**MAVEN-JENKINS-GITHUB**

1. Ensure you have all the required plugins installed in Jenkins.

Plugins; Role-based Strategy, build pipeline, Junit, Selenium.

1. Ensure you have the correct Eclipse downloaded, and the maven plugins installed in the Eclipse
2. Ensure to configure environmental variables

**To install a plugin**:

On the Jenkins Dashboard

Manage Jenkins🡪Plugin manager->Available plugin->(Search for the plugin you want to install

**How to create a role in Jenkins**:

Manage jenkins->Configure global Security->Authorization->Role-based Strategy->Apply & Save

After this got to Manage Jenkins->Manage & Assign role-> give name & permissions

Item Role: Ensure you add **.\* on the Pattern test**

**How to add users:**

Manage Jenkins🡪 Manage user🡪create User

**Using Jenkins on the CLI**

Localhost8080/cli/🡪download Jenins-cli.jar

Copy the Jenkins cli upto; 8080/build (job-name)->permission

**To log into Jenkins:**

Go to your command prompt🡪 cd downloads🡪 java -jar Jenkins.war

Then go to google and type: **localhost:8080** on the search bar

**How to create CI pipeline (Downstream & Upstream)**

Create more than 1 jobs; for this example, we create Job A, Job B and Job C

Manage Jenkins->Configure->Build steps->execute with windows batch-> type message starting “echo..->Apply & Save

Then go to Post build action🡪Build another project🡪Add job🡪Trigger if stable/fail/unstable🡪put jobs in the sequences you want them to run i.e, Job A, then Job B and Job C

On the jobs dashboard🡪click on + (New View)🡪>Give the pipeline a name🡪build pipeline view🡪create🡪select job that you want to want for upstream & downstream🡪run

**MAVEN**

Maven is a build management tool for java projects that helps to execute build life cycle frameworks. Maven uses POM.XML (project Object Model) file, to build projects. POM.XML file contains configurations and information regarding the project such as the dependencies, group Id, Artifact Id, Versions and plugins.

**Difference between Maven & Jenkins**

Jenkins is mostly a CI/CD tool, which uses other tools to automate the complete build cycle, while Maven is a build lifecycle management tool that helps with dependency management.

How to set up environment variables & path

-copy the path to where your apache maven files are stored i.e **C;\Users\Lillie\Desktop\apache-maven**

-copy the path to the maven bin folder is i.e **C:\users\Lillie\Desktop\apache-mave\bin**

**Right click on this PC🡪properties->Advanced setting->environment variables🡪New🡪give variable name i.e MAVEN-HOME🡪** Paste the path copied to maven folder

Specify the path to binary files HOME as well

Environment variable🡪click on path🡪edit🡪paste the bin path and paste

On your **cmd prompt; run the command >mvn --version** to confirm

**How to create a Maven Project in Eclipse**

File->New->Project->Maven project-> check create a simple project->Specify Group ID & Artifact ID->finish

Once created you will see your project on the far-left hand side. Click on the project to view src/main/java ; src/main/resources: src/test/java ; pom.xml file

-In the pom.xml file, you will see your Group Id, ArtifactId; Version

**To add a test case in your maven project;**

Click on your project🡪New->Class🡪Give it a name🡪 Finish

**How to set up your dependencies**

Google “Maven Repository🡪 junit testing dependencies🡪 copy them-> go to your maven project🡪 pom.xml file and add them; to do this type and opening tag for dependencies, paste then put a closing tag.

-you can add as many dependencies as your project needs e.g Selenium dependencies, web driver, etc

-Once you do this, SAVE it, and you will see It downloading and creating jar files for the specified dependencies.

**To integrate Maven with Github via SSH key**

On Maven/Eclipse Windows🡪Preferences🡪 General🡪Network Connection🡪SSH🡪SSH2->Key management🡪Generate RSA key🡪 Copy the RSA key

Go to Github🡪Profile->Settings🡪SSH &GPG key🡪New SSH key🡪Paste the Key

**On your maven project**;

Go to src/main/java🡪New->Class🡪Give it a name🡪Finish

Go to Maven Eclipse🡪click on Clone a Git Repo🡪Paste the github URL🡪Next

**Once the Repo has been created, you need to include your project inside your github**:

Right click on your project🡪Team🡪Share project🡪Select the github URL 🡪Finish

Go to your project again🡪right click🡪Team🡪Add to index

You need to stage all the unstaged files; You can do this by clicking on the +sign above; or drag and drop them from the unstaged area, to the staged space🡪Add a commit message🡪click on commit & Push🡪 add your github password

* Go to github->Create new Repo🡪Give it same name as your maven project🡪 commit

Go back to your Maven Project🡪 right click🡪Team🡪Push🡪paste the URL🡪Provide user Name & Password->next🡪

**How to integrate with Jenkins**

--Go to your Github &copy the URL

Go to your Jenkins dashboard🡪New Item🡪Give it a Name🡪Maven Project🡪Create

-On the general section🡪Github project🡪Paster the URL

On Source code management🡪 Git🡪Paster the URL🡪 Specify the Branch🡪Specify the build trigger->Specify build goal i.e compile:

On Post Step🡪 invoke top level Maven target🡪 enter the targets i.e pmd:checkstyle

* Apply 🡪 Save🡪 Build Now