Introduction to Apache Tomcat 7

In this chapter, we introduce the world of Apache Tomcat server.
Throughout this chapter, we

- Describe the Apache Tomcat architecture
- Discuss the requirements for installing and configuring Tomcat
- Describe the steps of installing and configuring Tomcat
- Test your Tomcat installation

At the end of this chapter, you will understand the Tomcat architecture, have an instance of Tomcat server installed and running on your computer, and have a sample web application displayed in your browser.

The Apache Tomcat Server

The Apache Tomcat server is an open source, Java-based web application container that was created to run servlet and JavaServer Pages (JSP) web applications. It was created under the Apache-Jakarta subproject; however, due to its popularity, it is now hosted as a separate Apache project, where it is supported and enhanced by a group of volunteers from the open source Java community.

Apache Tomcat is very stable and has all of the features of a commercial web application container—yet comes under Open Source Apache License. Tomcat also provides additional functionality that makes it a great choice for developing a complete web application solution. Some of the additional features provided by Tomcat—other than being open source and free—include the Tomcat Manager application, specialized realm implementations, and Tomcat valves.

Currently supported versions on Apache Tomcat are 5.5X, 6.0X, and 7.0X. Versions earlier than 5.5 are still available for download, but they are archived and no support is available for them, so users are encouraged to use the latest possible version of Tomcat where available.

Major versions on Apache Tomcat coincide with versions of the Java Servlet specification, or Java Servlet API, released. So, Tomcat 5.5X supports Servlet API 2.3, Tomcat 6.0X supports Servlet API 2.4, and the latest Tomcat 7.0 is a reference implementation of current Servlet API 3.0. In addition to Servlet API versions, Tomcat versions support corresponding JSP API versions.

The JVM compatibility also depends on the version chosen. Table 1-1 provides a cross-reference of Tomcat versions, supported JVM versions, and Servlet API and JSP API releases.
Table 1-1. Tomcat Versions and Supported API and JDK Versions

<table>
<thead>
<tr>
<th>Apache Tomcat</th>
<th>Servlet API</th>
<th>JSP API</th>
<th>JDK</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.0</td>
<td>3.0</td>
<td>2.2</td>
<td>1.6</td>
</tr>
<tr>
<td>6.0</td>
<td>2.5</td>
<td>2.1</td>
<td>1.5</td>
</tr>
<tr>
<td>5.5</td>
<td>2.4</td>
<td>2.0</td>
<td>1.4</td>
</tr>
<tr>
<td>4.1</td>
<td>2.3</td>
<td>1.2</td>
<td>1.3</td>
</tr>
<tr>
<td>3.0</td>
<td>2.2</td>
<td>1.1</td>
<td>1.1</td>
</tr>
</tbody>
</table>

This book will cover version 7 of the Apache Tomcat Server. However, most of the content can be applied to versions 5.5 and 6—where that is not possible, it will be clearly stated.

The Tomcat Manager Web Application

The Tomcat Manager web application is packaged with the Tomcat server. It is installed in the context path of /manager and provides the basic functionality to manage web applications running in the Tomcat server from any web browser. Some of the provided functionality includes the ability to install, start, stop, remove, and report on web applications. Chapter 4 covers the details of the Tomcat Manager web application.

Specialized Realm Implementations

Tomcat provides container-managed security methods for protecting resources within the container. These “databases” of users that can be authenticated by the container are called realms.

We will cover two types of realms supported by Tomcat in more detail: MemoryRealm, where user information is simply read from a file and stored in memory, and JDBCRealm, which uses relational database to store users. You can read more about realms with examples in Chapter 6.

Tomcat Valves

Tomcat valves are a technology introduced with Tomcat 4, and available in all later versions. Valves allow you to associate an instance of a Java class with a particular Catalina container. The configured valve class is then acting as a preprocessor for all requests coming to the container. Valves are proprietary to Tomcat and cannot, at this time, be used in a different servlet/JSP container.

Servlet API defines similar functionality in form of Filters. We will also discuss the differences between valves and servlet filter implementation in Chapter 8.

Further Information

Throughout this book, we will discuss all of these Tomcat-specific features, and a lot of other features that are common to all web application containers. More information about Tomcat can be found on its homepage at http://tomcat.apache.org, which is shown in Figure 1-1.