**Jenkins Configure (Maven-Setup) & Pipeline Workflow**

Before building a pipeline we have to create various project steps and configure it into pipeline

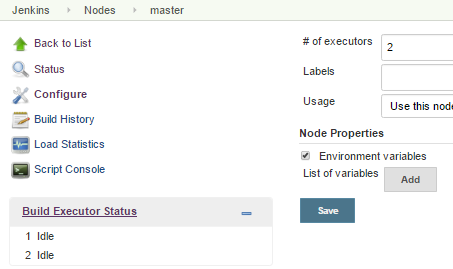
In this pipeline we are going to configure many project steps

**Initial Setup**

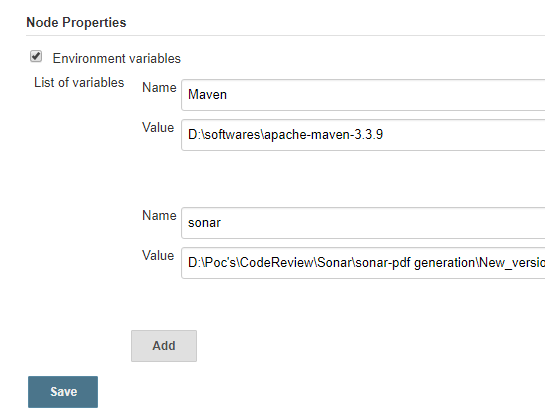
After successful installation initial steps to be done to configure the projects

**Note**:- Refer **1.Jenkins setup** **documentation** for Installation

Go to Jenkins->Manage Jenkins ->Manage Nodes->Master->Configure->Add



Set Environmental variables as follows



**Project Configuration**

# Checkout Source (GIT) and Unit Test- MavenFromGit\_UnitTest

1. **Plugins**

Jenkins -> Manage Jenkins -> Manage Plugins ->advanced

Install **Git Plugin** and dependency plugins to install software successfully

From this link <https://wiki.jenkins-ci.org/display/JENKINS/Git+Plugin>



Note:-Make sure that all the dependencies plugins must be added

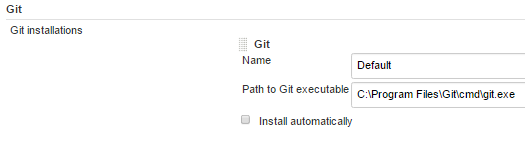
1. **Global Tool Configuration**

We can made the global configurations here

Jenkins -> Manage Jenkins ->Global Tool Configuration

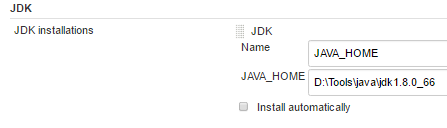
1. **Git Installation**

Set Git path



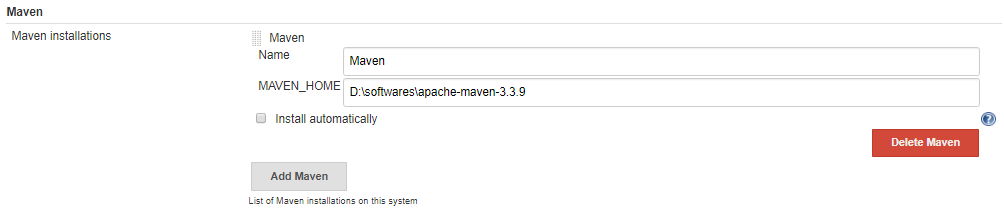
1. **JDK Installation**

Set Java\_Home



1. **Maven Installation**

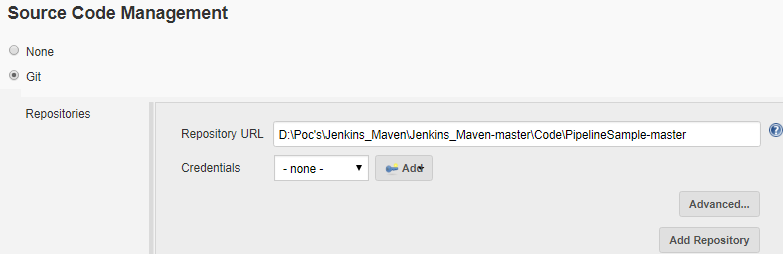
Set Maven Home



1. **Local Configuration**

Create New item->Maven project

1. Under **Source Code Management** -> git



Repository URL: Cloned git repository URL

1. **Build**

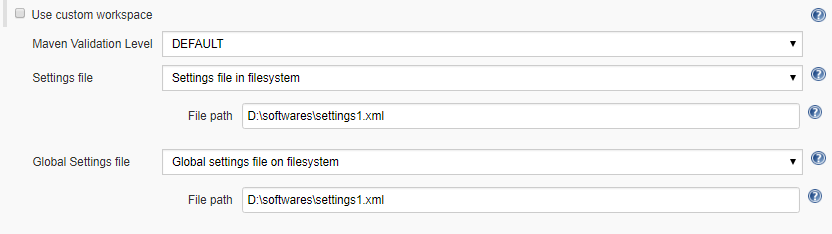


Root POM : pom.xml

Goals and options :Clean Install

(mention the goal)

Click **Advance** option in build and mention settings file details

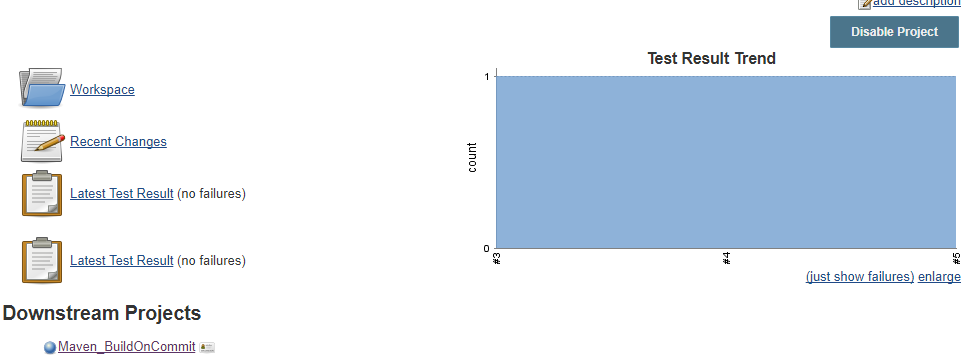


->Save->BuildNow

In Console



Build Successful and unit test report has been generated !!!

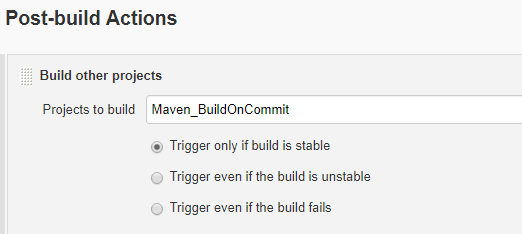


**note** : for unit test report – related plugins has to be included in the pom.xml of the source code

**note :-** Refer **Project-Maven Sample documentation** for Maven Project

**note :-** Refer **2.GIT documentation** for git installation

1. In **Post build action->build other project**



Give name of the other project which has to be run in sequential steps

# Build the code on post commit : Maven\_BuildOnCommit

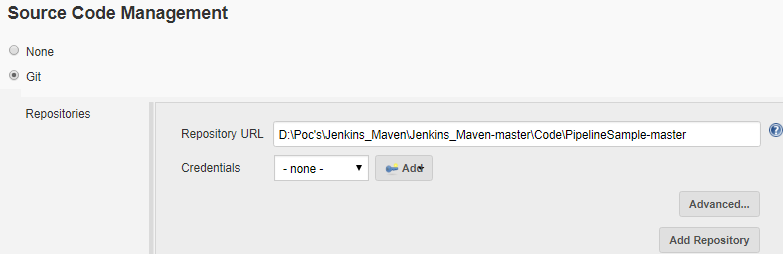
Once the code pulled from the git repository , if we made any changes and commit the code.

Code has to be build on post commit

1. **Local Configuration**

Create New item->Maven project

1. Under **Source Code Management** -> git



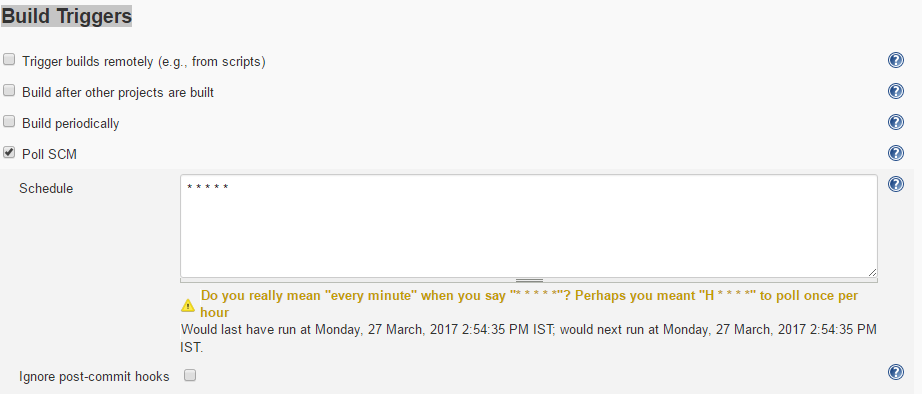
Repository URL: Cloned git repository URL

1. In **Build Triggers**

Select ->Poll SCM

And mention the details when the build should takes place after post commit

**(Note : Refer 2. Git documentation for post build setup)**



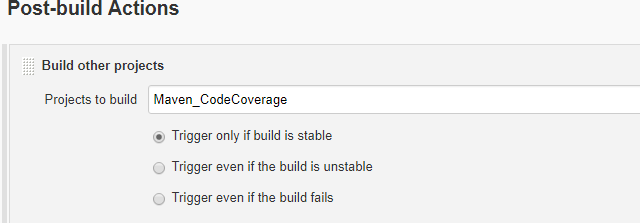
->Save->BuildNow

In Console



Build successful

1. In **Post build action->build other project**



Give name of the other project which has to be run in sequential steps

# MavenProject\_CodeCoverage

1. **Plugins**

Jenkins -> Manage Jenkins -> Manage Plugins ->advanced

Install **Git Plugin** and dependency plugins to install software successfully

From this link <https://wiki.jenkins-ci.org/display/JENKINS/Cobertura+Plugin>

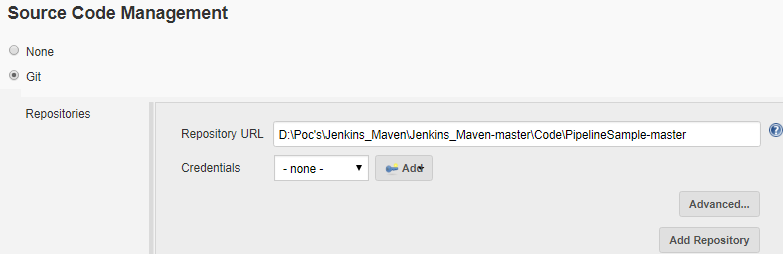


Note:-Make sure that all the dependencies plugins must be added

**b. Local Configuration**

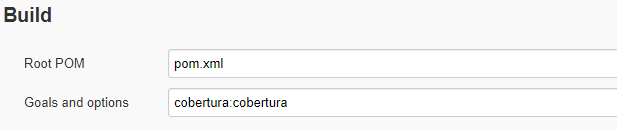
Create New item->Maven project

1. Under **Source Code Management ->** git



Repository URL: git repository URL

1. **Build**

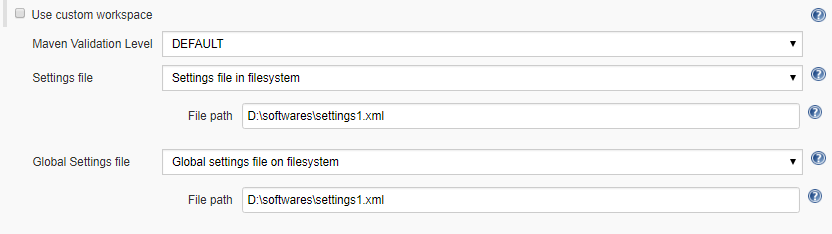


Root POM : pom.xml

Goals and options : cobertura:cobertura

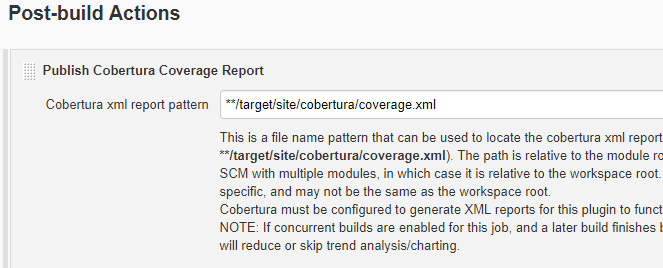
(mention the goal)

Click **Advance** option in build and mention settings file details



1. **Post-build Actions**

**For XML report**

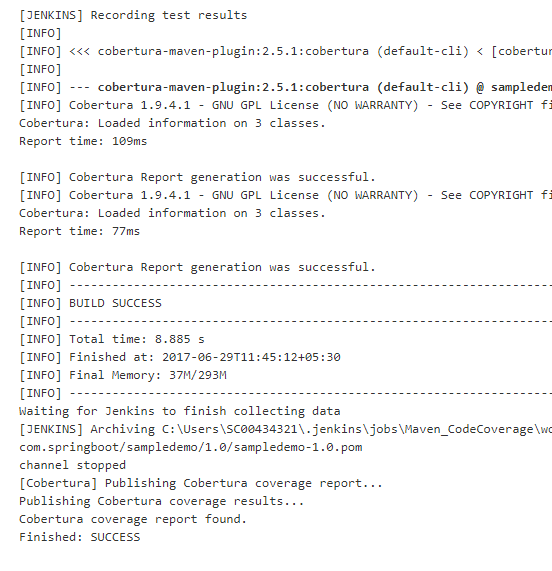


In **Post-build action** -> Publish Cobertura Coverage Reports

1. Cobertura xml report pattern –mention the space where the xml report found in the project structure

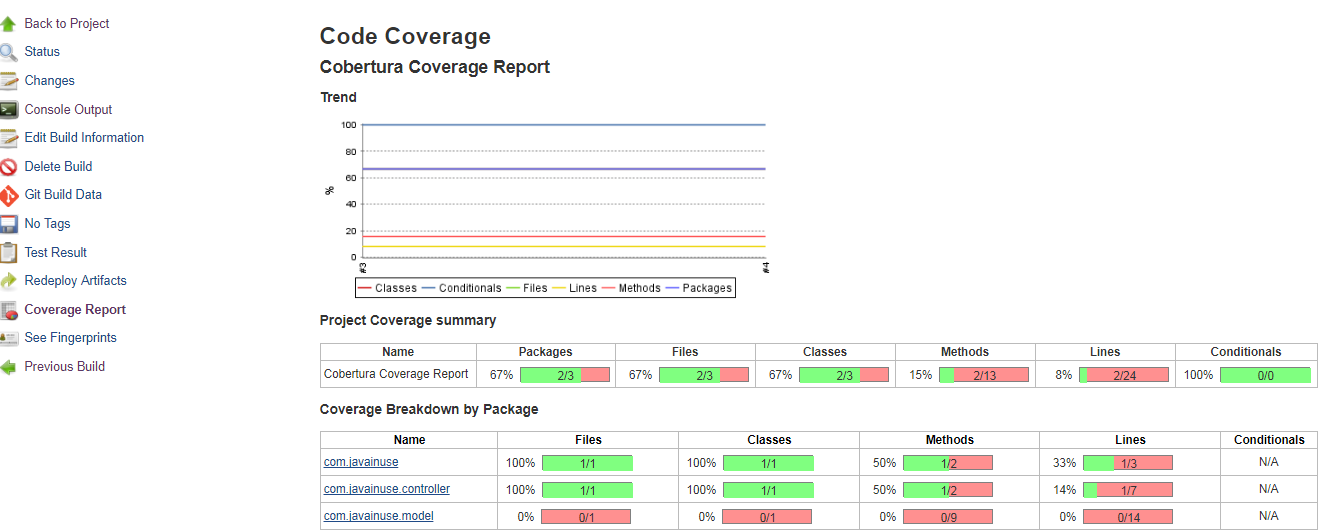
->Save->BuildNow

In Console



Build Successful!!!

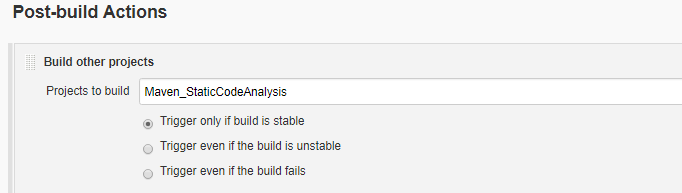
And report is shown in the graphical format



**note** : for Code coverage – related plugins has to be included in the pom.xml of the source code

**refer :- Project-Maven Doc** for Maven Project

1. In **Post build action**->build other project



Give name of the other project which has to be run in sequential step.

# Static code analysis and Quality Gate-Sonar

1. **Plugins**

Jenkins -> Manage Jenkins -> Manage Plugins ->advanced

Install **SonarQube Plugin** and dependency plugins to install software successfully

From this link <https://wiki.jenkins-ci.org/display/JENKINS/SonarQube+plugin>



Note:-Make sure that all the dependencies plugins must be added

To Install **Quality Gate Plugin**

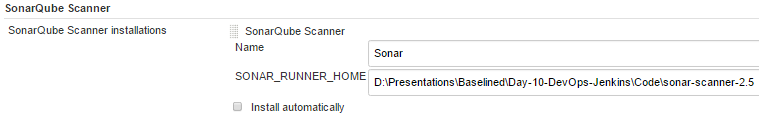
From this link <https://wiki.jenkins-ci.org/display/JENKINS/Quality+Gates+Plugin>

1. **Global Tool Configuration**

We can made the global configurations here

Jenkins -> Manage Jenkins ->Global Tool Configuration

1. **SonarQube Scanner**

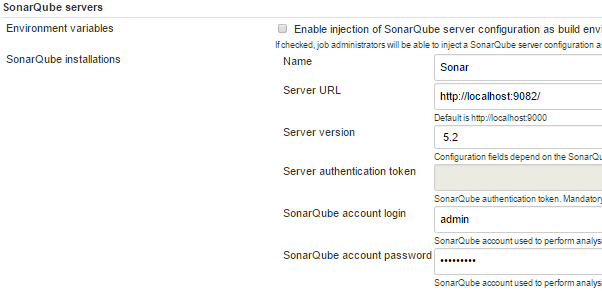


Name: name of our choice

Home: SonarScanner server home

**Note**:-server should be in running condition

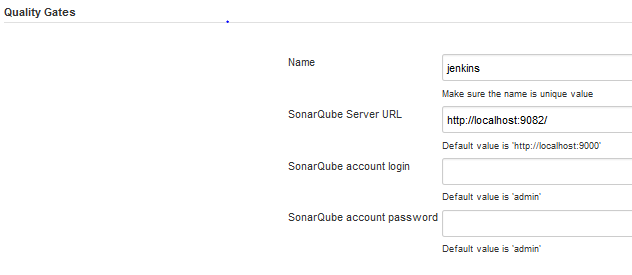
1. **Configuration System**
2. **SonarQube Servers**



Server URL: URL in which SonarQube server is running

**Note** :-server should be in running condition

1. **Quality Gates**



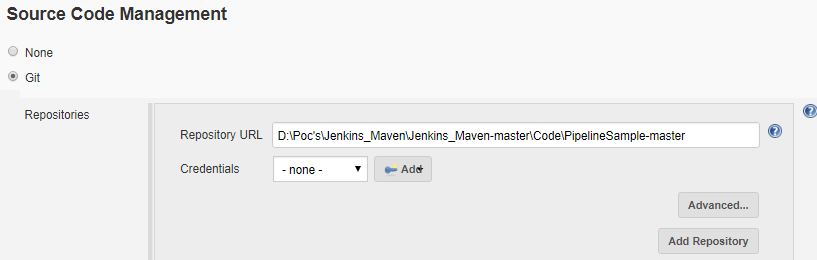
Server URL: URL in which SonarQube server is running

**Note** :-server should be in running condition

1. **Local Configuration**

Create New item->Maven project

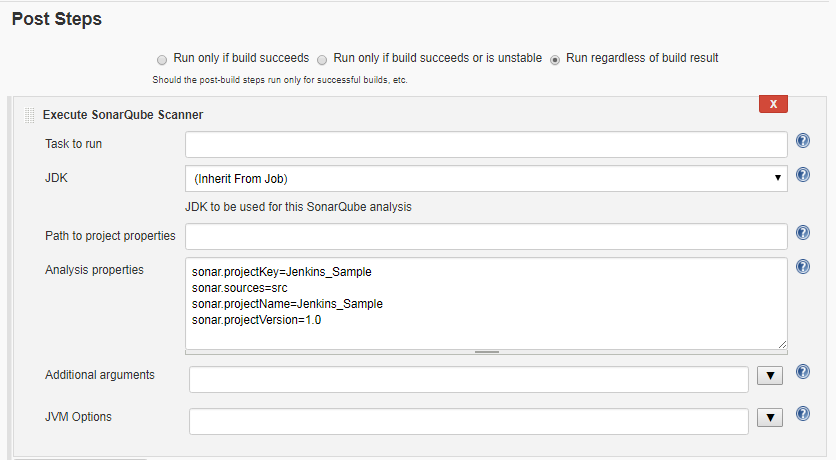
1. Under **Source Code Management** -> git



Repository URL: Cloned git repository URL

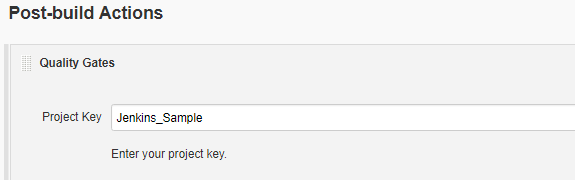
1. **Post Steps**

Select->Execute SonarQube Scanner



Have to mention the analysis properties (things mentioned in the sonar\_runner.properties file)

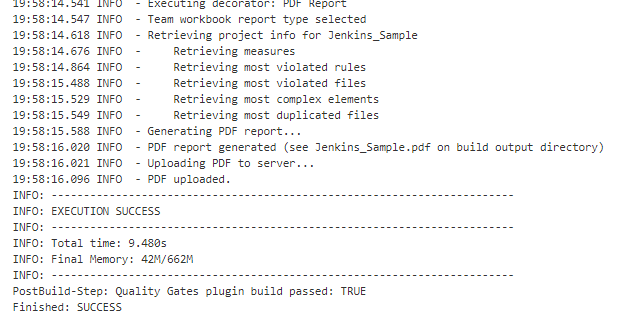
1. In **Post build action**->quality gates



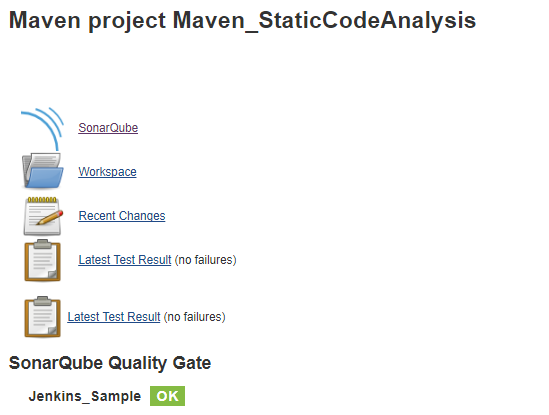
Mention the project key as mentioned in the sonar.projectKey in Build

->Save->BuildNow

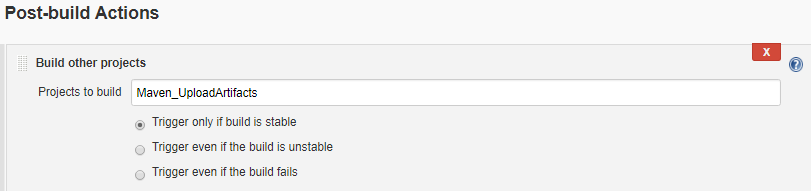
Console



We can able to see the report by clicking sonarqube



1. In **Post build action**->build other project



Give name of the other project which has to be run in sequential steps

# Maven\_Nexus\_ArtifactUpload

1. **Plugins**

Jenkins -> Manage Jenkins -> Manage Plugins ->advanced

Install **Sonatype Nexus Plugin** and dependency plugins to install software successfully

From this link <http://download.sonatype.com/nexus/ci/latest.hpi>

**Note**:-Make sure that all the dependencies plugins must be added

Refer:- 6. **Nexus Installation Doc** , for installation

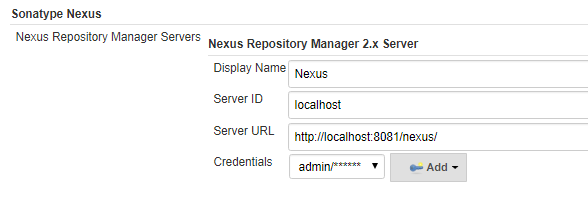
1. **Configure System**

We can made the global configurations here

Jenkins -> Manage Jenkins ->Configure System

**Sonatype Nexus -** to connect with Nexus Repository

Select -> **Nexus Repository 2.x Server**



Under **Sonatype nexus**

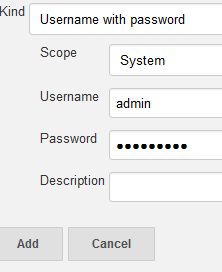
1.Display Name : Name to display for nexus

2.ServerID : IP Address for nexus repository

3.Server URL: Url to connect with your nexus repository

4.Credentials:

We have to select **username with password** option in the dropdown and enter the credentials



And click on add

If we click on **Test Connection** ->

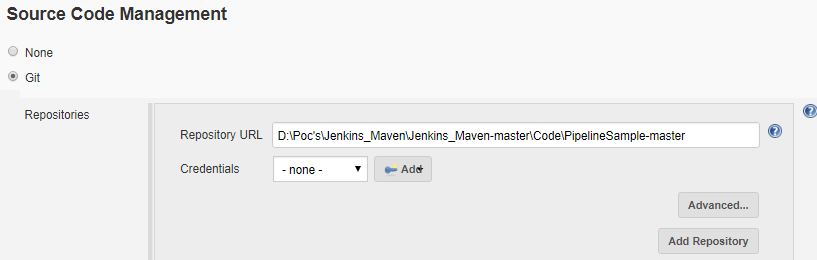
If connection established it will show like

**Nexus Repository Manager 2.x connection succeeded (2 hosted release Maven 2 repositories)**

1. **Local Configuration**

Create New item->Maven project

1. Under **Source Code Management** -> git



Repository URL: git repository URL

1. **Post step -Nexus Repository Manager Publisher** - to publish artifacts into repository

**Add build step->** **Nexus Repository Manager Publisher**

1. Nexus Instance: select the instance name from dropdown which we have created in global configuration section

2. Nexus Repository: select the repository to which we have permission to access

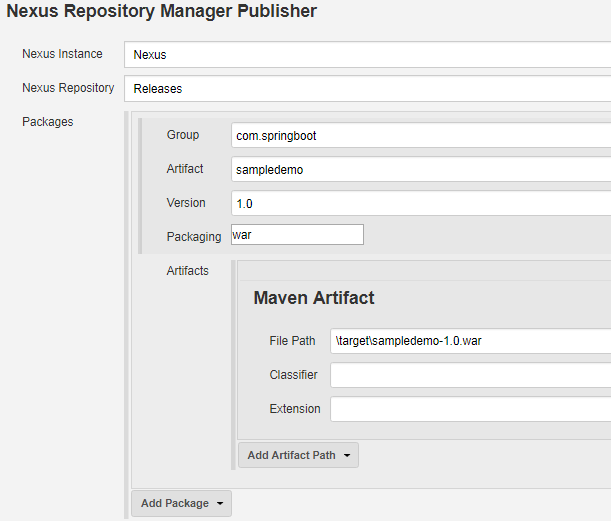
3. **Package**: select the package

Under while fill the details about the artifact which we are going to upload into nexus repository

Group, artifact , version and packaging(ex.war)

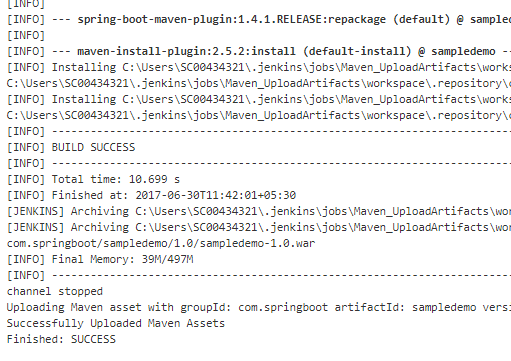
Click on **Artifact-Add artifact path-Maven artifact**

And mention the file path where the war or jar file stored

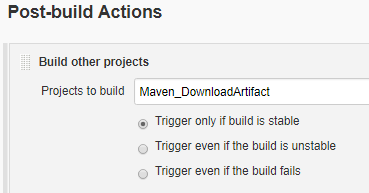


->Save->BuildNow

In Console



1. In **Post build action->**build other project



Give name of the other project which has to be run in sequential steps

# Maven\_Nexus\_DownloadArtifacts

1. **Plugins**

Jenkins -> Manage Jenkins -> Manage Plugins ->advanced

Install **Repository Connector plugin** and dependency plugins to install software successfully

From this link <https://wiki.jenkins-ci.org/display/JENKINS/Repository+Connector+Plugin>

Note:-Make sure that all the dependencies plugins must be added

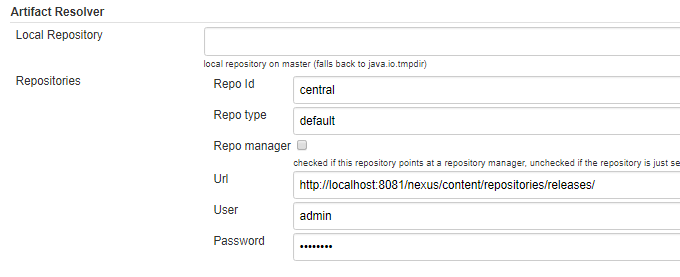
1. **Configure System**

We can made the global configurations here

Jenkins -> Manage Jenkins ->Configure System

**Artifact Resolver :**This Configuration has to be done once after the Artifacts stored into nexus then only we can retrieve .

**Artifact Resolver –** to retrieve artifact from Nexus



Under **Artifact Resolver**

1.Local Repository - we can able to give local directory path so that downloaded artifacts will be stored here

2.Repository

a)Repo Id –Central( by default)

b)Repo type- Default(by default)

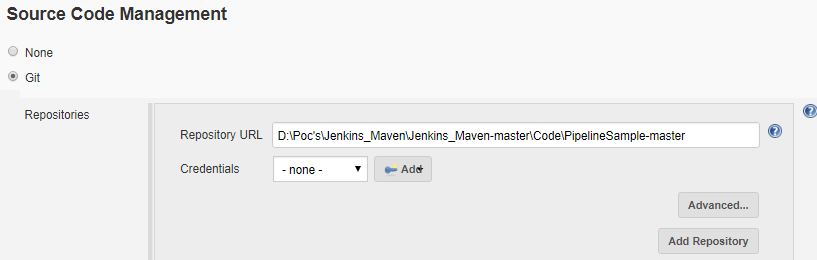
c)Url-path where your files stores inside the repository

d)username and password - which is used when connecting the nexus repository

1. **Local Configuration**

Create New item->Maven project

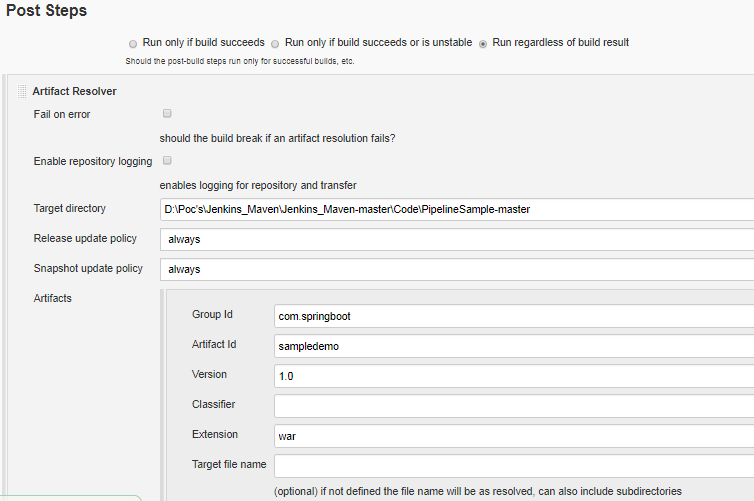
1. Under **Source Code Management** -> git



Repository URL: git repository URL

1. **Artifact Resolver –** to retrieve (download) artifact from nexus

**Add build step -> Artifact Resolver**



We have to mention the details about the artifact which we are going to download

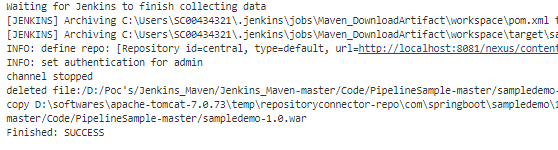
1.Target directory-the space where the downloaded file will be copied

2.Release update policy and snapshot update policy we can select ad per our wish

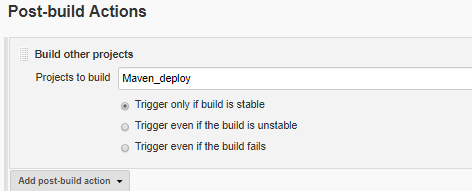
3.Artifact-have to mention the artifact details which we are going to download from the nexus repository

->Save->BuildNow

**In Console**



1. In **Post build action**->build other project



Give name of the other project which has to be run in sequential steps

# Deploy war into the server

1. **Plugins**

Jenkins -> Manage Jenkins -> Manage Plugins ->advanced

Install **Deploy Plugin** and dependency plugins to install software successfully

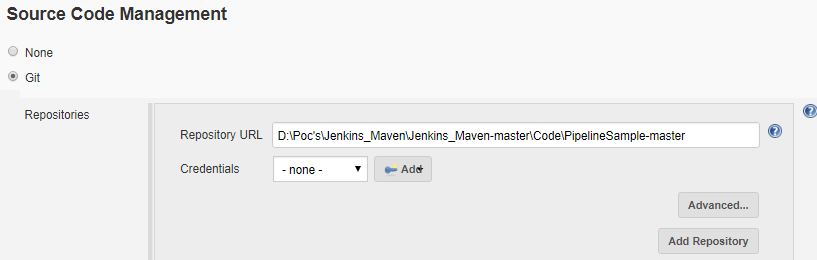
From this link <https://wiki.jenkins-ci.org/display/JENKINS/Deploy+Plugin>

Note:-Make sure that all the dependencies plugins must be added

1. **Local Configuration**

Create New item->Maven project

1. Under **source code management -> git**



Repository URL: git repository URL

1. **Post-build action**

In **Post-build action->deploy war/ear to a container**

1.War/ear file- \*\*\\*.war(by default it will take the copied war from the workspace which we have mentioned)

2.Context path- The context path that the container should use to publish the WAR/EAR

3.Container-Add container->can able to select the server container which is used for deployment

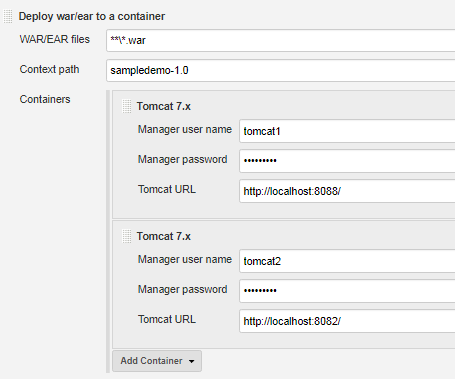
a. manager username-username of the tomcat server

b. manager password-password of the tomcat server

c. tomcat url- URL in which tomcat is running

we can also mention n number of servers

Refer:- **7.Deploy Document** for configuration



->Save->BuildNow

**In Console**



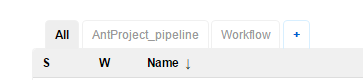
Project Steps completed Successfully!!!

**Jenkins Pipeline**

To create a build pipeline add **Pipeline Plugin**

From this link <https://wiki.jenkins-ci.org/display/JENKINS/Build+Pipeline+Plugin>

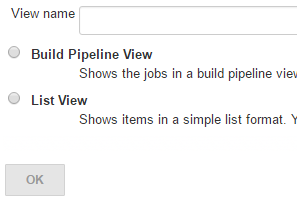
In Jenkins UI,Click on ”+” sign in order to add pipelin



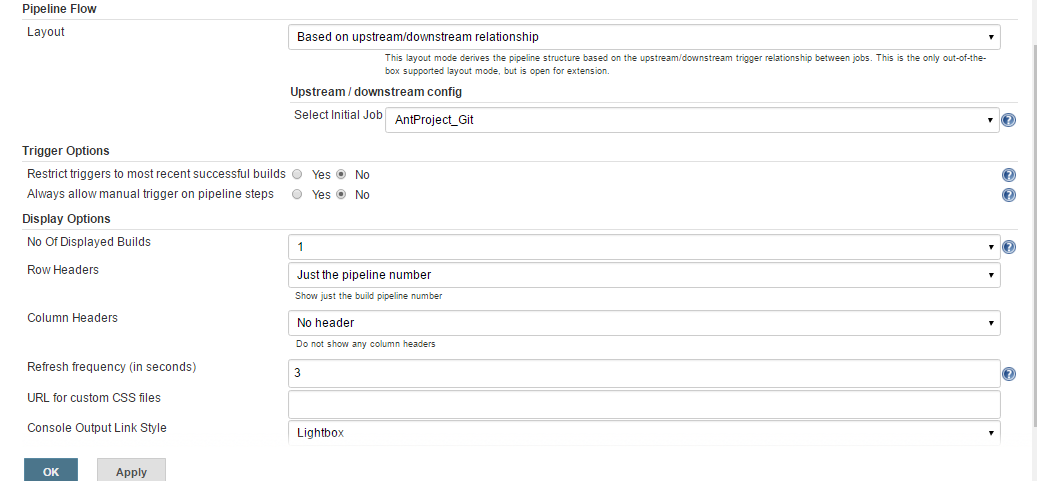
New pipeline

View name: name of the pipeline

click on build pipeline view



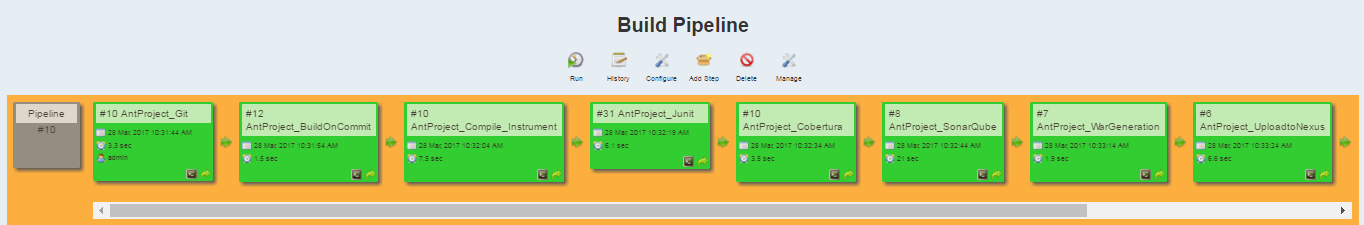
Click on OK



Select initial job: select the first project which has to be execute first in the pipeline

And click OK

Pipeline will appear ->click ->run



Pipeline executed successfully!!!