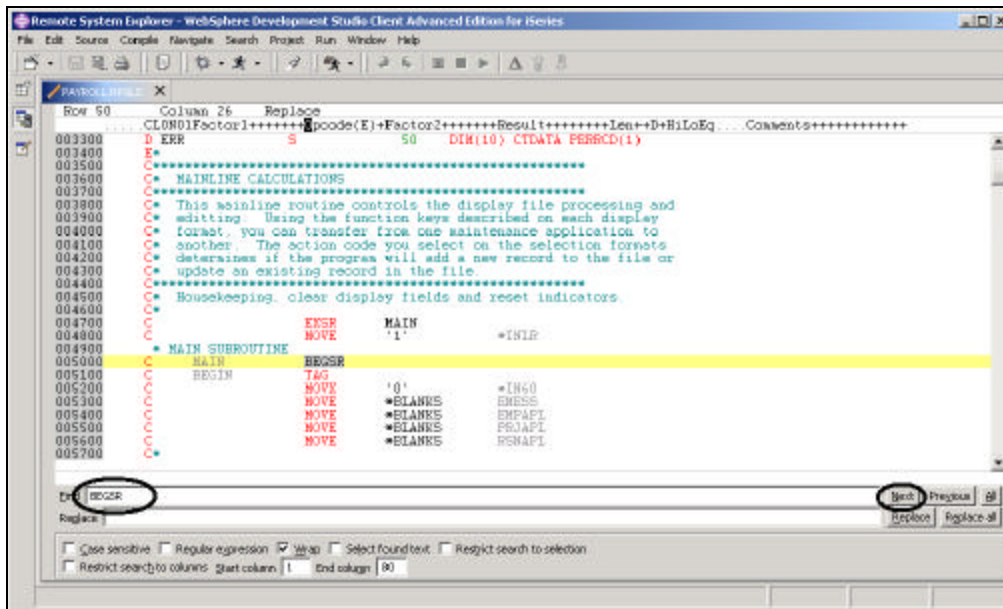


Figure 29: Find/Replace dialog

At the bottom of this dialog, you will notice that you have some options to select from i.e., search only in certain columns, etc. I want you to find the first occurrence of **BEGSR**,

- In the *Find* entry field, enter **BEGSR** to find the start of a subroutine. Make sure the *Replace* entry field is blank. You would use this field for text replacement.

You will notice that the editor moves the active line, to line 50, which contains the first **BEGSR** phrase in the file.

- Click the *Next* push button (or Press **Ctrl+N**) to go to the next location of **BEGSR** in the file.

Filtering Lines

The CODE 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.

I want you to filter for the operation code BEGSR so you can see all lines in your source code containing the phrase BEGSR



Figure 30: Selected operation code BEGSR

- Select the **BEGSR** operation code in the edit window
- Select the **Edit** option from the workbench menu bar
- Select the **Filter Selection** option from the pull down menu

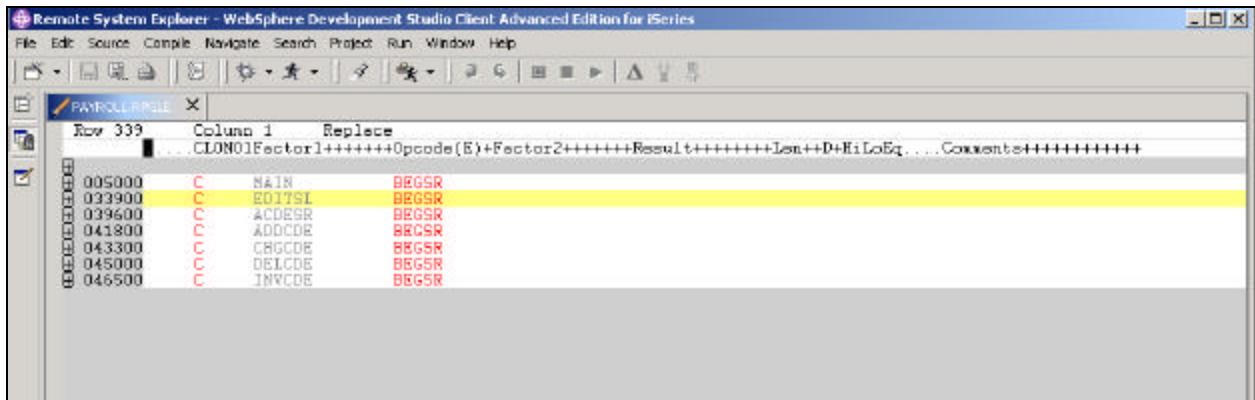


Figure 31: Editor window showing all lines with BEGSR phrase

- Move the **cursor** down **two lines**, to line 339
- Click on the **plus** beside the line to expand this section

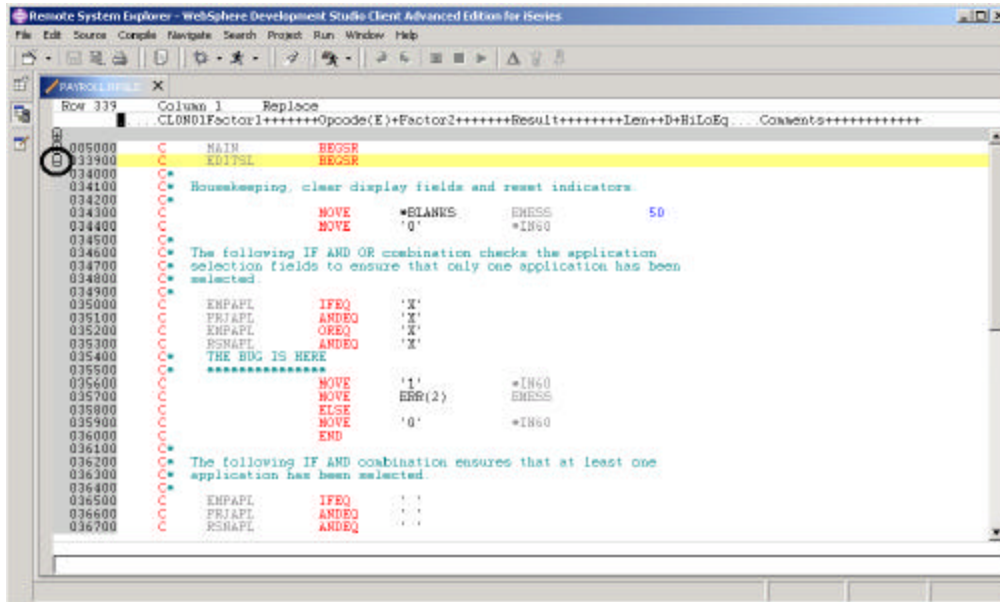


Figure 32: Section expanded

Show all of the source again by:

- Selecting the *Edit* option on the workbench menu bar
- Selecting the *Show all* option or by pressing **(Ctrl+W)**.

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

Using Filters

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

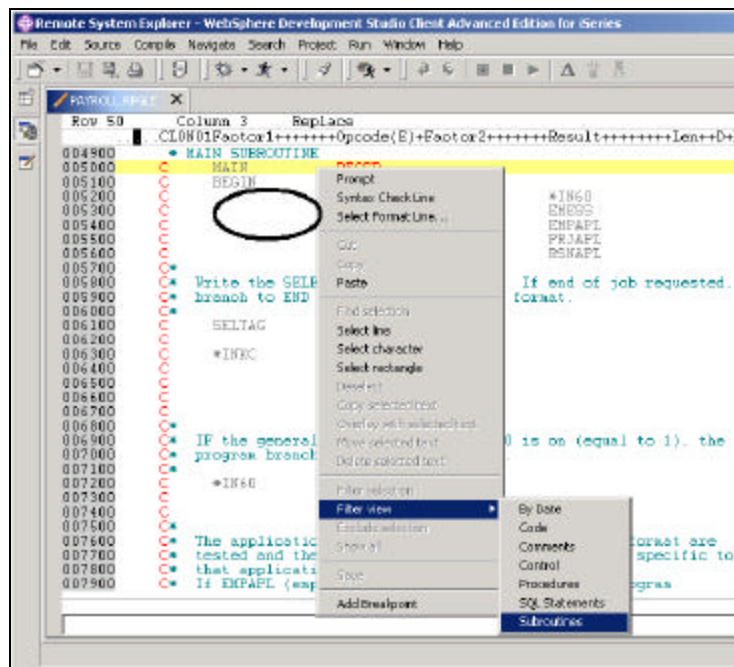


Figure 33: Editor pop up menu with Filter sub menu

- Right click in the *editor window*
- Select the *Filter view* option from the pop up menu
- Select the *Subroutines* option from the sub menu.

All subroutines specifications are displayed allowing you to move quickly and easily to the area in your file where the desired subroutine is.

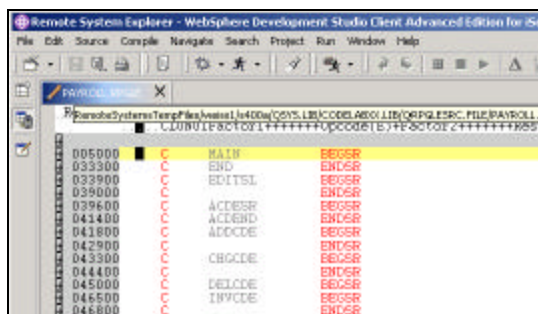


Figure 34: Subroutine filter in action

- Move your cursor to the line with the subroutine declaration **CHGCDE**. (line 433)
- Show all lines in this subroutine by clicking on the **+ sign** beside it

Now you can work with the source inside this subroutine.

Comparing Files

This compare file section is work in progress

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. You can edit your source directly in the utility -- it contains all of the CODE Editor's features.

Switch back to <OS400>CODELABxx/QRPGLESRC(PAYROLL) using the blue drop-down list beneath the toolbar.

From the **Actions** menu, select **Compare...** The **Compare** dialog appears.

In the entry field, type

<>CODELABxx/QRPGLESRC(PAYROLLG)

where <> means use the default host, **xx** is your workstation number. PAYROLLG is a version of the PAYROLL file where some compile errors have been corrected.

Click on the **Compare** push button. The editor now has the PAYROLL member loaded on the left and member PAYROLLG loaded on the right. In between the two members are two long vertical blue lines with horizontal yellow and red bars highlighting the differences in these members.

Use the vertical scroll bars to move within the files. As you scroll, you will see where the differences are in the members. RPG experts will notice that PAYROLL has some errors in it. We will fix these in a few moments.

From the **Compare** menu (which was inserted while performing this action), select **Exit Compare** to go back to the original view.

Syntax Checking

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 file. In this part of the exercise you will create a syntax error and will be immediately prompted to correct it.

- Move the cursor to **line 198** which contains **'EXSR ACDESR'**.

You might already be on that line but if not, type the line number in the sequence number column, or scroll down

- Append an **X** to the **EXSR** op-code to make it **EXSRX**.
- Move the **cursor off** of the line.

An error message appears to draw attention to the error.

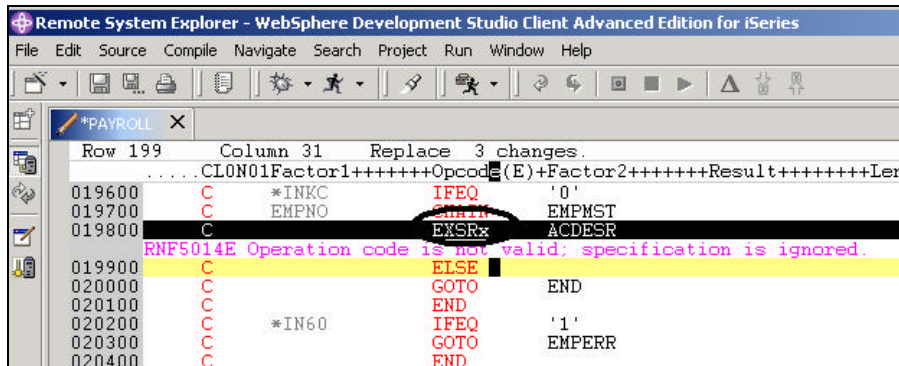


Figure 35: Editor window with syntax error

- Move the *cursor* onto the *pink error message*.
- Press **F1**.



Figure 36: Second level help for syntax error

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

- Minimize the *help window*.
- Change **EXSRX** to **EXSR** to correct the error.
- 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 by using the **Windows**→**Preferences** option on the workbench menu bar and the **Remote Systems + LPEditor Parsers + ILE RPG** nodes in the preference tree view



Figure 37: Preferences dialog to toggle syntax checking

Verifying Your Source

Now you get to play with one of the most powerful and unique features of the RSE -- 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 RSE 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, inserts the error messages directly into the source and helps you to navigate between the errors.

Invoking 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:

- Select the *Source* option from the workbench menu bar
- Select the *Verify* option from the pull down menu

After a moment the verifier will display an error list underneath the editor window.

ID	Message	Severity	Line	Location	Connection
RNF5178	ENDSR operation is missing for subroutine DELCDE; ENSDR statement is missu...	30	1	CODELAB\ORPGL5SRC\PAYROLL	s400a
RNF7030	The name or indicator SELTAX is not defined.	30	73	CODELAB\ORPGL5SRC\PAYROLL	s400a
RNF7043	Target SELTAX of GOTO or CABCS operation is not a label; the specification L...	30	73	CODELAB\ORPGL5SRC\PAYROLL	s400a
RNF7051	The name or indicator ACDEID is not referenced.	0	414	CODELAB\ORPGL5SRC\PAYROLL	s400a
RNF7066	Record Format EMPLSTCTL not used for input or output.	0	27	CODELAB\ORPGL5SRC\PAYROLL	s400a
RNF7066	Record Format EMPLSTSQL not used for input or output.	0	27	CODELAB\ORPGL5SRC\PAYROLL	s400a

Figure 38: Verifier error list

The error list will show you

1. The error message itself
2. The severity
3. The line number
4. The source location
5. The connection name

Fixing errors

To fix an error in your source go the error list and:

- Double-click on the error **RNF7043**

You are automatically brought back into the editor window to the line where the error occurred. The error on line **73** is a typo. **SELTAX** should really be **SELTAG**. Make the appropriate change.

The only serious remaining error is **RNF5178E**. This error is caused by a missing **ENDSR**.

Note: The verifier could not determine where the ENSDR is missing so the line number reported is 1, for this reason you can't just double click on the error message, you need to investigate where the missing statement belongs.

- Move to **line 462** in the editor
- Place the cursor in the **text area** of **line 462**
- Press the **Enter key** to insert a new line.
- In line **463**, type

C **ENDSR**

Hint: use the Tab key to quickly move to the appropriate column

All the non-informational errors are now fixed.

You can filter out different severities by using the filter menu

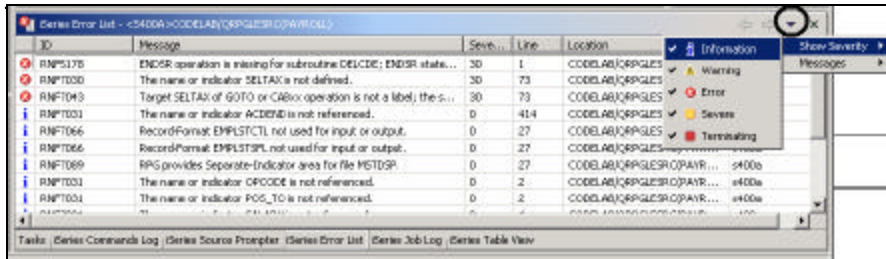



Figure 39: Filter for severity levels

- Click the arrow in the window bar as shown in Figure 39
- Deselect the severities you don't want to see in the list (Information for example)

Saving a Source Member

Before you lose any of your changes, it's a good idea to save them. You can save the member from either:

1. The **File** option on the workbench menu bar
2. The workbench **toolbar** (click on ) ,
3. Press **Ctrl+S**.

Changes are uploaded to the iSeries

- Verify your source again,

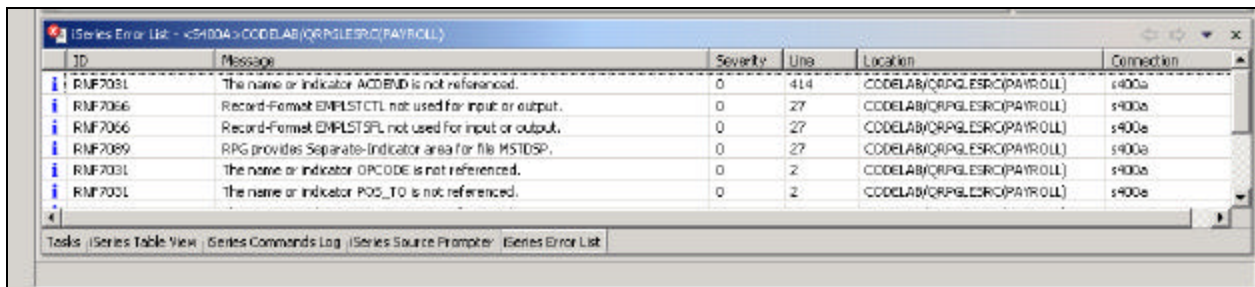


Figure 40: Error list, only informational message are left

Everything should be **OK**, you are ready to compile the program.

Remote compile

The remote compile capability is part of the RSE. It gives you a workstation interface to submit requests to the iSeries to compile, bind, or build objects on the 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 Error List window, 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 the lab environment you can run the compiles interactive.

Change the preferences to compile in interactive by:

- Selecting the **Windows** option from the workbench menu bar
- Selecting the **Preferences** option from the pull down menu

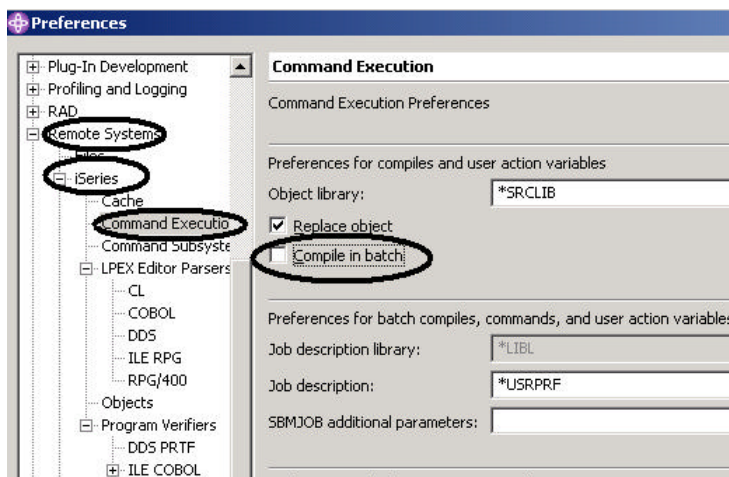


Figure 41: Change Compile command default

- Expanding **Remote Systems**
- Expanding **iSeries**
- Selecting **Command Execution**
- De-selecting the **Compile in batch** check box
- Click the **OK** push button, to get back to the RSE perspective

Starting the compile

Starting a compile is like using any other command in the RSE. In the RSE tree view that shows your expanded library CODELABxx:

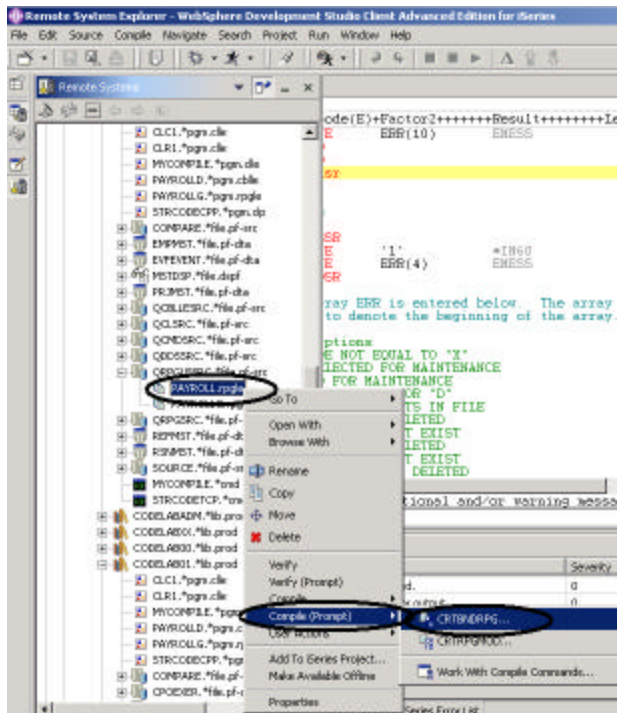


Figure 42: Select member to compile

- Right-click on the **PAYROLL** member in QRPGLSRC
- Select the **Compile (Prompt)** option from the pop-up menu
- Select the **CRTBNDRPG...** option from the sub menu

Compiling Your Source

You will now use the prompt for the CRTBNDRPG command to specify your compile parameters

All entry fields pertaining to names are already filled in with the correct information.

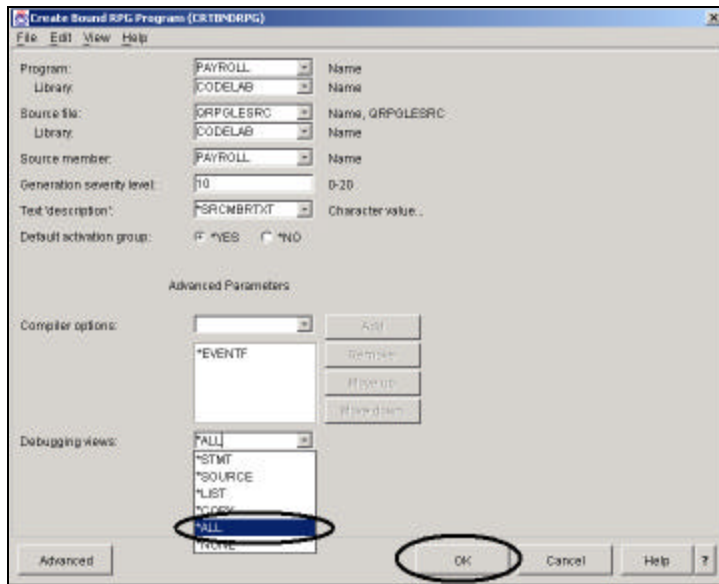


Figure 43: Prompt for CRTBNDRPG

- Change the **Debugging views** parameter to ***ALL**.

If you want check out the other parameters available by clicking the **Advanced** push button

- Click the **OK** push button when you are done.

A dialog box will show that the compile started

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 informational messages

If you are not sure that the compile was successful:

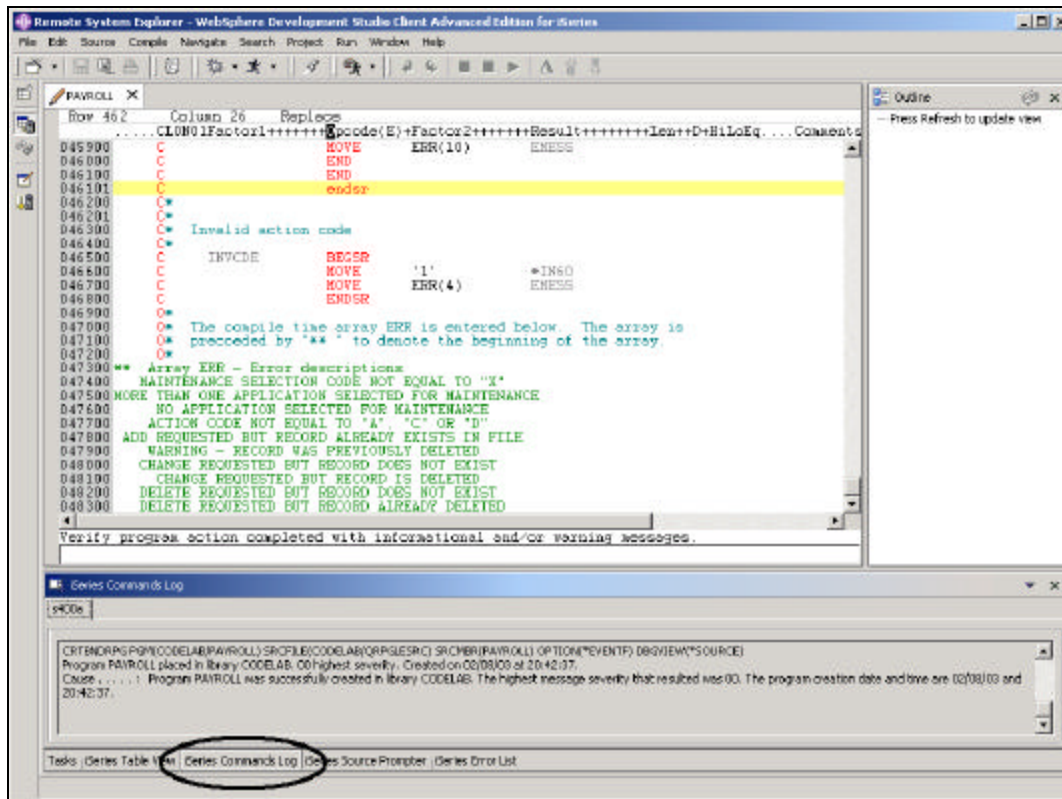


Figure 44: iSeries Commands Log

- Click on the *iSeries Commands Log* tab at the bottom of the workbench

The iSeries table view with Command entry field

You can use the **iSeries table view** inside the RSE to submit commands to the iSeries.

You could change your library list, for example:

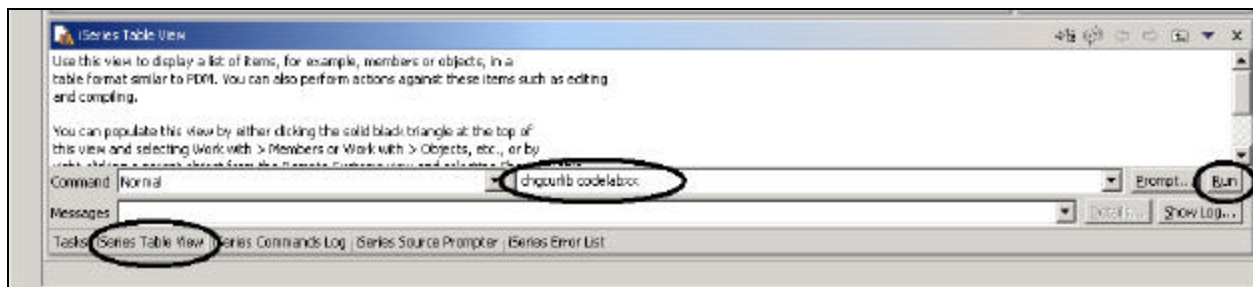


Figure 45: Table view with command entry

- Select the *iSeries Table view* tab from the views at the bottom of the workbench
- Enter in the command entry field: **CHGCURLIB CODELABxx**
- Click the **Run** push button

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

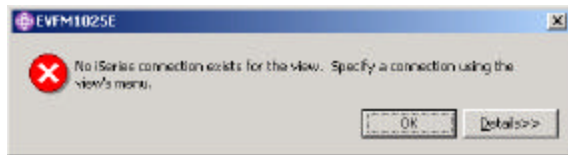


Figure 46: error message when using Table view without active connection

If you get this message:

- Go to the **RSE tree** view
- Right click on **QRPGLESRC** file

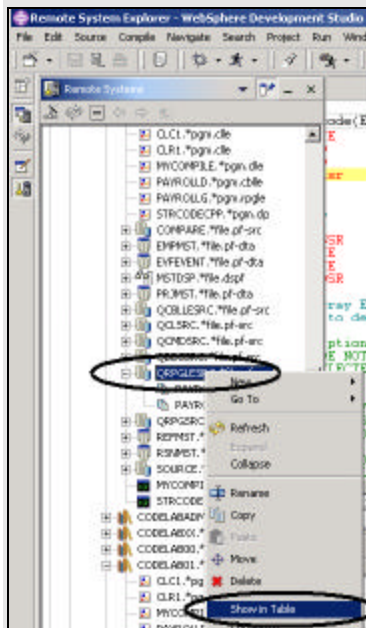


Figure 47: Select Table view to connect it to iSeries

- Select the **Show in Table** option from the pop up menu

The table view is now populated with the member in the selected source file

- Run the **CHGCURLIB** again

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

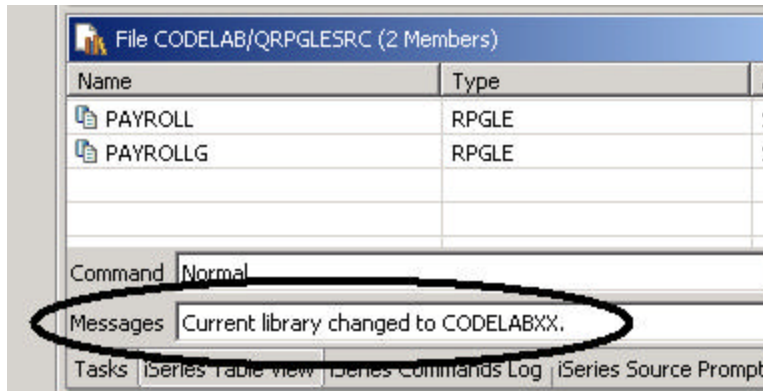


Figure 48: Completion message for command

Tip: You can also connect to other than iSeries systems with the Remote System Explorer and Launch commands for these systems as well, i.e. your local system, or LINUX.

Interactive Connection to the iSeries

Communications between the iSeries and your workstation can be configured for:

1. RSE server job, the one that you use currently
2. Interactive job to test 5250 applications

If you need an Interactive communications link to the iSeries server, you need a 5250 emulator to start the interactive job on the iSeries. The communications link will be established through the emulation session by using an OS/400 command to initiate the session to your workstation. For the next task you will be using an interactive connection in this Lab.

Using an interactive connectionI

Start a 5250-emulation session.

Sign on to the iSeries with your userid and password **WDSCLABxx** where **xx** is your workstation number (01, 02, etc.).

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

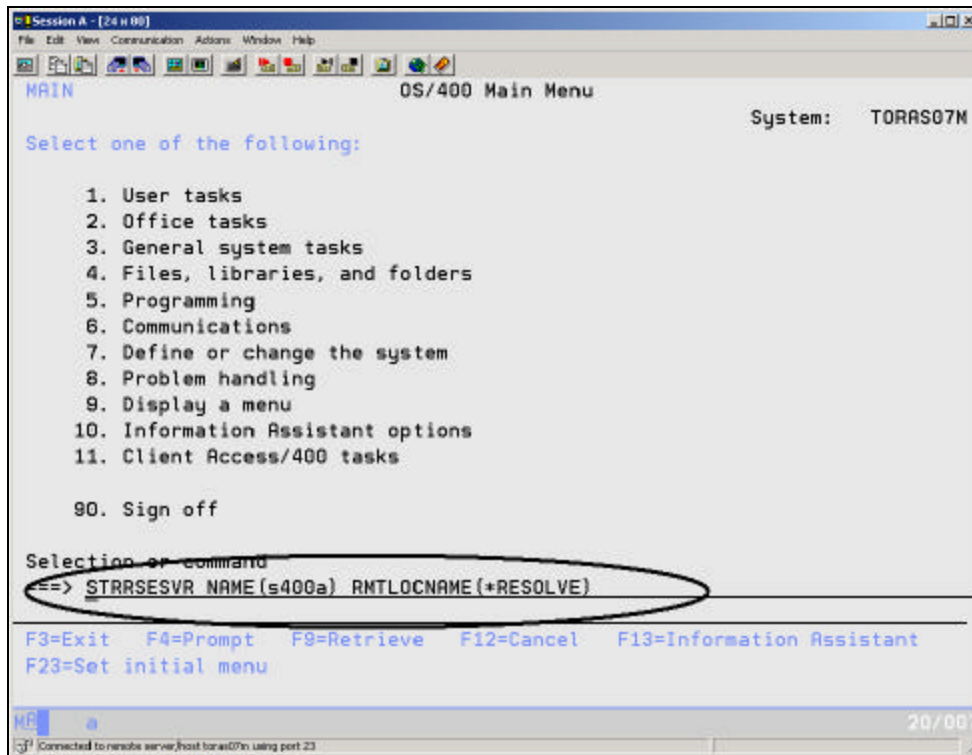


Figure 49: Start RSE interactive connection

- Key in the following command

STRRSESVR NAME (s400a) RMTLOCNAME(*RESOLVE

The ***RESOLVE** keyword will get the IP address of your workstation and with this information the **RSE server** will communicate with the **RSE daemon** that runs on your workstation.

You should see a screen as shown in Figure 50.

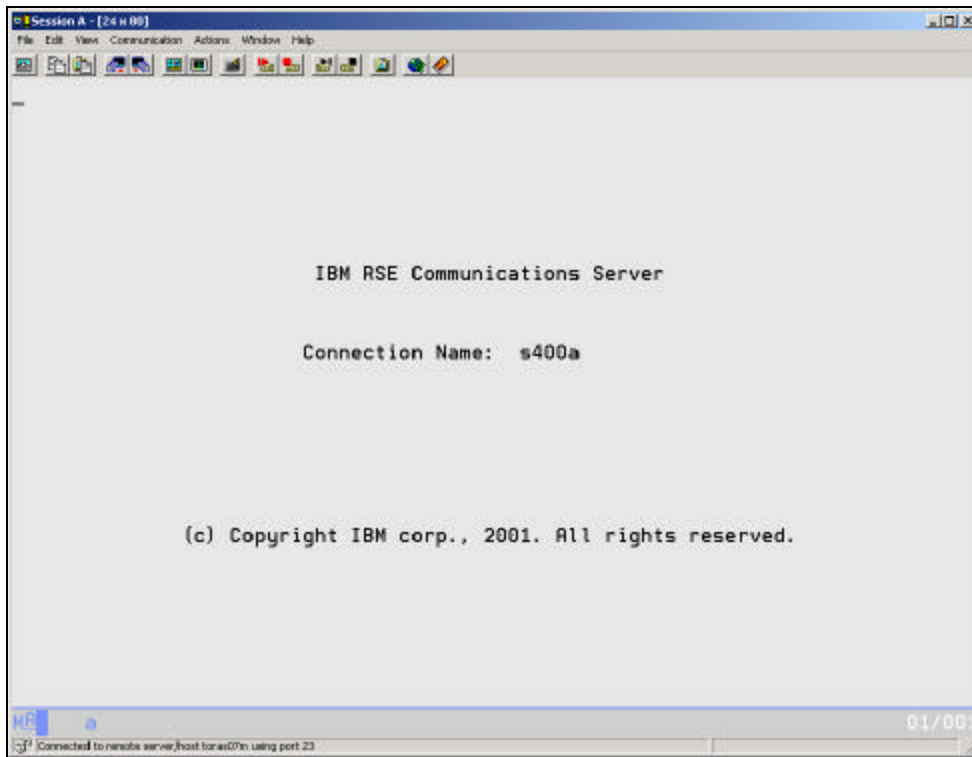


Figure 50: RSE connection screen

Note: This screen will stay this way, don't wait for it to finish, this session is the interactive session for interactive commands started from the RSE.

Running the PAYROLL Program

In this part of the exercise you will run the *PAYROLL* program that you just compiled.

In the RSE tree view

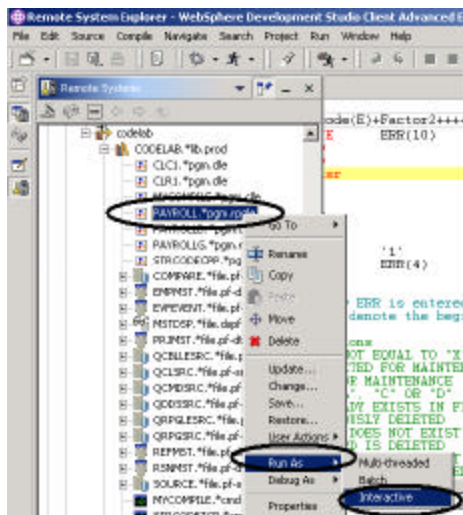


Figure 51: Run program in interactive job

- Locate the **PAYROLL** program that you created
- Right click on the **PAYROLL** program icon in the RSE tree view
- Select the **Run As** option from the pop up menu
- Select the **Interactive** option from the sub menu
- Switch to your **5250-emulation** session.

You will see the **Start Menu** of the payroll program

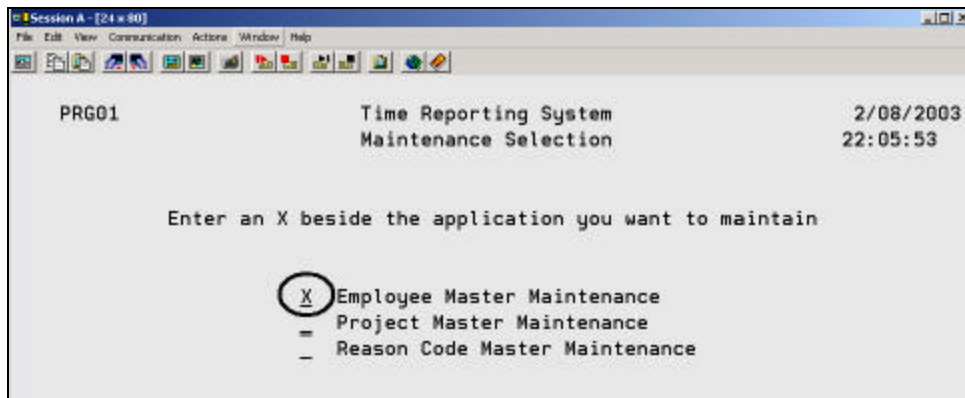
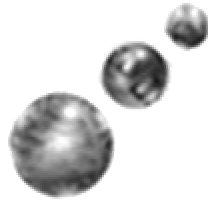


Figure 52: Payroll Start Menu

- Place an **'X'** beside **Employee Master Maintenance**.
- Press the **Enter** key.
- Type **123** for the Employee Number.
- Type **A** for the Action Code to add employee **123**.
- Press the **Enter** key.

- Type any information you like about the employee.
- Press the **Enter** key.
- Play in the application as much as you like.
- Press **F3** to end the PAYROLL program.

We hope you enjoyed the exercise now you will switch gears, you will look at the 5250 User Interface Design tool in WebSphere Development Studio



Exercise2: Working with 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 let's 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. We'll now step through each part of the interface and update some DDS as well.

Open a DDS display file member using the WebSphere workbench

In the workbench, in the RSE perspective use the connection that you used in the exercise before. In the list

- Expand the **Library List** filter or if it is still expanded use it as is,
- Expand the **QDDSSRC** file in library **CODELABxx**
- Right mouse click on the **MSTDSP** member and select:

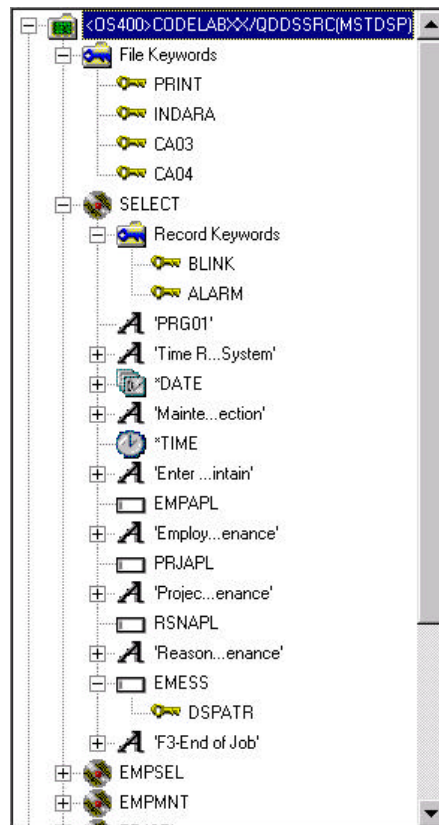
Open with --> Code designer

The member **MSTDSP** will be downloaded to the workstation and loaded into CODE designer.

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, we'll explain everything you need to know.

- ___ 1. Click on the + beside the folder <Servername>CODELABxx/QDDSSRC(MSTDSP).
- ___ 2. Click on the + beside the folder **File Keywords**.
- ___ 3. Click on the + beside the record **SELECT**.
- ___ 4. Click on the + beside the folder **Record Keywords**.
- ___ 5. Click on the + beside the field **EMESS**.



The DDS Tree is now showing you a nice summary of the file-level keywords and of the record SELECT

The Details Page

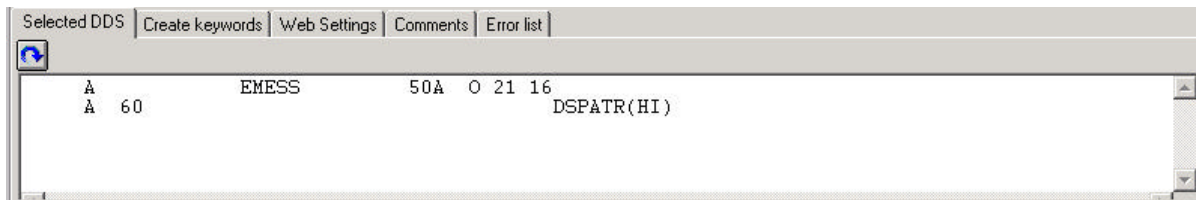
In the upper right-hand side of the Designer is the Workbook with several different tabbed pages. The top page is called Details and it lists the contents of the currently selected item in the DDS Tree.

In the bottom right-hand side of the Designer is the Utility Notebook. The **Selected DDS** page in the notebook shows the actual DDS source for the currently selected item.

- ___ 1. In the DDS Tree click on the record **SELECT**. 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	Position	Length	Type	Shift	Usage	Sample
A 'PRG01'	2, 5	5	Text constant			PRG01
A 'Time R...System'	2, 30	21	Text constant			Time Reporting System
<input checked="" type="checkbox"/> *DATE	2, 70	6	Date constant			YY/MM/DD
A 'Mainte...ection'	3, 30	21	Text constant			Maintenance Selection
<input checked="" type="checkbox"/> *TIME	3, 70	6	Time consta...			HH:MM:SS
A 'Enter ...intain'	6, 14	54	Text constant			Enter an X beside the application you
<input type="checkbox"/> EMPAPL	9, 25	1	Alphanumeric	A - Alphanumeric	Both	B
A 'Employ...enance'	9, 28	27	Text constant			Employee Master Maintenance
<input type="checkbox"/> PRJAPL	10, 25	1	Alphanumeric	A - Alphanumeric	Both	B
A 'Projec...enance'	10, 28	26	Text constant			Project Master Maintenance
<input type="checkbox"/> RSNAPL	11, 25	1	Alphanumeric	A - Alphanumeric	Both	B
A 'Reason...enance'	11, 28	30	Text constant			Reason Code Master Maintenance
<input type="checkbox"/> EMESS	21, 16	50	Alphanumeric	A - Alphanumeric	Output	00000000000000000000000000000000000000
A 'F3-End of Job'	23, 7	13	Text constant			F3-End of Job

- ___ 2. In the DDS Tree click on **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.
- ___ 3. In the DDS Tree click on the field **EMESS**. The Details page shows its field-level keywords. The Selected DDS page now shows the DDS for the EMESS field.

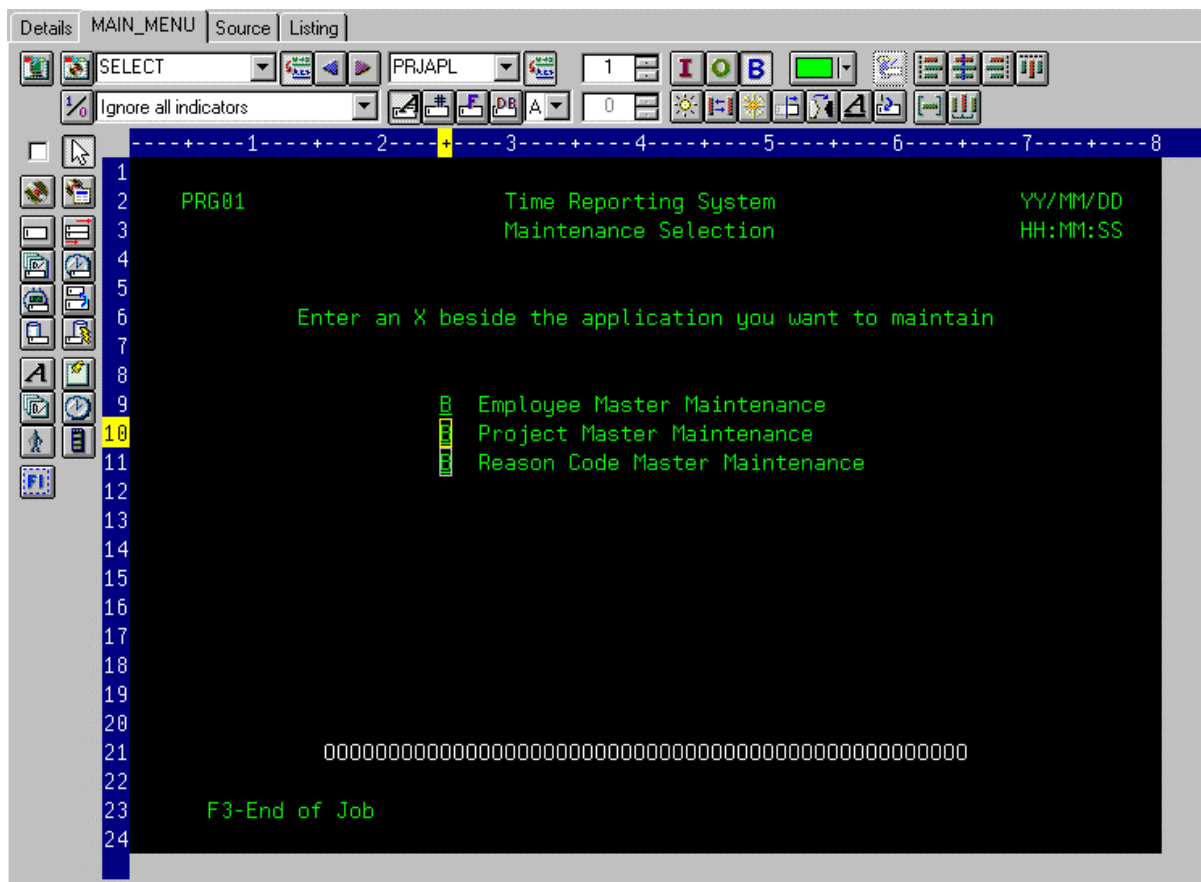


Even this relatively small and simple DDS source member demonstrates how much easier it is to use the Designer to navigate through your DDS source. The syntax is being interpreted in intuitive graphical ways making it an ideal tool for learning DDS. But to get orders of magnitude improvement in your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

The Design Page

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.

- ___ 1. Click on the tab labeled **MAIN_MENU** in the workbook.




In order to understand where MAIN_MENU came from, we need to describe the concept of a **group**. A group is simply a collection of one or more DDS records that make up a single screen or report. It is the set of records that is written by the application to the display or printer device at one time. Grouping records together allows you to work on one record while still seeing the related records in the background. The Workbook has a Design page tab for each group defined for quick access to each group of records.

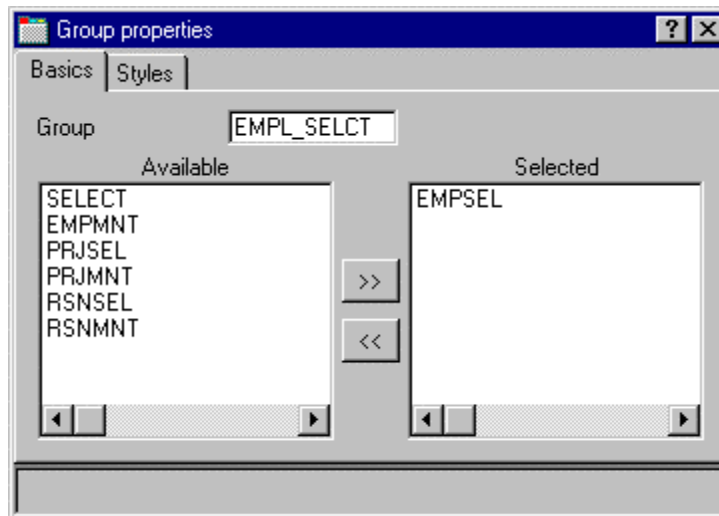
Creating Groups from Existing Records

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 we will create a group for the next screen, where the user selects which employee in the payroll database to maintain.

- ___ 1. Scroll to the **bottom** of the **DDS Tree** and **click** on the + beside the MAIN_MENU group. The SELECT record appears as the only record in this group.



- ___ 2. Right-click on the **MAIN_MENU** group.
- ___ 3. From the pop-up menu, select **Insert group...** A **Group properties** notebook appears and a blank Design page for the group **SCREEN1** also appears.
- ___ 4. On the Group properties notebook, click on the record **EMPSEL** in the Available list box and click on the  push button. For simplicity this is the only record we will add for now. The Design page now shows us what the record EMPSEL looks like.
- ___ 5. Name the group by over typing **SCREEN1** with **EMPL_SELECT**

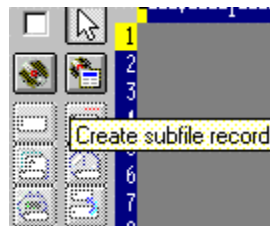


- ___ 6. Close the Group properties notebook by clicking on the **X** in the top right corner.

Well, 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.


Creating New Screens

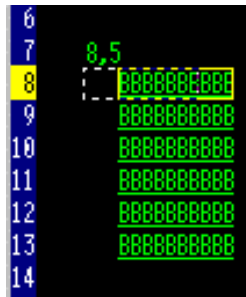
- ___ 1. Right-click on the new **EMPL_SELECT** group in the DDS Tree.
- ___ 2. From the pop-up menu, select **Insert group...** A **Group properties** notebook appears and a blank Design page for the group **SCREEN1** also appears.
- ___ 3. Rename the group to EMPL_LIST and close the Group properties 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 tools 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 on the **Create subfile record** button and then click in the dark gray area. A subfile and a subfile control record pair is created.

Field Creation and Design Page Toolbar

- ___ 1. Now **click** on 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** on 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 mouse over to the **right edge** of the field. It turns into a double-headed arrow.

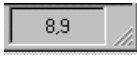
Hold down the mouse button and **move** it to the **left**. The size of the field will be reduced. The current size will appear just above the field. When the size is **3**, **let go** of the mouse.

- ___ 4. The toolbar at the top of the design page is a very convenient place to monitor and manipulate the currently selected field. Rename the record from RECORD1 to EMPLSTSFL and the field from FIELD1 to OPCODE by simply over typing the text in



the combo box. Change the color of the field by clicking on the **Color palette** button and selecting pink. Change the usage of the field to input only by clicking on the **I** button.

- ___ 5. Position the cursor at **row 8, column 9**. Note that the bottom right of the


Designer frame shows the current cursor position  If you can't see the field

with the cursor position on your screen, press the Maximize button in the top right corner. You can use the cursor keys or the mouse to move the cursor.

___ 6. If you are creating a long field with an exact length, the SDA syntax can be easier. Type:

+O(30)

and then hit the **back arrow (not Backspace!)** to select the text you created. Notice from Selected DDS page that you have created a text constant containing '+O(30)'.

___ 7. Hit 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.

___ 8. Rename the new field to **ENAME** using the toolbar. This will show the name of the employee.

___ 9. Position the cursor to **8, 41**.

___ 10. Now we will add a field for the employee's salary. Now wouldn't it be nice 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? **Type**

\$666,666.66

and then hit the **back arrow**.

___ 11. **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!

___ 12. Rename the field to **SALARY** and change its color to yellow, using the toolbar.

___ 13. Our subfile seems a little scrunched to the left. It would be nice to space it out evenly.



Just select the field and hit **Space horizontally** on the far right side of the toolbar. The other buttons in the vicinity will do your typical align, left, right, center and top.


___ 14. Just below the palette 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. **Set the Subfile size to 300** (by over typing) and the **Subfile page size to 9**. The design page is updated accordingly.

Switching between Multiple Records

- ___ 1. Now let's fix up the Subfile control record. The group we created contains 2 records.
You can verify this by dropping down the record combo box in the toolbar.



- Change the current record by selecting RECORD1CTL from the combo box or by hitting the  or simply by pressing **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.
- ___ 2. Rename the record to EMPLSTCTL using the toolbar.

Copy and Paste

Let's provide a 'position to' entry in the subfile control header.

___ 1. Position the cursor at **4,9** and type:

Position to:

___ 2. Now we need an employee name field. We could create a named field with the right characteristics like we did in the subfile, or we could create a source reference using the



button in the palette or we could reference the original database field using one of



the buttons. But there is an even simpler way. Use copy and paste! In the

DDS Tree expand (click on the +) the **EMPMNT** record.


___ 3. Click on the **ENAME** field and press **Ctrl+C**. (The pop-up menu or Edit pulldown would give the Copy menu item as well).

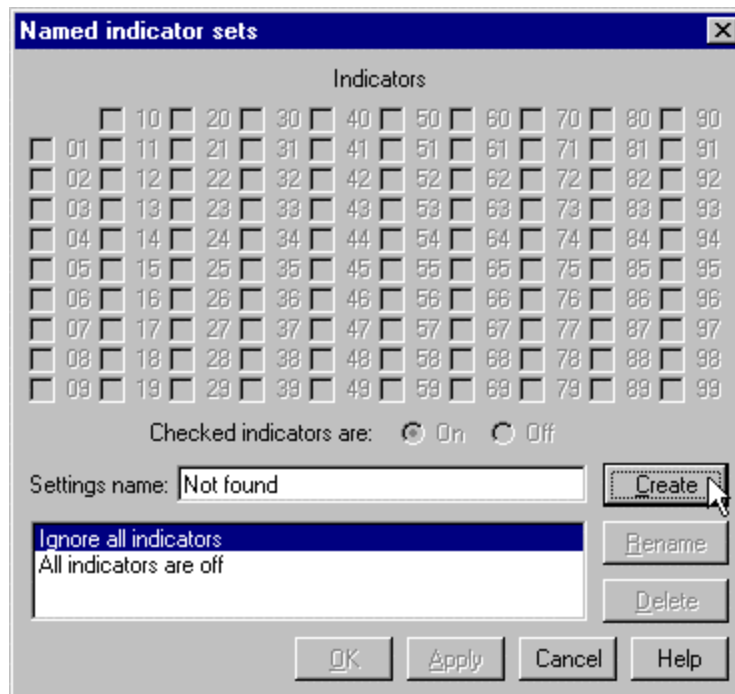
___ 4. Position the cursor to **4, 23** and press **Ctrl+V**. Voila! Now that was easy!

___ 5. Rename the field to **POS_TO**.

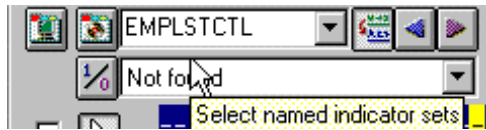
Working with Indicators

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 turn on indicator 60. The screen should turn the field red, reverse image and position the cursor to it. Now wouldn't it be nice if we could work something higher level and easier to remember than some arbitrary number from 1 to 99.

- ___ 1. Click on the  button on the design page toolbar (or **press F7**). The **Named indicator sets** dialog appears.
- ___ 2. In the **Settings name** field, type:
Not Found
and hit the **Create** button.



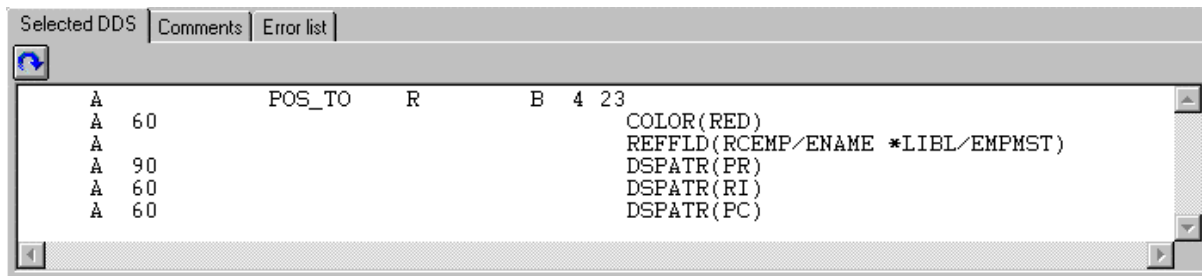
- ___ 1. **Click** on the checkbox next to **60** and press **OK**. The **Not found** indicator set is now in effect. The design area is shown as if indicator 60 was on and all other indicators were off. The design page toolbar shows the current indicator set in the combo box on the bottom left.



- ___ 2. Now select the **POS_TO** field.
- ___ 3. On the toolbar, select the color **red** and the display attributes **reverse image** and **position cursor**. (The set of latched toolbar buttons representing the current display attributes is found just below the color button). The toolbar should look as follows:

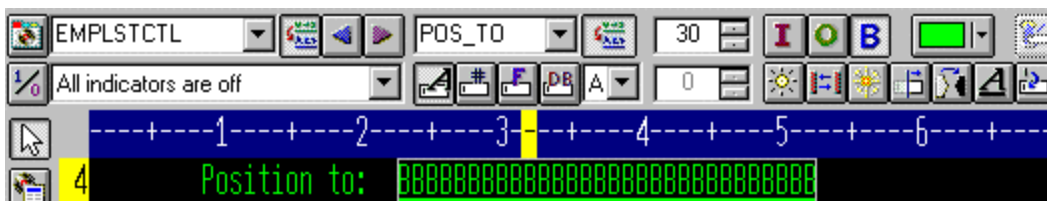


- ___ 4. **Examine** the DDS generated in the **Selected DDS** page.



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

- ___ 5. Now let's try it out! From the **Select named indicator sets** combo box, select **All indicators are off**.



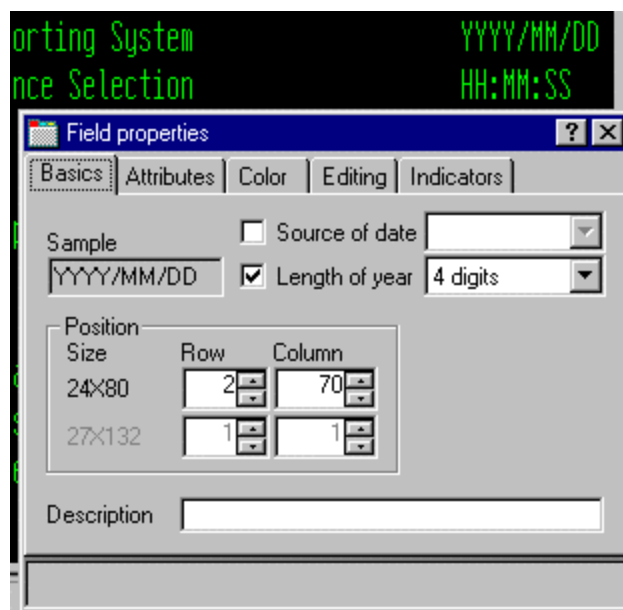
Hey! This stuff really works!

- ___ 6. From the **Select named indicator sets** combo box, select **Not found**. The field appears red and reverse imaged.

The Properties Notebook

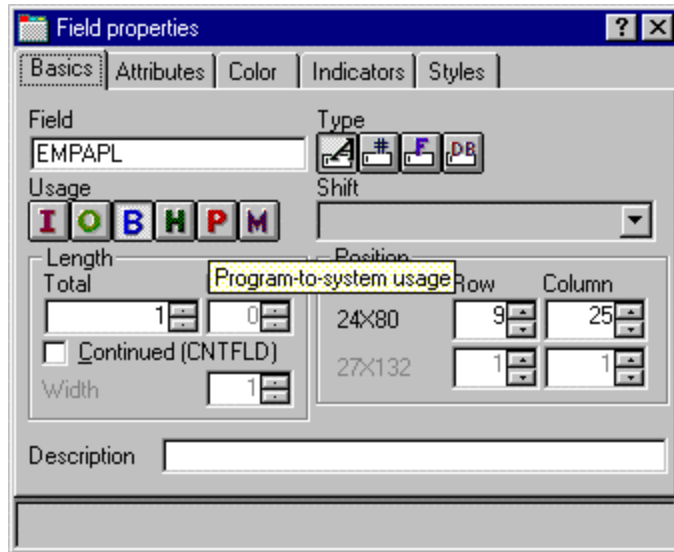
Second to the direct manipulation and the toolbar on the Design page, the easiest and quickest ways of getting access to the properties of a field, record, or entire file is a Properties notebook. You can get to a Properties notebook from the **Selected** menu, by pressing **F4**, or **double-clicking** on anything in the DDS Tree or the Details page or Design page.

- ___ 1. In the DDS Tree, click on 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. Now click on 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. Let's say we had to change the year from 2 to 4 digits. Click on the **Length of year** checkbox.
- ___ 4. Select **4 digits** from the combo box. Notice how the sample is updated on the properties notebook.



- ___ 5. To test the Design page, click on the **MAIN_MENU** tab in the workbook and look at the upper right corner of the screen. The date now has 4 digits.

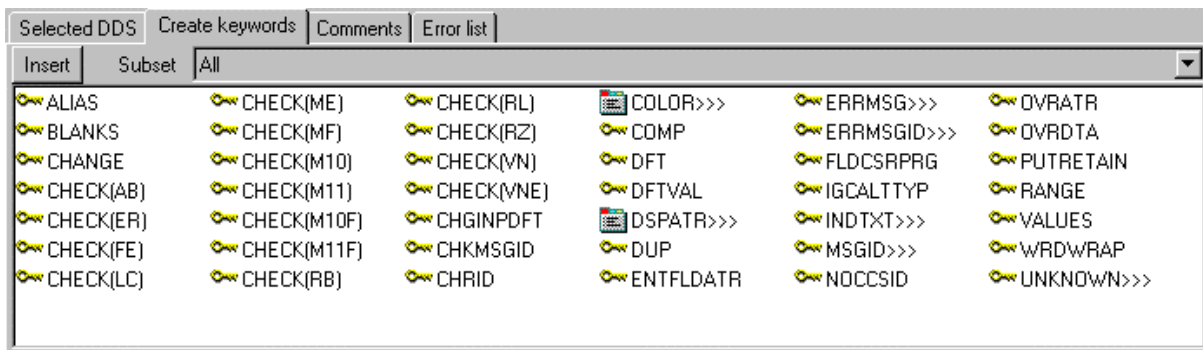
6. Now click on the **EMPAPL** field in the SELECT record. On the **Field Properties** notebook click on the **Basics** tab. On this page you can change the field's name, usage, length, type, screen position. The other pages give you quick access to other properties for this field.





Adding New Keywords

“OK”, you might think, “I can see where CODE Designer helps me manage the visual aspects of my displays and reports. But I have real work to do. I need access to the full power of DDS.”

- ___ 1. Click back on the **EMPAPL** field in the DDS Tree.
- ___ 2. Press **F5** or select **Insert keywords** from the pop-up menu. You are taken to the Details page for the EMPAPL field and the **Create keywords** tab is selected in the Utilities 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.




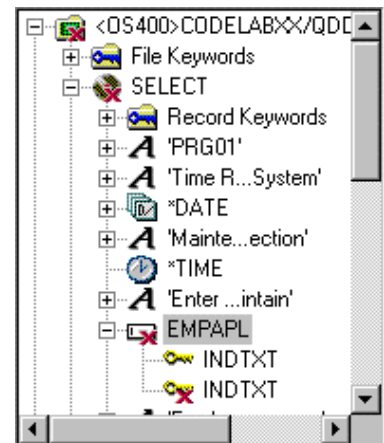
- ___ 3. With the **EMPAPL** properties notebook at the Basics page, click on the  button to change the field to numeric type. Notice that the list of keywords in the **Create keywords** page has changed.
- ___ 4. Click on the  button to change the field back to alphanumeric. Notice that the list of keywords in the **Create keywords** page has changed.
- ___ 5. Click on the **ALIAS** keyword and press **F1**. The DDS Reference help for the ALIAS keyword appears. It is important to mention that the CODE Designer has lots of on-line help. Feel free to press F1 anywhere you want to see help for an item, icon or notebook. You will get help relevant to what you are currently trying to do. From the **Help** menu you can get quick access to the DDS Language Reference as well as several other useful sources of information.
- ___ 6. Minimize the help window

- ___ 7. Double click on 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 at your leisure.
- ___ 8. Double click on the **INDTXT** keyword again. The keyword is created with the same default values creating a conflict.

Verifying Your Source

We have just added a new record and some new fields to our DDS source. Everything that the CODE Designer adds to your DDS source is certain to have the correct syntax. Now we need to make sure that there are no semantic errors. We just introduced one in the last section by creating two INDTXT keywords describing the same indicator.

- ___ 1. From the **Tools** menu, select **Verify file** (or click on the  button on the main toolbar). The DDS source is checked using the same verifier that the CODE Editor uses. A message appears on the status line at the bottom of the designer stating that the verify completed with errors.
- ___ 2. In the DDS Tree, there is a trail of red x's leading to the culprit. The file icon has a red x, as does the SELECT record, the EMPAPL field and finally the second INDTXT keyword.
- ___ 3. Click on the **MAIN_MENU** tab in the workbook. The EMPAPL field is highlighted in red.
- ___ 4. Click on 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. Click on the **Error list** tab.
- ___ 6. Double-click on the warning DDS7861 in the Error list. (Hitting F1 would get 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 nice if you could just zap 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.



```

.....AAN01N02N03.....Functions+++++++Co
A      you want to maintain'
A      EMPAPL      1  B  9 25INDTXT(01 'Default')
A      INDTXT(01 'Default')
DDS7861W Text already assigned to indicator.
A      9 28'Employee Master Maintenance'
A      PRJAPL      1A B 10 25
A      10 28'Project Master Maintenance'
A      RSNAPL      1A B 11 25
A      11 28'Reason Code Master Maintenance'


```

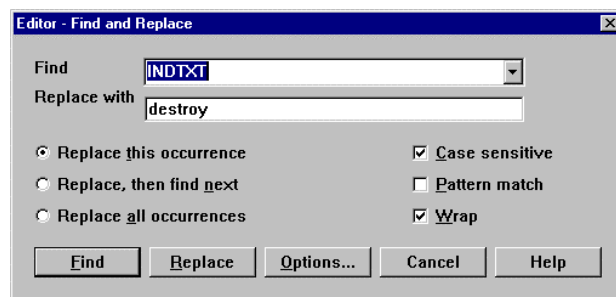
Selected DDS | Comments | Error list

<OS400>CODELABXX/QDDSSRC(MSTDSP)

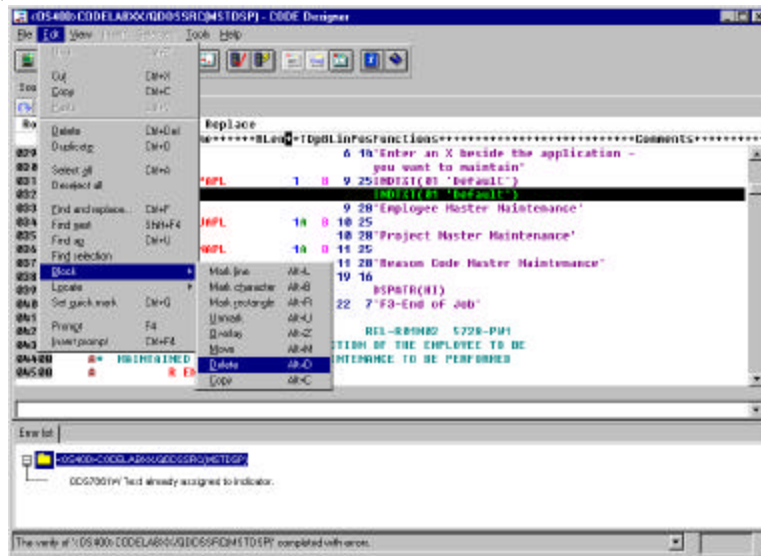
DDS7861W Text already assigned to indicator.

Switching between Design mode and Edit mode

- ___ 1. Click on the  button or select the **File** → **Edit DDS Source** menu item. You now have access to the full power of the editor.
- ___ 2. Explore the **Edit** and **View** pulldowns.
- ___ 3. Press **Ctrl-F** to bring up the Find/Replace dialog
- ___ 4. Type in **INDTXT** and hit the **Find** button.




- ___ 5. Press **Ctrl-N** to find the next occurrence.
- ___ 6. **Delete** the second **INDTXT** line.



Compiling Your Source

Now we will compile the source on the iSeries just as we did in the CODE Editor.

- ___ 1. From the **File** menu, select **Save** to save your source to the iSeries.
- ___ 2. From the **Tools** menu select **Compile** and then select **No prompt** (or click the  button on the main toolbar).
- ___ 3. A message will tell you when the compile is complete. Click on the **OK** push button in the message dialog.

If you run the PAYROLL program, you will see the 4 digit year change you made to the opening screen of the program.

Closing CODE Designer

- ___ 1. From the **File** menu, select **Exit**.

Now let's play with the debugger.

Exercise 3: Working with the Debugger

The debugger is a source-level debugger that allows you to debug and test an application running on a host iSeries from your workstation. 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 host.
- Set, change, delete, enable and disable line breakpoints in the application program. You can easily manage all your breakpoints using the Breakpoints list.
- Set watch breakpoints to make the program stop whenever a specified variable changes.
- View the call stack of your program in the Stacks page. As you debug, the call stack gets updated dynamically. You can view the source of any debuggable program by clicking on its call stack entry.
- Step through your code one line at a time.
- Step into or step over program calls and ILE procedure calls.
- Display a variable and its value in the Monitors page. 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 Programs page.
- Debug multithreaded applications, maintaining separate Stacks for each thread with the ability to enable and disable any individual thread.
- Load source from the workstation instead of the iSeries -- useful if you don't want the source code on a production machine.
- Debug client/server and distributed applications.

The Debugger supports Java as well as RPG/400 and ILE RPG, COBOL and ILE COBOL, C, C++ and CL.

In the following exercise you will be given the opportunity to learn about some of the basic features of the Debugger. For the purpose of this exercise we will be debugging an ILE RPG/400 program. Don't worry if you don't know RPG.

Starting the Distributed Debugger

You will be working with the ILE RPG program **PAYROLLG**.

Note: PAYROLLG is the same RPG program as PAYROLL but without compile errors. You are using it instead of PAYROLL in this debugger section, to accommodate everyone who decided to skip right to this section without completing the editor exercises.

You can start the Debugger in several ways: direct from the RSE tree view, or from the launch configuration dialog. Starting direct from the preview doesn't allow you to specify parameters to be passed to the program, the launch configuration allows to modify how the program is invoked including to specify parameters. To make life a little

more interesting you will be using CL program CLR1 to call PAYROLLG and you will pass one parameter to CLR1.

This means you will use the launch configuration dialog.

Use the Remote Systems Explorer to start the Debug launch configuration:

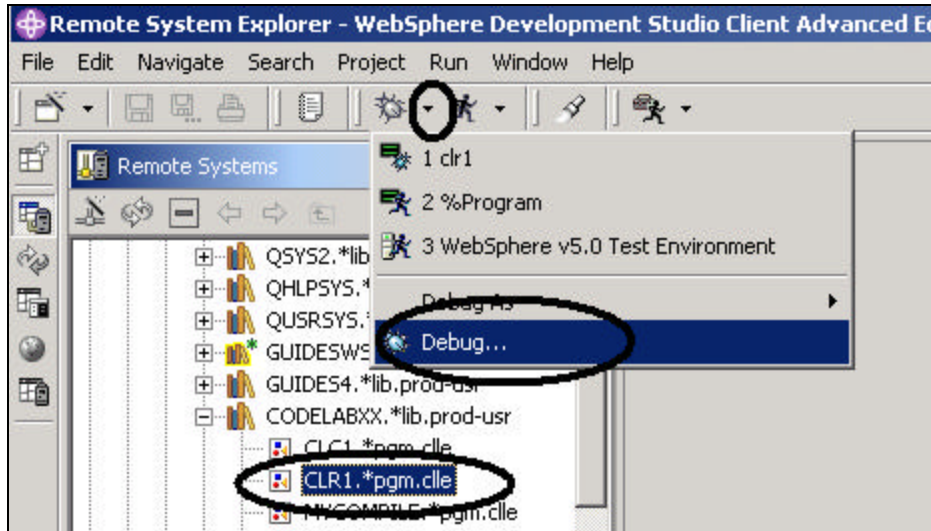



Figure 53: Start the Debug launch configuration

In the RSE tree view,

- Expand the **Library list filter**, if it isn't expanded already
- Expand library **CODELABxx**, if it isn't expanded already
- Select the program **CRLI** in library CODELABxx

On the workbench toolbar:

- Click on the **little arrow** beside the **DEBUG icon**  as shown in Figure 53
- Select the **Debug..** option from the pull down menu

The Debug launch configuration dialog appears

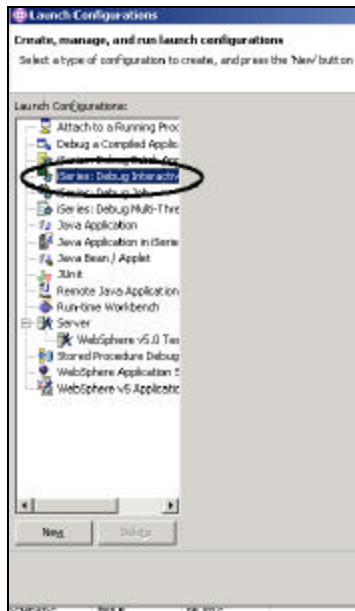


Figure 54: Launch configuration dialog

In the tree view:

- Select the *iSeries: Debug Interactive Application* node
- Click the *New* push button

The debug Interactive Application pane appears on the right side of the dialog

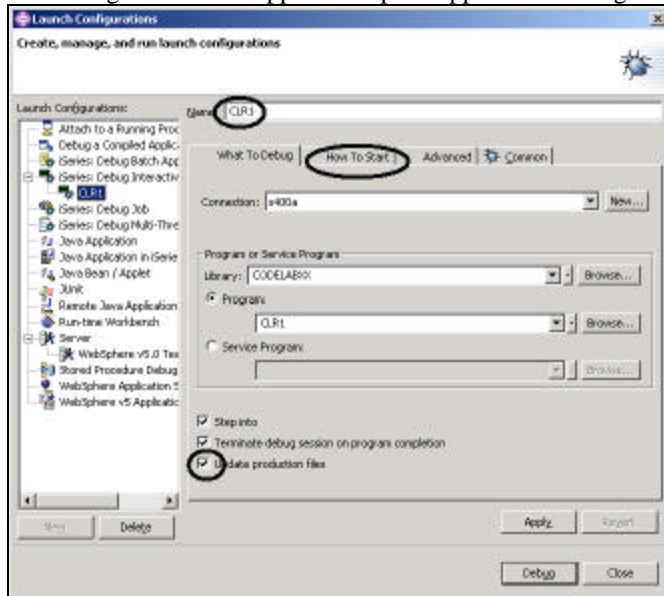


Figure 55: Debug Interactive Application

- Enter the program name *CLRI* in the *Name* entry field
- Select the *Update production files* check box

- Click the **How To Start** tab

On the How To Start page

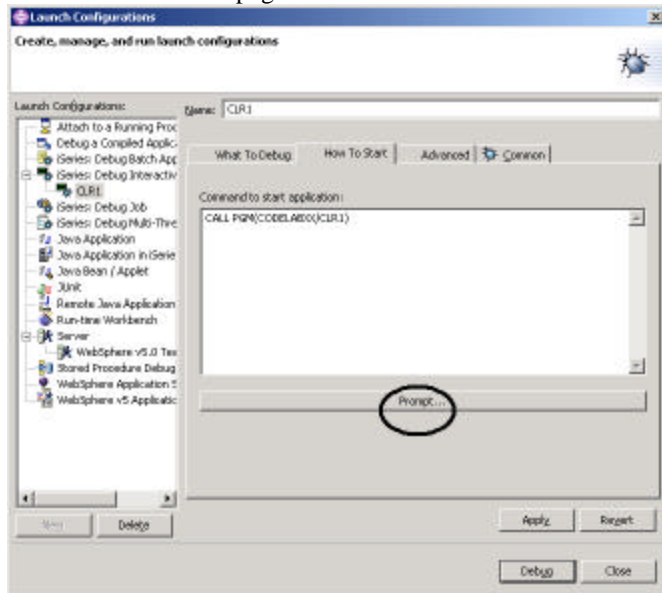


Figure 56: How to Start page

- Click the **Prompt** push button

In the Prompt dialog

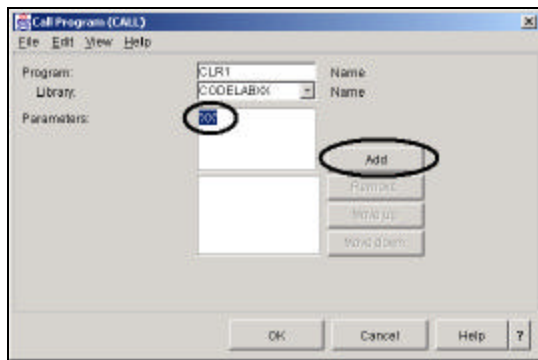


Figure 57: Prompt dialog

- Enter 'XX', XX being your team number, in the parameter field
- Click the **Add** push button

The parameter value will appear in the lower list box

- Click the **OK** push button

The complete start command for the program appears in the dialog

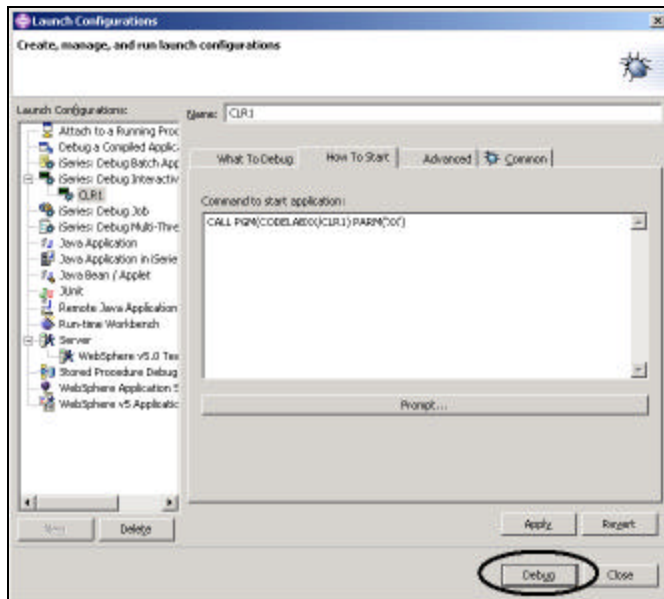


Figure 58: Ready to debug

- Click the **Debug** push button

If an error message like this appears,

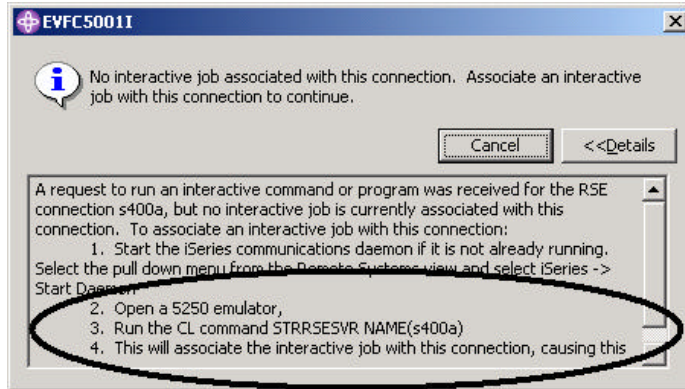


Figure 59: Error message if interactive session doesn't exist

The RSE server has been shut down in the meantime, go to your emulator and restart the RSE server following the instructions in the message.

Cancel the message, in the debugger perspective

Now the Debug perspective is loaded in the workbench

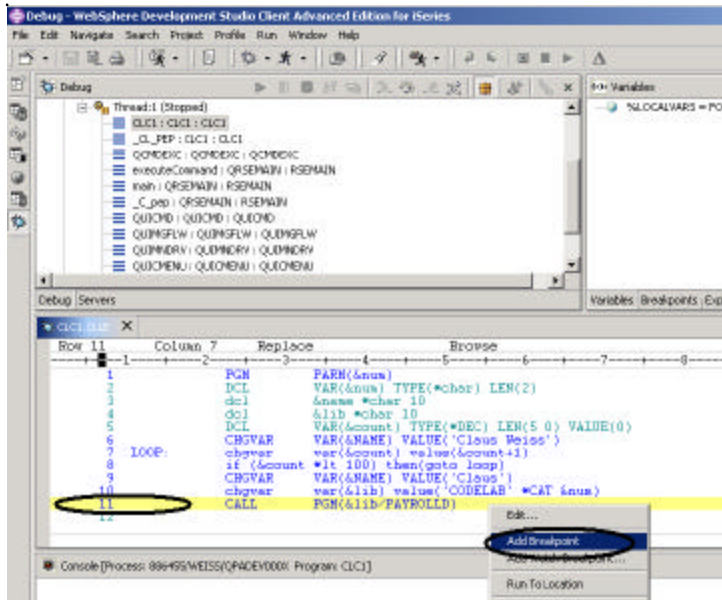


Figure 60: Debugger perspective, adding breakpoint

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

Setting Breakpoints

You can only set breakpoints at executable lines. All executables lines are displayed in blue.

The easiest way to set a breakpoint is to right-click on the line in the Source pane:

- Right-click anywhere on **line 11**
- Select the **Add breakpoint** option from the pop up menu.

To add a conditional breakpoint in the loop when it loops the 99th time:

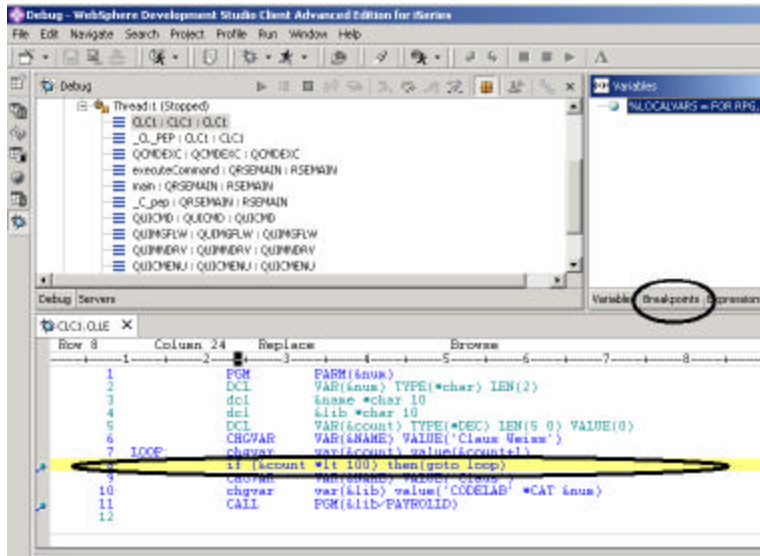


Figure 61: Adding a break point and selecting the breakpoint view

- Right-click on **line 8**
- Select the **Add breakpoint** option from the pop up menu.
- Select the Breakpoints tab in the upper right pane of the workbench as shown in Figure 61

In the breakpoints view:

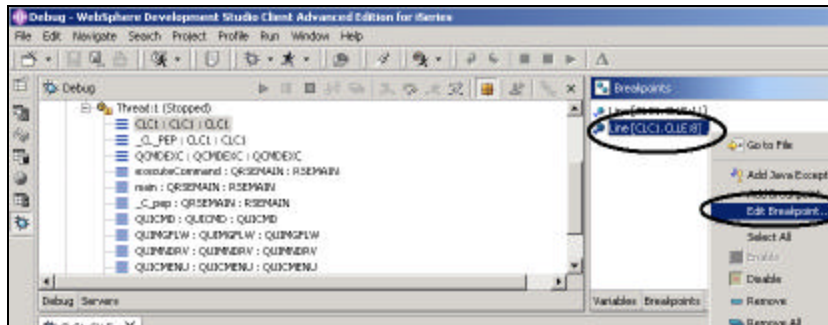


Figure 62: Edit breakpoint in breakpoint view

- Right-click on the **line 8** breakpoint
- Click the **Next>** push button in the Edit breakpoint dialog

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.

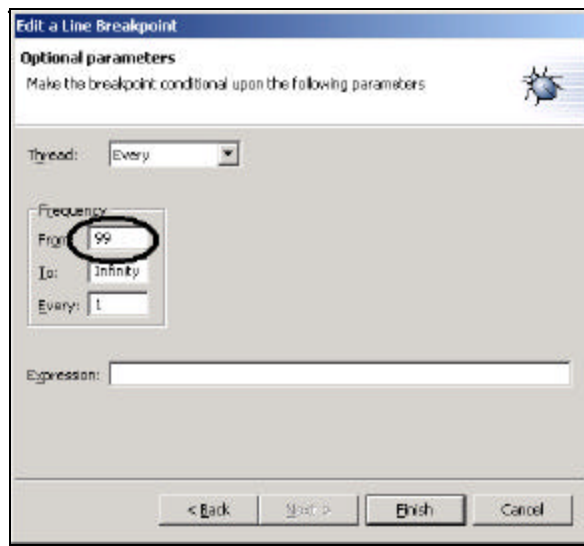


Figure 63: Specify condition for breaking

- Enter **99** in the **From** entry field
- Click the **Finish** push button

Monitoring Variables

You can monitor variables in the Monitors view. In this exercise, you will monitor the variable &COUNT.

In the Source pane

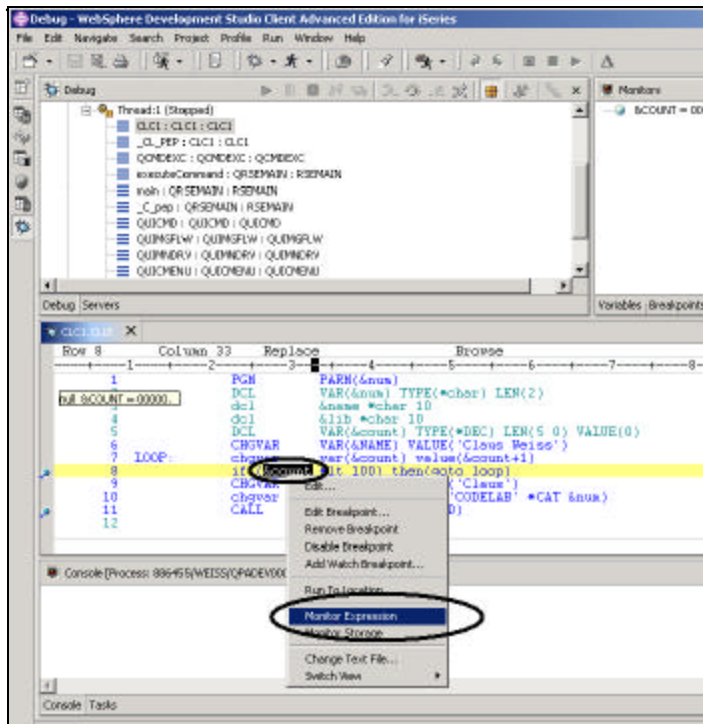


Figure 64: Select variable to be monitored

- Double-click on the variable **&COUNT**,
- Right click **&COUNT**
- Select the **Monitor Expression** option from the pop-up menu.

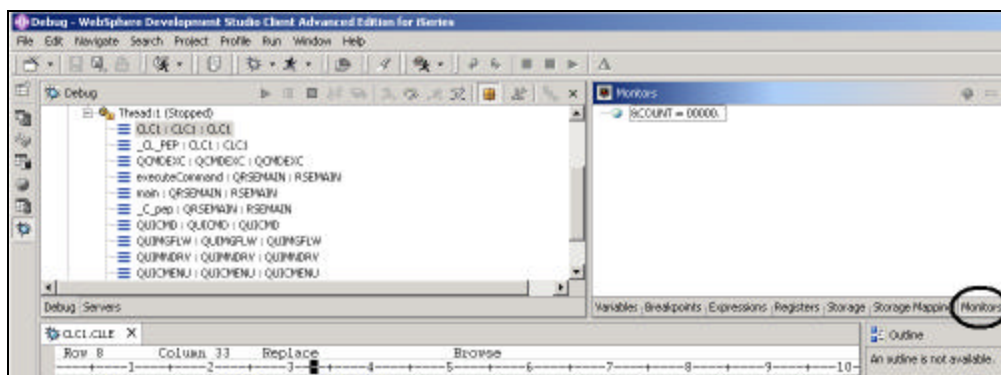


Figure 65: Monitor view with variable **&COUNT**

The variable appears in the Monitors pane

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 little popup dialog.

Running a Program

Now that some breakpoints are set, you can start running the application:



Figure 66: Run push button on debugger toolbar

- Click the **Run** button  from the debug toolbar.

The program starts running and hits 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**.

- Click on the **Run** button again.

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

- Click the **Run** push button once more so that the program hits the breakpoint at line 11.

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 (Step debug) the Payroll program.

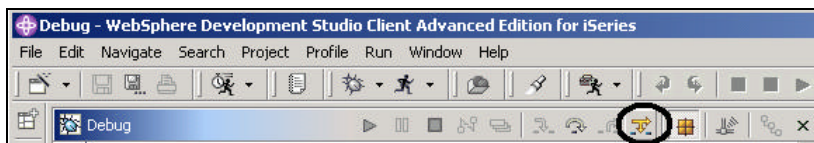



Figure 67: debugger toolbar with Step Debug push button

- Click the **Step Debug** button  on the debugger toolbar. The source of PAYROLLG is displayed.

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

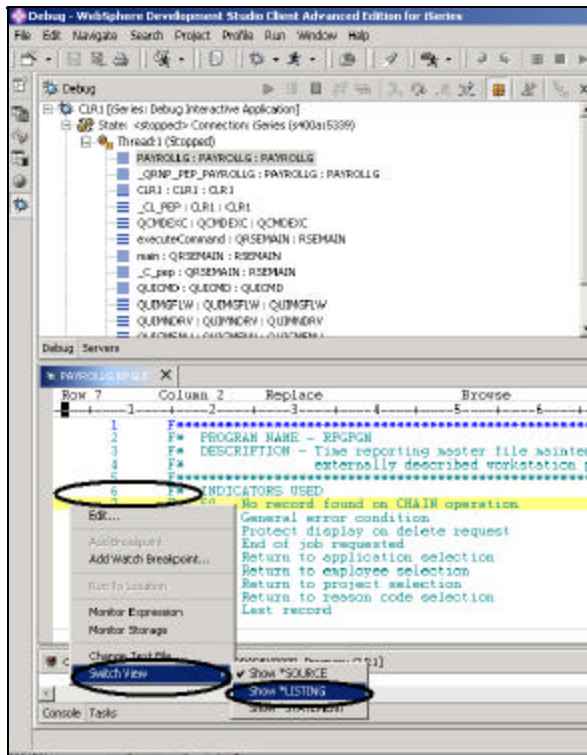
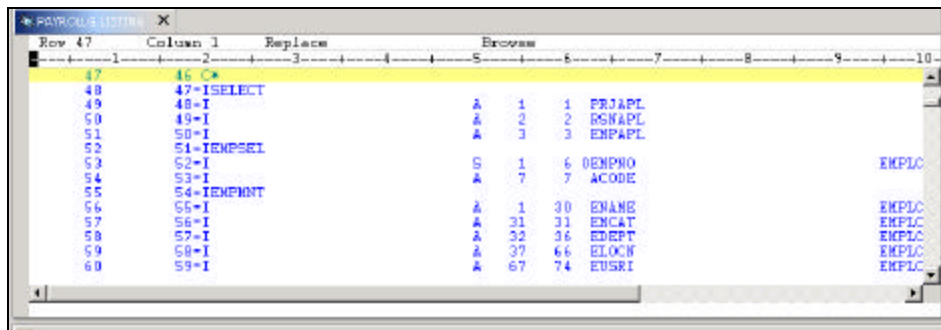


Figure 68: RPG source view

- Right-click *anywhere* in the *source view*
- Select the *Switch view* option from the pop-up menu
- Select the *Show *LISTING* option from the submenu.



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

You don't have any /Copy member in our PAYROLL program but these would also be shown in a listing view. Switch back to the source view.

- From the *Switch view* sub menu, select the *Show *Source* option

The Call Stack

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, procedure and method that is on the stack at the current execution point. Double clicking on a stack entry 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, by clicking on the + sign in front of it.

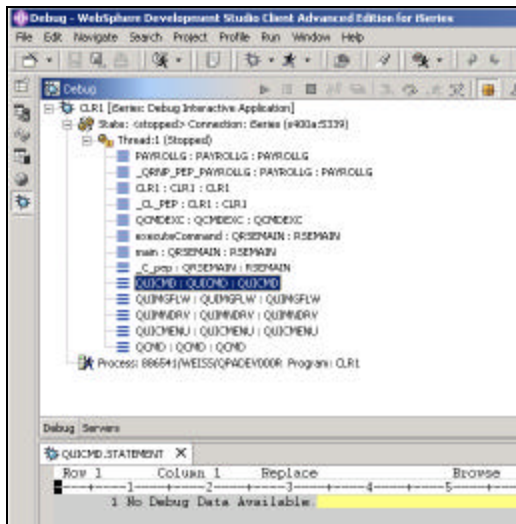


Figure 70: Call Stack in Debug view

It allows you to work with and switch between different programs and/or ILE modules.

Setting Breakpoints in PAYROLLG

- Right-click on **line 55**
- Select the **Add Breakpoint** option from the pop-up menu.

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

- Repeat the above step for **line 56**.
- Repeat the above step for **line 87**

To view all breakpoints, select the **Breakpoints** tab from the top left pane. This view shows all breakpoints currently set in your programs.

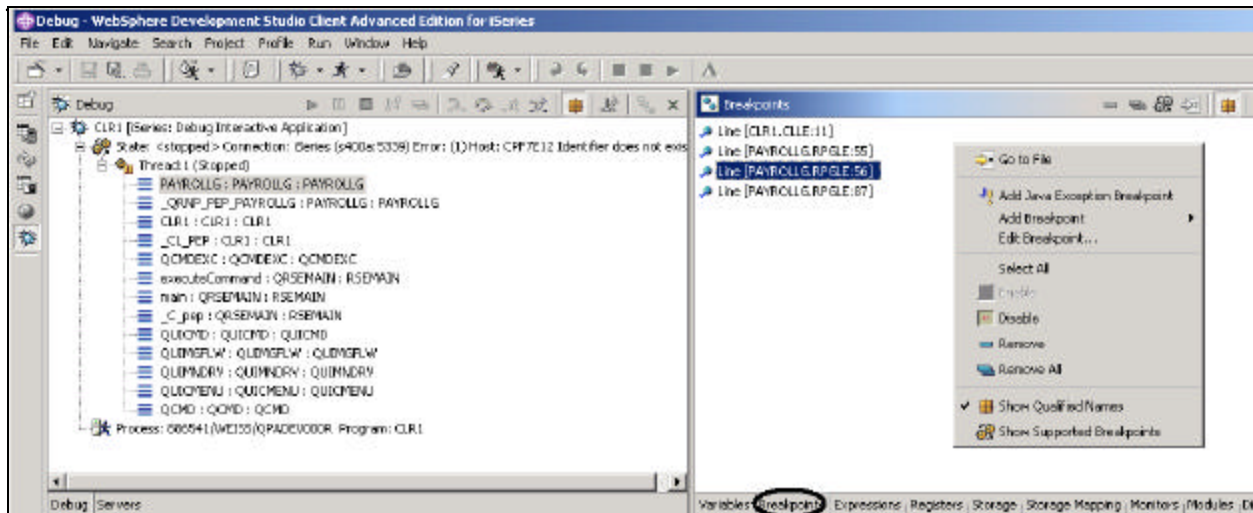


Figure 71: Breakpoints view

This is a convenient place to work with breakpoints. You can delete, disable/enable, or edit a breakpoint. These tasks are available from the pop-up menu when you right mouse click in the view area. Double clicking on any entry brings up the source where the breakpoint is set.

Removing a Breakpoint

It is also easy to manipulate breakpoints from the Source pane.

- Right click on *line 56*
- Select the **Remove Breakpoint** option

The blue icon is removed from the prefix area indicating that no breakpoint is set on that line.

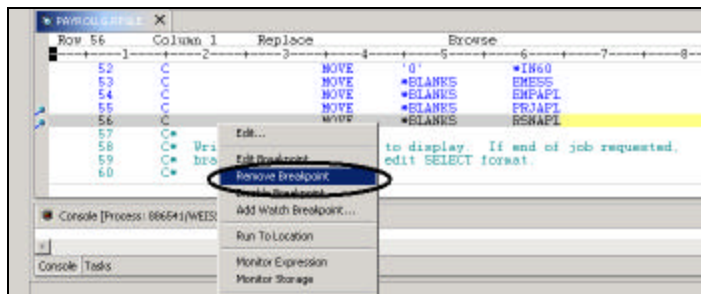


Figure 72: Remove breakpoint

Running the Program

- Click the **Run** button  on the debugger toolbar.

The program starts running and hits the breakpoint at line 55.

- Click on the **Run** button again.

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

- Type an **X** beside the **Project Master Maintenance** option,
- Press **Enter** in the emulation session.

The program hits the breakpoint at line **87**.

Monitoring Variables in PAYROLLG

You can monitor (or change) variables and indicators in the Monitors view. In this exercise, you will monitor variables and change their values.

In the Source pane,

- Double-click on the variable **EMPAPL** on line **87**
- Right click on the **variable**
- Select the **Monitor Expression**

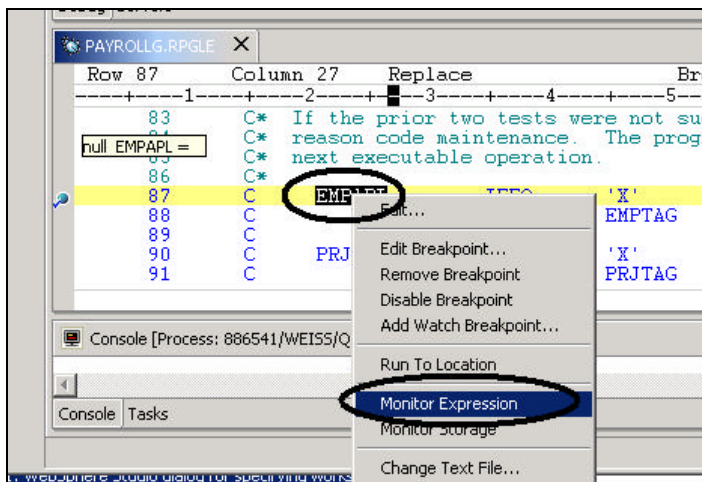


Figure 73: Add variable to monitor

- Click on 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.

- In the same way add the variables **PRJAPL** on **line 90**
- and **RSCDE** on **line 102** to the monitor.

Variable **PRJAPL** equals '**X**' because you did select the Project Master Maintenance option.

In the Monitors view

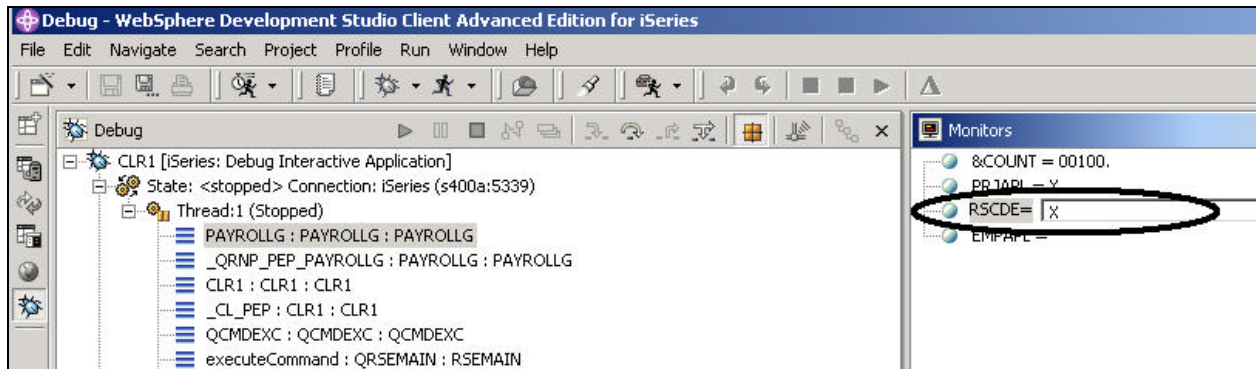


Figure 74: Change variable content

- Double-click on the variable **RSCDE**.

The value changes into an entry field.

- **Type** in the new value **X** for the variable.
- Leave the *entry field* by clicking on another variable.

The variable is successfully changed.

Adding a Storage Monitor

Adding a storage monitor for a variable allows you to view the storage starting with the address where the variable is located. You can choose between two display formats, hexadecimal and character or character only.

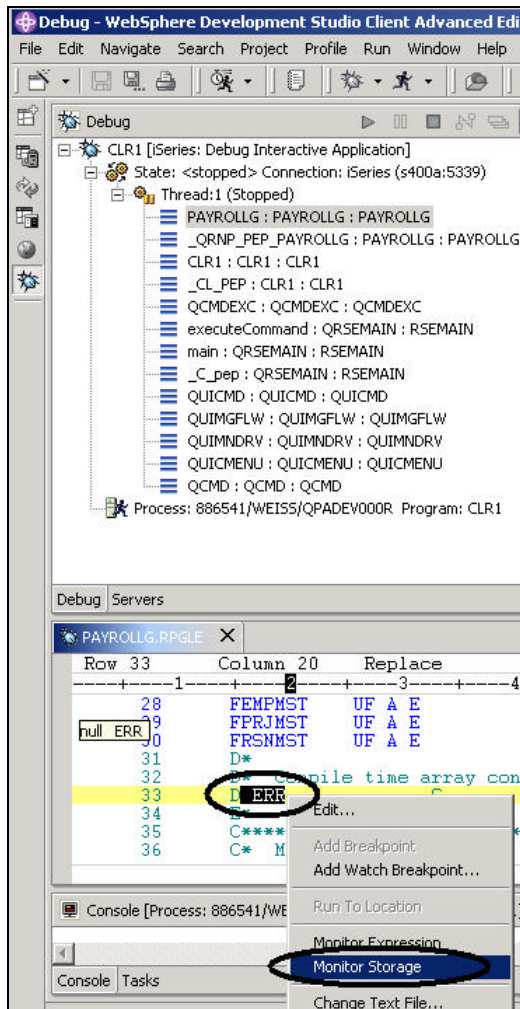


Figure 75: Adding a Storage Monitor

In the Source window:

- Double click on variable **ERR** in **line 33**,
- Right click and select the **Storage Monitor** option from the pop-up menu

A new page is added to the Storage view. The tab shows the name of the variable.

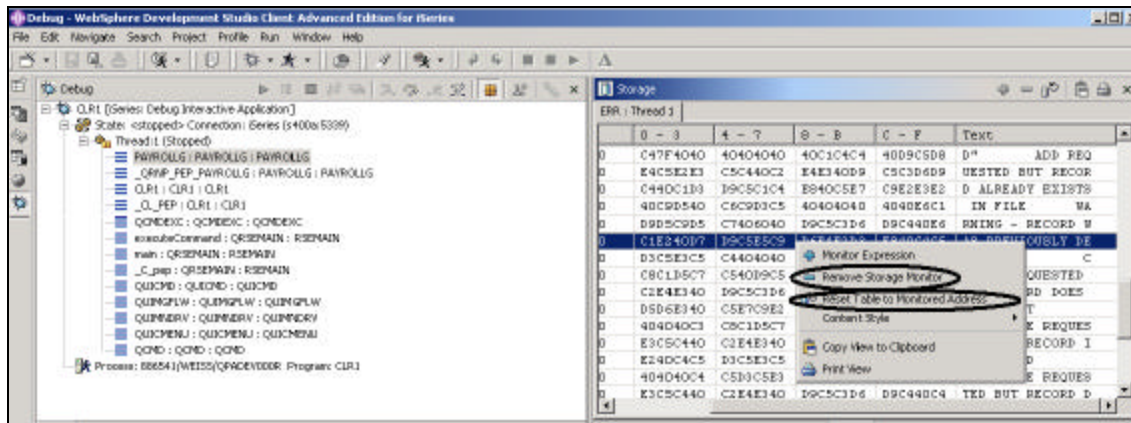


Figure 76: Change storage monitor

- Use the scroll bar on the right of the storage view to *scroll down*. You can see the content of the memory at this moment
- Right-click in the *view area*
- Select the **Reset table to Monitored Address** option to get back to the starting address.
- Right click in the *view area*
- Select the **Remove Storage Monitor** option to remove the storage monitor.

Watch Breakpoints

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

In the Source view:

- go to *line 122*.
- Double-click on variable **IN60* to highlight it.
- Select the **Add Watch Breakpoint** option.

The **Watch Breakpoint** window appears. The **Expression** entry field is pre-filled with the highlighted variable **IN60*.

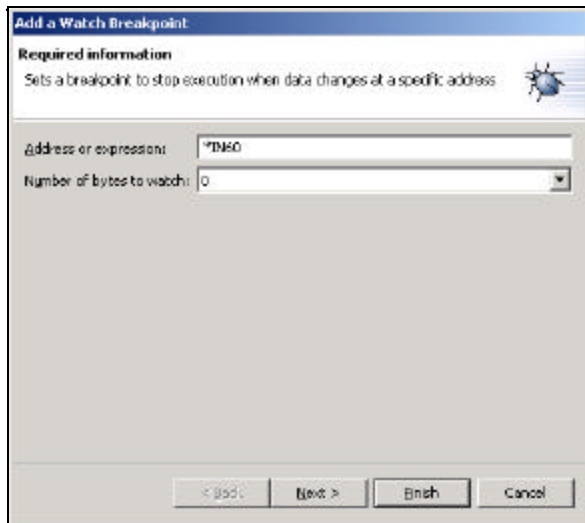


Figure 77: Add Watch Breakpoint

- Click the **Finish** push button.

The watch breakpoint is now set.

- Click on the **Run** button on the debugger toolbar.

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

In the 5250-emulation session:

- Type **123** for **Project Code**
- and **D** (for delete) in the **Action Code** field.
- Press **Enter**.

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

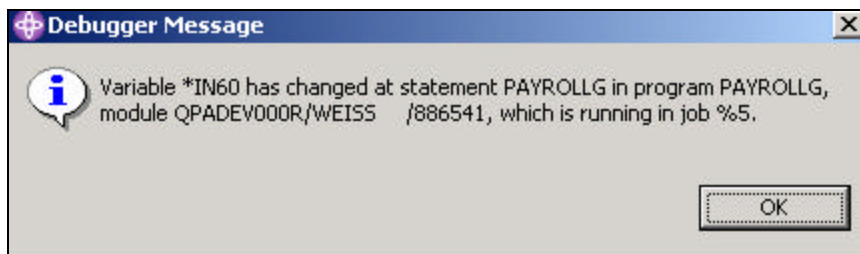




Figure 78: Watch breakpoint message

- Click the **OK** push button

The program stops at line **454**. This line is located immediately after the statement which caused the variable ***IN60** to change.

Stepping Through the Program

The CODE Debugger allows you to step through your source code one line at a time.

Press the step over  or step debug  buttons on the toolbar.

From the **Run** menu option, select **Step Over** or **Step Debug**.

Closing the Debug Session

- Click on the **Run** push button on the debugger toolbar.

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

- Switch to the *5250-emulation session*.
- Press **F3** to end the job.


A message **Debug session terminated** comes up.

- Click the **OK** push button

Close the debugger perspective by performing the following steps:



Figure 79: Close debug perspective

- Right click on its **icon**  on the left task bar in the workbench
- Select the **Close** option from the pop-up menu

This quick tour has demonstrated the main features of the Debugger. Now let's move on to the Remote System Explorer

Exercise 4: Exploring Remote System Explorer

Most of the functionality of the CODE Project Organizer has been replaced by WebSphere Studio functionality with the exception of accessing ADM parts.

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

It allows you to:

- Simplify your work by giving you quick access to lists of iSeries libraries, objects, members, IFS files, Unix files, and local files.
- Use the context-sensitive pop-up menus on these lists to perform actions such as start the CODE Editor, CODE Designer, or Distributed Debugger or other common iSeries actions.
- 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.
- 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.

In the following exercise you will use the **Remote Systems Explorer (RSE)** perspective to work with the iSeries objects that you used in the previous exercise. You will also see how easy it is to perform actions and define your own actions. In short, you'll see how **Remote Systems Explorer** can organize and integrate your work and make that work easier.

Creating a library filter

In the RSE perspective, you now need to get to the iSeries objects you want to work with

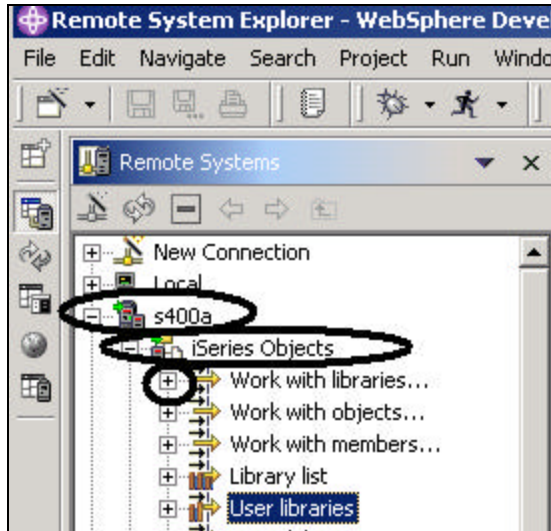


Figure 80: Expand Work with libraries node

In the previous exercises you have worked with the library list filter, now you will create your own library filter. First you will need to specify the library you want to work with:

- Expand the **connection node** that connects to your iSeries host, by clicking on the + sign beside it.
- Expand the **iSeries Objects** node

To create a new library filter

- Expand the **Work with libraries...** Node.

You see this dialog show up

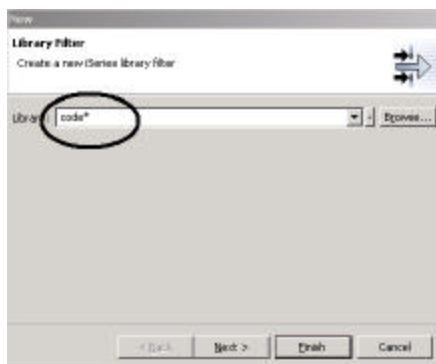


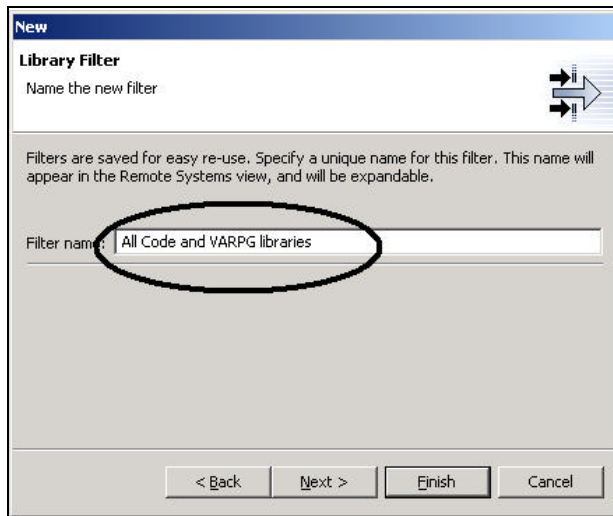
Figure 81: Specify a filter string

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

Specify the first filter string that selects the libraries starting with CODE by

- Keying into the **Library:** entry field **CODE***, using the * wild card character.
- Clicking the **Next >** push button, beside the **Filter Strings** list

The following dialog will show



Specify a name for this filter by:

- Keying into the **Filter name** entry field:
All CODE and VARPG libraries
- Click the **Finish** push button

Back in the **Remote System explorer** tree view you will see the new filter expanded, listing all CODE* libraries, now you need to add the VARPG libraries:

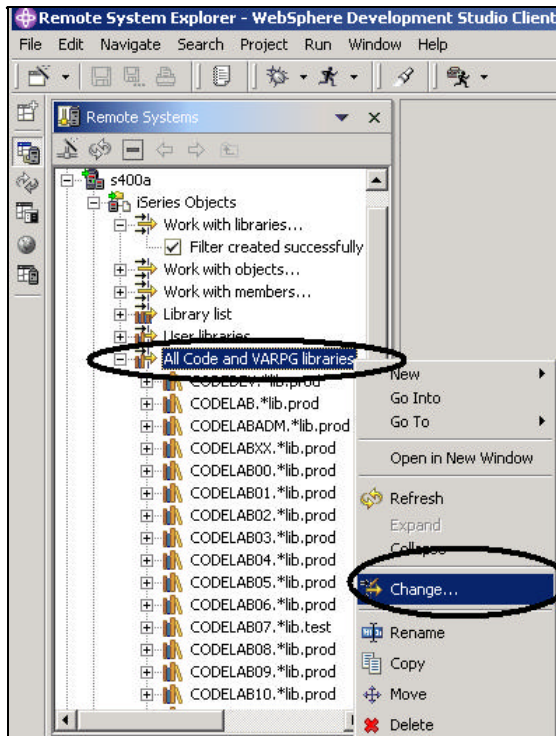


Figure 82: Select the Change option

- Right click on the Filter *All CODE VARPG libraries* icon
- Select the *Change...* option from the pop up menu

The **Change Library Filter** string dialog shows up

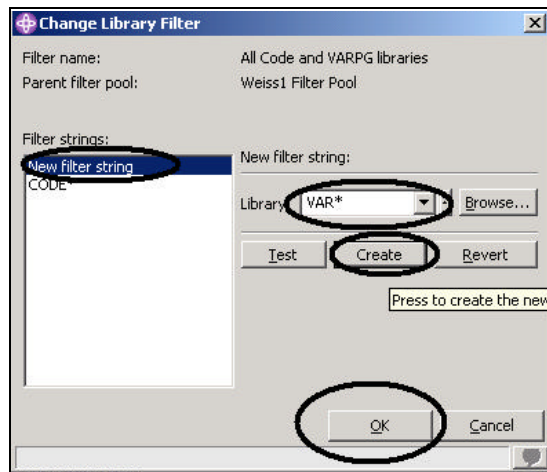


Figure 83: Add second filter string

- Click on the *New filter string* entry in the list box
- Enter *VAR** in the *Library* entry field

- Click the **Create** push button

The **VAR*** filter string gets added to the list box.

- Click the **OK** push button

You are now back in the RSE perspective dialog. You will see the list in the RSE perspective being expanded to include your filter

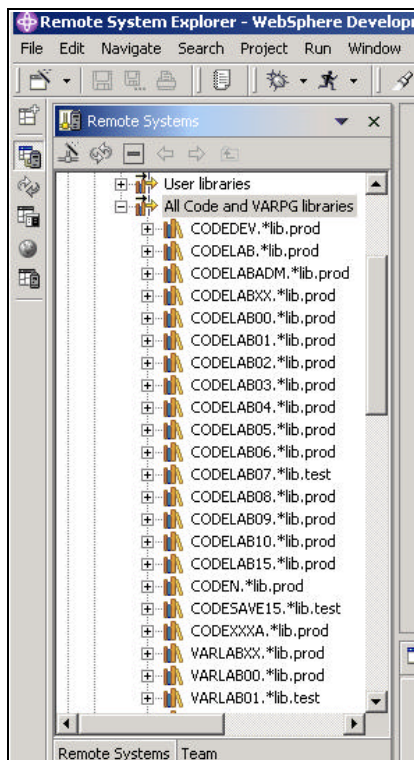


Figure 84: Expanded filter

Now you can work with the libraries directly and can drill down to the object you want to work with.
Now create an object filter

Creating an object filter

In the RSE list, under the connection node you are using,

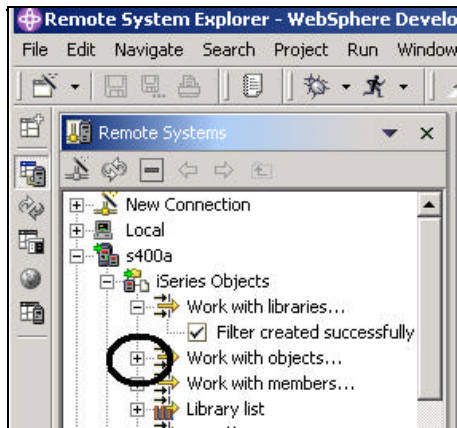


Figure 85: Create object filter

- Find the **Work with objects...** node,
- Expand *this node*

This will show the *New Object Filter* dialog

Create a filter to show all your source files in your **CODELABxx** library:

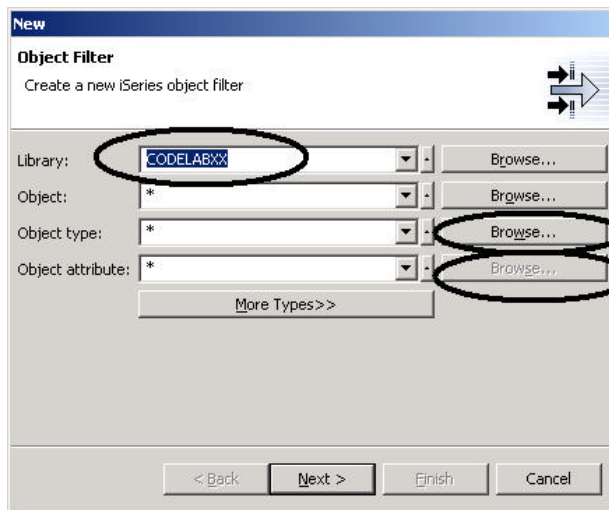


Figure 86: Specify filter string

- Enter **CODELABXX** in the *Library* entry field
- Click the **Browse** push button beside the *Object type* entry field

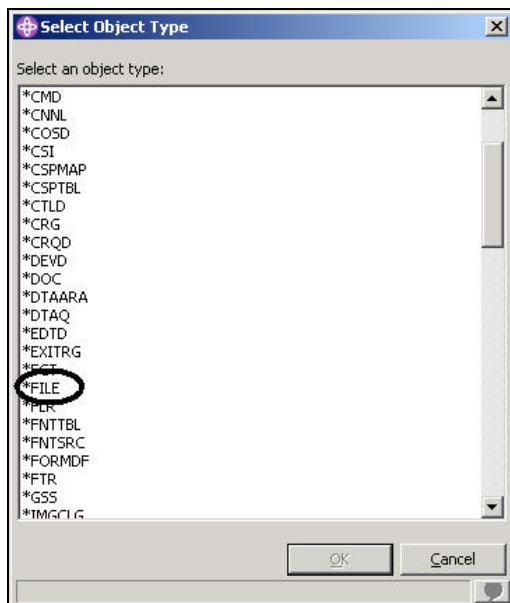


Figure 87: Select Object Type

- Select the ***File** object type from the list
- Click the **OK** push button

Back in the Object filter dialog

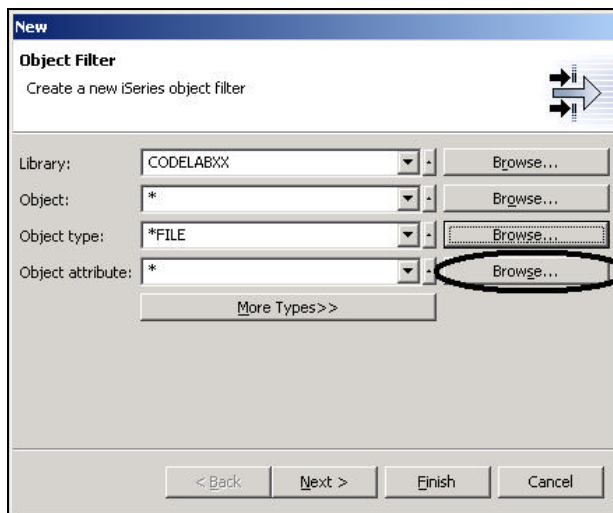


Figure 88: Browse for Object Attribute

- Click the **Browse** push button beside the **Object Attribute** entry field

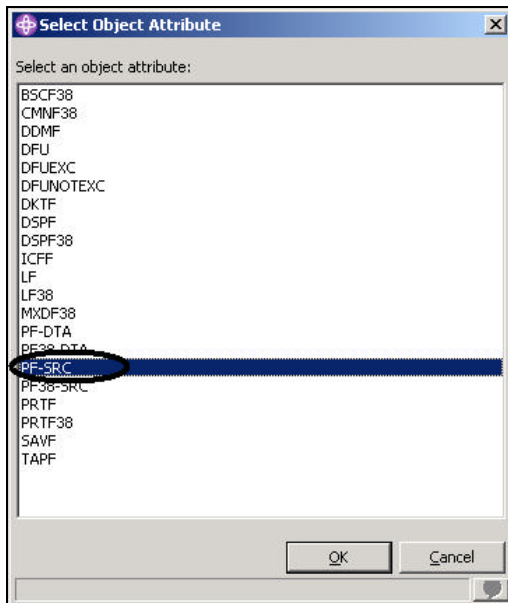


Figure 89: Select filter Object Attribute

- Select **PF-SRC** from the Object Attribute list
- Click the **OK** push button

On the Object Filter dialog:

- Click the **Next >** push button

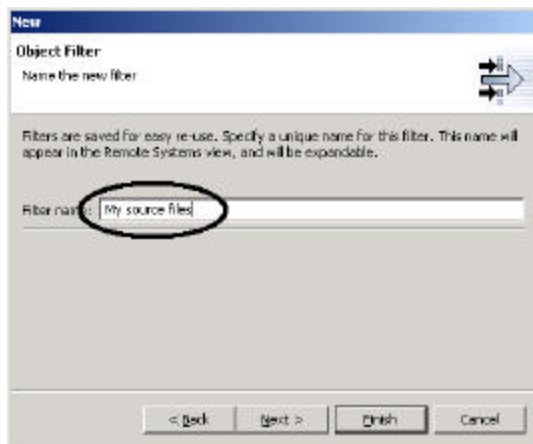


Figure 90: Specify filter name

- Specify the Filter name: **My source files**
- Click the **Finish** push button

The new filter will show up in the RSE list

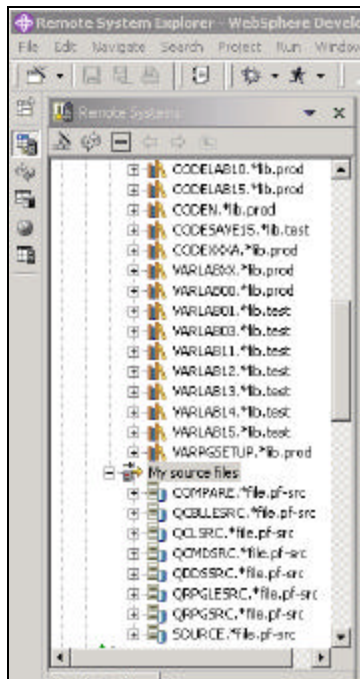


Figure 91: Object filter shows all source files in library

You got the idea how to create filters and tailor your development environment. Filters can also be specified for non iSeries servers and your local system.

Tip: If you end up with too many filters, look into creating filter pools, which allow you to group filters and tailor the filters to your environment.

Now you can work with the objects you have in your RSE list like you worked in PDM with libraries, objects, or members.

Assuming you want to edit the member **Payroll**, in QRPGLESRC you just:

- Expand node **QRPGLESRC**
- Right mouse click on the member **Payroll**
- From the pop up menu select **Open with**
- From the sub menu select **LPEX editor**

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 RSE list by using the pop up menu options on this member.

You can also create your own actions in addition to the default actions.

Creating a user action

In PDM you can create user actions in addition to the pre-supplied system actions, in the RSE you can do the same.

In the RSE perspective

- Locate your **connection node** and the **iSeries Object** node underneath it.

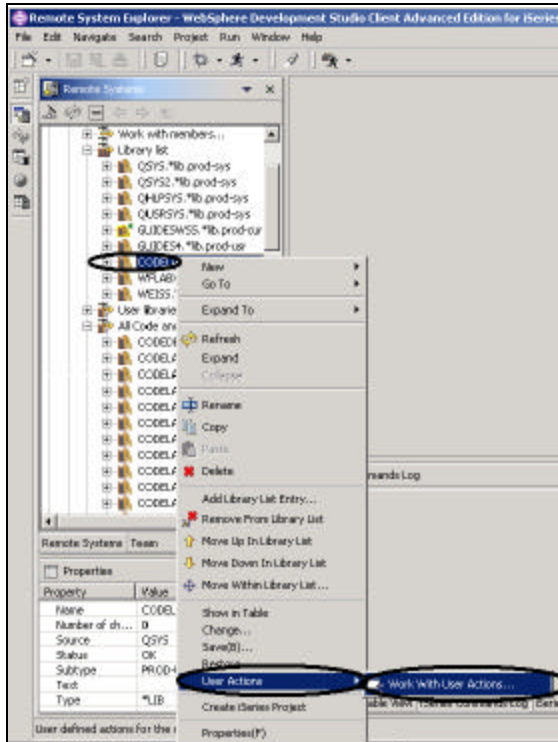


Figure 92: Work with user actions

- Expand the **Library list** filter
- Right-click on the **CODELABxx** node
- Select the **User Actions** option from the pop up menu
- Select the **Work With User Action...** option from the sub menu

You will see the **Work with User Actions** dialog

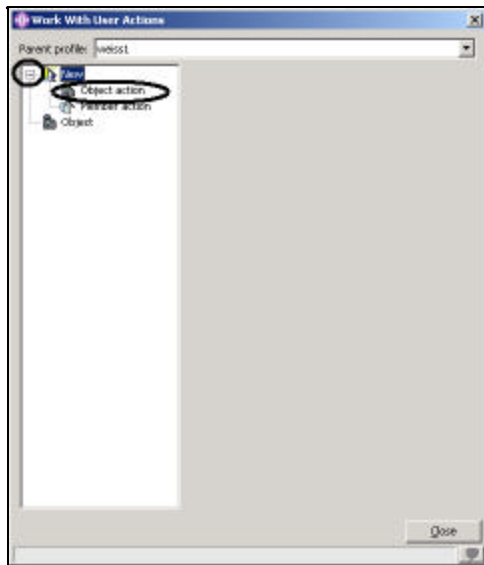


Figure 93: Work with User Actions dialog

- Expand the **New** node in the list, if it is not expanded already
- Click on the **Object action** node

The *Work with User Actions* dialog appears

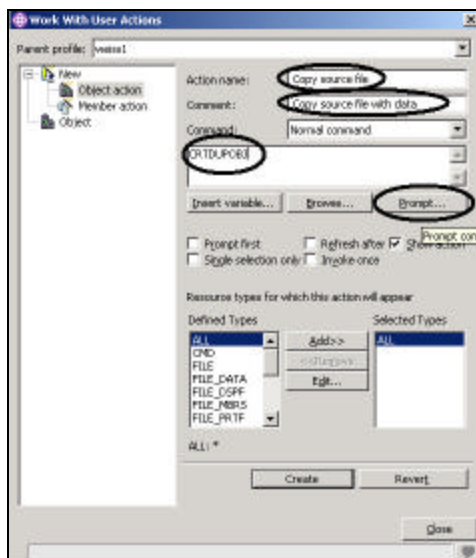


Figure 94: Create an Object action

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

- Enter a name for the user action in the **Action** entry field: **Copy source file**
- Enter a comment in the **Comment** entry field

- Key in the command to execute **CRTDUPOBJ**
- Click the **Prompt** push button, to get the command prompter for this command

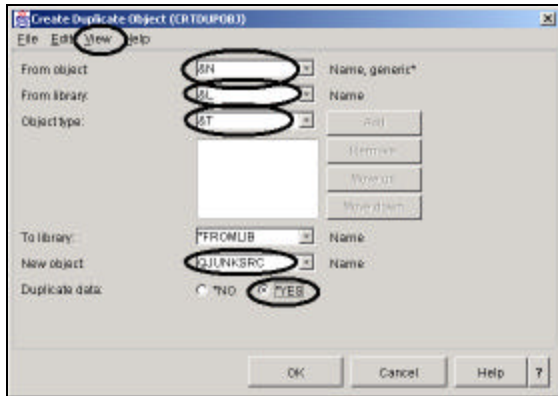


Figure 95: Prompt for CRTDUPOBJ command

This is the command you will be running

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

- Enter **&N** in the **From Object** entry field, to indicate to use the name of the selected object in the RSE list
- Enter **&L** in the **From Library** entry field, to pick up the library name form the selected object
- Enter **&T** in the **Object Type** entry field, to pick up the object type from the selected object
- Enter **QJUNKSRC** in the **New object** entry field

To get the additional Duplicate Data parameter:

- Select the **View** option on the menu bar of this dialog
- Select the **All parameters** option from the pull down menu

Now the **Duplicate Data** parameter is also shown on the prompt dialog

- Select the **Yes** radio button for the **Duplicate Data** parameter
- Click the **OK** push button

You will be back at the **Work with User Action** dialog

To refresh the RSE list after the action has been run:

- Select the **Refresh after** check box

Tip: Pressing the **Insert variable..** push button, brings up a list of valid replacement variables with explanation what they do.

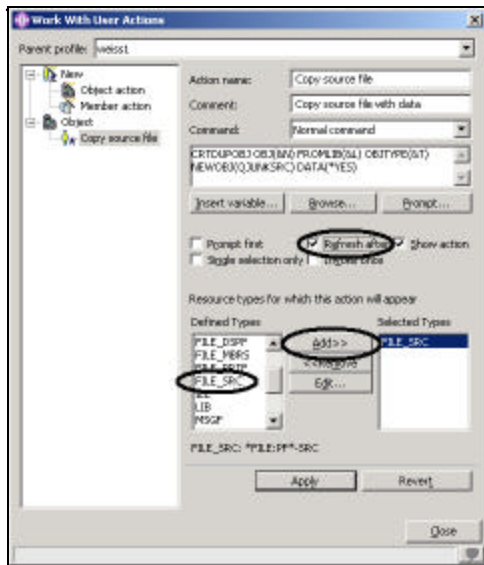


Figure 96: Limit user action to source files

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.

To specify this restriction

- Locate the **Defined Types** listbox
- Select **FILE_SRC** in the list
- Click the **Add>>** push button beside the list

FILE_SRC is now one of the Selected Types, actually since you only selected this one it is the only one.

- Select the **Refresh** check box to refresh the RSE list after the user action has been performed
- Click the **Close** push button

Now, only when you right mouse click on a source file, will this user action appear on the pop up menu selected, for any other object type it will not appear.

Back in the workbench and the RSE perspective, give it a try

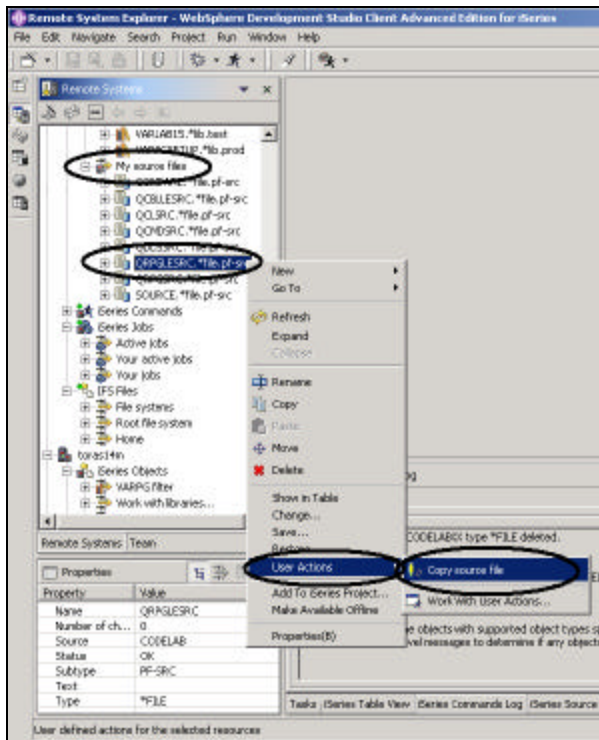


Figure 97: User action shows in pop up menu

- Locate your filter **My Source files**
- Expand the **filter**, if it is not already expanded
- Right mouse click on the **QRPGLSRC** file
- Select **User Actions** from the pop up menu
- Select **Copy source file** from the sub menu

The file gets duplicated and the list gets refreshed, your new source file will show in the list.

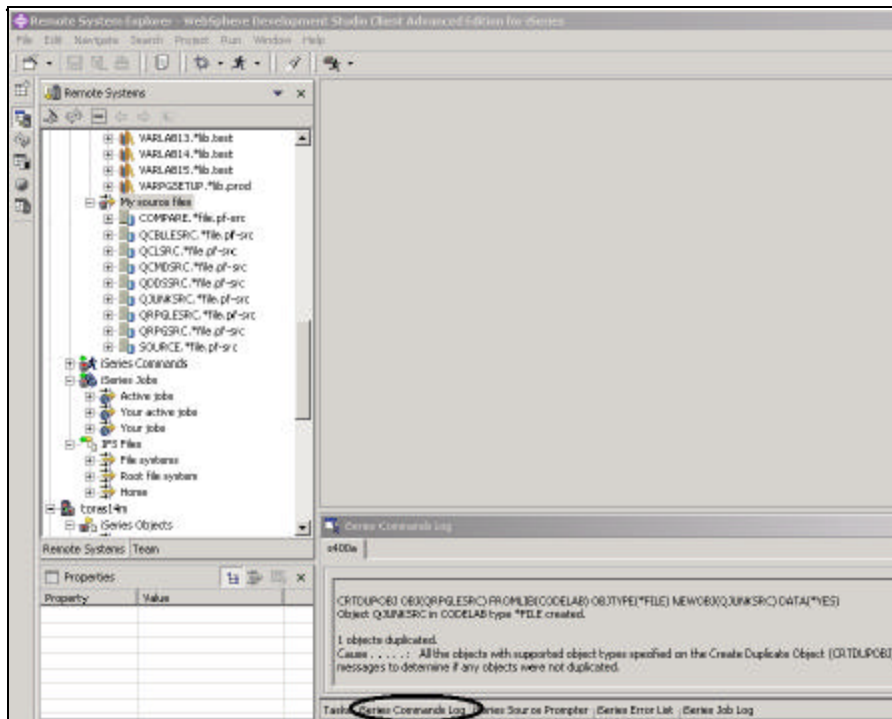


Figure 98 CRTDUPOBJ command in user action ran successful

You can check the messages of the CL commands you are running in the RSE job by looking at the Commands log in the right hand side bottom pane in the work bench.

To delete the source file **QJUNKSRC** that you just created:

- Right mouse click on its **node** in the list
- Select the **Delete** option from the pop up menu.

Running commands from the RSE

The Command entry is part of the RSE table view

- Check if you have a iSeries **Table view** tab in the bottom right view pane where your command log appeared in the previous task.
- If you have it click on *it*
- If you don't have it
 - In the RSE tree view, right mouse click on *your source file filter*
 - Select the **Show in table** option from the pop up menu

Now in the iSeries Table view you can run commands on the iSeries server that the table is connected to

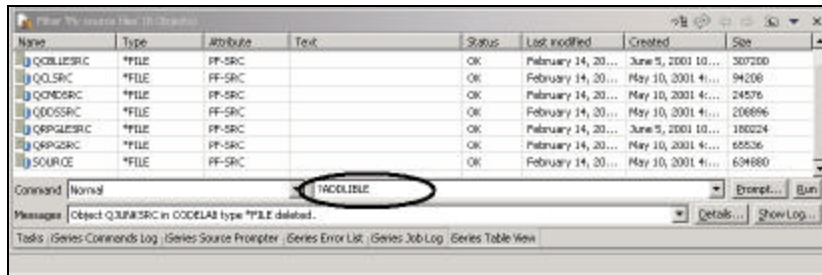


Figure 99: Table view with command entry

In the **command** entry field

- Key in an iSeries command for example **?ADDLIBLE**
The question mark is there to display a prompt screen

Tip: Instead of specifying a question mark you could use the Prompt push button

- Click the **Run** push button

The command prompt dialog for the ADDLIBLE command appears

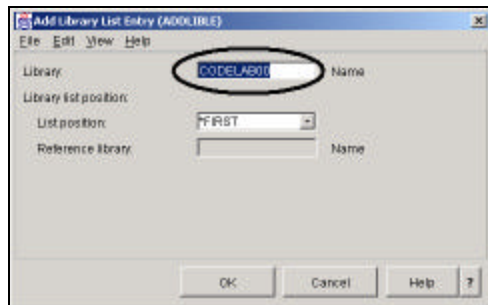


Figure 100: Command prompt dialog

In the prompt dialog:

- Key in **CODELAB00**, that will add this library to the library list of your RSE job on the iSeries server.

The messages field will confirm the successful completion of this command

To get to the command history, similar to F9 on the green screen, press the little arrow on the command entry field

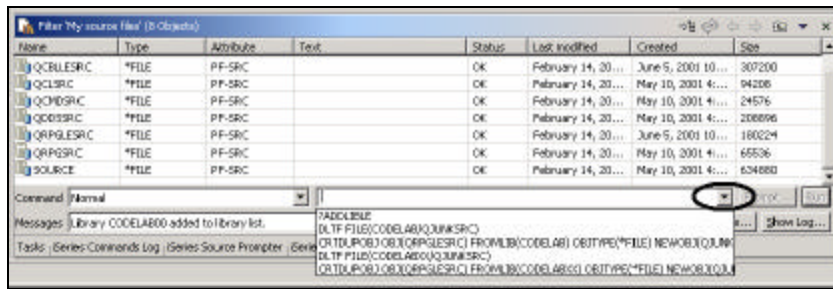


Figure 101: command history

You could also use the **iSeries** commands node in the RSE view underneath the iSeries objects node and run predefined commands or define your own commands.

We hope this exercise gave you a first taste of the capabilities the RSE perspective provides to iSeries Application Developers

Congratulations!

You have successfully completed the Introduction to RSE lab. More material can be found at our web site at <http://www.ibm.com/software/ad/iseriess>

Happy Coding!