**Setting Up Automated Build System (Jenkins) Over Cloud**

Contents

[**Chapter 1. Prerequisite** 2](#_Toc524974843)

[**Hardware Requirements:** 3](#_Toc524974844)

[**Software Requirements:** 3](#_Toc524974845)

[1. Java: As Jenkins itself a java application, hence proper supporting java version should be installed before installing Jenkins. 3](#_Toc524974846)

[2. Git: This is essential as we need to get project content from GIT repository, windows system may come with preinstalled Git but it’s always better to install latest version of GIT. 4](#_Toc524974847)

[3. Download and Install Notepad++ to edit configuration files in projects and Sonar. 6](#_Toc524974848)

[4. Download 7-Zip from https://www.7-zip.org/download.html 6](#_Toc524974849)

[5. Download and Install Android 6](#_Toc524974850)

[6. Download and Install MS Visual Studio Latest Version. 7](#_Toc524974851)

[**Chapter 2: Setup Sonar** 7](#_Toc524974852)

[**1)** **Downloading sonarqube server and scanner** 8](#_Toc524974853)

[**2)** **Setting up environment variables** 8](#_Toc524974854)

[**3)** **Running Sonar as a service (Install and Run from the locations)** 8](#_Toc524974855)

[4) **Modify the port where Sonar is running, go to at following**. 9](#_Toc524974856)

[**Chapter 3: Self Signed Certificates and Windows Credential Manager** 9](#_Toc524974857)

[**Configure Self Signed SSL on Git for Windows** 9](#_Toc524974858)

[1. Using Internet Explorer. 9](#_Toc524974859)

[2. Exporting a certificate using the Chrome browser 10](#_Toc524974860)

[**Configuring Credential Storage with GIT** 11](#_Toc524974861)

[**Chapter 4: Setting Up Jenkins:** 12](#_Toc524974862)

[**Download and Installation** 12](#_Toc524974863)

[**Post-installation setup wizard** 12](#_Toc524974864)

[Unlocking Jenkins 12](#_Toc524974865)

[Customizing Jenkins with plugins 14](#_Toc524974866)

[Creating the first administrator user 14](#_Toc524974867)

[**Initial Configuration:** 14](#_Toc524974868)

[Creating Sample Project 17](#_Toc524974869)

[**Install and Configure Plug-Ins** 17](#_Toc524974870)

[From the web UI: 17](#_Toc524974871)

[Advanced Installation: 18](#_Toc524974872)

[List of recommended plugins 18](#_Toc524974873)

[**Configure Plugins: Global Tools Configuration:** 20](#_Toc524974874)

[**Configure global settings and paths**. 25](#_Toc524974875)

[**Chapter 5: Setting up Project in Jenkins** 29](#_Toc524974876)

[**Configure Project** 29](#_Toc524974877)

[Sample sonar-project.properties as following 34](#_Toc524974878)

[**Chapter 6: Setting up Dashboard**. 35](#_Toc524974879)

[**Chapter 7: Distributed Execution (Master and Slave Node)** 37](#_Toc524974880)

**[Configure Master Machine](#_Toc524974881)** [37](#_Toc524974881)

**[Configure Slave Machine](#_Toc524974882)** [39](#_Toc524974882)

# **Chapter 1. Prerequisite**

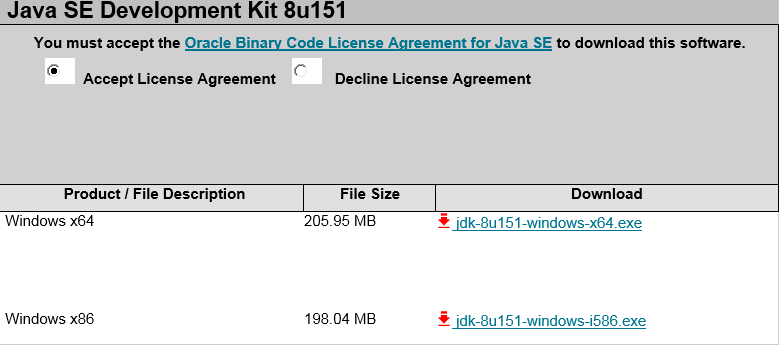
## **Hardware Requirements:**

1. A Physical/Virtual Machine
   1. Installed memory(RAM) 16.0 GB or 32.0 GB
   2. Either 32 or 64-bit Operating System.
      1. Now a day’s most of the server machines are coming with 64 bit configuration.
   3. Processor® CPU @2.80 GHz
      1. The more virtual CPU, more it will be beneficial to execute concurrent threads.
   4. Hard Disk : 200 GB
      1. The more hard disk space we will have the more it will be beneficial.
2. There may be 2 or more machine (virtual, physical) but more machines may be required as per project requirements.
   1. Setting up Slave machine is followed in coming sections.

## **Software Requirements:**

### **Java**: As Jenkins itself a java application, hence proper supporting java version should be installed before installing Jenkins.

* 1. Java 8 Latest Edition (I installed 8u\_151, 32 or 64 bit version or both together)
  2. Download java from <http://www.oracle.com/technetwork/java/javase/downloads/java-archive-javase8-2177648.html>



Select “Accept License Agreement” and Windows x86 and follow the installation instructions.

Set **JAVA\_HOME**:

Right click My Computer and select Properties.

1. On the Advanced tab, select Environment Variables, and then edit JAVA\_HOME to point to where the JDK software is located, for example, C:\Program Files (x86)\Java\%jdk%
   1. Here jdk = jdk1.8.0\_151.
2. Verify Java version at command prompt by running command **java –version.**
   1. **Command will run successfully it Java is installed properly and will show values as following.**



### **Git**: This is essential as we need to get project content from GIT repository, windows system may come with preinstalled Git but it’s always better to install latest version of GIT.

* 1. Download “GIT” from <https://gitforwindows.org/> , you will get latest version always. Or just go to <http://git-scm.com/download/win> and the download will start automatically. It will download latest 64 bit GIT version.
  2. Run Setup and follow the instructions, let default components be installed.
     1. Select Git Editor of your choice or keep the default one as shown in picture.



* 1. Either select Use Git from Git Bash only or “Use Git and optional Unix tools” optional UNIX tools may help in configuring having more command line options.
  2. Click next and select “Use open ssl Library”
  3. Click Next and configure “line ending conversions as mentioned below” 
  4. Click next and select default MinTTY terminal.
  5. Click Next and let default option be selected.
  6. Finally click install and setup will install selected components.
  7. Click Finish.
  8. Now Right Click My Computer -> Properties ->Advanced System Settings->Environment Variables->New
  9. Set New system variable GIT =C:\Program Files\Git;C:\Program Files\Git\cmd;C:\Program Files\Git\bin;
  10. Add %GIT% to Path variable.

### **Download and Install Notepad++ to edit configuration files in projects and Sonar**.

* 1. <https://notepad-plus-plus.org/download/v7.5.5.html>
  2. Either 32 bit or 64 bit.
  3. Let install with default options.

### **Download 7-Zip from** [**https://www.7-zip.org/download.html**](https://www.7-zip.org/download.html)

* 1. Just click and install.
  2. This utility is useful to extract compressed files and decompress the contents to send them over mail.

### Download and Install Android

* 1. Android Studio will install all necessary tools and Libraries required for any android project.
  2. Download latest android studio from <https://developer.android.com/studio/index.html>
  3. To install Android Studio on Windows, proceed as follows:

Launch the .exe file you downloaded.

Follow the setup wizard to install Android Studio and any necessary SDK tools   
 (Install all tools).

You may see short video at <https://developer.android.com/studio/install.html>

* 1. Setup “ANDROID\_HOME=C:\Users\dc-user\AppData\Local\Android\Sdk” environment variable, here dc-user is y username, in your case it may be different.

### Download and Install MS Visual Studio Latest Version.

* 1. This need only for the VS projects and No need to install on every Machine but on a specific machine as this take much space.
  2. All components need to be installed on a specific slave node as it takes about 40 GB of space.
  3. Download community edition from <https://www.visualstudio.com/downloads/>
     1. Community edition may expire after a month hence if you have full Visual Studio Professional with Key then use it.
  4. Follow the instruction mentioned at :
     1. <https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio>
  5. Add “MSBUILD\_HOME=C:\Program Files (x86)\Microsoft Visual Studio\2017\Community\MSBuild\15.0” in environment variable and add %MSBUILD\_HOME” in path variable.

# **Chapter 2: Setup Sonar**

**This includes setting up sonarqube and sonarqube scanner.**

## **Downloading sonarqube server and scanner**

* + - * 1. Download latest sonar version from <https://www.sonarqube.org/downloads/>
        2. You may either download latest or LTS (Long term supported) version.
        3. Once it is download then follow the instructions from <https://docs.sonarqube.org/display/SONAR/Get+Started+in+Two+Minutes>

1. Unzip - let's say in "C:\Apps", the SonarQube distribution once it's downloaded. ([Download Page](http://www.sonarsource.org/downloads/))

2. Start the SonarQube server:

|  |
| --- |
| C:\sonarqube\bin\windows-x86-xx\StartSonar.bat |
|  |
| <https://docs.sonarqube.org/display/SCAN/Analyzing+with+SonarQube+Scanner>C:\Users\e069511\Desktop\Capture.PNG |
|  |
| Either select Windows 64 bit or Any and extract under “C:\Apps”. I used Windows 64 bit ver. |

## **Setting up environment variables**

* + - * 1. Set SONARQUBE\_HOME=C:\Apps\sonarqube-7.0
        2. Set SONAR\_RUNNER\_HOME=C:\Apps\sonar-scanner-3.0.3.778
        3. Add %SONARQUBE\_HOME%\bin\windows-x86-32 to path variable.
        4. After updating path as per step Sonar can be run just by typing StartSonat.bat

## **Running Sonar as a service (Install and Run from the locations)**

* + - * 1. Run Sonar as a Windows Service  
           %SONARQUBE\_HOME%/bin/windows-x86-32/InstallNTService.bat
        2. Start Sonar service  
           %SONARQUBE\_HOME%/bin/windows-x86-32/StartNTService.bat
        3. Sonar service will run automatically and same need to configure with slave nodes as well.

## **Modify the port where Sonar is running, go to at following**.

**C:\Apps\sonarqube-6.7\conf (you will see sonar.properties and wrapper.conf)**

1. Change the sonar-properties files to have the port as 9010  
sonar.web.host: 0.0.0.0  
sonar.web.port: 9010  
sonar.web.context: /

2. Change the wrapper.conf and add following line  
wrapper.java.additional.3=-Djava.io.tmpdir=../../temp/

3. Run Sonar as a Windows Service  
%SONARQUBE\_HOME%/bin/windows-x86-32/InstallNTService.bat

4. Start Sonar service  
%SONARQUBE\_HOME%/bin/windows-x86-32/StartNTService.bat

5. Change the Sonar runner properties file  
sonar.host.url=http://localhost:9010

* + - 1. Other Sonar Related Configuration with Jenkins will be given in next chapters.

# **Chapter 3: Self Signed Certificates and Windows Credential Manager**

## **Configure Self Signed SSL on Git for Windows**

The self-signed certificate can be configured on Windows using the following steps:

### Using Internet Explorer.

* 1. open the URL
  2. Once the page has loaded click on the padlock next to the address bar
  3. Select the root certificate
  4. Now click on the "View Certificate" then "Details" tab and then the "Copy to file" button
  5. Select the "**Base64 encoded**" option and then export to a **.cer** file on your local file system
  6. Now configure Git to use the downloaded .cer file using the Git config command

|  |
| --- |
| **git config --global http.sslCAInfo C:/Users/e069511/certificate.cer** |

### Exporting a certificate using the Chrome browser

|  |  |
| --- | --- |
|  | 1. Connect to the website using SSL ([https://whatever](https://whatever/)) 2. Since Chrome version 56, you do the following: go to the Three Dots Menu -> More Tools -> Developer Tools, then click on the Security Tab. This will give you a **Security Overview** with a **View certificate** button. 3. Click on the **View certificate** button.   A modal window will open. It has two panes. The top one shows the trust hierarchy of the sites certificate (the last one listed), the intermediate certificate(s), and the root certificate (the topmost one).  The second, larger pane, shows the details of one of the certificates.  There may be zero or more intermediate certificates.  Note that the root certificate has a gold-bordered icon. The others have a blue border. See the screen shot below.   1. To export a certificate:    1. First click on the certificate's icon in the trust hierarchy.    2. The certificate will be shown in the main part of the modal.    3. Click on the certificate's large icon in the main part of the modal. Export the certificate to your desktop.   [enter image description here](https://i.stack.imgur.com/Hdmot.png) |

Use Unix2Dos for appending certificate information in existing certificate.

## **Configuring Credential Storage with GIT**

We are using Standard Windows Credential Manager with GIT to store credentials called as wincred.

Run Following command git config --global credential.helper wincred

And then do checkout/update from repository, a window will popup asking GIT credentials (username and password). Then it will store them permanently until changed with GIT itself.

Changes at Windows credential Manager can be made manually as well. Type “Credential Manager” inside Run

Click on “Credential Manager”, you may see options to Add, Edit and Delete under section “Generic Credentials”.

# **Chapter 4: Setting Up Jenkins:**

## **Download and Installation**

* 1. Download latest Jenkins windows installer from <https://jenkins.io/download/>

Either LTS or Weekly version

Windows

To install from the website, using the installer:

* [Download the latest package](http://mirrors.jenkins.io/windows/latest) (Jenkins.msi)
* Open the package and follow the instructions
  1. Check installation instructions at <https://jenkins.io/doc/book/installing/>

After Downloading Jenkins zip file, decompress it and install as following

WAR file

The Web application Archive (WAR) file version of Jenkins can be installed on any operating system or platform that supports Java.

**To download and run the WAR file version of Jenkins:**

1. Download the [latest stable Jenkins WAR file](http://mirrors.jenkins.io/war-stable/latest/jenkins.war) to an appropriate directory on your machine.
2. Open up a terminal/command prompt window to the download directory.
3. Run the command java -jar jenkins.war.
4. Browse to http://localhost:8080 and wait until the **Unlock Jenkins** page appears.
5. Continue on with the [Post-installation setup wizard](https://jenkins.io/doc/book/installing/#setup-wizard) below.

After Downloading and installing, we need to do post installation setup wizard.

<https://jenkins.io/doc/book/installing/#setup-wizard>

## **Post-installation setup wizard**

After downloading, installing and running Jenkins using one of the procedures above, the post-installation setup wizard begins.

This setup wizard takes you through are a few quick "one-off" steps to unlock Jenkins, customize it with plugins and create the first administrator user through which you can continue accessing Jenkins.

### Unlocking Jenkins

When you first access a new Jenkins instance, you are asked to unlock it using an automatically-generated password.

In windows it will be at following location: C:\Program Files (x86)\Jenkins\secrets

1. Browse to http://localhost:8080 (or whichever port you configured for Jenkins when installing it) and wait until the **Unlock Jenkins** page appears.



1. From the Jenkins console log output, copy the automatically-generated alphanumeric password (between the 2 sets of asterisks).



1. On the **Unlock Jenkins** page, paste this password into the **Administrator password** field and click **Continue**.  
   **Notes:**
   * If you ran Jenkins in Docker in detached mode, you can access the Jenkins console log from the Docker logs ([above](https://jenkins.io/doc/book/installing/#accessing-the-jenkins-console-log-through-docker-logs)).
   * The Jenkins console log indicates the location (in the Jenkins home directory) where this password can also be obtained. This password must be entered in the setup wizard on new Jenkins installations before you can access Jenkins’s main UI. This password also serves as the default administrator account’s password (with username "admin") if you happen to skip the subsequent user-creation step in the setup wizard.

### Customizing Jenkins with plugins

After [unlocking Jenkins](https://jenkins.io/doc/book/installing/#unlocking-jenkins), the **Customize Jenkins** page appears. Here you can install any number of useful plugins as part of your initial setup.

Click one of the two options shown:

* **Install suggested plugins** - to install the recommended set of plugins, which are based on most common use cases.
* **Select plugins to install** - to choose which set of plugins to initially install. When you first access the plugin selection page, the suggested plugins are selected by default.

|  |  |
| --- | --- |
|  | If you are not sure what plugins you need, choose **Install suggested plugins**. You can install (or remove) additional Jenkins plugins at a later point in time via the [**Manage Jenkins**](https://jenkins.io/doc/book/managing) > [**Manage Plugins**](https://jenkins.io/doc/book/managing/plugins/) page in Jenkins. |

The setup wizard shows the progression of Jenkins being configured and your chosen set of Jenkins plugins being installed. This process may take a few minutes.

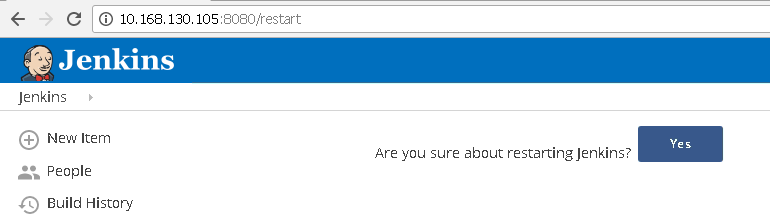
### Creating the first administrator user

Finally, after [customizing Jenkins with plugins](https://jenkins.io/doc/book/installing/#customizing-jenkins-with-plugins), Jenkins asks you to create your first administrator user.

1. When the **Create First Admin User** page appears, specify the details for your administrator user in the respective fields and click **Save and Finish**.
2. When the **Jenkins is ready** page appears, click **Start using Jenkins**.  
   **Notes:**
   * This page may indicate **Jenkins is almost ready!** Instead and if so, click **Restart**.
   * If the page does not automatically refresh after a minute, use your web browser to refresh the page manually.
3. If required, log in to Jenkins with the credentials of the user you just created and you are ready to start using Jenkins!

|  |  |
| --- | --- |
|  | From this point on, the Jenkins UI is only accessible by providing valid username and password credentials. |

## **Initial Configuration:**

1. Jenkins installs as a service. Hence it always restart as soon as system reboots for any update. No need to start manually. Just need to access with IP: Port combination with login credentials.
2. Is somehow Jenkins is stopped then same may be restarted by typing following in browser
   1. Type in browser IP:port/restart, as shown in following picture
   2. Or alternatively start services.msc (start->Run) and restart Jenkins as following.

->After creating first Administrator Account, any number of users can be added to Jenkins with credentials as following.

a) Go to Manage Jenkins -> Manage Users

b) Click on Manage Users and Then Create User

1. Fill the details and click on create user. As soon as user is created, then same credentials can be used to login with Jenkins.



## Creating Sample Project

* + - * 1. **Go to Jenkins home page and click on New Item**
        2. Select Freestyle Project and Enter Project Name
        3. Click Ok and a Project Configuration Wizard will open.

Refer Chapter 5 for complete project configuration steps.

## **Install and Configure Plug-Ins**

### From the web UI:

The simplest and most common way of installing plugins is through the

**Manage Jenkins** > **Manage Plugins** view, available to administrators of a Jenkins environment.

Under the **Available** tab, plugins available for download from the configured Update Center can be searched and considered:



Click on Manage Plug-In and search for the plugin you need to install. 

Select the Plugin and choose either Install without restart or Download now and install after restart.

### Advanced Installation:

Click on Advanced Tab, You may upload any downloaded .hpi (plugin file) and upload annually via below mentioned method.



All plugins can be found and browse from <https://plugins.jenkins.io/>

Now Following Plugins need to be installed by above mentioned methods

### List of recommended plugins

1. Git Plugin <https://wiki.jenkins.io/display/JENKINS/Git+Plugin>

Plugin Id: git

Dependencies:

[workflow-scm-step](https://plugins.jenkins.io/workflow-scm-step) (version:1.14.2)   
[credentials](https://plugins.jenkins.io/credentials) (version:2.1.14)   
[**git-client**](https://plugins.jenkins.io/git-client)**(version:2.5.0)**   
[mailer](https://plugins.jenkins.io/mailer) (version:1.18)   
[matrix-project](https://plugins.jenkins.io/matrix-project) (version:1.7.1)   
[parameterized-trigger](https://plugins.jenkins.io/parameterized-trigger) (version:2.33, optional)   
[promoted-builds](https://plugins.jenkins.io/promoted-builds) (version:2.27, optional)  
[**scm-api**](https://plugins.jenkins.io/scm-api)**(version:2.2.0)**   
[**ssh-credentials**](https://plugins.jenkins.io/ssh-credentials)**(version:1.13)**   
[**structs**](https://plugins.jenkins.io/structs)**(version:1.10)**   
[token-macro](https://plugins.jenkins.io/token-macro) (version:1.12.1, optional)

1. SonarQube plugin

Plugin id: sonar

Require dependencies:

[Maven-plugin](https://plugins.jenkins.io/maven-plugin) (version: 2.14, optional)   
[workflow-cps](https://plugins.jenkins.io/workflow-cps) (version: 2.25, optional)   
[configuration slicing](https://plugins.jenkins.io/configurationslicing) (version: 1.40, optional)   
[jQuery](https://plugins.jenkins.io/jquery) (version: 1.11.2-0)

1. SonarQube Scanner

Require dependencies:

[JQuery plugin v.1.11.2-0 (required)](https://plugins.jenkins.io/jquery)

[Maven Integration plugin v.2.14 (optional)](https://plugins.jenkins.io/maven-plugin)

[Pipeline: Groovy v.2.25 (optional)](https://plugins.jenkins.io/workflow-cps)

[Configuration slicing plugin v.1.40 (optional)](https://plugins.jenkins.io/configurationslicing)

1. Fire Line: is a static code analysis program to look for bugs in Java Code.
2. Sonar Quality Gates: If done setup, it fails the build whenever Quality Gate criteria in sonar analysis aren’t meet.
3. Junit Plugin: Allows Junit-format test results to be published
   1. Require Dependencies:
   2. [workflow-api](https://plugins.jenkins.io/workflow-api) (version:2.22)   
      [workflow-step-api](https://plugins.jenkins.io/workflow-step-api) (version:2.12)   
      [script-security](https://plugins.jenkins.io/script-security) (version:1.30)   
      [structs](https://plugins.jenkins.io/structs) (version:1.7)
4. Gradle Plugin, used to invoke a Gradle build script as the main build step.

Configured under Manage Jenkins 🡪 Global Tool Configuration

Dependency: [structs](https://plugins.jenkins.io/structs) (version: 1.3)

1. Apache Maven comes by default with standard Jenkins installation
2. MSBuild Plugin: allows MSBuild to build .NET Projects

<https://wiki.jenkins.io/display/JENKINS/MSBuild+Plugin>

Dependency: Structs (1.3)

1. Notification: Default plugin comes with Jenkins but another plugin also available in market from Tikal Knowledge which allows sending Job Status notifications in JSON and XML formats
2. Distributed Build: A Slave computer can be configured to offload build projects from master computer. It’s an inbuilt feature and can be done with manage Jenkins, where it asks the details of node slave machine.
   1. Workspace Cleanup plugin: Deletes workspace after a build is finished
   2. Windows slave plugin : connects to windows machine and start slave agents on them

## **Configure Plugins: Global Tools Configuration:**

Go to Jenkins 🡪 Manage Jenkins 🡪 Global Tool Configuration



1. Keep Maven Configuration as it is. “Default Maven Settings”



1. Configure JDK as shown in figure
2. Configure GIT as following.
3. Configure Gradle as following.



1. Configure MSBuild as following.
   1. Here installation path may also exist on Slave nodes, May not necessary all on master.
2. Configure SoanrQube Scanner for MSBuild. Select latest version from drop down list.
3. Configure SonarQube Scanner as following. Select Latest version from drop down
4. Configure ANT as following. Select latest version from dropdown list.
5. Configure MAVEN as following. Select latest version from dropdown list.

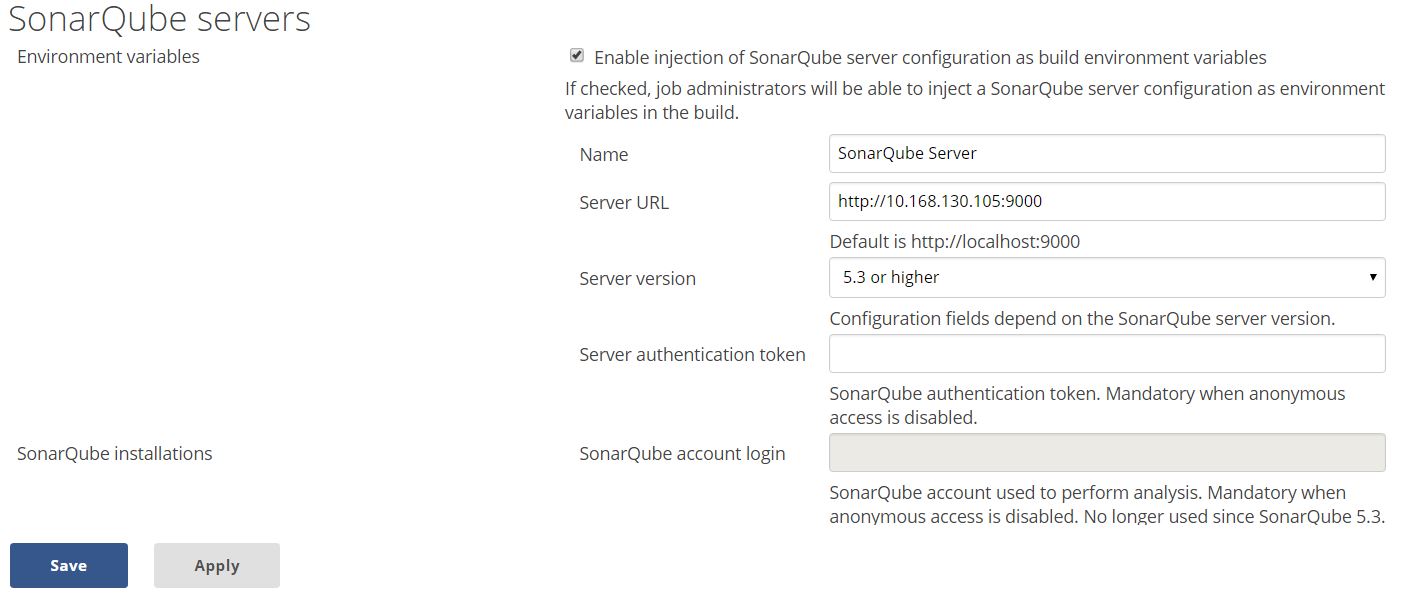
## **Configure global settings and paths**.



Go to Manage Jenkins🡪 Configure System



* 1. Configure Maven Project Configuration as following.
     1. I have selected default view as all, but any specific view can be selected.
  2. Configure setting for SonarQube server as following. Mention full url:port instead of localhost.

****

* 1. Configure Slave Status as following.

The status of slave machine can be shown as following.



* 1. Jenkins Location.



* 1. Configure Quality Gates - Sonarqube
  2. Configure Git plugin user.name and user.email values.
  3. Configure Extended Email configuration .
  4. Configure Email Notification.
     1. We need to configure JavaMail as same is used by Jenkins and it doesn’t comes with default java installation. Always take latest version of JavaMail API.

Configuring JavaMail for Email notification

* + 1. Update and configure latest JavaMail API, add jar file in classpath. Refer following documents
       1. <http://www.oracle.com/technetwork/java/javasebusiness/downloads/java-archive-downloads-eeplat-419426.html#javamail-1.4.5-oth-JPR>
       2. <https://javaee.github.io/javamail/#overview_description>
    2. Extract JavaMail latest version under Installed Java. i.e. C:\Program Files (x86)\Java
       1. Add JavaMail “mail.jar” into environment variable JAVAMAIL\_JAR as following.

JAVAMAIL\_JAR= C:\Program Files (x86)\Java\javamail-1.4.5\mail.jar

* + - 1. Add CLASSPATH = %JAVAMAIL\_JAR% into environment variable.
    1. If Using Windows Platform, need to do settings in McAfee
       1. Right-click Access Protection and select Properties
       2. Click the Access Protection tab
       3. Under Categories on the left, select Anti-virus Standard Protection
       4. In the right pane, select Prevent mass mailing worms from sending mail, then click Edit
       5. In the Processes to exclude section, type the process name, then click OK to close the Rule details window
       6. Click Apply then close the Access Protection Properties window
    2. Add following (processes to exclude) chrome.exe, eclipse.exe, java.exe, csrss.exe, jenkins.exe
  1. Configure ANDROID\_HOME as following



# **Chapter 5: Setting up Project in Jenkins**

## **Configure Project**

**Step 1** − Go to the Jenkins dashboard and Click on New Item



Step 2 − in the next screen, enter the Item name, in this case we have named it HelloWorld. Choose the ‘Freestyle project option’

**Step 3** − the following screen will come up in which you can specify the details of the job.



Give Label expression Name of Nodes (slaves), you may also specify custom workspace as mentioned.

**Step 4** − we need to specify the location of files which need to be built. Enter the url of that repository here. In addition to this, you would need to click on the Add button for the credentials to add a user name and password to the repository so that the code can be picked up from the remote repository.



Step 5: Configure Build Triggers

If Project have dependency on another project then select option “Build after other projects are built” as well.



**Step 6** − Now go to the Build section and click on Add build step → Execute Windows batch command, You may Invoke Ant, Gradle Script as per your project configuration.

**Step 7** − in the command window, enter the following commands and then click on the Save button.

javac HelloWorld.java

Java HelloWorld



Alternatively->



**Step 8** - Configure Sonar Scanner.

Analysis properties-> can be written as following.

## Sample sonar-project.properties as following

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

#Configure here general information about the environment, such as SonarQube DB details for example

#No information about specific project should appear here

#Default SonarQube server

sonar.host.url=http://10.168.130.105:9000

#Default source code encoding

sonar.sourceEncoding=UTF-8

# required metadata

sonar.projectKey= HelloWorld

#Java project analyzed with the SonarQube Runner

sonar.projectName=HelloWorld

sonar.projectVersion=1.0

#Language

sonar.language=java

#java version used by source files:

sonar.java.source=1.8

#Comma-separated paths to directories with sources (required)

sonar.sources=src/

#path to Java project compiled classes (optional)

sonar.java.binaries=bin/com

#sonar.java.binaries=target/classes

#comma-separated list of paths to libraries (optional

sonar.java.libraries=lib

#path to test source directories (optional)

#sonar.tests=testDir1, testDir2

**Step 9** – Configure Editable Email Notification

Once saved, you can click on the Build Now option to see if you have successfully defined the job.

**Step 10**− Once the build is completed, a status of the build will show if the build was successful or not. In our case, the following build has been executed successfully. Click on the #1 in the Build history to bring up the details of the build.

**Step 11** − Click on the Console Output link to see the details of the build

# **Chapter 6: Setting up Dashboard**.

**Visit**

[**https://plugins.jenkins.io/build-monitor-plugin**](https://plugins.jenkins.io/build-monitor-plugin)

**Go to archive and download latest from**

[**https://updates.jenkins.io/download/plugins/build-monitor-plugin/**](https://updates.jenkins.io/download/plugins/build-monitor-plugin/)

**Upload downloaded plugin (.hpi file) and restart the Jenkins.**

**Now go to** [**http://10.168.130.105:8080/newView**](http://10.168.130.105:8080/newView)

****

Now you may select the Build Monitor View to see the Status of Builds.

Give the Name Sample and Configure the Projects included.



# **Chapter 7: Distributed Execution (Master and Slave Node)**

**Note: Prepare VM or Physical Machines have same kind of environment configuration and software installed. Bur additional software may be installed on slave nodes for additional task but there is no need to install Jenkins on slave node.**

## **Configure Master Machine**

* + 1. On Master Machine go to Manage Jenkins -> Manage Nodes



* + 1. Click on New Node
    2. Enter Node Name and Select Permanent Agent then press OK.
    3. Fill the information as following.
       1. Set a **number of executors**
          1. (One or more) as needed.
       2. Set a **Remote FS Root**
          1. A home directory for the master on the slave machine.
          2. For a *Windows slave*, use something like: "C:\Jenkins\"
       3. Select the appropriate **Usage** setting:
          1. For an additional worker: *Utilize this slave as much as possible*
          2. For specialized jobs: *Leave this machine for tied jobs only*
       4. **Launch Method**:
          1. An easy way to control a Windows slave is by using *Launch slave agents via Java Web Start*  (Recommended for Windows)
          2. TODO: add steps for other methods.
       5. **Availability**
          1. *Keep this slave online as much as possible*
          2. TODO: add details for each option.
       6. Press **OK**.



## **Configure Slave Machine**

1. Open a browser on the **slave machine** and go to the **Jenkins master server** URL ([http://yourjenkinsmaster:8080](http://yourjenkinsmaster:8080/)).
2. Go to **Manage Jenkins** > **Manage Nodes**,

Click on the newly created slave machine.

Click on the **Launch** button to launch agent from browser on slave.



Agent will be downloaded and then you may start the same

