**MODULE 2**

**Installing Tomcat 7**

In the previous module, you saw a brief history of the Internet and the Web that built up to the development of servlets and the release of Tomcat and performed review of Linux virtualization technolies. Continuing in this abstract manner, you learned about Tomcat’s modular architecture. However, none of this is useful if you don’t have the Tomcat server, so in this module you’ll do the following:

* You’ll install Java if you haven’t done so already.
* You’ll install Tomcat on your platform.

You’ll also see how to compile Tomcat from the source code provided on the Tomcat web site. This process is the same on Windows and Linux and requires the Ant build tool, so you’ll see how to do it once all the other installation techniques have been covered.

**Installing Java**

Your choice of JVM can significantly affect the performance of your Tomcat server, and it’s worth evaluating a few to see which gives you the best performance. This is a subject that many people don’t concern themselves with or have never thought about, so you won’t be alone if you think that this isn’t an issue. Sun Microsystems’ JVM is all you need, right?

Well, if performance is really an issue and you want to squeeze as much out of your server setup as possible, you should look into this area. You can find a lot of information on the Internet, and Sun provides its own guidance at <http://java.com/docs/performance/.> IBM ([www.ibm.com/developerworks/java/jdk/)](http://www.ibm.com/developerworks/java/jdk/)) and the Blackdown project ([www.](http://www/)blackdown.org), which is a Linux port of source donated by Sun Microsystems, provide the main alternatives to Sun Microsystems’ Java development kit (JDK).

Download a suitable distribution from <http://java.com/j2se/downloads/.> Two types of download exist: a self-extracting binary file and an RPM package for systems supporting RPMs. As mentioned earlier, Tomcat 7 works with JDK 1.6 and later.

You can download the latest JDK here: <http://www.oracle.com/technetwork/java/javase/downloads/index.html>  
We'll install the latest JDK, which is JDK 7, Update 71. The JDK is specific to 32 and 64 bit versions. If you are on 32 bit, you'll need: jdk-7u71-linux-i586.tar.gz  
Start by creating a new directory /usr/java:

*[root@tomcat ~]# mkdir /usr/java*

Now change the directory to the one where you want to install Java, and execute the binary.

You must prefix the binary’s filename with any path information that’s necessary, like so:

*[root@tomcat: ~]# cd /usr/java*

*[root@tomcat: java ]#*

Download the appropriate JDK and save it to /usr/java directory we created above.  
Unpack jdk-7u5-linux-x64.tar.gz in the /usr/java directory using tar -xzf:

*[root@tomcat: java]# tar -xzf jdk-7u71-linux-x64.tar.gz*

This will create the directory /usr/java/jdk1.7.0\_71. This will be our JAVA\_HOME. We can now set JAVA\_HOME and put Java into the path of our users. To set it for your current session, you can issue the following from the CLI:

*[root@tomcat: java]# export JAVA\_HOME=/usr/java/jdk1.7.0\_71*

*[root@tomcat: java]# export PATH=$JAVA\_HOME/bin:$PATH*

# Installing Tomcat

Now that you’ve installed Java, it’s time for you to install the Tomcat server. The first step for all systems is obtaining the appropriate distribution. This may be a binary or source distribution, depending on your needs. Whatever your requirements, Tomcat is available from <http://tomcat.apache.org/download-70.cgi> Choose the most stable version of Tomcat 7 provided. At the time of the writing, this was the 7.0.71 release. If you’re interested in the latest version of Tomcat or want to download an older version, you’ll find both of these options below the binary downloads. You’ll also require Ant 1.65 or above for various deploy and build tasks later in the module.

*[root@tomcat: ~]# cd /usr/share*

Download apache-tomcat-7.0.29.tar.gz (or the latest version) <http://tomcat.apache.org/download-70.cgi> and save it to /usr/share directory. Once downloaded, you should verify the MD5 Checksum for your Tomcat download using the md5sum command.

*[root@tomcat: share ]#md5sum apache-tomcat-7.0.57.tar.gz*

*[root@tomcat: share ]# tar -xzf apache-tomcat-7.0.57.tar.gz*

This will create the directory /usr/share/apache-tomcat-7.0.71

You can now start Tomcat by running the following shell command:

*[root@tomcat: java]# $CATALINA\_HOME/bin/startup.sh*

You can shut down Tomcat using

*[root@tomcat: java]# $CATALINA\_HOME/bin/shutdown.sh*

*As a lab activity you’ll create redhat/centos service for tomcat7 to start/stop tomcat using init.d functionality and service command.*

## **Viewing the Default Installation**

## http://www.davidghedini.com/images/TomcatManager.jpg

## *Figure 2-1. Default Tomcats page*

To check that Tomcat is running, point your browser to http://localhost:8080. You should see a screen like the one in Figure 2-1.

To check that the dynamic side of Tomcat’s functionality is working, choose the JSP Examples link from the menu on the left, and select some of the examples. Check that they run without error messages. Do the same with the Servlet Examples link to test this functionality.

If you have any problems, refer to the “Troubleshooting and Tips” section later in this module.

# Installing Ant

Before you install Tomcat from source, or indeed before you start any serious Java-based project, you should install Ant. Ant is a Java-based build tool that has become ubiquitous. You use it to build and deploy applications. It benefits from platform independence and can use a single build file on multiple platforms. However, the build files must minimize dependency on a specific file path. (Windows paths, for example, will cause problems on Linux and vice versa.)

You can download the latest binary distribution of Ant from <http://ant.apache.org/>bindownload.cgi. Ant is easy to install; simply unpack the distribution to a convenient location. Because Ant is a program that you’ll use on a number of projects, you should make it available from any directory. To do this, add it to your path, and add an ANT\_HOME environment variable as you did with CATALINA\_HOME. It’s a good idea to set the entry in the path to ANT\_HOME\bin to allow for any updates to Ant that you may make.

To test that you’ve installed Ant, type **ant -version** in a terminal window. If everything has gone according to plan, you’ll see Ant’s usage message.

You won’t use Ant for anything but compiling the source code and deploying web applications in this module, so you won’t see the details of it here. However, you should be aware that it uses an XML file, called build.xml by default, to carry out its tasks.

**Installing Tomcat from Source**

If you want to obtain the latest version of Tomcat with the newest bug fixes and upgrades, then installing it from the source code is a good option. In Linux, it’s far more common for servers to be built for the system. However, this isn’t strictly necessary for a Java-based server such as Tomcat.

Tomcat is easily built using the Ant build utility. You use Ant for automated project building, including compilation and deployment. It has all the system-independent benefits that Java enjoys, because it’s written in Java.

You can also use Ant to carry out a number of administrative actions on Tomcat, each of which is described in Module 6. The deployer application mentioned previously also uses Ant.

It used to be the case that you had to manually download a huge number of libraries from many different sources to compile Tomcat, but now Ant can do it for you. All the instructions on how to build from the source are available at <http://tomcat.apache.org/tomcat-7.0-doc/>building.html. You will need to download a JDK, ANT, and the source code from the Subversion repository. In addition, you will need to create a build.properties file in the same directory with the appropriate lines from Listing 2-1. Those with # marks are commented out and can be ignored if they don’t apply to your installation. You should ensure that base.path points to the place you want to download.

**Listing 2-1.** *Ant’s* build.properties *File*

# ----- Default Base Path for Dependent Packages -----

# ----- Linux/Unix path ----- base.path=/usr/share/java

# ----- Proxy setup -----

# Uncomment if using a proxy server

#proxy.host=proxy.domain

#proxy.port=8080

#proxy.use=on

Once you’re satisfied with your setup, you can build Tomcat using the following line in the base directory:

*[root@tomcat: share ]#* ant

The build will take a few minutes, and the resultant build is the subdirectory tomcat-7/build. To deploy the new server, move (and rename) it out of the source folder and into a folder of its own, and set the CATALINA\_HOME variable using the instructions given previously. If you want to update the source code and recompile it, use the following commands in the source directory:

*[root@tomcat: share ]#ant checkout*

*[root@tomcat: share ]#ant build*

The second command will compile only those files that have changed, so you can also use it to compile the server if you’ve made any changes of your own to the source.

**Troubleshooting and Tips**

Finally, before I close this module, I’ll cover the typical problems that may occur when you install Tomcat. If you have further problems, you can find more material on the Tomcat web site at <http://apache.org/tomcat/> and at [http://java.sun.com,](http://java.sun.com/) as well as on various forums. You should also read the release notes available with each download. The following problems are typically encountered when first installing Tomcat.

**The Port Number Is in Use**

Tomcat uses port 8080 by default, as mentioned previously. You can check if another program is using this port by using netstat. Typing netstat –nlpt into your shell/command prompt will list open ports on your system and should show the process that’s interfering with Tomcat. You have two options: shut the process down or change Tomcat’s port as described earlier.

A common problem is trying to start a new Tomcat instance when one is still running. This is especially true if it’s running as a daemon thread. If you suspect this is the case, you can check it by using telnet to connect to the socket, as follows, and see if you’re given a connection:

*[root@tomcat: share ]# telnet localhost 8080*

If you’re awarded a connection, the screen goes blank rather than giving an error. When you’re connected, type **GET /** and press Return or Enter. (Echo is turned off by default on Windows, so it looks a little strange, because typing doesn’t appear to achieve anything.) This results in the following output:

*<?xml version="1.0" encoding="ISO-8859-1"?>*

*<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"* [*"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">*](http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd)

[*<html xmlns="http://www.w3.org/1999/xhtml"*](http://www.w3.org/1999/xhtml) *xml:lang="en" lang="en">*

*<head>*

*<title>Apache Tomcat</title>*

*<style type="text/css">*

*/\*<![CDATA[\*/ body {*

*color: #000000;*

*background-color: #FFFFFF;*

*font-family: Arial, "Times New Roman", Times, serif; margin: 10px 0px;*

*}*

*…Output trancated*

*</body>*

*</html>*

*Connection to host lost.*

If you see a different message, you might have another web server running on this port. Even if you’re refused a connection, this indicates that a process is sitting on that port. If the connection fails, then try one of the other possibilities.