

IBM WebSphere Development Studio Client for iSeries



Tutorial: Building an e-business application with RPG

Version 5.1

IBM WebSphere Development Studio Client for iSeries



Tutorial: Building an e-business application with RPG

Version 5.1

Contents

Introduction to this tutorial	v
Learning objectives	v
Prerequisite knowledge	vi

Chapter 1. Module 1: Introducing the Web customer inquiry application **1**

Exercise 1.1: Introducing iSeries Web Application Development	1
Exercise 1.2: Introducing Web customer inquiry application	2
Recap.	5

Chapter 2. Module 2: Creating a Web project **7**

Exercise 2.1: Starting the Development Studio Client workbench	7
Exercise 2.2: Opening the Web perspective	9
Perspectives	9
Exercise 2.3: Creating a Web project	10
Exercise 2.4: Setting up the iSeries server information	14
Recap	16

Chapter 3. Module 3: Visually constructing the Web application **17**

Exercise 3.1: Creating a new Web diagram	17
Struts	17
Exercise 3.2: Plotting your new Web project.	20
Recap	22

Chapter 4. Module 4: Creating a Web application **23**

Exercise 4.1: Invoking the Web interaction wizard	24
Exercise 4.2: Specifying the input and output page	25
Exercise 4.3: Defining the iSeries program invocation and parameters	27
Exercise 4.4: Invoking a service program.	27
Exercise 4.5: Defining the structure information for *PGM and *SRVPGM invocation	28
Exercise 4.6: Adding reference fields from DB2/400 to the structure	30
Exercise 4.7: Adding more parameters to the program definition	34
Exercise 4.8: Defining the structure as a parameter	35
Exercise 4.9: Adding a feedback parameter	36
Exercise 4.10: Defining the input page content.	38
Exercise 4.11: Defining the output page content	39
Exercise 4.12: Specifying error handling	41
Exercise 4.13: Enhancing the output page	42
Exercise 4.15: Ensuring customer number field cannot be modified.	44
Recap	45

Chapter 5. Module 5: Adding the iSeries program to your Web application **47**

Exercise 5.1: Opening the Remote System Explorer perspective	47
Exercise 5.2: Finding an iSeries source member	48
Exercise 5.3: Editing a source member	49
Exercise 5.4: Reviewing Remote System LPEX Editor features.	50
Highlighting specification fields	50
Displaying types of lines	51
Checking the syntax of a file	52
Keeping track of columns in a specification line	53
Exercise 5.5: Creating a *PGM object with RPG	53
Fixing errors	56
Exercise 5.6: Creating *SRVPGM with RPG	56
Fixing errors	59
Creating a service program	59
Recap	61

Chapter 6. Module 6: Running the Web application **63**

Exercise 6.1: Opening the Web perspective	63
Exercise 6.2: Finding the Web application	64
Exercise 6.3: Selecting the run on server option	64
Recap	67

Chapter 7. Module 7: Adding an error page. **69**

Exercise 7.1: Creating the flow control page.	69
Exercise 7.2: Modifying your Web interaction	71
Exercise 7.3: Modifying the RPG to implement flow control	76
GETDATA instructions	78
GETDATAS instructions	78
Exercise 7.4: Creating a *PGM object (for GETDATA)	79
Fixing errors	81
Exercise 7.5: Creating a *SRVPGM object (for GETDATAS)	81
Fixing errors	83
Creating a service program	83
Exercise 7.6: Testing the new error page	85
Recap	86

Chapter 8. Module 8: Enhancing the input page using Web tools. **89**

Exercise 8.1: Opening Page Designer	89
Exercise 8.2: Working with page properties	90
Exercise 8.3: Linking a cascading style sheet to the Web page	92
Exercise 8.4: Designing and adding a logo	94
Exercise 8.5: Adding a heading 1 tag to the page	102
Exercise 8.6: Adding a picture to the page	102
Exercise 8.7: Adding moving text to the page.	104

Exercise 8.8: Changing the text color. 107
Recap 109

Programming interface information 112
Trademarks 113

Appendix. Notices 111

Introduction to this tutorial

iSeries™ Web development tools give you the ability to create new e-business applications that use a Web-based front end to communicate with the business logic in an ILE or non-ILE language program residing on an iSeries server. You can create the high-level design of your Web site and apply page templates using Web site designer. Then, you can create the individual pages with Page Designer, or generate input and output JSP files with the Web Interaction wizard. You can also add iSeries Web components to your pages, for example, Web equivalents of iSeries command keys, input fields that accept only particular types of data, or output fields such as subfile names.

Tutorial purpose:

Before you begin, take a minute to read the learning objectives for the tutorial.

Length of time:

This tutorial takes approximately 1.5 hours to complete.

Modules:

Work on the modules in sequence. The pictures in the modules show similar tasks. Some of the names and icons may be different from the environment you will be working with when you complete the modules.

Prerequisites

You should also learn about the prerequisite knowledge for the tutorial before beginning. This section describes the type of knowledge you need to already have to fully benefit from this tutorial.

Learning objectives

In order to successfully develop and test an e-business RPG application, you will need to understand many concepts. This tutorial is designed to help you learn about those concepts while working through the steps. In this tutorial, you will create a simple e-business customer inquiry application. While creating a browser user interface, you will learn how to use the iSeries wizards in Development Studio Client to create the servlets to invoke an RPG program that performs the business logic. You will then learn how to run the application in the WebSphere® Test Environment that is part of Development Studio Client.

The tutorial is broken into seven modules, each with their own specific learning objectives. You can choose to complete all modules, or you can choose to complete only a few, depending on your own learning goals. Each module is composed of several exercises. These exercises must be completed in order to successfully achieve the module's goals.

The first module, Introducing the Web customer inquiry application, shows you the completed customer inquiry application that you will build in this tutorial. You will also learn about the iSeries Web development tools that you will use to create this Web application.

The second module, Creating a new Web project, helps you learn about the Web perspective and the tools in this perspective to help you create a Web project to contain your Web application files and to supply the information which the iSeries server will use for serving business information for this Web application.

The third module, Visually constructing the Web application explains how to create a representation of a Web interaction in the Web diagram. You will learn how to add JSP file nodes and an action mapping node and how to draw connections between these nodes. You will also learn about Struts-based Web applications.

The fourth module, Creating the Web interaction, helps you learn how to create a Web interaction to use an input and output page, and to create the servlet to invoke an RPG program to get data from the iSeries database. While creating this Web interaction you will also learn how to use the Web diagram that you already created in the third module.

The fifth module, Adding the iSeries program to the Web application, you learn how to use the Remote System Explorer perspective to edit and compile an iSeries program or service program that will be invoked by the Web application to get data from the iSeries.

The sixth module, Running the Web application explains how to run the Web application in the WebSphere Test Environment. You also learn about the WebSphere Test Environment.

The seventh module, Adding flow control, helps you learn how to create an informative error page for customers when an incorrect customer number is entered.

The eighth module, Enhancing the input page using Web tools, explains how to add colour and some pictures to make the input page of the Web application more attractive. You will learn how to use the Page Designer tool and some other related Web tools.

Be sure to read about the Prerequisite knowledge for the tutorial before beginning. This section describes the type of knowledge you need to already have to fully benefit from this tutorial.

Prerequisite knowledge

In order to complete this tutorial end to end, you should already have working knowledge of the following:

- Basic Microsoft® Windows® operations such as working with the desktop and basic mouse operations such as opening folders and performing drag-and-drop operations.
- How Web applications work.
- How to use a browser to navigate the Internet.

It is also useful, but not necessary, for you to have basic knowledge of the following:

- DDS
- Servlets and Java™ Server Pages (to understand the generated output of the Web Interaction wizard)

If you do not have current knowledge of these technologies and concepts, you might be interested in learning more about these after you complete this tutorial. You can find some additional information on these topics in the Help system that is part of WebSphere Development Studio Client for iSeries. There are also plenty of resources on the Internet at our product Web page <http://www.ibm.com/software/awdtools/iseries>.

When you are ready to begin, start with Module 1: Introducing the Web customer inquiry application.

Chapter 1. Module 1: Introducing the Web customer inquiry application

In this module, you will be introduced to the Web customer inquiry application that you will build in this tutorial. You will also learn about the iSeries Web tools included with Development Studio Client.

In order to gain an introductory understanding of customer inquiry application and the iSeries Web development tools, there are several steps you will need to learn about and follow:

- Explain the tools available to iSeries programmers for iSeries Web application development
- Describe the Web customer inquiry application

In order to accomplish these learning objectives, there are several steps that are involved, including:

Exercise 1.1: Introducing iSeries Web Development Tools

Exercise 1.2: Introducing the Web customer inquiry application

The exercises in this module must be completed in order. Start with Exercise 1.1 when you are ready to begin.

Length of time:

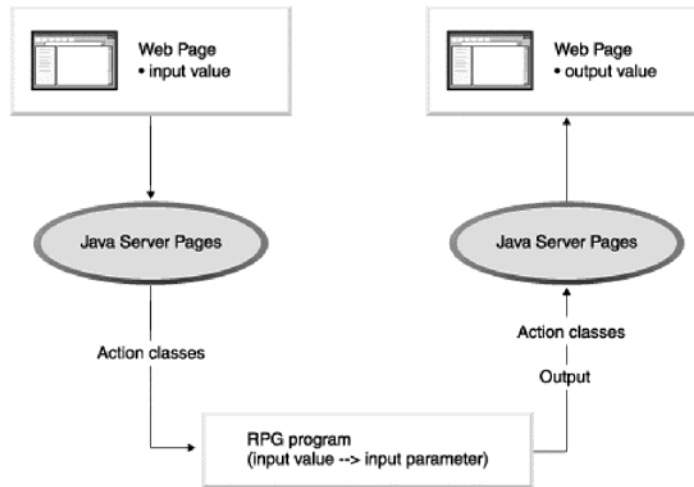
This module will take approximately 10 minutes to complete.

Exercise 1.1: Introducing iSeries Web Application Development

With Development Studio Client you can make your iSeries applications and data accessible beyond the green-screen interface. You can create a new Web interface that connects directly to your program's input and output parameters.

You can create interactive Web pages using Web development tools. Use the Web Interaction wizard to define how your pages interact with one or more ILE or non-ILE applications. This wizard generates Java Action classes and JSP files for use with data from HTML forms. When the end user enters data in a form, the input becomes data to your ILE and non-ILE programs, and the output from the programs is formatted for the Web. Your ILE and non-ILE logic can be separated into different programs for each stage of input and output (known as a Web interaction), or can be a single service program with entry points to handle each Web interaction. The following diagram illustrates the Web development tools

process:



You can easily customize your HTML and JSP files using the editing tools in Web development tools. You can use these tools to create and update input forms, change the appearance or placement of blocks of text, and add backgrounds and images to your pages. The iSeries-specific Web components help you create Web versions of your input and output pages with the same kinds of input validation, output formatting, and subfile controls that native DDS screens provide. You do not need a detailed knowledge of HTML or Java to accomplish these tasks.

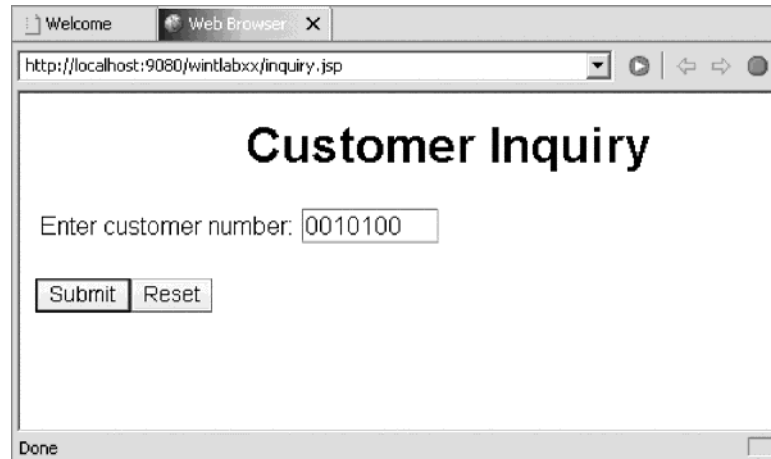
You have an easy test mechanism for your Web-enabled applications. You can easily run your program in the WebSphere test environment of the workbench, quickly make changes, and retest, rather than redeploy your application every time you want to verify functions. When you are finished, you can package and deploy your Web applications as J2EE-based Web archive (WAR) and Enterprise Archive (EAR) files, and then install them in a WebSphere Application Server.

Exercise 1.2: Introducing Web customer inquiry application

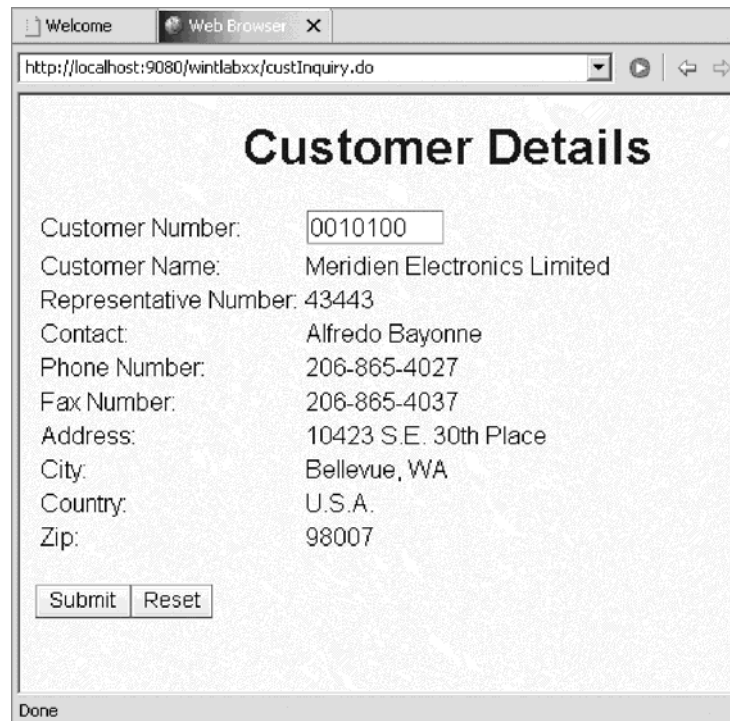
By the end of this tutorial you will have created a simple e-business customer inquiry application that uses a Web-based front end to communicate with RPG business logic residing on an iSeries server. You will first create a representation of a Web interaction in a Web diagram. You will then invoke the Web Interaction wizard to create the input and output pages and to create the servlet to invoke the RPG program or service program to get data from the iSeries database. Next you will add this RPG program or service program to your Web application. You will then run the application in the local test environment on your workstation. You will also create an informative error page for when a customer enters an incorrect customer number.


Here is what the finished e-business customer inquiry application looks like:

After the WebSphere server has started, your Web application Customer Inquiry input page (inquiry.jsp) will display in the workbench browser:

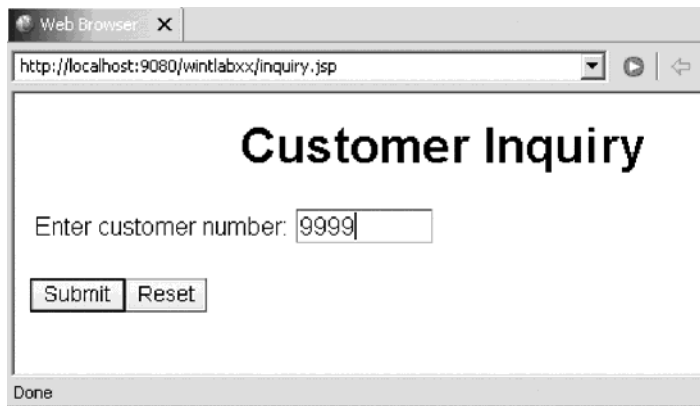


When you enter a customer number and click the **Submit** push button, the Customer Details output page (result.jsp) appears:

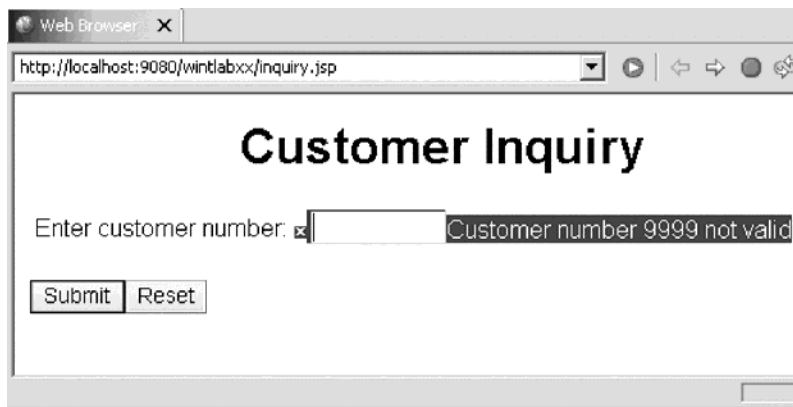


Next, you click the  Back button in the browser to return to the Customer Inquiry input page. Here you enter an incorrect customer number and click

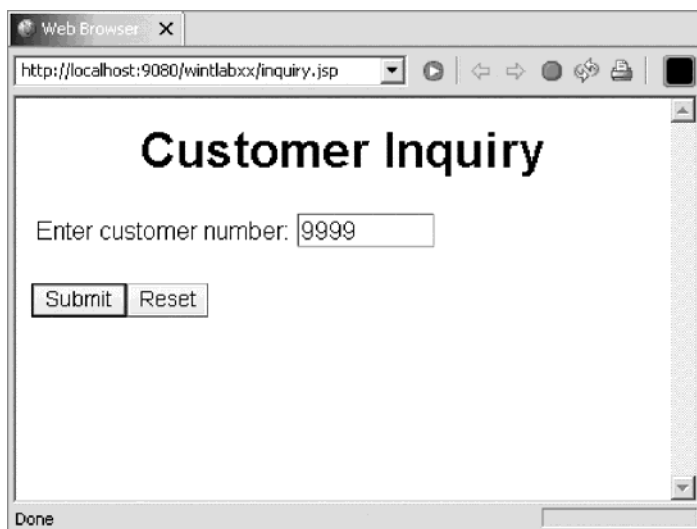
Submit.



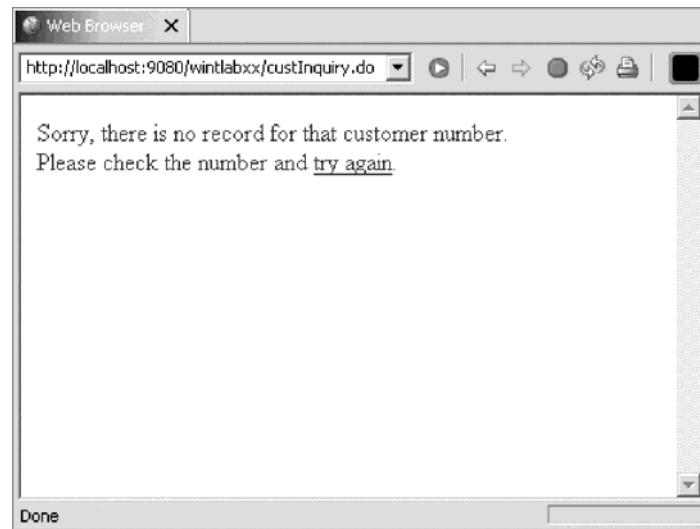
An error message appears.



You then add a new output error page to improve the message information so that if a wrong customer number is entered,



an error page opens instead of a message.



Recap

Congratulations! You have completed Module 1: Introducing iSeries Web Tools and Web customer inquiry application. You should now understand:

- What components make up Development Studio and Development Studio Client
- What components are included in the advanced edition of Development Studio Client
- What components are included in Application Developer
- What components are included in Site Developer
- What tool you use to manage your development cycle tasks
- What the benefits are of Remote System Explorer
- What tool is used to edit source members
- What tool is used to debug programs
- What tool is used to design screens
- What tool you used to save compile time
- The purpose of the Remote Commands view
- The capabilities of each iSeries development tool

Now that you have this introductory knowledge of Development Studio Client, you can continue with Module 2: Starting Development Studio Client and the Remote System Explorer.

Chapter 2. Module 2: Creating a Web project

In this module, you will learn how to create a Web project and then supply the information which the iSeries server will use for serving the business information for this Web application.

Remember: Before beginning this module, you should learn about the Prerequisite knowledge for the tutorial.

In order to create a Web project for the Web application and set iSeries server information, there are several steps you will need to learn about and follow:

- Understanding a perspective
- Understanding J2EE settings
- Understanding a Web project
- Understanding the iSeries Web Tools Run-Time Configuration wizard
- Creating a Web project
- Creating the iSeries server information

In order to accomplish these learning objectives, there are several steps that are involved, including:

- Exercise 2.1: Starting the Development Studio Client workbench
- Exercise 2.2: Opening the Web perspective
- Exercise 2.3: Creating a Web project
- Exercise 2.4: Setting up the iSeries server information

The exercises in this module must be completed in order. Start with Exercise 2.1 when you are ready to begin.

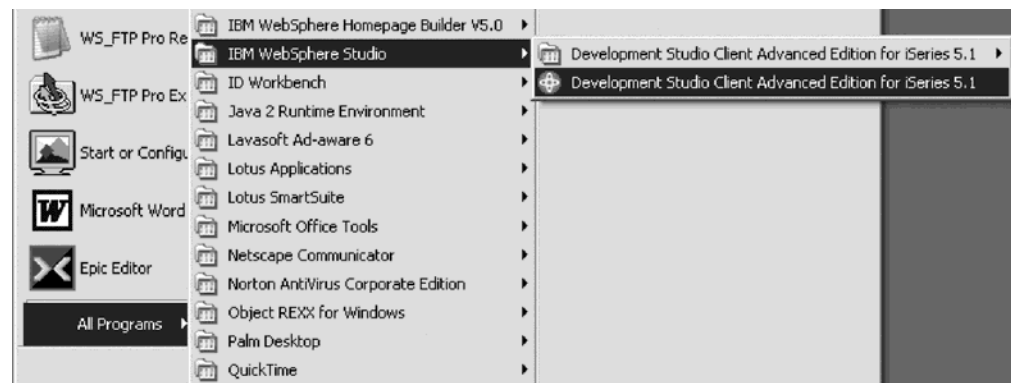
Length of time:

This module will take approximately 5 minutes to complete.

Exercise 2.1: Starting the Development Studio Client workbench

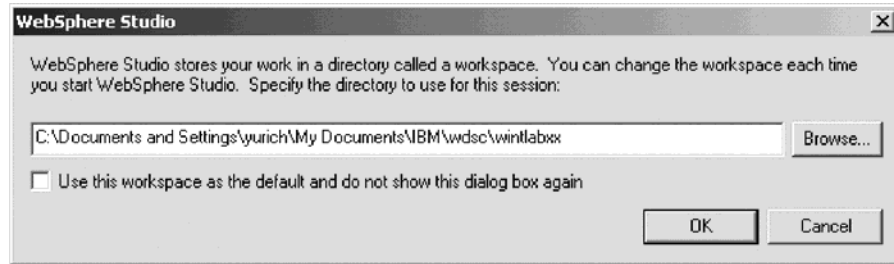
Now go ahead and start Development Studio Client.

On your desktop taskbar:



1. Click: **Start > All Programs > IBM® WebSphere Studio > Development Studio Client Advanced Edition for iSeries 5.1**

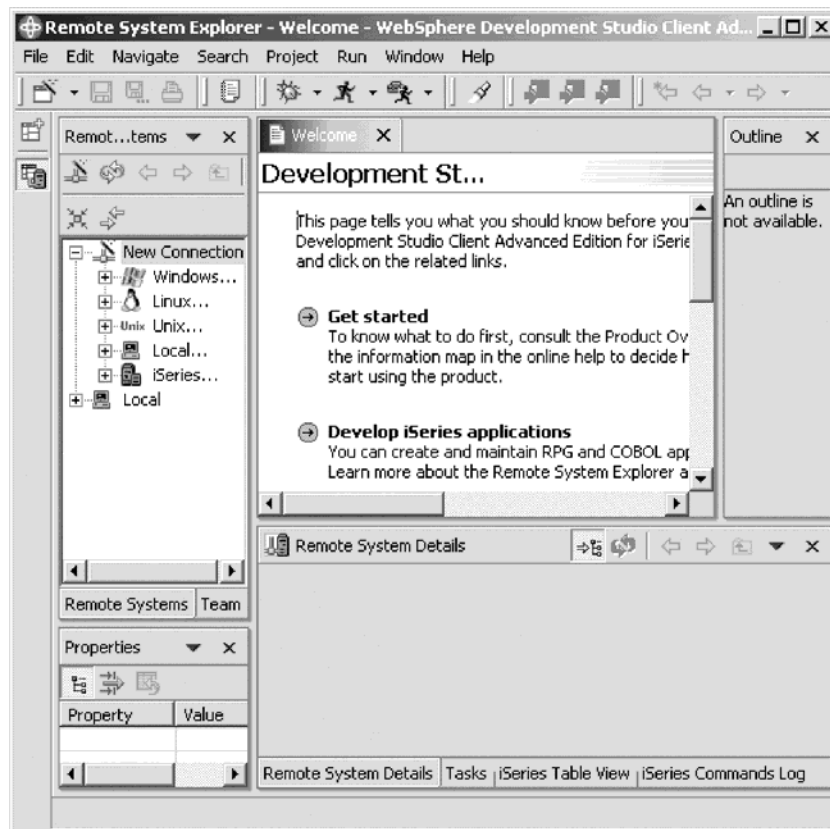
A dialog for workspace selection will appear asking you for the workspace location, unless you used the Development Studio Client before and selected not to show this dialog again.



The workspace contains all the information about your Development Studio projects. You can accept the default or store the work related to this tutorial, in a separate workspace.

2. To name the directory of the workspace as shown above, use the directory name `wintlabxx`.
3. Click **OK**.

After a few moments of loading, the workbench opens, and the initial window of Development Studio Client appears on your desktop:



Exercise 2.2: Opening the Web perspective

In this exercise you will use the Web perspective. This allows you to access tools and views specifically related to Web application development. However, when you first start the Development Studio Client workbench you see the Remote System Explorer (RSE) perspective by default. This is the perspective that you would use to work with iSeries objects. It allows specifying connections to iSeries servers and provides the programmer a similar interface to iSeries objects as the Program Development Manager (PDM) does in a green-screen environment.

Perspectives

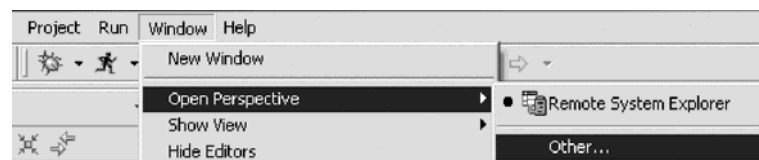
Before we go further, let's review perspectives. Perspectives are a collection of views and tools assembled to help different kinds of users to do their job when they use the workbench. These workbench users will have different jobs (roles), for example, one user needs to work with iSeries objects related to RPG/COBOL, another user is dealing with Java programs, and another user with Web page development. Each of these different user types will need different views of the files/objects they are working with and they need different tools. The perspectives present the user with a selection of specific views/tools geared towards the different roles the users have. The workbench has many predefined perspectives, like the:

- Web perspective for Web developers.
- Remote System Explorer perspective for iSeries programmers.

You can also create your own perspective by using the **Save Perspective as** option under the **Window** menu in the workbench, after you have modified a perspective. Now that you know what a perspective is we can go ahead with our exercise. You are a Web developer in this tutorial and that is the reason you will select the Web perspective. The Web environment provides its own perspective since it needs to give its users access to unique views and tools targeted towards Web tasks.

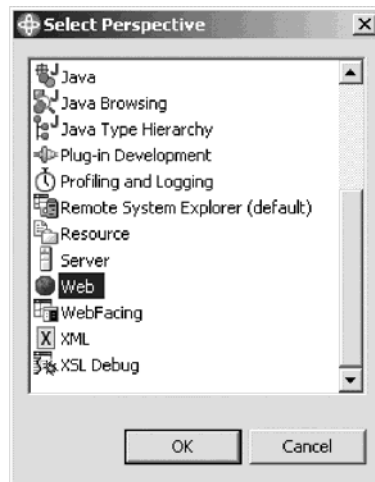
You create a Web project in the Web perspective.

To open the Web perspective:



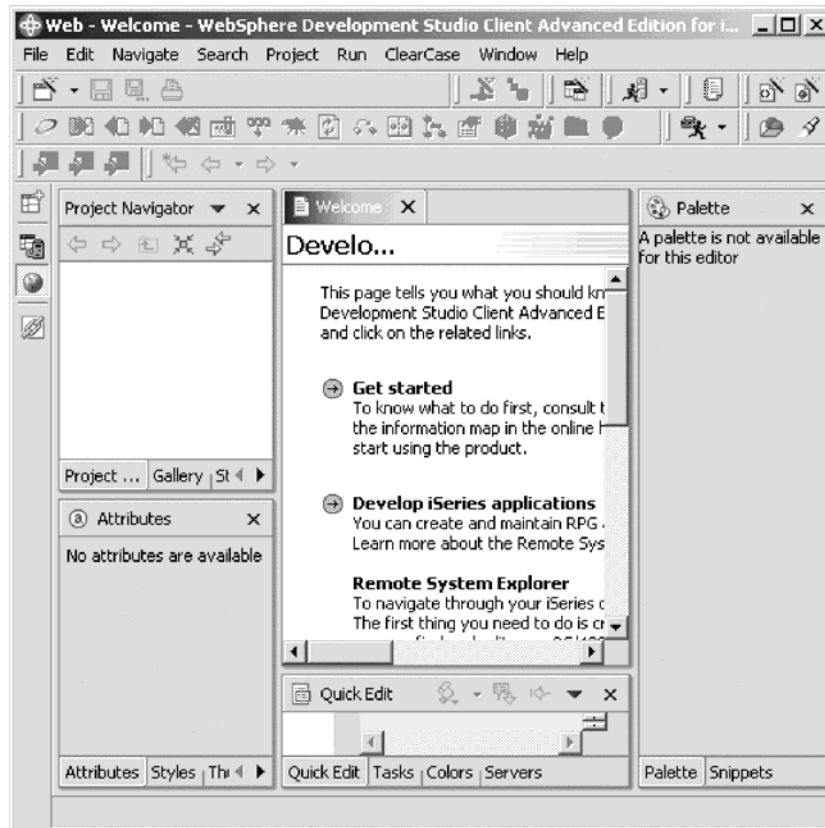
1. Click **Window > Open Perspective** on the workbench menu.
2. Then click **Other** on the pop-up menu.

The Select Perspective window opens:



3. Select **Web** from the list.
4. Click **OK**.

The workbench will now show the Web perspective with the Web Projects view open on the left hand side.



Exercise 2.3: Creating a Web project

Web projects hold all of the Web resources that are created and used when developing your Web application. The first step to creating or importing a Web application is to create either a static or a dynamic Web project. Static Web projects are meant to contain only simple Web site resources, such as HTML files. Dynamic

Web projects are used to structure Web applications that will use more complicated, dynamic Web technologies, such as JavaServer Page files, and possibly data access resources.

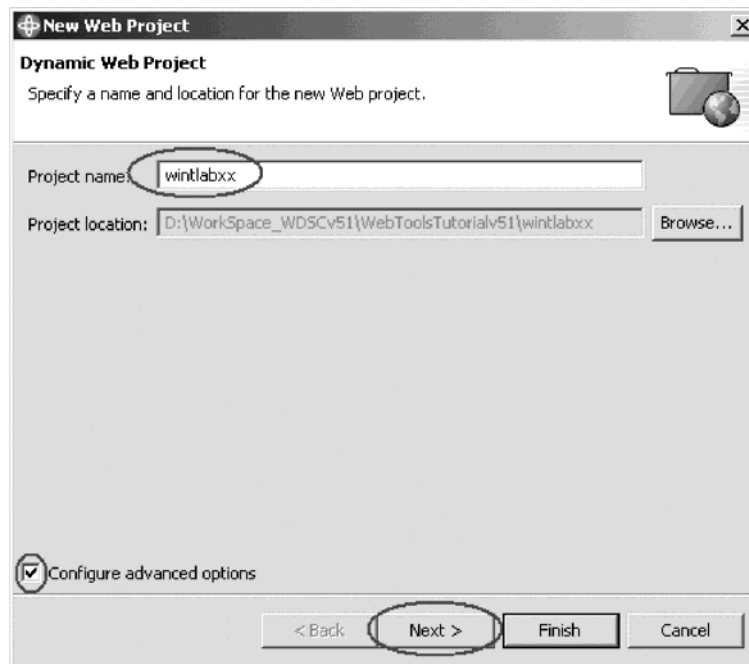
To create a Web project:

1. Click **File > New** on the workbench menu.
2. Then click **Dynamic Web Project** on the pop-up menu.



This starts the Web project wizard. The initial page of the Web project wizard opens.

On the Dynamic Web Project page:

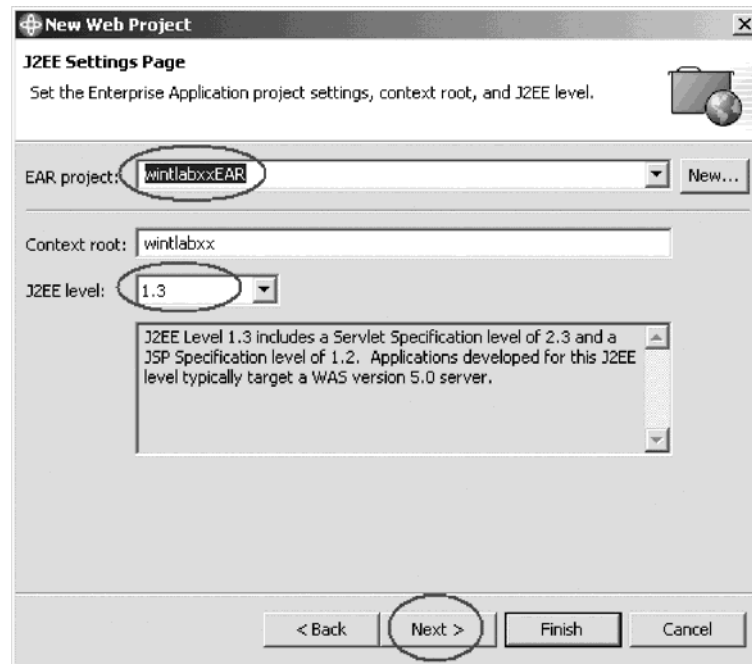


3. In the **Project name** field, type `wintlabxx`.
4. Accept the default value in the **Project location** field.
This is where your project is stored in your file system.

5. Select the **Configure advanced options** check box to specify or customize the project options, such as adding Struts support to your Web project.
6. Click **Next**.

The J2EE Settings page opens.

This page allows you to specify the J2EE level for the Web project. If you plan to use the WebSphere Application Server version 5 only then you select J2EE level 1.3. If you are using previous versions of WebSphere Application Server in your environment and are planning to deploy the Web application to these versions then you use J2EE level 1.2 since these earlier versions of WebSphere Application Server don't support the J2EE level 1.3. In this exercise you use the Web application in the WebSphere Test Environment, which is version 5, so you use J2EE level 1.3 which is the default.

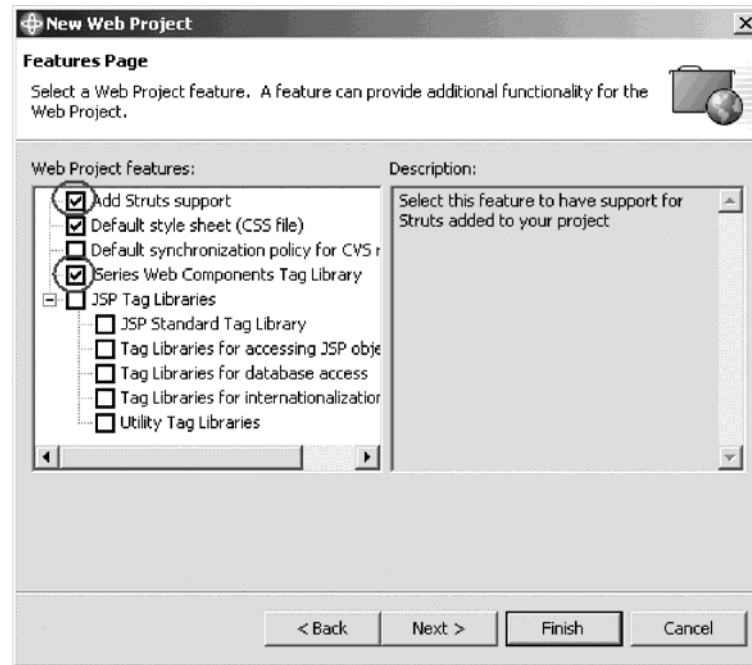


7. In the **EAR project** field, type wintlabxxEAR.
A new or existing Enterprise Application project must be associated with your new Web project for purposes of deployment.
The Enterprise Application project is new, so you typed the name of the new project. When your Web project is created at the end of the wizard, the new Enterprise Application project (EAR file) is also created. The default is an Enterprise Application project named DefaultEAR located in the same directory as your new Web project.
8. Leave the default value for the **Context root** field.
The default value for the context root is the name you provided for your Web project. The context root is the Web application root, the top-level directory of your application when it is deployed to a Web server. The context root can also be used by the links builder to ensure that your links remain ready to publish as you move and rename files inside your project.
9. Leave the default value in the **J2EE level** list.
By default, the Web project's J2EE level is set to the Workbench's J2EE level.

Any new servlets and JSP files that you create should adhere to the latest specification level available; previous specification levels are offered to accommodate any legacy dynamic elements that you expect to import into the project.

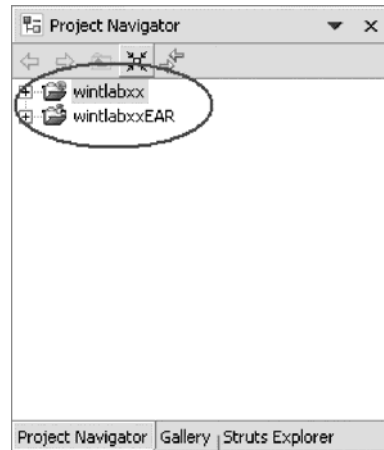
10. Click **Next**.

The Features Page opens:



11. Select the **Add Struts support** check box as you want to create a project that uses Struts technology.
12. Leave the default **Default style sheet (CSS)** check box selected to create a default CSS file (called Master.css) for any HTML and JSP files included in the project.
13. Select the **iSeries Web Components Tag Library** check box as you want to add iSeries fields and controls to the Web pages of your Web application.
14. Click **Next** to go to the Struts Settings page and ensure that **Create a Resource Bundle for the Struts Project** is selected. If not, select the **Override default settings** check box and then select **Create a Resource Bundle for the Struts Project**.
15. Click **Finish** to create the Web project.

Now you are back in the workbench and your new project and project files appear in the Project Navigator.

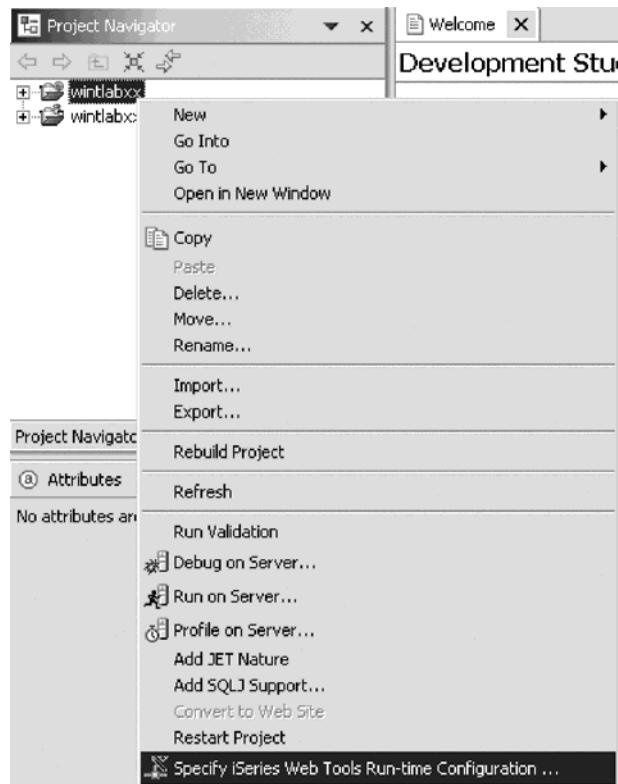


Exercise 2.4: Setting up the iSeries server information

Before you can start creating your application, you will need to define where the RPG program you will connect to later on in this tutorial is located. You need to specify the iSeries server name and the user-ID and password to be used when starting the job on the iSeries to run your program.

To set up the iSeries server:

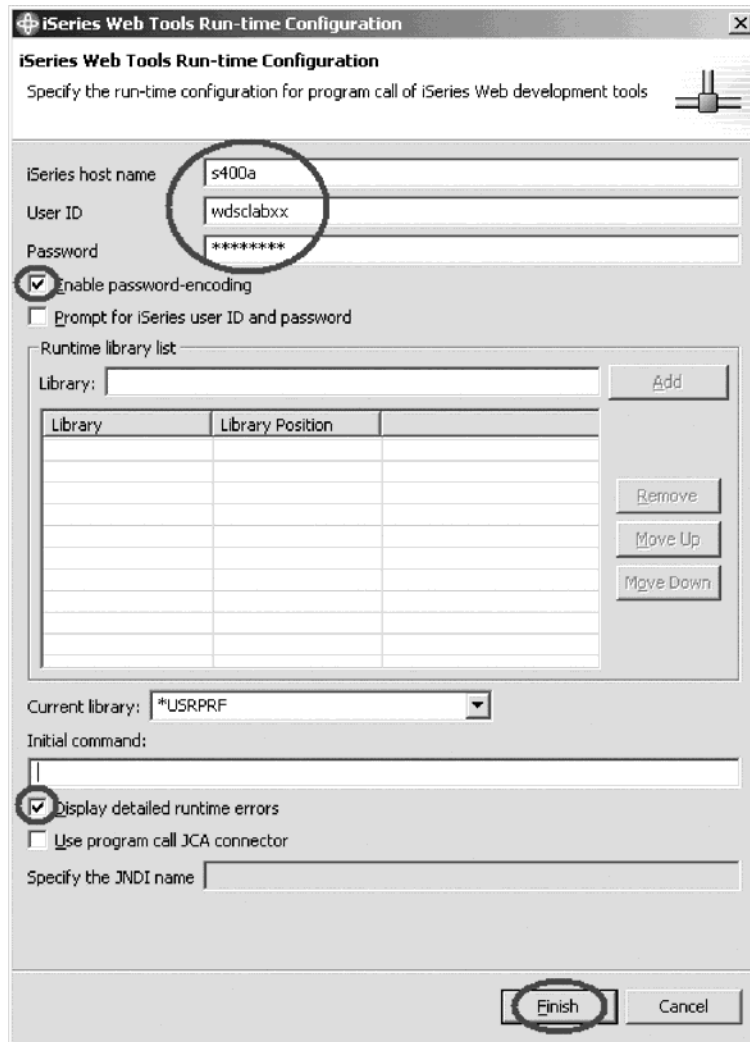
1. In the Web perspective, right-click the **wintlabxx** project in the Project Navigator.



2. Click **Specify iSeries Web Tools Run-Time configuration** on the pop-up menu.

Tip: You could also use the  button from the workbench toolbar to get the Runtime configuration wizard..

The iSeries Web Tools Run-Time Configuration wizard opens:



iSeries Web Tools Run-time Configuration

Specify the run-time configuration for program call of iSeries Web development tools

iSeries host name: s400a

User ID: wdsclabxx

Password: *****

Enable password-encoding

Prompt for iSeries user ID and password

Runtime library list

Library: Add

Library	Library Position

Remove

Move Up

Move Down

Current library: *USRPRF

Initial command:

Display detailed runtime errors

Use program call JCA connector

Specify the JNDI name:

Finish Cancel

You use this wizard to specify how Web interactions that perform program calls to an iSeries host should be connected and authenticated.

3. In the **iSeries host name** field, type the name of the iSeries host where your program is located, for example s400a.
4. Enter your user ID and password for the iSeries host in the appropriate fields. You need to ensure that your User ID has been set up in a way that it contains the correct library list. If not, you will have to use this table to add a library to the library list when the job gets started to run your program.
5. Select the **Enable password-encoding** check box. This option ensures that the password appears in encoded format in the web.xml file. If you clear the check box for this option, the password appears as plain text in the web.xml file.
6. Select the **Display detailed runtime errors** check box to see the details for any errors that occur during the run time of your Web application. This is useful for tracing and debugging development errors that may occur.
7. Click **Finish** to define authentication and run-time values for your host program or procedure call to be made by your Web interaction in the Web project.

Recap

Congratulations! You have completed Module 2: Creating a Web project. You should now understand:

- How to start the workbench
- The concepts of the workspace and perspectives, the Web perspective and J2EE settings
- The importance of creating a Web project
- The importance of setting up iSeries run-time configuration information
- How to open the Web perspective
- How to create a Web project
- How to set up iSeries server information

Now that you have created a Web project, you can continue to Module 3: Visually constructing the Web application.

Chapter 3. Module 3: Visually constructing the Web application

In this module you will learn how to create a representation of a Web interaction in the Web diagram by adding JSP file nodes and an action mapping node to the free-form surface. You will learn how to draw connections to or from the nodes and the action mapping node. In the next module you learn how to invoke the Web interaction wizard to create the JSP files and to define and generate the Web interaction.

In order to visually construct the Web application, there are several steps you will need to learn about and follow:

- Understanding Struts
- Understanding the Web Diagram palette
- Creating various Web objects
- Making connections between Web objects
- Using the Web Diagram tool to predefine variables for the Web Interaction wizard

In order to accomplish these learning objectives, there are several steps that are involved, including:

- Exercise 3.1: Creating a new Web diagram
- Exercise 3.2: Plotting your new Web project

The exercises within this module must be completed in order. Start with Exercise 2.1 when you are ready to begin.

Length of time:

This module will take approximately 10 minutes to complete.

Exercise 3.1: Creating a new Web diagram

A Web diagram is a view that helps you visualize the flow structure of a Struts-Based Web Application. Because of the indirectness involved with a Struts application, being able to visually see the application's flow can help you to better understand the application.

Struts

Before you create a Web diagram, let's first look at what Struts is all about. First, Struts is a set of Java classes and JSP tag libraries that provide a conceptual framework for developing Web applications. The Struts technology is open source and was developed as part of the Apache Software Foundation's Jakarta project.

Second, Struts provides numerous, custom JSP tags that are simple to use but are powerful in the sense that they hide information. The Page Designer does not need to know much about form beans, for example, beyond the bean names and the names of each field in a given bean.

Third, you can use the diagram editor to show all or part of a Struts application. For example, suppose you have a three-part Struts application. One part handles the login process, one part handles product inquiries, and a third part handles

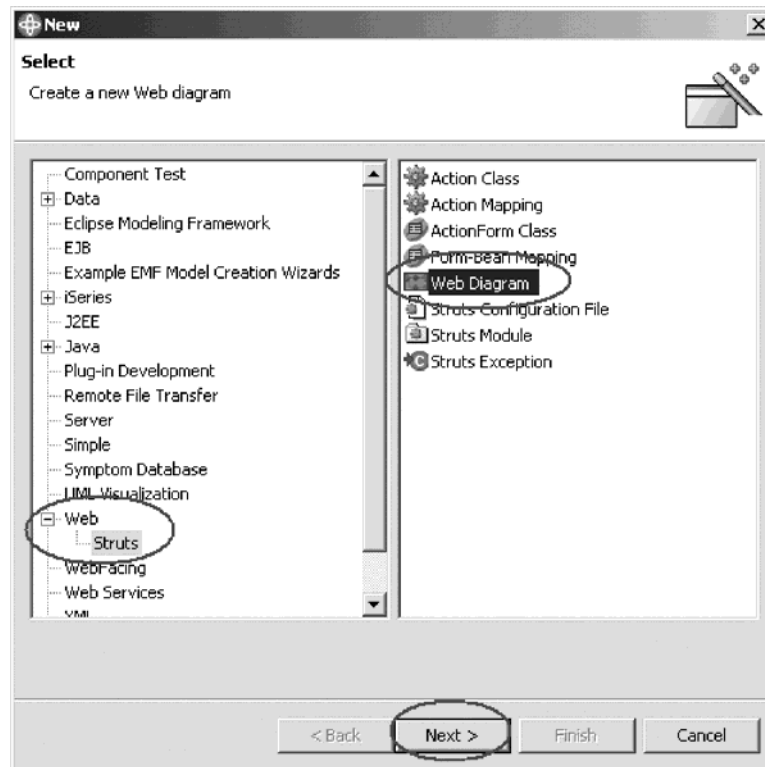
product updates. In this case you could draw three diagrams to represent this system, or you could draw the entire system in a single diagram. Because one diagram can be included inside another, it would probably make more sense to represent this Struts application using a set of three diagrams.

Now, that you know what Struts is, you can go ahead with this exercise.

To create your Web diagram:

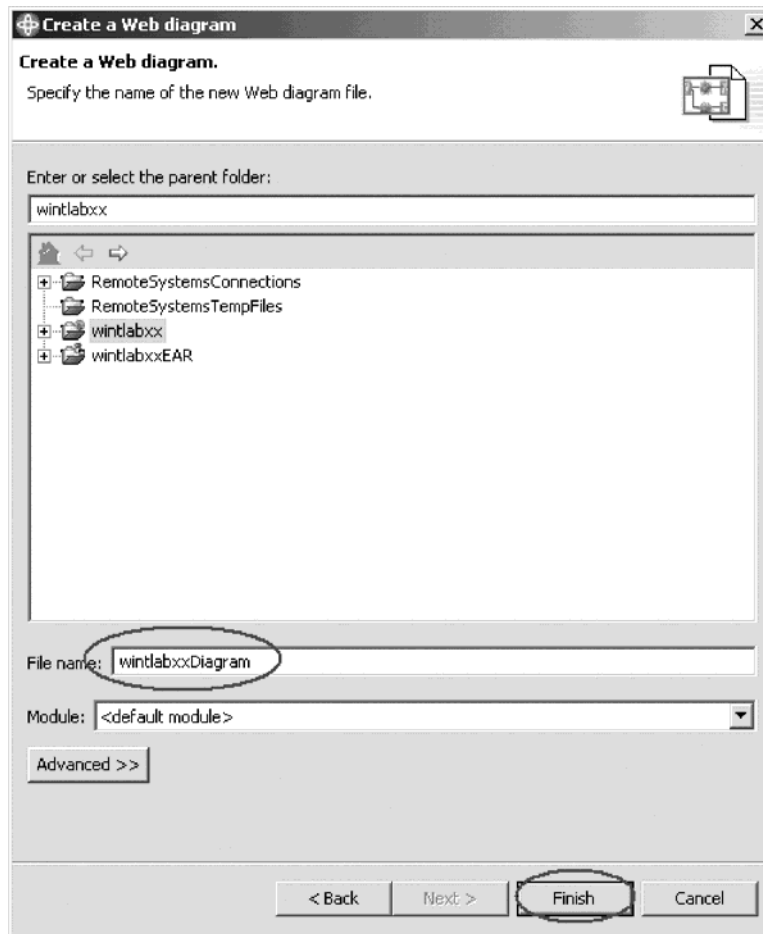
1. Click **File > New** on the workbench menu.
2. Then click **Other** on the pop-up menu.

The Create a new Web diagram page opens:



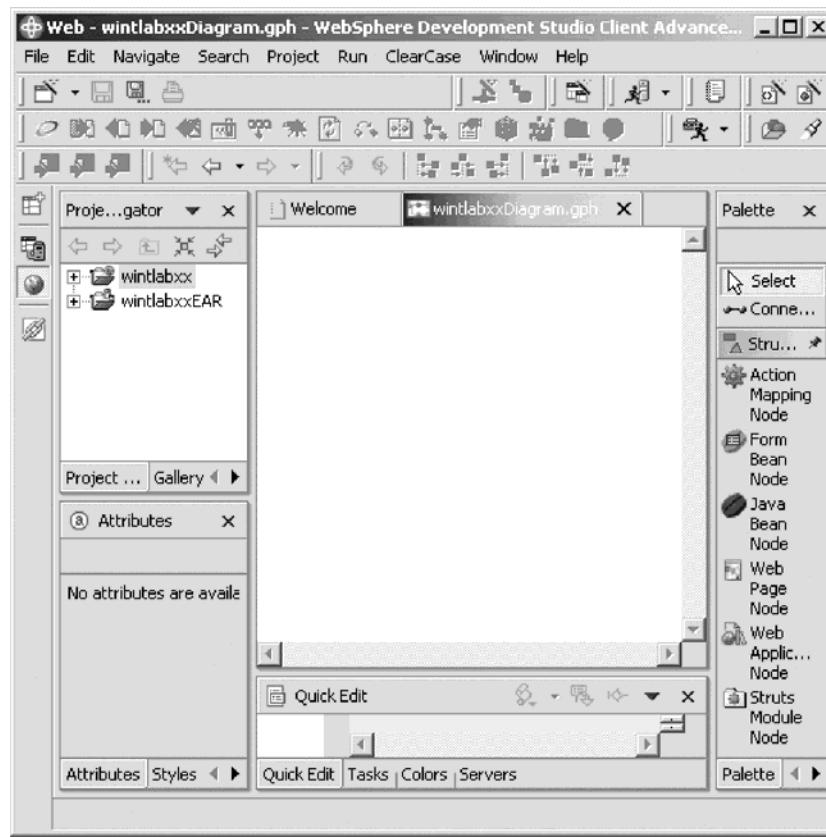
3. In the left pane of the Create a new Web diagram page, expand **Web** and then select **Struts**.
4. In the right pane of the Create a new Web diagram page, select **Web Diagram**.
5. Click **Next**.

The Specify the name of the new Web diagram file page opens:



6. Select your project **wintlabxx**.
7. In the **File name** field, type `wintlabxxDiagram` as the name of the diagram.
The `.gph` file extension is automatically applied to the name.
8. Click **Finish** to create the file in your project.




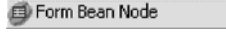
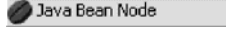
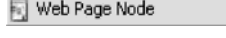
The file opens on a free-form surface in the editor view:



After you create a Web diagram, you can add JSP file nodes and an action mapping node to the free-form surface in the editor view to represent the Web interaction that you want to generate.







Exercise 3.2: Plotting your new Web project

Let's become familiar with some of the icons in the Palette. Locate these icons in the Palette view on the right of the free-form surface in the editor view.

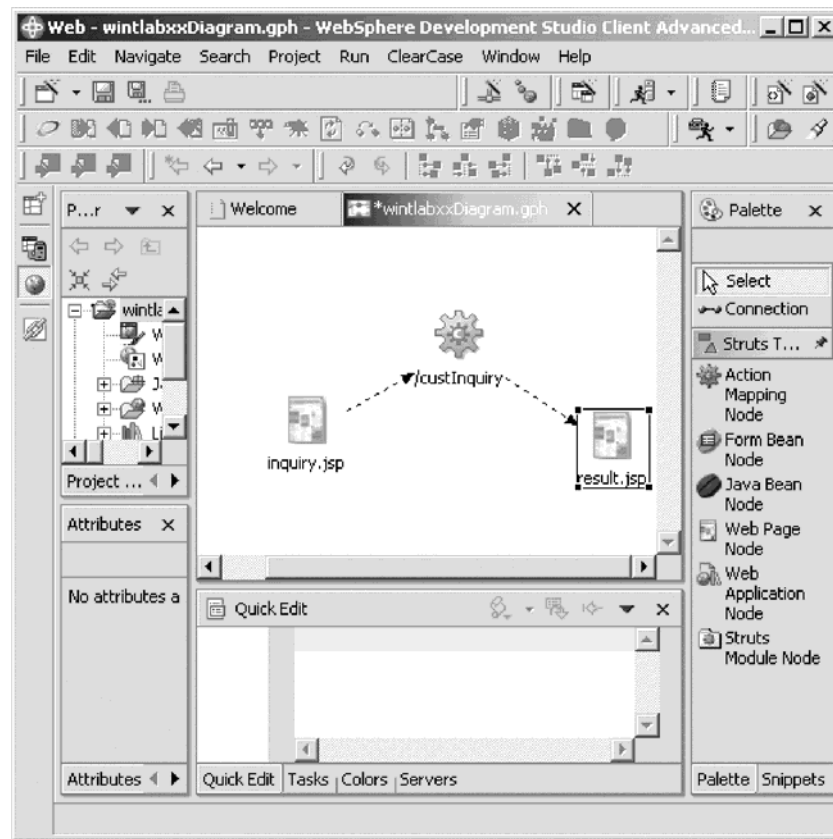
- Select  in the Palette view is used to select the objects on the diagram.
- Connect Two Nodes  in the Palette view is used to connect two objects (nodes) on the Web Diagram together to create a flow in the design.
- New Action Mapping Node  in the Palette view is used to create a Web Interaction Node.
- New Form Bean Node  in the Palette view is used to create a Form Bean Node.
- New Java Bean Node  in the Palette view is used to create a Java Bean Node.
- New Web Page Node  in the Palette view is used to create a JSP Node.

In this exercise you use the Web Interaction Node and the JSP Node to plot your Web Project.

To plot a Web project:

1. Select New Web Page Node  Web Page Node in the Palette view and click somewhere to the upper left in your Free Form Surface (FFS).
This creates a JSP Node.
2. Rename the JSP to `inquiry.jsp`.
3. Select New Web Page Node  again in the Palette view. To create another JSP node, click in your FFS again, somewhere to the right from the first jsp you created. Name this JSP node `result.jsp`.
4. Select New Action Mapping Node  Action Mapping Node in the Palette view and click somewhere in your FFS.
This creates a Web Interaction Node.
5. Rename the Web Interaction to `custInquiry`.
6. Select Connect Two Nodes  Connection in the Palette view to connect the nodes together.
This tool is direction sensitive, so make sure you do the following steps in order.
7. Click **`inquiry.jsp`** and click **`custInquiry`**.
This creates a link between these two nodes with an arrow that indicates that the page is input to the action
8. Connect Two Nodes  Connection is still selected. Now click **`custInquiry`** then click **`result.jsp`**.
9. Press **Enter** to accept the new name.
10. Select Select  Select in the Palette view to switch to select mode.

Your Web Diagram should now look like this:



11. Click the Save icon on the workbench toolbar to save the Web diagram.

Recap

Congratulations! You have completed Module 3: Visually constructing a Web application. You should now understand:

- The purpose of Struts
- The purpose of a Web diagram
- How to create a Web diagram
- How to plot a Web project
- The features in the Web Diagram palette

Now that you have your entire Web project planned out and ready for implementation, you can continue to Module 4: Creating a Web interaction.

Chapter 4. Module 4: Creating a Web application

In this module, you will learn how to create the interaction to use an input and output page, and to create the servlet to invoke an RPG program to get data from the iSeries database.

Remember that you have a Web diagram where you already defined the name of the interaction, the input and output pages. You will invoke the Web Interaction wizard from your Web diagram.

In order to create a Web interaction, there are several steps you will need to learn about and follow:

- Understanding an interaction
- Understanding input and output pages
- Accessing an iSeries database from a Web browser
- Using existing DB2/400 field definitions to create the necessary Program Call Markup Language (PCML) information to be able to communicate with an iSeries program
- Using externally described iSeries fields in your Web pages
- Getting data from the iSeries by invoking an iSeries program that accesses the database and returns the data through parameters
- Reading and writing data from and to a Web page
- Creating an input (inquiry) page for a Web application with the Web Interaction wizard using the Web diagram input page
- Creating an output (result) page for a Web application with the Web Interaction wizard using the Web diagram output page
- Generating Java code for servlets using the Web Interaction wizard.

In order to accomplish these learning objectives, there are several steps that are involved, including:

- Exercise 4.1: Invoking the Web Interaction wizard
- Exercise 4.2: Specifying input and output pages
- Exercise 4.3: Defining the iSeries program invocation and parameters
- Exercise 4.4: Invoking a service program
- Exercise 4.5: Defining the structure information
- Exercise 4.6: Adding reference fields from DB2/400 to the structure
- Exercise 4.7: Adding more parameters to the program definition
- Exercise 4.8: Defining the structure as a parameter
- Exercise 4.9: Adding a feedback parameter
- Exercise 4.10: Defining the input page content
- Exercise 4.11: Defining the output page content
- Exercise 4.12: Specifying error handling
- Exercise 4.13: Enhancing the output page
- Exercise 4.14: Ensuring the customer number field cannot be modified

The exercises in this module must be completed in order. Start with Exercise 4.1 when you are ready to begin.

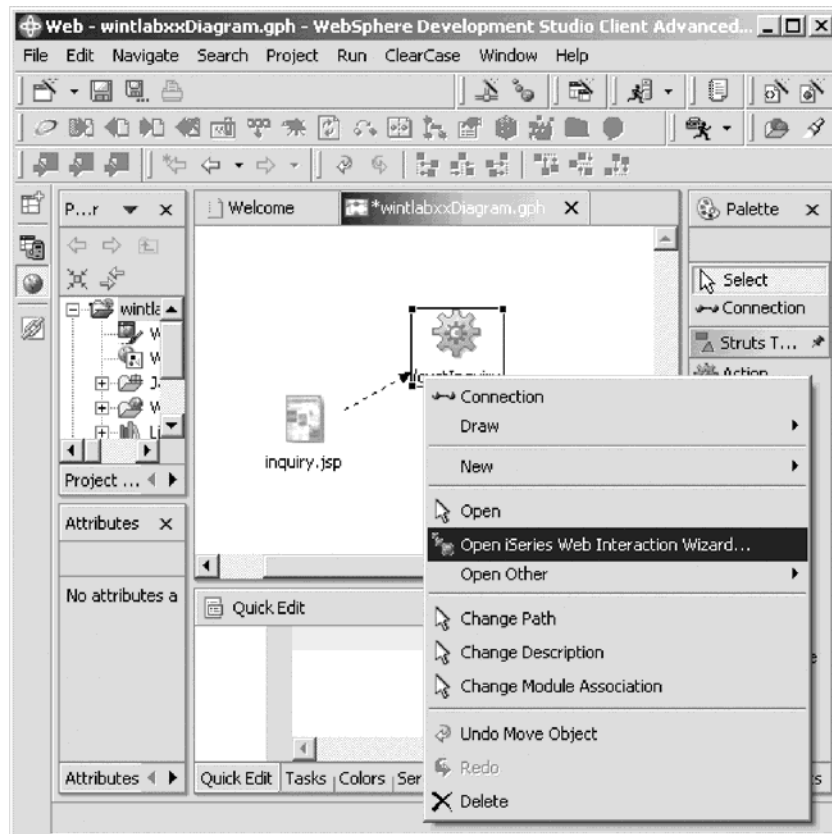
Length of time:

This module will take approximately 20 minutes to complete.

Exercise 4.1: Invoking the Web interaction wizard

During this exercise you will use the Web Interaction wizard to create the interface description for parameters to be passed to an iSeries program. You will also generate a Web page based on the parameters definition used by the program. Some of the parameters will be defined based on existing iSeries database field definitions.

To invoke the Web Interaction Wizard from your Web Diagram:

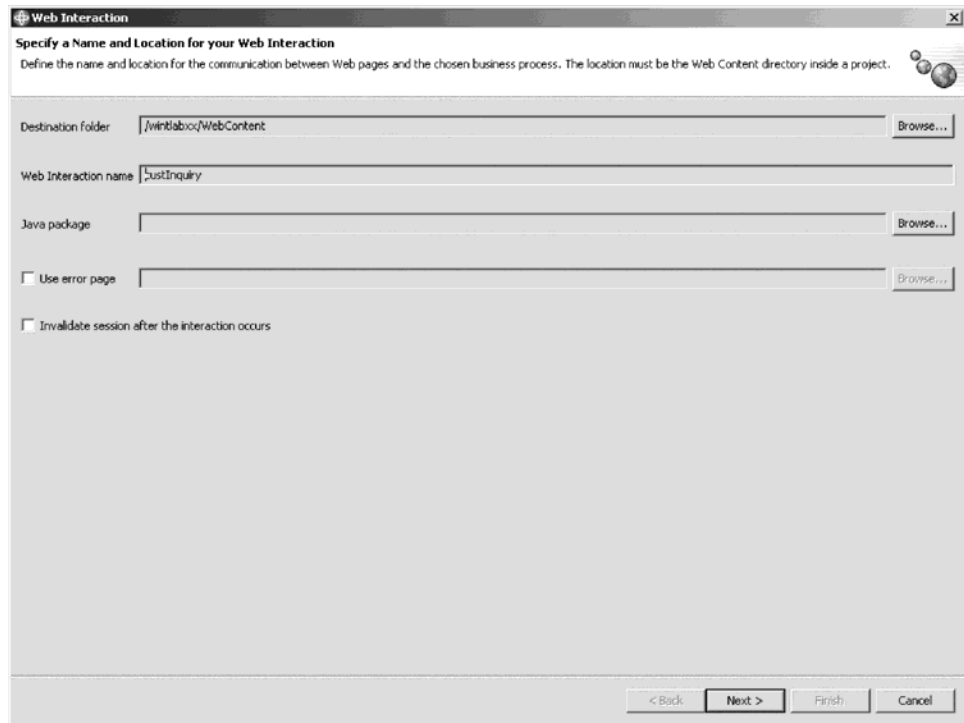


1. Right-click **custInquiry** action mapping node in your Web diagram.
2. Click **Open iSeries Web Interaction Wizard** on the pop-up menu.

An interaction is the span between the submit of a request from the current Web page to the post of a new Web page. An interaction could be a simple request to the HTTP server to load another page. In this environment an interaction will be made up of a call to an iSeries program, waiting for the return of the program with data and then invoking a JSP to show the next Web page containing the data returned from the program.

First you have to give the interaction a name so you can reference it later on, but since you invoked the interaction from your Web diagram, the name has already been filled in for you.

The Specify a Name and Location for your Web Interaction page opens:



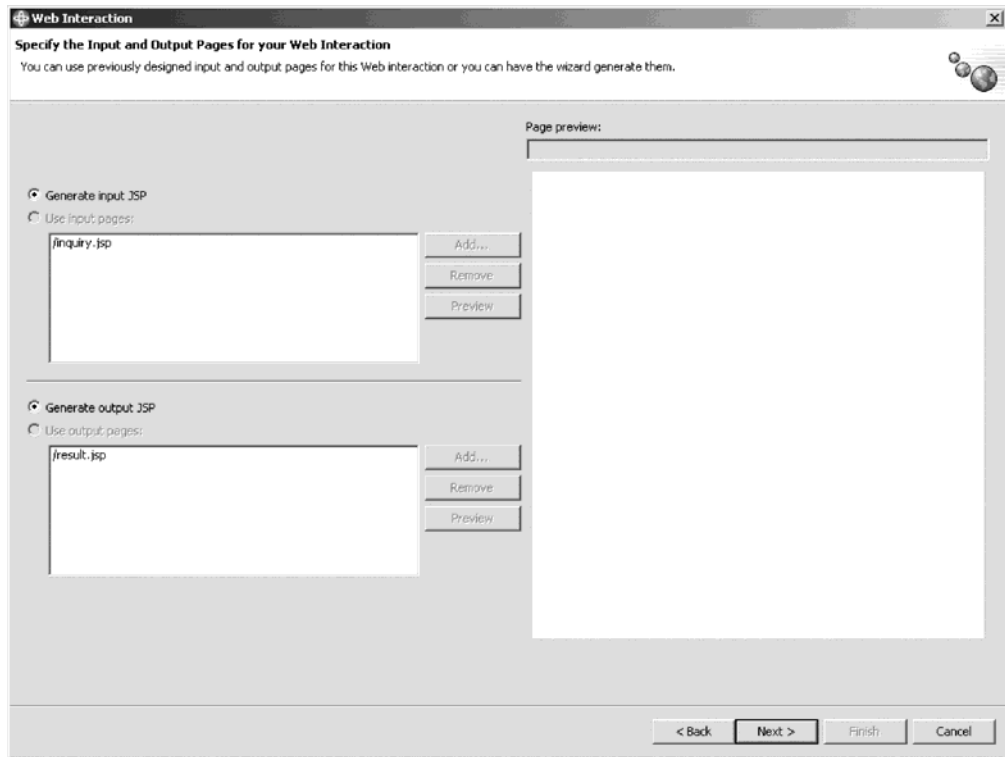
3. Click **Next** in the Web Interaction wizard.

The Specify the Input and Output Pages for your Web Interaction page opens.

Exercise 4.2: Specifying the input and output page

The next page of the Interaction wizard asks for information about the Web pages that are involved in this interaction. You have an input page that invokes the interaction and an output page that displays the results of interaction. The output page shows in the browser at the end of the interaction. The wizard generates both Web pages for you. Since you plotted the input and output pages in the Web Diagram, they are already listed as pages to be created by the Web Interaction

Wizard.

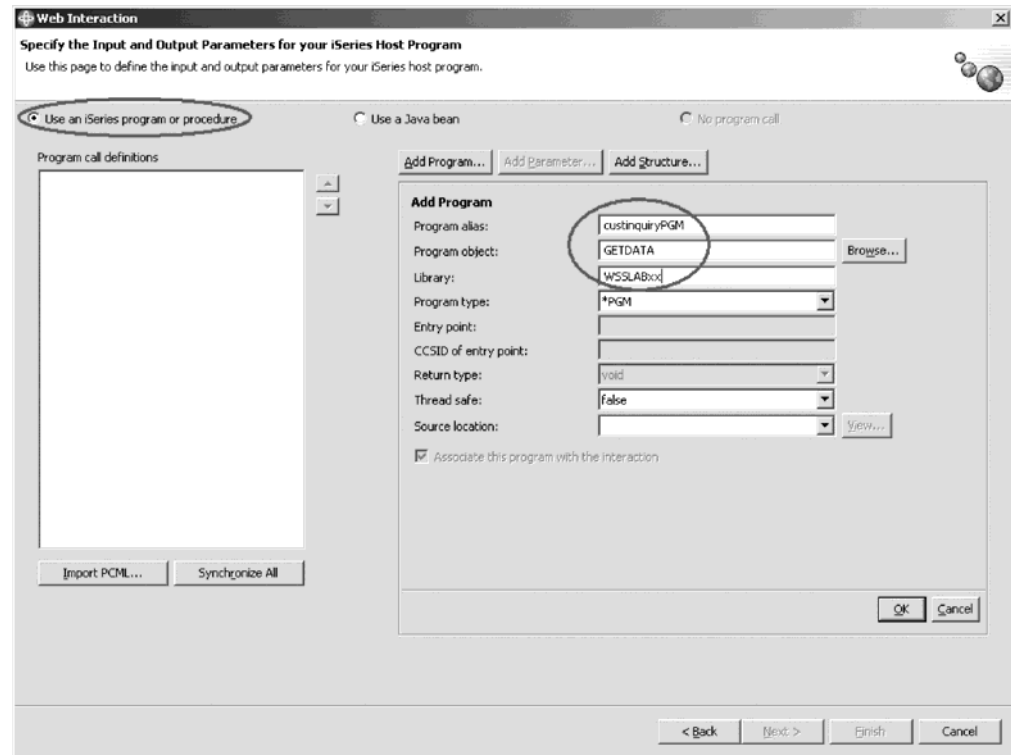


1. Click Next.

The Specify the Input and Output Parameters for your iSeries Host Program page opens.

Exercise 4.3: Defining the iSeries program invocation and parameters

The next page asks you to define the parameters you want to pass between the Web pages and the iSeries program that processes the requests.



To define the program that you want to call on the iSeries:

1. The **Use an iSeries program or procedure** radio button is selected by default. You use this page of the wizard to describe the program or procedure that is invoked as the business logic for this Web interaction.
2. In the **Program alias** field, type `custinquiryPGM`.
This value is used when creating the Java classes for this interaction.
3. In the **Program object** field, type `GETDATA`.
This is the name a program object or a service program object.
4. In the **Library name** field, type `WSSLABxx`.
This is the name of the library that contains the program object.
Now you have to make a choice for program type.
To work with Service programs and procedures in Service programs follow the steps in Exercise 4.4: Invoking a service program.
5. To work with a regular program, leave the **Program Type** field as `*PGM`.
6. Click **OK** in the middle of the Add Program pane to add the program definition to the Program call definitions area of the page.
7. Click **Yes**, if you are asked to save the changes.
8. Follow the steps in Exercise 4.5: Defining the structure information for `*PGM` and `*SRVPGM` invocation.

Exercise 4.4: Invoking a service program

Follow these steps if you want to work with procedures in service programs.

If you want to invoke a regular *PGM from your Web page then go to Exercise 4.5: Defining the structure information for *PGM and *SRVPGM invocation.

The Service program you call will have a different name, so first change the program name:

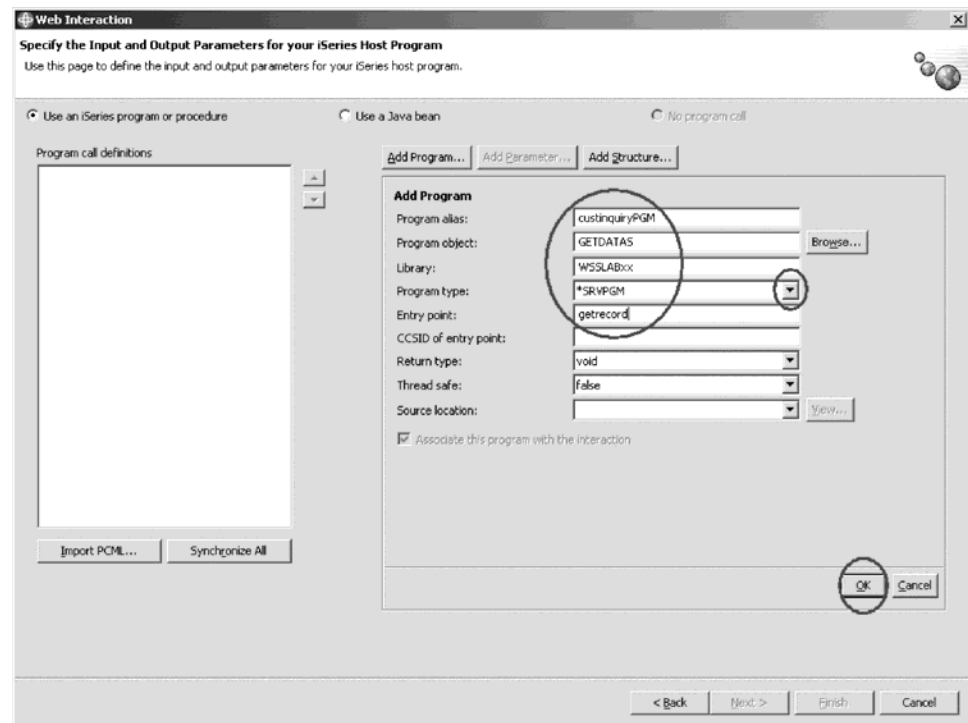
1. Select the **Program object** field.
2. In the **Program object** field, type GETDATAS
3. Select ***SRVPGM** from the **Program type** list.
4. In the **Entry point** field, type getrecord.

This is the procedure name in your service program.

Note: Enter the Entry point name in lower case.

5. Click **OK** to add the program definition to the Program call definitions area of the page.
6. Click **Yes** if you are asked to save the changes.

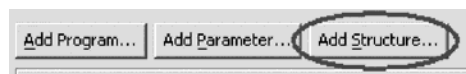
The special instructions for calling a Service program ends here.



Exercise 4.5: Defining the structure information for *PGM and *SRVPGM invocation

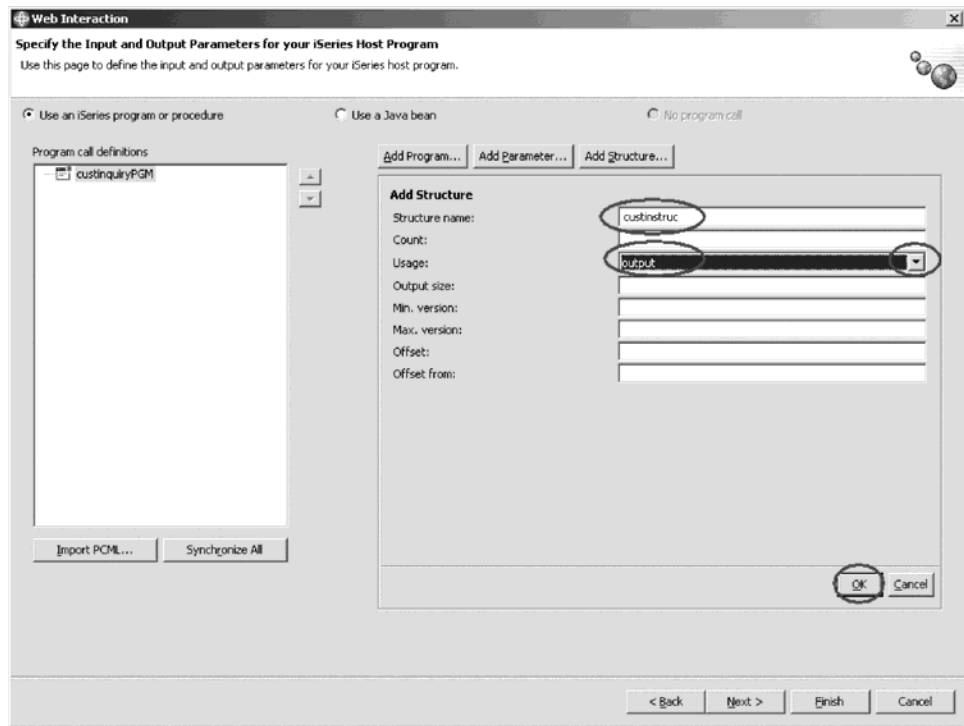
Now you add the parameter description for the program call. One of the parameters is a structure. First you describe the layout of the structure and then you describe the parameters.

To define the structure:

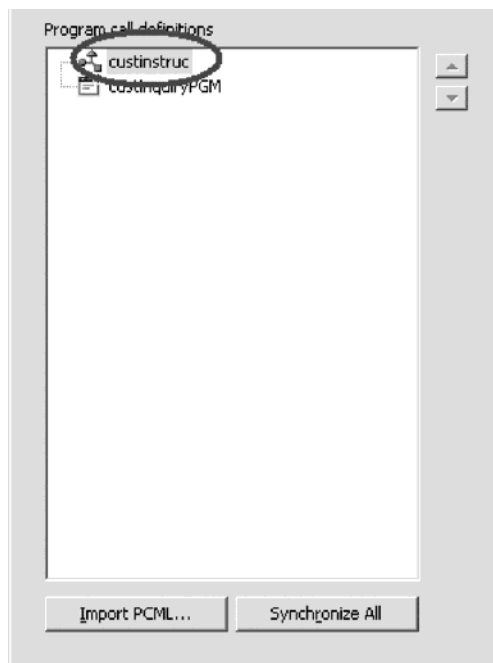


1. Click **Add Structure**.

In the right pane of the Specify the Input and Output Parameters for your iSeries Host Program page, the Add Structure pane opens:



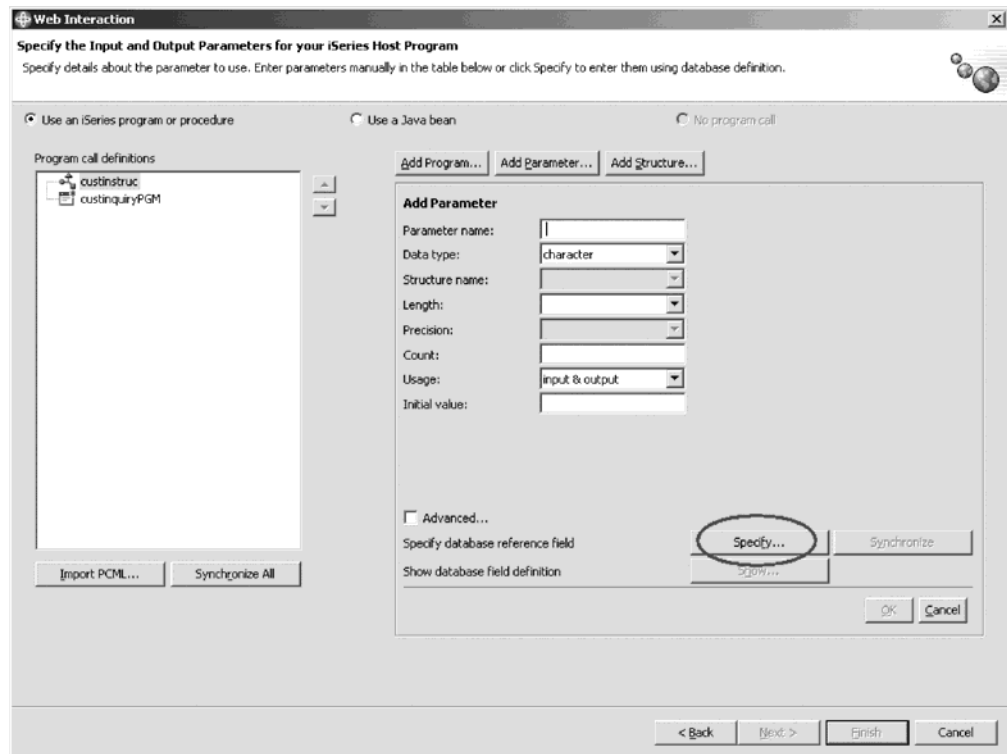
2. In the **Structure name field**, type `custinstruc` as the name of the structure.
3. Change the **Usage** list to **output**.
The structure receives output from the host program.
4. Click **OK** to add the structure definition to the Program call definitions area on this page of the wizard.



This tells the wizard that the fields in the structure will only be used to transfer data from the program to the Web page.

Exercise 4.6: Adding reference fields from DB2/400 to the structure

In the right pane of the Specify the Input and Output Parameters for your iSeries Host Program, the Add Parameter pane opens. It allows you to add fields to the structure. These fields can be reference fields from DB2/400.



You use a record format on the iSeries to describe the layout of the structure so you need to connect to the iSeries server to select the record format.

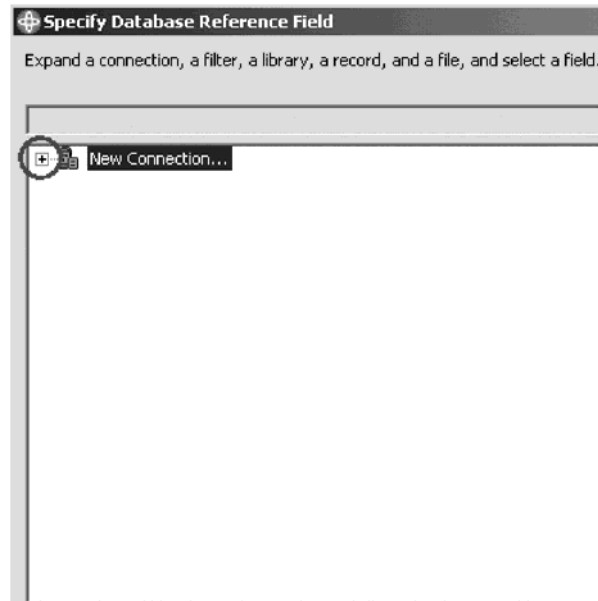
To add fields to the structure:

1. Click **Add Parameter**.



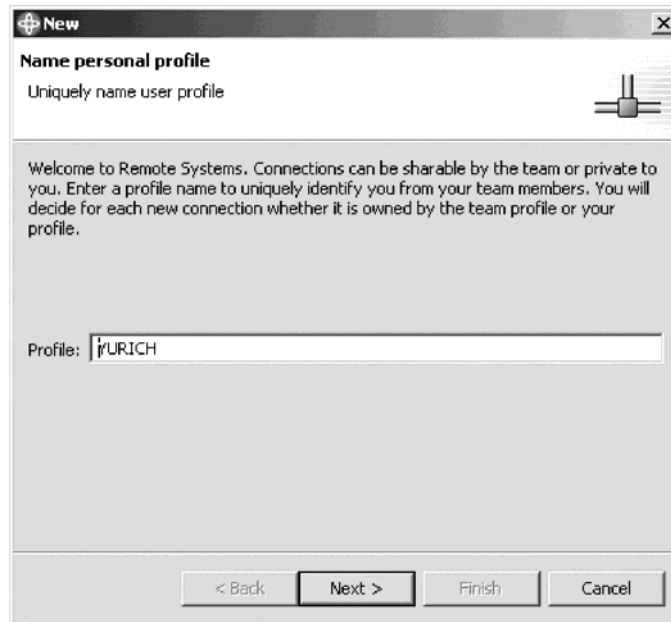
2. Click **Yes** if you are asked to save the changes.
3. Click **Specify**.

The Specify Database Reference Field window opens:



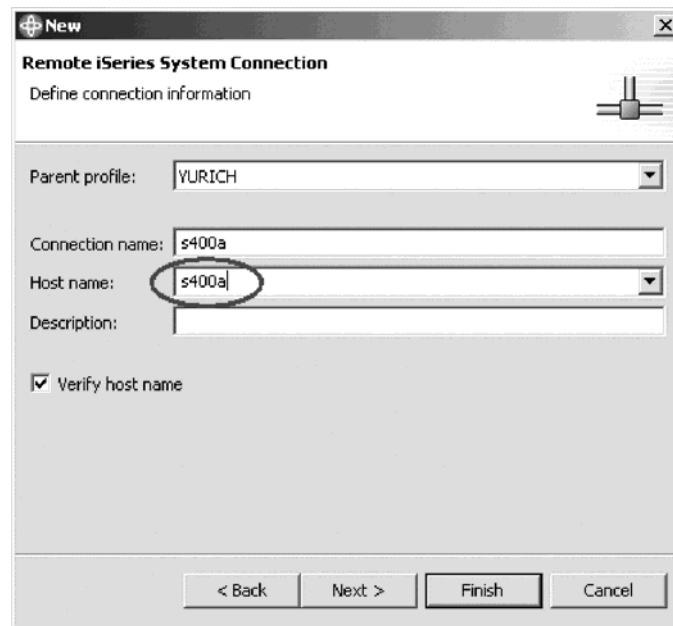
4. In the left pane of the Specify Database Reference Field window, expand **New Connection**.

The Name personal profile page opens:



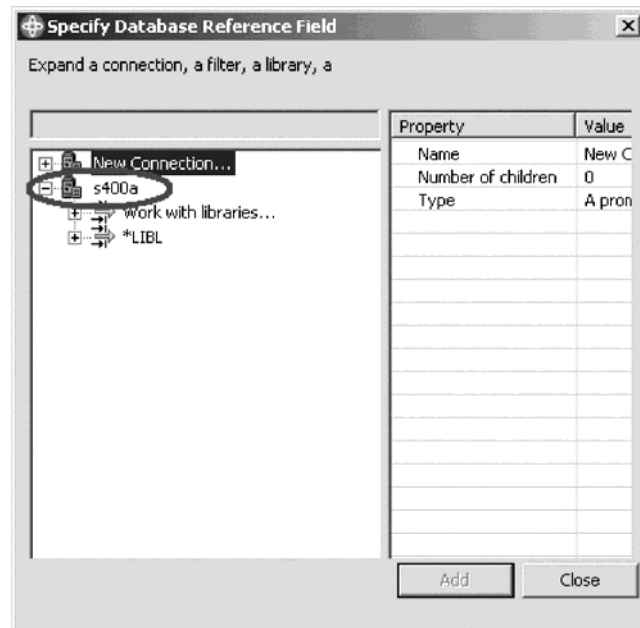
5. Click **Next** to accept the default profile.

The Remote iSeries System Connection page opens:



6. Leave the default value in the **Parent Profile** field.
7. In the **Host Name** field, type the iSeries server name, for example, s400a. (This is the same server name as you specified for the runtime.)
8. Click **Finish**.

You return to the Specify Database Reference Field window.

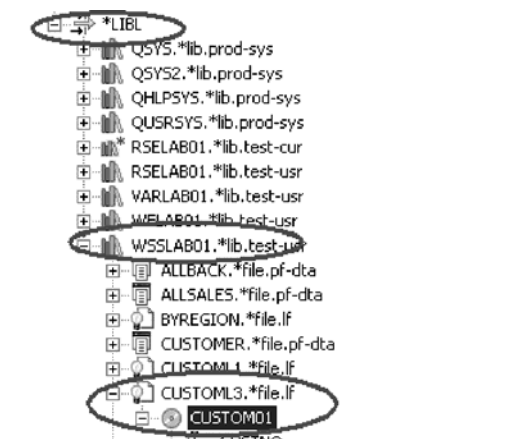


9. Expand your server, for example s400a.
10. Expand *LIBL..

11. Type your User ID and password, if prompted.

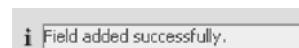


12. Select the check boxes to permanently store these two values then click **OK**.
13. Look for WSSLABxx.
14. If this library is not in the list of libraries, then expand Work with Libraries and type WSSLABXX in the **Library** field of the Work with Libraries dialog. Click **OK**.
15. Expand library WSSLABxx.



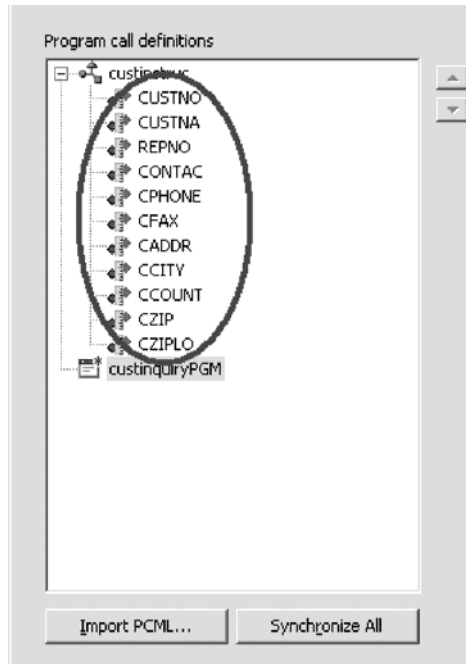
16. Expand the logical file CUSTOMML3.
17. Expand record format CUSTOM01.
18. Select the record format CUSTOM01.
19. Click **Add**.

You should see a message on the bottom of the dialog that fields have been added successfully.



20. Click **Close**.

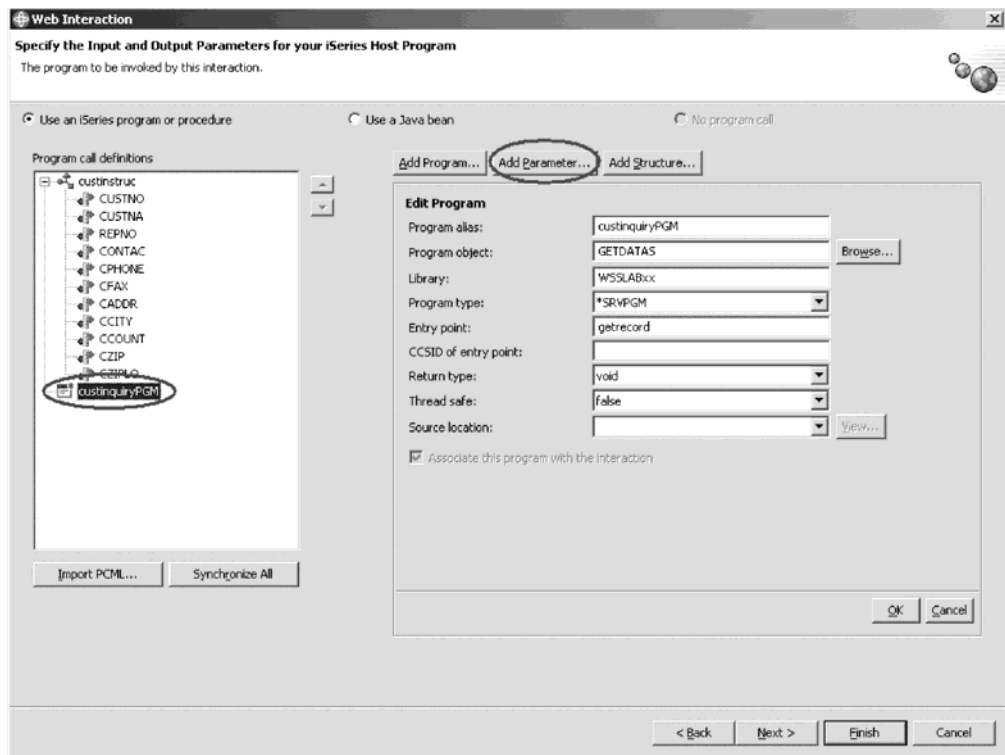
Parameters are added to the structure.



Exercise 4.7: Adding more parameters to the program definition

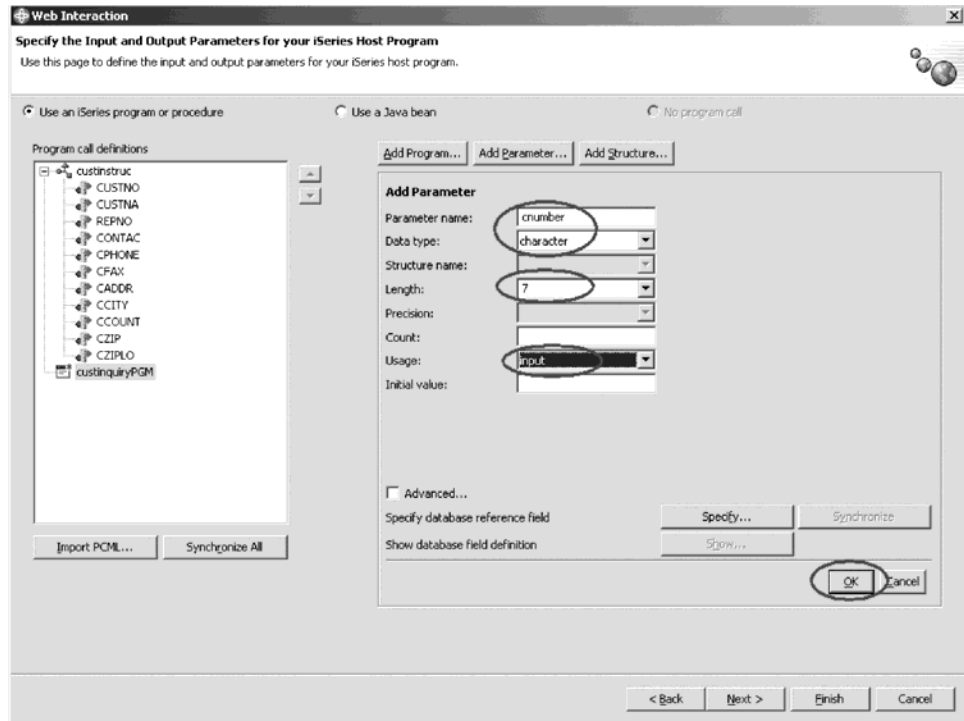
You continue to add more parameters to the program definition.

To add more parameters:



1. In the left pane of the Specify Input and Output Parameters for your iSeries Host Program page, select the `custinquiryPGM` program you added in Exercise 4.5.
2. Click **Add Parameter**.

Now enter the customer number as the input parameter:

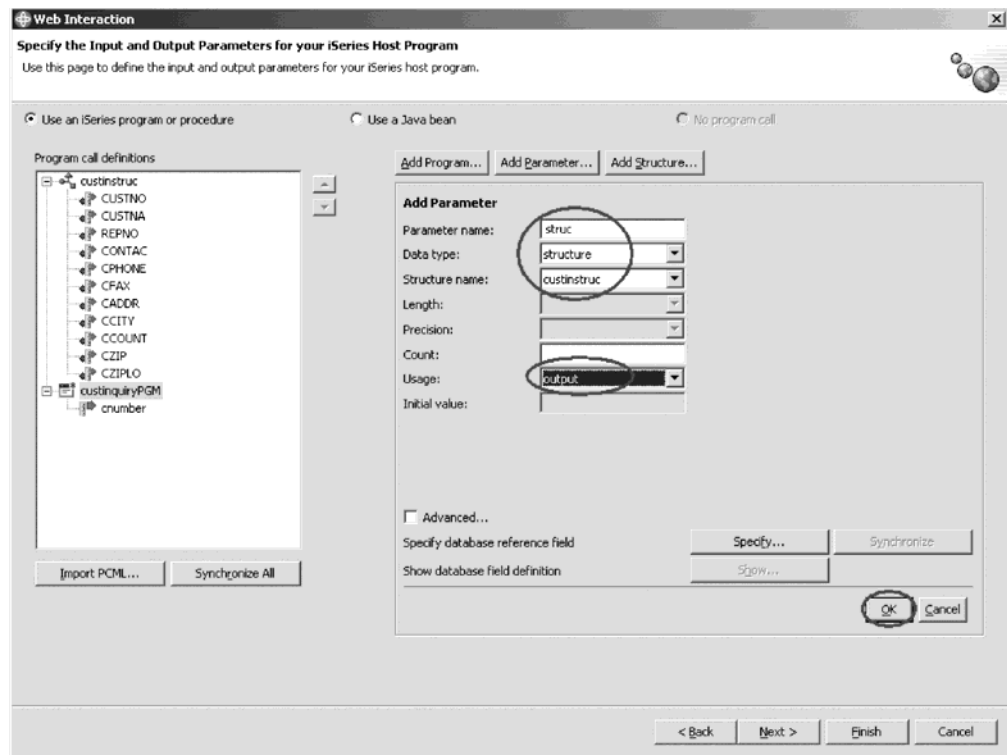


3. In the **Parameter name** field, type `cnumber`.
4. In the **Length** field, type 7.
5. In the **Usage** list, select **input**.
This is the Input parameter that will be sent from your Web page to the program.
6. Click **OK**.
7. If you are asked whether you want to save the changes, click **Yes**.
You have just defined the first parameter.

Exercise 4.8: Defining the structure as a parameter

Now you add the structure as a parameter to the program call definition.

To define the structure as a parameter:



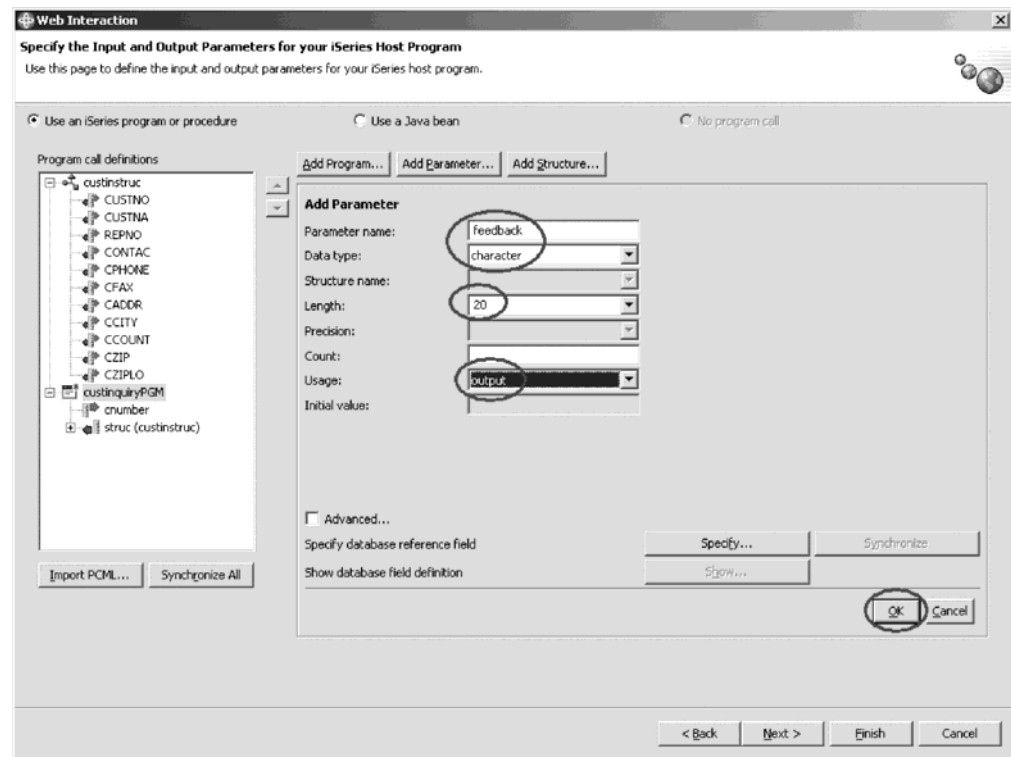
1. In the **Parameter name** field, type struc.
2. In the **Data type** list, select **structure**.
3. In the **Usage** list, select **output**.
4. Click **OK**.
5. If you are asked, whether you want to save the changes, click **Yes**.

Your structure is now an output parameter for the program call, which means it will pass data back from the program to the servlet running in the WebSphere Test Environment.

Exercise 4.9: Adding a feedback parameter

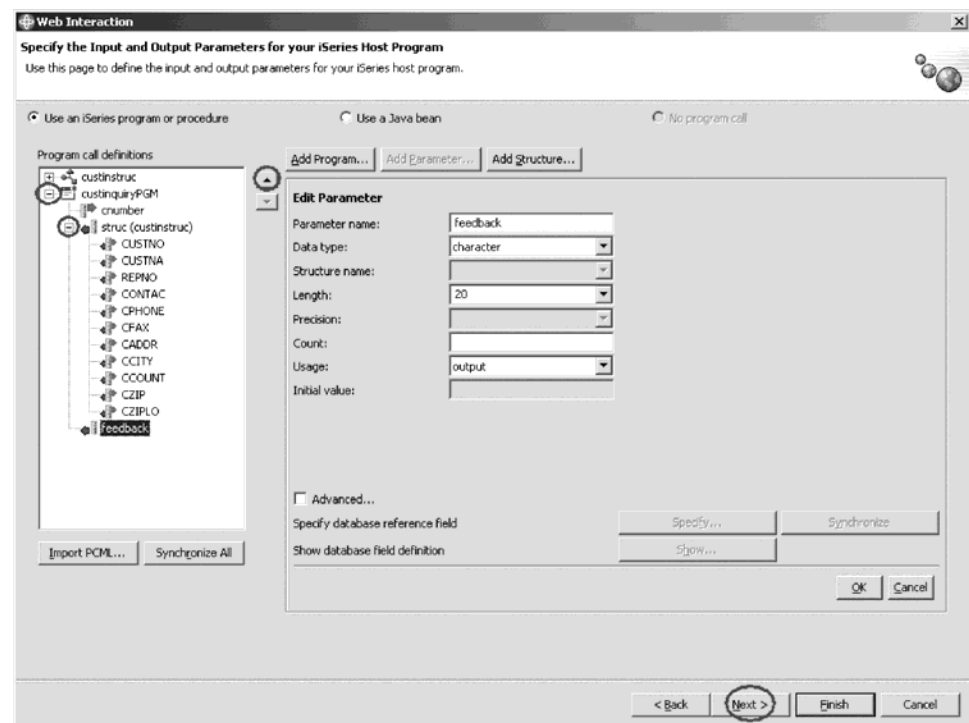
The program on the iSeries is written in a way that it will return a value to indicate whether the action (accessing a customer record) on iSeries was successful or not. If it was not successful the return value will contain a message number and a value for a substitution variable. You need to add a third parameter to the program call.

To add a feedback parameter:



1. In the **Parameter name** field, type feedback.
2. In the **Length** field, type 20.
3. Select **output** from the **Usage** list.
4. Click **OK**.

You have completed adding parameters.



In the left pane of the wizard, you can expand `custinquiryPGM` to see all the parameter definitions.

Tip: To move a parameter from its current position, use the arrows to the left of this pane.

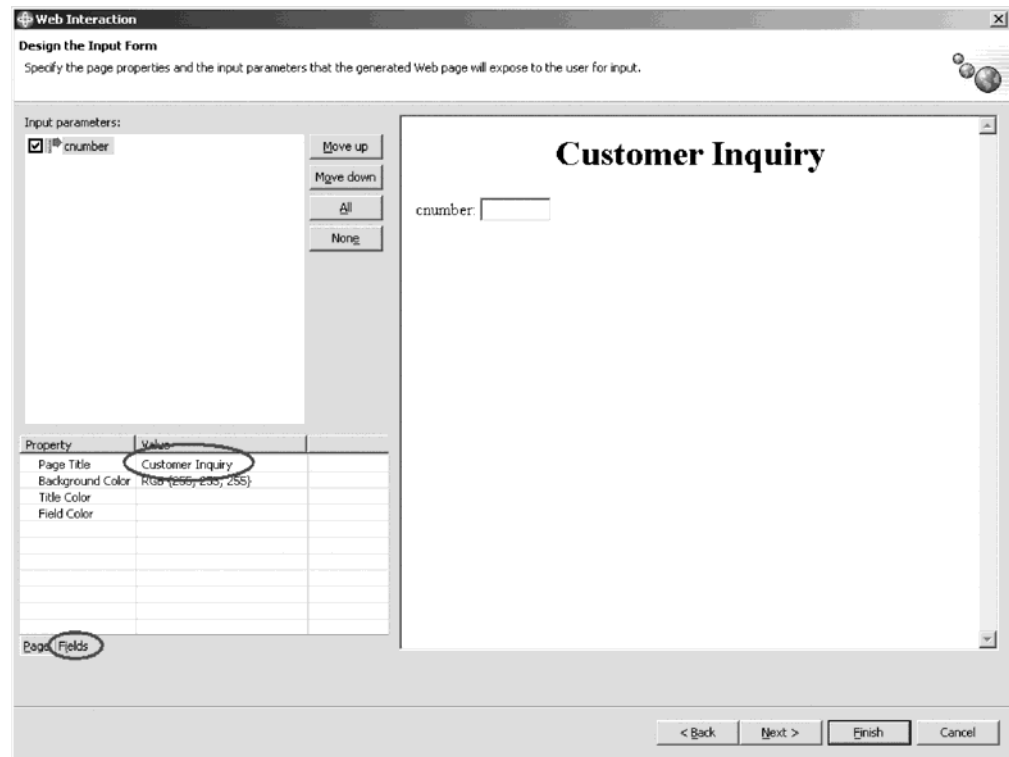
You are ready to go to the next page.

5. Click **Next**.

The interface between the servlet and the iSeries program is now defined.

Exercise 4.10: Defining the input page content

Now you need to specify which of the input parameters you specified you want to see on the input page for this interaction that gets generated by the wizard. This is easy in this application since you only have the customer number input field to handle. You also have the capability to enhance the page layout by changing colors of the page background and changing the heading. The Design the Input Form page contains a properties table that shows all input capable parameters, and a sample of the Web page that will be created.



To change the heading:

1. Click the field beside the **Page Title** property in the Property table and type Customer Inquiry.
2. Press **Enter** to apply this change to the sample page.
3. Click the **Fields** tab on the Properties table.

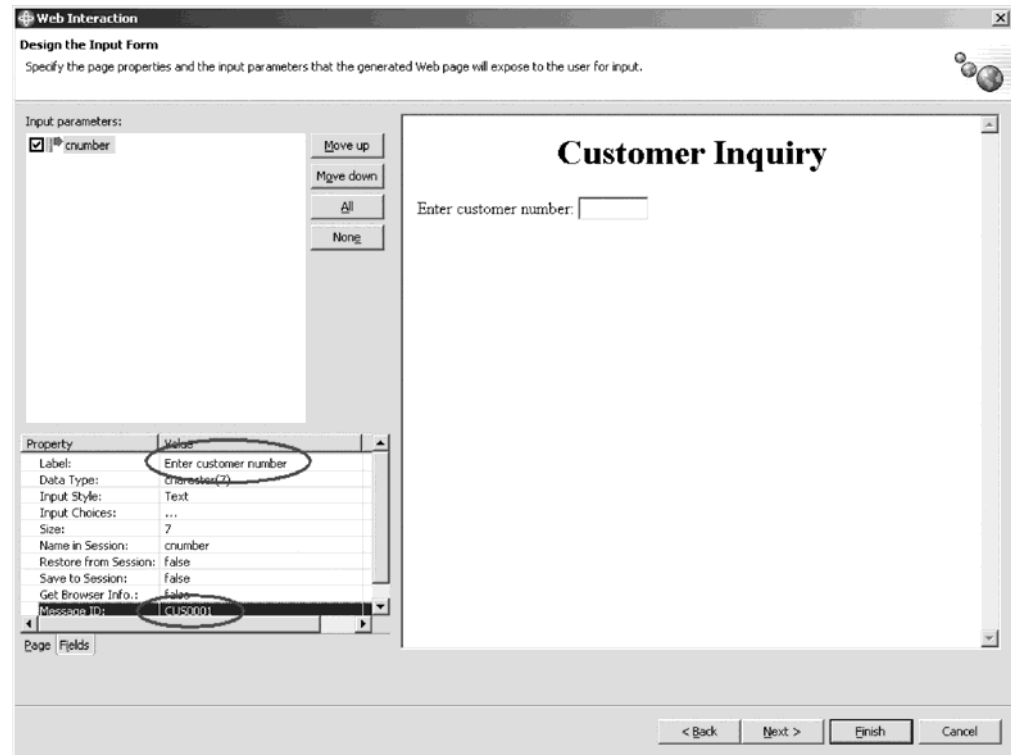
This opens the field properties for the selected field in the list.

You want to change the field label. You also want to change this application so it will be able to tell the user if there is a problem with the input data submitted. If the program fails to find a customer record for the customer number that was keyed into the input page, the program will send back a

message id in the feedback parameter. This error message text contains a substitution variable. You want to fill this substitution variable with the data from the cnumber input field. To enable the wizard to generate the code for this function you need to tell it that variable cnumber contains data for the substitution variable in the message.

You will make this change using the Properties table.

To change a field label:



1. In the left pane of the Design the Input Form page, select **cnumber**.
2. Click the field beside the **Label** property, and type Enter customer number.
3. Click the field beside the **Message ID** property, and type CUS0001.
4. Press **Enter**.

Note: Make sure to press **Enter** after making this entry.

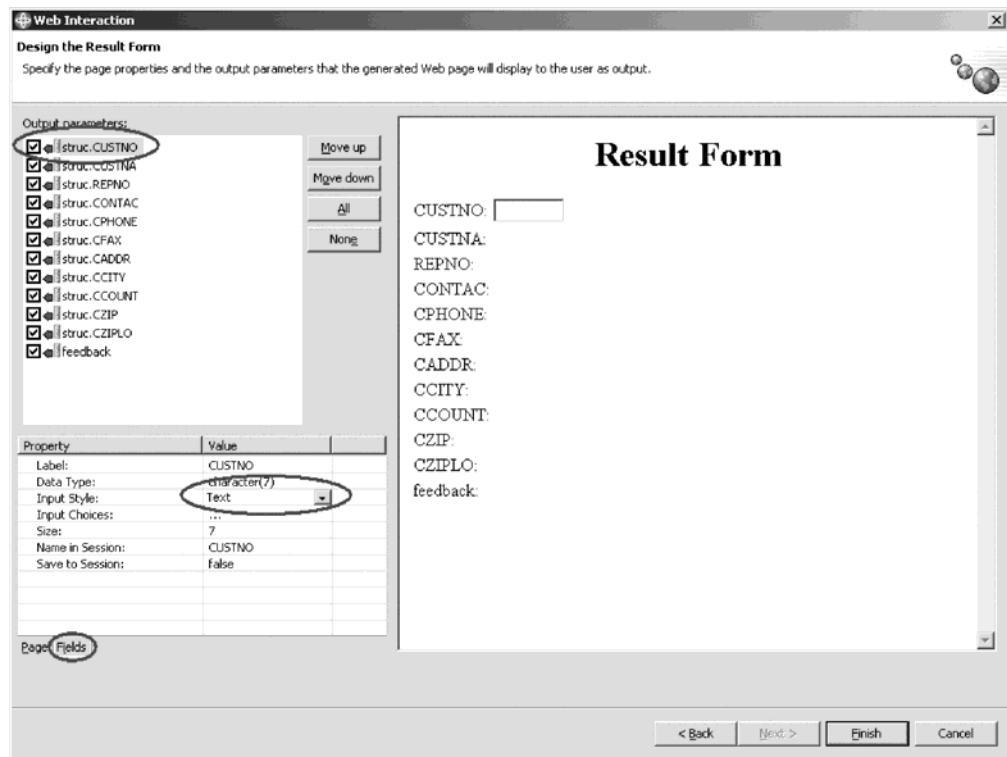
CUS0001 is the message id that contains a substitution variable that will be filled with the content of cnumber in case of an exception during the database retrieval.

5. Click **Next** to advance to the next wizard page.

Exercise 4.11: Defining the output page content

Now you can specify what data should be displayed on the Output Web page generated for you. In the right pane of the wizard you see a sample of the page that will be generated. First, change the Input style of the output parameter struc.CUSTNO. This will enable you to use customer number as an input parameter to future features of your Web project.

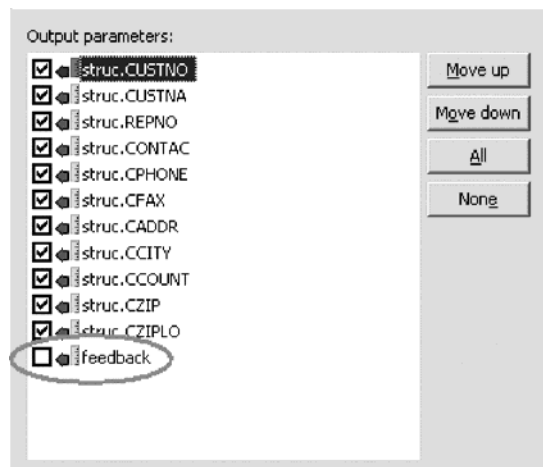
To define the output page:



1. Select struc.CUSTNO.
2. Click the **Fields** tab.
3. In the Properties table from the **Input Style** list, select **Text**.

By default all output parameters will be added to the Web page, but you don't want the data in the feedback parameter to be displayed.

Let's go through the steps to erase the feedback parameter from the output page.

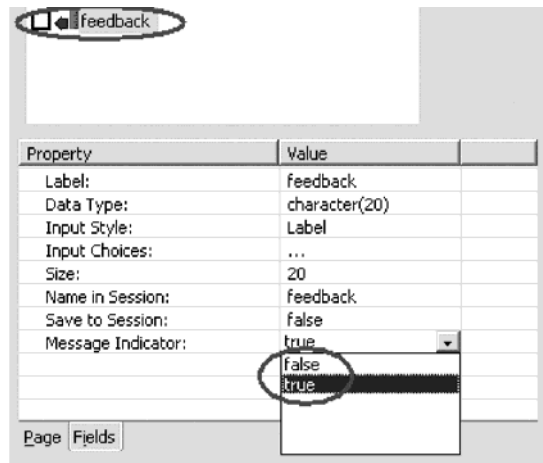


4. In the left pane of the Design the Result Form, clear the **feedback** check box. This will remove it from the Web page. You will stay on this page to specify the error handling.

Exercise 4.12: Specifying error handling

Now you can specify what should happen in case a customer number didn't match the customer numbers in your database.

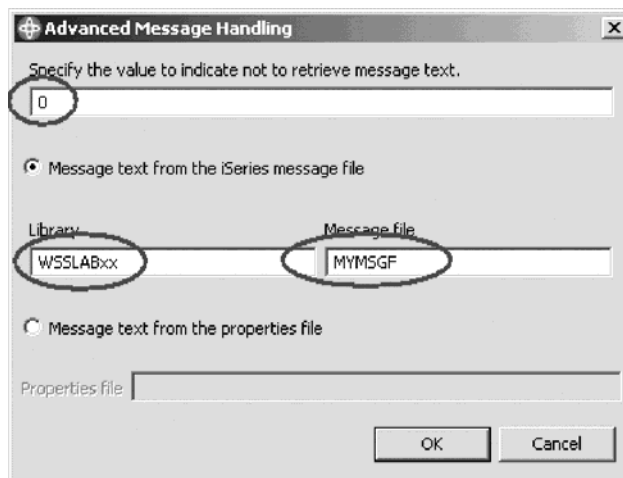
1. In the left pane of the Design the Result Form, select the **feedback** variable. You will see the properties for this variable in the Properties table.



2. Locate the **Message Indicator** heading in this Properties table.
3. Click the field beside the **Message Indicator** and set it to **true**.

This just tells the wizard that this parameter will contain an indication whether the program call was successful or not.

The Advanced Message Handling dialog opens:



4. In the **Specify the value to indicate not to retrieve message text**, type 0.
5. Leave the **Message text from the iSeries message file** radio button selected.
6. In the **Library** field, type WSSLABxx.
7. In the **Message file** field, type MYMSGF.
8. Click **OK**.

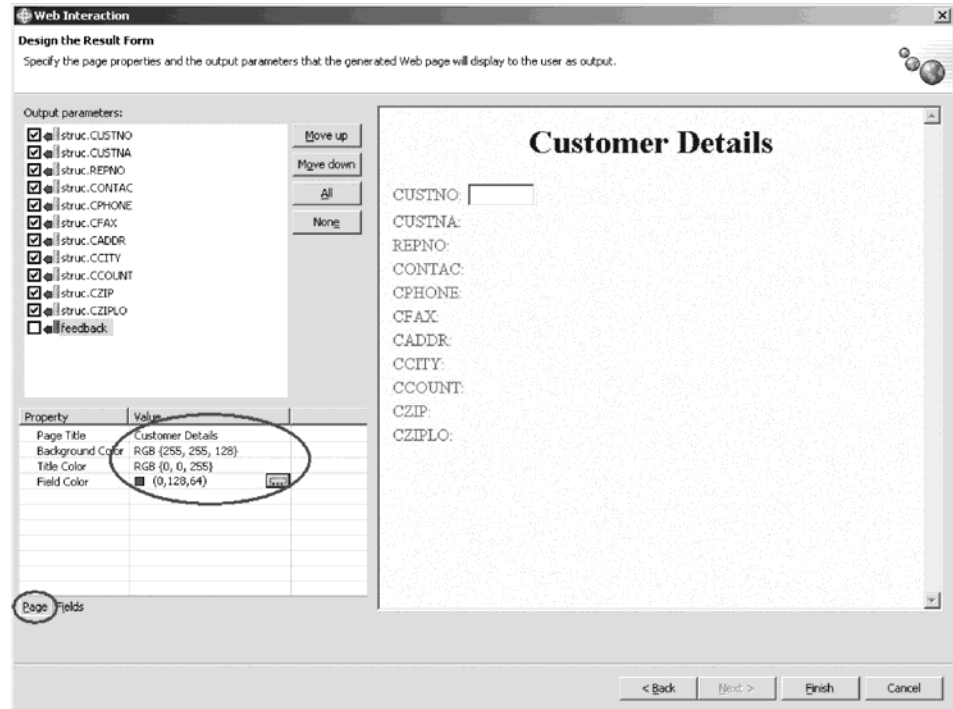
The message handling is defined.

Exercise 4.13: Enhancing the output page

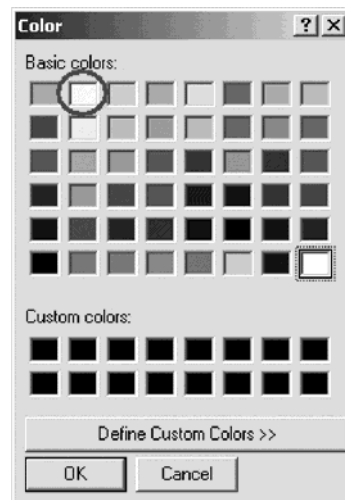
Next you enhance the look of the Web page.

To enhance the output page:

1. Click the **Page** tab in the Properties table.

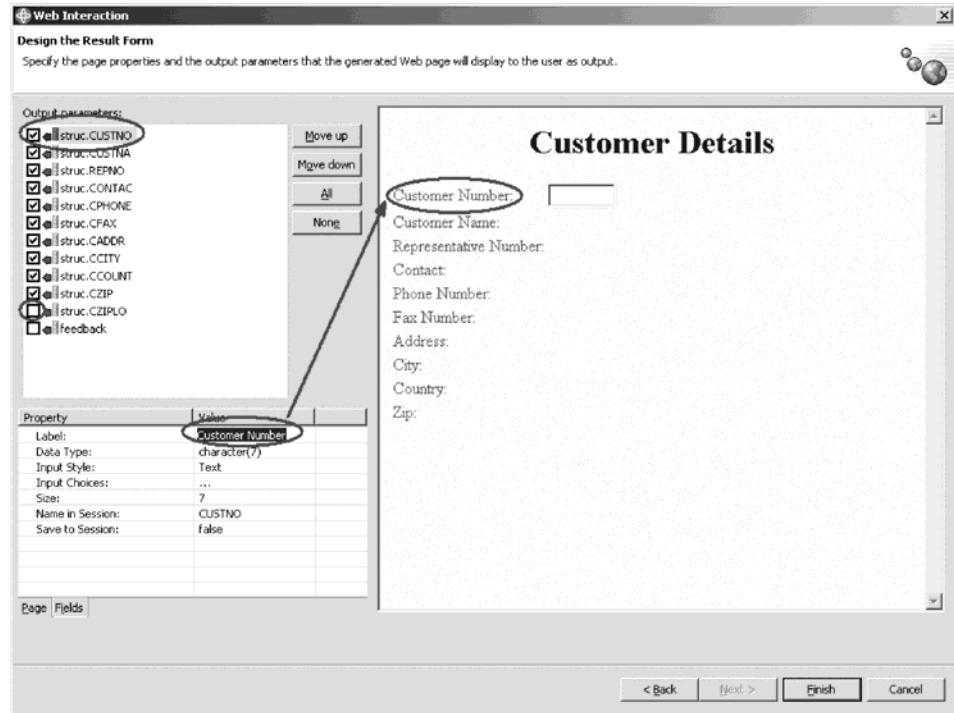


2. Click the field beside the **Page title** and type Customer Details.
3. Click the field beside the **Background Color** and click the selection button on the right side of the field.
The Color palette opens.
4. Select a **light yellow** from the Color palette.



5. Click **OK** on the Color palette.
6. Click the field beside the **Title Color** and select **Blue** from the Color palette.
7. Click **OK** on the Color palette.

8. Click the field beside the **Field Color** and select **Dark Green** from the Color palette.
9. Click **OK** on the Color palette.
 You notice the sample page gets updated as you choose different properties. If you don't like the suggested color choices, go ahead and create your own design.
 You need to modify the labels of the fields that show up on the output JSP.
10. Click the **Fields** tab.
11. Select each field under **Output parameters**, and for each one, change the **Label** value to the values described in the table below:



CUSTNO	Customer Number
CUSTNA	Customer Name
REPNO	Representative Number
CONTAC	Contact
CPHONE	Phone Number
CFAX	Fax Number
CADDR	Address
CCITY	City
CCOUNT	Country
CZIP	Zip

12. In the left pane of the Design the Result Form page, clear the struc.CZIPLO check box. You don't need to display this variable.
13. Click **Finish** on the Design the Result Form page.
 The Interaction wizard creates the necessary files for your application:

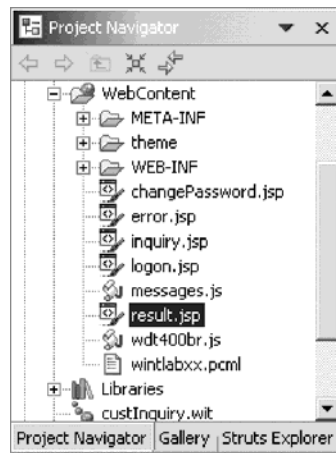
- The file custInquiry.wit is created in the project. This file reflects the data you entered on the pages of the Web Interaction wizard for the interaction you identified as cust. To modify these settings, double-click on the file to open the Web Interaction wizard.
- The default package under the Java Source folder contains the .java files generated by the wizard.
- The classes folder under WebContent/WEB-INF contains the .class files that were generated from the .java files that were created by the wizard.

Exercise 4.15: Ensuring customer number field cannot be modified

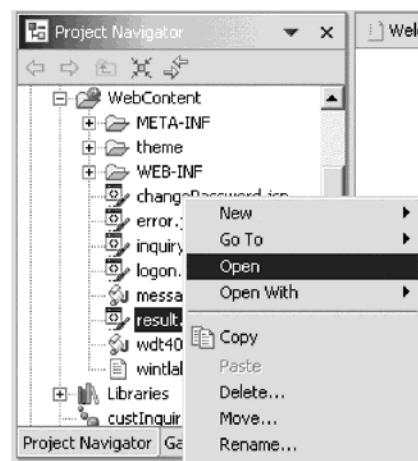
Let's make sure that the Customer Number field on your results.jsp page cannot be modified.

To ensure the customer number field cannot be modified:

1. In the Project Navigator, expand your project then WebContent and locate result.jsp.

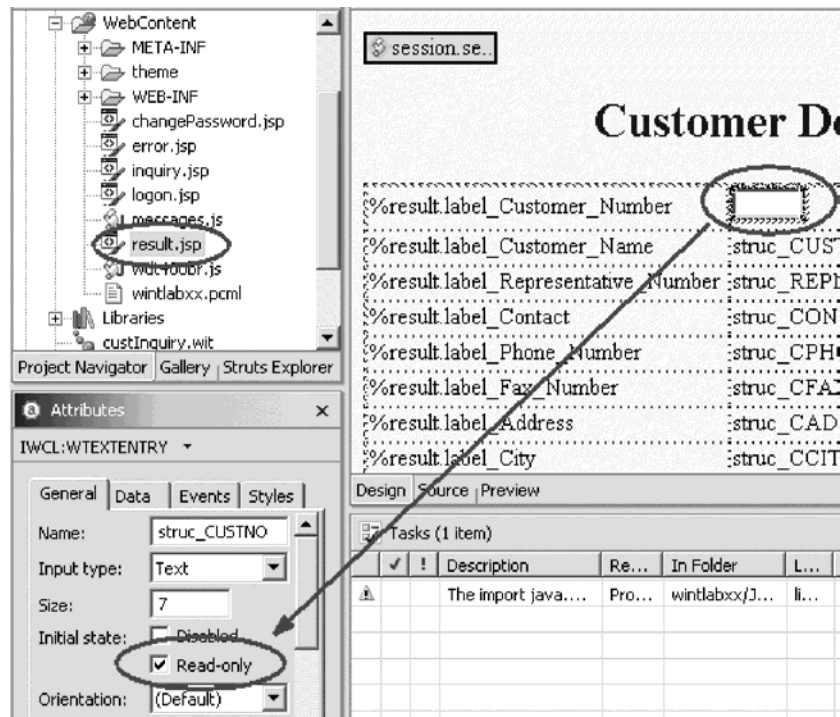


2. Right-click result.jsp.



3. Click **Open** on the pop-up menu.

This opens the result.jsp page on the Page Designer.



4. Select the **Text field of Customer Number** in Page Designer.
Notice the **Attributes** view of that Text field appears in Page Designer.
5. Select the **Read-only** check box.
This prevents any modifications to the Customer Number.
6. Close Page Designer. Click **Yes** to save the changes.

Recap

Congratulations! You have completed Module 4: Creating a Web interaction. You should know understand:

- The purpose of an interaction.
- How to invoke a Web interaction.
- How to specify input and output pages.
- How to define a program invocation and parameters.
- How to invoke a service program.
- How to define the program structure.
- How to add reference fields from DB2/400 to the program structure.
- How to add a structure as a parameter.
- How to add a feedback parameter.
- How to define the input page content.
- How to define the output page content.
- How to specify error handling.
- How to enhance the output page
- How to ensure the customer number field cannot be modified

Before you can test your Web application you have to create the iSeries program containing the business logic to return data to your result page. You can write the

logic with SEU if you wish. The next module guides you through using the Remote System Explorer perspective to edit and compile your iSeries program. Continue to Module 5: Adding the iSeries program to your Web application.

Chapter 5. Module 5: Adding the iSeries program to your Web application

In this module, you will learn how to add the iSeries program to your Web application using the Remote System Explorer perspective. Remote System Explorer provides a PDM like environment but with a GUI interface, and full integration with all iSeries programmer tools in the Development Studio Client workbench.

In order to add the iSeries program to your Web application, there are several steps you will need to learn about and follow:

- Understanding LPEX editor features
- Opening the RSE perspective
- Finding an iSeries source member
- Editing an iSeries source member
- Compiling an object.

Note: To access iSeries data and resources you will need to have a job description setup and associated with your user profile on your iSeries system.

In order to accomplish these learning objectives, there are several steps that are involved, including:

- Exercise 5.1: Opening the RSE perspective
- Exercise 5.2: Finding an iSeries source member in the RSE
- Exercise 5.3: Editing an iSeries source member in the RSE
- Exercise 5.4: Reviewing some LPEX Editor features
- Exercise 5.5: Creating a PGM object with RPGIV
- Exercise 5.6: Creating a SRVPGM object with RPGIV

The exercises in this module must be completed in order. Start with Exercise 5.1 when you are ready to begin.

Length of time:

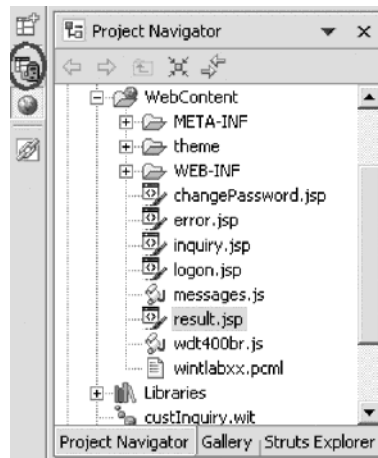
This module will take approximately 15 minutes to complete.

Exercise 5.1: Opening the Remote System Explorer perspective

To open the Remote System Explorer perspective:

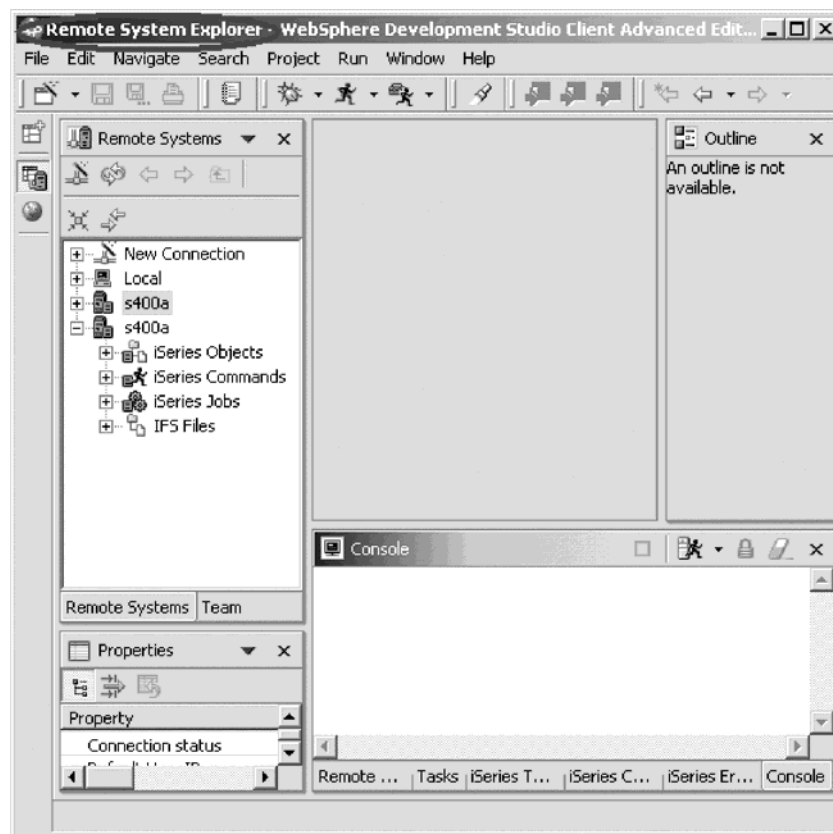
1. In the workbench, check whether you have the Remote System Explorer perspective already open. Look for the icon in the left taskbar of the workbench.

2. If you see it click the  RSE icon.



3. Otherwise, select **Window > Open Perspective > Remote System Explorer** on the workbench menu.

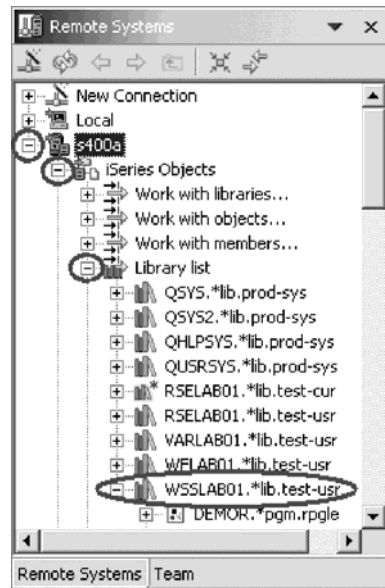
The Remote System Explorer perspective opens.



Exercise 5.2: Finding an iSeries source member

You will see a list of servers in the Remote Systems view. Now you locate the source you want to edit.

To find an iSeries source member:



1. Expand your iSeries server. Use the same server that you have used to specify the reference fields in the data structure.
2. Expand iSeries Objects.
3. Expand Library list. If prompted, type your user ID and password.
4. Look for WSSLABxx. If you do not see this library, then right-click Library list and click **Add Library List Entry** on the pop-up menu. Type WSSLABXX in the **Additional Library** field and click **OK**.
5. Expand the library WSSLABxx.
You will see all objects in this library.
6. Expand the source file QRPGLESRC.
7. Scroll down to the source members in the expanded list.

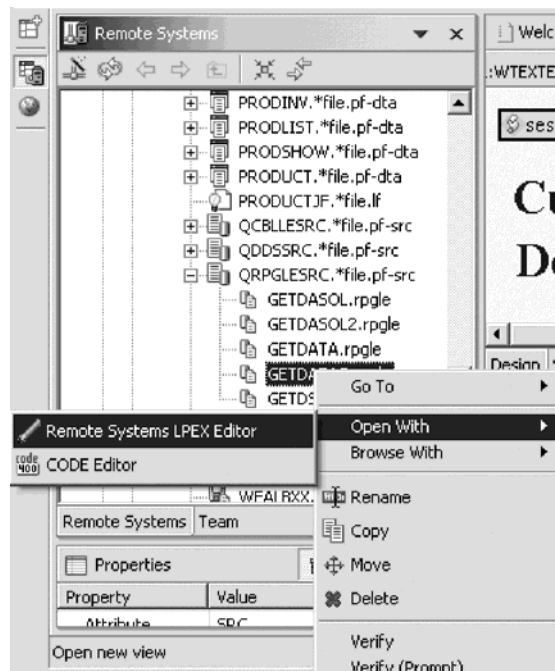
Exercise 5.3: Editing a source member

Two shell source members are prepared for you to use. Depending on whether you specified that you want to work with a program, or a procedure in a service program, you need to select a different member.

To edit a source member:

1. For a program call, right-click member **GETDATA**.
2. For a service program call, right-click member **GETDATAS**.

3. Select **Open with > Remote Systems LPEX Editor** on the pop-up menu.



The Editor window opens and you are ready to write your program.

Tip: You can also double-click on a member to bring up the Remote Systems LPEX editor.

For the two different choices provided, you need to go to the correct exercise:

- Write a regular RPG program — Exercise 5.5: Creating the *PGM object using RPG
- Write an RPG Service Program — Exercise 5.6: Creating the *SRVPGM using RPG

Before you do that, if you have not worked with the Remote Systems LPEX editor you might want to go through Exercise 5.4: Reviewing Remote Systems LPEX Editor Features.

Exercise 5.4: Reviewing Remote System LPEX Editor features

Before you start, here are some useful tips on using the LPEX editor. Let's look at some of the features of the LPEX editor, so you can later easily find your way around and use them:

- **ALT+L** marks a line.
- **ALT+U** unmarks selection
- **ALT+D** deletes a selected text
- **ALT+C** copies selected text
- **ALT+M** moves selected text
- Press **Enter** to insert a new line

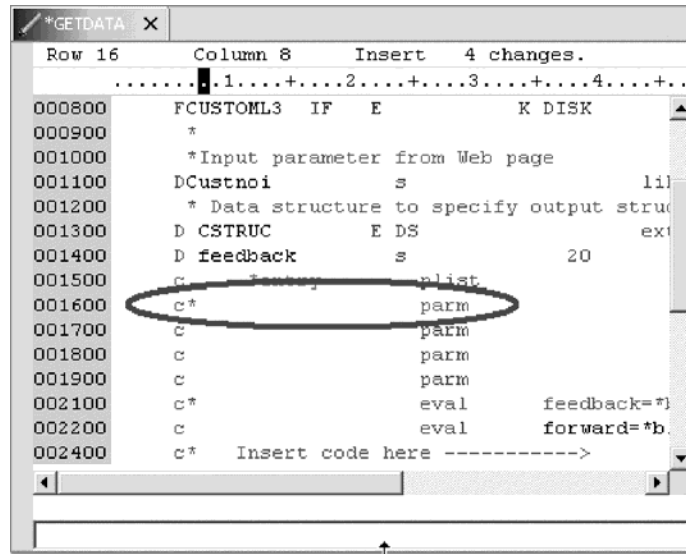
Highlighting specification fields

The LPEX editor highlights the Tokens (specification fields) of your RPG program source, providing a separate color for each, improving readability. When you make changes to a line, the token colors get updated only after you move your cursor off the line.

To see how token highlighting works:

1. Move the cursor to a Calculation statement.
2. Position the cursor to column 7 (right next to the 'C').
3. Type an asterisk (*).
4. Move the cursor off the line and watch what happens.

The line where the asterisk (*) was typed has become a comment line and its color changes accordingly.



5. Move the cursor back to column 7 and remove the asterisk (*).
6. Move the cursor off that line of code and the statement is tokenized.
The token highlighting changes to reflect that this is a non-commented 'C' specification.

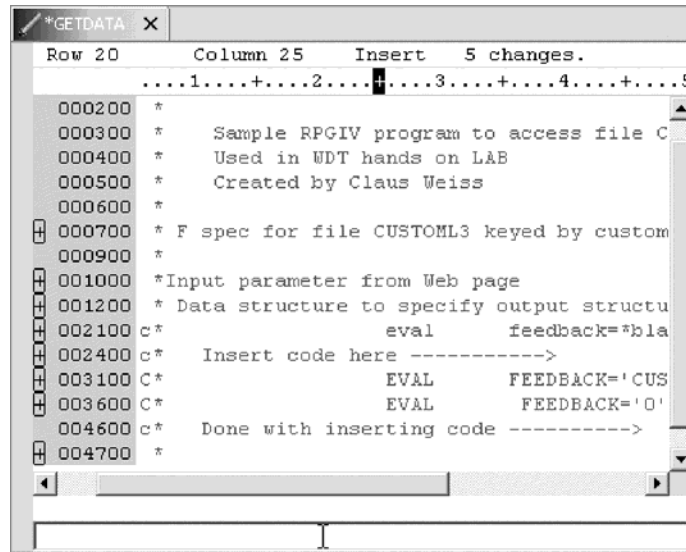
Displaying types of lines

Using the LPEX editor, it is possible to have only particular types of source lines displayed at a time.

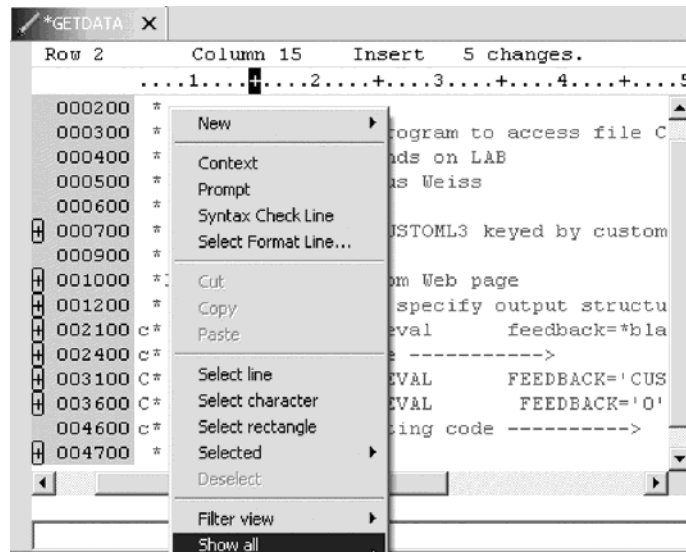
To display types of lines:

1. Right-click anywhere in the editor window.
2. Click **Filter view** on the pop-up menu.
The menu items list the types of line selections that can be made.
3. Click Comments.

The Editor window now contains only those RPG statements that are comments.



4. Right-click anywhere in the editor window.
5. Click **Show all** on the pop-up menu.



All statements types are displayed. In addition, the choices in the popup menu can be used to include only control specifications, user subroutines, and procedure and others. The Filter Selection option under the Selected option in the Edit menu pull down will allow the selection of only those lines containing a particular selected character string.

Checking the syntax of a file

Syntax checking is available for RPG code, and by default will be active. The syntax of the RPG code is checked automatically when a change is made to a line, and the cursor is moved off the line.

When errors are found, they are displayed following the statement with the error.

Keeping track of columns in a specification line

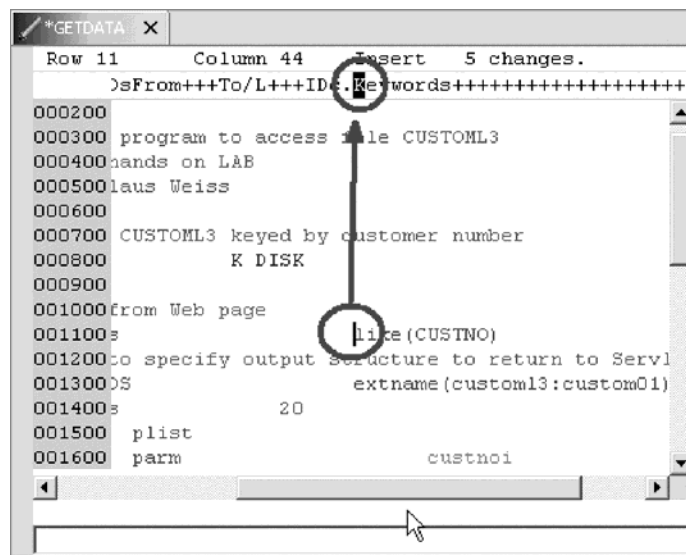
The format line is at the top of the editor window, just above the first statement. A format line is used to help keep track of the columns in a particular specification line. The content of the format line can vary to reflect the particular type of specification being keyed such as F specs, C specs, D specs and so on.

To display a format line:

1. Clicking a line or use the arrow keys and click the left mouse button to make a line active.

The format line gets updated as a line gets focus.

You can move the cursor right or left with the arrow keys to go from character to character, or with the **Tab** key to go from field to field. An indicator on the format line moves with the cursor to show in which column the cursor is positioned.



You can select a format line for any specification you want.

2. Click **Source** from the Editor menu.
3. Click **Select Format Line** on the pop-up menu.

The Format Line Selection window opens.



Now you can start and create the RPG program or service program.

4. Click **OK**.

Exercise 5.5: Creating a *PGM object with RPG

To make it easier for you, most of the RPG source is already prepared for you. You have already opened the correct source member GETDATA in the previous exercise using the Remote Systems view.

Read through the source. It contains an F spec to access file CUSTOML3, which contains the customer data keyed by customer number. The D specs defines the data structure CSTRUC that you pass back to your Web page and the CUSTNOI variable that gets passed from the Web page to this program. As well the Feedback variable is defined as a 20 length character field.

The Entry parameter list is defined as:

- CUSTNOI field (this is the input parameter).
- CSTRUC structure (this the data structure for the customer output data).
- Feedback field to indicate no success for file access.

The code you have to write here, fetches a customer record by chaining to the file with the CUSTNOI that gets passed into the program.


To create a *PGM object with RPG:

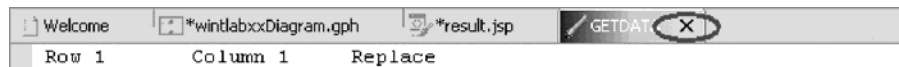
1. Add the code to get the customer record and check whether the chain was successful.

```

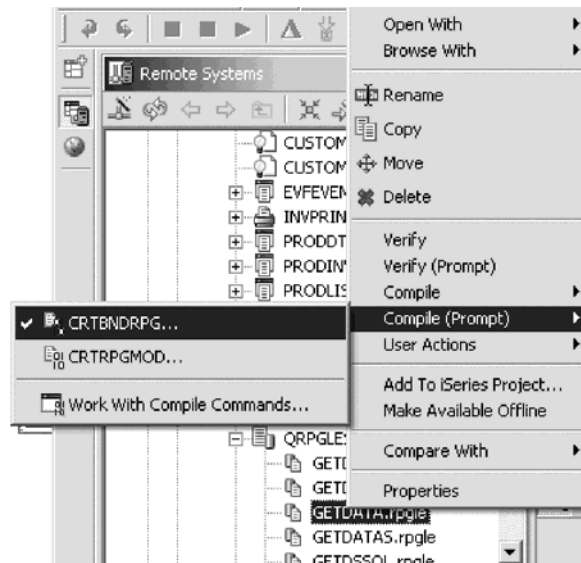
CL0N01Factor1+++++0pcode(E)+Factor2+++++Result+++++Len++D+HiLoEq
C      CUSTNOI      CHAIN      CUSTOML3                      5050
C
C
C              IF      *IN50
C              EVAL    FEEDBACK='CUS0001 ' + CUSTNOI
C              ELSE
C              EVAL    FEEDBACK='0'
C              ENDIF
C

```

2. Your coding is complete, so let's create the program.
3. Click the Save  icon on the workbench toolbar to save the member.
4. Click the X in the Editor window title bar to close the member.

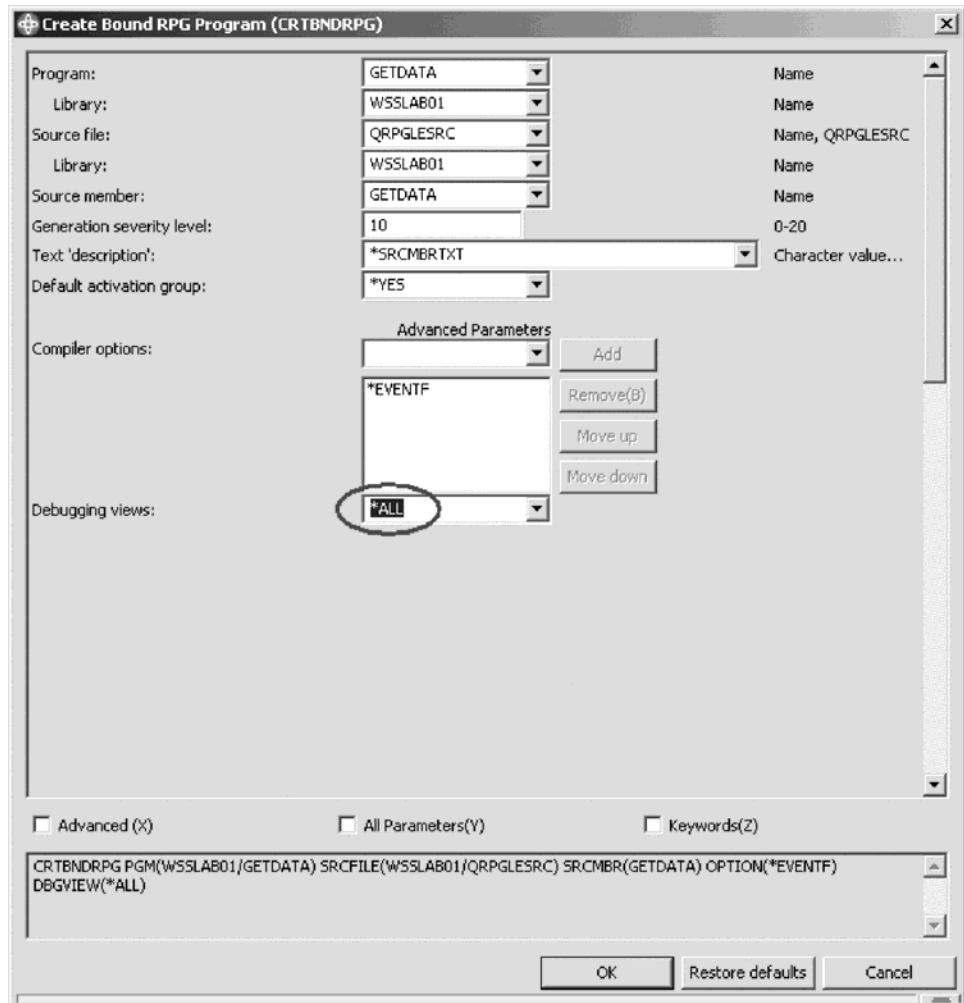


Be careful not to click the X on the workbench window title bar.



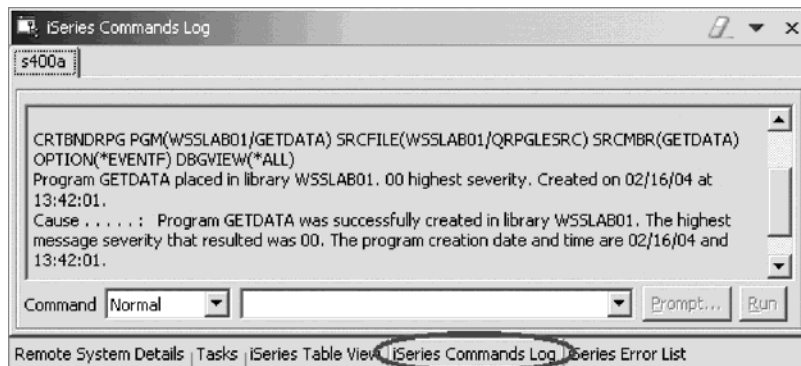
5. In the Remote Systems view, right-click the GETDATA member in source file QRPGLSRC and click **Compile (Prompt) > CRTBNDRPG** on the pop-up menu.

The Create Bound RPG Program (CRTBNDRPG) window opens:



6. In the **Debugging views** list, select ***ALL**
7. Click **OK** to start to compile.

After compile, the error list view is shown, listing all compile errors. You may see only information and warning messages which means your compile was completed and the program object was created.



Click the **iSeries Commands Log** tab to check that the compile was successful. This log shows the Remote System Explorer job messages.

Now you are ready to run your Web application. Go to Module 6: Running the Web Application and skip Exercise 5.6: Creating a *SRVPGM with RPGIV.

If you get a message that there are errors in your program, go through the edit compile fix cycle.

Fixing errors

In the error list view, check the errors:

1. Double-click the error.
This positions the cursor in the Editor window on the statement that is wrong.
2. Fix all errors.
3. Save the source member.
4. Go back to the Remote Systems view.
5. Right-click the member to run the CRTBNDRPG command.
6. Continue this cycle until you get a clean compile then go to Module 6: Running the Web Application.

Exercise 5.6: Creating *SRVPGM with RPG

To make it easier for you most of the RPG source is already prepared for you. You have already opened the correct source member GETDATAS in the previous exercise using the Remote Systems view. Read through the source. It contains an F spec to access file CUSTOML3 which contains the customer data keyed by customer number. The D specs define the data structure CSTRUC that gets passed back to your Web page and the CUSTNOI variable that gets passed from the Web page to this program. Also variable FEEDBACK is defined as 20 characters, to return message id and customer number in case the customer record was not found.

The prototype for procedure getrecord with the parameters is defined as the:

- CUSTNOI field (this is the input parameter)
- CSTRUC structure (this is the data structure for customer data)
- FEEDBACK field (returns error or success)

Note: The external name is case sensitive. It has to match exactly what you specified in the Web Interaction wizard.

The procedure interface defines the parameters in the same way as the prototype. Inside the procedure after the procedure interface you need to add the RPG code.


To add the RPG code:

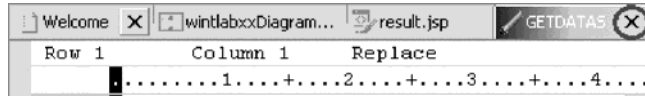
1.

```
CL0N01Factor1+++++++0opcode(E)+Factor2+++++++Result+++++++Len++D+HiLoEq
C      CUSTNOI      CHAIN      CUSTOML3      5050
C
C                  IF      not *IN50
C                  EVAL      cust1=cstruc
C                  EVAL      FEEDBACK='0'
C                  ELSE
C                  EVAL      FEEDBACK='CUS0001 '+ CUSTNOI
C                  ENDIF
```
2. You do a CHAIN to the database and move the database fields to the local procedure structure. If the database access fails, move the MSGID and Customer number to the FEEDBACK variable.

Your coding is complete, so let's create the service program module.

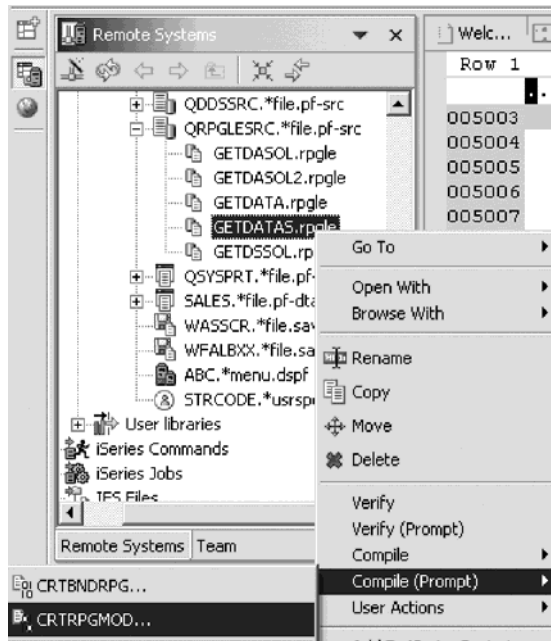
To create the service program module:

1. Click the Save  icon on the workbench toolbar to save the member.
2. Click X in the Editor window title bar to close the member.

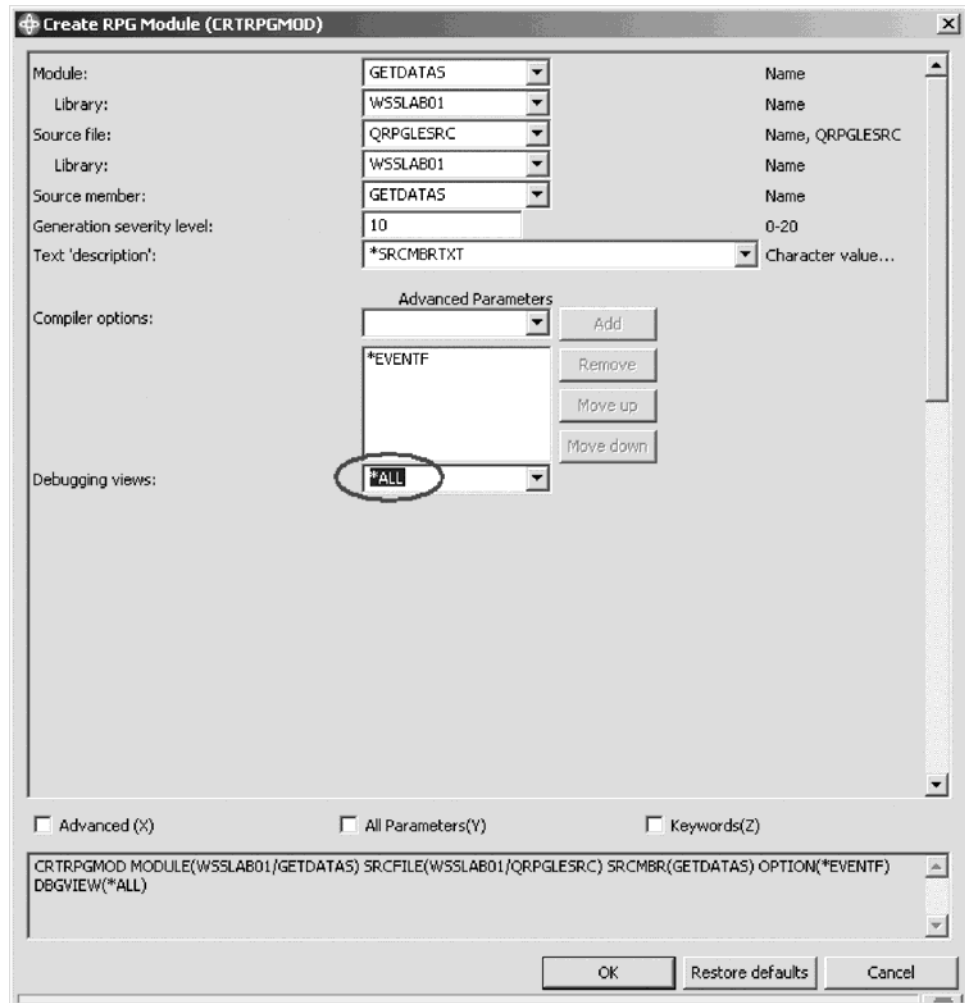


Be careful, don't click the X on the workbench window title bar.

3. In the Remote Systems view, right-click GETDATAS member in QRPGLSRC and select **Compile (Prompt) > CRTRPGMOD** on the pop-up menu.

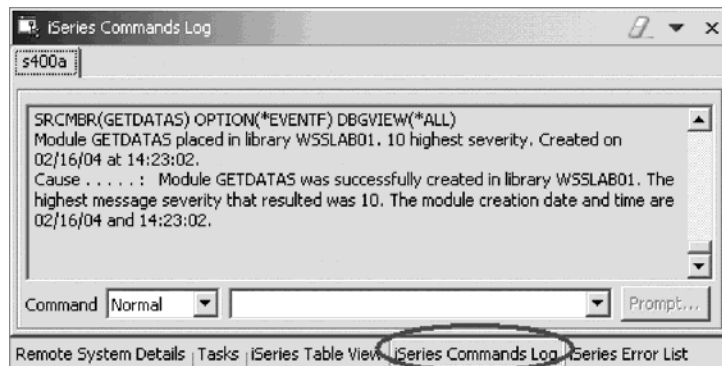


The Create RPG Module (CRTRPGMOD) opens:



4. In the **Debugging views** list, select ***ALL**
5. Click **OK** to submit the command.

After the compile the error list view is shown, listing all compile errors. You may only see information and warning messages that means your compile was completed and the program module was created.



Click the **iSeries Commands Log** tab to check if the compile was successful. This log shows the Remote System Explorer job messages. Now you are ready

to create a Service Program. Go to the section Creating a service program. If you get a message that there have been errors found in your program, go through the edit compile fix cycle.

Fixing errors

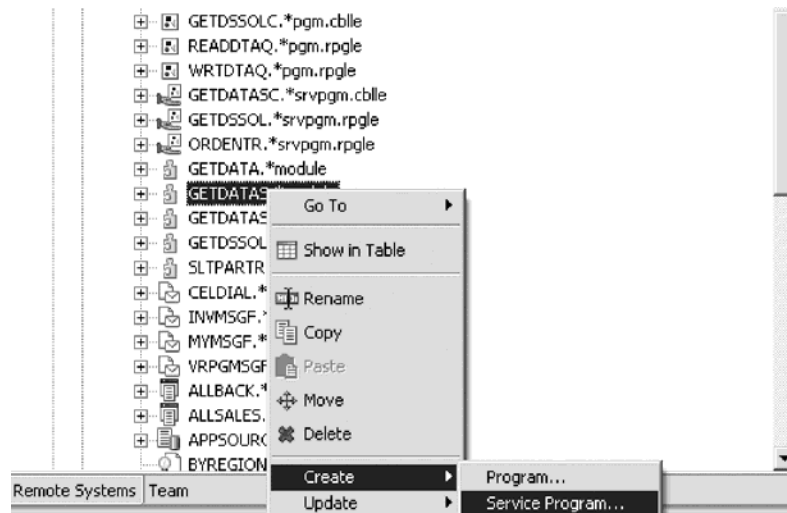
In the error list view, check the errors:

1. Double-click on the error.
This positions the cursor in the Editor window on the statement that is wrong.
2. Fix all errors.
3. Save the source member.
4. Go back to the Remote Systems view.
5. Right-click the member to run the CRTRPGMOD command.
6. Continue this cycle until you get a clean compile then proceed to create the service program.

Creating a service program

To create a service program:

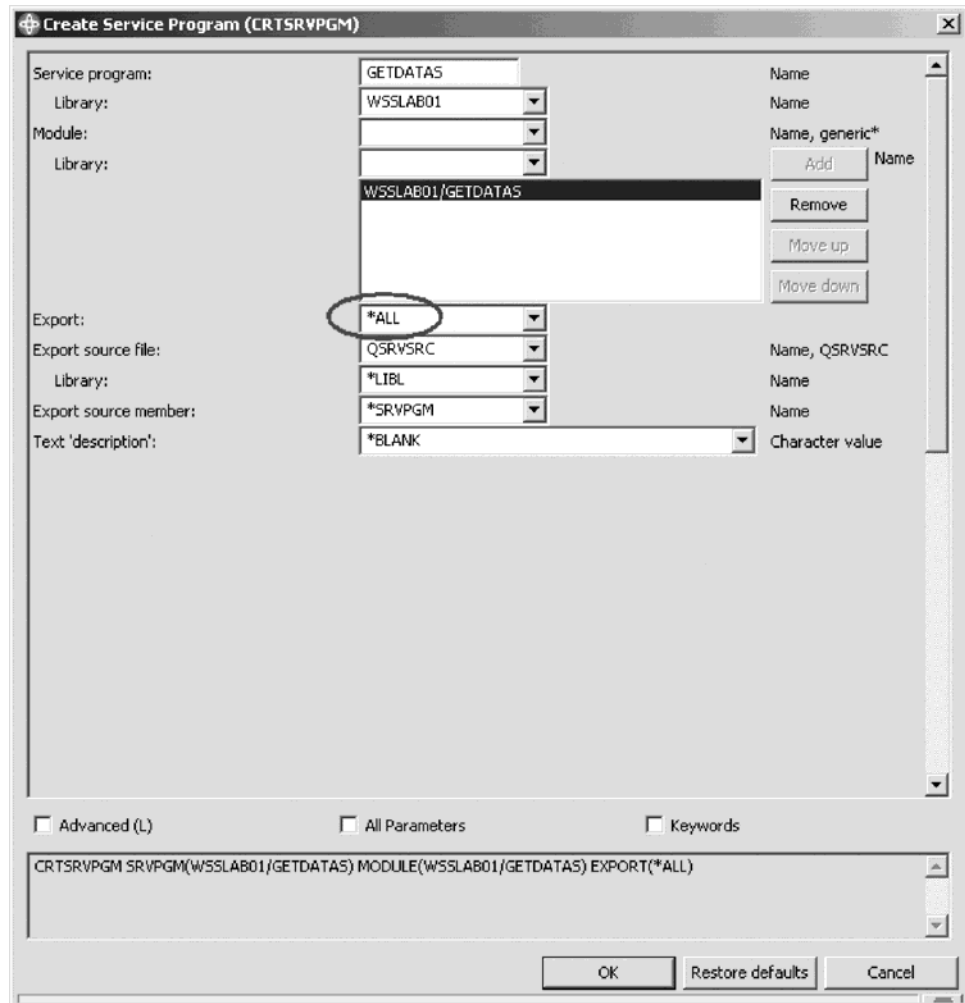
1. In the Remote Systems view find the GETDATAS module in library WSSLABxx.



Note: If you don't see the module, right-click library WSSLABxx and select **Refresh**.

2. Right-click GETDATAS and click **Create > Service Program** on the pop-up menu.

The Create Service Program (CRTSRVPGM) window opens:



3. In the Export list, select ***ALL**.
4. Click **OK**.
Check the command log to see that the Service program was created.
5. Click the **iSeries Commands Log** tab at the bottom of the workbench.
You should see a message that the Service program was created.



Finally you are ready to run your Web application.

Recap

Congratulations! You have completed Module 5: Adding the iSeries program to your Web application. You should now understand:

- LPEX editor features
- How to open the Remote System Explorer perspective
- How to find an iSeries source member
- How to edit an iSeries source member
- How to compile a program object or a service program object

Now that you have created a program object or service program object continue to Module 6: Running the Web application.

Chapter 6. Module 6: Running the Web application

In this module you learn how to test the Web application locally without deploying the files in the Web project to the host server. Testing your Web application locally includes using the iSeries Web Tools Run-time Configuration wizard to specify the authentication and run-time values for every host program or procedure call made by any Web interaction in the Web project. This is something you have already done in Exercise 2.4 and running your Web application in the WebSphere test environment.

In order to run the Web application, there are several steps you will need to learn about and follow:

- Opening the Web perspective and finding the Web application
- Running the Web application in the WebSphere test environment

In order to accomplish these learning objectives, there are several steps that are involved, including:

- Exercise 6.1: Opening the Web perspective
- Exercise 6.2: Finding the Web application
- Exercise 6.3: Selecting the Run on Server option

Note: To access iSeries data and resources you will need to have a job description setup and associated with your user profile on your iSeries system.

The exercises in this module must be completed in order. Start with Exercise 6.1 when you are ready to begin.

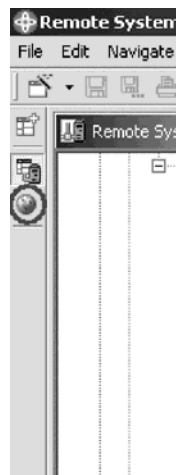
Length of time:

This module will take approximately 5 minutes to complete.

Exercise 6.1: Opening the Web perspective

To open the Web perspective:

1. Click the Web perspective  icon, on the left taskbar of the workbench.



2. If there is no Web perspective icon, select **Window > Open Perspective > Web** on the workbench menu.

Exercise 6.2: Finding the Web application

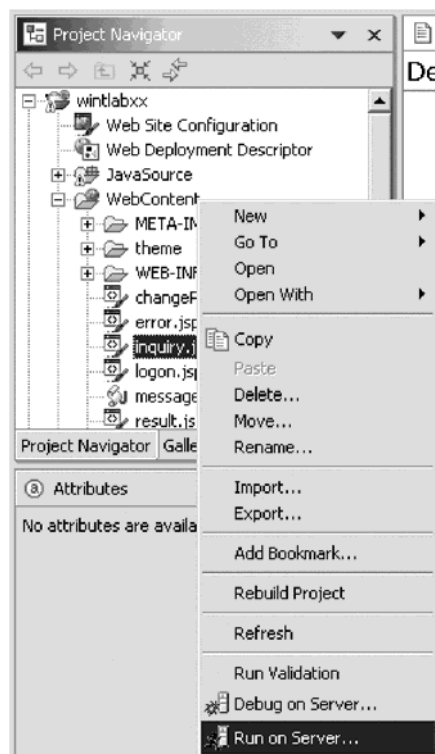
To find the Web application:

1. In the Project Navigator, expand your project wintlabxx if it is not already expanded.
2. Expand the WebContent folder if it is not already expanded.

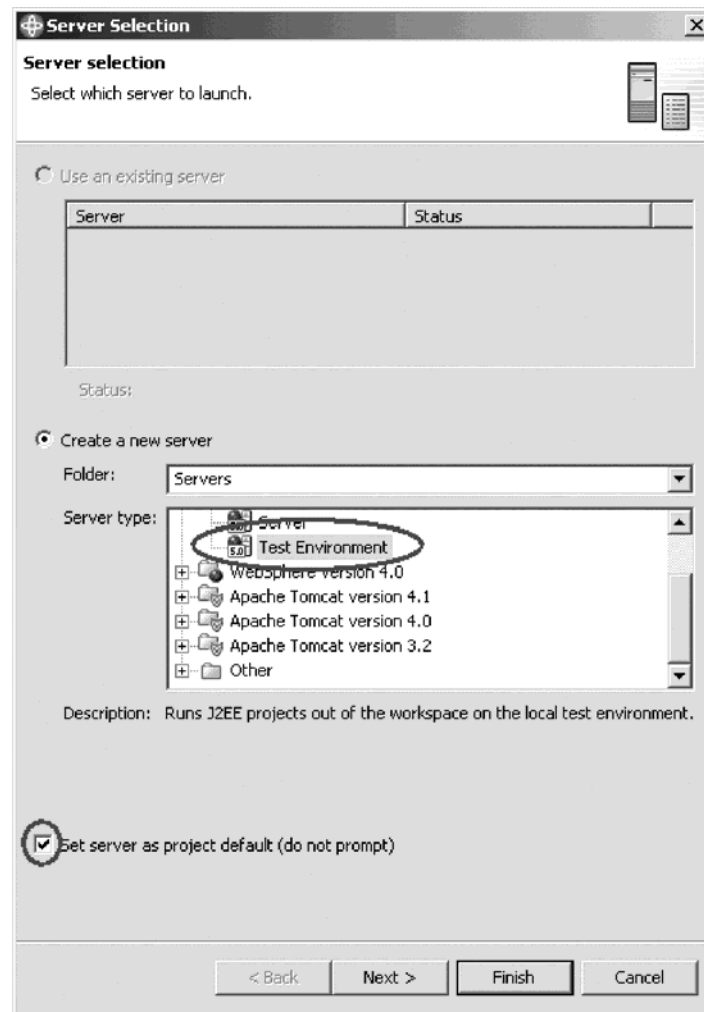
Exercise 6.3: Selecting the run on server option

To test your application locally using the WebSphere test environment:

1. Right-click inquiry.jsp and click **Run on Server** on the pop-up menu.

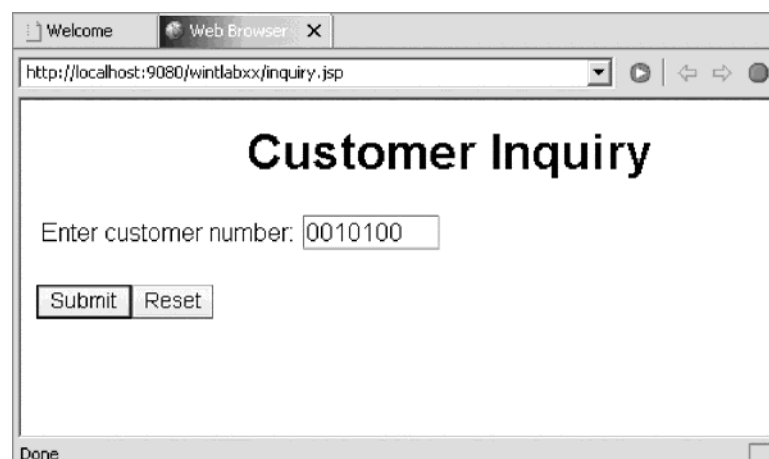


You will see a message to select the Server to run in the **Test Environment**:



2. Make sure **Test Environment** is selected in the **Server type** list.
3. Select the **Set server as project default (do not prompt)** check box.
4. Click **Finish** to configure and launch the server.

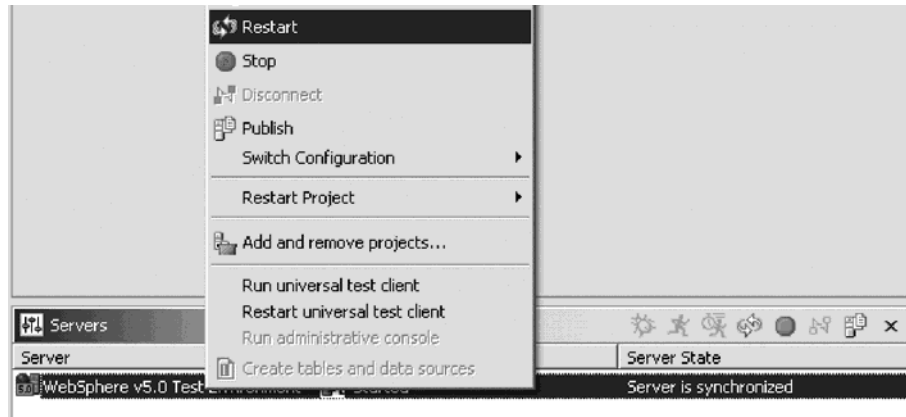
After the WebSphere server has started, your Web application input page will display in the workbench browser.



5. Type 0010100 as the **Customer Number**.
6. Click **Submit**.

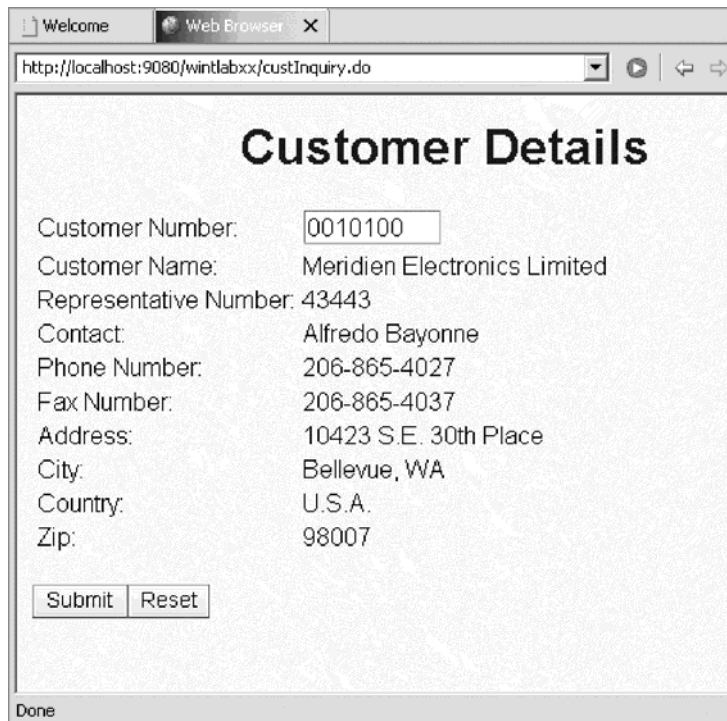
This launches the generated output or result page.

If you see a **Page not found error**, go to the **Servers** view.



7. Restart the Server instance.

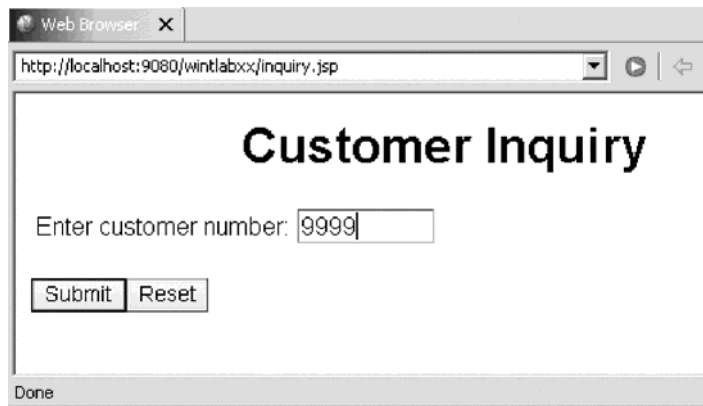
Then go back and try the **Run on Server** option again; the Server sometimes needs to be restarted to pick up the changes. You will see the Result page with customer data appearing in the workbench browser.



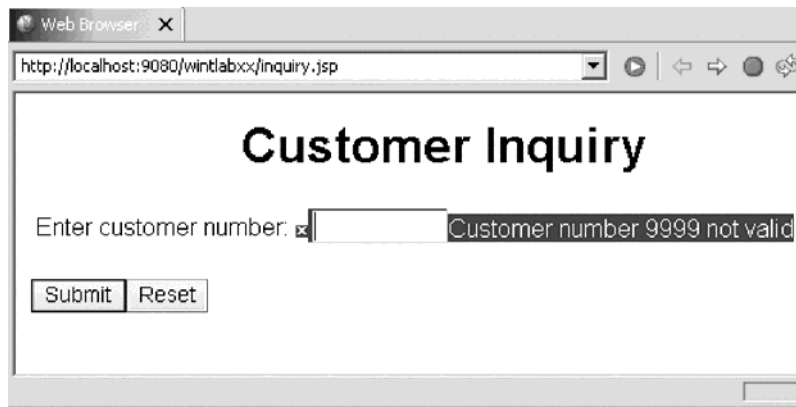
Next, test the message handling:

8. Click the  Back button in the browser to get back to the first page.

9. Enter a wrong customer number, for example, 9999.



10. Click **Submit**.
An error message appears.



This capability is generated by the Web Interaction wizard. Remember that you specified something for the feedback field; you specified the message file to be accessed if the feedback field contains a non 0 value; in this case the feedback parameter will contain the message ID to be used for the error message.

Recap

Congratulations! You have completed Module 6: Running the Web application. You should now understand:

- How to open the Web perspective
- How to find the Web application
- How to run the Web application

Now that you have run your Web application continue to Module 7: Adding Error page to your Web application.

Chapter 7. Module 7: Adding an error page

In this module, you will learn how to create an informative error page for when a customer enters an incorrect customer number. This way, the customer won't just be given an error message, but will have a more informative and helpful response from the Web Application, which is completely customizable.

In order to add error control to the customer inquiry Web application, there are several steps you will need to learn about and follow:

- Using Page Designer to add a new error page.
- Using the Web Interaction wizard to add the error page.
- Using the RSE perspective to modify the program or service program to call the error page.

In order to accomplish these learning objectives, there are several steps that are involved, including:

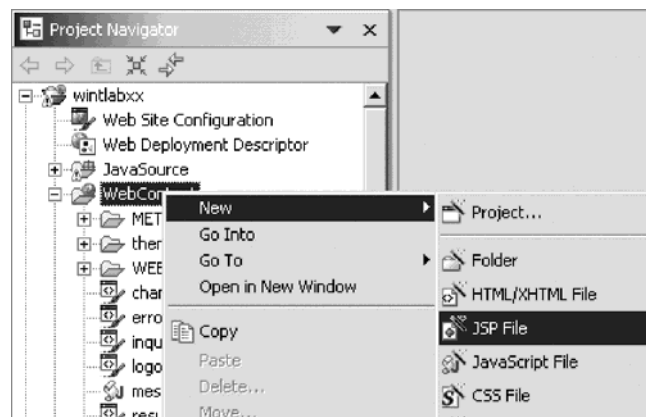
- Exercise 7.1: Creating the flow control page
- Exercise 7.2: Modifying your Web interaction
- Exercise 7.3: Modifying the RPG to implement flow control
- Exercise 7.4: Creating an RPG program (for GETDATA)
- Exercise 7.5: Creating a service program (for GETDATAS)
- Exercise 7.6: Testing the new error page

Exercise 7.1: Creating the flow control page

First we need to create the error page using Page Designer.

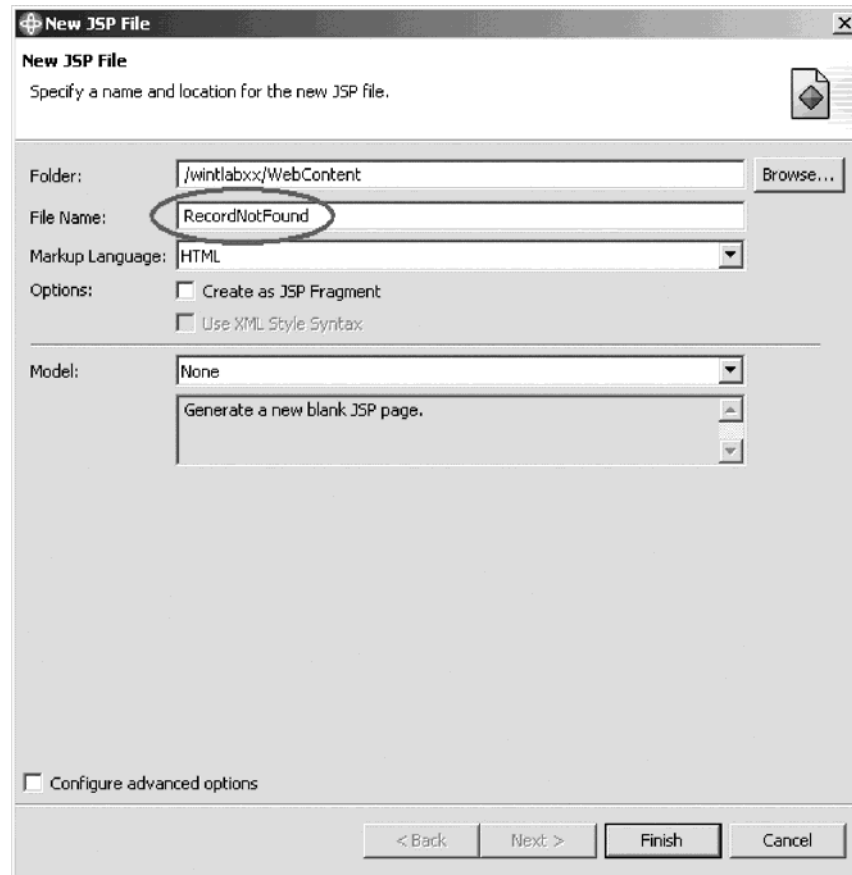
To create the error page:

1. In the Web Perspective, expand your project and drill down to the WebContent folder if not already expanded.



2. Right-click the WebContent folder and click **New > JSP** on the pop-up menu.

The New JSP File page opens:



3. In the **File Name** field, type RecordNotFound.
4. Click **Finish**.

The Page Designer automatically opens.

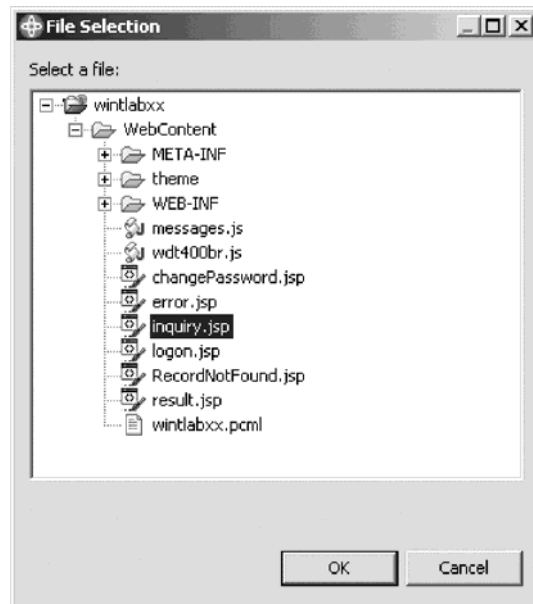
Your window will look like this:



Place content here.



5. Click on the words "**Place content here.**" and change it to something like Sorry, there is no record for that customer number.
6. Press **Enter**.
7. Type Please check the number and try again.
8. Highlight the words try again and right-click.
9. Select **Insert Link** on the pop-up menu.
The Insert Link window opens.
10. Click **Browse** next to the **URL** field.
11. Click **File** on the pop-up menu.
The File Selection window opens:



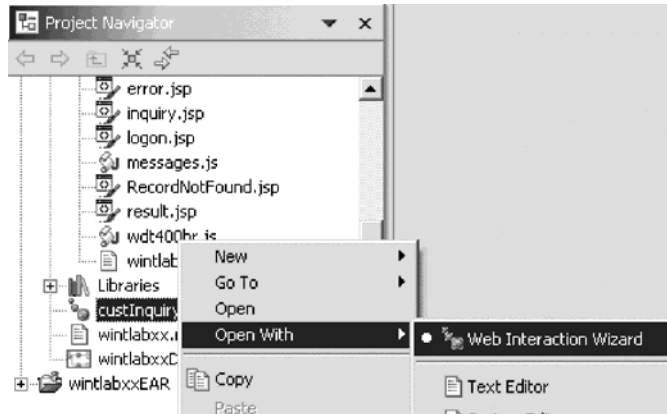
12. Select inquiry.jsp.
13. Click **OK** on the File Selection window.
The Insert Link window appears with the **URL** field updated.
14. Click **OK** on the Insert Link dialog.
You have completed creating the flow control page.
15. Save the file.
You could spend more time customizing this page, but for the purpose of this exercise you will leave the page as is.
16. Close Page Designer.

Exercise 7.2: Modifying your Web interaction

To modify the Web interaction:

1. Expand your project wintlabxx if not already expanded.

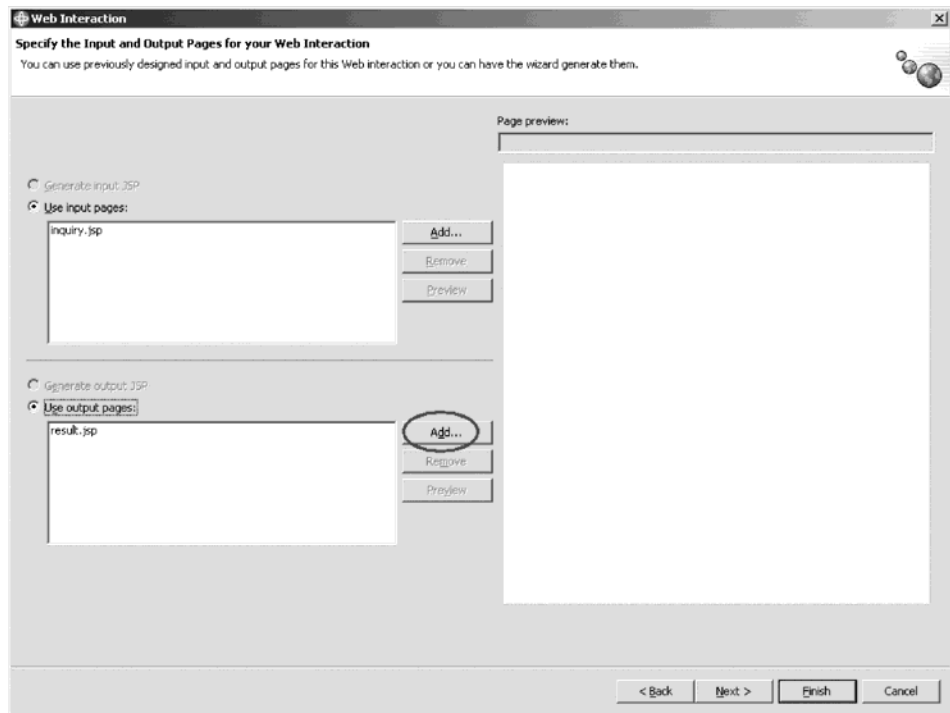
2. Right-click `custInquiry.wit` and select **Open With > Web Interaction Wizard** on the pop-up menu:



The Web Interaction wizard opens:

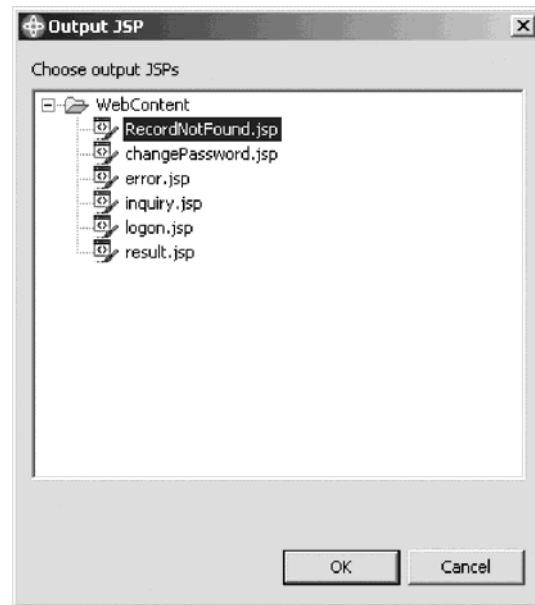
The name and location of your Web interaction are already specified.

3. Click **Next** to add a new output page.

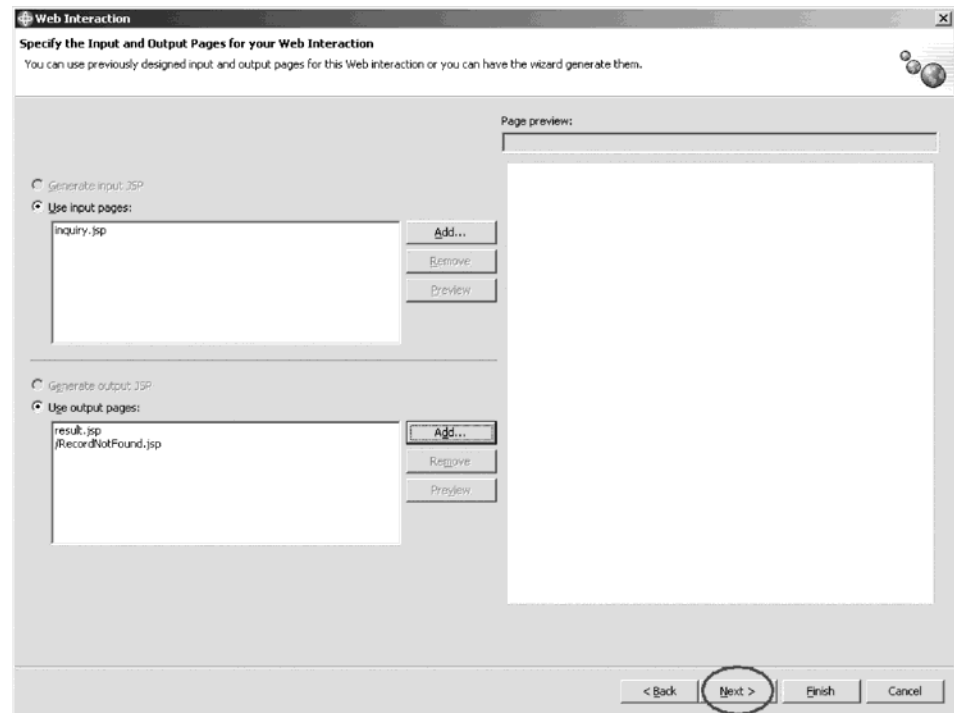


4. Next to the `result.jsp` section, click **Add**.

The Output JSP window opens:



5. Expand WebContent.
6. Select RecordNotFound.jsp.
7. Click OK to add the new output page.

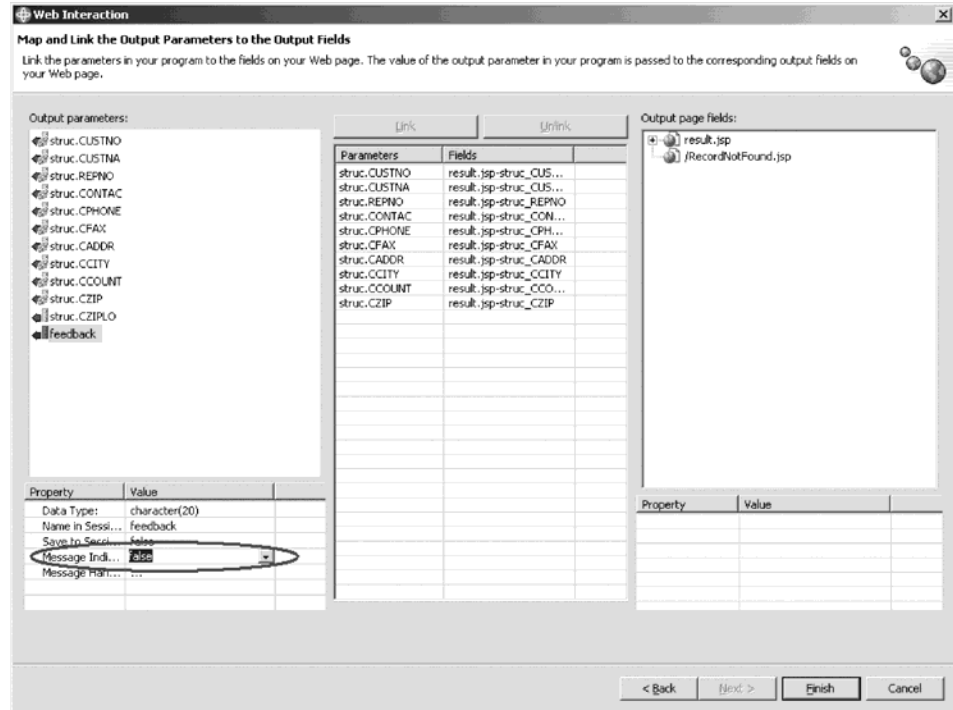


8. Click Next in the Web Interaction wizard.

You will re-use the Feedback parameter to control which page the Web Interaction will send next after it performs its action; the result.jsp or the RecordNotFound.jsp. So you don't have to add another parameter or change the parameters, you just change the values the feedback parameter returns depending on whether a customer record was found or not found.

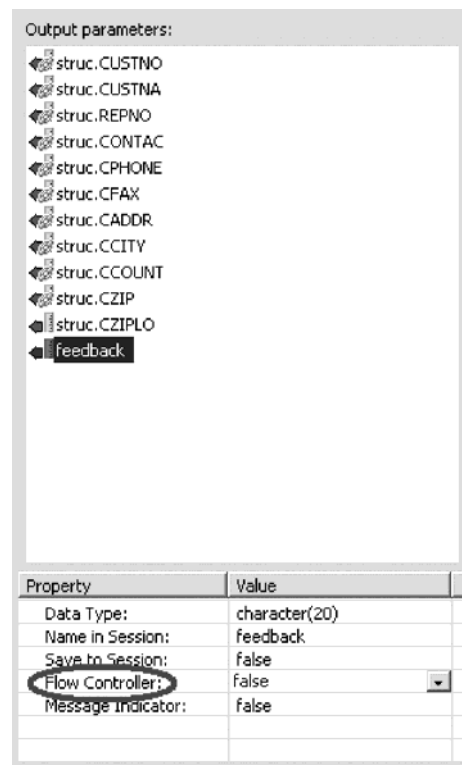
Since you don't need to define the program interface:

9. Click **Next**.
This opens the Input Parameters page.
10. Since you don't need to change any input parameters.
11. Click **Next** again.
This opens the Output Parameters page.

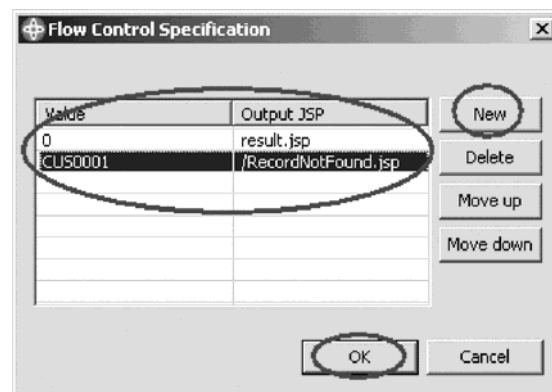


12. Select **feedback** under the **Output Parameters** list.
13. Change the **Message Indicator** property to **false**. Select **feedback** under the **Output Parameters** list again.

Now a **Flow controller** property is added to the Properties table.



14. Change the **Flow Controller** property to **true**.
The Flow Control Specification dialog opens.

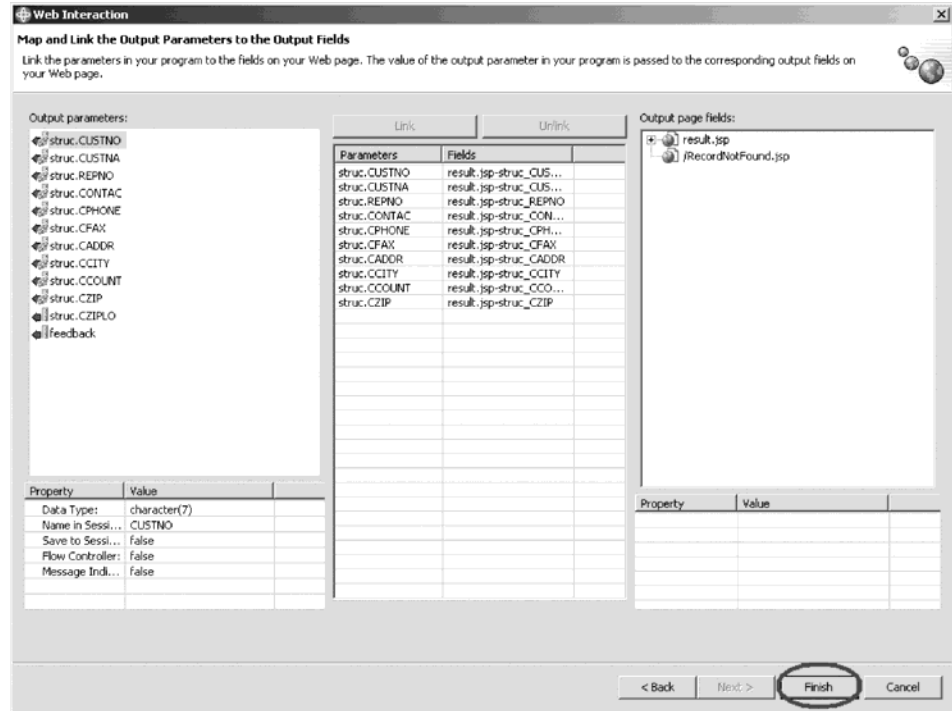


15. Click **New**.
This creates a new flow condition. The default page is result.jsp.
16. The value 0 is already in the **Value** field for the result.jsp.
17. Click **New** to create a new flow condition.
18. Double-click the **Output JSP** field to change the page.
19. Select **RecordNotFound.jsp**.
20. Type CUS0001 in the **Value** field for /RecordNotFound.jsp.
21. Click **OK**.

This will set the flow control such that if the value of Feedback is '0', then the program will output the customer information to the result.jsp page and display it. However, if the customer number doesn't exist in the database, the Feedback value will be "CUS0001", and instead of displaying a message near

the input field, it will show the user the RecordNotFound.jsp page for a more helpful response. After modifying the Web Interaction, you will need to modify your RPG program to remove the customer number from the parameter if a record is not found.

22. Click **Finish** on the Output Parameters page to apply your changes to the Web Interaction.



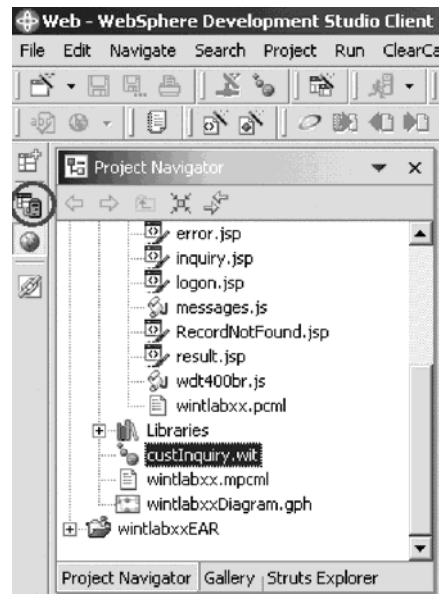
You may see a message indicating that the server needs to be restarted. Click **Yes** to do so.

Exercise 7.3: Modifying the RPG to implement flow control

You will now modify the RPG code to change the values returned in the Feedback parameter.

To modify the RPG code:

1. Look for the Remote System Explorer icon  in the left taskbar of the workbench.

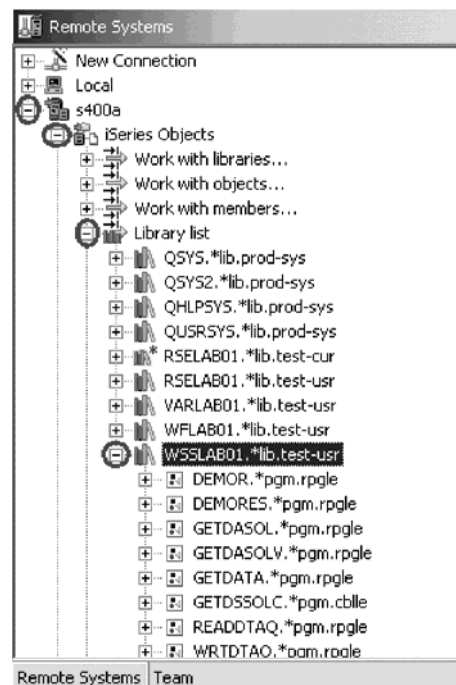


2. If you see it, click the Remote System Explorer icon.
3. Otherwise, select **Window > Open Perspective > Remote System Explorer** on the workbench menu.

The Remote System Explorer perspective opens.

A list of servers in the Remote Systems view displays.

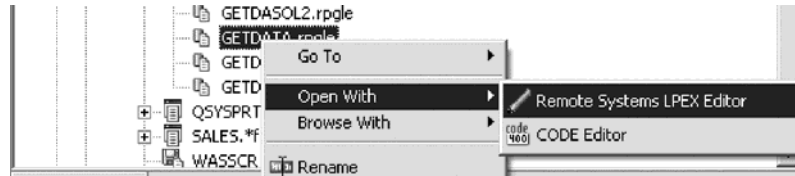
4. Expand iSeries server, for example, s400a if not already expanded.



Use the same server that you have used to specify the reference fields in the data structure.

5. Expand iSeries Objects if not already expanded.

6. Expand Library list if not already expanded.
7. If prompted, type your user ID and password.
8. Expand the library WSSLABxx.
You will see all objects in this library.
9. Expand the source file QRPGLSRC.



10. Scroll down to the source files in the expanded list.
Again you work with the same source members as before. Depending on whether you specified that you wanted to work with a program, or a procedure in a service program, you need to select a different member:
11. For the program call, right-click GETDATA member.
12. For the service program call, right-click GETDATAS member.
13. Select **Open with > Remote Systems LPEX Editor** on the pop-up menu.
14. Depending on your choice above, follow either the section named GETDATA instructions or skip to the section named GETDATAS instructions.

Let's switch to the Remote System Explorer perspective to begin.

GETDATA instructions

To update GETDATA instructions:

1. Locate the RPG code you edited earlier.
It should be marked between the comments "Insert code here" and "Done with inserting code".
2. Find this line of code:


```
C                                EVAL      FEEDBACK='CUS0001' + CUSTNO1
```
3. Change this line to:


```
C                                EVAL      FEEDBACK='CUS0001'
```

Now when the requested record is not found the value CUS0001 gets returned without the customer number in the parameter **Feedback**.
4. **Save** the file.
5. Go to Exercise 7:4 Creating a *PGM object (for GETDATA) .

GETDATAS instructions

To update GETDATAS instructions:

1. Locate the RPG code you edited earlier. It should be marked between the comments "add your code after this statement" and "your code ends here".
2. Locate the RPG code you edited earlier. It should be marked between the comments "Insert code here" and "Done with inserting code".
3. Find this line of code:


```
C                                EVAL      FEEDBACK='CUS0001' + CUSTNO1
```
4. Change this line to:


```
C                                EVAL      FEEDBACK='CUS0001'
```

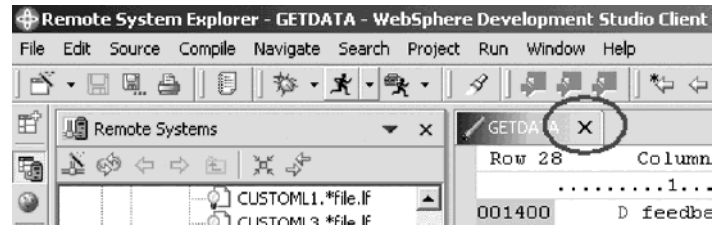
Now when the requested record is not found the value CUS0001 gets returned without the customer number in the parameter **Feedback**.

5. Save the file.
6. Go to Exercise 7:5 Creating a *SRVPGM object (for GETDATAS).

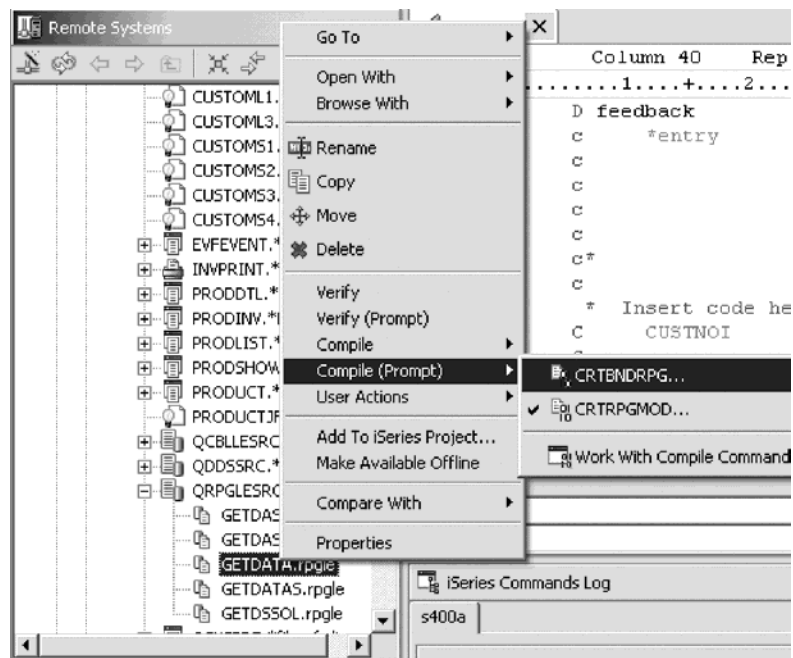
Exercise 7.4: Creating a *PGM object (for GETDATA)

Your code is complete, so let's re-create the program.

To create the program:

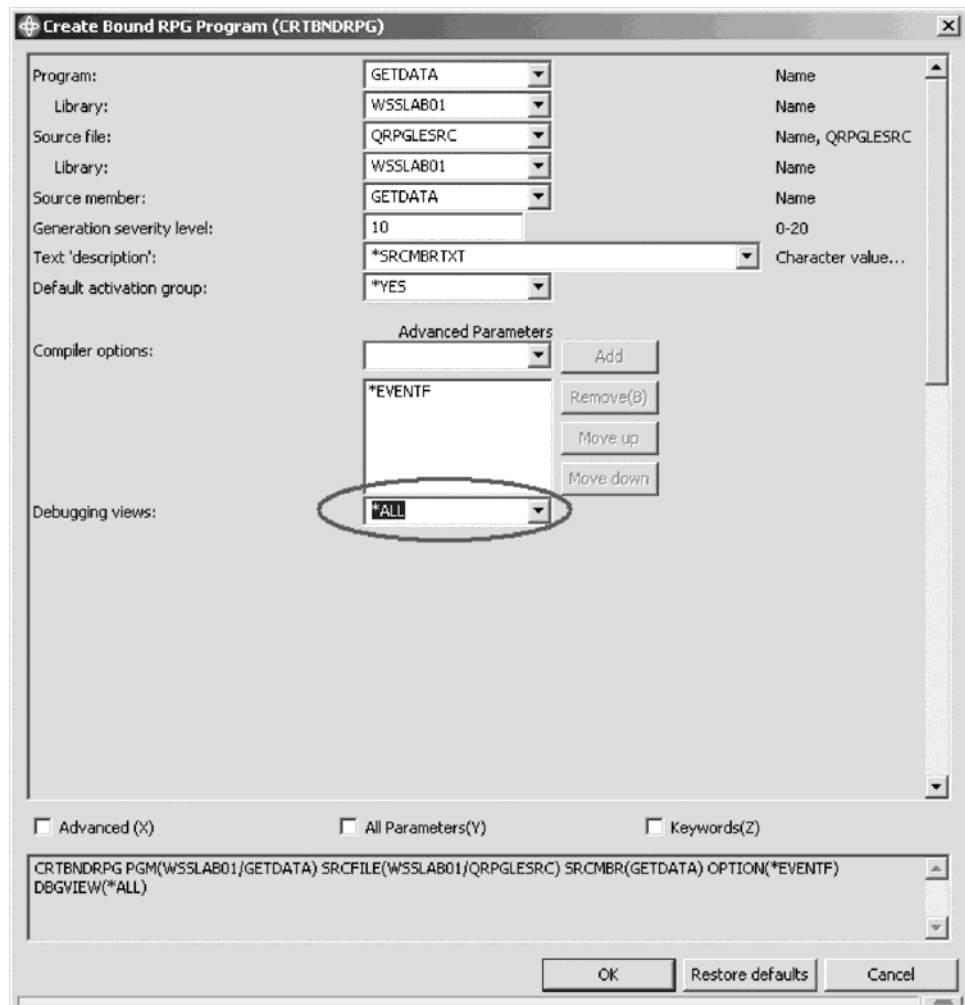


1. Click the X in the Editor window title bar to close the member.
Be careful to not click the X on the workbench window title bar.
2. In the Remote Systems view, right-click GETDATA member in QRPGLSRC.



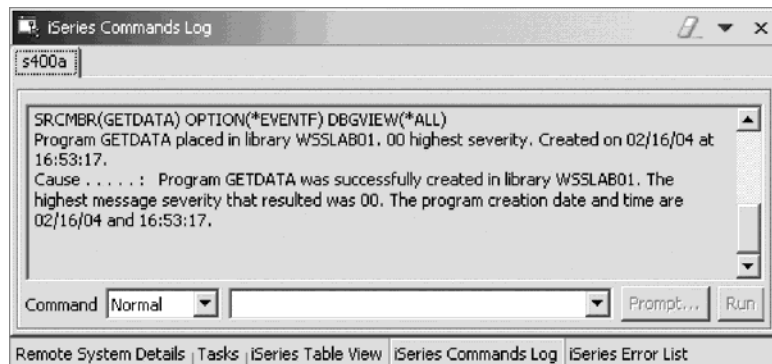
3. Click **Compile (Prompt) > CRTBNDRPG** on the pop-up menu.

The Create Bound RPG Program (CRTBNDRPG) window opens:



4. In the **Debugging views** list, select ***ALL**
5. Click **OK** to start to compile.

After compile completes the error list view is shown, listing all compile errors. You may see only information and warning messages which means your compile was completed and the program object was created.



Click the **iSeries Commands Log** tab to check that the compile completed successfully. This log shows the Remote System Explorer job messages. Now you are ready to run your Web application.

Go to the Exercise 7.6: Testing the new error page and skip Exercise 7.5: Creating a service program (for GETDATAS). If you get a message that there have been errors found in your program, go through the edit compile fix cycle.

Fixing errors

In the error list view check the errors:

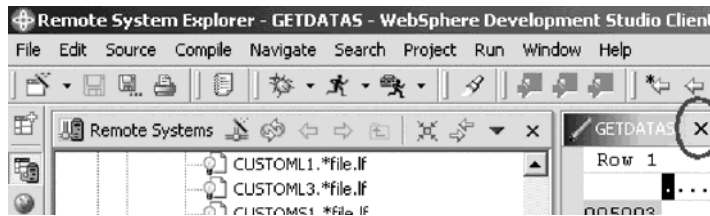
1. Double-click on the error.
This positions the cursor in the Edit window on the statement that is wrong.
2. Fix all errors.
3. Save the source member.
4. Go back to the Remote Systems view.
5. Right-click the member to run the CRTBNDRPG command.
6. Continue this cycle until you get a clean compile.

Exercise 7.5: Creating a *SRVPGM object (for GETDATAS)

Your coding is complete, so let's create the service program.

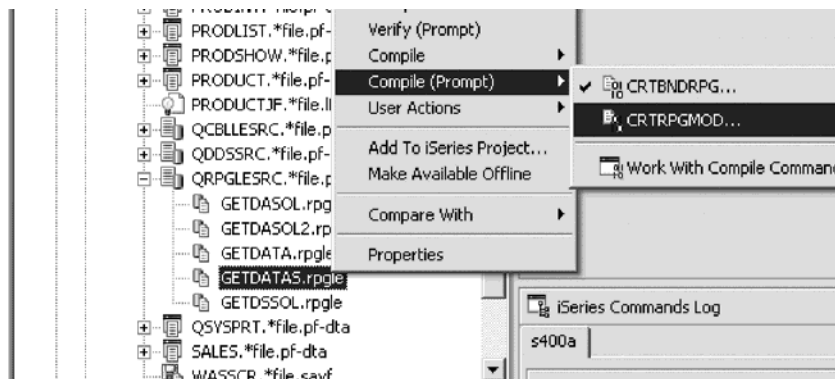
To create the service program:

1. Click the X in the Editor window title bar to close the member.



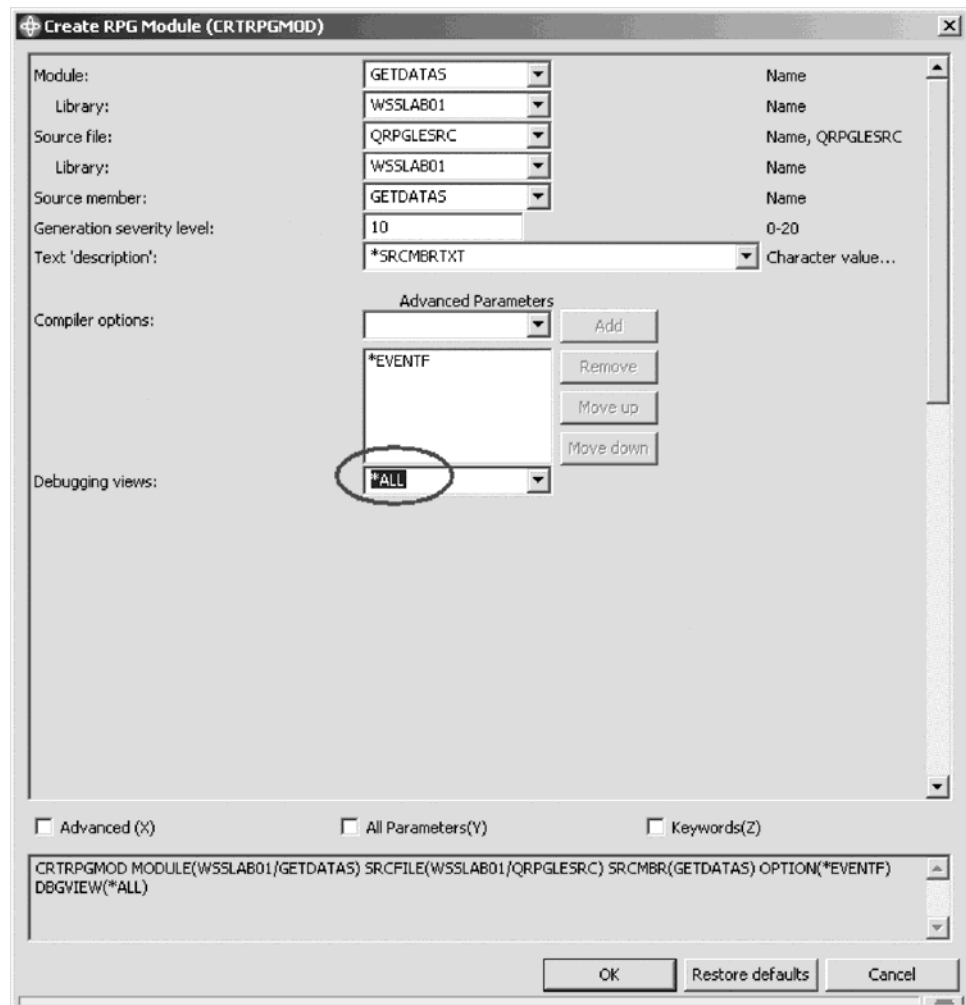
Be careful not to click the X on the workbench window title bar.

2. In the Remote Systems view, right-click the GETDATAS member in QRPGLSRC.



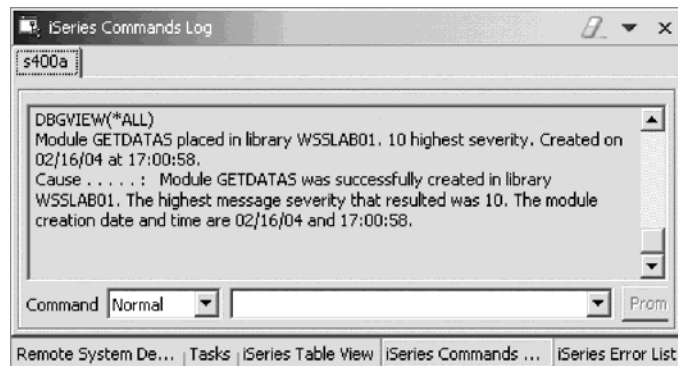
3. Click **Compile (Prompt) > CRTRPGMOD** on the pop-up menu.

The Create RPG Module (CRTRPGMOD) window opens:



4. In the **Debugging views** list, select ***ALL**.
5. Click **OK** to submit the command.

After compile completes the error list view is shown, listing all compile errors. If you see only information and warning messages that means your compile was completed and the program module was created.



Click the **iSeries Commands Log** tab to check that the compile was successful. This log shows the Remote System Explorer job messages. Now you are ready to create service program.

Go to the section Creating a service program. If you get a message that there have been errors found in your program, go through the edit compile fix cycle.

Fixing errors

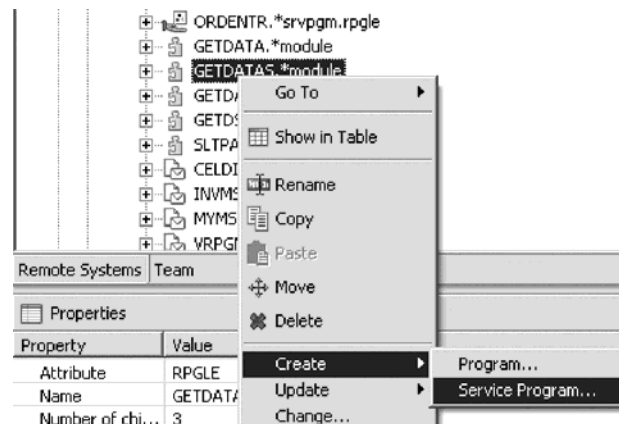
In the error list view check the errors:

1. Double-click on the error.
This positions the cursor in the Edit window on the statement that is wrong .
2. Fix all errors.
3. Save your source member.
4. Go back to the Remote Systems view.
5. Right-click the member to run the CRTRPGMOD command.
6. Continue this cycle until you get a clean compile then proceed down to the next step.

Creating a service program

To create a service program:

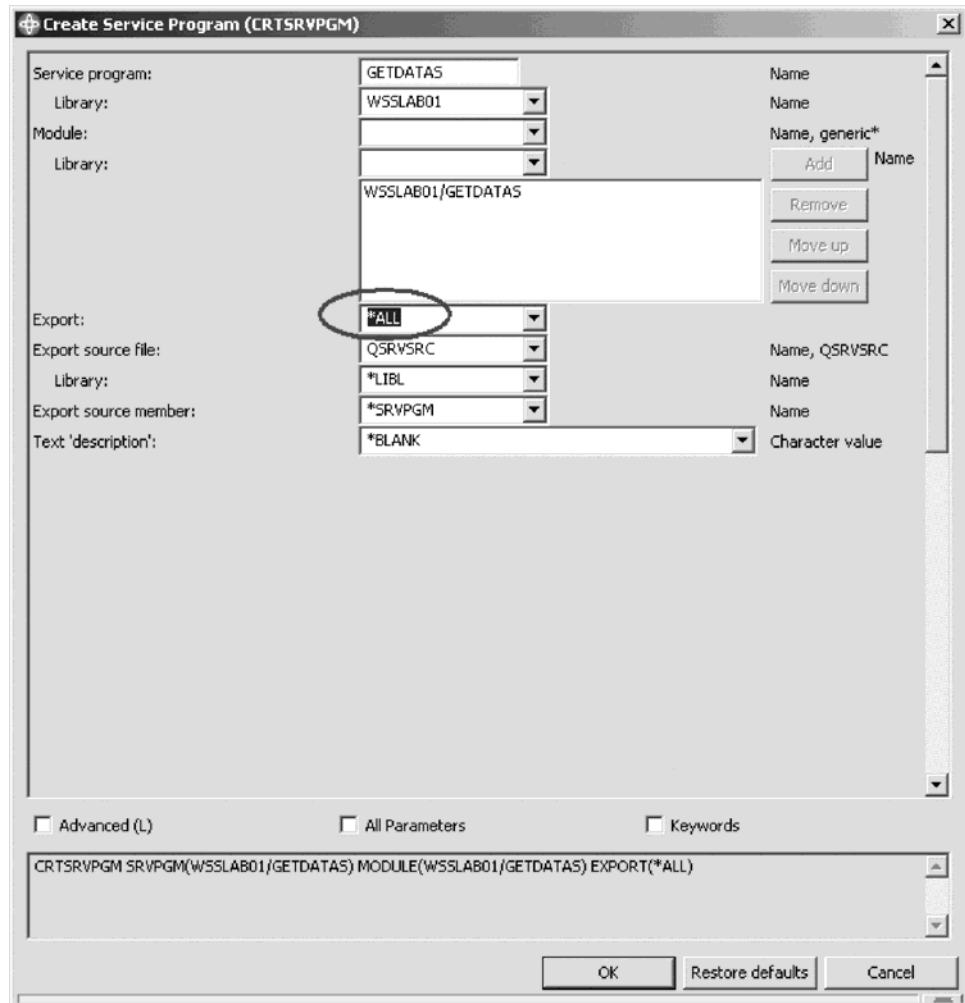
1. In the Remote Systems view, find the GETDATAS module in library WSSLABxx.



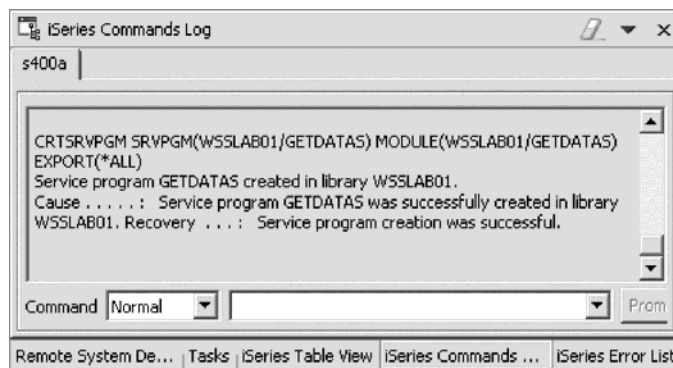
Tip: If you don't see the module, select the library, right-click and then select **Refresh**.

2. Right-click the module.
3. Click **Create > Service Program** on the pop-up menu.

The Create Service Program (CRTSRVPGM) window opens:



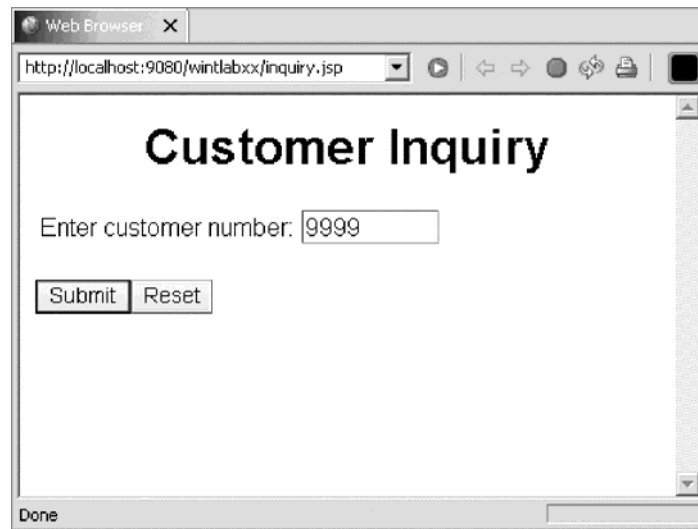
4. In the Export list, select *ALL
5. Click OK.
6. Check the **iSeries Commands log** to see that the service program was created.



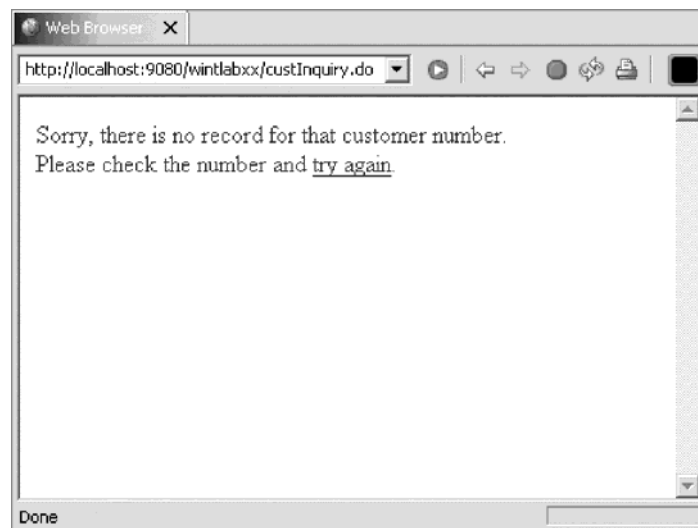
7. Click the **iSeries Commands Log** tab at the bottom of the workbench. You should see a message that the service program was created.

Exercise 7.6: Testing the new error page

Now that the flow control is in place, if you enter a wrong customer number,



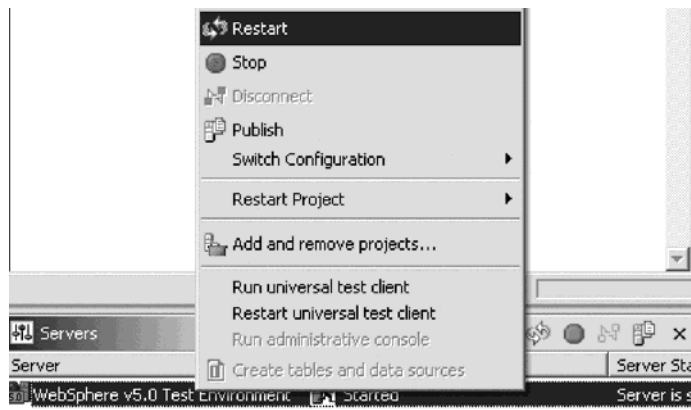
you will get the error page instead of a message.



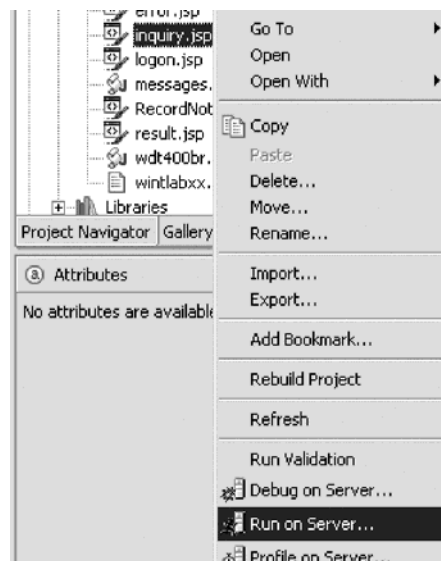
Now follow the instructions below to run the modified project on the WebSphere Test Environment to see the result of adding flow control to your Web application.

The server should still be running from the last time you ran the Web application. Every time you make a change to the Web application, you need to restart the server to pick up the changes.

To restart the server:



1. In the Web Perspective, click the **Servers** tab to open the Servers view.
2. Select the server instance. Right-click and click **Restart** on the pop-up menu.
The server instance should restart.
3. In the Web perspective Project Navigator locate the WebContent folder, and right-click inquiry.jsp.



4. Click **Run on Server** on the pop-up menu.
5. Enter a wrong customer number, for example 9999.
6. Click **Submit**.
The error page opens.

Recap

Congratulations! You have completed Module 7: Adding an error page. You should now understand how to:

- Use page designer to add a new error page.
- Use the Web interaction wizard to add the error page.
- Use the RSE perspective to modify the program or service program to call the error page.
- Create a program or service program object.

Now that you have added an error page to your Web application to handle incorrect customer numbers, you can continue to Module 8: Enhancing the customer inquiry input page using Web tools.

Chapter 8. Module 8: Enhancing the input page using Web tools

In this module, you will learn how to use the Web tools in the Development Studio Client workbench to update the input page of your customer inquiry Web application. The input page named `inquiry.jsp` is one of the files created by the Interaction wizard. This page is very plain. You will learn how to use Page Designer and some related Web tools in the workbench to add some color to the input page, as well as add some pictures to make it more interesting Web page.

In order to enhance the customer inquiry input page, there are several steps you will need to learn about and follow:

- Using Page Designer
- Using Page Designer components to create a more attractive browser interface
- Adding a style sheet to the page
- Adding a heading that moves
- Adding a graphic from the sample gallery
- Creating a Logo with a Web design tool
- Testing this new input page

In order to accomplish these learning objectives, there are several steps that are involved, including:

- Exercise 8.1: Opening Page Designer
- Exercise 8.2: Working with page properties
- Exercise 8.3: Linking a cascading style sheet to the Web page
- Exercise 8.4: Designing and adding a logo
- Exercise 8.5: Adding a heading1 tag to the page
- Exercise 8.6: Adding a picture to the page
- Exercise 8.7: Adding moving text to the page
- Exercise 8.8: Changing the text color in Page Designer

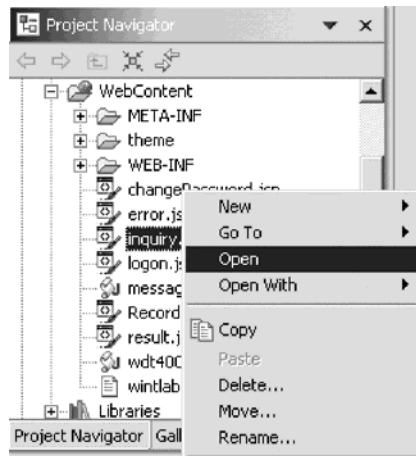
Exercise 8.1: Opening Page Designer

The first step is to locate the `inquiry.jsp` file and to start Page Designer. In Page Designer you will link the page to a cascading style sheet that comes as a sample with the workbench.

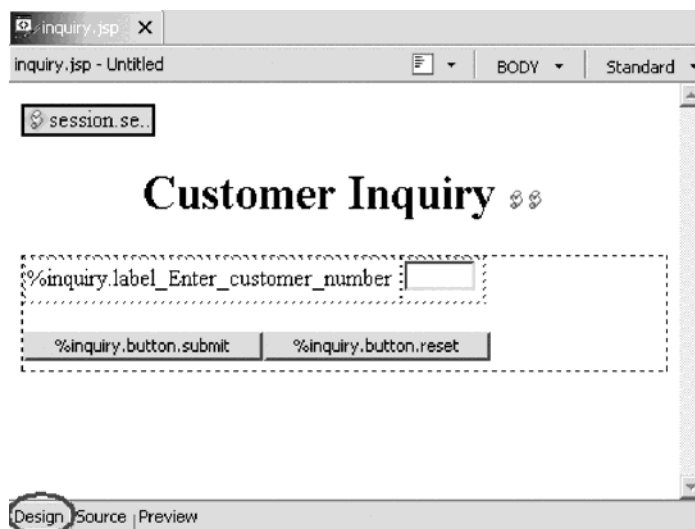
To open the Web perspective:

1. In the workbench, open the Web perspective.

You should now have the Project Navigator view in your workbench environment.



2. Expand the wintlabxx Web project if not already.
3. Expand the WebContent folder if not already.
4. Right-click inquiry.jsp and click **Open** on the pop-up menu.



The Page Designer appears in the upper right pane of the workbench and shows the inquiry.jsp page as the Web Interaction wizard created it.

Make sure that you are on the Design page in Page Designer.

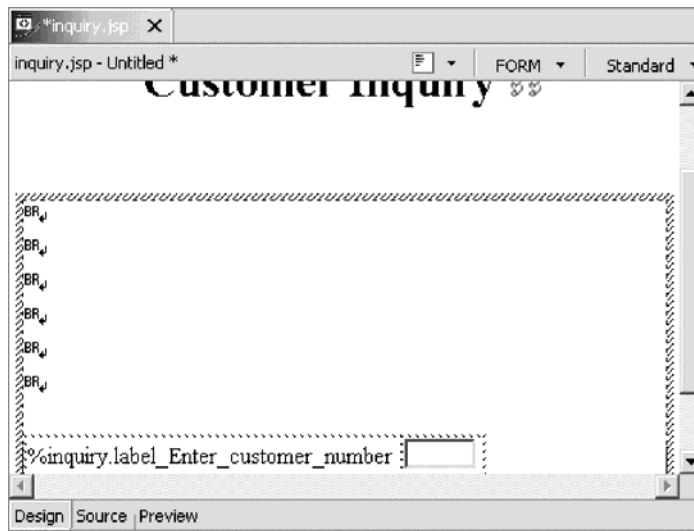
5. Click the **Design** tab.

Exercise 8.2: Working with page properties

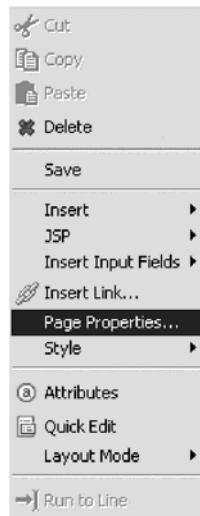
Next you move the form down.

To work with page properties:

1. Click underneath the heading Customer Inquiry.

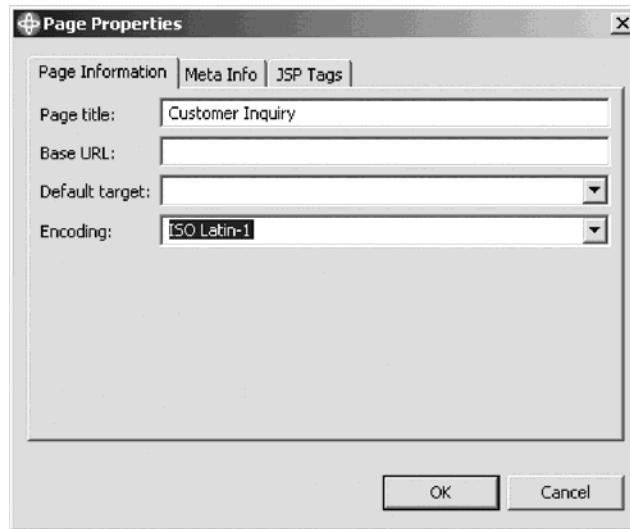


2. Press **Enter** number of times as shown above.
3. To change the page properties, right-click the background of the inquiry.jsp page in Page Designer.



4. Click **Page Properties** on the pop-up menu.

The Page Properties window opens.



This window allows you to change some of the page properties.

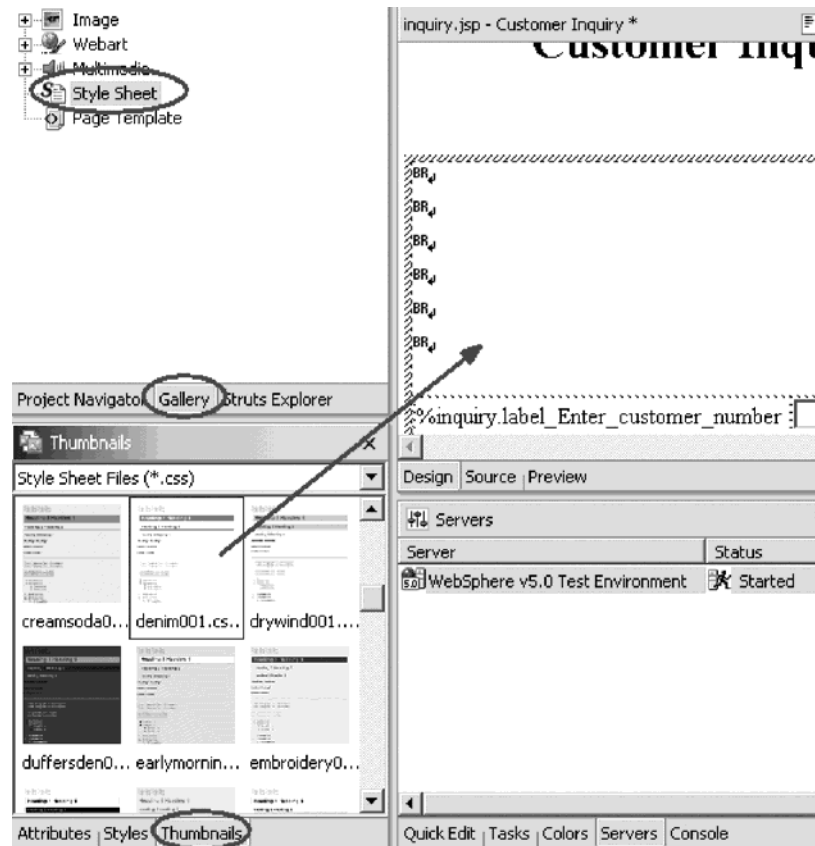
For example you could change the title. When this page is shown in a browser, the Window title bar of the browser will display Customer Inquiry.

5. Close the Page Properties dialog.



Exercise 8.3: Linking a cascading style sheet to the Web page

Now you will add a style to the Web page. You can use a style that is used in your company or you can use one of the sample style sheets that are provided in Development Studio Client.

To link a style sheet to a Web page:



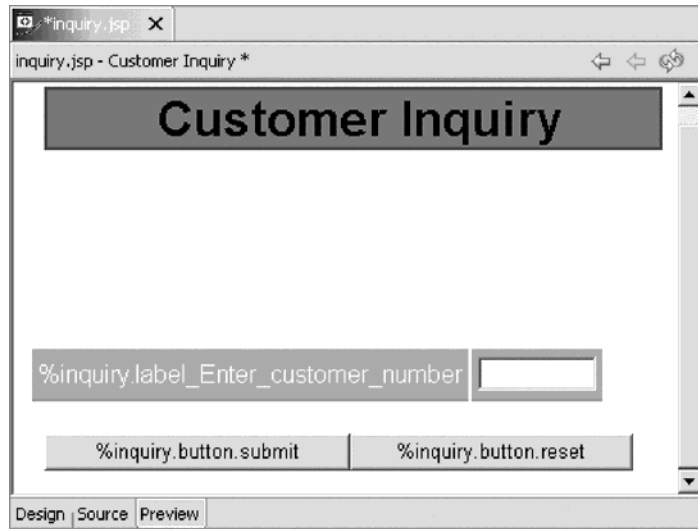
1. Click the **Gallery** tab, if the Gallery view isn't already displayed.
2. Click the **Style Sheet** icon in the Gallery list.
3. Click the **Thumbnails** tab in the bottom view in the workbench.
You should see thumbnail icons of all the styles available as shown above.
4. In the thumbnail view, scroll down to the bottom, until you see style sheet denim001.css in the list, or select a style sheet that you like best.
5. Click the thumbnail picture of denim001.css.
6. Hold the left mouse button down and drag the mouse cursor to the Page Designer window.

The cursor will change from this shape  , to this shape  . When the latter cursor shape appears in the Page Designer window then, let go of the mouse left button.

After a short while the Style sheet properties will be applied.

7. Click the **Preview** tab.

You will see the colors in the page have changed to the style sheet definitions as shown below:

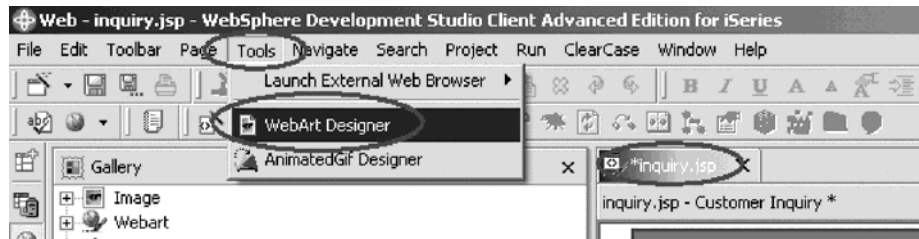


8. Click the **Design** tab to get back to the Design page.

Exercise 8.4: Designing and adding a logo

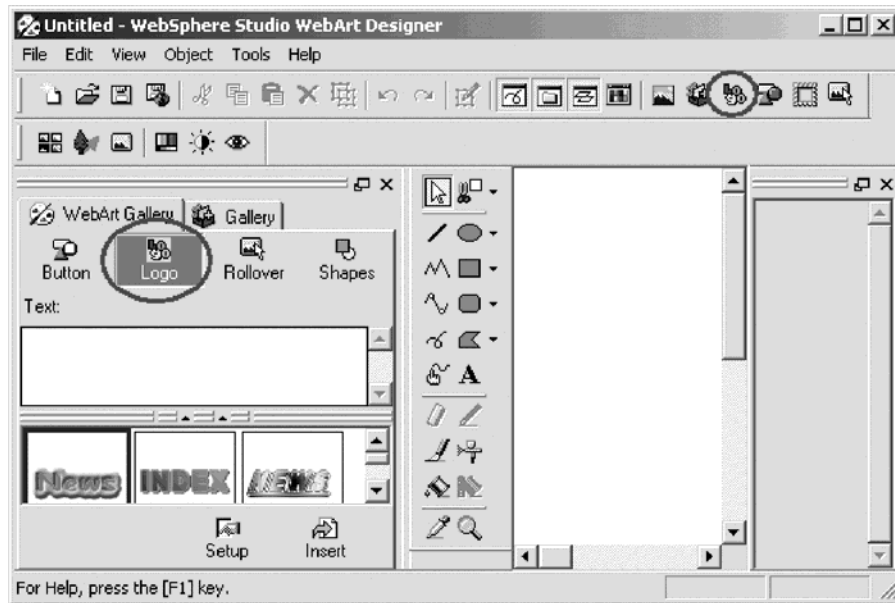
Now that you have the overall Web page look specified you will use the WebArt Designer to create a Logo that you then will add to this page.

To start the WebArt Designer:



1. Make sure the Page Designer is in focus.
2. Click **Tools > WebArt Designer** in the workbench menu.

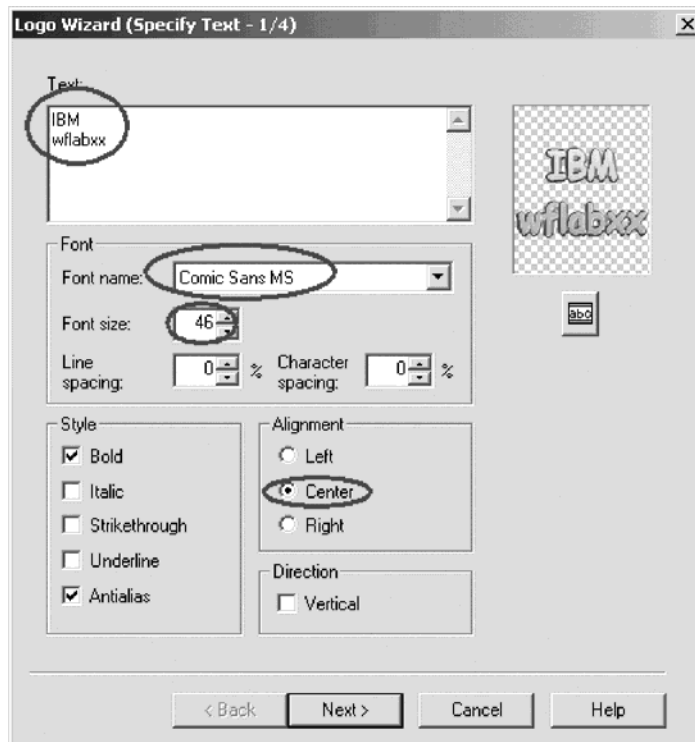
The WebArt designer opens as shown below.



The WebArt designer shows the Template Gallery on the left, where you find samples of logos, buttons, rollovers, images and more. The big white area in middle of the dialog is the canvas that is used to work with objects that you want to create or change. Now you will create a logo from scratch.

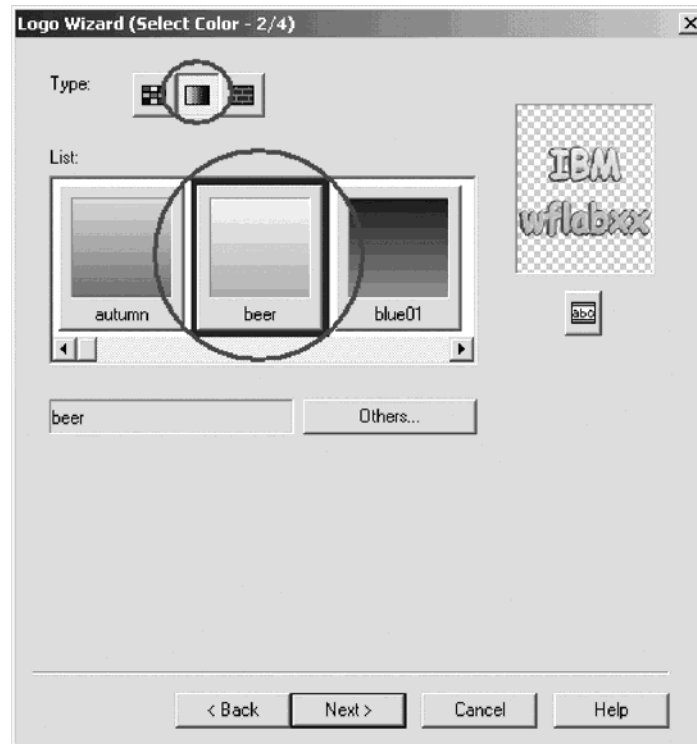
You could also select one from the template gallery as the base for your own logo.

3. Click the **Create Logo** button above the canvas or click **Object > Create Logo**. The Logo Wizard opens:



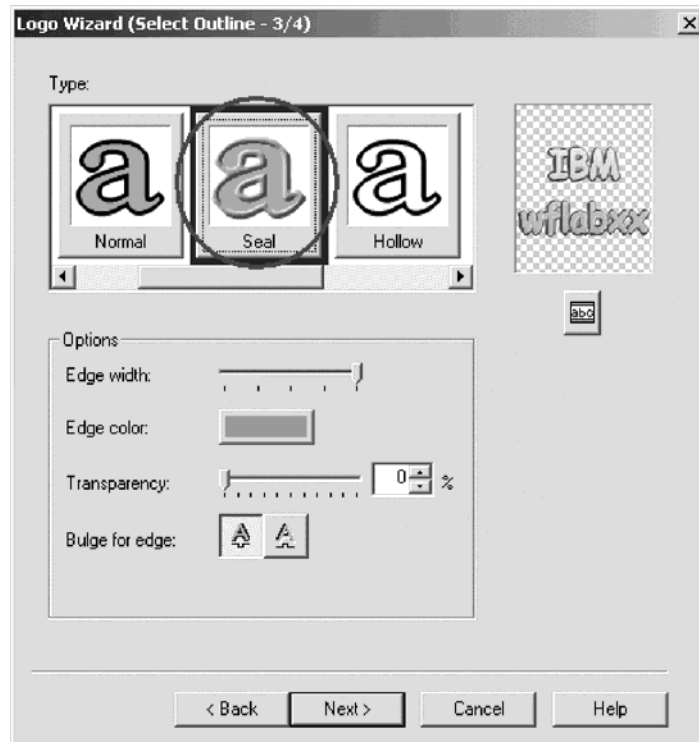
4. In the **Text** field, type your company name and in the next line in the **Text** field, type wintlabxx.
5. In the **Font name** list, select **Comic Sans MS**.
6. In the **Font size** list, select **46**.
7. Under **Alignment**, select the **Center radio** button.
Notice in the upper right corner of the dialog, a sample of the logo as specified at the moment is being displayed.
8. Click **Next** to go to the next page of the wizard.

The Select Color page opens:



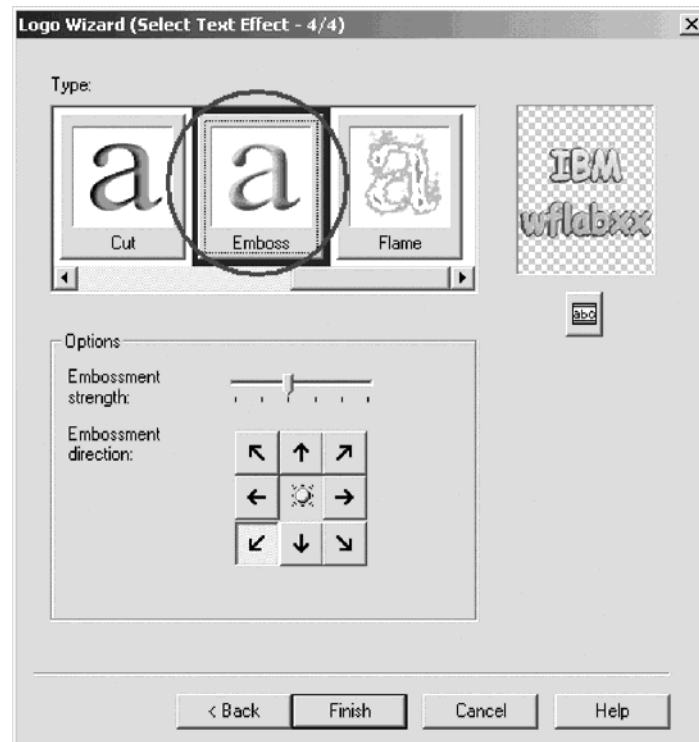
9. Select the **gradation type** button; the middle one of the three type push buttons.
Tip: The other buttons select color types: solid and textured.
10. Select beer from the colors available, or any other color you like best, just scroll through the list to find a gradation you like.
Tip: You can change the colors by clicking the Others button on this dialog and create you own gradation.
11. Click **Next** to go to the next page of the wizard.

The Select Outline page opens:



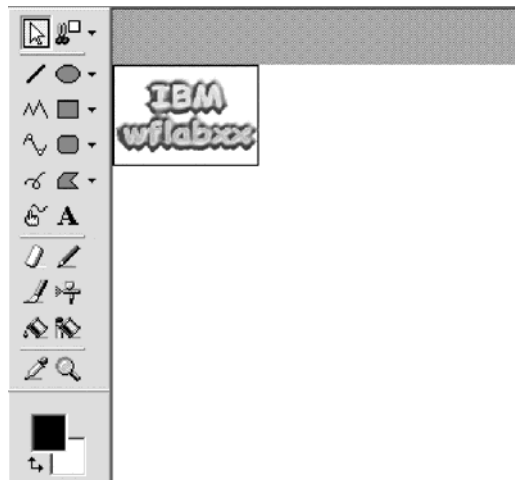
12. Select the **Seal outline** from the list, or any outline you like best.
13. Click **Next**.

The Select Text Effect page opens:



14. Select the **Emboss text effect**, or one that you like best.
15. Click **Finish**.

You return to the WebArt designer window:



Resize the logo object on the canvas:

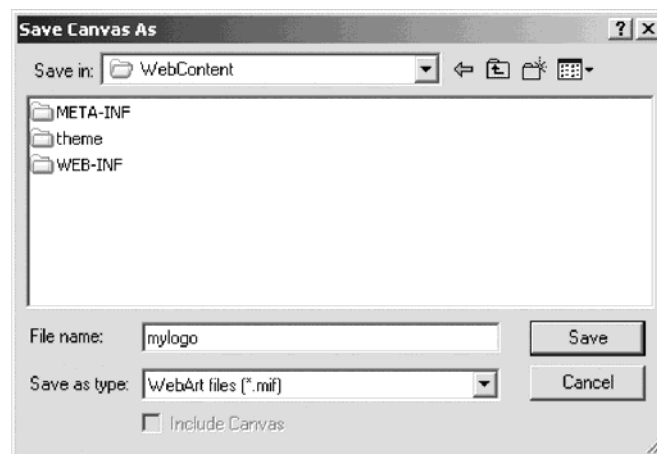
16. Click the logo object to select it.
17. Move the cursor to the rectangle at the right bottom corner of the object; watch the cursor changing shape.
18. Drag the rectangle up and to the left so the object becomes smaller.

Now you need to save this object. First save it as a WebArt object. This allows you later on to work with the object again in WebArt designer, but you can't use that format for your Web page.

To save this object:

1. Click **File > Save** on the WebArt Designer menu.

The Save Canvas As window opens:



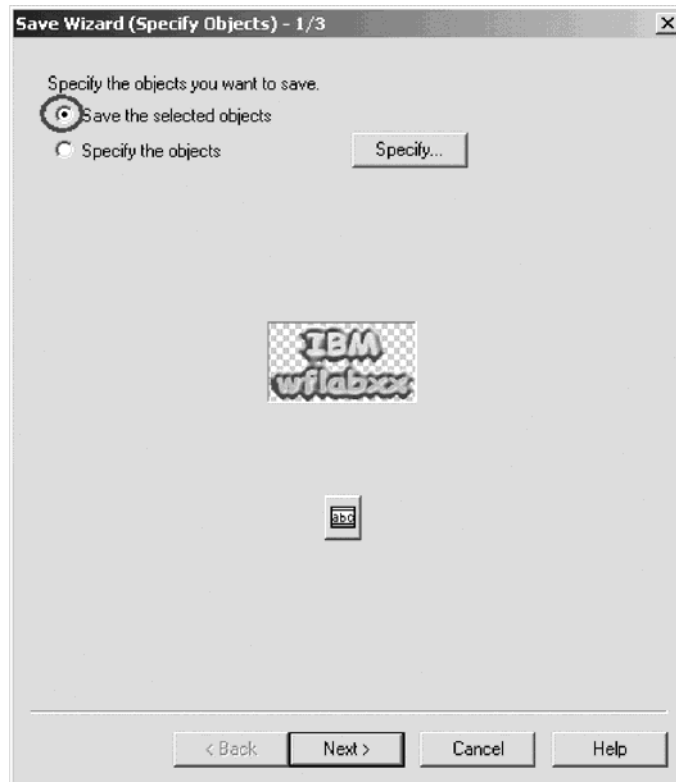
Make sure the directory is pointing to the WebContent directory in the workspace where your Web project is located.

2. In the **File name** field, type mylogo.
3. Click **Save**.

Now you need to save the object in a form that can be displayed on a Web page.

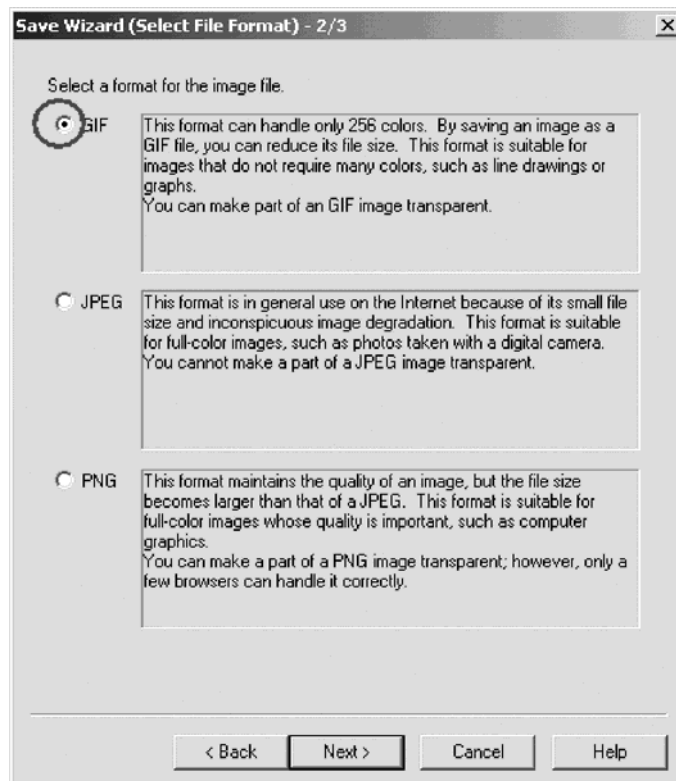
4. Click **File > Save wizard for Web** from the WebArt Designer menu.

The Save Wizard opens:



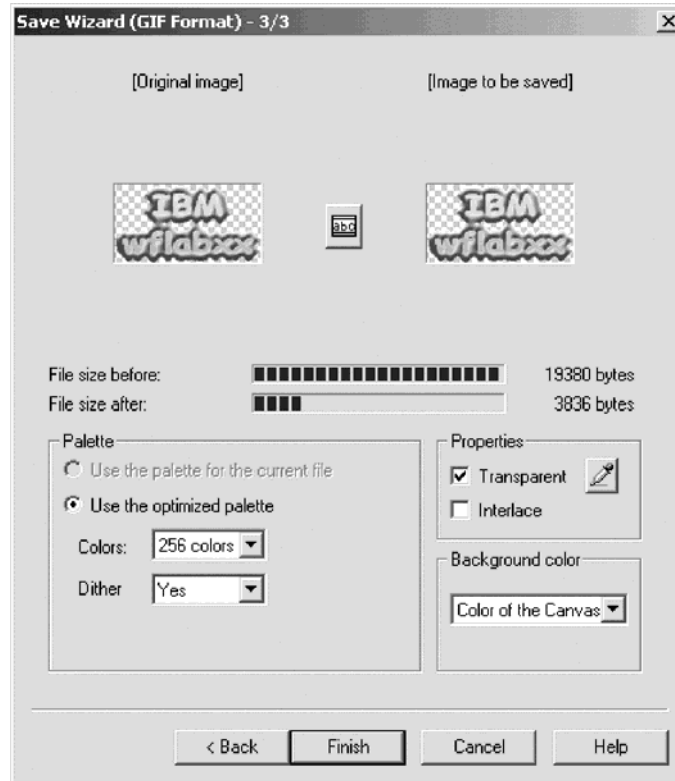
5. Click the **Save the selected object** radio button.
6. Click **Next**.

The Select File Format page opens:



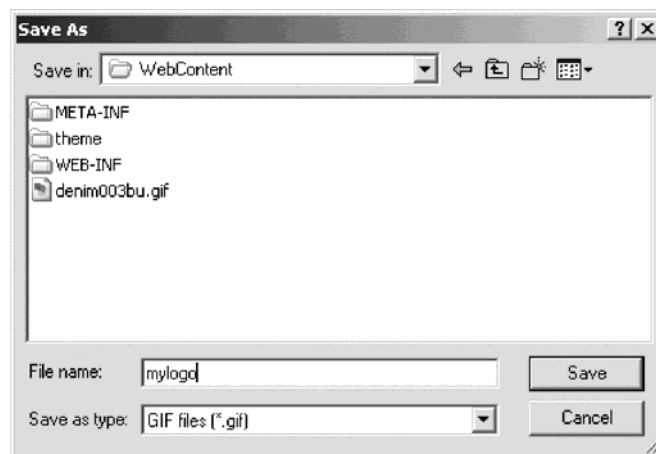
7. Click the **GIF** radio button.
8. Click **Next**.

The GIF Format page opens:



9. Click **Finish**.

The Save As window opens:



Make sure the directory is pointing to the WebContent directory in the workspace where your Web project is located.

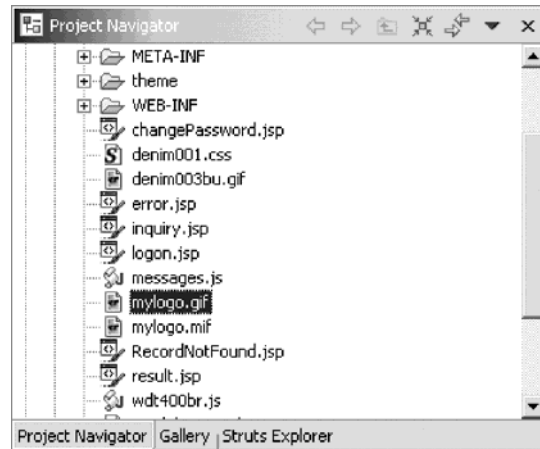
10. In the **File name** field, type mylogo.
11. Click **Save**.

Close the WebArt designer.

12. Select **File > Exit** from the menu.

You return to the workbench in the Web perspective and see the Project Navigator.

13. Make sure you switch from the Gallery view to the Project Navigator.
14. Expand the wintlabxx Web project if not already.
15. Expand the WebContent folder if not already.



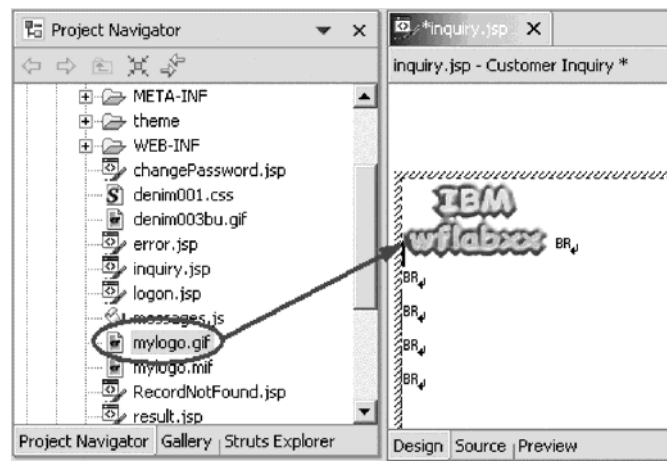
The mylogo.gif file should appear in the list as shown below. If it doesn't appear, the list might need to be refreshed.

16. Right-click the Web Project wintlabxx folder icon.
17. Select **Refresh**.

Hopefully you can see the file now in the WebContent folder, if not, go to Windows Explorer and search for the mylogo.gif on your hard drive. Move it to the WebContent folder in the Development Studio Client workspace. If you didn't use the default location as shown above, do a search for the workspace and the wintlabxx directory in it. Move the mylogo.gif file into the WebContent sub directory under the wintlabxx directory. Now you can take the logo and put it on your Web page that is still open in Page Designer. If Page Designer has been closed just open the inquiry.jsp file. Make sure you are on the Design page, not the Preview page In the Project Navigator view:

18. Select the mylogo.gif file.
19. Hold the left mouse button down.
20. Drag the file to the upper left side in the Page Designer window.
21. Let the mouse button go.

The logo is placed on the Design page:

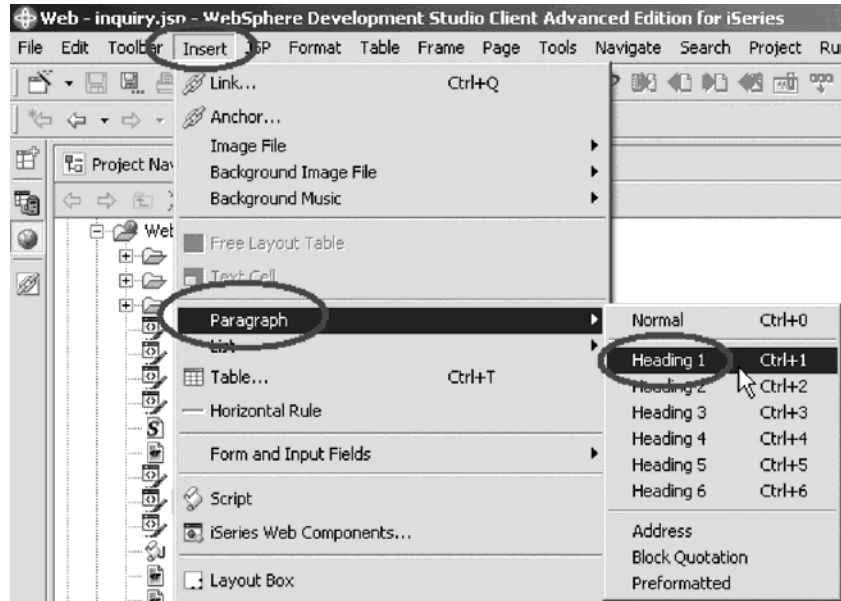


Exercise 8.5: Adding a heading 1 tag to the page

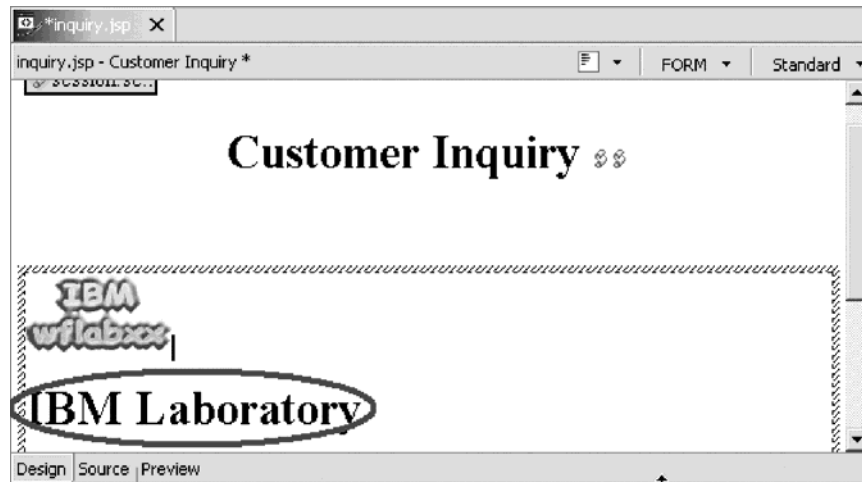
Now you want to insert a nice heading below the logo.

To add a heading 1 tag:

1. Position the cursor just below the Logo at the first BR tag.



2. Click **Insert** from the workbench menu.
3. Click **Paragraph > Heading 1** from the pop-up menu.
A frame appears that allows you to enter text.
4. Enter your company name.

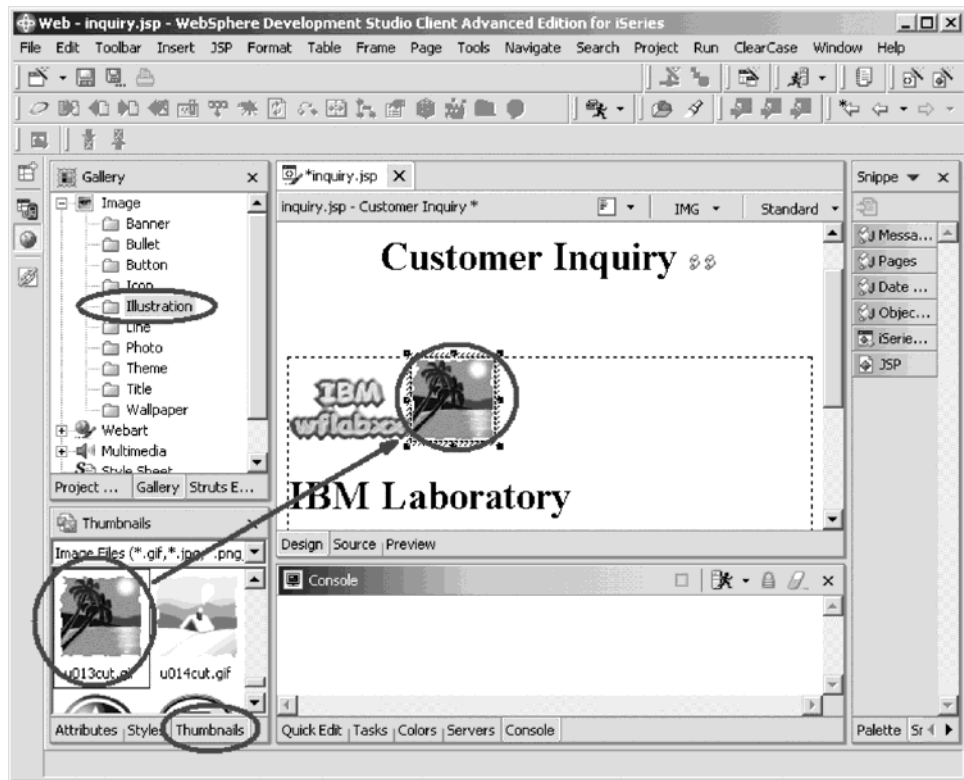


Now you want to use one of the sample pictures that comes with Development Studio Client and place this picture on the page.

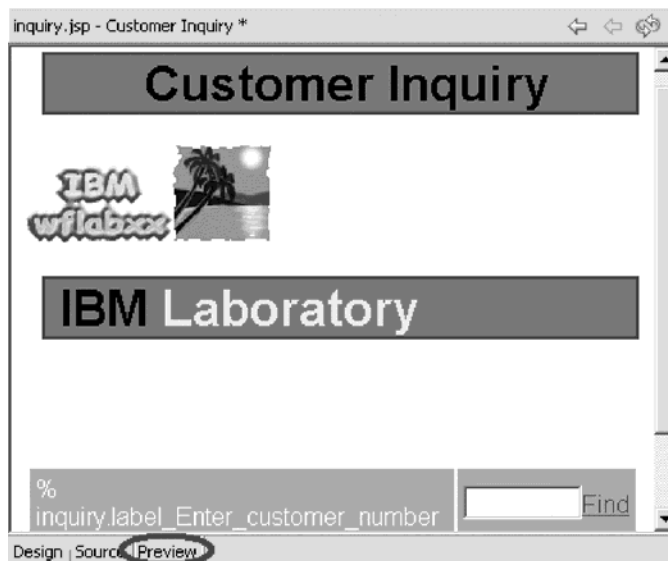
Exercise 8.6: Adding a picture to the page

To add a picture to the page:

1. In the Project Navigator, click the **Gallery** tab.



2. Expand the **Image** folder.
 3. Select the **Illustration** folder.
 4. On the Design page beside the Gallery view, select one of the sample illustrations, for example, use file u013cut.gif.
 5. Drag the picture onto the Design page beside the logo.
- The Design page now contains a picture. Click the **Preview** tab.

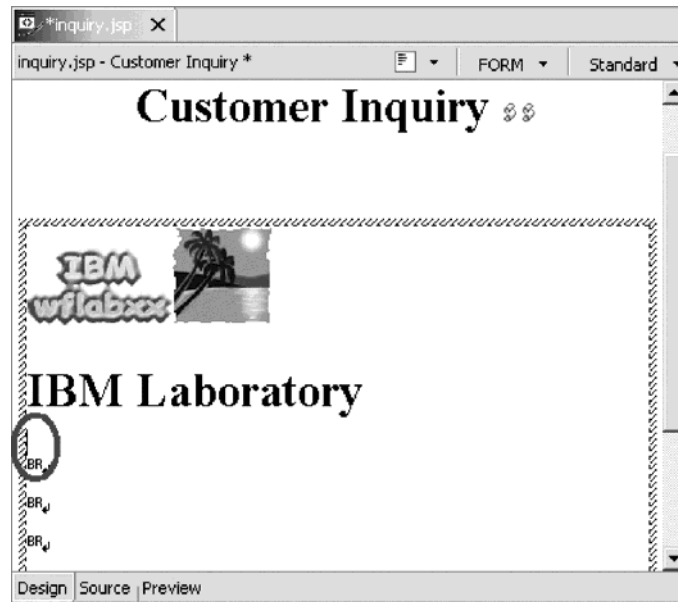


You are almost done. Next you add moving text to the page.

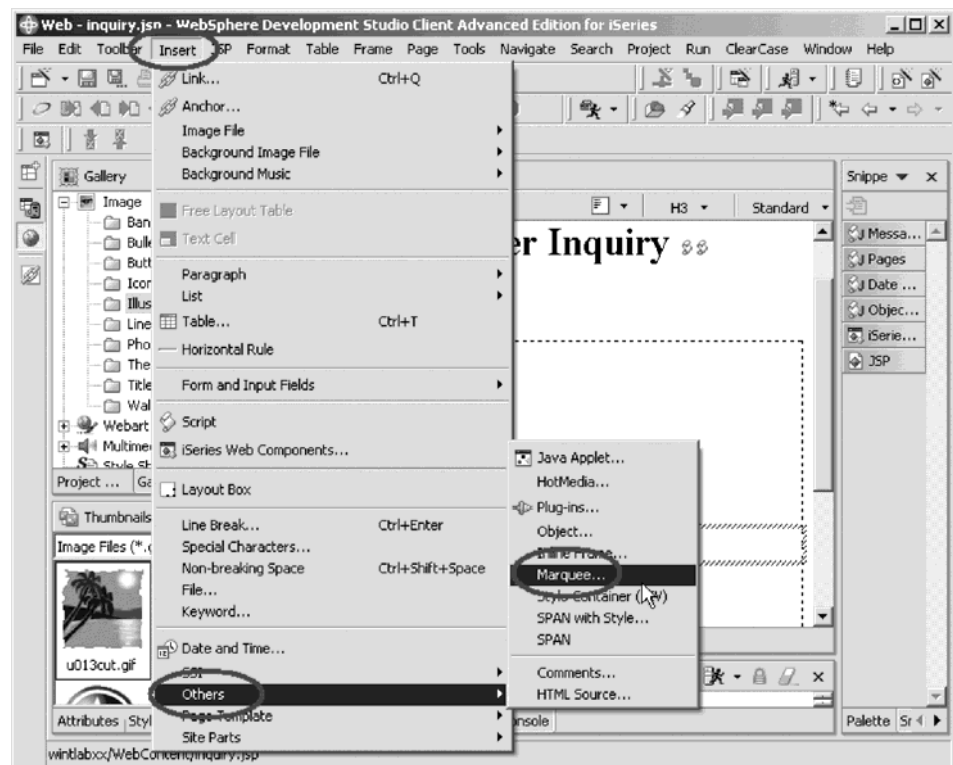
Exercise 8.7: Adding moving text to the page

Now you add moving text to the page:

1. Position the cursor in the Design page underneath the picture.
2. Click **Insert** from the workbench menu.
3. Click **Paragraph > Heading 3** on the pop-up menu.

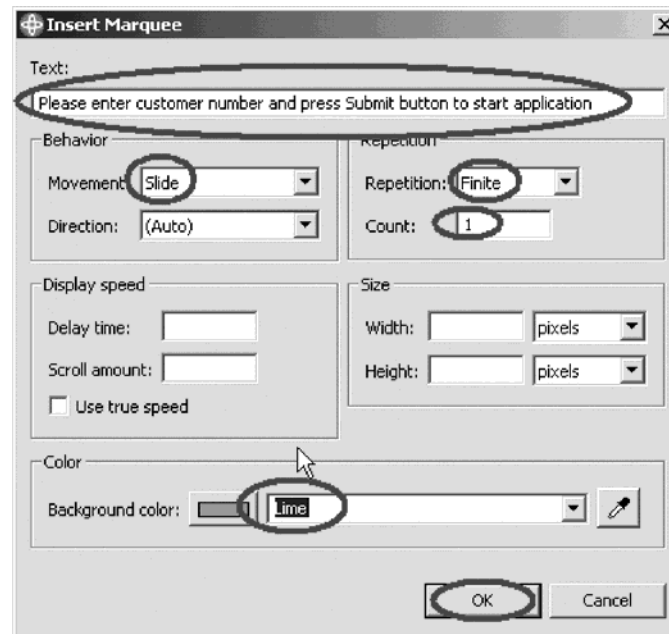


4. Leave the cursor positioned inside the heading 3 frame.
5. Click **Insert** from the workbench menu.



6. Click **Others > Marquee** on the pop-up menu

The Insert Marquee window opens:



7. Enter into the **Text** field: Please enter customer number and press Submit button to start application.
8. Select **Slide** from the **Movement** list.
9. Select **Finite** from the **Repetition** list.
10. Type **1** in the **Count** list.

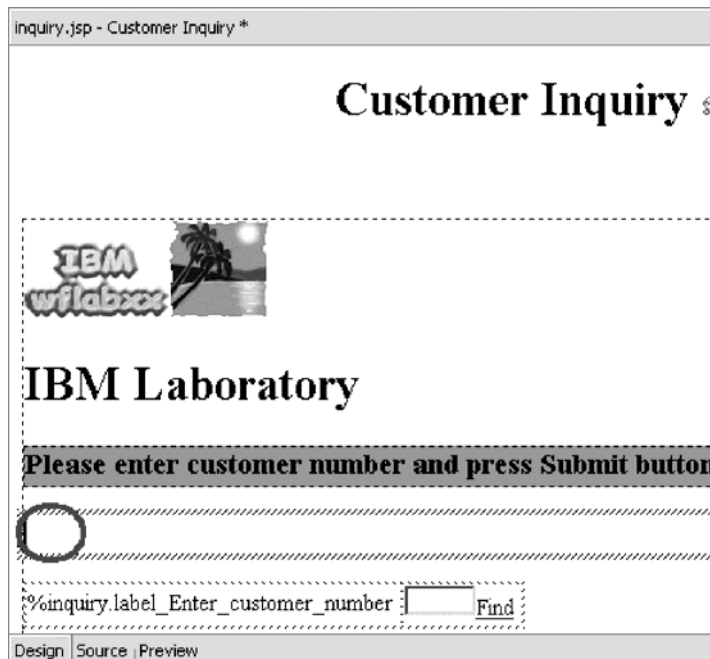
The two last selections just avoid the text sliding in forever and not standing still. If you want more movement on the page, you can change these settings.

11. Select **Lime** in the **Background color** list. Click **OK**.
12. Click **OK** again.

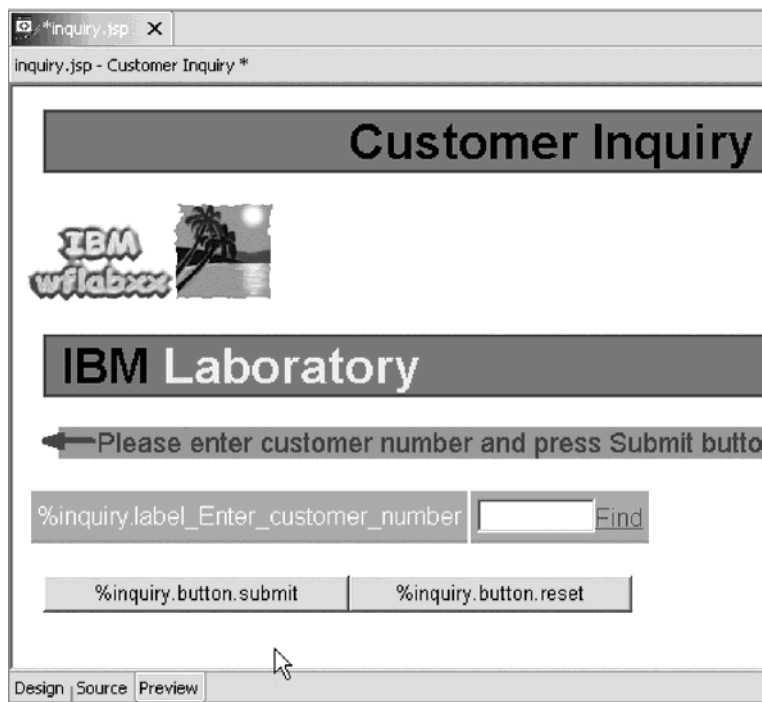
The Design page should look like:

To save some space, you can remove some of the line break tags.

13. Position the cursor on the **BR** tag.



14. Press the **Delete** key until the entry field and push buttons appear on your page.
Next you view the page as it would appear in a browser.
15. Click the **Preview** tab at the bottom of the Design page.
You will notice that your heading 3 text is sliding in:

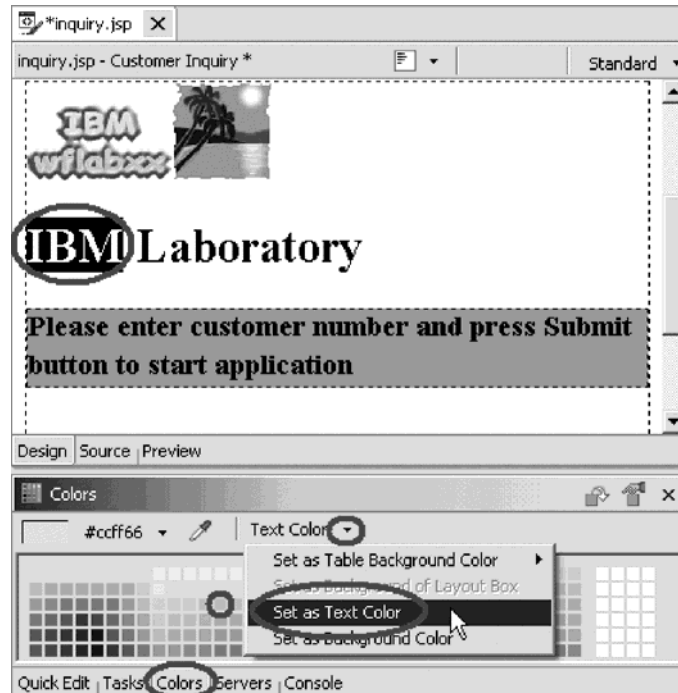


Exercise 8.8: Changing the text color

Sometimes you want to change the text color. There is an easy way to apply another color to certain areas of text. To do that you return to the Design page.

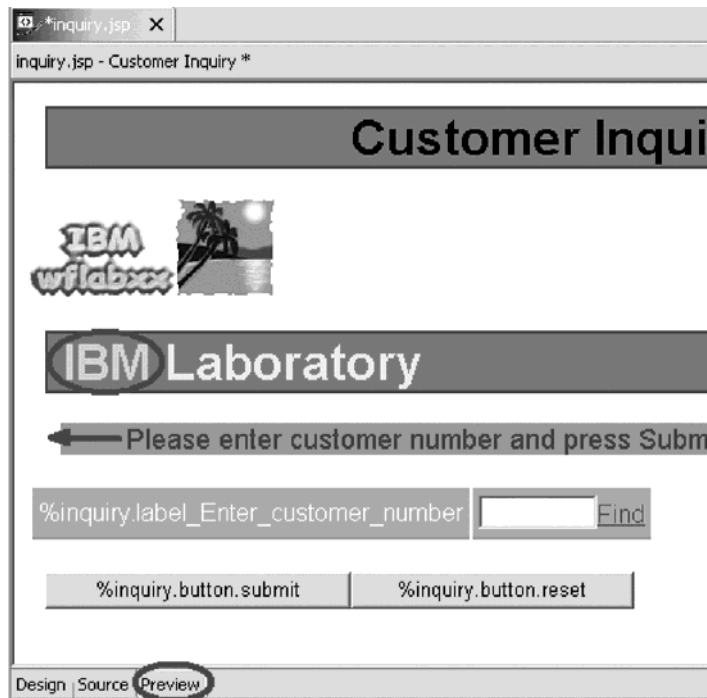
To change the text color:

1. Click the **Design** tab.

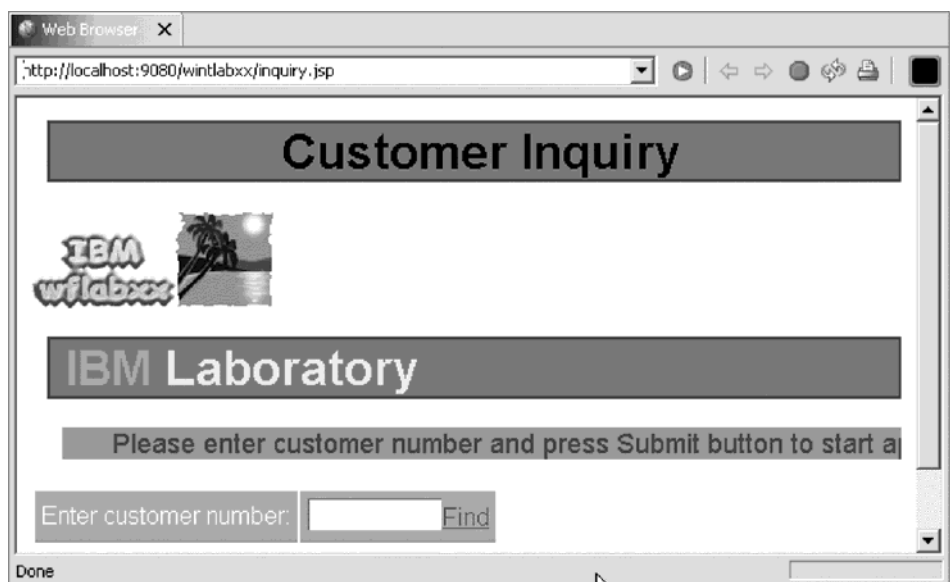


2. Select the company text (IBM) you want to change color on, by swiping the text area with the mouse cursor.
Below the Page Designer window there are several tabs.
3. Select the **Colors** tab.
4. Select a color from the Color palette.
5. Click **Text color** from the Color menu.
6. Click **Set as Text Color** on the pop-up menu.
You are done; now the selected text will be displayed with the color you selected for it.
7. Select the **Preview** tab to view your completed page.

8. Save the file inquiry.jsp.



9. Exit Page Designer.
 10. In the Project Navigator expand the wintlabxx folder if not already.
 11. Expand the WebContent folder.
 12. Right-click inquiry.jsp.
 13. Click **Run on server** on the pop-up menu.
- Your new designed input page opens and you can run your Web application.



Recap

Congratulations! You have completed Module 8: Enhancing the input page using Web tools. You should know understand:

- The purpose of Page Designer.
- How to open Page Designer.
- How to add a style sheet to a Web page.
- How to work with Web page properties.
- How to design and add a logo to a Web page
- How to add a heading 1 tag to a Web page
- How to add a picture to a Web page.
- How to create a logo with WebArt designer.
- How to add moving text to a Web page.
- How to change the text color in a Web page.

As you can see, the intent of the iSeries extensions to the Web Development Tools is to enable you to be immediately productive creating Web user interfaces on top of RPG business logic, leveraging your existing skills and potentially even existing code, and delaying the need to learn HTML, JavaScript™ and Java. These extensions include a palette of iSeries-unique JSP custom tags within the Web Page Designer, which offer attributes for formatting, validation and field referencing that you are accustomed to from your green screen development. These attributes include data type, length, decimals, formatting and validation. From these attributes, client-side JavaScript is generated. These palette parts defer the need to learn HTML and JavaScript.

The Web tools extensions also include a Web Interaction wizard, that given the signature information for an iSeries executable or entry point, will generate not only the Java bean wrapper to call it, but also a Struts action that uses it, and optionally input and output JavaServer Pages and form beans to prompt for the input parameters and display the output parameters.

A Web application that uses RPG for business logic, can be created by creating and stringing together Web interactions, where the output page from one interaction is the input page to the next interaction. When the JSPs are created first, the Web Interaction wizard allows mapping of input fields to input parameters and output fields to output parameters. This interaction wizard defers the need to learn Java, JSPs and Servlets, especially if combined with the Web Site Designer for creating page templates, preferably by someone else skilled in Web design.

Appendix. Notices

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

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

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

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

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

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

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

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

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

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

Lab Director
IBM Canada Ltd. Laboratory
8200 Warden Avenue
Markham, Ontario
Canada
L6G 1C7

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

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

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

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

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

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 1992, 2004. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

Programming interface information is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks

IBM
iSeries
OS/400
RPG/400
VisualAge
WebSphere

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

ActiveX, Microsoft, SourceSafe, Visual C++, Visual SourceSafe, Windows, Windows NT[®], Win32, Win32s and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX[®] is a registered trademark of The Open Group.

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



Printed in USA