



# Jenkins

## Contents:

1. Introduction.....	3
1.1 What is the Jenkins .....	3
1.2 Launching the Test Drive.....	3
2. Scenario: .....	3
3. Conclusion .....	24

# 1. Introduction

## 1.1 What is the Jenkins

Jenkins is an open-source continuous integration software tool written in the Java programming language for testing and reporting on isolated changes in a larger code base in real time. The software enables developers to find and solve defects in a code base rapidly and to automate testing of their builds.

### **Continuous Integration:**

Continuous Integration is a development practice in which the developers are required to commit changes to the source code in a shared repository several times a day or more frequently. Every commit made in the repository is then built. This allows the teams to detect the problems early. Apart from this, depending on the Continuous Integration tool, there are several other functions like deploying the build application on the test server, providing the concerned teams with the build and test results etc.

## 1.2 Launching the Test Drive

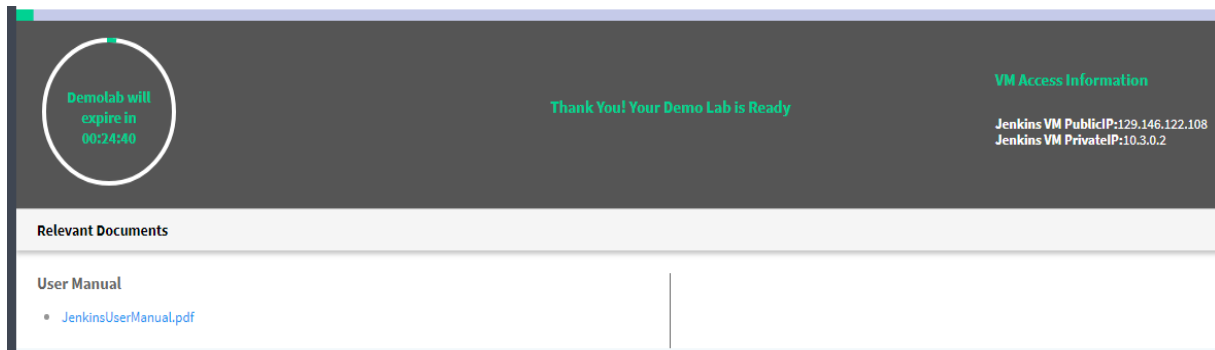
Once the test drive is ready, you should receive the credentials and URL(s) needed to access the test drive in an email sent to the address you used to sign up.

# 2. Scenario:

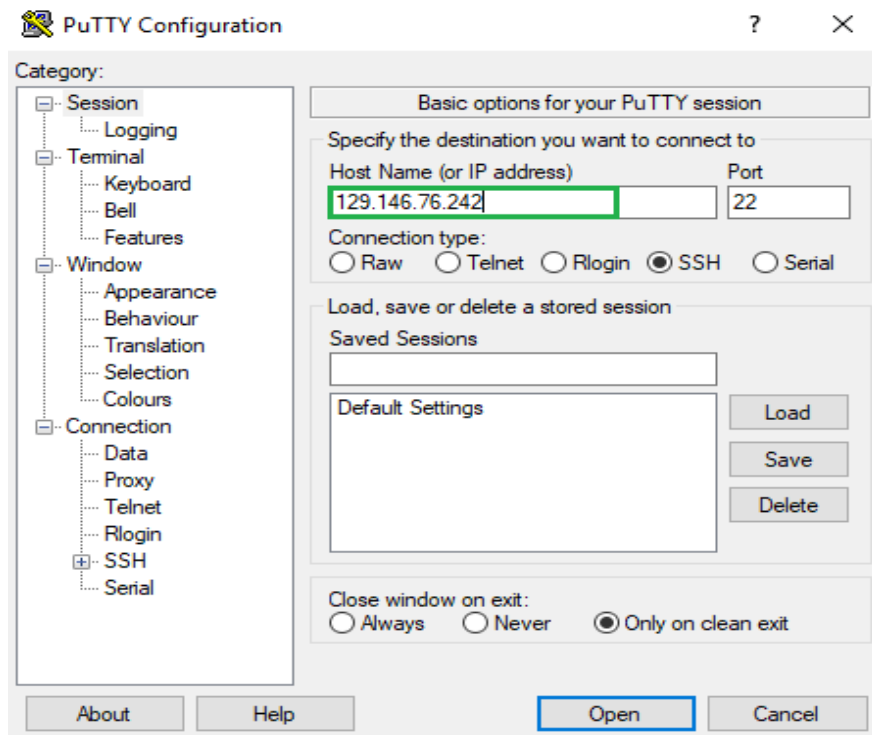
### **Objectives:**

- Launch the test drive.
- Setup and configure the GitHub repository.
- Configure Jenkins job.
- See Jenkins in action.

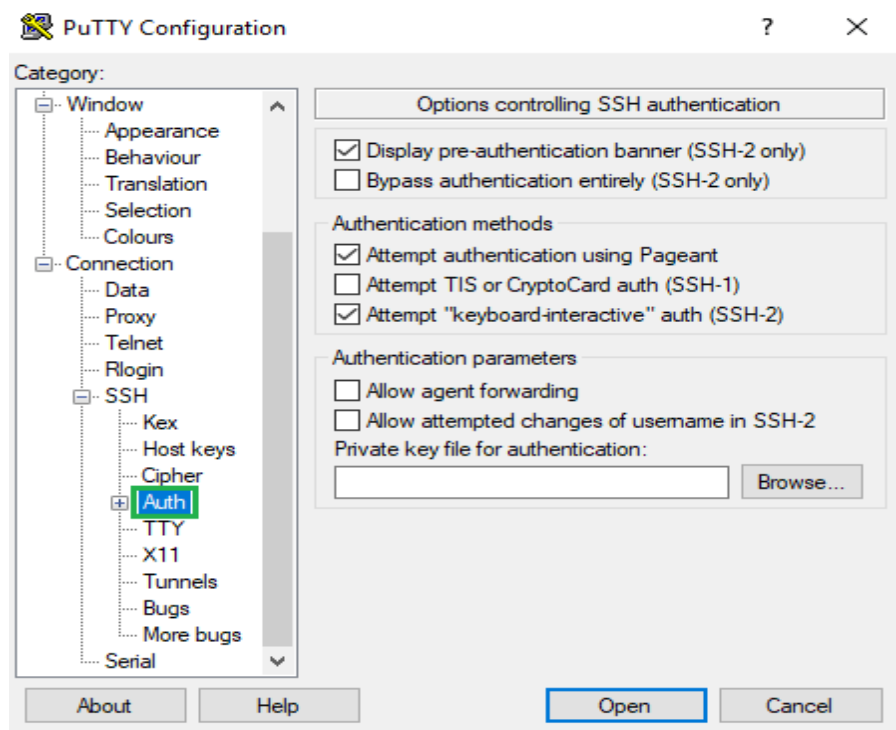
1. Log into your SSH session using client SSH URL provided in test drive launch page.

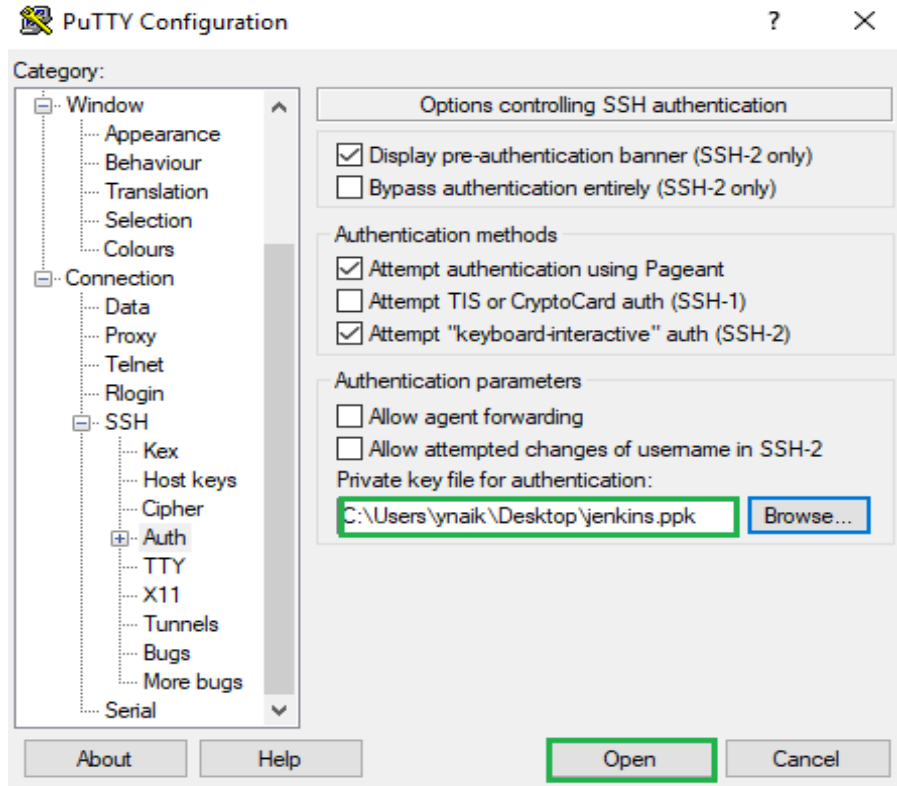


2. Open putty and SSH into the VM using the IP address



Click on browse and select the private key you have stored in your local system





```

ubuntu@jenkinsvm: ~
login as: ubuntu
Authenticating with public key "rsa-key-20171009"
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.4.0-97-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Get cloud support with Ubuntu Advantage Cloud Guest:
http://www.ubuntu.com/business/services/cloud

35 packages can be updated.
14 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@jenkinsvm:~$ █

```

3. Once you login into the VM you need to run a script to configure the job for the Test Drive

Run the below command:

***sudo wget <https://raw.githubusercontent.com/yougandar/test/master/script.sh>***

This command basically downloads the script from the given location

```
ubuntu@jenkinsvm: ~  
ubuntu@jenkinsvm:~$ sudo wget https://raw.githubusercontent.com/yougandar/test/master/script.sh  
--2017-11-02 10:40:48-- https://raw.githubusercontent.com/yougandar/test/master/script.sh  
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.200.133  
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.200.133|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 1607 (1.6K) [text/plain]  
Saving to: 'script.sh'  
  
script.sh          100%[=====>] 1.57K  --.-KB/s  in 0s  
  
2017-11-02 10:40:48 (446 MB/s) - 'script.sh' saved [1607/1607]  
  
ubuntu@jenkinsvm:~$
```

To run the script run the following command

***sudo sh script.sh***

```
ubuntu@jenkinsvm:~$ sudo sh script.sh  
Ign:1 http://pkg.jenkins-ci.org/debian-stable binary/ InRelease  
Hit:2 http://pkg.jenkins-ci.org/debian-stable binary/ Release  
Hit:4 http://security.ubuntu.com/ubuntu xenial-security InRelease  
Hit:5 http://iad-ad-2.clouds.archive.ubuntu.com/ubuntu xenial InRelease  
Hit:6 http://iad-ad-2.clouds.archive.ubuntu.com/ubuntu xenial-updates InRelease  
Hit:7 http://iad-ad-2.clouds.archive.ubuntu.com/ubuntu xenial-backports InRelease  
Reading package lists... Done  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following packages were automatically installed and are no longer required:  
  grub-pc-bin linux-headers-virtual linux-image-virtual  
Use 'sudo apt autoremove' to remove them.  
The following NEW packages will be installed:  
  html-xml-utils  
0 upgraded, 1 newly installed, 0 to remove and 31 not upgraded.  
Need to get 231 kB of archives.  
After this operation, 2,691 kB of additional disk space will be used.  
Get:1 http://iad-ad-2.clouds.archive.ubuntu.com/ubuntu xenial/universe amd64 html-xml-utils amd64 6.9-1 [231 kB]  
Fetched 231 kB in 0s (3,175 kB/s)  
Selecting previously unselected package html-xml-utils.  
(Reading database ... 63619 files and directories currently installed.)  
Preparing to unpack .../html-xml-utils_6.9-1_amd64.deb ...  
Unpacking html-xml-utils (6.9-1) ...  
Processing triggers for man-db (2.7.5-1) ...  
Setting up html-xml-utils (6.9-1) ...  
[redacted]
```

```

Saving to: '/usr/share/jenkins/job-configfile.xml'

job-configfile.xml          100%[=====>]   1.66K  --.-KB/s   in 0s

2017-11-02 10:41:39 (502 MB/s) - '/usr/share/jenkins/job-configfile.xml' saved [1698/1698]

---Configuring Jenkins---
--2017-11-02 10:41:59-- http://localhost:8080/jnlpJars/jenkins-cli.jar
Resolving localhost (localhost)... 127.0.0.1
Connecting to localhost (localhost)|127.0.0.1|:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2972859 (2.8M) [application/java-archive]
Saving to: '/usr/share/jenkins/jenkins-cli.jar'

jenkins-cli.jar             100%[=====>]   2.83M  --.-KB/s   in 0.02s

2017-11-02 10:41:59 (125 MB/s) - '/usr/share/jenkins/jenkins-cli.jar' saved [2972859/2972859]

Authenticated as: admin
Authorities:
  authenticated
    % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
100   46  100   46    0     0    972      0  --:--:-- --:--:-- --:--:--   978
00e37fef3980f722957057ec0f639180
Jenkins-Crumb:261ca6fb27dc951ec17900441a27a7f7
ubuntu@jenkinsvm:~$ ls
script.sh
ubuntu@jenkinsvm:~$

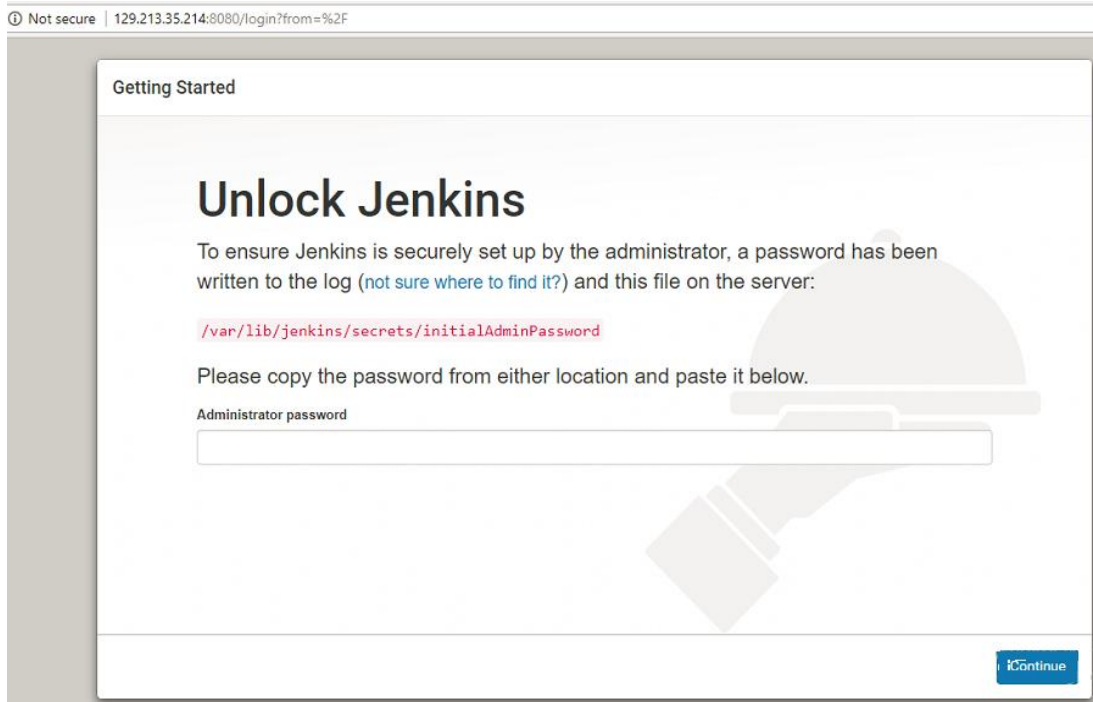
```

4. Now open any browser and paste the Jenkins IP address that we get in the access information to login to the Jenkins dashboard

Ex: <http://JenkinsVMIP:8080>

**<http://129.146.76.242:8080>**





5. To unlock Jenkins you need to login to the Jenkins machine and use the following command to access the initialAdminPassword

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

```
ubuntu@jenkinsvm: ~  
ubuntu@jenkinsvm:/var/lib/jenkins/jobs$ cd  
ubuntu@jenkinsvm:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword  
efel8c4b13e6437592b30db73fac15b6  
ubuntu@jenkinsvm:~$
```

Paste the password you get as shown below

129.213.35.214:8080/login?from=%2F

### Getting Started

## Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

Continue

When prompted please click on install suggested plugins, so Jenkins automatically installs the necessary plugins

129.213.35.214:8080

### Getting Started

## Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

#### Install suggested plugins

Install plugins the Jenkins community finds most useful.

#### Select plugins to install

Select and install plugins most suitable for your needs.

Getting Started

# Getting Started

✓ Folders Plugin	✓ OWASP Markup Formatter Plugin	✓ Build Timeout	✓ Credentials Binding Plugin	Folders Plugin
⌚ Timestampers	⌚ Workspace Cleanup Plugin	⌚ Ant Plugin	⌚ Gradle Plugin	** bouncycastle API Plugin
⌚ Pipeline	⌚ GitHub Branch Source Plugin	⌚ Pipeline: GitHub Groovy Libraries	⌚ Pipeline: Stage View Plugin	** Structs Plugin
⌚ Git plugin	⌚ Subversion Plug-in	⌚ SSH Slaves plugin	⌚ Matrix Authorization Strategy Plugin	** JUnit Plugin
⌚ PAM Authentication plugin	⌚ LDAP Plugin	⌚ Email Extension Plugin	⌚ Mailer Plugin	OWASP Markup Formatter Plugin
				** Pipeline: Step API
				** SCM API Plugin
				** Script Security Plugin
				** Pipeline: API
				** Pipeline: Supporting APIs
				** Pipeline: Job
				** Token Macro Plugin
				Build Timeout
				** Credentials Plugin
				** SSH Credentials Plugin
				** Plain Credentials Plugin
				Credentials Binding Plugin
				** - required dependency

Please wait while the plugins get installed

6. When prompted create a new admin username and password which you can use to access the dashboard.

Getting Started

# Create First Admin User

Username:

Password:

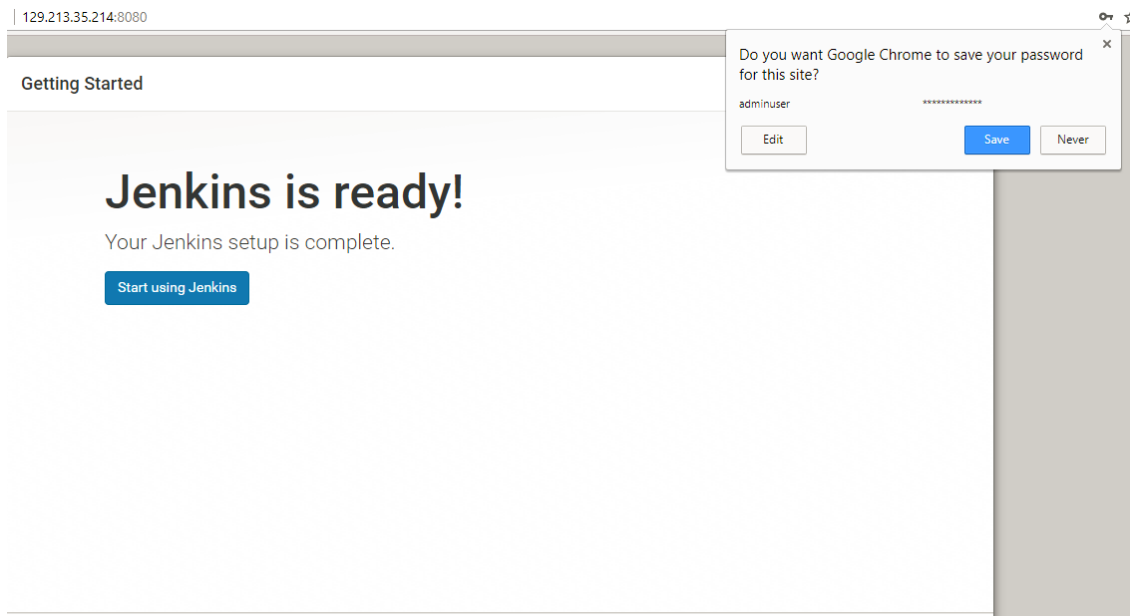
Confirm password:

Full name:

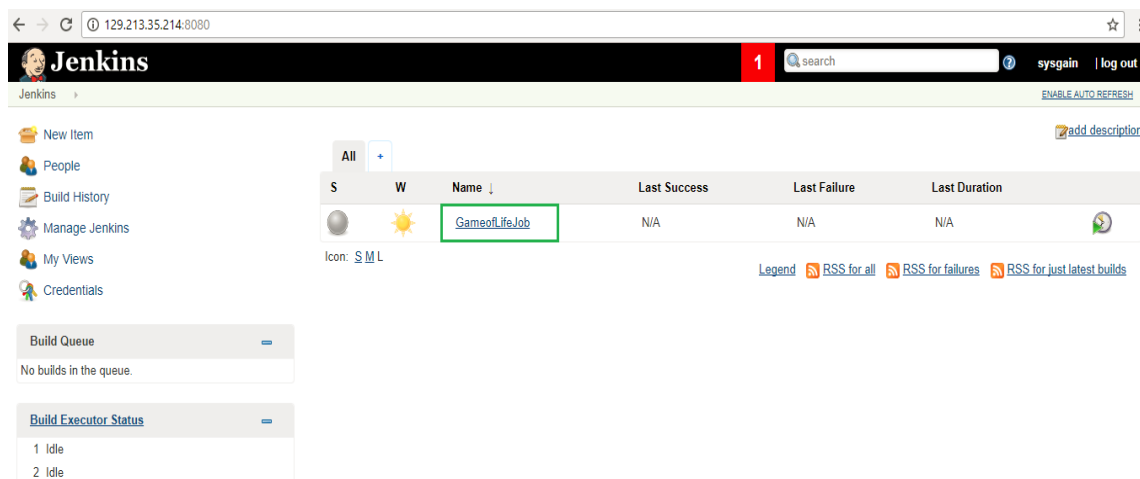
E-mail address:

Jenkins 2.73.2

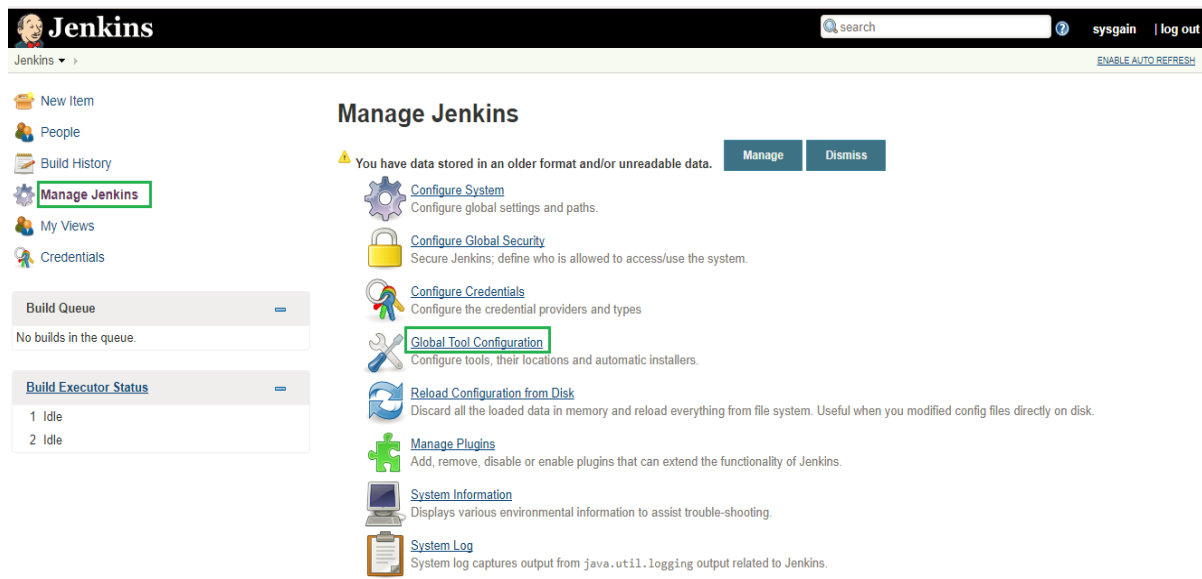
[Continue as admin](#) [Save and Finish](#)



7. When you access the dashboard you can see a job configured: i.e. **GameofLifeJob**

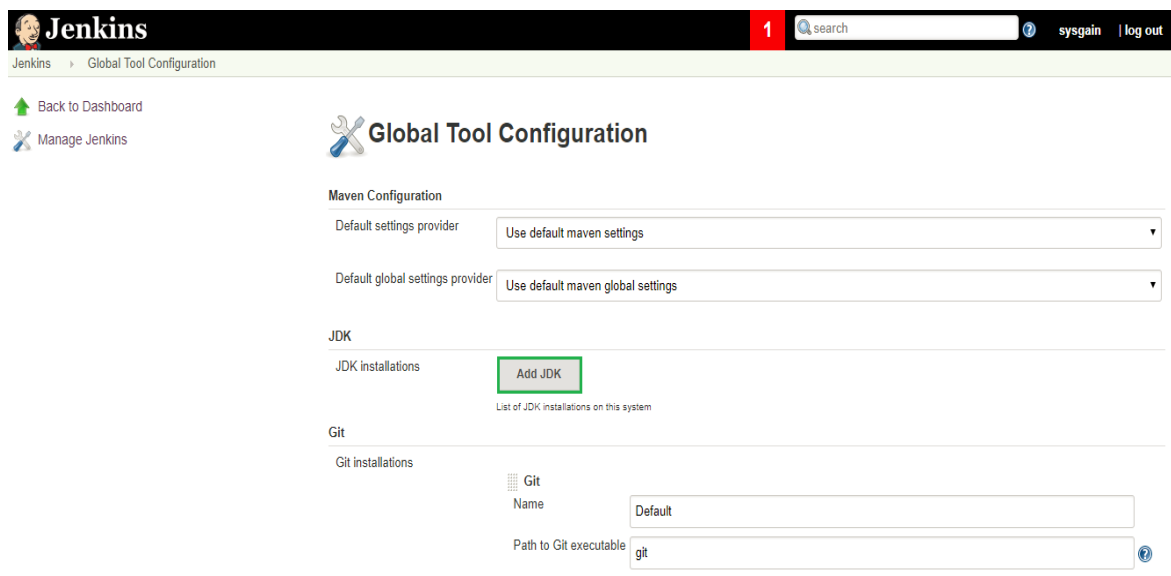


8. Go to manage Jenkins click on global tool configuration, here you need to add the JDK path as shown in the below screenshot



The screenshot shows the Jenkins 'Manage Jenkins' dashboard. On the left sidebar, 'Manage Jenkins' is highlighted with a green box. The main content area is titled 'Manage Jenkins' and contains a list of configuration options. 'Global Tool Configuration' is highlighted with a green box. A warning message at the top states: 'You have data stored in an older format and/or unreadable data.' with 'Manage' and 'Dismiss' buttons.

- [Configure System](#): Configure global settings and paths.
- [Configure Global Security](#): Secure Jenkins; define who is allowed to access/use the system.
- [Configure Credentials](#): Configure the credential providers and types.
- [Global Tool Configuration](#): Configure tools, their locations and automatic installers.
- [Reload Configuration from Disk](#): Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.
- [Manage Plugins](#): Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
- [System Information](#): Displays various environmental information to assist trouble-shooting.
- [System Log](#): System log captures output from java.util.logging output related to Jenkins.



The screenshot shows the 'Global Tool Configuration' page in Jenkins. The breadcrumb trail is 'Jenkins > Global Tool Configuration'. The page is divided into sections for 'Maven Configuration', 'JDK', and 'Git'. In the 'JDK' section, the 'Add JDK' button is highlighted with a green box. Below it, there is a table for 'List of JDK installations on this system'. In the 'Git' section, there are input fields for 'Name' (set to 'Default') and 'Path to Git executable' (set to 'git').

### Global Tool Configuration

#### Maven Configuration

Default settings provider:

Default global settings provider:

#### JDK

JDK installations:

List of JDK installations on this system

#### Git

Git installations

Git
Name: <input type="text" value="Default"/>
Path to Git executable: <input type="text" value="git"/>

Jenkins 1 search sysgain log out

Jenkins > Global Tool Configuration

[Back to Dashboard](#)  
[Manage Jenkins](#)

## Global Tool Configuration

**Maven Configuration**

Default settings provider:

Default global settings provider:

**JDK**

JDK installations

JDK	Name
<input checked="" type="checkbox"/>	<input type="text" value=""/>

- Required

☒ Install automatically

☐ Install from java.sun.com

Version:

☐ I agree to the Java SE Development Kit Licence Agreement

- Installing JDK requires Oracle account. [Please enter your username/password](#)

[Delete Installer](#)

9. Please paste the path as `/usr/lib/jvm/java-1.8.0-openjdk-amd64`. Choose any name

```
ubuntu@jen-jenkinsvm:~$ cd /usr/lib/jvm
ubuntu@jen-jenkinsvm:/usr/lib/jvm$ ls
default-java  java-1.8.0-openjdk-amd64  java-8-openjdk-amd64
ubuntu@jen-jenkinsvm:/usr/lib/jvm$ cd java-1.8.0-openjdk-amd64
ubuntu@jen-jenkinsvm:/usr/lib/jvm/java-1.8.0-openjdk-amd64$ ls
ASSEMBLY_EXCEPTION  bin  docs  include  jre  lib  man  src.zip  THIRD_PARTY_README
ubuntu@jen-jenkinsvm:/usr/lib/jvm/java-1.8.0-openjdk-amd64$
```

## Global Tool Configuration

### Maven Configuration

Default settings provider Use default maven settings

Default global settings provider Use default maven global settings

### JDK

#### JDK installations

 JDK  
Name JDK

JAVA\_HOME /usr/lib/jvm/java-1.8.0-openjdk-amd64/

☐ Install automatically



Delete JDK


Add JDK

List of JDK installations on this system


10. Under maven give any name and select the checkbox to install automatically

Maven

Maven installations

 Maven  
Name maven

☒ Install automatically

 Install from Apache  
Version 3.5.2

Add Installer

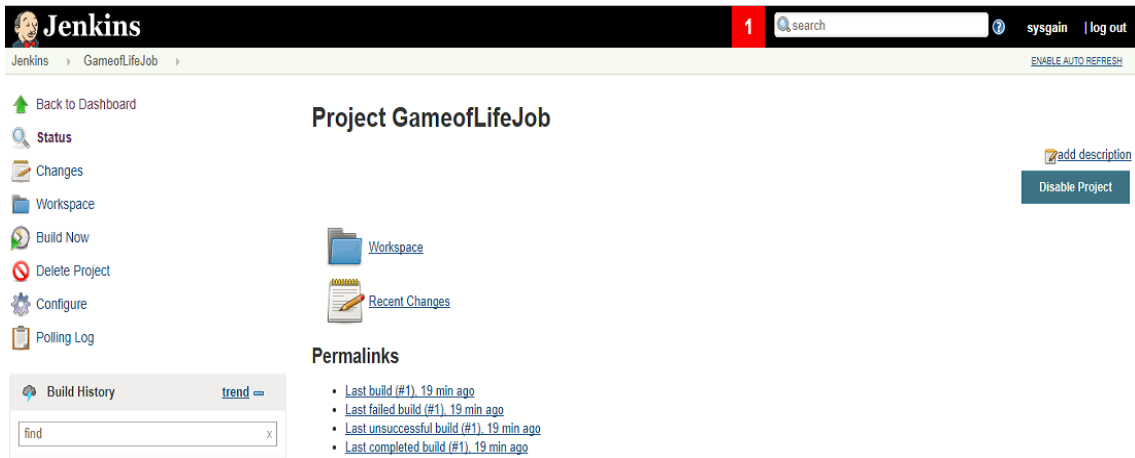
Add Maven

List of Maven installations on this system

Delete Installer

Delete Maven

11. Now click on the job and click on configure.

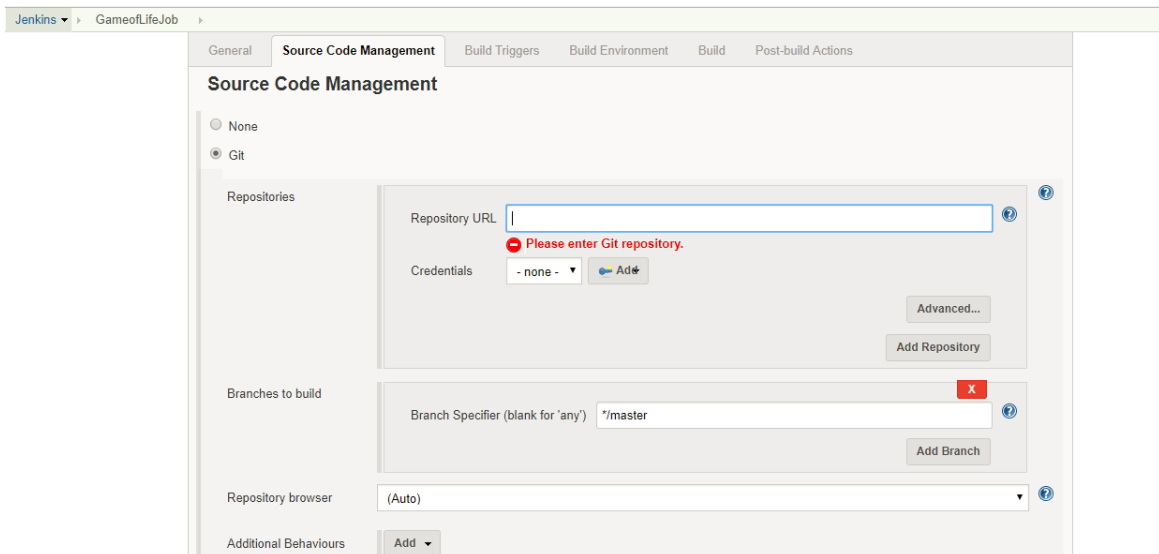


The screenshot shows the Jenkins interface for the 'Project GameofLifeJob'. The top navigation bar includes the Jenkins logo, a search bar, and links for 'sysgain' and 'log out'. Below the navigation bar, the left sidebar contains links for 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Delete Project', 'Configure', and 'Polling Log'. The main content area displays the project name 'Project GameofLifeJob' and a 'Disable Project' button. Below this, there are links for 'Workspace' and 'Recent Changes'. A 'Permalinks' section lists several build links, all indicating they are 19 minutes ago. At the bottom, there is a 'Build History' section with a search bar and a 'trend' button.

12. Under SourceCodeManagement give the GitHub repository URL (the game of life repository which you have forked earlier)

EX: <https://github.com/yougandar/game-of-life.git>

Under branch specifier select \*/master



The screenshot shows the Jenkins 'Source Code Management' configuration page. The 'General' tab is selected, and the 'Source Code Management' section is active. Under 'Repositories', the 'Git' option is selected. The 'Repository URL' field is empty, and a red error message 'Please enter Git repository.' is displayed. The 'Credentials' dropdown is set to '- none -'. The 'Branches to build' section shows the 'Branch Specifier (blank for 'any')' set to '\*/master'. The 'Repository browser' is set to '(Auto)'. There are buttons for 'Advanced...', 'Add Repository', 'Add Branch', and 'Add'.



yougandar / game-of-life  
forked from bogo-devops/game-of-life

Unwatch 1 Star 0 Fork 10,327

Code Pull requests 0 Projects 0 Wiki Insights Settings

Demo application for the 'Jenkins: The Definitive Guide' book <http://www.wakaleo.com/books/jenkins-...> Edit

Add topics

419 commits 42 branches 163 releases 4 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

This branch is even with bogo-devops:master.

bogotobogo Back to \* from +

gameoflife-acceptance-tests	Isolate the acceptance tests from the other modu	
gameoflife-build	Updated versions	
gameoflife-core	Back to * from +	3 years ago
gameoflife-deploy	Tidied up code	6 years ago
gameoflife-web	Fixed merge issue	5 years ago
.gitignore	Moved web tests into a separate module.	6 years ago
README.markdown	Changed README to markdown	5 years ago
infinittest.filters	First commit	7 years ago
pom.xml	Updated dependencies	3 years ago

Clone with HTTPS Use SSH

Use Git or checkout with SVN using the web URL.

<https://github.com/yougandar/game-of-life>

Open in Desktop Download ZIP

Click on Add -> Jenkins -> It will open the Jenkins Credentials provider

### Source Code Management

☐ None  
☒ Git

Repositories

Repository URL <https://github.com/yougandar/game-of-life.git>

Credentials - none - [Add](#)

[Advanced...](#)

[Add Repository](#)

Branches to build

Branch Specifier (blank for 'any')  [X](#)

[Add Branch](#)

Repository browser (Auto)

Additional Behaviours [Add](#)

☐ Subversion

13. Provide the credentials of your git account so Jenkins can access your repository

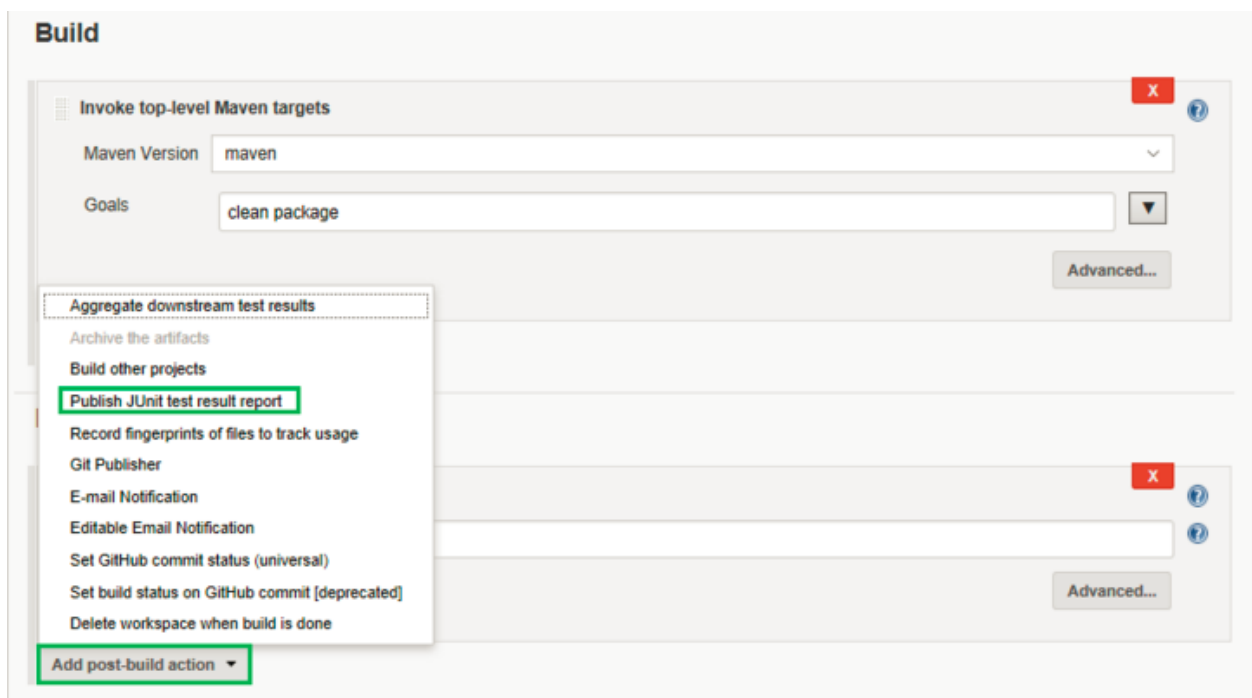
The screenshot shows the 'Jenkins Credentials Provider: Jenkins' dialog box. It has tabs for 'General', 'Source Code Management', 'Build Triggers', 'Build Environment', 'Build', and 'Post-build Actions'. The 'Source Code Management' tab is active. Under the 'Add Credentials' section, the following fields are visible: 'Domain' (Global credentials (unrestricted)), 'Kind' (Username with password), 'Scope' (Global (Jenkins, nodes, items, all child items, etc)), 'Username' (yougandar), 'Password' (masked with asterisks), 'ID' (empty), and 'Description' (empty). The 'Add' button is highlighted with a green box.

The screenshot shows the 'Source Code Management' configuration page. It has radio buttons for 'None', 'Git' (selected), and 'Subversion'. Under the 'Git' section, there are three main sections: 'Repositories', 'Branches to build', and 'Repository browser'. The 'Repositories' section has a 'Repository URL' field (https://github.com/yougandar/game-of-life.git) and a 'Credentials' dropdown (yougandar/\*\*\*\*\*). The 'Branches to build' section has a 'Branch Specifier (blank for 'any')' field (\*/\*master). The 'Repository browser' section has a dropdown set to '(Auto)'. The 'Additional Behaviours' section has an 'Add' button. The 'Add Repository' button is also visible.

The standard for test reporting in Java is an XML format used by JUnit.

Jenkins understands this format, so if our build produces JUnit XML test results, Jenkins can generate nice graphical test reports and statistics on test results over time, and also let us view the details of any test failures.

Jenkins also keeps track of how long our tests take to run, both globally, and per test-this can come in handy if we need to track down performance issues.



14) Go to the Post-build Actions section and check "Publish JUnit test result report" checkbox.

When Maven runs unit tests in a project, it automatically generates the XML test reports in a directory called surefire-reports in the target directory.

So, enter "\*\*/target/surefire-reports/\*.xml" in the "Test report XMLs" field. The two asterisks at the start of the path ("\*\*") are a best practice to make the configuration a bit more robust:

they allow Jenkins to find the target directory no matter how we have configured Jenkins to check out the source code.

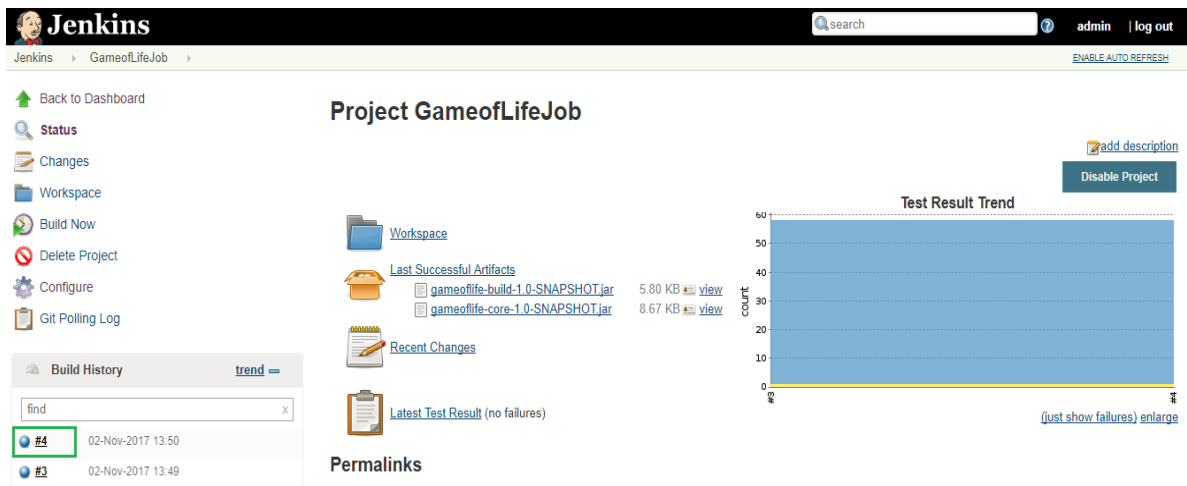
The screenshot shows the 'Post-build Actions' configuration page for a Jenkins job. It contains two main sections:

- Archive the artifacts:** A text field labeled 'Files to archive' contains the pattern `**/target/*.jar`. An 'Advanced...' button is located to the right.
- Publish JUnit test result report:** A text field labeled 'Test report XMLs' contains the pattern `**/target/surefire-reports/*.xml`. Below this field is a descriptive text: 'Fileset "includes" setting that specifies the generated raw XML report files, such as 'myproject/target/test-reports/\*.xml'. Basedir of the fileset is the workspace root.' There are two checkboxes: 'Retain long standard output/error' (unchecked) and 'Allow empty results' (unchecked). The 'Health report amplification factor' is set to '1.0', with a note below it: '1% failing tests scores as 99% health. 5% failing tests scores as 95% health'. At the bottom, there is a 'Do not fail the build on empty test results' checkbox (unchecked).

At the bottom left, there is a button labeled 'Add post-build action' with a dropdown arrow.

15. Now when you build the project Jenkins performs the tests and give the test results accordingly indicating success or failure.

The screenshot shows the Jenkins web interface for a job named 'Project GameofLifeJob'. The top navigation bar includes the Jenkins logo, a search bar, and links for 'sysgain' and 'log out'. The left sidebar contains a list of actions: 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now' (highlighted with a green box), 'Delete Project', 'Configure', and 'Git Polling Log'. The main content area displays the job's status, including a 'Workspace' icon, a 'Recent Changes' icon, and a 'Permalinks' section with a list of links: 'Last build (#1), 25 min ago', 'Last failed build (#1), 25 min ago', 'Last unsuccessful build (#1), 25 min ago', and 'Last completed build (#1), 25 min ago'. On the right side, there are buttons for 'add description' and 'Disable Project'. At the bottom left, there is a 'Build History' section with a search bar and a 'trend' link.



In the below screenshot we can see that the build was successful in the console output.

```
Jenkins > GameofLifeJob > #4

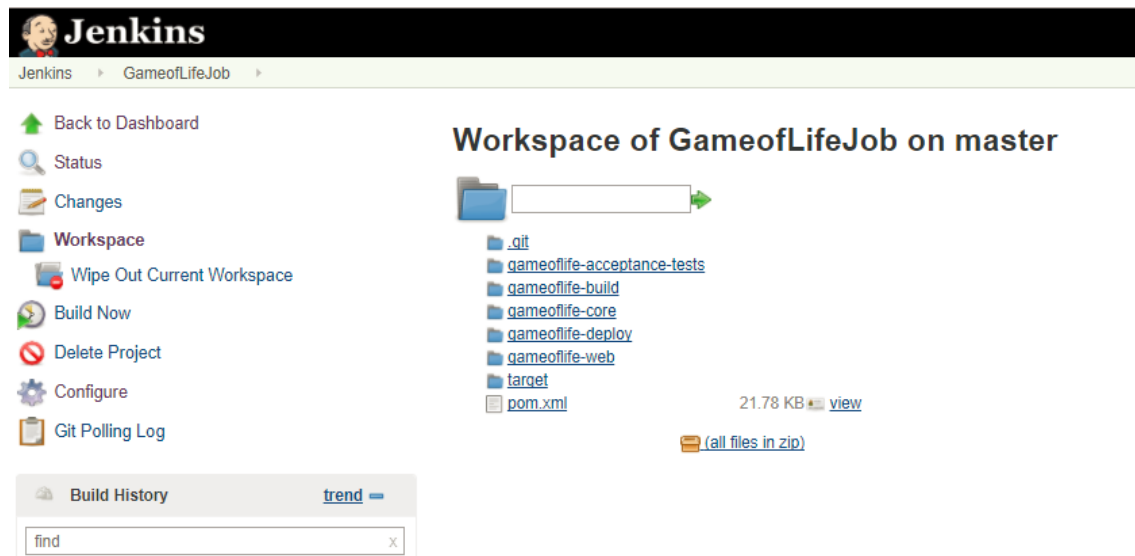
Tests run: 7, Failures: 0, Errors: 0, Skipped: 0
Running com.wakaleo.gameoflife.webtests.controllers.WhenCreatingNewGame
Tests run: 3, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.333 sec
Running com.wakaleo.gameoflife.webtests.controllers.WhenSpawningANewGeneration
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.028 sec

Results :

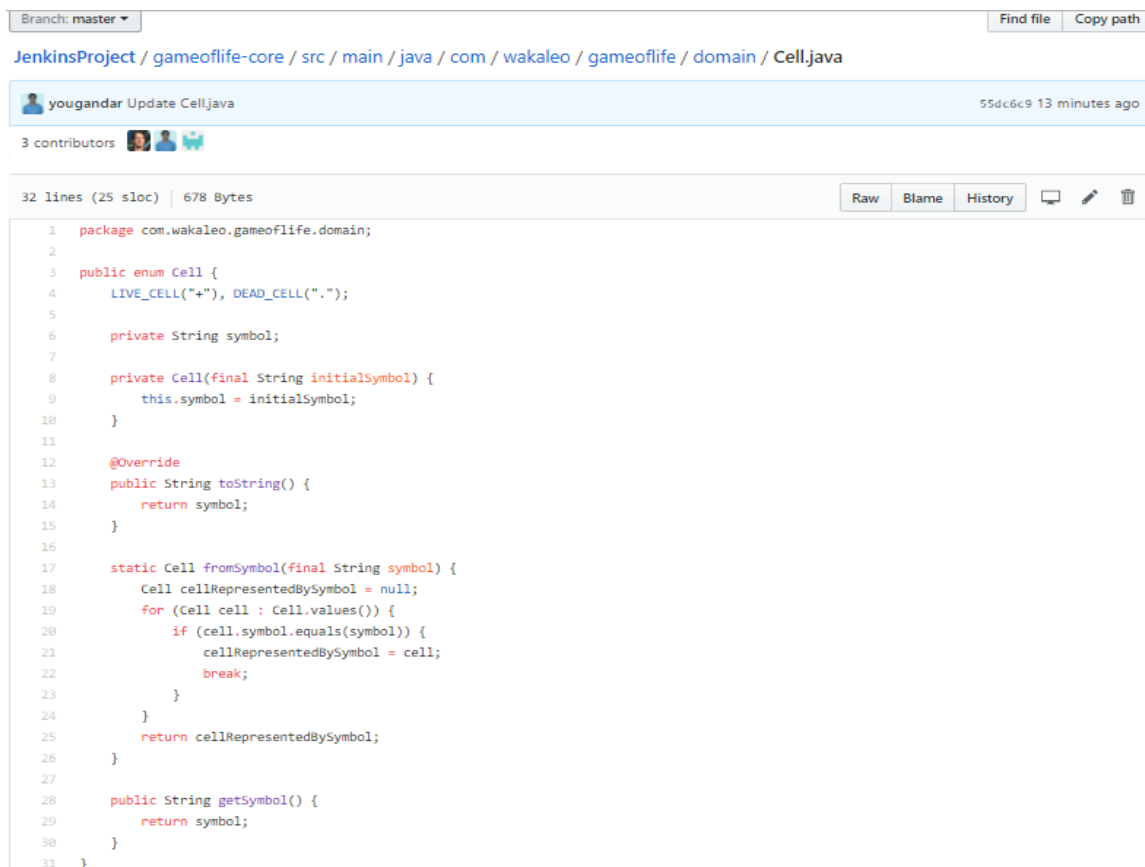
Tests run: 7, Failures: 0, Errors: 0, Skipped: 0

[INFO] --- maven-easyb-plugin:1.4:test (default) @ gameoflife-web ---
[INFO] /var/lib/jenkins/workspace/GameofLifeJob/gameoflife-web/src/test/stories does not exists. Skipping easyb testing
[INFO] --- maven-war-plugin:2.1.1:war (default-war) @ gameoflife-web ---
[INFO] Packaging webapp
[INFO] Assembling webapp [gameoflife-web] in [/var/lib/jenkins/workspace/GameofLifeJob/gameoflife-web/target/gameoflife]
[INFO] Processing war project
[INFO] Copying webapp resources [/var/lib/jenkins/workspace/GameofLifeJob/gameoflife-web/src/main/webapp]
[INFO] Webapp assembled in [35 msecs]
[INFO] Building war: /var/lib/jenkins/workspace/GameofLifeJob/gameoflife-web/target/gameoflife.war
[INFO] WEB-INF/web.xml already added, skipping
[INFO] -----
[INFO] Reactor Summary:
[INFO]
[INFO] gameoflife ..... SUCCESS [ 1.567 s]
[INFO] gameoflife-build ..... SUCCESS [ 0.775 s]
[INFO] gameoflife-core ..... SUCCESS [ 3.465 s]
[INFO] gameoflife-web ..... SUCCESS [ 1.681 s]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 7.730 s
[INFO] Finished at: 2017-11-02T13:50:35Z
[INFO] Final Memory: 30M/549M
[INFO] -----
Archiving artifacts
Recording test results
Finished: SUCCESS
```

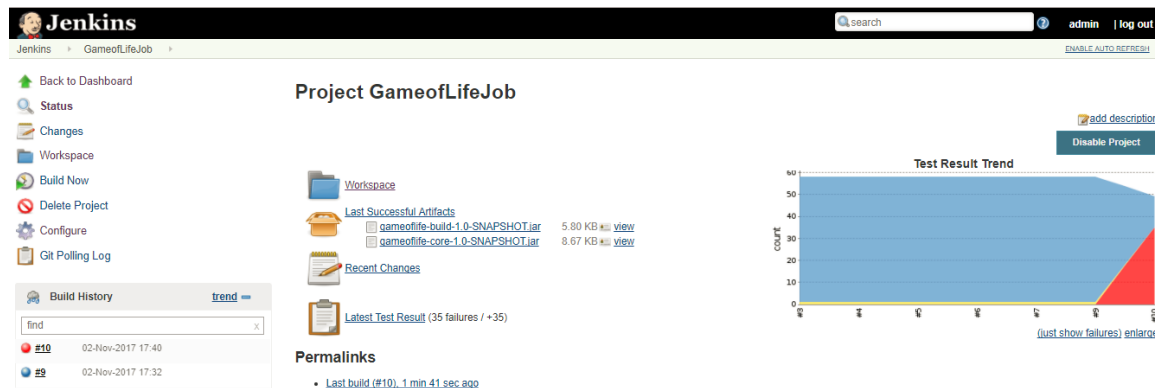
16. Click on the Workspace of project as show in the below screenshot.



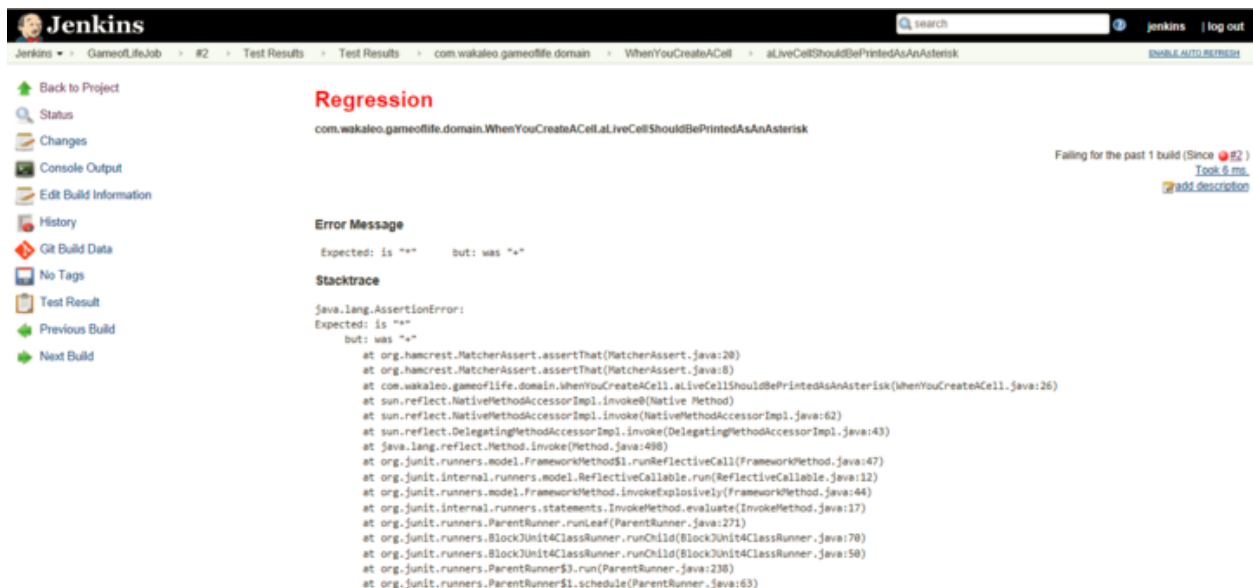
17. Now, what we can do is change to source code to demonstrate Jenkins failed test results. In the source code if we change the LIVE\_CELL (\*) to LIVE\_CELL (+) and build the job again.



18. Here we can see that the build has failed and also review the results as what has caused the failure.



As you can see in the screenshot below we can check the failed results and figure put what has caused the error ( as shown in the error message below : Expected is “\*” but: was “+”



19. Now once we revert back to the original source code, we can see that build is successful. And also we can see the test results trend.

Branch: master
Find file
Copy path

JenkinsProject / gameoflife-core / src / main / java / com / wakaleo / gameoflife / domain / Cell.java

wakaleo Restored working tests
118e888 on Apr 12, 2013
2 contributors

32 lines (25 sloc) 678 Bytes
Raw Blame History

```

1 package com.wakaleo.gameoflife.domain;
2
3 public enum Cell {
4     LIVE_CELL("*"), DEAD_CELL(".");
5
6     private String symbol;
7
8     private Cell(final String initialSymbol) {
9         this.symbol = initialSymbol;
10    }
11
12    @Override
13    public String toString() {
14        return symbol;
15    }
16
17    static Cell fromSymbol(final String symbol) {
18        Cell cellRepresentedBySymbol = null;
19        for (Cell cell : Cell.values()) {
20            if (cell.symbol.equals(symbol)) {
21                cellRepresentedBySymbol = cell;
22                break;
23            }
24        }
25        return cellRepresentedBySymbol;
26    }
27
28    public String getSymbol() {
29        return symbol;
30    }
31 }

```

Jenkins

admin | log out

Jenkins > GameofLifeJob >

ENABLE AUTO REFRESH

Back to Dashboard
Status
Changes
Workspace
Build Now
Delete Project
Configure
Git Polling Log

Project GameofLifeJob

Workspace

Last Successful Artifacts
gameoflife-build-1.0-SNAPSHOT.jar 5.80 KB view
gameoflife-core-1.0-SNAPSHOT.jar 8.67 KB view

Recent Changes

Latest Test Result (no failures)

Build History trend

Build	Status	Time
#11	Success	02-Nov-2017 17:43
#10	Failure	02-Nov-2017 17:40
#9	Success	02-Nov-2017 17:32

Permalinks

- Last build (#11), 16 sec ago
- Last stable build (#11), 16 sec ago
- Last successful build (#11), 16 sec ago

Test Result Trend

add description
Disable Project

### 3. Conclusion

Now that you've had the opportunity to try Jenkins for yourself and perform a pipeline deployment job, you should have a good understanding on how easy it is to incorporate a Continuous Integration process into your existing delivery workflow.