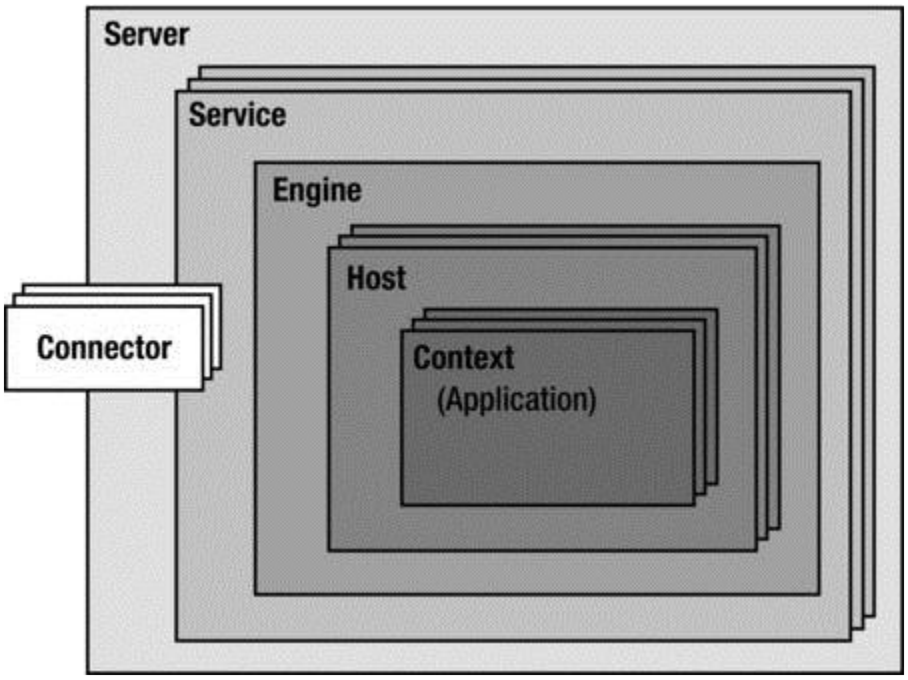
# Apache Tomcat

Apache Tomcat is an open-source web server and servlet container developed by the Apache Software Foundation (ASF). Tomcat implements several Java EE specifications including Java Servlet, JavaServer Pages (JSP), Java EL, and WebSocket, and provides a "pure Java" HTTP web server environment for Java code to run in

## Tomcat's Architecture:

Tomcat's architecture consists of a series of functional components that can be combined according to well-defined rules.



The structure of each server installation (via these functional components) is defined in the file server.xml, which is located in the /conf subdirectory of Tomcat's installation folder.

## Context:

A Context is the innermost element of a group of Tomcat components called containers, and it **represents a single web application**. Tomcat automatically instantiates and configures a standard context upon loading your application. As part of the configuration, Tomcat also processes the properties defined in the \WEB-INF\web.xml file of your application folder and makes them available to the application.

## Connector:

**A Connector handles communications with the client.** There are multiple connectors available with Tomcat e.g. HTTP connector for most of the HTTP traffic and AJP connector which implements the AJP protocol used when connecting Tomcat to another web server such as Apache HTTPD server.

The default configuration of Tomcat includes a connector to handle HTTP communication. By default, this connector waits for requests coming through port **8080**. This is why the URLs of our examples always start with http://localhost:8080/. Note that the requests for all applications go through a single instance of this connector. Each new request causes the instantiation of a new thread that remains alive within the connector for the duration of the request. Articles available on internet about Tomcat often refer to this connector as "**Coyote**".

The ***connectionTimeout*** attribute set to 20,000 means that a session is terminated after 5 hours, 33 minutes, and 20 seconds of inactivity, while ***redirectPort="8443"*** means that incoming requests that require Secure Socket Layer (SSL) transport are redirected to port 8443.

AJP connector lets Tomcat only handle dynamic web pages and lets a pure HTML server (e.g., the Apache Web Server) handle the requests for static pages. This maximizes the efficiency with which the requests are handled. You can probably comment out this connector as tomcat itself is pretty fast today OR simply if you don't plan on using a web server together with Tomcat.

## Host:

A Host is an association of a network name, e.g. www.yourdomain.com, to the Tomcat server. A host can contain any number of contexts (i.e. applications). You can define several hosts on the same server. For example, if you have registered the domain yourdomain.com, you can define host names such as w1.yourdomain.com and w2.yourdomain.com. Keep in mind that it will only be accessible from the Internet if a domain name server maps its name to the IP address of your computer.

The default configuration of Tomcat includes the host named **localhost**. The association between **localhost** and your computer is done instead by writing an entry in the file C:\Windows\System32\drivers\etc\hosts.

The Host attribute "***appBase***" defines the application directory within the Tomcat installation folder. Each application is then identified by its path within that directory. The only exception is the path ROOT, which is mapped to the empty string. The application base directory for localhost is webapps. This means that the application in directory "C:\Program Files\Apache Software Foundation\Tomcat 6.0\webapps\ROOT\" is identified by the empty string. Therefore, its URL is "http://localhost:8080/". For other applications, which reside in directories other than ROOT, as in "C:\Program Files\Apache Software Foundation\Tomcat 6.0\webapps\myapp\", the URL is like "http://localhost:8080/myapp/".

The attribute ***unpackWARs="true"*** means that if you drop a WAR file in the appBase directory, Tomcat will automatically expand it into a normal folder. If you set this attribute to false, the application will run directly from the WAR file. This obviously means a slower execution of the application, because Tomcat needs to unzip the WAR file at execution time.

The attribute ***autoDeploy="true"*** means that if you drop an application in the appBase directory while Tomcat is running, it will be deployed automatically.

## Listener:

A Listener is a Java object that, by implementing the org.apache.catalina.LifecycleListenerinterface, is able to respond to specific events.

- AprLifecycleListener : enables the Apache Portable Runtime (APR) library. This library provides OS level support to tomcat.
- JasperListener : enables Jasper, which is the JSP engine. This listener is what makes it possible to recompile JSP documents that have been updated.
- JreMemoryLeakPreventionListener : deal with different known situations that can cause memory leaks.
- GlobalResourcesLifecycleListener : is responsible for instantiating the managed beans associated with global Java Naming and Directory Interface (JNDI).
- ThreadLocalLeakPreventionListener : also deal with different known situations that can cause memory leaks.

## Global Naming Resources

The GlobalNamingResources element can only be defined inside the Server component. **It defines JNDI resources that are accessible throughout the server.** The only resource defined in the default server.xml is a user and password memory-based database defined via the file conf/tomcat-users.xml.

## Valve:

A Valve is an interceptor like element that, when inserted in a Container (Context, Host, or Engine), **intercepts all the incoming HTTP requests before they reach the application**. This gives you the ability to preprocess the requests directed to a particular application; to the applications running in a virtual host OR to all the applications running within an engine.

The RemoteAddrValve valve lets you selectively allow or block requests on the basis of their source IP address. It support two attributes – allow and block.

```
<Valve className="org.apache.catalina.valves.RemoteAddrValve" block="192\.168.*"/>
```

The RemoteHostValve valve operates like remote address filter but on client host names instead of client IP addresses.

```
<Valve className="org.apache.catalina.valves.RemoteHostValve" deny=".*badweb\.com"/>
```

The RequestDumperValve logs details of the incoming requests and therefore is useful for debugging purposes.

```
<Valve className="org.apache.catalina.valves.RequestDumperValve"/>
```