What is Tomcat?

1. It is an open-source Java servlet container that implements many Java Enterprise Specs such as the Websites API, Java-Server Pages and last but not least, the Java Servlet.

2. The complete name of Tomcat is "Apache Tomcat" it was developed in an open, participatory environment and released in 1998 for the very first time. It began as the reference implementation for the very first Java-Server Pages and the Java Servlet API.

3. It no longer works as the reference implementation for both of these technologies, but it is considered as the first choice among the users even after that.

4. It is still one of the most widely used java-sever due to several capabilities such as good extensibility, proven core engine, and well-test and durable. Here we used the term "servlet" many times, so what is java servlet; it is a kind of software that enables the webserver to handle the dynamic(java-based) content using the Http protocols.

So what exactly is Apache tomcat?

If you are a little familiar with the websites or have some basic knowledge about the websites, you must have heard about the HTTP protocol or may also know what actually are they. If you want to provide any web-services such as you want to provide a simple static content possibly by using HTML (or Hypertext Markup Language), or maybe you just want to send data from a server to point you, so you necessarily need a server and that server is HTTP(HyperText transfer protocol). So, as we all know that if anyone wants to make a simple, static website, he definitely requires an HTTP server, but if he wants to make website dynamic, he has to use servlet. We use the HTTP server if we want to send simple data. If we want to send dynamic data or to make our website dynamic, we need to use the servlet. Hence, we need an HTTP server and what else we need is a container where we will run or servlet, so when we combine the HTTP server and the servlet (or we can say servlet container), they both combine to become a single server know as tomcat server.

In simple words, we can say that The Apache Tomcat is actually a server and a servlet container.

What kind of server is Tomcat?

The Java ecosystem supports a wide variety of application servers, so let's have a little discussion on each of them and see where Tomcat fits in:

A servlet container is basically an implementation of the Java servlet specification, which is mainly used for the purpose of hosting Java servlets.

The Java enterprise application-server is an implementation of the Java specification.

A web- server is a kind of server designed to serve files using a local system such as Apache.

We can say that, at the center, the Tomcat is JSP (Java Server Pages) and Servlet. The JSP is one of the server-side programming technologies that enables the developers to create platform-independent dynamic content and also known as the server-side view rendering technology. A servlet is a java-based software component that helps in extending the capabilities of a server. However, it can also respond to several kinds of requests and generally implemented web server containers to host the web-applications on the webservers. As the developer's point of view, we just have to write the java server pages (or JSP) or the servlet and not required to worry about routing; the Tomcat will handle the routing.

The Tomcat also consists of the webserver known as the Coyote engine due to which it's possible to extend the capability of Tomcat to include several java enterprise specs, and including the Java Persistence API(JPA). The Tomcat also has an extended version known as the "TomEE" that contains more enterprise features.

Let's see how to install Tomcat. But before doing that, we are required to download the Tomcat. If you are a window ten user, you can use the following given steps for downloading and installing the Tomcat on your system:

How to Install Tomcat 9 on Ubuntu 18.04

Apache Tomcat is an open-source implementation of the Java Servlet, JavaServer Pages, Java Expression Language, and Java WebSocket technologies. It is one of the most widely adopted application and web servers in the world today. Tomcat is simple to use and has a robust ecosystem of add-ons.

This tutorial explains how to install and configure Tomcat 9 on Ubuntu 18.04. The same instructions apply for Ubuntu 16.04 and any Ubuntu-based distribution, including Linux Mint and Elementary OS.

Prerequisites

To be able to install packages on your Ubuntu system, you must be logged in as a user with sudo privileges .

Step 1: Install OpenJDK

Tomcat requires Java to be installed. We'll install OpenJDK , which is the default Java development and runtime in Ubuntu 18.04.

The installation of Java is pretty simple. Begin by updating the package index:

```
sudo apt update
```

Install the OpenJDK package by running:

```
sudo apt install default-jdk
```

Step 2: Create Tomcat User

For security purposes, Tomcat should not be run under the root user. We will create a new system user and group with home directory `/opt/tomcat` that will run the Tomcat service:

```
sudo useradd -r -m -U -d /opt/tomcat -s /bin/false tomcat
```

Step 3: Install Tomcat

We will download the latest binary release of Tomcat 9 from the Tomcat 9 downloads page .

At the time of writing, the latest version is `9.0.27`. Before continuing with the next step, you should check the download page for a new version. If there is a new version, copy the link to the Core `tar.gz` file, which is under the Binary Distributions section.

Start by download the Tomcat archive in the `/tmp` directory using the following `wget` command:

```
wget http://www-eu.apache.org/dist/tomcat/tomcat-9/v9.0.27/bin/apache-tomcat-9.0.27.tar.gz -P /tmp
```

Once the download is complete, extract the Tomcat archive and move it to the `/opt/tomcat` directory:

```
sudo tar xf /tmp/apache-tomcat-9*.tar.gz -C /opt/tomcat
```

To have more control over Tomcat versions and updates, create a symbolic link called `latest` that points to the Tomcat installation directory:

```
sudo ln -s /opt/tomcat/apache-tomcat-9.0.27 /opt/tomcat/latest
```

Later if you want to upgrade your Tomcat instance, simply unpack the newer version and change the symlink to point to the latest version.

As we mentioned in the previous section Tomcat will run under the `tomcat` user. This user needs to have access to the tomcat installation directory.

The following command changes the directory ownership to user and group tomcat:

```
sudo chown -RH tomcat: /opt/tomcat/latest
```

The scripts inside `bin` directory must have executable flag :

```
sudo sh -c 'chmod +x /opt/tomcat/latest/bin/*.sh'
```

Step 4: Create a systemd Unit File

To run Tomcat as a service you need to create a new unit file.

Open your text editor and create a file named `tomcat.service` in the `/etc/systemd/system/`:

```
sudo nano /etc/systemd/system/tomcat.service
```

Paste the following configuration:

/etc/systemd/system/tomcat.service

```
[Unit]
```

```
Description=Tomcat 9 servlet container
After=network.target

[Service]
Type=forking

User=tomcat
Group=tomcat

Environment="JAVA_HOME=/usr/lib/jvm/default-java"
Environment="JAVA_OPTS=-Djava.security.egd=file:///dev/urandom -Djava.awt.headless=true"

Environment="CATALINA_BASE=/opt/tomcat/latest"
Environment="CATALINA_HOME=/opt/tomcat/latest"
Environment="CATALINA_PID=/opt/tomcat/latest/temp/tomcat.pid"
Environment="CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"

ExecStart=/opt/tomcat/latest/bin/startup.sh
ExecStop=/opt/tomcat/latest/bin/shutdown.sh

[Install]
WantedBy=multi-user.target
```

Modify the value of `JAVA_HOME` if the path to your Java installation is different.

Save and close the file and notify systemd that we created a new unit file:

```
sudo systemctl daemon-reload
```

Start the Tomcat service by executing:

```
sudo systemctl start tomcat
```

Check the service status with the following command:

```
sudo systemctl status tomcat
```

```
* tomcat.service - Tomcat 9 servlet container
   Loaded: loaded (/etc/systemd/system/tomcat.service; disabled; vendor preset: enabled)
   Active: active (running) since Wed 2018-09-05 15:45:28 PDT; 20s ago
  Process: 1582 ExecStart=/opt/tomcat/latest/bin/startup.sh (code=exited, status=0/SUCCESS)
 Main PID: 1604 (java)
    Tasks: 47 (limit: 2319)
   CGroup: /system.slice/tomcat.service
```

If there are no errors enable the Tomcat service to be automatically started at boot time:

```
sudo systemctl enable tomcat
```

Step 5: Adjust the Firewall

If your server is protected by a firewall and you want to access Tomcat from the outside of your local network, you need to open port `8080`.

To allow traffic on port `8080` type the following command:

```
sudo ufw allow 8080/tcp
```

Usually when running a Tomcat application in a production environment you will have a load balancer or reverse proxy . It's a best practice to restrict access to port `8080` only to your internal network.

Step 6: Configure Tomcat Web Management Interface

Now that Tomcat is installed and running, the next step is to create a user with access the web management interface.

Tomcat users and roles are defined in the `tomcat-users.xml` file. This file is a template with comments and examples describing how to configure user or role.

```
sudo nano /opt/tomcat/latest/conf/tomcat-users.xml
```

To add a new user with access to the Tomcat web interface (manager-gui and admin-gui) we need to define the user in the `tomcat-users.xml` file, as shown below. Make sure you change the username and password to something more secure:

/opt/tomcat/latest/conf/tomcat-users.xml

```
<tomcat-users>
<!--
   Comments
-->
  <role rolename="admin-gui"/>
  <role rolename="manager-gui"/>
  <user username="admin" password="admin_password" roles="admin-gui,manager-gui"/>
</tomcat-users>
```

By default Tomcat web management interface is configured to restrict access to the Manager and Host Manager apps only from the localhost.

If you want to be able to access the web interface from a remote IP, you will have to remove these restrictions. This may have various security implications, and it is not recommended for production systems.

To enable access to the web interface from anywhere open the following two files and comment or remove the lines highlighted in yellow.

For the Manager app, open the following file:

```
sudo nano /opt/tomcat/latest/webapps/manager/META-INF/context.xml
```

For the Host Manager app, open the following file:

```
sudo nano /opt/tomcat/latest/webapps/host-manager/META-INF/context.xml
```

context.xml

```
<Context antiResourceLocking="false" privileged="true" >
<!--
  <Valve className="org.apache.catalina.valves.RemoteAddrValve"
         allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />
-->
</Context>
```

Another option is to allow access to the to the Manager and Host Manager apps only from a specific IP. Instead of commenting the blocks you can simply add your IP address to the list.

For example if your public IP is `45.45.45.45` you would make the following change:

context.xml

```
<Context antiResourceLocking="false" privileged="true" >
  <Valve className="org.apache.catalina.valves.RemoteAddrValve"
         allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1|45.45.45.45" />
</Context>
```

The list of allowed IP addresses is a list separated with vertical bar `|`. You can add single IP addresses or use a regular expressions.

Remember to restart the Tomcat service each time you edit Tomcat configuration files for changes to take effect:

Step 6: Test the Tomcat Installation

Open your browser and type: `http://<your_domain_or_IP_address>:8080`

Assuming the installation is successful, a screen similar to the following should appear:

Tomcat web application manager dashboard is available at `http://<your_domain_or_IP_address>:8080/manager/html`. From here, you can deploy, undeploy, start, stop, and reload your applications.

You can sign in with the user you have created in `Step 6`.

Tomcat virtual host manager dashboard is available at `http://<your_domain_or_IP_address>:8080/host-manager/html`. From here, you can create, delete and manage Tomcat virtual hosts.

Conclusion

You have successfully installed Tomcat 9 on your Ubuntu 18.04 system. You can now visit the official Apache Tomcat 9 Documentation and learn more about the Apache Tomcat features.

Advantages of Tomcat:

Some significant advantages of Tomcat are as follows:

- **It is open-source**
  It means anyone from anywhere can download, install, and use it free of cost, which makes it the first choice among the new developers and new users.
- **Incredibly Lightweight**
  It is actually a very light application, even with the JavaEE's certification. However, it provides all necessary and standard functionalities required to operate a server, which means it gives very fast load and redeploys as compared to its various alternatives.
  Yes, it is right that it does not offer so many features in case you want a number of features, it might be good for you, but if you want to have an easy and fast means in order to run your application, it is the best option for you.
- **Highly flexible**
  Due to its built-in customization options, extensive and lightweight nature, it offers high flexibility, a user can run it in any fashion he wants, and it will still work as fine without any issues. Since it is open-source, anyone who has knowledge can tweak it according to his requirements.
- **Stability**
  It is one of the most stable platforms available today to build on and using it to run our applications. It is incredibly stable because it runs independently of our Apache installation. In case if there is a big failure in Tomcat due to which it to stop working, the rest of our server would run just well.
- **It provides us an extra level of security**
  As the several organizations usually like to position their Tomcat's installation behind the protection of an extra firewall which can be accessible only from the Apache installation.
- **It is well documented**
  It has several excellent documentation available, including a vast range of

freely available online tutorials that can be downloaded or viewed directly online by the user, which makes it one of the best choices to fill the requirement of an application server in mostly every java web-application.Whether a user is looking for the installation instructions, startup settings, server configuration notes, all kind of information about the Tomcat is already available on the internet.

- **It is one of the most widely used application servers**
  According to an estimation, it holds almost 60 percent of the market share almost all java application server deployments, which makes it one of the most popular application servers used for java web-based applications. However, we cannot say that it implements all of the features required for a JavaEE application server; instead, it enables us to run Java EE application.
  Tomcat acts as a "webserver" or "servlet container." However, there is a plethora of terminology for anything.
- **It's mature**
  We take a look back in the past; we will find that it has existed for almost 20 years, which is quite a significant time, in which it gets mature over time passage. Since the Tomcat is open-source software, it's updated, and new releases come out nearly on a regular basis, and the open-source community maintains it. The maturity makes it one of the most extremely stable application servers for the development of software, applications, and deploying java applications. Since now, it is extremely a stable option that becomes more powerful with excellent community support.

Let discuss some disadvantages of Tomcat

- It is not as fast as the Apache if we are working with the static pages
- It has some issues like a memory leak
- It's way to handle the logs.
- Issues in the SSL installations
- Its user interface is inferior and basic