**DevOps Setup**

**# Installing Jenkins**

|  |
| --- |
| **Jenkins Setup Guideline**  **1. Prerequisite**   Installing the Oracle JDK If you want to install the Oracle JDK, which is the official version distributed by Oracle, you will need to follow a few more steps. If you need Java 6 or 7, which are not available in the default Ubuntu 16.04 repositories (not recommended), this installation method is also available.  First, add Oracle's PPA, then update your package repository.   * sudo add-apt-repository ppa:webupd8team/java * sudo apt-get update   Then, depending on the version you want to install, execute one of the following commands: Oracle JDK 6 or 7 These are very old versions of Java which reached end of life in February 2013 and April 2015 respectively. It's not recommended to use them, but they might still be required for some programs.  To install JDK 6, use the following command:   * sudo apt-get install oracle-java6-installer   To install JDK 7, use the following command:   * sudo apt-get install oracle-java7-installer  Oracle JDK 8 This is the latest stable version of Java at time of writing, and the recommended version to install. You can do so using the following command:   * sudo apt-get install oracle-java8-installer  Oracle JDK 9 This is a developer preview and the general release is scheduled for March 2017. It's not recommended that you use this version because there may still be security issues and bugs. There is more information about Java 9 on the [official JDK 9 website](https://jdk9.java.net/).  To install JDK 9, use the following command:   * sudo apt-get install oracle-java9-installer  Managing Java There can be multiple Java installations on one server. You can configure which version is the default for use in the command line by using update-alternatives, which manages which symbolic links are used for different commands.   * sudo update-alternatives --config java   The output will look something like the following. In this case, this is what the output will look like with all Java versions mentioned above installed.  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:  You can now choose the number to use as a default. This can also be done for other Java commands, such as the compiler (javac), the documentation generator (javadoc), the JAR signing tool (jarsigner), and more. You can use the following command, filling in the command you want to customize.   * sudo update-alternatives --config command  Setting the JAVA\_HOME Environment Variable Many programs, such as Java servers, use the JAVA\_HOME environment variable to determine the Java installation location. To set this environment variable, we will first need to find out where Java is installed. You can do this by executing the same command as in the previous section:   * sudo update-alternatives --config java   Copy the path from your preferred installation and then open /etc/environment using nano or your favorite text editor.   * sudo nano /etc/environment   At the end of this file, add the following line, making sure to replace the highlighted path with your own copied path.  /etc/environment  JAVA\_HOME="/usr/lib/jvm/java-8-oracle"  Save and exit the file, and reload it.   * source /etc/environment   You can now test whether the environment variable has been set by executing the following command:   * echo $JAVA\_HOME   This will return the path you just set.  **2. Jenkins Setup**  **A. Download Jenkin latest war from https://updates.jenkins-ci.org/download/war.**  **Follow Step to setup Jenkins (USE WINSCP software to see the folder structure and do the following steps):-**    **a) After successfully download war file put that war file into tomcat/webapp folder.**  **b) Open tomcat/bin folder and run startup.bat file to run tomcat server.**  **c) After successfully start tomcat open browser and enter http://localhost:8080/jenkins.**  **d) Setup your jenkins.**  **e) Create credential for login by click on sign up link.**  **f) Login into jenkin application.**    **Follow Step to Configure jenkins.**  **Jenkins Mangement :-**  **a) After successfully login, click on Manage Jenkins Link.**  **b) Click Configure System.**  **c) Goto Jenkins Location section and enter your jenkins location in Jenkins URL input field like “http://<IP Address>:<Port>/jenkins/” and also enter your email id in to System Admin e-mail address input field.**    **d). Email configuration**  **i) Goto E-mail Notification section and enter infrasofttech-com- smtp.mail.na.collabserv.com in SMTP server input field.**  **ii) Click on checkbox name Use SMTP Authentication**  **iii) Enter email id into User Name input field.**  **iv) Enter password of your email id into password input field.**  **v) Click on SSL checkbox field.**  **vi) Enter 465 into SMTP Port input field.**  **vii) Enter UTF-8 into Charset input field.**    **Plugin Management :-**  **a) Download plugin as per requirement from Manage jenkins→Plugin Manager→Available.**  **b) For SVN download Subversion Plug-in from Available section in plugin Manger.**    **Project And Build Management :-**  **a) Click NewItem link to create new project for build.**  **b) Enter project name in and select FreeStyle project.**    **c). Source Code Management :-**  **i) Select subversion from Source Code Management section.**  **ii) Enter your repository url into Repository URL input field.**  **iii) Enter your repository credential into Credentials input filed.**    **d). Build Triggers :-**  **i) Checked Build Trigger Remotely checkbox.**  **ii) Enter any String in Authentication Token input field.**  **iii) Checked Build periodically checkbox for build your war periodically.**  **iv) Enter your build schedule in Schedule input field for periodic war.**  **v) Checked Poll SCM checkbox for get latest code from svn before make build.**  **vi) Enter your build schedule in Schedule input field.**  **e). Build Environment :-**  **Checked Add timestamps to the Console Output for show timstamp in build output.**  **f). Build :-**  **Execute Window Batch Command :-**  **i) Enter your project path that you have configured into jenkins into Command input field.**  **ii) Enter mvn clean install -DskipTests into Command input field for Maven build.**  **g). Post-build Actions :-**  **Click Add Post-Build Action dropdown to add any operation after build.**  **Select E-mail Notification from dropdown to add notification after build.**    **i) E-mail Notification:-**  **i) Add E-mail recipient into Recipients input field those you want to send email after build.**  **ii) Checked E-mail for every unstable build checkbox for after build failure E-mail notification must be go.**  **Also install Deploy tomcat container plugin to auto deploy tomcat to the required server.**  **Once installed In post-build Actions add Deploy tomcat container . Give war -> \*\*/\*.war context path-> webapps**  **Username & password should be the username and password of tomcat user with “manager-scripts” role. For this go to tomcat installation folder -> confs-> tomcat-users.xml and add a user with “manager-scripts” role. Provide this same details to Jenkins. URL-> Tomcat localhost URL with IP address and port eg : 172.21.0.33:8080** |

**#JavaMelody integration with Jenkins:**

* 1. **Install plugin “Monitoring” in Jenkins.**
  2. **Goto the url of Jenkins with monitoring**

**Eg:** [**http://localhost:9595/jenkins/monitoring**](http://localhost:9595/jenkins/monitoring)

**Or** [**http://localhost:9595/jenkins/monitoring/nodes**](http://localhost:9595/jenkins/monitoring/nodes)

**Jenkins handles the rest of it.**

**# Finally packaging your linux setup with vagrant for distribution of your Cookbook you just made:**

* 1. When this is all done we can package the vagrant in a “.box” file and use it .
  2. Syntax in CMD -> vagrant package --output mynewbox.box
  3. Its will take some time to package it all, finally you mynewbox.box file will be saved in **E**:/**Vagrant**/ **ProjDevOps/** mynewbox.box
  4. You can use this file to install a virtual server any time you want with command:  
     >vagrant box add mybasebox mynewbox.box  
     >vagrant init mybasebox  
     >vagrant up
  5. These set of commands with install linux with all the softwares pre installed.

**------------------------END-------------------------**