IBM WebSphere Development Tools for iSeries

CODE/400 - Introduction

Session Id   230129
Agenda Key 17LF

Inge Weiss and the iSeries Team

iweiss@ca.ibm.com

IBM Toronto Laboratory

Version 5 Release 1

homepage: http://www.ibm.com/software/ad/wdt400
newsgroup: news://news.software.ibm.com/ibm.software.code400

Seventh Edition (October, 2001)

The information contained in this document has not been submitted to any formal IBM test and is distributed on an "as is" basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will result elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Comments concerning this notebook and its usefulness for its intended purpose are welcome. You may send written comments to:

IBM Canada Ltd.

8200 Warden Avenue, Markham, Ontario, L6G 1C7

Attention: Inge Weiss, CODE Advanced Topics

or e-mail to**: iweiss@ca.ibm.com**

Technical Information

For more technical information on CODE or WebSphere Development Tools for iSeries contact either
    Dave Slater at slater@ca.ibm.com
    Claus Weiss at weiss@ca.ibm.com

Education

CODE/400 courses:

S6186   CODE/400 for iSeries - Basic (2 days)
S6205   CODE/400 for iSeries - Advanced (1 day)

Trademarks

IBM is a registered trademark of International Business Machines Corporation.

The following are trademarks of International Business Machines Corporation.

iSeries
AS/400
VisualAge
DB2/400
ILE
Integrated Language Environment
IBM
OS/400
RPG/400
WebSphere

Trademarks of other companies as shown

'Intel'   'Intel Corporation'

'Microsoft'   'Microsoft Corporation'

'Windows'   'Microsoft Corporation

Overall Lab Guide

The objective of the CODE Introduction is to explore some of the basic features of the tools that make up CODE, Editor, Designer, Debugger and Project Organizer. 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:

- Connect to the iSeries.
- Open source in the Editor, make changes, and use different ways to manouver through the source.
- Invoke the Program Verifier and work with the Error list.
- Start a host compile using the Program Generator.
- 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 Project Organizer, invoke actions and create new actions.

The exercises for each tool, Editor, Designer, Debugger, and Project Organizer 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 similar tasks. Some of the names and directories 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.

Introduction to CODE

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

Workstation Components
- WebSphere Studio for iSeries
- WebFacing
- VisualAge RPG
- CODE
- VisualAge for Java for iSeries

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; and completely organize your programming projects.

The CODE product includes the following tools:
- **CODE Editor**
  A powerful language-sensitive editor that you can easily 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.
- **CODE Program Generator**
  An interface that allows you to submit requests to the iSeries to compile, bind, or build objects on the host. The tool gives you a graphical interface to all the compile options available for all the supported create commands (CRTxxx).
- **CODE Designer**
  A rich graphical interface that makes designing or maintaining display file screens, printer file reports, and physical file databases easy and fun.
- **IBM Distributed 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.

- **CODE Project Organizer**

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

In this session, you will learn some of the basic features and functionality of each of these tools. We are confident that CODE 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 CODE

The CODE tool of the WebSphere Development Tools for iSeries (WDT400) product consists of two parts:
1. The 'back-end' which resides on the iSeries.
   This part is responsible for handling all the workstation requests such as getting or saving source members, etc. The back-end is shipped with the ADTS host utilities (SEU, PDM, DFU, SDA, ...).
2. The 'front-end' which is installed on your workstation.
   These workstation files can be installed from:
   - a local CD drive
   - a LAN drive (assumes that an installable image has been set up on the LAN)
     an iSeries (assumes that the workstation files have been installed to the iSeries ifs).
     The workstation install uses the Windows Installer.

The minimum hardware requirements for CODE are an Intel® Pentium II processor or faster with 64MB of memory, a SVGA 800 x 600 monitor, CD-ROM drive, and a mouse or pointing device. The recommended workstation hardware is a processor with 96MB of memory, and a SVGA 1024 x 768 monitor. A complete install of CODE including the help files uses about 235MB of disk space.

Connecting to the iSeries

Communications between the iSeries and your workstation can be configured for:
•TCP/IP communications using the native Windows built in TCP/IP support. You can use any 5250 emulator that supports TCP/IP.
• SNA (System Network Architecture) / APPC (advanced program-to-program communications). This setup requires either: Client Access Express; Personal Communications; or RUMBA to handle the communications.

For this lab session, you will use TCP/IP communications. On the workstation, the CODE daemon needs to be running in order to allow TCP/IP communication with the iSeries. When your PC is restarted after the CODE installation, the CODE daemon is started for you. If you closed the daemon or want to start it manually, you can do so from Start → Programs → IBM WebSphere Development Tools for iSeries → Communications → Communication Daemon.

___ 1. Ensure that the Code daemon is running.
This program waits and listens for an iSeries to contact it on a specific TCP/IP port and then makes a connection.
An icon will appear in your system tray (bottom right of your screen).
You can interact with CODE communications by using the pop-up menu of this icon.
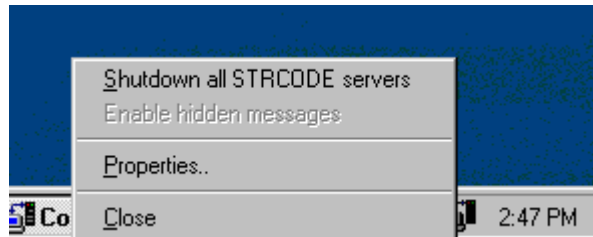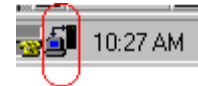
___ 2. **Start** a 5250-emulation session.
___ 3. Sign on to the iSeries. Your userid and password should both be **CODELABxx** where **xx** is your workstation number (01, 02, etc.). Instead of the **Enter** key you may have to use the **Ctrl** key in your 5250-emulation session.
___ 4. At the iSeries command line type:
   STRCODETCP
and press **Enter** (or Ctrl).
This will call a CL program which automatically figures out which IP address your emulator is using and invokes the STRCODE command that is shipped with the product. This CL program can be found in the QCLSRC source file in the CODELABxx library.

You should see a screen that has **EVFCLOGO** in the upper left-hand corner. The CODE server is now active and waiting for commands from the workstation.

If you did not have this CL program, you could type or prompt the STRCODE command:

STRCODE RMTLOCNAME(PC_hostname) CMNTYPE(*TCPIP)

```
                        Start CODE (STRCODE)

 Type choices, press Enter.


 Host server name . . . . . . .   OS400        Character value
 Remote location name . . . . .   PC hostname_____
 Communications type  . . . . .   *TCPIP       *PRV, *APPC, *TCPIP
```

The PC_hostname would be either the TCP/IP host name of your PC or its dotted IP address in quotes. For example '9.21.99.99'. You can determine your PC host name, by typing :

hostname

at a DOS command line.

CODE does not require you to be continuously connected to an iSeries -- many of its features are designed to function in 'disconnected' mode as well. This gives you the ability to develop your applications at home or while on the road (oh joy!). In fact, we've added several features, like the caching of iSeries information (copy files, database reference fields, etc.) on your workstation, to help you work in 'disconnected' mode.

The CODE Editor

The CODE Editor has many powerful features that make it the premiere editor for iSeries programmers:
- It supports RPG/400 and ILE RPG, COBOL and ILE COBOL, C, C++, DDS, CL, REXX, HTML and Java language-sensitive editing.
- Seamless access to iSeries source members, ADM parts, and local files.
- Token highlighting of source code elements.
- SEU-like prompting to help you edit column-sensitive languages such as RPG and DDS.
- Built-in syntax checking and program verification (a compile with no object generation) on the workstation for RPG, COBOL, and DDS. With this you can work while disconnected from an iSeries -- saving iSeries cycles and saving you time.
- Lots of on-line help including programming guides and language references.

In this section you will be introduced to some of the features which make the CODE Editor so powerful. For the purpose of the exercise we will be using some ILE RPG source. Don't worry if RPG is unfamiliar to you. Let's get started.

Starting the Editor

The editor can be started from the Windows Start menu, a DOS prompt, or CODE Project Organizer:

___ 1. From the **Start→ Programs→ IBM WebSphere Development Tools for iSeries→ IBM CODE400** menu, select **CODE Editor**. The CODE Editor appears.

```
CODE - Untitled Document 1                              _ □ ×
File  Edit  View  Actions  Options  Windows  Help
Row 1          Column 1        Insert
-----+----1----+----2----+----3----+----4----+----5--
00001
IBM Live Parsing Editor
```
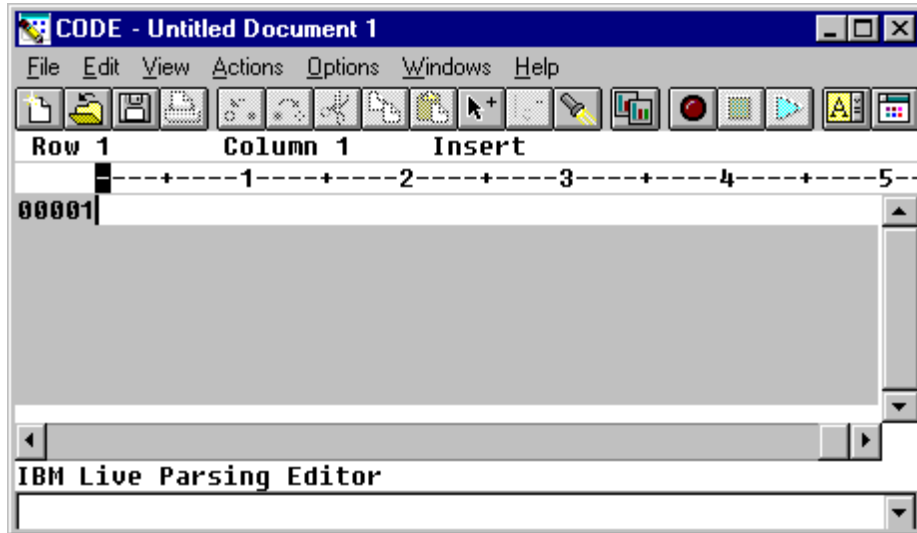
Enabling Extras

___ 1. In the editor, select **Actions→Enable Extras...** The Enable Extras dialog appears.
___ 2. Just hit **OK** on the dialog. The next time the editor gets refreshed, when opening a file for example, the Extras menu will be added to the menu bar.

Opening a Source Member

The editor provides seamless access to workstation files and to source members on the host.

• Source members are automatically downloaded to the workstation.
• Has member locking that is similar to that of SEU.
• iSeries members and local files have a source type (C, DSPF, RPGLE, ...) associated with them. This allows the editor to load the file in the correct language environment.
• You need an iSeries connection to edit iSeries files directly. However, if you download the files to your workstation then you can work on them without an iSeries connection and then upload them whenever you want.

Now We're Ready to Open a Member

___ 1. From the **File** menu, select **Open...** This will bring up the Open dialog.
___ 2. **Click** the + beside **<OS400>** to show the iSeries library list.
___ 3. **Click** the + beside **CODELABxx** to show the library.
___ 4. **Click** on the source physical file **QRPGLESRC** to show its contents on the right.
___ 5. **Click** on **PAYROLL RPGLE**.

___ 6. **Click** on the **OK** push button. The source will be loaded into the editor. Notice also, that the Extras menu now has been added to the menu bar.
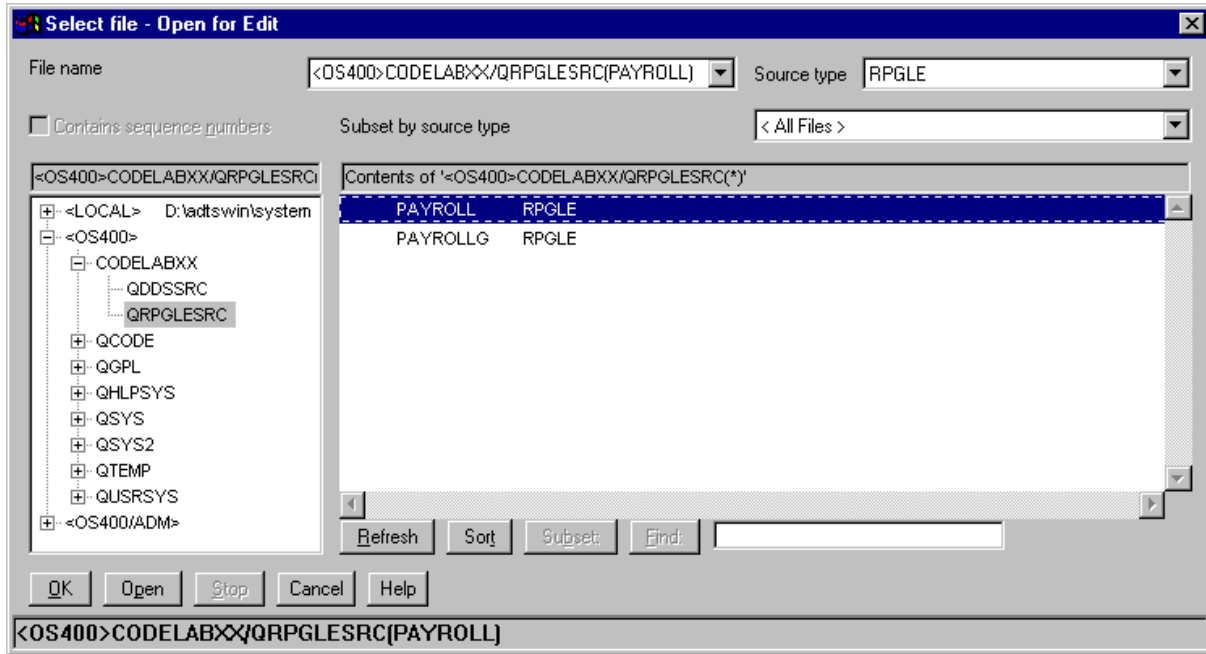
If you know the name of the source then you can load it without starting the Open dialog:

___ 1. Press **Esc**. The cursor moves to the command line at the bottom of the CODE Editor window.

___ 2. Type in the following edit command in the Command Line:

LX <OS400>CODELAB*xx*/QDDSSRC(MSTDSP)

where '*xx*' is your workstation number. .


```
lx <os400>codelabxx/qddssrc(mstdsp)
```
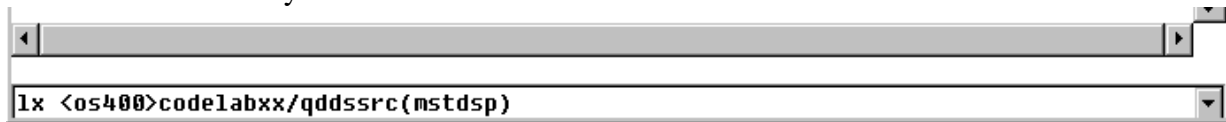
___ 3. Press **Enter**. The display file source appears in the editor.

___ 4. **Click** on the blue drop-down list immediately underneath the toolbar. Notice the drop-down list of files. You can use this list to quickly switch between any number of files loaded in the editor. Select **<OS400>CODELAB*xx*/QRPGLESRC(PAYROLL)** to switch back to the RPG .

```
CODE - <OS400>CODELABXX/QDDSSRC(MSTDSP)                        _ □ ✕
File  Edit  View  Actions  Options  Windows  Help  Extras

<OS400>CODELABXX/QDDSSRC(MSTDSP)                                    ▼

<OS400>CODELABXX/QDDSSRC(MSTDSP)
Untitled Document 1
<OS400>CODELABXX/QRPGLESRC(PAYROLL)
000200    A*  89/04/10  16:28:41      SING        REL-R01M02  5728-PW1
000300    A*****************************************************************
000400    A*  THIS DISPLAY FILE PROVIDES MAINTENANCE FORMATS FOR ALL THE
```
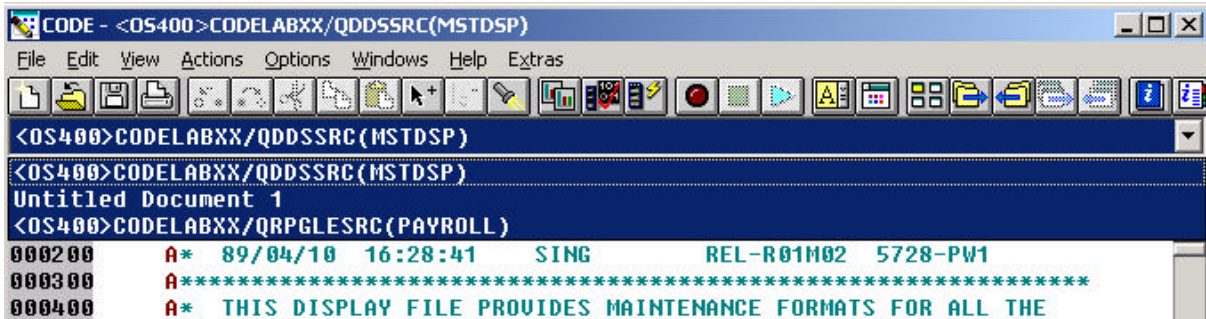
Basic Editor Features

The CODE 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, overlay, 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
CODE provides special support for insertion and deletion in column-sensitive languages.
Important! If you are using WDT/400 V5R1 with Service Pack 4, do the following:
___ 1. Select the **Extras** menu and click on **Column sensitive editing**.
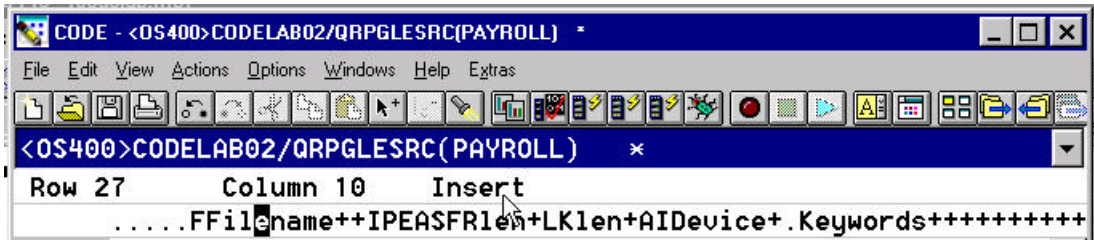***If you are using any older version of CODE, do the following:***
___ 1. **Press** the **Esc** key to move the cursor to the CODE command line.

___ 2. **Type:**
       CODE FIELDS ON
     Press Enter.

Now let's see what this does.
___ 1. **Move** the cursor to **row 27, column 10**.
___ 2. **Make sure** the editor is in **Insert mode**. If the status area says 'Replace', **press** the
     Insert key.

___ 3. **Hit the spacebar** 3 times. Notice that only the filename is shifted but none of the other
     columns to the right are affected.

___ 4. **Press 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:
___ 1. Move the cursor into the gray sequence number area to the left of the edit area.
___ 2. On any sequence number type **dd**
___ 3. Go down a few lines and type **dd** again and press **Enter**. Notice that the lines have been
     deleted.
___ 4. Now type **i5** in the sequence number area. Make sure the cursor is within the sequence
     number area and 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.
___ 1. From the **Edit** menu, select **Undo.** Notice that the 5 new lines disappear.
___ 2. Do another undo action by pressing **Ctrl+Z**. Notice that the deleted lines reappear.
___ 3. From the **Edit** menu, select **Redo.** 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:
___ 1. From the **File** menu, select **Close view**. A dialog will appear asking you to save the
     latest changes. **Click** the **No** button.

___ 2. From the **File** menu, select **<OS400>CODELABxx/QRPGLESRC(PAYROLL)** to reload the original source.

## 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 CODE Editor window.

___ 1. **Position** the cursor over the word **MOVE** in row **56** of the ILE RPG source.
___ 2. **Press F1**. Language-sensitive help for the MOVE operation code appears.
___ 3. Scroll to the very top of the Help page and press the **Synchronize** button in the upper right hand corner. A help navigation bar appears on the left side.
___ 4. Play around in the help window to see what else is available.
___ 5. Minimize the Help window.
___ 6. Select the **Help** menu, to see what help is available. The **ILE RPG help** menu item will show you all the on-line help that is available for ILE RPG. Select any menu item to find more RPG information. Minimize the Help window when you are done.

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.

___ 1. **Move** your cursor to the D-spec on row 33.
___ 2. From the **Edit** menu, select **Prompt** (or press F4). A window appears showing the specification line broken down into its individual fields.
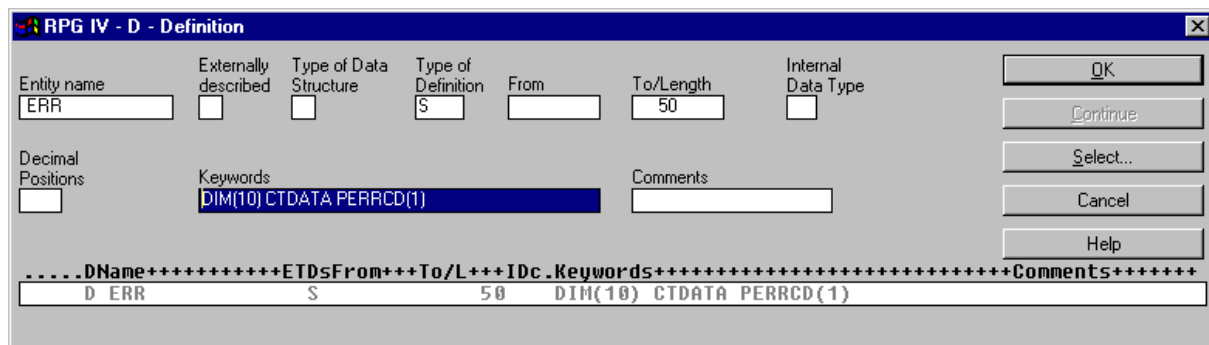

___ 3. **Move** the cursor to the **Keywords** field using the **Tab** key. **Press F1** or click on the **Help** push button to see help for this field.
___ 4. You will see words in the help that appear in a different color than the regular text. These are help links, and they show that there is additional help available on that word or phrase. Click on any link to see specific help for that item.
___ 5. Minimize the Help window.
___ 6. **Click** on the **Cancel** push button to close the prompt window without changing your source.

Note: When you use prompts to edit or add to your source and then click on the **OK** push button, the changes are added to the program in the appropriate column and the syntax is checked automatically.

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.

___ 1. From the **View** menu, select **Indent**. The indented view is by default displayed in a split view in the bottom half of your edit window.

___ 2. To get the indented view displayed as full view, select **Ring manager** from the **Windows** menu. The Ring Manager window comes up.
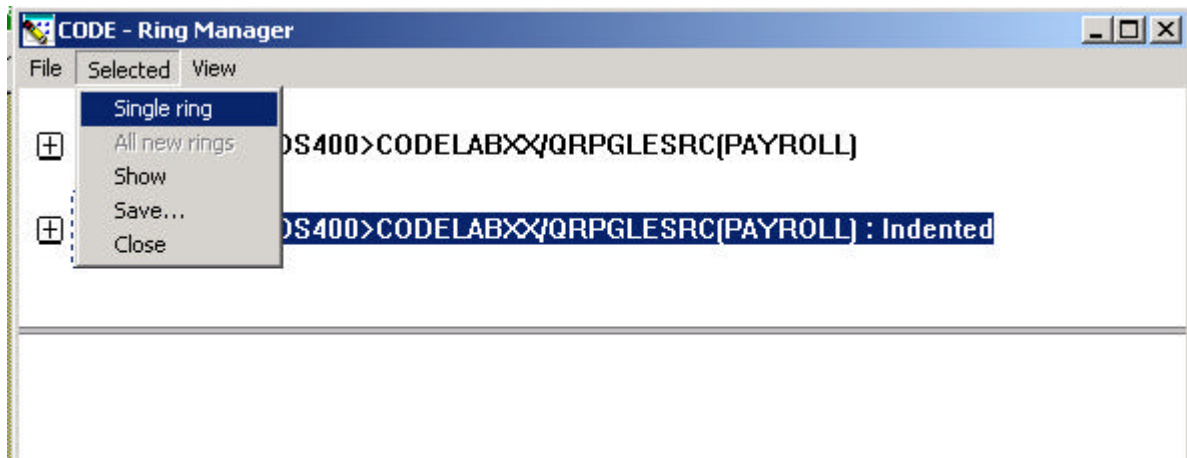
___ 3. In the Ring Manager window select the **Indented** entry, then click on **Single ring** from the **Selected** menu. All views are now on one ring. Close the Ring manager and switch back to the edit window. The indented source is now the current view.

.
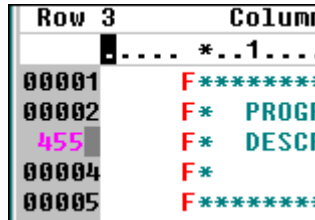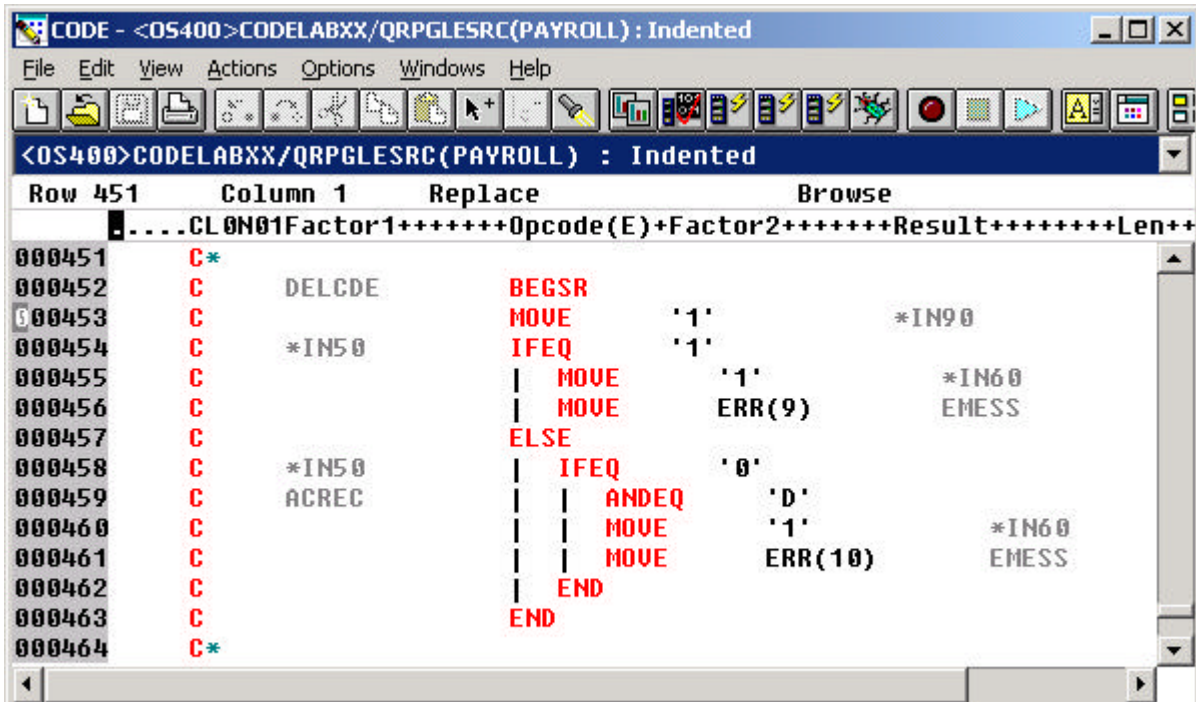
___ 4.
> **Move** the cursor to row **455** by typing **455** in the sequence number area, just as you would using SEU and check what happened to the IFEQ - END blocks. Use the blue drop-down list below the toolbar to switch between the normal view and indented view to see the difference.
>
> **Note:** The INDENTED view is Browse mode only and cannot be edited.

___ 5. **Switch** to the INDENTED view and **close** it by pressing **F3**.

___ 6. **Switch** back to **<OS400>CODELABxx/QRPGLESRC(PAYROLL)**.
> (**Note:** you can also switch between source members using Alt+right arrow or Alt+left arrow)

Find and Replace

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

___ 1. Press **Ctrl+Home** to go to the top of the file.

___ 2. From the **Edit** menu, select the **Find and replace...** (or **press Ctrl+F**, or click on 🔦 in the toolbar). The Find and Replace dialog appears.
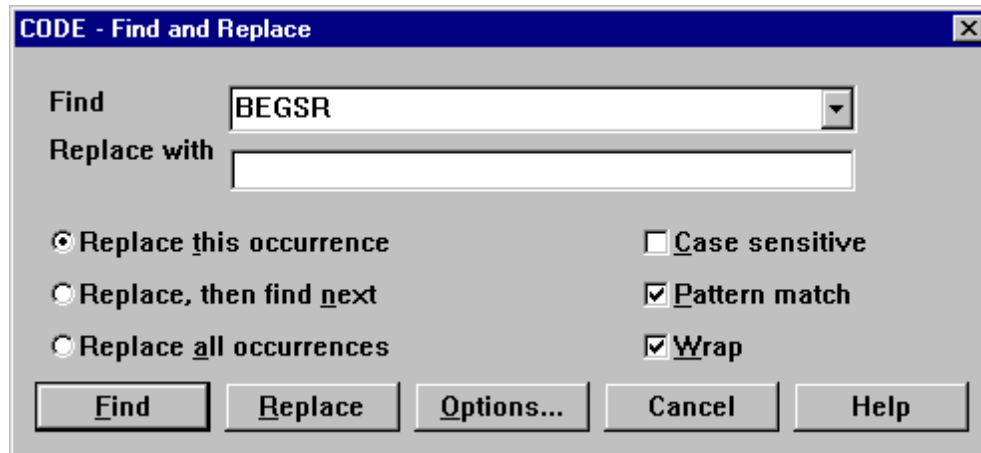

___ 3. **Click** on the **Options** push button to see what we could do if we wanted to -- ie. search all the files loaded in the editor, search only in certain columns, etc.

___ 4. **Click** on the **Cancel** push button.

___ 5. In the **Find** entry field, enter **BEGSR** to find the start of a subroutine. Make sure the **Replace with** entry field is blank. You would use this field for text replacement.

___ 6. **Click** on the **Find** push button. The cursor is positioned at the first **BEGSR** in the file.

___ 7. 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.

___ 1. From the **View** menu, select **Filter** (or **press Ctrl+I**).

___ 2. **Type BEGSR** in the entry field.
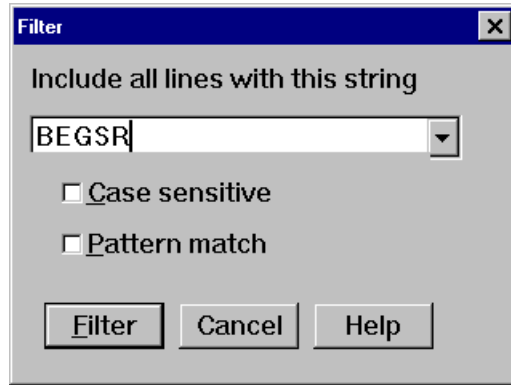
___ 3. Click on the **Filter** push button. Only lines that contain the string BEGSR appear in the editor.
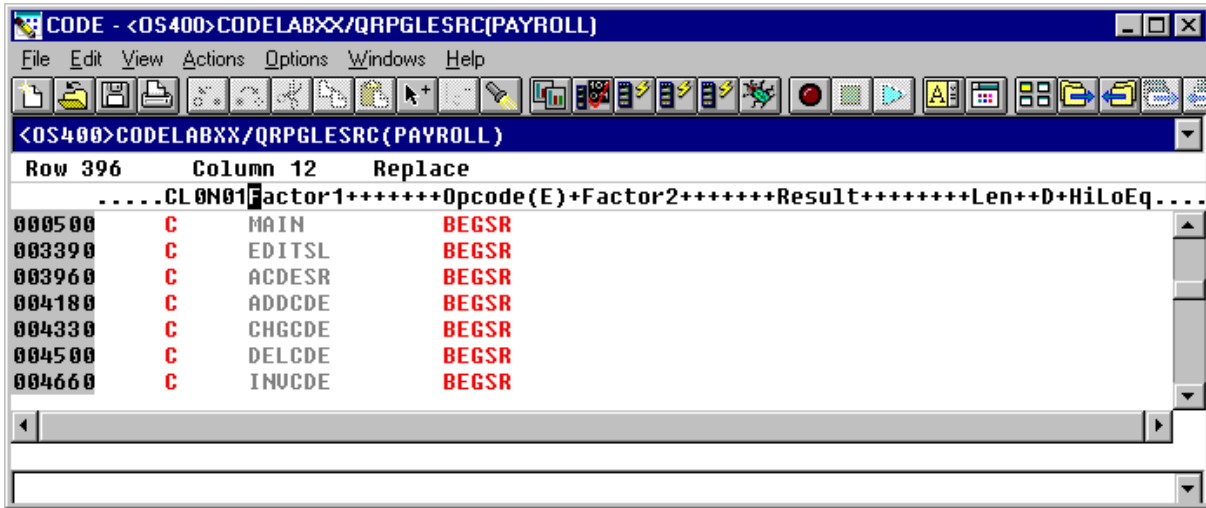
___ 4. **Move** the cursor down a line or two
___ 5. Show all of the source again by selecting **View→Show all** or by pressing **Ctrl+A.**
    Your cursor is still positioned on the same line that you moved the cursor to, even though all lines are now showing.

Using the Show Feature

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

___ 1. From the **View** menu, select **Show** and then select **Subroutines**. All subroutines specifications are displayed allowing you to move quickly and easily to the area in your file where the desired subroutine is.
___ 2. **Move** your cursor to the line with the subroutine declaration **CHGCDE**. (line 433)
___ 3. Show all lines by pressing **Ctrl+A**.
    Your cursor is still positioned on the line for the declaration of the subroutine CHGCDE.

Multi-File Search Utility

If you would like to search through the members in a source physical file or though the files in a local directory, you can use the CODE Multi-File Search tool.

___ 1. From the editor **Windows** menu, select **Multi-file search**. The Search dialog appears.

___ 2. In the **String** entry field, type **ENHRS**
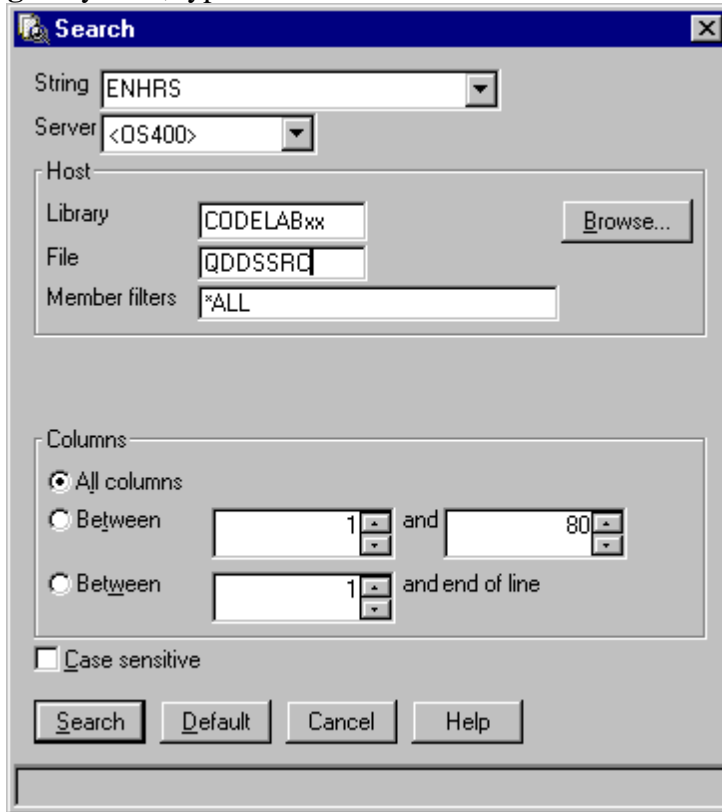
___ 3. **Click** on the **Server** combo-box and select **<OS400>.**

___ 4. In the **Library** entry field, type **CODELABxx** where **xx** is your workstation number.

___ 5. In the **File** entry field, type **QDDSSRC** to search all members in this source physical file.

___ 6. Press the **Search** button. The Multi-File Search window lists all the lines in all the files that reference ENHRS.


```
ENHRS - Multi-File Search                                      _ □ ✕
File  Edit  View  Options  Tools  Help

 □  <OS400>CODELABXX/QDDSSRC(EMPMST)
    ─── A     ENHRS   R
 □  <OS400>CODELABXX/QDDSSRC(MSTDSP)
    ─── A     ENHRS   R    B 11 37REFFLD(RCEMP/ENHRS *LIBL/EMPMST)
 □  <OS400>CODELABXX/QDDSSRC(REFMST)
    ─── A     ENHRS    3 1    TEXT('EMPLOYEE NORMAL WEEK HOURS')

Found 3 occurrences of 'ENHRS'.
```
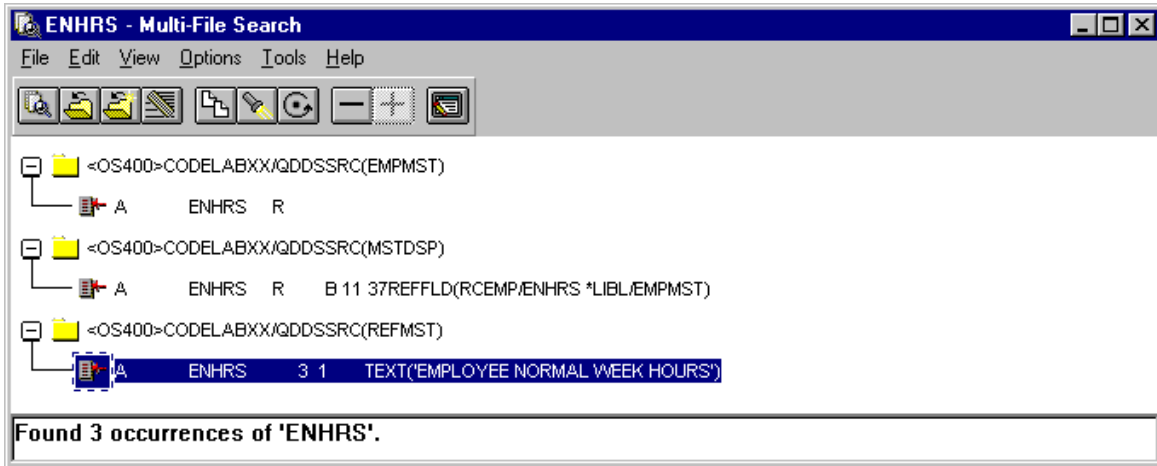
___ 7. **Double-click** on the last line in the list
> **A   ENHRS   3 1   TEXT('EMPLOYEE NORMAL WEEK HOURS')**

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

Comparing Files

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.

___ 1. **Switch** back to **<OS400>CODELABxx/QRPGLESRC(PAYROLL)** using the blue drop-down list beneath the toolbar.

___ 2. From the **Actions** menu, select **Compare...** The **Compare** dialog appears.

___ 3. In the entry field, type
> **<>CODELABxx/QRPGLESRC(PAYROLLG)**

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

___ 4. **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.
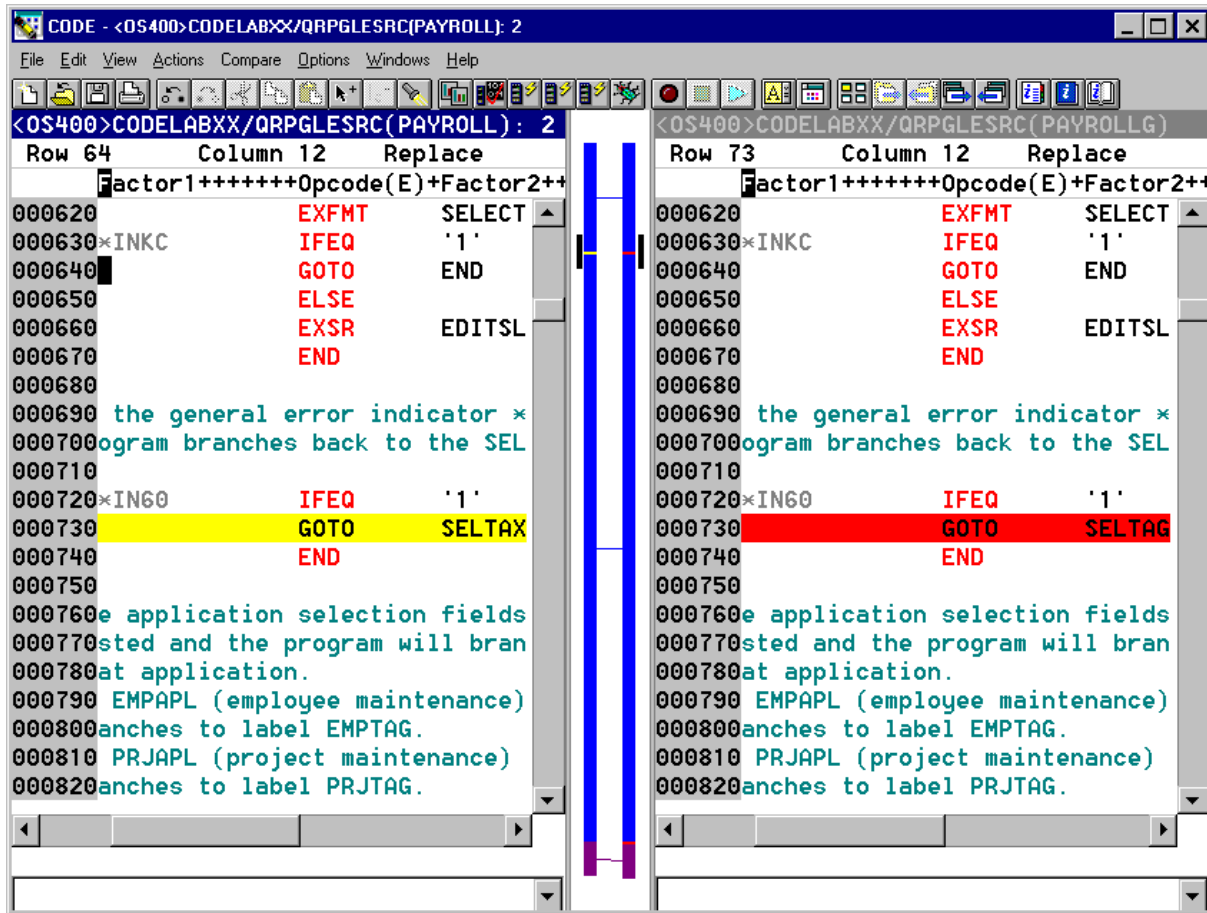
___ 5. Use the vertical scroll bars to move within the files. As you scroll, you will see where the differences are in the members. RPG experts will notice that PAYROLL has some errors in it. We will fix these in a few moments.

___ 6. From the **Compare** menu (which was inserted while performing this action), select **Exit Compare** to go back to the original view.
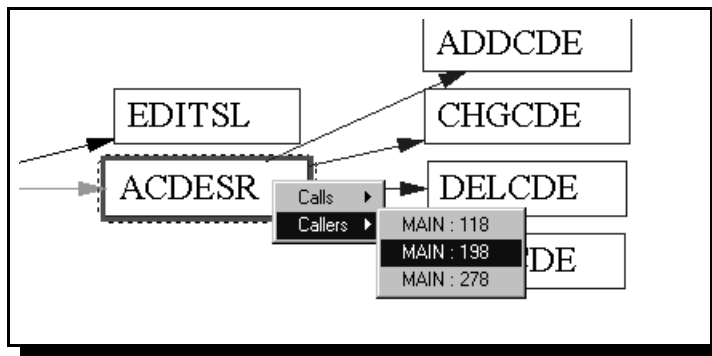
The Navigator

Take full advantage of the Navigator to view how calling and called procedures or functions are related. The arrow direction indicates which procedures or functions are calling and which ones are being called. The Navigator is available for ILE RPG, C, and COBOL languages.

___ 1. **Switch** back to **<OS400>CODELABxx/QRPGLESRC(PAYROLL)** using the blue drop-down list box beneath the toolbar.

___ 2. From the **View** menu, select **Navigator** menu and then select **Open**. A Navigator window opens up showing that the MAIN procedure calls EDITSL and ACDESR. The

subroutine ACDESR calls ADDCDE, CHGCDE, DELCDE, and INVCDE.
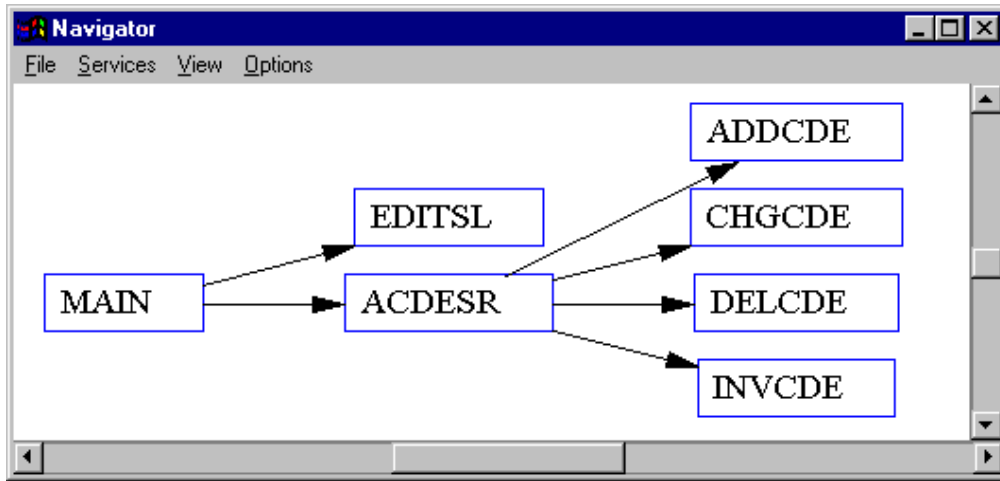
___ 3. If you right click on the ACDESR subroutine you can go right to the line where that subroutines is called or where it calls other subroutines.

___ 4. From the **Services** menu, select **Topology**. This opens a file called NAVIG.OUT that lists the called and calling functions. For large, complex programs these tools are invaluable.

___ 5. Press **F3** to close NAVIG.OUT.

___ 6. Select the Navigator window and from the **File** menu, select **Close**.

Syntax Checking

One of the powerful features that the CODE 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.

___ 1. **Move** the cursor to row 198 which contains 'EXSR ACDESR'. You might already be on that line but if not, you already know different ways of getting there. You could type the line number in the sequence number column, or you could use **Locate Line** (Ctrl+L), **Find and Replace** (Ctrl+F), **Filter** (Ctrl+I) or the **Navigator** function.

___ 2. **Append** an **X** to the **EXSR** op-code to make it **EXSRX**.

___ 3. **Move** the cursor off of the line. An error message appears to draw attention to the error.

```
001960     C     *INKC        IFEQ        '0'
001970     C     EMPNO        CHAIN       EMPMST                          50
001980     C                  EXSRX       ACDESR
001980RNF5014E Operation entry is not valid; specification is ignored.
001990     C                  ELSE
002000     C                  GOTO        END
002010     C                  END
```

___ 4. **Move** the cursor onto the pink error message.
___ 5. **Press F1**. This opens a window with second level help for the error.
___ 6. **Minimize** the help window.
___ 7. Change **EXSRX** to **EXSR** to correct the error.
___ 8. **Move** the cursor off the line you just fixed. The error message is automatically removed from the editor.
___ 9. You can toggle automatic syntax checking by using the **Options→Language editing→Syntax checking** menu item.
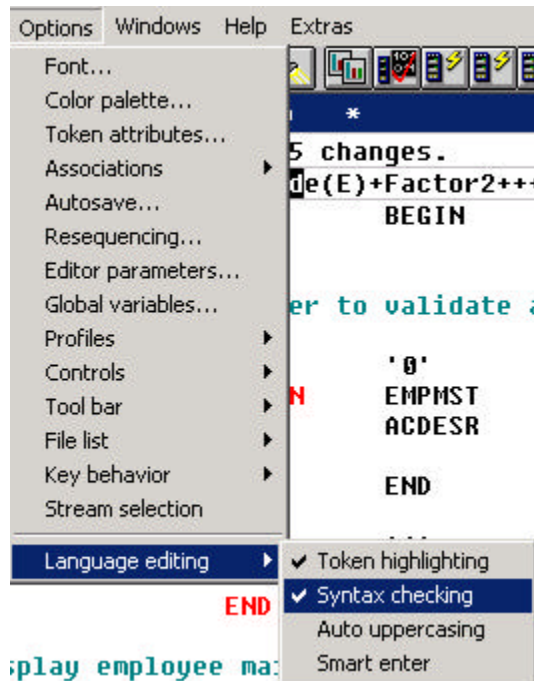
Verifying Your Source

Now we get to play with one of the most powerful and unique features of the CODE Editor -- 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 are disconnected from an iSeries. You can do this because CODE 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:

___ 1. From the **Actions** menu, select **Verify program** and then select **No prompt** (or click on

). A verify is performed and the **Error List** appears. By default, you will see a number of informational messages that are not shown in the picture below. To filter out all the informational messages toggle the **Options**→**Include**→**Information** menu.
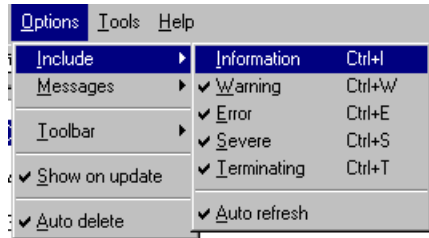

To determine the severity and line number of an error:

___ 1. **Click** on any error in the list. The status line at the bottom displays the severity of the error and the line number in the source.
___ 2. **Press F1**. Second-level help for that error message appears. Read the help.
___ 3. Minimize the Help window.

To insert an error message in the source and place your cursor there.

___ 1. **Double-click** on the error **RNF7043E**. The error is inserted into the editor and you are taken to the offending line.
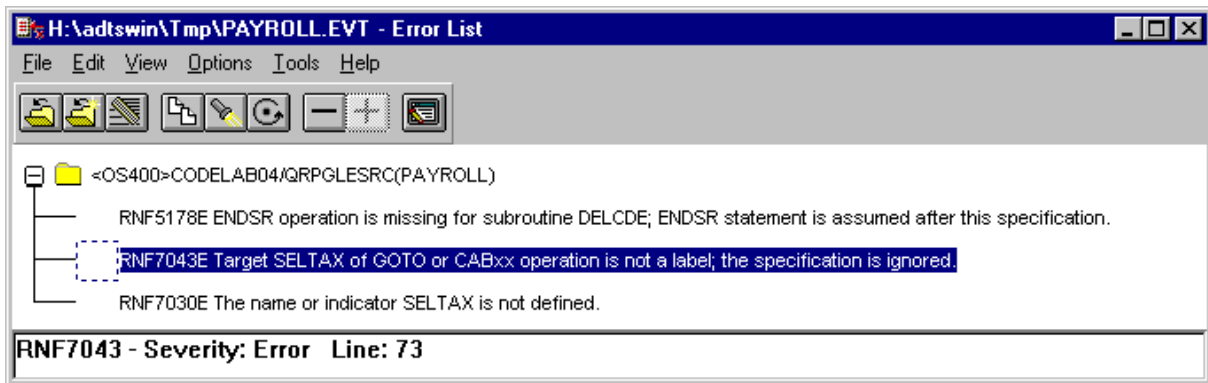
Fixing Errors

Now you will use both the editor and the error list to fix the errors found when verifying the program.

___ 1. The error on line **73** is a typo. **SELTAX** should really be **SELTAG**. Make the appropriate change.
___ 2. **Switch** back to the Error List window. Several of the errors now have a checkmark beside them. This indicates that you have modified these lines. The only serious

remaining error is **RFN5178E**. This error is caused by a missing **ENDSR**.
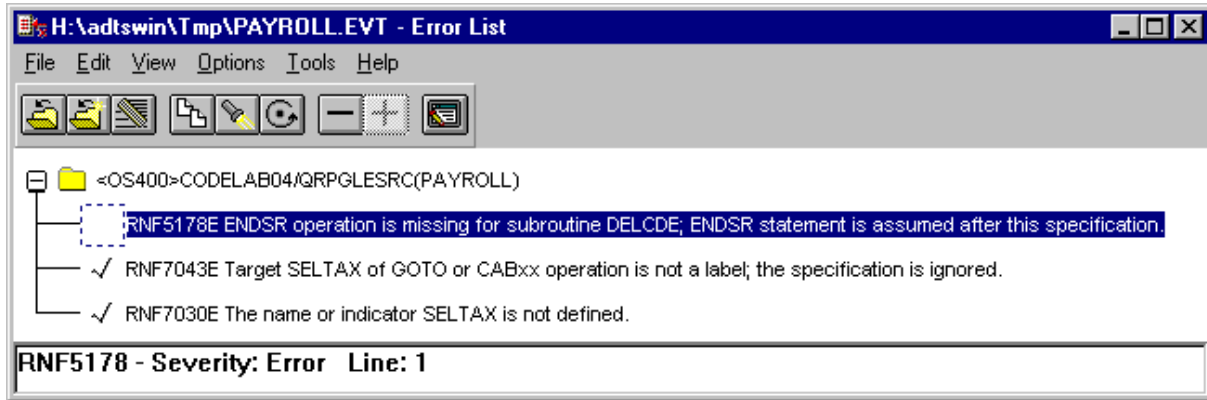
___ 3. **Move** the cursor to row **461** in the editor. From the **View** menu, select **Indent** if you are having troubles seeing the error (or look at the picture in the **Indenting Source** section again).

___ 4. **Press Enter** to insert a new line.

___ 5. On row **463**, type

      C   ENDSR

(**Hint:** use the Tab key to quickly move to the appropriate column)
All the non-informational errors are now fixed.

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 the **File** menu, the toolbar (click on [disk icon]), or:

___ 1. Press **Ctrl+S**. Changes are uploaded to the iSeries.

CODE Program Generator

The Program Generator is another tool within CODE. 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. It remembers your compile options so that they only need to be entered once. The Program Generator also supports several features for Java, such as compiling on the iSeries host.
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.

Starting the Program Generator

Whenever you request that source be compiled from within the editor, the Program Generator is started.

___ 1. From the editor's **Actions** menu, select **Compile** and then select **Prompt...** The Program Generator window appears.
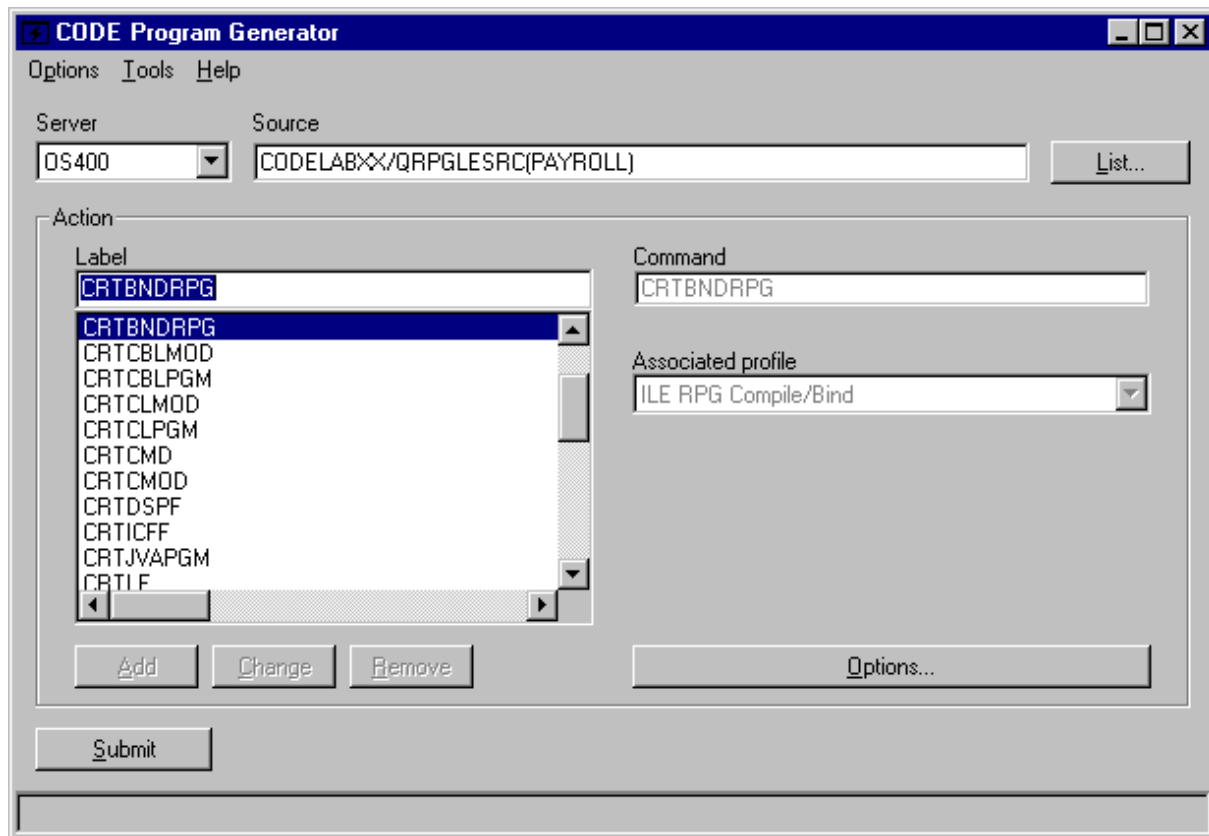
Compiling Your Source

We will now use the Program Generator to select compile options. To start, we have to specify some settings and the program we want to compile.

___ 1. From the **Options** menu, select **Settings...** The **Settings** dialog appears.

___ 2. On the **General** page **click** on the **Interactive** action mode and make sure the **Notify on completion** is checked**.**

___ 3. **Click** on the **OK** push button to save your settings and dismiss the dialog.

___ 4. Both the **Server** and the **Source** entry fields should already be filled in with **OS400** and **CODELABxx/QRPGLESRC(PAYROLL)**, respectively.

___ 5. The **Label** identifies the command and options for the command that will be used to compile the program. **Select** the CRTBNDRPG label.

___ 6. **Click** on the **Options...** push button near the bottom right-hand side of the notebook to select the compile options for the program.

___ 7. **Click** on the **Compile** tab.

___ 8. Change the **Debugging views** from ***CMDDFT** to ***ALL**.
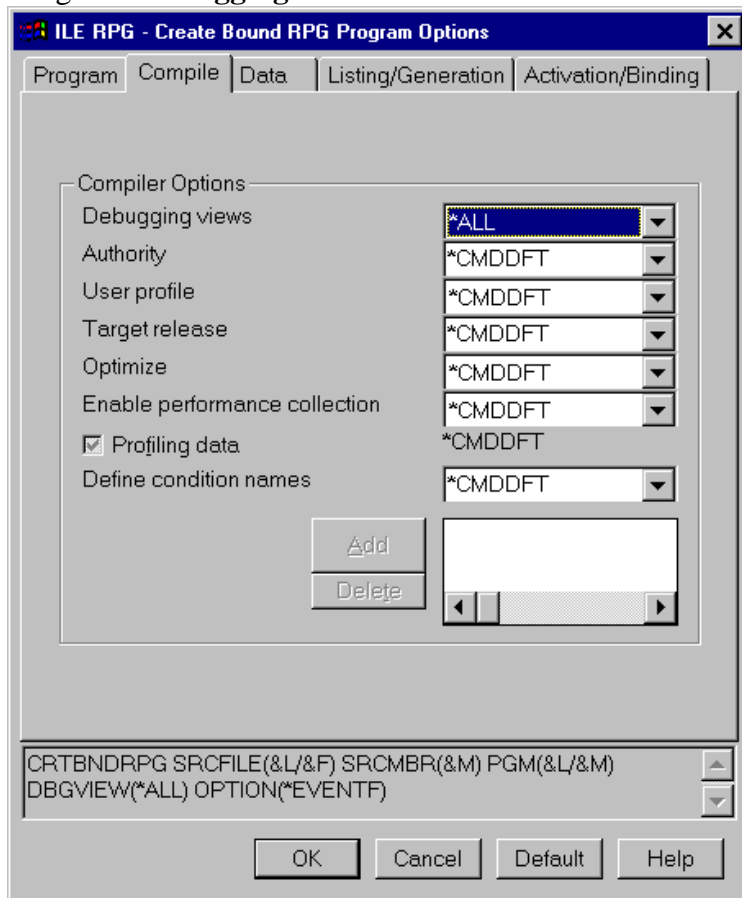
___ 9. Select any other tab to explore the ILE RPG compile options. Click on the **OK** push button when you are done.

___ 10. **Click** on the **Submit** push button to submit your compile request to the iSeries.

___ 11. A message will tell you when the compile is complete. **Click** on the **OK** push button in the message dialog. **Click** on the **X** in the upper right-hand corner of the Program Generator notebook to close it.
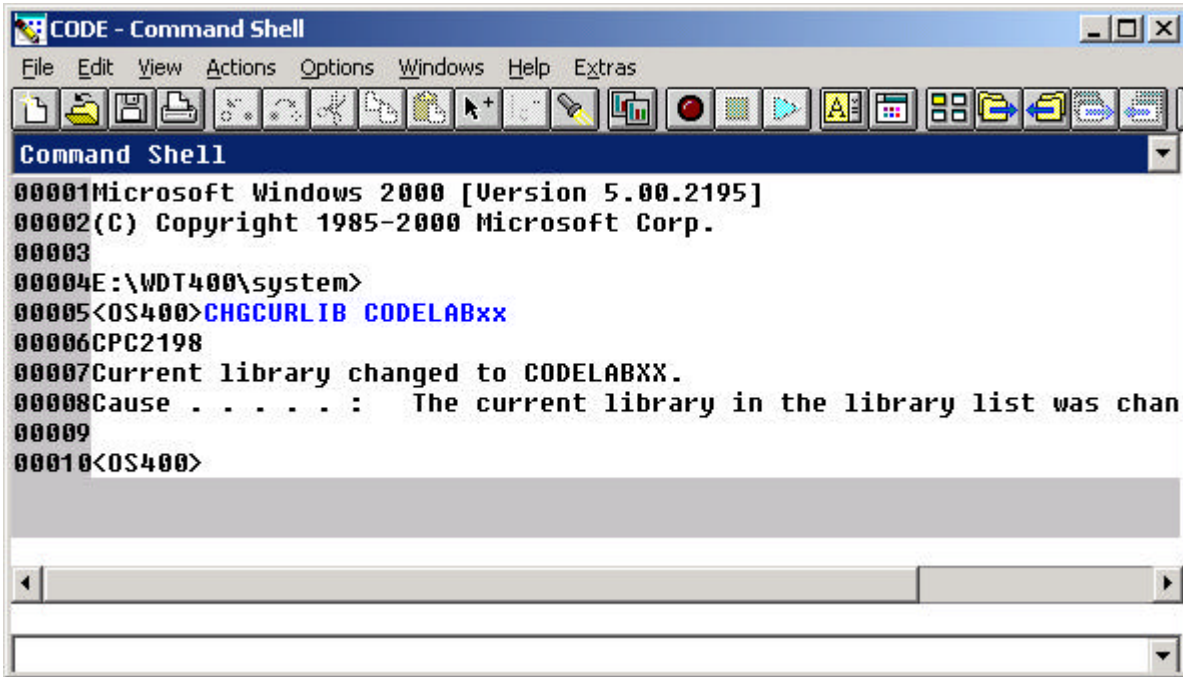
The Command Shell

You can use the **Command Shell** inside the CODE Editor to submit commands to the iSeries. You could change your library list, for example.

___ 1.

From the **Windows** menu, select **Command Shell** (or simply press **F9**). This will bring up the Command Shell where you can enter both workstation and host commands.

___ 2. The library you will use is **CODELABxx** where **xx** is your workstation number. At the **<OS400>** prompt, type **CHGCURLIB CODELABxx** and press Enter.

```
CODE - Command Shell                                          _ □ ×
File  Edit  View  Actions  Options  Windows  Help  Extras
Command Shell                                                      ▼
00001Microsoft Windows 2000 [Version 5.00.2195]
00002(C) Copyright 1985-2000 Microsoft Corp.
00003
00004E:\WDT400\system>
00005<OS400>CHGCURLIB CODELABxx
00006CPC2198
00007Current library changed to CODELABXX.
00008Cause . . . . . :    The current library in the library list was chan
00009
00010<OS400>
```

___ 3. From the **Options** menu, select **Servers** and then select **LOCAL**. This will switch the prompt in the Command Shell to a workstation prompt.
___ 4. Type **c:** and press Enter.
___ 5. Type **DIR** and press Enter. Wow! Now you can even scroll up and down to view all files and directories in the c drive.
___ 6. From the **Options** menu, select **Servers** and then select **OS400** to switch back to host prompt.

Running the PAYROLL Program

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

___ 1. In the editor, **switch** to the Command Shell by pressing **F9**.

```
PRG01                   Time Reporting System              98/04/09
                        Maintenance Selection              17:26:38


          Enter an X beside the application you want to maintain


                    _   Employee Master Maintenance
                    _   Project Master Maintenance
                    _   Reason Code Master Maintenance
```

___ 2. At the **<OS400>** prompt, type in the following command:

      CALL PAYROLL

then press **Enter**.

___ 3. **Switch** to your 5250-emulation session.

___ 4. Place an 'x' beside **Employee Master Maintenance**. Press **Enter**.

___ 5. Type **123** for the Employee Number.

___ 6. Type **A** for the Action Code to add employee **123**. Press **Enter**.

___ 7. Type any information you like about the employee. Press **Enter**. Play in the application as much as you like.

___ 8. Press **F3** to end the PAYROLL job.

The 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.

Starting the CODE Designer

You can jump straight to this section without completing the CODE Editor section but you must have established a connection with the iSeries as discussed in the Connecting to the iSeries < PageRef Connecting to the AS/400>

Let's launch the Designer and load a display file.

___ 1. From the **Start → Programs→ IBM WebSphere Development Tools for iSeries → IBM CODE400** menu, select **CODE Designer**. The Designer appears with no DDS loaded.

___ 2. From the **File** menu, select **Open...** The Open dialog appears.
___ 3. **Click** the + beside **<OS400>** to show the iSeries library list.
___ 4. **Click** the + beside **CODELABxx** to show the library.
___ 5. **Click** on the source physical file **QDDSSRC** to show its contents on the right.
___ 6. **Click** on **MSTDSP DSPF** and then **click** on the **OK** push button. The source is loaded into the Designer.

The DDS Tree

What you are looking at now is basically a "Windows Explorer"-like 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 **<OS400>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 <u>Workbook</u> with several different tabbed pages. The top page is called <u>Details</u> and it is the other half of the "Windows Explorer"-like interface. It lists the contents of the currently selected item in the DDS Tree.

In the bottom right-hand side of the Designer is the <u>Utility Notebook</u>. 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.
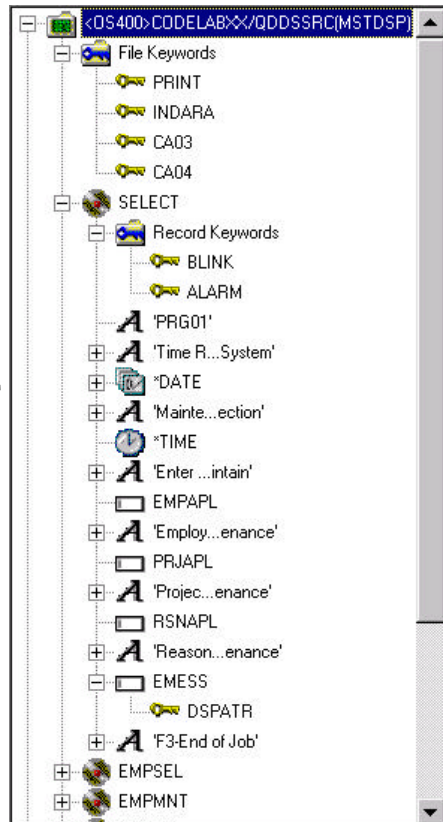
Field	Position	Length	Type	Shift	Usage	Sample
A 'PRG01'	2, 5	5	Text constant			PRG01
A 'Time R...System'	2, 30	21	Text constant			Time Reporting System
*DATE	2, 70	6	Date constant			YY/MM/DD
A 'Mainte...ection'	3, 30	21	Text constant			Maintenance Selection
*TIME	3, 70	6	Time consta...			HH:MM:SS
A 'Enter ...intain'	6, 14	54	Text constant			Enter an X beside the application you
EMPAPL	9, 25	1	Alphanumeric	A - Alphanumeric	Both	B
A 'Employ...enance'	9, 28	27	Text constant			Employee Master Maintenance
PRJAPL	10, 25	1	Alphanumeric	A - Alphanumeric	Both	B
A 'Projec...enance'	10, 28	26	Text constant			Project Master Maintenance
RSNAPL	11, 25	1	Alphanumeric	A - Alphanumeric	Both	B
A 'Reason...enance'	11, 28	30	Text constant			Reason Code Master Maintenance
EMESS	21, 16	50	Alphanumeric	A - Alphanumeric	Output	000000000000000000000000
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.

```
Selected DDS | Create keywords | Comments | Error list

    A              EMESS           50A  O 21 16
    A   60                                 DSPATR(HI)
```

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.
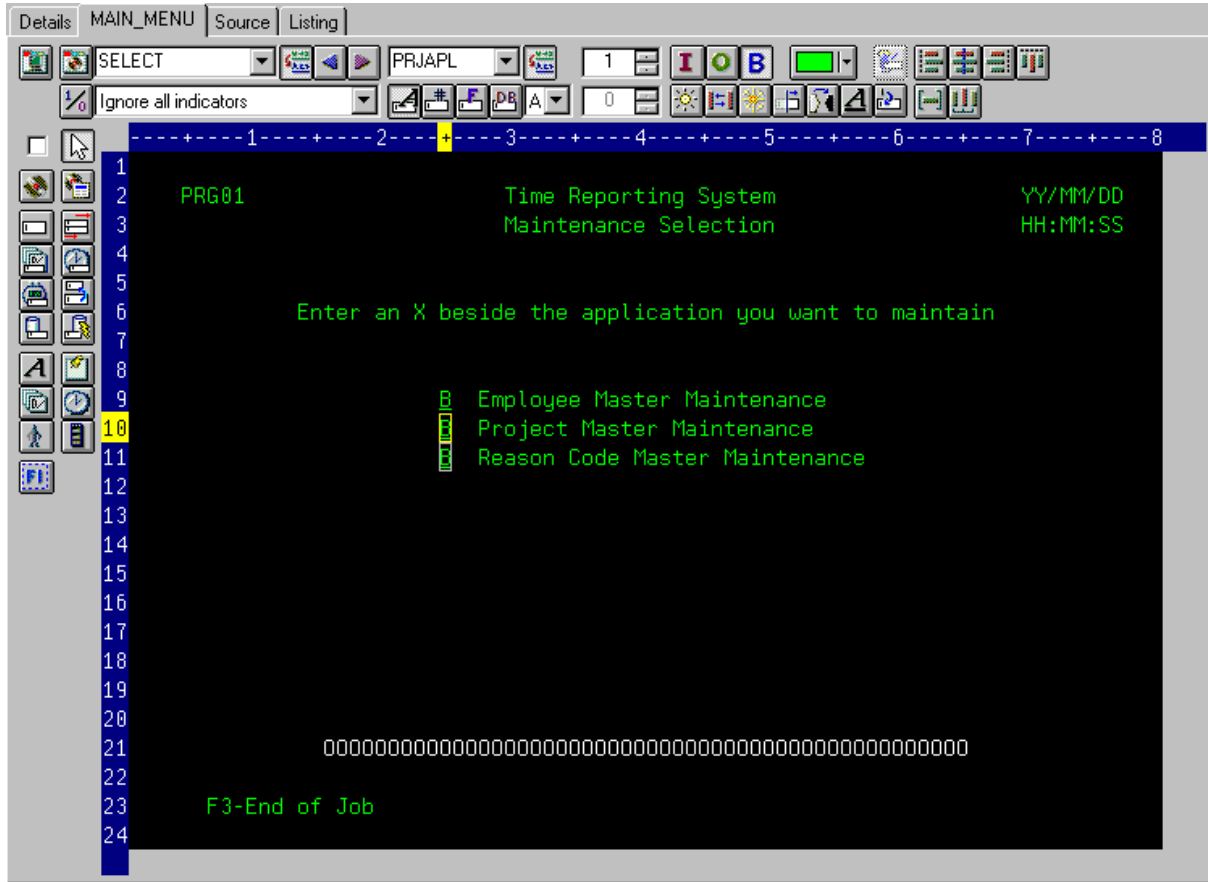
In order to understand where MAIN_MENU came from, we need to describe the concept of a **group**. A group is simply a collection of one or more DDS records that make up a single screen or report. It is the set of records that is written by the application to the display or printer device at one time. 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_SELCT**
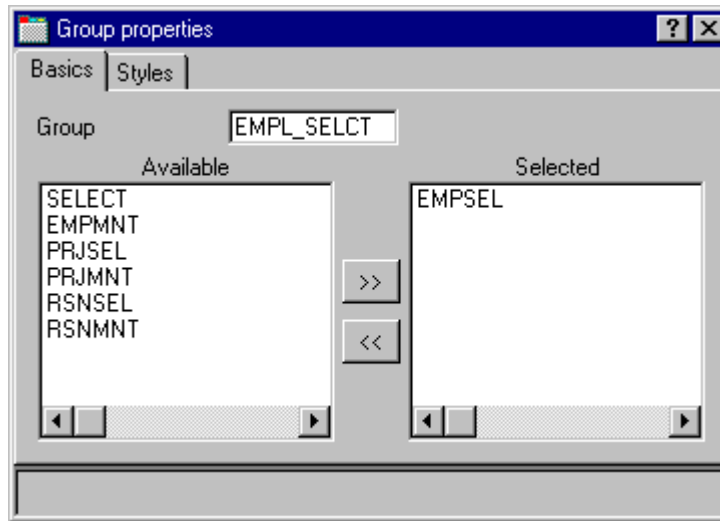
___ 6. **Close** the Group properties notebook by clicking on the X in the top right corner

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_SELCT 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. **Click** on the **Create subfile record** button and then **click** in 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.
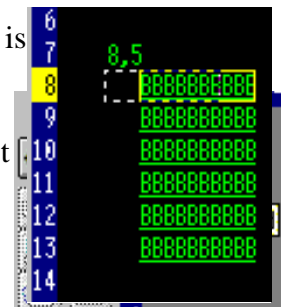
___ 3. **Move** the mouse over to the **right edge** of the field. It turns into a double-headed arrow.
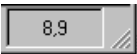

Hold down the mouse button and **move** it to the **left**. The size of the field will be reduced. The current size will appear just above the field. When the size is **3**, **let go** of the mouse. Very nice!

___ 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 8,9 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**
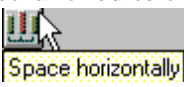
   $666,666.66

   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

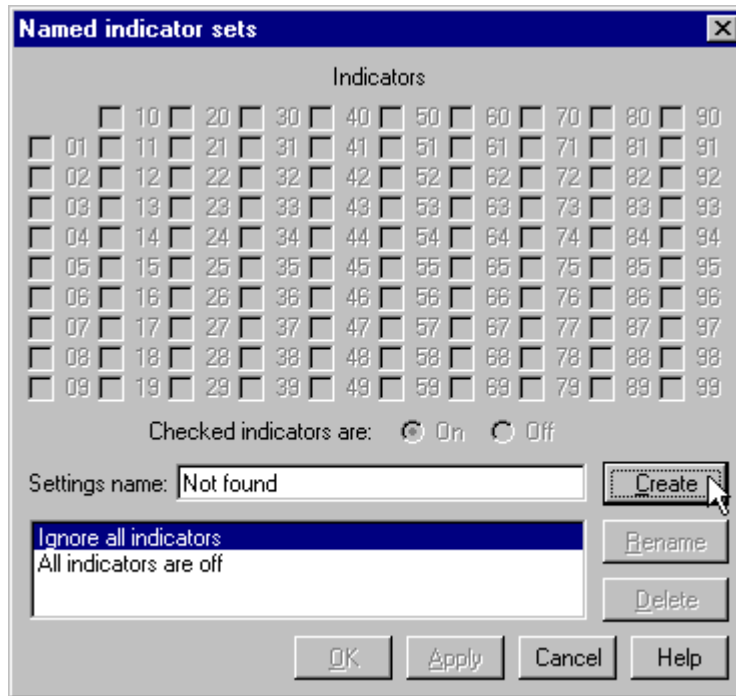Let's provide a 'position to' entry in the subfile control header.

___ 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 ½ 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.

Named indicator sets

Indicators

```
        10   20   30   40   50   60   70   80   90
  01   11   21   31   41   51   61   71   81   91
  02   12   22   32   42   52   62   72   82   92
  03   13   23   33   43   53   63   73   83   93
  04   14   24   34   44   54   64   74   84   94
  05   15   25   35   45   55   65   75   85   95
  06   16   26   36   46   56   66   76   86   96
  07   17   27   37   47   57   67   77   87   97
  08   18   28   38   48   58   68   78   88   98
  09   19   29   39   49   59   69   79   89   99
```

Checked indicators are: ⦿ On   ○ Off

Settings name: | Not found | [Create]

Ignore all indicators
All indicators are off

[Rename]

[Delete]

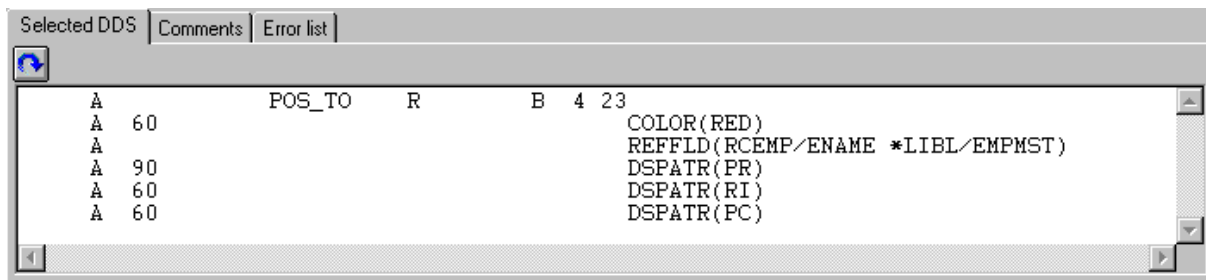[OK]   [Apply]   [Cancel]   [Help]

___ 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.

EMPLSTCTL

Not found
Select named indicator sets

___ 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 latchable toolbar buttons representing the current display attributes is found just below the color button). The toolbar should look as follows:

EMPLSTCTL   POS_TO   30   I O B
Not found   0

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

Selected DDS | Comments | Error list

```
     A                POS_TO    R          B   4 23
     A   60                                    COLOR(RED)
     A                                         REFFLD(RCEMP/ENAME *LIBL/EMPMST)
     A   90                                    DSPATR(PR)
     A   60                                    DSPATR(RI)
     A   60                                    DSPATR(PC)
```

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**.
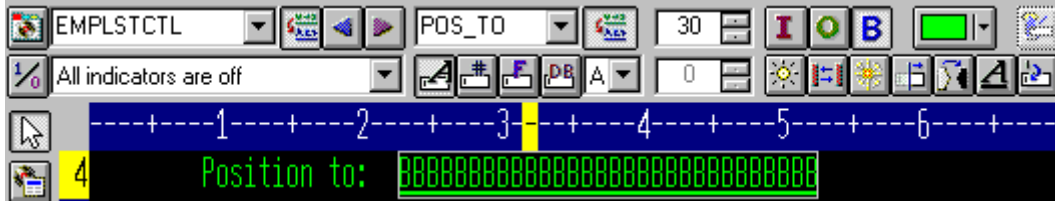

Hey! This stuff really works!

___ 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
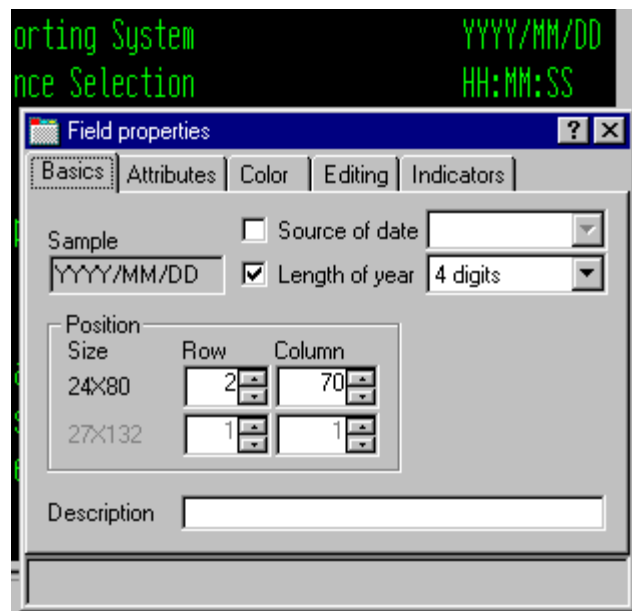
properties for this field.
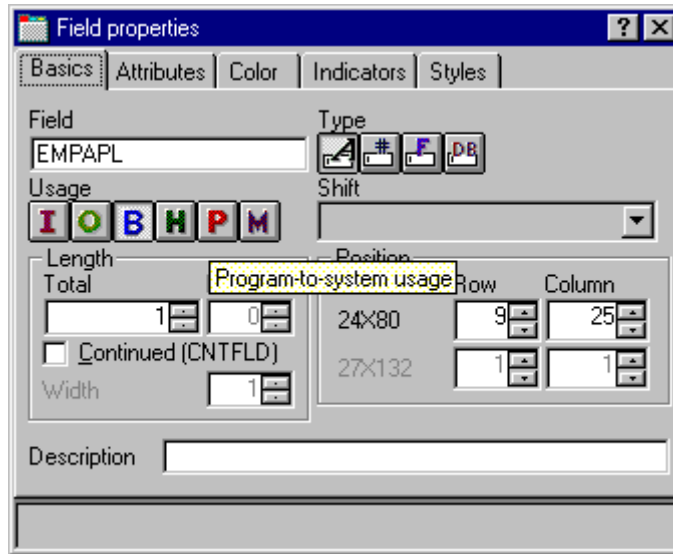
Adding New Keywords

"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.
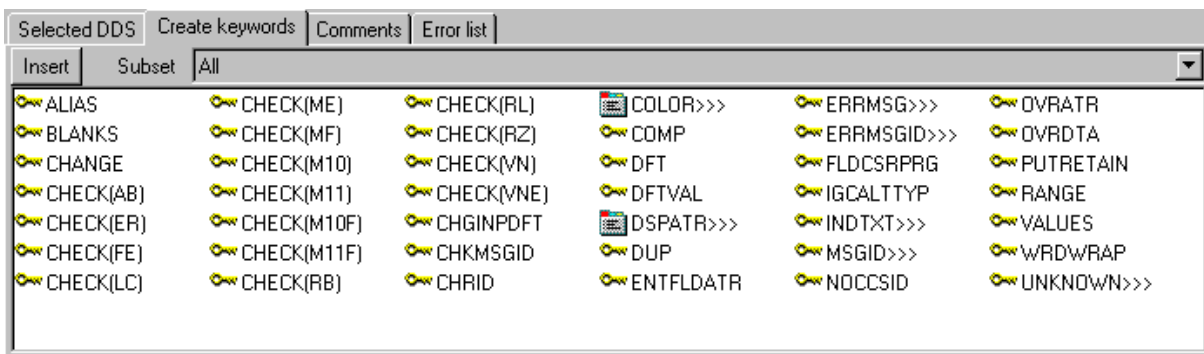

___ 3. With the **EMPAPL** properties notebook at the Basics page, **click** on the ⊞ button to change the field to numeric type. Notice that the list of keywords in the **Create keywords** page has changed.

___ 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 [icon] 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
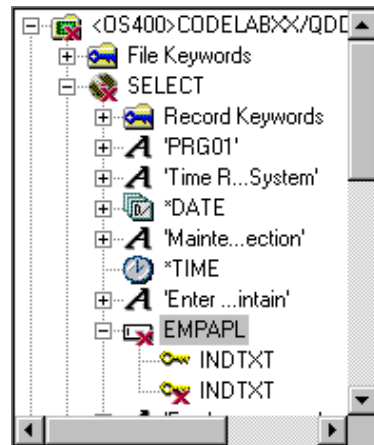
modes at the push of a button.
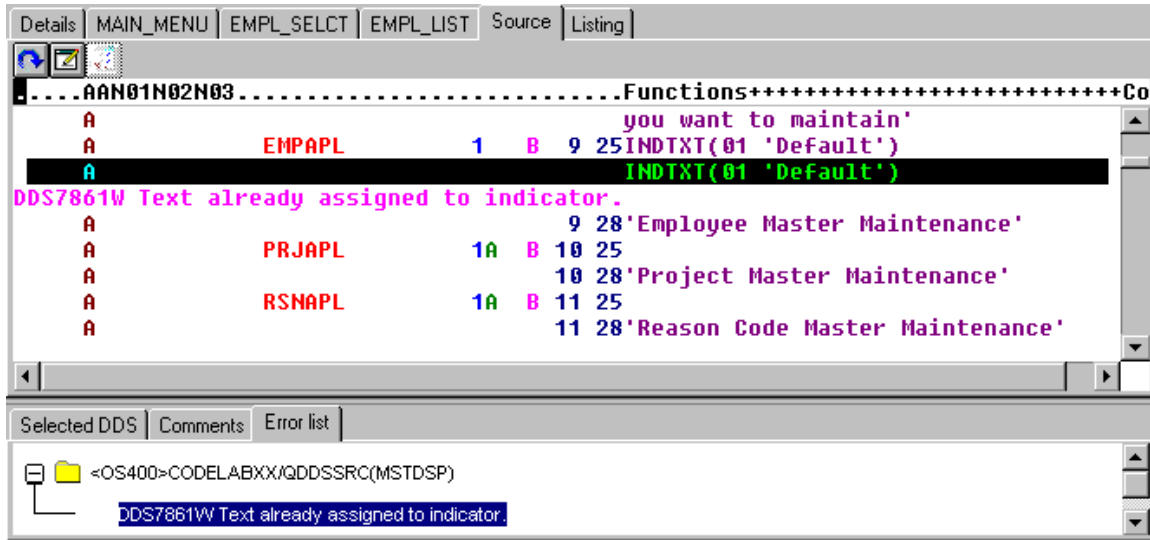
Switching between Design mode and Edit mode

___ 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 using **Alt-Backspace**.
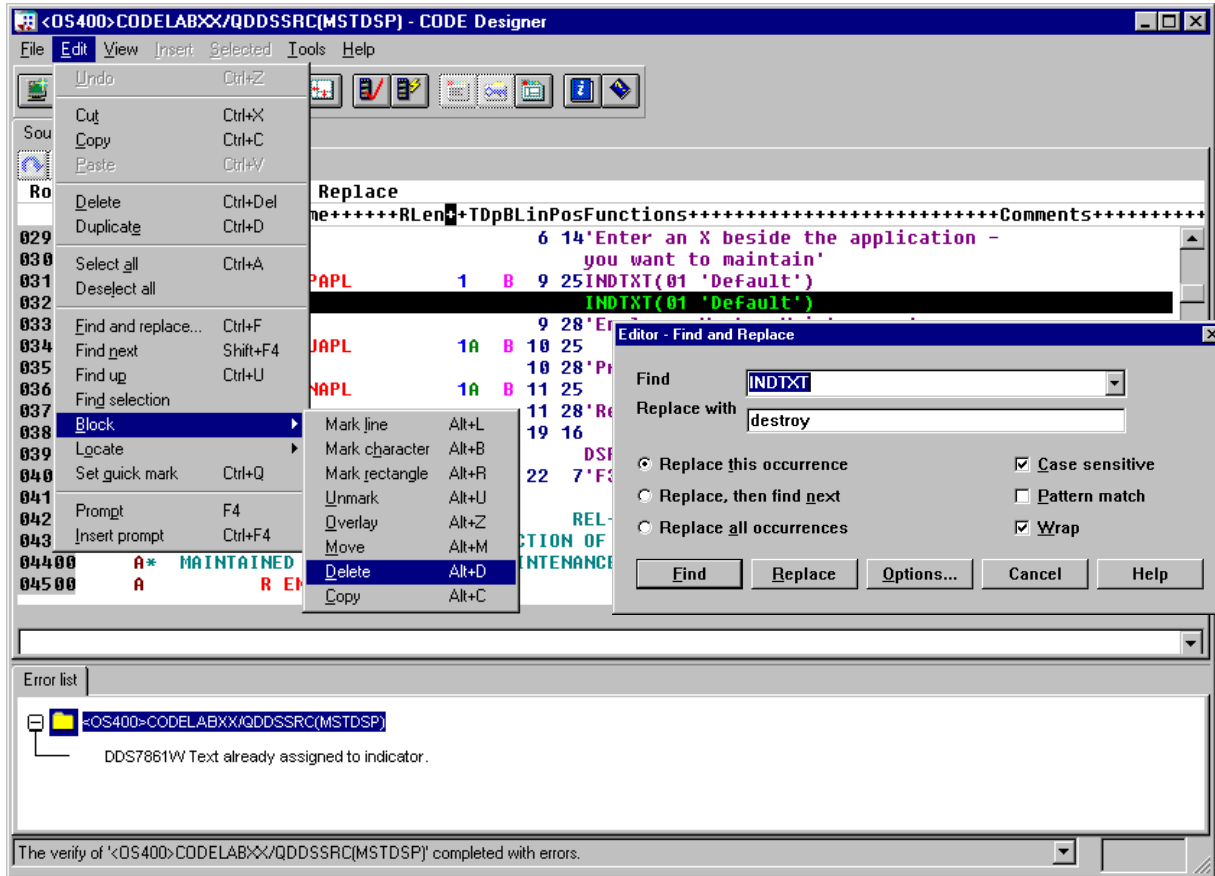
Compiling Your Source

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.

The Distributed Debugger

The CODE 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. We are using it instead of PAYROLL in the 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: from a DOS prompt, from the Windows Start menu; from the CODE Project Organizer; or from the CODE Editor:
___ 1. If the CODE Editor is not up already open it by selecting **Start → Programs→ WebSphere Development Tools for iSeries → IBM CODE400 → CODE Editor**.
___ 2. Load the **PAYROLLG** program into the editor. You can use the Open dialog to do this or hit Esc to go to the editor's command line and type**:**
     LX <>CODELABxx/QRPGLESRC(PAYROLLG)
where **xx** is your workstation number.

___ 3. From the **Actions** menu, select **Debug** and then select **Interactive application** (or click on [icon] in the toolbar or even faster **press Ctrl+Shift+D**). Since this is the first time you invoke the debugger and you did not specify a default server to be used for debugging, the Servers dialog comes up.
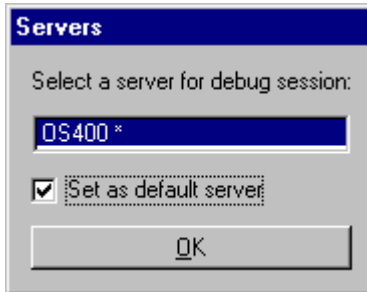
___ 4. Select Set as default server and then press **OK**. This server will now be used for your subsequent debug sesssions. Since we only started one server and are using it for editing and debugging, the following message is displayed:
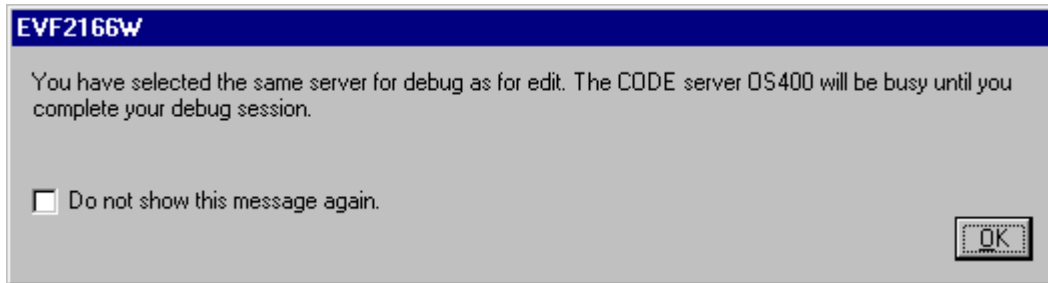
___ 5. Select Do not show this message again and click **OK**. If you want to re-enable the message, select Enable hidden messages from the CODE Daemon context menu.
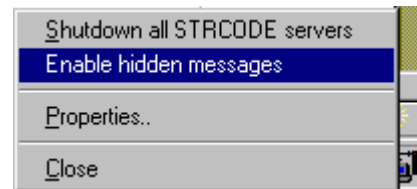

The **Call Program** dialog appears next. Program and Library are pre filled with the respective entries from the Editor. If your program requires parameters you have to enter them here. Our PAYROLLG program does not require parameters so we can click OK

---

without changing anything.

___ 6. The debugger window comes up. This window is open as long as the debugger is active. If this is the first time the debugger is invoked, you will be presented with a Logon dialog. Enter your userid and password (**CODELABxx**) and click **OK.** The host name for the iSeries you are connecting to is already pre filled for you.
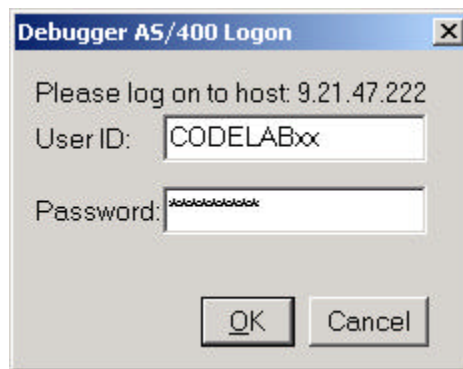
Note: There is a server that runs on the iSeries which services requests from the CODE Debugger. This server is started with the command **STRDBGSVR** and will run until the next IPL or until it is explicitly terminated. If no one has run this command since the last IPL you may get the following message:
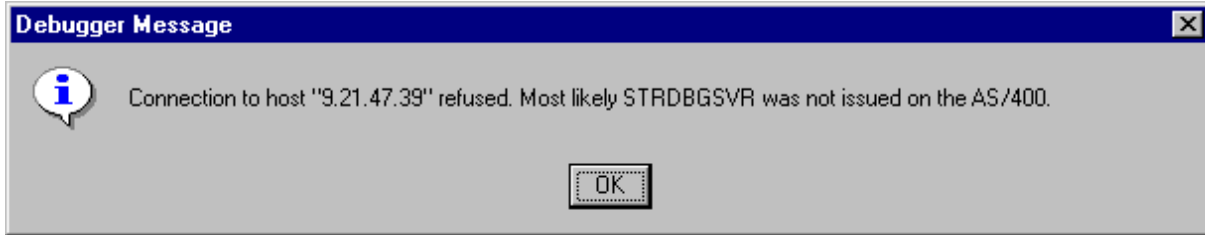
___ 1. Now that the program is active on the iSeries and stopped at the first executable statement the debugger displays the source. Depending on the option you used to compile the program (*SRCDBG for CL, or *SOURCE, *LIST, or *ALL for ILE CL), this window displays either the Source or Listing view.
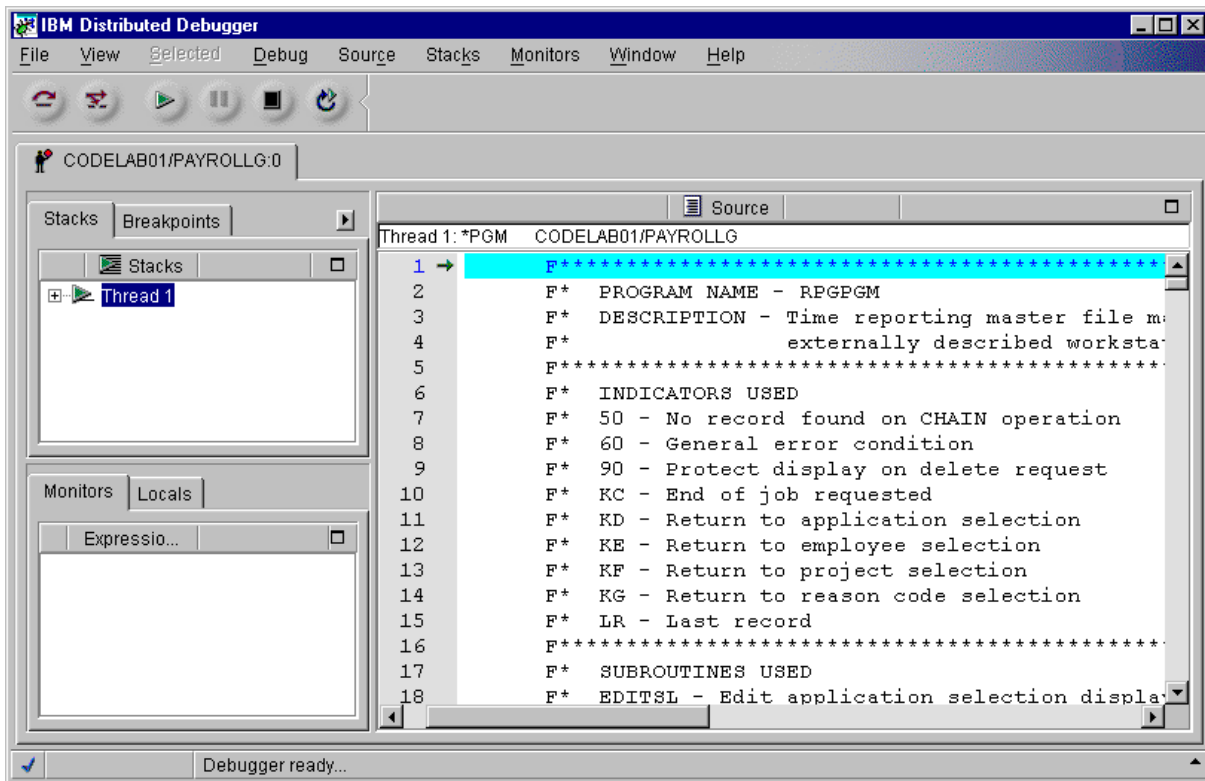
___ 2. From the **Source** menu, select **Listing View**. Page down in the source and take a look at the expanded file descriptions. We don't have any /Copy members in our PAYROLL program but these would also be shown in a listing view.
___ 3. Switch back to the source view. From the **Source** menu, select **Source View.**

The Call Stack

The **Stacks** page 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. Clicking on a stack entry will display the corresponding source if it is available. Otherwise the message **Context not available** appears in the source pane.

___ 1. In the **Stacks** page, expand the stack entry of Thread1 by clicking on the + sign in front of it or by double clicking on Thread1.
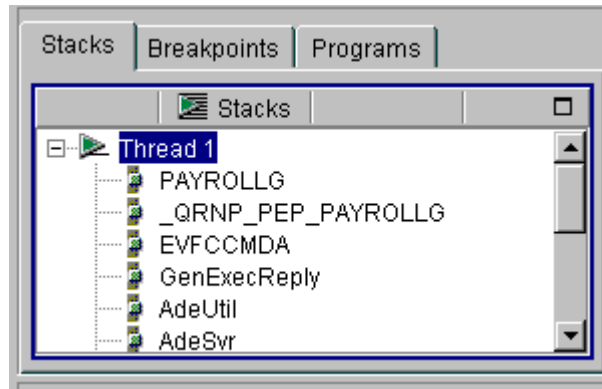
___ 2. To view detailed information about each stack entry, select **Stacks** from the menu bar and click on **Show all Stack Information**.
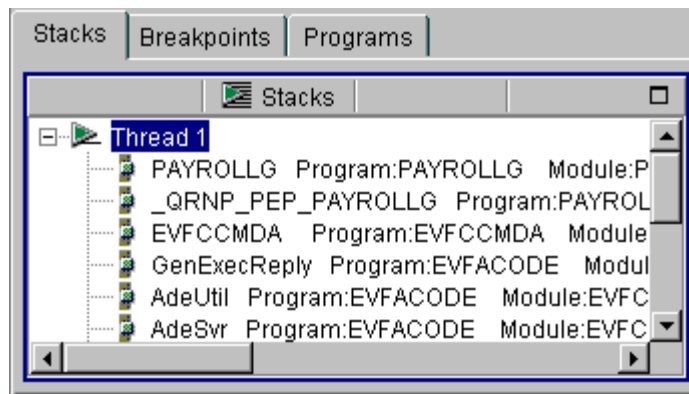
It allows you to work with and switch between different programs and/or ILE modules.

The Programs page

The Programs page lists all programs that are currently known to the debugger. Programs can be expanded to show modules and procedures. Clicking on a module or procedure entry displays its source. Adding or removing programs is available from the Programs menu.

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 double-click in the prefix area of the Source pane:

___ 1. **Move** the mouse over the prefix area (left margin) of line **55** and **double-click**. A red
dot is added to the prefix area of this line to indicate that a breakpoint is set.

___ 2. **Repeat** the above step for line 56.

___ 3. **Right click** in the prefix area of line 87 and select **Set Breakpoint** from the pop-up
menu.

___ 4. Select the **Breakpoints** tab from the top left pane. Expand the **Line Breakpoints** entry
by clicking on the + sign in front of it or by double clicking on **Line Breakpoints**. This
gives you a tree view of all line breakpoints currently set in your program.
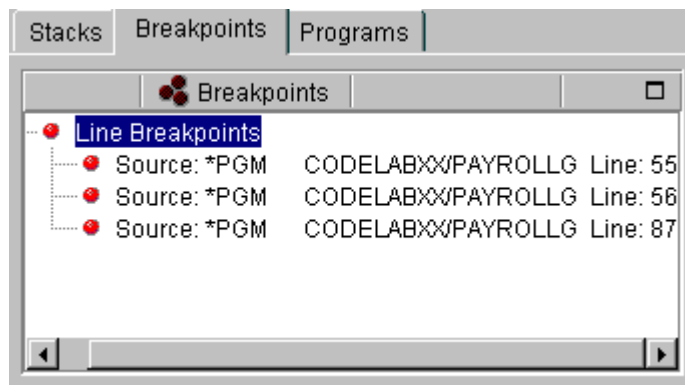
This is a convenient place to work with breakpoints. You can delete, disable/enable, or
modify a breakpoint. These tasks are available from the Selected menu or from the
breakpoints pop-up menu. Clicking on any entry selects it and also brings up the source
where the breakpoint is set. We will use a different method to work with breakpoints.

Deleting a Breakpoint

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

___ 1. **Move** the mouse over the prefix area of line **56** and double-click on it. The red dot is
removed from the prefix area indicating that no breakpoint is set on that line.

Note: You could also right-click in the prefix area and select **Delete breakpoint** from
the pop-up menu.
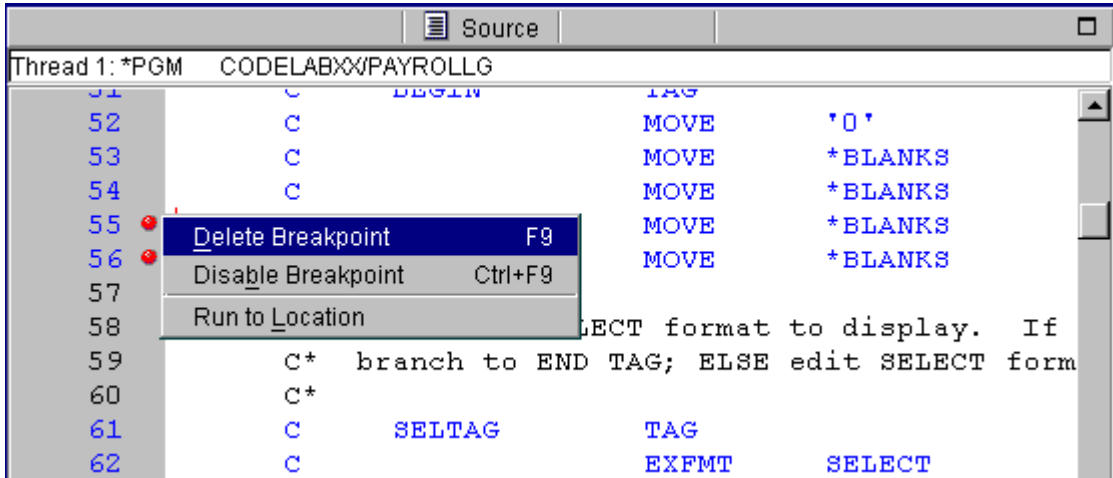
Running a Program

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

___ 1. **Click** the **Run** button [image] from the toolbar. The program starts running and hits the

breakpoint at line **55**.

___ 2. **Click** on the **Run** button again. The program waits for input from the 5250-emulation session.

___ 3. **Type** an **X** beside the **Project Master Maintenance** option, then press **Enter**. The program hits the breakpoint at line **87**.

Monitoring Variables

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

___ 1. In the Source pane, **double-click** on the variable **EMPAPL** on line **87,** right click and select **Add to Program monitor**.
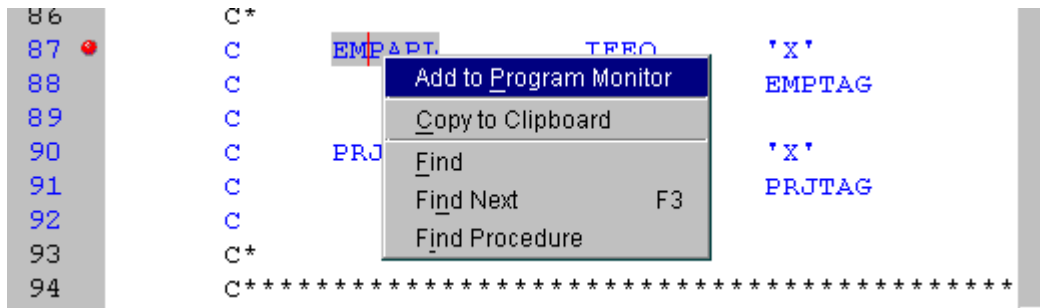
___ 2. The variable appears in the Monitors page. Its value is blank because you did not select the Employee Master Maintenance option.

___ 3. 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.

___ 4. In the Monitors page, **double-click** on the variable **RSCDE**. The value changes into an entry field.

___ 5. **Type** in the new value **X** for the variable.

___ 6. **Press Enter.** The variable is successfully changed.

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.

___ 1. In the Source window, go to line **122**.

___ 2. **Double click** on the variable ***IN60** to highlight it.

___ 3. From the **Breakpoints** menu, select **Set watch...** The Watch Breakpoint window appears. The **Expression** entry field is pre filled with the highlighted variable ***IN60**.

___ 4. **Click** on the **OK** push button. The watch breakpoint is now set.

___ 5. **Click** on the **Run** button from the toolbar. The application waits for input from the 5250-emulation session.

___ 6. In the 5250-emulation session, **type 123** for **Project Code** and **D** (for delete) in the **Action Code** field.

___ 7. **Press Enter**. A message is displayed indicating that the variable *IN60 has changed.

___ 8. **Click OK.** The program stops at line **454**. This line is located immediately after the statement which caused the variable ***IN60** to change.
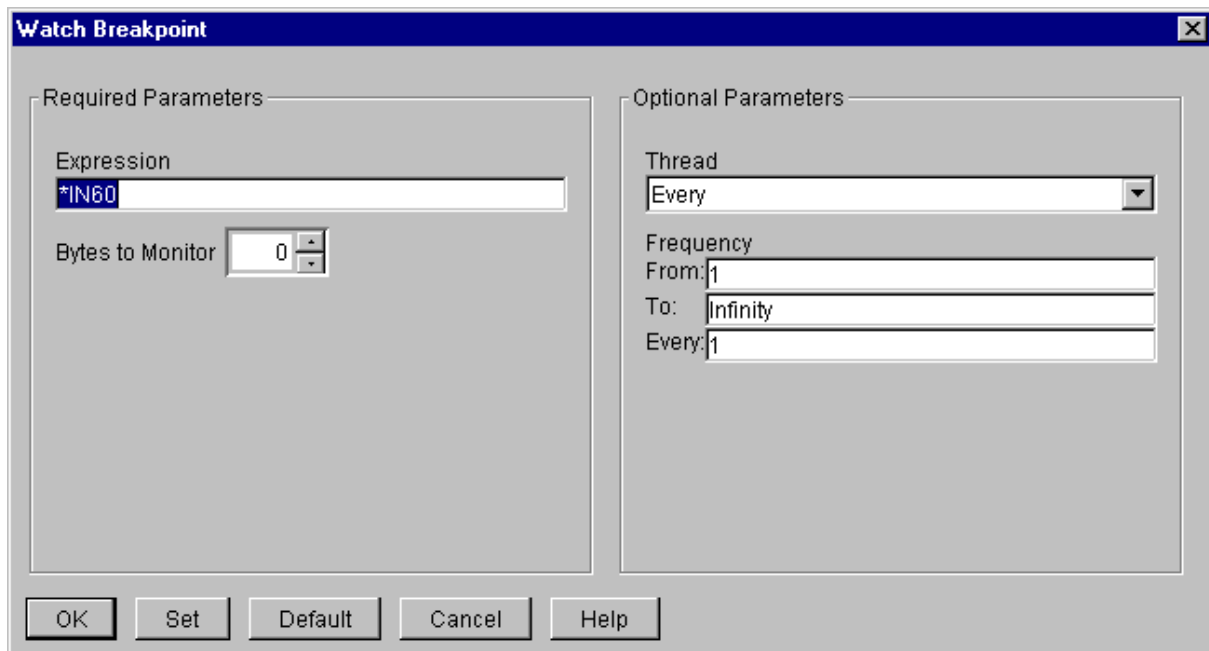
Stepping Through a Program

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

___ 1. Press the step over ⟳ and step debug ⤵ buttons on the toolbar.

___ 2. From the **Debug** menu, select **Step Over** and **Step Debug**.

Closing the Debug Session

This quick tour has demonstrated the main features of the CODE Debugger. Now let's move on to the final major piece of the CODE tool kit -- the CODE Project Organizer.

___ 1. **Click** on the **Run** button from the toolbar. The application waits for input from the 5250-emulation session.

___ 2. Switch to the 5250-emulation session. Press **F3** to end the job.

___ 3. A message **Debug session terminated** comes up. Click on the **OK** and from the **File** menu, select **Exit**.

CODE Project Organizer

The CODE Project Organizer is PDM (Program Development Manager) for the workstation. It allows you to:

• Simplify your work by giving you quick access to lists of iSeries libraries, objects, members, ADM parts, and local files.

• Use the context-sensitive pop-up menus on these lists to perform actions such as start the CODE Editor, Designer, or Debugger or common iSeries actions.

• Use the Project Organizer Actions notebook to create and manage your own user-defined actions and have them appear in the menus.

• Use the Project Organizer's command line to increase your productivity by allowing you to enter and repeat iSeries or local commands without switching to an emulator session.

• Import and export your projects and actions with team members to reduce setup time.

In the following exercise you will use the CODE Project Organizer (CPO) to set up a PAYROLL project which provides quick access to all the pieces that make up the PAYROLL program. You will also see how easy it is to perform actions and define your own actions. In short, you'll see how CPO can organize and integrate your work and make that work easier.

Starting CODE Project Organizer

The CODE Project Organizer organizes your work in terms of projects. A project is a collection of filters (lists that you define) and actions that you use to access and operate on whatever currently interests you.

___ 1. To open a project, locate the **CODE Projects** folder on your desktop and double click on it. The CODE Projects window appears.

___ 2. **Double-click** on the project **new.cpo** to create a new project. The **New Project** dialog appears.

___ 3. In the **Name** entry field, type **Payroll**

___ 4. **Click** on the **OK** push button to create the Payroll project. Both, a Payroll CPO and the **Project Setup** window appear. The Project Setup lets you set some project wide options. These can be changed at any time.

___ 5. **Click** on the **OK** push button. You may get an error that the local directory does not exist. Click **Yes** on the error message to create the local directory.

___ 6. **Click** on the **Project** menu to learn what you can do with projects.

___ 7. Move the mouse over any Project menu item and press **F1** to see help for it.

___ 8. Minimize the help window.
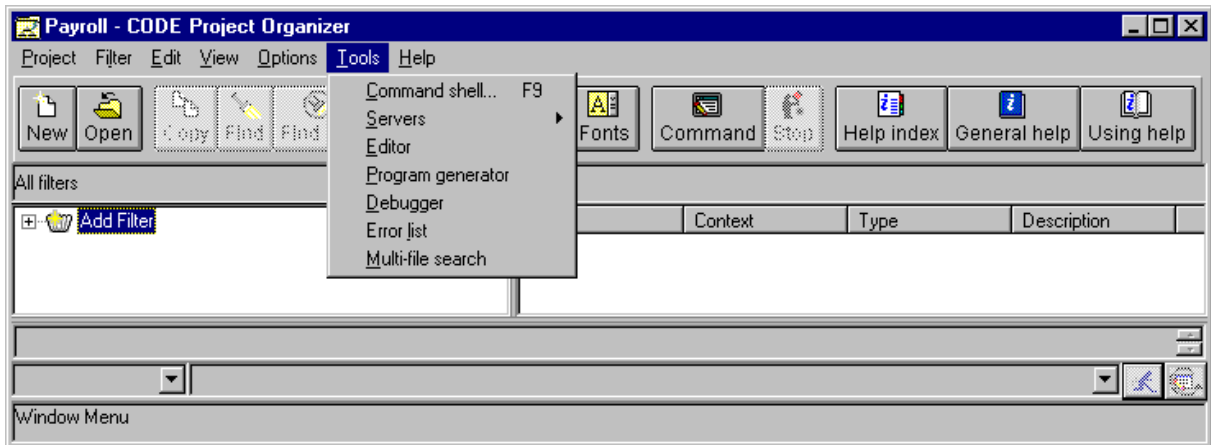
___ 9. **Click** on the **Tools** and **Help** menus. These menus give you quick access to the CODE tools and help. When you are done minimize the help window again.

Creating a Member Filter

A filter is simply a quick way to access some subset of libraries, objects, members, and local files. We are going to start by creating several filters to look at the source members in the **CODELABxx** library.

___ 1. **Click** on the + beside **Add filter** to expand it.

___ 2. **Click** on the + beside **AS/400** to expand it.

___ 3. **Click** on **Member...** to bring up the **Add Member Filter** dialog.

___ 4. In the **File** entry field type *

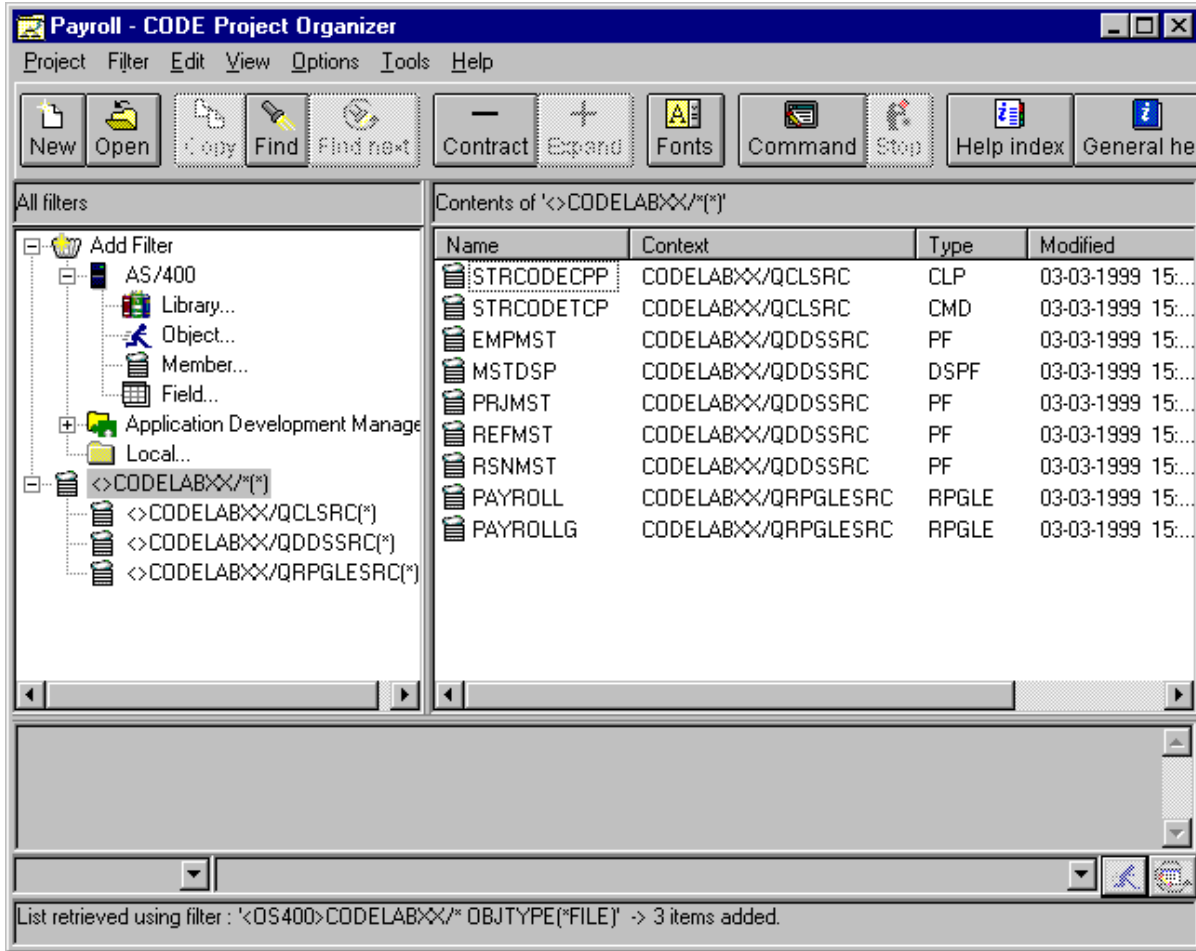___ 5. In the **Library** entry field type **CODELABxx** where xx is your workstation Number.
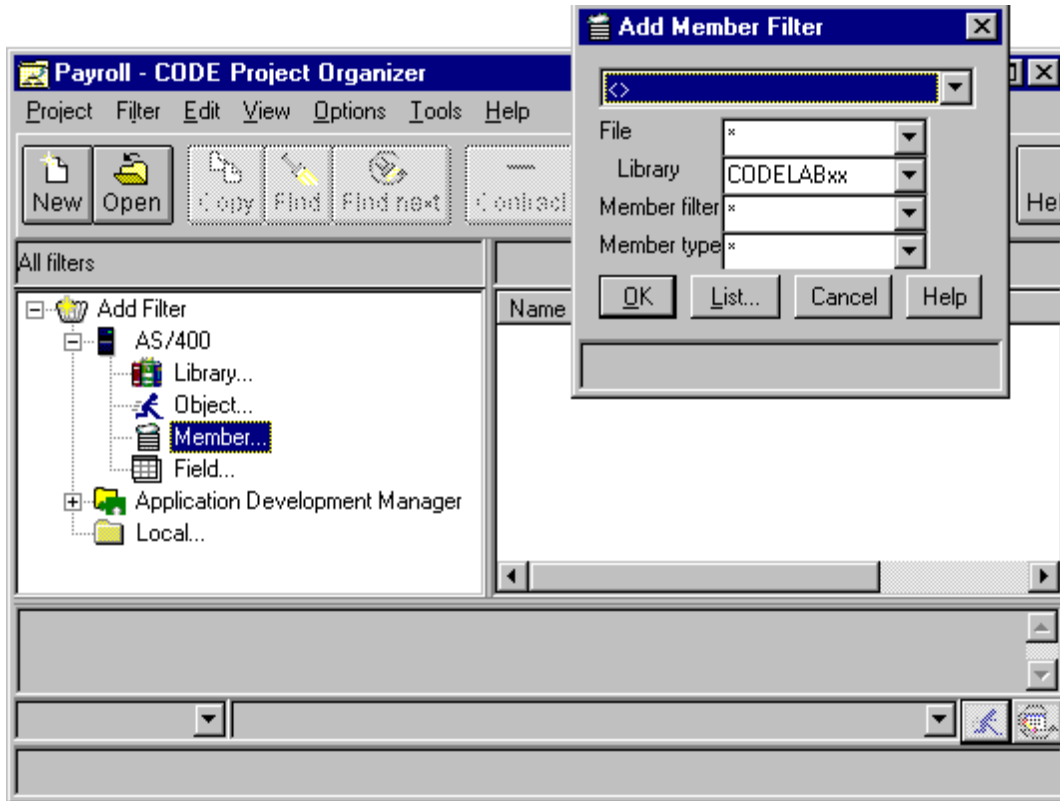
___ 6. **Click** on the **OK** push button. This creates the filter
>CODELABxx/*(*)
which basically says for the current server show all the members in all the
source-physical files in library CODELABxx. The filter is automatically opened and the
matching objects are added to the file list on the right side of the window.

___ 7. **Click** on the + beside the filter <>CODELABxx/*(*). There are three sub filters that let
you see the members in QCLSRC, QDDSSRC or QRPGLESRC.

___ 8. **Click** on each of these filters to see what is in the file list on the right side.


Creating an Object Filter

Now let's create an object filter to look at the objects in library CODELABxx

___ 1. From the **Filter** menu, select the **Add** and then select **Object filter...**

___ 2. Type **CODELABxx** in the **Object library** entry field and leave the defaults in the other
fields.

___ 3. **Click** on the **OK** push button. This creates the filter
>CODELABxx/* OBJTYPE(*)
which shows all objects of all types in the library CODELABxx for the current server.
The filter is automatically opened and the objects are shown in the file list on the right
side of the Window.

___ 4. Let's create another object filter. **Click** the right mouse button on a blank spot beside **Add Filter**. From the pop-up menu select **Add** and then select **Object filter...**
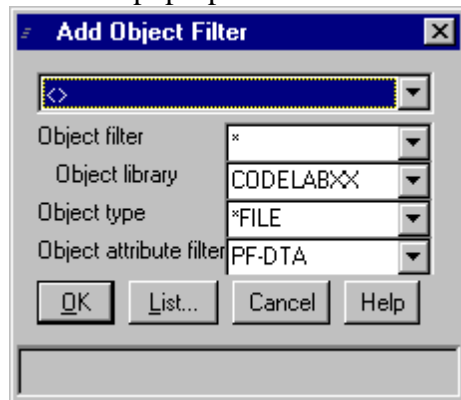
___ 5. Type **CODELABxx** in the **Object library** entry field.
___ 6. Type ***FILE** in the **Object type** entry field.
___ 7. Type **PF-DTA** in the **Object attribute filter** entry field. Leave the defaults in the other fields.

___ 8. **Click** on the **OK** push button. This creates the filter
>CODELABxx/* OBJTYPE(*FILE) OBJATTR(PF-DTA)
which shows all the physical files in the library CODELABxx.

___ 9. **Click** on this filter with the right mouse button to bring up the pop-up menu. This menu gives you a quick way to create and maintain the filter list. Click anywhere outside of the menu to make it disappear.

Creating a Local File Filter
You can also create filters to look at local workstation files.

___ 1. From the **Filter** menu, select **Add** and then select **Local files filter**. The **Add File Filter** dialog appears.

___ 2. In the **Subdirectory** entry field change **x:\WDT400\local** to **x:\WDT400** where **x** is the drive where CODE is installed.

___ 3. Make sure the **File filter** is set to *

___ 4. Press the **OK** button. The filter is created and opened.

___ 5. **Click** on the + sign beside the new filter to expand it and select one of the subfilters. All files that match that filter appear. These are the directories and files that make up most of WDT/400.

Invoking Action*s*
Now you are going to learn how to use CPO to run iSeries and workstation commands or actions.

Invoking a Project-Level Action
Project- level actions are actions that are not specific to the items listed in your filters.

___ 1. From the **Project** menu, select **Actions**. Notice the number of predefined commands. You will learn how to add to this list in a few moments.

___ 2. From the cascaded menu, select the item **Work with spooled files**.

___ 3. The **Confirm action** dialog appears.

___ 4. **Click** on the **OK** button to submit the command to the iSeries. The command is logged in the Monitor area below where the filters are shown. Your 5250 emulation shows the **Work with All Spooled Files** screen.

___ 5. Press **F12** to get back to the EVFCLOGO.

Disabling the Confirm Actions Dialog
You may not want to see the **Confirm action** dialog every time you invoke an action.

___ 1. From the **Options** menu, select **Confirm actions**. The checkmark beside this menu item disappears. The next time you perform an action the dialog will not appear.

Invoking an Object-Level Action
An object-level action is an action that is done on an object in a filter list.

___ 1. Click on the object filter
>CODELABxx/*(*)
to show the objects in the library CODELABxx.

___ 2. In the list of objects, use the right mouse button to **click** on PAYROLL. The context menu has a list of *PGM-related actions. We could start the CODE Debugger from here or run the program.
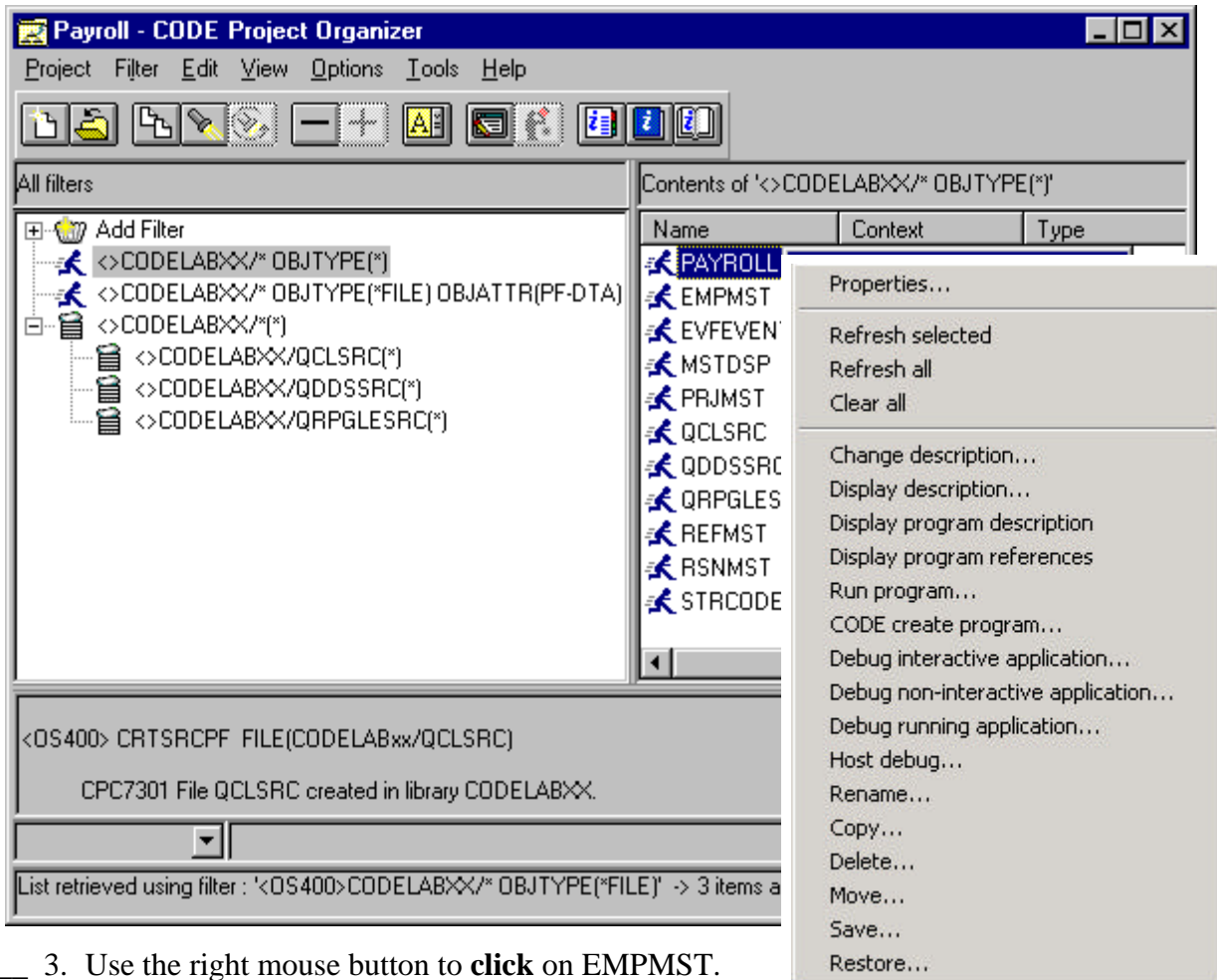
___ 3. Use the right mouse button to **click** on EMPMST. The pop-up menu has a list of *FILE-related actions.
___ 4. Select **DFU update...** from the list. This starts the Data File Utility for you in the 5250-emulation session.
___ 5. **Switch** to the 5250-emulation session. You should be in DFU on the **Update Data with Temp Program** screen.
___ 6. Press **Enter**. This will take you to a screen that would allow you to update the data file. We won't bother doing this right now.
___ 7. **Press F3** to go to the End Data Entry screen.
___ 8. **Press F3** again to leave DFU and switch back to CPO.

Now let's use CPO to load source members into the CODE Designer.
___ 1. Click on the member filter
        <>CODELABxx/*(*)
    to show the source members in the library CODELABxx.
___ 2. In the filter list, right-click on the member **MSTDSP** and select **CODE design** from the pop-up menu. The CODE Designer appears and loads the member.

Now you will invoke an action against several objects at the same time.

___ 1. In the filter list, **click** on the member **EMPMST.**

___ 2. While holding down the **Ctrl** key, **click** on the member **PRJMST**.

___ 3. While holding down the **Ctrl** key, **right-click** on the member **RSNMST**.

___ 4. From the pop-up menu select **CODE edit**. All three members are loaded into the CODE Editor.

You can also hold down the **Shift** key while selecting objects. This will select everything between your first and last selection.

While you were loading members into the CODE Designer and Editor you may have also noticed an action call **Get from host**. This action makes it easy to download source members to your workstation. You can then use the **Put to host** action to upload them back to the iSeries. Automatic checking makes sure that you do not overwrite someone else's changes. As you can see, CPO can be the central launching point for all your CODE and iSeries actions.

The Actions Notebook

The Actions notebook is where you create, change, add, delete and reorder actions that appear on the Project Action menu and on the object pop-up menus.

___ 1. From the **Options** menu, select **Actions...** The CODE Actions notebook appears.

Across the top are tabs describing the types of objects that you can have actions for. There is a set of actions for projects, local files, iSeries objects and members, and ADM objects (ADM is IBM's change management tool for the iSeries). On the left side of each notebook page is a list of the current actions for the object type. On the right side are buttons that allow you to define new actions, and change, delete and reorder existing actions. There is also a Command field that displays the actual command string that CPO invokes.
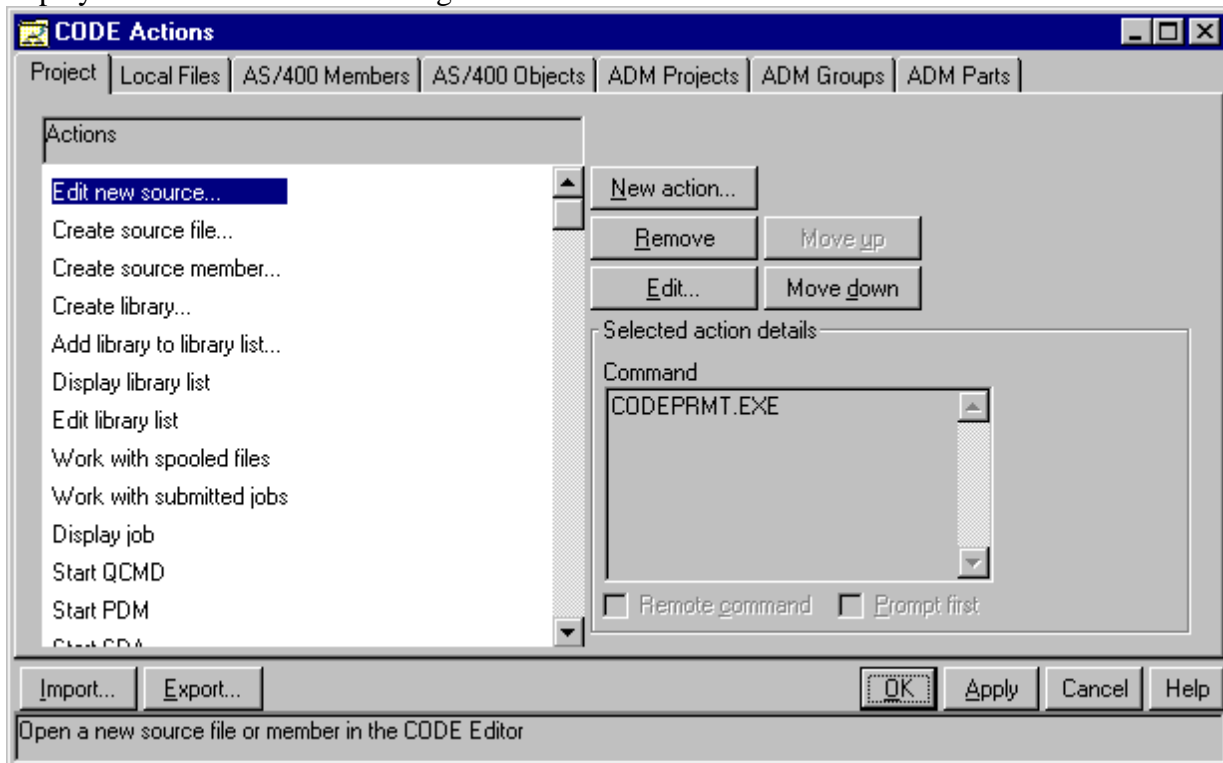
Creating a New Action

Let's say that you need to backup your work regularly. We'll create a new action that will copy the contents of the CODELABxx library into another library CODESAVExx.

___ 1. On the **Project** page, **click** on the **New Action** push button. The **Create Project Action** dialog appears.

___ 2. In the **Action** entry field, type **Backup CODELABxx**.

___ 3. In the **Comment** entry field, type **Save CODELABxx to CODESAVExx**.

___ 4. In the **Command** entry field, type

       CPYLIB FROMLIB(CODELABxx) TOLIB(CODESAVExx)

   (You could also **click** on the **Prompt...** push button for help entering the command.)
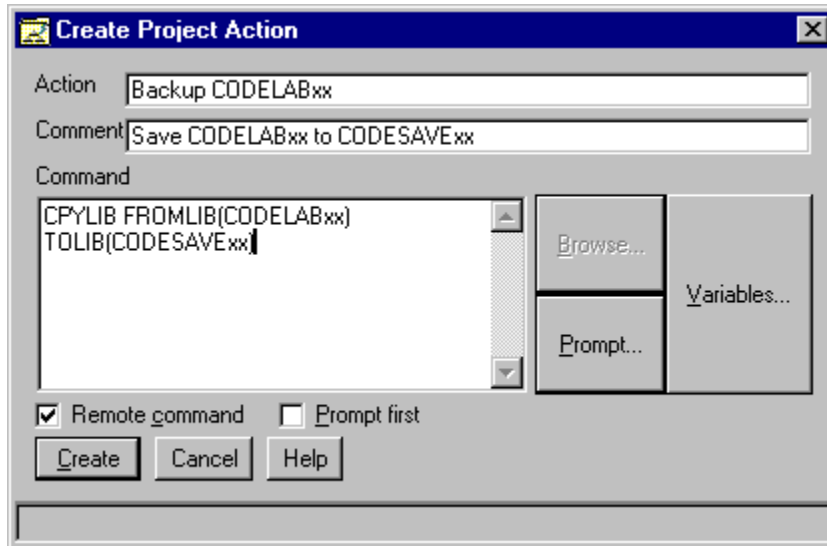
___ 5. **Click** on the **Create** push button. The action is added to the list at the left side of the window.

___ 6. **Click** on the **Apply** push button to activate your changes in CPO.

Now let's have a quick look at some of the iSeries Member actions so you can see some of the powerful features available to you for creating actions in CPO.

___ 1. **Click** on the **AS/400 Members** tab in the **CODE Actions** notebook.

___ 2. Select **SEU edit** in the Actions list.

___ 3. **Click** on the **Edit...** push button. The **Change Member Action** dialog appears. The command string uses several PDM-like replacement variables.

___ 4. **Click** on the **Variables...** push button to see the available variables.

___ 5. **Click** on the **Cancel** push button to return.
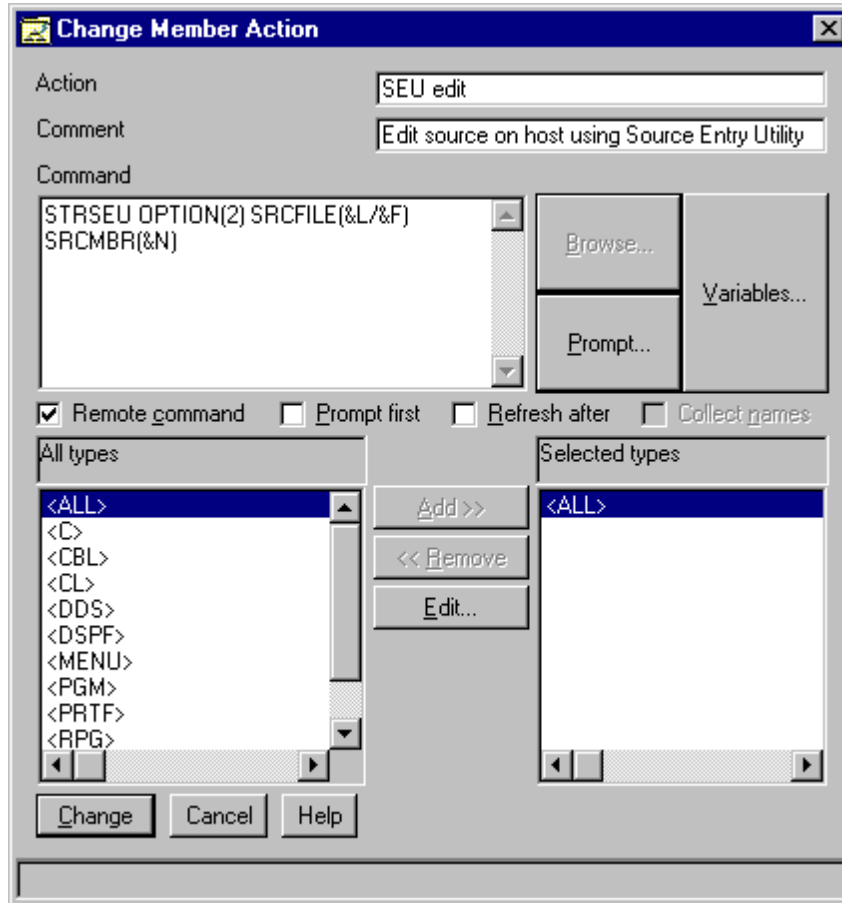
On the bottom of the Change Member Action dialog are two list boxes. These let you select the member types that this command can act on. SEU can act on <ALL> types. If you look at the STRSDA (Start Screen Design Aid) you will see that it can only act on type DSPF (display files). CPO lets you extend its capability even further by allowing you to define your own object types. For example, if you have documentation written in Lotus WordPro, you could define a new source type of .lwp that has an action that calls Lotus WordPro. With this functionality you can use CPO to organize your entire project.

___ 1. **Click** on the **Cancel** push button to leave the dialog.

___ 2. **Click** on the **OK** push button to leave the **CODE Actions** dialog.

___ 3. From the CPO **Project** menu, select **Actions**.

___ 4. From the cascaded menu select **Backup CODELABxx**. The command will execute on the iSeries and the results will appear in the Monitor area of the CPO window.

Running Commands from CPO

Immediately under the Monitor Window is a line that lets you enter commands to be run on either the workstation or an iSeries.

___ 1. Select **<LOCAL>** from the drop-down combo-box beside the CPO command line.

___ 2. On the CPO command line type

c:\dir

and press **Enter** (or click on [icon]   ). A directory for C: appears in the Monitor

Window.

___ 3. Select **<OS400>** from the drop-down combo-box beside the CPO command line.

___ 4. On the CPO command line type

dsplibl

and press **Enter** to see the iSeries library list.

| <OS400> | ▼ | dsplibl | ▼ | [icon] [icon] |

___ 5. **Switch** to the 5250-emulation session, **admire** the library list, and press **F3**. You return
to CPO.

You can run just about any workstation or iSeries command from this command line.

Closing CODE Project Organizer

To close the CODE Project Organizer:

___ 1. From the **Project** menu, select **Exit Project Organizer**.

Congratulations!

You have successfully completed the Introduction to CODE lab. More material can be found
at our web site at http://www.ibm.com/software/ad/wdt400

Happy CODEing!