**ASSIGNMENT NO. 1 A**

**Title:** Installation and Configuration of Web Application Servers.

**Objectives:**

1. To install and configure web application server.
2. To learn about application manager role in web server
3. To deploy web application on server access it in network

**Problem Statement:**

Installation and Configuration of Web Application Servers Tomcat, Apache, IIS.

**Outcomes:**

On completion of the course, student will be able to–

1. install Web Application Server
2. deploy their own web application on server
3. configure server for different service
4. manage their own web application through console

**Tools Required:**

1. Tomcat any version (we use 8.5) binary distribution for windows/linux or .exe
2. Operating system –Windows operating system/ open source linux
3. JDK 1.8
4. IIS

**Theory:**

**Apache Tomcat HTTP Server**

Apache Tomcat is a Java-capable HTTP server, which could execute special Java programs known as "Java Servlet" and "Java Server Pages (JSP)". Tomcat is an open-source project, under the "Apache Software Foundation" (which also provides the most use, open-source, industrial-strength Apache HTTP Server). The mother site for Tomcat is http://tomcat.apache.org. Alternatively, you can find tomcat via the Apache mother site @ http://www.apache.org.

Tomcat was originally written by James Duncan Davison (then working in Sun), in 1998, based on an earlier Sun's server called Java Web Server (JWS). It began at version 3.0 after JSWDK 2.1 it replaced. Sun subsequently made Tomcat open-source and gave it to Apache.

**The various Tomcat releases are:**

Tomcat 3.x (1999): RI for Servlet 2.2 and JSP 1.1.

Tomcat 4.x (2001): RI for Servlet 2.3 and JSP 1.2.

Tomcat 5.x (2002): RI for Servlet 2.4 and JSP 2.0.

Tomcat 6.x (2006): RI for Servlet 2.5 and JSP 2.1.

Tomcat 7.x (2010): RI for Servlet 3.0, JSP 2.2 and EL 2.2.

Tomcat 8.x (2013): RI for Servlet 3.1, JSP 2.3, EL 3.0 and Java WebSocket 1.0.

Tomcat is an HTTP application runs over TCP/IP. In other words, the Tomcat server runs on a specific TCP port from a specific IP address. The default TCP port number for HTTP protocol is 80, which is used for the production HTTP server. For test HTTP server, you can choose any unused port number between 1024 and 65535.

**STEP 1: Download and Install Tomcat**

**For Windows**

Go to http://tomcat.apache.org ⇒ Under "Tomcat 8.5.{xx} Released" (where {xx} is the latest upgrade number) ⇒ Downloads ⇒ Under "8.5.{xx}" ⇒ Binary Distributions ⇒ Core ⇒ "ZIP" package (e.g., "apache-tomcat-8.5.{xx}.zip", about 9 MB).

Create your project directory, say "d:\myProject" or "c:\myProject". UNZIP the downloaded file into your project directory. Tomcat will be unzipped into directory "d:\myProject\apache-tomcat-8.0.{xx}".

For ease of use, we shall shorten and rename this directory to "d:\myProject\tomcat".

Take note of Your Tomcat Installed Directory. Hereafter, I shall refer to the Tomcat installed directory as <TOMCAT\_HOME>.

**Tomcat's Directories**

Take a quick look at the Tomcat installed directory. It contains the following sub-directories:

**bin:** contains the binaries; and startup script (startup.bat for Windows and startup.sh for Unixes and Mac OS), shutdown script (shutdown.bat for Windows and shutdown.sh for Unix and Mac OS), and other binaries and scripts.

**conf:** contains the system-wide configuration files, such as server.xml, web.xml, context.xml, and tomcat-users.xml.

**lib:** contains the Tomcat's system-wide JAR files, accessible by all webapps. You could also place external JAR file (such as MySQL JDBC Driver) here.

**logs:** contains Tomcat's log files. You may need to check for error messages here.

**webapps:** contains the webapps to be deployed. You can also place the WAR (Webapp Archive) file for deployment here.

**work:** Tomcat's working directory used by JSP, for JSP-to-Servlet conversion.

**temp:** Temporary files.

**STEP 2: Create an Environment Variable JAVA\_HOME**

**(For Windows)**

**You need to create an environment variable called "JAVA\_HOME" and set it to your JDK installed directory.**

First, find your JDK installed directory. The default is "c:\Program Files\Java\jdk1.8.0\_{xx}", where {xx} is the upgrade number. Take note of your JDK installed directory.

To set the environment variable JAVA\_HOME in Windows 7/8/10: Start "Control Panel" ⇒ System and Security (Optional) ⇒ System ⇒ Advanced system settings ⇒ Switch to "Advanced" tab ⇒ Environment Variables ⇒ System Variables ⇒ "New" ⇒ In "Variable Name", enter "JAVA\_HOME" ⇒ In "Variable Value", enter your JDK installed directory as noted in Step 1.

To verify, RE-START a CMD shell (restart needed to refresh the environment) and issue:

**SET JAVA\_HOME**

**JAVA\_HOME=c:\Program Files\Java\jdk1.8.0\_{xx} <== Verify that this is YOUR JDK installed directory**

**STEP 3: Configure Tomcat Server**

The Tomcat configuration files are located in the "conf" sub-directory of your Tomcat installed directory, e.g. "d:\myProject\tomcat\conf" (for Windows) or "/Applications/tomcat/conf" (for Mac OS). There are 4 configuration XML files:

server.xml

web.xml

context.xml

tomcat-users.xml

Make a BACKUP of the configuration files before you proceed!!!

**Step 3(a) "conf\server.xml" - Set the TCP Port Number**

Use a programming text editor (e.g., NotePad++, TextPad, Sublime, Atom for Windows; or gEdit, jEdit, Sublime, Atom for Mac OS) to open the configuration file "server.xml", under the "conf" sub-directory of Tomcat installed directory.

The default TCP port number configured in Tomcat is 8080, you may choose any number between 1024 and 65535, which is not used by an existing application. We shall choose 9999 in this article. (For production server, you should use port 80, which is pre-assigned to HTTP server as the default port number.)

Locate the following lines (around Line 69) that define the HTTP connector, and change port="8080" to port="9999".

***<!-- A "Connector" represents an endpoint by which requests are received***

***and responses are returned. Documentation at :***

***Java HTTP Connector: /docs/config/http.html (blocking & non-blocking)***

***Java AJP Connector: /docs/config/ajp.html***

***APR (HTTP/AJP) Connector: /docs/apr.html***

***Define a non-SSL HTTP/1.1 Connector on port 8080***

***-->***

***<Connector port="9999" protocol="HTTP/1.1"***

***connectionTimeout="20000"***

***redirectPort="8443" />***

**Step 3(b) "conf\web.xml" - Enabling Directory Listing**

Again, use a programming text editor to open the configuration file "web.xml", under the "conf" sub-directory of Tomcat installed directory.

We shall enable directory listing by changing "listings" from "false" to "true" for the "default" servlet. This is handy for test system, but not for production system for security reasons.

Locate the following lines (around Line 103) that define the "default" servlet; and change the "listings" from "false" to "true".

***<!-- The default servlet for all web applications, that serves static -->***

***<!-- resources. It processes all requests that are not mapped to other -->***

***<!-- servlets with servlet mappings. -->***

***<servlet>***

***<servlet-name>default</servlet-name>***

***<servlet-class>org.apache.catalina.servlets.DefaultServlet</servlet-class>***

***<init-param>***

***<param-name>debug</param-name>***

***<param-value>0</param-value>***

***</init-param>***

***<init-param>***

***<param-name>listings</param-name>***

***<param-value>true</param-value>***

***</init-param>***

***<load-on-startup>1</load-on-startup>***

***</servlet>***

**Step 3(c) "conf\context.xml" - Enabling Automatic Reload**

We shall add the attribute reloadable="true" to the <Context> element to enable automatic reload after code changes. Again, this is handy for test system but not for production, due to the overhead of detecting changes.

Locate the <Context> start element (around Line 19), and change it to <Context reloadable="true">.

***<Context reloadable="true">***

***......***

***......***

***</Context>***

**Step 3(d) (Optional) "conf\tomcat-users.xml"**

Enable the Tomcat's manager by adding the highlighted lines, inside the <tomcat-users> elements:

***<tomcat-users>***

***<role rolename="manager-gui"/>***

***<user username="manager" password="xxxx" roles="manager-gui"/>***

***</tomcat-users>***

This enables the manager GUI app for managing Tomcat server.

**STEP 4: Start Tomcat Server**

The Tomcat's executable programs and scripts are kept in the "bin" sub-directory of the Tomcat installed directory, e.g., "d:\myProject\tomcat\bin" (for Windows)

**Step 4(a) Start Server**

**For Windows**

Launch a CMD shell. Set the current directory to "<TOMCAT\_HOME>\bin", and run "startup.bat" as follows:

// Change the current directory to Tomcat's "bin"

// Assume that Tomcat is installed in "d:\myProject\tomcat"

d: // Change the current drive

cd \myProject\tomcat\bin // Change Directory to YOUR Tomcat's "bin" directory

**// Start Tomcat Server**

**startup**

**Step 4(b) Start a Client to Access the Server**

Start a browser (as HTTP client). Issue URL "http://localhost:9999" to access the Tomcat server's welcome page. The hostname "localhost" (with IP address of 127.0.0.1) is meant for local loop-back testing inside the same machine. For users on the other machines over the net, they have to use the server's IP address or DNS domain name or hostname in the format of "http://serverHostnameOrIPAddress:9999".

TomcatHomePage.png

Try issuing URL http://localhost:9999/examples to view the servlet and JSP examples. Try running some of the servlet examples.

(Optional) Try issuing URL http://localhost:9999/manager/html to run the Tomcat Web Manager. Enter the username and password configured earlier in tomcat-users.xml.

**Step 4(c) Shutdown Server**

For Windows

You can shutdown the tomcat server by either:

***Press Ctrl-C on the Tomcat console; OR***

***Run "<TOMCAT\_HOME>\bin\shutdown.bat" script. Open a new "cmd" and issue:***

***// Change the current directory to Tomcat's "bin"***

***d: // Change the current drive***

***cd \myProject\tomcat\bin // Change Directory to YOUR Tomcat's "bin" directory***

***// Shutdown the server***

***shutdown***

**STEP 5: Develop and Deploy a WebApp**

**Step 5(a) Create the Directory Structure for your WebApp**

First of all, choose a name for your webapp. Let's call it "hello". Goto Tomcat's "webapps" sub-directory. Create the following directory structure for you webapp "hello" (as illustrated):

Under Tomcat's "webapps", create your webapp root directory "hello" (i.e., "<TOMCAT\_HOME>\webapps\hello").

Under "hello", create a sub-directory "WEB-INF" (case sensitive, a "dash" not an underscore) (i.e., "<TOMCAT\_HOME>\webapps\hello\WEB-INF").

Under "WEB-INF", create a sub-sub-directory "classes" (case sensitive, plural) (i.e., "<TOMCAT\_HOME>\webapps\hello\WEB-INF\classes").

You need to keep your web resources (e.g., HTMLs, CSSs, images, scripts, servlets, JSPs) in the proper directories:

"hello": The is called the context root (or document base directory) of your webapp. You should keep all your HTML files and resources visible to the web users (e.g., HTMLs, CSSs, images, scripts, JSPs) under this context root.

"hello/WEB-INF": This directory, although under the context root, is not visible to the web users. This is where you keep your application's web descriptor file "web.xml".

"hello/WEB-INF/classes": This is where you keep all the Java classes such as servlet class-files.

You should RE-START your Tomcat server to pick up the hello webapp. Check the Tomcat's console to confirm that "hello" application has been properly deployed:

**You can issue the following URL to access the web application "hello":**

**http://localhost:9999/hello**

You should see the directory listing of the directory "<TOMCAT\_HOME>\webapps\hello", which shall be empty (provided you have enabled directory listing in web.xml earlier).

Step 5(b) Write a Welcome Page

Create the following HTML page and save as "HelloHome.html" in your application's root directory "hello".

***<html>***

***<head><title>My Home Page</title></head>***

***<body>***

***<h1>My Name is so and so. This is my HOME.</h1>***

***</body>***

***</html>***

**You can browse this page by issuing this URL:**

[**http://localhost:9999/hello/HelloHome.html**](http://localhost:9999/hello/HelloHome.html)