PACKAGING AND DEPLOYING A WEB APPLICATION INTO STANDALONE APACHE TOMCAT

You need to package your web application as a .war file to deploy it directly to Apache Tomcat. WAR stands for web archive. A different server might require a different format, for instance a glassfish server will need a .jar file whereas a websphere application server will require an .ear.

Installing and verifying a standalone Apache Tomcat

We will use a stand-alone tomcat for this example. Just download latest tomcat (.zip) version, unzip into a folder.

Now start tomcat server by running the startup.bat under the bin folder of tomcat installation.

You can verify if the server is running by typing localhost:<port> in the browser (e.g. http://localhost:8081).

You can view and change the default port for tomcat in the file server.xml under the conf folder of tomcat installation. Look for the element Connector configured for the HTTP protocol:

<Connector port="**8081**" protocol="HTTP/1.1"
connectionTimeout="20000"

redirectPort="8443" />

You can stop tomcat by running shutdown.bat under the bin folder of tomcat installation.

Packaging a war file to deploy into Apache Tomcat

We can package the application either from eclipse or using java jar command. We will see both.

In eclipse,

- right click your dynamic web project,
- select export > WAR file,
- specify a destination folder
- and click finish.

You can also optionally select optimize to a particular server already configured and also export source files along with class files.

Alternatively, you can create the war using java jar command as: Go to the workspace location of your project by right clicking the project, select Properties, and see the value for Location. Go inside the WebContent folder and run the below command from the command line:

jar -cvf myDynaServletApp.war myWebApp *

Note: Make sure that there is a folder WebContent/WEB-INF/classes and is your output folder with your class files. Else, right click your project in eclipse, go to Build Path > Configure Build Path and change the output folder in the Source tab to <project>/ WebContent/WEB-INF/classes. Browse and select the folder creating any folders if necessary than typing the location manually, as it is less error prone.

Deploying a war file into Apache Tomcat

Copy paste the war file into the webapps folder and start (startup.bat) if it is not running or restart (shutdown.bat, startup.bat) if already running. Server will unzip the war file and create a project root folder.

After you start the server you will see an additional project root folder under webapps.

To redeploy, you will have to shutdown, remove both war and folder, place new war and then startup again.

Format the servlet url and verify the servlet/jsp deployment

The url will be of the form host:port/<projectrootfolder>/servleturlpattern. Server is localhost and port it 8081 in my case. You can view and change the default port for tomcat in the file server.xml under the conf folder of tomcat installation. Url pattern is either configured in a web.xml file (web deployment descriptor) or through annotations.

I am using the same project that we have done for 'Hello World Dynamic Servlet' where we have configured the url patern through annotations:

```
@WebServlet("/HelloWorldServlet")
public class HelloWorldServlet extends HttpServlet {
```

…

Also, my project root folder created is DynaServletProject.

So my final url is:

http://localhost:8081/DynaServletProject/HelloWorldServlet

Execute the url in any browser and you will get the output as:

Hello World!

Source : http://javajee.com/packaging-and-deploying-a-web-application-into-standalone-apache-tomcat