**JENKINS**

* **Youtube :** [Raghav Pal](https://www.youtube.com/channel/UCTt7pyY-o0eltq14glaG5dg) <https://www.youtube.com/watch?v=89yWXXIOisk>
* My Installed Jenkins is installed on den01cvy: <http://den01cvy.us.oracle.com:8080/>
* JENKINS\_HOME = /scratch/supvenka/Jenkins/jenkinshome
* export JAVA\_HOME=/scratch/supvenka/software/java/jdk1.8.0\_45
* /usr/bin/java 1.8.0\_191
* Apache tomcat : /scratch/Tools/apacahe-tomcat-8.5.40
* Admin password is stored in file under $JENKINS\_HOME/secrets/initialAdminPassword

TOMCAT = /scratch/Tools

* Jenkins is a Java application. Hence it is platform independent.
* Used in Continuous integration and continuous delivery.
* When any code is checked in Jenkins picks up the latest code and builds it using Jenkins
* If there are any issues then it will trigger a notification about any issues.
* We can integrate our unit test , acceptance tests with Jenkins.
* When the build is successful Jenkins will trigger tests that is integrated and hence when the tests run we will know if any check-in broke or encountered bugs etc

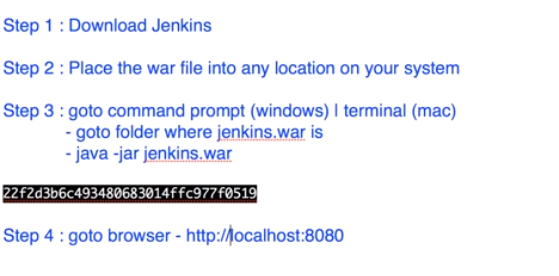
Please change the JENKINS\_HOME: Since the above will download all the binaries/libraries in the user’s home folder.

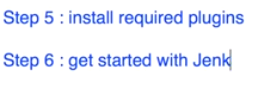
So set the JENKINS\_HOME

export JENKINS\_HOME=/scratch/supvenka/Jenkins/jenkinshome

ctrl C to kill the previous jenkins.war

Then run java –jar Jenkins.war





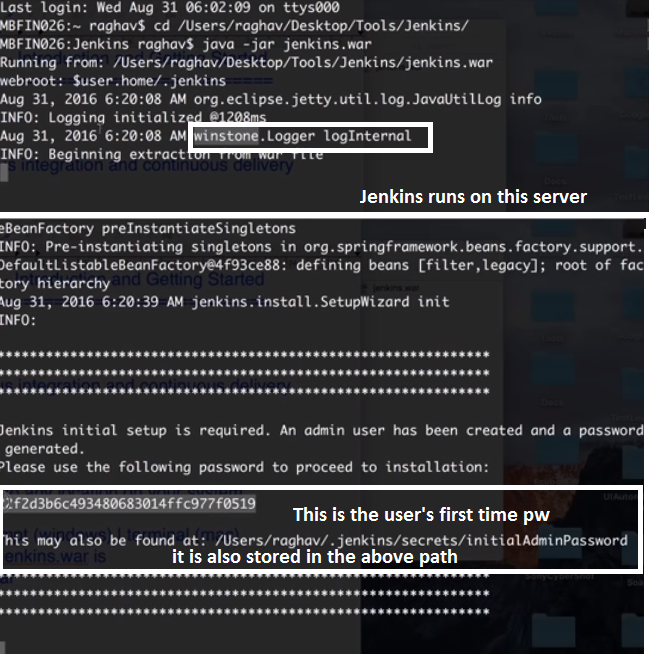
Before we execute java –jar Jenkins:

Please change the JENKINS\_HOME: Since the above will download all the binaries/libraries in the user’s home folder.

So set the JENKINS\_HOME

Export JENKINS\_HOME=/scratch/supvenka/Jenkins/jenkinshome

**Then run java –jar Jenkins**

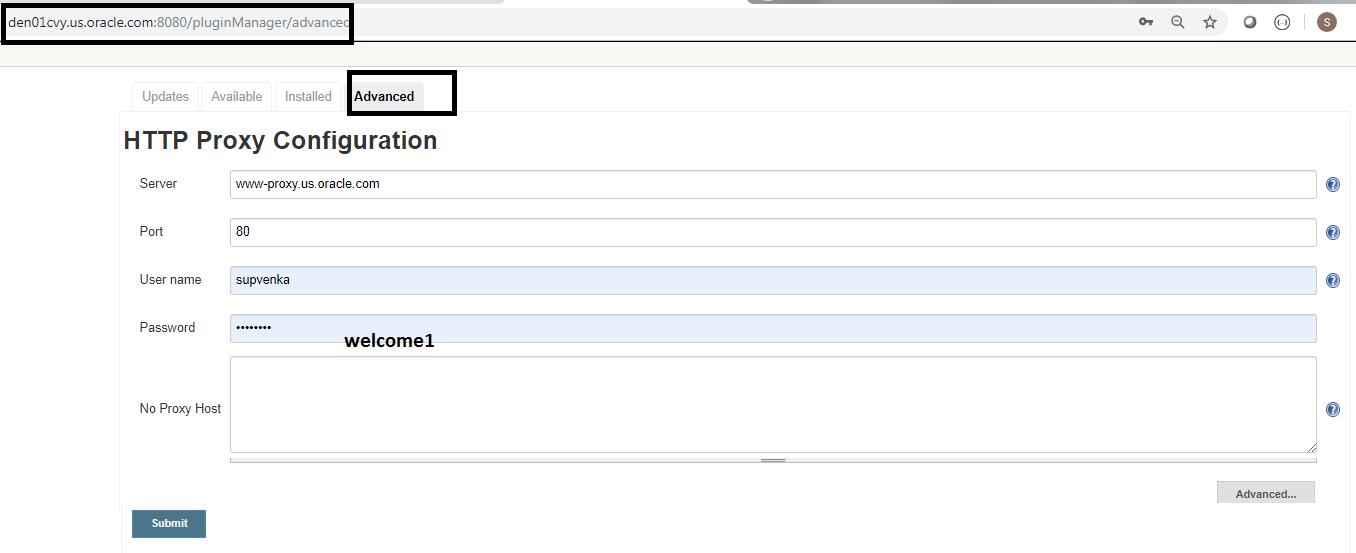


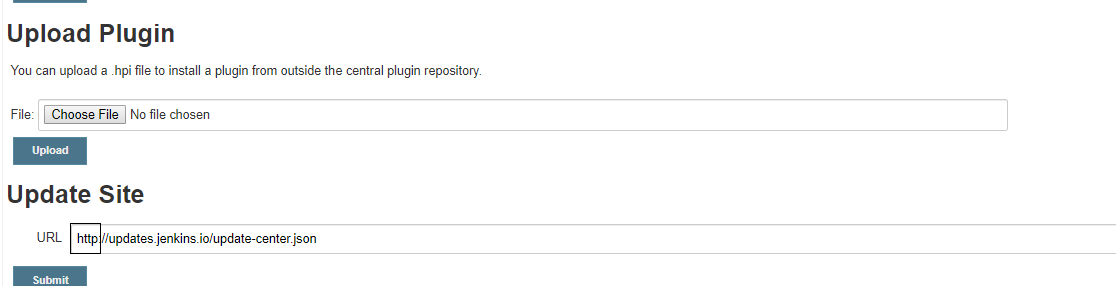
Once Jenkins is up and running

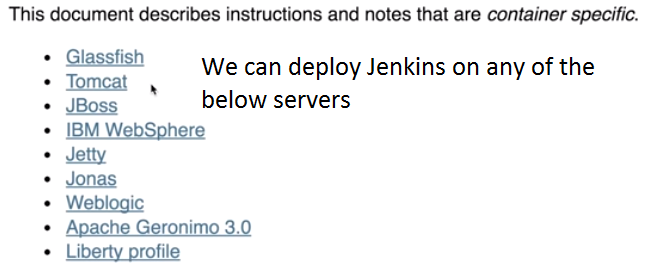
Go to <http://localshost:8080> (default port = 8080)

First time login give the password provided .

Then we can set up the user name and pw later (supvenka/welcome1)





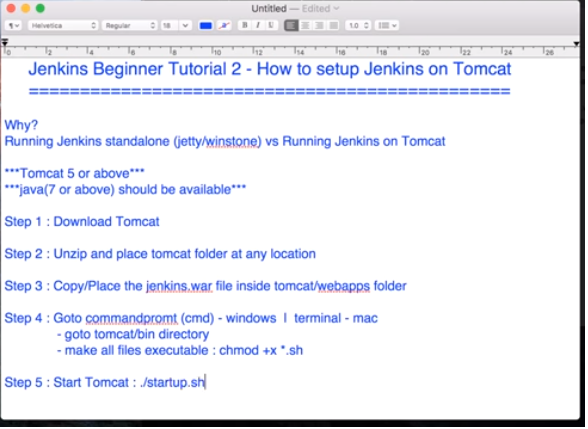


**How to setup Jenkins on Tomcat**

**Why should we deploy Jenkins on tomcat , because Jenkins has it own standalone serverlet container Jety and windstone What is difference in using its own servlet container v/s using tomcat**

If we have tomcat we can start all applications on a single server. Most web applications are installed on a common server like tomcat.

* We need Tomcat 5 or above for Jenins
* We need Java ( 7 or above )

****

**Some issues I encountered**

Since Jenkins using Jetty and Tomcat on which we deployed Jenkins both used the **same** port we need to shut down one

I shut down Jenkins (place where I executed java -jar jenkins.war that terminal I did a ctrl C)

Then also I got an error when I did ./startup.sh in apacaheTomcat/logs/catalina.out folder when I viewed the catalina.out saying hudson did not load some class.

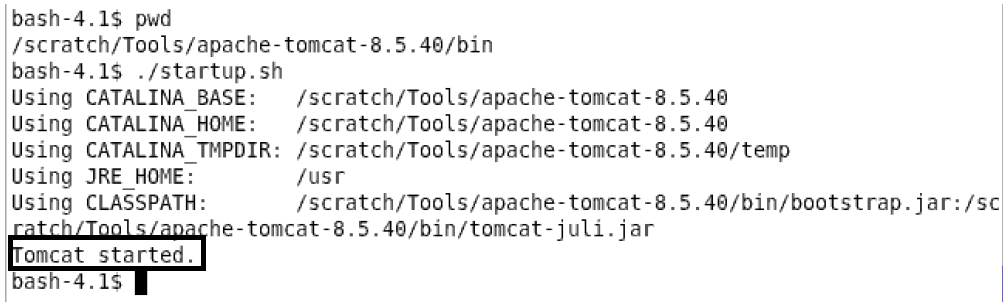
I then had to set

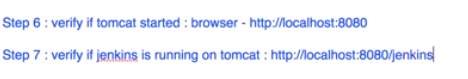
export JENKINS\_HOME=/scratch/supvenka/Jenkins/jenkinshome

./shutdown.sh

./startup.sh

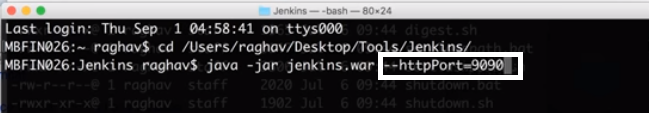
localhost:8080/jenkins/ came up





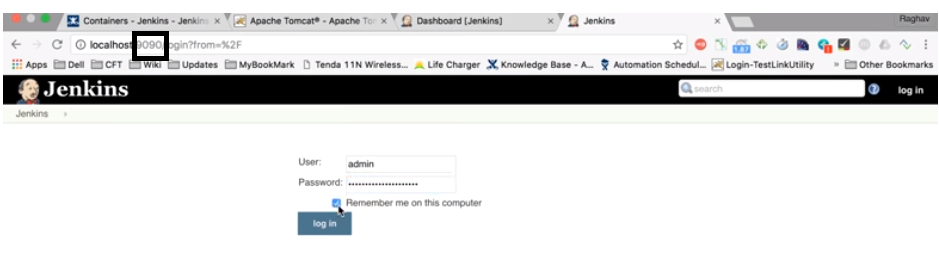
**Want to run Jenkins standalone also then specify a different port number since even Jenkins server uses port 8080 by default this will clash with the Jenkins running on port 8080**

**Hence we start Jenkins using a different port like below:**

****

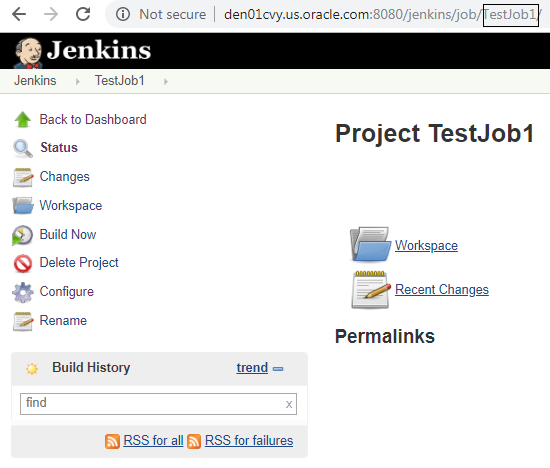
**Jenkins on port 9090 launched as shown below**

**localhost:9090/Jenkins/**

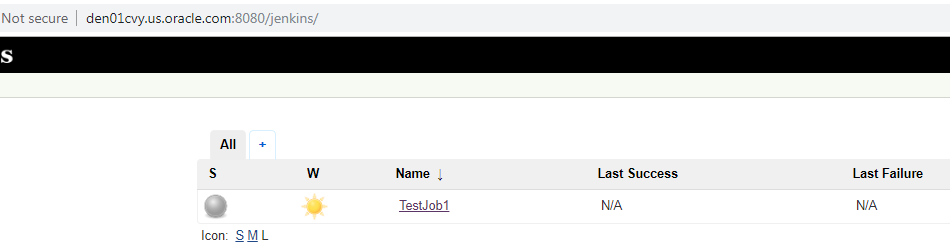
****

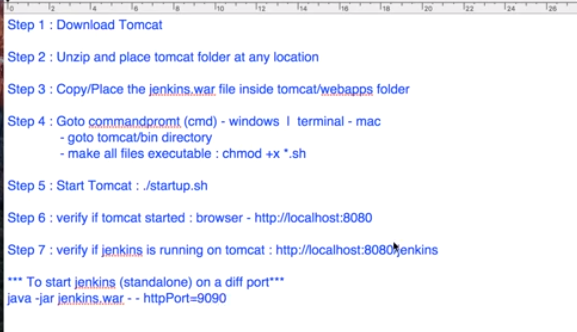
**Create job**

**TestJob1--🡪FreeStyleProject -🡪Save**

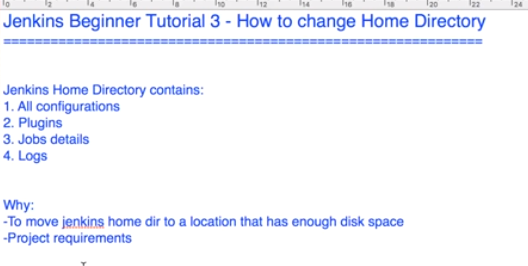
****

We can also see the job created here

****

****

**How to change your Jenkins Home directory:**

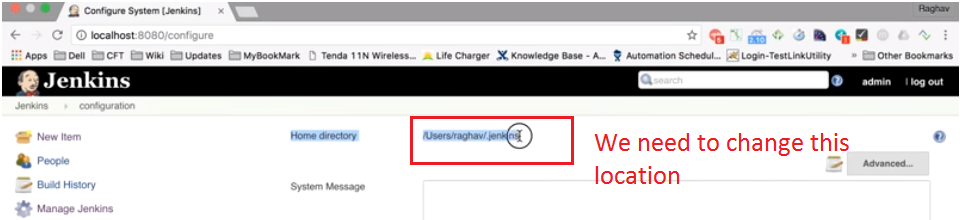
****

By default it will be where your ./Jenkins folder is there where we extracted ran

Java –jar Jenkins.war

This will create a ./Jenkins folder this is usually under /user/supvenka/Jenkins.

* I need to change this as this directory will need to be kept in a place where there is enough diskspace and for project requirements



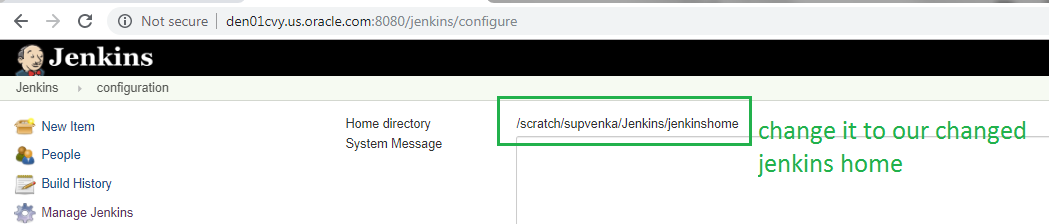
For that execute java –jar jenkins.war in a path

cd /scratch/supvenka/Jenkins/jenkinshome > java –jar jenkins.war

The copy the ./jenkins folder that Is created to a place where there is **sufficient space**

Then now in our jenkins we need to go to

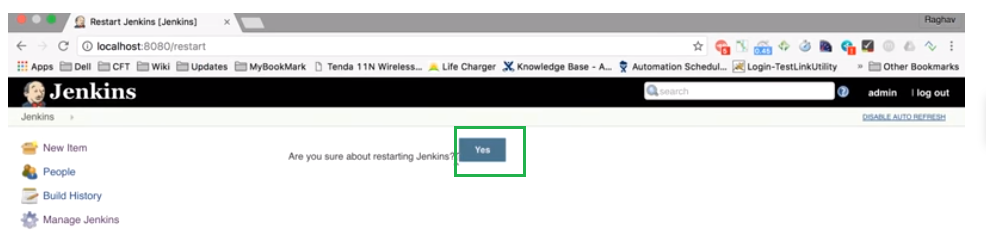




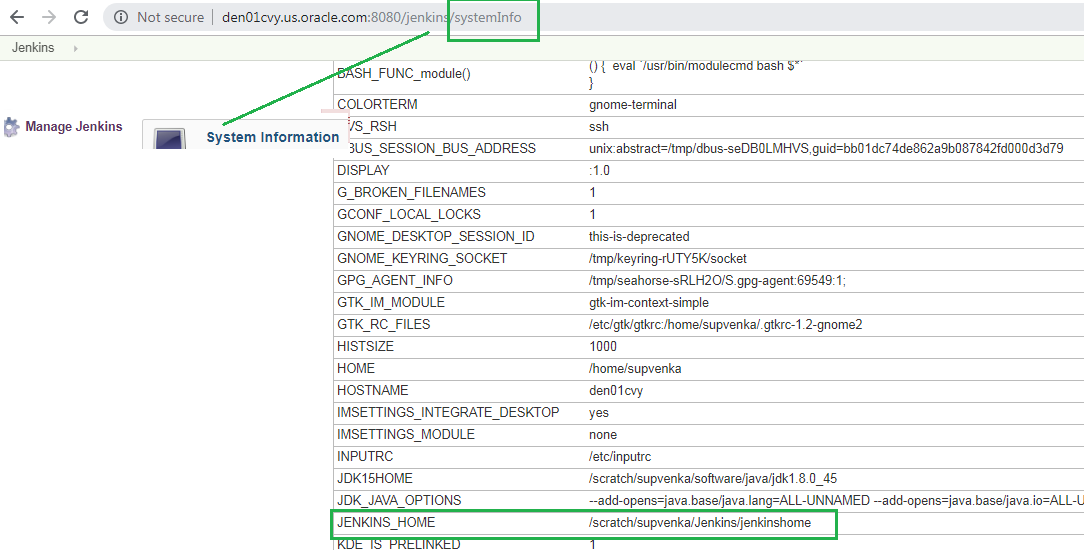
We need to set JENKINS\_HOME use the export command

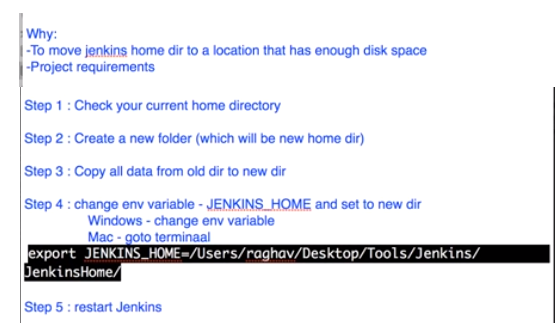
Then restart jenkins we can do it either by ctrl +C on the terminal we executed java –jar jenkins.war.

OR



Check if the JENKINS\_HOME has been set in the systemInfo



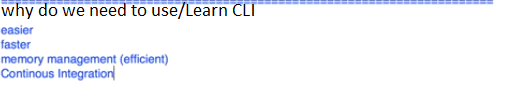


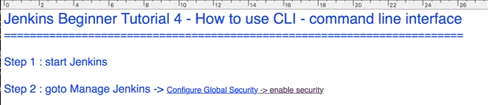
**HOW TO USE CLI (Command Line Interface)**

**Jenkins CLI documentation :** [**https://jenkins.io/doc/book/managing/cli/**](https://jenkins.io/doc/book/managing/cli/)

**CLI is useful because it is:**

* easier
* faster
* memory management is more efficient than UI : It consumes lesser memory
* Continuous Integration: When implementing CI command line is used





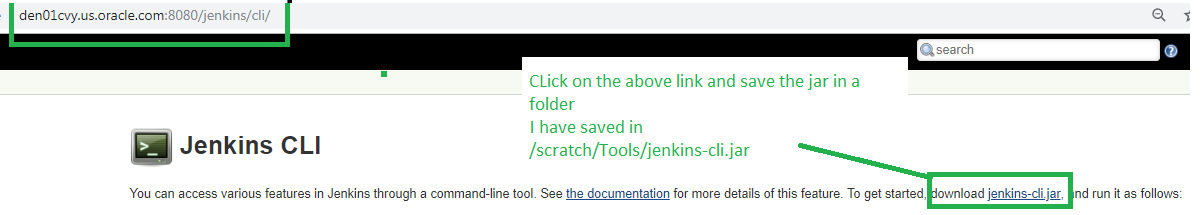


****

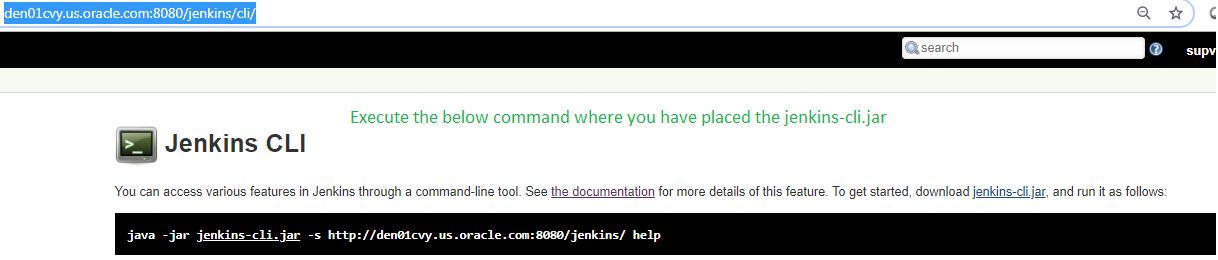
The above step 3 is using Jenkins standalone server hence the above url is fine.

If we have deployed it on tomcat we need to use url something like this

<http://den01cvy.us.oracle.com:8080/jenkins/cli/>

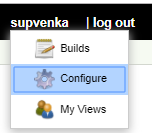
****

****

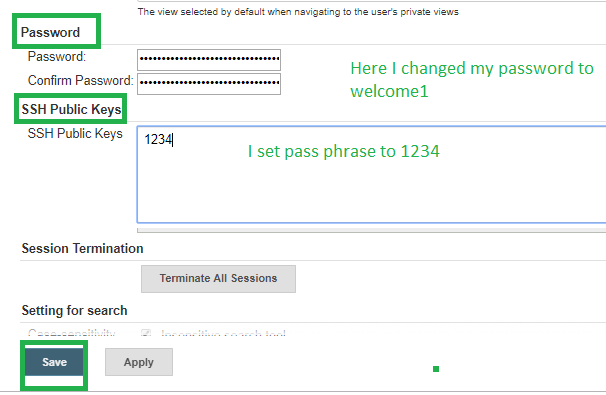
****

**Issues faced.**

* I did not have a pass phrase set. The above command when executed asks for a passphrase.
* I also did not remember my default auto assigned pwd and the initial admin pw file was not present
* So I logged in as supvenka in jenkins: <http://den01cvy.us.oracle.com:8080/jenkins/user/supvenka/>
* Click on configure as shown below



Then go to



Shut down apache ./shutdown then execute ./start-up.sh

Then go to <http://den01cvy.us.oracle.com:8080/jenkins/cli/>

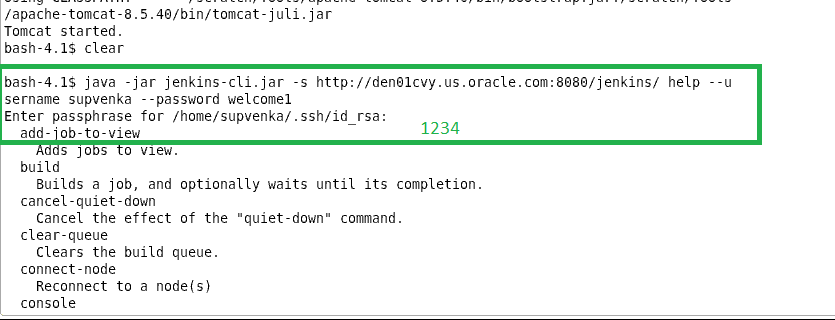
Copy the complete command including the help to clipboard



And execute it in the path where **your jenkins-cli.jar is located**

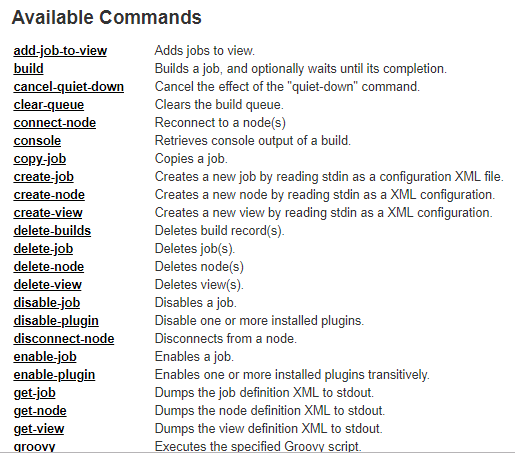
**For me I needed to provide my user name and password so I executed the following command**

**java -jar jenkins-cli.jar -s http://den01cvy.us.oracle.com:8080/jenkins/ help --username supvenka --password welcome1**

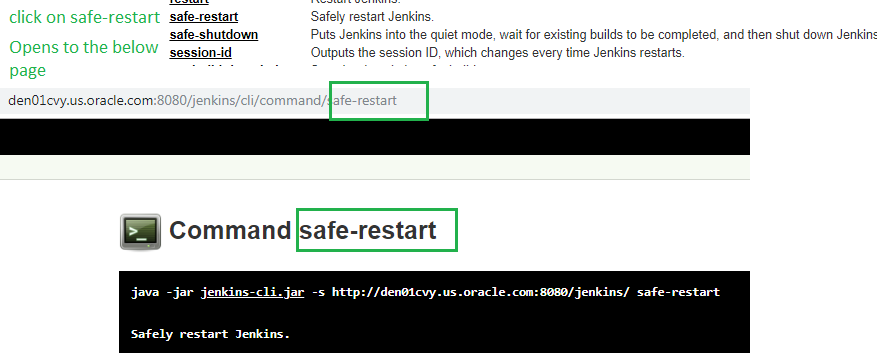


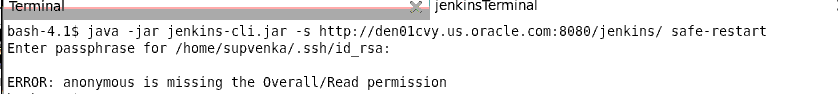
After we run the command we get all the possible options like above we see connect-node build etc.

We will be able to execute the commands shown below



Clicking on any of the above command opens another tab with the exact command to be executed

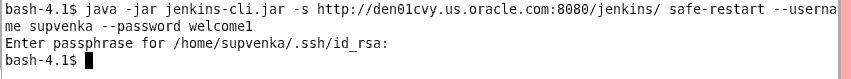




Got the error so I needed to provide username and pw

I can now copy the above command go to my terminal where and run the above command

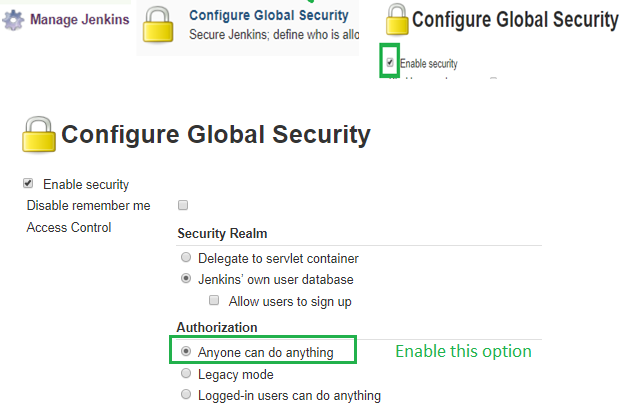
With user name and password and passphrase



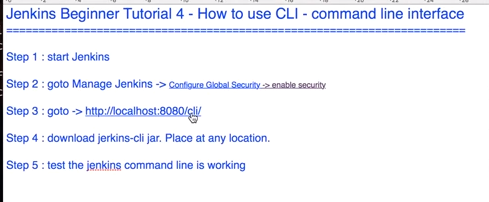
To avoid the Overall/Read permissions we can do the below

We can also set that anyone can do anything (This is however not really advised)

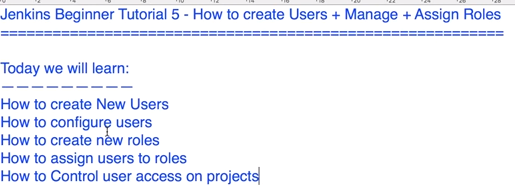
Manage Jenkins-🡪Configure Global Security-🡪Access Control –Authorization---Anyone can do anything (enable this)



**Summary** : In the *above CLI we have executed the below*



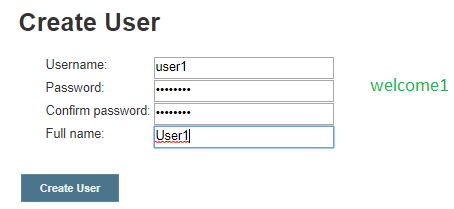
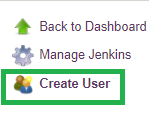
**How to create USERS, MANAGE and ASSIGN ROLES**

****

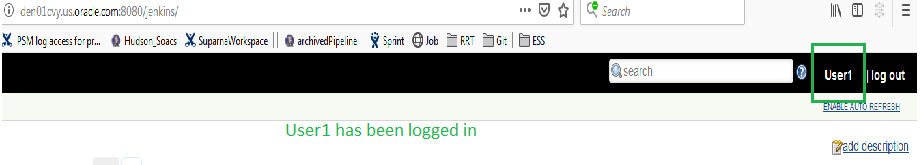
**Create Users:**

Go to Manage Jenkins-🡪Manage Users-🡪Create User





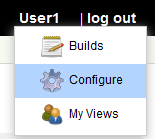
Login as the above user created to verify it works fine.

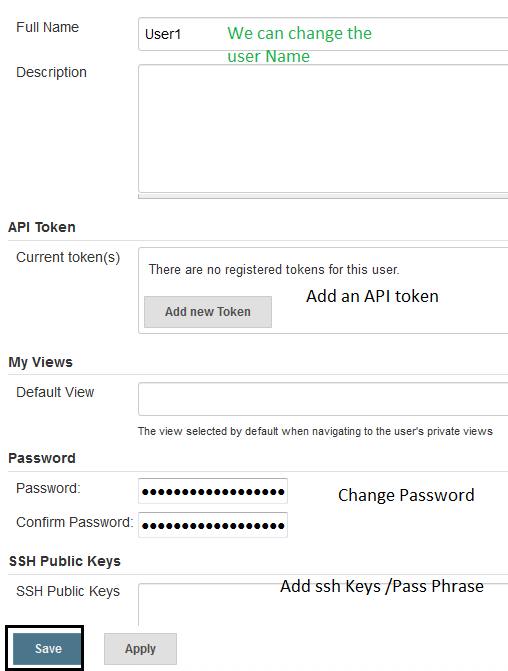


**Configure Users:**

**User configuration**

Login as the user.

Click on Username🡪configure🡪



**Manage User Roles**

For Managing the Roles we use a plugin called **Roles Strategy Plugin**

**Need to be logged in as Admin (supvenka)**

****

* Go to url : <https://wiki.jenkins.io/display/JENKINS/Role+Strategy+Plugin>
* Download the Latest: it downloads a role-strategy.hpi file gets downloaded.
* We then do to $JENKINS\_HOME/plugins/and add the role-strategy.hpi
* Restart your Jenkins

**Second Way is**

The below should be executed as admin User(supvenka)

* Go to Jenkins-🡪Manage Jenkins🡪Manage Plugins
* Go to Available Tab and search for role.
* Then choose Role-basedAuthorizationStrategy plugin



* Download now and Install after restart.

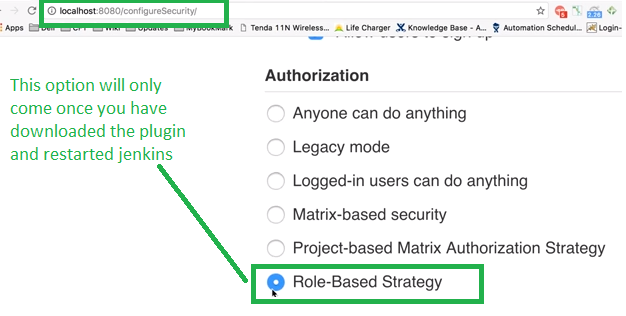


* ManageJenkins🡪Configure Global Settings-ConfigureSecurity

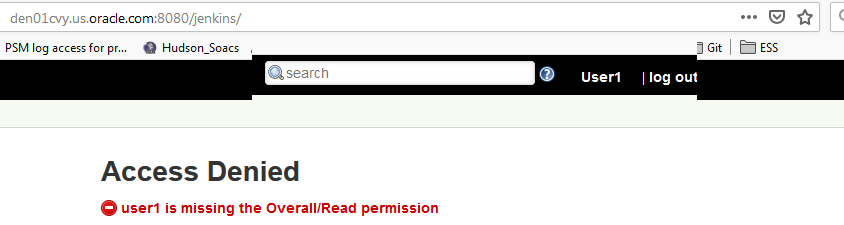
Ensure Enable Security is checked



* Once the plugin is downloaded and **Jenkins restarted** then we see the below option



Now log off and log in as user1/welcome: Notice the following User1 does not have permissions for anything.

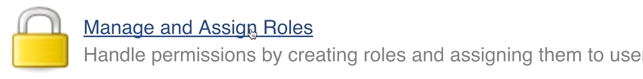


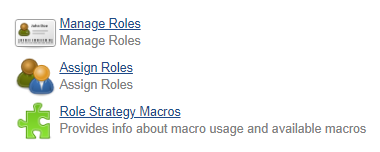
To give permissions to the user

* Login as admin
* Go to Manage Jenkins.

**Managing Roles:**

* Notice the Manage and Assign Roles option. This will **come only when** the role-strategy plugin has been installed





* Click **on Manage Roles**

We have Different Roles:

**Global Role**: Global roles provides authorization and access on global level

**Project Role:** Gives the user project level access based on the **pattern of project name** and access provided

**Slave Role:**

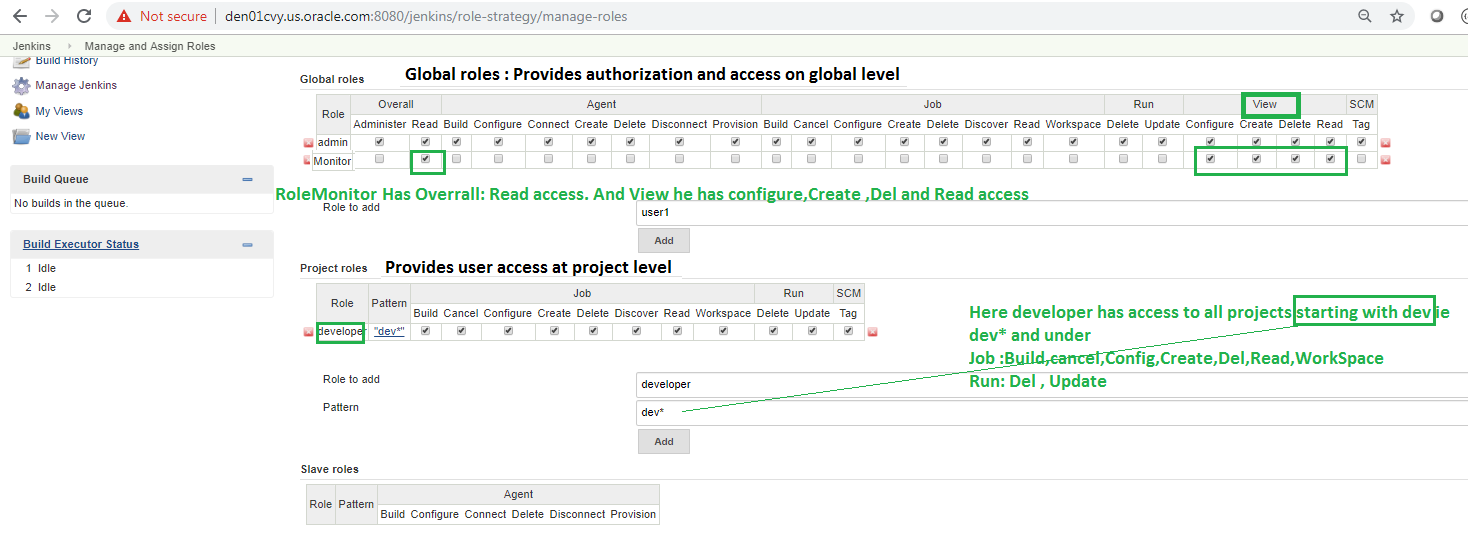
* Under Global Role: Provide role Name ex Monitor
* click add
* We see that under Global Roles: check/Tick roles that this global role has
* Under Project Role:

Provide role Name ex developer and Pattern = dev\* 🡪implies the developer role can access any project starting with dev as pattern is dev\* and build,configure,del..etc which ahs been checked

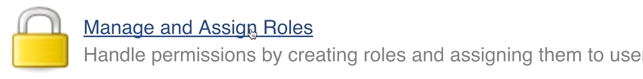
Provide role Name ex developer and Pattern = test\*

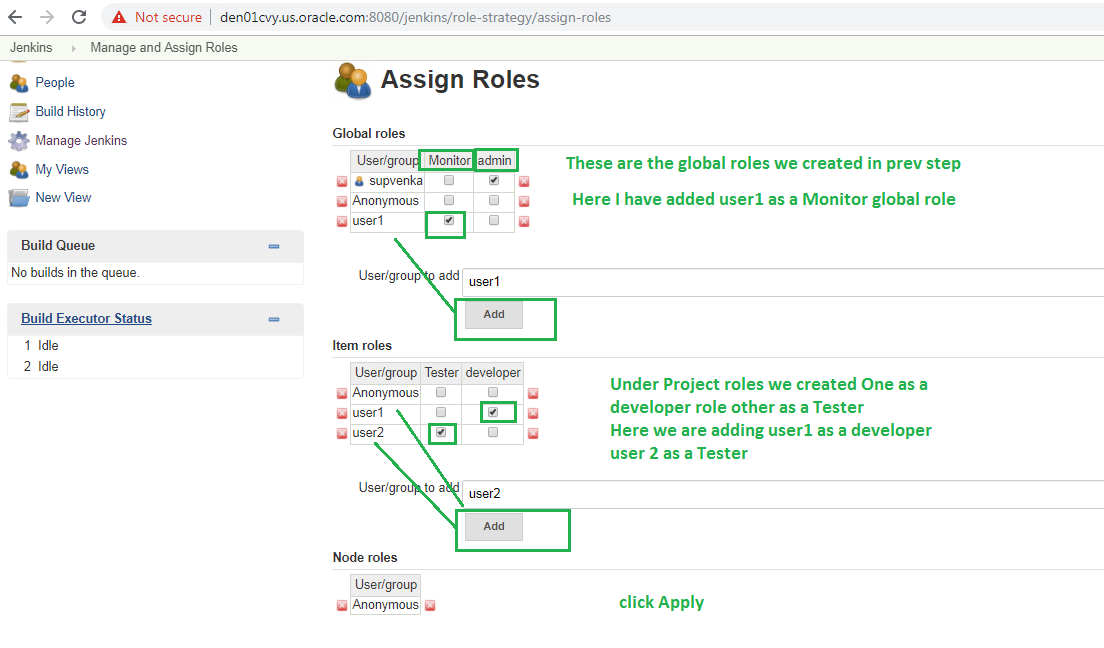
implies the Tester role can access any project starting with test as pattern is test\* and build,configure,del..etc which has been checked

* We see that under Project Roles: check/Tick roles that this applicable / you want



**Assigning Roles to the users:**

* Login as admin
* Go to Manage Jenkins.
* 
* Click **on Assign Roles**
* We see under Global Roles the roles we created like the **Monitor**
* We give a user name : say user1
* Click Add
* Then check Monitor box under the user’s column to give him Monitor privilege
* We see under Item Roles the roles we created like the **Tester and Developer**
* We give a user name : say user1, user2
* Click Add
* Then check **Tester and Developer** box under the user’s column to give him a Developer or Tester’s role/ privilege

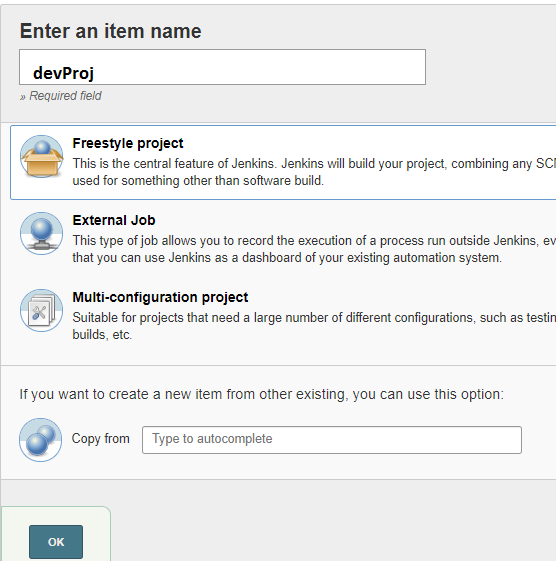
****

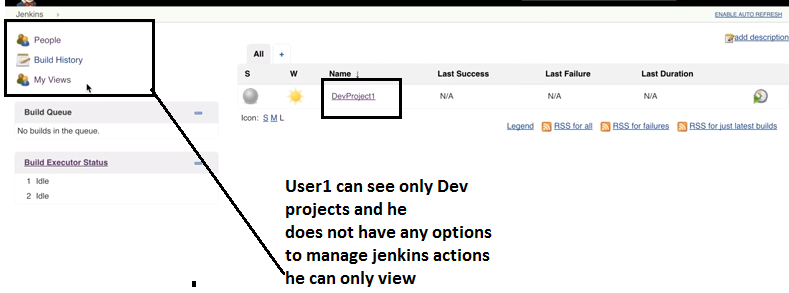
**Now to Test the above:**

* We will create two jobs with name DeveloperProj and TesterProj
* When user1 logins in From above he has monitor global role + developer role
* So he can Just see DeveloperProj job (and not TesterProj job) and he can just view he should not have any configurations to change anything.

Click New Item

Create devProj





Issue I faced My Dev could not see dev projects

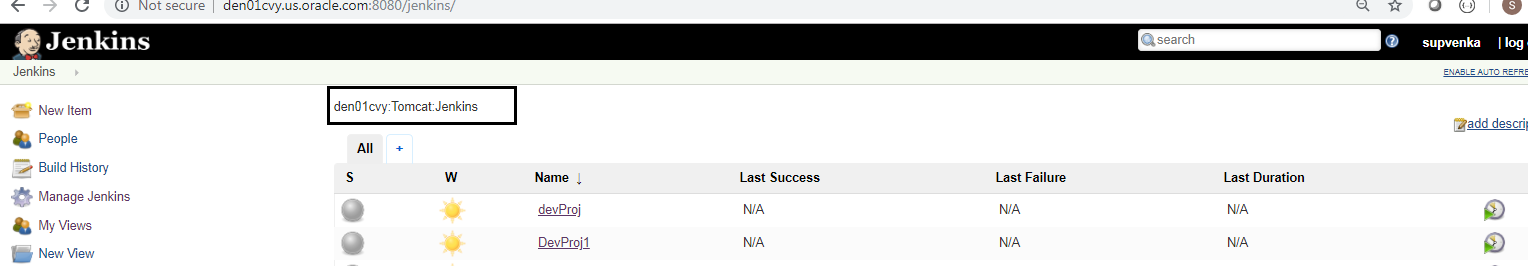
I could not find role-strategy.hpi plugin o the site to download

**BASIC CONFIGURATIONS**

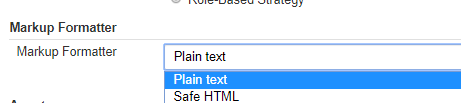
* **Home directory** :/scratch/supvenka/Jenkins/jenkinshome

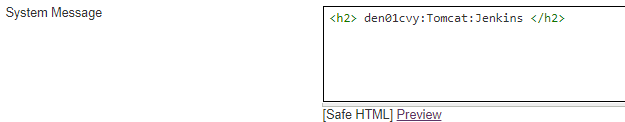
Jenkins stores all of **its data** in this directory on the file system.

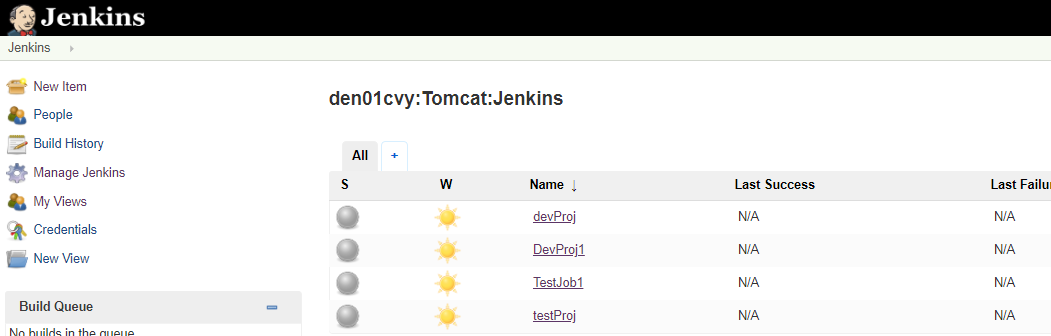
* **Build Record Root Directory:** Stores all the build records,logs etc
* **System Message:** This message will be displayed at the top of the Jenkins main page.
  + **Plain Text:** Write any Text : DO a preview to see how it would look



* + **Safe Html Text** : We can use html Tags for display to check if html is enabled
    - Manage-jenkins-Configure Security
    - Markup Formatter -->
      * choose from drop down :**Safe HTML**







* **Number of executors:** Maximum Number of jobs Jenkins can run at a time
* **Labels:** (We first need to understand master and slave concept)

Jenkins has system for a distributed build: Master and slave

Make Jenkins Host as a Master machine and add different other machines as nodes.

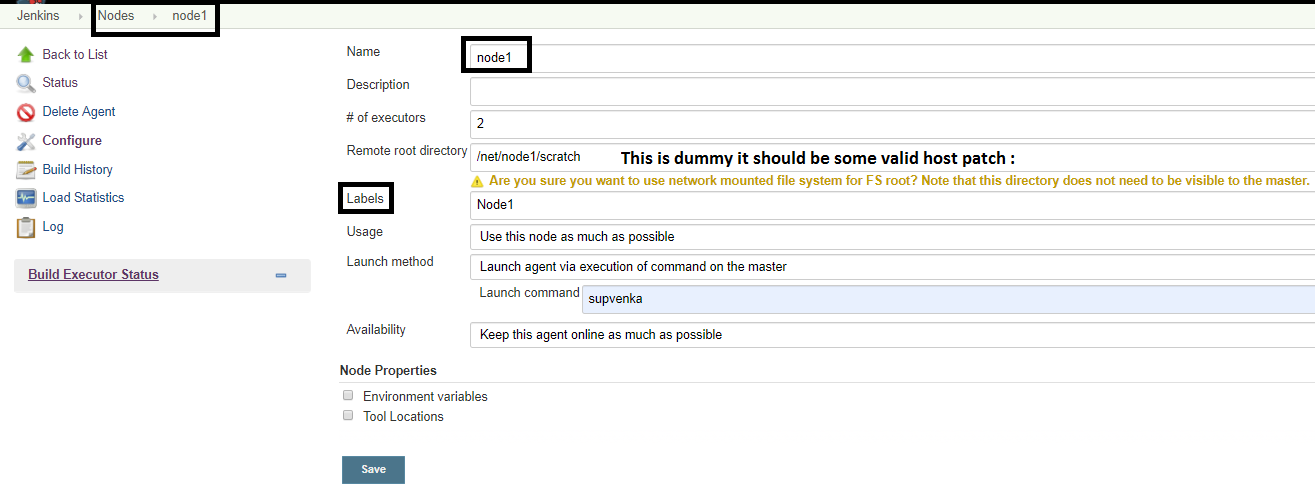
SO we can execute jobs on other nodes and NOT the master.

* + Manage Node:
  + Add New Nodes.

number of executors = Maximum number jobs that we can execute on this node at a time

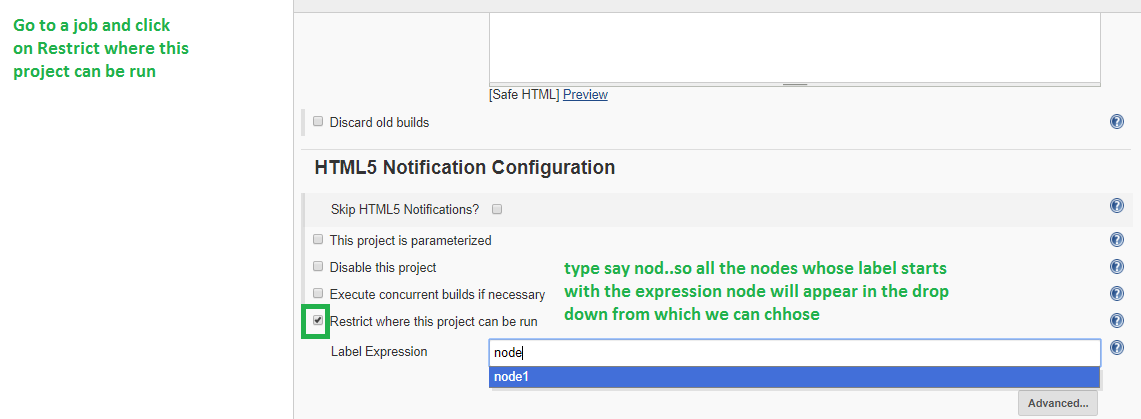
Label: Give label/name to the node

Root dir: The root dir where Jenkins can store all its records like build records etc.



We have added few node machine and given some labels to these nodes as shown in the above image

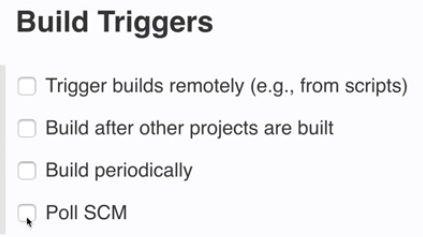
Go to job and click restrict where this proj can be run: Here it gives a Label expression. All the Labels we have given for the nodes will appear in this drop down. When can select It will mention something like "THis label is serviced by 1(n number of) nodes. This implies that this job will be run on only that node whose label matches.



* **Quiet Period:** Is the number of seconds the Jenkins instance will wait before it triggers the job. If there is a delay in some commit of a file to happen. It is good to have this quiet period else this job will get trigger and may and fail while commit in progress, or commit failed
* **SCM checkout Tries**

In Build Trigggers: We have

Pools SCM: It will try to connect to a source code repo and gets the latest build information. In case it fails what is the maximum retry count our Jenkins should do.



SCM checkout Tries--- This number is specified here as to how many times it can try

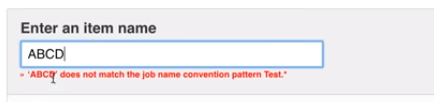
**Restrict Project Naming:**

By default it is default.

We can specify a pattern as well: When we give a pattern. Then we can create Jobs/projects only matching that pattern else it will give an error like below

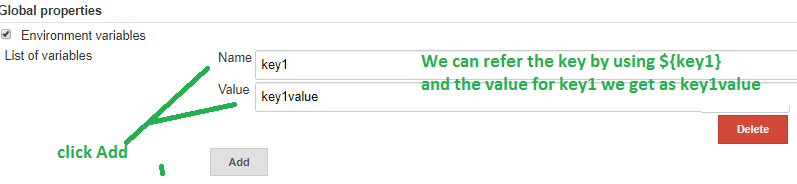


I gave a name ABC which does not match the pattern Test so it gives an error as shown below:



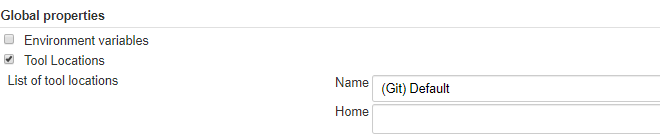
**Global Properties**:

Environment variables properties at global which can be used by all the jobs at Jenkins. We can refer the key by using ${key1}



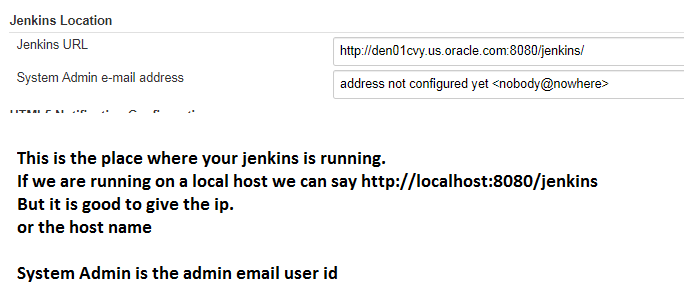
Environment variables: Values that can be accessed at a global level which can be referred in any job.

eg Git value etc



Jenkins url: The url where Jenkins is installed and running

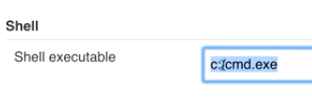
System Admin e-mail address :is same as the email address of the admin user.



**SSh Server:** In case you want Jenkins to run Job remotely or connect to a remote server.

**Shell:** By default has **bash shell** on which it executes its commands.

If you on run on any other shell.or run on cygwin we need to give the path of the cygwin exe ex on windows we run on cmd.exe so we gives its **path**

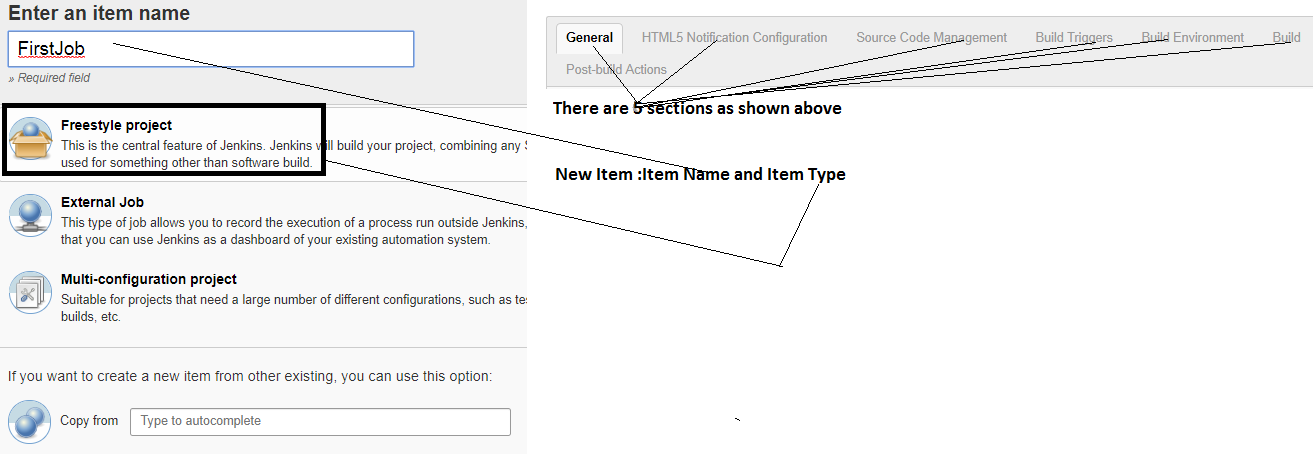


**JOBS**

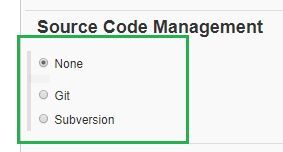
****

**Create a New job:**

****

****

**Source Code Management:** Say we want to take the source code **system like git** or a sub version. We need to add plugins to get the source.



**Build Triggers:**

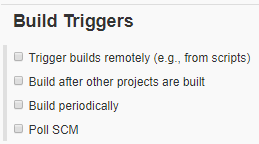
Trigger build remotely (using scripts)

Build after other projects are built like dependency.

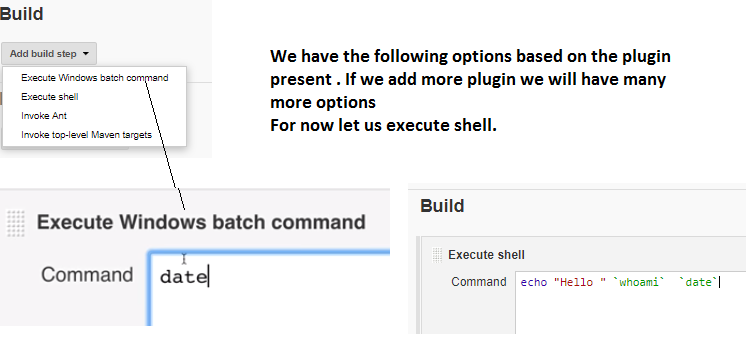
Build periodically

Poll SCM when we have source code Management set to any SCM like git.

When this option is enabled our jobs gets triggered automatically when there is any build check-in available in our SCM like git.



Build: We can any build steps Like execute commands on Windows batch command Or shell etc we use this.



**Build Periodically:**

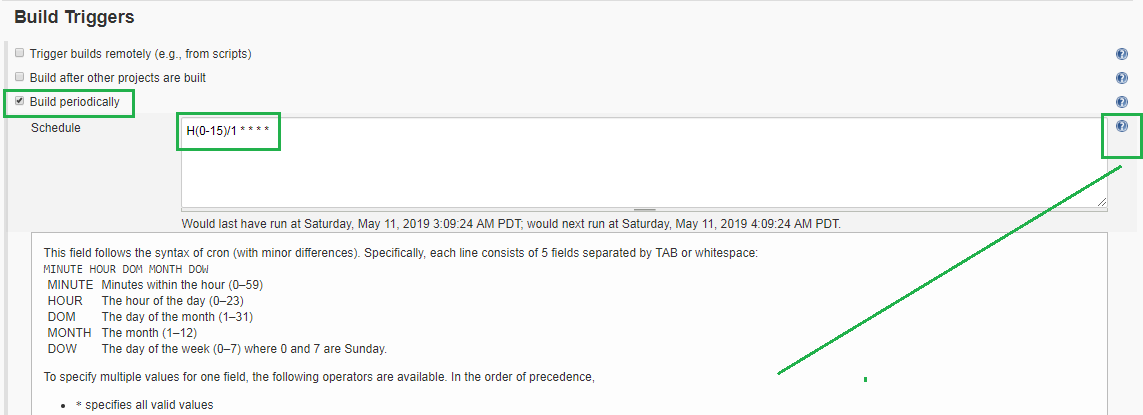
**Schedule:** Here we provide some expression based on our expression our job will be executed at some particular interval

Clicking on the Help it will tell us what format we need to follow

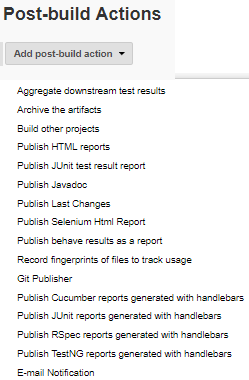
Ex: every minute H/1\*\*\*\*

It will also say when we schedule it the last run was when and when the next run will be.

ex \*\*\*\* : Means every minute



**Post build Actions:** After we build our project. Say we want to build another project or Report notification Or execute another Test like acceptance etc



**How to trigger Jobs remotely**

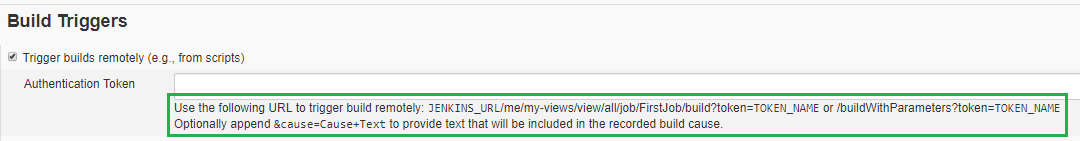
****

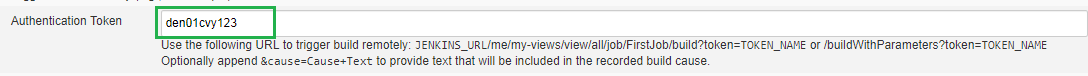
Go to your Job

Click on configure

Click on Build Triggers 🡪 Trigger Builds remotely

In Authentication Token : It gives us the url to trigger remotely like below:





Jenkins gives the url in the Authorization Token section.

TOKENNAME: What we provided in AuthenticationToken as the value

JENKINS\_URL/me/my-views/view/all/job/FirstJob/build?token=TOKEN\_NAME or /buildWithParameters?token=TOKEN\_NAME

Copy that url in the browser. Since we are running on den01cvy we specify the host. In the Authentication Token we can specify some token such that we use that token in the parameterized key TOKEN\_NAME

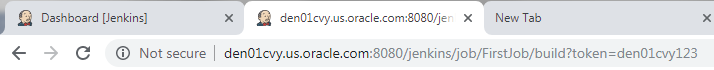
http://den01cvy.us.oracle.com:8080/jenkins/job/FirstJob/build?token=den01cvy123

Replacing the values to use the token I have provided

http://den01cvy.us.oracle.com:8080/jenkins/me/my-views/view/all/job/FirstJob/build?token=den01cvy123

OR

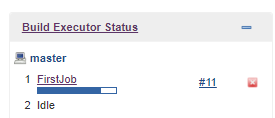
<http://den01cvy.us.oracle.com:8080/jenkins/job/FirstJob/build?token=den01cvy123>



We see the JOB triggered and present in the **BUILD QUEUE** Section



It runs as there are no jobs in the queue



**How to chain Job executions: One job triggers after another Job completed**

****

Click on a new item (New Item)

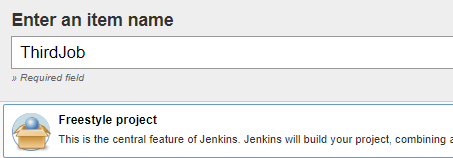
Give the item name and choose FreeStyle Project

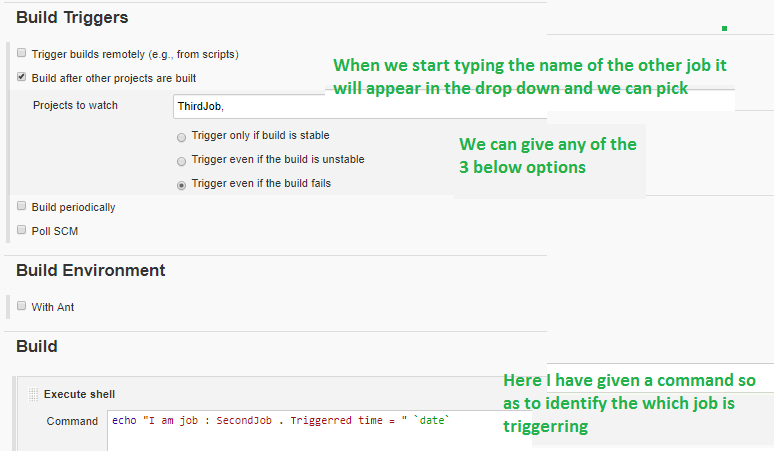
Go to the Job created.

Go to Build Trigger: Build after other projects are built

Here specify the Job name after which this job will trigger

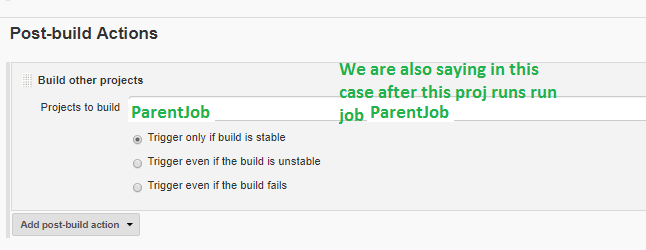
Here we have different options as shown below

****

****

Here SecondJob will run only after ThirdJob completes ie Trigger for this project will come from another project (in this case ThirdJob) and only when the other project completes this will run

**We also add Post Buld Actions**

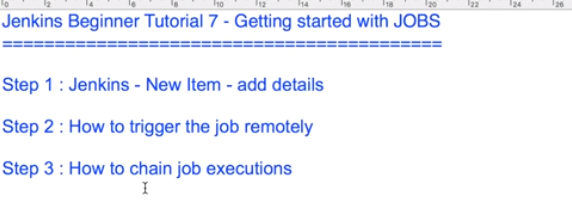
****

So I will Trigger ThirdJob

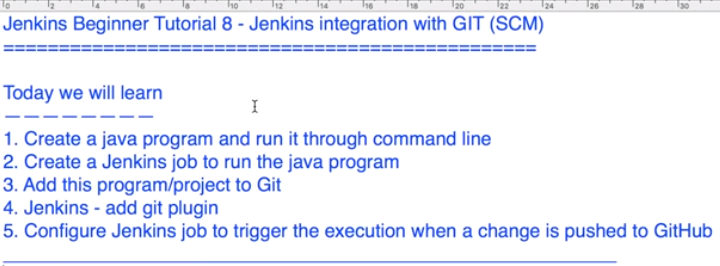
After which SecondJob gets Triggered

After which ParentJob Gets triggered.

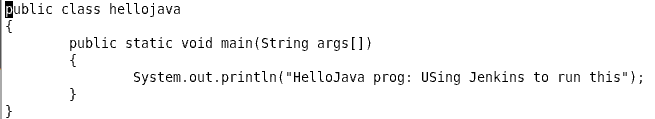
For us it worked fine ☺



**Integration with GIT**

****

I created a java class under /scratch/supvenka/Jenkins/javaprogs/hello.java



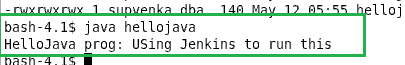
Compiled it using java hellojava.java

Executed using java hellojava







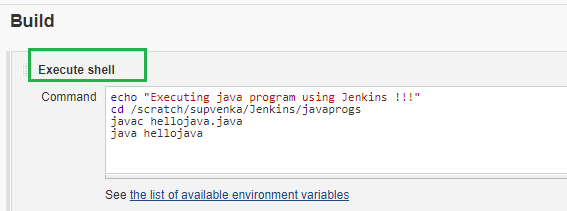




To test the java program works

* Create New Item (new Job) : FreeStyle Proj : ex HellojavaProg
* In the above Job go to Build
* Build : choose option Execute Shell
* Go to the location where the java file is present
* Compile using javac hellojava.java
* Execute the java program using

java hellojava

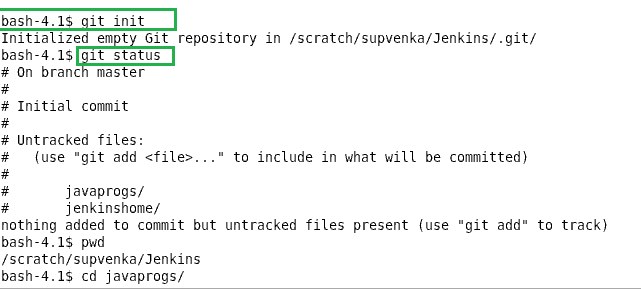


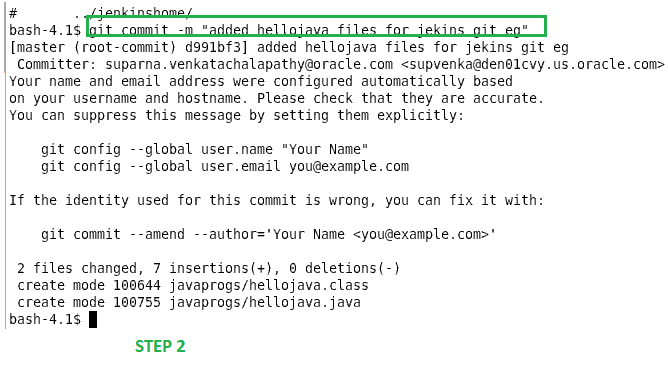




To initialize the git repository we do

* git init
* git status





I created a repository called Jenkins here

ssh = git@github.com:supvenka/Jenkins.git

http = <https://github.com/supvenka/Jenkins.git>



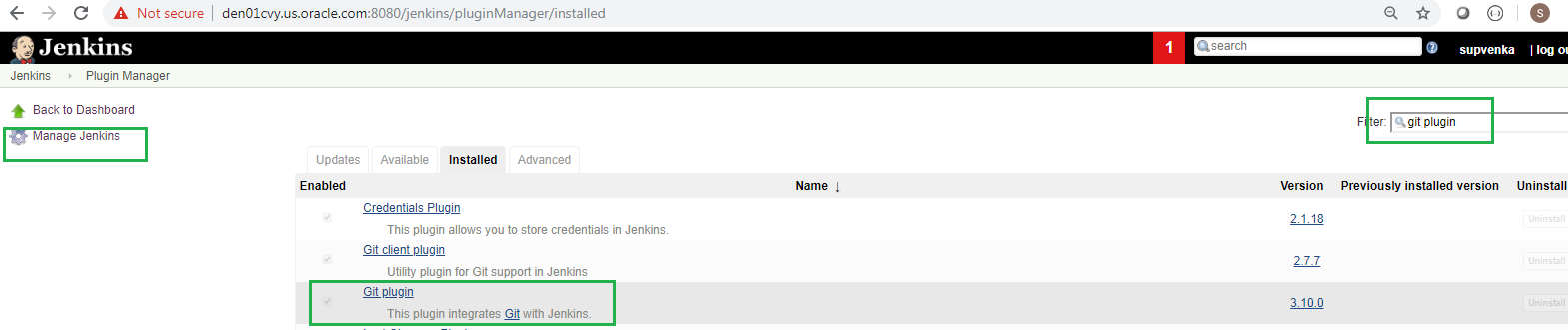
Todo : mine didn’t work need to check.

Once that is done the committed files get reflected in the repository.



* Check if the Jenkins has added the github plugin

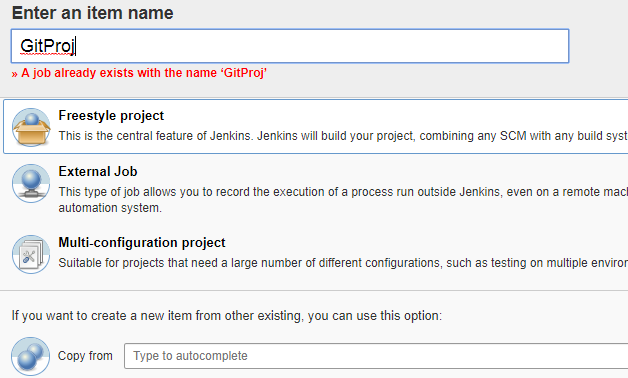
Manage Jenkins🡪Manage Plugins🡪Installed : search for git plugin: My case it is present else go to Available search for git plugin and then do Download and restart after download



**Configure a Job to trigger the when there is a change in the SCM / git repo**



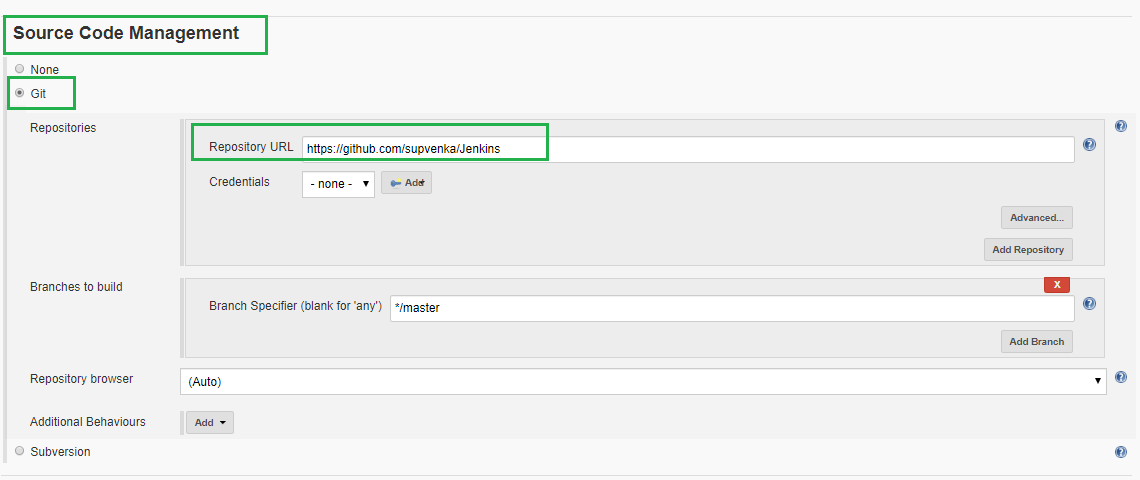
Go to Job /Project that has been created



* Configure
* Source Code Management

Git

Repository Name :

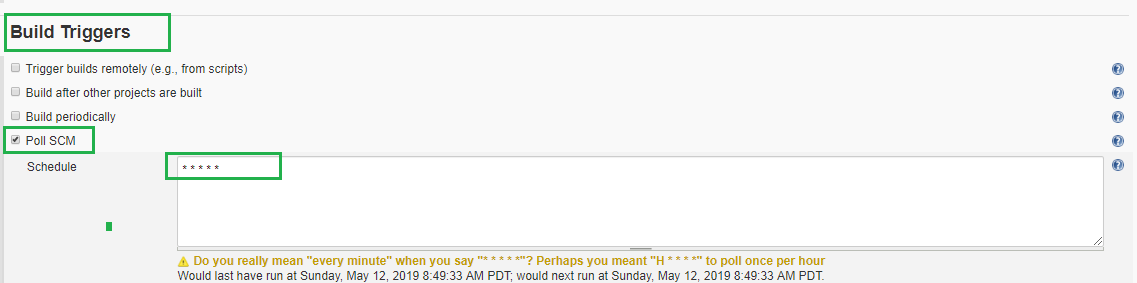


* Build Triggers:

Poll SCM

Schedule: Here we give the cron expressions

Ex: \*\*\*\*\* : Searches for changes every minute if there is a change in the SCM then it will trigger the job



* Apply
* Save.

Now we can create a new file in the repository

git add.

git status

Now let us make a change in our source code by adding say a readme file

We need to be in the same dir /folder structure where are git is

* git commit add -m "added readme"
* git remote add origin https://github.com/supvenka/Jenkins : **Don’t think this is need**
* git push origin master

Now notice that the Job that you created with SCM (Git) and Build trigger Poll Scm has triggered automatically.When we go to the console of the job we see in the console the **first** statement

**“Triggered by a SCM change”**

With the git repo information from where it is downloading and running the job

