Configuring & Using Apache Tomcat 4

A Tutorial on Installing and Using Tomcat for Servlet and JSP Development

Taken from Using-Tomcat-4 found on http://courses.coreservlets.com

Following is a summary of installing and configuring Apache Tomcat 4 for use as a standalone Web server that supports servlets 2.3 and JSP 1.2. Integrating Tomcat as a plugin within the regular Apache server or a commercial Web server is more complicated (for details, see http://jakarta.apache.org/tomcat/tomcat-4.0-doc/).

Integrating Tomcat with a regular Web server is valuable for a deployment scenario, but my goal here is to show how to use Tomcat as a development server on your desktop. Regardless of what deployment server you use, you'll want a standalone server on your desktop to use for development. (Note: Tomcat is sometimes referred to as Jakarta Tomcat since the Apache Java effort is known as "The Jakarta Project").

The examples here assume you are using Windows, but they can be easily adapted for Solaris, Linux, and other versions of Unix. I've gotten reports of successful use on MacOS X, but don't know the setup details. Except when I refer to specific Windows paths (e.g., C:\blah\blah), I use URL-style forward slashes for path separators (e.g., install_dir/webapps/ROOT). Adapt as necessary.

The information here is adapted from More Servlets and JavaServer Pages from Sun Microsystems Press. For the book table of contents, index, source code, etc., please see http://www.moreservlets.com/. For information on servlet and JSP training courses (either at public venues or on-site at your company), please see http://courses.coreservlets.com. To report errors or omissions in this writeup or to inquire about an on-site servlet and JSP training course, please contact Marty at hall@coreservlets.com.

Install the JDK

Your first step is to download and install Java. The servlet 2.3 specification requires Java 2; I suggest JDK 1.3 or later. See the following sites for download and installation information. Once you've installed Java, confirm that everything including your PATH is configured properly by opening a DOS window and typing "java -version" and "javac -help". You should see a real result both times, not an error message about an unknown command. Or, if you use an IDE, compile and run a simple program to confirm that the IDE knows where you installed Java.

- JDK 1.3 (Windows, Linux, Solaris): http://java.sun.com/j2se/1.3/.
- JDK 1.4 (Windows, Linux, Solaris): http://java.sun.com/j2se/1.4/download.html. Be sure you download the full Software Development Kit (SDK) from entries in the right hand column, not just the JRE (Java Runtime Environment). The JRE is only for running already-compiled class files, and lacks a compiler.
- Java 2 (Other platforms): http://java.sun.com/cgi-bin/java-ports.cgi.

Configure Tomcat

Configuring Tomcat involves six steps:

1. Downloading the Jakarta Tomcat software.
2. Enabling the ROOT Context.

3. Changing the port from 8080 to 80.
4. Telling Tomcat to reload servlets when they are modified.
5. Setting the JAVA_HOME variable.
6. Changing the DOS memory settings.
7. Setting the CATALINA_HOME variable.

Details of each step are given below.

1. Download the Tomcat Software

Go to http://jakarta.apache.org/builds/jakarta-tomcat-4.0/release/, select the latest version number, and choose "binary". Download the zip file to your PC and unzip it into a location of your choice. You specify the top-level directory (e.g., C:\). The zip file has embedded subdirectories (e.g., jakarta-tomcat-4.0.1). Thus, C:\jakarta-tomcat-4.0.1 is a common resultant installation directory. Note: from this point forward, I'll refer to that location as install_dir. (Note: both JDK and Tomcat are available in the Documents folder at http://csis.pace.edu/~wolf/.)

2. Enable the ROOT Context.

The ROOT context is the default Web application in Tomcat, and is convenient to use when you are first learning about servlets and JSP (although you'll use your own Web applications once you're more experienced). The default Web application is already enabled in Tomcat, Tomcat 4.0.1-4.0.3, and Tomcat 4.1. But, in Tomcat 4.0.4 it is disabled by default. To enable it, uncomment the following line in install_dir/conf/server.xml:

```
<Context path="" docBase="ROOT" debug="0"/>
```

3. Change the Port to 80

Assuming you have no other server already running on port 80, you'll find it convenient to configure Tomcat to run on the default HTTP port (80) instead of 8080. Making this change lets you use URLs of the form http://localhost/blah instead of http://localhost:8080/blah. Note that you need admin privileges to make this change on Unix. Also note that Windows XP automatically starts IIS on port 80. So, if you use XP and want to use port 80 for Tomcat, you'll need to disable IIS.

To change the port, edit install_dir/conf/server.xml and change the port attribute of the Connector element from 8080 to 80, yielding the result below.

```
<Connector
  className="org.apache.catalina.connector.http.HttpConnector"
  port="80" ...
  ... />
```

4. Turn on Servlet Reloading

The next step is to tell Tomcat to check the modification dates of the class files of requested servlets and reload ones that have changed since they were loaded into the server's memory. This degrades performance in deployment situations, so is turned off by default. However, if you fail

to turn it on for your development server, you'll have to restart the server every time you recompile a servlet that has already been loaded into the server's memory.
To turn on servlet reloading, edit install_dir/conf/server.xml and add a DefaultContext subelement to the main Service element and supply true for the reloadable attribute. The easiest way to do this is to find the following comment:

```
<!-- Define properties for each web application.  This is only needed
    if you want to set non-default properties, or have web application
    document roots in places other than the virtual host's appBase
    directory.  -->
```

and insert the following line just below it:

```
<DefaultContext reloadable="true"/>
```

Be sure to make a backup copy of server.xml before making the above change.

5. Set the JAVA_HOME Variable

Next, you must set the JAVA_HOME environment variable to tell Tomcat where to find Java. Failing to properly set this variable prevents Tomcat from handling JSP pages. This variable should list the base JDK installation directory, not the bin subdirectory. For example, if you are on Windows 98/Me and installed the JDK in C:\JDK1.3_01, you might put the following line in your autoexec.bat file.

```
set JAVA_HOME=C:\JDK1.3_01
```

On Windows NT/2000/XP, you would go to the Start menu and select Settings, then Control Panel, then System, then Environment. Then, you would enter the JAVA_HOME value.

Rather than setting the JAVA_HOME environment variable globally in the operating system, some developers prefer to edit the startup script to set it there. If you prefer this strategy, edit install_dir/bin/catalina.bat and change the following:

```
if not "%JAVA_HOME%" == "" goto gotJavaHome
echo You must set JAVA_HOME to point at ...
goto cleanup
:gotJavaHome
```

to:

```
if not "%JAVA_HOME%" == "" goto gotJavaHome
set JAVA_HOME=C:\JDK1.3_01
:gotJavaHome
```

Be sure to make a backup copy of catalina.bat before making the changes. Also note that the exact details of the existing catalina.bat file change a bit in different minor releases of Tomcat 4.

6. Change DOS Memory Settings

If you use Windows, you may also have to change the DOS memory settings for the startup and shutdown scripts. If you get an "Out of Environment Space" error message when you start the server, you will need to right-click on install_dir/bin/startup.bat, select Properties, select Memory, and change the Initial Environment entry from Auto to at least 2816. Repeat the process for install_dir/bin/shutdown.bat.

7. Set the CATALINA_HOME Variable

If you are going to make copies of the Tomcat startup or shutdown scripts, it is also helpful to set the CATALINA_HOME environment variable to refer to the top-level directory of the Apache Tomcat installation (e.g. C:\jakarta-tomcat-4.0). This variable identifies the Tomcat installation directory to the server. However, if you are careful to avoid copying the server startup scripts and you use only shortcuts (called "symbolic links" on Unix/Linux) instead, you are not required to set this variable.

Test the Server

Testing the server involves two steps:
1. Verifying that the server can even start.
2. Checking that you can access your own HTML and JSP pages.

1. Verify That the Server Can Start

Before trying your own servlets or JSP pages, you should make sure that the server is installed and configured properly. For Tomcat, click on install_dir/bin/startup.bat (or execute install_dir/bin/startup.sh on Unix/Linux). Next, enter the URL http://localhost/ in your browser and make sure you get the Tomcat welcome page, not an error message saying that the page cannot be displayed or that the server cannot be found. If you chose not to change the port number to 80 as described above, you will need to use a URL like http://localhost:8080/ that includes the port number.

To halt the server, double click on install_dir/bin/shutdown.bat. I recommend that you make shortcuts to (not copies of) the startup and shutdown scripts and place those shortcuts on the desktop or in your main development directory. If you use an IDE, you'll have to tell it where these scripts are.

2. Try Some Simple HTML and JSP Pages

After you have verified that the server is running, you should make sure that you can install and access simple HTML and JSP pages. This test, if successful, shows two important things. First, successfully accessing an HTML page shows that you understand which directories should hold HTML and JSP files. Second, successfully accessing a new JSP page shows that the Java compiler (not just the Java virtual machine) is configured properly.