Jenkins Configure (Maven-Setup) & Pipeline Workflow

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

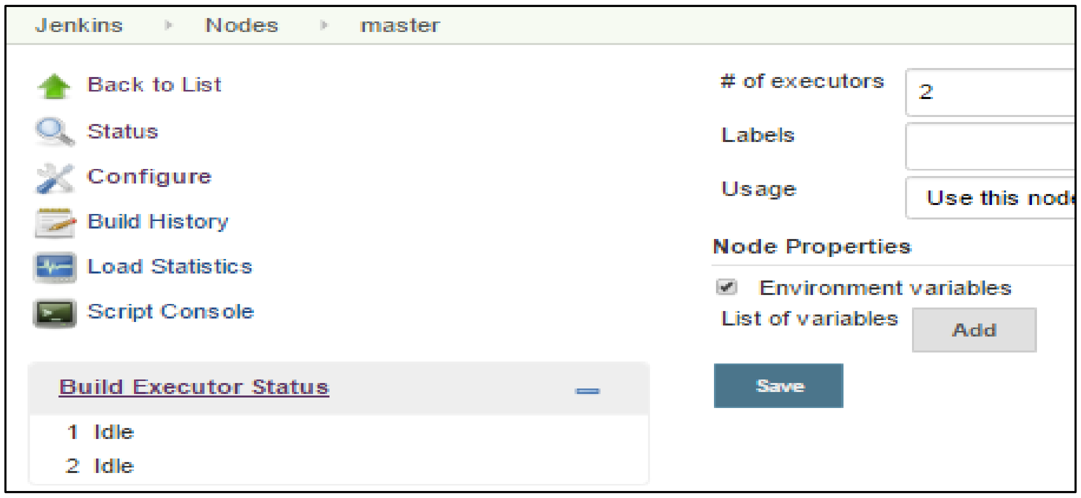
In this pipeline we are going to configure many project steps

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



Jenkins › Nodes › master

Back to List
Status
Configure
Build History
Load Statistics
Script Console

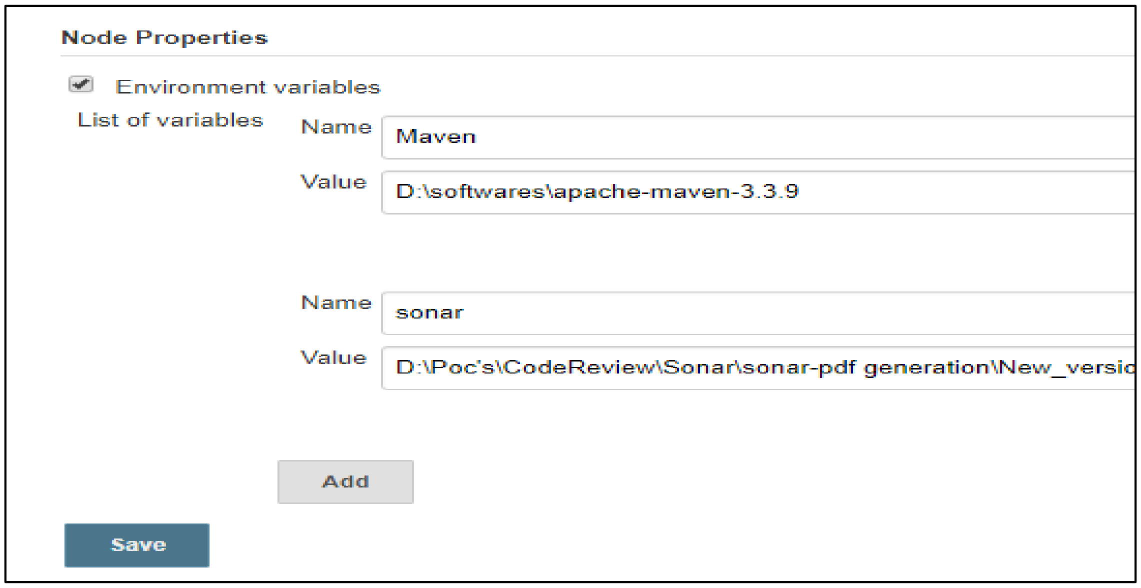
Build Executor Status
1 Idle
2 Idle

# of executors	2
Labels	
Usage	Use this nod

Node Properties
✓ Environment variables
List of variables Add

Save

Set Environmental variables as follows


Node Properties

☑ Environment variables

List of variables

Name	Maven
Value	D:\softwares\apache-maven-3.3.9
Name	sonar
Value	D:\Poc's\CodeReview\Sonar\sonar-pdf generation\New_versio

Project Configuration

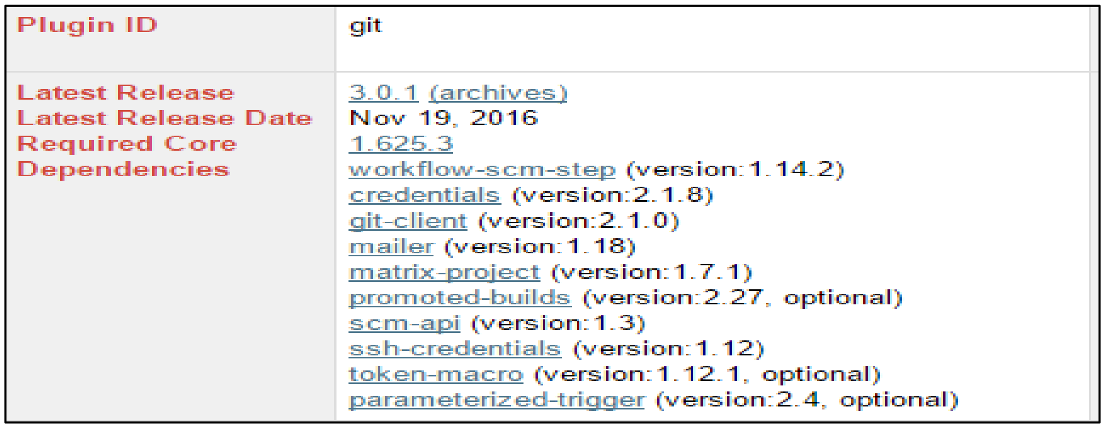
1. Checkout Source (GIT) and Unit Test- MavenFromGit_UnitTest

a. 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



Plugin ID	git
Latest Release	3.0.1 (archives)
Latest Release Date	Nov 19, 2016
Required Core	1.625.3
Dependencies	workflow-scm-step (version:1.14.2) credentials (version:2.1.8) git-client (version:2.1.0) mailer (version:1.18) matrix-project (version:1.7.1) promoted-builds (version:2.27, optional) scm-api (version:1.3) ssh-credentials (version:1.12) token-macro (version:1.12.1, optional) parameterized-trigger (version:2.4, optional)

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

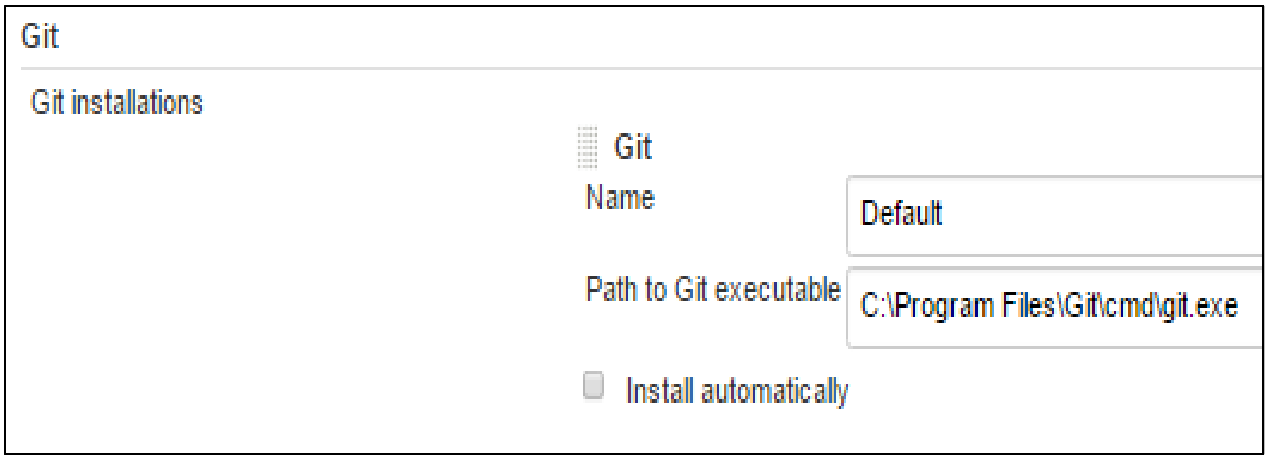
b. Global Tool Configuration

We can made the global configurations here

Jenkins -> Manage Jenkins ->Global Tool Configuration

I. Git Installation

Set Git path



Git

Git installations

Git

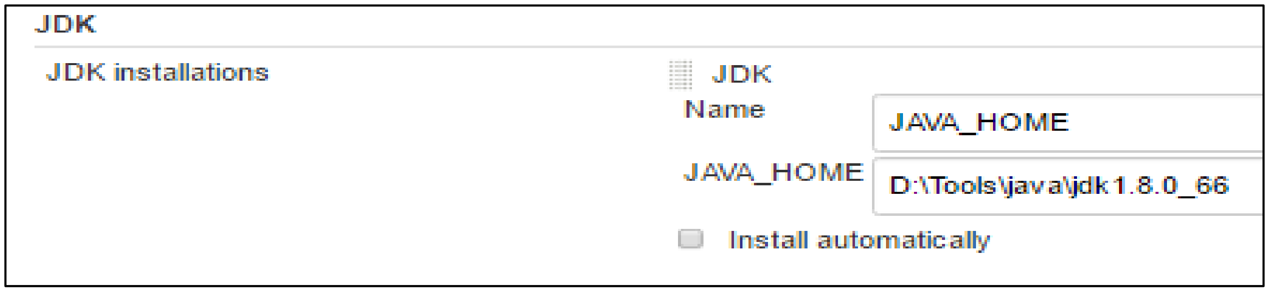
Name: Default

Path to Git executable: C:\Program Files\Git\cmd\git.exe

Install automatically

II. JDK Installation

Set Java_Home



JDK

JDK installations

JDK

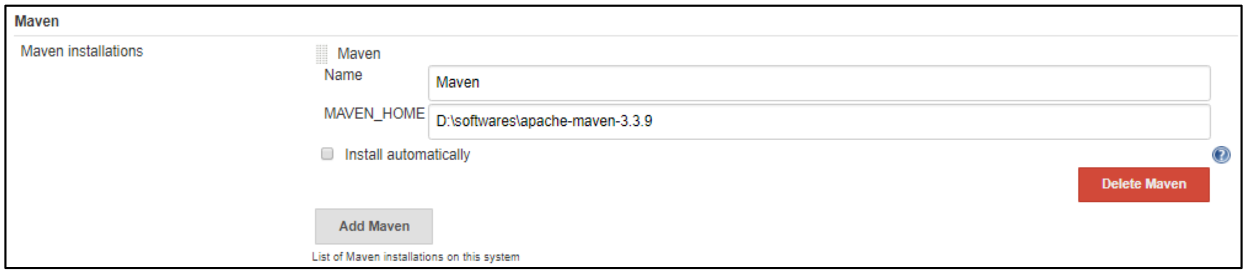
Name: JAVA_HOME

JAVA_HOME: D:\Tools\java\jdk1.8.0_66

Install automatically

III. Maven Installation

Set Maven Home



Maven

Maven installations

Maven

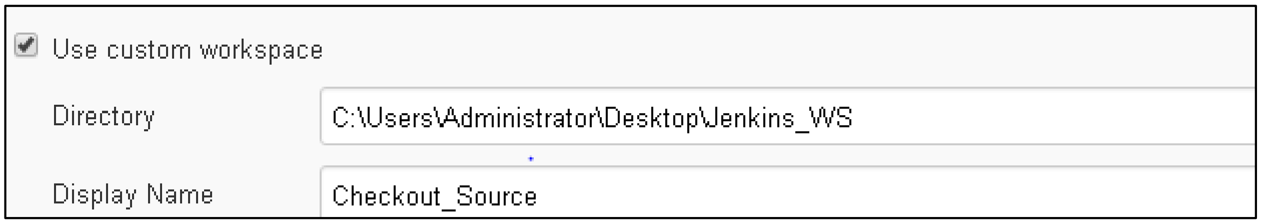
Name: Maven

MAVEN_HOME: D:\softwares\apache-maven-3.3.9

Install automatically

c. Local Configuration

I. Under Advanced Project Options



[✓] Use custom workspace

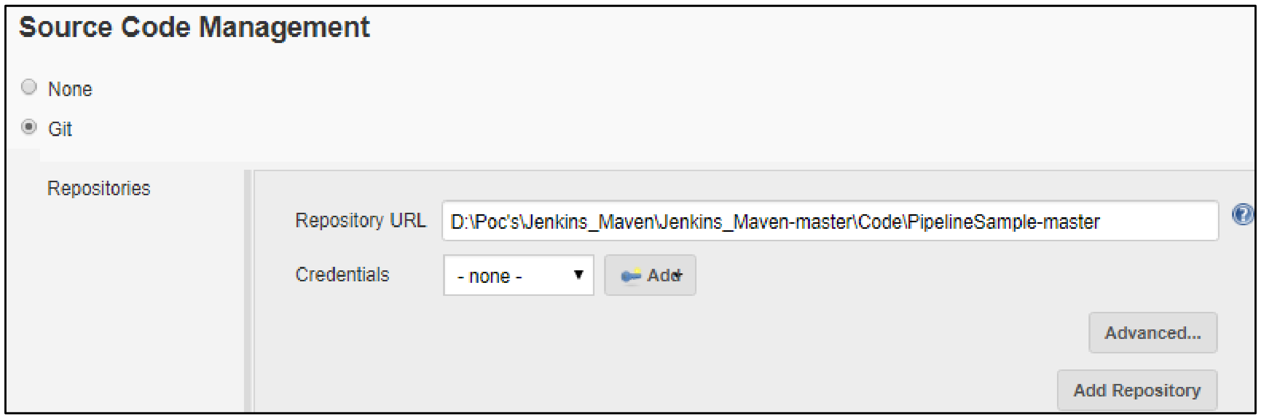
Directory: C:\Users\Administrator\Desktop\Jenkins_WS

Display Name: Checkout_Source

Directory : Mention the custom workspace directory(cloned source will be available in this directory)

DisplayName :Display name of the project

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



Source Code Management

None [✓ Git]

Repositories

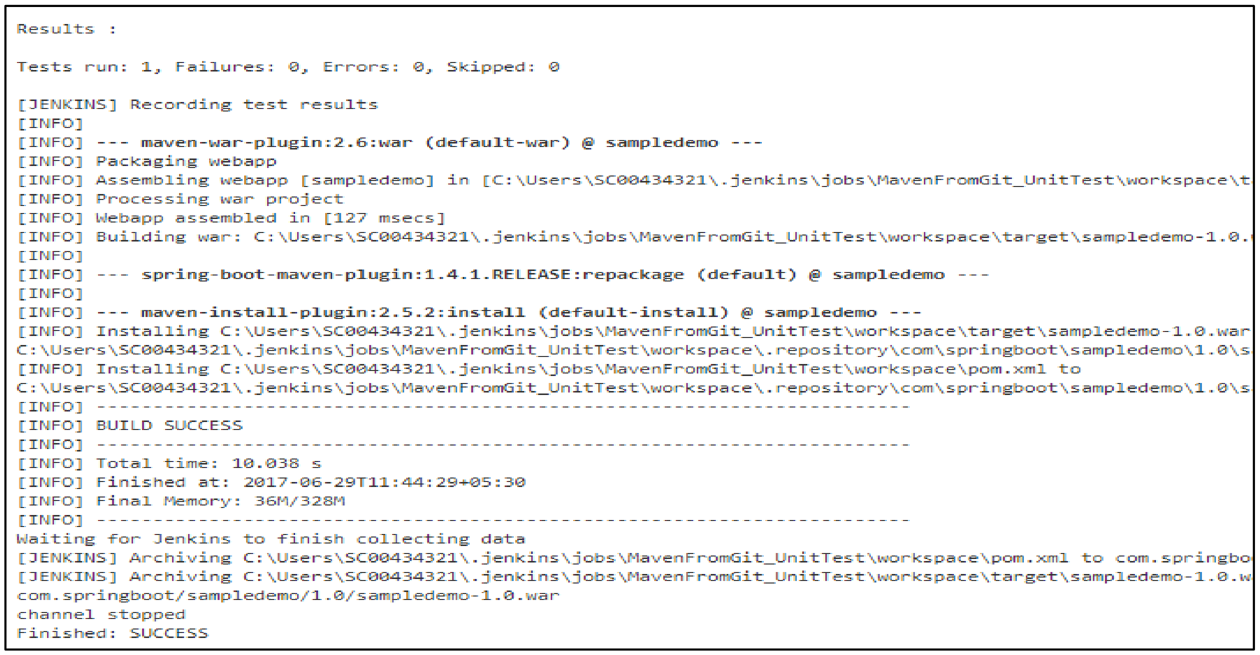
Repository URL: D:\Poc's\Jenkins_Maven\Jenkins_Maven-master\Code\PipelineSample-master

Credentials: - none -

Repository URL: Cloned git repository URL

->Save->BuildNow

In Console



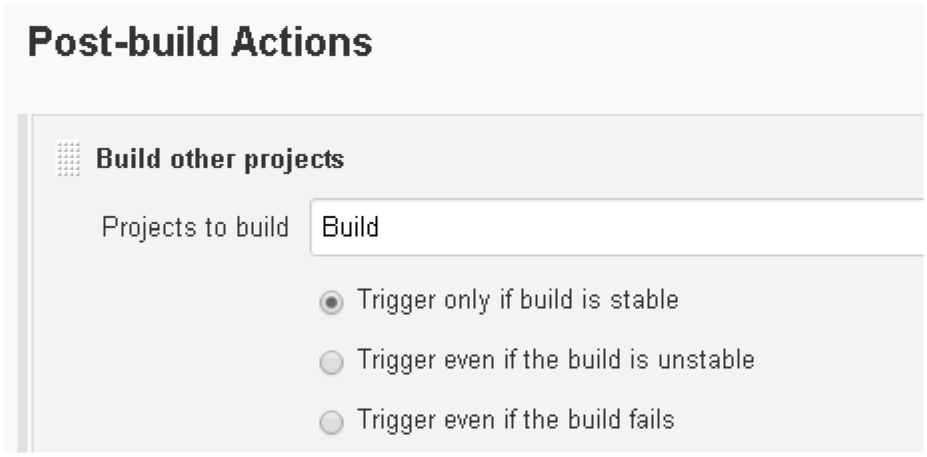
```
Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[JENKINS] Recording test results
[INFO]
[INFO] --- maven-war-plugin:2.6:war (default-war) @ sampledemo ---
[INFO] Packaging webapp
[INFO] Assembling webapp [sampledemo] in [C:\Users\SC00434321\.jenkins\jobs\MavenFromGit_UnitTest\workspace\t
[INFO] Processing war project
[INFO] Webapp assembled in [127 msecs]
[INFO] Building war: C:\Users\SC00434321\.jenkins\jobs\MavenFromGit_UnitTest\workspace\target\sampledemo-1.0.
[INFO]
[INFO] --- spring-boot-maven-plugin:1.4.1.RELEASE:repackage (default) @ sampledemo ---
[INFO]
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ sampledemo ---
[INFO] Installing C:\Users\SC00434321\.jenkins\jobs\MavenFromGit_UnitTest\workspace\target\sampledemo-1.0.war
C:\Users\SC00434321\.jenkins\jobs\MavenFromGit_UnitTest\workspace\.repository\com\springboot\sampledemo\1.0\s
[INFO] Installing C:\Users\SC00434321\.jenkins\jobs\MavenFromGit_UnitTest\workspace\pom.xml to
C:\Users\SC00434321\.jenkins\jobs\MavenFromGit_UnitTest\workspace\.repository\com\springboot\sampledemo\1.0\s
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 10.038 s
[INFO] Finished at: 2017-06-29T11:44:29+05:30
[INFO] Final Memory: 36M/328M
[INFO] ------------------------------------------------------------------------
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving C:\Users\SC00434321\.jenkins\jobs\MavenFromGit_UnitTest\workspace\pom.xml to com.springbo
[JENKINS] Archiving C:\Users\SC00434321\.jenkins\jobs\MavenFromGit_UnitTest\workspace\target\sampledemo-1.0.w
com.springboot/sampledemo/1.0/sampledemo-1.0.war
channel stopped
Finished: SUCCESS
```

Build Successful!!!

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

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

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



Post-build Actions

Build other projects

Projects to build: Build

[✓ Trigger only if build is stable]
Trigger even if the build is unstable
Trigger even if the build fails

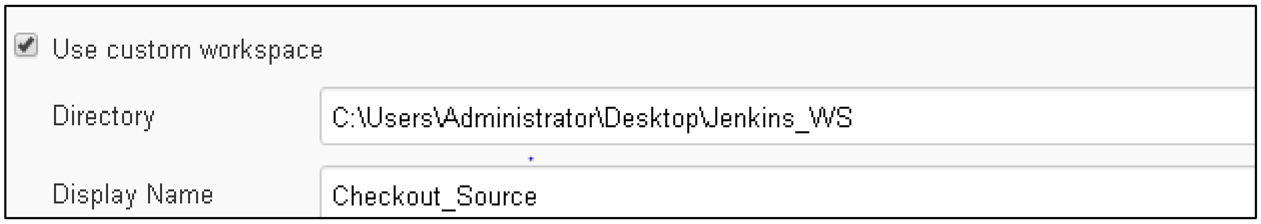
Give name of the other project which has to be run in sequential steps

2. Build

a) Local Configuration

Create New item->freestyle project

I. Under Advanced Project Options



☑ Use custom workspace

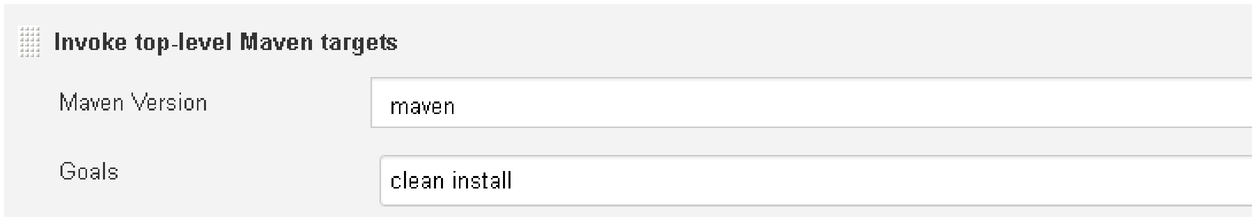
Directory	C:\Users\Administrator\Desktop\Jenkins_WS
Display Name	Checkout_Source

Directory : Mention the custom workspace directory(cloned repository)

DisplayName :Display name of the project

II. In **Build**

Select "Invoke top level maven"



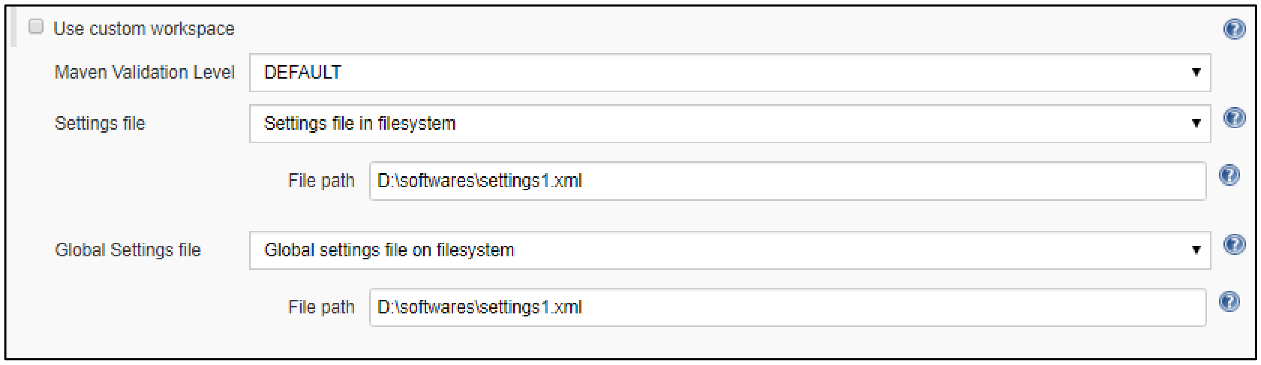
Invoke top-level Maven targets

Maven Version	maven
Goals	clean install

Maven version :Which is mentioned in the global tool config

Goals : clean install

Click **Advance** option in build and mention settings file details(If not open network)

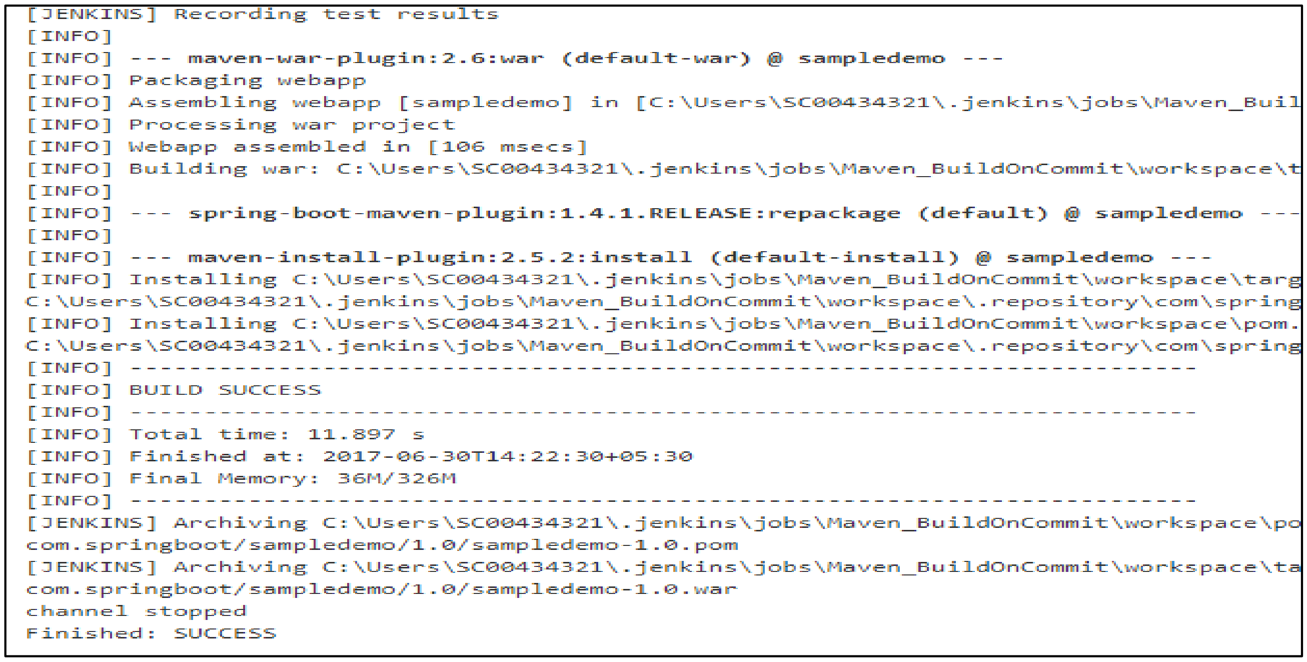


☐ Use custom workspace

Maven Validation Level	DEFAULT
Settings file	Settings file in filesystem
File path	D:\softwares\settings1.xml
Global Settings file	Global settings file on filesystem
File path	D:\softwares\settings1.xml

->Save->BuildNow

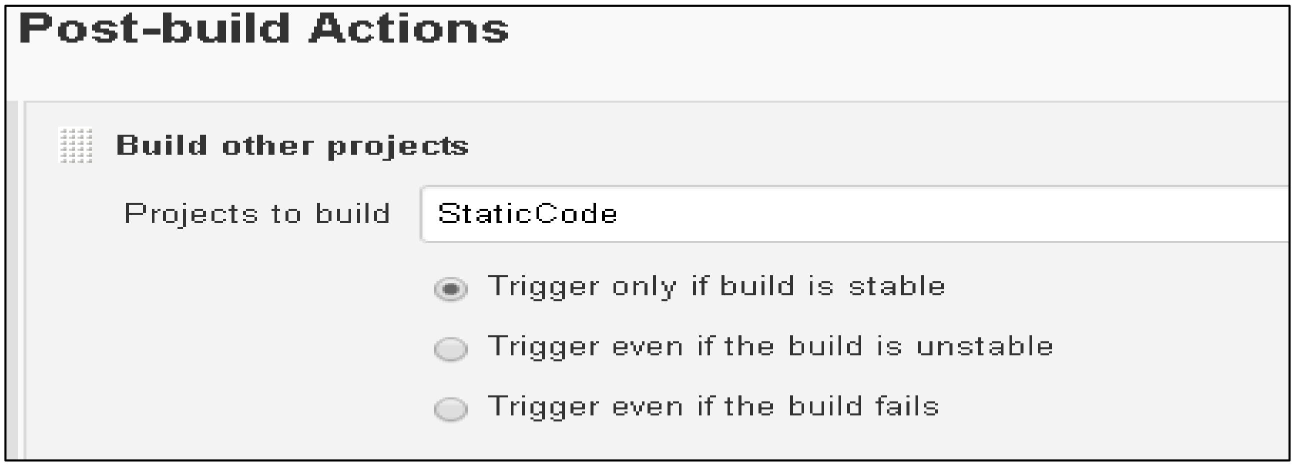
In Console



```
[JENKINS] Recording test results
[INFO]
[INFO] --- maven-war-plugin:2.6:war (default-war) @ sampledemo ---
[INFO] Packaging webapp
[INFO] Assembling webapp [sampledemo] in [C:\Users\SC00434321\.jenkins\jobs\Maven_Buil
[INFO] Processing war project
[INFO] Webapp assembled in [106 msecs]
[INFO] Building war: C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\t
[INFO]
[INFO] --- spring-boot-maven-plugin:1.4.1.RELEASE:repackage (default) @ sampledemo ---
[INFO]
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ sampledemo ---
[INFO] Installing C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\targ
C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\.repository\com\spring
[INFO] Installing C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\pom.
C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\.repository\com\spring
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 11.897 s
[INFO] Finished at: 2017-06-30T14:22:30+05:30
[INFO] Final Memory: 36M/326M
[INFO] ------------------------------------------------------------------------
[JENKINS] Archiving C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\po
com.springboot/sampledemo/1.0/sampledemo-1.0.pom
[JENKINS] Archiving C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\ta
com.springboot/sampledemo/1.0/sampledemo-1.0.war
channel stopped
Finished: SUCCESS
```

Build successful

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



Post-build Actions

Build other projects

Projects to build: StaticCode

[✓ Trigger only if build is stable]
Trigger even if the build is unstable
Trigger even if the build fails

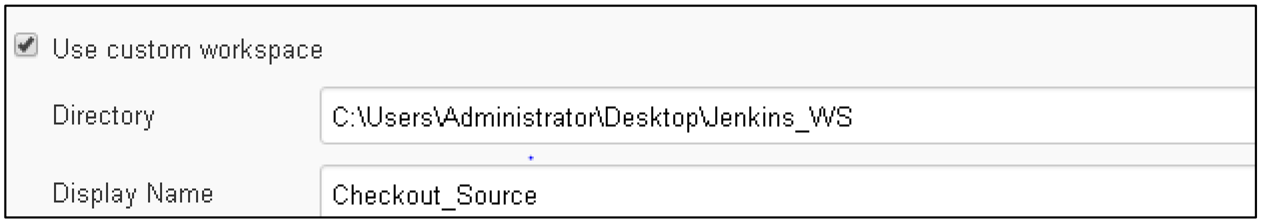
Give name of the other project which has to be run in sequential steps

3. 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

a) Local Configuration

I. Under Advanced Project Options



☑ Use custom workspace

Directory C:\Users\Administrator\Desktop\Jenkins_WS

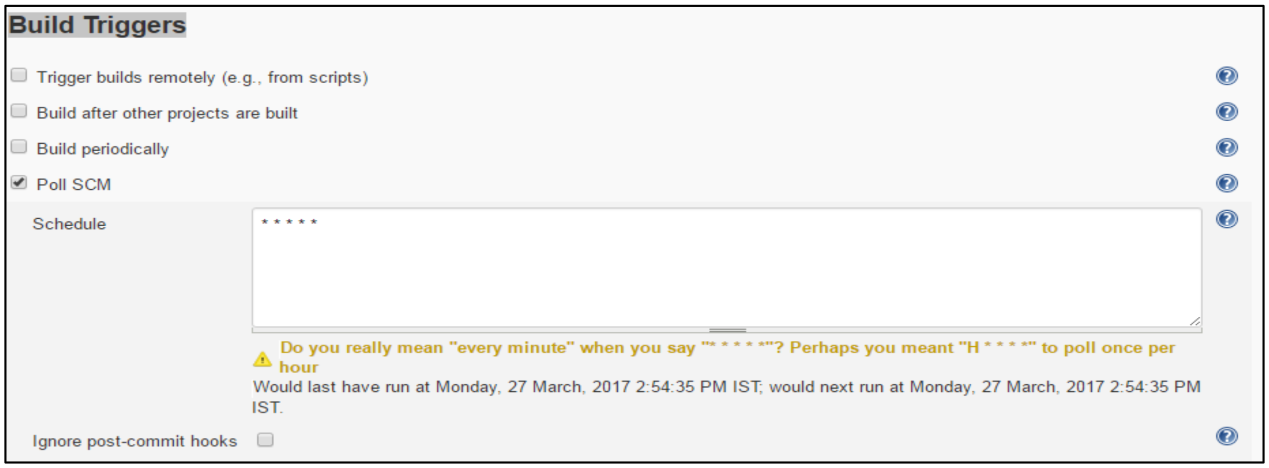
Display Name Checkout_Source

Directory : Mention the custom workspace directory(cloned repository)
DisplayName :Display name of the project

II. 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)



Build Triggers

☐ Trigger builds remotely (e.g., from scripts)
☐ Build after other projects are built
☐ Build periodically
☑ Poll SCM

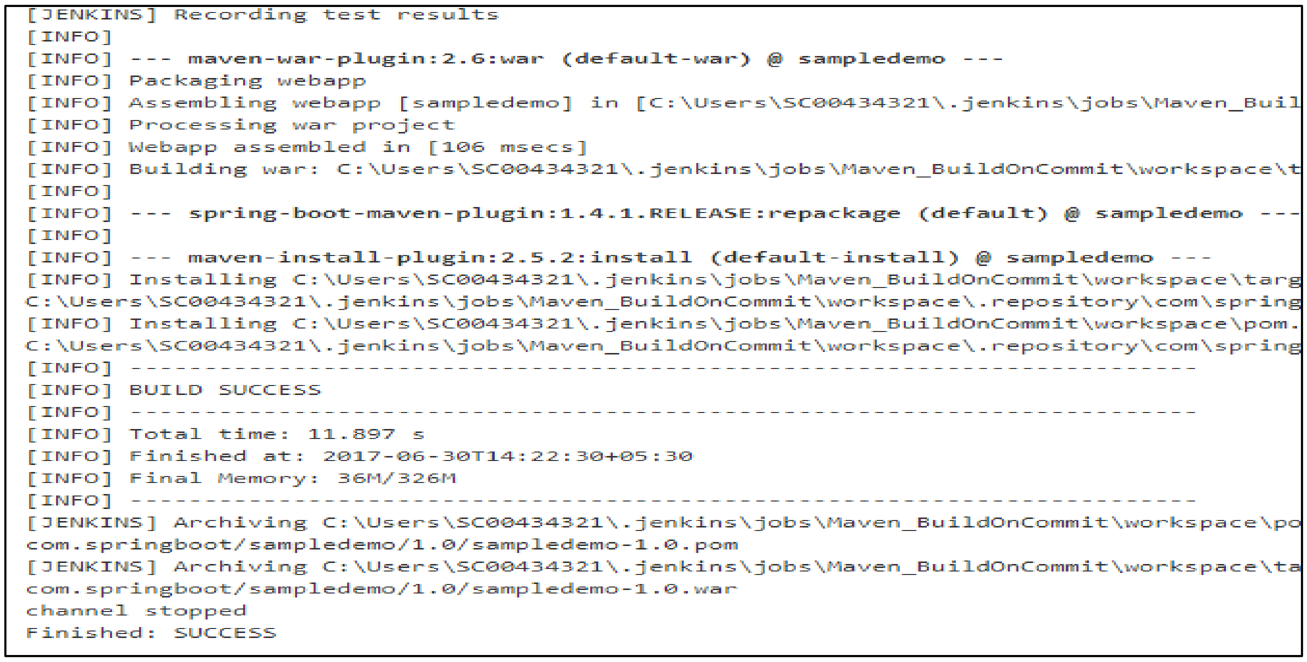
Schedule * * * * *

Do you really mean "every minute" when you say "* * * * *"? Perhaps you meant "H * * * *" to poll once per hour
Would last have run at Monday, 27 March, 2017 2:54:35 PM IST; would next run at Monday, 27 March, 2017 2:54:35 PM IST.

Ignore post-commit hooks ☐

->Save->BuildNow

In Console



```
[JENKINS] Recording test results
[INFO]
[INFO] --- maven-war-plugin:2.6:war (default-war) @ sampledemo ---
[INFO] Packaging webapp
[INFO] Assembling webapp [sampledemo] in [C:\Users\SC00434321\.jenkins\jobs\Maven_Buil
[INFO] Processing war project
[INFO] Webapp assembled in [106 msecs]
[INFO] Building war: C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\t
[INFO]
[INFO] --- spring-boot-maven-plugin:1.4.1.RELEASE:repackage (default) @ sampledemo ---
[INFO]
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ sampledemo ---
[INFO] Installing C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\targ
C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\.repository\com\spring
[INFO] Installing C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\pom.
C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\.repository\com\spring
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 11.897 s
[INFO] Finished at: 2017-06-30T14:22:30+05:30
[INFO] Final Memory: 36M/326M
[INFO] ------------------------------------------------------------------------
[JENKINS] Archiving C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\po
com.springboot/sampledemo/1.0/sampledemo-1.0.pom
[JENKINS] Archiving C:\Users\SC00434321\.jenkins\jobs\Maven_BuildOnCommit\workspace\ta
com.springboot/sampledemo/1.0/sampledemo-1.0.war
channel stopped
Finished: SUCCESS
```

Build successful

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

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

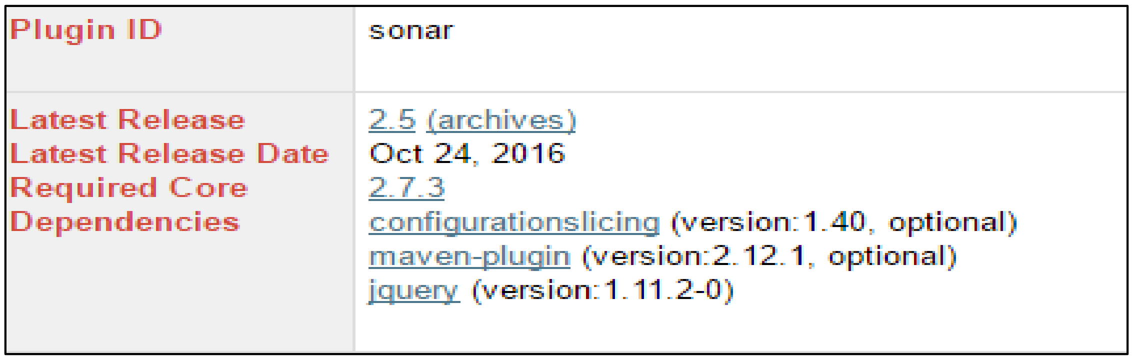
4. Static code analysis and Quality Gate-Sonar

a. 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



Plugin ID	sonar
Latest Release Latest Release Date Required Core Dependencies	2.5 (archives) Oct 24, 2016 2.7.3 configurationslicing (version:1.40, optional) maven-plugin (version:2.12.1, optional) jquery (version:1.11.2-0)

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

To Install **Quality Gate Plugin**

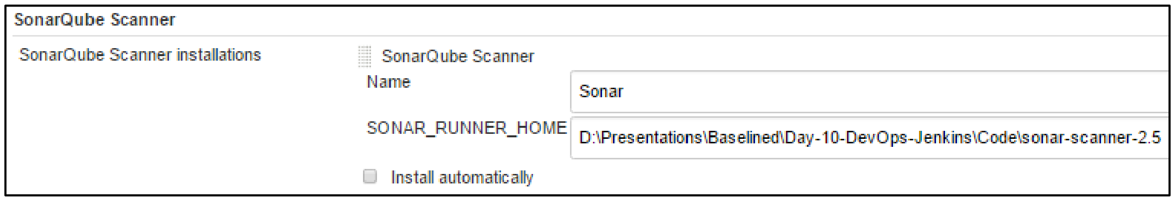
From this link https://wiki.jenkins-ci.org/display/JENKINS/Quality+Gates+Plugin

b. Global Tool Configuration

We can made the global configurations here

Jenkins -> Manage Jenkins ->Global Tool Configuration

I. SonarQube Scanner



SonarQube Scanner

SonarQube Scanner installations

SonarQube Scanner

Name: Sonar

SONAR_RUNNER_HOME: D:\Presentations\Baselined\Day-10-DevOps-Jenkins\Code\sonar-scanner-2.5

☐ Install automatically

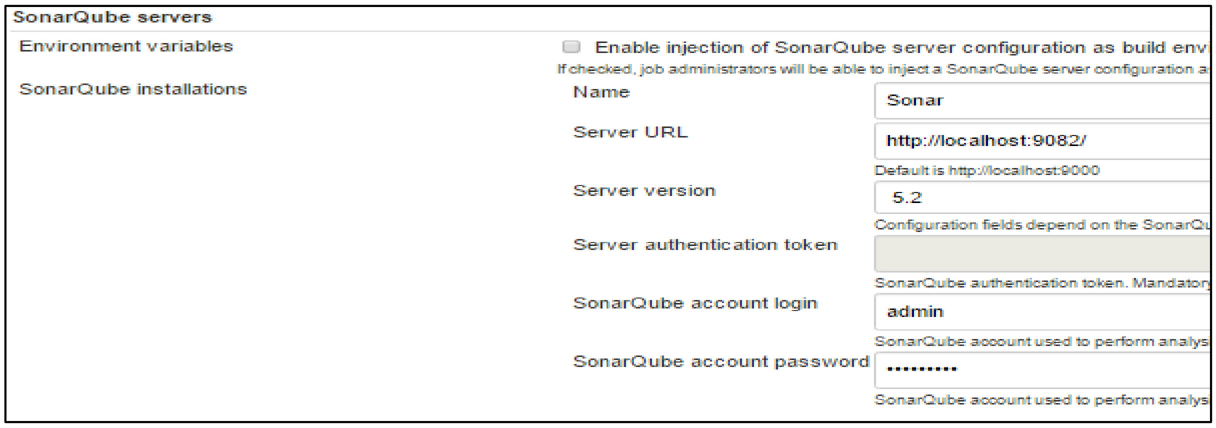
Name: name of our choice

Home: SonarScanner server home

Note:-server should be in running condition

c. Configuration System

I. SonarQube Servers



SonarQube servers

Environment variables — Enable injection of SonarQube server configuration as build env

If checked, job administrators will be able to inject a SonarQube server configuration a

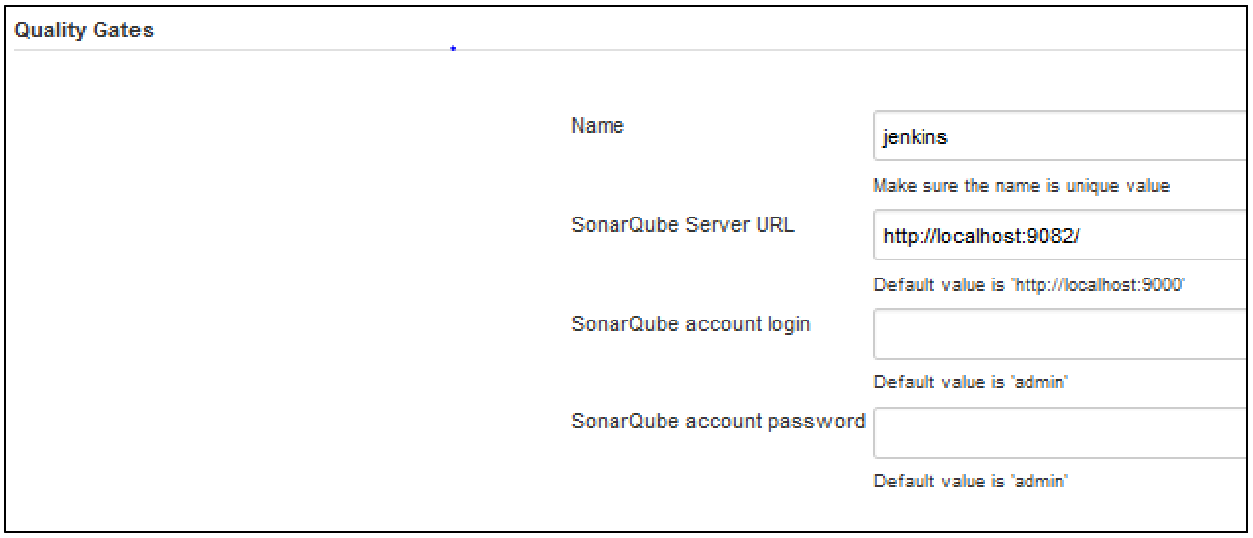
SonarQube installations

Name	Sonar
Server URL	http://localhost:9082/
	Default is http://localhost:9000
Server version	5.2
	Configuration fields depend on the SonarQu
Server authentication token	
	SonarQube authentication token. Mandatory
SonarQube account login	admin
	SonarQube account used to perform analys
SonarQube account password	•••••••••
	SonarQube account used to perform analys

Server URL: URL in which SonarQube server is running

Note :-server should be in running condition

II. Quality Gates



Quality Gates

Name	jenkins
	Make sure the name is unique value
SonarQube Server URL	http://localhost:9082/
	Default value is 'http://localhost:9000'
SonarQube account login	
	Default value is 'admin'
SonarQube account password	
	Default value is 'admin'

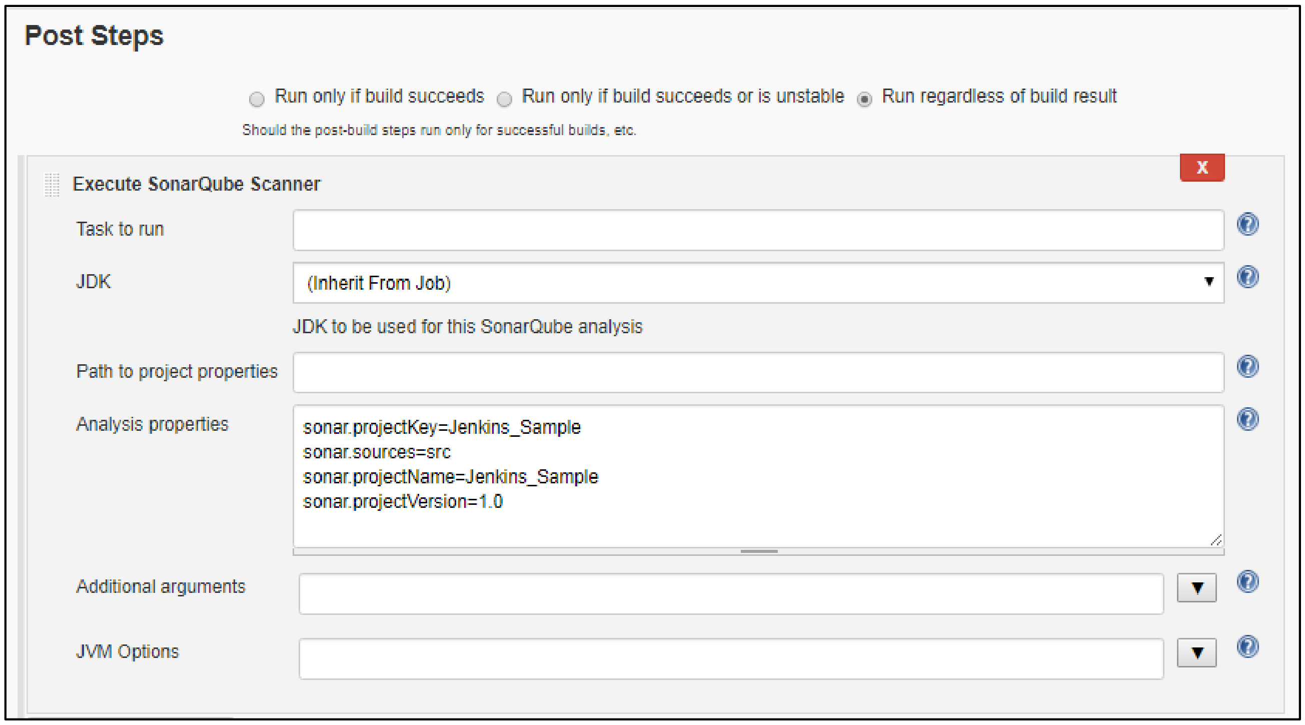
Server URL: URL in which SonarQube server is running

Note :-server should be in running condition

d. Local Configuration

I. Post Steps

Select->Execute SonarQube Scanner


Post Steps

( ) Run only if build succeeds ( ) Run only if build succeeds or is unstable [✓ Run regardless of build result]

Should the post-build steps run only for successful builds, etc.

Execute SonarQube Scanner

Task to run:

JDK: (Inherit From Job)

JDK to be used for this SonarQube analysis

Path to project properties:

Analysis properties:

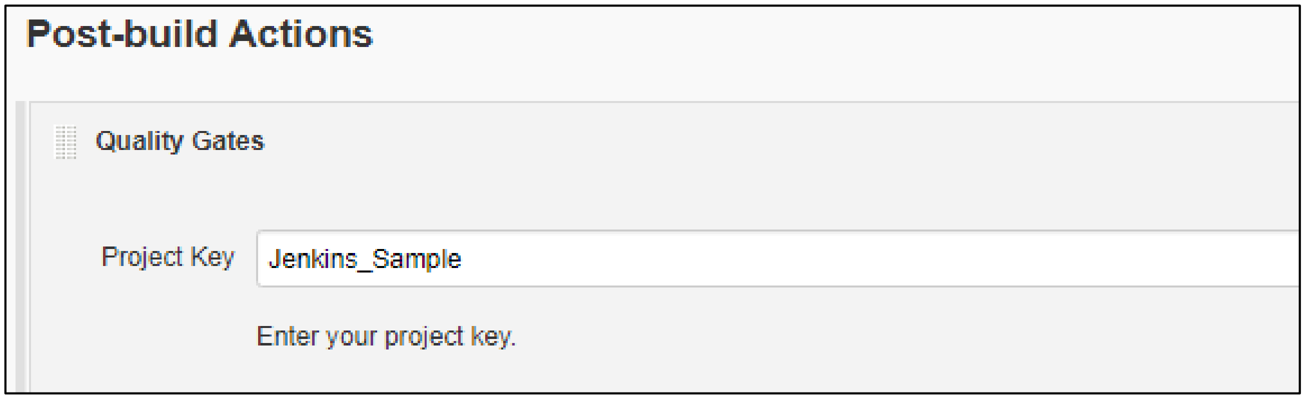
```
sonar.projectKey=Jenkins_Sample
sonar.sources=src
sonar.projectName=Jenkins_Sample
sonar.projectVersion=1.0
```

Additional arguments:

JVM Options:

Have to mention the analysis properties (things mentioned in the sonar_runner.properties file)

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


Post-build Actions

Quality Gates

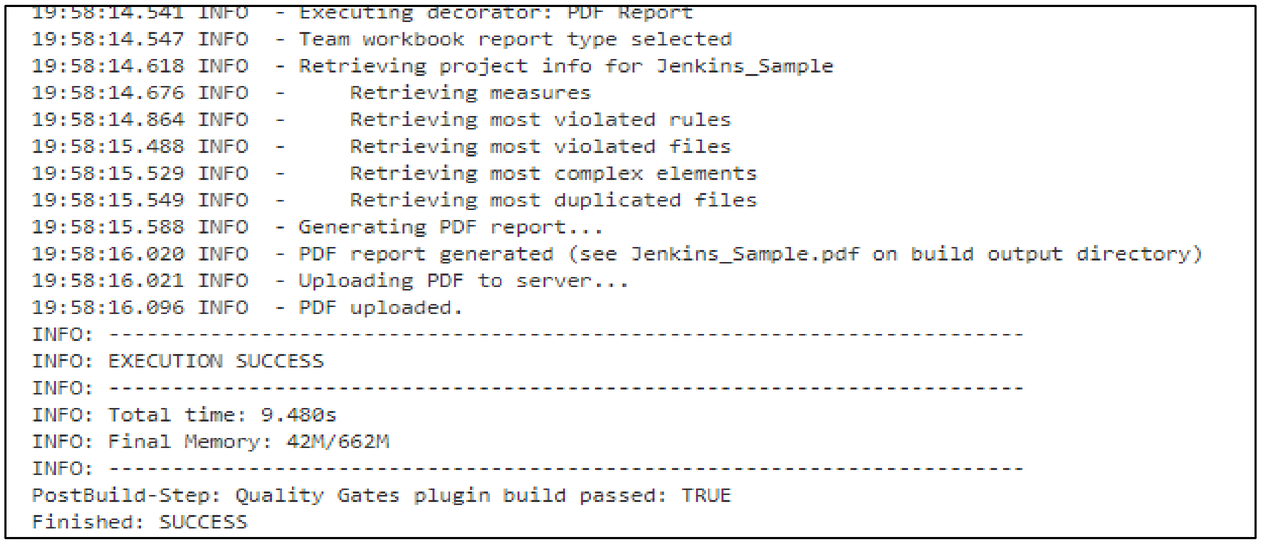
Project Key: Jenkins_Sample

Enter your project key.

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

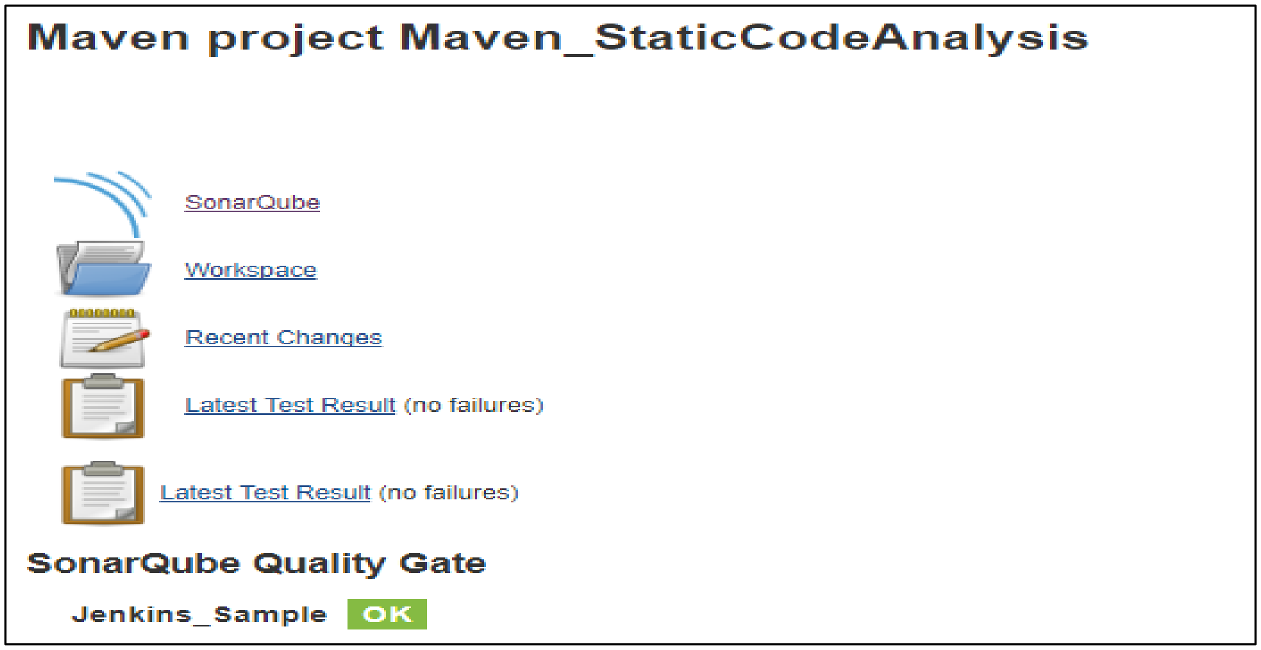
->Save->BuildNow

Console



```
19:58:14.541 INFO  - Executing decorator: PDF Report
19:58:14.547 INFO  - Team workbook report type selected
19:58:14.618 INFO  - Retrieving project info for Jenkins_Sample
19:58:14.676 INFO  -     Retrieving measures
19:58:14.864 INFO  -     Retrieving most violated rules
19:58:15.488 INFO  -     Retrieving most violated files
19:58:15.529 INFO  -     Retrieving most complex elements
19:58:15.549 INFO  -     Retrieving most duplicated files
19:58:15.588 INFO  - Generating PDF report...
19:58:16.020 INFO  - PDF report generated (see Jenkins_Sample.pdf on build output directory)
19:58:16.021 INFO  - Uploading PDF to server...
19:58:16.096 INFO  - PDF uploaded.
INFO: ------------------------------------------------------------------------
INFO: EXECUTION SUCCESS
INFO: ------------------------------------------------------------------------
INFO: Total time: 9.480s
INFO: Final Memory: 42M/662M
INFO: ------------------------------------------------------------------------
PostBuild-Step: Quality Gates plugin build passed: TRUE
Finished: SUCCESS
```

We can able to see the report by clicking sonarqube



Maven project Maven_StaticCodeAnalysis

SonarQube

Workspace

Recent Changes

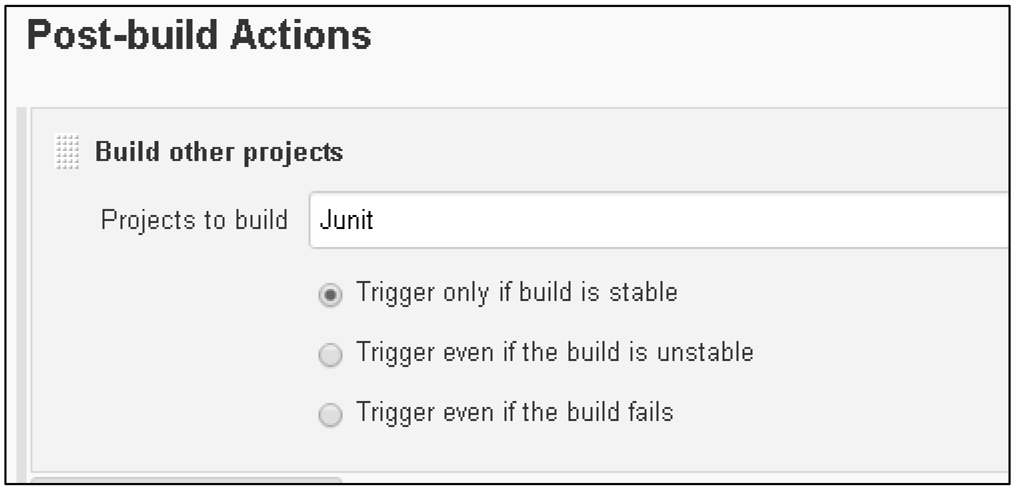
Latest Test Result (no failures)

Latest Test Result (no failures)

SonarQube Quality Gate

Jenkins_Sample OK

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


Post-build Actions

Build other projects

Projects to build: Junit

[✓ Trigger only if build is stable]

Trigger even if the build is unstable

Trigger even if the build fails

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

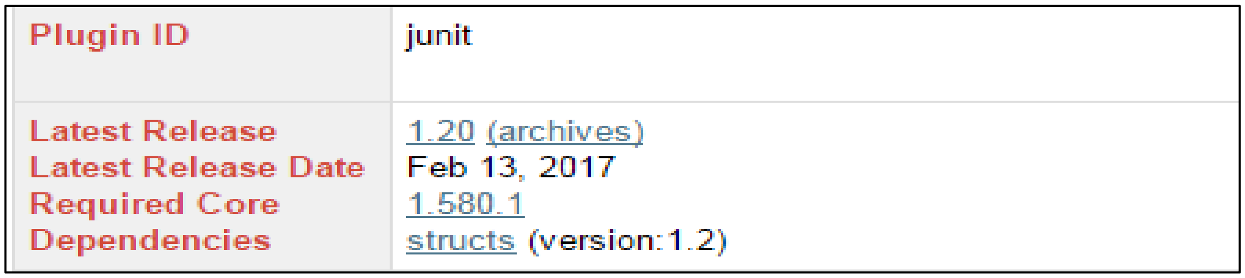
5. JUnit

a.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/JUnit+Plugin



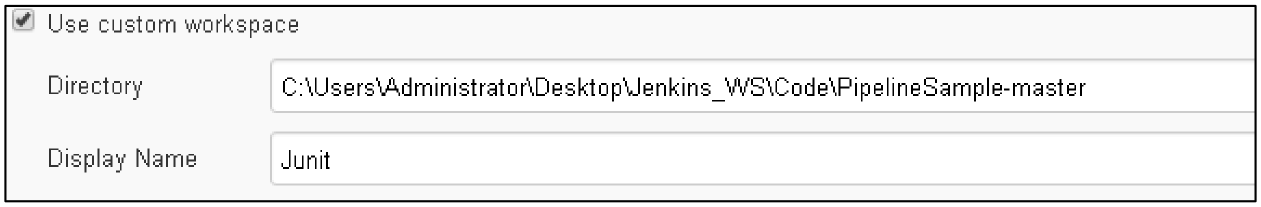
Plugin ID	junit
Latest Release	1.20 (archives)
Latest Release Date	Feb 13, 2017
Required Core	1.580.1
Dependencies	structs (version:1.2)

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

b. Local Configuration

Create New item->freestyle project

I.Under Advanced Project Options



☑ Use custom workspace

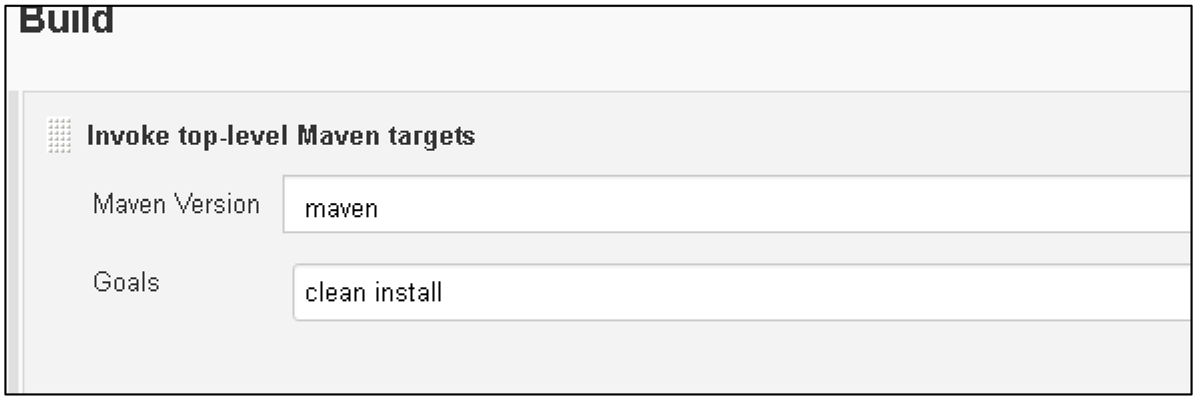
Directory: C:\Users\Administrator\Desktop\Jenkins_WS\Code\PipelineSample-master

Display Name: Junit

Directory : Mention the custom workspace directory

DisplayName :Display name of the project

II. Build



Build

Invoke top-level Maven targets

Maven Version: maven

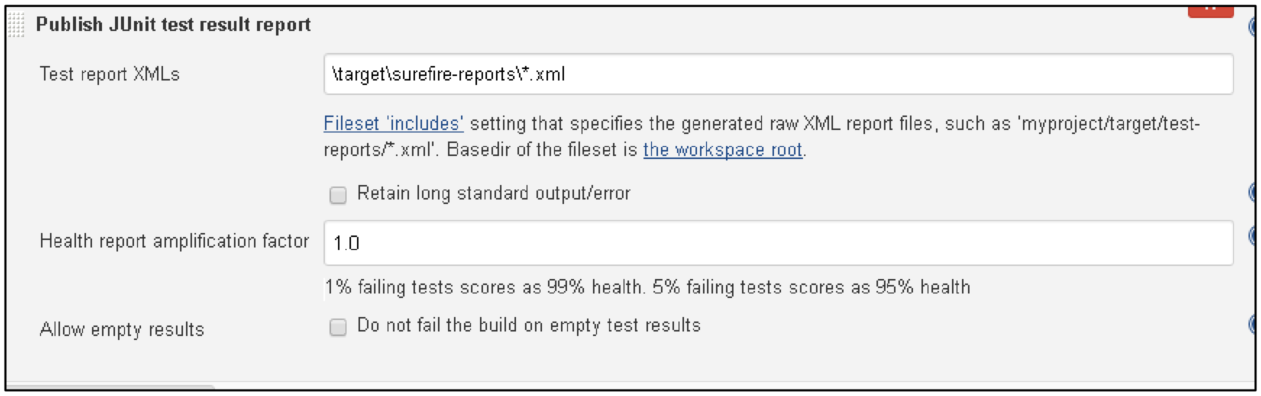
Goals: clean install

Select->invoke ant

Ant version : select the ant home in the dropdown

Targets : mention the target which has to be takes place(Target name as same as <target> specified in the junit.xml, **Note : refer junit document**)

IV.Post-build Actions


Publish JUnit test result report

Test report XMLs: \target\surefire-reports\*.xml

Fileset 'includes' setting that specifies the generated raw XML report files, such as 'myproject/target/test-reports/*.xml'. Basedir of the fileset is the workspace root.

Retain long standard output/error

Health report amplification factor: 1.0

1% failing tests scores as 99% health. 5% failing tests scores as 95% health

Allow empty results: Do not fail the build on empty test results

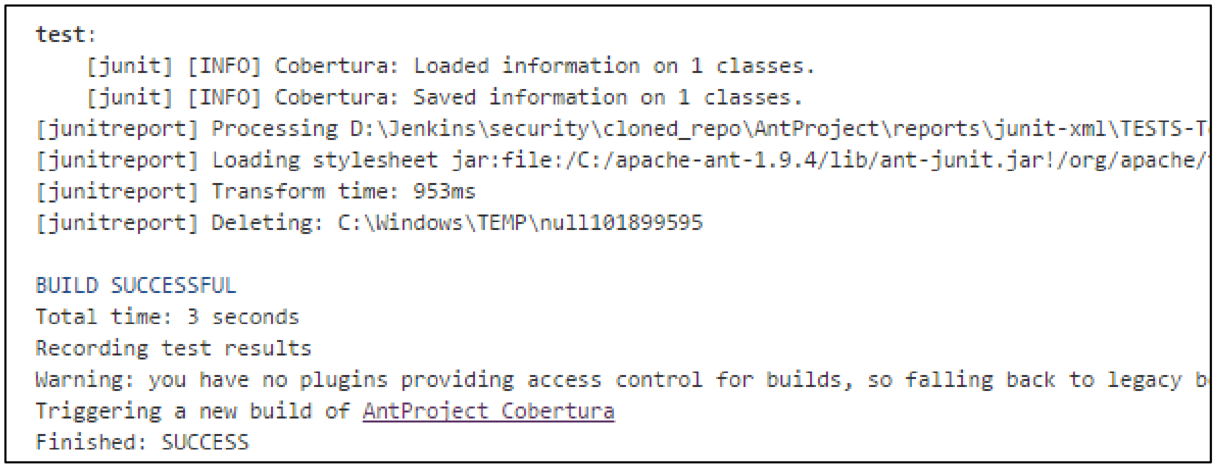
In **Post-build action** -> Publish Junit test results reports

a.Test repost XMLs –mention the space where the xml report found in the project structure

Fileset 'includes' setting that specifies the generated raw XML report files, such as 'myproject/target/test-reports/*.xml'. Basedir of the fileset is the workspace root.

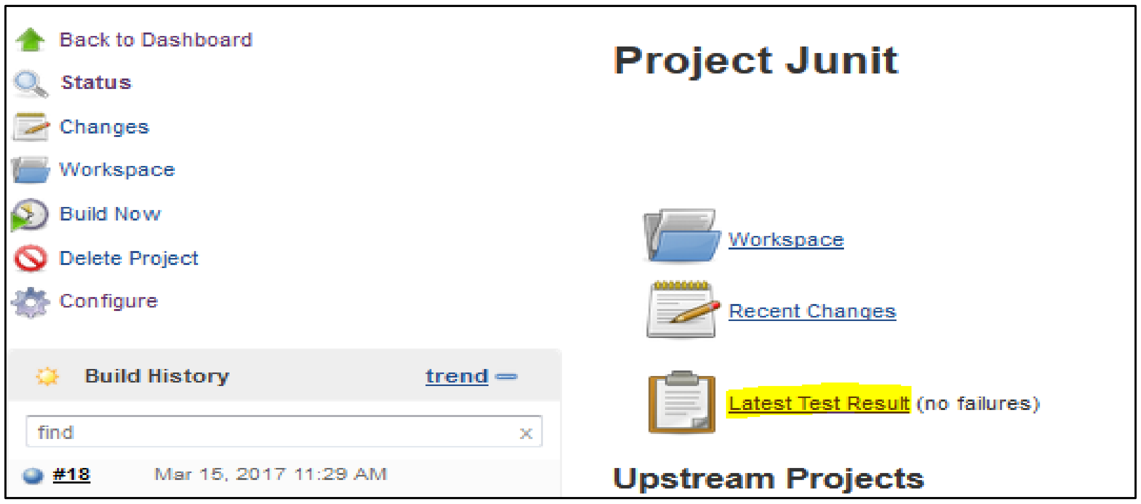
->Save->BuildNow

In Console


```
test:
    [junit] [INFO] Cobertura: Loaded information on 1 classes.
    [junit] [INFO] Cobertura: Saved information on 1 classes.
[junitreport] Processing D:\Jenkins\security\cloned_repo\AntProject\reports\junit-xml\TESTS-T
[junitreport] Loading stylesheet jar:file:/C:/apache-ant-1.9.4/lib/ant-junit.jar!/org/apache/
[junitreport] Transform time: 953ms
[junitreport] Deleting: C:\Windows\TEMP\null101899595

BUILD SUCCESSFUL
Total time: 3 seconds
Recording test results
Warning: you have no plugins providing access control for builds, so falling back to legacy b
Triggering a new build of AntProject Cobertura
Finished: SUCCESS
```

Build Successful!!!

Also we can get the latest test report by clicking it



#18 Mar 15, 2017 11:29 AM

Project Junit

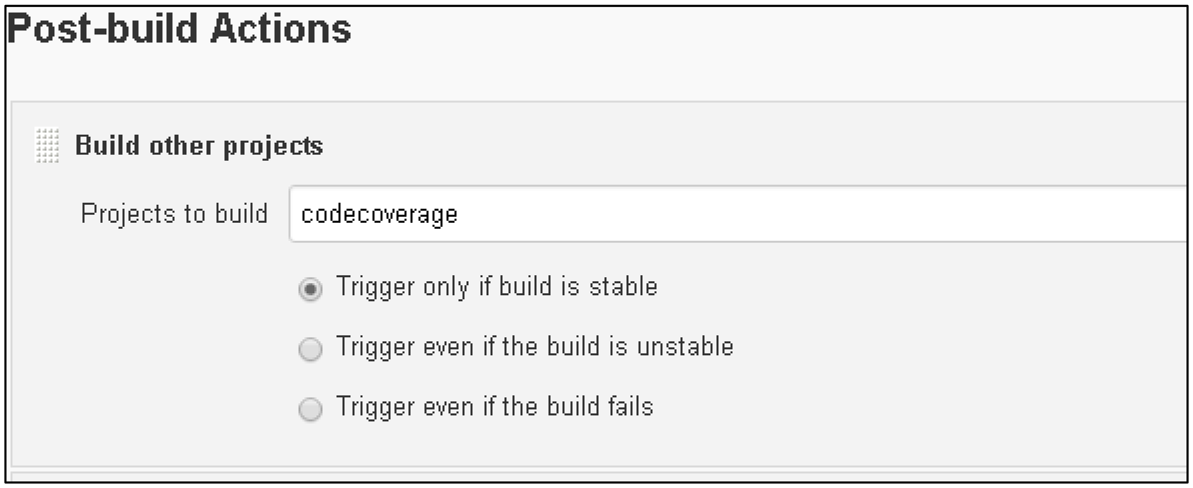
Workspace

Recent Changes

Latest Test Result (no failures)

Upstream Projects

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



Post-build Actions

Build other projects

Projects to build: codecoverage

[✓ Trigger only if build is stable]

Trigger even if the build is unstable

Trigger even if the build fails

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

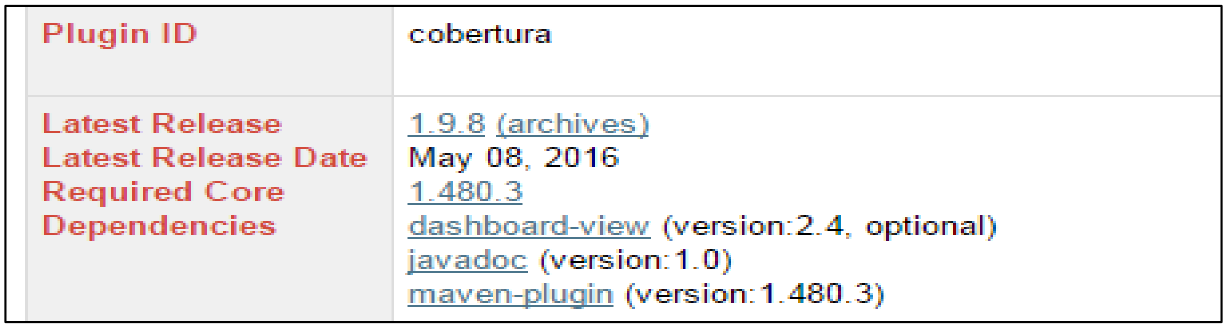
5. CodeCoverage

a. 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



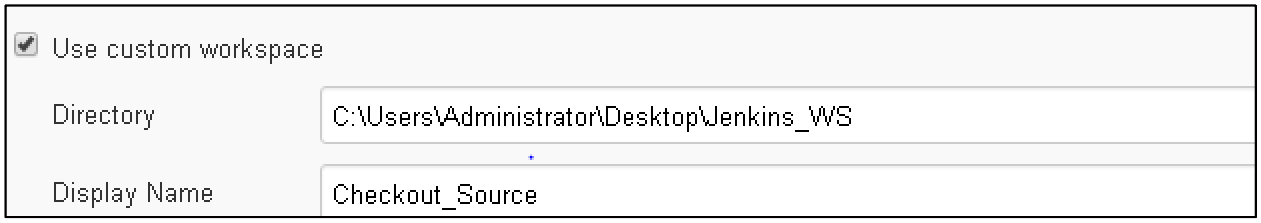
Plugin ID	cobertura
Latest Release Latest Release Date Required Core Dependencies	1.9.8 (archives) May 08, 2016 1.480.3 dashboard-view (version:2.4, optional) javadoc (version:1.0) maven-plugin (version:1.480.3)

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

b. Local Configuration

Create New item->freestyle project

I. Under Advanced Project Options



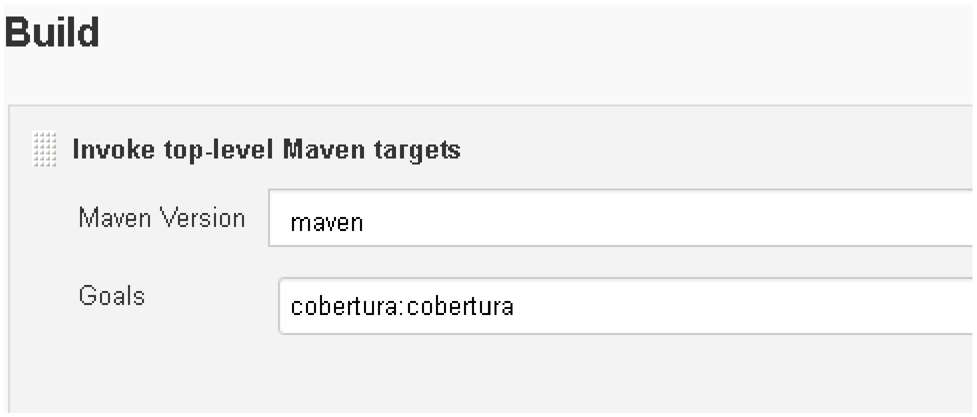
✓ Use custom workspace

Directory	C:\Users\Administrator\Desktop\Jenkins_WS
Display Name	Checkout_Source

Directory : Mention the custom workspace directory(cloned repository)

DisplayName :Display name of the project

II. Build



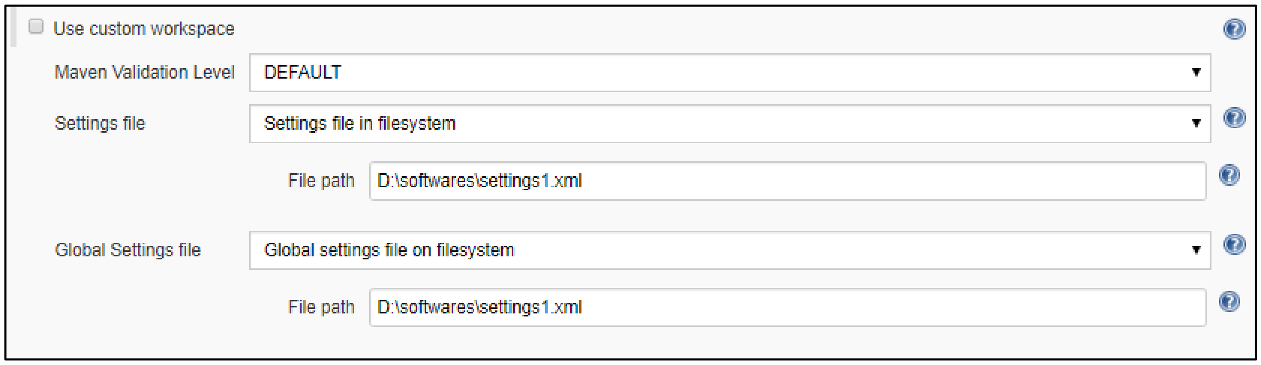
Build

Invoke top-level Maven targets

Maven Version	maven
Goals	cobertura:cobertura

Goals and options : cobertura:cobertura

Click **Advance** option in build and mention settings file details(If not open network)



Use custom workspace

Maven Validation Level: DEFAULT

Settings file: Settings file in filesystem

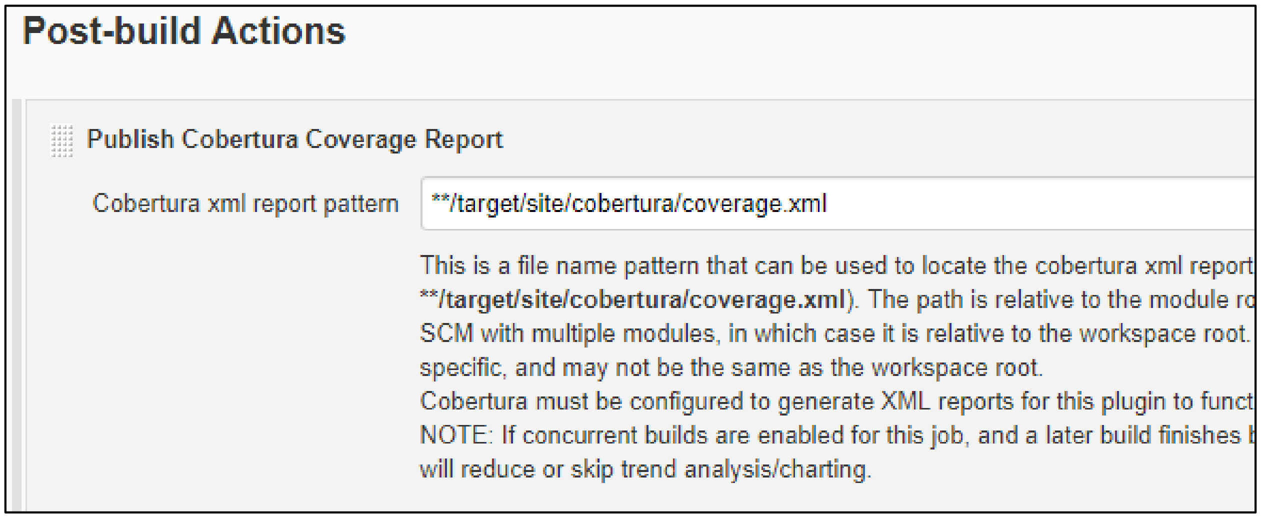
File path: D:\softwares\settings1.xml

Global Settings file: Global settings file on filesystem

File path: D:\softwares\settings1.xml

III. Post-build Actions

For XML report



Post-build Actions

Publish Cobertura Coverage Report

Cobertura xml report pattern: **/target/site/cobertura/coverage.xml

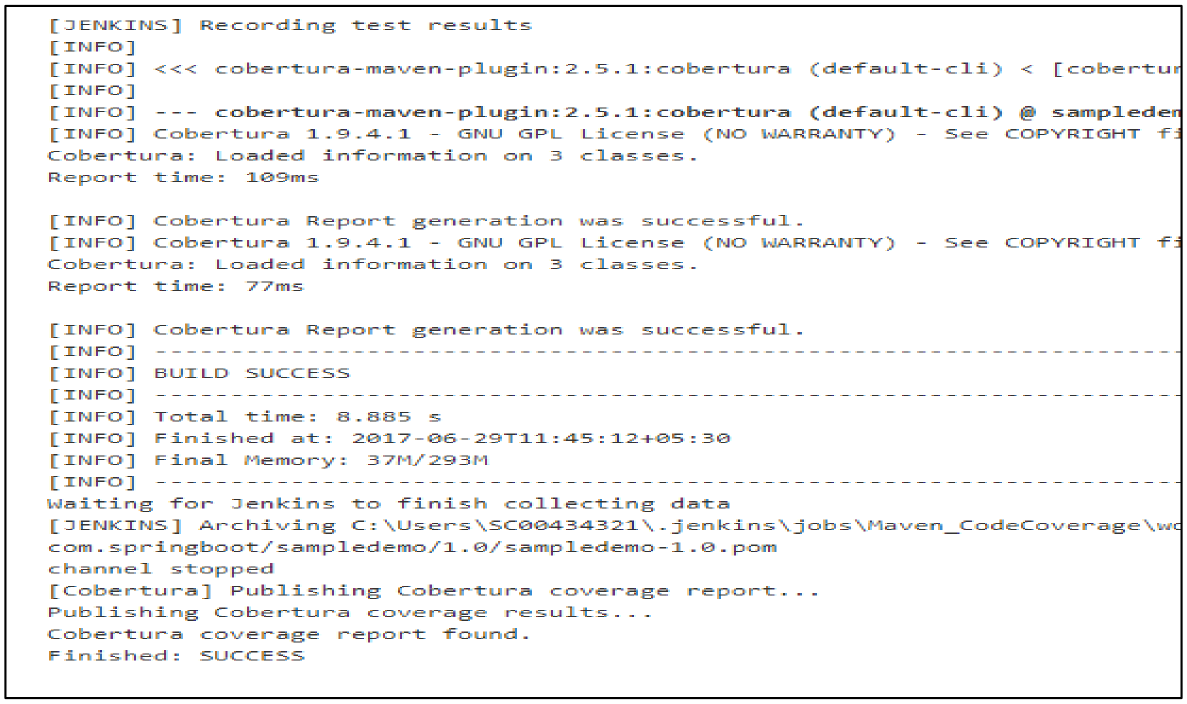
This is a file name pattern that can be used to locate the cobertura xml report
**/target/site/cobertura/coverage.xml). The path is relative to the module ro
SCM with multiple modules, in which case it is relative to the workspace root.
specific, and may not be the same as the workspace root.
Cobertura must be configured to generate XML reports for this plugin to funct
NOTE: If concurrent builds are enabled for this job, and a later build finishes b
will reduce or skip trend analysis/charting.

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

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

->Save->BuildNow

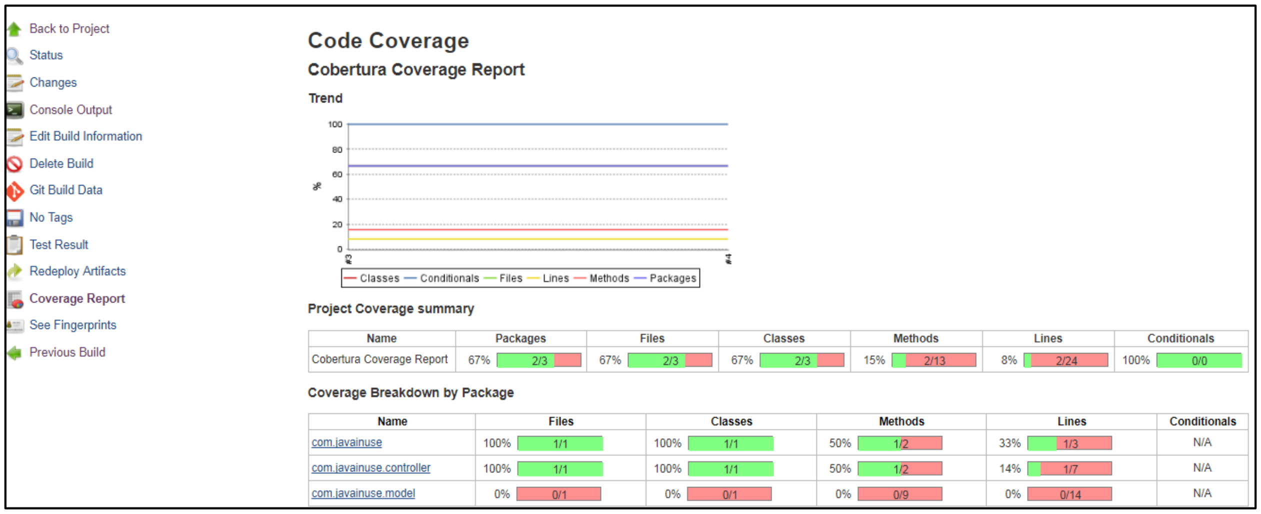
In Console



```
[JENKINS] Recording test results
[INFO]
[INFO] <<< cobertura-maven-plugin:2.5.1:cobertura (default-cli) < [cobertur
[INFO]
[INFO] --- cobertura-maven-plugin:2.5.1:cobertura (default-cli) @ sampleden
[INFO] Cobertura 1.9.4.1 - GNU GPL License (NO WARRANTY) - See COPYRIGHT fi
Cobertura: Loaded information on 3 classes.
Report time: 109ms

[INFO] Cobertura Report generation was successful.
[INFO] Cobertura 1.9.4.1 - GNU GPL License (NO WARRANTY) - See COPYRIGHT fi
Cobertura: Loaded information on 3 classes.
Report time: 77ms

[INFO] Cobertura Report generation was successful.
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 8.885 s
[INFO] Finished at: 2017-06-29T11:45:12+05:30
[INFO] Final Memory: 37M/293M
[INFO] ------------------------------------------------------------------------
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving C:\Users\SC00434321\.jenkins\jobs\Maven_CodeCoverage\wo
com.springboot/sampledemo/1.0/sampledemo-1.0.pom
channel stopped
[Cobertura] Publishing Cobertura coverage report...
Publishing Cobertura coverage results...
Cobertura coverage report found.
Finished: SUCCESS
```

Build Successful!!!

And report is shown in the graphical format



Project Coverage summary

Name	Packages	Files	Classes	Methods	Lines	Conditionals
Cobertura Coverage Report	67% 2/3	67% 2/3	67% 2/3	15% 2/13	8% 2/24	100% 0/0

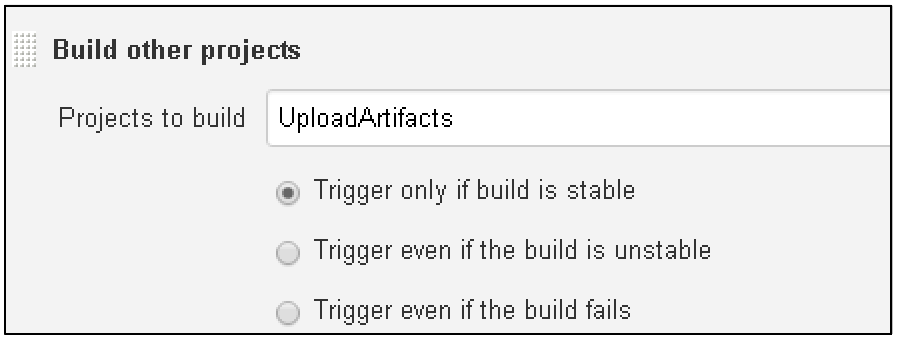
Coverage Breakdown by Package

Name	Files	Classes	Methods	Lines	Conditionals
com.javainuse	100% 1/1	100% 1/1	50% 1/2	33% 1/3	N/A
com.javainuse.controller	100% 1/1	100% 1/1	50% 1/2	14% 1/7	N/A
com.javainuse.model	0% 0/1	0% 0/1	0% 0/9	0% 0/14	N/A

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

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


Build other projects

Projects to build: UploadArtifacts

(● Trigger only if build is stable)
○ Trigger even if the build is unstable
○ Trigger even if the build fails

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

6. ArtifactUpload

a. 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

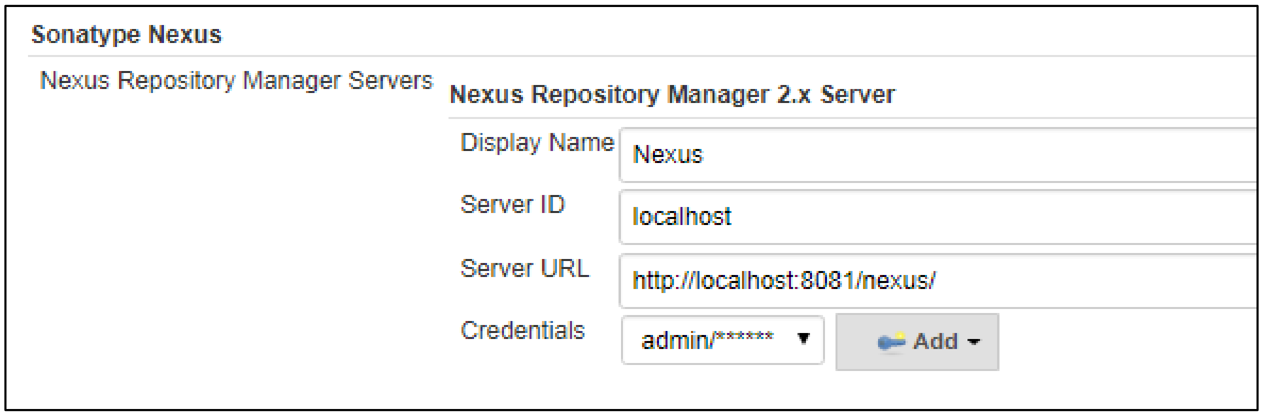
b. Configure System

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**


Sonatype Nexus

Nexus Repository Manager Servers

Nexus Repository Manager 2.x Server

Display Name	Nexus
Server ID	localhost
Server URL	http://localhost:8081/nexus/
Credentials	admin/******

Under **Sonatype nexus**

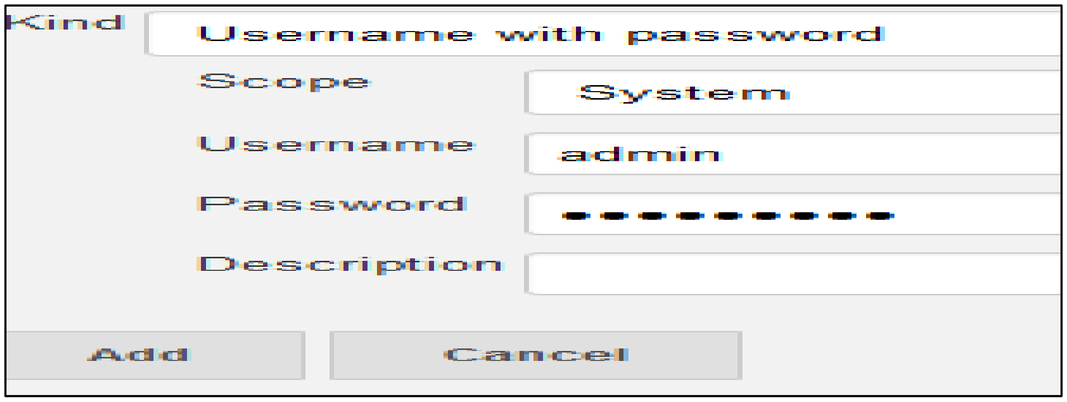
1.Display Name : Name to display for nexus

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


Kind	Username with password
Scope	System
Username	admin
Password	●●●●●●●●●
Description	

And click on add

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

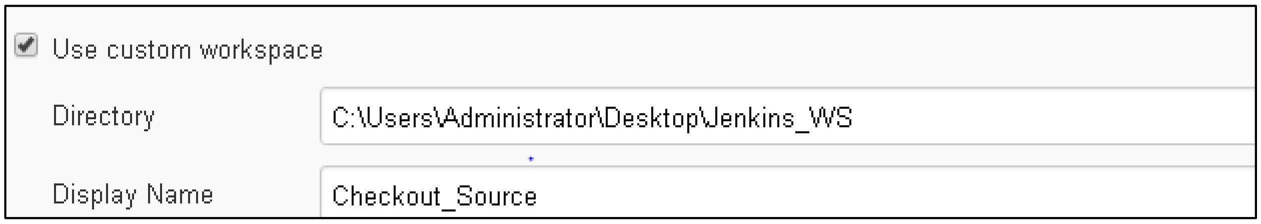
If connection established it will show like

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

c. **Local Configuration**

Create New item->freestyle project

I. **Under Advanced Project Options**



- [✓] Use custom workspace

Directory	C:\Users\Administrator\Desktop\Jenkins_WS
Display Name	Checkout_Source

Directory : Mention the custom workspace directory(cloned repository)
DisplayName :Display name of the project

II. **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

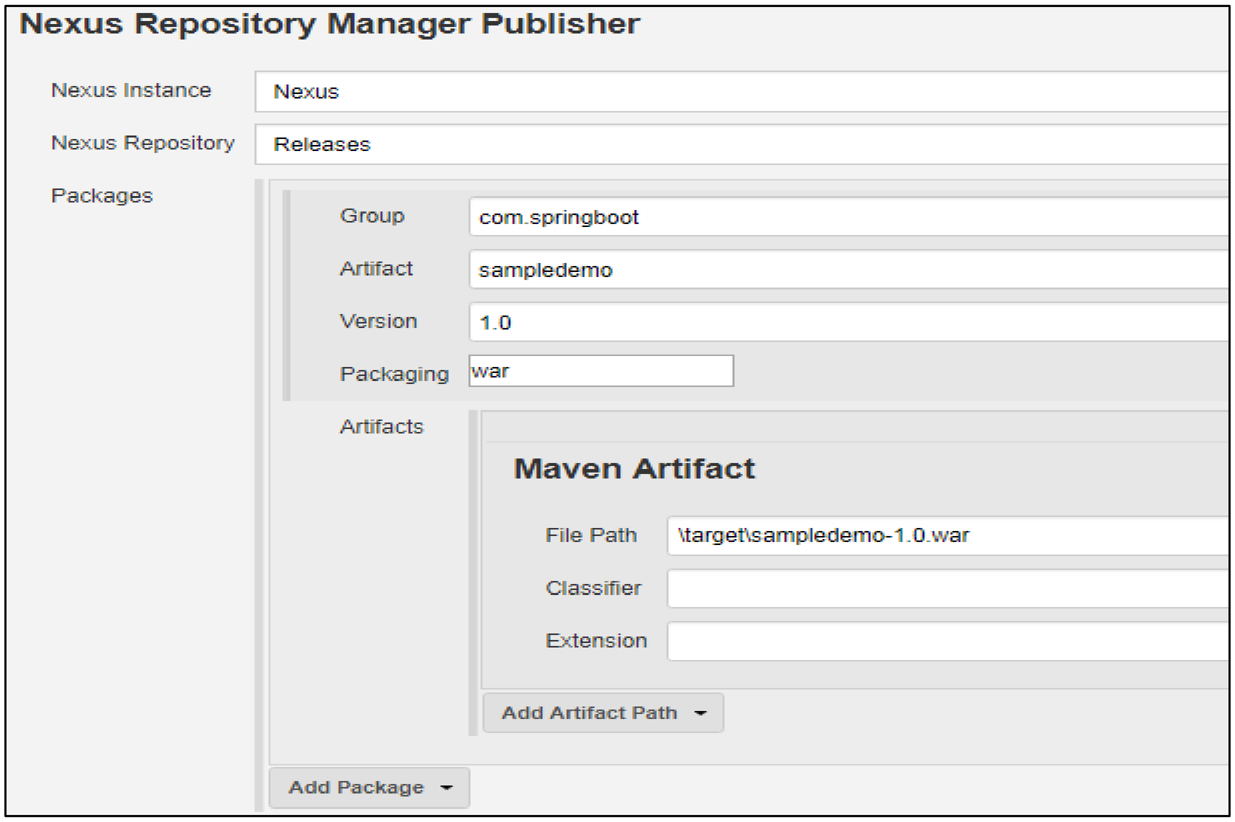
3. **Package**: select the package

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



Nexus Repository Manager Publisher

Nexus Instance: Nexus

Nexus Repository: Releases

Packages

Group: com.springboot

Artifact: sampledemo

Version: 1.0

Packaging: war

Artifacts

Maven Artifact

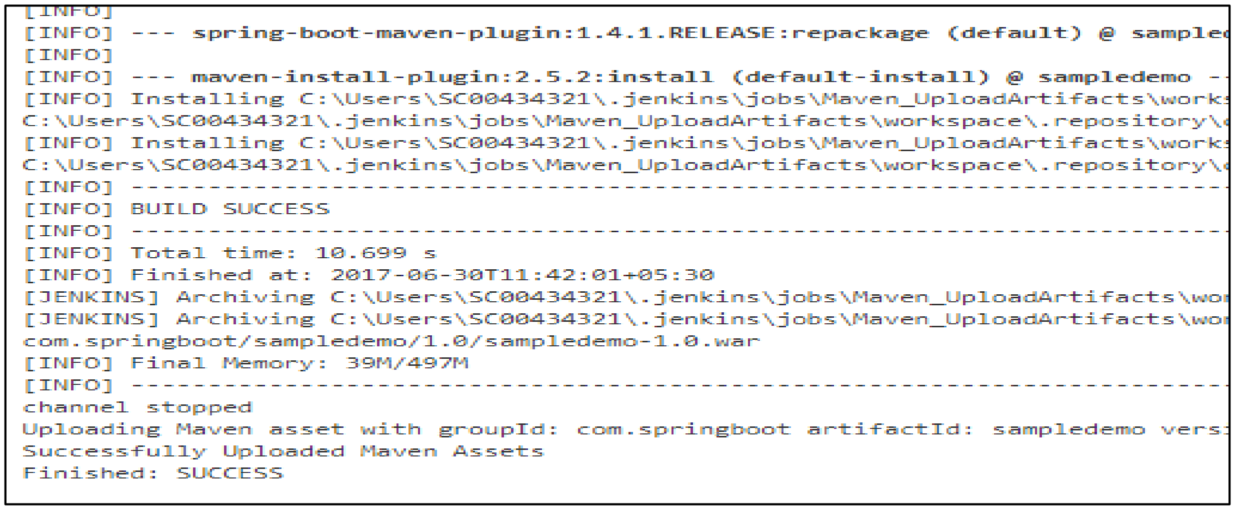
File Path: \target\sampledemo-1.0.war

Classifier:

Extension:

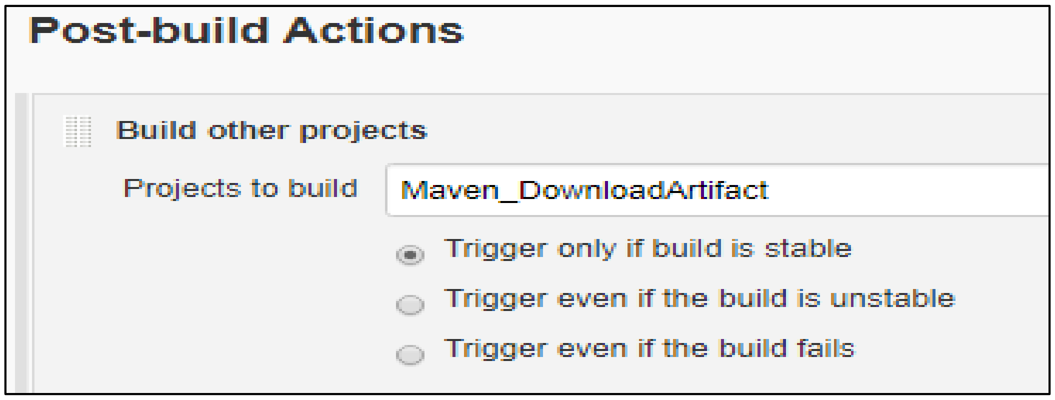
->Save->BuildNow

In Console



```
[INFO]
[INFO] --- spring-boot-maven-plugin:1.4.1.RELEASE:repackage (default) @ samplec
[INFO]
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ sampledemo --
[INFO] Installing C:\Users\SC00434321\.jenkins\jobs\Maven_UploadArtifacts\works
C:\Users\SC00434321\.jenkins\jobs\Maven_UploadArtifacts\workspace\.repository\c
[INFO] Installing C:\Users\SC00434321\.jenkins\jobs\Maven_UploadArtifacts\works
C:\Users\SC00434321\.jenkins\jobs\Maven_UploadArtifacts\workspace\.repository\c
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 10.699 s
[INFO] Finished at: 2017-06-30T11:42:01+05:30
[JENKINS] Archiving C:\Users\SC00434321\.jenkins\jobs\Maven_UploadArtifacts\wor
[JENKINS] Archiving C:\Users\SC00434321\.jenkins\jobs\Maven_UploadArtifacts\wor
com.springboot/sampledemo/1.0/sampledemo-1.0.war
[INFO] Final Memory: 39M/497M
[INFO] ------------------------------------------------------------------------
channel stopped
Uploading Maven asset with groupId: com.springboot artifactId: sampledemo vers
Successfully Uploaded Maven Assets
Finished: SUCCESS
```

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


Post-build Actions

Build other projects

Projects to build: Maven_DownloadArtifact

- [✓ Trigger only if build is stable]
- Trigger even if the build is unstable
- Trigger even if the build fails

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

7. Maven_Nexus_DownloadArtifacts

a. 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

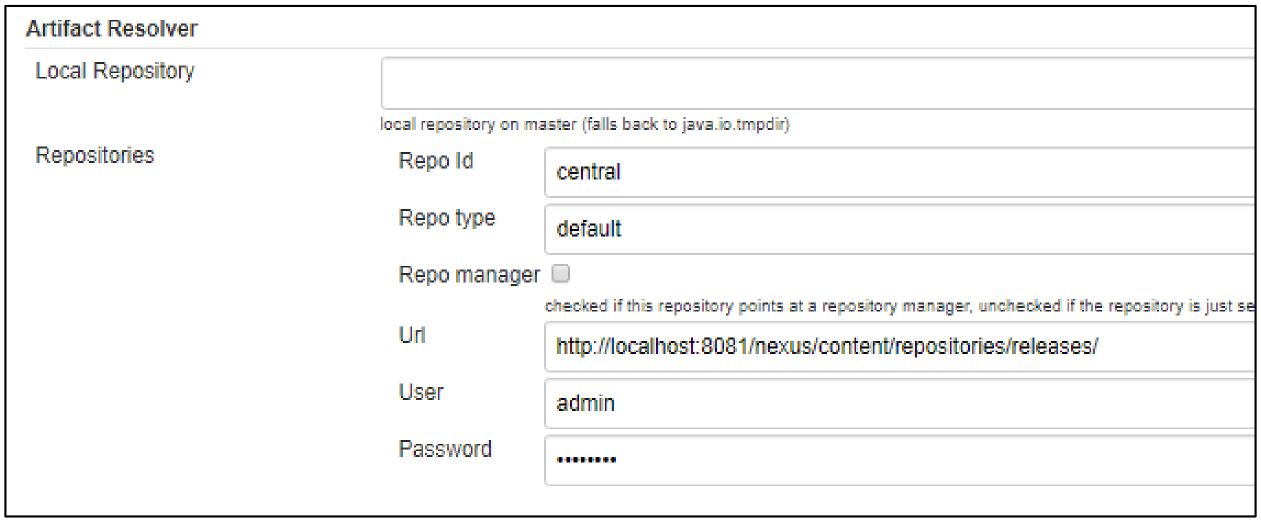
b. Configure System

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


Artifact Resolver

Local Repository		
	local repository on master (falls back to java.io.tmpdir)	
Repositories	Repo Id	central
	Repo type	default
	Repo manager	☐
		checked if this repository points at a repository manager, unchecked if the repository is just se
	Url	http://localhost:8081/nexus/content/repositories/releases/
	User	admin
	Password	••••••••

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)

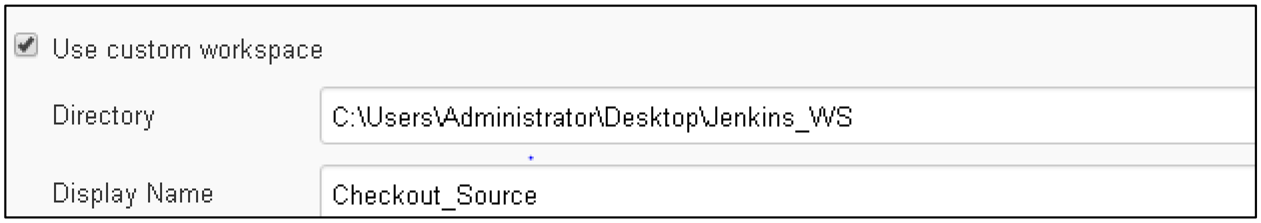
c)Url-path where your files stores inside the repository

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

c. Local Configuration

Create New item->freestyle project

I. Under Advanced Project Options

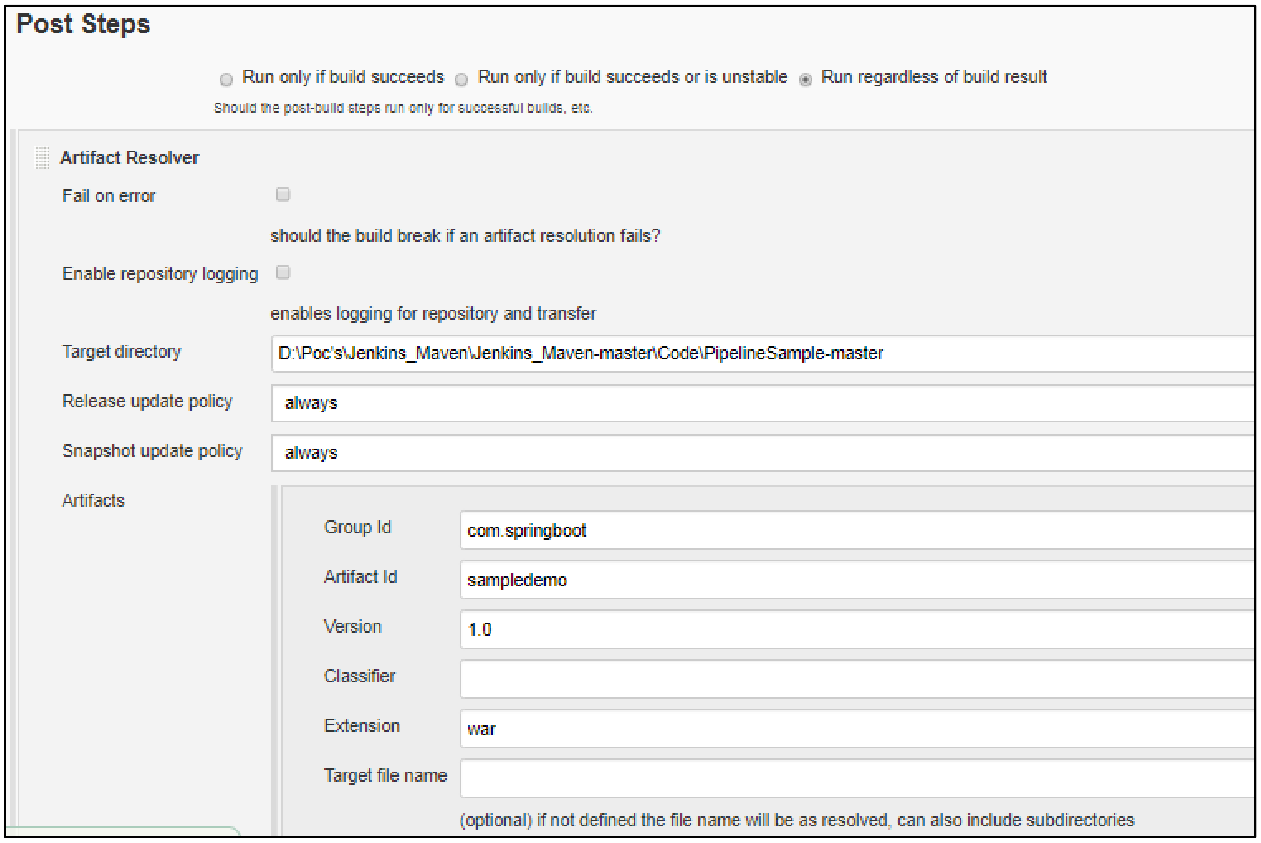

☑ Use custom workspace

Directory	C:\Users\Administrator\Desktop\Jenkins_WS
Display Name	Checkout_Source

Directory : Mention the custom workspace directory(cloned repository)
DisplayName :Display name of the project

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

Add build step -> Artifact Resolver


Post Steps

Run only if build succeeds ( ) Run only if build succeeds or is unstable ( ) Run regardless of build result (●)

Should the post-build steps run only for successful builds, etc.

Artifact Resolver

Fail on error	☐ should the build break if an artifact resolution fails?
Enable repository logging	☐ enables logging for repository and transfer
Target directory	D:\Poc's\Jenkins_Maven\Jenkins_Maven-master\Code\PipelineSample-master
Release update policy	always
Snapshot update policy	always

Artifacts

Group Id	com.springboot
Artifact Id	sampledemo
Version	1.0
Classifier	
Extension	war
Target file name	(optional) if not defined the file name will be as resolved, can also include subdirectories

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

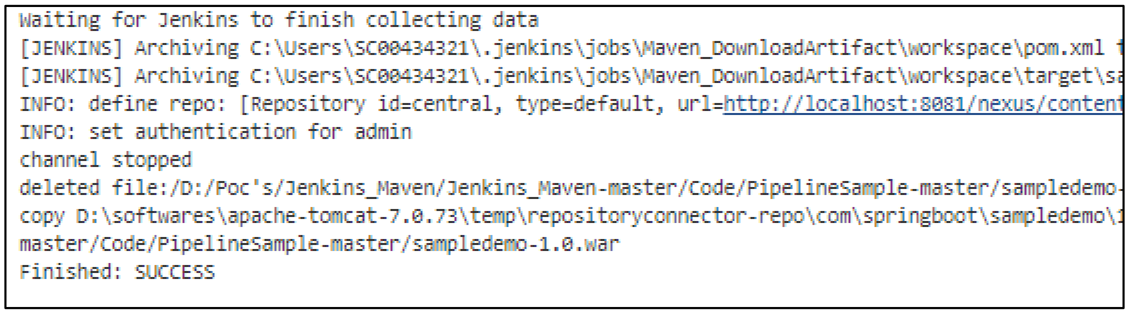
1.Target directory-the space where the downloaded file will be copied

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

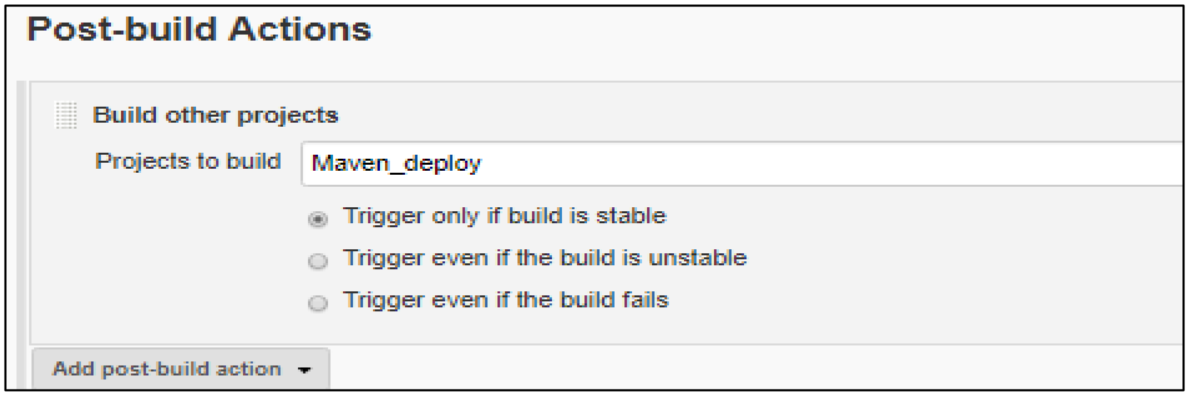
->Save->BuildNow

In Console



```
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving C:\Users\SC00434321\.jenkins\jobs\Maven_DownloadArtifact\workspace\pom.xml t
[JENKINS] Archiving C:\Users\SC00434321\.jenkins\jobs\Maven_DownloadArtifact\workspace\target\sa
INFO: define repo: [Repository id=central, type=default, url=http://localhost:8081/nexus/content
INFO: set authentication for admin
channel stopped
deleted file:/D:/Poc's/Jenkins_Maven/Jenkins_Maven-master/Code/PipelineSample-master/sampledemo-
copy D:\softwares\apache-tomcat-7.0.73\temp\repositoryconnector-repo\com\springboot\sampledemo\1
master/Code/PipelineSample-master/sampledemo-1.0.war
Finished: SUCCESS
```

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



Post-build Actions

Build other projects

Projects to build: Maven_deploy

[✓ Trigger only if build is stable]
Trigger even if the build is unstable
Trigger even if the build fails

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

8. A. Deploy war into the Container

a. Plugins

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

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

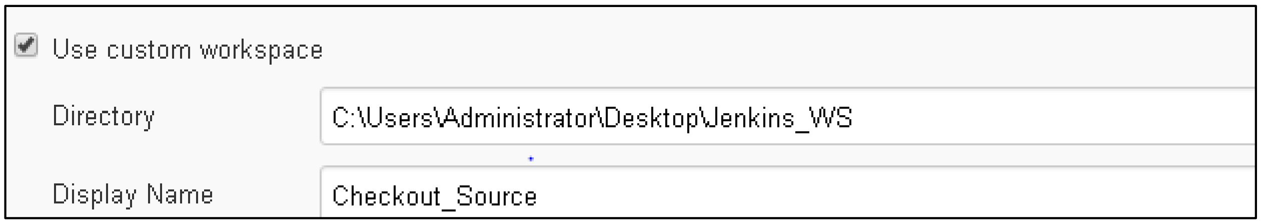
From this link https://wiki.jenkins-ci.org/display/JENKINS/Deploy+Plugin

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

b. Local Configuration

Create New item->freestyle project

I. Under Advanced Project Options



Use custom workspace ✓

Directory C:\Users\Administrator\Desktop\Jenkins_WS

Display Name Checkout_Source

Directory : Mention the custom workspace directory(cloned repository)
DisplayName :Display name of the project

I. 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

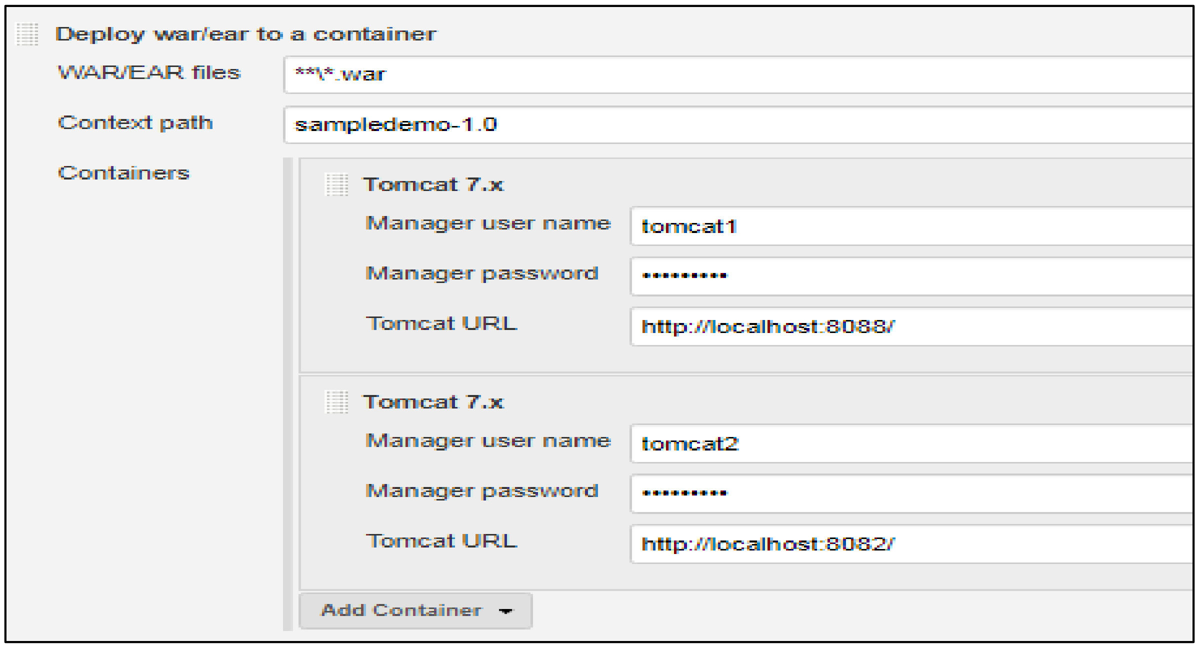
a. manager username-username of the tomcat server

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


Deploy war/ear to a container

WAR/EAR files: ***.war

Context path: sampledemo-1.0

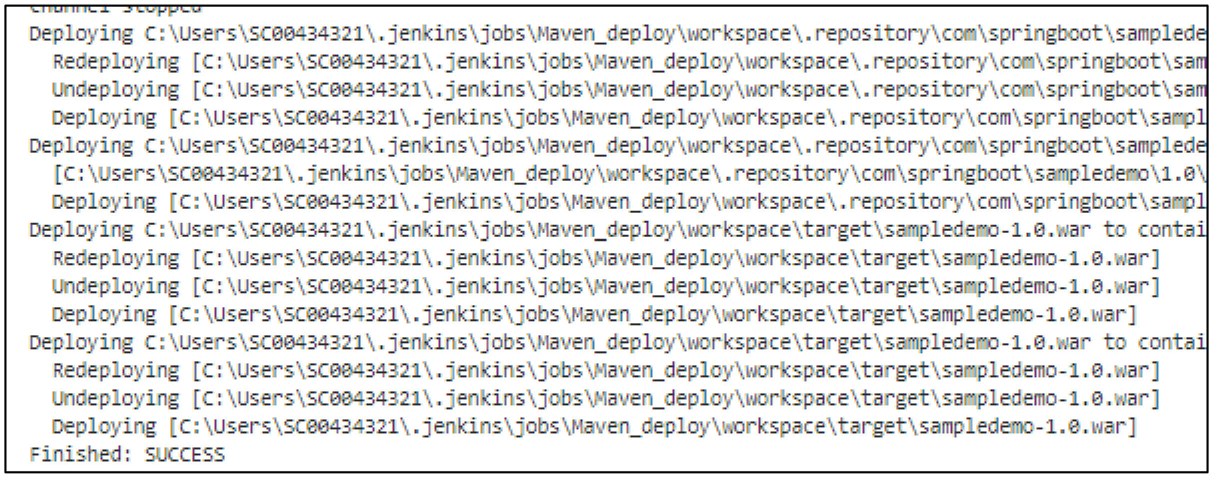
Containers:

Tomcat 7.x
Manager user name: tomcat1
Manager password: ••••••••••
Tomcat URL: http://localhost:8088/

Tomcat 7.x
Manager user name: tomcat2
Manager password: ••••••••••
Tomcat URL: http://localhost:8082/

->Save->BuildNow

In Console


```
Deploying C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\.repository\com\springboot\samplede
  Redeploying [C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\.repository\com\springboot\sam
  Undeploying [C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\.repository\com\springboot\sam
  Deploying [C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\.repository\com\springboot\sampl
Deploying C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\.repository\com\springboot\samplede
  [C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\.repository\com\springboot\sampledemo\1.0\
  Deploying [C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\.repository\com\springboot\sampl
Deploying C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\target\sampledemo-1.0.war to contai
  Redeploying [C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\target\sampledemo-1.0.war]
  Undeploying [C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\target\sampledemo-1.0.war]
  Deploying [C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\target\sampledemo-1.0.war]
Deploying C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\target\sampledemo-1.0.war to contai
  Redeploying [C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\target\sampledemo-1.0.war]
  Undeploying [C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\target\sampledemo-1.0.war]
  Deploying [C:\Users\SC00434321\.jenkins\jobs\Maven_deploy\workspace\target\sampledemo-1.0.war]
Finished: SUCCESS
```

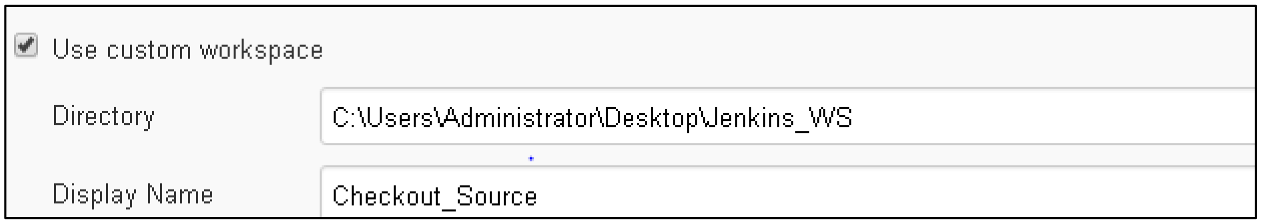
Project Steps completed Successfully!!!

8.B. Deploy SpringBoot Application

a) Local Configuration

Create New item->freestyle project

I. **Under Advanced Project Options**



Use custom workspace [✓]

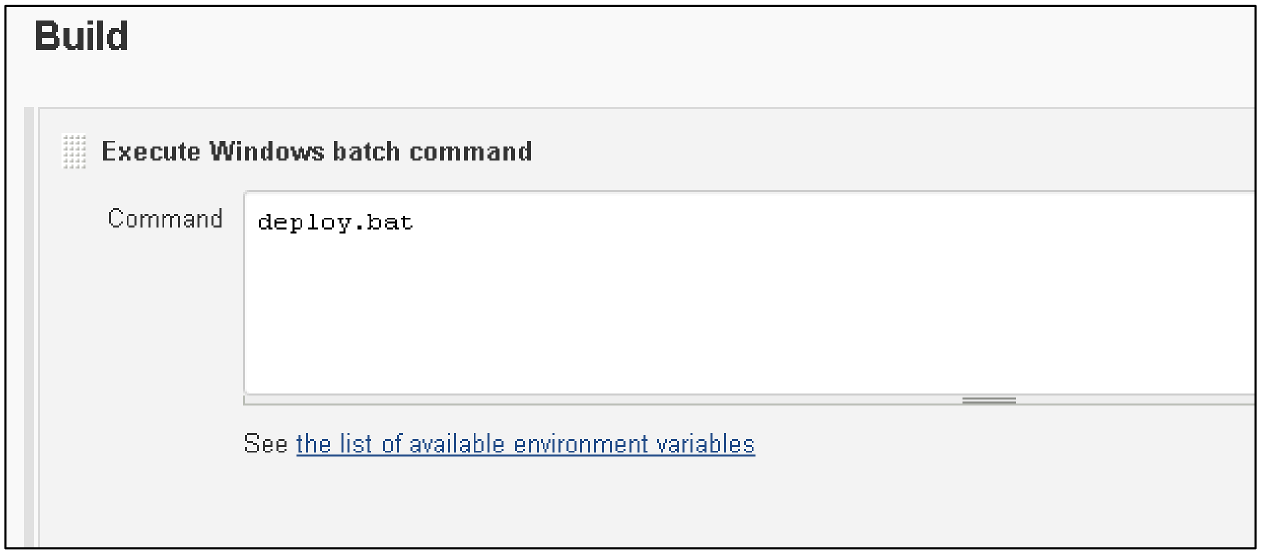
Directory: C:\Users\Administrator\Desktop\Jenkins_WS

Display Name: Checkout_Source

Directory : Mention the custom workspace directory(Directory where the downloaded war file stored)

DisplayName :Display name of the project

II. Build



Build

Execute Windows batch command

Command

```
deploy.bat
```

See the list of available environment variables

Execute Windows batch command

Command: execute batch file

Batch file content:

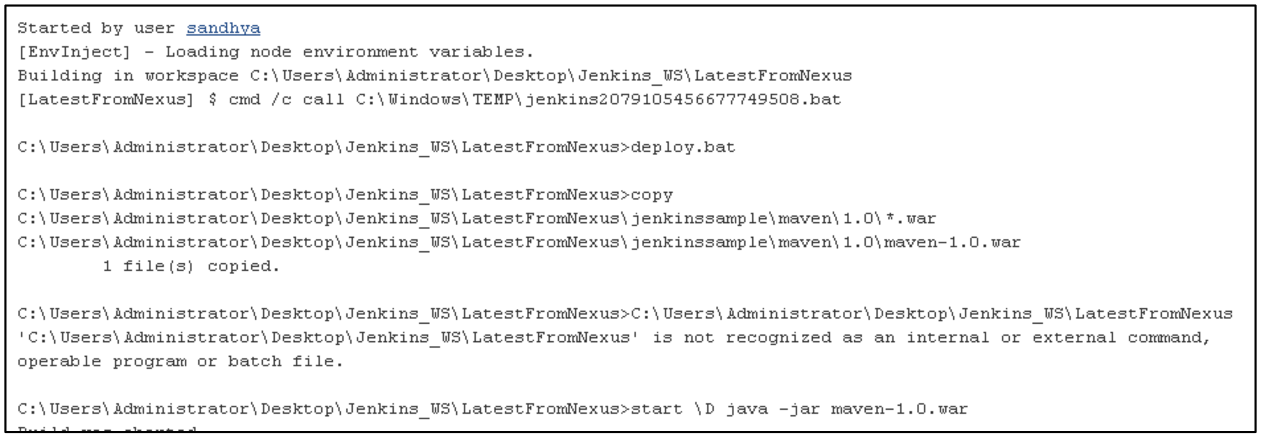
copy C:\Users\Administrator\Desktop\Jenkins_WS\LatestFromNexus\jenkinssample\maven\1.0\*.war

C:\Users\Administrator\Desktop\Jenkins_WS\LatestFromNexus

java -jar maven-1.0.war

->Save->BuildNow

In Console



```
Started by user sandhya
[EnvInject] - Loading node environment variables.
Building in workspace C:\Users\Administrator\Desktop\Jenkins_WS\LatestFromNexus
[LatestFromNexus] $ cmd /c call C:\Windows\TEMP\jenkins2079105456677749508.bat

C:\Users\Administrator\Desktop\Jenkins_WS\LatestFromNexus>deploy.bat

C:\Users\Administrator\Desktop\Jenkins_WS\LatestFromNexus>copy
C:\Users\Administrator\Desktop\Jenkins_WS\LatestFromNexus\jenkinssample\maven\1.0\*.war
C:\Users\Administrator\Desktop\Jenkins_WS\LatestFromNexus\jenkinssample\maven\1.0\maven-1.0.war
        1 file(s) copied.

C:\Users\Administrator\Desktop\Jenkins_WS\LatestFromNexus>C:\Users\Administrator\Desktop\Jenkins_WS\LatestFromNexus
'C:\Users\Administrator\Desktop\Jenkins_WS\LatestFromNexus' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\Administrator\Desktop\Jenkins_WS\LatestFromNexus>start \D java -jar maven-1.0.war
```

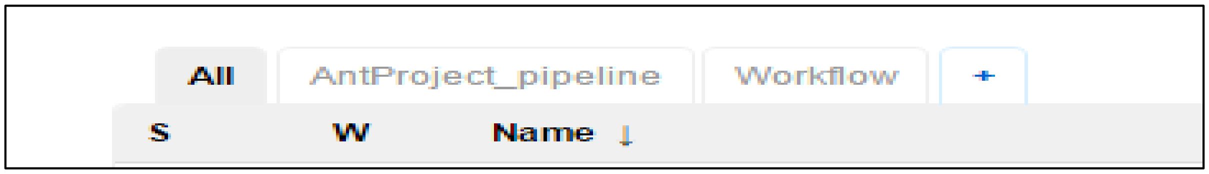
Project Steps completed Successfully!!!

<u>Jenkins Pipeline</u>

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

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

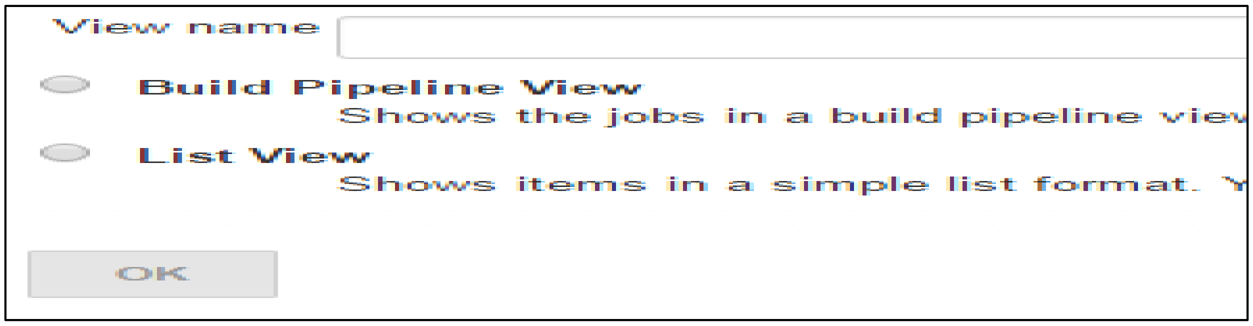
In Jenkins UI,Click on ”+” sign in order to add pipelin



New pipeline

View name: name of the pipeline

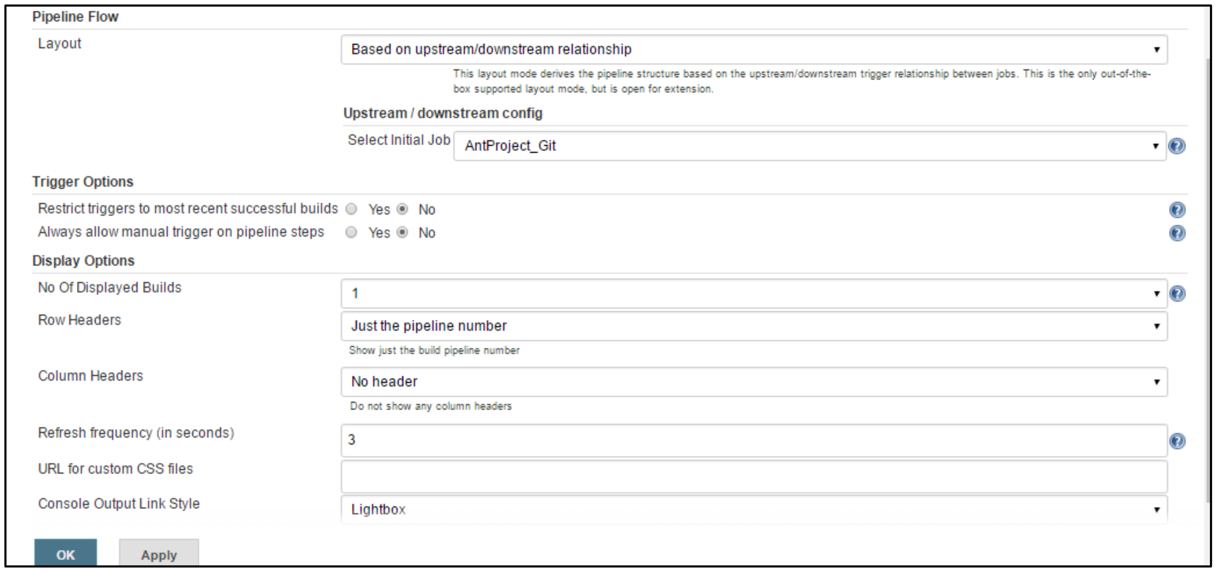
click on build pipeline view


View name

Build Pipeline View
Shows the jobs in a build pipeline view

List View
Shows items in a simple list format. Y

Click on OK


Pipeline Flow

Layout: Based on upstream/downstream relationship

This layout mode derives the pipeline structure based on the upstream/downstream trigger relationship between jobs. This is the only out-of-the-box supported layout mode, but is open for extension.

Upstream / downstream config

Select Initial Job: AntProject_Git

Trigger Options

Restrict triggers to most recent successful builds: Yes [✓ No]

Always allow manual trigger on pipeline steps: Yes [✓ No]

Display Options

No Of Displayed Builds: 1

Row Headers: Just the pipeline number

Show just the build pipeline number

Column Headers: No header

Do not show any column headers

Refresh frequency (in seconds): 3

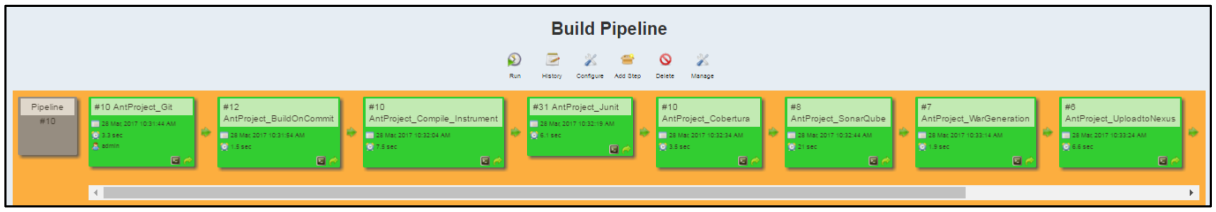
URL for custom CSS files:

Console Output Link Style: Lightbox

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

And click OK

Pipeline will appear ->click ->run


Build Pipeline

Pipeline #10

#10 AntProject_Git
28 Mar, 2017 10:31:44 AM
3.3 sec
admin

#12 AntProject_BuildOnCommit
28 Mar, 2017 10:31:54 AM
1.5 sec

#10 AntProject_Compile_Instrument
28 Mar, 2017 10:32:04 AM
7.5 sec

#31 AntProject_Junit
28 Mar, 2017 10:32:19 AM
6.1 sec

#10 AntProject_Cobertura
28 Mar, 2017 10:32:34 AM
3.5 sec

#8 AntProject_SonarQube
28 Mar, 2017 10:32:44 AM
21 sec

#7 AntProject_WarGeneration
28 Mar, 2017 10:33:14 AM
1.9 sec

#6 AntProject_UploadtoNexus
28 Mar, 2017 10:33:24 AM
6.6 sec

Pipeline executed successfully!!!