Web Tools

in WebSphere Development Studio Client 5.1

Phil Coulthard
coulthar@ca.ibm.com

e business software

This presentation will introduce the Web Tools that are in WebSphere Development Studio Client, as of the 5.1 release.

Disclaimer

Acknowledgement:

- This presentation is a collaborative effort of the IBM Toronto iSeries Application Development presentation team, including work done by:
  - ► *Phil Coulthard, George Farr, Claus Weiss, Don Yantzi*

Disclaimer:

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

Reproduction:

- The base presentation is the property of IBM Corporation. Permission must be obtained PRIOR to making copies of this material for any reason.

This is a collaborative effort

AGENDA

E-business Primer

Intro to some technology and terms

Web Tools

In Development Studio Client

Application Server Tools

In Development Studio Client

ISeries Web Tools

In Development Studio Client

We will first introduce some terms and technology needed to understand the Web Tools.

Then we will cover the Web Tools as they are inherited from WebSphere Studio Site Developer.

We will also cover the built-in Application Server Tools, which nicely complement the Web Tools.

Finally we discuss the extensions which have been added uniquely for iSeries developers.

What is J2EE?

- **Java 2 Enterprise Edition**
- A standard, as specified by Sun Microsystems Inc
    With considerable help/input from friends like IBM
- Defines a runtime that application server vendors have to implement
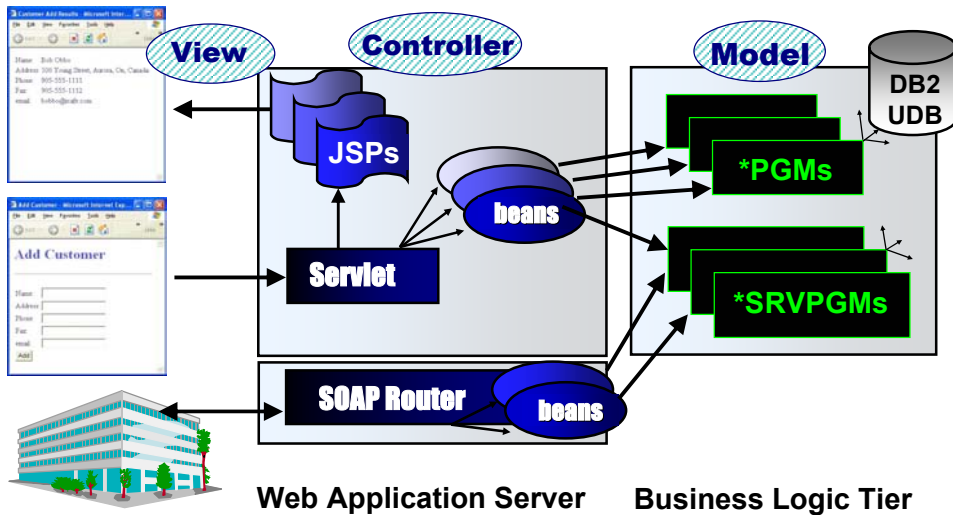   Vendors like IBM with WebSphere Application Server

- Primary requirements for a J2EE vendor:
    EJB Container for running **Enterprise JavaBeans**
      - These are an advanced topic we don't cover here
    Web Container for running **Servlets** and **JSPs**
      - These are what we talk about here, and all many customers really need

You will see we discuss here terminology that comes from the J2EE specification. While the J2EE specification covers both Servlets and JSPs, and Enterprise JavaBeans (EJBs), we are only concerned here with Servlets and JSPs, and the tooling for them.

It is not necessary to have EJBs in a J2EE application, and indeed when using simple Java, or RPG/COBOL, for the business logic, they normally are not used. They are good though when you need to build a portable and highly-scalable Web application.

IBM

Typical e-business Architecture

View **Controller** **Model**

DB2 UDB

JSPs

*PGMs

beans

Servlet

*SRVPGMs

SOAP Router beans

Web Application Server   **Business Logic Tier**

Architecture of a "Better Architecture"

Here, you can see a rather detailed architectural picture of a Web application. The second tier, known as the Web application tier, consists of a single servlet that acts as a controller deciding which business logic Java bean (not an EJB!) is to be called for a given input request. The job of the business logic Java bean is to invoke the actual business logic on the third tier, which from your perspective at this point, is an RPG or COBOL program or procedure. The user inputs are passed to the business logic and the results are returned. The controller servlet then decides what Web page (that is, what JSP) to show next and passes control to it, and at the same time, passing it the business logic Java bean that now contains the results. The results are inserted into the JSP and the resulting page is shown to the user.

For Web services, the caller invokes a generated client, which calls the SOAP router to find the appropriate Java bean to be called. Again, this Java bean is a wrapper of the business logic in the procedure or program. *[SOAP (Simple Object Access Protocol) is an XML-syntax-based messaging protocol to access Web services. It provides enterprises with more capabilities to connect systems internally and externally via the Internet. Its role is similar to that of DCOM and CORBA® distributed object systems—but is lighter weight and less programmer-intensive. SOAP is becoming more widely used to invoke services and as a messaging system over the Web.]*

What Are Servlets?

J2EE Servlet Spec 2.3

- **Servlets are . . .**
  - ► **Java classes (programs written in Java)**
- **Servlets run . . .**
  - ► **On an application server such as WebSphere**
- **Servlets are called . . .**
  - ► **By your HTTP Server software**
  - ► **When a user goes to your Web page**
- **The input to Servlets are . . .**
  - ► **User-entered data from a Web page**
- **The output of a Servlet is . . .**
  - ► **Java Bean, passed to a JavaServer Page**

Servlets are a key part of any new e-business application. So what are they? In a nutshell, they are CGI-bin programs written in Java.

Servlets are server-side Java programs that use the *Sun Microsystems Java Servlet API* and its associated classes and methods, as defined in the *Sun Microsystems Java Servlet 2.3 Specification*. These Java programs extend the functional of a Web server by generating dynamic content and responding to Web client requests. When a browser sends a request to the server, the server can send the request information to a servlet, so that the servlet constructs the response that is sent back to the browser.

Just as applets run on a Web browser and extend the browser's capabilities, servlets run on a Java-enabled Web server, such as the WebSphere Application Server, to extend the server's capabilities. Servlets are commonly used to allow businesses to connect databases to the Web, due to their flexibility, scalability, and the their processing economy when developed in the WebSphere Studio Web development environment.

IBM

What Are JSPs?

- **JavaServer Pages (JSPs) are . . .**
  - ► **.jsp files**
    - containing html tags plus JSP tags

J2EE JSP Spec 1.2

- **JSP tags . . .**
  - ► **Allow dynamic data to be inserted into the static HTML**
    - Data is extracted from Java Beans passed to the JSP
- **JSPs are called . . .**
  - ► **By your servlet**
  - ► **The input to JSPs are . . .**
    - Java Beans passed from your Servlet
  - ► **The output of a JSP is . . .**
    - A full Web page, displayed to user

JavaServer Pages are also a key part of any new e-business application. So what are they? In a nutshell, they are just like display files in that they are "side files" that contain constant information (HTML tags versus DDS text fields) and dynamic information that is inserted at runtime (using special JSP tags that extract data from a simple Java Bean, versus DDS named fields).

JavaServer Pages enable you to generate dynamic web content, such as HTML, DHTML, XHTML, and XML files, to include in a Web application. JSP files are one way that the WebSphere Studio implements server-side dynamic page content. JSP files allow a Web server, such as WebSphere Application Server or Apache Tomcat, to dynamically add content to your HTML pages before they are sent to a requesting browser. Typically, an application has a single servlet, which decides which business logic to call, and which JSPs to invoke, passing the business logic data to the servlet via a Java Bean. While you need not worry about, under the covers a JSP is converted into a servlet itself, on first touch.

This is in contrast with client-side JavaScript (within <SCRIPT> tags), which is executed in a browser. A JSP page is ideal for tasks that are better suited to execution on the server, such as accessing databases or calling Enterprise Java beans.

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.

A *Struts-based Web application*, also known as a *Struts application*, is a Web application that has Struts support added. Struts applications use a Struts framework to implement a model-view-controller design approach to building Web applications.

You use Struts to help you more easily develop applications that are divided into model/view/controller parts:

Model

> The model is the business logic, which in most cases involves access of data stores like relational databases. The development team that handles the model may be expert at writing DB2 COBOL programs, or EJB entity beans, or some other technology appropriate for storing and manipulating enterprise data.
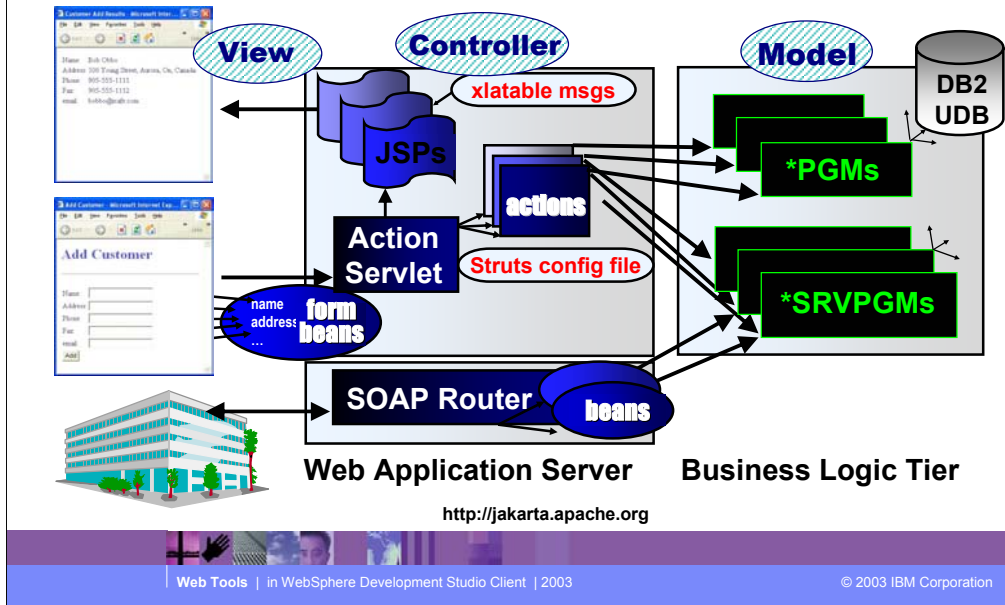
View

> The view is the code that presents images and data on web pages. The code comprises JSPs and the Java beans that store data for use by the JSPs.

Controller

> The controller is the code that determines the overall flow of events.

Here we see the view controller components of a Web application. While third tier (which is usually outside of the Web server) contains the model.

Even Better Architecture with Struts

View

Controller

Model

DB2 UDB

JSPs

xlatable msgs

actions

Action Servlet

Struts config file

*PGMs

name address …

form beans

*SRVPGMs

SOAP Router

beans

Web Application Server

Business Logic Tier

http://jakarta.apache.org

Even Better Architecture with Struts

In this enhanced version of the same Web application architecture you just looked at, a very popular open-source Web application framework, called Struts, is being leveraged. Apache's Struts Web Application Framework is becoming a standard as more companies are using it for their Web applications. It is a method for resolving Web application issues such as: Should the servlet controller perform error handling? Is there a simple way to change the panel-to-panel flow? What about internationalization? With Struts, the servlet is supplied, and whatever needs to be coded becomes an Action class in Java, which in turn, calls the actual business logic. The base Action class is supplied by Struts. The decision about which Action to invoke for a given Web page form, and which JSP to show next, is guided by a Struts config file (shown here in red text). This allows the flow to be altered without modifying the code. Struts also make it easier to build internationalized Web pages by using externally described text. (See "xlatable msgs" shown here in red text.) Struts further simplifies the processing of the user input by using "form beans" to capture all the user input from the input page (shown here in red text). For each input field in the input page, the data is copied to a property in the bean with the same name as the field. This bean is then passed to the Action, which can further pass it along to the business logic.

The iSeries Web tools include all the necessary tooling for Struts and will generate everything in the Web application tier, after you pictorially "layout" the application flow. Again, this allows the iSeries developer to focus on the RPG or COBOL business logic, using existing programming skills.
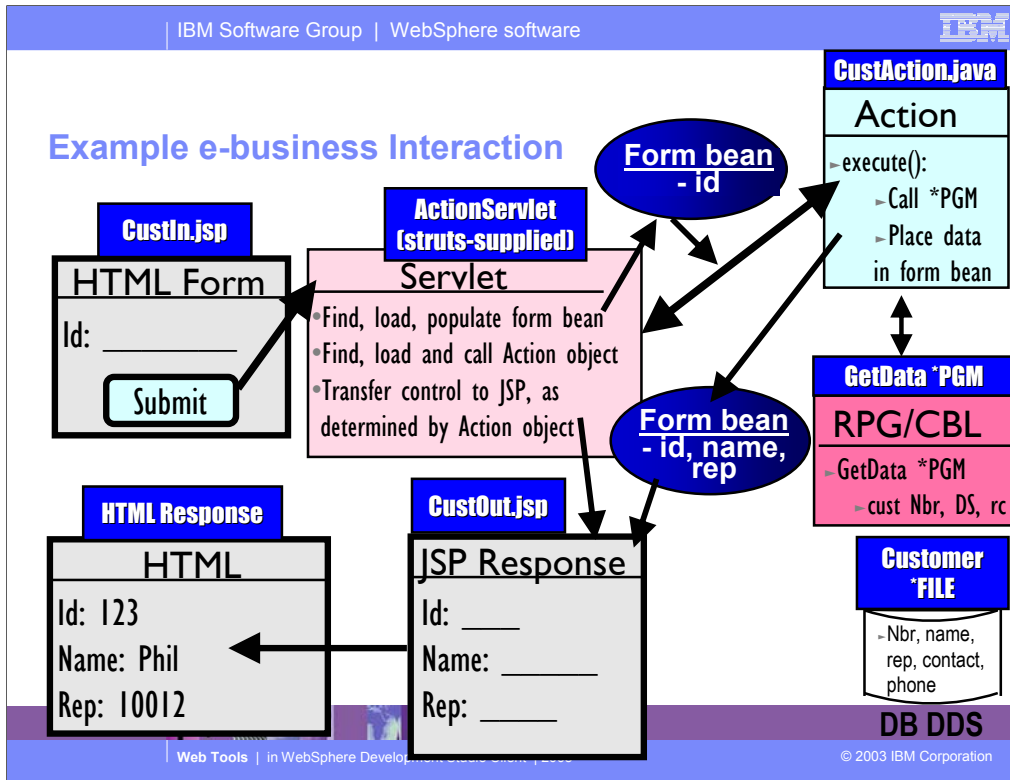
Here we show an example of a typical interaction in a Web application, where the logic is written in RPG.

#1. Customer is presented with a Web page that prompts for a customer Id. Any time a Web page has input fields, those fields exist within an HTML form tag, which identifies the server-side logic to call. For J2EE applications, this is a servlet.

#2. When the Submit button is pressed in the form, the servlet is invoked. Remember, for Struts the servlet is supplied by the framework. The servlet's job is to read the user-entered input, and place into appropriate form bean for the form, and then optionally call the validate method on the bean to verify the user input is valid. The servlet then determines which action to call for this form, and calls the action, passing the form bean. Which form bean and action to use for a given input-capable web page is determined via the Struts configuration file.

#3. The action code (which is still in java) then invokes the business logic, which in our case is RPG or COBOL logic on the 3rd tier. The user specified customer ID is passed as the input parameter. The program updates output parameters with the customer information data structure, and a return code. The action then determines which JSP to show next, and prepares the form bean for that JSP with the data returned from the program.
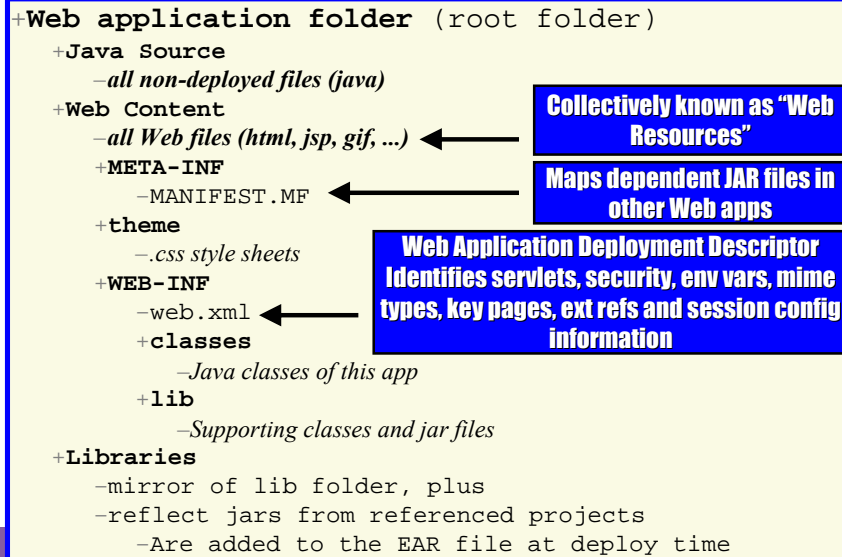
#4. The action returns control to the servlet, telling which JSP to show next. The servlet then invokes that JSP. The JSP uses tags to find the prepared data and inserts in into the page.

#5. The resolved HTML page is returned to the Web browser.

Terms: Web Application

● Web App folder structure:

J2EE Servlet Spec 2.3

```
+Web application folder (root folder)
  +Java Source
    –all non-deployed files (java)
  +Web Content
    –all Web files (html, jsp, gif, ...)  ◄─── [Collectively known as "Web Resources"]
    +META-INF
      –MANIFEST.MF  ◄─── [Maps dependent JAR files in other Web apps]
    +theme
      –.css style sheets
    +WEB-INF       [Web Application Deployment Descriptor
      –web.xml  ◄─── Identifies servlets, security, env vars, mime
                     types, key pages, ext refs and session config
                     information]
      +classes
        –Java classes of this app
      +lib
        –Supporting classes and jar files
  +Libraries
    –mirror of lib folder, plus
    –reflect jars from referenced projects
      –Are added to the EAR file at deploy time
```

The Web application will run in an application server. The application server expects the application to conform to certain standards in the way its files are stored. Here you see the standard for the current J2EE Specification supported by the current release of WebSphere Application Server. The Web tools in WebSphere Studio will generate and work with Web applications conforming to this standard.

Terms: Web Application

●Example Web Application

```
+accounts
   +Java Source
   +Web Content
      -index.html          www.mydomain.com/accounts
      +receivable
         -page1.html     www.mydomain.com/accounts/receivable/page1.html
      +payable
         -page1.html     www.mydomain.com/accounts/payable/page1.html
      +META-INF
      +theme
         -corporate.css
      +WEB-INF
         -web.xml
         +classes
         +lib
   +Libraries
      -j2ee.jar
```

2003 IBM Corporation

Here is an example of the Web application folder structure and where in this folder structure the files of a Web application have to be stored.

The default directory structure in the Web project adheres to the J2EE specification for Web modules. (Web modules are also known as Web applications, in servlet programming.) This specification defines a project directory structure that specifies the location of Web content files, class files, class paths, deployment descriptors, and supporting metadata.

Terms: WAR Files

● **Web Archive Files (WAR)**

► **One file containing**
- Whole folder structure of Web application
- Including web.xml file
- Optionally including source

► **Used to**
- Install and configure Web application in an application server

J2EE Servlet Spec 2.3

```
+accounts
    +Java Source
    +Web Content
        -index.html
        +receivable
            -page1.html
        +payable
            -page1.html
        +META-INF
        +theme
            -corporate.css
        +WEB-INF
            -web.xml
            +classes
            +lib
    +Libraries
        -j2ee.jar
```

MyWebProject.war

A Web application is a group of HTML pages, JSP pages, servlets, and other resources, along with source files, optionally, that can be managed as a single unit. A Web archive (WAR) file is a packaged Web application. WAR files can be used to import a Web application into a Web server.

In addition to project resources, the WAR file includes a Web deployment descriptor file. The Web deployment descriptor is an XML file that contains deployment information, MIME types, session configuration details, etc., for a Web application. The Web deployment descriptor file (web.xml) provides information about the WAR file that must be shared with the developers, assemblers, and deployers in a J2EE environment.

Terms: EAR Files

● Enterprise Archive Files (EAR)

► One file containing:
- Zero or more Web Archive (war) files
- Zero or more EJB jar files
- A J2EE deployment descriptor (**application.xml**)

► Used to install and configure:
- All pieces of a J2EE <u>Enterprise Application</u>
  - Web application plus EBJs plus EJB clients
- All Web applications for a Web site (say)

J2EE Servlet Spec 2.3

```
+accounts
+accounts
+accounts
    +Java Source
    +Web Content
        -index.html
        +receivable
            -page1.html
        +payable
            -page1.html
    +META-INF
    +theme
        -corporate.css
    +WEB-INF
        -web.xml
        +classes
        +lib
    +Libraries
        -j2ee.jar
```

MyEntApp●ear
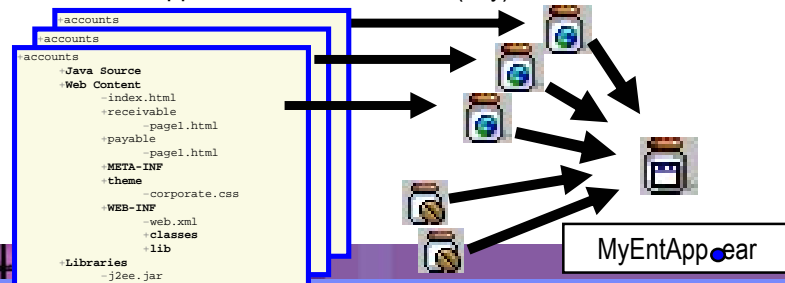
Enterprise Application projects are exported as EAR (enterprise archive) files that include all files defined in the Enterprise Application project as well as the appropriate module archive file for each J2EE module project defined in the deployment descriptor, such as Web archive (WAR) files and EJB JAR files.

AGENDA

E-business Primer

Intro to some technology and terms

Web Tools

In Development Studio Client

Application Server Tools

In Development Studio Client

ISeries Web Tools

In Development Studio Client

Now that the brief terminology primer is done, we turn our attention to the Web Tools as we inherit them from WebSphere Studio Site Developer or WebSphere Studio Application Developer, which WebSphere Development Studio Client and WebSphere Development Studio Client Advanced are based on, respectively.

These Web Tools are generic, in that they apply to both iSeries and non-iSeries programmers. Later we will see the iSeries extensions that are added on top of these tools to optimize them for iSeries development.

Web Tools

- **WDSc Web Tools At A Glance:**
  - ► **Web Projects and Perspective**
  - ► **Web Site Designer and Page Templates (NEW IN 5.1!)**
  - ► **Web Editors**
    - ▪ For JSP and HTML files, Cascading Style Sheets, logos, images, and animation
  - ► **Link viewing and management**
  - ► **Import/Export**
    - ▪ Numerous formats
  - ► **Wizards**
    - ▪ for Web pages from DB or JavaBean
  - ► **Debugging**
    - ▪ Of JSPs, Servlets and Java
  - ► **WAS Test Environment and Application Server support**

• Web project creation, using the J2EE-defined hierarchy, or a static version that reduces project overhead when dynamic elements are not required. Static Web projects can later be converted to J2EE Web projects.

• New tools for helping design Web sites with consistent-looking web pages, each which implement an common template.

• JSP and HTML file creation, validation, editing, and debugging

• JavaScript editing and validation

• Image editing and animation

• Cascading Syle Sheet (CSS) editing support

• HTTP/FTP import

• FTP export (simple resource copy) to a server

• WAR file import, export, and validation

• Link viewing

• Link parsing, validation, and management, which includes converting links, flagging broken links, and fixing up links as linked resources are moved or renamed

• Generation of Web pages using wizards that create Web resources from database (SQL) queries and beans

• Integration with the WebSphere unit test environment

• Publishing support for multiple Web server types
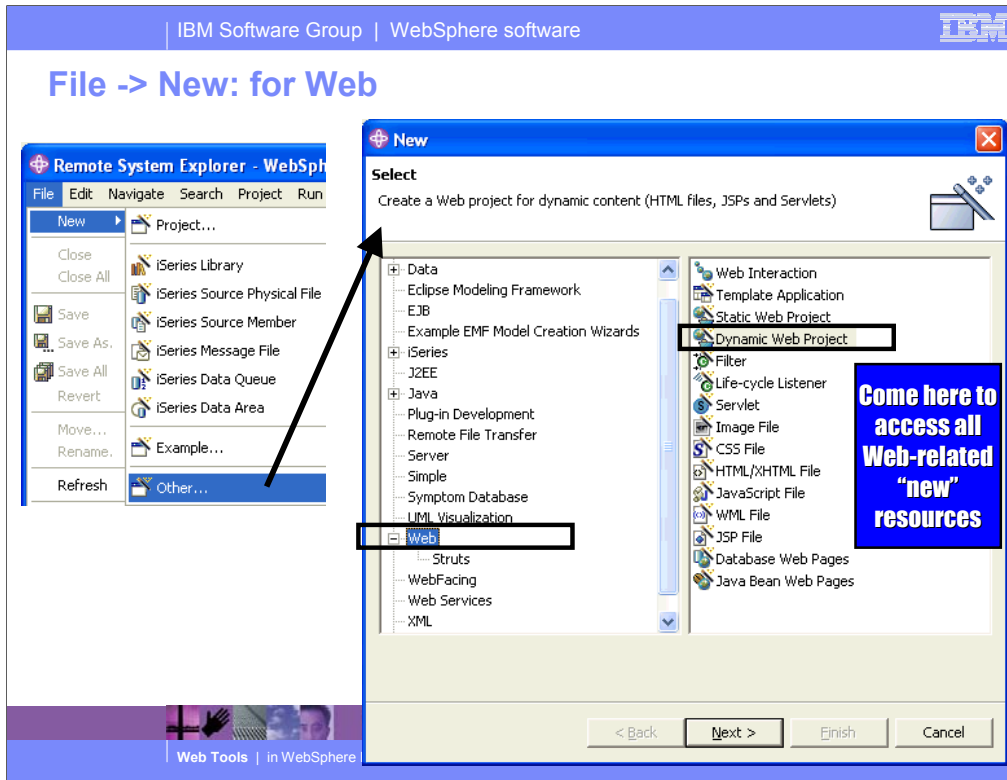
Click **File > New > Project**.

Select **Web** in the left pane of the wizard, select **Dynamic Web Project** in the right pane, and then click **Next** to open the **Create a Web Project** wizard.

Web Projects

- **Web Projects**
  - ► **Special type of project**
    - With its own "new" wizard
    - With its own perspective
    - With its own tools
  - ► **Created with J2EE folder layout**
  - ► **Created with simple web.xml file**
    - Automatically updated as resources are created
    - Has specialized **web.xml** editor

- **File->New->Other...**
  - ► **->Web->Dynamic Web Project**

A Web Project is a standard eclipse project, in that it contains folders and files and can be shared by a team. But it also has unique capabilities and tools, and a dedicated perspective to make it easy to access these capabilities and tools.

A Web project is created with the required layout for a J2EE web application, and with an initial Web.xml file which the J2EE specification insists be supplied to describe the capabilities of the application. This configuration file will be automatically updated for you as you create things in the project, although you can also edit it with the dedicated web.xml editor, if you desire.

You create a new Web project by selecting File->New->Other… from the menu, and then Web on the left, and Dynamic Web Project on the right, as we see next…
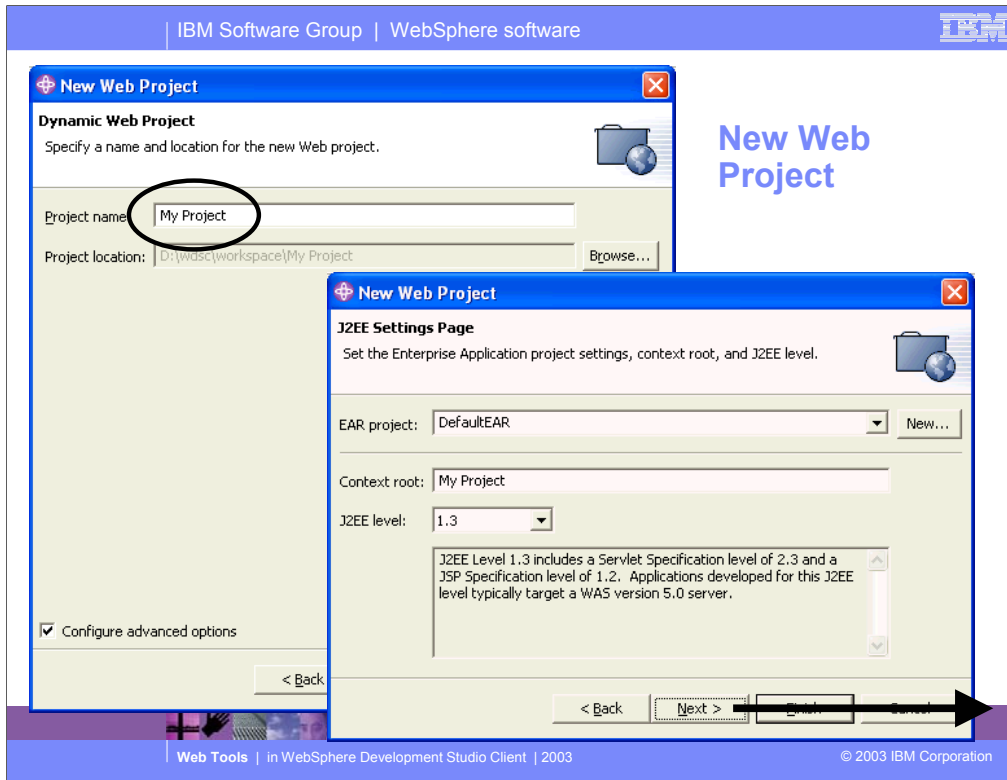
This shows the first and second pages of the New Web Project wizard.

First page: enter a project name, and optionally where to put that project in the file system.

Second page: select the EAR file project to hold the Web project. We haven't discussed an EAR file project yet, so you will just take the default. Indeed, you will normally take the defaults for the next items as well.

New Web Project: Page 3

Select Struts support

Select iSeries components

Change later using Properties dialog for the project

New Web Project

Features Page
Select a Web Project feature. A feature can provide additional functionality for the Web Project.

Web Project features:
- ☑ Add Struts support
- ☑ Default style sheet (CSS file)
- ☐ Default synchronization policy for CVS re
- ☑ iSeries Web Components Tag Library
- ☐ JSP Tag Libraries
  - ☐ JSP Standard Tag Library
  - ☐ Tag Libraries for accessing JSP objec
  - ☐ Tag Libraries for database access
  - ☐ Tag Libraries for internationalization
  - ☐ Utility Tag Libraries

Description:
Select this feature to have support for Struts added to your project

< Back   Next >   Finish   Cancel

Web Tools | in WebSphere Development Studio Client | 2003   © 2003 IBM Corporation

This shows the third page of the New Web Project wizard.

Third page: specify features to enable in your new project. This is just a short-cut, you can always add these later via the Properties dialog for the Web Project. Typically, you will select to Add Struts support and iSeries Web Components Tag Library. If using CVS for team support, then you would also likely select the Default synchronization policy for CVS repository.

New Web Project: Page 4

New Web Project

Struts Settings
Select the initial settings for Struts

☑ Override default settings

Specify Struts information if desired

Settings

Struts version: 1.1

Default Java package prefix: myproject

☑ Create a Resource Bundle for the Struts Project

Resource bundle

Java package: myproject.resources   Browse...

Resource bundle name: ApplicationResources

Translatable strings will go here

< Back   Next >   Finish   Cancel

This shows the fourth page of the New Web Project wizard.

Fourth page: if you selected to enable struts support on page three, here you will be given the opportunity to change information about the struts support. The defaults are shown here, and you rarely will want or need to change them.

New Web Project: Page 5

New for 5.1: Page Templates!

This template will be applied to all new pages created via the Web Site Designer

New Web Project

Select a Page Template for the Web Site

Select a default Page Template for the Web Site. This Page Template will be applied to new pages created using Web Site Designer.

☑ Use a default Page Template for the Web Site

Page Template Type
◉ Sample Page Template   ○ User-Defined Page Template

IBM supplied or user defined

Page Template File

Location: [   ] Browse... ▾

File name: A-02_blue.htpl

Thumbnail:

A-02_blue.htpl   A-02_gray.htpl   A-02_green.htpl

< Back   Next >   Finish   Cancel

This shows the fifth and final page of the New Web Project wizard.

Fifth page: this was new for 5.1, and it enables you to specify a default page template. This is only used if you happen to use the new Web Site Designer editor, and create pages from there. Even then, you can always change the template used for any page, no matter how created, by using the popup menu for pages selected in the Web Site Designer.
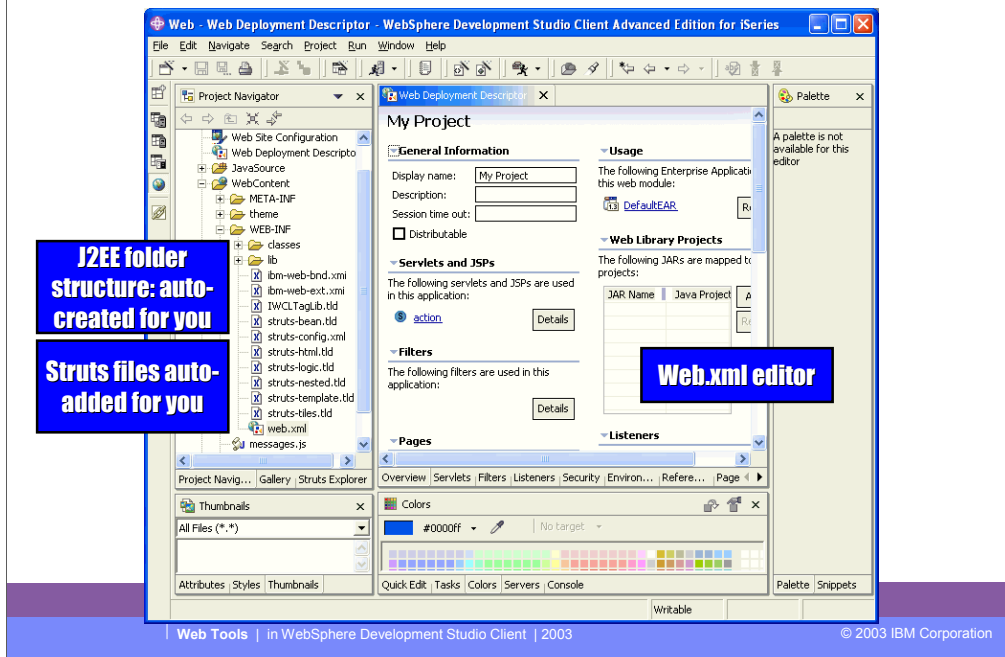
This shows the output of the New Web Project wizard.

When the project is created, the Web perspective opens with the project in the Project Navigator view. The new project will reflect the J2EE folder structure that specifies the location of web content files, class files, classpaths, the deployment descriptor, and supporting metadata. A Web project's web.xml file is used in building a WAR file from a project and provides information necessary for deploying a Web application module. Whenever you create a new Web project, a minimal web.xml file is created in WEB-INF under the project's Web Content folder.

If you chose to create a Struts web project, the necessary Struts files are pre-supplied as well. Similarly for iSeries Web Components.
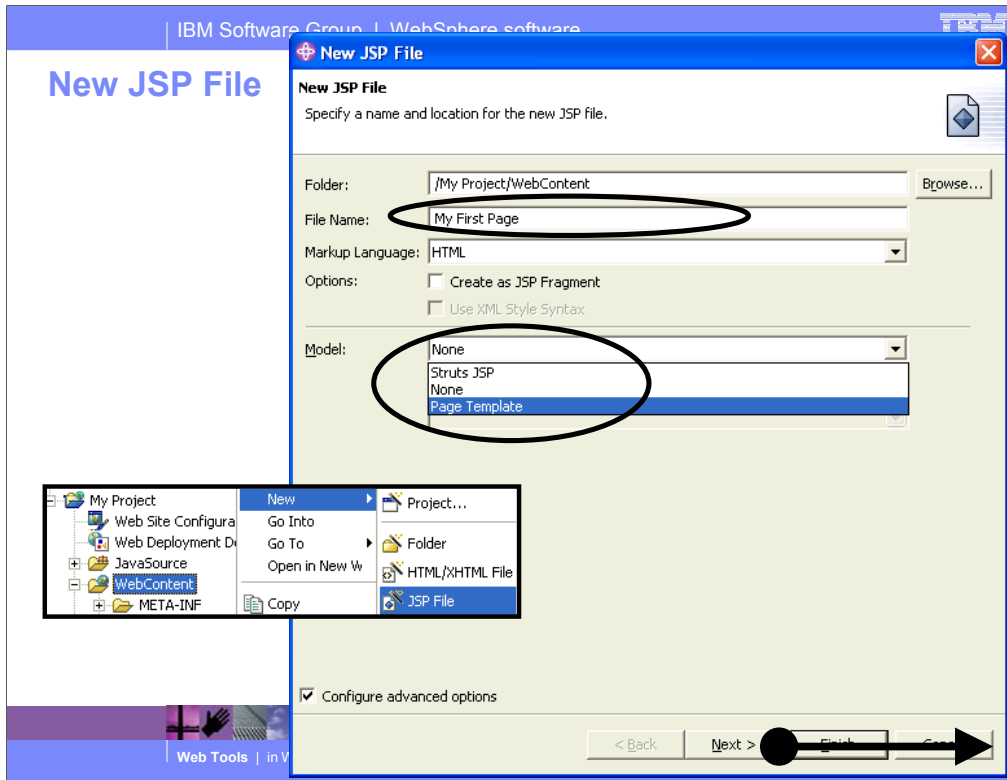
New JSP File

Here we see the wizard for creating a new JSP file, something you will likely do immediately and often.

First, to launch the wizard select the **WebContent** folder of the project, and then select **New->JSP File** from the popup menu.

In the first page of the wizard, enter the **File Name** to give the file (minus the .jsp extension) and the **Model**, which will normally be **Struts JSP** if creating a Struts application.

There are numerous other pages in the wizard, which you can see only if you select "Configure advanced options". However, normally you can just press Finish at this point.
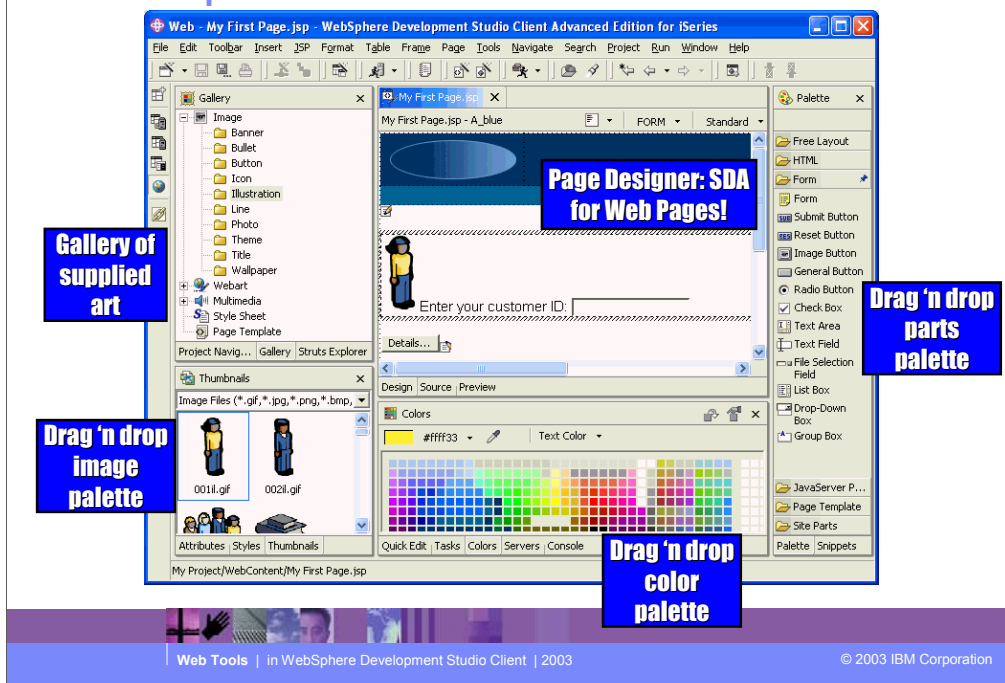
The New JSP File wizard creates your JSP file and opens it in the Page Designer editor.

The Page Designer allows you to work with HTML files, JSP files, and embedded JavaScript. Within the Page Designer, you can move among three pages that provide different ways for you to work with the file that you are editing. You can switch pages by clicking on the tabs at the bottom of the editor pane. These pages work in conjunction with the workbench Outline and Properties views, tool bar buttons, menu bar options, and context (right-click) menus. It also works together with the Palette/Snippet views on the right, the Quick Edit and Colors views on the bottom, and the Gallery, Attributes, Styles and Thumbnails views on the left.

The Design page is the WYSIWYG environment that enables you to create and work with a file while viewing its elements on the page. The Source page enables you to view and work with a file's source code directly. Preview shows you how the current page is likely to look when viewed in a Web browser.

Gallery - Contains a variety of catalogs of reusable files that can be applied to Web pages. The file types available include images, wallpaper, Webart, sound files, and style sheet files.

Thumbnail - Shows thumbnails of the images in the selected project, folder, or file. This view is especially valuable when used in conjunction with the Gallery view to add images from the artwork libraries supplied by WebSphere Studio to your page designs. You can drag and drop from this view into the J2EE Navigator view or the Design or Source page of Page Designer.

Palette - Show a number of palettes of parts which can be dragged and dropped onto the Web page.

Snippets - Shows a number of palettes of re-usable JavaScript scripts which can be dragged and dropped onto the Web page. This is also where you get the iSeries Web Components palette, as we will see.

Colors - Shows many colors, which you can drag and drop the Page Designer to change a color.

AGENDA

Web Tools

Web Site Designer

Now we drill down a bit, into the new Web Site Designer capability introduced in the 5.1 release.

Web Site Designer

- **New in 5.1 !**

- Allows you pre-define which pages site (project) will contain, and their relationship to each other

  The pages are then be iteratively created

  Each are given the page template associated with the project

- The relationships are then used to populate a site map and navigation bar

  These are components you can drag and drop onto your page template or any page

The purpose of the Web Site Designer is to enable you to build a Web site where each page is defined using a page template, so that every page looks the same and can be quickly and easily changed by merely changing the template.

In addition to this, it allows to visualize the high level view of the website, such that a web site map and navigation bar can automatically created for the site (or added to your own page template).
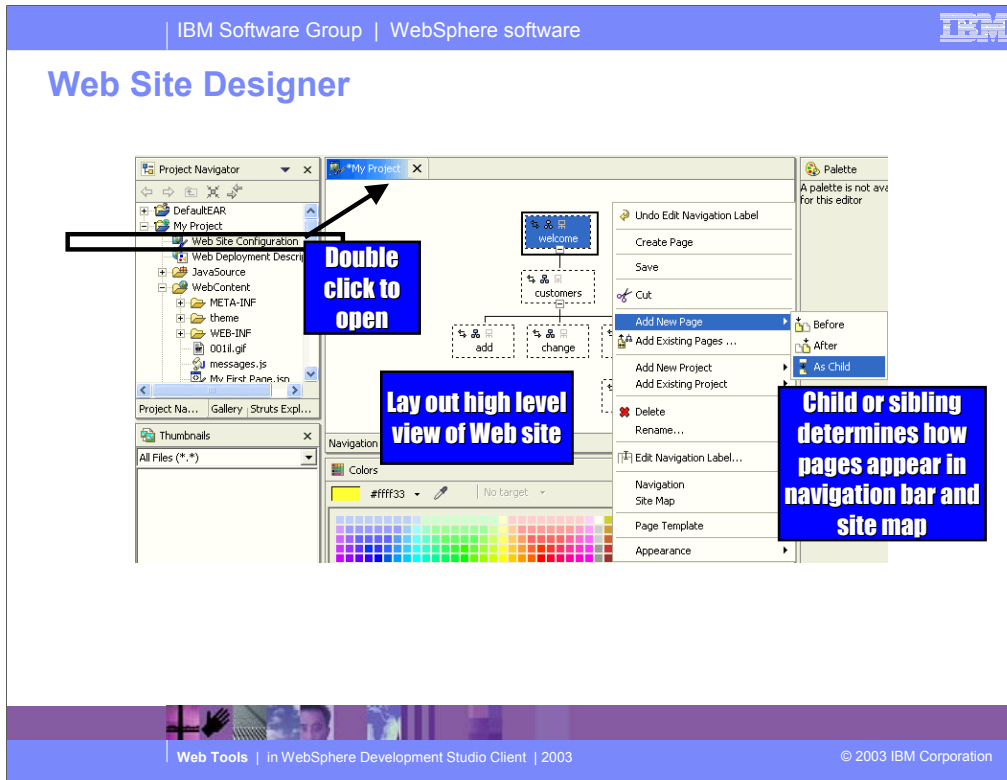
This is the Web Site Designer. You get to it by double clicking on the Web Site Configuration entry in the Project Navigator view.

You use the popup menu to add nodes representing Web pages. For each page you decide if it's a child or a sibling of the selected page, from the user's conceptual point of view. This allows the navigation bar and site map to be automated. You can further refine this using the Navigation and Site Map cascading menus in the popup menu for a selected Web page.

Once you've laid out a Web site, it does not actually exist. You still have to actually create the Web pages, and then of course put in all the logic behind to drive the pages and do the links.

Normally you will not start with this though. Normally you will use the Web Diagram Editor to layout your screens and their flow (actual flow, not conceptual) and then use the wizards there to realize or create the Web pages and struts artifacts. Later you can come here to the Web Site Designer and use Add Existing Pages to tell the tool how to build the site map and navigator bar. You will also use the Page Template popup menu item to specify the page template to assign to all the pages created in the Web Diagram Editor.

Web Site Designer

Make pages "real" by creating them

Depending on template chosen, you get a navigator bar and site map for free!

© 2003 IBM Corporation

Here we see how to start creating actual Web pages from the nodes in the Web site designer: right click and select Create Page. This takes you into the Page Designer for the new JSP file. If you chose a default page template in the New Web Project wizard, it will be applied immediately as the page is created this way.

As pages are created, in the Web Site Designer, their boxes appear with solid lines around them versus dotted lines.
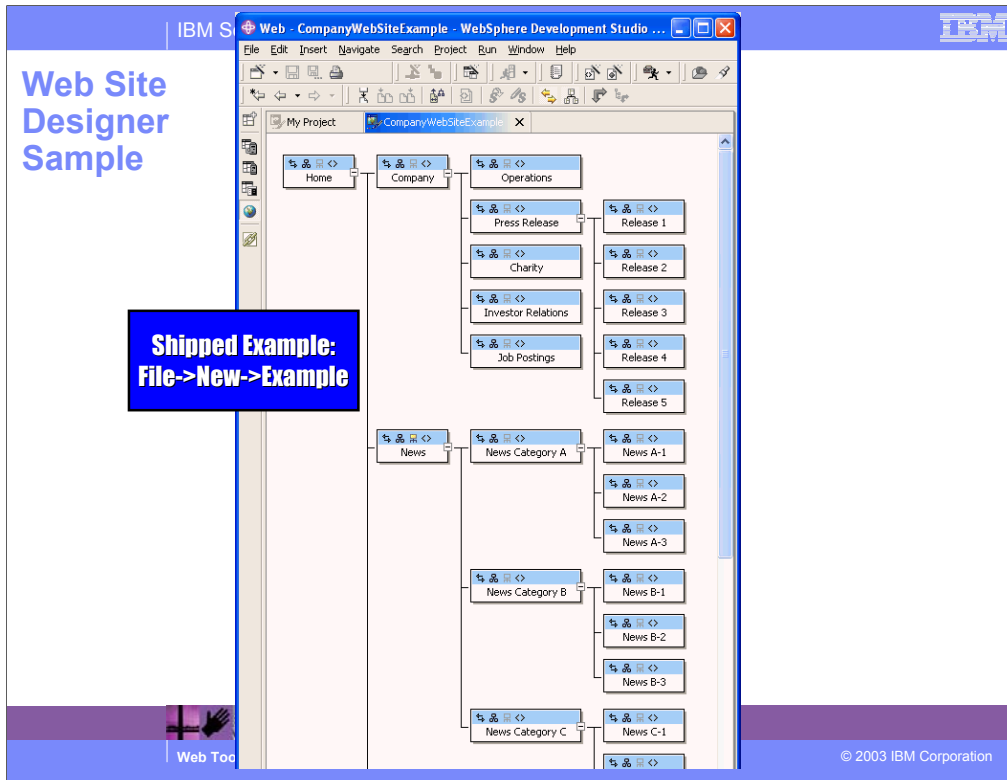
Here is an interesting example web project in the Web Site designer. You can create this yourself by selecting Examples from File->New.
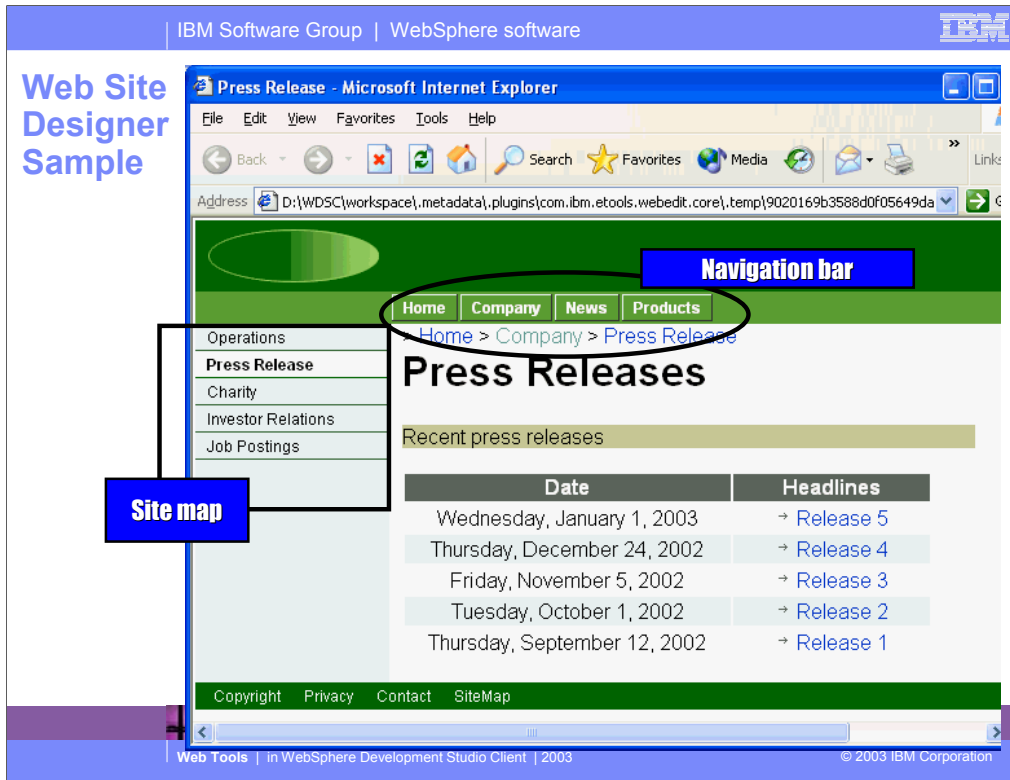
Here is one of the pages in that Web Site designer example, showing the navigation bar at the top and the site map on the left, as supplied for free in the chosen page template.

Unfortunately, this example is for a static web site, not a dynamic site.

AGENDA

Web Tools

Database Web Pages Wizard

Now we drill down into the Database Web Pages wizard that comes with the Web Tools.

Database Web Wizard

- ## Database Web Pages Wizard
  - ► **Input: SQL Query Statement (sub-wizard)**
  - ► **Output:**
    - Input JSP page prompting for input parameters
    - Output JSP page displaying output data
    - Servlet that ties them together

Web Interaction
Template Application
Web Project
Servlet
Filter
Image File
CSS File
JSP File
JavaScript File
Life-cycle Listener
HTML/XHTML File
Database Web Pages
Java Bean Web Pages
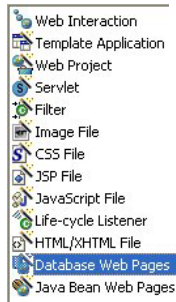
To launch the Database Web Pages wizard, select **File -> New -> Other**. Then select the **Web** option, **Database Web Pages**, and then the **Next** button. You can create an input page that collects information to filter the SQL statement, and design the result and detail pages that will format and display the resulting filtered statements to users.

Database Web Wizard
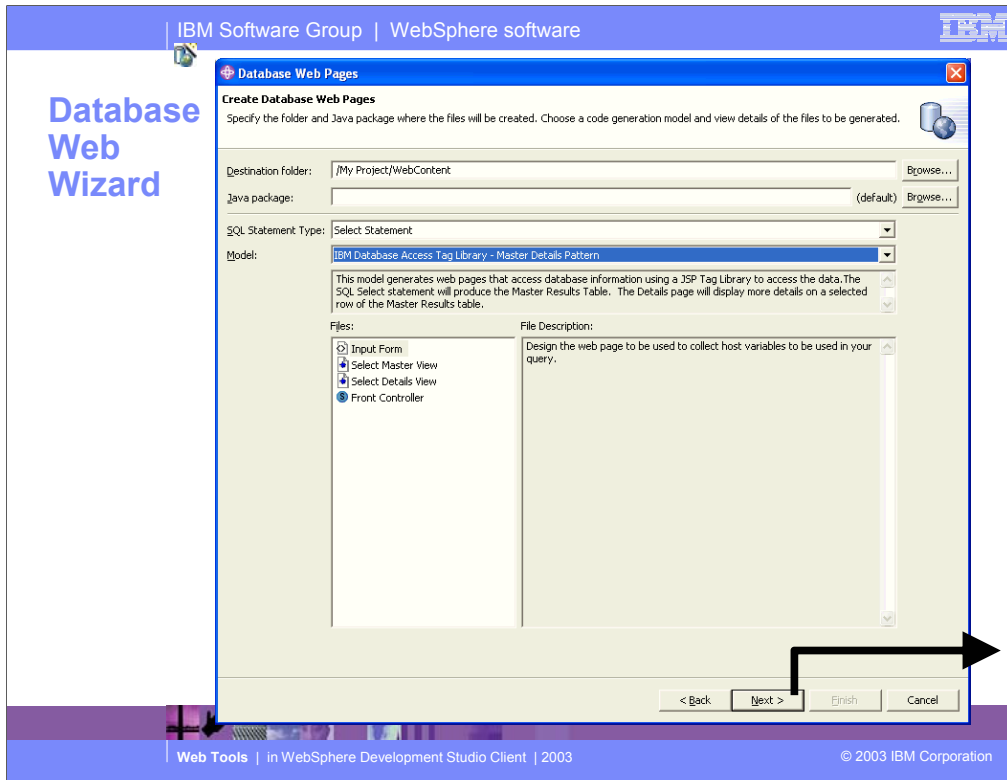
Specify the location into which the generated JSP and HTML files will be placed, as well as where all servlets will be mapped. You should select the **Web Content** folder, or one of the folders under Web Content. The **Java package** name specifies where all Java source code will be placed. A default package is assumed if you do not designate a package name.

Select one of the SQL statement types from the **SQL Statement Type** list box.

Select one of the generation models from the **Models** list box. The **Description** area provides information about the generation model that you have selected. Based on the statement type and generation model combination, the **Files** list box lists the pages that will be generated. If you click on a file, a short description is displayed in the **File Description** box.

You also have the option to create and generate an input form, the Master View table or just a row of that table, or a Web region front controller. In the course of completing the wizard, you will have an opportunity to customize the statement content and layout of these forms. The **File Description** area provides information about each form that you select.

Click **Next**.

Database Web Wizard

Database Web Pages

Choose SQL Method
Choose an SQL method.

How would you like to create your SQL statement?
- Be guided through creating an SQL statement
- Manually type an SQL statement

Choose a database model for the SQL statement.
- Use existing database model
  - Database model: _____ Browse...
- Connect to a database and import a new database model

☐ Save SQL Statement
Statement Name: NewStatement

< Back   Next >   Finish   Cancel

Choose how to create the SQL statement. If you already have an SQL statement written, from the **Choose an existing select statement** page, you can browse to locate a statement that exists in a workbench project. If you want to create a new SQL statement, click **Next**.

To create a new SQL statement from the **Specify SQL statement information** page, choose whether to be guided by the Construct an SQL statement wizard or to manually type an SQL statement into a text editor. In addition, select the appropriate radio button to locate the database model by either browsing for an existing database model in the **Database model** dialog, or by connecting to a database and importing a new model.

Optionally, choose whether to save the new SQL statement that you will create. If you select the **Save SQL Statement?** check box, you should supply a name for the statement by overtyping in the **Statement Name** field.

Click **Next**.

IBM Software Group |

DB Web Wizard

Database Web Pages

Connection Page
Connect to your database.

☐ Save Database Metadata

Connection name: Con1
Database: ISERIES
User ID: coulthar
Password: ****

Database vendor type: DB2 Universal Database for iSeries, V5R1
JDBC driver: AS/400 Toolbox for Java JDBC Driver
Host: TORAS1FB
(Optional) Port number:
Server name:
Database Location:
JDBC driver class: com.ibm.as400.access.AS400JDBCDriver
Class location: D:\WDSC\iseries\eclipse\plugins\com.ibm.etools
Connection URL: jdbc:as400://TORAS1FB

Filters...
Connect to Database

Tip: your JDBC Class location is
d:\wdsc\iseries\eclipse\plugins\com.ibm.etools.iseries.toolbox_5.0.0\runtime\jt400.jar

< Back   Next >   Finish   Cancel

© 2003 IBM Corporation

Connection Filters

Connection Filters

| Schema | Table | Stored Procedure | User-Defined Function |

Enabled	Target	Predicate	Filter	Operator
☑	SCHEMA	LIKE	CARDEMO	

Add Filter
Remove
Move up
Move down
Enable
Disable

Tip: Press Filters and subset by library (="SCHEMA")

Filters...
(SCHEMA LIKE CARDEMO)

Stored procedures and user-defined functions types
☑ SQL   ☑ Java
☑ C   ☑ Other

☑ Exclude system schemas

OK   Cancel

This is the 2nd page of the Database Web pages wizard.

If you chose to import a model from a database, you will need to supply information in the **Database Connection** page:

Connection name - A name used to identify the database connection (this can be any string).

Database - The name of the Database to connect to. For iSeries, the whole system is the database normally, so this can be any string too.

User ID / Password - ID and Password required to connect to the database.

Database vendor type - A list of supported database types and levels. Choose DB2 Universal Database for iSeries, V5R1.

JDBC driver - A list of available database connection drivers. It will pick the right one for you.

Host - The name of your iSeries server.

Class location -The location of the JDBC driver class identified for you in **JDBC driver class**. Click **Browse** to locate the jar file, which for iSeries is jt400.jar, which is shipped with WebSphere Development Studio Client here:

> d:\wdsc\iseries\eclipse\plugins\com.ibm.etools.iseries.toolbox_5.0.0\runtime\jt400.jar

> Where d:\wdsc is where you installed the product.

Connection filters (from the Filters push button) - Specify tables you want to filter out of your view of the database. Once a filter is defined, it is available whenever you define a connection. You can filter out system schemas (that define basic database structures like tables and views) by selecting the **Exclude system schemas** check box.

> For iSeries, add a Schema (library) filter or else it will take forever to leave this page of the wizard! You can specify a generic name, but be sure to use '%' as the wildcard character, not '*'.

Connect to Database push button - Connects to the database prior to you advancing to the next page of the wizard. Just handy in case there are errors connecting.

Click **Next**.
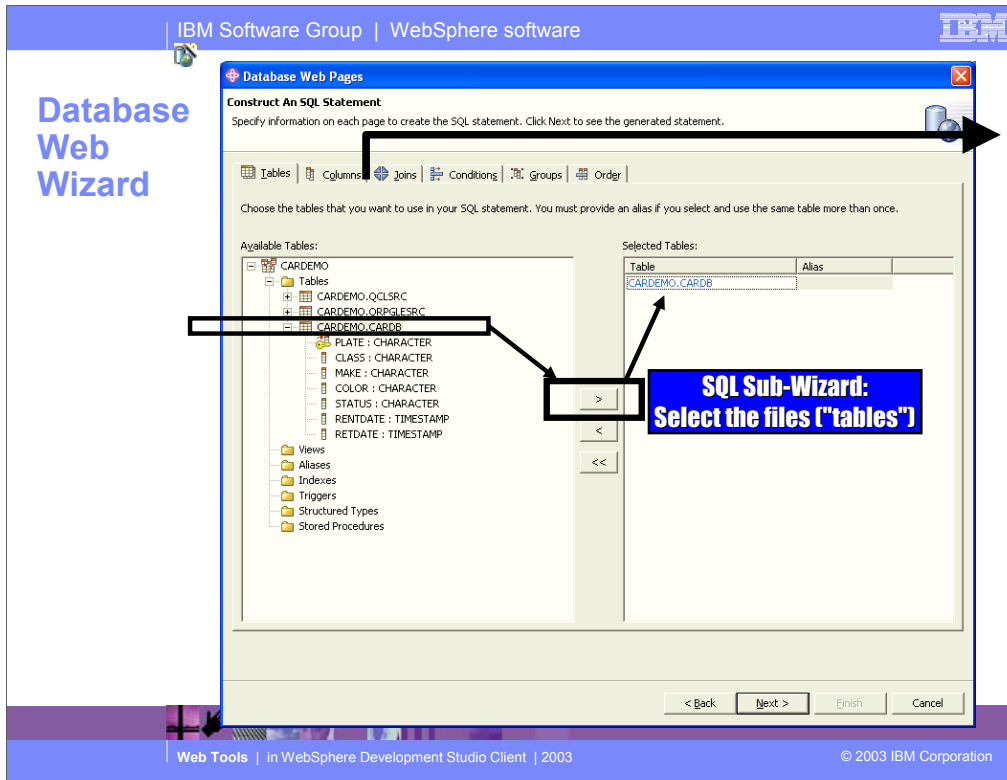
Select the tables (files) you wish to access from your Web pages, and press **>** to select them for the wizard.

Click the **Columns** tab to continue working on your statement.

Database Web Wizard

Database Web Pages

Construct An SQL Statement
Specify information on each page to create the SQL statement. Click Next to see the generated statement.

Tables | Columns | Joins | Conditions | Groups | Order

Choose the output columns to include in your SQL statement. You can add column expressions and specify an alias for your output columns.

Available Columns:

- CARDEMO.CARDB
  - PLATE
  - CLASS
  - MAKE
  - COLOR
  - STATUS
  - RENTDATE
  - RETDATE

Selected Columns:

Column	Alias
PLATE	
CLASS	
MAKE	
COLOR	
STATUS	
RENTDATE	
RETDATE	

SQL Sub-Wizard: Select the fields ("columns")

Move Up
Move Down
Add...
Change...
Remove

☐ Show schema and table names

< Back   Next >   Finish   Cancel

Web Tools | in WebSphere Development Studio Client | 2003   © 2003 IBM Corporation

If you don't want to work with all the fields in the file, select the subset here. Note for performance, it is best to only select those you need.

Select the **Conditions** tab to continue evolving your statement.

Database Web Wizard

Database Web Pages

Construct An SQL Statement
Specify information on each page to create the SQL statement. Click Next to see the generated statement.

Tables | Columns | Joins | Conditions | Groups | Order

Define the conditions that you want to use to select rows. Click in the appropiate cell to specif...

Column	Operator	Value	And/Or
CARDEMO.CARDB.STATUS	=	:status	

SQL Sub-Wizard: Select the records ("rows") by using conditions

Tip: use :variable-name for values you want to prompt the user for

< Back   Next >   Finish   Cancel

If you do not want to list all the data in the file, then on this page specify the criteria for what data to show. If you want to ask the user for one or more values, use :variable, where variable is any arbitrary name. This will generate an input page that prompts the users for the values for those variables.

Press **Next** to finish with the SQL statement.
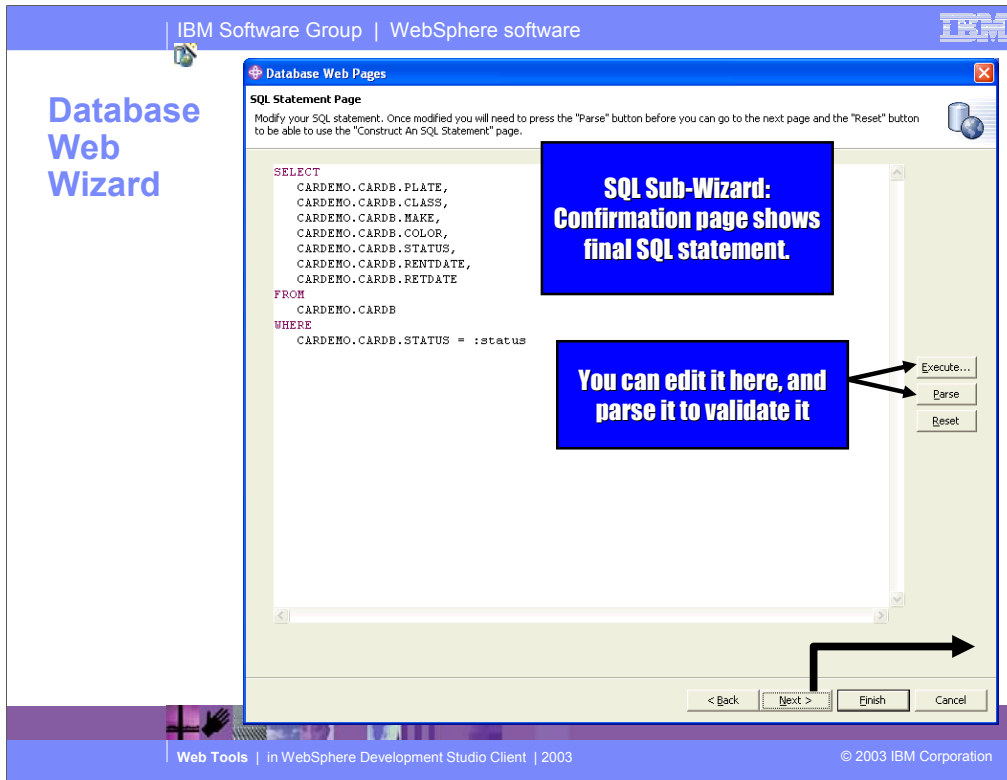
Perform any necessary actions in the **SQL Statement** page. The available actions include the following:

Edit the SQL statement

> Using the SQL Statement Editor, which is a text editing window, you can verify the accuracy of the SQL statement and directly edit the SQL statement with the help of the editor's syntax highlighting for SQL keywords.

Execute

> Opens the **Execute SQL Statement** dialog. If the SQL statement requires a parameter, you will be prompted for input. The SQL statement is executed until you click **Close**.

Parse

> Parse the SQL statement to verify its validity.

> **Note:**

>> Once an SQL statement is parsed, you can make updates in the editing window. However, if you wish to return to the **Construct an SQL statement** page to update the statement, you must first click the **Reset** button to restore the original statement statement.

Reset

> Resets the session so that you can return to the **Construct an SQL statement** page notebook to make changes to the SQL statement.

Press **Next** to continue.

Database Web Wizard

Database Web Pages

Runtime Connection Page
Specify the database connection you would like to use at runtime.

○ Use data source connection

Data source/JNDI name: [ ]

● Use driver manager connection

Driver name: [com.ibm.as400.access.AS400JDBCDriver]

URL: [jdbc:as400://TORAS1FB]

User ID: [coulthar]

Password: [*******]

Re-enter password: [*******]

⚠ Note that the password given here will be stored unencrypted in this project's web.xml file

< Back   Next >   Finish   Cancel

Choose whether to use a data source connection or a driver manager connection in the **Specify Runtime Database Connection Information** page. A data source connection name should be the JNDI name defined in the server configuration. A data source is the preferred database connection type for a Web application. A data source is defined in a Web server. Usually, it is a pooled connection, so that the server makes a pool of connections to the database on startup. Each time a JSP file requires a database connection, the server provides one of its pooled connections, which minimizes the connection overhead for the JSP file. When the JSP is done executing, it releases the database connection back to the pool.

For a driver manager connection, the JSP file requires a new connection to the database on each execution. Supply the driver name and associated URL.

Supply the user ID and password, along with a verification of the password, necessary to access the target database. If this is a new connection, the userid and password specified in the **Database Connection** page are reflected here. If you used an existing database model or an existing SQL statement, these are the values originally used to load the existing database model. The ID/password will be added to the web.xml file as an initialization parameter of the servlets associated with the Main and Details result pages.

Click **Next**.

Database Web Wizard
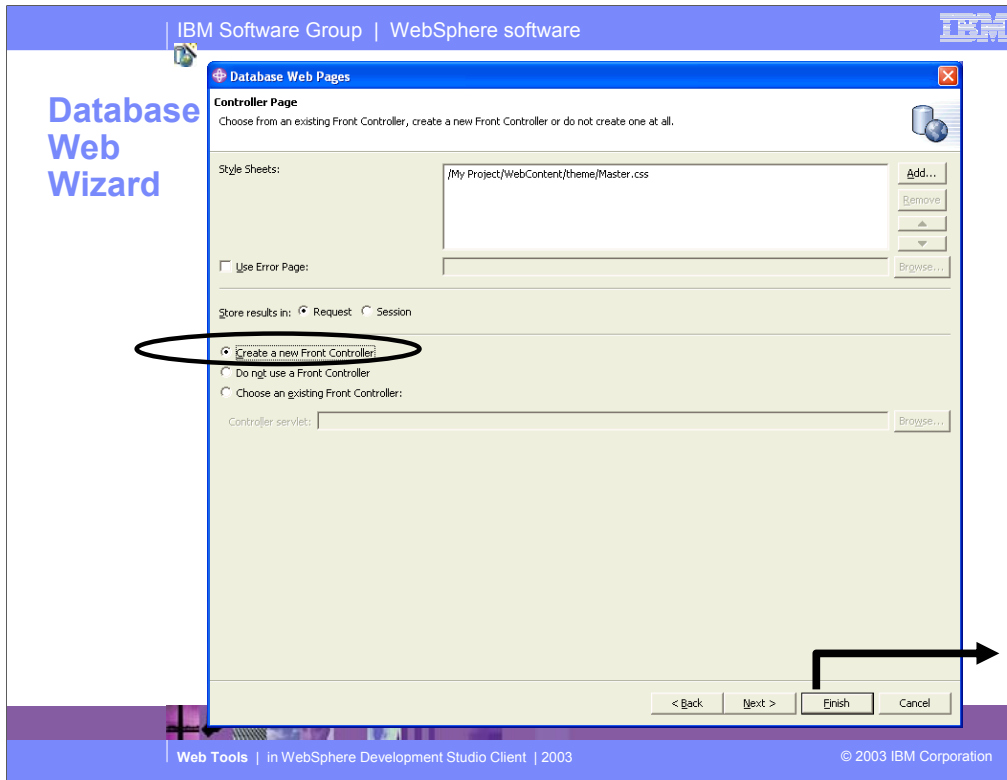
In the Controller Page, select the check box and use the **Browse** button to associate a style sheet or error page from the current project with the generated Web pages. Style Sheets allow you to define colors, fonts and many other attributes of your HTML tagging. The error page enables you specify a URL that is forwarded to a user's browser if an error is encountered while executing one of the generated resources.

Select the **Request** or **Session** radio button to identify the model type. Storing the results in a session will allow the data to be used in multiple pages for the duration of the session. Storing the results in a request will allow access to the data only for the duration of the request; the result set is destroyed after the request is processed, freeing up memory on the server. The default is **Request**.

Optionally, you can include or remove processing layers from the model used to generate Web pages. A front controller is a servlet that serves as the single entry point for all requests to a Web application region. A controller provides a single point to perform all authentication, logging, debugging, and to all subsequent views, including the input, master results, and details results views.. You can choose to automatically create a new controller, reuse an existing controller, or not use a controller at all. If you choose not to use a controller, that functional layer of processing is not performed.

Click **Next**.
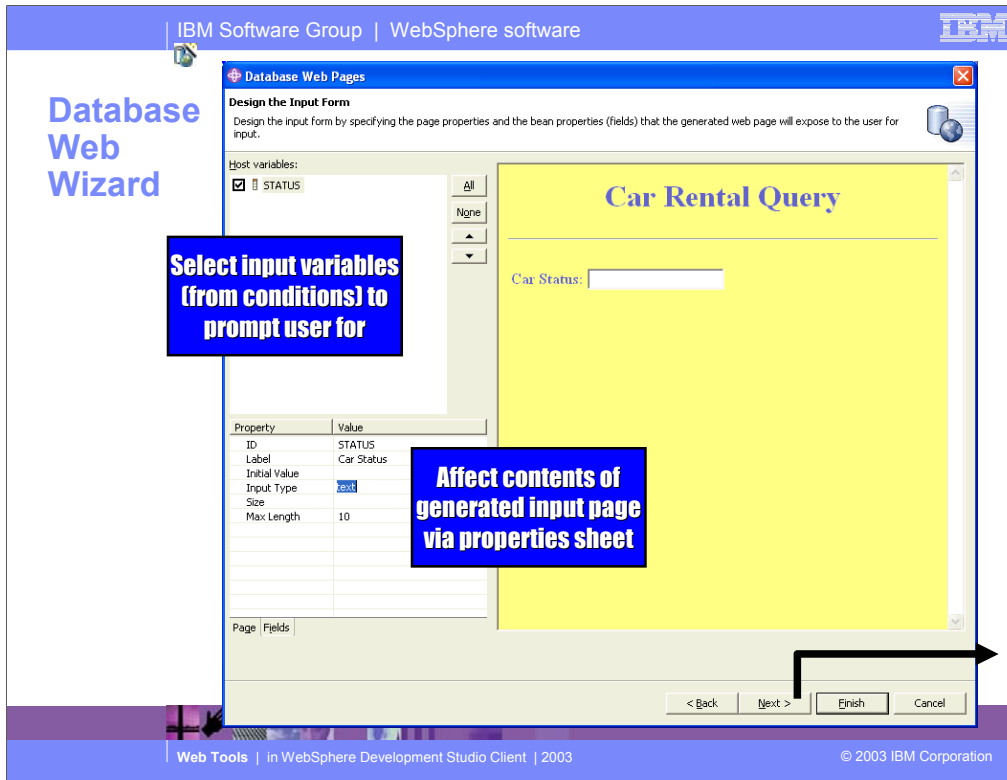
Select and update, as needed, values in the **Host variables list** and properties table to determine the content, layout, and design attributes of the input form. The scrolling panel on the right side of this page approximates the look and layout of the resulting input page. You can also use the **All** or **None** button to select or deselect the entire list. Click the up and down arrow buttons to reorder the columns in the input page. At least one column must be specified as a **Key Value** if you chose to generate a Details form.

Click **Next**.
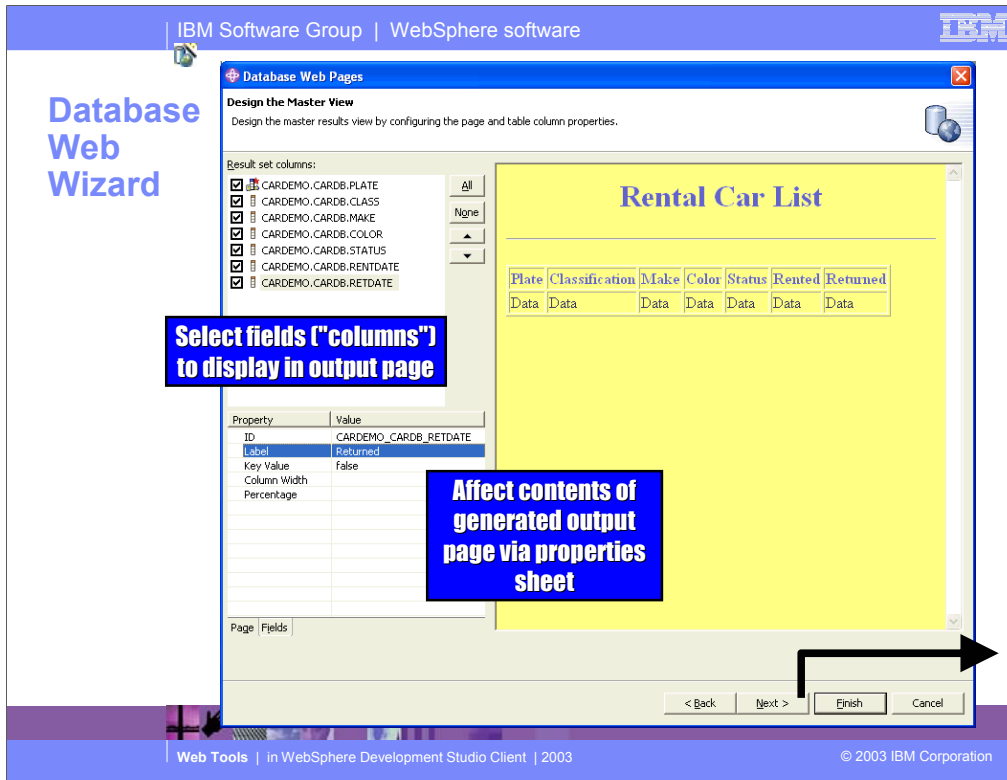
Select and update, as needed, values in the **Result set columns** list and properties table to determine the content, layout, and design attributes of the Master View Web page. The scrolling panel on the right side of this page approximates the look and layout of the resulting page. You can also use the **All** or **None** button to select or deselect the entire list. Click the up and down arrow buttons to reorder the columns in the result table. At least one column must be specified as a **Key Value**.

Click **Next**.
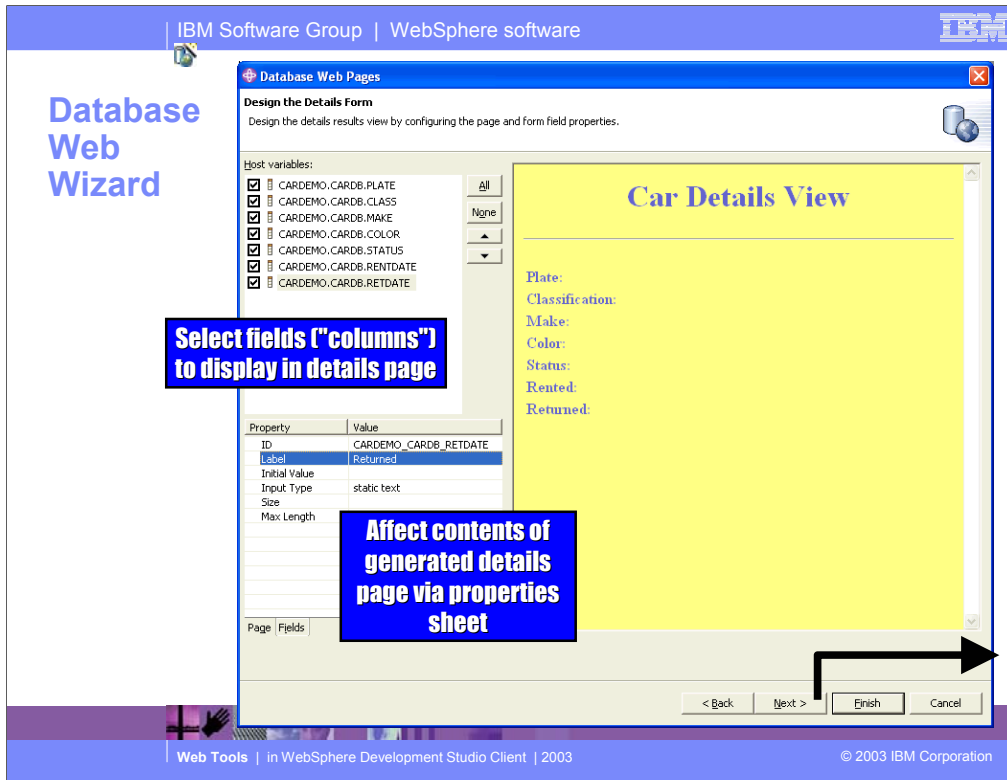
Select and update, as needed, values in the **Host variables** list and properties table to determine the content, layout, and design attributes of the details results Web page. The scrolling panel on the right side of this page approximates the look and layout of the resulting page. You can also use the **All** or **None** button to select or deselect the entire list. Click the up and down arrow buttons to reorder the columns in the details table.

Click **Next**.
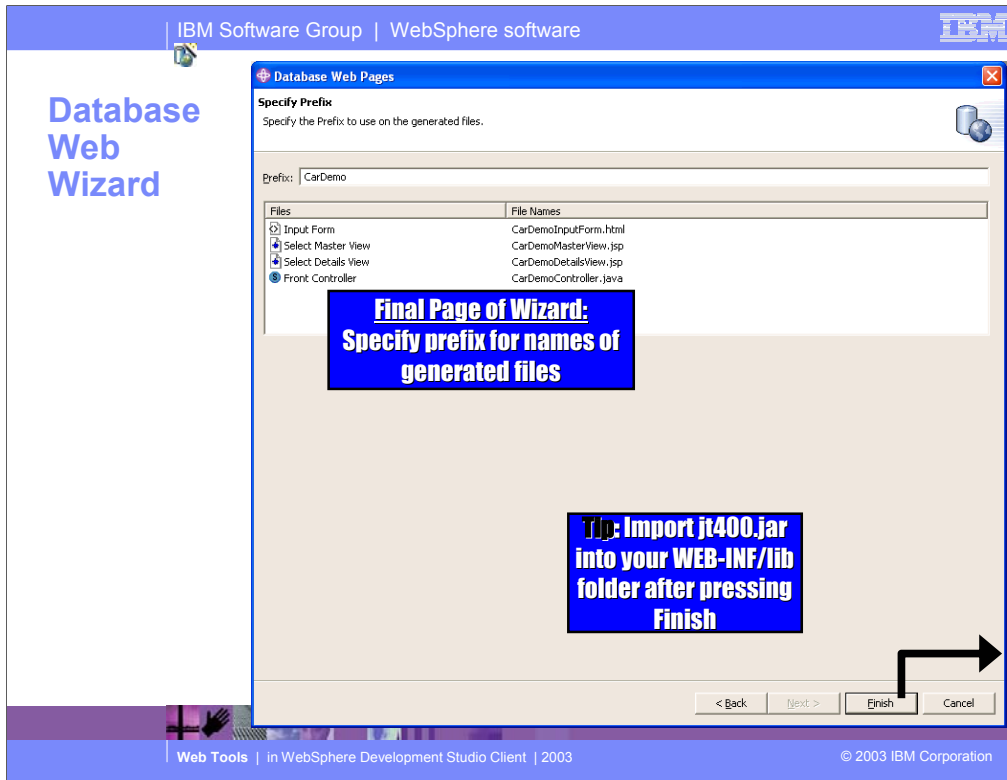
Provide a common Java package prefix for the pages that are generated from the statement. Note that the list of generated pages and resources will reflect any changes to the prefix as you type in the **Java package prefix** field.

Click **Finish** to generate the Web pages.
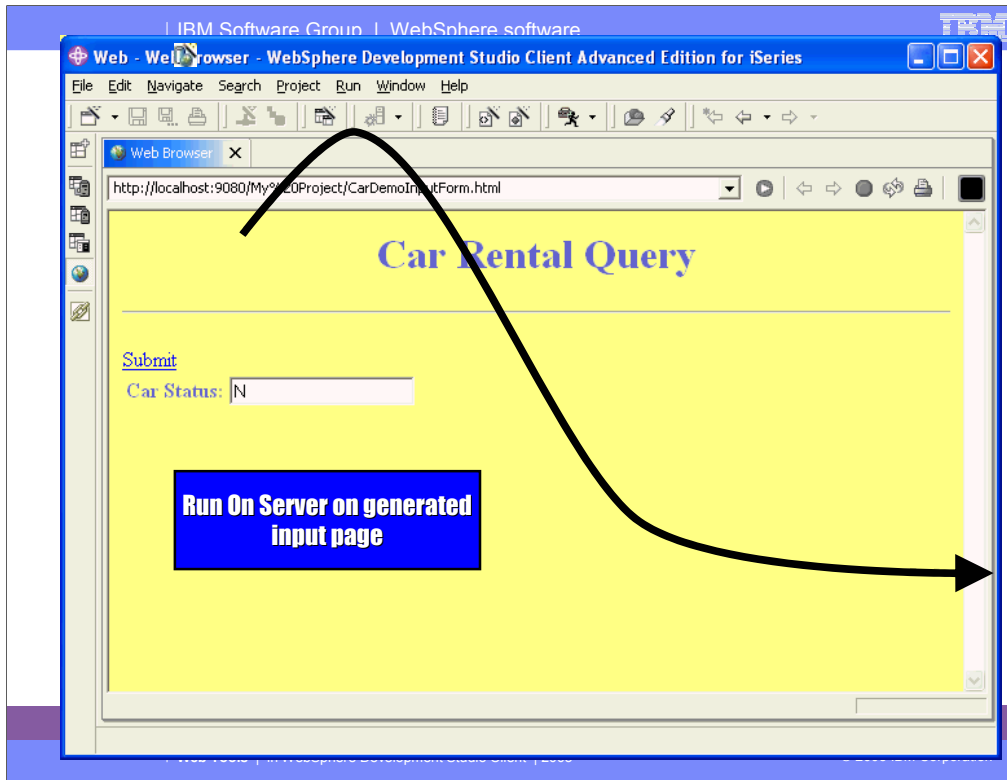
Running the Web application locally uses the WebSphere test environment. To use the WebSphere test environment, do the following:

Right-click the file you want to run and select **Run on server**.

If this is the first time you are doing this, select the **Create a new server** radio button.

Leave Servers in the **Folder** field.

In the **Server type** field, expand **WebSphere version 5.0** and select **Test Environment**.

Click **Finish** to configure the test environment and launch the server.

Enter a value in the **Car Status** field on the Car Rental Query input page and click **Submit**.
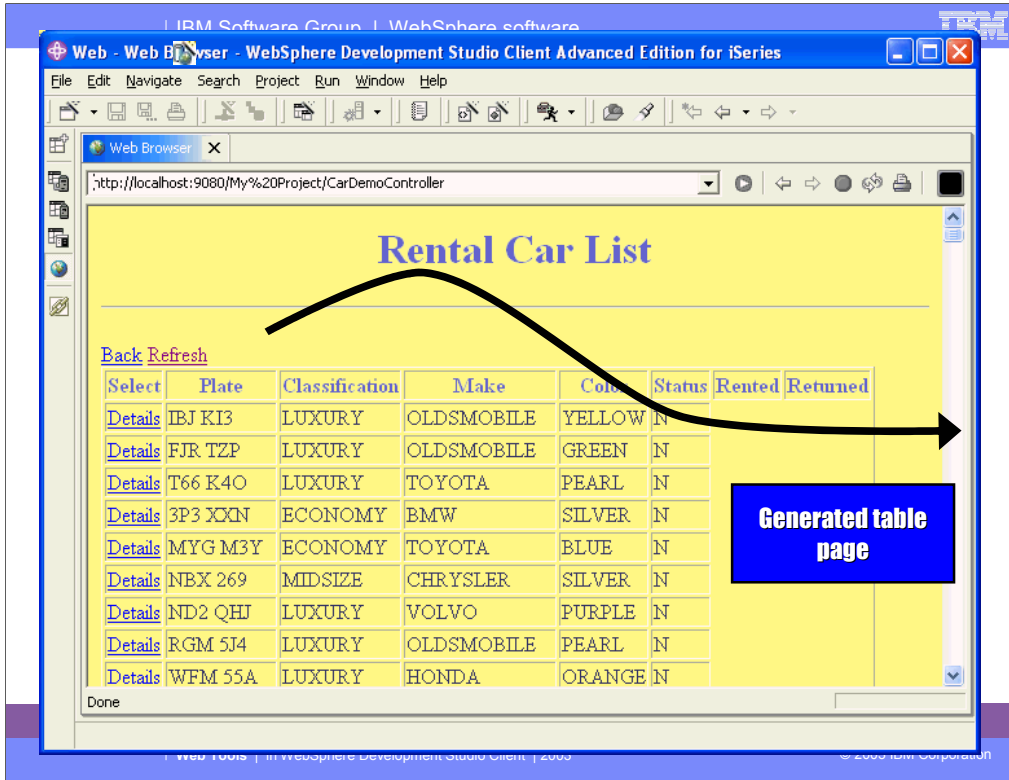
This is the resulting list of items meeting the conditioning criteria.

To see details of one item, select it in the list.
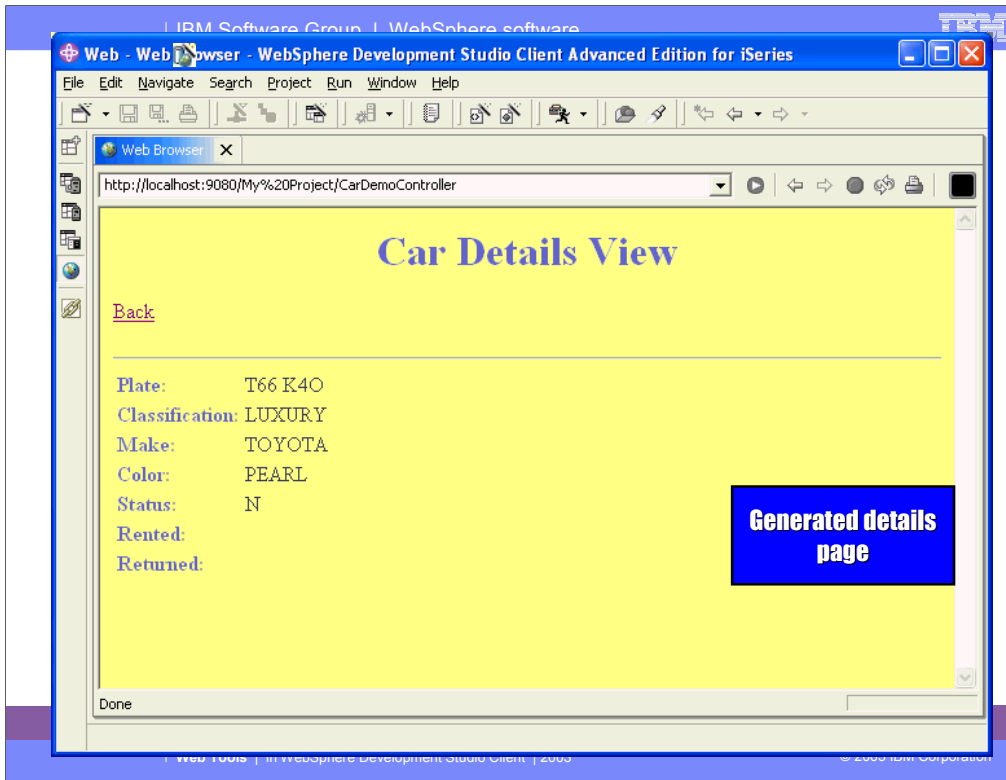
Now you see the output of the selected car details.

AGENDA

Web Tools

Java Bean Web Pages Wizard

Now we discuss the Java Bean Web Pages wizard

Java Bean Web Wizard

● **Java Bean Web Pages Wizard**
  ► **Input: Java Bean**
  ► **Output:**
    ▪ Input JSP page prompting for input properties
    ▪ Output JSP page displaying output properties
    ▪ Servlet that ties them together

  ► **To launch:**
    ▪ Select **File->New->Other** from Web perspective
      ▪ then **Web** and **Java Bean Web Pages**

Web Interaction
Template Application
Web Project
Servlet
Filter
Image File
CSS File
JSP File
JavaScript File
Life-cycle Listener
HTML/XHTML File
Database Web Pages
Java Bean Web Pages
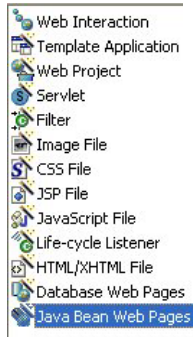
The Java Bean Web Pages wizard takes as input a Java Bean, and generates from it a Web page that prompts the user for the input properties (setXXX methods), and a Web page that displays to the user the output properties (getXXX methods). You get to select the properties to prompt for, and display, and can easily affect the Web page fields generated for these properties. You also get to select which method or methods to call when Submit is pressed in the first generated Web page. The generated Servlet will call the selected method(s), and then transfer control to the generated output page.
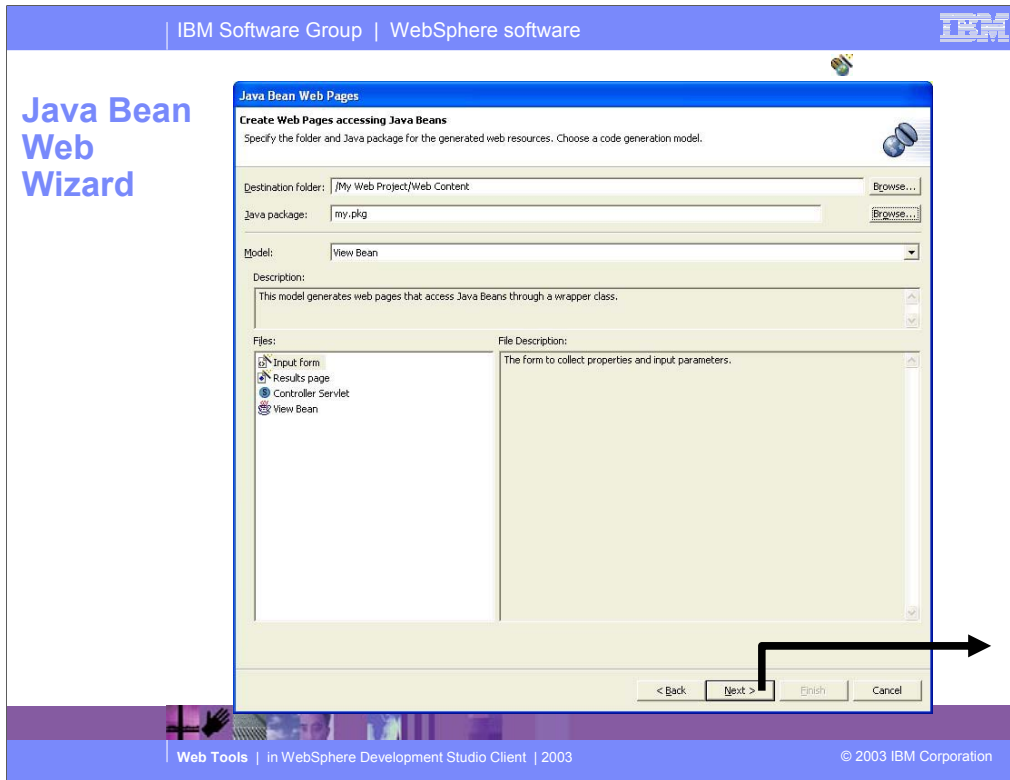
This is the first page of the Java Bean Web Pages wizard. Here you select the folder in which to generate the output (defaults to the Web Content folder in the current project), and the name to give the Java package for the generated code.

You also specify the "model" here for the generated code (the files that will be generated are listed at the bottom). As of 5.1, your only choice is View Bean, which generates a view bean that wraps the model bean, such that when data is extracted from the bean you have the opportunity to write code to massage that data for display in a Web page. This might involve adding html tags to adorn it, for example, or formatting it in some other way.
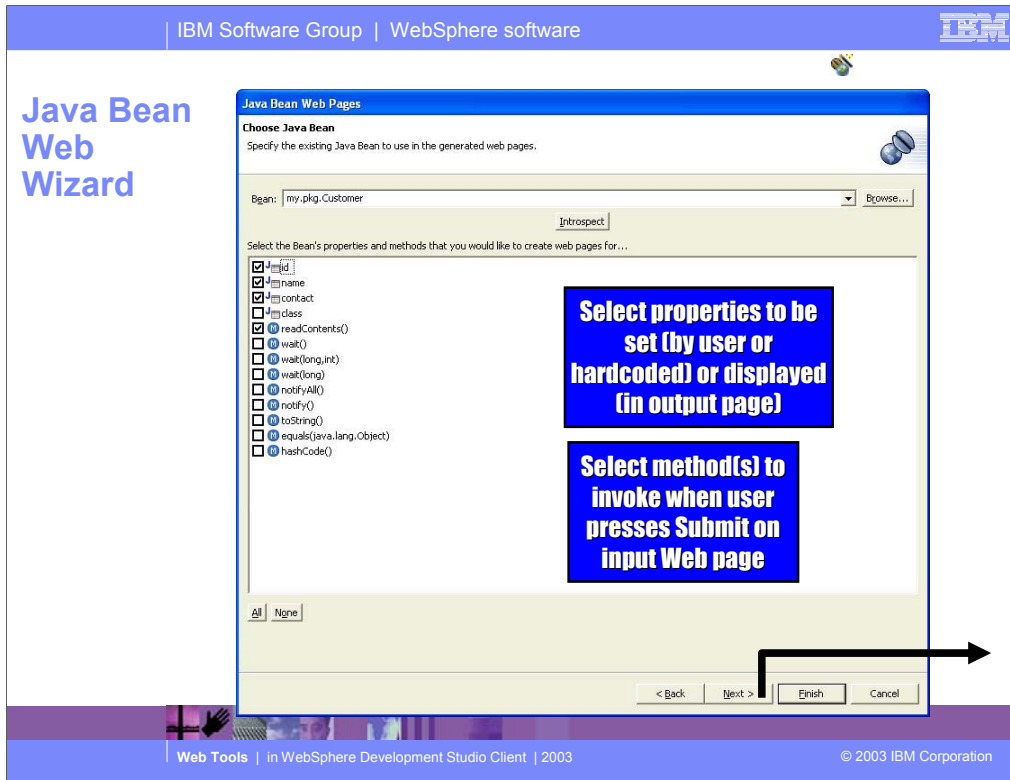
This is the second page of the Java Bean Web Pages wizard.

Here you select the properties to prompt for and/or display, and the methods to call when the user presses Submit on the first generated Web page.

Java Bean Web Wizard

Java Bean Web Pages

View Bean Data Page
Choose options for Front Controller or View Bean data wrappers.

Style Sheets: /My Web Project/Web Content/theme/Master.css

Add...
Remove

☐ Use Error Page:

Browse...

Store results in: ⦿ Request ○ Session

○ Create a new Front Controller
⦿ Do not use a Front Controller
○ Choose an existing Front Controller:

Controller servlet:

Browse...

☑ Create View Bean(s) to wrapper your data object(s)

< Back   Next >   Finish   Cancel

This is the third page of the Java Bean Web Pages wizard.

Here you can select

  - the cascading style sheets the generated pages will use (the default is the Master.css file created by the Web project wizard).

  - your own Web page to show when then is an error in the processing

  - whether to store the bean data in the request or the session

  - whether to generate a servlet, generate only JSPs, or re-use an existing servlet. Generally you should always use a servlet, but its best to have only one per Web application, so re-using is best. To do this, create one the first time you use the wizard, and re-use it subsequent times. You'll have to manually edit it though, to add the logic to process the newly generated pages.

You also can decide here whether to generate that view bean as described in the View Bean model on the first page.
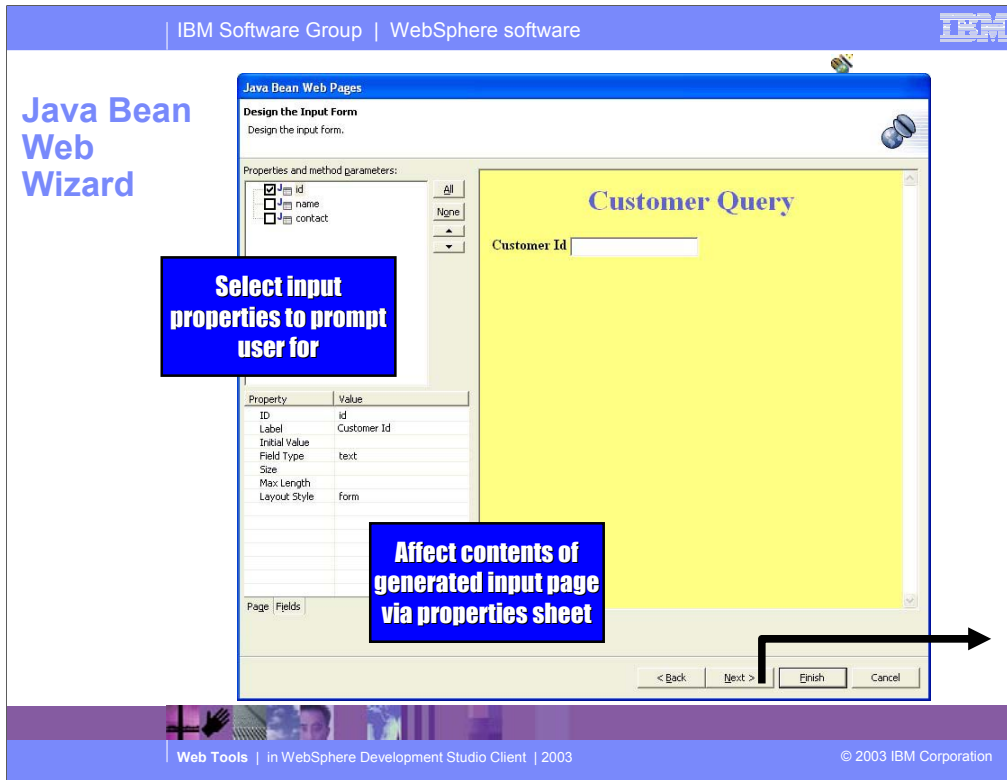
This is the fourth page of the Java Bean Web Pages wizard.

Here you select which of the bean's properties (of those you selected on the second page of the wizard) to prompt the user for. Note if you want to hard-code a property and not prompt for it, select it here anyway, then set the Initial Value in the property sheet, and specify the Field Type as hidden.

For each field, you can easily specify properties that affect its appearance, such as whether to use a text field, hidden field, password field, dropdown or other Field Type.

You can also set properties for the overall page, such as the background color. Of course, all these can still be changed after by editing the generated page.
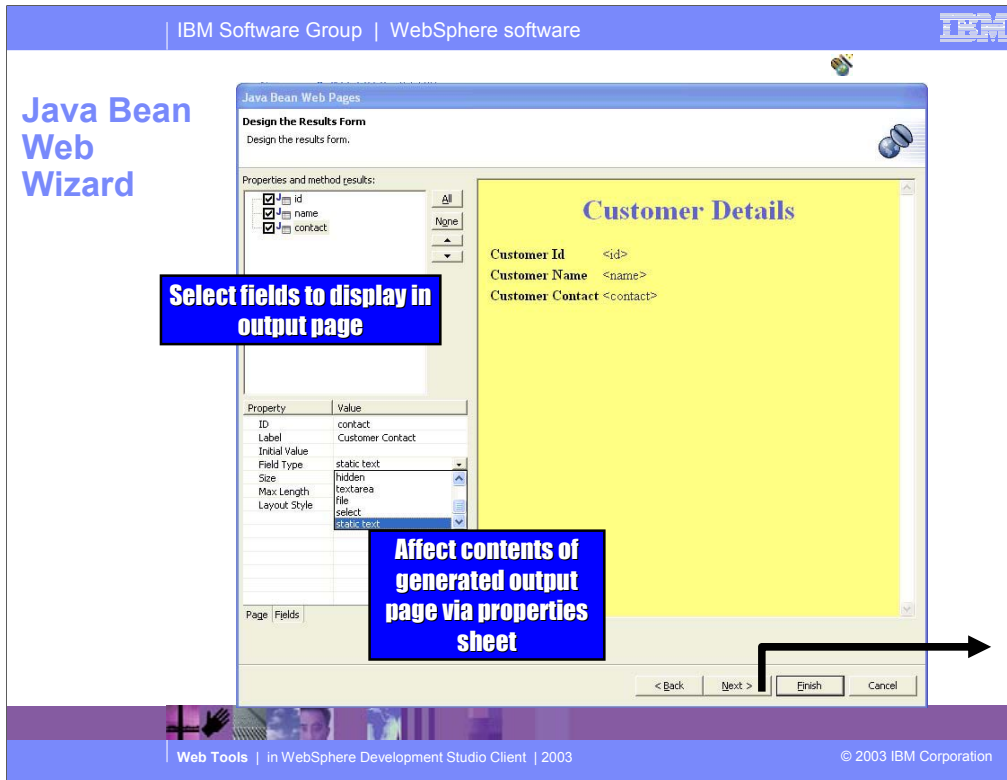
This is the fifth page of the Java Bean Web Pages wizard.

Here you select which of the bean's properties (of those you selected on the second page of the wizard) to display to the user, after setting the input properties and running the methods selected to be invoked.

For each field, you can easily specify properties that affect its appearance, such as whether to use a text field, hidden field, password field, dropdown or other Field Type.

You can also set properties for the overall page, such as the background color. Of course, all these can still be changed after by editing the generated page. For indexed properties you may want to change the Layout Style from Form to Table.

Java Bean Web Wizard

Java Bean Web Pages

Specify Prefix
Specify the Prefix to use on the generated files.

Prefix: Customer

Files	File Names
Input form	CustomerInputForm.html
Results page	CustomerResultsForm.jsp
View Bean	CustomerViewBean.java

Final Page of Wizard:
Specify prefix for names of generated files

< Back   Next >   Finish   Cancel
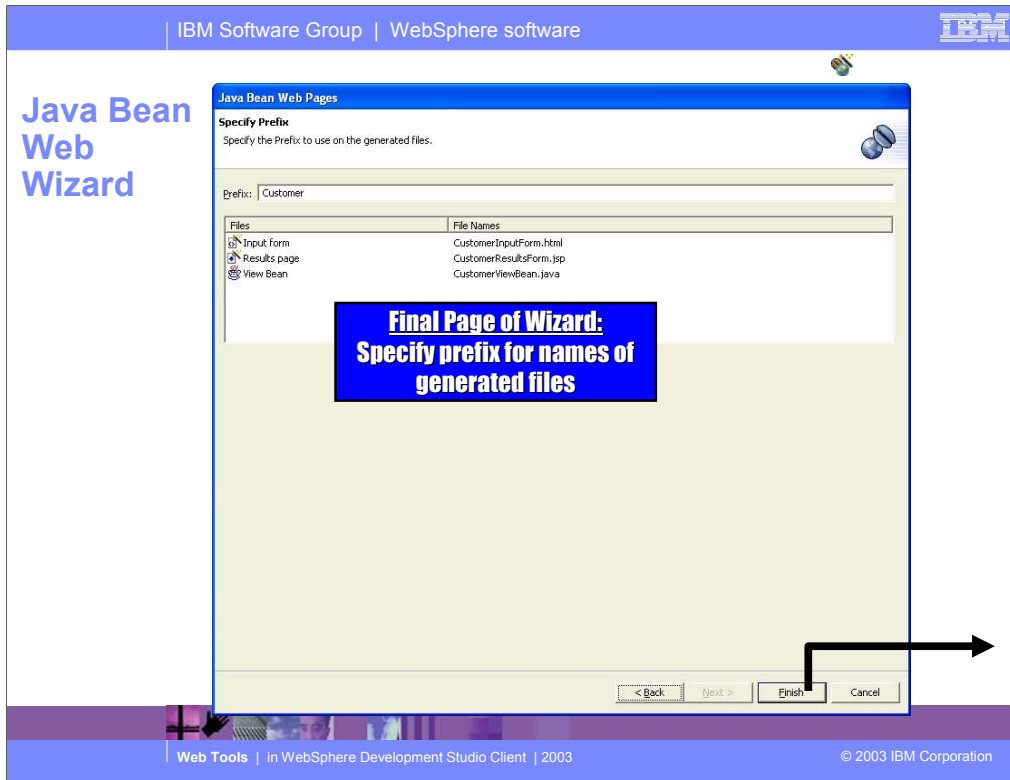
This is the sixth and final page of the Java Bean Web Pages wizard.

It merely shows the files that will be generated. You can affect the names of those files by specifying the prefix to apply to each.
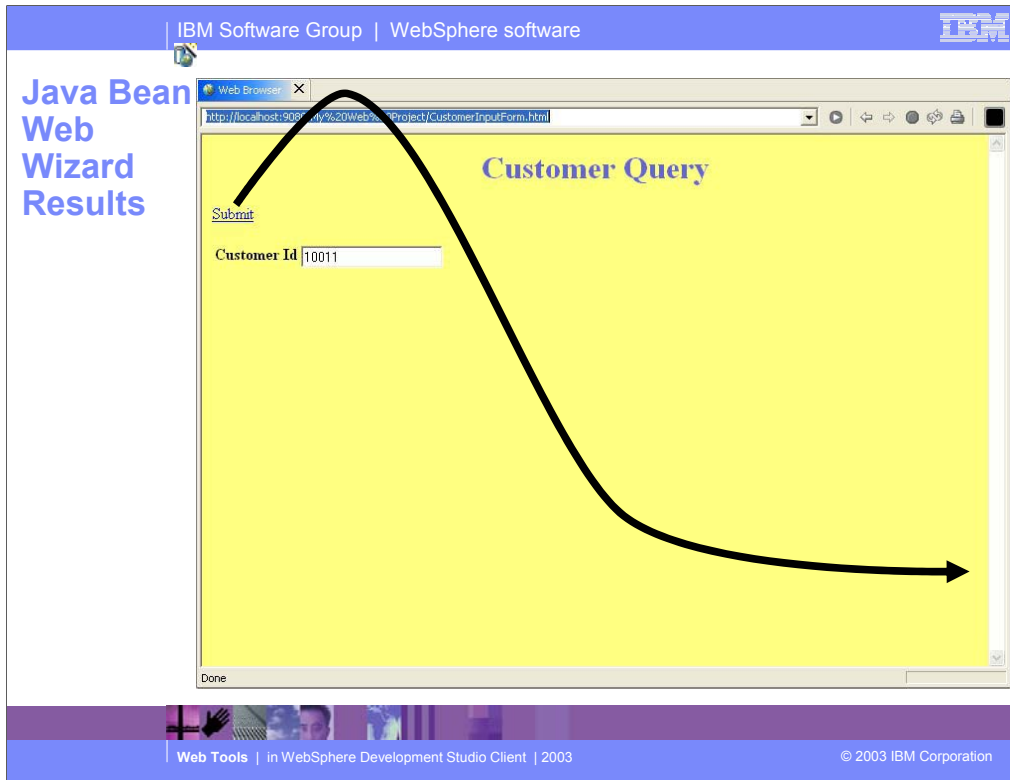
This is result of running the generated input Web page (right click on it and select Run On Server). Here we see we are prompted for the properties we selected to prompt for, and when we press Submit, the generated controller servlet will be called. It will in turn pass control to the output JSP…
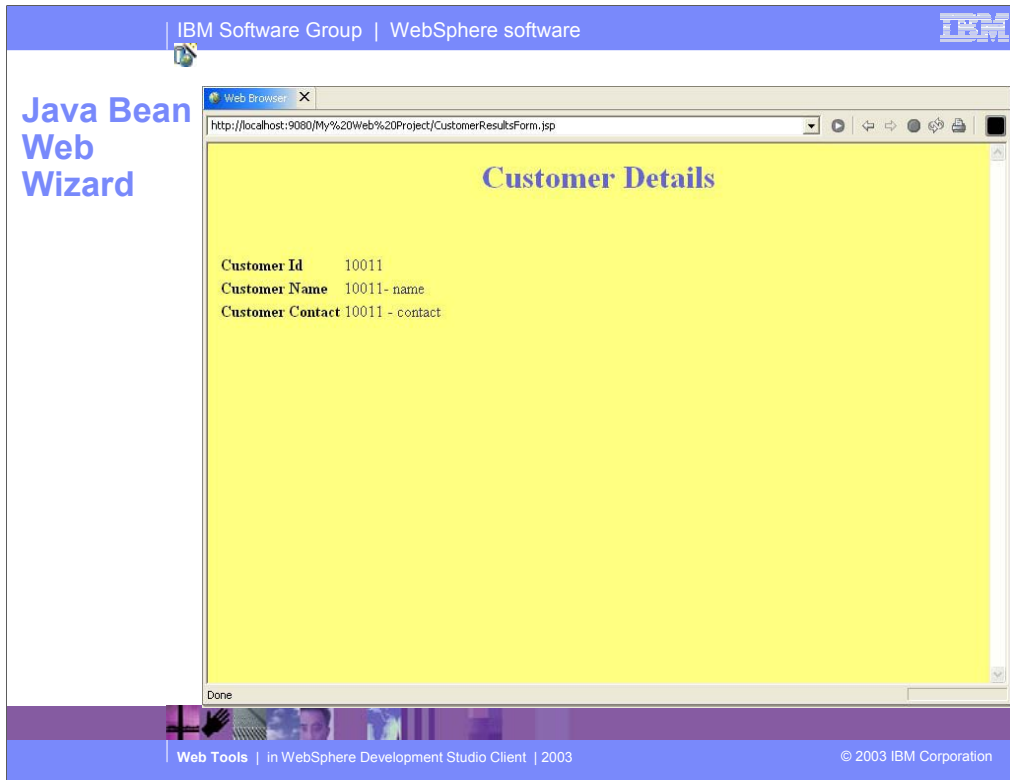
… and this is the generated output JSP. If you look at its source, you see that it instantiates the view bean (which in turn instantiates the model bean), and then sets its input from the fields on the input page, and calls the methods you selected to call, and then extracts the data from the bean and places it into the generated fields on this output page.

Struts Tooling

► **Struts Tools**

We will now drill down a bit on the Struts tooling.

Struts Tooling in WDSC

- **Struts support within Web projects**
  - ► An option when creating the project, and in project properties
- **Struts preferences settings**
  - ► Preferences -> Web Tools -> Struts Tools
- **Web Diagram Editor**
  - ► A cool way to develop Web apps:
    - Layout the "flow" of your app by dropping icons representing Web pages and Struts actions (code), drawing lines between them to represent flow
    - "Realize" the nodes by right-clicking and selecting appropriate editor or wizard
- **Struts Explorer view**
  - ► A struts-resource-based view of a Web project
- **Config file editor**
  - ► Easy way to edit the struts-config.xml file

Here the highlights of the Struts tooling is listed, "at a glance".

The Struts application development tools make it easy for you to build and manage a Struts-based Web application.

•Lets you set up a Struts project so that tag libraries and other Struts-related resources are located properly; as a result, you can reference those resources without fail as you develop your application

•Provides Wizards to create form beans and action classes so that you have a head start in developing the logic that is specific to your application

•Includes a specialized editor to create and modify the Struts configuration file

•Provides Struts support for validation and editing; for example, by helping you to use Struts tag libraries

•Provides a visual assembly tool, which has the following benefits:

•Helps architects to design the flow of a Struts-based Web application and to communicate that design to other professionals

•Embeds Struts code in several components

•Provides quick access to resource-appropriate editors and wizards

•Separates team responsibilities for greater productivity and focus
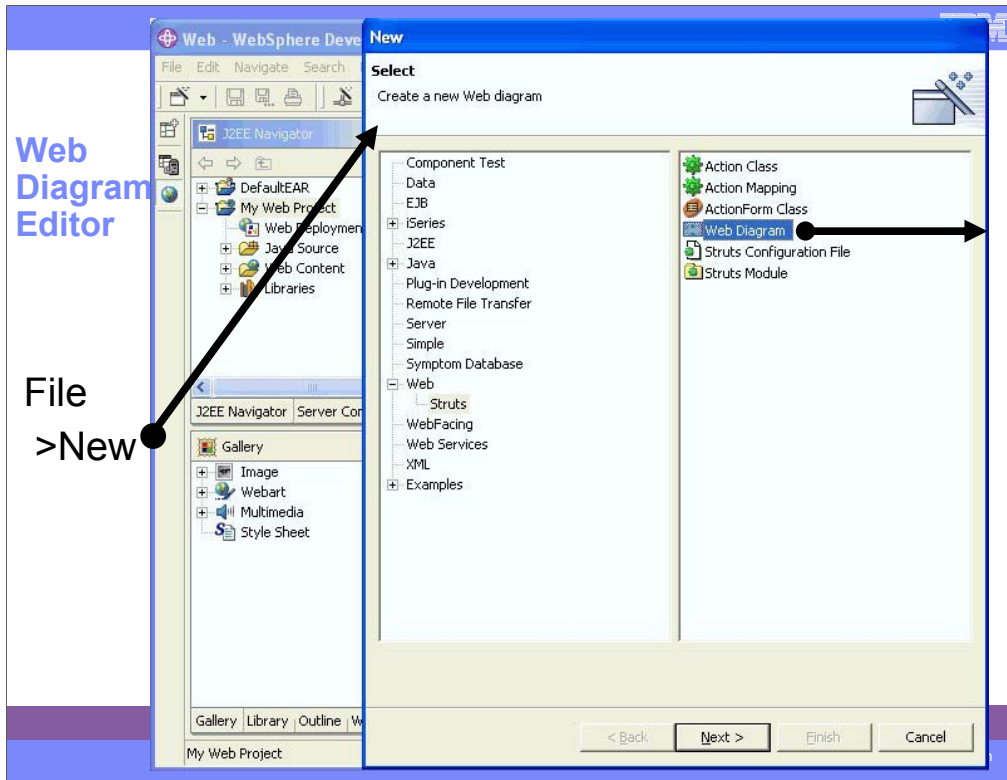
A *Web diagram* is a file that helps you visualize the flow structure of a Struts-based Web application. Because of the indirectness involved with a Struts application, being able to visually see the application's flow can help you to better understand the application.

In the J2EE Navigator, select the name of a project.

Right-click and select **New** and then **Other**.

Select **Web** and then **Struts** in the left pane and **Web diagram** in the right pane.

Click **Next**.

In the **File name** box, type the name of the Web diagram file. Click **Finish**.
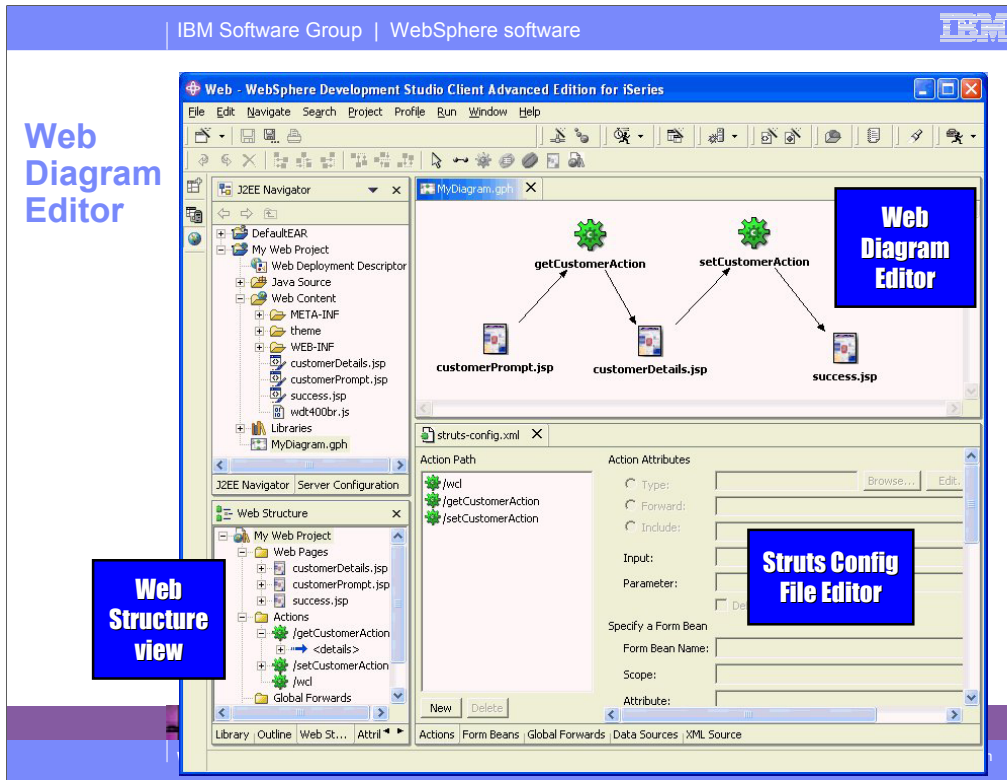
The Web dialog editor allows you to "draw" the flow of your web application but creating nodes for web pages and actions (among others), and drawing connecting lines between them for flow. Initially, the things you create are "greyed out" because they don't yet exist. As you double-click on nodes to create them, they nodes become colored and the lines fully drawn (versus dotted). Typically you use form beans to represent the data from input pages, although the diagram here doesn't show them.

This tool makes it very easy to create a web site that actually works, flowing from page to page, very quickly. Indeed, a trick is to create the form beans first (by double clicking) and then generating the pages from them using a right-click action. Then the only work left is to make them "look good". The struts config file is automatically maintained for you as you use this editor.

Run On Server

► **Running Any Web Application**

Now we talk about something exciting … how to quickly see your Web applications running, using the built-in copy of WebSphere Application Server, or a remote WebSphere Application Server.

Run On Server

- **Run On Server**
  - ► **Now this is VERY COOL!**
- **When ready to test your Web app**
  - ► **Right click on initial html or jsp file**
    - ▪ or whole project, which implies the index.html file
  - ► **Select "Run on Project"**
  - ► **Wait for the magic...**
- **Your Web application will run!**
  - ► **Opens Server perspective**
    - ▪ Publishes it to built-in copy of WAS
    - ▪ Starts built-in copy of WAS
    - ▪ Brings up a Web Browser
    - ▪ **Runs your application!!**

Tip: you can set breakpoints in your java code

Once you have a Web application, it comes time to see it running. You don't have to install and configure websphere first in order to do this. WebSphere Application Server is actually built into the toolset, and ready to use.

To use it, merely right click on your first Web page (or the project, which implicitly tries to start index.html or index.jsp), and select Run On Server. This starts the built-in copy of WebSphere Applications Server, publishes your project to it, and runs your application

Run On Server
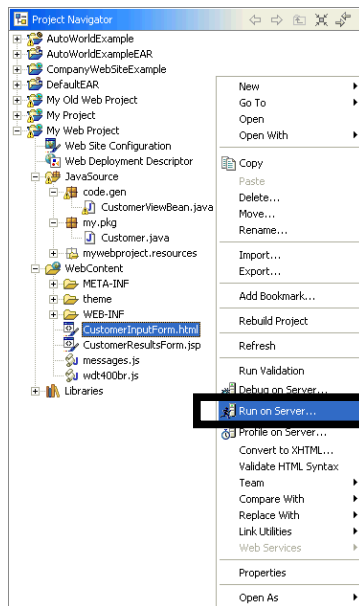
Select your initial Web page, right click, and select Run On Server…

Run On Server

Server Selection

Server selection
Select which server to launch.

○ Use an existing server

Server	Status
WebSphere v5.0 Test Environment	Configured
WebSphere v5.0 Test Environment (2)	Configured
WebSphere v5.0 Test Environment (3)	Configured

Status: The server is already configured for this project

○ Create a new server

Folder: Servers

Server type:
- Server
- Test Environment
- WebSphere v4.0 Server
- Apache Tomcat version 4.1
- Apache Tomcat version 4.0
- Apache Tomcat version 3.2
- Other

Description:

□ Set server as project default (do not prompt)

< Back   Next >   Finish   Cancel

You are asked for the server configuration you want to use, which by default is the one for the built-in copy of WebSphere Application Server.

Run On Server

Manage multiple instances of built-in WAS here in Servers view
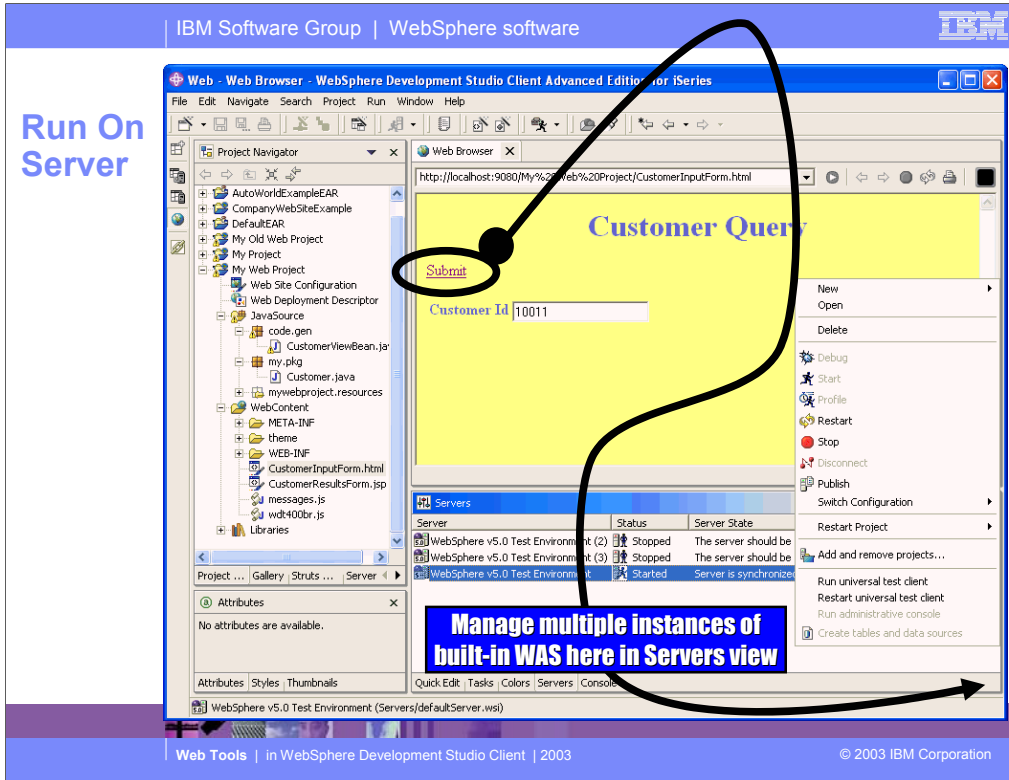
When you do Run On Server, the Servers view is started at the bottom of the perspective, and in it you can see the status of the servers you have configured. The Web Browser is also started, where the selected (or index.html) file is shown.

As you run your application, the application server (and your application if so coded) writes status information to the console, which is another view that takes focus when it is

WAS Test Environment

- **Full copies of WAS 4.0, 5.0.2 Express and 5.0.2 are embedded in the IDE:**
  - ► **Integrated with Server Tools to enable instant testing of Web projects within WDSc**
    - ✓ Standalone all-in-one testing
    - ✓ No dependency on WAS installation or availability
  - ► **Provides**
    - ✓ Ability to debug live server-side code on client
    - ✓ Support for configuring multiple Web applications
    - ✓ Support for multiple servers run at the same time
    - ✓ Access to the profiling feature that is available in the workbench
    - ✓ Ability to version server configurations
    - ✓ Access to the WAS Administration Client

The WebSphere Test Environment provides a fully function test bed for Web applications without having to setup WebSphere Application Server on a remote server.

AGENDA

E-business Primer

Intro to some technology and terms

Web Tools

In Development Studio Client

Application Server Tools

In Development Studio Client

ISeries Web Tools

In Development Studio Client

Now we will go into a little more detail on the rest of the Application Server Tools capability available, which the Run On Server support hints at.

Application Server Tools

- **Rich set of integrated tools**
  - **Supports:**
    - ✓ Testing Web projects with an Application Server
    - ✓ WebSphere Application Server and Tomcat
    - ✓ **built-in** WAS, **local** WAS/Tomcat, **remote** WAS
  - **Comprised of:**
    - **Server Instances** (where to publish)
      - ✓ Refer to a particular WAS or Tomcat installation
      - ✓ Are associated with one Server Configuration
    - **Server Configurations** (what to publish)
      - ✓ Define EAR files (WAS) or WAR files (Tomcat) to deploy
        - ■ EAR files map to Enterprise Application projects (EAR projects)
        - ■ WAR files map to Web projects
      - ✓ Can be associated with multiple Server Instances
    - **Server Projects**
      - ✓ Just a way to group instances and configurations

Servers and *configurations* are used to test your projects. Servers identify where you can test your projects. Server configurations contain setup information. You can either have WebSphere Studio create the servers and configurations automatically for you, or you can create them using a wizard.

You can create servers and configurations that can run resources from the following types of projects:

Web projects, which may contain JSP files, HTML files, servlets, and beans

Enterprise Application projects, which may contain JavaTM Archive (JAR) files or Web Archive (WAR) files or both, and pointers to other Web or J2EE projects

J2EE projects, which contain enterprise beans

Java Application client projects

After testing your application locally, you can publish the application either locally or to another machine.

Server Perspective

- **For managing your application servers:**
  - **Various views:**
    - **Navigator** view
      - ✓ Standard Eclipse project explorer
    - **Servers** view
      - ✓ Lists all defined server instance/configuration pairs
      - ✓ For starting, stopping, restarting and publishing
    - **Server Configuration** view
      - ✓ Shows all server instances and server configurations
      - ✓ Doesn't show the Server Projects they exist in
    - **Console** view
      - ✓ Shows data written to standard out by the Web applications
    - **Web browser** view
      - ✓ Once a Web application starts running
    - **TCP/IP monitor** view (you must open this view)
      - ✓ Shows all traffic between Web browser and application server

The Server Tools feature provides you with several views. Here are the views you can monitor and manage your servers and server configurations:

Server Configuration view

The Server Configuration view (similar to the one shown below) allows you to manage the servers and configurations. This view displays a list of all the servers that reside in your workspace (without displaying their associated server projects).

Servers view

The Servers view (similar to the one shown below) allows you to manage the servers and configurations. This view displays a list of all your servers and configurations that are associated with that server. You can use this view to start, start in debug mode, restart, or stop the servers.

TCP/IP Monitor view

The TCP/IP Monitor is a simple server that monitors all the requests and the responses between a Web browser and an application server. If you want the TCP/IP Monitor view to open when there is TCP/IP activity, select the **Show TCP/IP Monitor view when there is activity** check box on the Preferences - Server - TCP/IP Monitor.
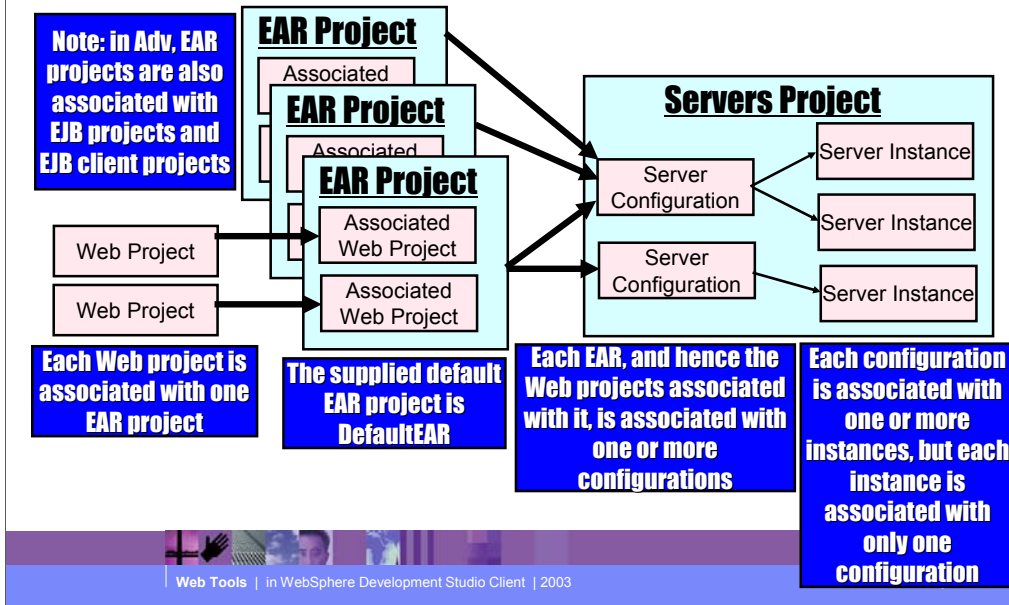
This shows the relationships between EAR projects and Server projects, which contain server configurations.

Each Web project is associated with one EAR project, which is how that project gets published or deployed.

Each EAR project is associated with one or more configuration, which is one or more instances. Eg, local WAS might be one instance, and remote WAS on iSeries might be another. These might share the same server configuration. In the servers view, you are seeing the server instances.

Behind the scenes there is one project named Servers that hold all the server configurations and instances.

Server Tools

Server - WebSphere Development Studio Client Advanced Edition for iSeries

File   Edit   Navigate   Search   Project   Run   Window   Help

Navigator
- AutoWorldExample
- AutoWorldExampleEAR
- CompanyWebSiteExample
- DefaultEAR
- My Old Web Project
- My Project
- My Web Project
- Servers
  - defaultConfiguration.wsc
  - defaultConfiguration2.wsc
  - defaultConfiguration3.wsc
  - .project
  - defaultServer.wsi
  - defaultServer2.wsi
  - defaultServer3.wsi
- toras07m_ORDLIB

Server Perspective

When you start a server instance, the Web projects associated with it are available to be run via the built-in browser

Server Configuration
- Servers
  - WebSphere v5.0 Test Environment
  - WebSphere v5.0 Test Environment (2)
  - WebSphere v5.0 Test Environment (3)
- Server Configurations
  - WebSphere Server Configuration
    - DefaultEAR
      - My Web Project
  - WebSphere Server Configuration (
  - WebSphere Server Configuration (

Right click to add EAR projects to configurations

New
Open
Delete
Add and remove projects...

Servers

Server	Status	Server State
ment (2)	Stopped	The server should be re
ment (3)	Stopped	The server should be re
ment	Started	The server should be re

WebSphere v5.0 Server Configuration (Servers/defaultConfiguration.wsc)

Here you see an example of the Server perspective.

Server Tools

- **Remote Publishing**
  - ▶ **You can use Servers view to manage a remote WAS instance**
    - Start, stop, restart, publish
    - Cool!
  - ▶ **But**
    - You must first install the supplied Agent Controller on the server, and then start it
    - This is made possible by PTF for iSeries
      - ✓ See www.ibm.com/software/awdtools/wds400

- **When must you restart WAS?**
  - ▶ **The documentation will tell you**

WebSphere Studio allows you to test projects on a remote installation of WebSphere Application Server. Before you can test on a remote installation of WebSphere Application Server, you need to specify remote server information for your server. To test your project remotely, you must install WebSphere Application Server on the remote machine where you want to test your project. **Important:** It is recommended that you install WebSphere Application Server before you install Agent Controller. Agent Controller is included with most configurations of WebSphere Studio.

AGENDA

E-business Primer

Intro to some technology and terms

Web Tools

In Development Studio Client

Application Server Tools

In Development Studio Client

ISeries Web Tools

In Development Studio Client

Finally, we drill down now on the iSeries-specific tools

iSeries Web Tools

- **iSeries Web Tools, at a glance**
  - ► **Tools optimized for iSeries developers!**
    - ■ **Host Information wizard**
      - ✓ Set runtime information such as library list and sign-on information, to be used by glue generated by all Web Interaction wizards for this Web project
    - ■ **Web Interaction wizard**
      - ✓ You define the parameters to a *PGM/*SRVPGM, wizard generates input JSP prompting for input parm, output JSP showing output parms, and all the glue in-between
      - ✓ Or you pre-create the input and/or output pages, and map the input/output fields on the pages to the input/output parameters in the *PGM/*SRVPGM, and it generates the glue to bind them
    - ■ **Web Components palette**
      - ✓ Web GUI Widgets that support DSPF-like attributes such as error checking by datatype, and edit-code and edit-word

The tools specifically for iSeries developers are all designed to allow you to easily create Web applications that use RPG or COBOL for the business logic. This means:

 - a wizard to specify runtime information such as the library list

 - a wizard to generate Web pages that call RPG or COBOL programs/procedures. This is just like the Java Bean Web Pages wizard, but it calls programs or procedures instead of Java beans.

 - a palette of parts that have attributes SDA users will recognize.

The iSeries Web Tools Run-time Configuration wizard is used to define the host-related run-time information. When this is done, you can unit test a Web application that communicates with an ILE program on an iSeries host without deploying the JSP files and servlets to the host.

You use the wizard to identify the host where your ILE program is located, to define and modify your run-time library list, and to specify whether you want to see the details for any errors that occur during the run time of your Web application.

New for 5.1, you can also specify how to get the user Id and password, and a current library and initial command to call for things like file overrides.

Web Interaction Wizard

- ● **Two modes to interaction wizard:**
  - **1. Generate input/output Web pages**
    - ■ **Given the parameter description of the API to call**
  - **2. Generate mappings**
    - ■ **Given the input/output pages**
    - ■ **Given the parameter description of the API to call**
    - ■ **Given the mappings**
      - ✓ between input parms & input fields
      - ✓ between ouput parms & output fields

You can create a Web interaction by using the Web Interaction wizard with the JSP files that you create. The wizard adds coding within the JSP files, and generates the iSeries run-time classes used to communicate with your business processes.

The Web Interaction wizard helps you define input and output pages, fields, and parameters. You also use the wizard to identify the host program or Java bean to call for each interaction. An interaction is defined by the communication that occurs between your Web pages and the business processes. In addition, you can use the wizard to define message handling for the Web pages in your application. The wizard uses all of this data to add code to the JSP files and to generate the iSeries classes that communicate between the Web pages in your application and the ILE host programs or Java beans that perform the business processing for your application.

Invoking Web Interaction Wizard

You can invoke Web Interaction Wizard from popup menu of an action in the Struts Web Diagram editor

Use New->Other…, Then Web->Struts to open a Web Diagram Editor

Use Web Diagram editor:

1.- place input and output web page nodes using right click, New->**Web Page Node**

2.- place action nodes using right click, New->**Action Mapping Node**

3.- draw connection lines from input page to action: rt-click, **Connection**

4.- invoke Web Interaction wizard by right clicking on action node, selecting **Open iSeries web interaction wizard**

You can also invoke Web Interaction Wizard from Web project toolbar

Create or modify a web interaction

You can use a Web diagram to visually represent a Web interaction and then invoke the Web Interaction wizard to generate the necessary iSeries run-time classes for the interaction. This can be done several ways:

Create a representation of a Web interaction in the Web diagram by adding JSP file nodes and an action icon to the free-form surface.

Create the Web pages (JSP files) in your Web project, and drag and drop the files on to the diagram free-form surface.

Create the Web interaction before using the Web diagram.

Right-click the project name in the J2EE Navigator view and select **New > Other**.

In the left pane of the wizard, expand **Web** and click **Struts**.

Select **Web Diagram** in the right pane and click **Next**.

Enter a name for the diagram in the **File name** field (the .gph file extension is automatically applied to the name), and click **Finish** to create the file in your project. The file opens on a free-form surface in the editor view.

iSeries Web Tools
► iSeries Web Interaction Wizard
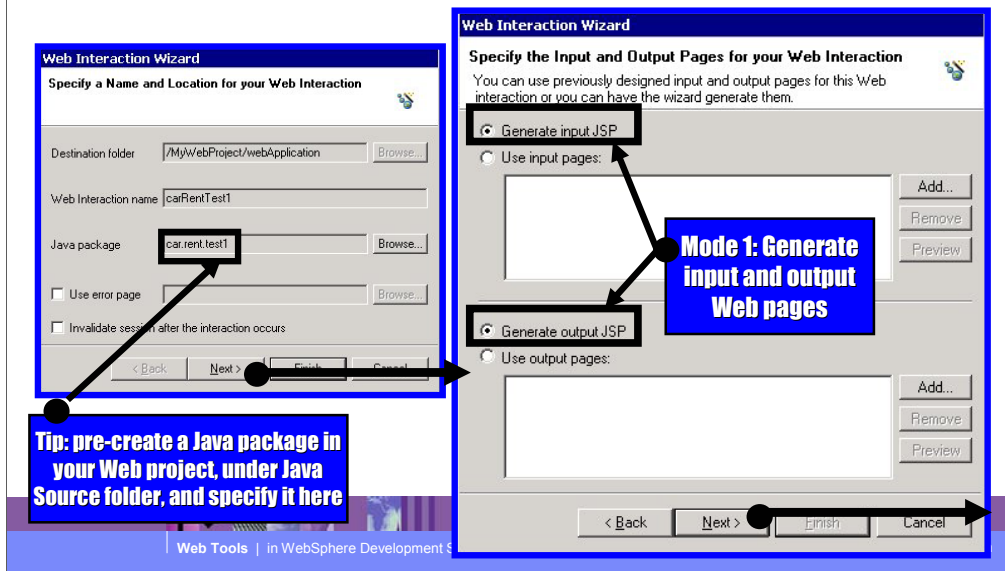
In the first page of the Web Interaction wizard, you specify the name for the interaction (arbitrary), and the Java package into which the code will be generated (tip, create a package first), and whether to use your own error page or not.

In the second page of the Web Interaction wizard, you specify whether you want to use existing Web pages (and subsequently map their fields to parameters), or you want to have the wizard generate the Web pages for you.

On the third page of the wizard, you first identify the program or procedure that you wish to invoke when Submit is pressed on the first Web page.

Note that previously defined programs or procedures will appear in the tree on the left, for easy re-use.

After specifying the program or procedure and pressing OK, you will be prompted for the parameters …

iSeries Web Interaction Wizard

Web Interaction Wizard

Specify the Input and Output Parameters for your iSeries Host Program

Use this page to define the input and output parameters for your iSeries host program.

○ Use an iSeries ILE program   ○ Use a Java bean   ○ No program call

Describe parameters

Program call definitions

Add Program...   Add Parameter...   Add Structure...

☐ CarRent
  ■ carClass
  ■ carMake
  ■ carColor
  ■ retPlate

Edit Parameter

Parameter name: carClass
Data type: character
Structure name:
Length: 10
Precision:

► **For arrays**

Count:
Usage: input
Initial value: *ANY

☐ Advanced...

Specify database reference field   Specify...   Synchronize
Show database field definition   Show...

OK   Cancel

Import PCML...   Synchronize All

< Back   Next >

► **Input: read by program**
► **Output: updated by program**
► **Input/Output: both**

► **2a. Press "Add Parameter",**
► **2b. Enter parameter information**
► **2c. Press Ok**
► **2d. Repeat for each parm**

► **3a. Pre-define structures before referencing them for parms**

Reference DB fields!

2a **2b** **2c** **3**

Still on the third page of the wizard, you now define each parameter of the program or procedure. The parameter can either be simple or a structure, and either can be an array.

iSeries Web Interaction Wizard
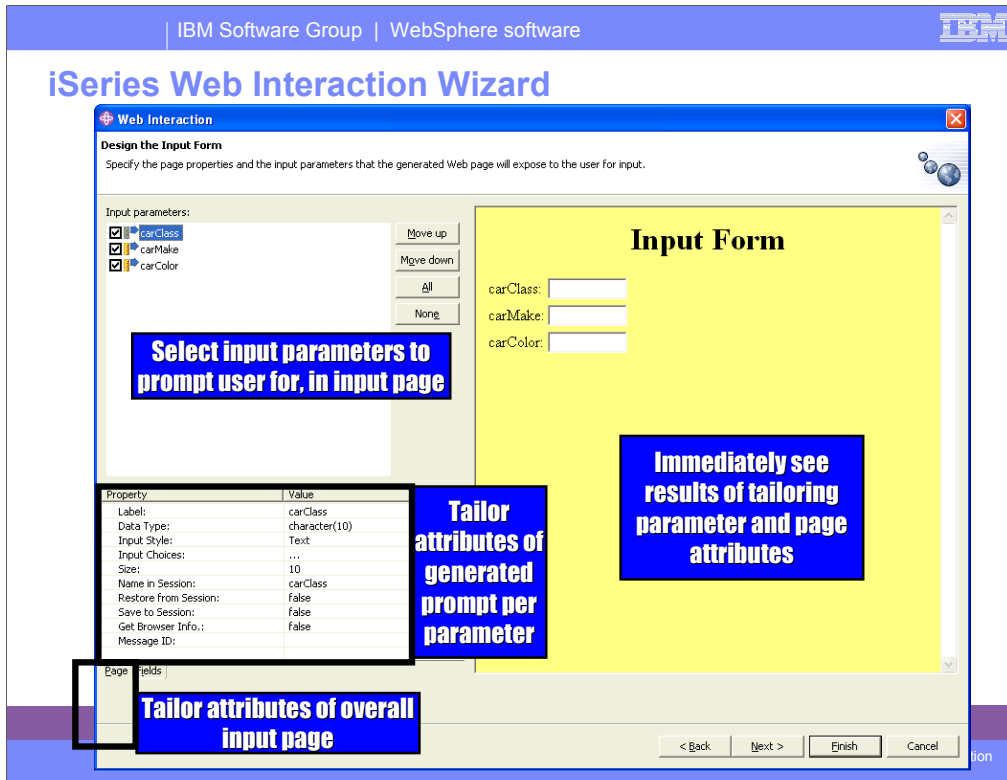
On the fourth page of the wizard, you refine how the generated input Web page will look. You select which input parameters to prompt for, and for each use the property sheet to affect the prompt label, UI field type, and so on. You can also select the Page tab at the bottom, and use the property sheet to change the page's title, color and more.

You can also edit the generated page after the wizard runs, of course, but this makes it quite easy to affect the generation, without deep HTML skills.
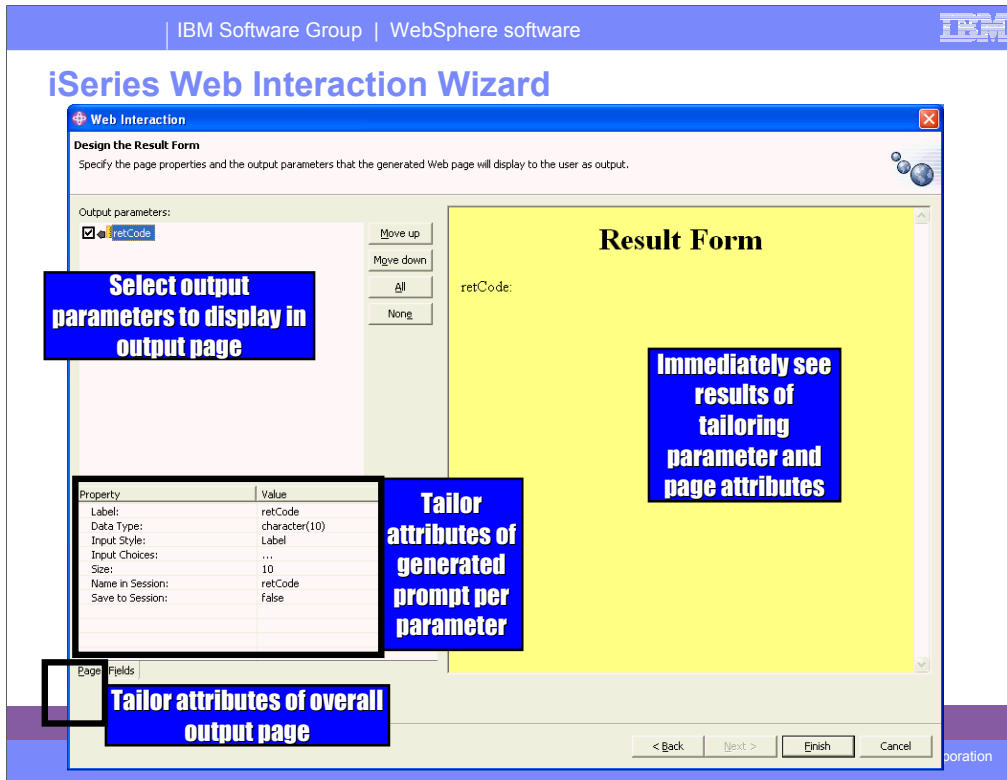
This fifth and final page of the Web Interaction wizard allows you to affect the generated output Web page. again, you can select which output parameters to show, and what to generate for them, and you can affect the overall Web page.

Pressing Finish on this page generates the Java code and JavaServer pages.

iSeries Web Interaction Wizard

If the Web Interaction wizard was launched from the Web Diagram Editor, then the nodes will now change to be colored, indicating that the files represented by those nodes now exist.

Results of Run on Server for Web Interaction Wizard

Web Tools | in WebSphere Development Studio Client | 2003

© 2003 IBM Corporation

Of course, you can now run the generated Web pages by selecting the input page and the Run On Server option.

On the first page, you enter information and press Submit. This calls the generated Java class (a Struts action), which will read the user entered-data and then call the program or procedure you specified, passing the user-entered data as parameters. It will then show the generated output page, passing the output parameters to it as data.

iSeries Web Components

► **Web Components: Visual Custom Tags**
  ▪ **Emerging technology**
  ▪ **Blessed by IBM and SUN**

► **Tag library defined by:**
  ▪ **www.sun.com**
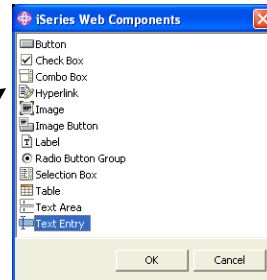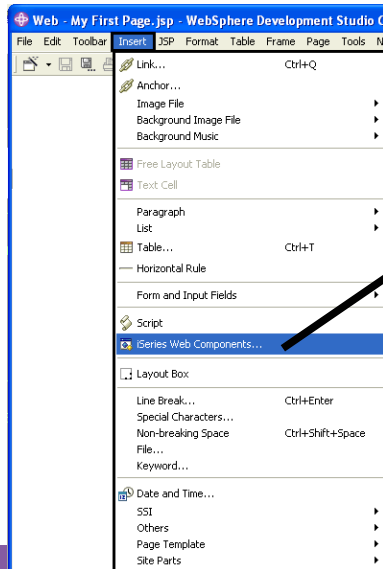
iSeries-unique Web widgets

While the Web Interaction wizard allows you to generate Web pages, sometimes you prefer to create them yourself. To help with this, IBM supplies a palette of Web user interface parts that are designed for iSeries programmers. To access these, you can use the iSeries Web Components menu action under the Insert menu, in the Web Page Designer when editing a JavaServer Page file.

iSeries Web Components

Another way to access the iSeries Web Components palette is to from the Snippets tab in the bottom right of the Web perspective. Either way, once you have the palette, you can drag and drop the widgets to the Web page.
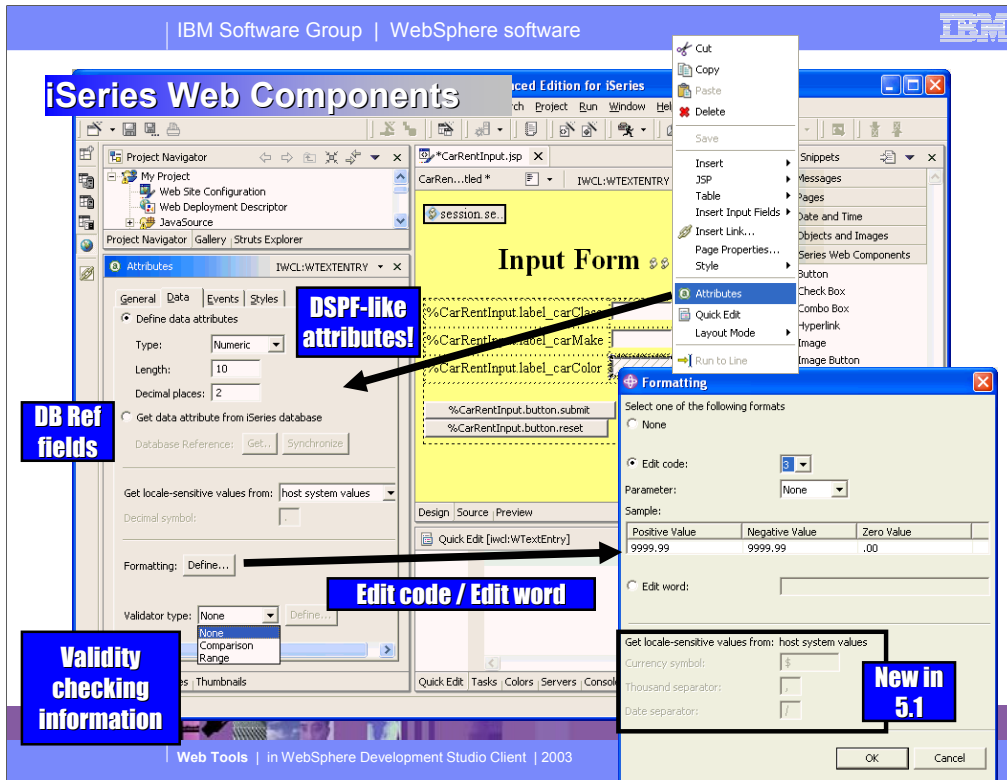
When you select an iSeries Web component in the page designer, the attributes view in the lower left allows you to specify information about the component. This information is often display file like. For example, for an entry field you can set its datatype, length, decimals, edit-code/edit-word, and validation information. All the javascript to support this is generated for you.

New as the 5.1 release is support for internationalization in these widgets:

  - you can specify %key-value for the label, and the actual value is retrieved from the project's resource bundle file, which is created when the project is created if its struts-enabled.

  - you can specify how to determine the decimal separator: from the OS/400 system value, or according to the user's Web browser, or hard-coded

  - you can specify how to determine the currency symbol, thousands-separator and date separator for edit-codes and edit words: again, from the OS/400 system or job values, from the user's Web browser codepage, or hard-coded.

One special Web component worth an extra mention is the table component. This is the equivalent of a subfile.

To use a table, drag and drop it, and use the attributes page to add the columns (or drag and drop other components into the table). If you want to populate and manage the table from RPG or COBOL logic, then specify the service program that will do that. This service program has to behave a certain way …

… the service program to manage a subfile needs to have four procedures named INIT, PGUP, PGDN and CLUP. This procedures are called when the user pages up and down, and are called with the table is initialized and closed. There

What's new in iSeries Web Tools for Version 5.0

- Re-written Page Designer for WYSIWYG Web design
- V4.0 Design Time Controls replaced with Web Components
  - Uses JSP 1.2 custom tag technology
  - Based on Web widgets from Tivoli
- Supports for Struts in Interaction Wizard
  - Launched from new Web Diagram editor
- Many miscellaneous enhancements
- <u>Advanced</u> support for JCA connector

Some enhancements in Version 5.0 of the Web Tools are listed here

What's new in iSeries Web Tools for Version 5.1

- Support for Struts 1.1 and WAS 5.0.2
- Support for internationalization:
  - ✓ Web pages generated with UTF-8 encoding in Web Interaction Wizard
  - ✓ Support in iSeries Web Components for translatable strings coming from a resource bundle
  - ✓ Support for placing translatable strings in resource bundle in generated Web pages in Web Interaction wizard
  - ✓ Support in iSeries Web Components for retrieving editcode/editword dollar sign, thousands-separator and decimal separator from iSeries host or Web page locale.
- Support for importing PCML generated by compilers
- Ease of use enhancements to Web Interaction Wizard

Some of the enhancements in Version 5.1 of the Web Tools

More Information?

► **eclipse.org**
  - **Website for the open source Workbench**

► **WDS and WDSc**
  - **ibm.com/software/awdtools/wds400/**

► **WebSphere Developer Domain**
  - **ibm.com/websphere/developer**

Trademarks & Disclaimers

References in this document to IBM products or services do not imply that IBM intends to make them available in every country.

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

AS/400	IBM(logo)
AS/400e	iSeries
e (logo) business	OS/400
IBM	

Lotus, Freelance Graphics, and Word Pro are registered trademarks of Lotus Development Corporation and/or IBM Corporation.
Domino is a trademark of Lotus Development Corporation and/or IBM Corporation.

C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.
Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.
UNIX is a registered trademark of The Open Group in the United States and other countries.
SET and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.
Other company, product and service names may be trademarks or service marks of others.

Information is provided "AS IS" without warranty of any kind.

All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.

Information in this presentation concerning non-IBM products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. Sources for non-IBM list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. IBM has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-IBM products. Questions on the capability of non-IBM products should be addressed to the supplier of those products.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. Contact your local IBM office or IBM authorized reseller for the full text of the specific Statement of Direction.

Some information in this presentation addresses anticipated future capabilities. Such information is not intended as a definitive statement of a commitment to specific levels of performance, function or delivery schedules with respect to any future products. Such commitments are only made in IBM product announcements. The information is presented here to communicate IBM's current investment and development activities as a good faith effort to help with our customers' future planning.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Photographs shown are of engineering prototypes. Changes may be incorporated in production models.

IBM

Web Tools

in WebSphere Development Studio Client 5.1

Phil Coulthard

coulthar@ca.ibm.com

e business software