**: Jenkins winsows does not launch the browser**

**: Jenkins.war file will launch the browser**

**Maven and Jenkin :**

-Step 1: Open Eclipse IDE  
-Step 2: Download and Install Maven

-Go to ‘Help‘ and click on ‘Install New Software‘ . OR

\_Go to Eclipse Marketplace - search 'Maven' - install 'Maven Integration for Eclipse'.

-Click on ‘Add‘ button.

-Enter the ‘Name‘ as per your wish. Here I am adding as ‘Maven’. Add location as “http://download.eclipse.org/technology/m2e/releases/” and click on ‘OK‘ button .

-In the Available Software list, you could see **Maven** option. Select “Maven” and click on ‘Next‘ button.

-Click ‘I accept the terms of the license agreement‘ then click Finish.

-To create a project – Right click on Package Explorer and navigate through New – Other .

-Select Maven Project and click on Next .

-Select ‘Create a simple project‘ check box and click Next .

-Fill the Group Id, Artifact Id and click the Finish button .

-In the maven project, we use POM file to add the required dependencies. Double click on pom.xml file and click on pom.xml tab.

-Create a TestNG class and generate testng.xml file. Check the below link to do this step.

package tests;

public class NewTest {

public WebDriver driver;

@Test

public void openMyBlog() {

driver.get("https://www.softwaretestingmaterial.com/");

}

@BeforeClass

public void beforeClass() {

System.setProperty("webdriver.gecko.driver", "D:\\Selenium\\Drivers\\geckodriver.exe");

driver = new FirefoxDriver();

}

@AfterClass

public void afterClass() {

driver.quit();

}

}

testng.xml: testng.xml file looks like below. Copy and place it in your testng.xml.

<?xml version="1.0" encoding="UTF-8"?>

<suite name="Suite" parallel="false">

<test name="Test">

<classes>

<class name="tests.NewTest"/>

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

-To run the project using testng.xml, right click on ‘testng.xml’ file and go to Run As TestNG Suite

To run the project using pom.xml, we need to add some more entries in the pom.xml file. We need to add ‘maven-compiler-plugin’ and ‘maven-surefire-plugin’

pom.xml

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>mavenPackage</groupId>

<artifactId>MavenProject</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<!-- Selenium -->

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-java</artifactId>

<version>2.53.1</version>

</dependency>

<!-- TestNG -->

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>6.8</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.6.1</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.20</version>

<configuration>

<suiteXmlFiles>

<suiteXmlFile>testng.xml</suiteXmlFile>

<!-- <suiteXmlFile>src/main/resources/testng.xml</suiteXmlFile> -->

</suiteXmlFiles>

</configuration>

</plugin>

</plugins>

</build>

</project>

=> To add dependency go to POM > dependecies > add > typeName > select version > ok.

=> right click on project > Maven > add dependency > typeName > select version > ok.

How to run pom.xml file ?

-To run the project using pom.xml, right click on ‘pom.xml‘ file and go to ‘Run As‘ ‘Maven test‘

How to run pom.xml file from cmd(command line) ?

open cmd

- type [cd project-path] (right click on project > properties > copy path)

- type [mvn validate] it will verify the project is correct.

- type [mvn clean] it will clean previous execution record/history.

- type [mvn compile] it will compile all the source code

- type [mvn test] it will run all the tests.

**=============================== jenkins =============================**

**Jenkins Credentials:**

**62e68cf79d334b94a3cdcc71d980dc70**

**This may also be found at: C:\Users\jawad\.jenkins\secrets\initialAdminPassword**

**How to open Jenkins :**

**localhost:8080 or ipAddress:8080**

**To get ipAddress in cmd - ipconfig /all** then **enter.**

**: open jenkins using different port number-**

**cmd > folder-path > java –jar jenkins.war - -httpPort=9090 >enter**

**cmd > folder-path > java -Dhudson.model.DirectoryBrowserSupport.CSP="" –jar C:\Users\jawad\Downloads\jenkinsFile\jenkins.war --httpPort=9191 (for html Extent Report view)**

**How to clean/delete previous jenkins from the system**

**Go to - C:/user/userName/.jenkins file (delete the file**

**Note: if you have to save any file- copy from .jenkins file & paste into any folder then delete .jenkins file.**

**Check Jenkins :**

**New Ietem – do project name - write some description –**

**add build step windows batch command /in the box type( java –version),**

**add build step windows batch command/in the box type( mvn –version) –**

**apply/save. go dashboard – select project – build now..**

**Configuration:**

Dashboard - Manage Jenkins – Global Tool Configuration –

Add Jdk - set name JAVA\_HOME – paste the path of JAVA\_home.

Add Maven – set name MAVEN\_HOME – paste the path of maven\_home – apply/save.

**System message :** Manage Jenkins – configure global security – select text type : safe html.

**Dashboard - Manage Jenkins** **– Configuration :**

**Project view :** I can make my own view, just click on (+) top on the project then customise the view by selecting columns.

**# number of executors** : 5 (can run 5 jobs in parallel).

**Labels :** I can make my pc as master machine and can add more nodes (machine) and can direct to run any test on any node.

(dash board – Manage Jenkins – Manage nodes and clouds – add node then I need to open more Jenkins-Tab to run more Labels.

**Quiet period** : 10 (seconds). Wait 10 seconds before starting next job.

**SCM checkout retry count** : defines how many times will it try to connect server if it fails.

**Restrict project naming:** it will restrict doing any name other than defined name type.

**Jenkins URL** : localhost:8080 (hostname:portNumber).

**System Admin e-mail address** : you can set your email address as for admin.

**Global properties** : make global properties which can be access from any job.

**Extended E-mail Notification** : if you want to have email notification to your email. Need email plugin down-loaded. Also set the triggers… if you don’t use email plugin then u can use default **email notification**. After configuration email u can verify it using **Test configuration by sending test e-mail** option. Apply / save.

**Then :**

Dashboard – Manage Jenkins – Global Tool Configuration:

**Add JDK** : JAVA\_HOME and path.

**Add Git** :

**Add Maven** : MAVEN\_HOME then path.

**Create a Job:**

**dash-board** - New Item.

-Enter an item name (here I am adding a name ‘MavenProject’)

select FreeStyle Project then OK .

**description** : add some description like (this is maven test or in html <h1>this is maven test<h1/>.

**source code** : none/Git (copy repo path from Git.)

set Credentials: [+ add ] – jenkins – Secret text – copy/paste git token – save.

**Build Triggers :** how frequently tests will be run.

Trigger builds remotely (e.g., from scripts) : this option will give u an url just copy n paste to new tab and change the Jenkins url to my (localhost)IP address:8080. Put authentication token, its like a password then same do at URL. Then save. then go to that tab and hit enter… test will be executed.

Build after other projects are built : name the build after which (1 or more) builds this build will run.

Build periodically : test will run on specified time or period. (Cron timer).

GitHub hook trigger for GITScm polling :

Poll SCM : poll source code management means as soon as build get ready from SCM.

**Build Environment :**

//attach Extent Report in jenkins

**Post-build Actions :** what action you want after this build execution.

> Publish html Report

> HTML directory to archive[report folder path]

> Index pages[Spark.html]

> Report title[Extent Report]

> save.

**Build** :

**Execute local Project from Eclipse work space:**

Jenkins > new item > freeStyle project > Advanced > Use custom workspace > paste project paht > Build > invoke Top level maven project > Goals [test]

-------------------------------------------------------------------------------------

**For Execute Windows batch command:**

-Scroll to ‘Build‘ - ‘Add Build Step‘ then ‘**Execute Windows batch command**’ or ‘Invoke top-level Maven targets‘ from the drop down list.

**First** – set testng.xml in the maven surefire in the pom.xml file also add maven compiter including source and target version.

**new Item** – do name then select **FreeStyle project** – OK –

**build** – **add build step** – **Execute Windows batch command** in **command** box

cd C:\Users\jawad\eclipse-workspace\project1 (cd project path)

mvn clean test

**apply – save**.

Go to **Dash board** – Select project – **Build Now**.

//(project directory) right click on project – properties – location [=>] button –

//copy project path – paste in command box .

**For Invoke top-level Maven Project:**

**New Item** – **FreeStyle project** or can copy existing project – paste to item name but change the name **Build** – Invoke Top level Maven target –

**Maven version** ( MAVEN\_HOME ) –

**Goals** (clean test) –

**Advanced –**

**POM**

C:\Users\jawad\eclipse-workspace\project1/pom.xml

**Apply – save**.

Then **Build now**.

**Execute test using Maven Plugins:**

**Manage Jenkins** – **Manage Plugins** –

**Available** – type(maven) – check (**maven integration**) –

**install without restart**.

Go back **Dash board** –

**new Item** – do project name

now **maven project** is available(select it) – **ok**.

**Build** – **add build step**

Root POM (pom path- pom.xml) –

C:\Users\jawad\eclipse-workspace\project1\ pom.xml

**Goals** (clean test) –

**Advanced** – optional

POM - cd C:\Users\jawad\eclipse-workspace\project1

**Apply** – **save**. then **Build Now**.

**Execute Git Project in Jenkins:**

Login Jenkins

New Item

Select project – maven project > ok.

Source code – Git > enter URL

Select Branch

**Root pom – projectName(from GitHub)/pom.xml**

Goals – clean test

Apply / Save.

**Problem:**

Jenkins can not connect git-

Just add git path till bin to Jenkins Global tool configuration.

**How to set Git credential in Jenkins ?**

Job/project > configure > SCM > Credential > [+Add] > Jenkins > provide Git-user name/email and password / token …

add description (it will help to understand the purpose of Credential).

OR Job/project > configure > SCM > Credential > [+Add] > Jenkins > Secret text > paste Github access Token > do a unique name in ID field > do some description >add/save.

**How to delete Git credential in Jenkins ?**

Dashboard > Manage Jenkins > Security > Credentials > > >

**How to create parameter in jenkins ?**

This project is parameterized > Add Parameter > Choice Parameter > do name > do choices[paramete names] > use parameter name in any field inside jenkins.

When you build job select an parameter option > build now. (can create multiple parameter).

**How to chain execution:** I can chain the tests. for example –

I have 3 tests… **test1, test2, test3**.

Now in test2 set **build trigger** as **Build after other projects are built** - enter project name (test1)**.**

Set **Post-build Actions** as **build other project** and enter project name (test3).

Now execute test1… then all 3 tests will be executed…

: after chain we can create a view- All | click(+) – do name – select delivery pipeline view – OK.

Click component – do a name – select initial job – apply.

Then we can see the view of chained jobs.

**How to create a pipeline:** (need to install pipeline pluggins)

New Item – do project name - select pipleline – OK.

Configure – Advance Project Option – do display name – pipeline – definition – select Pipeline script…

--------------------------------------------------------------------------------------

**E-mail notification**:

Dashboard – Configuration –

first go to **E-mail notification** – SMTP(smtp.gmail.com)

select Advanced option – select smtp authentication and select **use ssl**.

Enter User name([jawadkarim646@gmail.com](mailto:jawadkarim646@gmail.com))

Enter password (app password from gmail)[ rglm vuzm iihs ajvs]

Enter SMTP Port 465.

Select test configuration by sending email

Enter recipient email.

Click test configuration. Email will be verified. Then check your email…

Now in gmail setting select **Allow less secure app.**

**Extended E-mail notification:**

SMTP(smtp.gmail.com), Port 465

Click on **Advanced,**

User name([abcd@gmail.com](mailto:abcd@gmail.com)) – enter password. Select **ssl.**

Select content type : HTML

Default recipient : if any emails …

Default triggers : select any one.. like **always**..

Apply/Save.

**Now** Go to **add-post-build-action**

> select Editable Email Notification

> fill the required fields

> write subject and content for the receipents.

> attach file path

> save.

**Jenkins with GitHub:**

In Jenkins – Dashboard – new Item – name project – Maven project – ok.

**Source code management – Git –** clone repository from Git – paste it –

**Add credential :** select any if needed.

**Branch to build :** default master or any branch name if you have ….

**Build triggers: select any..**

**Build : root pom(**from Git **projectName/pom.xml)**

**Goals** : clean install.

Apply/Save.

**To change the jenkins default url to new url :**

( in cmd >java -jar jenkins.war --httpPort=6161 then hit enter.) = locahost:6161

**How to add testNG report in jenkins**

In jenkins install **TestNG Resuls** pluggin.

Go to configure – Post-Build Action –Publish TestNG Result

After execution

Go to build history – TestNG Result – click

**Jenkins Pipeline**

**1. dev team:**

:developers develop their code and store into repository.

**2. dev ops team:**

:will create the build with the developer's code from the repository.

:they need to do certain numbers of jobs(activities) for build, test, and release.

:will test the build with a QA automation script from the repository. They also write some kind of automation.

:do build deployment process (create build and then test).

:after build and test completion build will be released (build > test > release).

**3. QA team:**

:qa team creates smoke test script and store into repository

:after build deployment qa team install the build to qa enviroment

:will write automation for the build and store into repository.

\*\* This whole process happens continuously on a daily basis and is called continuous integration(CI).

\*\* after build and test completion build will be released continuously is called continuous delivery(CD).

Jenkins Pipeline process

Build > Deploy > Test > Release.

Step 1: Create - chain required jobs related to build, deploy, test and release.

example-

i) sampleBuildJob (freestyle)

ii) sampleDeployJob (freestyle)

iii) sampleTestJob (freestyle)

iv) sampleReleaseJob (freestyle)

Now chain the jobs:

go to sample deploy project > Build Triggers > Build after other projects are built enter name sampleBuildJob .

go to sample test project > Build Triggers > Build after other projects are built enter name sampleDeployJob .

go to sample release project > Build Triggers > Build after other projects are built enter name sampleTestJob.

Now run SampleBuildProject then all of them will be executed.

we need a Pipline to run all jobs at a sequential like above..

for Pipeline we need to install BuildPipelinePlugin in Manage jenkins

now create a Build Pipeline: click(+) > BuildPipelineView > do name > OK > select initial job > No of deployed builds[3,4,5 or any] > apply > ok.

now it will show a 'Build Pipeline'. just run the Build Pipeline then all the jobs will be run. after execution we can see the run history.

To create a DeliveryPipeline: click(+) > DeliveryPipelineView > do name > OK > Pipelines-Components[add] > >do name(optional) > select initial job(sampleBuildJob) > apply > ok. once you click ok then you can the complete report here. it is just for report purpose.