Agile and DevOps: are they same?

No, Agile and DevOps are not same but both should work together in an organization where DevOps is implemented.

Agile :

In Agile Methodology Software development will be divided into small parts ( Sprints /Iterations).

Each Sprint is delivered as a final product with core basic features and missed features will be added in the next sprint.

In Agile we cannot make changes continuously in the final product and release it often.

As Agile works with cross functional teams like Design, Coding, and Testing.

Without including Operations Team Continuos change and Continuos Delivery will be not possible.

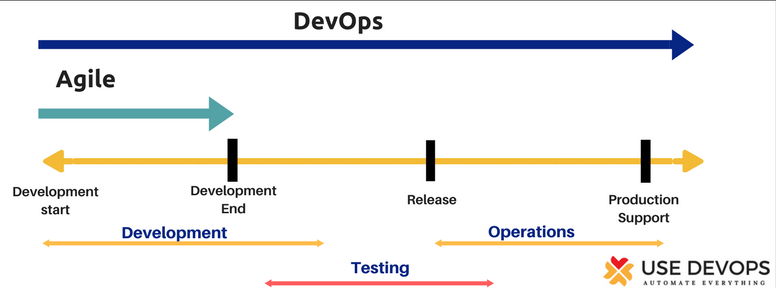
DevOps :

DevOps is a Practice of coordinating Development and Operations teams. It gives more flexibility on top of the Agile model.

With Continuous Integration (CI) and Continuous Delivery (CD) pipelines, we can make sure that we can release the product often.

DevOps practice not only focuses on the delivery of working software but also on overall service of the software to the business.

DevOps is not a separate concept but a mere extension of Agile. which includes operations as well in the definition of the cross-functional Agile team, collaborate together and work as ONE team with an objective to deliver software fully to the customer.



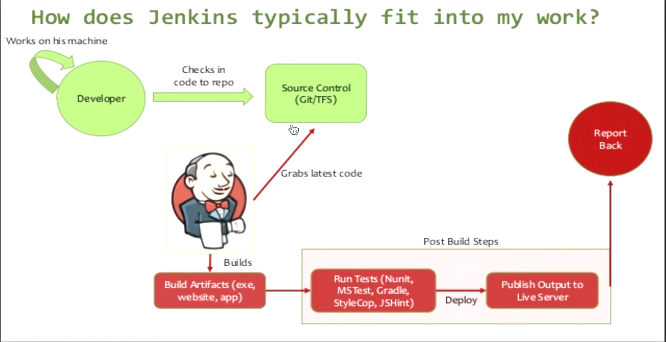
**Jenkins Beginner Tutorial 1 - Introduction**

1. Jenkins is a java application

Because it is java application it is platform independent and it does not required any specific platform. You can use Jenkins the same whether you are on windows or you are on mac or any other operating system

What we mean by this continuous integration and continuous delivery?

1. used for continuous integration and continuous delivery



Green-> when we working in teams we have different developers who developed a code and they keep on working on changing or enhancing the code and

As the developer works on his own machine and changes code he will check in the code into some shared repository which can be our gutbucket /TFS anything

Now what happens is:: let's suppose through the entire day there are many developers who are changing the Code and are checking in the Shared repository.

At the end of the day when you do a build …, let’s suppose there is some bug introduced in any of the code and now the build Failed.

Now: it would be very difficult for us to identify

* what exact code Build failed and
* at what point was this Bug introduced in the code

And then you have to go back and check in all the code that was created that day and there will be a lot of confusion

okay so here comes Jenkins

Here comes Jenkins, Continuous Integration

What Jenkins does is: As soon as any developer commits code in the shared repository. Jenkins will latest code and trigger build. And Build notification will be sent out.

We can check that:

* if there is any issue due to any commit ,we can reverse that
* We will not waste a lot of time in finding out what code cause the brea
* Let’s suppose Build successful. We can also integrate our unit tests or acceptance tests or performance tests along with the build as post-build actions in Jenkins and it will be automated.

(As soon as build gets deployed, Jenkins will trigger some test cases/some testing which will be automated and it will send out a report back to us.)

* Okay so let's suppose the build was successful there was no exception, but due to the changes in the code .There was something which broke in the application. So we will come to know instantly that there is some break in Code and we can troubleshoot it.

So you can see how are efficient and powerful, entire system can be and this is what is called as continuous integration.

Continuous Delivery

1. We are taking code and doing build, as soon as code gets commit and also performing tests and sending out reports.
2. When change in code, Entire cycle is triggered and we can know any issue with application

So this is what Continuous Delivery is all about…

|  |
| --- |
| Dir:  Plugins-> .jenkins folder ->Config, jobs, plugins, logs  Java –jar jenkins.war –httpPort=9090  Localhost:8080/restart  Localhost:8080/configure  Localhost:8080/systeminfo  Admin-> Configure->ssh public keys—passphrase    Jenkins-> Manage Jenkins->global security->authorization \_>anyone can do anything (for now) |

**Setup Jenkins standalone**

1. Download Jenkins… Jenkins.io (LTS)..stable ,plugins OR <http://mirrors.jenkins-ci.org/windows/>
2. Place war file into any location on your system.
3. Start Jenkins🡪 cmd/terminal

Java –jar Jenkins.war

Process: Extracts war file

Winstone: Server

1. **Gives initialAdminPassword**
2. We can access it On web browser on port 8080

http://localhost:8080

|  |
| --- |
| * C:\Users\vebalusu\Downloads>java -jar jenkins.war * webroot: $user.home/.jenkins * Jenkins home directory: C:\Users\vebalusu\.jenkins found at: $user.home/.jenkins * Jenkins initial setup is required. An admin user has been created and a password   C:\Users\vebalusu\.jenkins\secrets\initialAdminPassword |

1. **Install Required Jenkins (plugins)**: 1.) Select suggested plugins 2.) Install plugins
2. I can see what plugins available for building tool
3. if I want to see a notification and publishing click here
4. Create Jenkins admin user
5. Get started with Jenkins home window.

**Jenkins Beginner Tutorial 2 - How to setup Jenkins on Tomcat**

**Setup Jenkins on Tomcat or any other servlet container.**

**Why should deploy Jenkins on tomcat?**

Because we can start Jenkins on its own standalone server.

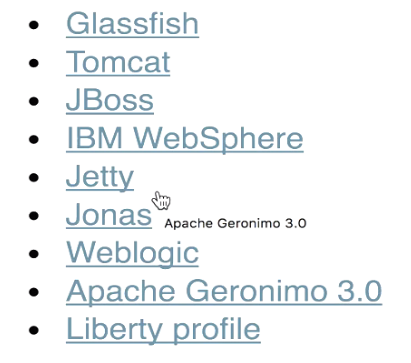
Jenkins comes by default with its own servlet container which is jetty and winstone We can also start and deployment case on other subject containers like tomcat

tomcat is one of the most common used to servlet containers

What is a difference between running Jenkins standalone (default) vs running Jenkins on Tomcat on any other Servlet container?

One of the benefits of running it on tomcat is:

* We can start all your applications on a single server.
* So mostly we have most of our Web applications and other applications installed on our Common server like Tomcat.



Java

Servlets

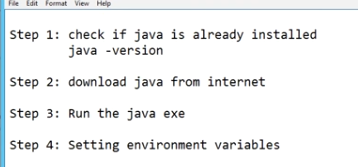
1. Java 7 or above---Java SE development kit 8---jdk

|  |
| --- |
| You are pointing to a JRE. You need to point to a JDK, which given the location of your JRE might be something like C:\Program Files (x86)\Java\jdk1.6.0\_26 or similar. Or, if you only have a JRE installed, you'll need to install a full JDK. Tomcat needs to be able to compile JSPs into .class files, which a JRE can't do.  JDK-🡪Java Development Kit |

JAVA\_HOME ->C:\Program Files (x86)\Java\jdk

PATH-> ; C:\Program Files (x86)\Java\jdk\bin or ;%JAVA\_HOME%\bin

If we change JAVA installation path, no need to change PATH Env variable



1. Tomcat 5 or above

|  |
| --- |
|  |

**In step-7:** Jenkins on tomcat and with Different port are same.

We can use Jenkins as standalone and in Tomcat

Note: Verify by creating job on each of these.

How to change Home Directory

Jenkins Beginner Tutorial 3 - How to change Home Directory

.jenkins---HOME dir

* webroot: $user.home/.jenkins
* Jenkins home directory: C:\Users\vebalusu\.jenkins found at: $user.home/.jenkins

What is a home directory of Jenkins ?

Whenever we are deploy Jenkins on any system, there is the folder which is created by default (. Jenkins), at the default location will be the user profile of that system and this directory contains all the information about the

* logs,
* configuration,
* job details,
* plugins , everything about Jenkin

Why do we want to change this home directory??

* DiskSpace: this is getting installed in the user profile and we want to move it to a place where we have enough disk space

Because they will be a lot of jobs being added to our Jenkins there will be a lot of plugins being added and other configurations and we want this home directly to be placed in such a location where we have sufficient space

* Project requirement

1. check your current home directory

You have to go to location where you have downloaded and you have put Jenkins folder. (Jenkins War file location)

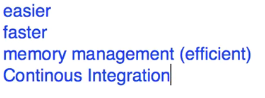
1. Start Jenkins

manage Jenkins -> configure system -> home directory

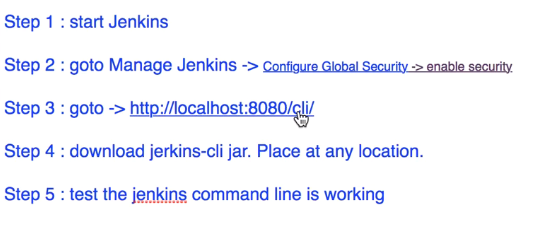
So you will see your current home directory here

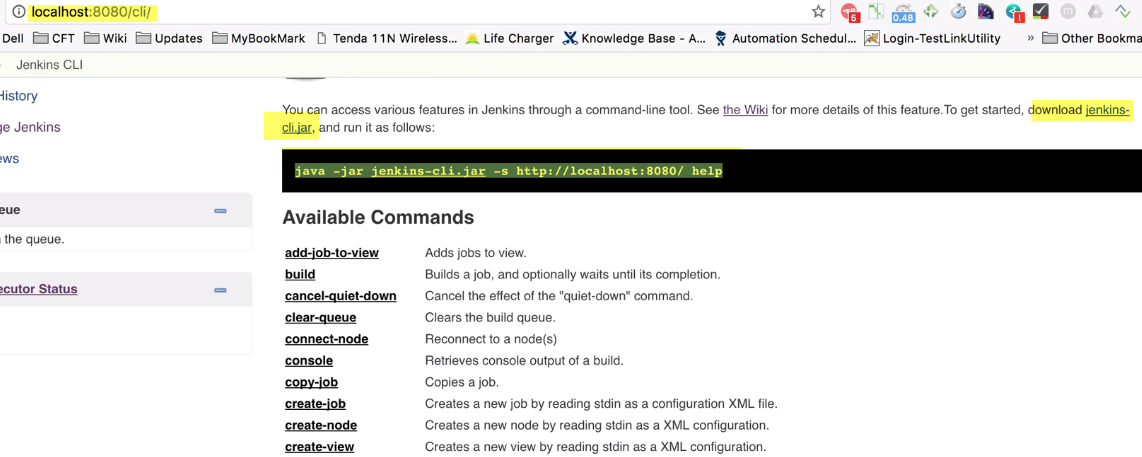


**Jenkins Beginner Tutorial 4 - How to use CLI (command line interface)**



* Consumes less memory than UI
* When we have to do CI, setup build -🡪 we use CLI





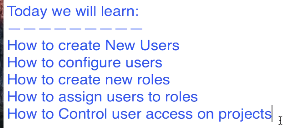
Go to: Admin(or user)-> Configure->ssh public keys(set anything here)—passphrase



Ex: Safe-restart

Jenkins-> Manage Jenkins->global security->authorization \_>anyone can do anything (for now)

Jenkins Beginner Tutorial 5 - How to create Users + Manage + Assign Roles



password

Login with the admin

Manage Jenkins ->configure system ->home directory -> secrets ->initial admin password.

Create new users:

Manage Jenkins->manage users-> create user

Configure

Whenever you log into your Jenkins the top right corner you will get the username (we have a has logged in and besides that there is a drop down and you can go to )

User-> configure window

* api token
* change your password
* setup any public keys for authentication

user1 has all that permissions as admin, he can go to manage and he can do any changes in the configurations ,manage the plugins and everything ok which is not the ideal scenario.

Roles

In general we have roles to define to every users.

Ex: (1.) admin user has all that roles, he can go to the Settings he can configure the system and everything and then we have.

(2.) Manager can only be able to view the projects

(3). Developers should be able to do the development projects and do anything with the development projects but not with any other project and

(4.) Testers should be able to only access that testing projects and not the development projects

Roles strategy plugin

For this, to manage , We have a plug-in called Roles strategy plugin.

Make all the changes as admin.

Step-1: Install:

1. Download-> hpi -> plugins directory ->restart your Jenkins
2. Manage Jenkins-> manage plugins-> go to available section->(type) role strategy plug -> (Install without restart/Download now and install after restart )
3. If downloaded plugin(hpi) , Advanced section -> have downloaded the HPA file -> upload plugin

Step-2: Manage Jenkins-> configure global security ->enable security( check it)

Step-3: Authorization -> role based strategy

Manage Jenkins -> manage and assigned roles

Three options In manage & Assign roles

1. Manage roles

Manage Roles

1. Global rules:

**Provide us authorization and access on a global level.**

Let me create a new role, Employee

Now, I can control what access i need to give to this role.

So I want that employee should have an overall **Read** access and then

He should also have an overall **view** access

1. Project roles
2. Slave rules

First one ->manage roles ->

Here, we can create rules specific to project.

Let me create a role,

Field: Developer and

Pattern: Dev.\*

Person having our role as developer should be able to access that projects which have their name starting with Dev.

and I'm giving him all the permissions on these type of projects and he should not be able to view or use any other project

Field: tester

Pattern: Test\*

He should only be able to access that test projects. So i'm giving him all the permissions on any project which starting like test and anything after that.

We have created employee Role at a global level and

Assign roles

We have created roles developer and tester at the project level

1. Assign roles : Assign roles to users

Now we have to assign the roles to our users ok so .We created 2 users: user 1 user 2 so. Now again in assign roles, you have

1. global goals and
2. project roles

Let me add our user1, user2 to global and both of them are added to employees.

project roles: i will add my users, but user1 is a developer role and user2 to tester.

We have created roles and assigned to users.

Role strategy macros

I will create new projects and give name Dev project, freestyle project

Another project test project

We have two projects and obviously one is that dev project and another one is that test project

User1 given as a developer role and developer can only view and access the projects which have named starting like Dev and

Tester can access to projects which have named starting like test

so let us now are validate that i'm logging out as an admin and

i will login as user1 and you see now this particular

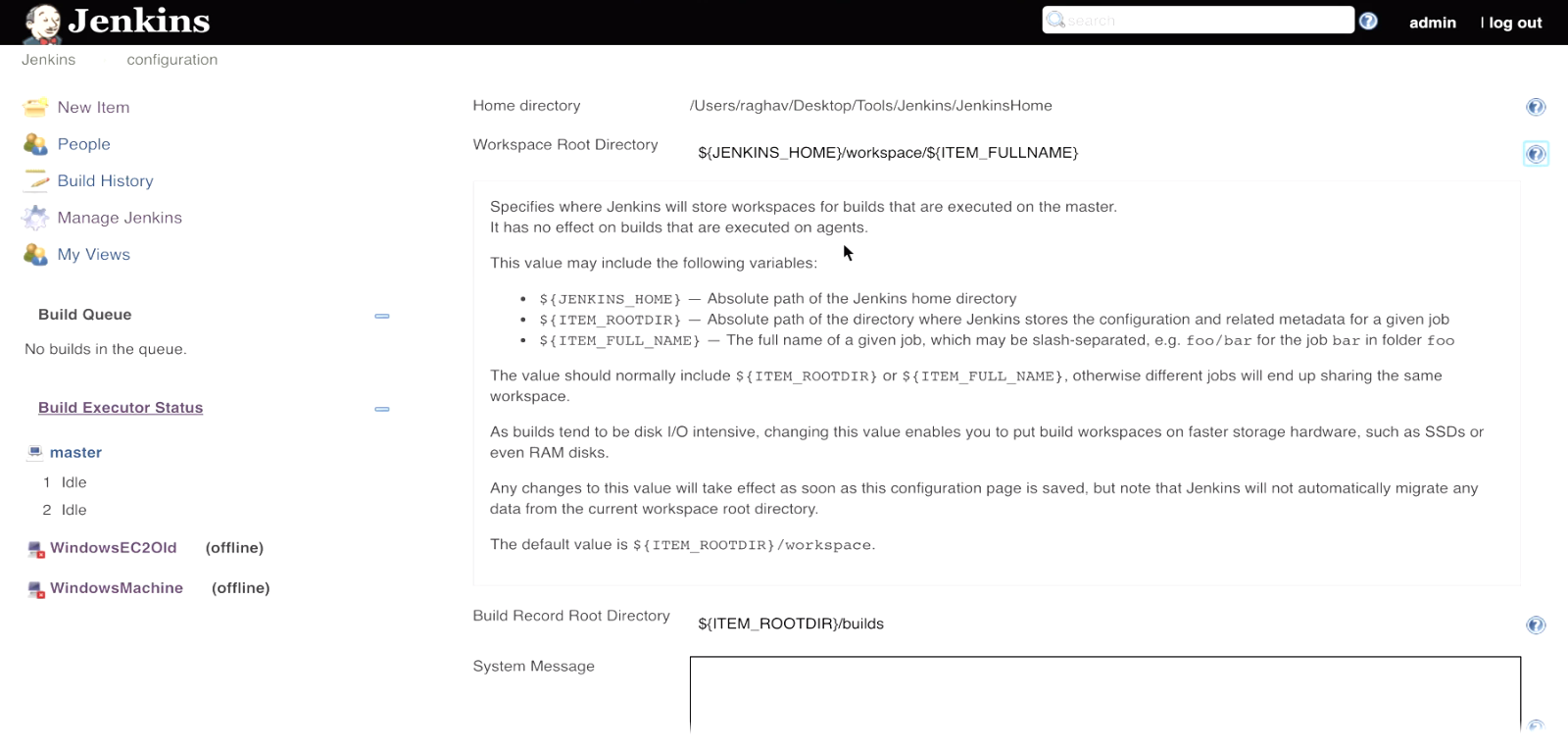
* user1 can see the Dev project and
* not any other project and also
* he does not have any access to the manage Jenkins menu so he cannot change the settings

This is how we can

1. create new users
2. manage users
3. assigned roles and
4. control the authorization and authentication in Jenkins

**Jenkins Beginner Tutorial 6 - Basic Configurations**

Manage Jenkins-> Configure system



1. Home Directory (plugins, logs, jobs stored)

(ADVANCED)

1. Workspace Root Dir: Jenkins store build info
2. Build Record Root Dir: Where Jenkins store build records
3. System Message:

* On Jenkins home screen, we can see this message displayed.
* If we want to notify users with message, we can use this…
* We can also use html tags.

<h1>This is example system generated </h1>

* If not able to use this, **Manage Jenkins-> Configure Global Security->MarkUp Formatter-> Safe Html**, instead of Plane text

1. # of executors: No. of parallel jobs, this particular Jenkins instance should be able to run.

Jenkins should run max. 5 parallel jobs one time.

1. Labels:

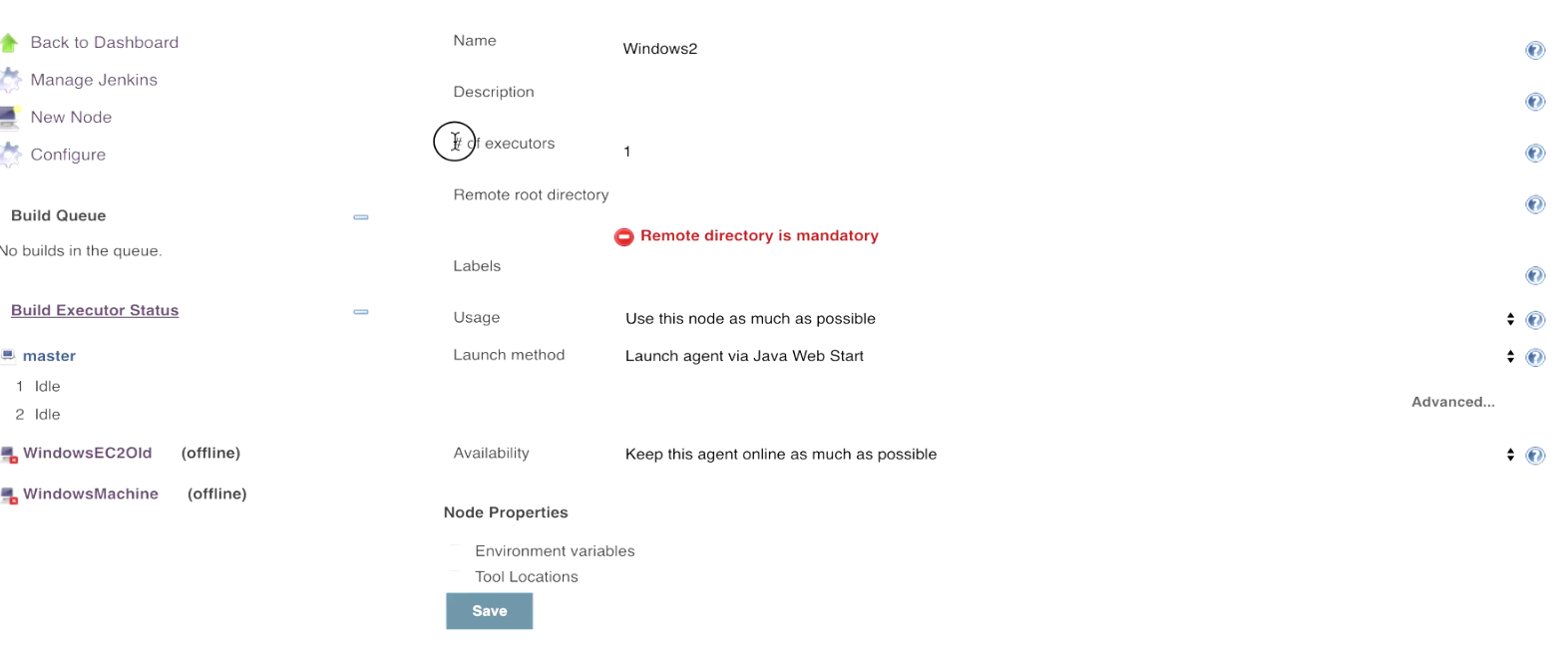
In Jenkins, we have system for distributed builds. This means, I can make my Jenkins as master machine and then can add nodes to this particular Jenkins.

(I can add different machines to Jenkins)

Execution should take on different nodes, but not on master.

Manage JENKINS-> Manage nodes-> New Node->

(Adding new nodes meaning adding new machine)



Name: windows

# of executors: Number of concurrent builds, that Jenkins may perform on this agent.

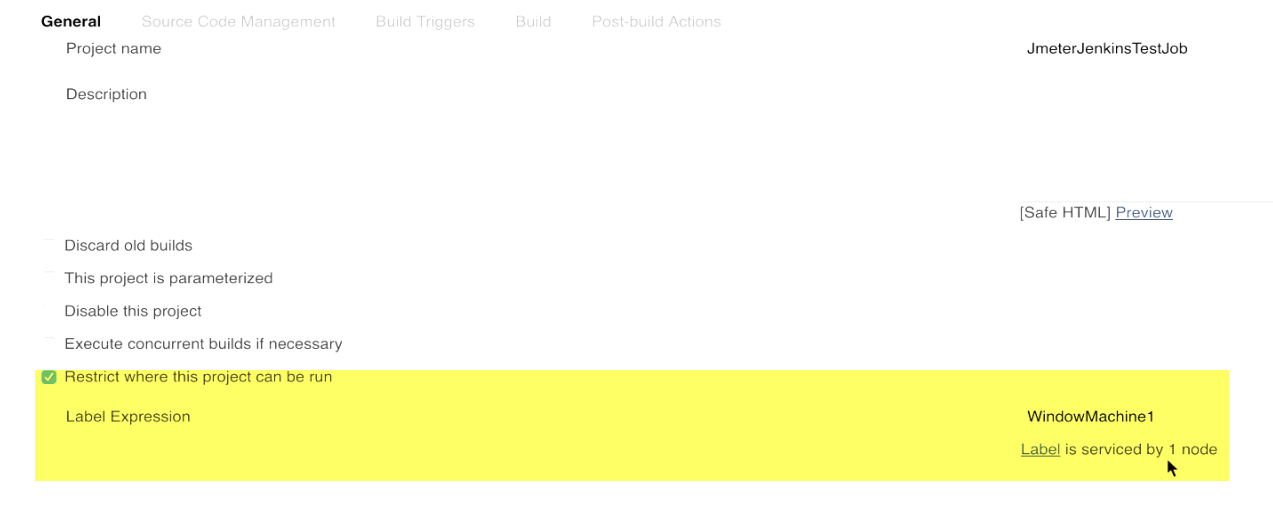
Labels: WindowsMachine

Usage: Only build jobs with label expressions matching this node.

I have added nodes and added label to those machines.

Now; let me go to existing jobs-> Configure->

General-> Restrict where this project can be run

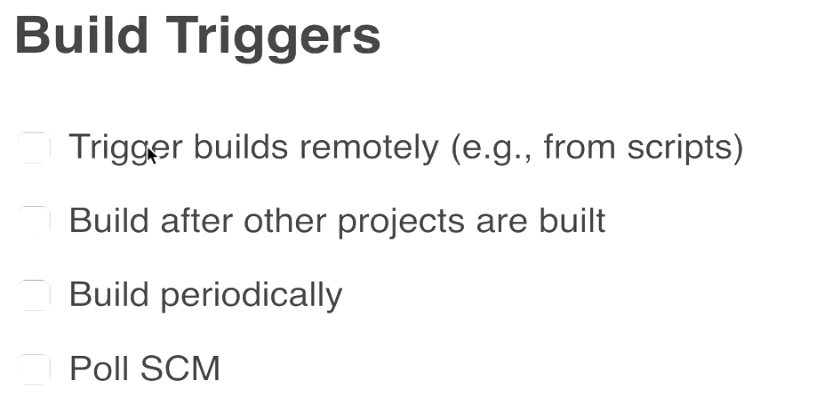


1. Quiet period:

No. of seconds that this particular Jenkins instance should wait before triggering a job.

The job can be triggered as soon as build takes place.

In Job configuration, we have build triggers:



Poll SCM: as soon as someone commit code in code repo. Like git the job will get triggered.

But there be a time gap/sleep interval between job getting triggered and build actually getting into source code repository.

Why?

If Quiet period =0, job will get triggered as soon as trigger coming to Jenkins.

For ex: Build committed into git, job will trigger.

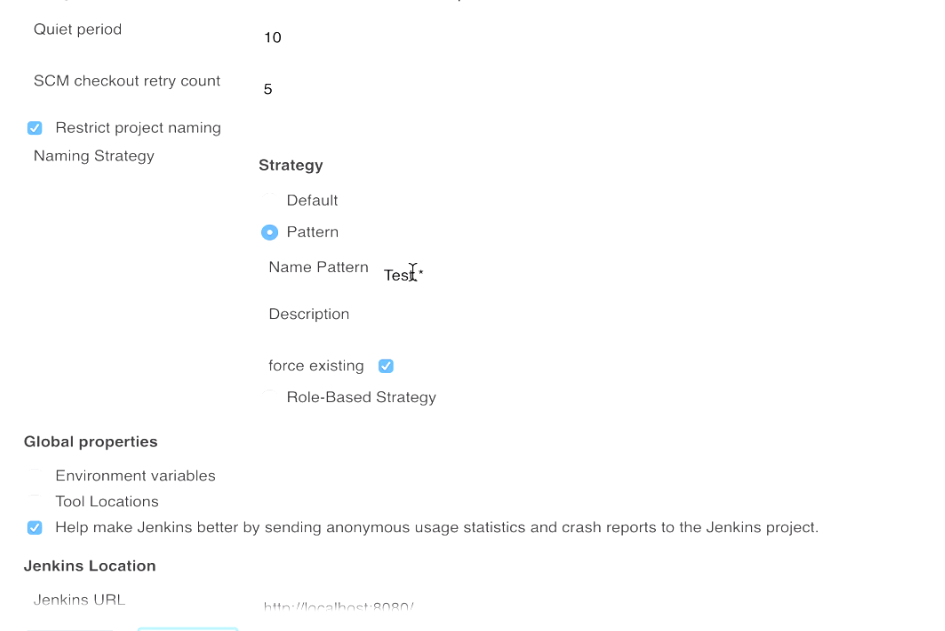
**Sometimes, Commit is not successful or it takes some time to commit all the files.**

1. SCM Checkout retry count:

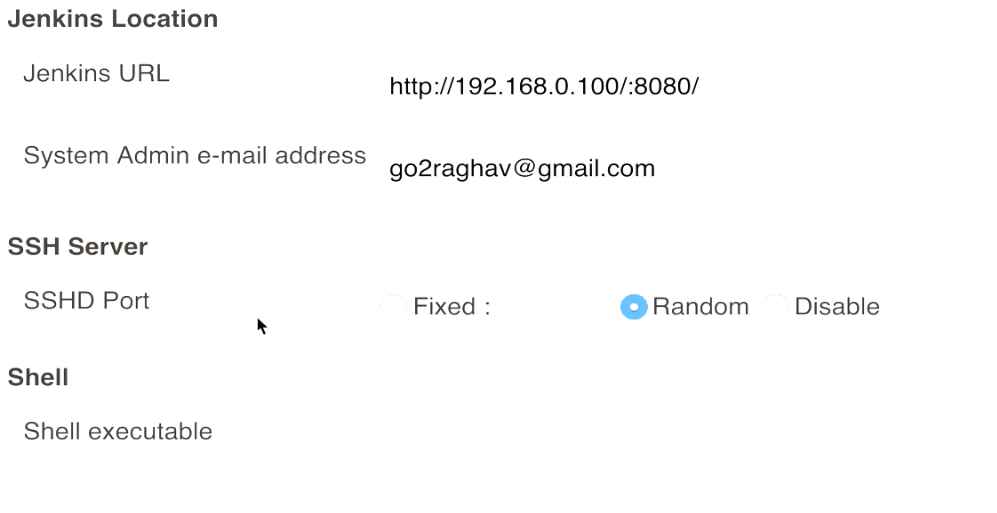
In Poll SCM, it will try to connect to Source code repo. And will try to get latest build information.

In case it fails, what is maximum retry count, Jenkins should do.

1. Restrict Project Naming:

If we want to restrict project naming like starting with Test etc..

1. Global Properties
2. Jenkins url



1. Admin email
2. **SSH server:**

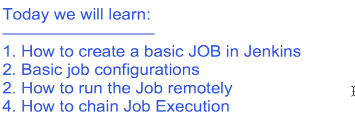
If we want jenkins to run jobs remotely and connect to servers, we can give port number here.

1. **SHELL**: By default, Jenkins have bash shell, which it executes shell commands.

Suppose, if we want to give some root dir or some shell (suppose we have Cygwin and want all executables commands to be executed by Cygwin…we can give path of executable).

Ex: C:\windows\sys32\cmd.exe

**Jenkins Beginner Tutorial 7 - Getting started with JOBS**



Create Job

Create Job:

Jenkins->new item 🡪 Create a program 🡪hello world freestyle project



1. **sections:**

1.) General

2.) Source Code Management

1. Build Triggers
2. Build
3. Post-build actions
4. **General:** Quick description of project



1. **Source Code Management**

If we need to build from Source Code Management tools like git, SVN. We will need to add plugin

1. Build Triggers

Poll SCM : We have source code mgmt system.

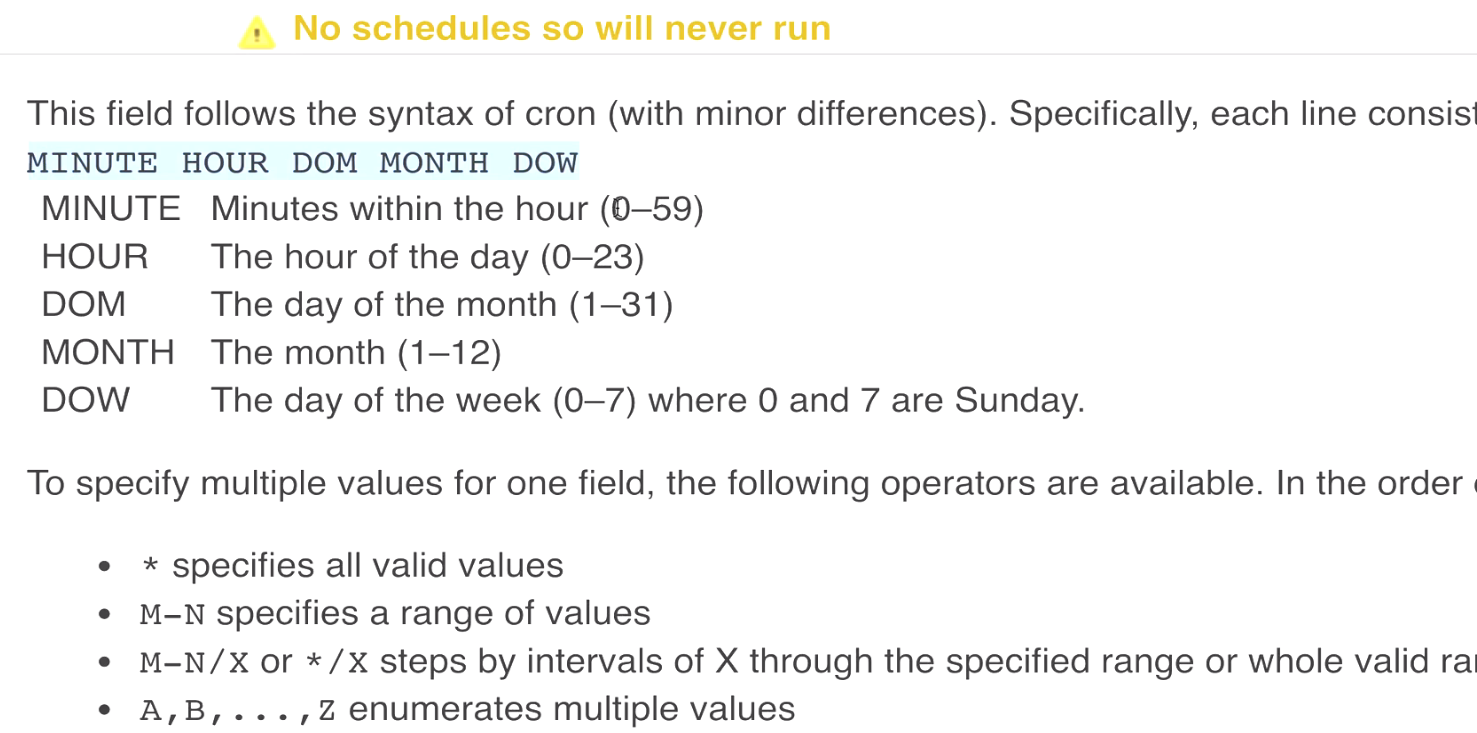
In case, if we have configured with any SCM like git.

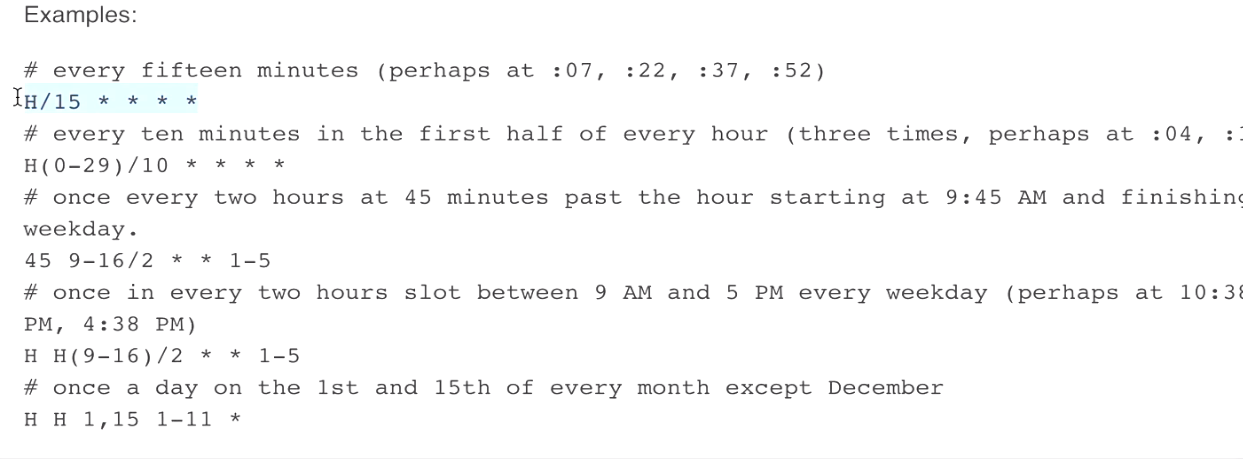
As soon as build takes place in the repo…Our job should get executed.

**Build Periodically: We can assign expression here.**

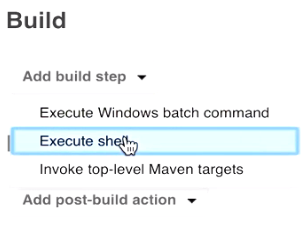
**Our job will be executed at some particular interval**

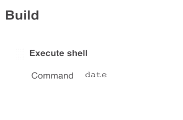




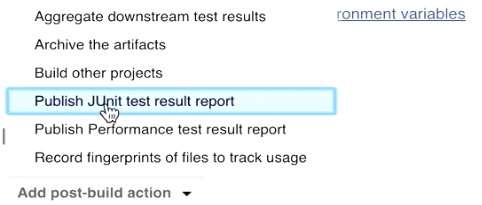


1. **Build**





1. **Post-build actions:**



**Dashboard:**

1. Status of last build: Build—Blue(success), (red-failed), not build (grey)
2. Weather report(Based on status of last 5 builds)

: Aggregate status of recent builds.

Ex: (1 out of last 4 builds failed.----cloud

3 out of 5 -🡪 rainy cloud

1. Last success: 46 sec, build number
2. Last failure: 1min, build number
3. Last duration: Duration of last build

**This column takes into account, last 5 run of this particular job/project.Based on count of successful/failures in last 5 runs. Icon gets changed.**

**Console output**

**Build now🡪 Console output**



**Build history**

**Build history**

Step-2: Trigger Job remotely

**Jenkins-> Project (test1)-> Configure**

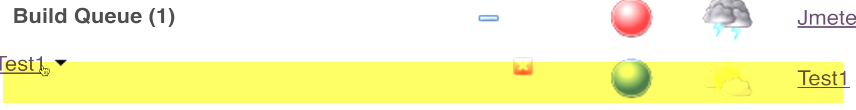
**Build triggers**



**JENKINS\_URL: IP:port**

**TOKEN\_NAME: Complex string-**🡪**password(1234)**





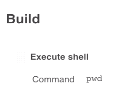


Step-3: How to chain Job executions

**New-> Test2-> freestyle**

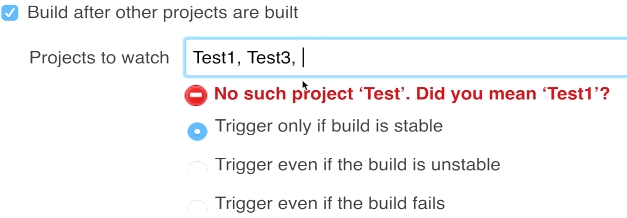


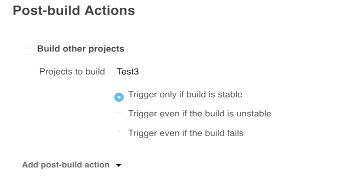
**New-> Test3-> freestyle**



**We have 3 projects test1,2,3**

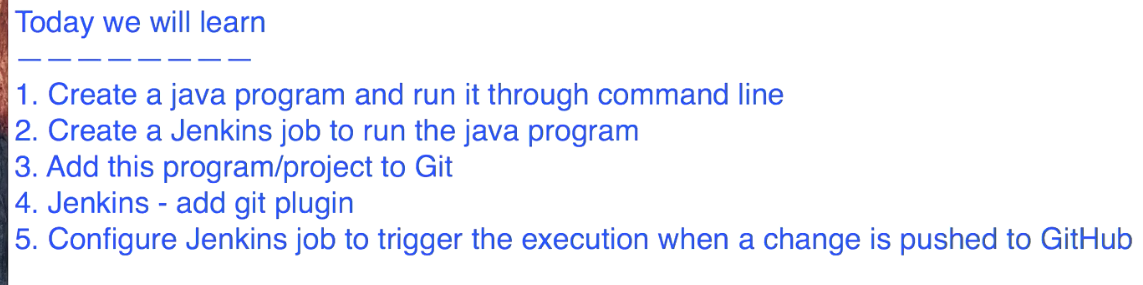
**Test2->Configure**





**As soon as test1 completed->test2-> test3->test3**

**Jenkins Beginner Tutorial 8 - Jenkins integration with GIT (SCM)**



Step-1: Create Java program

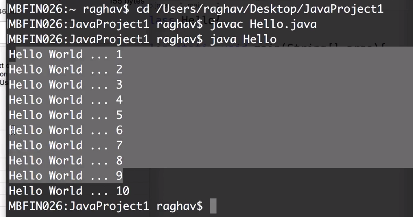
Printing hello world or 10

Eclipse/any editor / notepad

We’ll create a class hello and we will create the main method and

We will have our program written inside. For loop for iteration 1 to10

And inside the for loop i will print something “hello world”

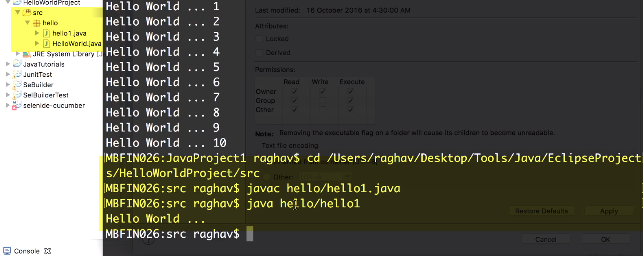


If we created program Java on Eclipse.

In that case also, you can run it through command-line. What we have to do is you have to the location just above your package,

(If you have created a custom package then you will go to directly just about the package so in my case the SRC)

Go to this particular directory on my command line



Step-2:

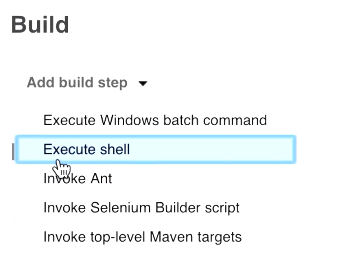
Create a Jenkins job

Create Job:

I will go to new item 🡪 Create a program 🡪hello world freestyle project

Windows: execute windows batch command

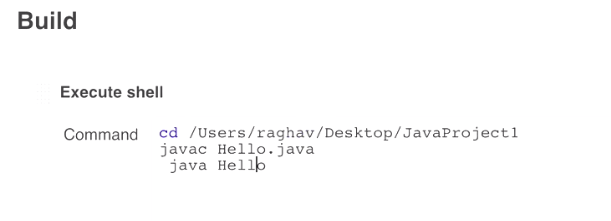
Linux: you will see execute shell



Go to my location of my java file first.

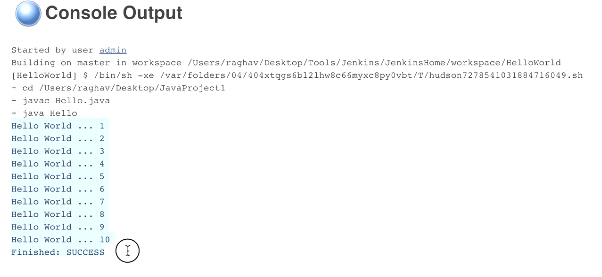
Javac hello.java

java hello



**Build Now**

**Console Output:**



Step-3: Add Project to Git and GitHub

Initialize the repository

Git init

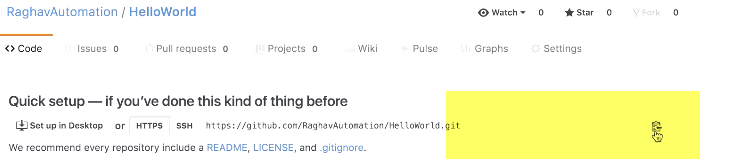
Git status-🡪two files which are not yet added

Git add🡪 git commit- m hello world program

I have to push it to a remote repository

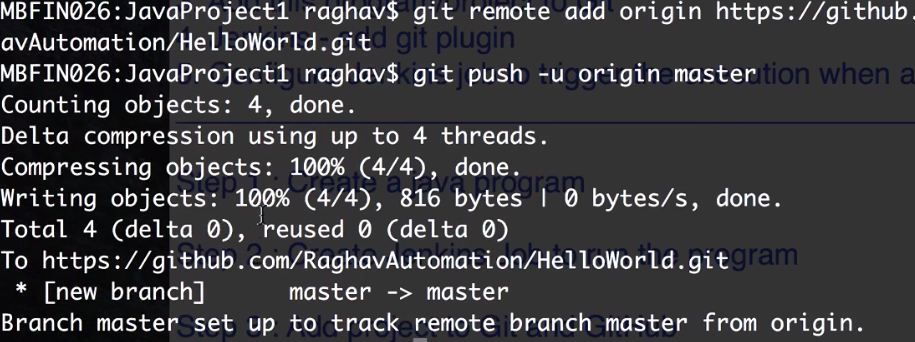
**Github** account: Create new repo..

i will copy this location of the repository

 **Client:** git remote add origin and the location of the repository

For SSH: we need to generate public and private keys, and add to github website

Use https 

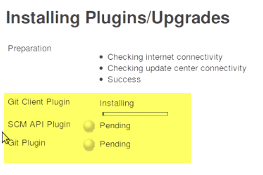


**Jenkins🡪 Manage Jenkins🡪 manage plugins -🡪 Git Plugin**

**(Available and find get plug-in and install download and install it**

Step-4:

Create Jenkins job to triggered whenever you make any changes in your project



**Dependencies of git**

This is the project🡪 configure 🡪 Source Code Management 🡪git option

name of the repository: urlll

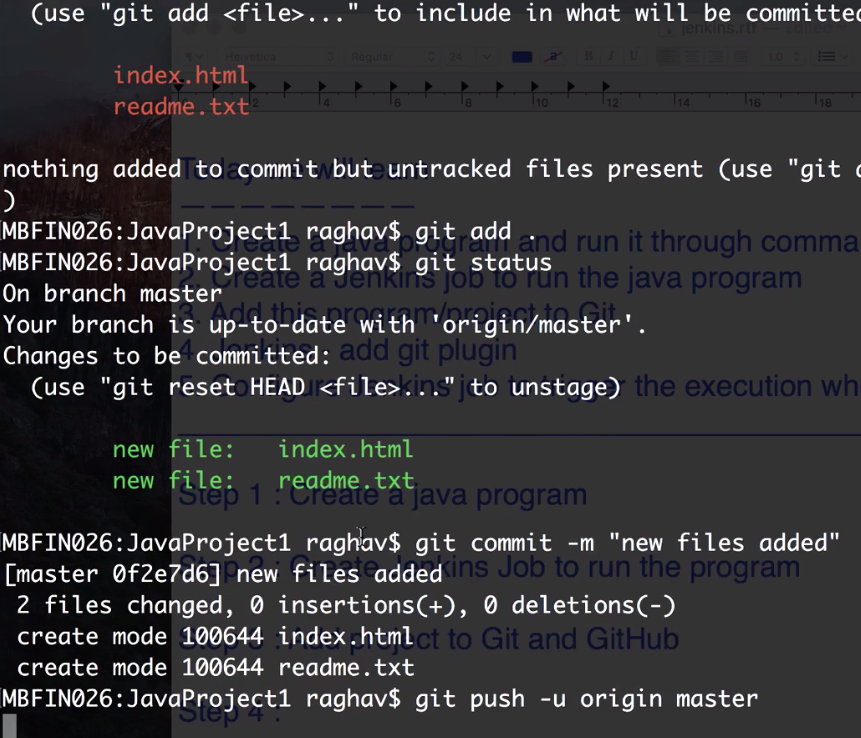
userid/passwd

Build triggers: Poll SCM

I have to give the Cron expression that how often and what water intervals, you want this particular job to check the source code repository to look for the changes.

You can give any expression here based on your requirement of the project.

Five stars 🡪check the repository every min for any changes

This job will check to git repository every minute. If any change it will trigger this job. 



You can see there is just one iteration and execution of this job, which we have done manually.

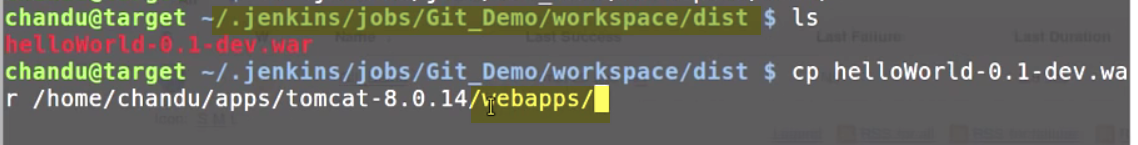
Next iteration will happen whenever we make any changes and push to our github.

Now you can see this job, it has started triggering that build.

It has started the job on itself and if you see the console output.

You can see this is started by an SCM change.

Deploy web application to tomcat server



Copy war file from .jenkins workspace dir to tomcat webapps

Localhost:8080(tomcat)--manage apps🡪



