

Rational Application Development certification prep, Part 5: Running Java applications

Skill Level: Introductory

[Gregory Scott \(glscott1@us.ibm.com\)](mailto:glscott1@us.ibm.com)
Senior Learning Specialist
IBM

04 Apr 2006

This is the fifth of [seven tutorials](#) created to help you prepare for the IBM Certification Test 255, Developing with IBM Rational® Application Developer for WebSphere® Software V6. Explore features for running both standalone and server-side J2EE enterprise applications.

Section 1. Before you start

About this series

Rational® Application Developer for WebSphere® Software is the IBM Software Development Platform that allows you to quickly design, develop, analyze, test, profile and deploy Web, Web services, Java™, J2EE, and portal applications. This series of seven tutorials helps you prepare to take the IBM certification Test 255, Developing with IBM Rational Application Developer for WebSphere Software V6 to become an IBM Certified Associate Developer. This certification targets entry level developers and is intended for new adopters of IBM Rational Web Developer or IBM Rational Application Developer for WebSphere Software V6.0, specifically professionals and students entering into Web development using IBM products.

About this tutorial

This tutorial explores how to run standalone Java applications using Rational

Application Developer and how to deploy desktop or server-side J2EE enterprise applications (including a Web module) to WebSphere Application Server 6. Learn how to change workspaces, as well.

Objectives

This tutorial demonstrates not only the ways of running Java applications in Application Developer 6, but some of the differences with earlier versions of the tool. Users of the earlier versions (WebSphere Studio Application Developer versions 4 and 5) will find Rational Application Developer 6 almost identical, except for features related to the associated server. After completing this tutorial, you will be able to run both standalone and server-side applications (such as Web applications) with confidence, use command line arguments from within Application Developer, and create or remove multiple launches.

Prerequisites

Before taking the first part of this tutorial, [Running standalone applications](#), you need to understand the concepts covered in the first three tutorials of this series:

- [Rational Application Development certification prep: Part 1, Workbench basics](#)
- [Rational Application Development certification prep: Part 2, Java development](#)
- [Rational Application Development certification prep: Part 3, Web development](#)

The second half of this tutorial continues with related issues pertaining to WebSphere Application Server 6. [Rational Application Development certification prep: Part 4, Working with databases](#) on the data perspective is not necessary for this tutorial, although if you finished it successfully you will save time by continuing with the completed project.

System requirements

To run the examples in this tutorial, install Rational Application Developer for WebSphere Software or Rational Web Developer for WebSphere Software. Download a free trial version of [Rational Application Developer for WebSphere Software](#) if you don't already have a copy of it.

The hardware and software requirements for this software can be located at [IBM](#)

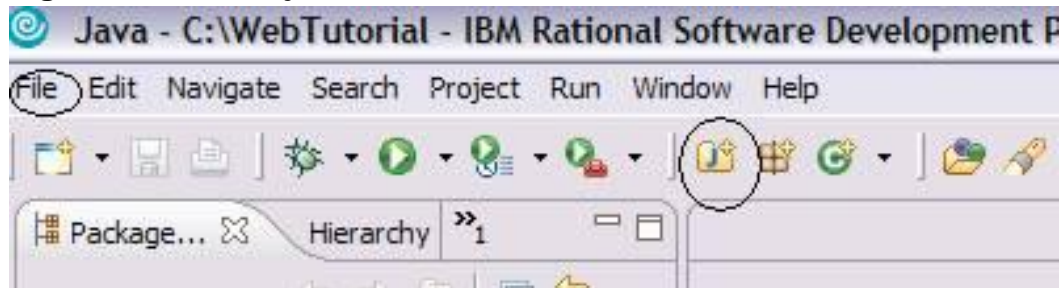
Rational Application Developer System Requirements.

Section 2. Running standalone applications in Application Developer

Step 1: Create or import a standalone application and run it

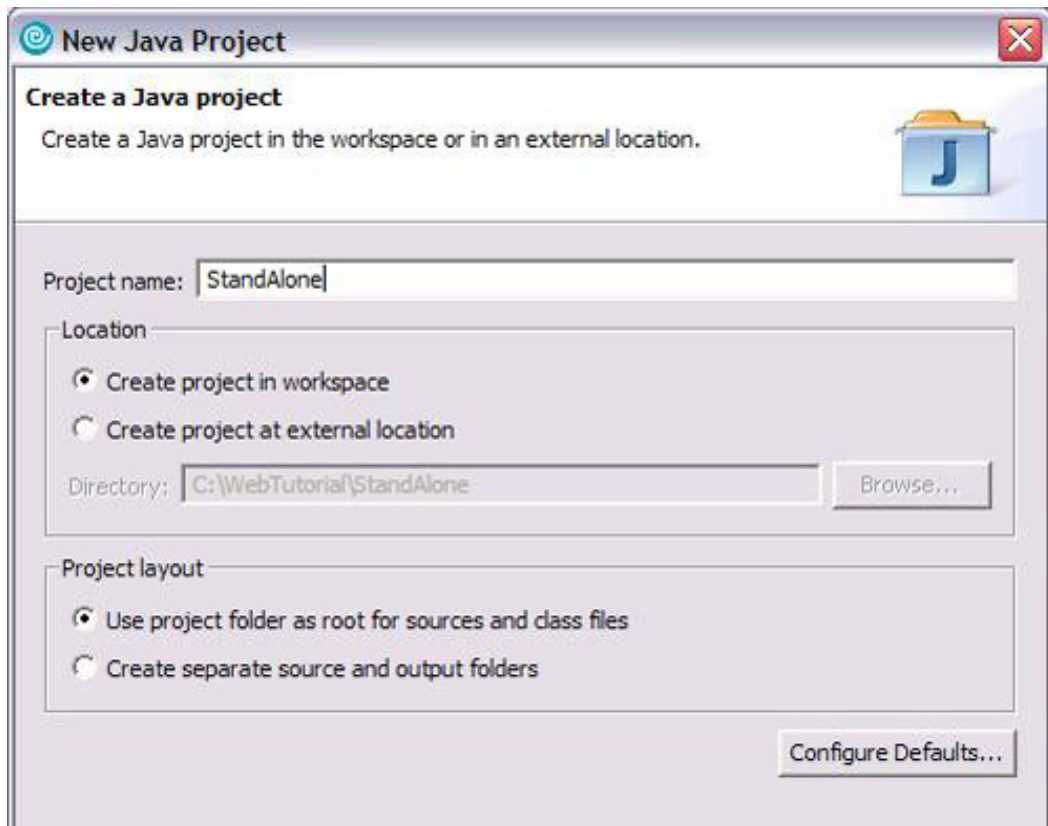
1. Open Application Developer and accept the default workspace or choose your own workspace. This tutorial assumes C:\WebTutorial, which was used by Tutorials 3 and 4, but you can replace that directory with your own.
2. Open the Java perspective by clicking the icon in the upper-right corner, and close any files showing in the editor pane.
3. Create a new Java project called `StandAlone`. One way to do this is to select **File > New > Project > Java**. Or, click the icon in the menu bar.

Figure 1. Java Project icon



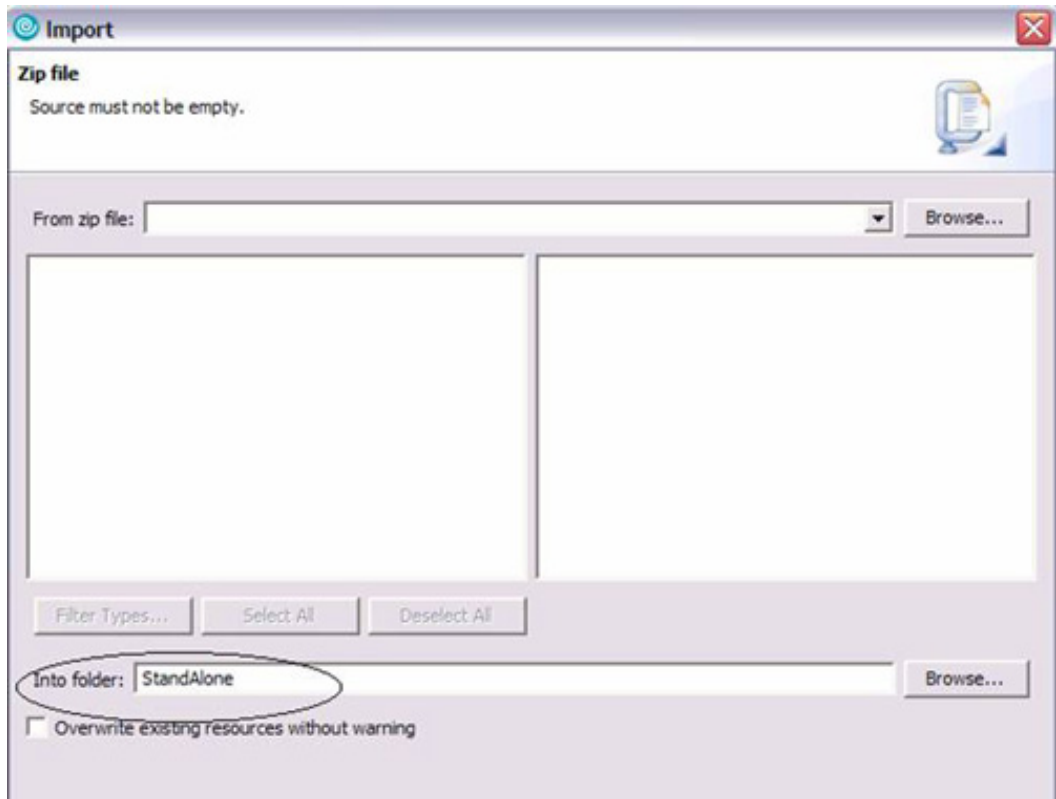
4. In the New Java Project window, type in `StandAlone` as the name, and leave the defaults.

Figure 2. New Java Project screen



5. In this tutorial you keep the source and bytecode in the current project, but as you see, you have options for managing other locations.
6. Either click **Next** to see other options available for other projects (but leave the defaults) or click **Finish**.
7. The StandAlone project now appears in the Package Explorer view. You can code the application yourself, as the following instructions show. Alternatively, using the techniques learned in your previous tutorials, you can import the StandAlone.jar from the [Download](#) section to run the application. Please insure, though, that after selecting **File > Import**, you select *Zip file* rather than one of the specialized .jar files that relate to J2EE development (such as App Client JAR file, which itself means a file with an xml deployment descriptor in it). This is because for purposes of importing, Application Developer makes no distinction between a standard jar and a zip (although for exporting it does). Also make sure that, when you arrive at the following screen and before clicking **Finish**, the word StandAlone is showing in the Into folder:

Figure 3. Zip file screen

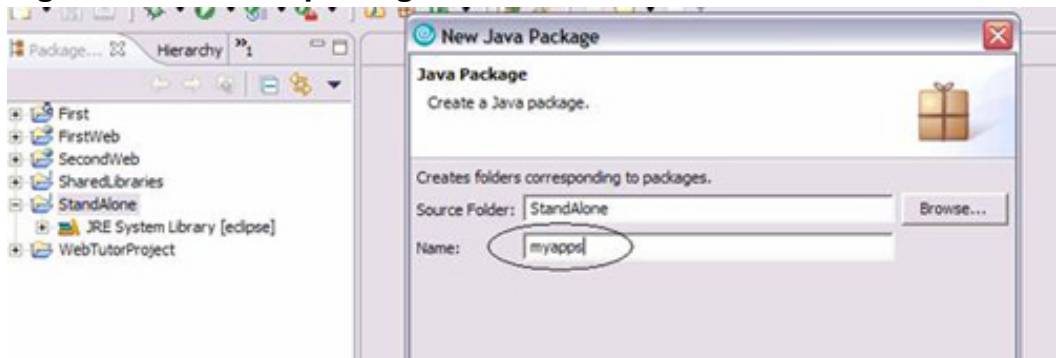


The StandAlone.jar is unzipped and the classes are added to your StandAlone folder. You can inspect them if you want.

To code the application yourself: Select the StandAlone project and either right-click and select **New > Package** or click the Package icon on the menu bar.

8. Name the package myapps.

Figure 4. New Java package screen



9. With the myapps package selected, click on the menu bar the New Class icon, which has the C on it (next to the package icon), or select **File > New > Class**. Fill in the name of the Java class as MyApplic1, and

make sure the main method is checked, as follows:

Figure 5. New Java Class screen

The screenshot shows the 'New Java Class' dialog box. The 'Package' field is set to 'myapps' and the 'Name' field is set to 'MyAppic1'. The 'Modifiers' section has 'public' selected. The 'Superclass' field is set to 'java.lang.Object'. The 'Which method stubs would you like to create?' section has 'public static void main(String[] args)' and 'Inherited abstract methods' checked.

10. Click **Finish**.

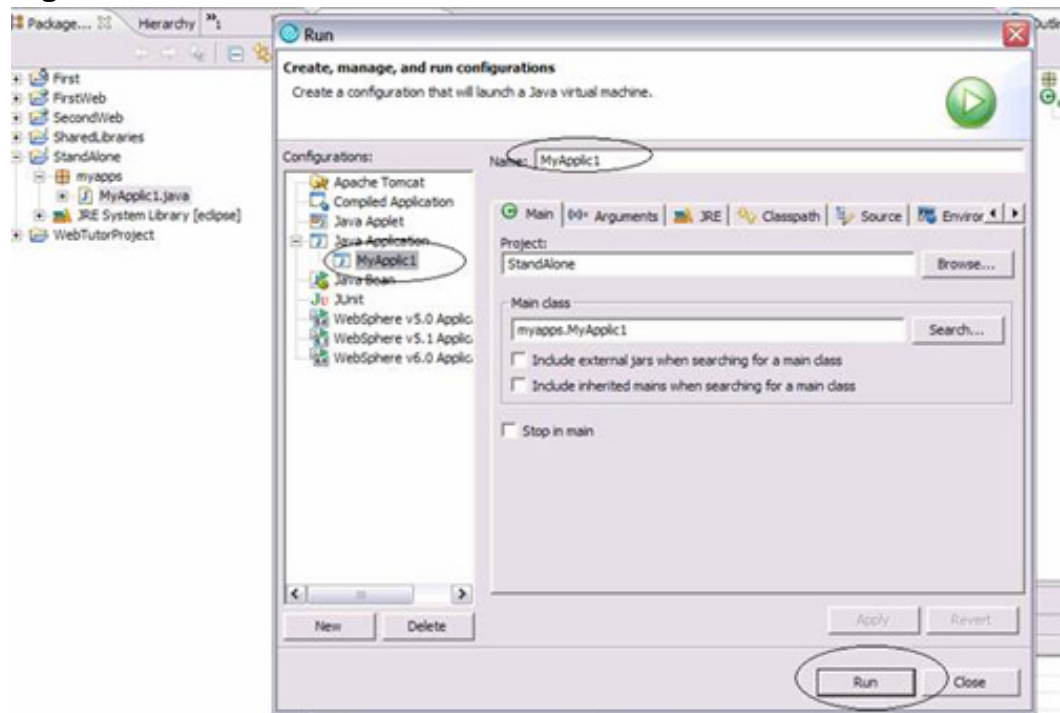
11. Add the following code to the main method:

```
System.out.println("In main of MyAppic1 as of :");
System.out.println(new java.util.Date());
if (args.length > 0){
    System.out.println("Your first argument is: " +
args[0]);
    System.out.println("Your second argument is: " +
args[1]);
}
else {System.out.println("You entered no command line
arguments");}
```

}

- Save the file. There should be no errors. Select **Run > Run** from the menu. The launch window opens.

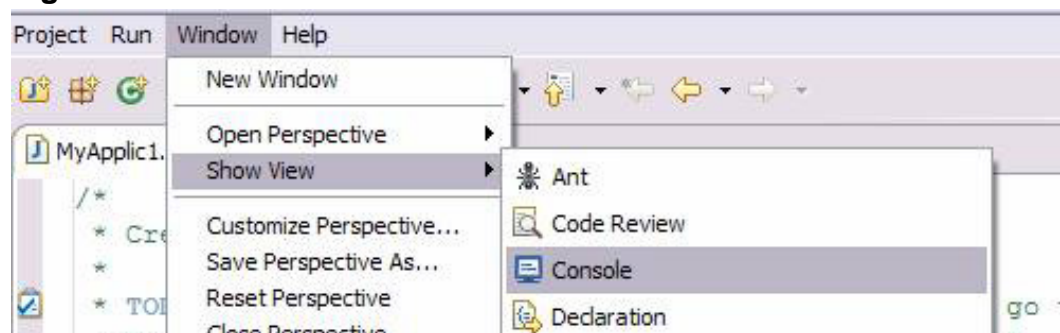
Figure 6. Run screen



If MyAppic1 is not showing under Java Application in the Configurations section, simply click **New** and create it. (Alternatively, you can choose from the menu, **Run > Run As > Java Application** and the new launch gets created for you.)

- Click **Run** in the lower-right window. It looks as if nothing happened (unless you happened to have your Console showing, in which case you will see the print statements).
- If your Console is not showing, click **Window > Show View > Console**.

Figure 7. Show View: Console



- The Console appears in the bottom pane. You should see the print statements, including the date, similar to this:

Figure 8. Console output

```
<terminated> MyApplic1 [Java Application] C:\Program Files\IBM\Rational\SDP\6.0\eclipse\jre\
In main of MyApplic1 as of :
Wed Jan 18 15:13:17 EST 2006
You entered no command line arguments
```

- You can re-run the application a number of different ways: repeat the **Run > Run** step from above; use the **Run > Run Last Launched** sequence; or use the hotkey **Ctrl-F11**.

Now it is time to work with command line arguments in Application Developer.

Step 2: Use command line arguments

- From the menu bar, click **Run > Run again**.
- Click the (x)=Arguments tab. Enter two arguments into the Program Arguments section, similar to this:

Figure 9. Program arguments text area



- Click **Run**. You should see the correct print statements: Anthony is the first argument, and Randazzo is the second.
- As with command line tools, if you want a string of words to be one argument, use double quotation marks. If you are not familiar with command line arguments, try running MyApplic1 again with a phrase similar to the following:

Figure 10. Program arguments text area: 2nd ex



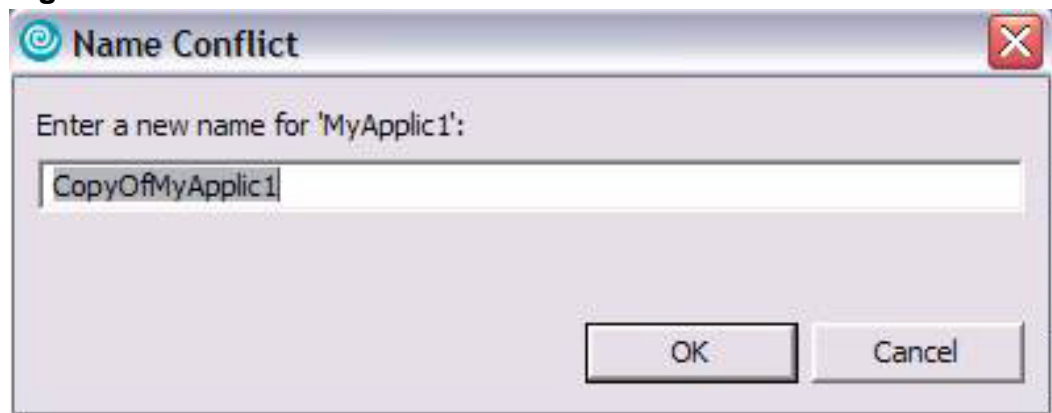
As you see, the entire phrase gets returned as the second command line argument.

Step 3: Create multiple launches

What happens if you are switching back and forth between multiple standalone applications? Create a second application and find out, of course!

1. To keep this simple, right-click **MyApplic1** and select **Copy**. Then paste MyApplic1 into the same package.

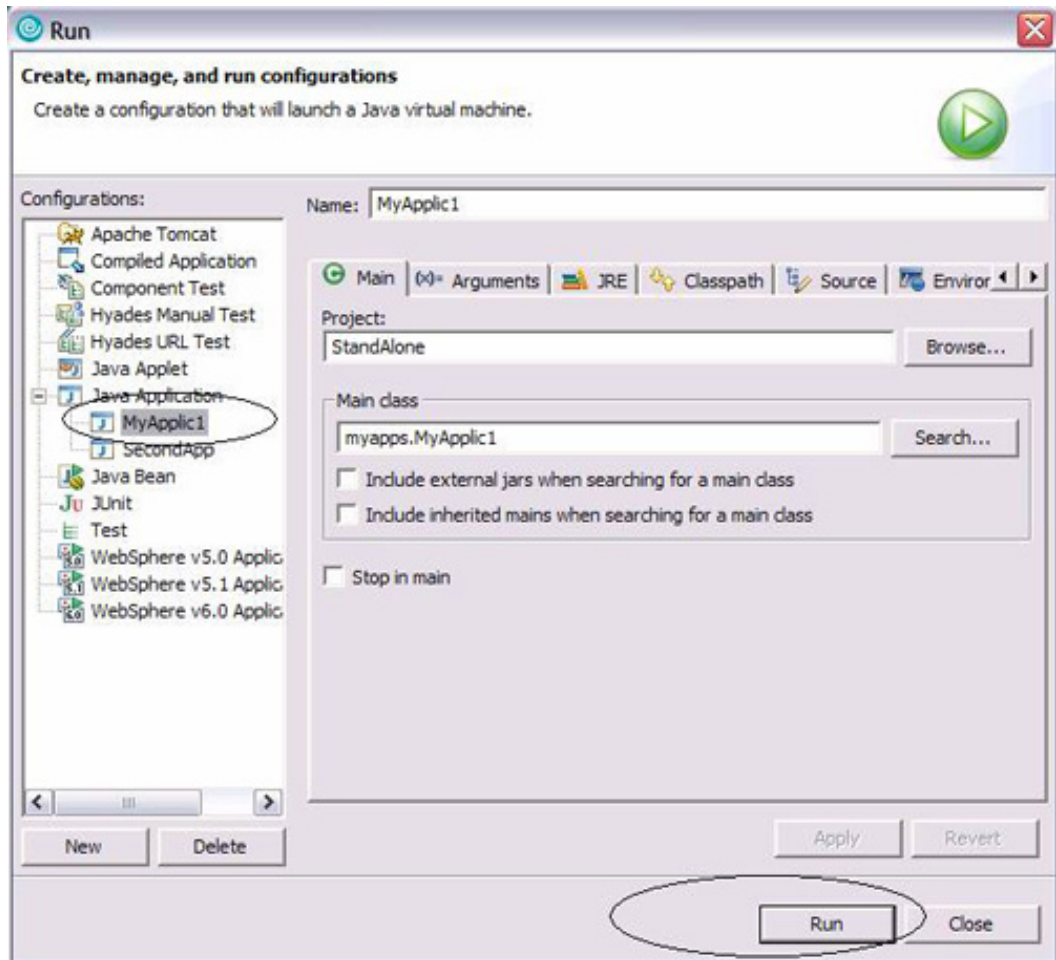
Figure 11. Name Conflict screen



2. Type `SecondApp` (or `MyApplic2` or whatever you want) as the new name.
3. Change the one line of code in `SecondApp`'s main method so the printout is relevant: `System.out.println("In main of SecondApp as of : ");`
4. If you try to run the selected `SecondApp` by pressing **Ctrl-F11** you still get the printout from `MyApplic1`.
5. Therefore, select **Run > Run**, and use the New button again to define the new launch; then click the Run button in the lower-right (or, from the beginning, with the `SecondApp` selected, simply click **Run > Run As > Java Application**).

- As indicated before, the launch gets created and the class's main method runs. Now you will have the correct print statements.
- At this point, if you use Ctrl-F11, it is SecondApp that runs again. If you want to revert to MyApplic1, then click **Run > Run** and select it from the list under Java Applications.

Figure 12. Run screen: Configurations



- Once you no longer need a launch, you can select it and click **Delete** (next to New) shown above.

You have explored the ways to run multiple applications in Application Developer. It is time for server-side considerations. Close any open files in the editor.

Section 3. Running Web applications and changing workspaces using Application Developer and WebSphere Application Server 6

In the first part of the tutorial, Rational Application Developer certification prep, Part 3, you saw that you could deploy and run a Web application in one of three ways from the Web perspective:

1. Right-click the servlet class file and click **Run on Server** (if the servlet .java file does not give the Run on Server option, use the servlet's class file instead, as found under the WEB-INF/classes directory).
2. Right-click the desired Web application (for example, FirstWeb in the tutorial [Rational Application Developer certification prep, Part 3](#)) and click **Run on Server**. This option searches the Welcome file list in the web.xml for the first file, for example, index.html, and if found, runs that index.html. If not, it looks for the second file in the Welcome file list and tries to run it, and so forth, through the list. You must create these files to allow this automatic search to work.
3. In the Servers pane, right-click the server itself and select **Add and Remove Projects**. Add the desired project to the server. This starts the server automatically, at which point you can run a particular servlet or JSP or a Welcome file according to (1) or (2) above.

The following section covers additional development issues resulting from using Application Developer 6 and WebSphere Application Server 6, and is especially relevant for previous users of WebSphere Studio Application Developer 4 and 5.

Differences between Application Developer and WebSphere Studio Application Developer

A little background: in versions 4 and 5 of Application Developer, the so-called WebSphere Test Environments were essentially WebSphere Application Server 4 and 5 respectively, but could be thought of as being internal to Application Developer. That is, if and when one switched workspaces in Application Developer 4 or 5 from, say, C:\One to C:\Two, and started up the server in the second workspace, a new instance of WebSphere Application Server 4 or 5 would be created that was independent of the server in the first workspace. Hence, there was no problem in having an enterprise application called, for instance, Library in both

workspaces.

Now, with Application Developer 6, the situation is different. Application Developer 6 gets installed with a basic edition of WebSphere Application Server 6 alongside of it. No longer is there a Server perspective in Application Developer, as there was in versions 4 and 5. Moreover, there is *only* one set of binaries for WebSphere Application Server 6, and any and all workspaces in Application Developer normally point to a *default profile* in WebSphere Application Server 6.

What is a *default profile*? You can think of a profile as being an instance of WebSphere Application Server 6. Having profiles allows WebSphere Application Server administrators to create multiple instances of WebSphere Application Server with only one set of binaries. (For more info on profiles, see the [Resources](#) section.) Hence, the default profile is the one automatically run if users do not create their own profiles, and indeed developers using Application Developer 6 need not care about profiles as long as they do not change workspaces.

However, if the workspace is changed in Application Developer 6, a potential problem occurs, as the following example describes: Assume an enterprise application named Library had been created in C:\One and deployed (which means that Library will also exist in the default profile of WebSphere Application Server 6). If a duplicate, similar, or new enterprise application called Library is created in C:\Two and you attempt to deploy it to WebSphere Application Server 6, an error will occur, indicating that the application already exists.

There are two solutions to this problem:

1. Create a second WebSphere Application Server profile to be used by the second Application Developer workspace . This solution is beyond the scope of this tutorial, but more information can be found in the [Resources](#) section.
2. Uninstall the enterprise application from the server in the first workspace before trying to create or deploy it from the second workspace. You get practice using this solution in the following section using the example from the tutorial "Rational Application Developer certification prep, Part 3", which contained the enterprise application called First (which contains the Web application called FirstWeb).

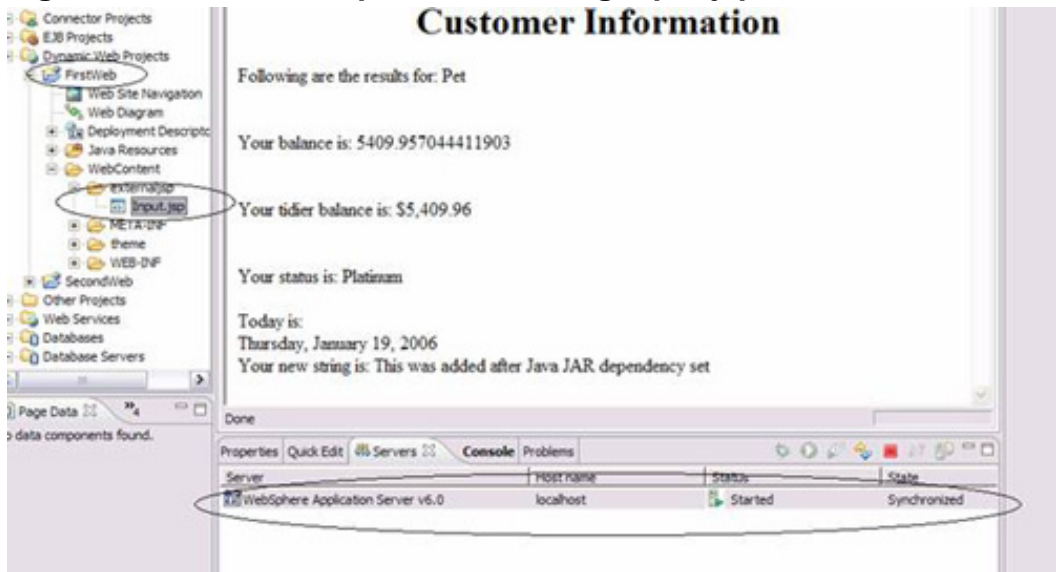
Step 1: Deploy identically named enterprise applications from different workspaces

This section assumes FirstWeb was run successfully in Part 3 or Part 4 of this tutorial series. If you did not successfully complete Part 4, use the solution from

tutorial Part 3, as the database will not have been set up and connection errors will result from running Part 4 solution code. Before continuing, run whichever solution you are employing to make sure that it has been deployed successfully to the server. You can download the easiest solution from Part 3 if needed.

1. At this stage, your Web application is running (presumably in the C:\WebTutorial workspace). You should see something similar to this:

Figure 13. Possible output from running Input.jsp of FirstWeb



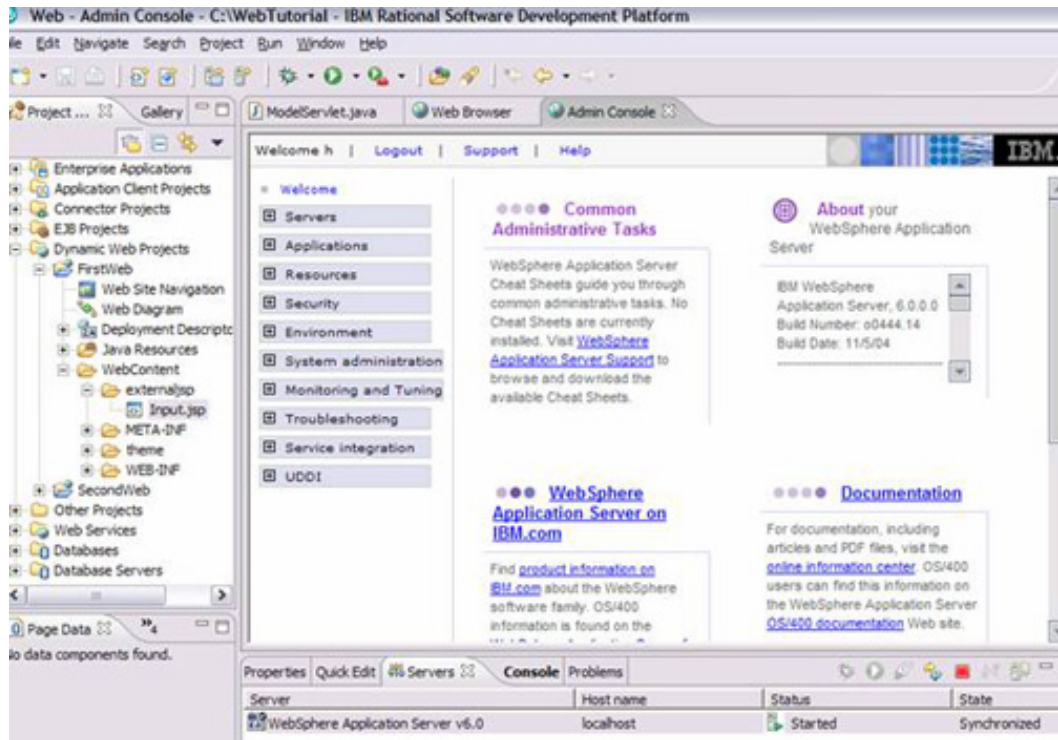
2. In the admin console for WebSphere Application Server 6, right-click the server under Servers and select **Run administrative console**.

Figure 14. Run administrative console choice

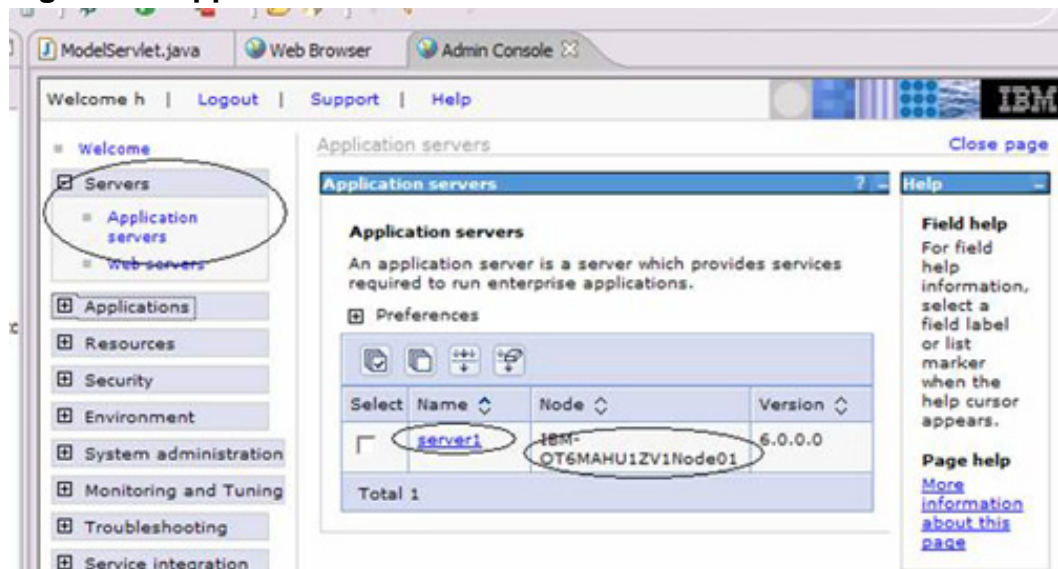


3. The login screen opens. Because security is not turned on, you can enter any name or single character (it is for logging purposes only), then click **LogIn**. The Admin console screen opens. This is the same admin console that administrators work with to control WebSphere Application Server 6.

Figure 15. Admin Console screen



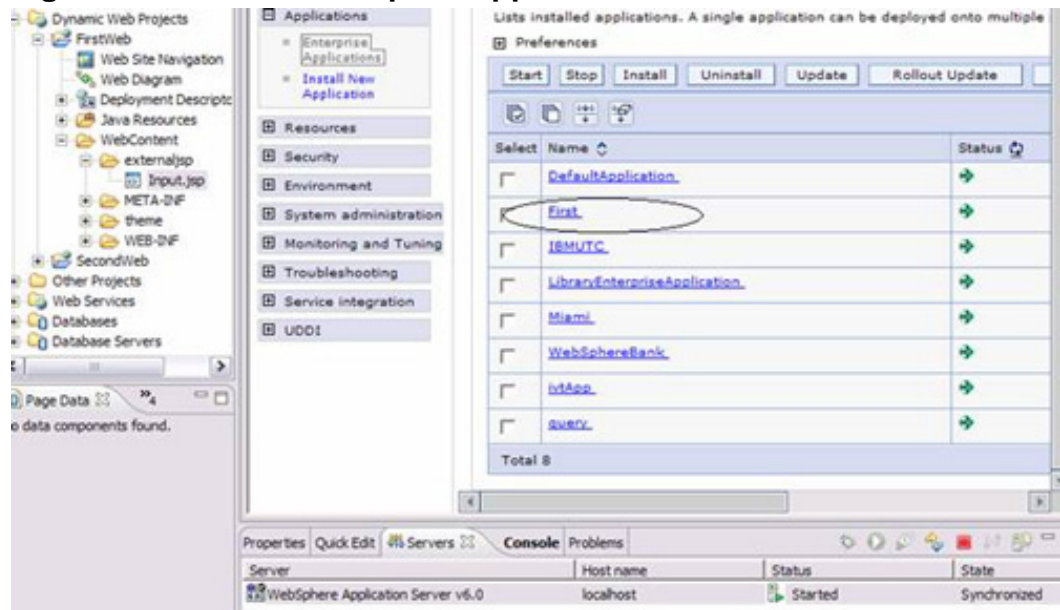
- Expand **Servers > Application Servers** under the left Welcome column. You see the default name (server1) and your machine's name (the node). **Figure 16. Application servers**



- Likewise, expand **Applications > Enterprise Applications**. You now see all of the enterprise applications that have been deployed to the application server, no matter what the Application Developer workspace. In my case, I see the following, including the enterprise application First

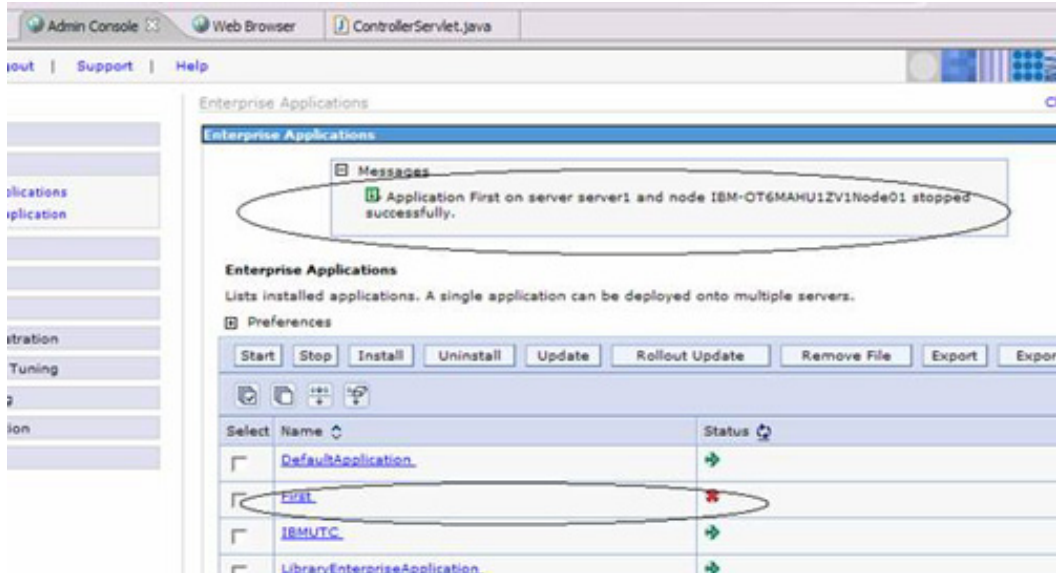
that was created in the third tutorial and is now running, as shown by the green arrow under "Status".

Figure 17. Installed enterprise applications



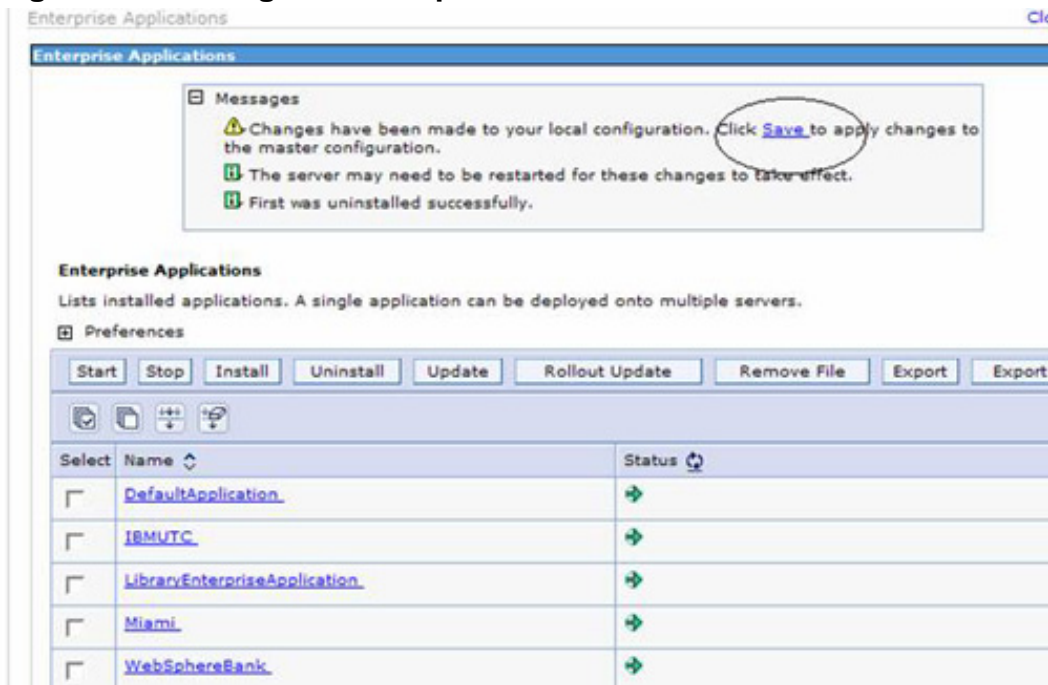
6. You can uninstall the First enterprise application either by using the admin console or by removing the project from the Servers pane in the Web perspective (with Add and Remove projects). Because you already know how to use the latter feature, stop and uninstall the First application from the admin console, as explained below.
7. Check the box next to the First application and then click **Stop**. A message indicating success appears, and the red x appears under **Status**.

Figure 18. Enterprise Applications: Messages



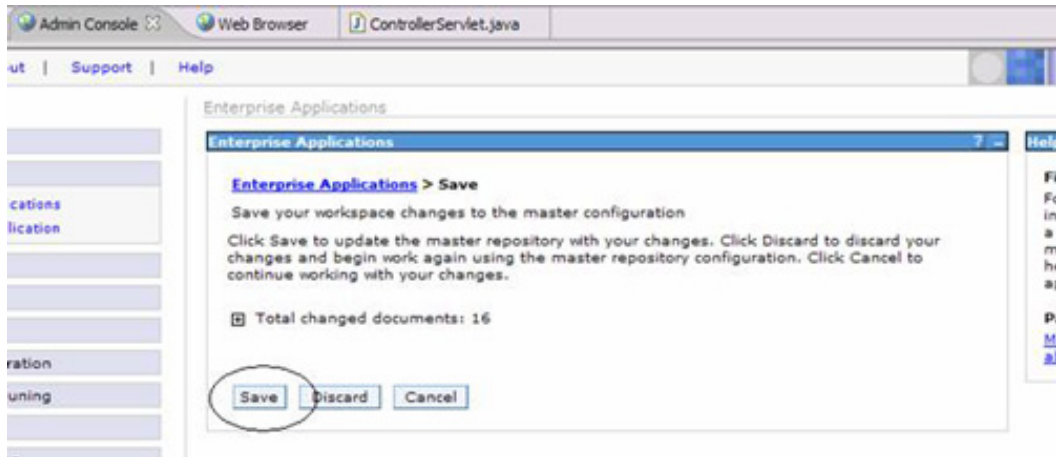
- Then click **First** again, and choose **Uninstall**. Click **OK** to confirm the uninstall. You should no longer see the First application listed. Equally important, you should see the Save message. You must click the Save link to finish the task, as shown:

Figure 19. Messages: Save option



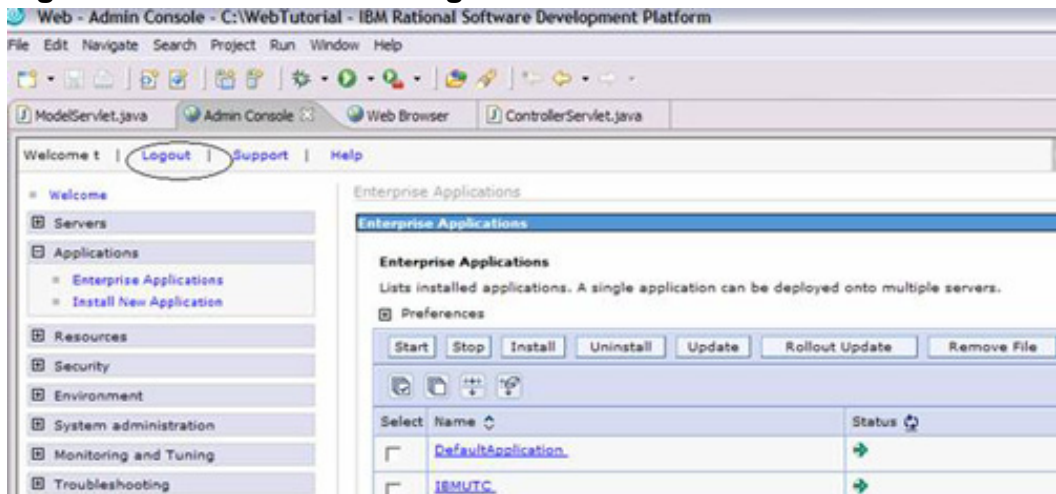
- A Save confirmation window appears.

Figure 20. Save confirmation



10. Click **Save**. First is now completely uninstalled.
11. There are many other options you can explore in the admin console, but then you are entering much further into the administrator's realm. For more information on administering WebSphere Application Server, see the [Resources](#) section.
12. For the moment, log out of the admin console by clicking the Logout option at the top.

Figure 21. Admin console: logout



13. Now that you have removed First from WebSphere Application Server, change your workspace in Application Developer, using the steps in the following section, and import the First application into that new workspace. To cover some other issues, export First from the C:\WebTutorial workspace and then re-deploy First to insure that it runs from the new workspace.

Step 2: Change the workspace and re-deploy

1. Close all files within Application Developer (**File > Close All**), but leave it running for the moment.
2. Check the running applications in your operating system. If you're running Windows, click **Ctrl-Alt-Delete** and choose **Task Manager > Processes**. You should see a javaw.exe for Application Developer and another java.exe for WebSphere Application Server and, depending on other software you might be running, there could be other java.exe processes. You will re-examine the Task Manager in a few minutes.

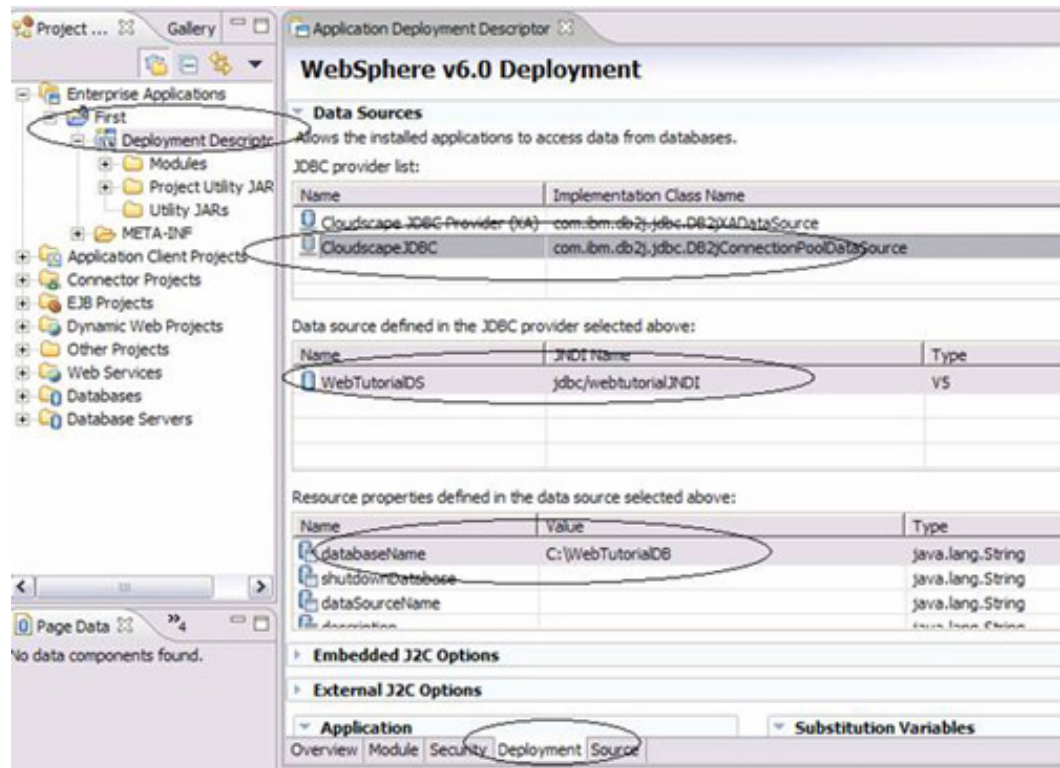
Figure 22. Task Manager results: one possibility

ntaskldr.exe	glscott1	00	29,276 K
isamsmt.exe	SYSTEM	00	2,220 K
javaw.exe	glscott1	00	255,908 K
eclipse.exe	glscott1	00	3,300 K
iPodService.exe	SYSTEM	00	3,912 K
vsmon.exe	SYSTEM	00	26,896 K
wdfmgr.exe	LOCAL SERVICE	00	1,744 K
TpKmpSvc.exe	SYSTEM	00	1,304 K
PCS_AGNT.EXE	SYSTEM	00	4,512 K
RASvc.exe	SYSTEM	00	18,936 K
TPHDEXLG.exe	SYSTEM	00	1,824 K
BTTray.exe	glscott1	00	7,452 K
BTStackServer.exe	glscott1	00	10,320 K
ccSetMgr.exe	SYSTEM	00	4,056 K
java.exe	glscott1	01	266,352 K
Ad-Watch.exe	glscott1	04	9,576 K
ctfmon.exe	glscott1	00	4,440 K

3. Back in Application Developer, drill down in Project Explorer to **First > Deployment Descriptor** and open it. Click the Deployment tab. This tab is new in Application Developer 6. It *enhances* your .ear (more information later). For the moment, simply note that if you did not run Part 4 on databases of this tutorial series, you do not see any custom values. If you finished Tutorial 4, select the Cloudscape JDBC provider to remind yourself how you configured it.
4. In WebSphere Studio Application Developer 4 and 5, the database information was not contained in the .ear file. You had to export separately any server information (be it in a project or not) and manually configure the testing WebSphere Application Server to accept the relevant information. In this tutorial that information is merely the database JNDI name and location (assuming you finished Tutorial 4). If you completed Tutorial 4 you should see the following; if you skipped

the tutorial, examine a little more carefully the whole screenshot here, simply for your edification:

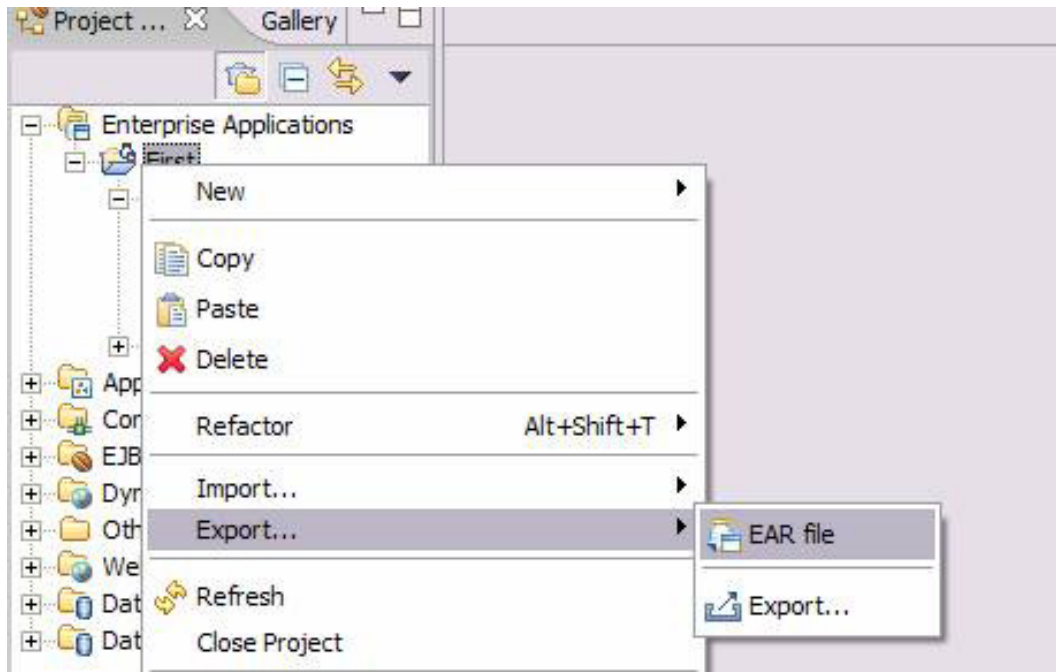
Figure 23. Enhanced .ear: Application Deployment Descriptor



Now that you have confirmed the information that can exist in an enhanced .ear file, close the descriptor and export First as an .ear:

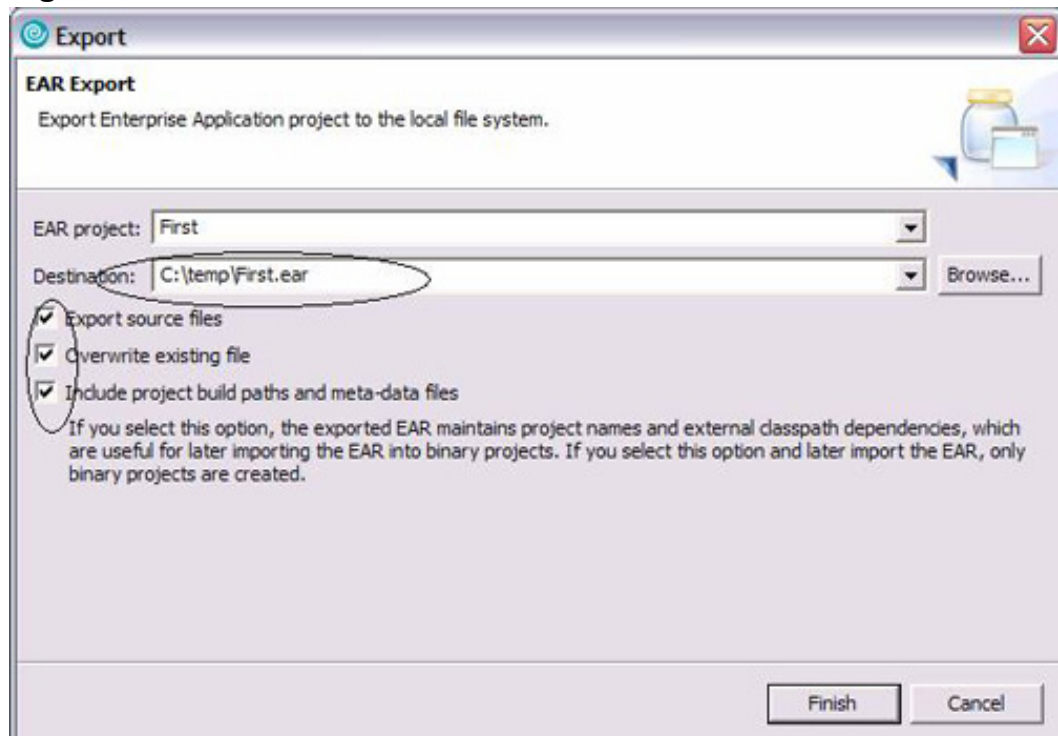
1. Right-click **First** and select **Export > Ear file.**

Figure 24. Export EAR



2. Choose your own temporary Destination. Check **Export source files** and **Include project build** (Overwrite is optional in this case).

Figure 25. Destination



3. Click **Finish**. Confirm the ear file is in the Destination directory, then close

Application Developer.

4. Now go back to the Task Manager in your operating system, and use the j key to quickly find any process beginning with j. You should find only java.exe -- the javaw.exe representing Application Developer was closed, but the server is still running!
5. To be sure your server does not continue to run, stop it in one of three ways:
 1. Before closing Application Developer, stop it from your Servers pane (in a relevant perspective like Web or J2EE)
 2. Kill the process from Task Manager
 3. Use your WebSphere Application Server command line options to stop it:
Go to the following directory, or its equivalent, on your system:
C:\Program Files\IBM\Rational\SDP\6.0\runtimes\base_v6\bin. Run the stopserver batch file with the name of your server (recall from the admin console that it was server1). For example:

```
C:\. . .base_v6\bin>stopServer server1
```

You see:

Figure 26. Windows command screen



```
C:\WINDOWS\System32\cmd.exe
C:\Program Files\IBM\Rational\SDP\6.0\runtimes\base_v6\bin>stopserver server1
ADMU0116I: Tool information is being logged in file C:\Program
Files\IBM\Rational\SDP\6.0\runtimes\base_v6\profiles\default\logs\ser
ver1\stopServer.log
ADMU0128I: Starting tool with the default profile
ADMU3100I: Reading configuration for server: server1
ADMU3201I: Server stop request issued. Waiting for stop status.
ADMU4000I: Server server1 stop completed.

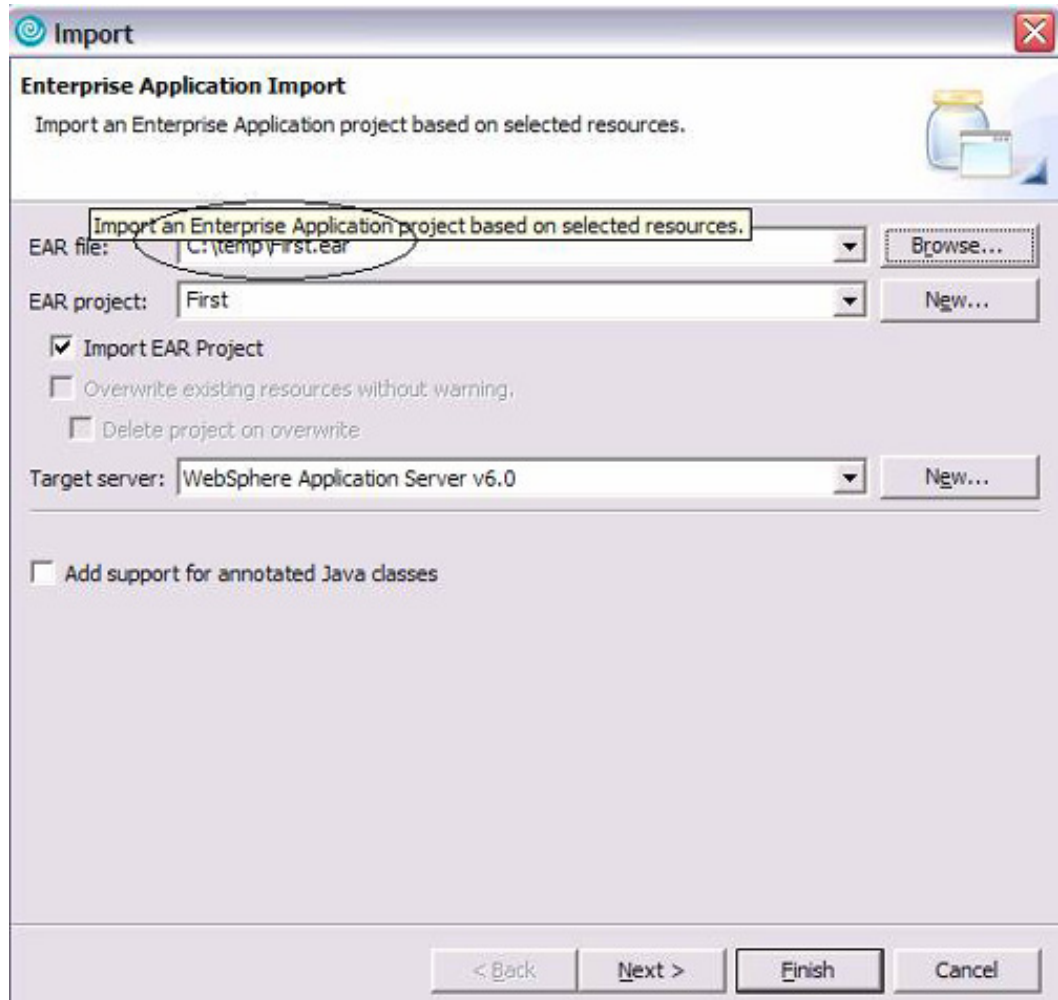
C:\Program Files\IBM\Rational\SDP\6.0\runtimes\base_v6\bin>_
```

Your server is stopped and the java.exe is gone from the Task Manager process.

6. Open Application Developer into a different, new workspace. For the following steps, it is assumed C:\Tutorial5 is the new workspace name you enter when opening the Application Developer.
7. Notice that the tool subsequently opens with the default Welcome window.
8. Close the Welcome pane and switch to the Web perspective if it is not

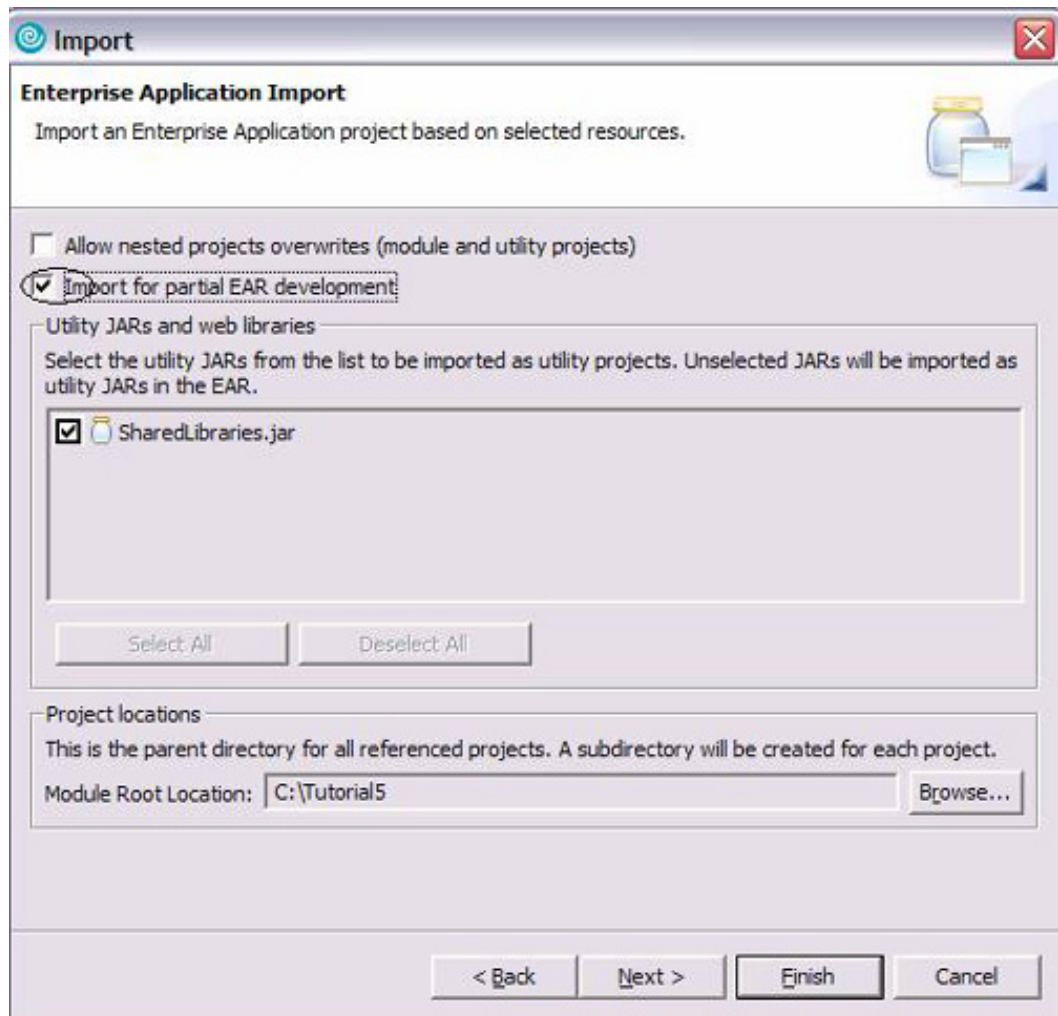
your default.

- In your Project Explorer window, right-click **Enterprise Applications** and click **Import > EAR file**.
- Browse to the temporary directory with the First.ear file that you exported a few steps back and select **First.ear**. The other fields are filled in for you.

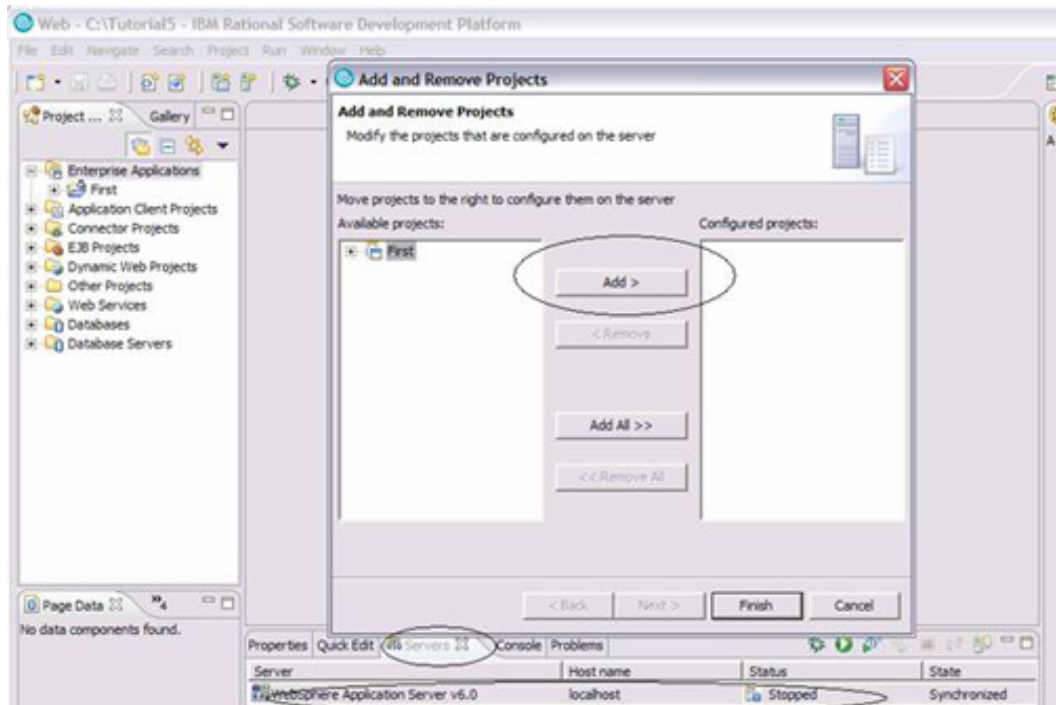


- Click **Next**. In the next screen, check **Import for partial EAR development**. Notice the .jar file gets checked automatically.

Figure 28. Partial development

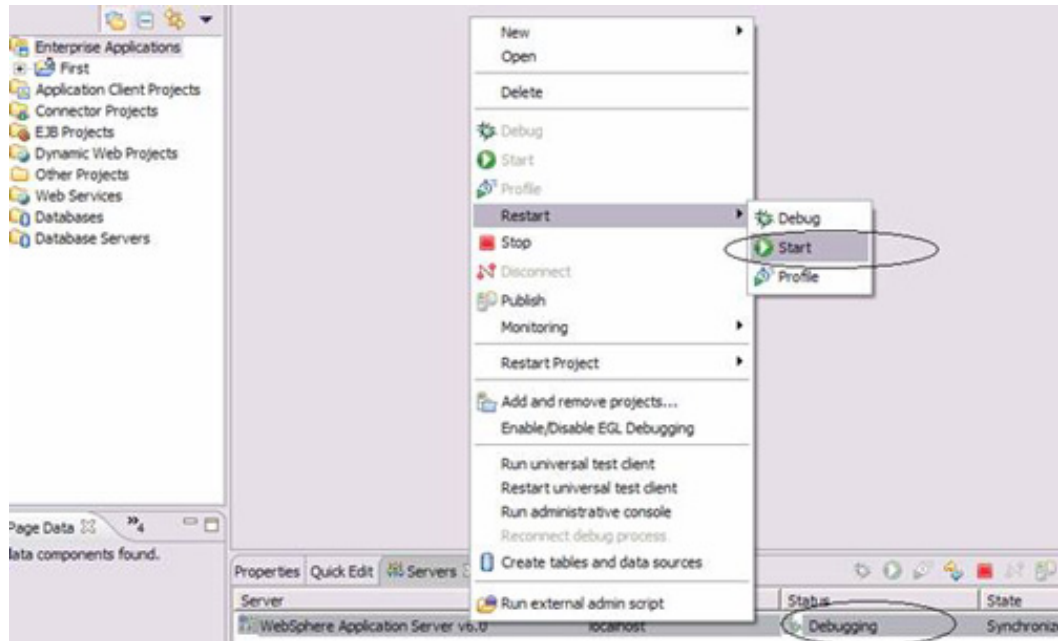


- To see any module (namely, FirstWeb) being brought in under First, click **Next**. Otherwise, click **Finish**. Stay in the Web perspective if you are asked whether you want to go to the J2EE perspective.
- In the Servers pane, right-click **WebSphere Application Server 6** and select **Add and Remove Projects**. The following window opens:
Figure 29. Add and Remove Projects

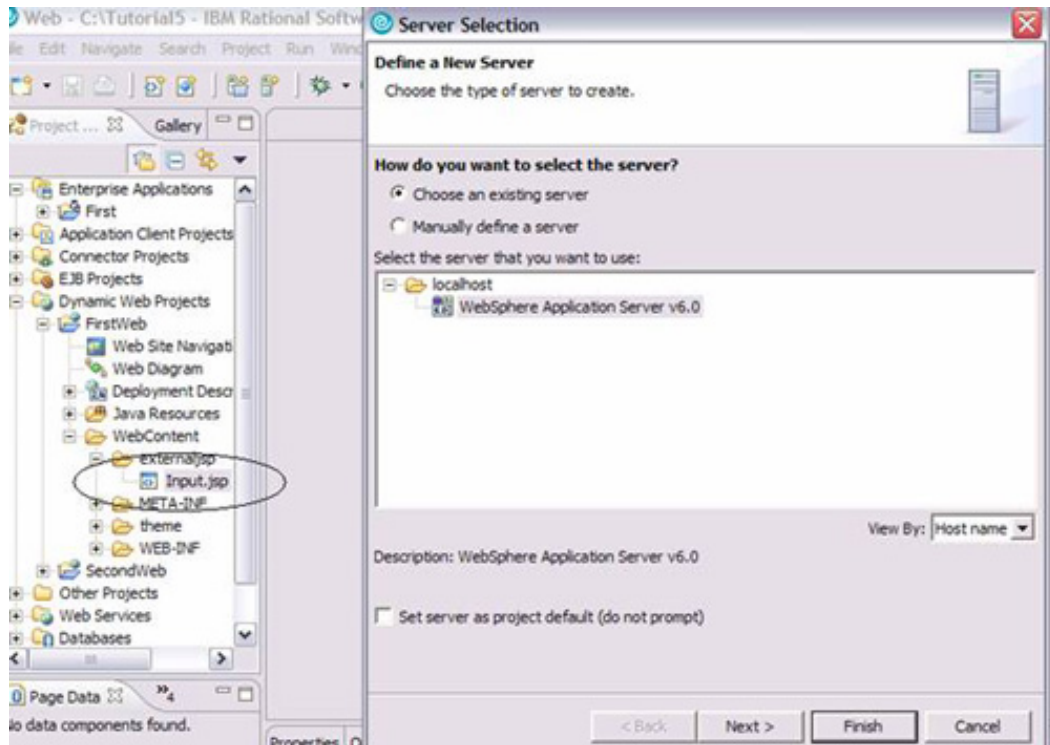


14. With First selected as the Available project, click **Add**, then **Finish**.
15. Notice the server is now starting (and the project gets added to it under the covers). Your project is now deployed to WebSphere Application Server 6, more precisely, again, to the default profile of Application Server. If the Server status bar displays *Debugging* rather than *Started*, restart the server in *Start* mode.

Figure 30. Restart option



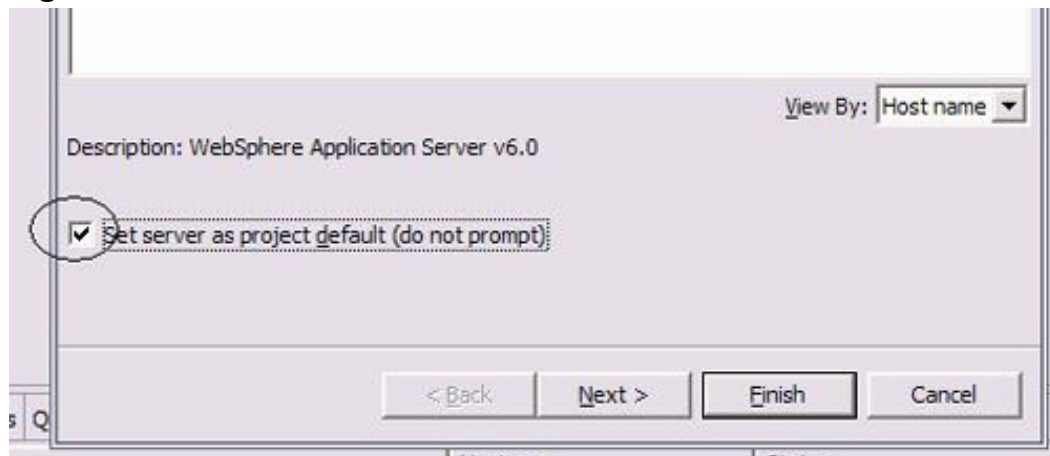
16. Depending on your hardware and processor speed, you might have to wait a minute or three for the restart to finally be successful. Examine the Console output to see the status. You need to see the line "Server1 open for e-business" before continuing.
17. To test click **Dynamic Web Projects > FirstWeb > WebContent > externaljsp > Input.jsp > Run on Server**.
18. Perhaps, surprisingly, you might be asked to select a server.
Figure 31. Server Selection screen



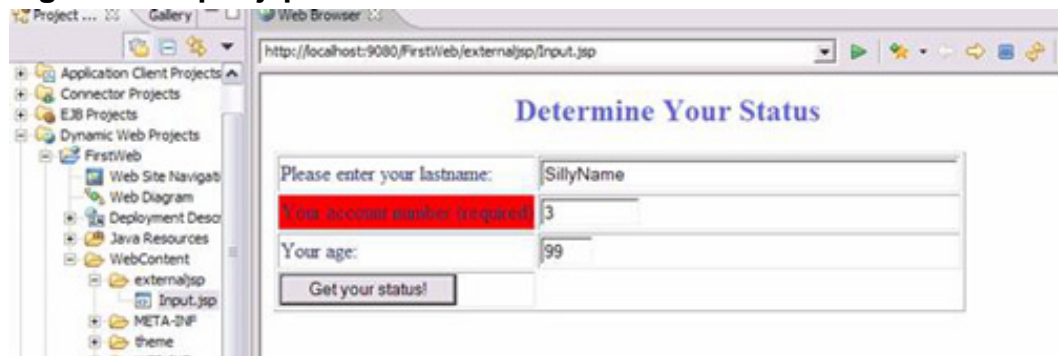
This is surprising, and perhaps an oddity of Application Developer for this version only, because after the server started, an Add and Remove Projects check indeed showed that First was added to WebSphere Application Server 6.

19. In any event, check **Set server as project default**.

Figure 32. Set Server screen



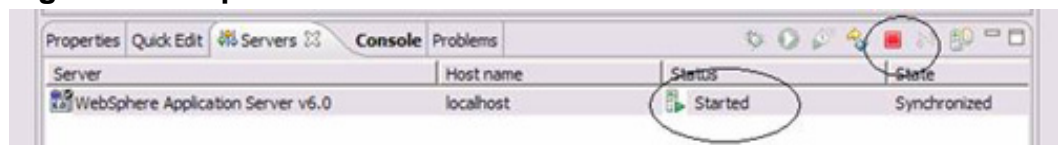
20. To confirm that First is an added project, click **Next** or simply click **Finish**.
21. Your Input.jsp form should now appear, as follows, and you can enter data.

Figure 33. Input.jsp screen

If you used the solution from Tutorial 3 (Web application) you get simulated results back, and it does not matter what lastname or account number you used (age was never implemented on the server side in the previous tutorials).

If you used the solution from Tutorial 4 (Data perspective), remember that, for the sake of simplicity, the lastname field was never checked against the database in the business logic (although it does get echoed back in the output.jsp). Implementing the code to insure that lastname matches the correct account number is left to the reader as a "bonus" exercise. At this stage, the account number is the only crucial element for data input, and must match a relevant piece of data in your database; otherwise, no status is returned from the InitialOutput.jsp (which resides under FirstWeb > WEB-INF > jsp).

22. You probably want to stop the server before leaving Application Developer so you don't leave java.exe running in your processes. In the Web perspective Servers pane, right-click the server and select **Stop** or, if the server is selected, simply click the Stop icon, the red square.

Figure 34. Stop icon

If you have drastic problems

Sometimes, your application code for a server-side application does not get reflected in the tested code, and simply restarting the server does not solve the problem. Here are some general solutions, from basic to advanced:

1. In the test server's popup menu, select **Add and Remove Projects**.

Remove your project, click **Finish**, and then add it back again. The server needs to be running to do this because adding or removing a project invokes wsadmin tasks on the server.

2. Alternatively, in Application Developer's menu, select **Project > Clean....** Be sure to build again after the cleaning, either automatically or through **Project > Build....** Do the add/remove projects step detailed above.
3. Incremental building and smart publishing, which is new with Application Developer 6, can get out of sync. If none of the previous fixes work, you must rebuild the default profile, which is the most drastic step of all, short of re-installing Application Developer. Rebuilding the profile can fix a multitude of server-related problems, but obviously will require some work (eight to twenty minutes, depending naturally on your experience). See the [Resources](#) section for links to information about rebuilding profiles.

Congratulations. You have now completed Tutorial 5.

Downloads

Description	Name	Size	Download method
Startup code for this tutorial	StandAlone.jar	1KB	HTTP

[Information about download methods](#)

Resources

Learn

- [The Source for Java Developers](#) is the place to get information on the latest Java JDK (J2SE 5, previously known as J2SE 1.5), including basic Java *standalone* applications that require a main method or server-side applications, making use of servlets, JSP pages or EJB components, with examples and tutorials.
- To get more information on applets, go to [J2SE 5 Java DemonstrationApplets](#).
- At [J2EE Connector Architecture](#) you will find a number of links on resource adapters and connecting to legacy or enterprise information systems.
- The [Technical library view](#) provides access to a multitude of free tutorials for mastering development, including how to run various applications, in Application Developer.
- The article, [IBM Rational Developer: Powerful support for rapid Java and J2EE development](#), presents an introduction into the various ways to develop and run Java applications using Application Developer.
- To learn more about EJB components, take the tutorial [Getting Started with Enterprise JavaBeans Technology](#).
- To learn more about version 6 and its profiles, read the article [System management for WebSphere Application Server V6](#).
- The forum posting at [WebSphere 6.0 -- Creating multiple app server instances on one physical server](#) presents information on using profiles, including instructions on deleting and re-creating the default profile.
- The IBM Redbook, [WebSphere Application Server V6 System Management & Configuration Handbook](#), provides information to help you administer WebSphere Application Server 6.
- Visit the [developerWorks Application Developer zone](#) to expand your Application Developer skills and knowledge.
- Get certified as an "IBM Certified Associate Developer". Check out the [objectives](#), [sample assessment tests](#), and [training resources](#) for test 255, "Developing with IBM Rational Application Developer for WebSphere Software V6".
- Stay current with [developerWorks technical events and webcasts](#).

Get products and technologies

- Download a free trial version of [IBM Rational Application Developer](#).

- Build your next development project with [IBM trial software](#), available for download directly from developerWorks.

Discuss

- [Participate in the discussion forum for this content.](#)
- Participate in [developerWorks blogs](#) and get involved in the developerWorks community.

About the author

Gregory Scott

Greg is an instructor in private and public national settings, internal and external to IBM, primarily related to Java client-side and server-side (J2EE) programming. Specializations include servlets, JSPs, Enterprise JavaBeans, portlets, XML and Web Services.