



IBM Global Services - IBM eServer iSeries

A32
WDS*c*: Java Tooling
Don Yantzi



Miami, FL

Oct 17-21, 2005

why "i"? it's simple.

© 2005 IBM Corporation

Agenda

- ➔ ▪ Getting Started
- The Java Editor
- Visual Editor
- Java Views
- Compile / Run / Debug
- iSeries Additions
- Reference

Getting Started

- Java Perspective
 - Customizes the workbench for Java development
 - Java related views, actions, wizards
- Java Projects
 - Used for organizing your Java code
 - Associated properties
 - Compiler settings
 - Javadoc location
 - Validators
 - Build path (same as classpath)
 - Associated builders (compiler)
 - When you save changes to a Java source file the builder automatically compiles it
 - Compiler is incremental so it only compiles your changes!

Projects provide the highest level of grouping resources (source files, graphics, executables, ...) in the workspace. Projects always have a type such as Java, WebFacing, Web. Those projects that are not linked to a specific type of development should use the "Simple" type. The project type specifies what properties are associated with the project. For example Java projects have a build path property where users can specify other projects and .jar files that this project requires. Projects also have an associated builder which knows how to build resources in the project (i.e. compile .java files into .class files.) By default projects map to a subdirectory of the workspace directory on the local file system with the same name as the project.

Web Perspective and Projects

- Customizes Workbench for Web development
 - Java development is frequently done as part of a Web application
 - Servlets
 - Action classes (struts)
 - Java business logic
- Web projects are also Java Projects
 - Include same properties and actions as Java projects
 - Plus properties and actions related to Web development
 - Follows J2EE guidelines for file structure of a Web application
 - All Java classes go under the Java Resources folder
 - All Web resources (HTML pages, JSPs, graphics) go under the WebContent folder

Perspectives are used in the Workbench to provide coherent selection and layout of views related to a specific type of development (Java, XML, WebFacing, ...) By default the Java perspective shows the packages, outline, hierarchy and tasks views. The Outline view is a standard Workbench view for showing the outline of the resource currently opened in the editor. It works for Java and other resource types like XML and SQL scripts. The Tasks view is another standard workbench view and shows all errors, warnings and tasks in the workspace not just Java errors. The packages and hierarchy views are Java specific. Each of the four views are outlined in more detail on the following slides.

•Packages view

- Shows only the Java projects in your workspace
- Collapses package directories into a single folder in the tree view

•Hierarchy view

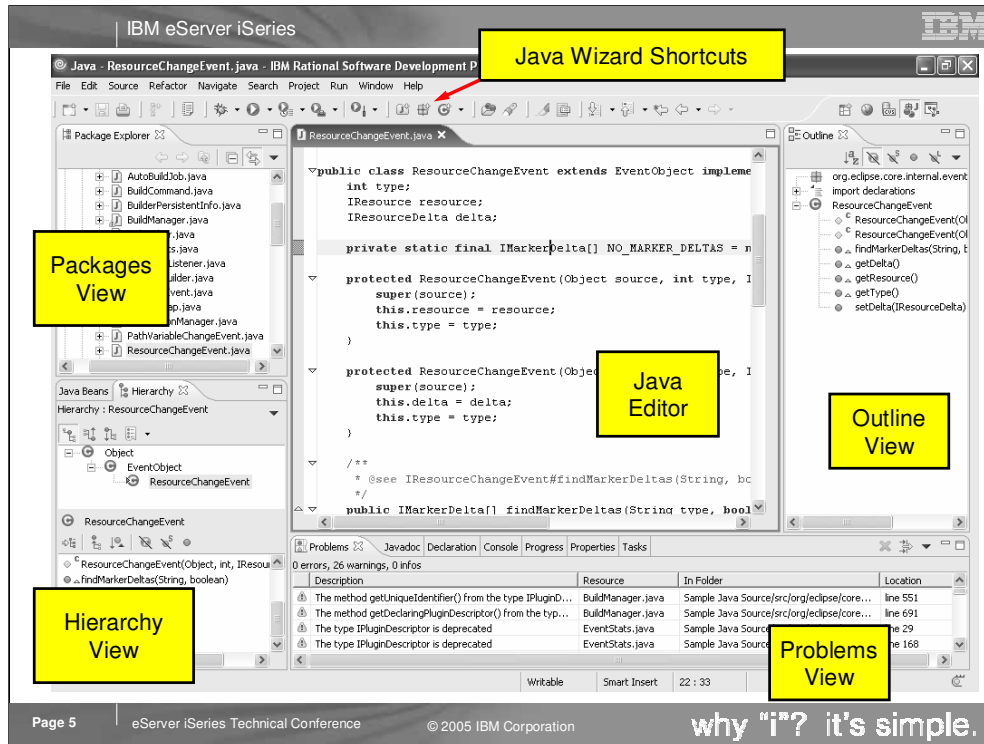
- Inheritance hierarchy of the Java class

•Outline view (standard workbench view)

- High level view overview of the structure of your Java class

•Tasks view (standard workbench view)

- Shows compile errors and warnings



Here you see a screen shot of the Java perspective (default layout.) All open perspectives are shown in the left hand column of the workbench window. Simply click the perspectives icon to switch to the open perspective (this applies to all perspectives not just the Java perspective.) Each perspective typically contributes shortcuts to the workbench toolbar. The Java perspective adds some icons for creating a new Java project, package, class, interface and scrapbook page.

IBM eServer iSeries

Web - LogonServlet.java - IBM Rational Software Development Platform

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer Gallery LogonServlet.java X Palette Outline

com.ibm.samples.controller
import declarations
LogonServlet
doPost(HttpServletRequestRequest)
doGet(HttpServletRequestRequest)
performTask(HttpServletRequestRequest)
handleError(HttpServletRequestRequest)
printStackToString(Throwable)

```

public void performTask(javax.servlet.http.HttpServletRequest request,
javax.servlet.http.HttpServletResponse response)
{
    try {
        String userID = request.getParameter("userID");
        String password = request.getParameter("password");
        String driver = request.getParameter("driver");
        String url = request.getParameter("url");

        // Obtain the session object from the session
        javax.servlet.http.HttpSession session = request.getSession();

        DBConnectionSpec connectionSpec = new DBConnectionSpec();
        connectionDBBean = (DBConnectionBean) session.getAttribute("connectionDBBean");
        if (connectionDBBean == null)
            connectionDBBean = new DBConnectionBean();
        connectionDBBean.setUser(userID);
        connectionDBBean.setPassword(password);
        connectionDBBean.setDriver(driver);
        if (request.getParameter("dataSourceName") != null)
            connectionDBBean.setURL(url);
        else
            connectionDBBean.setDataSourceName(getInitParameter("dataSourceName"));
    }
}

```

Editing Java servlet in the Web perspective

J2EE Project Explorer

Properties Quick Edit Servers Console Problems

0 errors, 26 warnings, 0 infos

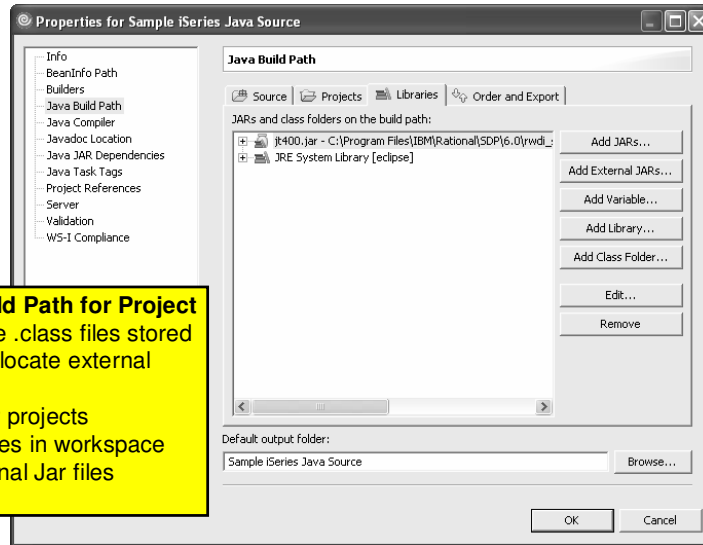
Description	Resource	In Folder
The method getUniqueIdentifier() from the type IPluginDescriptor is deprecated	BulkManager.java	Sample Java S
The method getDeclaringPluginDescriptor() from the type IPluginDescriptor is deprecated	BulkManager.java	Sample Java S
The type IPluginDescriptor is deprecated	EventStats.java	Sample Java S
The type IPluginDescriptor is deprecated	EventStats.java	Sample Java S

Client ... Page Data Styles


Writable Smart Insert 40 : 28

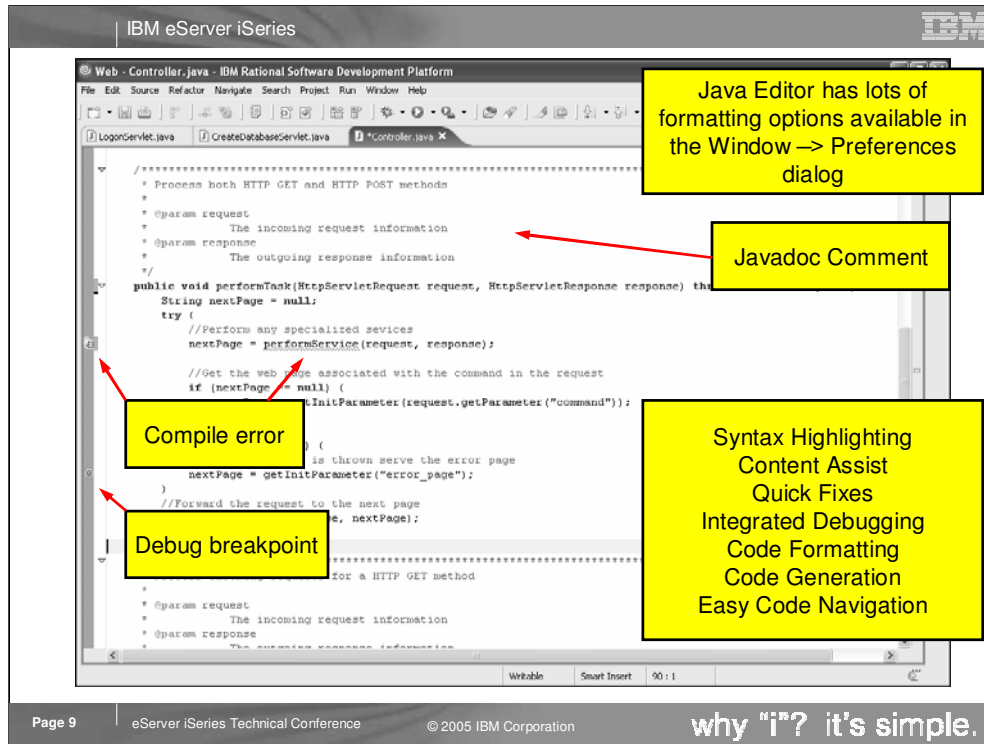
Page 6 eServer iSeries Technical Conference © 2005 IBM Corporation why "i"? it's simple.

Java Build Path



Agenda

- Getting Started
-  The Java Editor
- Visual Editor
- Java Views
- Compile / Run / Debug
- iSeries Additions
- Reference



After completing the new Java Class wizard the .java file is created in the workspace and the file is opened in the Java editor.

Content Assist – invoke by Ctrl + Space

- Suggesting class, method and field names
 - Select class / method / field from list of suggestions
 - Javadoc comments shown for selection
- Overriding Methods
 - Use code assist to display list of methods in super classes
 - Select method from list to override in current class
- Adding import statements
 - Invoke code assist after entering class name to automatically import the class

The screenshot shows a code editor with the following code:

```
try {
    DriverManager.registerDriver(new AS400JDBCDriver());
    Connection connection = DriverManager.getConnection(
} catch (SQLException e) {
}
```

A tooltip is displayed over the `getConnection` method call, showing the following information:

Attempts to establish a connection to the given database URL. The DriverManager attempts to select an appropriate driver from the set of registered JDBC drivers.

Parameters:
url a database url of the form jdbc:subprotocol:subname
user the database user on whose behalf the connection is being made
password the user's password

A list of suggestions is shown to the right of the tooltip:

- getConnection(String url) Connection - DriverManager
- getConnection(String url, Properties info) Connection - DriverManager
- getConnection(String url, String user, String password) Connection - DriverManager

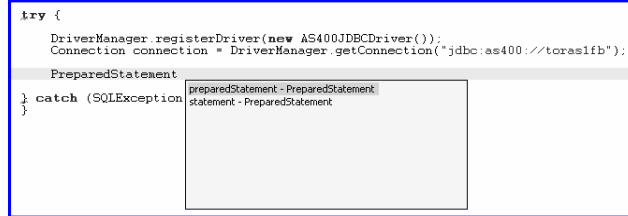
Content assist provides a list of class names, interfaces, fields and methods that are available in the current scope. To invoke content assist press Ctrl + space (or right click and select content assist from the popup menu). The list of options is subset by what you have already typed. For example: entering get and then pressing Ctrl + space will show only methods starting with get. If you keep typing then the list is further subset.

Tip: When using the workbench the user can double click on the title bar of any editor or view to maximize the editor / view to take up the entire workbench window.

More Content Assist – Methods and Variable Names

- Displaying parameters for methods
 - Invoke content assist inside the method call's parameters
 - MyClass.myMethod(*Ctrl+space*)
- Suggesting variable names

```
try {  
    DriverManager.registerDriver(new AS400JDBCDriver());  
    Connection connection = DriverManager.getConnection("jdbc:as400://toras1fb");  
    PreparedStatement  
} catch (SQLException  
}
```

A screenshot of an IDE showing a code editor with a blue border. The code is a try-catch block. The line 'PreparedStatement' is highlighted in grey. A pop-up window shows two suggestions: 'preparedStatement - PreparedStatement;' and 'statement - PreparedStatement'.

```
preparedStatement - PreparedStatement;  
statement - PreparedStatement;
```

Content assist will display parameters for methods and suggest variable names.

And More Content Assist - Javadoc

Content assist for
HTML inside
Javadoc comments

```
/**
 *
 * Sample class to highlight the features of the
 * <b>Java Development Tools</b> in WebSphere Development Studio Client V5.0.
 * This class reads an OS/400 <code>the results
 * to the console.
 *
 * @author yantzi
 */
public class JDBCExample {
    public static void main(
        try {
```

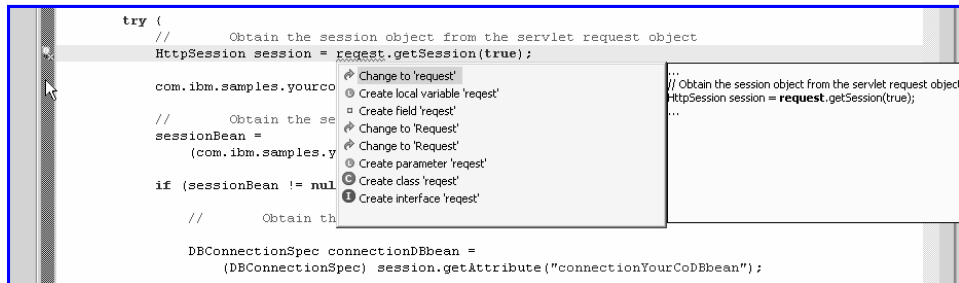
```
/**
 *
 * Sample class to highlight the features of the
 * <b>Java Development Tools</b> in WebSphere Development Studio Client V5.0.
 * This class reads an OS/400 physical file using JDBC and displays the results
 * to the console.
 *
 * @author yantzi
 */
public class JDBCExample {
    public static void main(
        try {
            new AS400JDBCDriver();
            Manager.getConnection("jdbc:as400://toras1fb");
```

Content assist for
Javadoc comments

Content assist is also available for Javadoc comments inside the Java editor. Content assist for Javadoc can provide suggestions for HTML tag names and Javadoc keywords. Again the list will be subset by what is already typed.

Quick Fixes

- The editor can suggest fixes when you have errors in your code
- To invoke "Quick Fixes":
 - Click on the light bulb in the left hand margin, or
 - Press Ctrl + 1
- Editor lists suggestions along with a preview of the code if the fix is applied



```
try (  
    // Obtain the session object from the servlet request object  
    HttpSession session = request.getSession(true);  
  
    com.ibm.samples.yourco  
  
    // Obtain the se  
    sessionBean =  
    (com.ibm.samples.y  
  
    if (sessionBean != nul  
  
    // Obtain th  
  
    DBConnectionSpec connectionDBbean =  
    (DBConnectionSpec) session.getAttribute("connectionYourCoDBbean");
```

Templates

- Templates can be used to insert frequently used code patterns with content assist, for example:
- Lots of templates are predefined for Java Tools
 - Iterate over an array using for loop
 - `for <ctrl + space>`
 - Inserting a try / catch block
 - `try <ctrl + space>`
 - Public, protected or private method
 - `public <ctrl + space>`
- Define your own in the Java -> Templates preferences page
 - Insert a customized Javadoc comment for your methods

Creating Templates

Preferences

Templates

Create, edit or remove templates

Name	Context	Description
<input checked="" type="checkbox"/> <pre>	javadoc	<pre> </pre>
<input checked="" type="checkbox"/> @author	javadoc	author name
<input checked="" type="checkbox"/> cast	java	dynamic cast
<input checked="" type="checkbox"/> catch	java	catch block
<input checked="" type="checkbox"/> do	java	do while statement
<input checked="" type="checkbox"/> else	java	else block
<input checked="" type="checkbox"/> elseif	java	else if block
<input checked="" type="checkbox"/> false	javadoc	<code>false</code>
<input checked="" type="checkbox"/> for	java	iterate over array
<input checked="" type="checkbox"/> for	java	iterate over array with t...
<input checked="" type="checkbox"/> for	java	iterate over collection
<input checked="" type="checkbox"/> if	java	if statement
<input checked="" type="checkbox"/> false	java	if else statement

Preview:

Pattern:

```
for (int ${index} = 0; ${index} < ${array}.length;
    ${array_type} ${array_element} = ${array}[${
    ${cursor}
    ]
)
```

Use code formatter

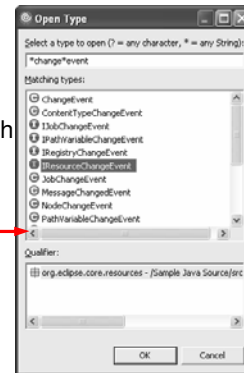
"Name" is used to match in content assist for listing templates

This code is inserted into the editor when the template is selected

Substitution variables can be filled in automatically or by the user

Navigating Through Your Code

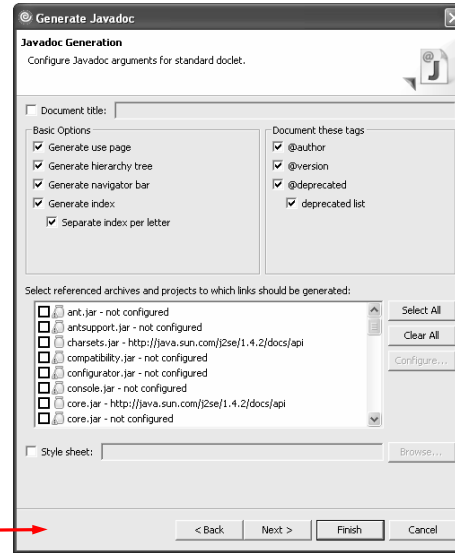
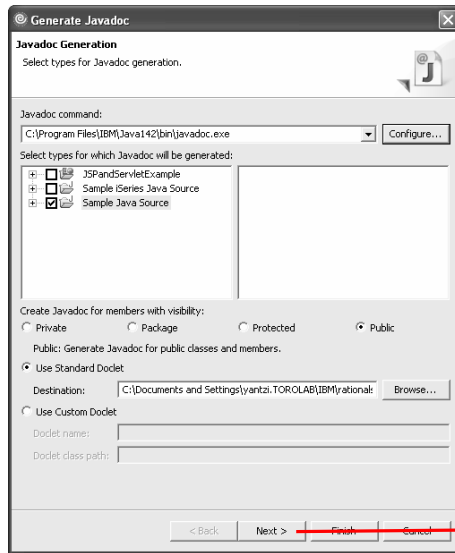
- There are lots of different ways to quickly navigate through your Java classes, here are some tips:
 - To open a class
 - Use the “Open Type” Dialog
 - Shortcut: Ctrl + Shift + T
 - Select the class name in the editor and press F3
 - Use the pop-up outline view in the editor
 - Shortcut: Ctrl + O
 - Use the navigation aids in the icon bar
 - Go to last edit location
 - Back to last location, forward to next location
 - Go to next problem, previous problem

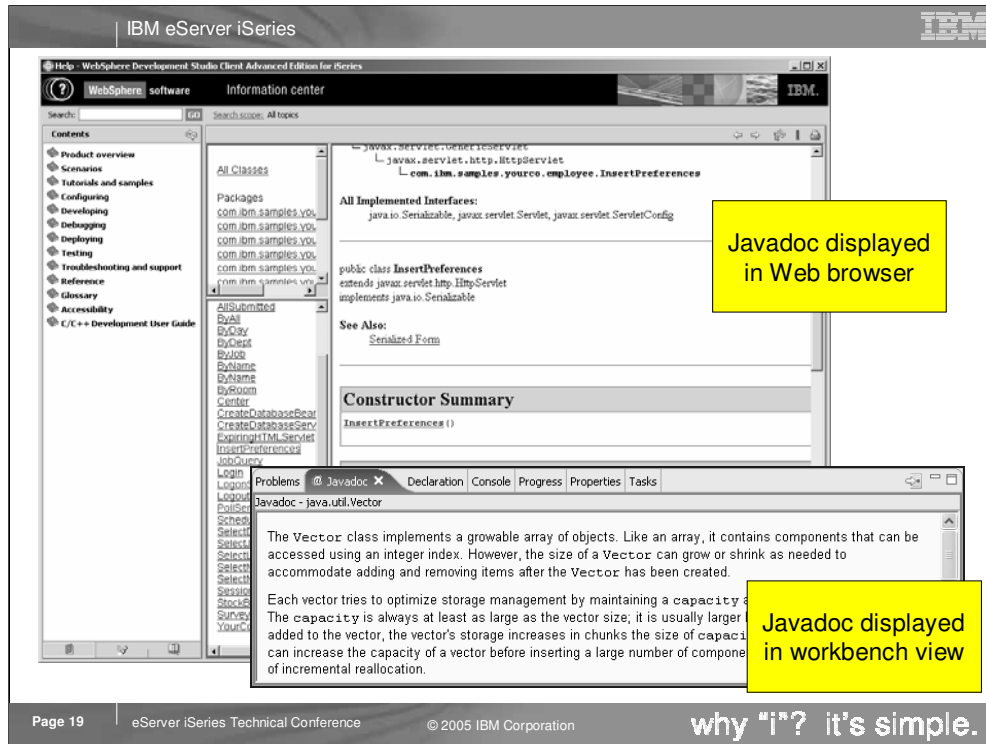


Javadoc – Easiest Way to Document Your Code


- **Java Documentation**
 - Standard way of documenting Java code that automatically generates HTML documentation for your classes
 - You insert Javadoc comments for your class, public constants and methods then run the class through the javadoc utility
 - Comments can include HTML markup and specialized Javadoc keywords
- **Generating Documentation**
 - Select the Project > Generate Javadoc... menu
 - Customize javadoc generation options
- **Viewing Documentation**
 - Select your project in the Workbench and select Navigate > Open External Javadoc
 - or
 - Use Javadoc view in the workbench

Exporting Javadoc

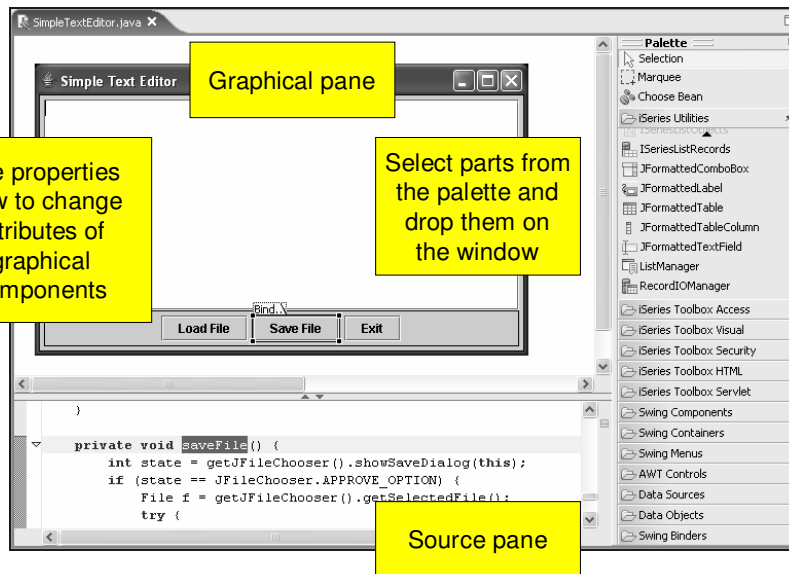




Agenda

- Getting Started
- The Java Editor
-  ▪ Visual Editor
- Java Views
- Compile / Run / Debug
- iSeries Additions
- Reference

Java Visual Editor

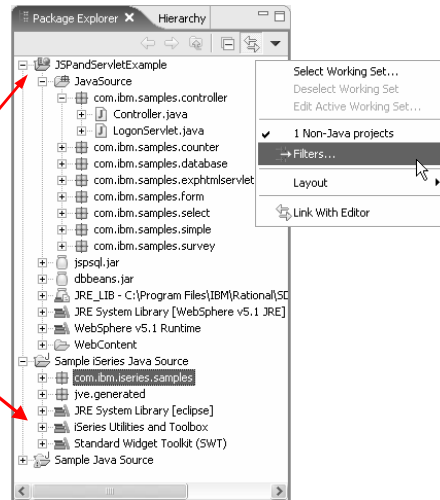


Agenda

- Getting Started
- The Java Editor
- Visual Editor
-  ▪ Java Views
- Compile / Run / Debug
- iSeries Additions
- Reference

Package Explorer

- Central location for managing your Java resources
 - Shows only Java related projects
 - Web projects are also Java projects
- Use **Working Sets** to show only a subset of all Java projects
- Displays external jar files included in the project's build path
 - Can be turned off with filters
- Use filters to control what is shown in the packages view



Packages are shown as their corresponding folders (com / ibm / etools / iseries / ...) which takes up a lot of the room in the tree view. The Packages view addresses both of these issues by showing only Java projects (and those projects which have a Java nature like Web projects) and showing Java packages as a single entry in the tree view. The Packages view also lets you see any referenced libraries directly in the tree view. Reference libraries are setup in the properties for the Java project and are covered later in this presentation.

IBM eServer iSeries

Outline View

- Works in concert with the Java Editor
 - Shows imports, fields, methods
 - Can subset what is shown
 - Can sort by name
 - Can show only selected member in editor
 - Generate getter and setter methods for fields
 - Generate Javadoc comments

Refactoring allows you to easily rename or move a method and the tool automatically finds and updates all references

Shortcut for menu options for searching

Page 24 eServer iSeries Technical Conference © 2005 IBM Corporation why "i"? it's simple.

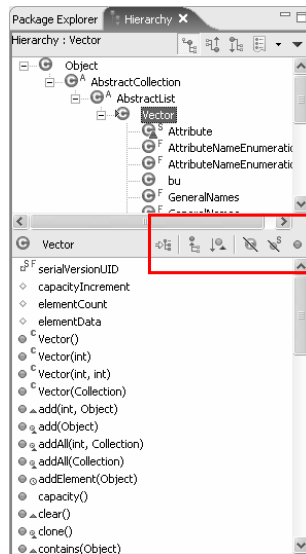
The Outline view shows the structure of the Java class currently being edited. It's main function is to provide you with a high level view of the structure of the class and provide easy navigation in the editor. Selecting a method / inner class in the outline view automatically positions the editor to the corresponding source code. The Outline view also provides many additional actions. These are available when you right click on the different items (classes, methods, fields) in the Outline view. For example: Right click on a field in the Outline view to see the option to generate standard getter and setter methods for the field. To remove a method right click on the method in the Outline view and select "Delete". The method and its corresponding code are deleted from the editor. The Outline view also provides some additional functionality such as:

- Toggling showing fields on / off
- Toggling showing static methods on / off
- Sorting the list of fields / methods
- Showing only the selected field or method in the editor instead of the entire class file

Hierarchy View

Hierarchy view shows a class relative to its superclass and subclasses.

To open a class in the hierarchy view highlight the class name in any source file or view and press F4.



- Shows selected members in hierarchy at all locations where they are declared.
- Show / hide inherited members
- Show / hide fields
- Show / hide static members
- Show / hide nonpublic members

Here you see the Hierarchy view with descriptions for the various features of the view. The first option (show selected members in hierarchy) lets you select a member from the list at the bottom of the view and the tree view at the top will be updated to show which classes have defined this method. This allows you to quickly see where the method has been overridden.

IBM eServer iSeries

Problems and Tasks Views

Problems view shows build errors and warnings in source code

Error ☹️

Warnings 😊

Filter problems and tasks

Resource specific task

General task

User defined and program detected tasks

Description	Resource	In Folder
The method getParameter(String) is undefined for the ty...	Controller.java	JSPandServletExample/JavaSourc
The method getUniqueIdentifier() from the type IPluginD...	BuildManager.java	Sample Java Source/src/org/eclips
The method getDeclaringPluginDescriptor() from the typ...	BuildManager.java	Sample Java Source/src/org/eclips
The type IPluginDescriptor is deprecated	EventStats.java	Sample Java Source/src/org/eclips
The type IPluginDescriptor is deprecated	EventStats.java	Sample Java Source/src/org/eclips
The type IPluginDescriptor is deprecated	EventStats.java	Sample Java Source/src/org/eclips
The type IPluginDescriptor is deprecated	EventStats.java	Sample Java Source/src/org/eclips

Description	Resource	In Folder	Location
And elseif condition	Controller.java	JSPandServletExample/JavaSource/com...	line 60
Extract method into superclass	Controller.java	JSPandServletExample/JavaSource/com...	line 32
Pickup milk on the way home			

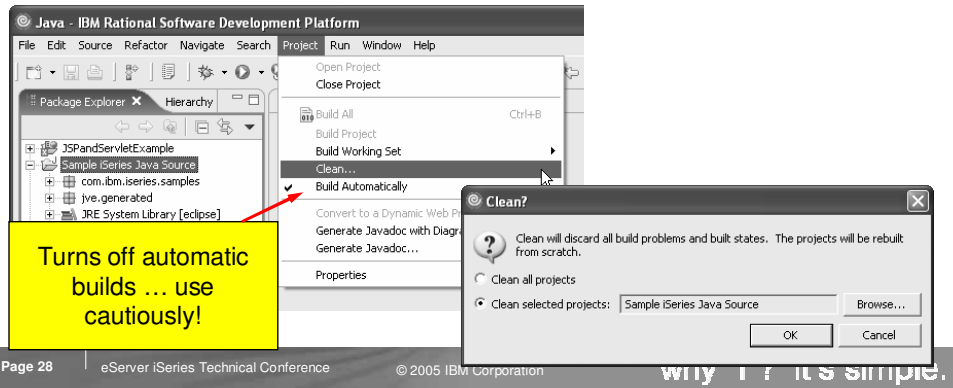
The Tasks view is a standard workbench view showing all errors, warnings and user defined tasks in the workspace. Because it shows all workspace tasks it provides filtering capabilities to scope what is shown in the actual view by the type of problem / task or based on what is selected in the Navigator view or Packages view.

Agenda

- Getting Started
- The Java Editor
- Visual Editor
- Java Views
-  ▪ Compile / Run / Debug
- iSeries Additions
- Reference

Compiling your Java class

- Every time you save your file it is automatically compiled (this is called building in the Workbench)
- Or you can force a recompile
 - Select project in package explorer, then select either
 - Project > Clean...



When you save a Java class (.java file) it is automatically compiled and all errors / warnings are shown in the tasks view. However you can also force a recompile using either the build or rebuild project actions. The Build Project action is an incremental build that builds only resources that have changed since the last build. The Rebuild Project is a full build that discards the previous build stat and rebuilds everything.

Running your Java Application

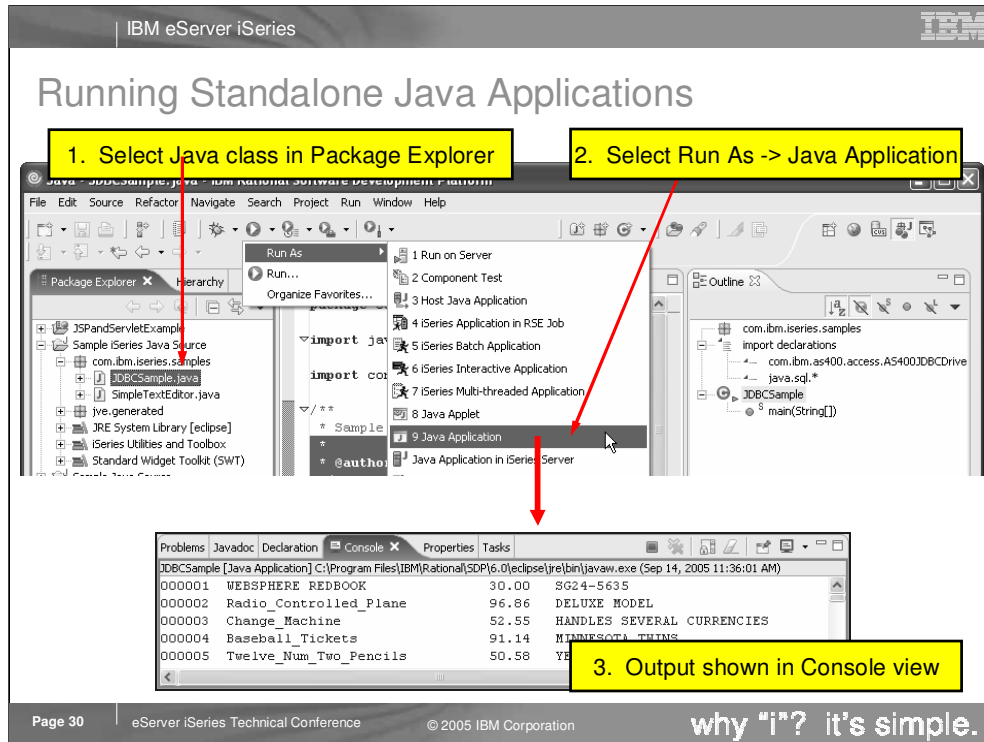
- Standalone Java application
 - Class implements “public static void main” method
 - Run locally, or
 - Run remotely on the iSeries
- Servlet (Web Application)
 - Class extends javax.servlet.http.HttpServlet
 - Servlet can be run on different test environments and servers

How you run a Java class will depend on what type of Java application it is. If it is a standalone Java application (i.e. it contains a public static void main(String[] args) method) then it can be run locally using either the JDK shipped with the workbench (this is the default) or any JDK installed on the local workstation. You can specify which JDK to use in the Java project properties. You can also run the standalone Java application remotely using iSeries extensions.

If the Java class is a servlet then it must be run in either the local WebSphere Test Environment (default) or any other

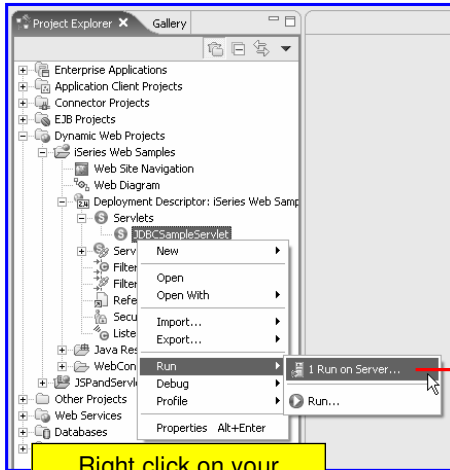
application server environment supported by the workbench. Currently this includes:

- Local WebSphere Test Environment
- Local Tomcat Test Environment
- Local Tomcat Application Server
- Remote WebSphere Application Server

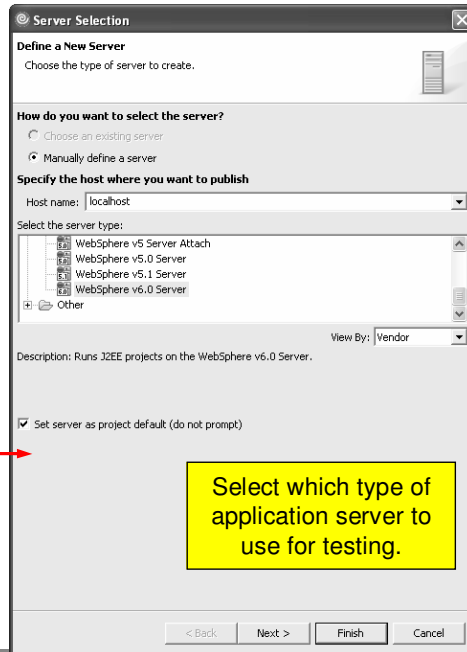


Here you see how to run the standalone JDBCSample Java application used through out this presentation. To run the application simply select the class in the packages view and from the Run icon in the toolbar select Run -> Java Application. If the application writes any output to either System.out or System.err then the workbench console view will appear with the output. If the Java class were a servlet then to run it in the local WTE you would select the class from the packages view and select "Run on Server" from the popup menu.

Running Java Servlets



Right click on your servlet and select Run > Run on Server...



Select which type of application server to use for testing.

Running Java Servlets



```
000001 WEBSHERE REDBOOK      30.00  SG24-5635
000002 Radio_Controlled_Plane  96.86  DELUXE MODEL
000003 Change_Machine          52.55  HANDLES SEVERAL CURRENCIES
000004 Baseball_Tickets        91.14  MINNESOTA TWINS
000005 Twelve_Num_Two_Pencils  50.58  YELLOW IN COLOR
000006 Over_Under_Shotgun      79.66  12 GUAGE
000007 Feel_Good_Vitamins       29.81  A,B,C AND D
000008 Cross_Country_Ski_Set   93.00  ENTRY LEVEL
000009 Rubber_Baby_Buggy_Wheel 98.71  4 INCH SIZE
000010 ITSO REDBOOK SG24-2152  50.00  ACCESSING THE AS/400 WITH JAVA
000011 ITSO REDBOOK SG24-2163  20.87  BUILDING AS/400 APPLICATIONS
      WITH JAVA
000012 Plastic_Garbage_Pail     22.40  36 GALLON CAPACITY
000013 Doll_House_Furniture     24.22  KITCHEN ONLY
000014 Plain_Pockets_Pants       10.28  SEVERAL COLORS AVAILABLE
000015 25_Inch_Color_TVs       63.75  BUILT-IN VCR
000016 Multi-System_Paper      71.57  STANDARD US SIZE
000017 Magical_Mystery_Maze     39.83  FOR BEGINNERS
Done
```

Output from servlet
appears in embedded
web browser

IBM eServer iSeries

Debug

Launch the debugger the same way you run the Java program

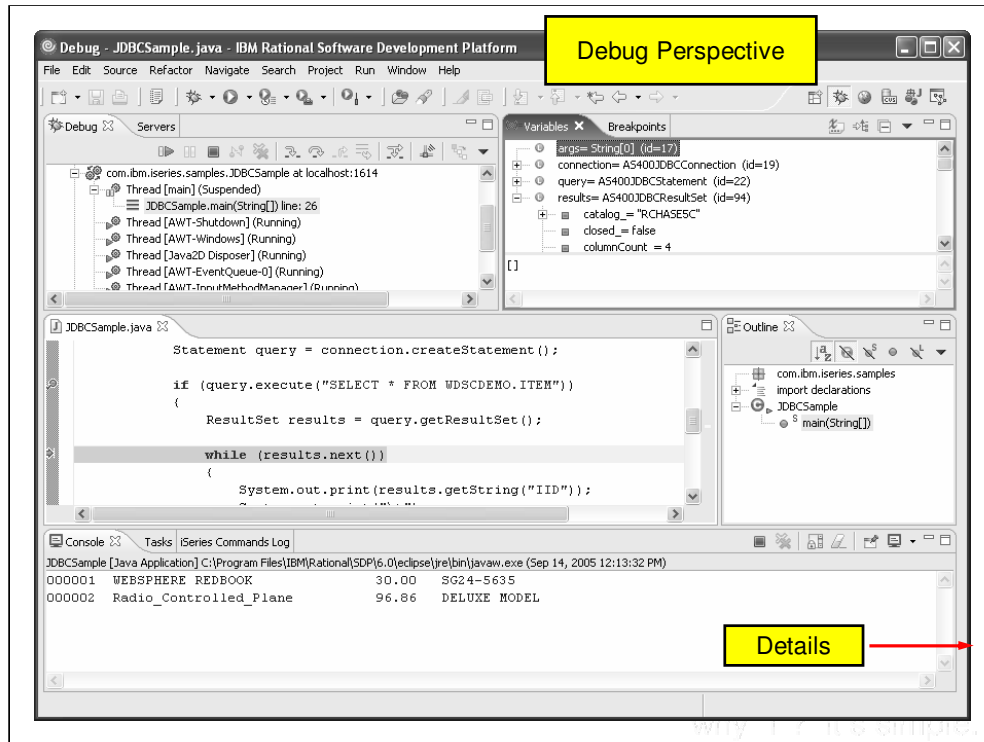
Tip: Set breakpoints in your source code before launching the debugger.

You can set breakpoints in the Java editor by double clicking in the left hand margin of the editor.

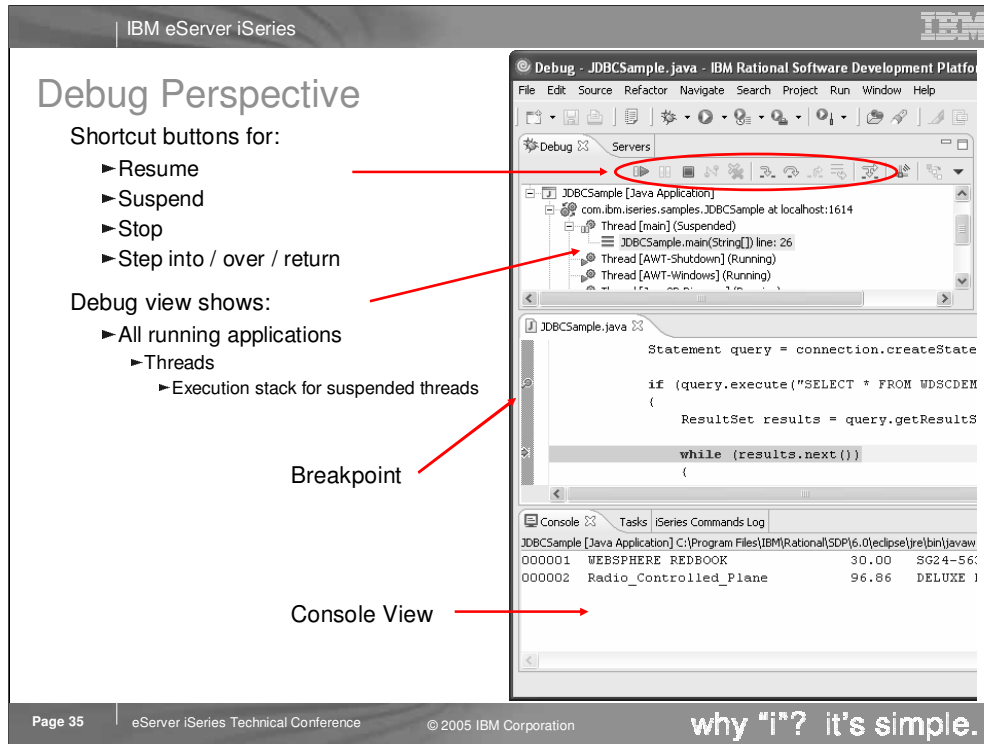
Opens debug perspective ...

Page 33 | eServer iSeries Technical Conference | © 2005 IBM Corporation | why "i"? it's simple.

To debug the Java application select the class in the packages view and select Debug -> Java Application from the Debug toolbar icon drop down menu. It is a good idea to set breakpoints in your source code before launching the application in debug mode. This will cause the debugger to stop at the first breakpoint it hits in the application.



Here you see the Debug perspective.



The top view shown in the screen shot (Debug view) shows all applications that are currently being debugged in a tree view. Below each application all threads are shown, and for each suspended thread the corresponding stack trace. The Debug view contains icons for resuming a suspended thread, suspending a thread, stopping an application and for stepping into / over a line of code or running to the end of a method. These options are also available from the

Debug menu and have associated shortcut keys. Below the Debug view is the source view where source files are opened if a breakpoint is hit.

Debug Perspective

- Variables View
- Breakpoints View
- Expressions View
- Display View
- Threads and Monitors

Page 36 | eServer iSeries Technical Conference | © 2005 IBM Corporation | why "i"? it's simple.

To the right of the Debug view is another pane showing the 4 views listed on this slide. The two main views are the Breakpoints view (used for managing breakpoints) and the variables view (used for viewing and changing the contents of variables in the application).

Breakpoints View

Manage breakpoints in the workspace

Add Java exception breakpoints

Stop when a NullPointerException is thrown

This can be very handle in locating problems!

Inspector view

Detailed view for inspecting variables

Variables View

Shows all variables that are visible in the current stack frame

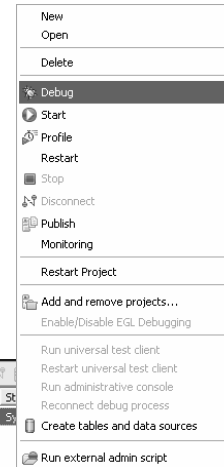
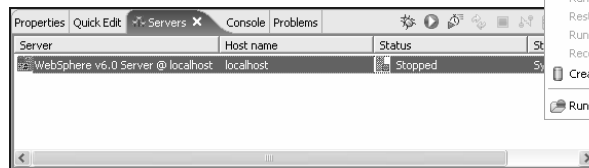
Display View

Displays result of executing an expression in context of current stack frame

Highlight expression in editor and select "Display" from the popup menu

Debugging Web Applications

- First start the application server in debug mode
- Then it's the same as debugging standalone Java applications
 - Set breakpoints inside your Java servlets or JavaServer pages
 - Debugger stops when a break point is hit



Searching – Now Where Did That Go???

- Two main ways to search your Java code
 1. Context sensitive searching
 - Most useful
 - Scope to workspace or class hierarchy
 - Packages / Classes / Interfaces / Variables / Members
 - Interfaces
 2. Global workspace
 - Search the entire workspace
 - Simple file name searching
- Results are shown in the Search view

There are two main ways to search for "things" in the workspace. The first is just a simple raw text search that will search for a string pattern in any type of file.

The second is a "Java-aware" search that will let you search Java files for strings that are context specific. Such as Declarations of all methods that start with getA*

References to all methods that start with getA*

IBM eServer iSeries

Searching

Search remote iSeries source members Simple text searching Java "aware" searching

What to search for

Scope of search

Tip: Use the actions from the searching shortcut actions from the outline view to avoid having to type this information

Page 39 eServer iSeries Technical Conference © 2005 IBM Corporation why "i"? it's simple.

Here you see the "Java-aware" search dialog. You use the radio buttons to define the context of the search.

Agenda

- Getting Started
- The Java Editor
- Visual Editor
- Java Views
- Compile / Run / Debug
-  ▪ iSeries Additions
- Reference

Java Tools for iSeries

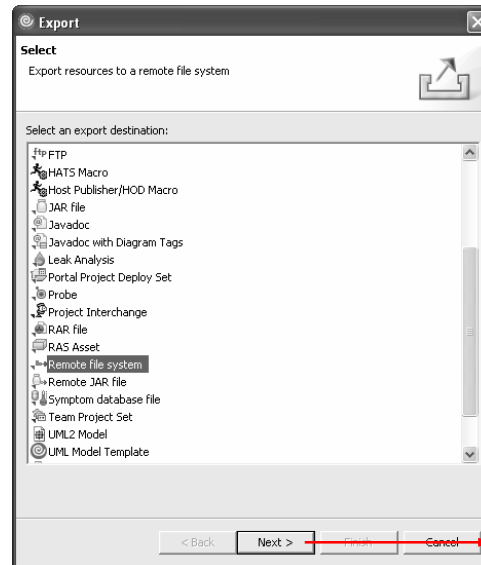
- Import and export from/to remote system
- Remote run and debug launch configurations
- Program Call Bean Wizard
 - Generate Java code to invoke a *PGM or *SRVPGM
- Toolbox for Java built-in
- Supplied Java-beans

Here is an overview of those extensions. The following slides will look at each one in more details.

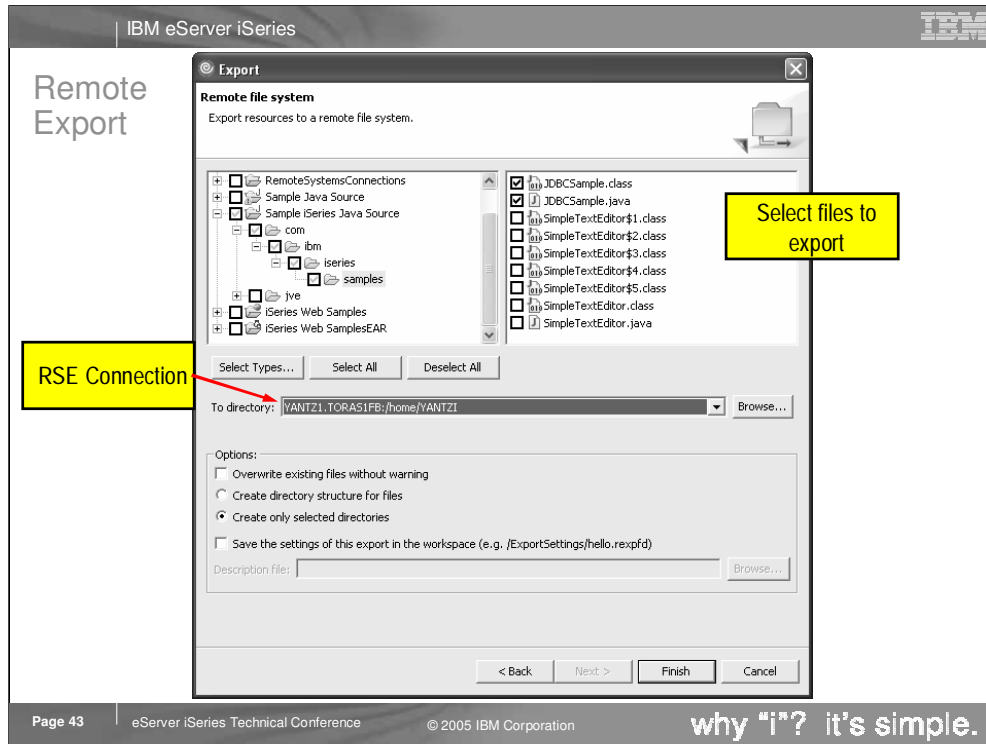
- Export and import to/from IFS, through common remote system frame-work support
- Special Java "compile" actoin/view for remotely compiling Java classes
- Special Java run action/view for remotely running Java applications
- Special Java debug action/view for remotely debugging Java applications
- Program Call wizard to "wrapper" any *PGM/*SRVPGM as a Java Bean
- Toolbox for Java built-in: integrated help and runnable samples
- Previously supplied Java beans: datafile updates/access (DFU beans), object lists beans (PDM – library, object, member, field, record lists), and Swing GUI beans (dspf-like function)

Remote Import and Export

- Import files from a remote system into a local project
- Export files from a local project to a remote system
- Supports iSeries, Linux, Unix, Windows
- Uses the IFS on iSeries
- Can also export to a JAR file on a remote system
 - Useful for Java

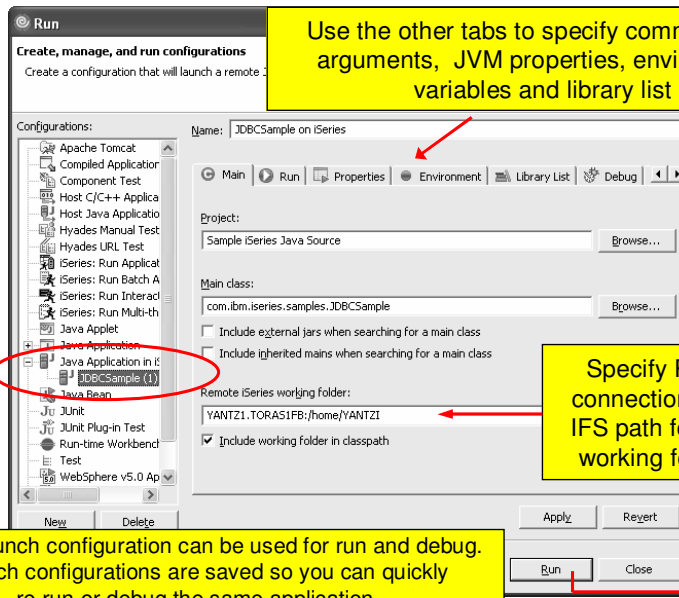


The workbench allows you to import and export your code (Java and non-Java code) from / to a variety of locations. To this list we added the ability to import from and export to a remote system (iSeries or non-iSeries). Like all the other import / export wizards, this ability is not restricted to Java only. On the iSeries this wizard exports to and imports from the integrated filesystem (IFS).



Here you see the next page of the export to remote file system wizard. Here you specify which files to export from their workspace and to which remote system and directory on that remote system. The import wizard has a similar page.

Remote Run and Debug Launch Configuration



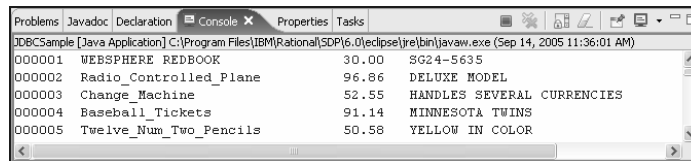
Use the other tabs to specify command line arguments, JVM properties, environment variables and library list

Specify RSE connection and IFS path for the working folder

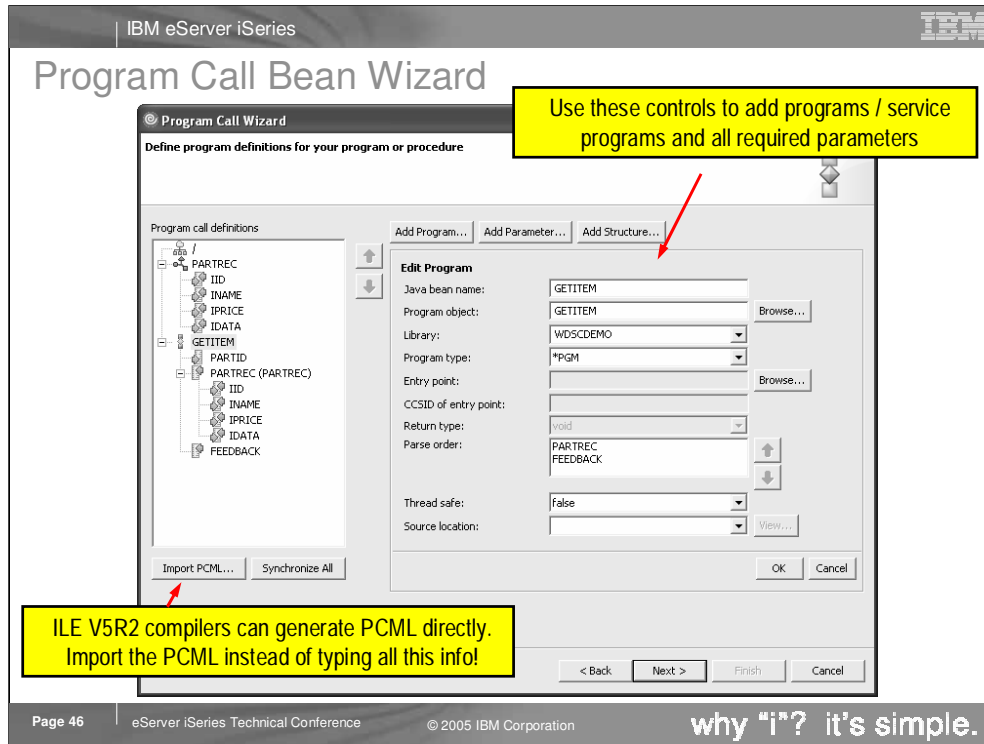
Same launch configuration can be used for run and debug. Launch configurations are saved so you can quickly re-run or debug the same application.

Remote Run and Debug on iSeries

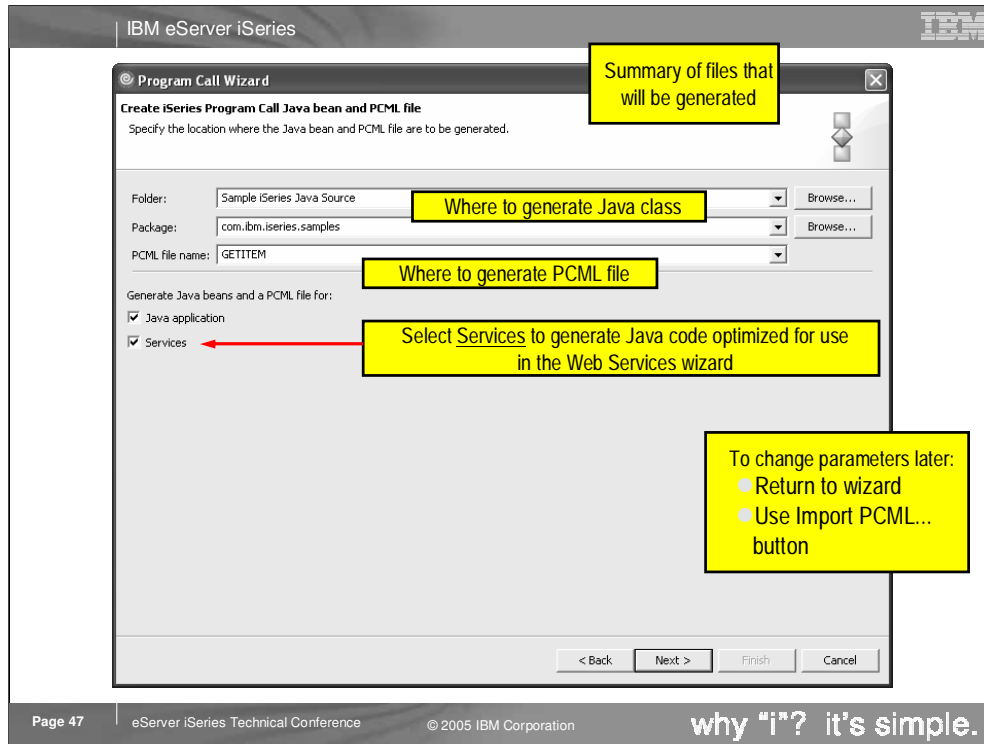
- Output from the Java application running (or being debugged) remotely appears in the Workbench's Console view
- The same launch configuration can be used to debug the Java application remotely
 - Same Java debugger is used for debugging Java applications running locally or remotely!!!



```
Problems Javadoc Declaration Console Properties Tasks
JDBC_Sample [Java Application] C:\Program Files\IBM\Rational\SDP\6.0\ eclipse\jre\bin\javaw.exe (Sep 14, 2005 11:36:01 AM)
000001 WEBSPHERE REDBOOK 30.00 SG24-5635
000002 Radio_Controlled_Plane 96.86 DELUXE MODEL
000003 Change_Machine 52.55 HANDLES SEVERAL CURRENCIES
000004 Baseball_Tickets 91.14 MINNESOTA TWINS
000005 Twelve_Num_Two_Pencils 50.58 YELLOW IN COLOR
```



On the File->New->Other wizard selection page, select the Program Call Bean wizard and press Next. In the Program Call wizard, select the program or ILE procedure to be called from Java. This must be a non-interactive program or procedure.



Here you see that you can confirm where you want the Java code to be generated, including the Program Call Markup Language (PCML) file that is also generated.

For the generated Java bean to be used as a Web Service, select the Web Services checkbox. This generates a Java bean that is easily consumable by the Web Services wizard. Notice how the wizard will automatically update our project's classpath to include the necessary AS/400 Toolbox for Java JAR files so that our

generated code will compile. Finally, press Finish to generate the Java bean!!

Authentication and library list settings

Program Call Wizard

Configure Authentication
Please enter the host information

Authentication | Library List

Generate configuration file

Configuration file name: defaultPCW

Specify signon values

Host name:

User ID:

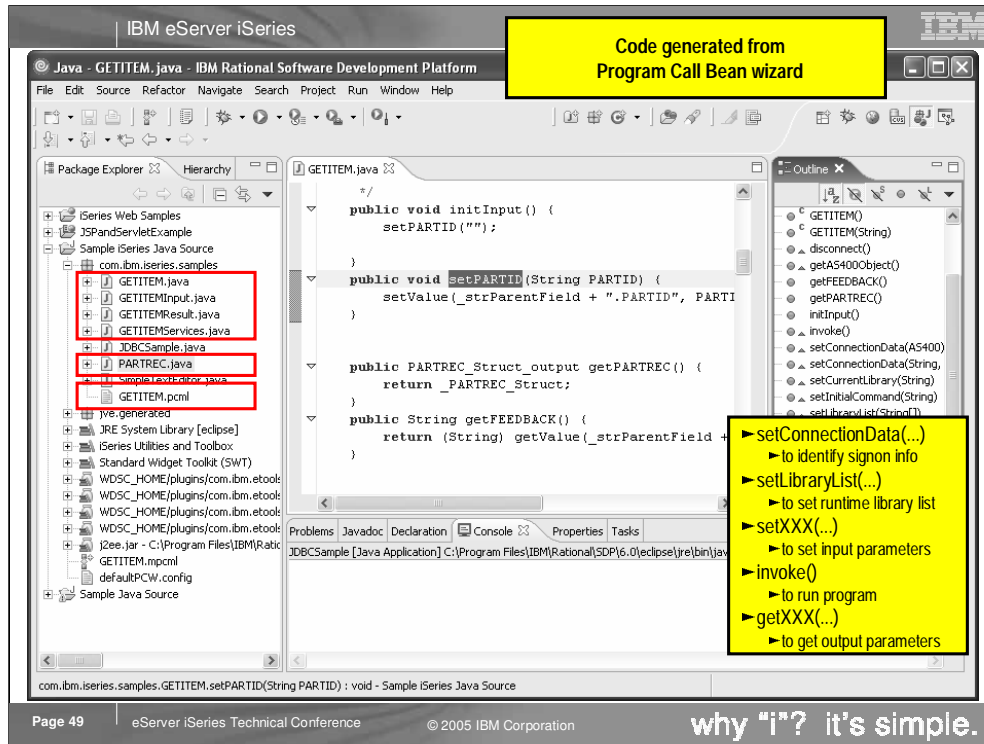
Password:

Enable password encoding

Specify the use of JCA connector and JAAS credentials

JNDI name:

< Back Next > Finish Cancel



Here we see the result of the Program Call wizard. The selected Java package in the upper right now contains all the generated Java classes, one of which is shown in the editor in the middle view. To use the generated Java bean, we write Java code to instantiate it and call the important methods in it, which are listed in the information box in the lower left. An example of Java code to use the bean is shown in the lower middle information box. The pattern for using the generated bean is this:

1. Instantiate the bean using the New operator
2. Call the setLibraryList method with an array of library names to set up the library list
3. Call the setXXX methods to set the input parameter data. There will be one such method for each input parameter.
4. Call the invoke method to call the program or procedure
5. Test the boolean result of invoke for true or false to determine if the invoke was successful
6. Call the getXXX methods to read the output parameter data that the program updated. There will be one such method for each output parameter.

Agenda

- Getting Started
- The Java Editor
- Visual Editor
- Java Views
- Compile / Run / Debug
- iSeries Additions
-  ▪ Summary and References

Summary

- Java Development Tools included with Development Studio Client
 - Part of IBM Rational Web Developer v6.0
- Java perspective provides optimized view for Java development
 - Java Editor + Java Views = Easier Development
 - Integrated Compile/Run/Debug
 - iSeries Additions for using Java with iSeries
- Lots more
 - Team development
 - Integrated, online help system
 - Extendable platform – you can add your own tools
 - Integrated Java, Web, XML, iSeries development

Resources

- Eclipse
 - www.eclipse.org
- WebSphere Development Studio Client
 - www.ibm.com/software/awdtools/wdt400/
- WebSphere Developer Domain
 - www.ibm.com/websphere/developer
- IBM Developer Domain
 - www.ibm.com/developer
- Java
 - www.java.sun.com



Reference: iSeries JavaBeans

GUI Beans for iSeries

- Java Beans supplied for DDS-like field error checking and formatting
 - JFormattedTextField (smart entry field)
 - JFormattedLabel (smart label constant)
 - JFormattedComboBox (smart dropdown)
 - JFormattedTable (smart-multi-column list)

Included with the iSeries extensions to the Java tools are some GUI and non-GUI Java beans that can be used in applications developed by WebSphere Development Studio Client customers. Here you see the GUI beans included with Development Studio Client. These beans extend their Swing counterparts but add customizations that iSeries developers are used to with display files such as:

Restricting a value to numeric, string, ...

Specifying the values length and precision

Specifying formatting using edit codes / words

JFormattedTextField

- Error checking based on data type, length, decimals
- DDS-like validity checking: range or comparison
- Editcode or editword formatting and masking
- Auto-advance

JFormattedLabel

- Editcode or editword formatting

JFormattedComboBox

- Combo of JFormattedTextField and JFormattedLabels

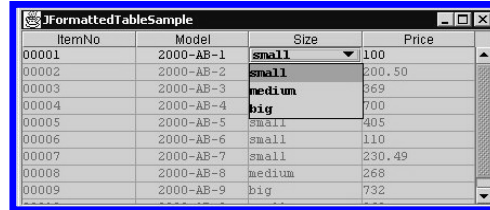
JFormattedTable

- A subfile or multi-file column list box
- Each cell is a JFormattedTextField or JFormatted Label or JFormattedComboBox

GUI Beans Examples

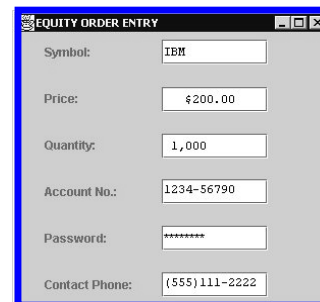
•File->New->Other->Examples->iSeries

► JFormattedTable



ItemNo	Model	Size	Price
00001	2000-AB-1	small	100
00002	2000-AB-2	small	200.50
00003	2000-AB-3	medium	369
00004	2000-AB-4	big	700
00005	2000-AB-5	small	405
00006	2000-AB-6	small	110
00007	2000-AB-7	small	230.49
00008	2000-AB-8	medium	268
00009	2000-AB-9	big	732

► JFormattedText



Symbol:	<input type="text" value="IBM"/>
Price:	<input type="text" value="\$200.00"/>
Quantity:	<input type="text" value="1,000"/>
Account No.:	<input type="text" value="1234-56790"/>
Password:	<input type="password" value="*****"/>
Contact Phone:	<input type="text" value="(555) 111-2222"/>

DFU Beans for iSeries (non-GUI)

- Java Beans supplied for querying data from DB2/400, using direct record access (versus SQL):
 - RecordIOManager
 - ListManager
 - FormManager

Here you see some non-GUI beans that can be used in a Java application to assist the developer in reading records from an iSeries physical file. The ListManager retrieves lists of records for display in a GUI list, whereas the FormManager can be used to retrieve a single record for display in a GUI form.

RecordIOManager

- Query, add, update, or delete records in a database
- Vastly simplifies task of writing database access code using Toolbox for Java classes

ListManager

- Maps list of records from RecordIOManager to a Swing GUI list
- Makes it easy to populate GUI list with DB2/400 data

FormManager

- Maps details of a single row from RecordIOManager to a single GUI dialog page
- Makes it easy to populate GUI form with DB2/400 data

DFU Beans Examples

- ▶ File->New->Other->Examples->iSeries
- ▶ ListManager

FormManager with a multiple record format logical file

Order: DEF456 Next Order

Order Date: 1998-03-24 MAREC
SALESPEC

Customer Order: 88888888

Customer: bbbbbbbbbbbb

Ship Via: ccc

Ship To: 0000000

Price: 55555.33

Goods: 6666

PARTNO	MODEL	PARTPRI	PARTMSR	PARTDIS	PARTSHIP
00562	AR-10	26.10	29.00	Y	1991-05-10
00562	AR-11	32.04	36.00	Y	1992-12-02
00562	AR-12	7.92	9.00	N	1994-05-07
00562	AR-13	11.31	13.00	N	1991-09-25
00074	AR-1	35.64	36.00	N	1990-07-07
00074	AR-11	31.15	35.00	N	1990-01-08
00074	AR-12	29.92	34.00	N	1994-03-19
00074	AR-14	39.56	46.00	N	1993-12-22
01807	AR-10	24.30	27.00	N	1994-06-16
01807	AR-11	20.47	23.00	N	1993-10-13

readAllRecords

▶ FormManager

List Beans for iSeries (non-GUI)

- For querying lists of libraries, objects, members, records or fields
 - AS400ListLibraries
 - AS400ListObjects
 - AS400ListMembers
 - AS400ListRecords
 - AS400ListFields
- Four levels of detail possible per object
- Some OS/400 API wrappers!

Here you see some additional non-GUI beans for retrieving lists of libraries, objects, members, records and fields from an iSeries.

AS400ListLibraries

- List libraries given simple or generic or special name

AS400ListObjects

- List objects given simple or generic library and object name
- Can also subset by type and attribute (one or multi, simple or generic)

AS400ListMembers

- List members given simple or generic library, file, member name
- Can also subset by member type (one or multi, simple or generic)

AS400ListRecords

- List records given simple or generic library, file, record name

AS400ListFields

- List fields given simple or generic library, file, record, field name



Trademarks & Disclaimers

© IBM Corporation 1994-2003. All rights reserved.

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.

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, David Slater, Alison Butteril, Linda Cole
- **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.