**Apache Tomcat**

=====================================================================================

**Introduction:**

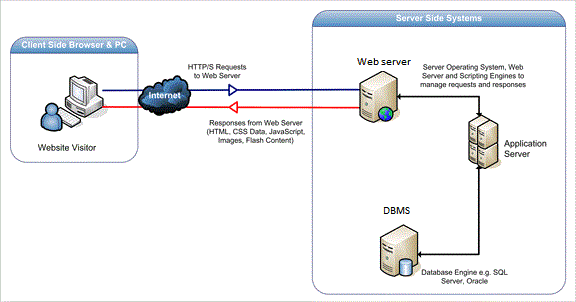
Tomcat, developed by Apache **(**[**www.apache.org**](http://www.apache.org)**).**Apache Tomcat is a web container which allows running servlet and JavaServer Pages (JSP) based web applications. Most of the modern Java web frameworks are based on servlets, e.g. JavaServer Faces, Struts, Spring.

Apache Tomcat also provides by default a HTTP connector on port 8080, i.e., Tomcat can also be used as HTTP server. But the performance of Tomcat is not as good as the performance of a designated web server, like the Apache HTTP server.

**Difference Between App Server and Web Server:**

|  |  |
| --- | --- |
| **Application Server** | **Web Server** |
| An application server serves business logic to application programs through any number of protocols, possibly including HTTP. It can be provided other protocol support such as RMI/RPC | A Web server exclusively handles HTTP/HTTPS requests. It serves content to the web using HTTP/HTTPS protocol. |
| Additionally App Server have components and features to support Application level services such as Connection Pooling, Object Pooling, Transaction Support, Messaging services etc. | Web server delegation model is fairly simple, when the request comes into the web server, it simply passes the request to the program best able to handle it(Server side program). It may not support transactions and database connection pooling. |
| Application server is more capable of dynamic behaviour than web server. We can also configure application server to work as a web server. Simply applic! Action server is a superset of web server. | Web Server is mostly designed to serve static content, though most Web Servers have plug-ins to support scripting languages like Perl, PHP, ASP, JSP etc. through which these servers can generate dynamic HTTP content. |

**Request Flow:**



**Different types of App Servers and Web Servers:**

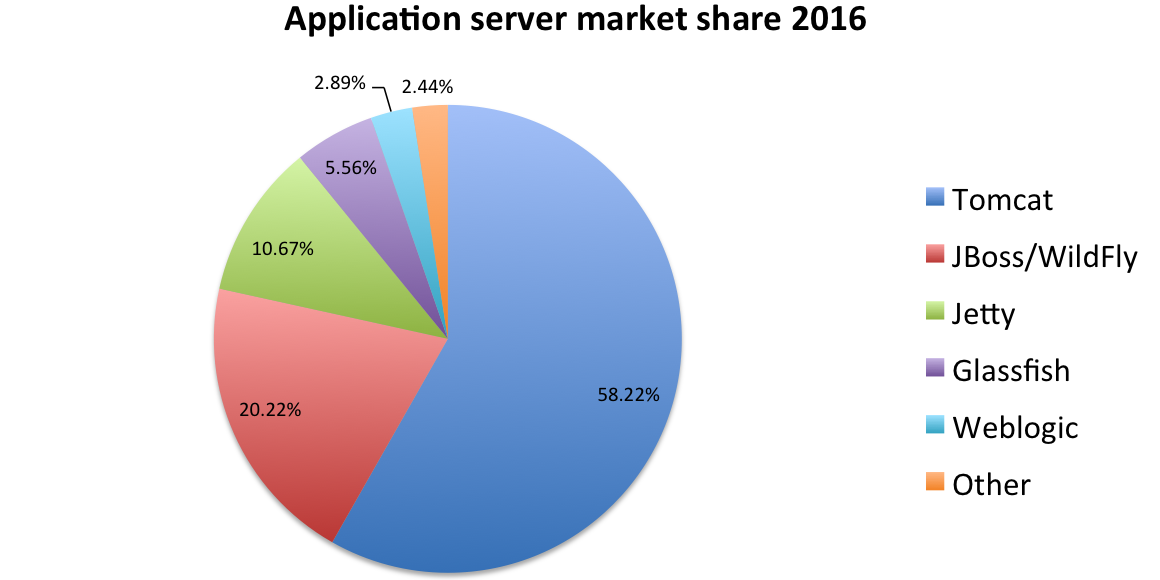
**Application Servers:**

|  |  |
| --- | --- |
| Application server | Vendor |
| WebSphere Application Server | IBM |
| WebLogic Server | Oracle |
| JBoss Application Server | RedHat |
| Tomcat | Apache |

**Web Servers:**

|  |  |
| --- | --- |
| Web server | Vendor |
| IBM HTTP Server | IBM |
| iPlanet | Oracle |
| Apache | Apache |
| Microsoft IIS Server | Microsoft |

**Application Servers occupancy in the leading Market:**



Tomcat installation base exceeded the 50% threshold for the second year in a row. The 58.22% share of the pie left no question about the winner. In addition to Tomcat, next four vendors having significant deployment base are:

●JBoss/WildFly installations, having 20.22% of the market share

●Jetty, with 10.67% of the market

●GlassFish, having another 5.56% of the pie

●Oracle WebLogic deployments with 2.44% of the installations

The “Other” category represented less than 2.5% of the installations. This consisted of Resin, Orion, OC4J, SAP NetWeaver and IBM WebSphere deployments, all detected in less than five deployments.

#### Tomcat Installation Step by Step:

#### Step 1 - Download the Latest Binary Distribution

You can download the latest version of Tomcat from the Apache [project site](https://tomcat.apache.org/). Click here to see the list of available versions.  Most Linux users will want to use the latest TAR package.

To download the package directly from the Linux command line, you'll use a command that looks something like this:

$ wget <http://mirror.23media.de/apache/tomcat/tomcat-8/v8.0.32/bin/apache-tomcat-8.0.32.tar.gz>

#### Step 2- Download JDK for Java environment

$wget --no-check-certificate -c --header "Cookie: oraclelicense=accept-securebackup-cookie" http://download.oracle.com/otn-pub/java/jdk/8u111-b14/jdk-8u111-linux-x64.tar.gz

#### Step 3: Extract the Tar balls

Create 2 different folders with the similar names of the tar files, like: Java and tomcat, for extract them separately.

**mkdir Java, mkdir Tomcat**

[root@HOST1 opt]# ls -ltr

drwxrwxrwx 3 root root 4096 Oct 26 12:46 Java

drwxrwxrwx 3 root root 4096 Oct 26 12:47 Tomcat

[root@HOST1 Java]# tar -xvf jdk-8u111-linux-x64.tar.gz

[root@HOST1 Tomcat]# tar -xvf apache-tomcat-8.0.32.tar.gz

#### Step 4 - Set the Required Environment Variables

If you haven't already done so during a different application's install process, you'll need to set the JAVA\_HOME environment variable in order for Tomcat to run.

Set the following environment variables and append them to the "/home/tomcat/.bash\_profile" so they are set for subsequent logins

[root@HOST1 ~]# export JAVA\_HOME=/opt/Java/jdk1.8.0\_77

[root@HOST1 ~]# export JRE\_HOME=/opt/Java/jdk1.8.0\_111/jre

[root@HOST1 ~]# export CATALINA\_HOME=/opt/Tomcat/apache-tomcat-8.0.32

[root@HOST1 ~]# export CATALINA\_BASE=$CATALINA\_HOME

[root@HOST1 ~]# $CATALINA\_HOME/bin/version.sh

#### Step 5 – Start Tomcat Server

If you followed all these steps correctly, you should be able to start Tomcat via its included startup script, startup.sh:  
$ $CATALINA\_HOME/bin/startup.sh  
Tomcat runs on port 8080 by default.  To check if your server is up and running correctly and port status.

$ ps -ef | grep java

$netstat -nlp | grep 8080

If this command returns the [Catalina](https://www.mulesoft.com/tomcat-catalina) process, Tomcat is up and running. You should now be able to access the Tomcat Welcome Page at http://<localhost>:8080/

[root@HOST1 ~]# curl -I http://localhost:8080

#### Step 6: Configuration Files

The main locations of configuration and log information are shown below.

Release Notes : $CATALINA\_HOME

Config : $CATALINA\_HOME/conf

Bin Directory : $CATALINA\_HOME/bin

Webapps : $CATALINA\_HOME/webapps

Logs : $CATALINA\_HOME/logs

#### Step 7: Enabling HTML Management Access

Edit the "$CATALINA\_HOME/conf/tomcat-users.xml" file, adding the following entries inside "tomcat-users" tag. Adjust the password as required.

<role rolename="manager-gui"/>

<role rolename="admin-gui"/>

<user username="tomcat" password="tomcat" roles="manager-gui,admin-gui"/>

Restart Tomcat for the configuration to take effect.

#### Step 8. Deploying Applications

The standard deployment format for web applications is a .war file. If you create a war application just put this application into the webapps folder. The next time tomcat starts it will unpack the war and make the application available.

Web applications may require external libraries. Typically, web application contain their own libraries but if you want to make certain libraries available for all applications you can put them into the folder "lib" and a subfolder below "lib". These libraries are then available for all web applications.

You can get a sample application WAR file to test with from "http://tomcat.apache.org/tomcat-8.0-doc/appdev/sample/".

Place the "sample.war" file in the "$CATALINA\_HOME/webapps" directory and Tomcat with automatically deploy it. You will see a "sample" directory appear.

You need to stop and start Tomcat for this to work, changes to effect.

**Important Commands & Points**

* **How to start tomcat services**

$CATALINA\_HOME/bin/startup.sh

* **How to stop tomcat services**

$CATALINA\_HOME/bin/shutdown.sh

* **Available log files**

catalina.out

manager log

localhost\_access\_log

**Various Directories description for Tomcat**

* **bin :** Contains startup/shutdown… scripts
* **conf:** Contains various configuration files including server.xml (Tomcat’s main configuration file) and web.xml that sets the default values for the various web applications deployed in Tomcat.
* **doc:** Contains miscellaneous documents regarding Tomcat.
* **lib:** Contains various jar files that are used by Tomcat. On UNIX any file in this directory is appended to Tomcat’s class path.
* **logs:** This is where Tomcat places it’s log files.
* **src:** The servlet APIs source files. Don’t get excited, though; these are only the empty interfaces and abstract classes that should be implemented by any servlet container.
* **Web apps:** Contains sample web applications.

#### Server.xml

server.xml is tomcat’s main configuration file. It serves two goals:  
1. Providing initial configuration for the Tomcat components.  
2. Specifying structure for Tomcat, meaning, letting Tomcat boot and build itself by instantiating components as specified in server.xml

**Connection to the Database:**

Add below code in the tomcat server.xml file. The code should be added in the **GlobalNamingResourceselement**. Also make sure that database driver is present in the tomcat lib directory, so in this case tomcat-jdbc.jar have to be present in the tomcat lib.

<Resource name="jdbc/MyDB"

global="jdbc/MyDB"

auth="Container"

type="javax.sql.DataSource"

driverClassName="com.mysql.jdbc.Driver"

url="jdbc:mysql://localhost:3306/UserDB"

username="pankaj"

password="pankaj123"

maxActive="100"

maxIdle="20"

minIdle="5"

maxWait="10000"/>

**Tomcat DataSource JNDI Resource Link Configuration – context.xml**

Add below code in the server context.xml file.

<ResourceLink name="jdbc/MyLocalDB"

global="jdbc/MyDB"

auth="Container"

type="javax.sql.DataSource" />