Remover Open JDK de Ubuntu

Java 8 install

$ sudo add-apt-repository ppa:webupd8team/java

$ sudo apt-get update

$ sudo apt-get install oracle-java8-installer

rahul@tecadmin:~$ java -version

java version "1.8.0\_131"

Java(TM) SE Runtime Environment (build 1.8.0\_131-b11)

Java HotSpot(TM) 64-Bit Server VM (build 25.131-b11, mixed mode)

Environment

$ sudo apt-get install oracle-java8-set-default

Now edit **/etc/environment** configuration file and add following entries to set **JAVA\_HOME**and **JRE\_HOME** environment variables.

JAVA\_HOME=/usr/lib/jvm/java-8-oracle

JRE\_HOME=/usr/lib/jvm/java-8-oracle/jre

JAVA DESDE CERO

Las instrucciones que figuran a continuación son para la instalación de la versión Java 8 Update 73 (8u73). Si va a instalar una versión distinta, cambie el número de la versión según corresponda cuando escriba los comandos en el terminal. **Ejemplo**: para Java 8u79, sustituya **8u73** por **8u79**. Tenga en cuenta que, como en el ejemplo anterior, el número de versión va a veces precedido por la letra u, y otras veces por un guion bajo, por ejemplo, jre1.8.0\_73. 

**Nota para el acceso raíz:** *Para instalar Java en una ubicación que afecte a todo el sistema, como /usr/local, debe conectarse como el usuario raíz para contar con todos los permisos necesarios. Si no tiene acceso de usuario root, instale Java en su directorio de inicio o en un subdirectorio para el que disponga de permiso de escritura.*

1. **Cambie al directorio en el que desee efectuar la instalación.** Escriba:  
   cd *nombre\_ruta\_acceso\_directorio*  
   Por ejemplo, para instalar el software en el directorio /usr/java, escriba:  
   cd /usr/java/
2. Mueva el archivo binario de almacenamiento .tar.gz al directorio actual.
3. **Desempaquete el tarball e instale Java**  
   tar zxvf jre-8u73-linux-x64.tar.gz  
     
   Los archivos de Java se instalan en un directorio denominado jre1.8.0\_73 en el directorio actual. En este ejemplo, se ha instalado en el directorio /usr/java/jre1.8.0\_73. Una vez finalizada la instalación se mostrará la palabra **Terminado**.
4. **Suprima el archivo**.tar.gz si desea ahorrar espacio en el disco.

Instalación por Defecto JRE/JDK

La opción más fácil para la instalación de Java es utilizando la versión empaquetada con Ubuntu. En concreto, esto instalará OpenJDK 8, la última versión recomendada.

En primer lugar, actualizaremos el índice de paquetes.

* sudo apt-get update

A continuación, instalaremos Java. Específicamente, este comando instalará el entorno de ejecución de Java (JRE).

* sudo apt-get install default-jre

Hay otra instalación por defecto de Java llamada JDK (Java Development Kit). El JDK por lo general sólo se necesita si va a compilar programas Java o si el software que va a utilizar Java lo requiere específicamente.

El JDK contiene el JRE, así que no hay inconvenientes si se instala el JDK en lugar de la JRE, excepto por el tamaño del archivo.

Puede instalar el JDK con el siguiente comando:

* sudo apt-get install default-jdk

Instalación del JDK de Oracle

Si desea instalar el JDK de Oracle, que es la versión oficial distribuida por Oracle, tendrá que seguir unos pasos más.

En primer lugar, agregue PPA de Oracle, a continuación, luego actualice el repositorio de paquetes.

* sudo add-apt-repository ppa:webupd8team/java
* sudo apt-get update

Luego, dependiendo de la versión que desee instalar, ejecutaremos uno de los siguientes comandos:

**Oracle JDK 8**

Esta es la última versión estable de Java por el momento, y la versión recomendada para instalar. Puede hacerlo utilizando el siguiente comando:

* sudo apt-get install oracle-java8-installer

**Oracle JDK 9**

Esta es una vista previa para desarrolladores y la liberación general está prevista para marzo de 2017. No se recomienda que utilice esta versión, porque todavía puede haber problemas de seguridad y errores. Hay más información acerca de Java 9 en la [página oficial de JDK 9](http://jdk.java.net/9/).

Para instalar JDK 9, utilice el siguiente comando:

* sudo apt-get install oracle-java9-installer

Gestionando Java

Puede haber varias instalaciones de Java en un servidor. Puede configurar cual será la versión por defecto para su uso mediante la línea de comandos usando update-alternatives, que gestiona cuales enlaces simbólicos se utilizan para diferentes comandos.

* sudo update-alternatives --config java

La salida será algo así como lo siguiente. En este caso, esto es lo que la salida mostrará con todas las versiones de Java instalada antes mencionados.

Output

There are 5 choices for the alternative java (providing /usr/bin/java).

Selection Path Priority Status

------------------------------------------------------------

\* 0 /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java 1081 auto mode

1 /usr/lib/jvm/java-6-oracle/jre/bin/java 1 manual mode

2 /usr/lib/jvm/java-7-oracle/jre/bin/java 2 manual mode

3 /usr/lib/jvm/java-8-openjdk-amd64/jre/bin/java 1081 manual mode

4 /usr/lib/jvm/java-8-oracle/jre/bin/java 3 manual mode

5 /usr/lib/jvm/java-9-oracle/bin/java 4 manual mode

Press <enter> to keep the current choice[\*], or type selection number:

Ahora puede elegir el número que desea usar como predeterminado. Esto también se puede hacer para otros comandos Java, como el compilador (javac), el generador de documentación (javadoc), la herramienta JAR de firma (jarsigner), y más. Se puede utilizar el siguiente comando, rellenando el comando que desea personalizar:

* sudo update-alternatives --config command

Definiendo la Variable de Entorno JAVA\_HOME

Muchos programas, como los servidores de Java, usan la variable de entorno JAVA\_HOME para determinar la ubicación de la instalación de Java. Para establecer esta variable de entorno, primero debe averiguar donde está instalado Java. Puede hacer esto mediante la ejecución del mismo comando que en el apartado anterior.

* sudo update-alternatives --config java

Copiar la ruta de la instalación preferida y luego abrir /etc/environment usando nano o su editor de texto favorito.

* sudo nano /etc/environment

Al final de este archivo, agregue la siguiente línea, asegurándose de sustituir la ruta resaltada con su ruta copiada.

/etc/environment

JAVA\_HOME="/usr/lib/jvm/java-8-oracle"

Guarde, salga del archivo y vuelva a cargarlo.

* source /etc/environment

Ahora puede probar si la variable de entorno se ha establecido mediante la ejecución del siguiente comando:

* echo $JAVA\_HOME

# **How To Manually Install Oracle Java on a Debian or Ubuntu VPS**

PostedApril 17, 2014 428.6kviews [JAVA](https://www.digitalocean.com/community/tags/java?type=tutorials) [UBUNTU](https://www.digitalocean.com/community/tags/ubuntu?type=tutorials) [DEBIAN](https://www.digitalocean.com/community/tags/debian?type=tutorials)

### Introduction

Java is a programming technology originally developed by Sun Microsystems and later acquired by Oracle. Oracle Java is a proprietary implementation for Java that is free to download and use for commercial use, but not to redistribute, therefore it is not included in a officially maintained repository.

There are many reasons why you would want to install Oracle Java over OpenJDK. In this tutorial, we will not discuss the differences between the above mentioned implementations.

## Assumptions

This tutorial assumes that you have an account with DigitalOcean, as well as a Droplet running Debian 7 or Ubuntu 12.04 or above. You will need root privileges (via sudo) to complete the tutorial.

You will need to know whether you are running a 32 bit or a 64 bit OS:

uname -m

* **x86\_64**: 64 bit kernel
* **i686**: 32 bit kernel

## Downloading Oracle Java JDK

Using your web browser, go to the [Oracle Java SE (Standard Edition) website](http://www.oracle.com/technetwork/java/javase/downloads/index.html) and decide which version you want to install:

* **JDK:** Java Development Kit. Includes a complete JRE plus tools for developing, debugging, and monitoring Java applications.
* **Server JRE:** Java Runtime Environment. For deploying Java applications on servers. Includes tools for JVM monitoring and tools commonly required for server applications.

In this tutorial we will be installing the JDK Java SE Development Kit 8 x64 bits. Accept the license and **copy the download link** into your clipboard. Remember to **choose the right tar.gz** (64 or 32 bits). Use wget to download the archive into your server:

wget --header "Cookie: oraclelicense=accept-securebackup-cookie" http://download.oracle.com/otn-pub/java/jdk/8u5-b13/jdk-8u5-linux-x64.tar.gz

Oracle does not allow downloads without accepting their license, therefore we needed to modify the header of our request. Alternatively, you can just download the compressed file using your browser and manually upload it using a SFTP/FTP client.

**Always get the latest version from Oracle's website** and modify the commands from this tutorial accordingly to your downloaded file.

## Installing Oracle JDK

In this section, you will need sudo privileges:

sudo su

The **/opt** directory is reserved for all the software and add-on packages that are not part of the default installation. Create a directory for your JDK installation:

mkdir /opt/jdk

and extract java into the **/opt/jdk** directory:

tar -zxf jdk-8u5-linux-x64.tar.gz -C /opt/jdk

Verify that the file has been extracted into the **/opt/jdk** directory.

ls /opt/jdk

## Setting Oracle JDK as the default JVM

In our case, the java executable is located under **/opt/jdk/jdk1.8.0\_05/bin/java** . To set it as the default JVM in your machine run:

update-alternatives --install /usr/bin/java java /opt/jdk/jdk1.8.0\_05/bin/java 100

and

update-alternatives --install /usr/bin/javac javac /opt/jdk/jdk1.8.0\_05/bin/javac 100

## Verify your installation

Verify that java has been successfully configured by running:

update-alternatives --display java

and

update-alternatives --display javac

The output should look like this:

java - auto mode

link currently points to /opt/jdk/jdk1.8.0\_05/bin/java

/opt/jdk/jdk1.8.0\_05/bin/java - priority 100

Current 'best' version is '/opt/jdk/jdk1.8.0\_05/bin/java'.

javac - auto mode

link currently points to /opt/jdk/jdk1.8.0\_05/bin/javac

/opt/jdk/jdk1.8.0\_05/bin/javac - priority 100

Current 'best' version is '/opt/jdk/jdk1.8.0\_05/bin/javac'.

Another easy way to check your installation is:

java -version

The output should look like this:

java version "1.8.0\_05"

Java(TM) SE Runtime Environment (build 1.8.0\_05-b13)

Java HotSpot(TM) 64-Bit Server VM (build 25.5-b02, mixed mode)

## (Optional) Updating Java

To update Java, simply download an updated version from Oracle's website and extract it under the **/opt/jdk** directory, then set it up as the default JVM with a higher priority number (in this case 110):

update-alternatives --install /usr/bin/java java /opt/jdk/jdk.new.version/bin/java 110

update-alternatives --install /usr/bin/javac javac /opt/jdk/jdk.new.version/bin/javac 110

You can keep the old version or delete it:

update-alternatives --remove java /opt/jdk/jdk.old.version/bin/java

update-alternatives --remove javac /opt/jdk/jdk.old.version/bin/javac

rm -rf /opt/jdk/jdk.old.version

The installation procedure documented above is confirmed to work on a Debian server, but can also be applied to an Ubuntu server. If you encounter any problem after following all the steps, please post a comment below.

# [How To Install GlassFish on Ubuntu 16.04](http://idroot.net/linux/install-glassfish-ubuntu-16-04/)

r00t July 14, 2016



In this tutorial we will show you how to install and configuration of GlassFish on your Ubuntu 16.04 server. For those of you who didn’t know, GlassFish is a popular app server that can run java based web applications for you. GlassFish 4.1 release supports the latest Java Platform: Enterprise Edition 7. It supports Enterprise JavaBeans, JPA, JavaServer Faces, JMS, RMI, JavaServer Pages, servlets, etc.

This article assumes you have at least basic knowledge of linux, know how to use the shell, and most importantly, you host your site on your own VPS. The installation is quite simple and assumes you are running in the root account, if not you may need to add ‘sudo’ to the commands to get root privileges. I will show you through the step by step installation GlassFish on a Ubuntu 16.04 (Xenial Xerus) server.

## Install GlassFish on Ubuntu 16.04

Step 1. First make sure that all your system packages are up-to-date by running these following apt-get commands in the terminal.



|  |  |
| --- | --- |
| 1  2 | sudo apt-get update  sudo apt-get upgrade |

Step 2. [Installing Java](http://idroot.net/tutorials/install-java-jre-jdk-ubuntu-16-04/) (JRE or JDK).

Choose the type of Java installation that you want with one the following:



|  |  |
| --- | --- |
| 1  2 | sudo apt-get install openjdk-7-jre  sudo apt-get install openjdk-7-jdk |

Another alternative Java install is with Oracle JRE and JDK. However, we would need to install additional repositories for a proper installation:



|  |  |
| --- | --- |
| 1  2 | sudo apt-get install python-software-properties  sudo add-apt-repository ppa:webupd8team/java |

Then, you will need to fully update the system with the following command and install it:



|  |  |
| --- | --- |
| 1  2 | sudo apt-get update  sudo apt-get install oracle-java8-installer |

Verify Installed Java Version.



|  |  |
| --- | --- |
| 1 | java -version |

Result:



|  |  |
| --- | --- |
| 1  2  3 | java version "1.8.0\_74"  Java(TM) SE Runtime Environment (build 1.8.0\_74-b02)  Java HotSpot(TM) 64-Bit Server VM (build 25.74-b02, mixed mode) |

Step 3. Install GlassFish.

First thing to do is to go to [GlassFish’s download page](https://glassfish.java.net/download.html) and download the latest stable version of GlassFish, At the moment of writing this article it is version 4.1.1:



|  |  |
| --- | --- |
| 1 | wget http://download.java.net/glassfish/4.1.1/release/glassfish-4.1.1.zip |

Once the Download completed successfully, extract the GlassFish archive to the document root directory on your server:



|  |  |
| --- | --- |
| 1 | unzip glassfish-4.1.1.zip |

Start the GlassFish server by using the following command as follows:



|  |  |
| --- | --- |
| 1 | glassfish4/bin/asadmin start-domain |

Step 4. Accessing GlassFish.

The default GlassFish Server’s port is 8080 and administration server’s port is 4848 with the administration user name as admin with no password. We can visit http://ip-address:8080/ to check the homepage of GlassFish Server and http://ip-address:4848/ to get the admin login page in our web browser and complete the required the steps to finish the installation. If you are using a firewall, please open port 4848 and 8080 to enable access to the control panel.

Congratulations! You have successfully [installed GlassFish](http://idroot.net/tutorials/how-to-install-glassfish-on-centos-7/). Thanks for using this tutorial for installing GlassFish in Ubuntu 16.04 LTS (Xenial Xerus) system. For additional help or useful information, we recommend you to check [the official GlassFish web site](https://glassfish.java.net/).

# Mark as Complete

# **How To Install Apache Tomcat 8 on Ubuntu 16.04**

PostedApril 27, 2016 284.2kviews [JAVA](https://www.digitalocean.com/community/tags/java?type=tutorials) [APPLICATIONS](https://www.digitalocean.com/community/tags/applications?type=tutorials) [UBUNTU](https://www.digitalocean.com/community/tags/ubuntu?type=tutorials) [UBUNTU 16.04](https://www.digitalocean.com/community/tags/ubuntu-16-04?type=tutorials)

### Introduction

Apache Tomcat is a web server and servlet container that is used to serve Java applications. Tomcat is an open source implementation of the Java Servlet and JavaServer Pages technologies, released by the Apache Software Foundation. This tutorial covers the basic installation and some configuration of the latest release of Tomcat 8 on your Ubuntu 16.04 server.

## Prerequisites

Before you begin with this guide, you should have a non-root user with sudo privileges set up on your server. You can learn how to do this by completing our [Ubuntu 16.04 initial server setup guide](https://www.digitalocean.com/community/tutorials/initial-server-setup-with-ubuntu-16-04).

## Step 1: Install Java

Tomcat requires Java to be installed on the server so that any Java web application code can be executed. We can satisfy that requirement by installing OpenJDK with apt-get.

First, update your apt-get package index:

* sudo apt-get update

Then install the Java Development Kit package with apt-get:

* sudo apt-get install default-jdk

Now that Java is installed, we can create a tomcat user, which will be used to run the Tomcat service.

## Step 2: Create Tomcat User

For security purposes, Tomcat should be run as an unprivileged user (i.e. not root). We will create a new user and group that will run the Tomcat service.

First, create a new tomcat group:

* sudo groupadd tomcat

Next, create a new tomcat user. We'll make this user a member of the tomcat group, with a home directory of /opt/tomcat (where we will install Tomcat), and with a shell of /bin/false (so nobody can log into the account):

* sudo useradd -s /bin/false -g tomcat -d /opt/tomcat tomcat

Now that our tomcat user is set up, let's download and install Tomcat.

## Step 3: Install Tomcat

The best way to install Tomcat 8 is to download the latest binary release then configure it manually.

Find the latest version of Tomcat 8 at the [Tomcat 8 Downloads page](http://tomcat.apache.org/download-80.cgi). At the time of writing, the latest version is **8.5.5**, but you should use a later stable version if it is available. Under the **Binary Distributions**section, then under the **Core** list, copy the link to the "tar.gz".

Next, change to the /tmp directory on your server. This is a good directory to download ephemeral items, like the Tomcat tarball, which we won't need after extracting the Tomcat contents:

* cd /tmp

Use curl to download the link that you copied from the Tomcat website:

* curl -O http://apache.mirrors.ionfish.org/tomcat/tomcat-8/v8.5.5/bin/apache-tomcat-8.5.5.tar.gz

We will install Tomcat to the /opt/tomcat directory. Create the directory, then extract the archive to it with these commands:

* sudo mkdir /opt/tomcat
* sudo tar xzvf apache-tomcat-8\*tar.gz -C /opt/tomcat --strip-components=1

Next, we can set up the proper user permissions for our installation.

## Step 4: Update Permissions

The tomcat user that we set up needs to have access to the Tomcat installation. We'll set that up now.

Change to the directory where we unpacked the Tomcat installation:

* cd /opt/tomcat

Give the tomcat group ownership over the entire installation directory:

* sudo chgrp -R tomcat /opt/tomcat

Next, give the tomcat group read access to the conf directory and all of its contents, and **execute** access to the directory itself:

* sudo chmod -R g+r conf
* sudo chmod g+x conf

Make the tomcat user the owner of the webapps, work, temp, and logs directories:

* sudo chown -R tomcat webapps/ work/ temp/ logs/

Now that the proper permissions are set up, we can create a systemd service file to manage the Tomcat process.

## Step 5: Create a systemd Service File

We want to be able to run Tomcat as a service, so we will set up systemd service file.

Tomcat needs to know where Java is installed. This path is commonly referred to as "JAVA\_HOME". The easiest way to look up that location is by running this command:

* sudo update-java-alternatives -l

Output

java-1.8.0-openjdk-amd64 1081 /usr/lib/jvm/java-1.8.0-openjdk-amd64

The correct JAVA\_HOME variable can be constructed by taking the output from the last column (highlighted in red) and appending /jre to the end. Given the example above, the correct JAVA\_HOME for this server would be:

JAVA\_HOME

/usr/lib/jvm/java-1.8.0-openjdk-amd64/jre

Your JAVA\_HOME may be different.

With this piece of information, we can create the systemd service file. Open a file called tomcat.servicein the /etc/systemd/system directory by typing:

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

Paste the following contents into your service file. Modify the value of JAVA\_HOME if necessary to match the value you found on your system. You may also want to modify the memory allocation settings that are specified in CATALINA\_OPTS:

/etc/systemd/system/tomcat.service

[Unit]

Description=Apache Tomcat Web Application Container

After=network.target

[Service]

Type=forking

Environment=JAVA\_HOME=/usr/lib/jvm/java-1.8.0-openjdk-amd64/jre

Environment=CATALINA\_PID=/opt/tomcat/temp/tomcat.pid

Environment=CATALINA\_HOME=/opt/tomcat

Environment=CATALINA\_BASE=/opt/tomcat

Environment='CATALINA\_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC'

Environment='JAVA\_OPTS=-Djava.awt.headless=true -Djava.security.egd=file:/dev/./urandom'

ExecStart=/opt/tomcat/bin/startup.sh

ExecStop=/opt/tomcat/bin/shutdown.sh

User=tomcat

Group=tomcat

UMask=0007

RestartSec=10

Restart=always

[Install]

WantedBy=multi-user.target

When you are finished, save and close the file.

Next, reload the systemd daemon so that it knows about our service file:

* sudo systemctl daemon-reload

Start the Tomcat service by typing:

* sudo systemctl start tomcat

Double check that it started without errors by typing:

* sudo systemctl status tomcat

## Step 6: Adjust the Firewall and Test the Tomcat Server

Now that the Tomcat service is started, we can test to make sure the default page is available.

Before we do that, we need to adjust the firewall to allow our requests to get to the service. If you followed the prerequisites, you will have a ufw firewall enabled currently.

Tomcat uses port 8080 to accept conventional requests. Allow traffic to that port by typing:

* sudo ufw allow 8080

With the firewall modified, you can access the default splash page by going to your domain or IP address followed by :8080 in a web browser:

Open in web browser

http://server\_domain\_or\_IP:8080

You will see the default Tomcat splash page, in addition to other information. However, if you click the links for the Manager App, for instance, you will be denied access. We can configure that access next.

If you were able to successfully accessed Tomcat, now is a good time to enable the service file so that Tomcat automatically starts at boot:

* sudo systemctl enable tomcat

## Step 7: Configure Tomcat Web Management Interface

In order to use the manager web app that comes with Tomcat, we must add a login to our Tomcat server. We will do this by editing the tomcat-users.xml file:

* sudo nano /opt/tomcat/conf/tomcat-users.xml

You will want to add a user who can access the manager-gui and admin-gui (web apps that come with Tomcat). You can do so by defining a user, similar to the example below, between the tomcat-users tags. Be sure to change the username and password to something secure:

tomcat-users.xml — Admin User

<tomcat-users . . .>

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

</tomcat-users>

Save and close the file when you are finished.

By default, newer versions of Tomcat restrict access to the Manager and Host Manager apps to connections coming from the server itself. Since we are installing on a remote machine, you will probably want to remove or alter this restriction. To change the IP address restrictions on these, open the appropriate context.xml files.

For the Manager app, type:

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

For the Host Manager app, type:

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

Inside, comment out the IP address restriction to allow connections from anywhere. Alternatively, if you would like to allow access only to connections coming from your own IP address, you can add your public IP address to the list:

context.xml files for Tomcat webapps

<Context antiResourceLocking="false" privileged="true" >

</Context>

Save and close the files when you are finished.

To put our changes into effect, restart the Tomcat service:

* sudo systemctl restart tomcat

## Step 8: Access the Web Interface

Now that we have create a user, we can access the web management interface again in a web browser. Once again, you can get to the correct interface by entering your server's domain name or IP address followed on port 8080 in your browser:

Open in web browser

http://server\_domain\_or\_IP:8080

The page you see should be the same one you were given when you tested earlier:

Let's take a look at the Manager App, accessible via the link or http://server\_domain\_or\_IP:8080/manager/html. You will need to enter the account credentials that you added to the tomcat-users.xml file. Afterwards, you should see a page that looks like this:

The Web Application Manager is used to manage your Java applications. You can Start, Stop, Reload, Deploy, and Undeploy here. You can also run some diagnostics on your apps (i.e. find memory leaks). Lastly, information about your server is available at the very bottom of this page.

Now let's take a look at the Host Manager, accessible via the link or http://server\_domain\_or\_IP:8080/host-manager/html/:

From the Virtual Host Manager page, you can add virtual hosts to serve your applications from.

## Conclusion

Your installation of Tomcat is complete! Your are now free to deploy your own Java web applications!

Currently, your Tomcat installation is functional, but entirely unencrypted. This means that all data, including sensitive items like passwords, are sent in plain text that can be intercepted and read by other parties on the internet. In order to prevent this from happening, it is strongly recommended that you encrypt your connections with SSL. You can find out how to encrypt your connections to Tomcat by following [this guide](https://www.digitalocean.com/community/tutorials/how-to-encrypt-tomcat-8-connections-with-apache-or-nginx-on-ubuntu-16-04).