Installing Jenkins on AWS EC2-Instance Redhat system :

Launch AWS EC2-Instance Process:

1. We have to login AWS console.
2. Then we have to select RedHat ec2 instance.
3. In Choose AMI Tab we have to select RedHat enterprise linux needs to be select
4. In Choose Instance type we have to select t2.micro General purpose.
5. Then click on Review and Launch Button
6. Then click on Launch Button.
7. When click on launch we have to download key pair with selecting option called new keypair.
8. Then click on Launch Instances button.

Connect to AWS EC2 Instance now:

Now Redhat ec2 instance has been created so we have to go to EC2 instance dash board and we have to provide name for new instance. Like “Jenkins master server”.

1. Select just now created EC2 instance and click on connect button above one.
2. Then copy the ssh command which is available under example one.
3. We have to open terminal and go to Downloads folder location since our pem file is available there.
   Cd Downloads
4. And enter sopied ssh command
5. And says yes and enter
6. Now we can see we have been connected successfully to AWS instance with ec2 user.

Now we have to follow the installation steps to install Jenkins on this instance.

1. sudo yum install wget
2. `sudo wget -O /etc/yum.repos.d/jenkins.repo http://pkg.jenkins-ci.org/redhat/jenkins.repo`
3. `sudo rpm --import https://jenkins-ci.org/redhat/jenkins-ci.org.key`
4. `sudo yum install jenkins`
5. sudo yum install java
6. sudo service Jenkins start
7. sudo chkconfig Jenkins on

Success Jenkins is installed now and running fine.

Now Access Jenkins from a web browser.

For this Enable TCP/IP connections for the RHEL instance that we just know launched.

How to Enable TCP/IP connections to instance.

Now go to AWS RED HAT instance and find the security wizard used.   Go to security groups .   Select the security wizard …   Actions -→ Edit inbound Rules -> Allow all TCP/IP or Allow all traffic and select anywhere not custom as source tab.

And now go to browser and type public ip:8080.

Unlock Jenkins using initial admin password, copy the location shown on the screen and go to instance and get the password using below command

```
sudo cat /var/lib/Jenkins/secrets/initialAdminPassword
```

And copy the password and paste into the browser and click on enter.

And select plugins installation button.

And try to create a new user or else continue with admin user.

Now our Jenkins installation setup is done on AWS redhat ec2 instance successfully.

Install GIT and GIT HUB tools and plug ins in Jenkins as well.

So here we are going to run a simple maven project.

So We have to install Maven here and configure in Global tool configuration tab under manage Jenkins tab and maven plug in in Jenkins as well.

We need to install Java as well

```
sudo yum install java*
```

We need to install MAVEN and we have to set up MAVEN like windows for this please follow below steps,

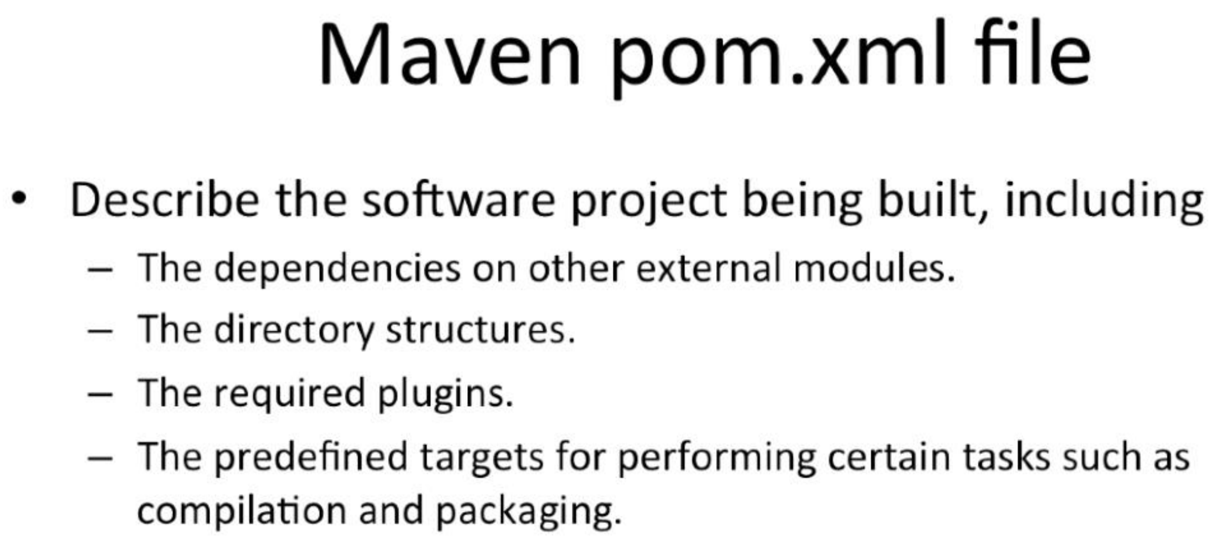
http://www.techoism.com/install-apache-maven-on-centos-765/

What does Maven do ?

- Maven describes how the software is built.
- Maven describes the project’s dependencies.

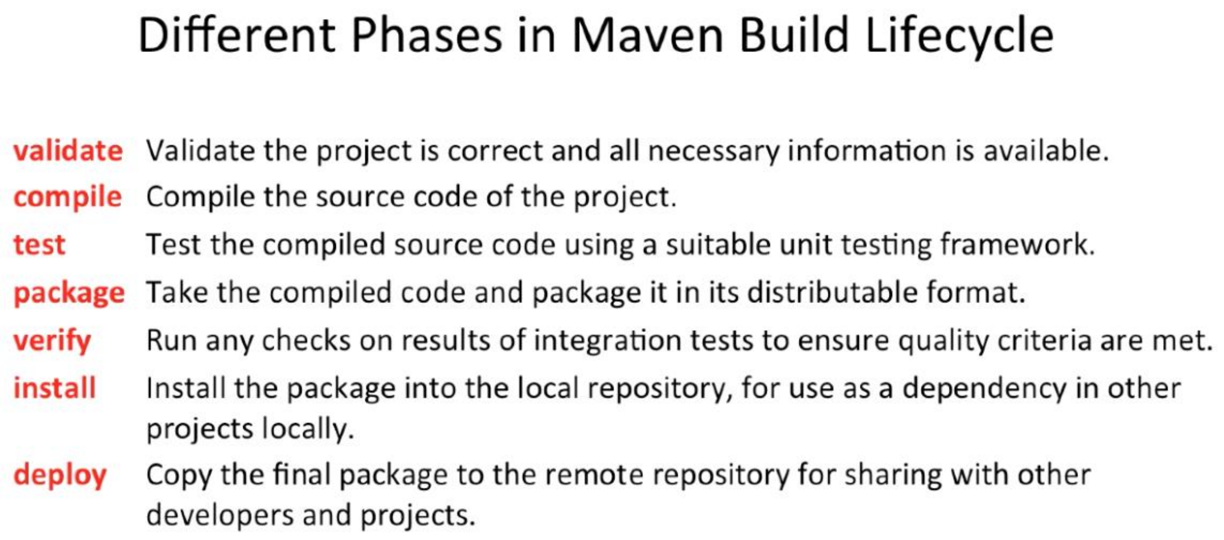
<u>Configure Jenkins for our Maven – Based project</u>.

We have to go Manage Jenkins and Global tool configuration tab.



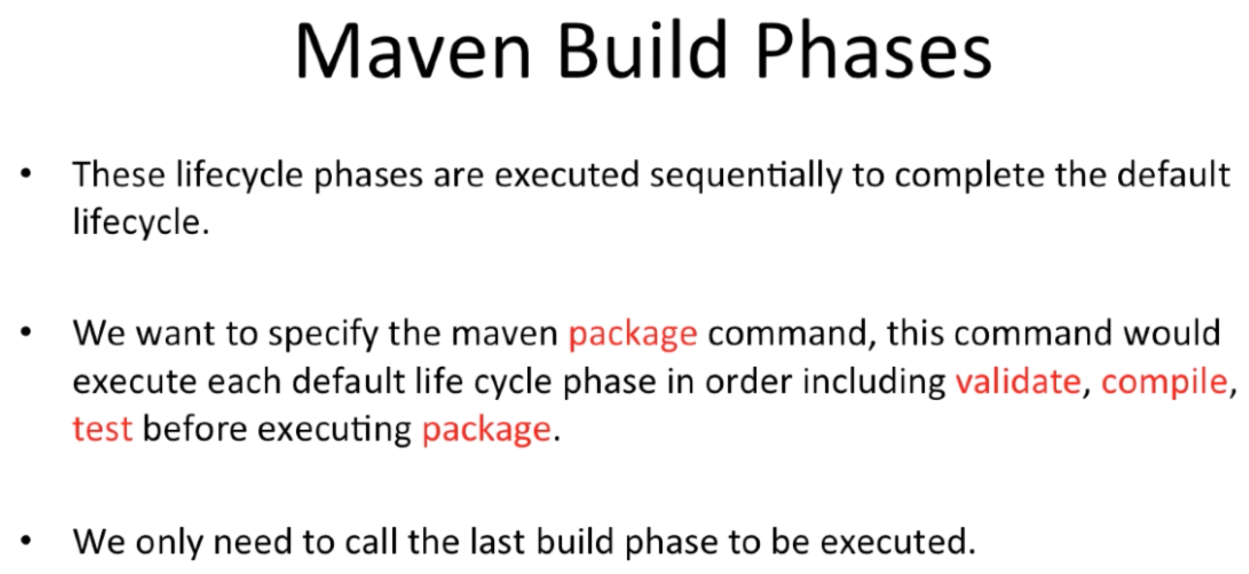
Maven pom.xml file

- Describe the software project being built, including
  - The dependencies on other external modules.
  - The directory structures.
  - The required plugins.
  - The predefined targets for performing certain tasks such as compilation and packaging.



Different Phases in Maven Build Lifecycle

validate	Validate the project is correct and all necessary information is available.
compile	Compile the source code of the project.
test	Test the compiled source code using a suitable unit testing framework.
package	Take the compiled code and package it in its distributable format.
verify	Run any checks on results of integration tests to ensure quality criteria are met.
install	Install the package into the local repository, for use as a dependency in other projects locally.
deploy	Copy the final package to the remote repository for sharing with other developers and projects.



Maven Build Phases

- These lifecycle phases are executed sequentially to complete the default lifecycle.
- We want to specify the maven package command, this command would execute each default life cycle phase in order including validate, compile, test before executing package.
- We only need to call the last build phase to be executed.

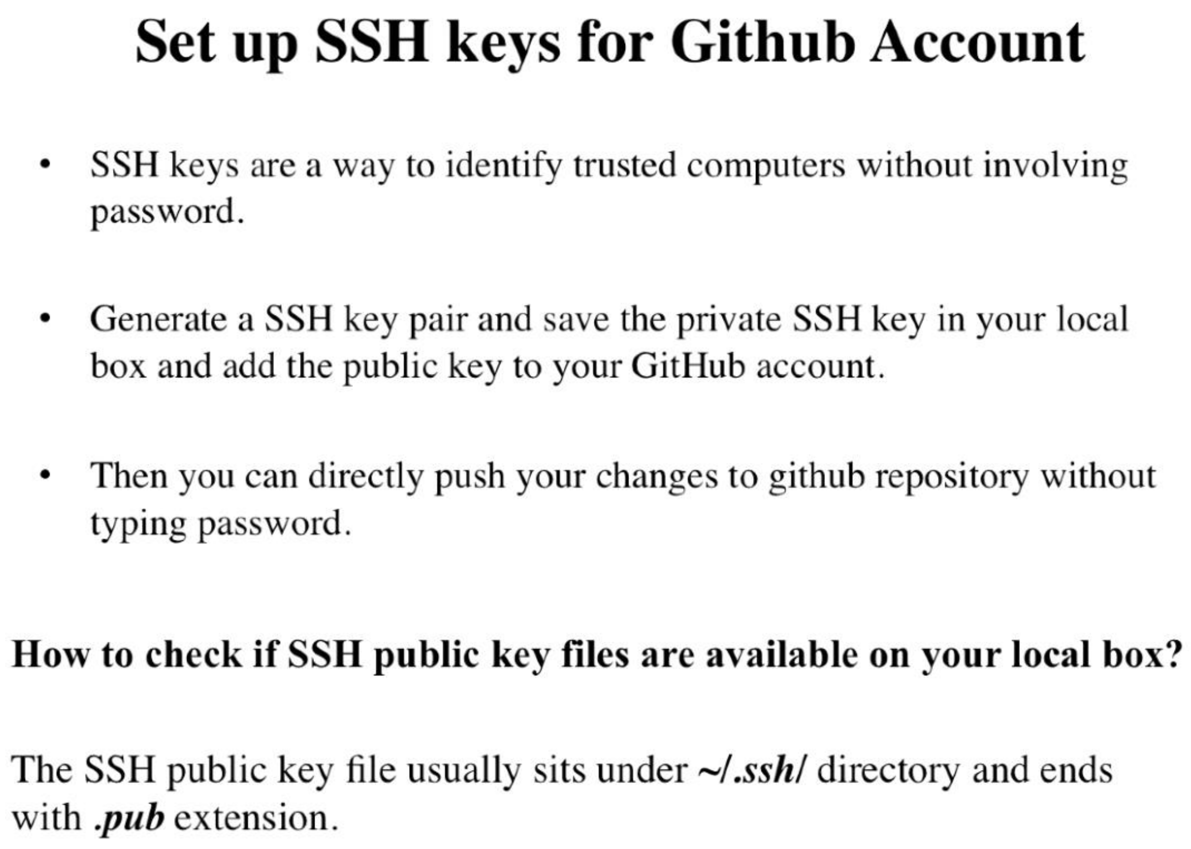
<u>Configure Jenkins for our Maven – Based project</u>.

- Click on New Item and give project name as maven_project
- Configure github repo
- Configure build step as invoke maven based job
- And give goal as **clean package**

And click on save and click on build now.

It can be successful build.

Now we can make same Maven build automatic way using poll scm



Set up SSH keys for Github Account

- SSH keys are a way to identify trusted computers without involving password.
- Generate a SSH key pair and save the private SSH key in your local box and add the public key to your GitHub account.
- Then you can directly push your changes to github repository without typing password.

How to check if SSH public key files are available on your local box?

The SSH public key file usually sits under ***~/.ssh/*** directory and ends with ***.pub*** extension.

Checking for existing SSH keys:

https://help.github.com/articles/checking-for-existing-ssh-keys/

Generating a new SSH key and adding it to the ssh-agent:

https://help.github.com/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent/

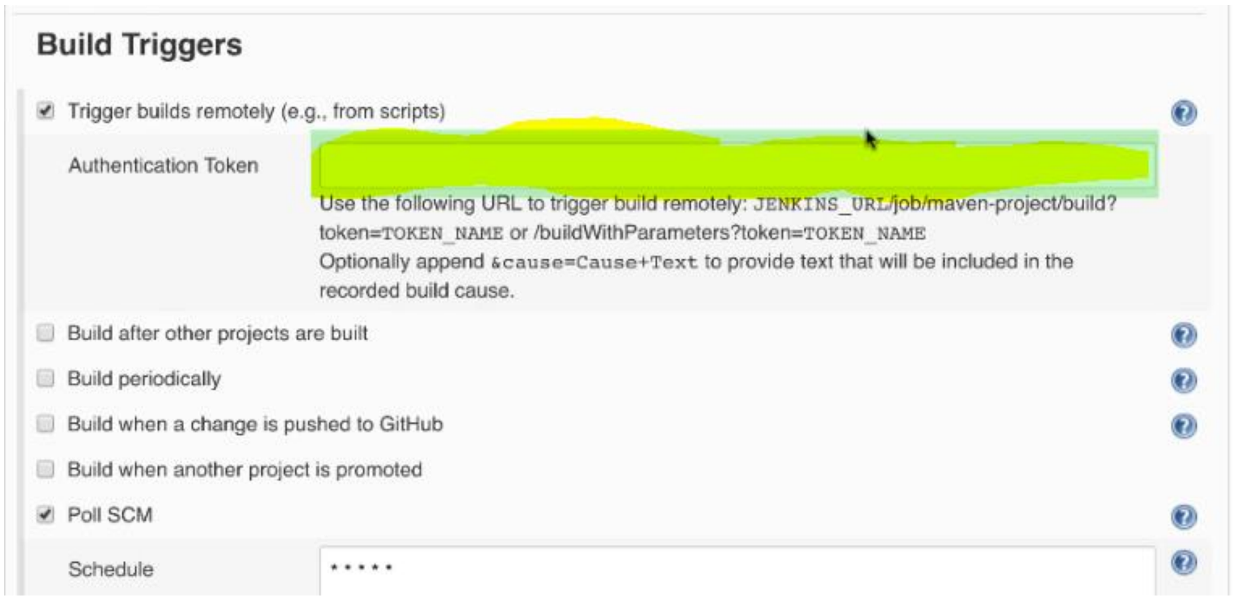
Adding a new SSH key to your GitHub account:

https://help.github.com/articles/adding-a-new-ssh-key-to-your-github-account/

After doing above steps when we want to push changes from our system to gthub no need to provide any password. This way we can do poll scm when we can have new files in git hub account repository.

Now Other build type,

Other Build Triggers of Jenkins



Build Triggers

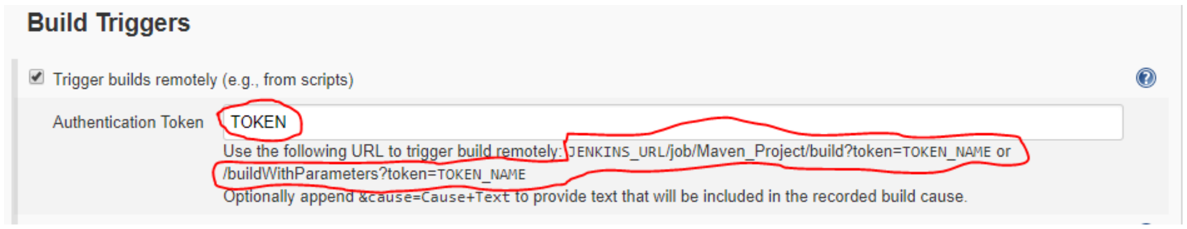
- [x] Trigger builds remotely (e.g., from scripts)

Authentication Token

Use the following URL to trigger build remotely: `JENKINS_URL`/job/maven-project/build?token=`TOKEN_NAME` or /buildWithParameters?token=`TOKEN_NAME`
Optionally append `&cause=Cause+Text` to provide text that will be included in the recorded build cause.

- [ ] Build after other projects are built
- [ ] Build periodically
- [ ] Build when a change is pushed to GitHub
- [ ] Build when another project is promoted
- [x] Poll SCM

Schedule `* * * * *`



Build Triggers

- [x] Trigger builds remotely (e.g., from scripts)

Authentication Token TOKEN

Use the following URL to trigger build remotely: `JENKINS_URL`/job/Maven_Project/build?token=`TOKEN_NAME` or /buildWithParameters?token=`TOKEN_NAME`
Optionally append `&cause=Cause+Text` to provide text that will be included in the recorded build cause.

Using this option we can generate builds from outside using TOKENS.

Lets take a example here for my maven project I have selected Trigger builds remotely option and I have provided token name as “TOKEN” only.

Now I have copied below link and saved the build.

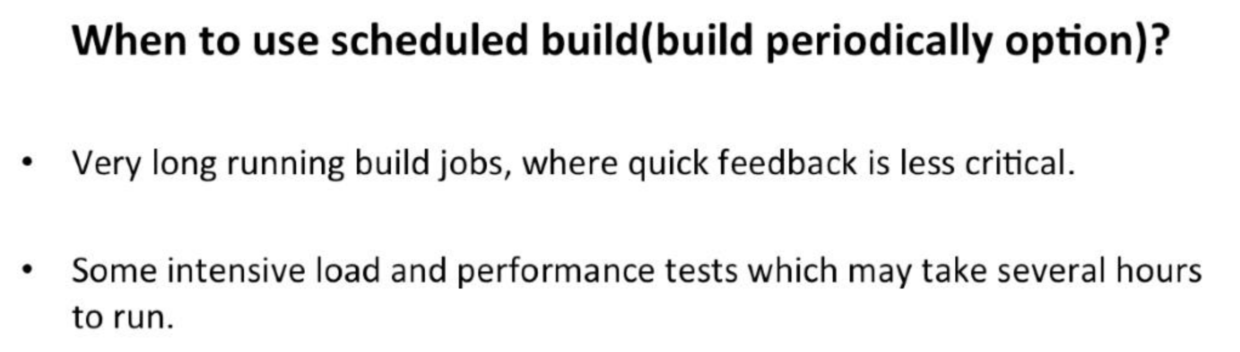
Now went to new browser window and paste the copied link and given token name as “TOKEN” and I did press enter. Now the build has been run you can go to that build history and check new build has been run. Below is the example link for new browser.

http://13.126.253.76:8080/job/Maven_Project/build?token=TOKEN

We have to replace with actual ip or localhost:8080 with Jenkins URL in above copied link.

Next is Build project other builds are built: this will come when we have to make builds are in pipeline.

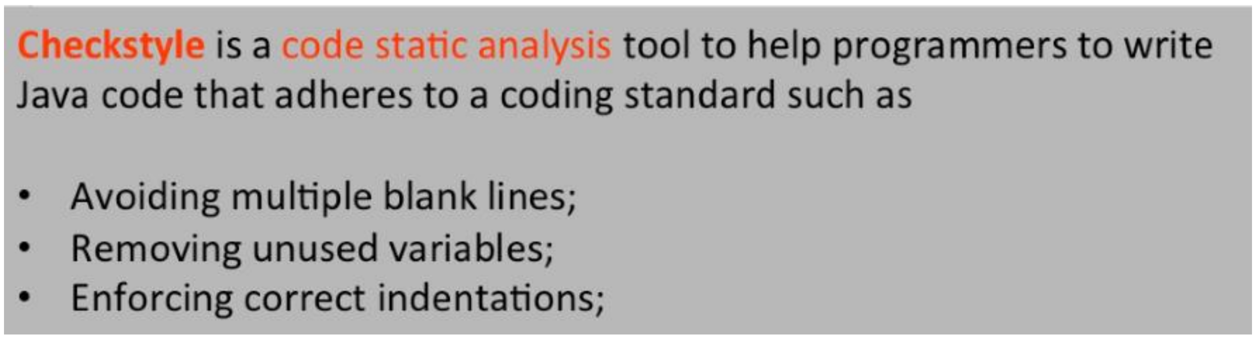
Next one build is periodically.



When to use scheduled build(build periodically option)?

- Very long running build jobs, where quick feedback is less critical.
- Some intensive load and performance tests which may take several hours to run.

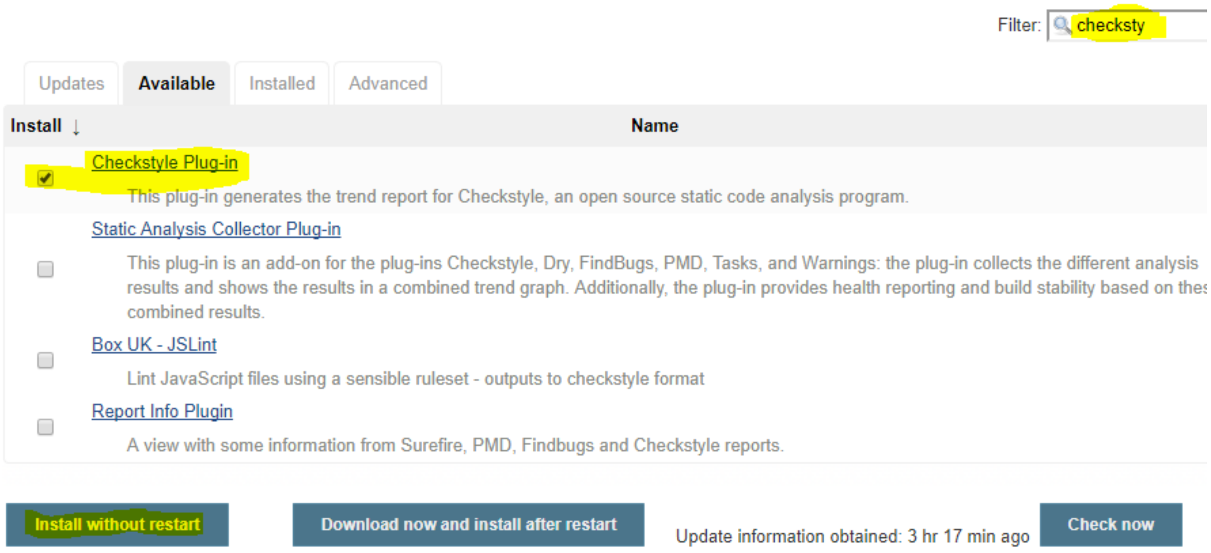
Jenkins project with code quality metrics report:



Checkstyle is a code static analysis tool to help programmers to write Java code that adheres to a coding standard such as

- Avoiding multiple blank lines;
- Removing unused variables;
- Enforcing correct indentations;

For this we have to install checkstyle plugin into Jenkins.

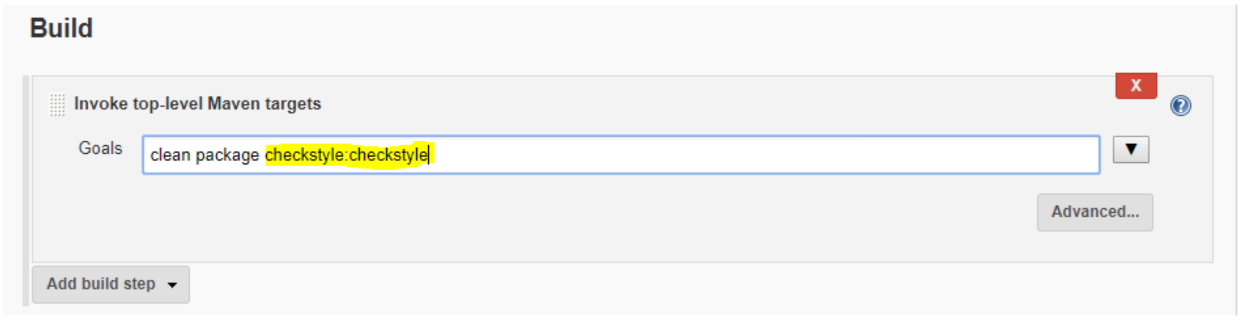

Filter: checksty

Updates | Available | Installed | Advanced

Install ↓	Name
✓	Checkstyle Plug-in This plug-in generates the trend report for Checkstyle, an open source static code analysis program.
☐	Static Analysis Collector Plug-in This plug-in is an add-on for the plug-ins Checkstyle, Dry, FindBugs, PMD, Tasks, and Warnings: the plug-in collects the different analysis results and shows the results in a combined trend graph. Additionally, the plug-in provides health reporting and build stability based on thes combined results.
☐	Box UK - JSLint Lint JavaScript files using a sensible ruleset - outputs to checkstyle format
☐	Report Info Plugin A view with some information from Surefire, PMD, Findbugs and Checkstyle reports.

Install without restart | Download now and install after restart | Update information obtained: 3 hr 17 min ago | Check now

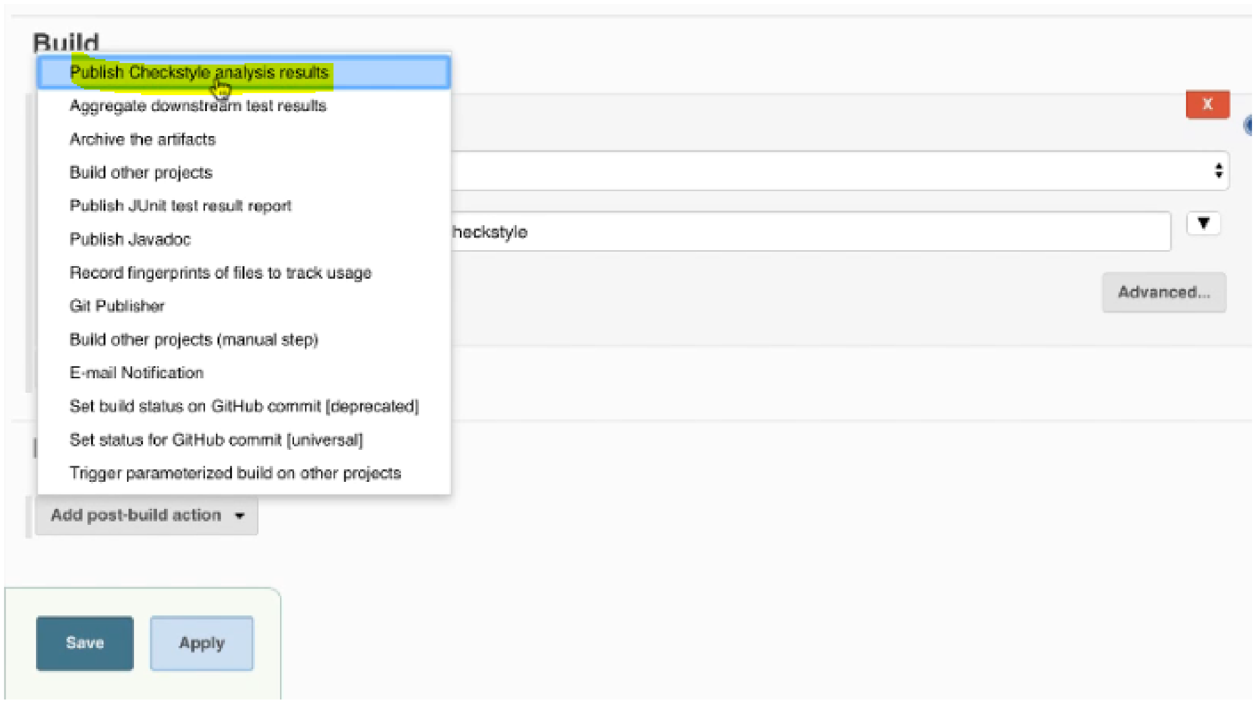
For doing checkstyle plugin we need to enter goal name as checkstyle:checkstyle like below

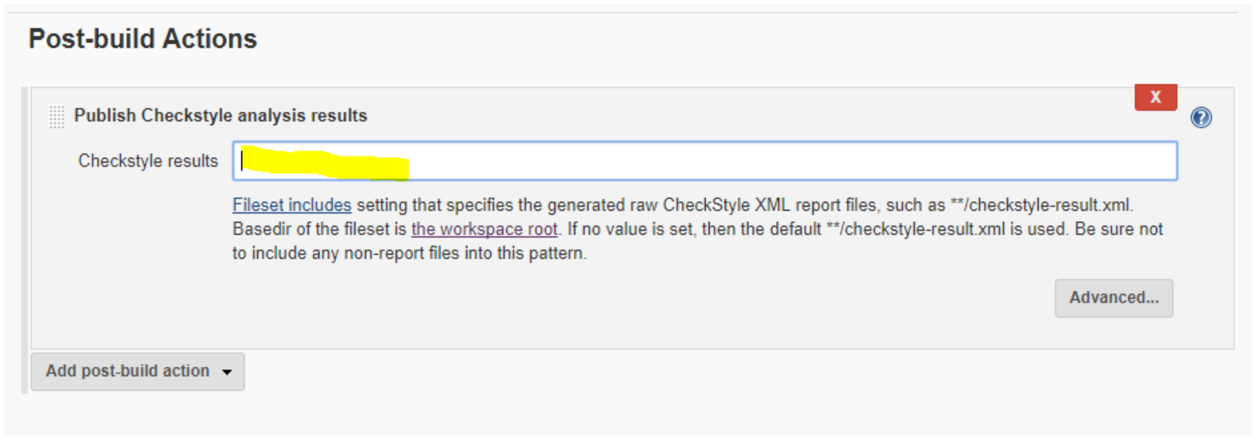

Build

Invoke top-level Maven targets

Goals: clean package checkstyle:checkstyle

And now I want to publish the results outcome of checkstyle plugin. For that we have to add one more step on job page cluster in post build actions like below


Build
Publish Checkstyle analysis results
Aggregate downstream test results
Archive the artifacts
Build other projects
Publish JUnit test result report
Publish Javadoc
Record fingerprints of files to track usage
Git Publisher
Build other projects (manual step)
E-mail Notification
Set build status on GitHub commit [deprecated]
Set status for GitHub commit [universal]
Trigger parameterized build on other projects
heckstyle
Advanced...
Add post-build action
Save
Apply

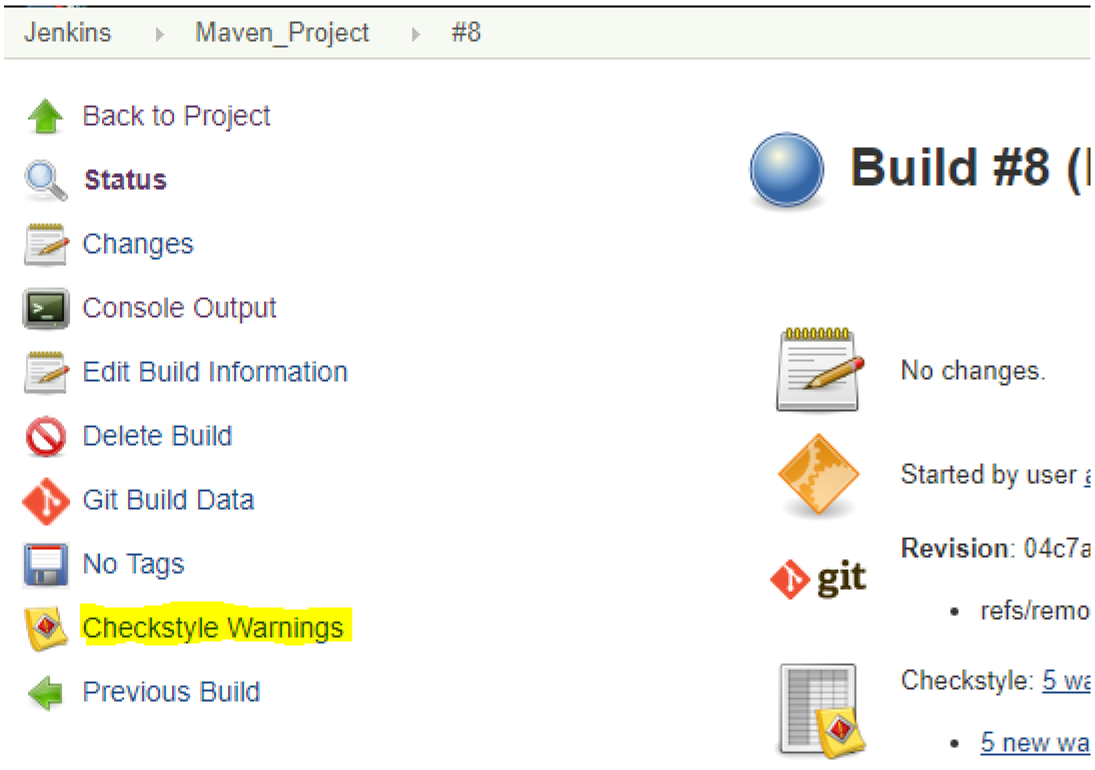


Fileset includes setting that specifies the generated raw CheckStyle XML report files, such as **/checkstyle-result.xml. Basedir of the fileset is the workspace root. If no value is set, then the default **/checkstyle-result.xml is used. Be sure not to include any non-report files into this pattern.

So here we can leave it as empty Jenkins will take care about that.

Now iam going to trigger a new build by clicking on buld now option.

After running the please click on build number example please look at below figure.



No changes.

Started by user a

Revision: 04c7a

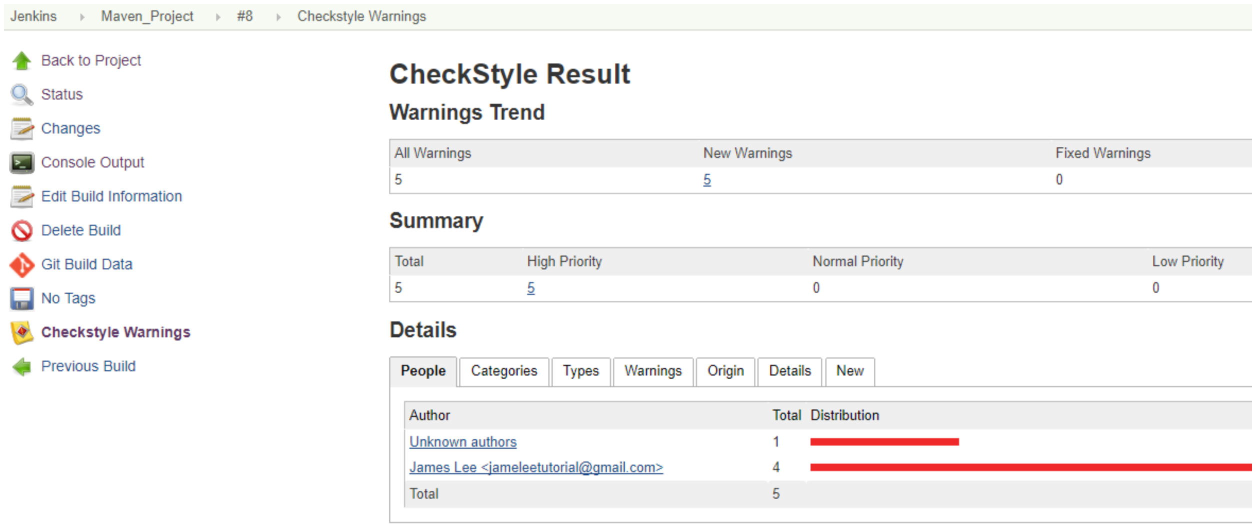
- refs/remo

Checkstyle: 5 wa

- 5 new wa

Now we are able to see new option called checkstyle warnings on home page of build number 8.

Just click on that



CheckStyle Result

Warnings Trend

All Warnings	New Warnings	Fixed Warnings
5	5	0

Summary

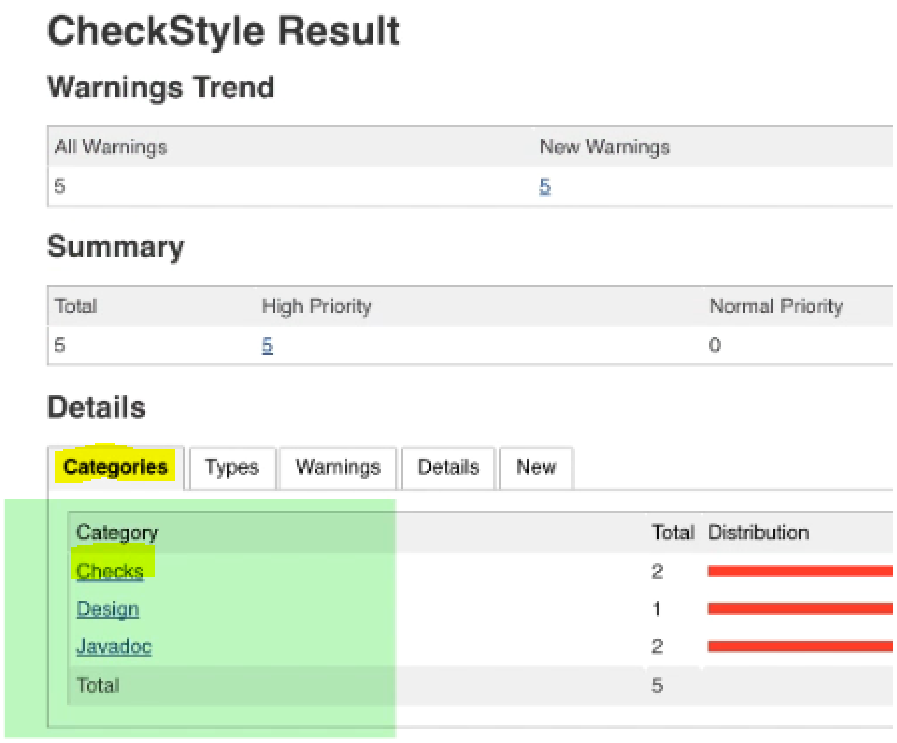
Total	High Priority	Normal Priority	Low Priority
5	5	0	0

Details

Author	Total	Distribution
Unknown authors	1	🟥
James Lee <jameleetutorial@gmail.com>	4	🟥
Total	5	

We are able to see there are 5 failures with error lines and click on that it will take us to exact line of code .All of them high priority.

How to fix those now,



CheckStyle Result

Warnings Trend

All Warnings	New Warnings
5	5

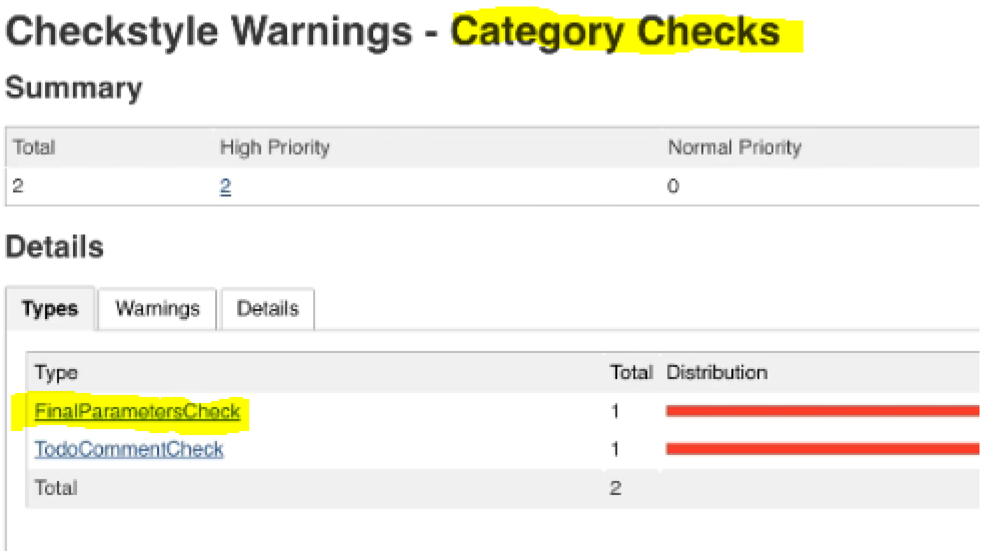
Summary

Total	High Priority	Normal Priority
5	5	0

Details

Category	Total	Distribution
Checks	2	
Design	1	
Javadoc	2	
Total	5	

Go to Checks category.



Checkstyle Warnings - Category Checks

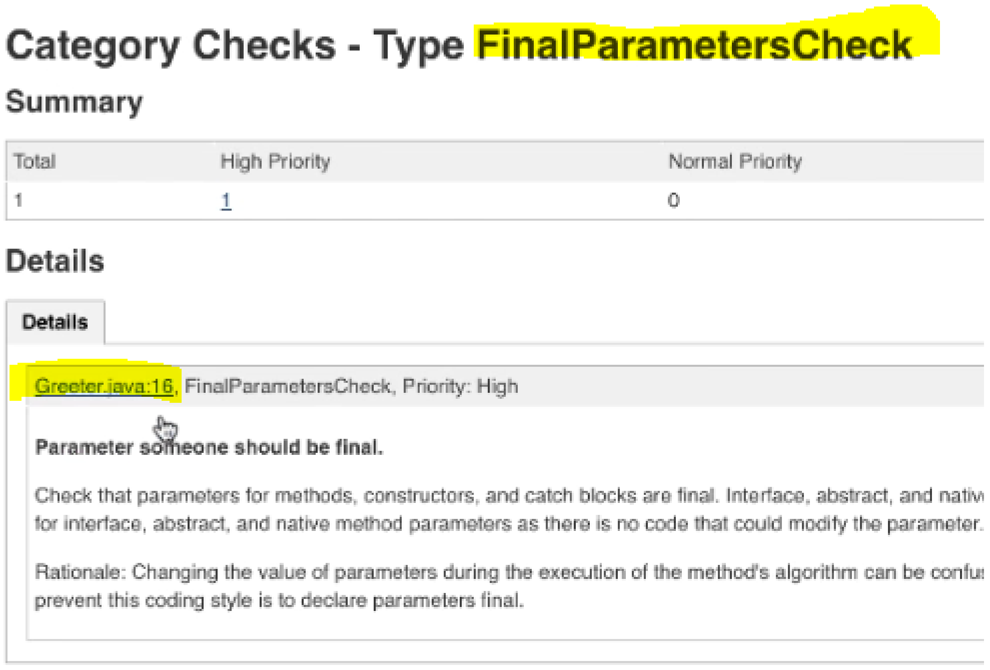
Summary

Total	High Priority	Normal Priority
2	2	0

Details

Type	Total	Distribution
FinalParametersCheck	1	
TodoCommentCheck	1	
Total	2	

Click on FinalParametersCheck for exact warning.



Category Checks - Type FinalParametersCheck

Summary

Total	High Priority	Normal Priority
1	1	0

Details

Details

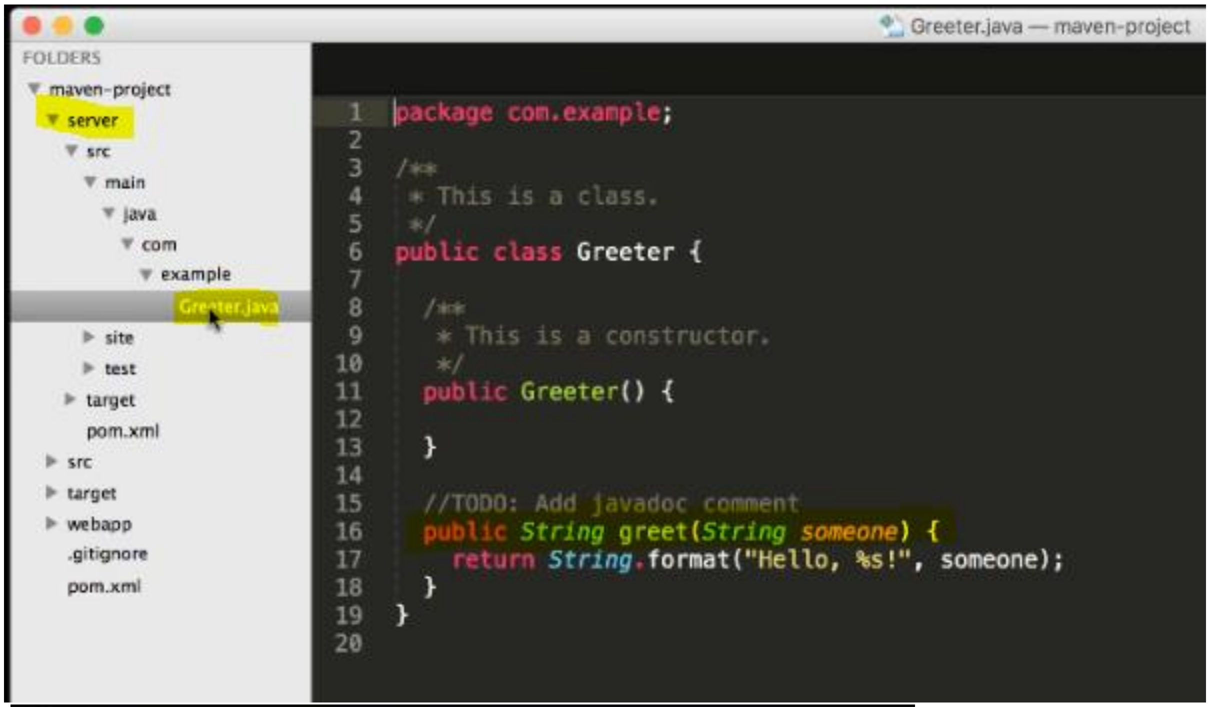
Greeter.java:16, FinalParametersCheck, Priority: High

Parameter someone should be final.

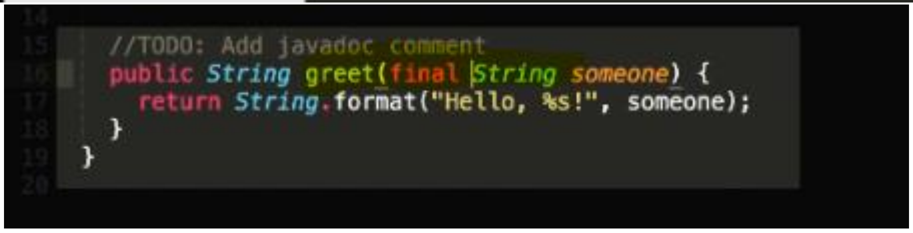
Check that parameters for methods, constructors, and catch blocks are final. Interface, abstract, and nativ
for interface, abstract, and native method parameters as there is no code that could modify the parameter.

Rationale: Changing the value of parameters during the execution of the method's algorithm can be confu
prevent this coding style is to declare parameters final.

There is a line number 16 we are having warning message here click on that t will take us exact line of code now.



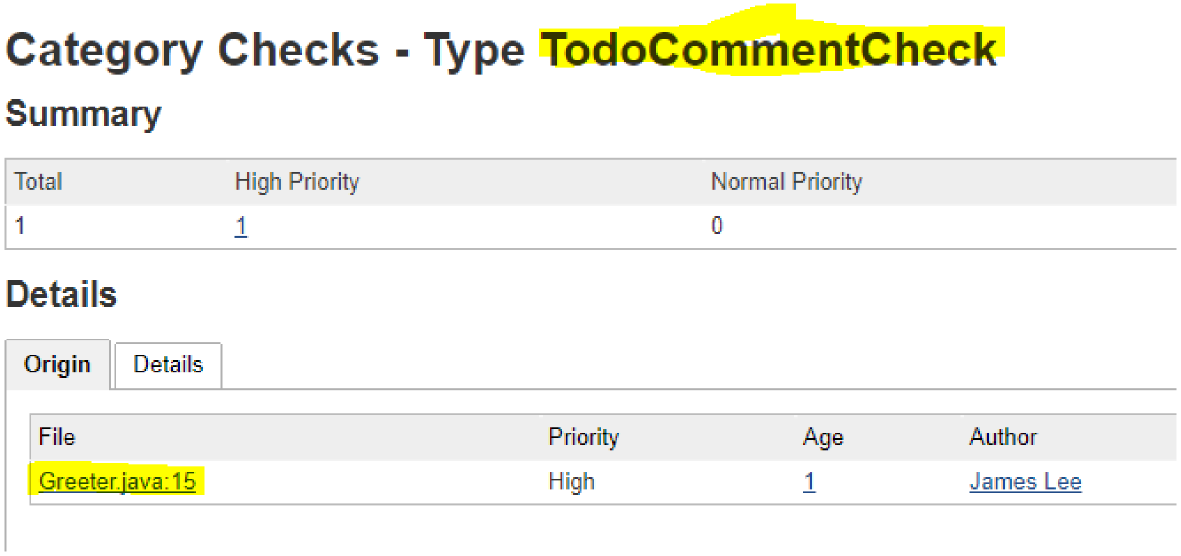
```
package com.example;

/**
 * This is a class.
 */
public class Greeter {

  /**
   * This is a constructor.
   */
  public Greeter() {

  }

  //TODO: Add javadoc comment
  public String greet(String someone) {
    return String.format("Hello, %s!", someone);
  }
}
```



```
  //TODO: Add javadoc comment
  public String greet(final String someone) {
    return String.format("Hello, %s!", someone);
  }
}
```

Only thing we have to add final keyword infront string parameter. So one warning fixed it seems.

In check category we have to go to next warning and click on that



Category Checks - Type TodoCommentCheck

Summary

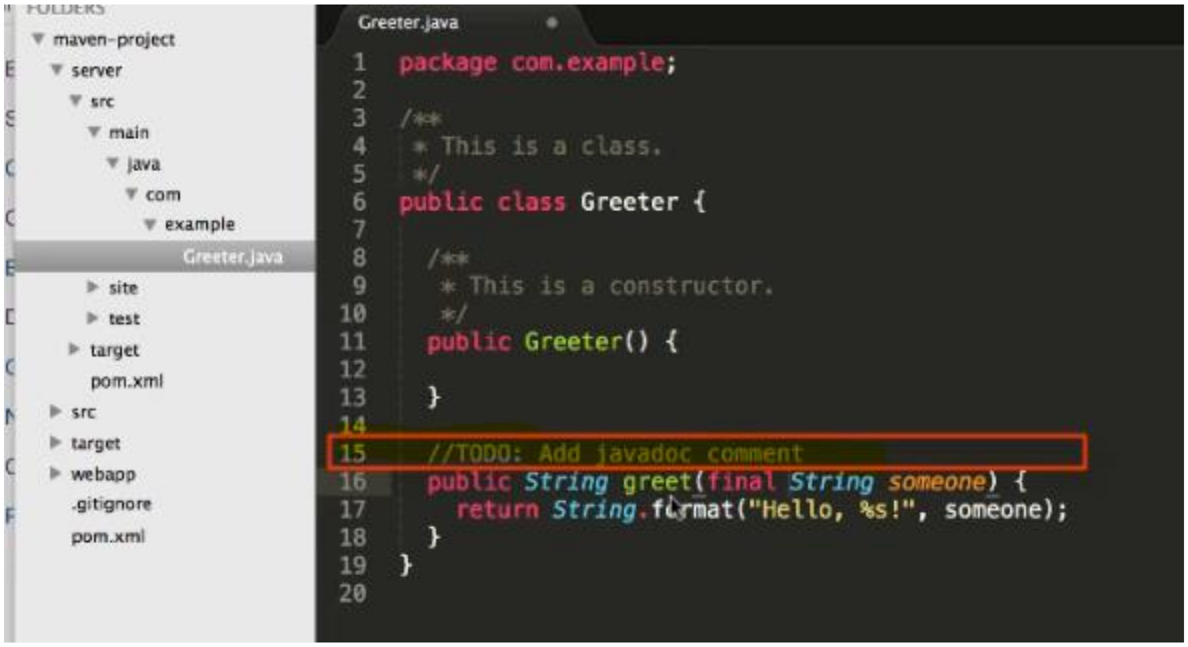
Total	High Priority	Normal Priority
1	1	0

Details

Origin | Details

File	Priority	Age	Author
Greeter.java:15	High	1	James Lee

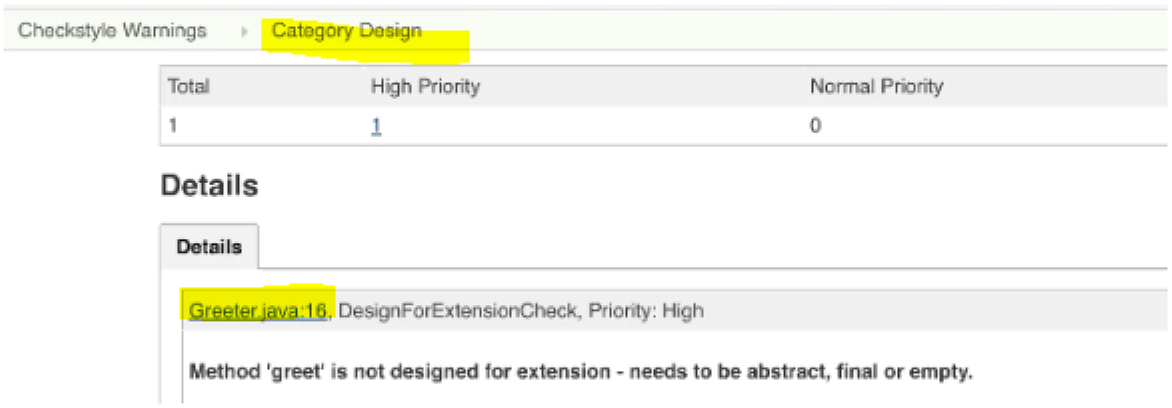
Click on that



```java
package com.example;

/**
 * This is a class.
 */
public class Greeter {

  /**
   * This is a constructor.
   */
  public Greeter() {

  }

  //TODO: Add javadoc comment
  public String greet(final String someone) {
    return String.format("Hello, %s!", someone);
  }
}
```

We have to remove that line

Third warning



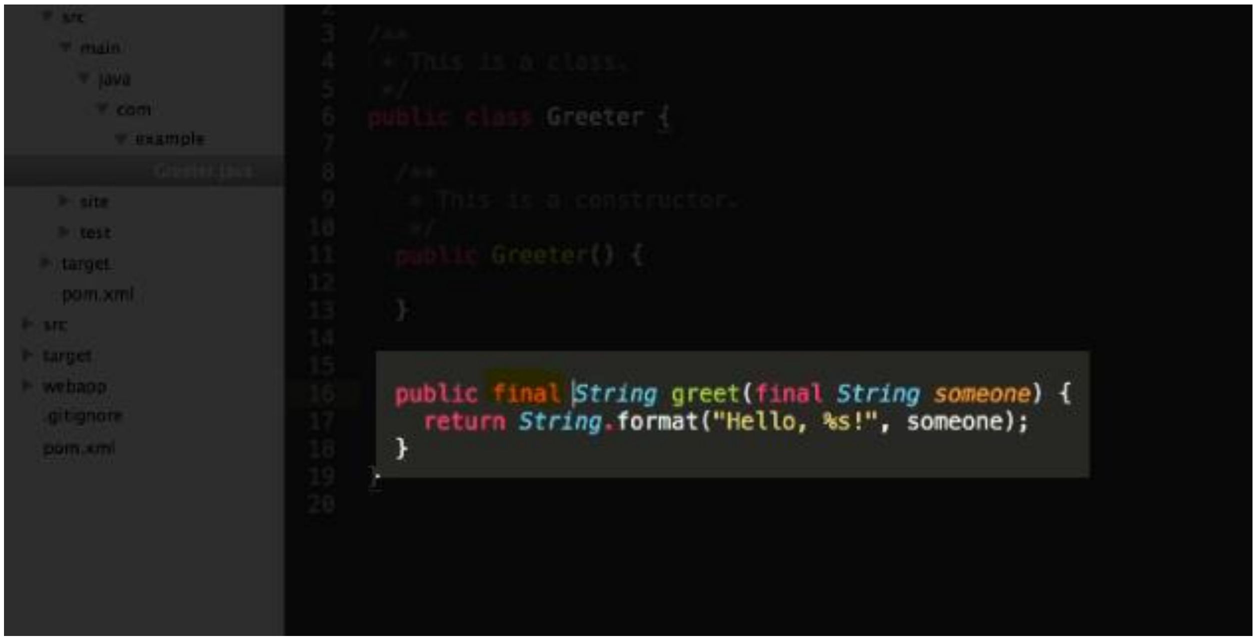
Total	High Priority	Normal Priority
1	1	0

Details

Details

Greeter.java:16, DesignForExtensionCheck, Priority: High

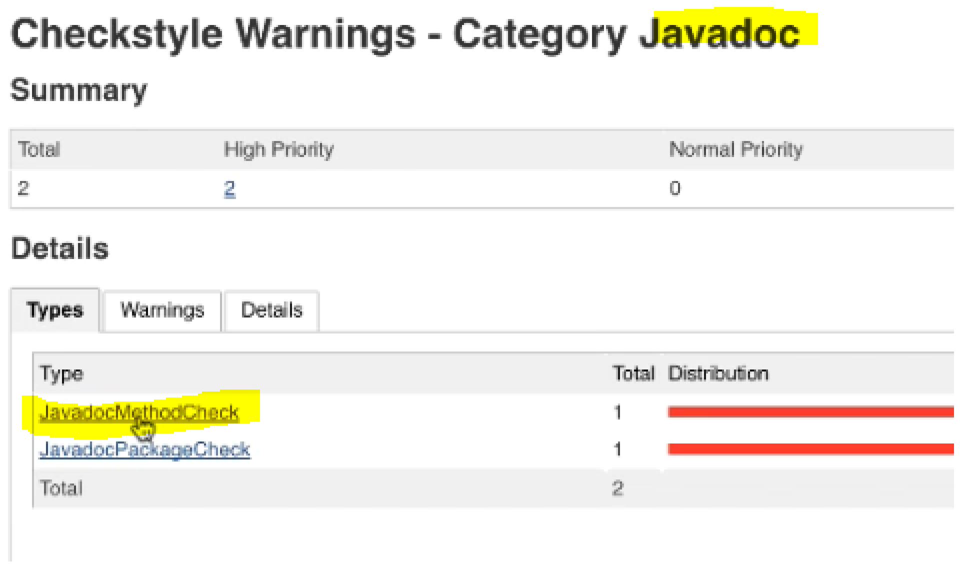
Method 'greet' is not designed for extension - needs to be abstract, final or empty.



```
/**
 * This is a class.
 */
public class Greeter {

  /**
   * This is a constructor.
   */
  public Greeter() {

  }

  public final String greet(final String someone) {
    return String.format("Hello, %s!", someone);
  }
}
```

We have to add final word again infront string.

Fourth warning



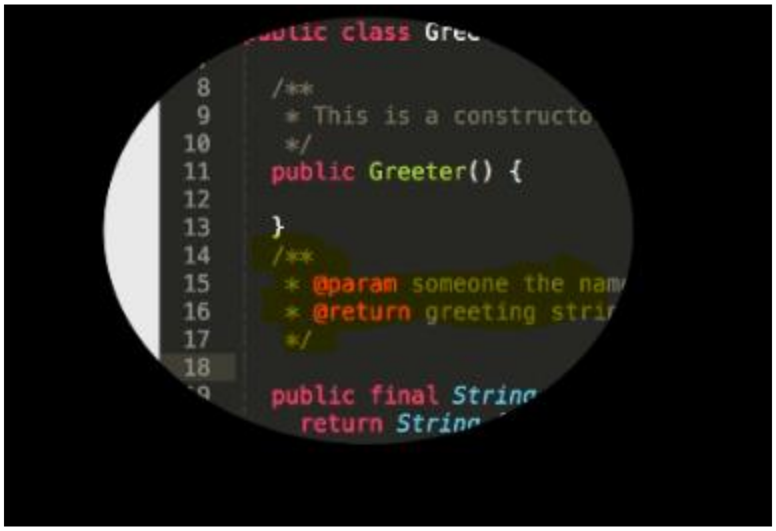
Checkstyle Warnings - Category Javadoc

Summary

Total	High Priority	Normal Priority
2	2	0

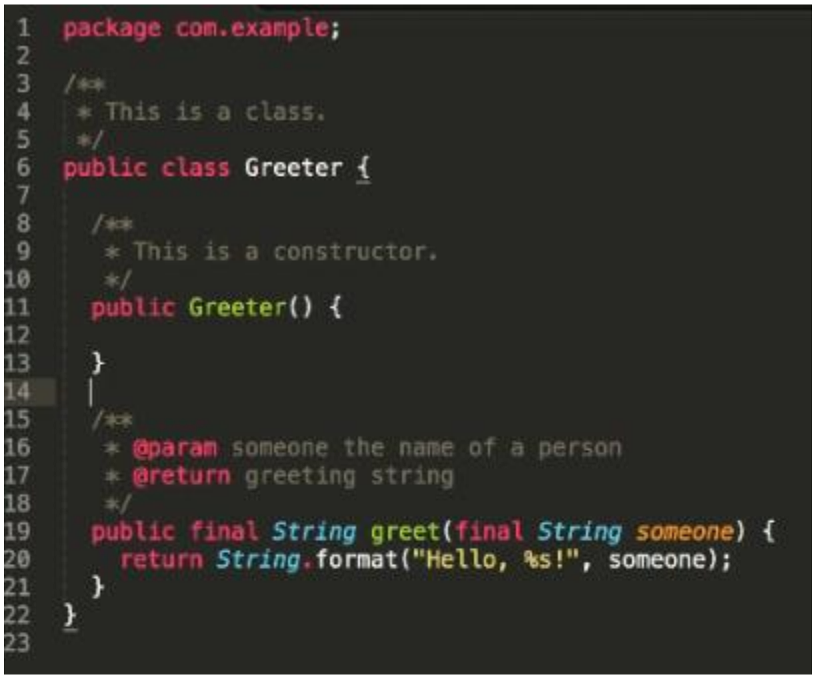
Details

Type	Total	Distribution
JavadocMethodCheck	1	🟥
JavadocPackageCheck	1	🟥
Total	2	



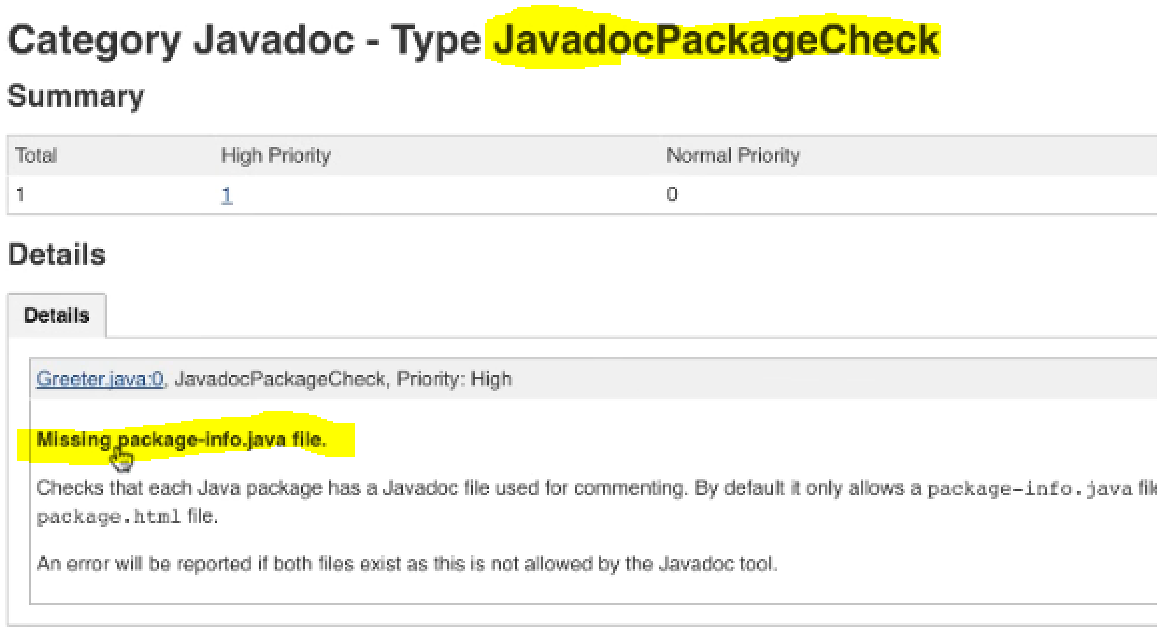
```
public class Gre
    /**
     * This is a constructo
     */
    public Greeter() {

    }
    /**
     * @param someone the nam
     * @return greeting strir
     */

    public final Strin
      return Strin
```

Comments has been missed so added between the methods



```
package com.example;

/**
 * This is a class.
 */
public class Greeter {

  /**
   * This is a constructor.
   */
  public Greeter() {

  }

  /**
   * @param someone the name of a person
   * @return greeting string
   */
  public final String greet(final String someone) {
    return String.format("Hello, %s!", someone);
  }
}
```

Fifth warning :



Category Javadoc - Type JavadocPackageCheck

Summary

Total	High Priority	Normal Priority
1	1	0

Details

Details

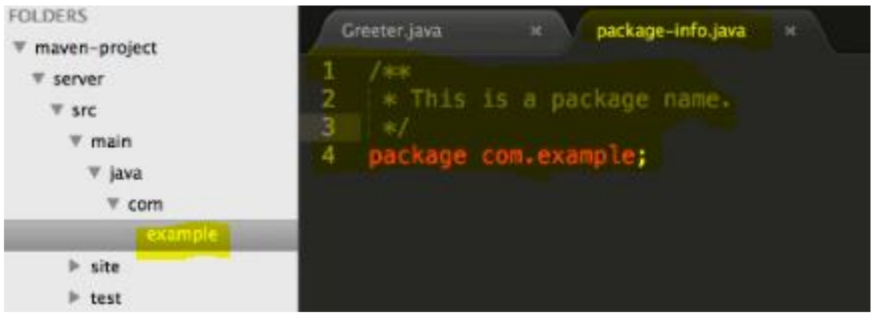
Greeter.java:0, JavadocPackageCheck, Priority: High

Missing package-info.java file.

Checks that each Java package has a Javadoc file used for commenting. By default it only allows a `package-info.java` fil
`package.html` file.

An error will be reported if both files exist as this is not allowed by the Javadoc tool.

File has been missed there



```java
/**
 * This is a package name.
 */
package com.example;
```

So under example folder I have created a new file called package-info.java with above content. Now all warnings has been fixed so we can re run the build now.

Git status

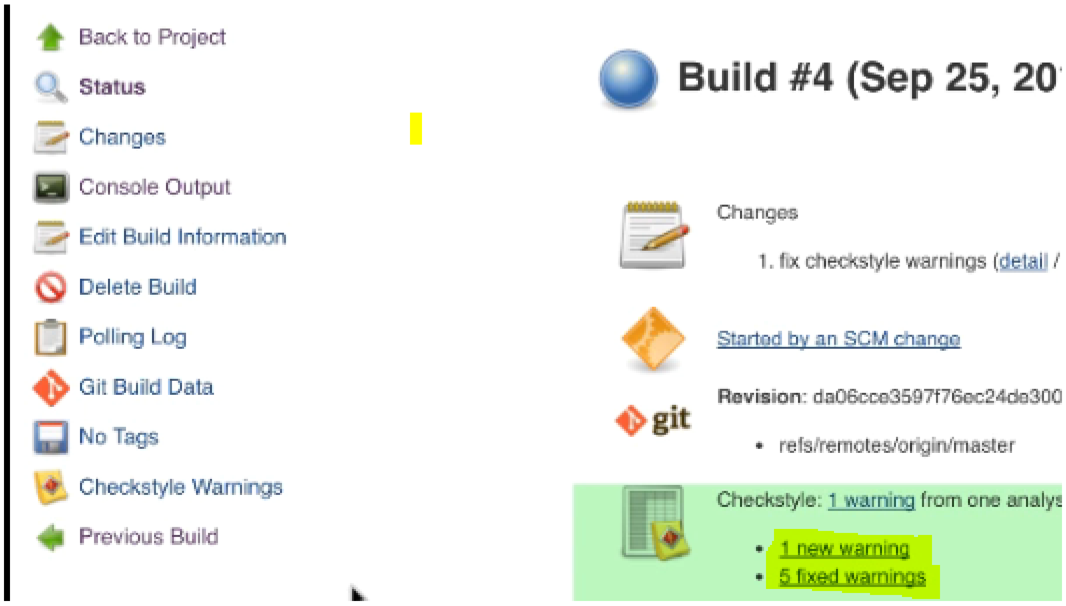
Git add .

Git commit –m “message”

Git push origin master

And clcik on build now or configure poll scm for every minit.

Now we can go to home page of new build number and check for the checkstyle report now.



Build #4 (Sep 25, 20

Changes

1. fix checkstyle warnings (detail /

Started by an SCM change

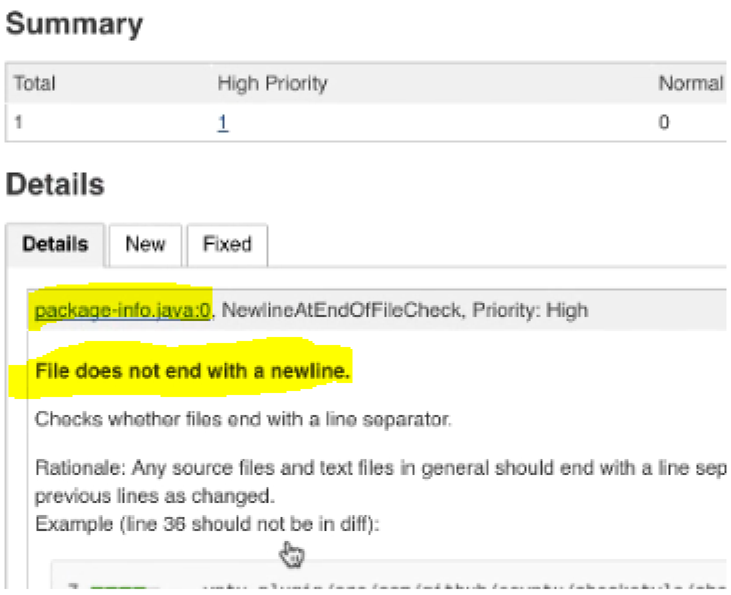
Revision: da06cce3597f76ec24de300

- refs/remotes/origin/master

Checkstyle: 1 warning from one analys

- 1 new warning
- 5 fixed warnings

Now 5 warnings are fixed and new warning came into picture



Summary

Total	High Priority	Normal
1	1	0

Details

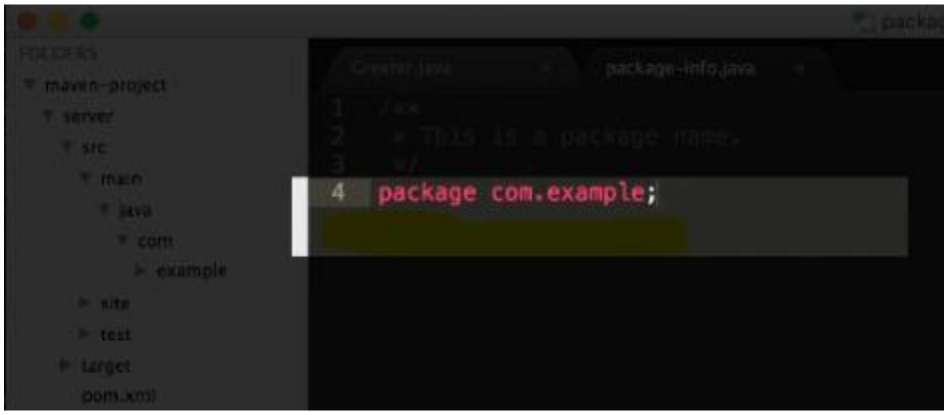
package-info.java:0, NewlineAtEndOfFileCheck, Priority: High

File does not end with a newline.

Checks whether files end with a line separator.

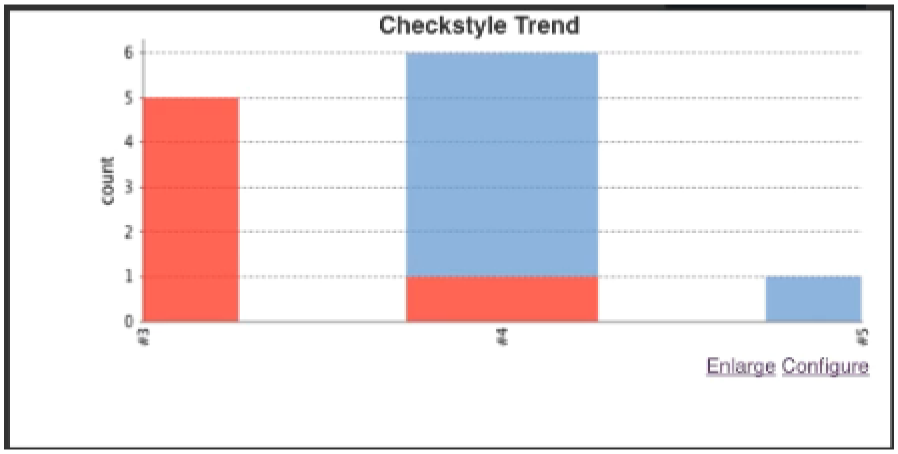
Rationale: Any source files and text files in general should end with a line sep
previous lines as changed.
Example (line 36 should not be in diff):

Am going to newly created file after last code line am putting one enter means adding new line after last line of code.



```
package com.example;
```

Now again git add commit push and build will get run and check the report again.



There are other tools are available like checkstyle. They are PMD and FindBugs and many.

If you are intrested just go through the below links

PMD Jenkins plugin:

https://wiki.jenkins-ci.org/display/JENKINS/PMD+Plugin

Findbugs Jenkins Plugin:

https://wiki.jenkins-ci.org/display/JENKINS/FindBugs+Plugin

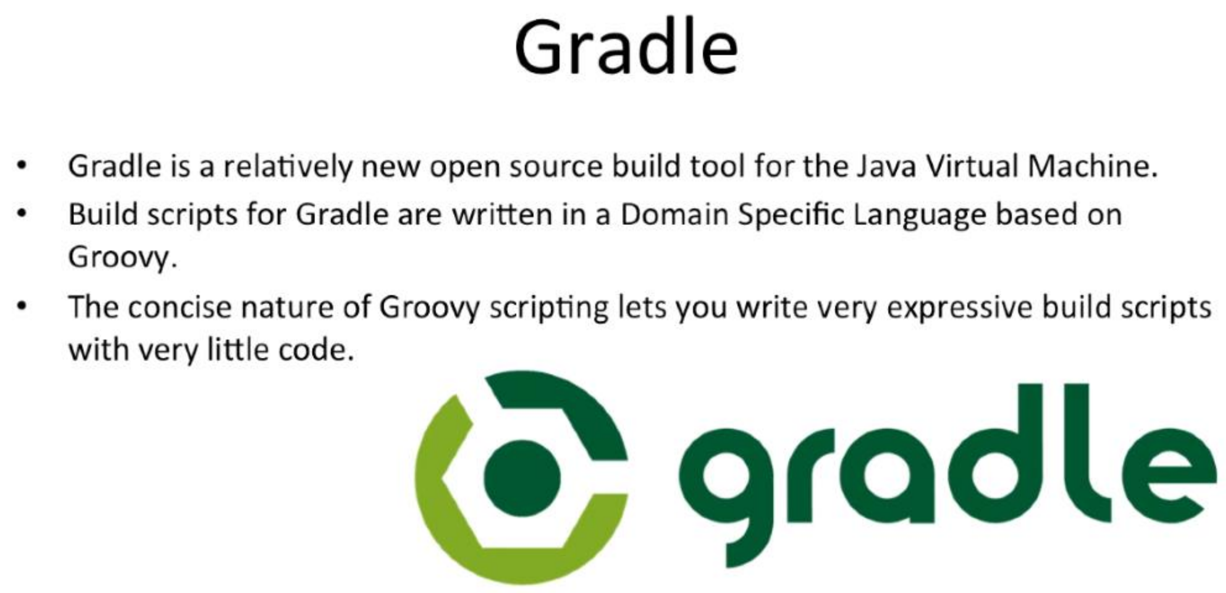
Jenkins' support for other build systems (Ant, Gradle and shell scripts)

Jenkins

Apache Ant

- Widely-used and very well-known build scripting language for Java.
- Flexible, extensible, relatively low-level scripting language.
- An Ant build script is made up of a number of targets, each target performs a particular job in the build process.

<APACHE ANT>



Gradle

- Gradle is a relatively new open source build tool for the Java Virtual Machine.
- Build scripts for Gradle are written in a Domain Specific Language based on Groovy.
- The concise nature of Groovy scripting lets you write very expressive build scripts with very little code.

Apache Ant:

http://ant.apache.org/

Ant Targets:

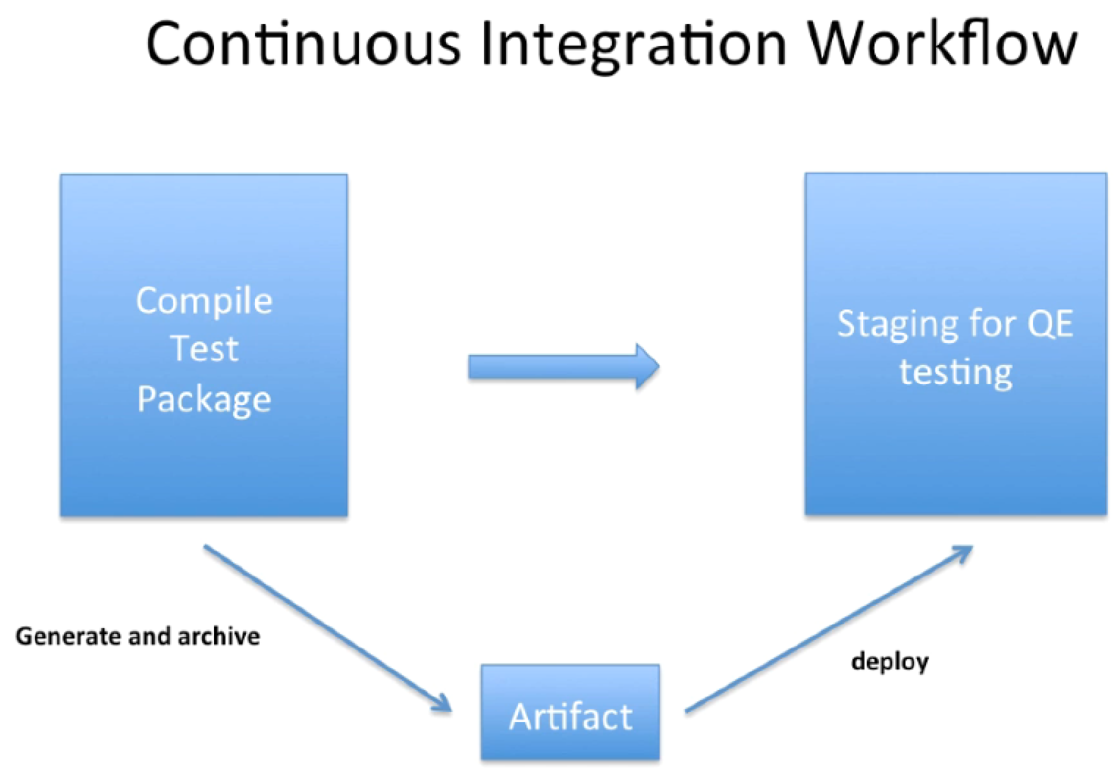
http://ant.apache.org/manual/targets.html

Gradle:

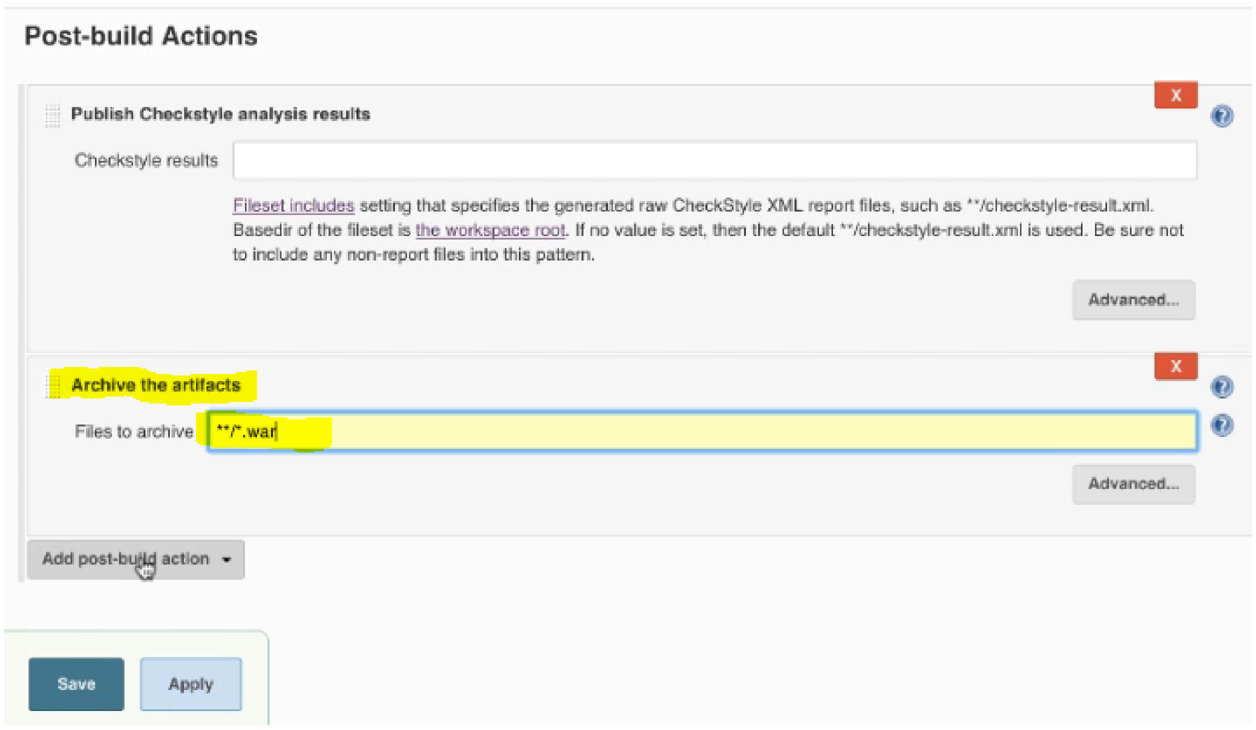
https://docs.gradle.org/current/dsl/index.html

Archive generated artifacts

So far we have just seen about compile test and package preparation. Now are going to do archive the artifact and deploying into container.



To archive the artifact we have to add one more step into post build action step like below



Post-build Actions

Publish Checkstyle analysis results

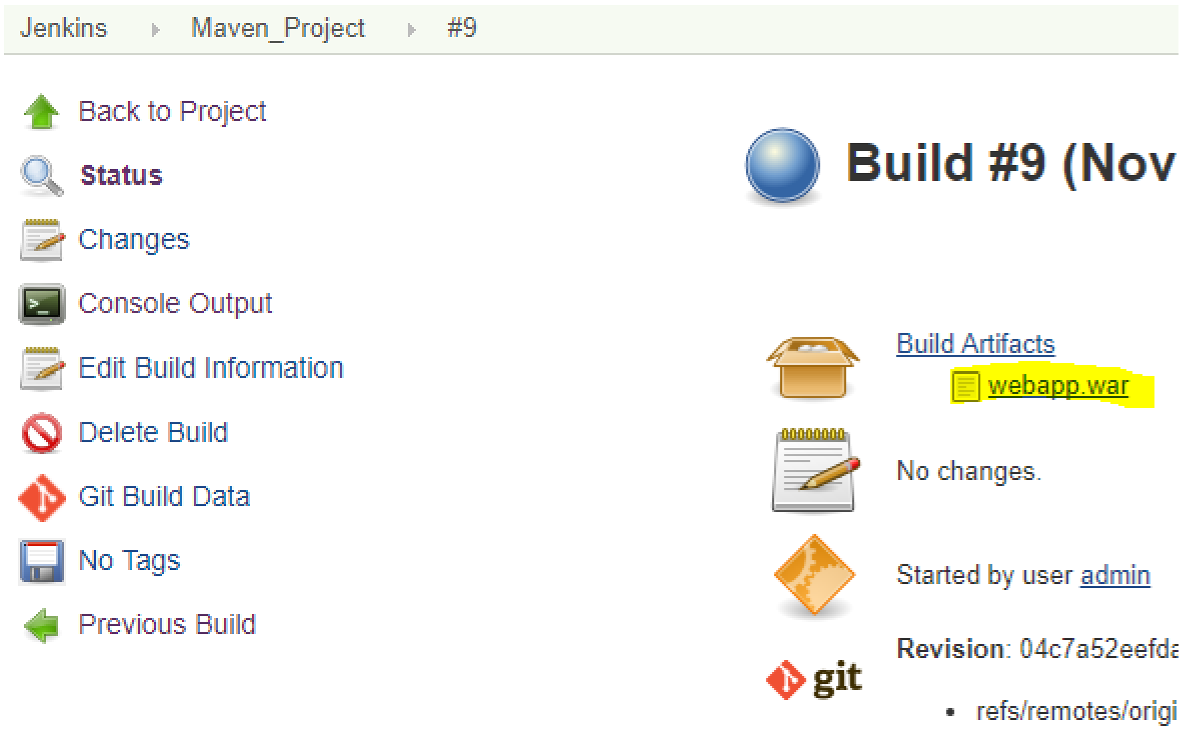
Checkstyle results

Fileset includes setting that specifies the generated raw CheckStyle XML report files, such as **/checkstyle-result.xml. Basedir of the fileset is the workspace root. If no value is set, then the default **/checkstyle-result.xml is used. Be sure not to include any non-report files into this pattern.

Archive the artifacts

Files to archive: **/*.war

Here Jenkins is going to take all .war files to do archive. After running the build we can see archived artifact on build number home page.



Build #9 (Nov

Build Artifacts
webapp.war

No changes.

Started by user admin

Revision: 04c7a52eefda

- refs/remotes/origi

And you can go and check console output as well what process has been done.

Install and configure Tomcat as a staging environment

Jenkins

Tomcat

Tomcat is an open-source web server and provides a "pure Java" HTTP web server environment in which Java code can run.

Apache Tomcat

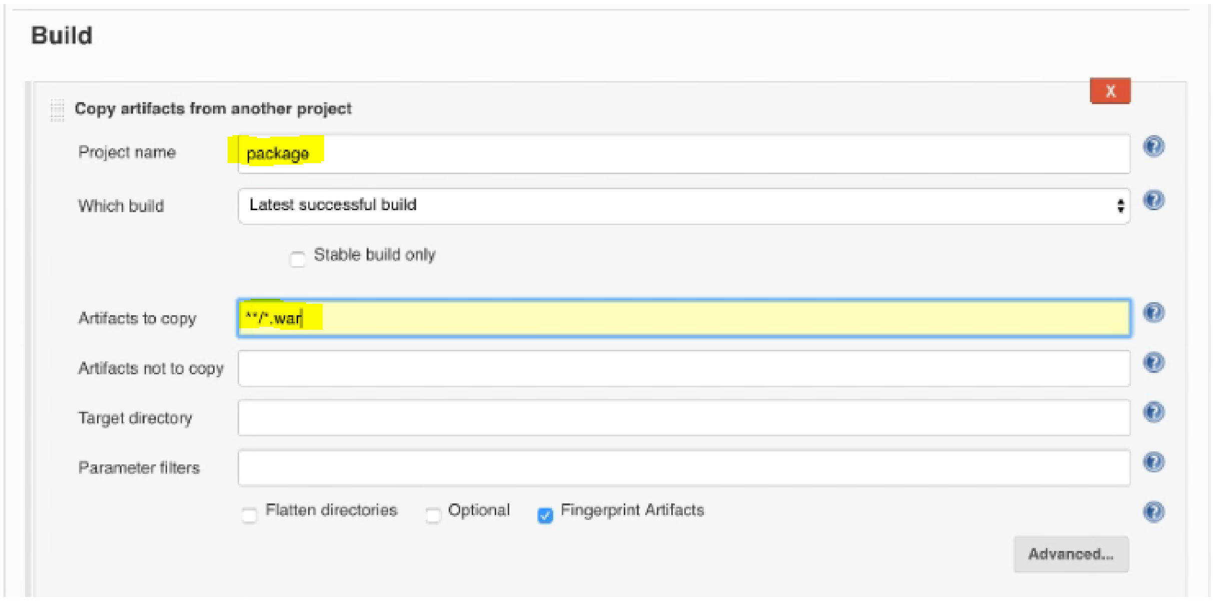
We have to install tomcat server and then we are able to deploy to this container.

Change Tomcat Server Port

- Jenkins runs on port 8080.
- The default port of Tomcat is also 8080.

- Install *copy artifact* and *deploy to container* plugins
- Deploy our application to staging environment

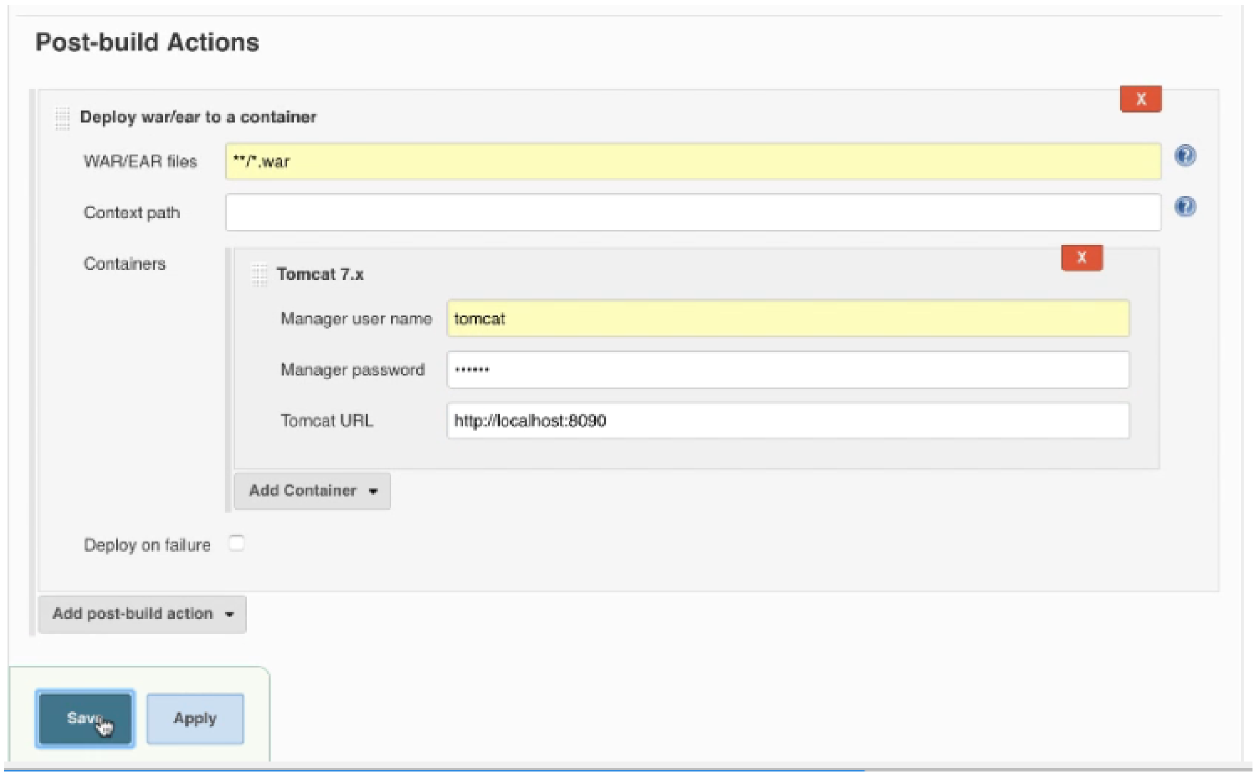
Here we have to install 2 plugins one is for copying artifact from previous build and another plugin is for deploying to container. And below configuration we have to do in build step and post buld step.



Build

Copy artifacts from another project

Project name	package
Which build	Latest successful build
	Stable build only
Artifacts to copy	**/*.war
Artifacts not to copy	
Target directory	
Parameter filters	
	Flatten directories Optional [✓ Fingerprint Artifacts]


Post-build Actions

Deploy war/ear to a container

WAR/EAR files: **/*.war

Context path:

Containers

Tomcat 7.x

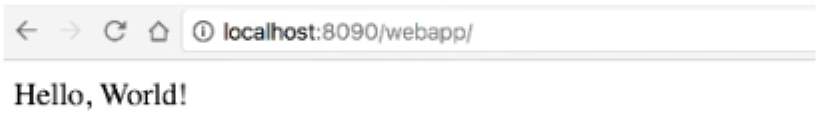
Manager user name: tomcat

Manager password: ••••••

Tomcat URL: http://localhost:8090

Deploy on failure: ☐

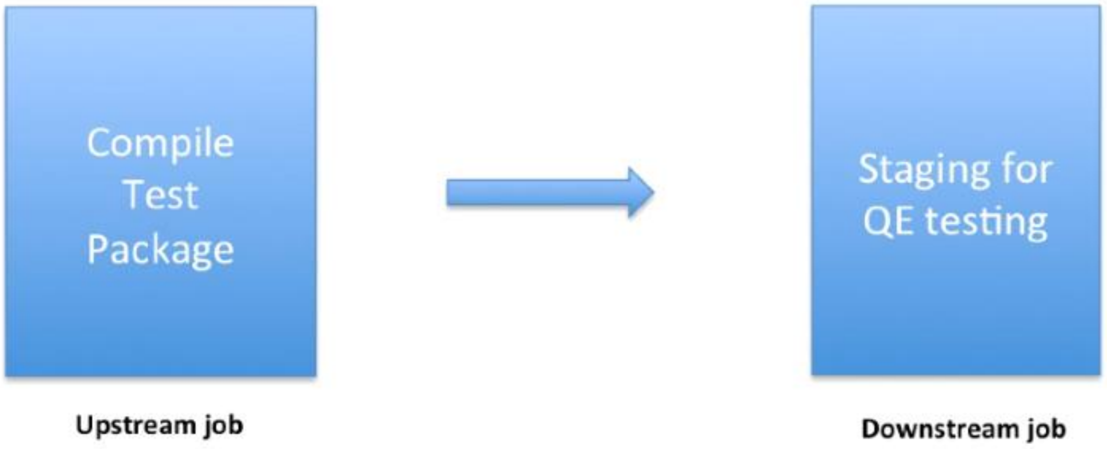
So here I have to create 2 builds and one is for preparing package and the second one is for taking that artifact and it is able to deploy into staging area. And here we have to make these 2 builds like upstream and down stream projects in post build action tab.


localhost:8090/webapp/

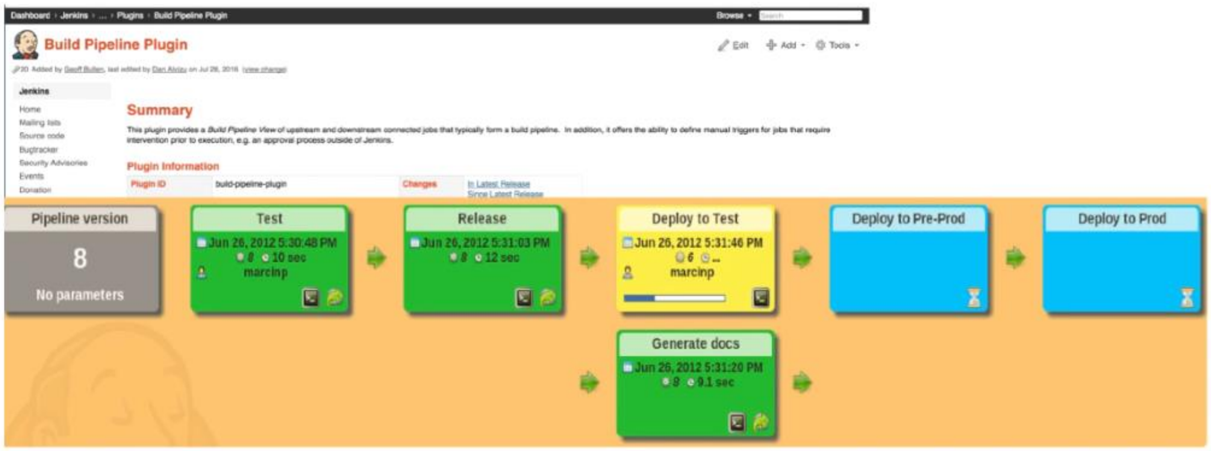
Hello, World!

When you run the frst build it is going have this output.

Our Current Build Pipeline



Build Pipeline Plugin



Build Pipeline Plugin

20 Added by Geoff Bullen, last edited by Dan Alvizu on Jul 28, 2016 (view change)

Summary

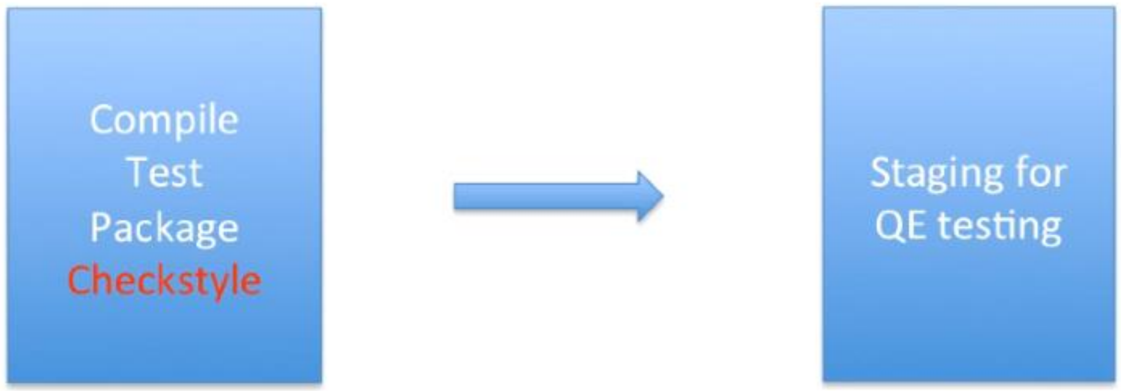
This plugin provides a *Build Pipeline View* of upstream and downstream connected jobs that typically form a build pipeline. In addition, it offers the ability to define manual triggers for jobs that require intervention prior to execution, e.g. an approval process outside of Jenkins.

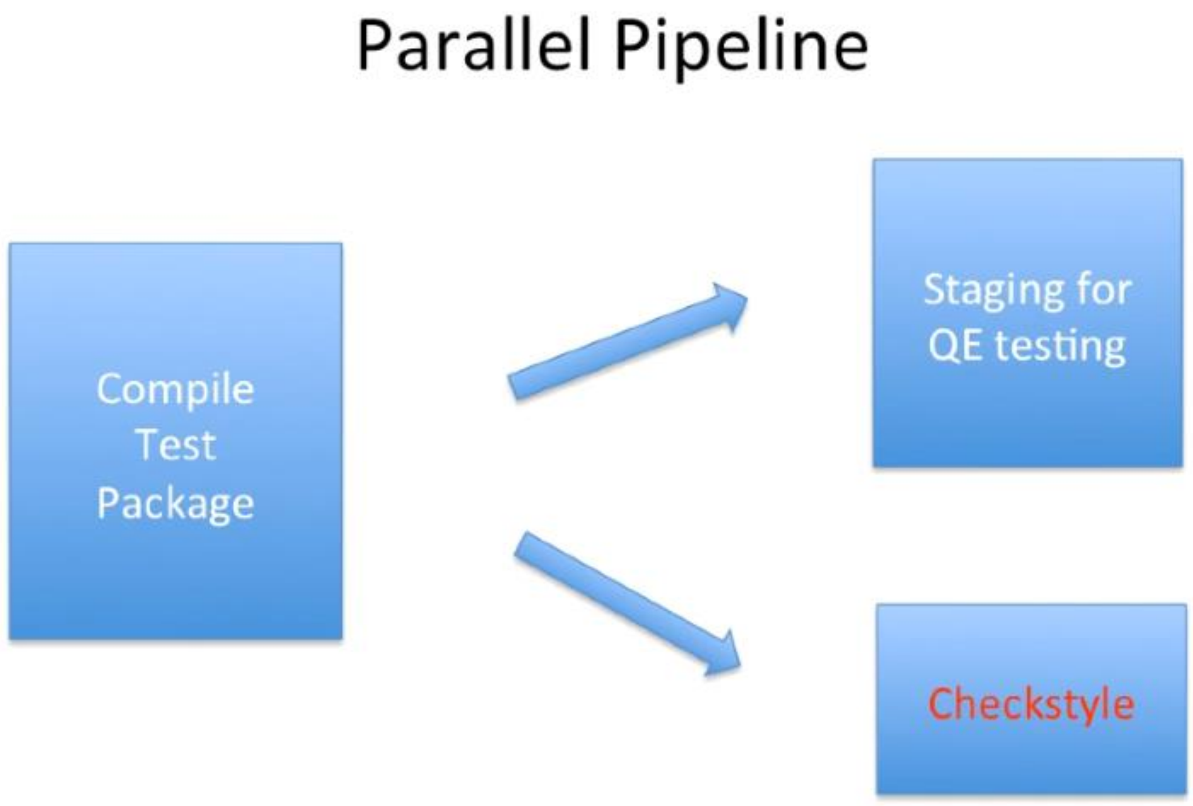
Plugin Information

Plugin ID	build-pipeline-plugin	Changes	In Latest Release Since Latest Release

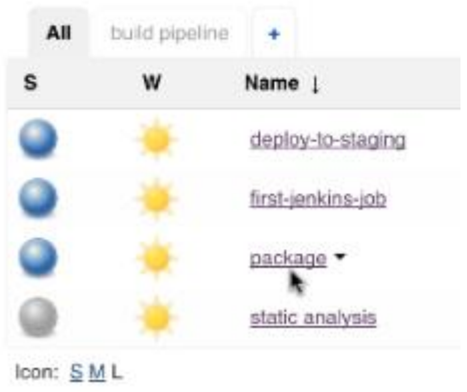
Parallel Jenkins Build

Linear Pipeline





We are going to create 3 builds

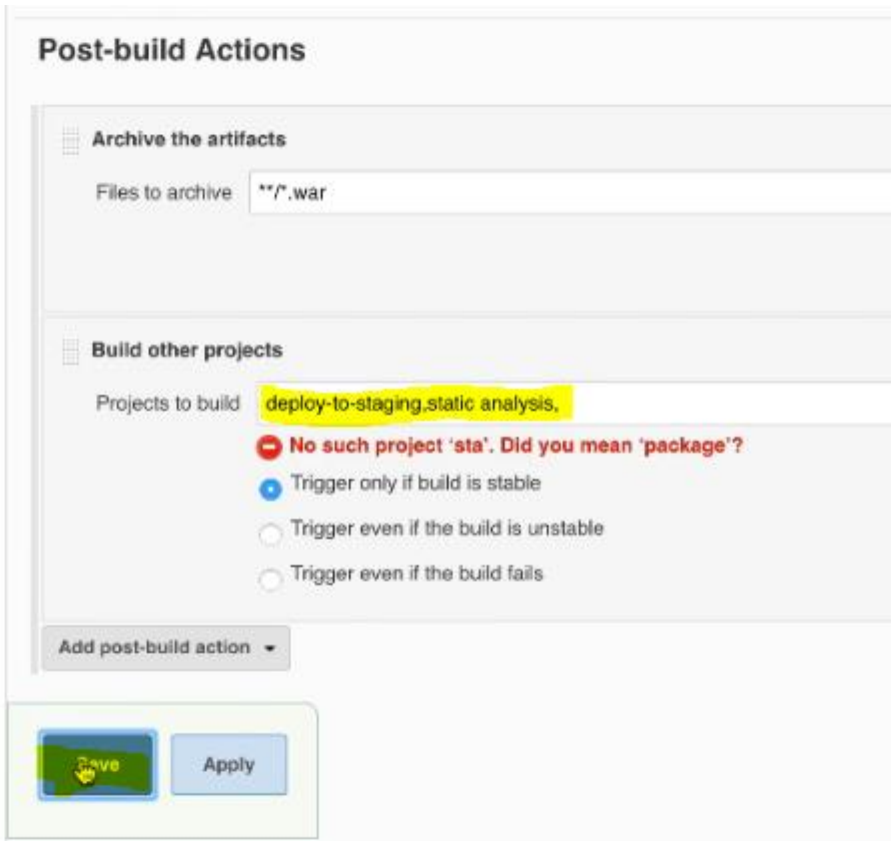


S	W	Name ↓
●	☀	deploy-to-staging
●	☀	first-jenkins-job
●	☀	package
○	☀	static analysis

Package will have goals to prepare war so in build actions tab we have to provide goal called clean and package no checkstyle goal.

Now second buld is static analysis here we have to write checkstyle:checkstyle goal in build action tab.

So in Package buld we are going to 2 bulds at a time like below screen shot



Post-build Actions

Archive the artifacts

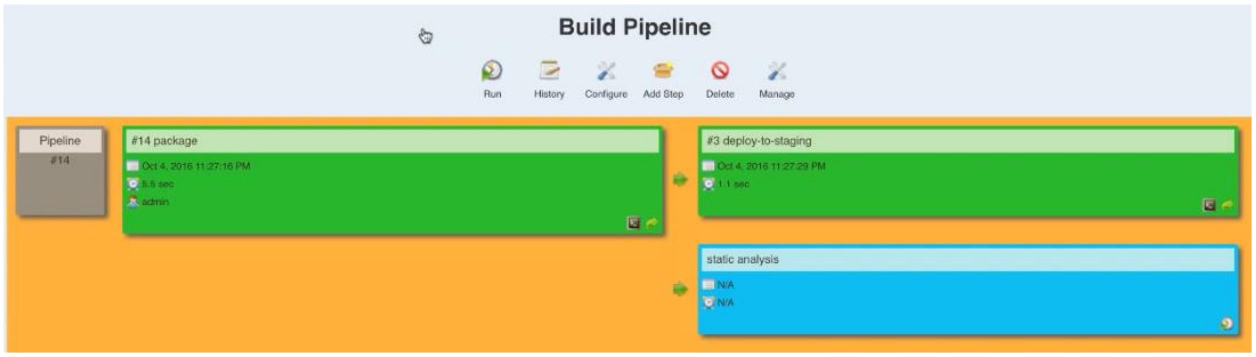
Files to archive: `**/*.war`

Build other projects

Projects to build: deploy-to-staging,static analysis,

No such project 'sta'. Did you mean 'package'?

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



Build Pipeline

Pipeline #14

#14 package
Oct 4, 2016 11:27:16 PM
5.5 sec
admin

#3 deploy-to-staging
Oct 4, 2016 11:27:29 PM
1.1 sec

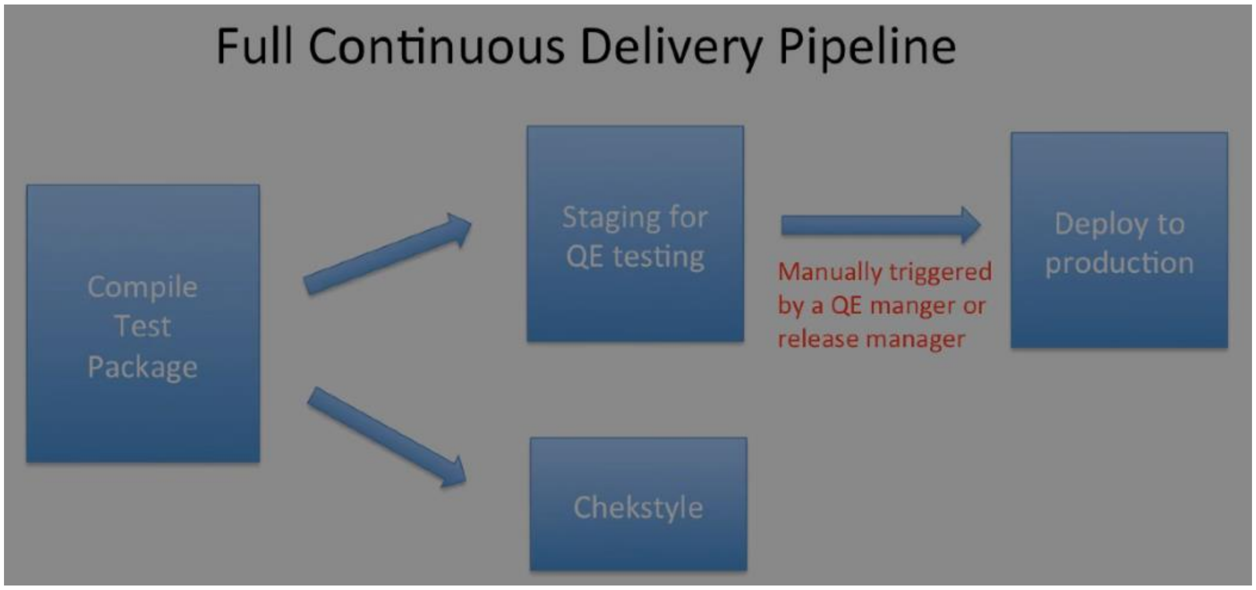
static analysis
N/A
N/A

So next one is we are going to setup deploy to production.

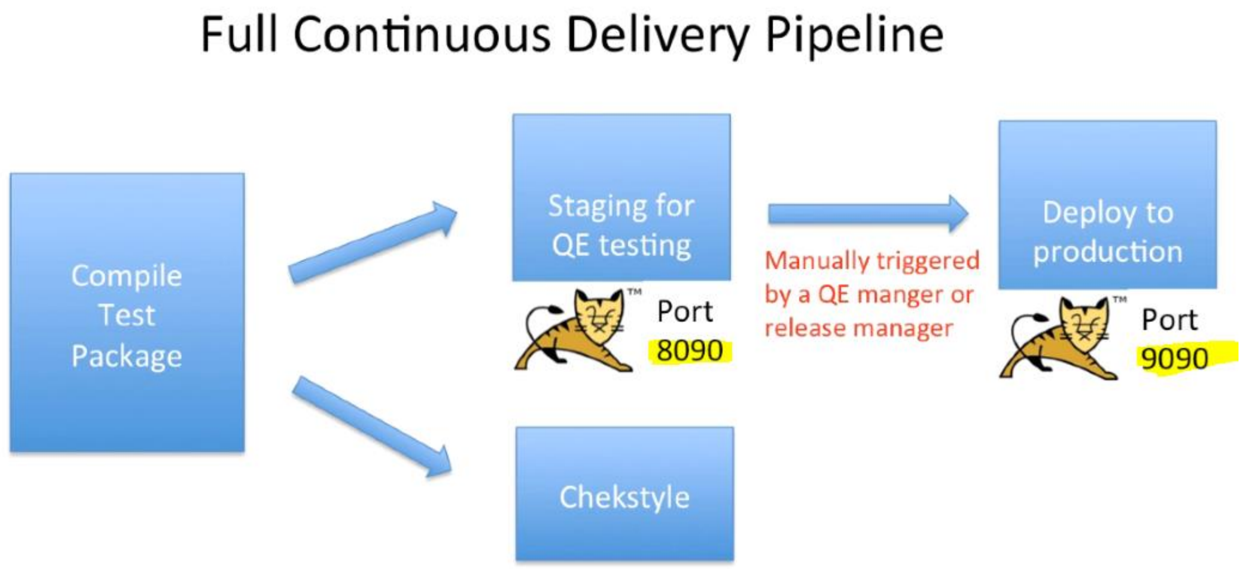
Continuous Delivery

Deploy our app to production

Jenkins



Full Continuous Delivery Pipeline

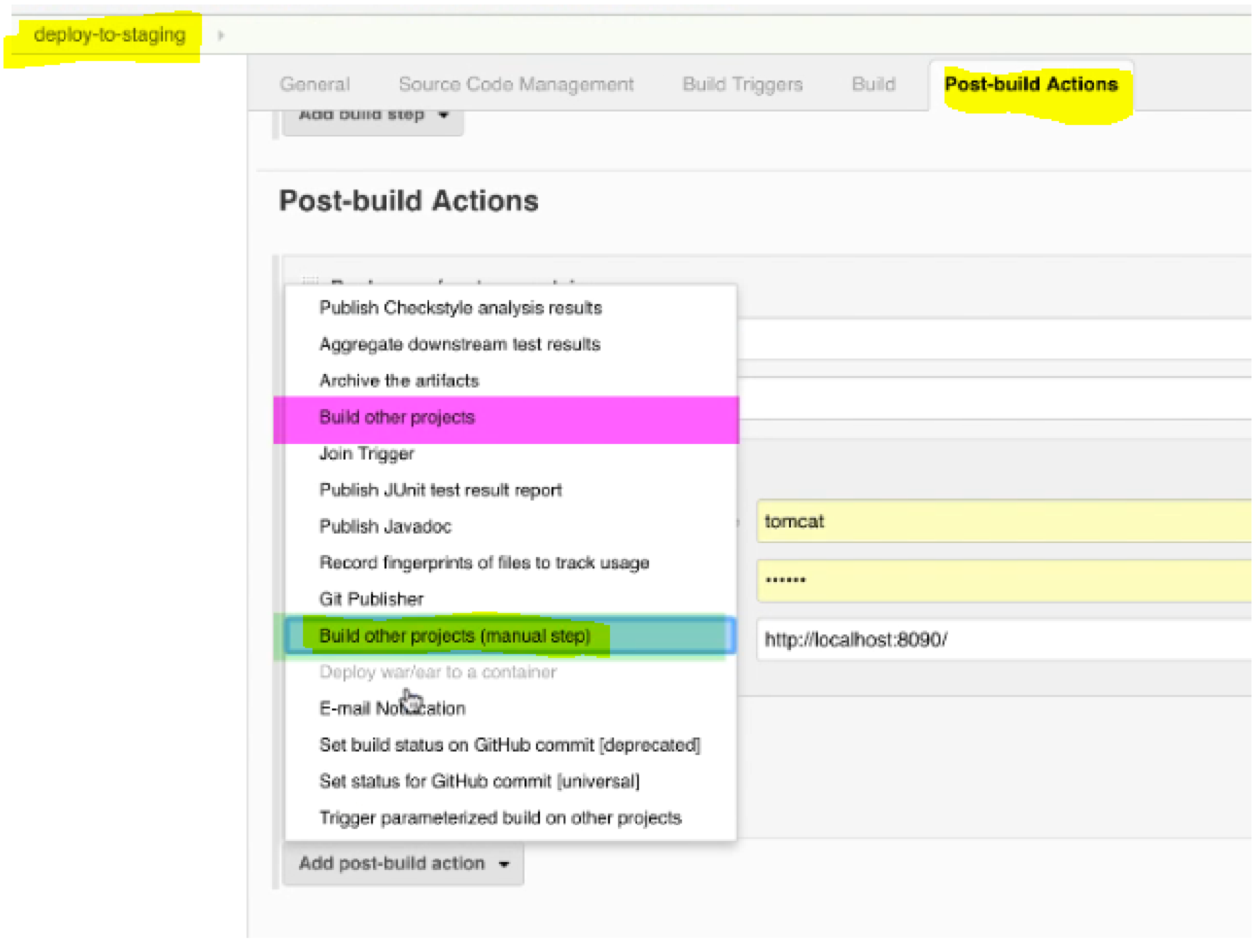


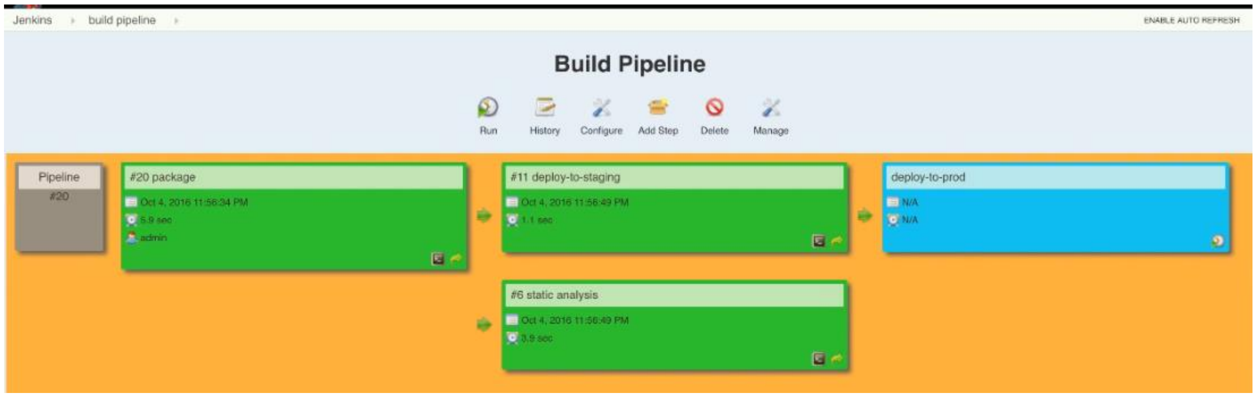
we are going to have 2 tomcats with 2 diff port numbers in 2 diff folders and in a same system.

So we have to start both the servelet containers then we can create above 4 builds and we can set up pipeline.

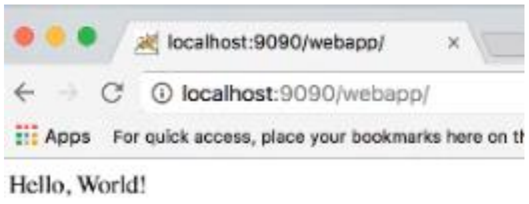
We have to create new job for prod env same like staging we have done like in build step we have to selct copy artifact from package build and in same way we have to select deploy to container in post build actions tab.

And then this prod job calling we have to set up in staging deployment build in post build actions, since we have to do manually we have to select option called deploy other builds manually in post buld actions of staging area build setup.


deploy-to-staging
General
Source Code Management
Build Triggers
Build
Post-build Actions
Add build step
Post-build Actions
Publish Checkstyle analysis results
Aggregate downstream test results
Archive the artifacts
Build other projects
Join Trigger
Publish JUnit test result report
Publish Javadoc
Record fingerprints of files to track usage
Git Publisher
Build other projects (manual step)
Deploy war/ear to a container
E-mail Notification
Set build status on GitHub commit [deprecated]
Set status for GitHub commit [universal]
Trigger parameterized build on other projects
Add post-build action
tomcat
••••••
http://localhost:8090/



Last build we have to run manually only.



Hello, World!

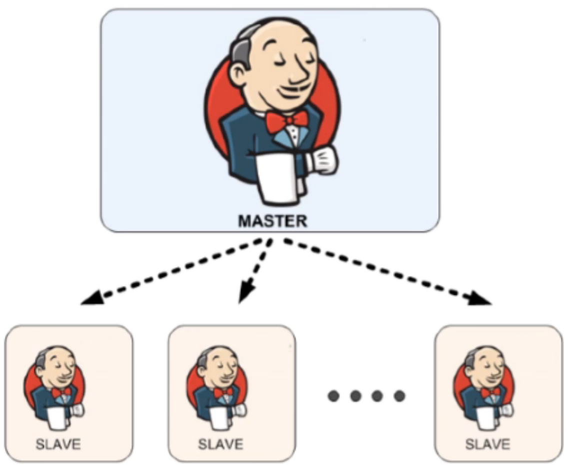
Install Jenkins on ubuntu system :

Install Jenkins on the Master Node

```
wget -q -O - http://pkg.jenkins-ci.org/debian/jenkins-ci.org.key | apt-key
add -

echo deb http://pkg.jenkins-ci.org/debian binary/ >
/etc/apt/sources.list.d/jenkins.list

apt-get update

apt-get install Jenkins
```

Introduction to Distributed Jenkins Builds

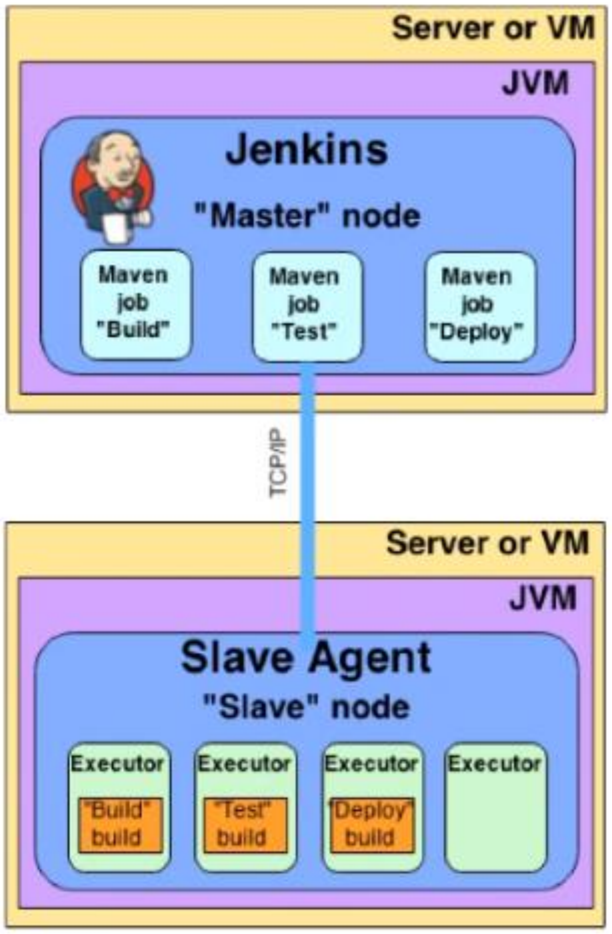
Jenkins' Master and Slave Architecture



Master:
- Schedule build jobs.
- Dispatch builds to the slaves for the actual job execution.
- Monitor the slaves and record the build results.
- Can also execute build jobs directly.

Slave:
- Execute build jobs dispatched by the master.

Jenkins Slave Agent



Different ways to start slave agent

- The master can start the slave agents via SSH.
- Start the slave agent manually using Java Web Start.
- Install the slave agent as a Window service.
- Start the slave agent directly from the command line on the slave machine.