

<http://stackoverflow.com/questions/15042235/tomcat-7-manager-how-to-authenticate>

<https://janardhanareddy.wordpress.com/category/version-control-system/rational-clearcase/><https://tecadmin.net/install-git-2-x-on-centos-rhel-and-fedora/>

GIT install

<https://www.linuxhint.com/smartgit-git-client-ubuntu-centos/>

<http://stackoverflow.com/questions/11403407/git-asks-for-username-everytime-i-push>

|  |
| --- |
| $ git config credential.helper store  $ git push http://example.com/repo.git  Username: <type your username>  Password: <type your password>  Also I suggest you to read $ git help credentials |

egrep -l 'ERROR|FAIL' target/sure\*/\*.txt | xargs less

**Temporary**

**[vobadm@devops1 compile\_ex1]$ git push**

**fatal: You are not currently on a branch.**

**To push the history leading to the current (detached HEAD)**

**state now, use**

**git push origin HEAD:<name-of-remote-branch>**

**[vobadm@devops1 compile\_ex1]$ git push origin HEAD:master --force**

**Everything up-to-date**

**[vobadm@devops1 Compile]$ git checkout master**

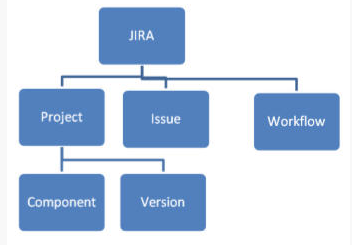
**Branch master set up to track remote branch master from origin.**

**Switched to a new branch 'master'**

**[vobadm@devops1 Compile]$ git status**

JIRA in its total is in light of 3 ideas.

**Issue:** Every task, bug, enhancement request; basically anything to be created and tracked via JIRA is considered an Issue.  
**Project:** A collection of issues  
**Workflow:** A workflow is simply the series of steps an issue goes through starting from creation to completion.



<https://phamthanhnhan14.wordpress.com/2016/10/10/install-jira-on-centos-7/>

<https://answers.atlassian.com/questions/12263911/centos-7-systemd-startup-scripts-for-jira-fisheye>

<https://it.megocollector.com/scripts/create-a-service-for-jira-on-centos-5-or-rhel-5/>

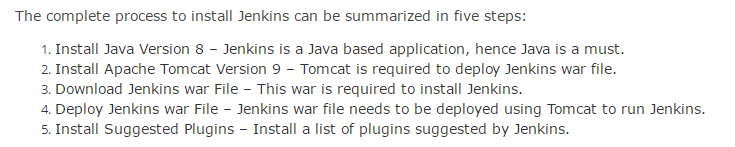
<http://www.techoism.com/steps-to-connecting-jira-with-mysql/>

<https://z3nsamlearn.zendesk.com/hc/en-us/articles/218748027-Installing-JIRA-7-x-on-Centos-7>

java -version

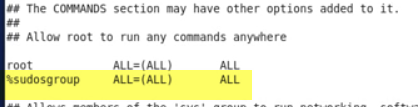
java version "1.8.0\_31"

|  |
| --- |
| **Step 1: Download Archive File**  Download latest version of java from its  **For 32 Bit –**  # cd /opt/  # wget --no-cookies --no-check-certificate --header "Cookie: gpw\_e24=http%3A%2F%2Fwww.oracle.com%2F; oraclelicense=accept-securebackup-cookie" "http://download.oracle.com/otn-pub/java/jdk/8u25-b17/jdk-8u25-linux-i586.tar.gz"  # tar jdk-8u25-linux-i586.tar.gz  **For 64 Bit –**  # cd /opt/  # wget --no-cookies --no-check-certificate --header "Cookie: gpw\_e24=http%3A%2F%2Fwww.oracle.com%2F; oraclelicense=accept-securebackup-cookie" "http://download.oracle.com/otn-pub/java/jdk/8u25-b17/jdk-8u25-linux-x64.tar.gz"  # tar xzf jdk-8u25-linux-x64.tar.gz  **Step 2: Install JAVA**  After extracting tar file, we just need to set up new version of java using alternatives. Use the following commands to do it.  # cd /opt/jdk1.8.0\_25/  # alternatives --install /usr/bin/java java /opt/jdk1.8.0\_25/bin/java 2  # alternatives --config java  There are 3 programs which provide 'java'.  Selection Command  -----------------------------------------------  \* 1 /opt/jdk1.8.0/bin/java  + 2 /opt/jdk1.7.0\_55/bin/java  3 /opt/jdk1.8.0\_25/bin/java  Enter to keep the current selection[+], or type selection number: 3  JAVA 8 has been installed on your Server. Now we have required to setup javac and jarring using following commands  # alternatives --install /usr/bin/jar jar /opt/jdk1.8.0\_25/bin/jar 2  # alternatives --install /usr/bin/javac javac /opt/jdk1.8.0\_25/bin/javac 2  # alternatives --install /usr/bin/javaws javaws /opt/jdk1.8.0\_25/bin/javaws 2  # alternatives --set jar /opt/jdk1.8.0\_25/bin/jar  # alternatives --set javac /opt/jdk1.8.0\_25/bin/javac  **Step 3: Check JAVA Version**  Use following command to check which version of java.  # java -version  java version "1.8.0\_25"  Java(TM) SE Runtime Environment (build 1.8.0\_25-b17)  Java HotSpot(TM) 64-Bit Server VM (build 25.25-b02, mixed mode)  **Step 4: Setup Environment Variables**  Some application’s are Java base. So follow below steps to set Java environment.   * Setup JAVA\_HOME Variable * # export JAVA\_HOME=/opt/jdk1.8.0\_25 * Setup JRE\_HOME Variable * # export JRE\_HOME=/opt/jdk1.8.0\_25/jre * Setup PATH Variable   # export PATH=$PATH:/opt/jdk1.8.0\_25/bin:/opt/jdk1.8.0\_25/jre/bin |



**Visudo**

Add group sudogroup-> Add users to it



**sudo yum install java-1.8.0-openjdk**

**sudo yum install wget**

**wget** <http://mirror.fibergrid.in/apache/tomcat/tomcat-9/v9.0.0.M10/bin/apache-tomcat-9.0.0.M10.tar.gz>

**tar xzf apache-tomat-9.0.0.M10.tar.gz**

**mv apache-tomcat-9.0.0.M10 tomcat9**

Our next step is to provide a username and password for Apache Tomcat

**gedit /home/edureka/tomcat9/conf/tomcat-users.xml**

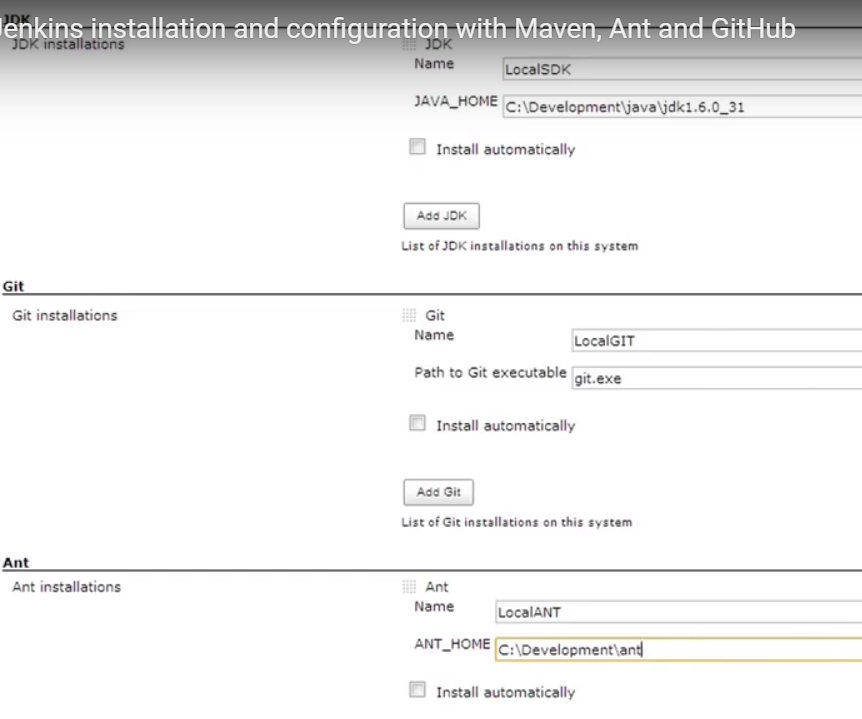
|  |
| --- |
| Now delete the content of the tomcat-users.xml file. Copy the below block and paste it in tomcat-users.xml file.  <?xml version="1.0" encoding="UTF-8?>  <tomcat-users>  <pre><role rolename="manager-gui"/>  <role rolename="manager-script"/>  <role rolename="manager-jmx"/>  <role rolename="manager-status"/>  <role rolename="admin-gui"/>  <role rolename="admin-script"/>  <user username="vobadm" password="111" roles="manager-gui,manager-<wbr />script,manager-jmx,manager-<wbr />status,admin-gui,admin-script"<wbr />/>  </tomcat-users> |

In the above image, as you can see that I have defined several roles and for all these roles I have given one single username and password i.e. edureka.

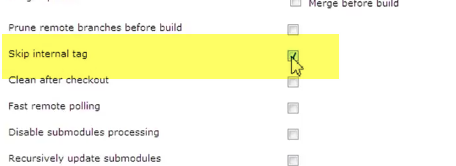
**Maven lifecycle:**

Validate, compile, test, unit testing using unit testing framework, package like jar, verify, install, deploy for production release.

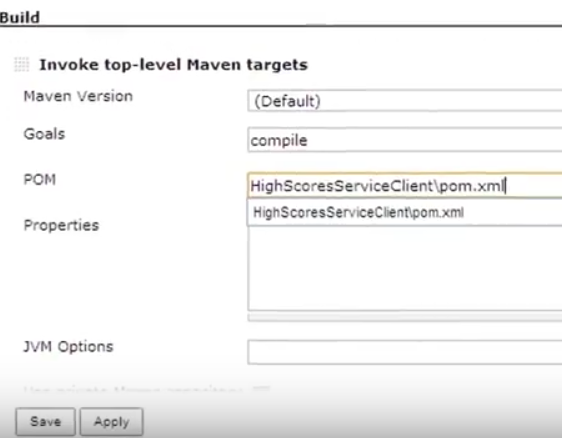


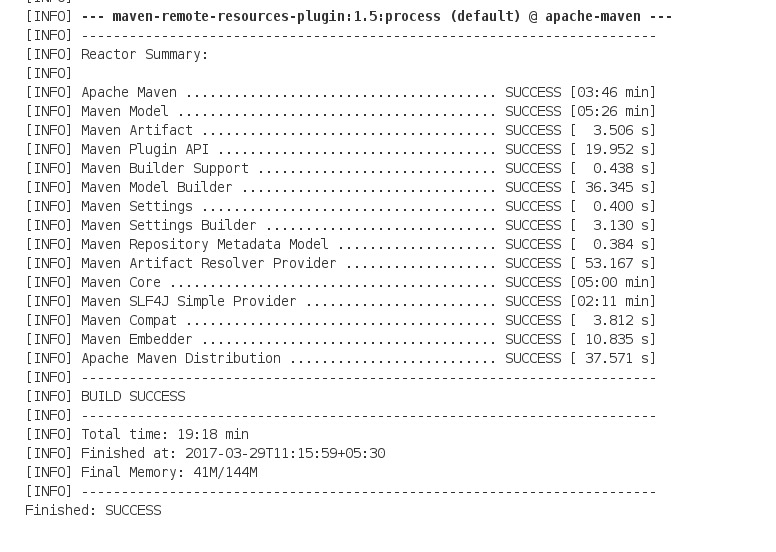


GIT…Jenkins create create foreach build in repo



Maven compile-🡪 Give path to pom.xml









Yes you can. Download the Plugin (\*.hpi File) and put it in the following directory:

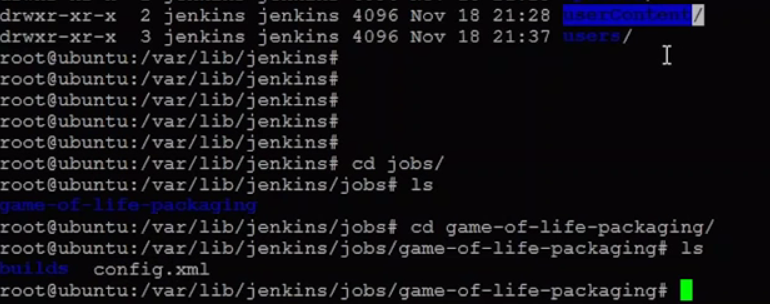
<jenkinsHome>/plugins/

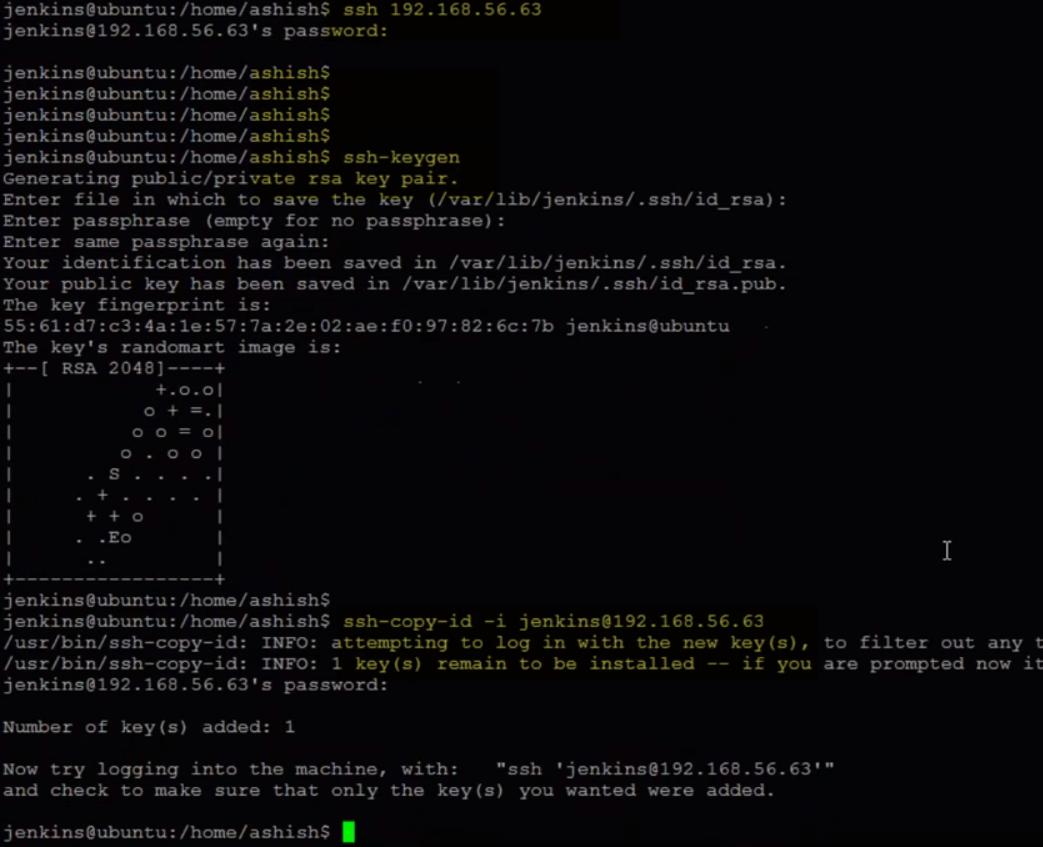
Afterward you will need to restart Jenkins.

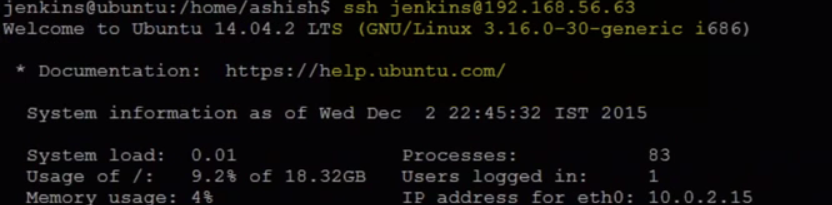
1. Download the plugin
2. Inside Jenkins: Manage Jenkins > Manage Plugins > There is a tab called 'Advance configuration' and in that page there is an option to upload a plugin (the extension of the file must be hpi)

Sometimes, when you download plugins you may get (.zip) files then just rename with (.hpi) and use the UI to install the plugin

<https://gist.github.com/micw/e80d739c6099078ce0f3#file-install_jenkins_plugin-sh>

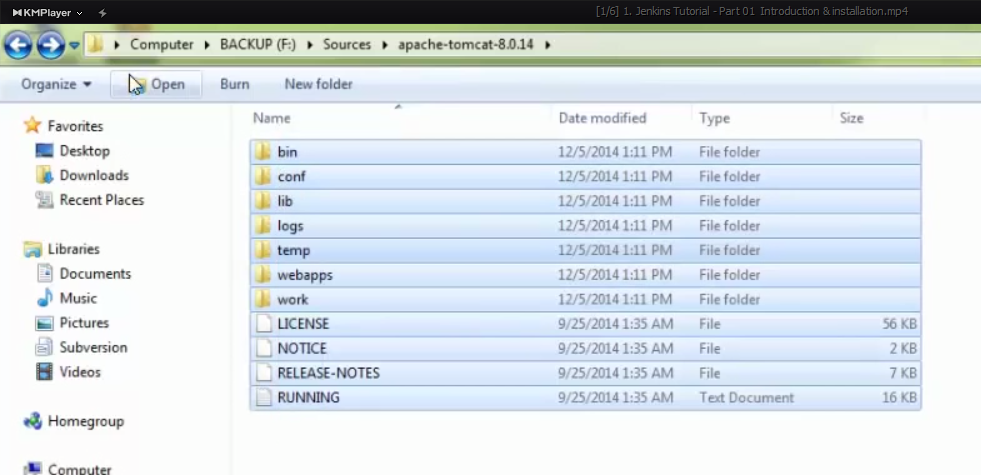








<http://127.0.0.1:8080/safeRestart>



Windows:

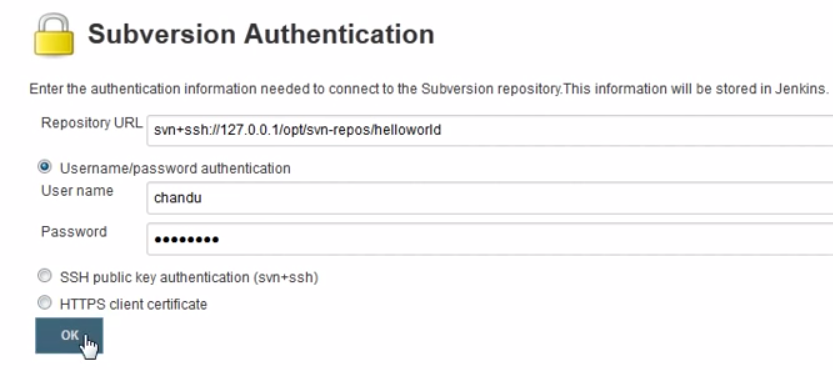
Extract tomcat

Put Jenkins in webapps

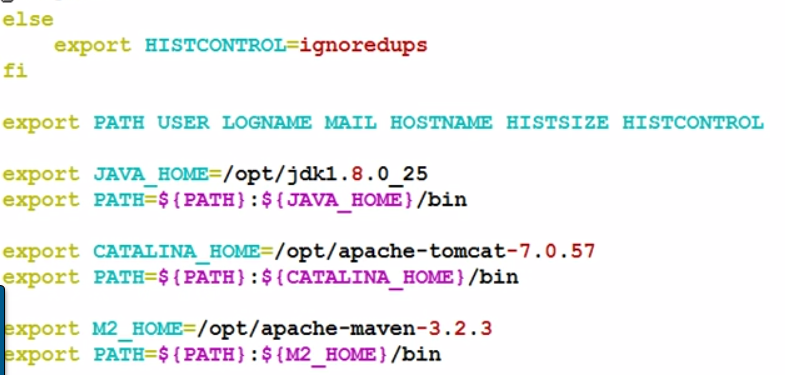
Set Catelina home,Jenkins\_home(for .jnekins), java\_home

$ nohup java -jar jenkins.war > $LOGFILE 2>&1

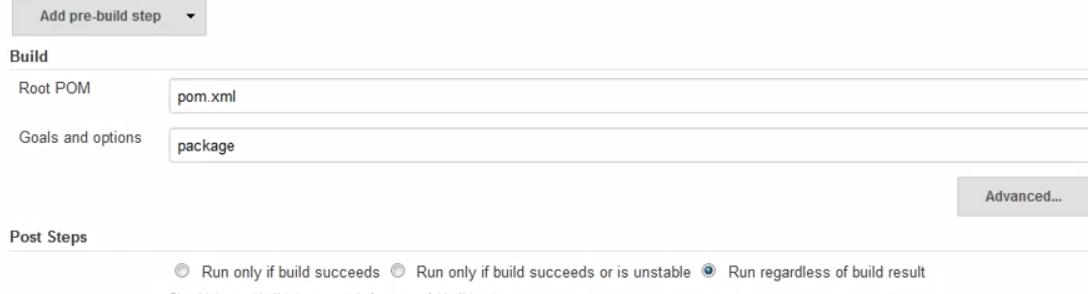
|  |  |
| --- | --- |
|  | Go to your %JENKINS\_HOME% and modify the jenkins.xml. Where you see --httpPort=8080change it to --httpPort=-1 --httpsPort=8080 you can make the ports anything you want of course, but in my testing (a while ago, it may have changed) if you don't keep --httpPort=<something> then Jenkins will always use 8080. So if you simply change --httpPort=8080 to --httpsPort=8080, port 8080 will still use http.  Also, if you want to use your own certificate, there are some instructions at the bottom of this page.  <http://wiki.jenkins-ci.org/display/JENKINS/Starting+and+Accessing+Jenkins> |



**Global Configuration**



MAVEN PROJECT:





**Issue: I use Java 1.7**

Tomcat 9

Solution: Upgrade java to 1.8

**sudo yum install java-1.8.0-openjdk**

[vobadm@centos\_BKP bin]$ ./shutdown.sh

Using CATALINA\_BASE: /home/vobadm/Desktop/tomcat

Using CATALINA\_HOME: /home/vobadm/Desktop/tomcat

Using CATALINA\_TMPDIR: /home/vobadm/Desktop/tomcat/temp

Using JRE\_HOME: /usr

Using CLASSPATH: /home/vobadm/Desktop/tomcat/bin/bootstrap.jar:/home/vobadm/Desktop/tomcat/bin/tomcat-juli.jar

Exception in thread "main" java.lang.UnsupportedClassVersionError: org/apache/catalina/startup/Bootstrap : Unsupported major.minor version 52.0

at java.lang.ClassLoader.defineClass1(Native Method)

at java.lang.ClassLoader.defineClass(ClassLoader.java:803)

at java.security.SecureClassLoader.defineClass(SecureClassLoader.java:142)

at java.net.URLClassLoader.defineClass(URLClassLoader.java:449)

at java.net.URLClassLoader.access$100(URLClassLoader.java:71)

at java.net.URLClassLoader$1.run(URLClassLoader.java:361)

at java.net.URLClassLoader$1.run(URLClassLoader.java:355)

at java.security.AccessController.doPrivileged(Native Method)

at java.net.URLClassLoader.findClass(URLClassLoader.java:354)