

Using the VisualAge for Java WebSphere Test Environment

Craig Pelkie
Bits & Bytes Programming, Inc.
craig@web400.com

Using the VisualAge for Java WebSphere Test Environment

Edition VAJWTE_20010930

Published by

Bits & Bytes Programming, Inc.
Valley Center, CA 92082
craig@web400.com

Copyright © 2001, Craig Pelkie. All Rights Reserved

No part of this presentation or the accompanying computer source code may be reproduced or distributed in any form or by any means, or stored in a database or data retrieval system, without the prior written permission of Craig Pelkie, who is the author of the presentation and the computer source code.

All computer source code distributed with this presentation, either on diskettes, CD-ROM, or available for downloading from sources such as the Internet is Copyright © 2001 Craig Pelkie, All Rights Reserved. The source code is for use in computer programs that you develop for internal use within your company, or for use within programs that you develop for the use of your clients, when such programs are compiled and distributed in an executable computer program such that no part of the source code is readily visible without the aid of a computer program disassembler. No part of the computer source code distributed with this presentation shall be reproduced in source code format, either printed or in electronic format, by you or by others who you allow to have access to the source code. You shall not cause the source code to be stored on any information retrieval system, such as computer Bulletin Board Systems or the Internet. You shall not develop any written articles, books, seminar materials, or other presentations that include the source code provided on the diskettes accompanying this manual or within the manual itself.

For any questions regarding your rights and responsibilities using the computer source code distributed with this presentation, contact Craig Pelkie, Bits & Bytes Programming, Inc., who is the owner of the source code.

LIMITATION OF LIABILITY AND DISCLAIMER OF WARRANTY

No representation is made that any of the programs, computer source code, commands, or configurations described and depicted in this manual and on the computer source code accompanying this presentation are error-free and suitable for any application that you may develop. Craig Pelkie makes no warranty of any kind, expressed or implied, including the warranties of merchantability or fitness for a particular purpose, with regard to the information, examples, and computer source code presented in this presentation and on the accompanying diskettes. Everything provided in this manual and on the accompanying diskettes is provided "as is". Craig Pelkie shall not be liable in any event for incidental or consequential damages or any other claims, pursuant to your use of any of the techniques presented in this presentation, or your use of the computer source code in programs that you develop, even if Craig Pelkie has been advised of the possibility of such damages.

You are responsible for testing any and all programs, configurations, commands, and procedures presented in this manual prior to using the programs, configurations, commands, and procedures with important user data. You must ensure that adequate and sufficient backup of important user data is available, in the event that recovery of the important user data is required.

Table of Contents

Introducing the WebSphere Test Environment	2
Installing the WTE.....	3
Using the WebSphere Test Environment.....	6
Run the Java Servlet and JSP Samples	10
Testing a Servlet	13
Testing a JSP.....	15
About the URLs	16
WTE Summary	17

The WebSphere Test Environment

- **Problem:**

- *Servlets developed in VA Java*
- *JavaServer Pages (JSP) developed on PC*
- *How to test before deploying to WebSphere*

- **VisualAge for Java WebSphere Test Environment (WTE)**

- *Can run servlets / JSPs on development PC*
- *Environment is built on WAS code base*
- *Can quickly debug, fix, retest*

- **Environment**

- *VisualAge for Java 3.5 or higher*
- *Windows 98 / ME / NT 4.0 / 2000*

Copyright © 2001 Craig Pelkie. All rights reserved worldwide

Introducing the WebSphere Test Environment

Many iSeries 400 shops are starting to move their development efforts to web enablement using WebSphere Application Server (WAS). Because WAS is a Java-based environment, it makes sense to consider using IBM's VisualAge for Java (VA Java) as one of your tools for developing WAS applications. VA Java is an especially good choice for WAS development when you use its built-in WebSphere Test Environment (WTE).

The WTE provides the support needed to run and test servlets and JavaServer Pages from inside VA Java. If you don't use the WTE, you need to either set up a Java application server such as Tomcat on your PC or export your code from your PC to WAS. If you use either of those approaches, you are then forced to test your code outside of the VA Java environment, meaning that you no longer have access to the integrated debugger.

Perhaps the best news about the WTE is that it is provided with VisualAge for Java Professional Edition, version 3.5 and up. That version is included with the new IBM WebSphere Development Tools for iSeries Version 5.1, provided with OS/400 V5R1. If you don't have the WebSphere Development Tools, you can acquire VisualAge for Java by itself; the Professional Edition is significantly less expensive than the Enterprise Edition. Although it is possible to install and run VA Java on a low-end Pentium class PC with as little as 128MB of memory, you will be much more pleased with the performance of the WTE if you use a PC with a fast CPU and at least 256MB of memory. You can install VA Java and the WTE on a Windows 98, Windows Me, Windows NT 4.0 or Windows 2000 PC.

Installing the WTE

- **VisualAge for Java install option**
 - **Complete install – includes WTE**
 - **Custom install – select JSP Development Environment option**
- **Add to VA Java Workspace**
 - **F2 (File, Quick Start), Add Feature – IBM WebSphere Test Environment**
 - **Verify that IBM WebSphere Test Env is listed in All Projects list**

Copyright © 2001 Craig Pelkie. All rights reserved worldwide

Installing the WTE

The WebSphere Test Environment is an optionally installable part of VisualAge for Java. When you install VA Java, you can choose to install the complete product or perform a custom install. If you choose the complete option, the WTE is installed. If you choose a custom install, you need to select the JSP Development Environment as a feature to install, as shown in the InstallShield Wizard panel (Figure 1). If you already have VisualAge for Java installed on your PC but did not install the WTE, you can run the install program again and add the WTE to your existing configuration.

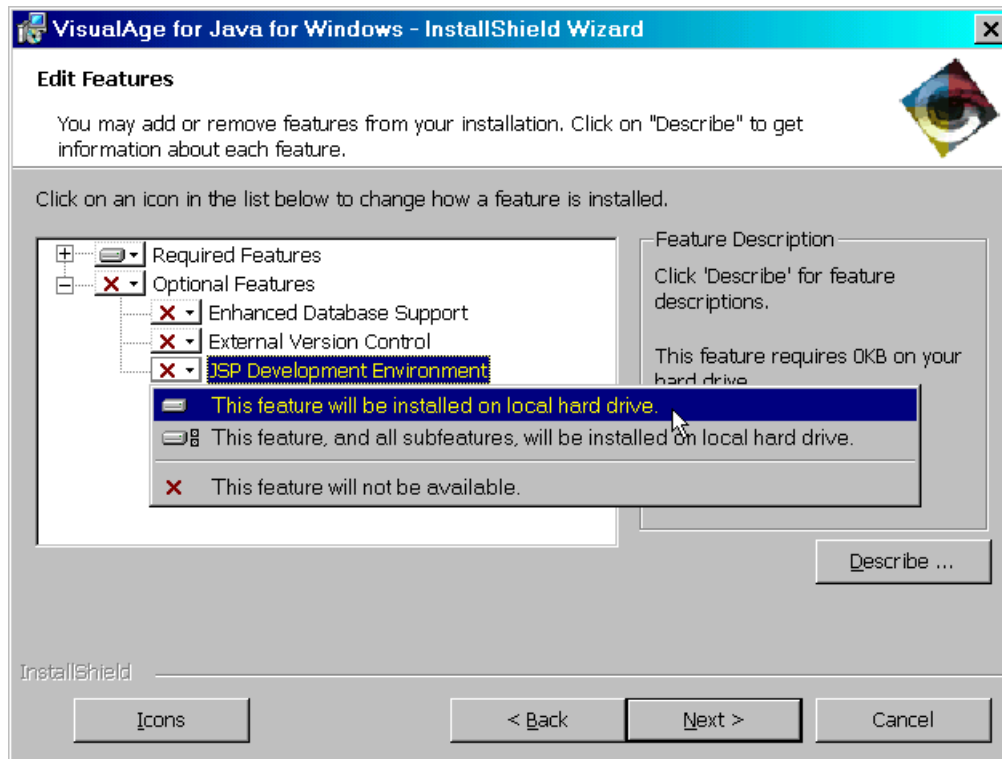


Figure 1: If you perform a custom install of VisualAge for Java, select the JSP Development Environment to install the WTE. WTE001

After installing the WTE, you need to add the feature to the VA Java workspace. In the VA Java Workbench (the default view of the Integrated Development Environment that is shown when you start VA Java), press the F2 key or select the File, Quick Start menu item.

On the Quick Start dialog (Figure 2), select the Feature, Add Feature option and click OK. In the Selection Required dialog shown in Figure 3, select the IBM WebSphere Test Environment and click OK. You can select more than one feature to add to the VA Java workspace if you want. When you add a feature, VA Java retrieves the required code from its repository and adds new projects, packages and classes to the workspace. You should only add features that you intend to use, since the start up and shut down times for VA Java are significantly longer when you have more features in the workspace.

After adding the feature to the workspace, you can verify that the feature is available by looking for the IBM WebSphere Test Environment project in the All Projects frame, as shown in Figure 4. Note that even if you select only the WTE, VA Java also adds other projects that are required to support the WTE. You should not remove any of those automatically added projects, as they are required for the WTE.

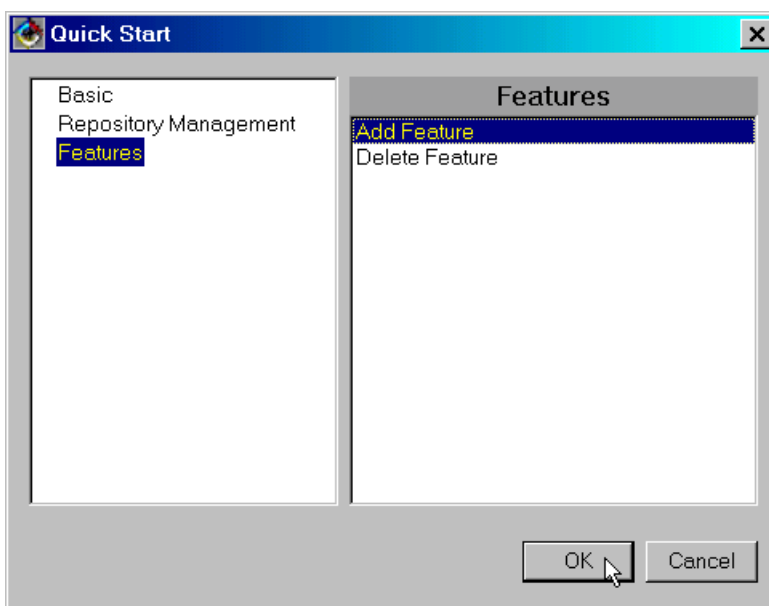


Figure 2: Select the Feature, Add Feature option on the Quick Start dialog to install additional VA Java features.

WTE002

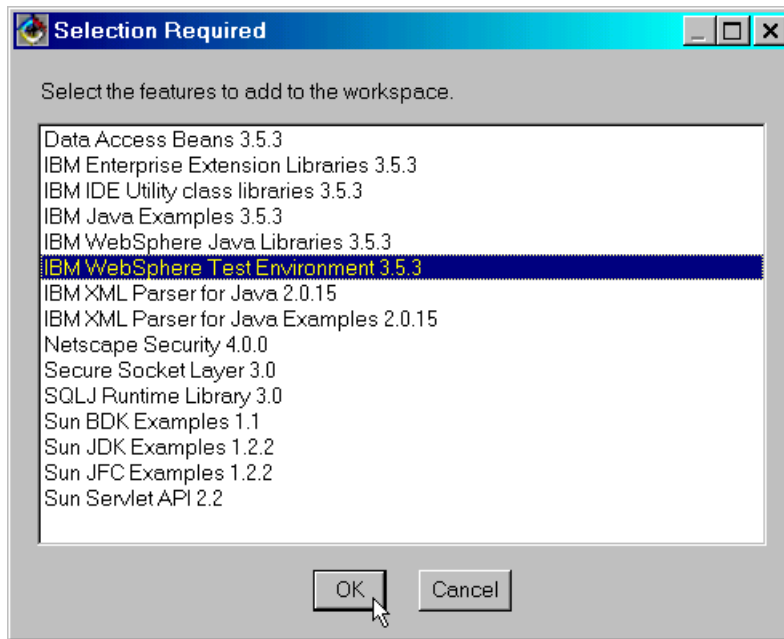


Figure 3: Select the IBM WebSphere Test Environment from the list of available features.

WTE003

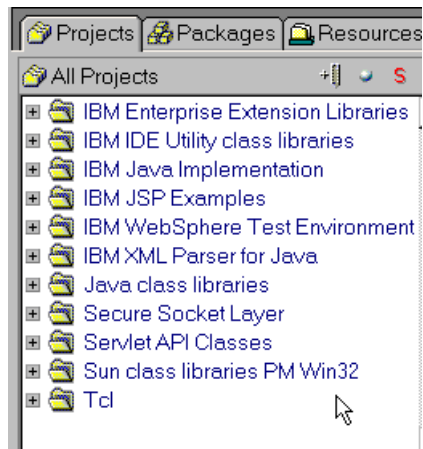


Figure 4: After adding the WTE, verify that the IBM WebSphere Test Environment project is in the All Projects list.

WTE004

Using the WTE

- **Develop web applications**
 - *Servlets in VA Java*
 - *JSPs in any editor*
- **Start WTE – VAJ menu**
 - *Workspace, Tools, WebSphere Test Environment*
- **WTE Control Center**
 - *Edit Class Path*
 - *Display trace messages*
 - *JSP Settings*

Copyright © 2001 Craig Pelkie. All rights reserved worldwide

Using the WTE, JSP Execution Monitor

- **Internal port number – 8082**
 - *Used to communicate between JEM and servlet test environment (on port 8080)*
- **Check – Enable monitoring JSP execution**
 - *Causes JEM to be displayed, active*
 - *Interactive debugger*
- **Check – Retrieve syntax error information**
 - *Causes “verbose” messages to be sent to browser for JSP syntax errors*

Copyright © 2001 Craig Pelkie. All rights reserved worldwide

Using the WebSphere Test Environment

Now that you have the WTE incorporated into VA Java, you can use it to test and debug your servlets and JavaServer Pages (JSP). If you already have a servlet or JSP that you want to test, you can start using the WTE immediately. If you have not yet developed a servlet or JSP, you can still start the WTE and begin working with it, using the sample servlets and JSP files that are included with the WTE.

You start the WTE by selecting the Workspace, Tools, WebSphere Test Environment menu item, as shown in Figure 5. After selecting that option, the WebSphere Test Environment Control Center is displayed (Figure 6). It may take up to a minute for the Control Center to appear, since there is a lot of initialization work done when you start the WTE. After the Control Center appears, click the plus sign next to the Servers item to expand the options, as shown in Figure 6. Note: the Control Center is a Java GUI application. In some cases, the buttons and other GUI elements are not rendered correctly when the Control Center is displayed. You can click the minimize icon in the upper right corner, then click the icon in the Windows tray to make the Control Center reappear. By forcing the GUI to repaint itself, you will usually be able to make it appear correctly.

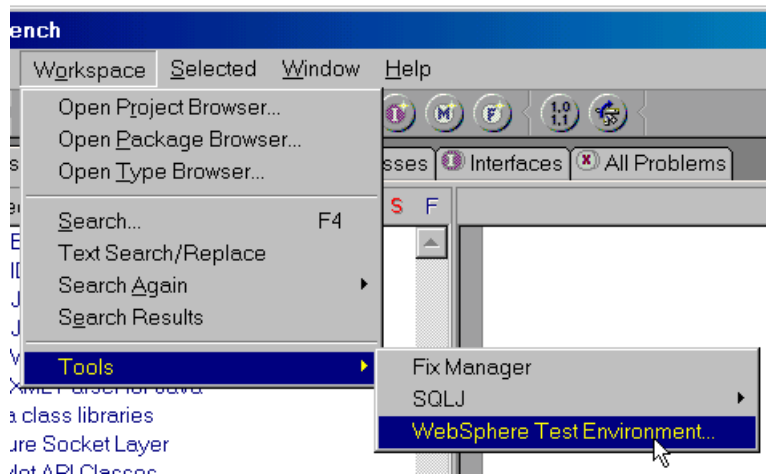


Figure 5: Start the WTE with the Workspace, Tools, WebSphere Test Environment menu item.

WTE005

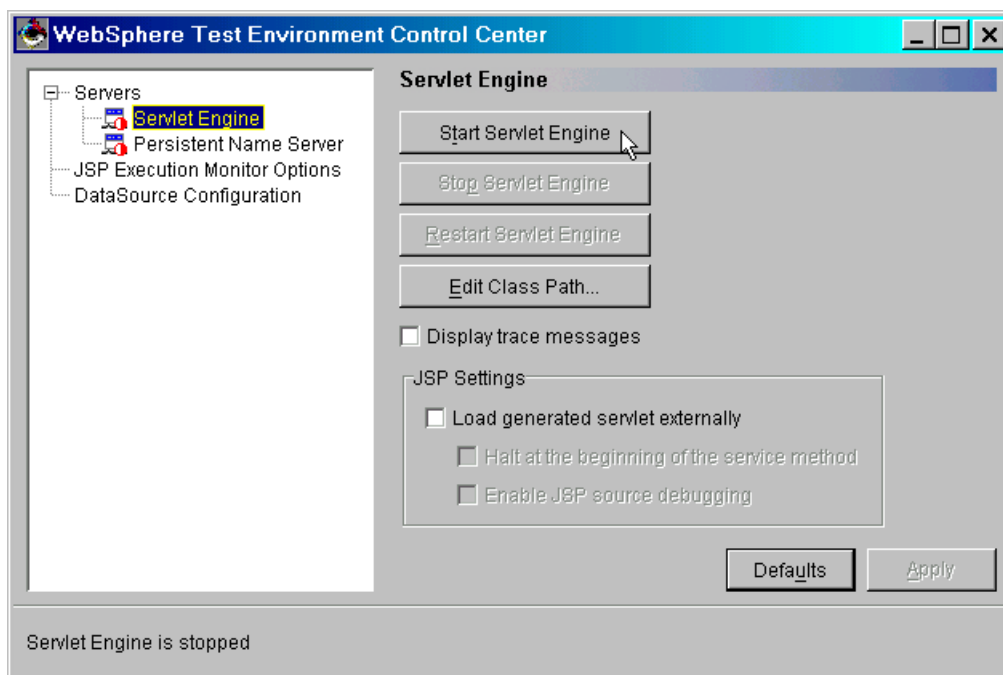
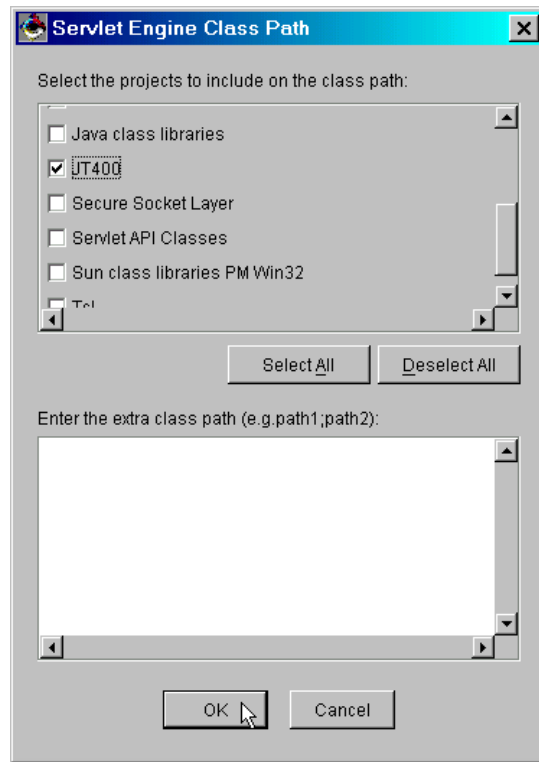


Figure 6: The WebSphere Test Environment Control Center is used to start and set options for the WTE.

WTE006

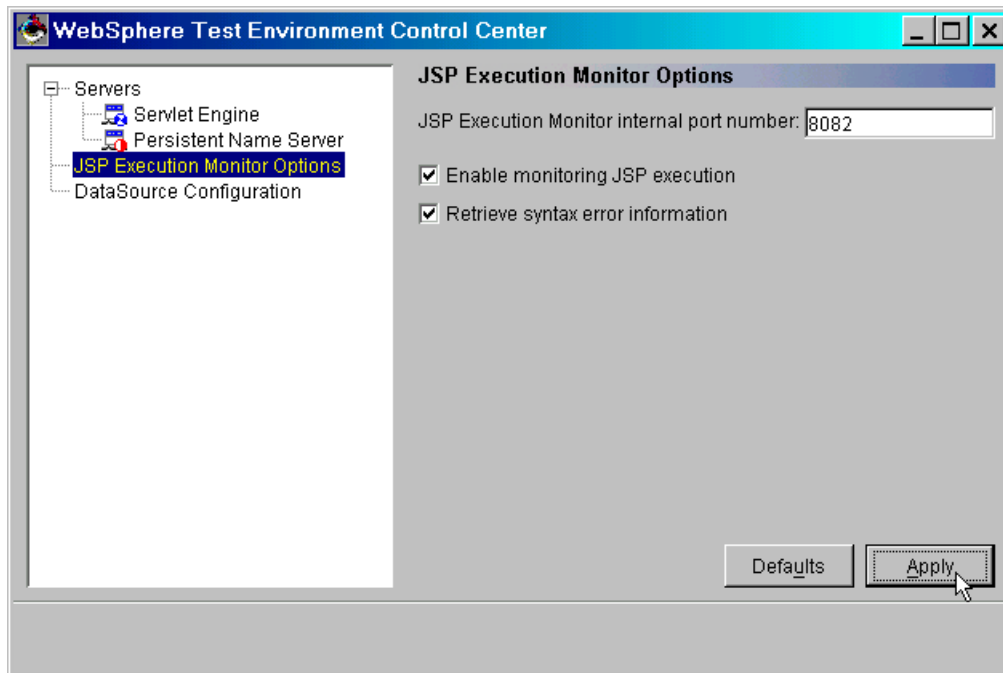
The first thing to do in the Control Center is click the Servlet Engine option, then click the Edit Class Path button. In the Servlet Engine Class Path (Figure 7), scroll through the list of projects and select the projects that include the Java classes you are using in your servlets and JSPs. Although you can click the Select All button to choose all of the current projects, you may not want to do that if you have many projects in your workspace, since it might cause your servlets and JSPs to run more slowly in the test environment. After selecting the projects to include in the class path, click the OK button to return to the Control Center.



WTE007

Figure 7: You can select a list of projects in your VA Java workspace to appear in the WTE class path.

On the Control Center, you can now click the JSP Execution Monitor Options. The options are shown in Figure 8. You should select both the Enable monitoring and Retrieve syntax options. You should leave the JSP execution monitor port set to its default of 8082 unless you know that you are running another TCP/IP application on your PC that uses that port. After checking the options, be sure to click the Apply button, as the options will not take effect unless you do.



WTE008

Figure 8: Check both of the options on the JSP Execution Monitor Options panel to start monitoring JavaServer Pages.

You can now click the Servlet Engine option again to return to the options shown in Figure 6. Click the Display trace messages option, then click the Start Servlet Engine button. The WTE servlet engine now loads, which is similar to starting the actual WebSphere Application Server. You will see the VA Java Console open (if it does not open as a visible window, click its icon in the Windows tray to bring the Console window to the foreground). In the Console you will see a series of status messages marking the progress of the WTE start up (see Figure 9). When you see the message Servlet Engine is started, you can begin testing servlets and JSPs.

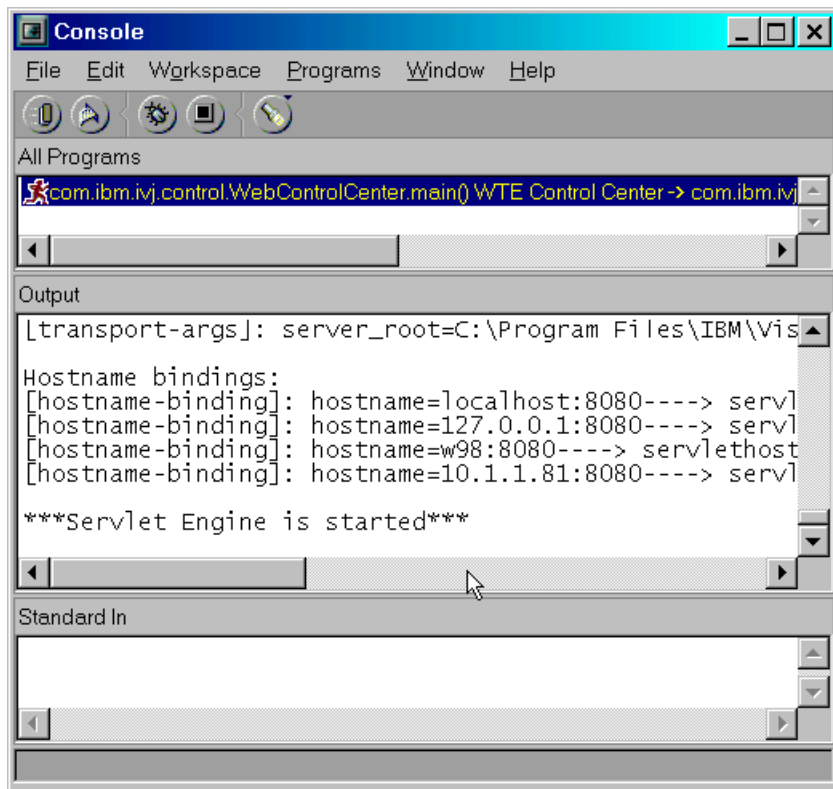


Figure 9: The VA Java Console displays status messages when you start the WTE.

WTE009

Start WTE / Initial Tests

- **WTE Control Center, Servlet Engine**
 - **Start Servlet Engine**
 - **Review trace messages in Console**
 - **Look for **** Servlet Engine is started ******
- **Run VAJ samples**
 - ***http://127.0.0.1:8080/jsp/index.html***
 - **Verifies that WTE started correctly**
 - **Use Sample 2 – Servlet Engine Configuration to review WTE environment**

Copyright © 2001 Craig Pelkie. All rights reserved worldwide

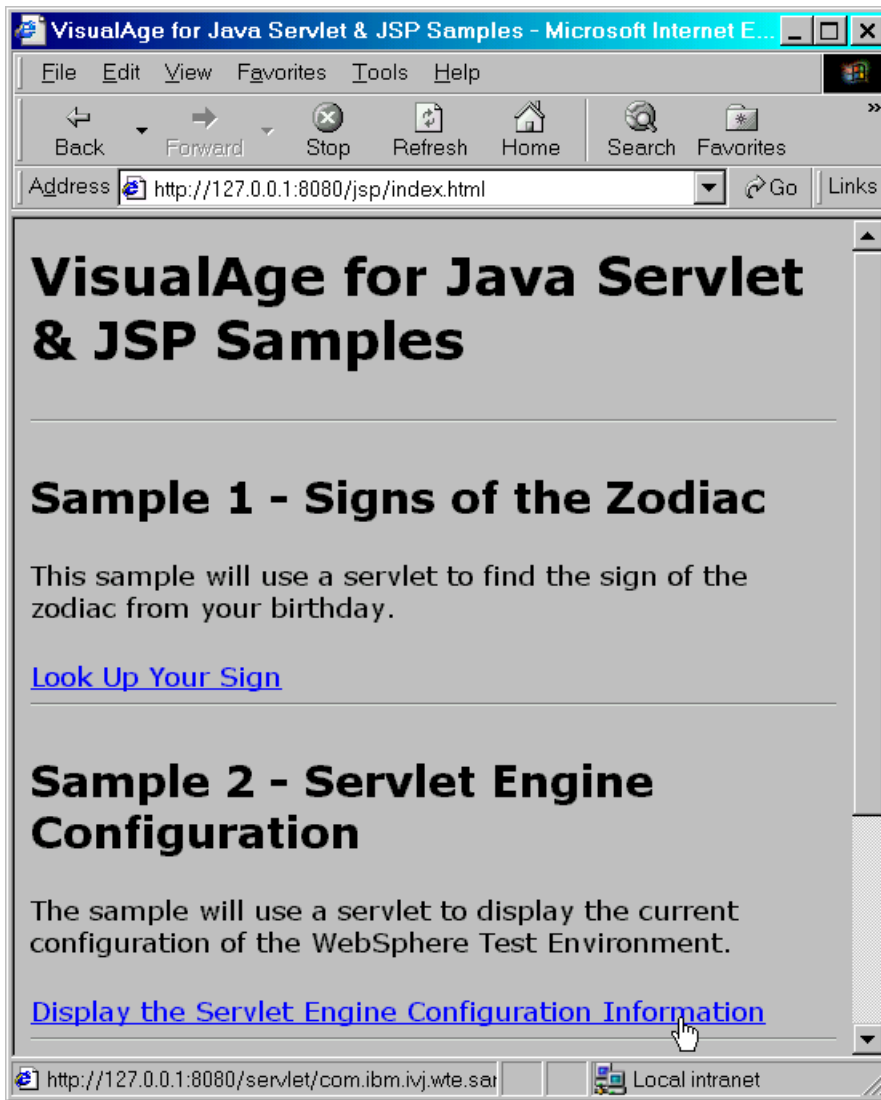
Run the Java Servlet and JSP Samples

Before testing your own servlets and JSPs, you should run the samples that are included with the WTE. I make it a practice to run the samples every time I start the WTE to verify that the WTE is started correctly and ready to work. If the samples do not run, it is unlikely that your own code will run either.

To load the samples, open your browser and enter the following URL:

`http://127.0.0.1:8080/jsp/index.html`

A page similar to Figure 10 should be displayed. There are three samples on the page; samples 1 and 3 are trivial programs that perform some calculations and display output to your browser. Sample 2, the Servlet Engine Configuration sample, is actually quite useful as it displays all of the options used by the WTE itself when it starts. The configuration options are stored in several properties files in the VA Java directories and can be edited if it is necessary to change the behavior of the WTE. In most cases, you will simply use the default configuration of the WTE to test your servlets and JSPs before exporting them to WAS on your server. There are no particular configuration tools provided with the WTE to change its configuration, beyond the options provided in the Control Center. IBM does document the properties files and the options you might want to change, but the explanations of the options are not always clear.



WTE010

Figure 10: Use the Servlet Engine Configuration link on the VA Java servlet and JSP Samples page to view the WTE configuration options.

The value of displaying the configuration, as shown in Figure 11, is that you can determine the document root directory. It is the incredibly long and complicated path shown in Figure 11. The significance of the document root is that you need to save your HTML and JSP files to that directory or in a subdirectory located in the document root directory so that the WTE will be able to locate and serve the files.

Testing a Servlet

- Set breakpoints in VA Java workspace before starting test
- Invoke servlet from browser:
 - › `http://127.0.0.1:8080/servlet/com.package.servlet`
 - › Note: *com.package.servlet is the complete package and class name of the servlet class*
- Output to the browser
- If any breakpoints set, VA Java Debugger window opens

Copyright © 2001 Craig Pelkie. All rights reserved worldwide

Testing a Servlet

To test a servlet, you enter a URL like this:

```
http://127.0.0.1:8080/servlet/com.web400.JdbcPartServlet
```

You need to specify the directory name `/servlet`, as that special name indicates to the WTE that it will be running a servlet. After that, you specify the full package-qualified name of the servlet class file. In the URL shown above, the servlet class file name is `JdbcPartServlet` and it is contained in a package named `com.web400`. Note that you request a servlet by its *package name*, which is not the same as the *project* that contains the package. A project is a VA Java construct used to organize groups of packages. The WTE does not care about your project name, but it does need to know the package name so that it can locate and serve your servlet.

Because the WTE is running under the control of VA Java, you can set breakpoints in your servlet class file in the VA Java Source window. When the WTE runs your servlet, it will stop at the breakpoints. You can then use the VA Java breakpoint tools to step through the source, examine and change values, or stop executing the servlet. The ability to debug a servlet at the source code level is quite valuable, especially when you have a servlet that runs but does not produce any visible output.

Testing a JSP

- **Develop JSP source code in your editor of choice**
- **Save JSP source code in default WTE directory:**
 - `\IBMVAJava\IDE\project_resources\IBM WebSphere Test Environment\hosts\default_host\default_app\web`
- **Start JSP from browser:**
 - ***http://127.0.0.1:8080/jspName.jsp***

Copyright © 2001 Craig Pelkie. All rights reserved worldwide

Working with the JSP Execution Monitor (JEM)

- **JSP Execution Monitor**
 - ***Opens after JSP is compiled to servlet***
 - ***3 panes:***
 - Upper left: Original JSP source
 - Upper right: compiled servlet
 - Lower: resulting HTML as generated
- **Use Actions to run JSP**
 - ***Step – steps through JSP source***
 - ***Run – run (show execution), “fast forward”***
 - ***Step into IDE debugger (debug the compiled servlet)***

Copyright © 2001 Craig Pelkie. All rights reserved worldwide

JSP Errors

- **Error trapped in JEM**
 - ***Stops at JSP source statement and servlet statement***
 - ***Shows as much of generated HTML as possible***
 - ***In browser: shows Java error(s), stack trace***

Copyright © 2001 Craig Pelkie. All rights reserved worldwide

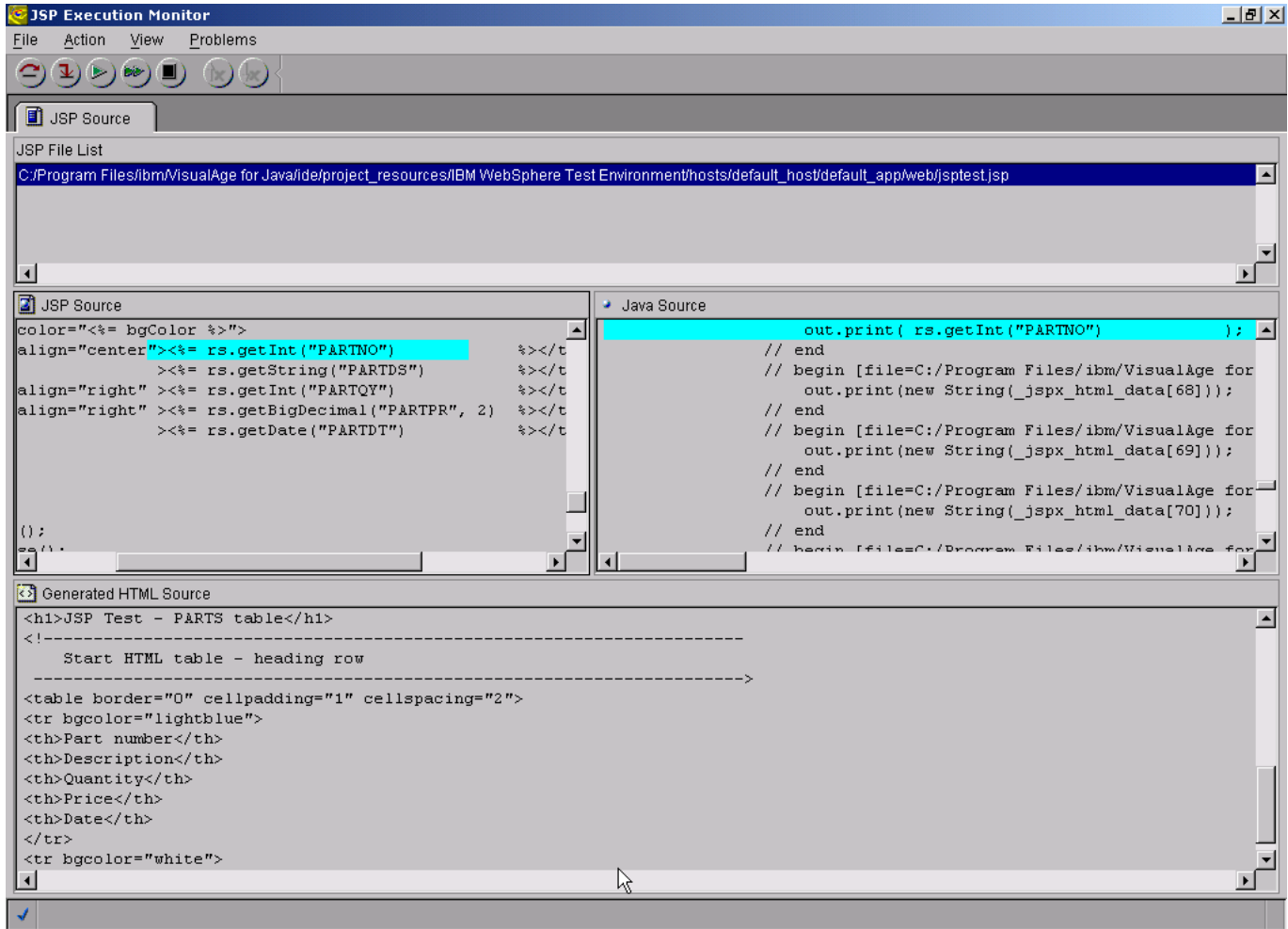
Testing a JSP

To test a JSP, you enter a URL like this:

`http://127.0.0.1:8080/jspTest.jsp`

In this case, the file `jspTest.jsp` is located in the document root directory, as shown in Figure 11.

When you run a JSP with the monitor option set (shown in Figure 8, JSP Execution Monitor Options), the JSP Execution Monitor ("JEM") opens, as shown in Figure 12. The JEM displays the source code of the JSP, the source code for the servlet that was created when the JSP was compiled and the resulting HTML from the execution of the JSP. Note, the JSP is automatically compiled to a servlet when you initially request it or after you make any changes to the JSP source code itself; you simply create, test and work with the JSP source.



WTE012

Figure 12: The JSP Execution Monitor is used to step through a JSP as it runs.

Using the JEM, you can step through the JSP code a line at a time. As you step through, you can view the corresponding servlet code that is executed and the resulting HTML. If there are any errors, you can open the VA Java Debugger and examine the value of any variables. You can also have the JSP continue executing in "run" mode (you can visually track its execution), or continue execution in "fast forward" mode (you simply see the resulting output, not each step of the execution).

Although JSPs are very easy to write, compared to the equivalent servlet code, they can be quite difficult to debug if you are simply depending on text output to the browser. Being able to perform breakpoint debugging on a JSP while developing it is a tremendous productivity aid.

About the URLs

You have probably noticed that the URL you use to invoke the WTE starts like this:

```
http://127.0.0.1:8080
```

If you are familiar with TCP/IP, you recognize the TCP/IP address as being the loopback address. In fact, you can enter any of the hostnames as shown in Figure 9 to start running a servlet or JSP in the WTE (`localhost`, `127.0.0.1`, the PC's network name or its unique TCP/IP address). The TCP/IP address or name in each case points back to the PC you are running the WTE on. The important part of the URL is the port 8080, which is assigned to the WTE when you start it running on your PC. In the unlikely event that you have another TCP/IP application running on your PC that uses that port, you will either have to reassign the other application's port number, or delve into the WTE configuration files and assign a different port number.

Note that the WTE itself uses port 8080 and the JEM uses port 8082 (see Figure 8). When you test a JSP, you test it by passing your request to port 8080, as shown in the URL above. The WTE uses port 8082 internally to talk with the JEM. You never enter 8082 as the URL port to invoke WTE services from the browser.

WTE Summary

The point of knowing about the WTE and how to work with it is twofold:

- 1) You can have a complete development, test and debugging environment on your PC. You can test servlet and JSP code without needing to export it to and configure it in WAS. If you copy some test data from your iSeries 400 database files to PC files (for example, to an Access database), you can use the JDBC-ODBC driver to connect to the test files and completely simulate the run-time environment of the servlet or JSP.
- 2) It is much simpler to test a servlet or JSP with the VA Java breakpoint and debugger facilities. Although you can establish a remote debugging session from VA Java to WAS running on the iSeries 400, it is much simpler to just start up the WTE on your PC and test the code there. Also, if you need to make modifications to a servlet or JSP that is already in production, it will be far more comfortable to make your modifications and tests off-line, rather than trying to change code that is in use.

Another point about the WTE is that it is not obvious from the VA Java packaging or environment that such a feature even exists. I remember being quite surprised to find out that this was included with VA Java, and especially surprised to find that IBM includes it in the Professional Edition of VA Java. The documentation for the WTE is included in the set of PDF manuals that are installed with VA Java, and can be accessed through the VA Java Help system (look for the "WebSphere Test Environment" and "JSP/Servlet Development Environment" manuals).

If you are just getting started with WebSphere programming, you should make the VisualAge for Java WebSphere Test Environment one of your first stops. If you are familiar with servlet and JSP programming, you will probably be pleasantly surprised to find out about this tool. Properly used, it will save you hours of debugging time.

