



Download Best Udemy Course	2
Download Full Version of This Course	2
Jenkins Installation	2
Jenkins on Windows	3
Jenkins on Ubuntu	9
Jenkins on Mac	12
Jenkins the Basic	14
Create “Hello world” Job	14
Add Build Step	15
Start the First Build	16
Console Output	17
Intro GOL Project (or World without Jenkins)	18
Git Installation and Clone GOL Repository	18
Maven Installation	20
Run Maven Build Manually	22
Install Tomcat Server as Service	23
Deploy GOL to Tomcat Server	25
Continuous Integration with Jenkins	26
Jenkins in the Big Picture of CI, CD and DevOps	26
Create GOL Job	26
Source Code Management	28

Check if git is installed in local machine	28
Specify source code repository	29
Check if setting working fine	30
Build Triggers	33
Build	37
Configure Build Step	37
Check Log and Workspace	39

Download Best Udemy Development Course

Download at <http://devcourses.co>

What is Good Course Look Like ?

- Start from Beginner
- Step by Step Explained Method
- Alot of Example to Make Thing Clear
- Alot of Exercise, Quize
- Real Life Hand on Projects Solve Real Problems
- Boot up to Launch New Carrier

Download Full Version of This Course

<https://www.udemy.com/jkcourse/?couponCode=DOWNLOAD>

2.5 hours of full HD screencast cover all content of this book.

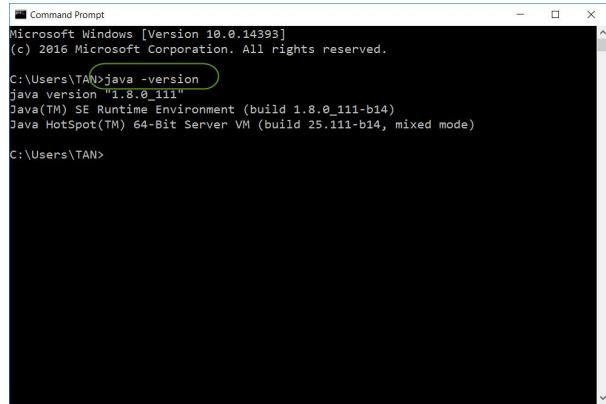
Jenkins Installation

In this chapter, we will jump into how to install and start up Jenkins in 3 environment : Window, Ubuntu and Mac.

In all environment, Jenkins will be installed as service, it mean Jenkins will automatically start up every time you machine power on or restart machine.

Jenkins on Windows

Check if java is installed : Java should be installed in Windows machine, to check if Java is installed, start a command prompt and typing in ***java -version*** Java version show up as below. Incase java not yet install, please install java before process to installation.



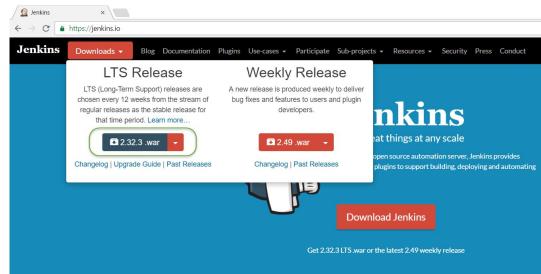
```
Command Prompt
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\TAN>java -version
java version "1.8.0_111"
Java(TM) SE Runtime Environment (build 1.8.0_111-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.111-b14, mixed mode)

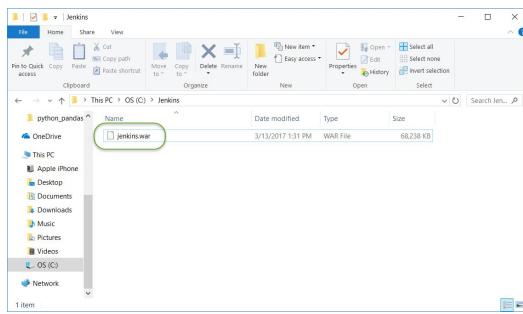
C:\Users\TAN>
```

Install process

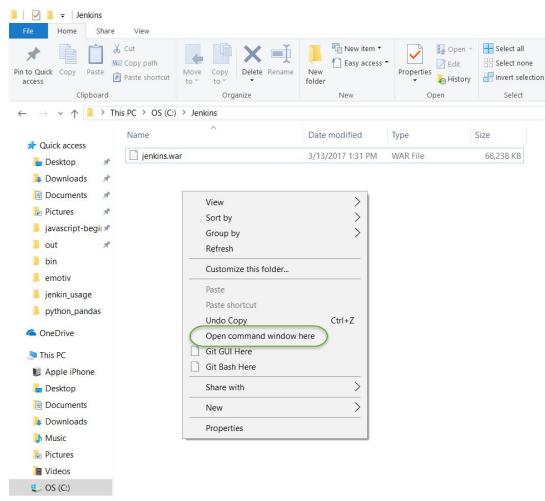
Step 1 : Go to <https://jenkins.io/> and download for stable version of Jenkins



Step 2 : Create a new folder for Jenkins (for example **C:\Jenkins**) and put file **jenkins.war** inside this folder

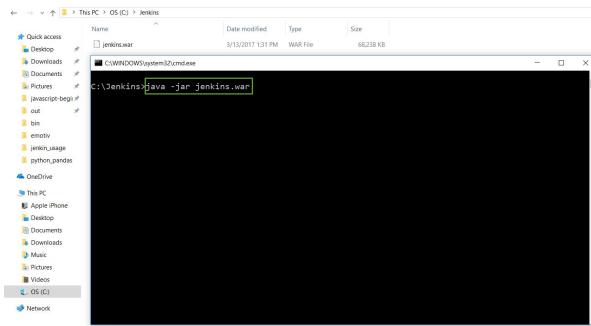


Step 3 : Press to shift and right click at same time to open context menu and select ***Open command window here***



Step 4 : Typing in command prompt following to start installation `java -jar jenkins.war`

Then wait a while for complete running.



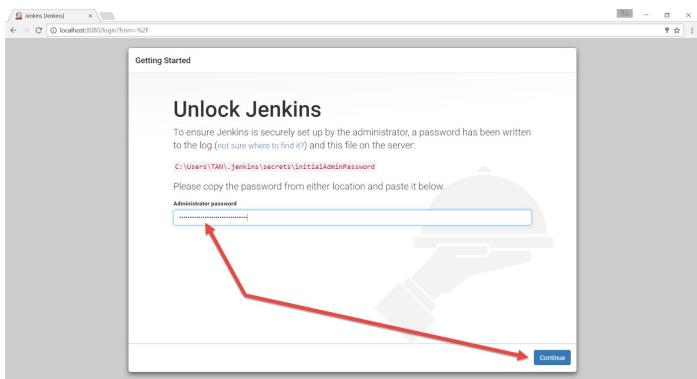
In the output log, you will see default password for first login of admin user, copy this password to notepad and save it for use on next step.

```

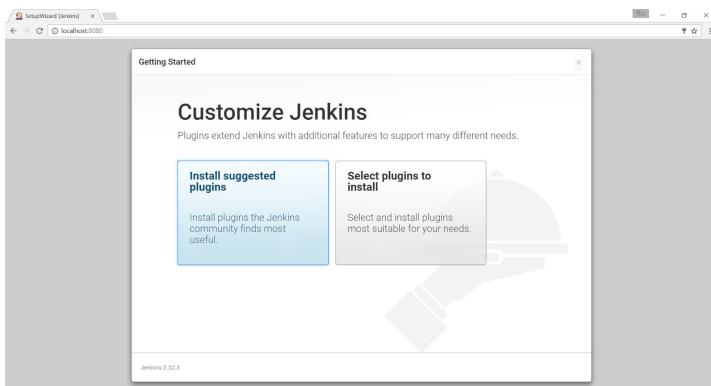
INFO:
*****
Denkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:
6043cce10c554047a3e958b60a4b427b
This may also be found at: C:\Users\TAN\.jenkins\secrets\initialAdminPassword
*****

```

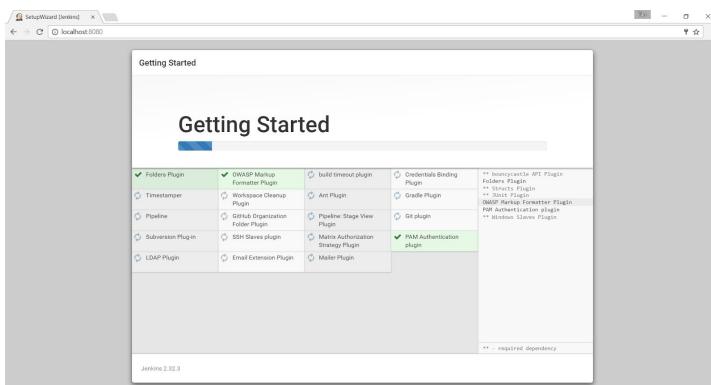
Step 5 : Now open browser and access to <http://localhost:8080> and then paste admin password to unlock Jenkins



Wait a while and following screen show up, select default option **Install suggested plugins**



At this step default plugins will be installed.



Step 6 : Enter your user name, password for a new admin, then click to **Save and Finish**

Getting Started

Create First Admin User

Username: tan

Password:

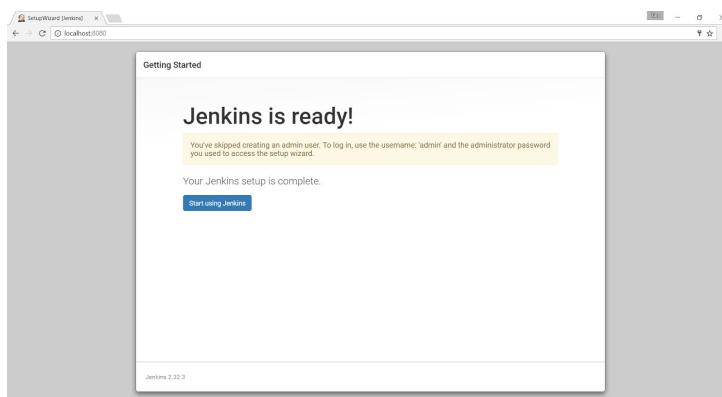
Confirm password:

Full name: tan

Email address: tanpham@gmail.com

Jenkins 2.32.3 Continue as admin Save and Finish

Step 7 : Click to **Start using Jenkins**, Congregation at this step you already complete for Jenkins installation



Jenkins auto login by new admin user already created

New Item

People

Build History

Manage Jenkins

My Views

Credentials

Build Queue

No builds in the queue

Build Executor Status

1 idle

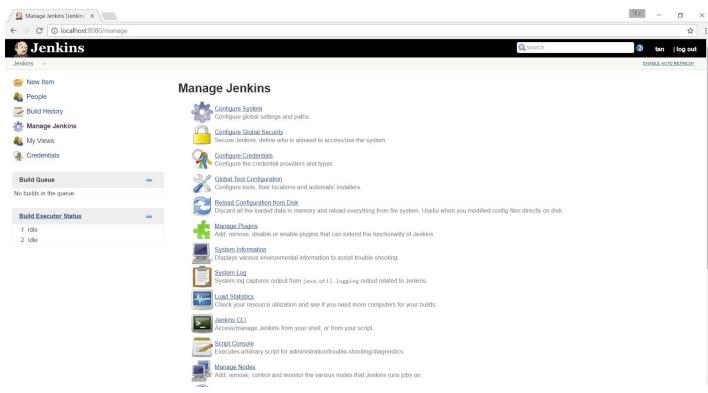
2 idle

Welcome to Jenkins!

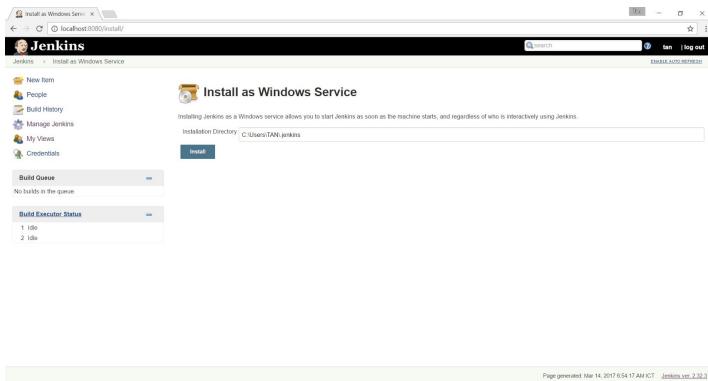
Please create new jobs to get started.

Page generated Mar 14, 2017 6:30:30 AM IST - REST API - Jenkins ver. 2.32.3

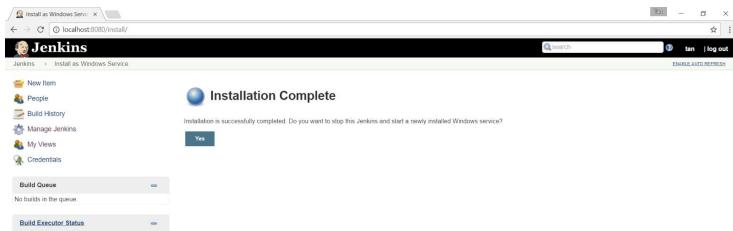
Step 8 : Click to **Manage Jenkins** from main screen, scroll down and click to **Install as Windows Service**



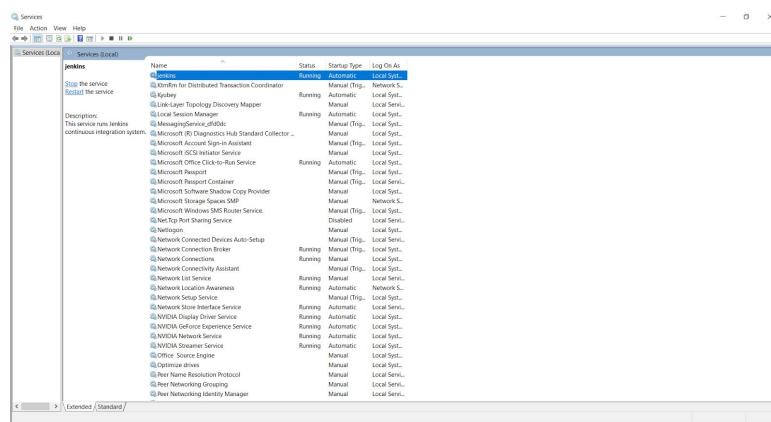
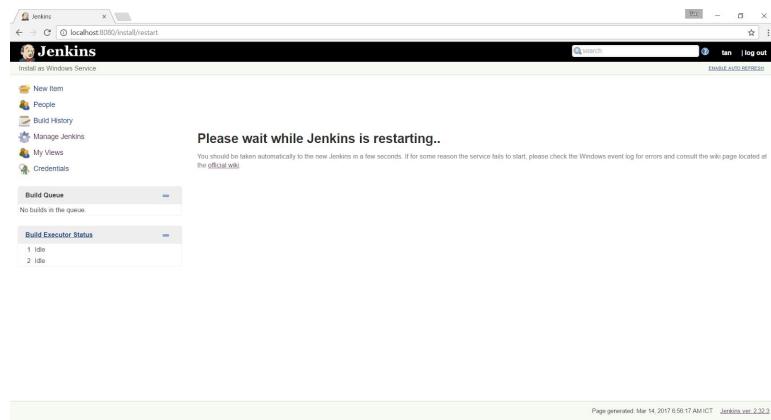
Click to **Install**



Click to **Yes**

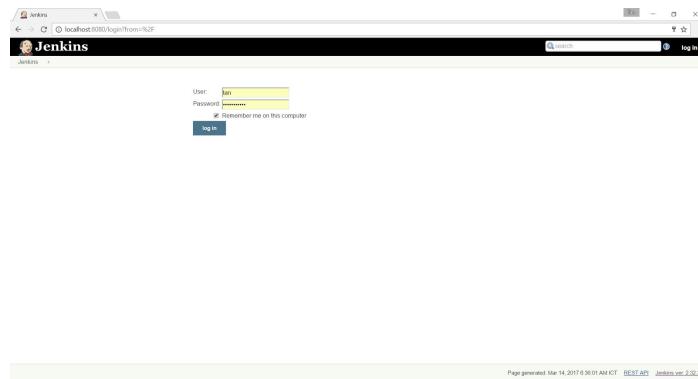


Wait some time and you will see jenkins service is running from windows service manager



That it, You complete install Jenkins on Windows as service. From now on every time Windows power up or restart, Jenkins will ready to access at <http://localhost:8080>

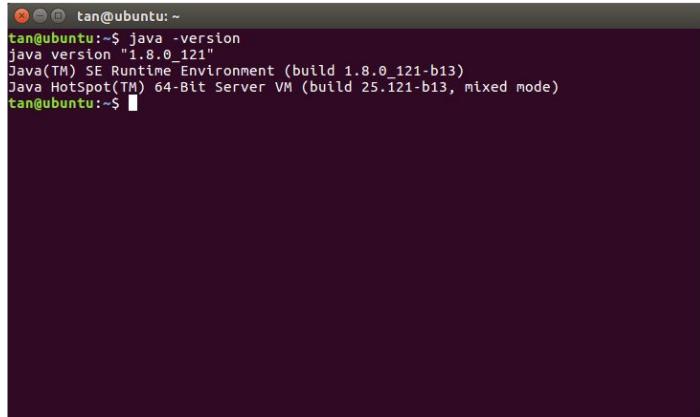
You can try to login Jenkins with above user created.



Jenkins on Ubuntu

Check if java is installed

Java should be installed in Ubuntu machine, to check if Java is installed, start a terminal and typing in **java -version** , If Java already installed, you will see Java version show up.

A screenshot of a terminal window on an Ubuntu system. The command 'java -version' is entered and its output is displayed. The output shows Java version 1.8.0_121, Java(TM) SE Runtime Environment (build 1.8.0_121-b13), and Java HotSpot(TM) 64-Bit Server VM (build 25.121-b13, mixed mode).

```
tan@ubuntu:~$ java -version
java version "1.8.0_121"
Java(TM) SE Runtime Environment (build 1.8.0_121-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.121-b13, mixed mode)
tan@ubuntu:~$
```

Install Process

Step 1 : Open terminal and typing in following command

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

 sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ >
 /etc/apt/sources.list.d/jenkins.list'

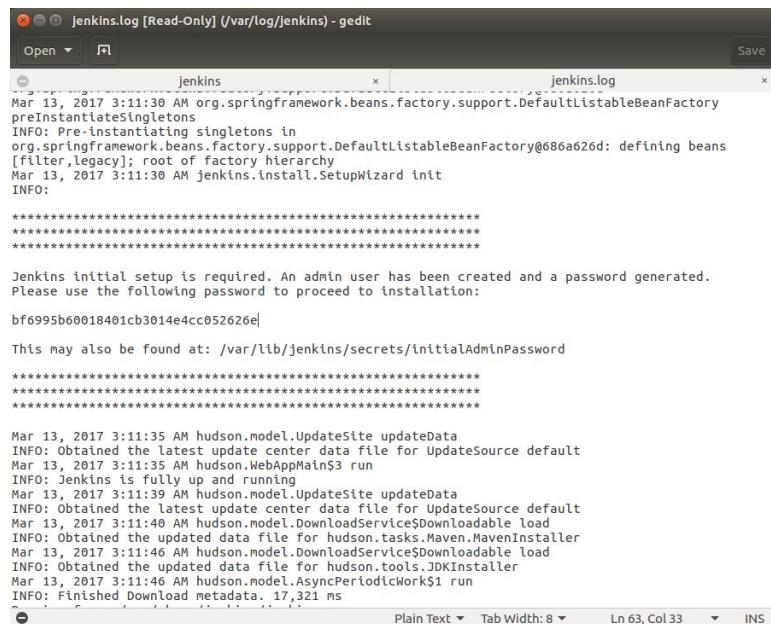
 sudo apt-get update

 sudo apt-get install jenkins
```

The above code do following job :

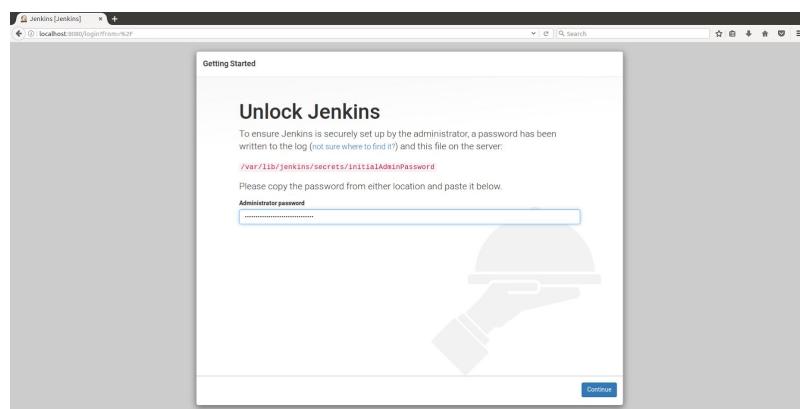
- Jenkins will be launched as a daemon up on start. See /etc/init.d/jenkins for more details.
- The 'jenkins' user is created to run this service.
- Log file will be placed in /var/log/jenkins/jenkins.log. Check this file if you are troubleshooting Jenkins.
- /etc/default/jenkins will capture configuration parameters for the launch like e.g JENKINS_HOME
- By default, Jenkins listen on port 8080. Access this port with your browser to start configuration.

Step 2 : Open the file **/var/log/jenkins/jenkins.log** and copy the default password which is automatically generated during installation

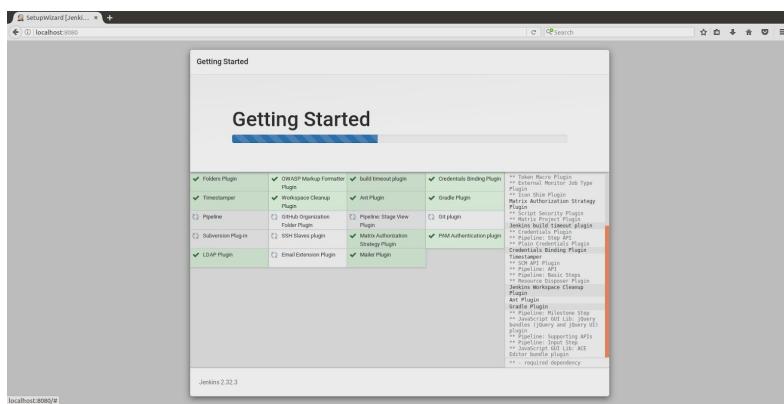
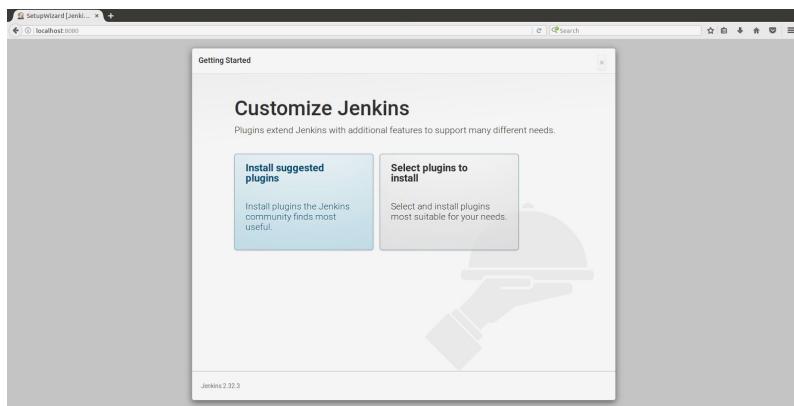


```
jenkins.log [Read-Only] (/var/log/jenkins) - gedit
Open Save
jenkins jenkins.log
Mar 13, 2017 3:11:30 AM org.springframework.beans.factory.support.DefaultListableBeanFactory
preInstantiateSingletons
INFO: Pre-instantiating singletons in
org.springframework.beans.factory.support.DefaultListableBeanFactory@686a626d: defining beans
[filter,legacy]; root of factory hierarchy
Mar 13, 2017 3:11:30 AM jenkins.install.SetupWizard init
INFO:
*****
Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:
bf6995b60018401cb3014e4cc052626e
This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
*****
Mar 13, 2017 3:11:35 AM hudson.model.UpdateSite updateData
INFO: Obtained the latest update center data file for UpdateSource default
Mar 13, 2017 3:11:35 AM hudson.WebAppMain$3 run
INFO: Jenkins is fully up and running
Mar 13, 2017 3:11:39 AM hudson.model.UpdateSite updateData
INFO: Obtained the latest update center data file for UpdateSource default
Mar 13, 2017 3:11:40 AM hudson.model.DownloadService$Downloadable load
INFO: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
Mar 13, 2017 3:11:40 AM hudson.model.DownloadService$Downloadable load
INFO: Obtained the updated data file for hudson.tools.JDKInstaller
Mar 13, 2017 3:11:45 AM hudson.model.AsyncPeriodicWork$1 run
INFO: Finished Download metadata., 17,321 ms
Plain Text Tab Width: 8 Ln 63, Col 33 INS
```

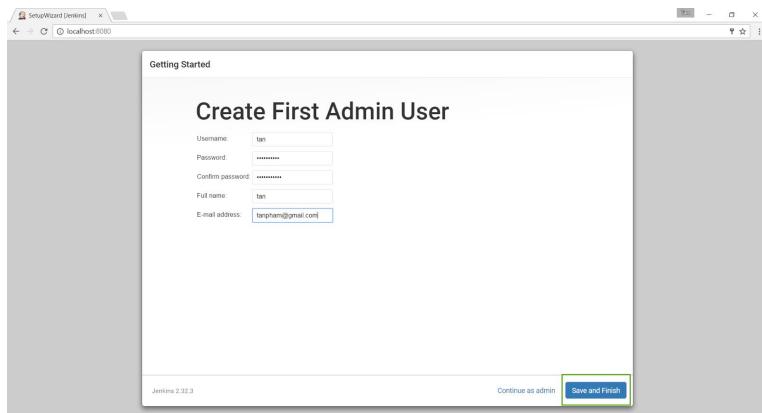
Step 3 : Open browser and access to Jenkins at <http://localhost:8080/> then paste the password above to unlock jenkins. Click to **Continue** button



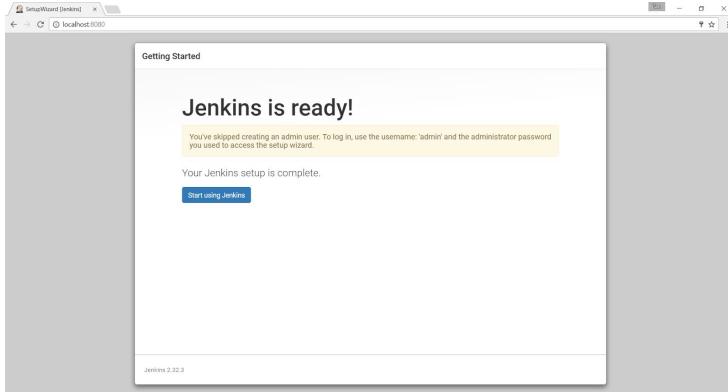
Step 4 : Click on **Install suggested plugins**



Step 5 : Enter your user name, password for a new admin.



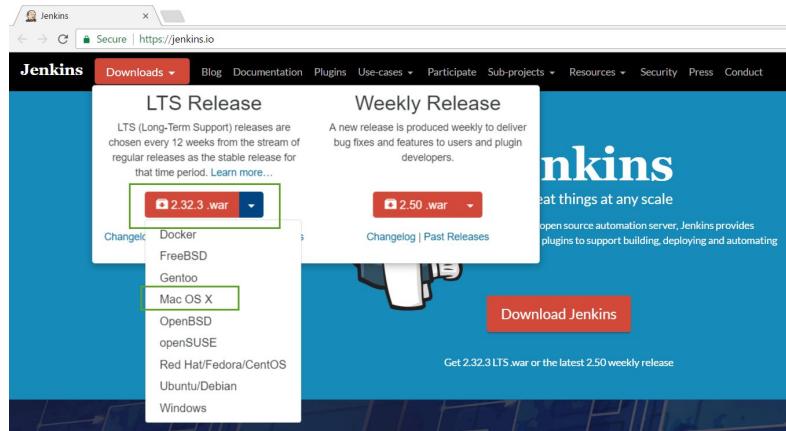
Step 6 : That it, now jenkins ready to use, click to ***Start using Jenkins***



From now Jenkins already installed in Ubuntu as daemon service, so every time you power on machine, Jenkins will start and ready to access at <http://localhost:8080/>

Jenkins on Mac

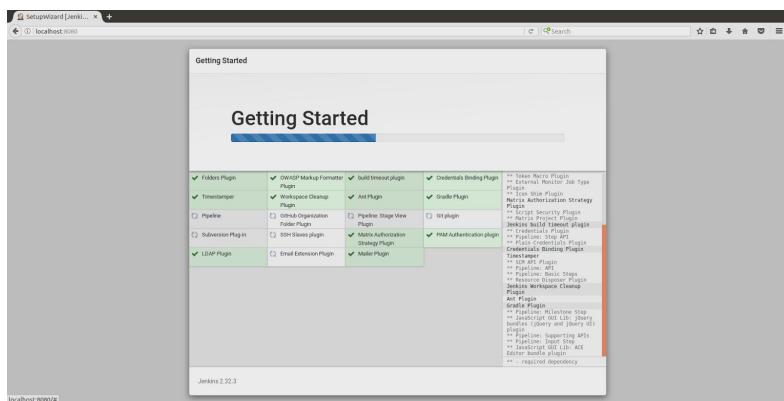
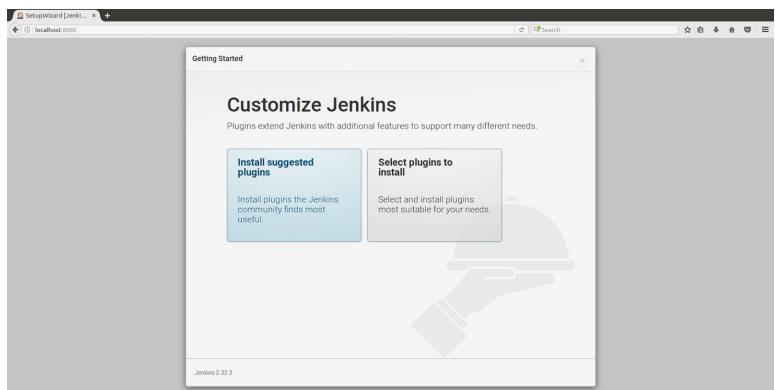
Step 1 : Download installer for Mac from <https://jenkins.io/> . Then run the installer file.



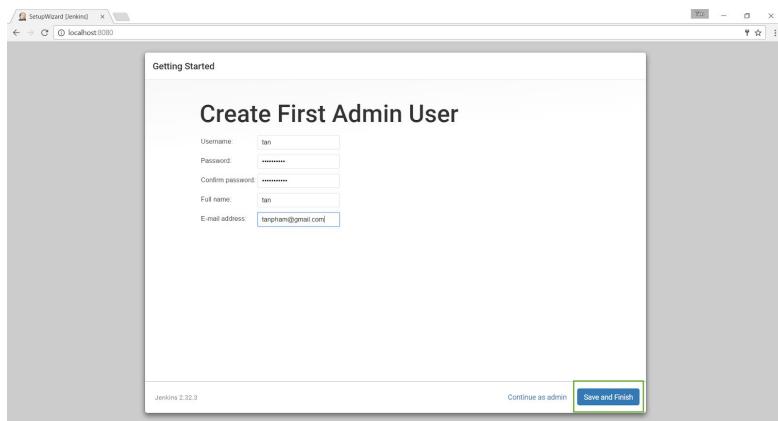
Step 2 : After complete install, you access Jenkins at <http://localhost:8080/>

Step 3 : Open file **/Users/Shared/Jenkins/Home/secrets/initialAdminPassword** and copy password use for unlock Jenkins

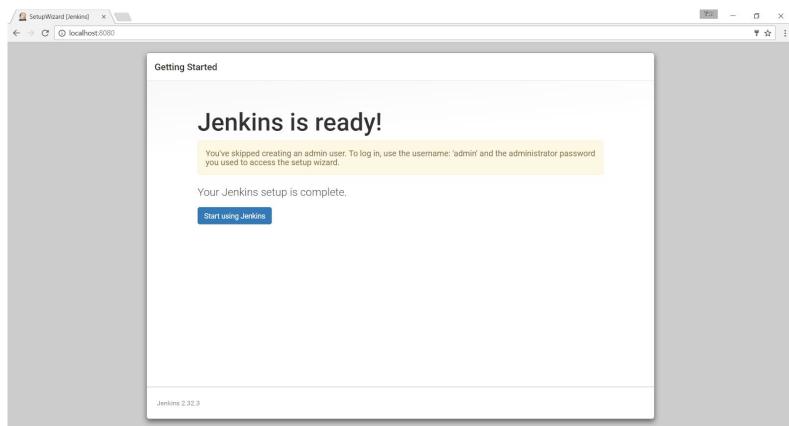
Step 4 : Click on **Install suggested plugins**



Step 5 : Enter your user name, password for a new admin.



Step 6 : That it, now jenkins ready to use, click to **Start using Jenkins**



From now Jenkins already installed in Mac as a service, so every time you power on machine, Jenkins will start and ready to access at <http://localhost:8080/>

Jenkins the Basic

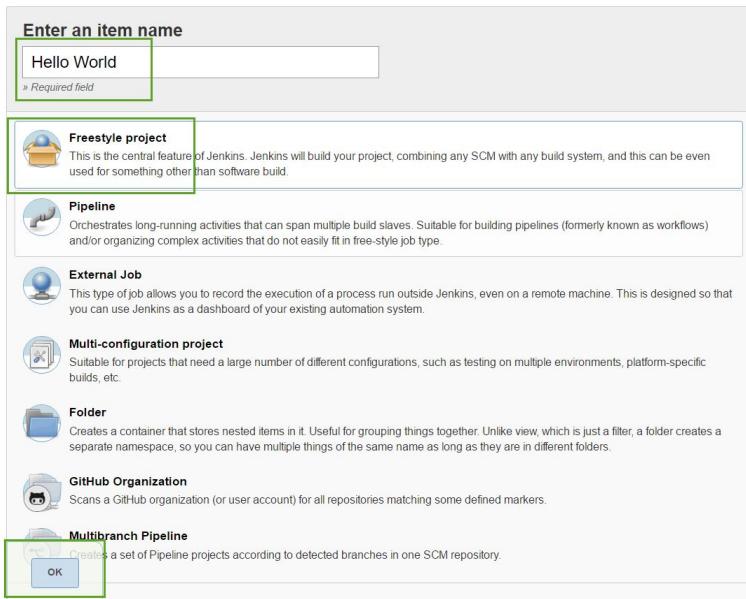
In this section I will explain Jenkins role in the world of continuous integration, continuous delivery and devops. And then we will go through steps in order to create “Hello World” Jenkins job.

Create “Hello world” Job

From Jenkins home page, click to **New Item**

A screenshot of the Jenkins dashboard. The title bar says 'Dashboard [Jenkins]'. The address bar shows 'localhost:8080'. The main menu bar has 'Jenkins' selected. Below the menu is a sidebar with links: 'New Item' (highlighted with a green border), 'People', 'Build History', 'Manage Jenkins', 'My Views', and 'Credentials'. Underneath the sidebar are two sections: 'Build Queue' (with a note 'No builds in the queue.') and 'Build Executor Status' (listing '1 Idle' and '2 Idle').

In create project page, typing in “Hello World” to project name, select **Freestyle project** and then click to save button.



Add Build Step

In configure job page, select **Build** tab, click to **Add build step** and select **Execute Windows batch command** (in case you are using Linux please select **Execute shell**)

Typing in command "echo Hello World". The purpose of this job is just echo to log message "Hello World" text.



Click **Save** button and we complete the first job!

Start the First Build

From job dashboard, click **Build Now** and you will see job run. This job just do one thing, invoke the command “echo hello world”

A screenshot of the Jenkins project dashboard for 'Hello World'. The title bar says 'Hello World [Jenkins]'. The URL in the browser is 'localhost:8080/job/Hello%20World/'. The main area is titled 'Project Hello World'. On the left, there is a sidebar with various options: 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now' (which is highlighted with a green box), 'Delete Project', 'Configure', and 'Move'. To the right of the sidebar, there are two links: 'Workspace' and 'Recent Changes'. At the bottom, there is a section titled 'Permalinks' with a 'Build History' table. The table has columns for 'Build' (with a gear icon), 'Time' (with a clock icon), 'Status' (with a green circle icon), and 'Trend' (with a downward arrow icon). There is also a search bar with 'find' and a 'trend' dropdown. At the very bottom, there are RSS feed links: 'RSS for all' and 'RSS for failures'.

The screenshot shows the Jenkins interface for the 'Hello World' project. At the top, there's a navigation bar with back, forward, and search buttons, and the URL 'localhost:8080/job/Hello%20World/'. Below it is the Jenkins logo and the project name 'Hello World'. A sidebar on the left contains links: Back to Dashboard, Status, Changes, Workspace, Build Now, Delete Project, Configure, and Move. The main content area is titled 'Project Hello World' and features a 'Workspace' link and a 'Recent Changes' link. Below that is a section titled 'Permalinks' with a 'Build History' table. The table has a 'find' input field, a dropdown for '#1' (selected), and a date 'Mar 19, 2017 9:14 PM'. It also includes 'RSS for all' and 'RSS for failures' links.

Console Output

Now job run completely, click to build number and select **Console Output**. You will see message “Hello World” show up inside build log. That it you just complete the first Jenkins job.

This screenshot is similar to the one above, showing the Jenkins interface for the 'Hello World' project. The build history table now includes a context menu for the first build (#1). The menu items are: Changes, Console Output, Edit Build Information, and Delete Build. The 'Console Output' option is highlighted with a green border and a green arrow pointing to it from the bottom right.

The screenshot shows the Jenkins interface for a 'Hello World' project. The top navigation bar includes links for 'Back to Project', 'Status', 'Changes', 'Console Output' (which is currently selected), 'View as plain text', 'Edit Build Information', and 'Delete Build'. The main content area is titled 'Console Output' and displays the following log entries:

```

Started by user tan
Building in workspace C:/Users/TAN/.jenkins/workspace>Hello World
[Hello World] $ cmd /c call C:/WINDOWS/TEMP/hudson9141285156312016941.bat
C:/Users/TAN/.jenkins/workspace>Hello World>echo Hello World
Hello World
C:/Users/TAN/.jenkins/workspace>Hello World>exit 0
Finished: SUCCESS

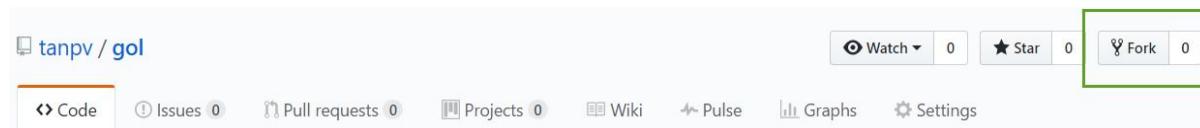
```

Intro GOL Project (or World without Jenkins)

In this book, project called GOL (game of life) is used to express the features related to Jenkins. So it will be helpful to review some information related to this project.

Git Installation and Clone GOL Repository

GOL project is hosted on Github at <https://github.com/tanpv/gol>. Please **fork** this repo to your Github account so you can try Jenkins features by yourself later.



To install Git, download software at <https://git-scm.com/>, and install with default option.

After install, open a cmd prompt and typing in **git --help** to check if install successfully

The screenshot shows a Windows Command Prompt window. The title bar says 'Command Prompt'. The command 'git --help' is run, displaying the usage information for the Git command-line interface. The output is as follows:

```

Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\TAN>git --help
usage: git [--version] [--help] [-C <path>] [-c name=value]
           [--exec-path=<path>] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           <command> [<args>]

```

Now create a folder name GOL and init it with command **git init**

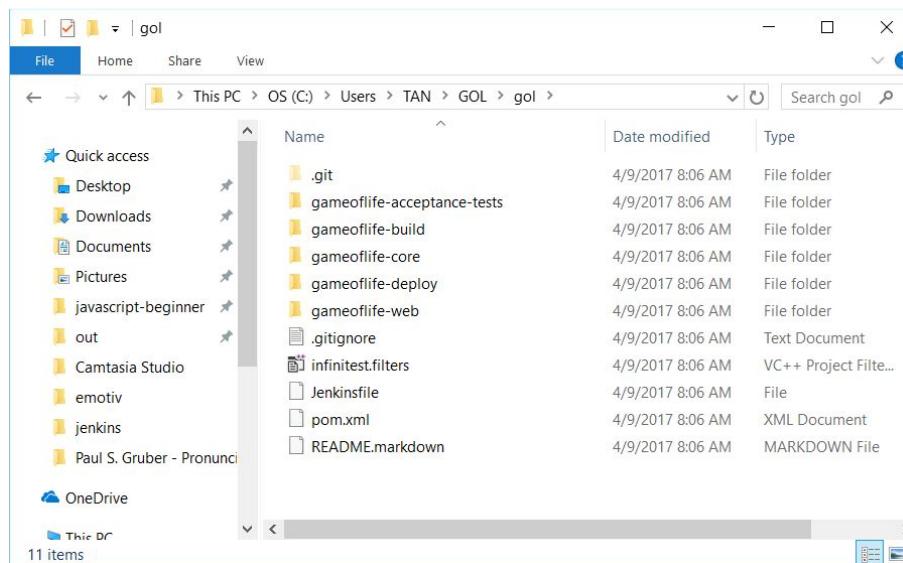
```
C:\WINDOWS\system32\cmd.exe
```

```
C:\Users\TAN\GOL>git init  
Initialized empty Git repository in C:/Users/TAN/GOL/.git/
```

And finally, clone the repo from Github to local with command **git clone**
<https://github.com/tanpv/gol.git>

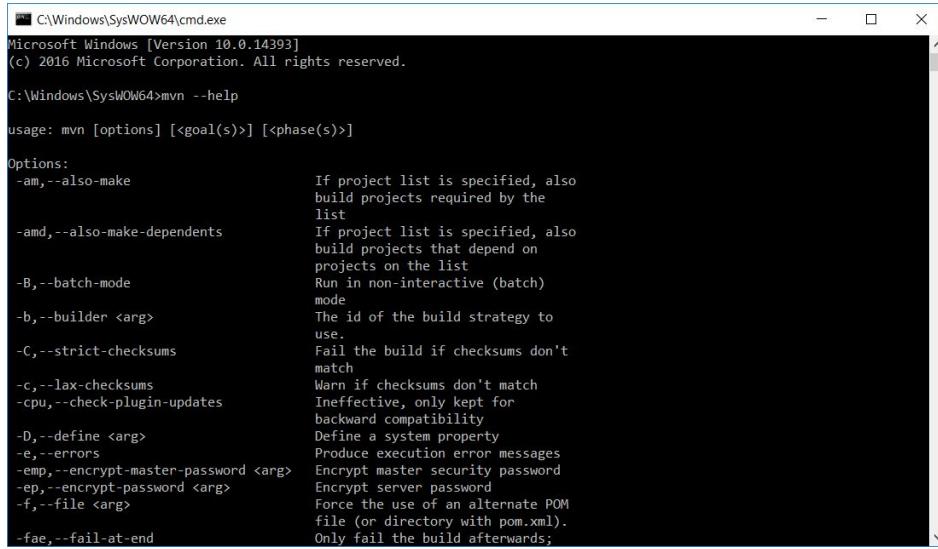
```
C:\WINDOWS\system32\cmd.exe - git clone https://github.com/tanpv/gol.git  
  
C:\Users\TAN\GOL>git init  
Initialized empty Git repository in C:/Users/TAN/GOL/.git/  
  
C:\Users\TAN\GOL>git clone https://github.com/tanpv/gol.git  
Cloning into 'gol'...  
remote: Counting objects: 1399, done.  
remote: Compressing objects: 100% (6/6), done.  
remote: Total 1399 (delta 2), reused 0 (delta 0), pack-reused 1393 receiving objects:  
Receiving objects: 99% (1386/1399), 1004.01 KiB | 594.00 KiB/s  
Receiving objects: 100% (1399/1399), 1.60 MiB | 594.00 KiB/s, done.  
Resolving deltas: 100% (1017/1017), done.
```

And finally, you will see GOL project is completely clone to local



Maven Installation

First step, maven should be installed in machine which running Jenkins job. To check this, just open command prompt and typing in **mvn --help**, you will see some all available maven command.



```
C:\Windows\SysWOW64\cmd.exe
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\SysWOW64>mvn --help

usage: mvn [options] [<goal(s)>] [<phase(s)>]

Options:
  -am,--also-make           If project list is specified, also
                           build projects required by the
                           list
  -amd,--also-make-dependents   If project list is specified, also
                           build projects that depend on
                           projects on the list
  -B,--batch-mode          Run in non-interactive (batch)
                           mode
  -b,--builder <arg>       The id of the build strategy to
                           use.
  -C,--strict-checksums    Fail the build if checksums don't
                           match
  -c,--lax-checksums       Warn if checksums don't match
                           Ineffective, only kept for
                           backward compatibility
  -cpu,--check-plugin-updates  Define a system property
                           Produce execution error messages
  -D,--define <arg>        Encrypt master security password
  -e,--errors               Encrypt server password
  -emp,--encrypt-master-password <arg>  Force the use of an alternate POM
  -ep,--encrypt-password <arg>   file (or directory with pom.xml).
  -f,--file <arg>          Only fail the build afterwards,
                           -fae,--fail-at-end
```

In case maven not yet install follow these steps to do maven installation (for Window)

- Download maven from link <http://maven.apache.org/download.cgi>

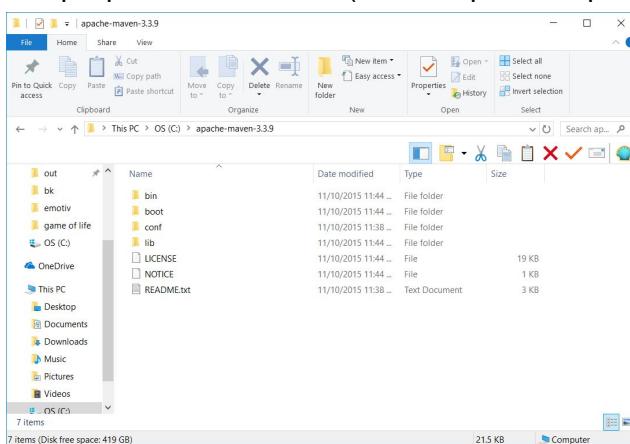
Files

Maven is distributed in several formats for your convenience. Simply pick a ready-made binary distribution archive and follow the [Installation instructions](#). Use a source archive if you intend to build Maven yourself.

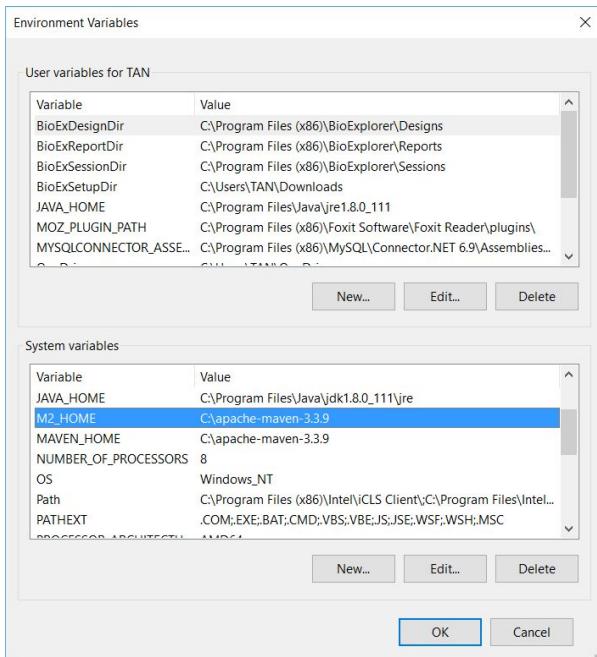
In order to guard against corrupted downloads/installations, it is highly recommended to [verify the signature](#) of the release bundles against the public [KEYS](#) used by the Apache Maven developers.

Link	Checksum	Signature
Binary tar.gz archive	apache-maven-3.3.9-bin.tar.gz	apache-maven-3.3.9-bin.tar.gz.asc
Binary zip archive	apache-maven-3.3.9-bin.zip	apache-maven-3.3.9-bin.zip.asc
Source tar.gz archive	apache-maven-3.3.9-src.tar.gz	apache-maven-3.3.9-src.tar.gz.asc
Source zip archive	apache-maven-3.3.9-src.zip	apache-maven-3.3.9-src.zip.asc

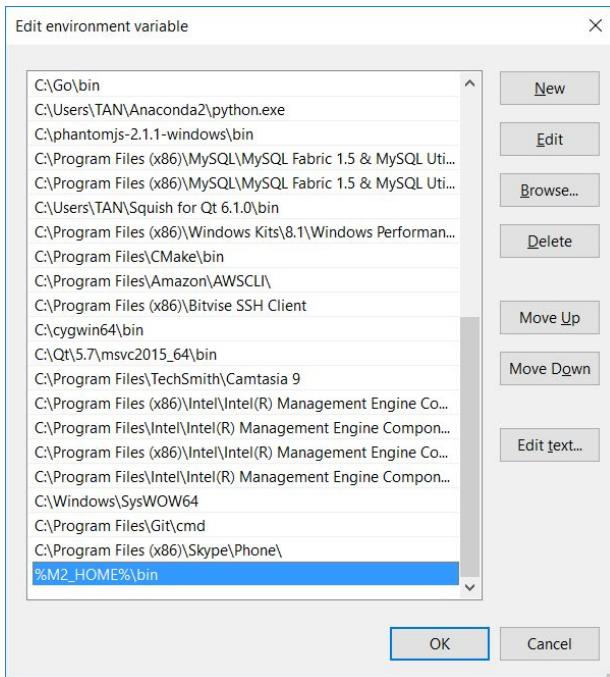
- Unzip zip file to one folder (for example I unzip to C:\apache-maven-3.3.9)



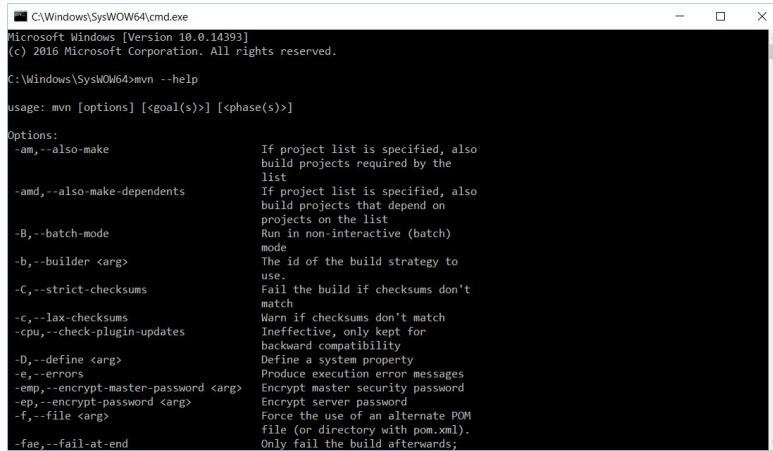
- Add M2_HOME and MAVEN_HOME variable point to above folder



- Add mavin bin to Path variable so you can call maven from any place



That it, to check maven installed successfully, open cmd and typing in ***mvn --help***
Help content for mvn should show up correctly



```
C:\Windows\SysWOW64\cmd.exe
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\SysWOW64\mvn --help

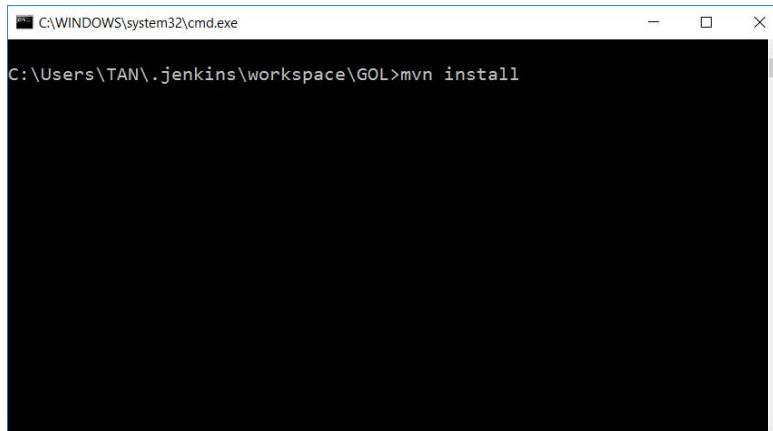
usage: mvn [options] [<goal(s)>] [<phase(s)>]

Options:
  -am,--also-make           If project list is specified, also
                           build projects required by the
                           list
  -amd,--also-make-dependents If project list is specified, also
                           build projects that depend on
                           projects on the list
  -B,--batch-mode           Run in non-interactive (batch)
                           mode
  -b,--builder <arg>        The id of the build strategy to
                           use.
  -c,--strict-checksums    Fail the build if checksums don't
                           match
  -c,--lax-checksums       Warn if checksums don't match
  -cpu,--check-plugin-updates Ineffective, only kept for
                           backward compatibility
  -D,--define <arg>        Define a system property
  -e,--errors               Produce execution error messages
  -emp,--encrypt-master-password <arg> Encrypt master security password
  -ep,--encrypt-password <arg> Encrypt server password
  -f,--file <arg>          Force the use of an alternate POM
                           file (or directory with pom.xml).
  -fae,--fail-at-end       Only fail the build afterwards;
```

Run Maven Build Manually

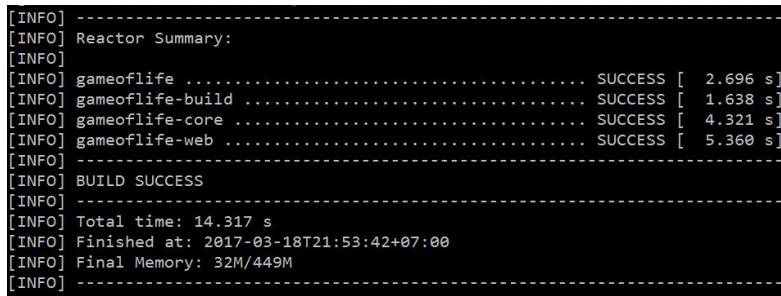
Now maven is ready, we can try to run the build for source code which we just clone from Github repository at **source code management** step.

Open **cmd** and change directory to repository GOL folder. And then typing in **mvn install**. This command will run java source unit test and then build this web application to a .war file



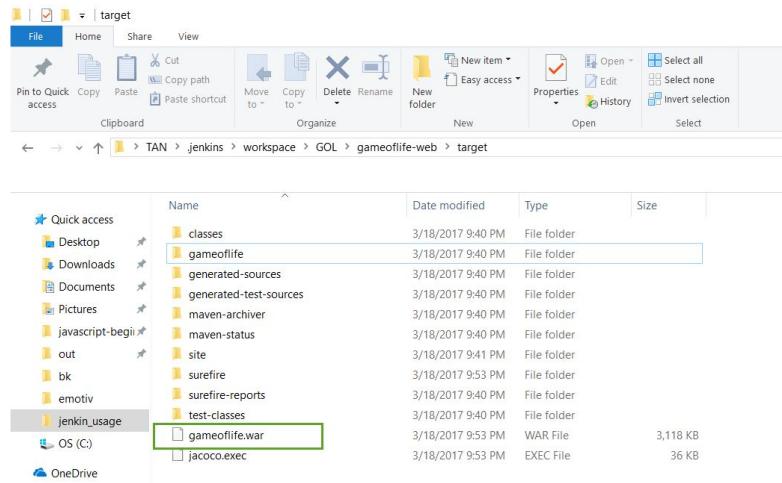
```
C:\WINDOWS\system32\cmd.exe
C:\Users\TAN\.jenkins\workspace\GOL>mvn install
```

You will build result from command prompt



```
[INFO] -----
[INFO] Reactor Summary:
[INFO] 
[INFO] gameoflife ..... SUCCESS [ 2.696 s]
[INFO] gameoflife-build .. SUCCESS [ 1.638 s]
[INFO] gameoflife-core .... SUCCESS [ 4.321 s]
[INFO] gameoflife-web ..... SUCCESS [ 5.360 s]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 14.317 s
[INFO] Finished at: 2017-03-18T21:53:42+07:00
[INFO] Final Memory: 32M/449M
[INFO] -----
```

And can see actual .war file inside target folder of **gameoflife-web**. This folder could be used to deploy with web application server.



Install Tomcat Server as Service

To install Tomcat 7 server, you just follow these steps:

- Download Tomcat 7 from this page <http://tomcat.apache.org/download-70.cgi>, note that we will download version [Windows Service Installer](#) so Tomcat will be installed as window service

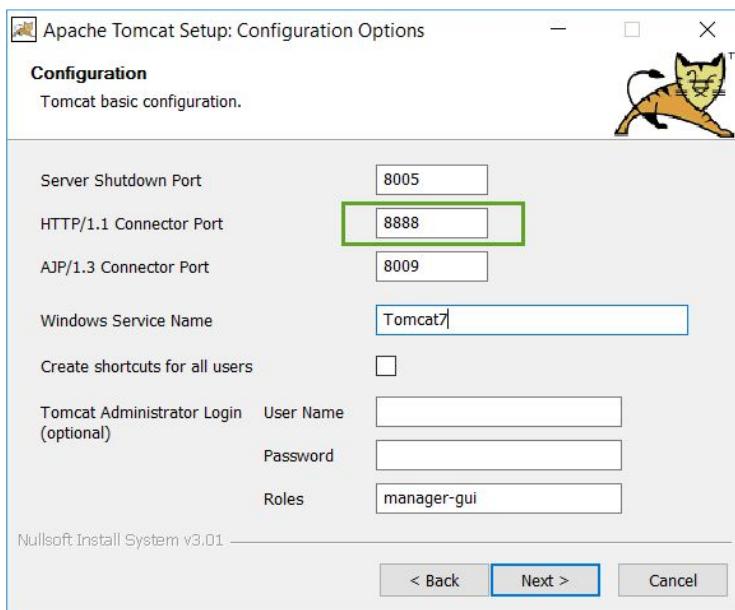
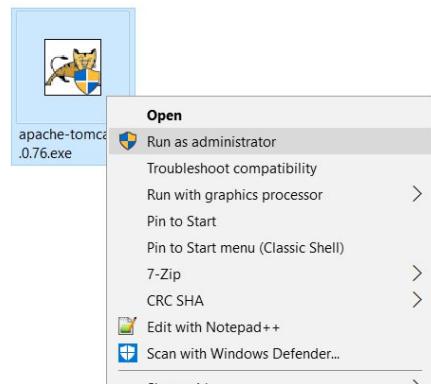
7.0.76

Please see the [README](#) file for packaging information. It explains what every distribution contains.

Binary Distributions

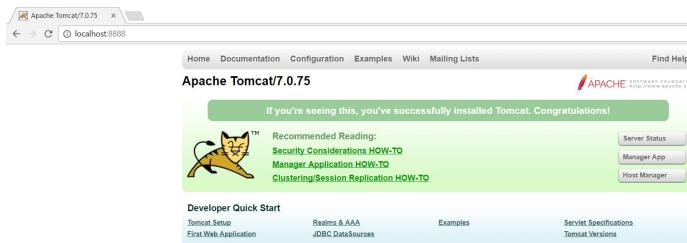
- Core:
 - zip (pgp, md5, sha1)
 - tar.gz (pgp, md5, sha1)
 - 32-bit Windows zip (pgp, md5, sha1)
 - 64-bit Windows zip (pgp, md5, sha1)
 - 32-bit/64-bit Windows Service Installer (pgp, md5, sha1)
- Full documentation:
 - tar.gz (pgp, md5, sha1)
- Deployer:
 - zip (pgp, md5, sha1)
 - tar.gz (pgp, md5, sha1)
- Extras:
 - JMX Remote Jar (pgp, md5, sha1)
 - Web services jar (pgp, md5, sha1)
 - JULI adapters jar (pgp, md5, sha1)
 - JULI log4j jar (pgp, md5, sha1)
- Embedded:
 - tar.gz (pgp, md5, sha1)
 - zip (pgp, md5, sha1)

- Run installer and select port 8888 for running Tomcat (We already use 8080 for Jenkins installation)



- After install successfully, a Tomcat service will running and you can access Tomcat server from localhost

Services (Local)					
	Name	Status	Startup Type	Log On As	
Stop the service	Apache Tomcat 7.0 Tomcat7	Running	Automatic	Local Syst...	
Restart the service	Time Broker	Running	Manual (Trigger Start)	Local Servi...	
Description:	Tilt Data model server	Running	Automatic	Local Syst...	
Apache Tomcat 7.0.75 Server - http://tomcat.apache.org/	Themes	Running	Automatic	Local Syst...	
	Remote Desktop Services	Running	Manual	Network S...	
	Telephone	Running	Manual	Network S...	
	System Events Broker	Running	Automatic (Trigger Start)	Local Syst...	
	Surround	Running	Automatic	Local Syst...	



- To control Tomcat from Jenkins, one user with role "manager-script" should be added to file **conf/tomcat-users.xml**. Then restart Tomcat server by service.

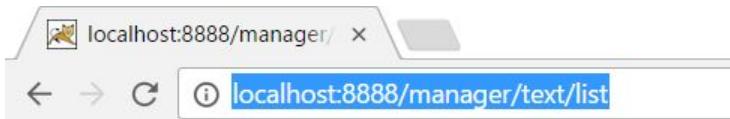
```

tomcat-users.xml
1 <?xml version='1.0' encoding='cp1252'?>
2 <!--
3 Licensed to the Apache Software Foundation (ASF) under one or more
4 contributor license agreements. See the NOTICE file distributed with
5 this work for additional information regarding copyright ownership.
6 The ASF licenses this file to You under the Apache License, Version 2.0
7 (the "License"); you may not use this file except in compliance with
8 the License. You may obtain a copy of the License at
9
10 http://www.apache.org/licenses/LICENSE-2.0
11
12 Unless required by applicable law or agreed to in writing, software
13 distributed under the License is distributed on an "AS IS" BASIS,
14 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
15 See the License for the specific language governing permissions and
16 limitations under the License.
17 -->
18
19 <tomcat-users>
20   <role rolename="manager-script"/>
21   <user username="admin" password="admin" roles="manager-script"/>
22 </tomcat-users>

```

Services (Local)					
	Name	Status	Startup Type	Log On As	
Apache Tomcat 7.0 Tomcat7	Apache Tomcat 7.0 Tomcat7	Start	Running	Automatic	Local Syst...
Stop the service	Time Broker	Running	Manual (Trigger Start)	Local Servi...	
Restart the service	Tile Data model server	Running	Automatic	Local Syst...	
Description:	Themes	Pause	Automatic	Local Syst...	
Apache Tomcat 7.0.75 Server -	Remote Desktop Services	Running	Automatic	Network S...	
http://tomcat.apache.org/	Telephony	Resume	Manual	Network S...	
	System Events Broker	Restart	Manual	Network S...	
	Superfetch	All Tasks	Running	Automatic (Trigger Start)	Local Syst...
	Dell SupportAssist Agent	Refresh	Running	Automatic	Local Syst...
	Storage Service	Properties	Running	Automatic (Delayed Start)	Local Syst...
	Windows Image Acquisition	Help	Running	Manual (Trigger Start)	Local Servi...
	State Repository Service		Running	Automatic	Local Syst...
	Secure Socket Tunneling Protocol Service		Running	Manual	Local Servi...
	SSDP Discovery		Running	Manual	Local Servi...

To check if admin user working, access to <http://localhost:8888/manager/text/list> and then put on user / password you just specify



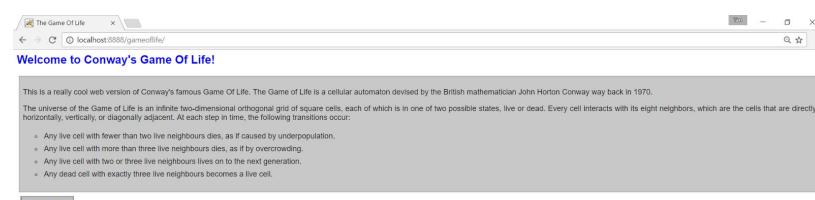
```

OK - Listed applications for virtual host localhost
/:running:0:ROOT
/gameoflife:running:0:gameoflife
/manager:running:0:manager
/docs:running:0:docs

```

Deploy GOL to Tomcat Server

Deploy this application to Tomcat server, and you will see some the app from browser and actually start a new game !.





Continuous Integration with Jenkins

This chapter will show you how to automate software build with Jenkins.

Jenkins in the Big Picture of CI, CD and DevOps

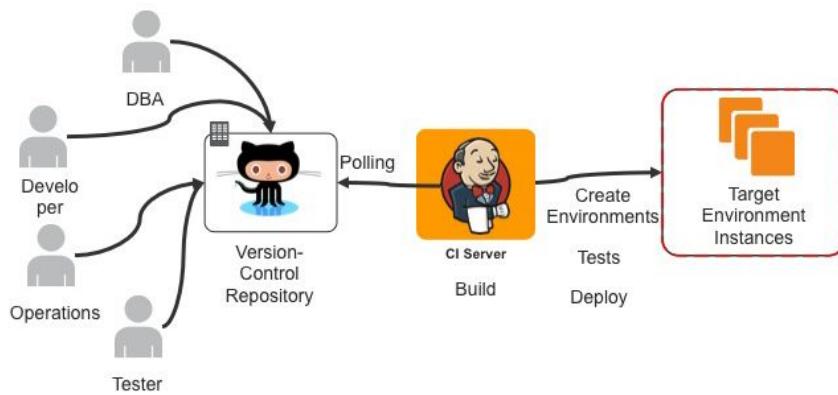
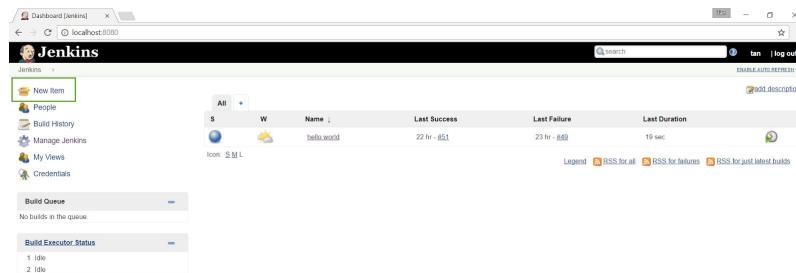


Image above show quite clear about Jenkins role in software development process.

- Developers do coding and commit their code to a version control repository like Github
- Jenkins actively check Github to see if there are any changes; in case there are changes, Jenkins will pull the latest code from Github to build machine (the machine which runs Jenkins job)
- Jenkins builds software artifacts from pulled source code (for example war file from .java source file)
- Jenkins runs and shows up unit test reports
- Jenkins deploys artifacts to target environments (for example a web server like Tomcat, JBoss, WebSphere...)

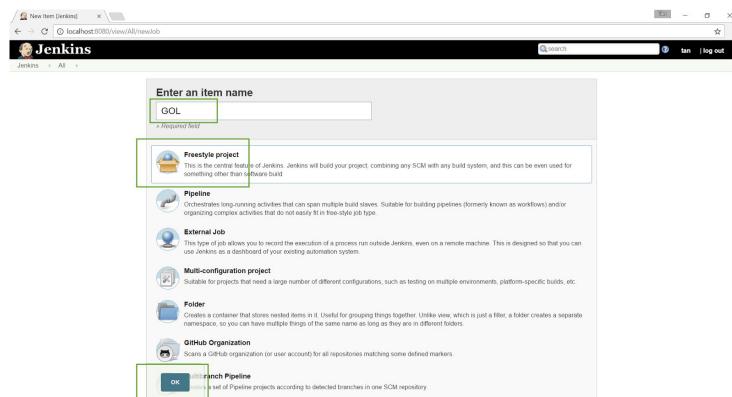
Create GOL Job

From home page, click to **New Item** link



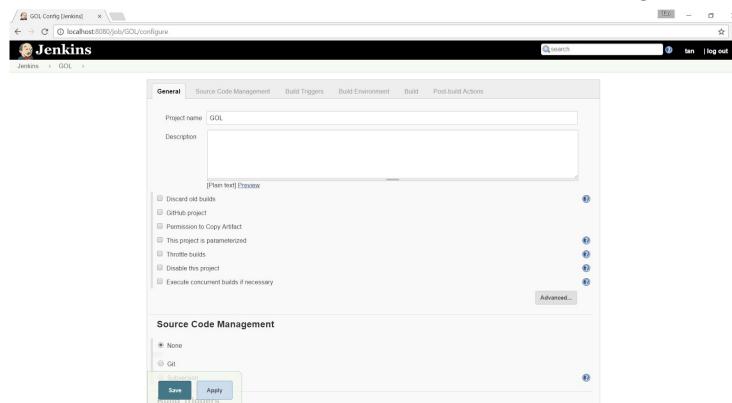
The screenshot shows the Jenkins dashboard with the 'New Item' link highlighted in green. The dashboard includes sections for Jenkins, Build Queue, and Build Executor Status.

Enter **GOL** for project name and then chose **Freestyle project** and finally click to **OK** button to create a new job with name is **GOL**



The screenshot shows the 'Enter an item name' dialog box with 'GOL' entered. Below it, the 'Freestyle project' option is selected, and the 'OK' button is highlighted.

Then a new screen will show up allow us to configure everything for **GOL** job.

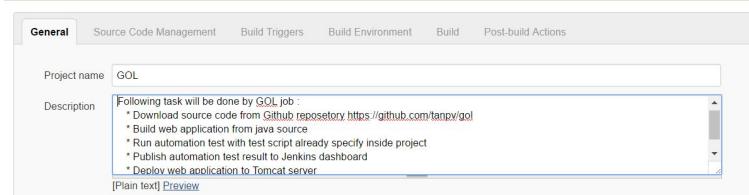


The screenshot shows the 'Configure Job' screen for the 'GOL' job. The 'General' tab is selected, showing fields for 'Project name' (set to 'GOL') and 'Description' (containing '[Plan test] Deploy'). Other options like 'Discard old builds', 'GitHub project', and 'Source Code Management' (set to 'Git') are also visible.

Description is place to describe what mission of this job and steps inside the job. Put follow text into **Description** session.

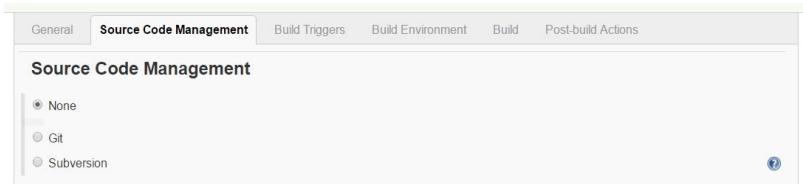
" Following task will be done by GOL job :

- Check change on Github repo and automatically download source code from Github repository <https://github.com/tanpv/gol>
- Build web application from java source
- Run automation test with test script already specify inside project
- Publish automation test result to Jenkins dashboard
- Deploy web application to Tomcat server ”



Source Code Management

Purpose of this part is answer for question **Where to get source code ?** for our software build. From job configure, click into **Source Code Management** tab



Check if git is installed in local machine

In order to clone source code from Github, local machine should install Git.

To check if local machine already install **git** or not, open command prompt and typing
git --help

```

C:\Windows\SysWOW64\cmd.exe
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\SysWOW64>git --help
usage: git [--version] [--help] [-C <path>] [-c name=value]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
  clone      Clone a repository into a new directory
  init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
  add        Add file contents to the index
  mv        Move or rename a file, a directory, or a symlink
  reset     Reset current HEAD to the specified state
  rm        Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
  bisect    Use binary search to find the commit that introduced a bug
  grep      Print lines matching a pattern
  log       Show commit logs
  show      Show various types of objects
  status    Show the working tree status

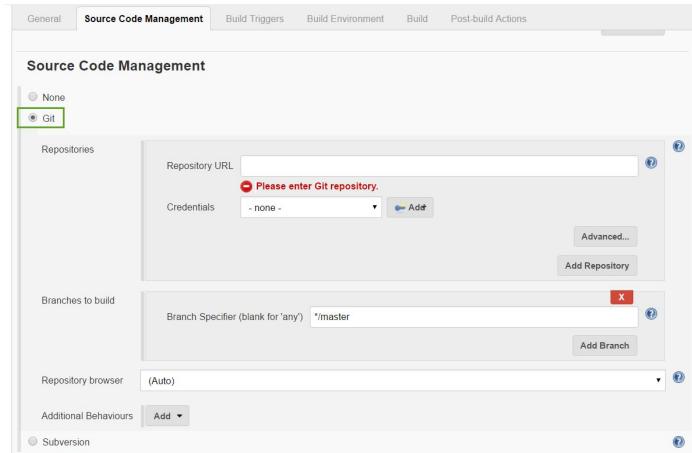
grow, mark and tweak your common history

```

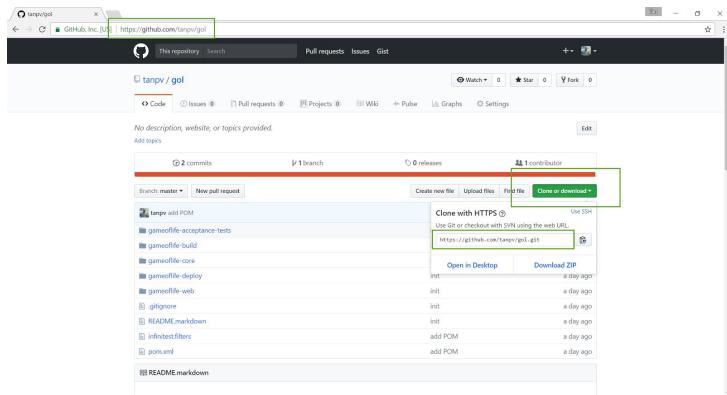
To download and install git, please refer to this link <https://git-scm.com/downloads>

Specify source code repository

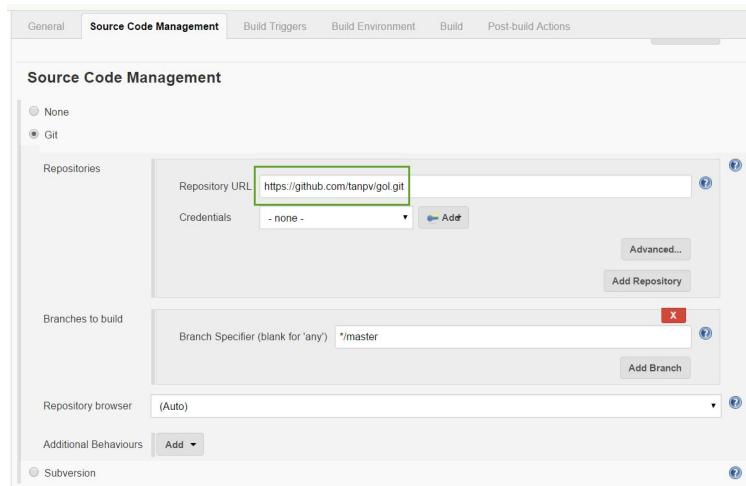
Because we host project source code on Github, so you will choose **Git** as below.



To get repo link, go to Github <https://github.com/tanpv/gol> then click to **Clone or download** button, you will see the repo link. Copy this link.



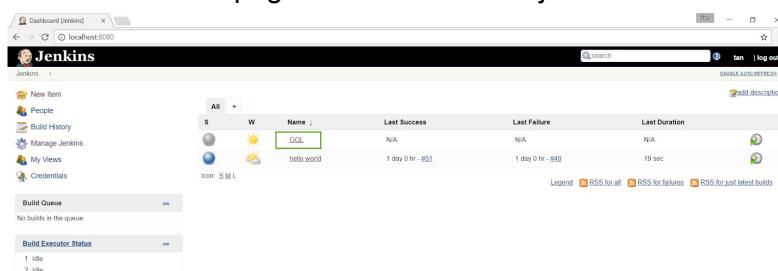
Put the repo link <https://github.com/tanpv/gol.git> in to **Repository URL** section. The branch will be default as ***/master**



Click to **Save** button and that it, we already tell to Jenkins where to get source code.

Check if setting working fine

This is the time to try for first running and see if source code from Github is clone or not. Go back to home page then click to GOL job



The dashboard for GOL will show up, you will see describe which we just added before

The screenshot shows the Jenkins Project GOL dashboard. On the left, there's a sidebar with links: Back to Dashboard, Status, Changes, Workspace, Build Now (which is highlighted), Delete Project, Configure, and Move. Below the sidebar is a 'Build History' section with a search bar and RSS links. To the right of the sidebar are 'Workspace' and 'Recent Changes' links. At the bottom right is a 'Permalinks' section with RSS links.

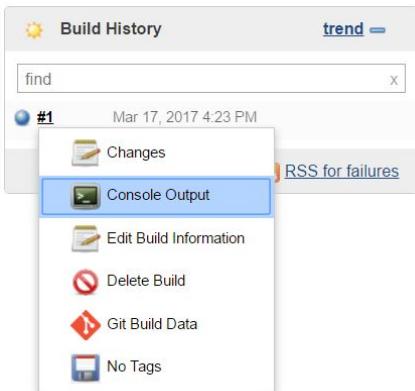
Click in to **Build Now** to activate job run immediately.

The screenshot shows the Jenkins Project GOL dashboard after clicking 'Build Now'. The 'Build Now' button is now highlighted in green. The rest of the interface remains the same as the previous screenshot, with the sidebar and build history section visible.

You will see GOL start running as expected

The screenshot shows the Jenkins Build History page. It displays a single build entry: '#1 Mar 17, 2017 4:23 PM'. The progress bar for this build is partially filled with blue, indicating it is currently running. Below the build list are RSS feed links.

To see how job running, select **Console Output** from context menu while you click to build number



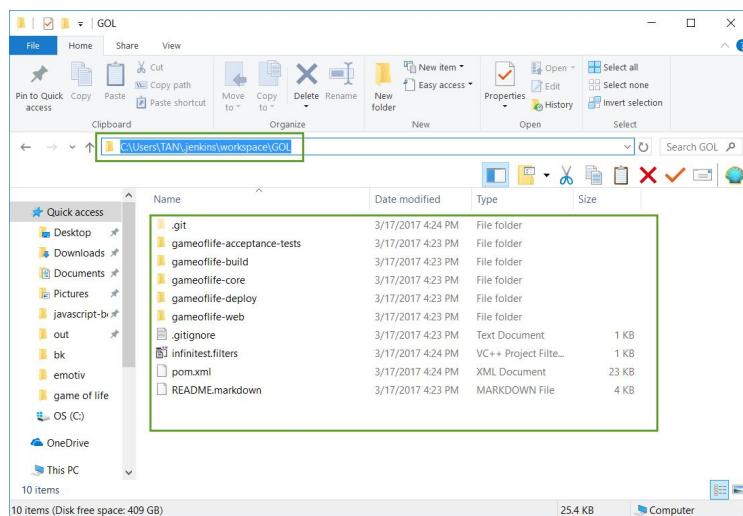
Then the job log will show up every thing happen in order to clone source code from Github repository. You can see that the build already success running.

```

Started by user tan
Building in workspace C:\Users\TAN\jenkins\workspace\GOL
Cloning the remote Git repository
Cloning repository https://github.com/tanpu/gol.git
> git.exe init C:/Users/TAN/jenkins/workspace/GOL # timeout=10
Fetching upstream changes from https://github.com/tanpu/gol.git
> git.exe fetch -tags +refs/heads/*:refs/remotes/origin/*
> git.exe config remote.origin.url https://github.com/tanpu/gol.git # timeout=10
> git.exe config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe config remote.origin.fetch https://github.com/tanpu/gol.git # timeout=10
Fetching upstream changes from https://github.com/tanpu/gol.git
> git.exe fetch --tags +refs/heads/*:refs/remotes/origin/*
> git.exe rebase -f refs/remotes/origin/master # timeout=10
> git.exe branch "refs/remotes/origin/master" "comitt" # timeout=10
Checking out Revision 38b0942a6ec3857632fd392fc0527f0526f06c25 (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f 38b0942a6ec3857632fd392fc0527f0526f06c25
First time build. Skipping changelog.
Finished: SUCCESS

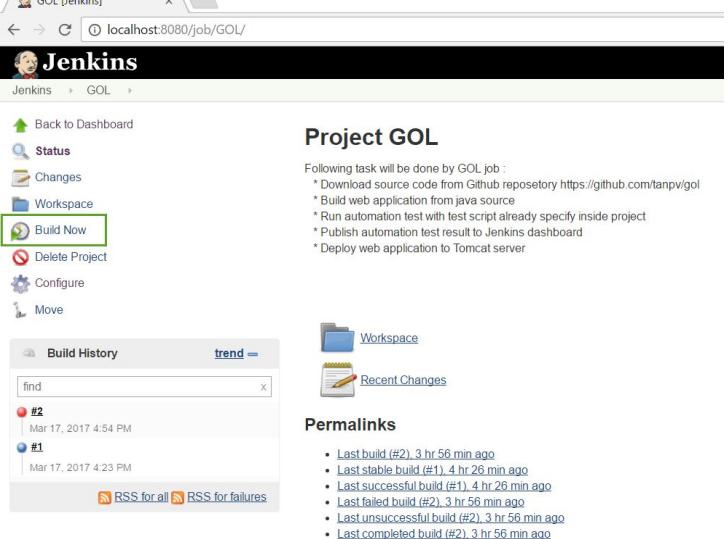
```

Now we could go to local repository to see the source code already downloaded. Each job will have it's own folder inside **workspace** folder



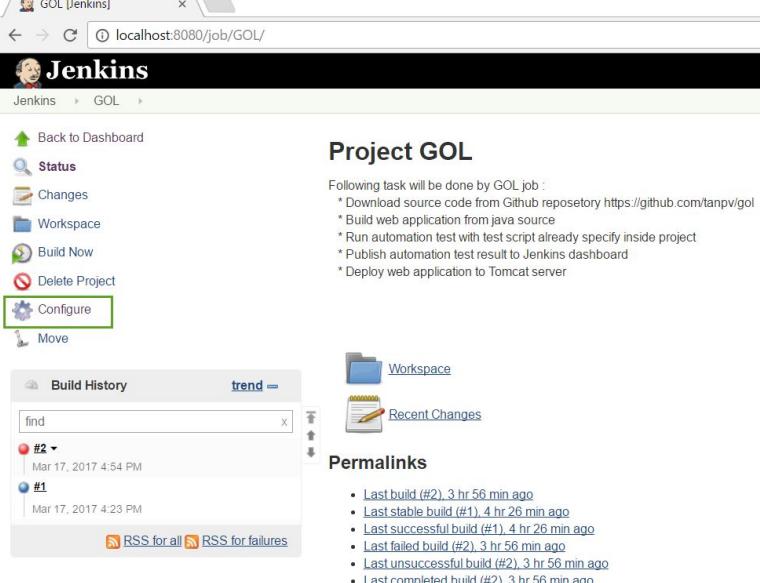
That it, we complete the first step of this job.

Build Triggers



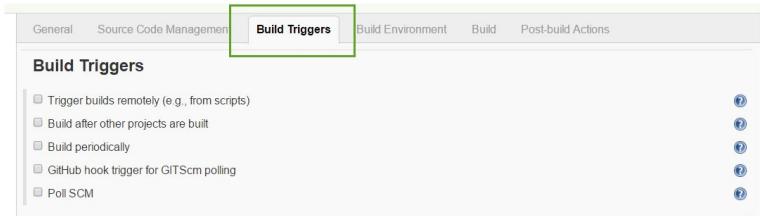
The screenshot shows the Jenkins Project GOL dashboard. On the left, there's a sidebar with links: Back to Dashboard, Status, Changes, Workspace, Build Now (which is highlighted with a green border), Delete Project, Configure (which is also highlighted with a green border), and Move. Below the sidebar is a 'Build History' section with two entries: #2 (Mar 17, 2017 4:54 PM) and #1 (Mar 17, 2017 4:23 PM). At the bottom of this section are 'RSS for all' and 'RSS for failures'. To the right of the sidebar is the main content area titled 'Project GOL'. It lists the tasks performed by the GOL job, including downloading source code from GitHub, building a web application from Java source, running automation tests, publishing results to the Jenkins dashboard, and deploying to Tomcat. Below this is a 'Recent Changes' section with a link to 'Workspace'. At the bottom of the main content area is a 'Permalinks' section with a list of build links.

The first way and most simple way to start LOG job is just click in to **Build Now** button. But this way is manually and not so cool. We want to trigger Jenkins job running automatically, so to do this, click to **Configure** from LOG dashboard.

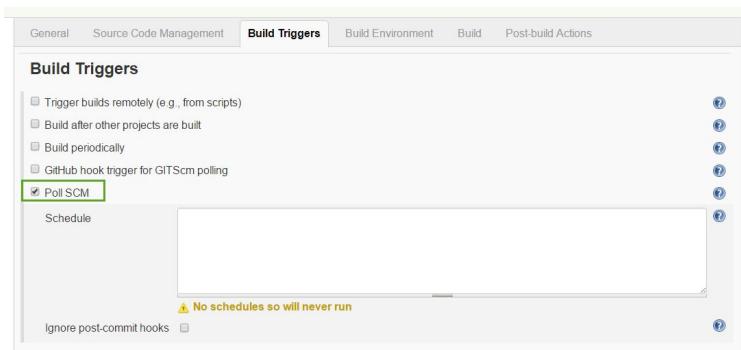


This screenshot is identical to the one above, showing the Jenkins Project GOL dashboard. The 'Configure' button in the sidebar is highlighted with a green border. The rest of the interface, including the build history, task list, recent changes, and permalinks, is the same as the previous screenshot.

Then select to **Build Triggers** tab you will some option to configure so job could trigger automatically



Have 5 ways to automate trigger Jenkins job as show above. In this book I will focus on **Poll SCM**, the good way in configure for a continuous system. Now you click on **Poll SCM**, you will see a text box show up for setting schedule



So following are steps show up how **Poll SCM** work :

- Base on schedule setting, Jenkins will actively check Github repository to see if have any change from repository
- If have any change on Github repo Jenkins job will trigger

Entry	Description	Equivalent To
@yearly (or @annually)	Run once a year at midnight in the morning of January 1	0 0 1 1 *
@monthly	Run once a month at midnight in the morning of the first of the month	0 0 1 * *
@weekly	Run once a week at midnight in the morning of Sunday	0 0 * * 0
@daily	Run once a day at midnight	0 0 * * *
@hourly	Run once an hour at the beginning of the hour	0 * * * *
@reboot	Run at startup	@reboot

* * * * * command to be executed

┌───────────┐
 | * * * * * |
 | └── day of week (0 - 7) (0 or 7 are Sunday, or use names)
 | ┌── month (1 - 12)
 | | ┌── day of month (1 - 31)
 | | ┌── hour (0 - 23)
 | | ┌── min (0 - 59)

Schedule with **Poll SCM** work follow cron schedule rule, above image show some cron schedule example. Basically, we want to catch the change on Github repository as soon as possible, so follow text will be add to **Schedule** * * * * *. This mean we want to check change on Github every minute !

General Source Code Management **Build Triggers** Build Environment Build Post-build Actions

Build Triggers

- Trigger builds remotely (e.g., from scripts)
- Build after other projects are built
- Build periodically
- GitHub hook trigger for GITScm polling
- Poll SCM

Schedule: `*****`

Do you really mean "every minute" when you say *** * * *? Perhaps you meant "H * * * * to poll"

Would last have run at Friday, March 17, 2017 9:41:54 PM ICT, would next run at Friday, March 17, 2017 9:41:54 PM ICT.

Ignore post-commit hooks

Click to **Save** and Jenkins will move you back to job dashboard



Now is the time to check if this setting really work ?

I will go to my GOL project repo on Github and change content of file **README.markdown** right from web browser (Github support commit change right from web browser)

This repository Search

tampv/gol

Code Issues Pull requests Projects Wiki Graph Settings

No description, website, or topics provided. Add topics

2 commits 1 branch 0 releases 1 contributor

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

Latest commit: 38b4a42 a day ago

File	Commit	Author	Date
src/main/java/com/tampv/gameoflife/GameOfLife.java	init	a day ago	
src/main/java/com/tampv/gameoflife/acceptance-tests/GameOfLifeAcceptanceTests.java	init	a day ago	
src/main/java/com/tampv/gameoflife/build/GameOfLifeBuild.java	init	a day ago	
src/main/java/com/tampv/gameoflife/core/GameOfLifeCore.java	init	a day ago	
src/main/java/com/tampv/gameoflife/deploy/GameOfLifeDeploy.java	init	a day ago	
src/main/java/com/tampv/gameoflife/web/GameOfLifeWeb.java	init	a day ago	
.gitignore	init	a day ago	
README.markdown	init	a day ago	
src/main/resources/filtering/filtering-filters.xml	add POM	a day ago	
pom.xml	add POM	a day ago	

<https://github.com/tampv/gol/blob/master/README.markdown>

This is a simple demonstration application used in the Jenkins: The Definitive Guide book.

Building the project

The project is a simple multi-module Maven project. To build the whole project, just run `mvn install` from the root directory.

Running the game

The application is a very simple online version of Conway's 'game of life'. To see what the game does, run `mvn install` as described above, then go to the `gameoflife-web` directory and run `mvn jetty:run`. The application will be running on `http://localhost:9090`.

Running the acceptance tests

The acceptance tests are written using `Mochajs` and `Thucydides`. They are designed to run against a remote server. Run the tests using `mvn test` during the integration and acceptance testing stages of a web site release.

32
33
34 Test change

Commit changes

Update README.md

Test change

Commit directly to the `master` branch.
 Create a new branch for this commit and start a pull request. Learn more about pull requests.

Commit changes **Cancel**

Now come back to job dash board, wait for about 1 minute, you will see new job is planning and run.

Project GOL

Following task will be done by GOL job :

- * Download source code from Github repository <https://github.com/tanpv/gol>
- * Build web application from java source
- * Run automation test with test script already specify inside project
- * Publish automation test result to Jenkins dashboard
- * Deploy web application to Tomcat server

Workspace

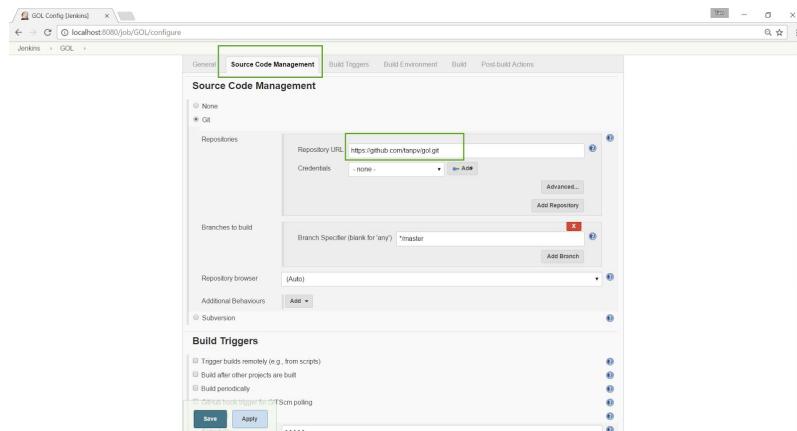
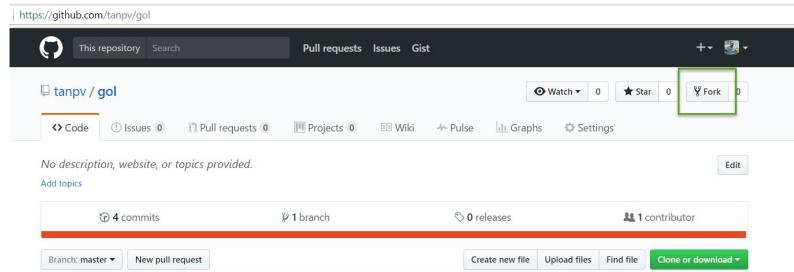
Recent Changes

Permalinks

- [Last build \(#3\), 6 hr 1 sec ago](#)
- [Last stable build \(#1\), 5 hr 43 min ago](#)
- [Last successful build \(#1\), 5 hr 43 min ago](#)
- [Last failed build \(#2\), 5 hr 13 min ago](#)
- [Last unsuccessful build \(#2\), 5 hr 13 min ago](#)
- [Last completed build \(#2\), 5 hr 13 min ago](#)

[RSS for all](#) [RSS for failures](#)

That it, it work !!!, to practice by yourself, just clone GOL project to your account by **Fork** button and then change the GOL configuration point to your github account as show below.



Build

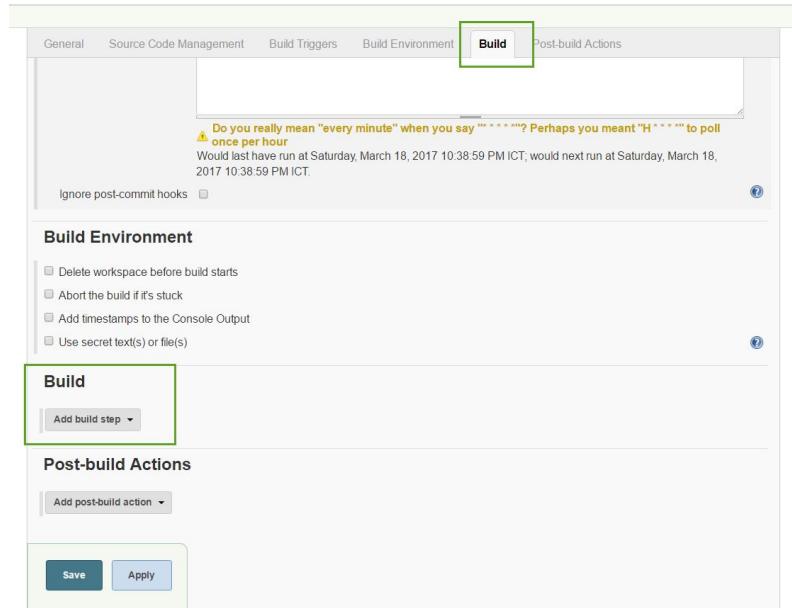
In this section, We will setting to build java web app with maven.

Configure Build Step

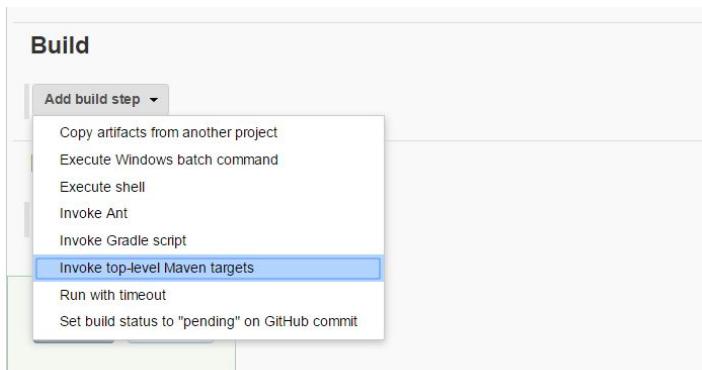
From GOL job dashboard, click to **Configure** link, the configure page will show up

A screenshot of a Jenkins project dashboard for 'Project GOL'. On the left, there's a sidebar with links for 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Delete Project', 'Configure' (which is highlighted with a green box), 'Git Polling Log', and 'Move'. The main content area is titled 'Project GOL' and contains a list of tasks: 'Download source code from Github repository https://github.com/tanpv/gol', 'Build web application from java source', 'Run automation test with test script already specify inside project', 'Publish automation test result to Jenkins dashboard', and 'Deploy web application to Tomcat server'. Below this, there are sections for 'Workspace' (with a blue folder icon) and 'Recent Changes' (with a notebook icon). At the bottom, there's a 'Permalinks' section with a list of build links and an 'RSS for all' and 'RSS for failures' button.

From configure page, select **Build** tab, page will scroll down to **Add build step**



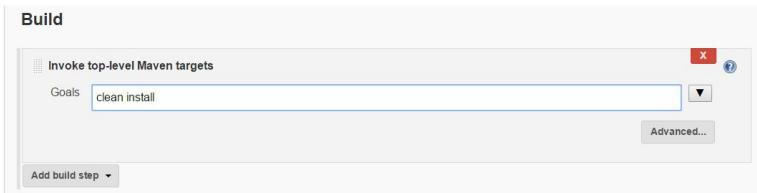
Click to **Add build step** and select **Invoke top-level Maven targets**



Section to adding maven command will show up as below



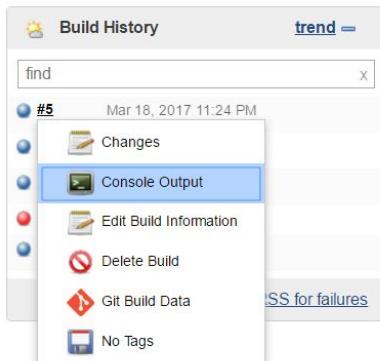
Put **Goals** maven command **clean install**



Then finally click in to **Save** button. That it we already complete configure for build maven step.

Check Log and Workspace

Now from dashboard of GOL, just click to **Build Now** to see how Jenkins job work by open console log



Scroll this log to bottom you will see build job run successfully

```
Results :
Tests run: 0, Failures: 0, Errors: 0, Skipped: 0

[INFO] ... jacoco-maven-plugin:0.7.2.201409121642:report (jacoco-site) @ gameoflife-web ...
[INFO] Analyzed bundle 'gameoflife-web' with 2 classes
[INFO]
[INFO] --- maven-thucydides-plugin:0.9.268:aggregate (thucydides-reports) @ gameoflife-web ...
[INFO] log4j:WARN No appenders could be found for logger (org.jboss.logging).
[INFO] log4j:WARN Please initialize the log4j system
[INFO] Reading requirements from net.thucydides.core.requirements.FileSystemRequirementTagProvider@22d4adda7
[INFO] Reading requirements from net.thucydides.core.requirements.PackageRequirementBasedTagProvider@75b6d41
[INFO] Generating requirements for: []
[INFO] ...
[INFO] --- maven-integration-plugin:2.4:install (default-install) @ gameoflife-web ...
[INFO] Installing C:\Users\TAN\jenkins\workspace\GOL\gameoflife-web\target\gameoflife.war to C:\Windows\system32\config\systemprofile\.m2\repository\com\valasco\gameoflife\gameoflife-web\1.0-SNAPSHOT\gameoflife-web-1.0-SNAPSHOT.war
[INFO] Installing C:\Users\TAN\jenkins\workspace\GOL\gameoflife-web\pom.xml to C:\Windows\system32\config\systemprofile\.m2\repository\com\valasco\gameoflife\gameoflife-web\1.0-SNAPSHOT\gameoflife-web-1.0-SNAPSHOT.pom
[INFO] -----
[INFO] Reactor Summary:
[INFO] gameoflife ..... SUCCESS [ 10.519 s]
[INFO] gameoflife-build ..... SUCCESS [ 4.644 s]
[INFO] gameoflife-core ..... SUCCESS [ 7.861 s]
[INFO] gameoflife-web ..... SUCCESS [ 14.009 s]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 38.349 s
[INFO] Final Memory: 38M/249M
[INFO] -----
Finished: SUCCESS
```

Page generated Mar 18, 2017 11:31:19 PM

From GOL home page, click to **Workspace**



Jenkins

Jenkins > GOL >

 Back to Dashboard

 Status

 Changes

 Workspace

 Build Now

 Delete Project

 Configure

 Git Polling Log

 Move

You will access directory where build happen and actually could see the war file inside **Gameoflife-web/target/**. You could actually download war file from here.



Jenkins

Jenkins > GOL >

 Back to Dashboard

 Status

 Changes

 Workspace

 Build Now

 Delete Project

 Configure

 Git Polling Log

 Move

 Build History 

 find X

Workspace of GOL on Windows_Agent

 gameoflife-web / target /

- classes
- gameoflife
- generated-sources/annotations
- generated-test-sources/test-annotations
- maven-archiver
- maven-status/maven-compiler-plugin
- site
- surefire
- surefire-reports
- test-classes/com/wakaleo/gameoflife/webtests/controllers

 gameoflife.war 3.04 MB [view](#)

 jacoco.exec 17.93 KB [view](#)

 (all files in zip)

That it, We already finish setting up build step.