**Automation | BDD Process**

***Knowledge Transfer Document***

Prepared by:

Mohit Manchanda

Contents

[Steps to start with BDD: 3](#_Toc476057347)

[GitHub 10](#_Toc476057348)

[Download GitHub 10](#_Toc476057349)

[To commit code in Github 10](#_Toc476057350)

[Jenkins 12](#_Toc476057351)

[Create Jenkins job for new project 12](#_Toc476057352)

[Optional 14](#_Toc476057353)

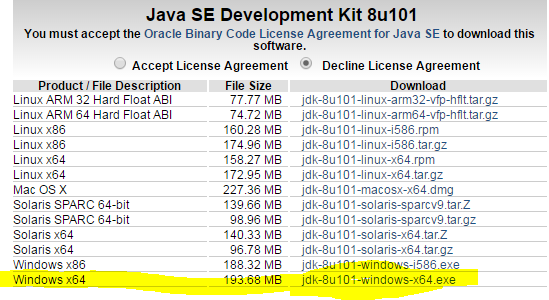
# Steps to start with BDD:

1. Download and Install **JDK** with default settings for windows :- <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

Click on download button present



Choose latest version of JDK for windows



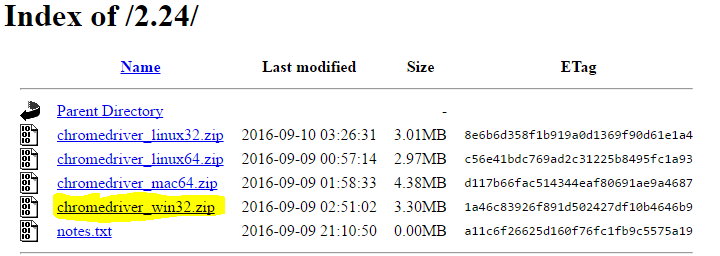
Install it and after installing it open windows explorer, right click on computer 🡪 properties 🡪 Advance system settings 🡪 Environment Variable 🡪 System variable

Ensure path must contains your **JDK installation path**, if not then add after semicolon’**;**’ and at the last again put semicolon

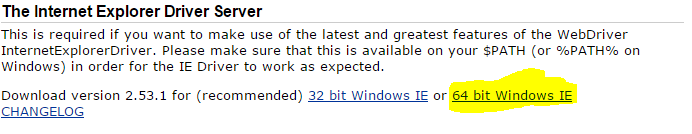
Example: C:\Program Files\Java\jdk1.8.0\_91\bin;

1. Download chrome and IE webdriver

Chrome: <http://chromedriver.storage.googleapis.com/index.html?path=2.24/>

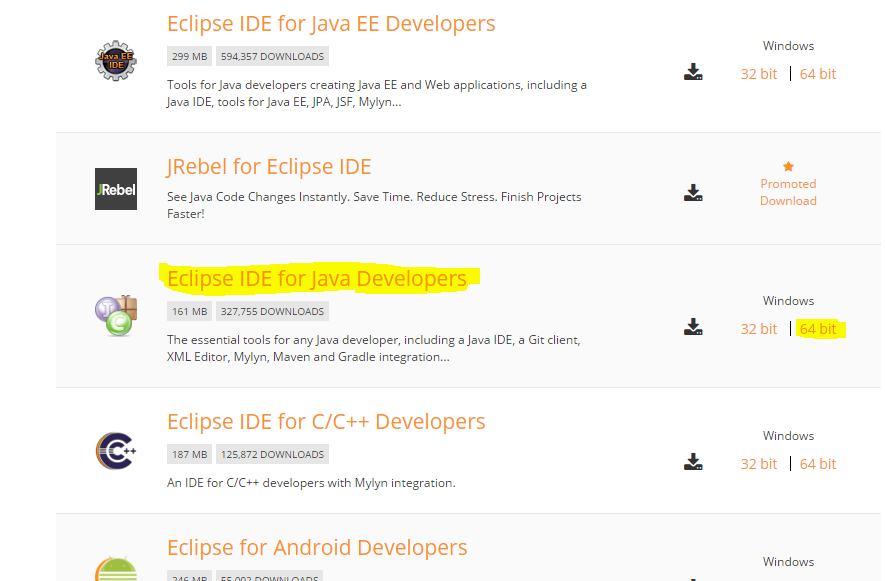


IE: <http://www.seleniumhq.org/download/>



1. Download and install **Eclipse** with default setting :- <https://eclipse.org/downloads/eclipse-packages/>

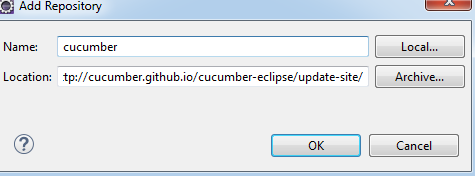
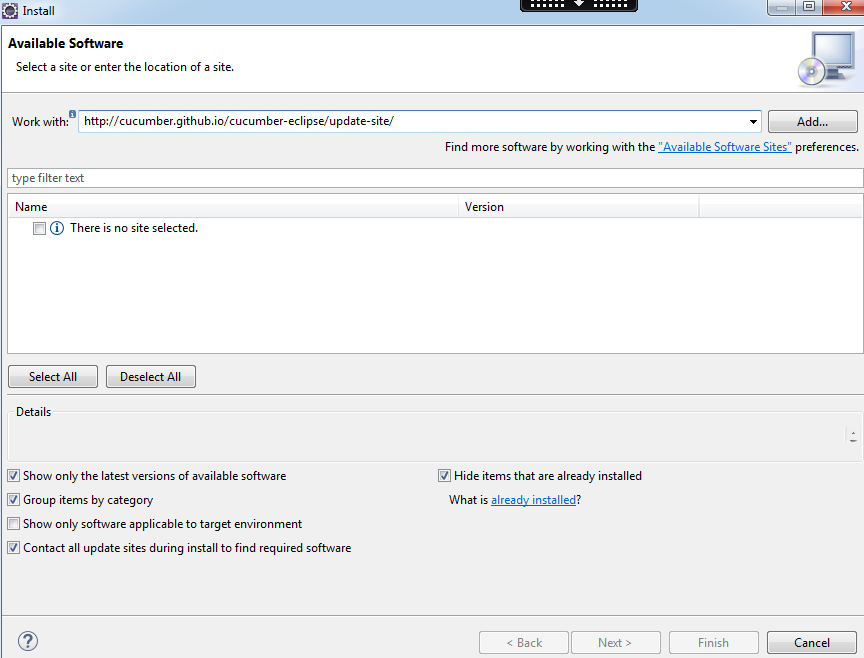
Click on below highlighted path

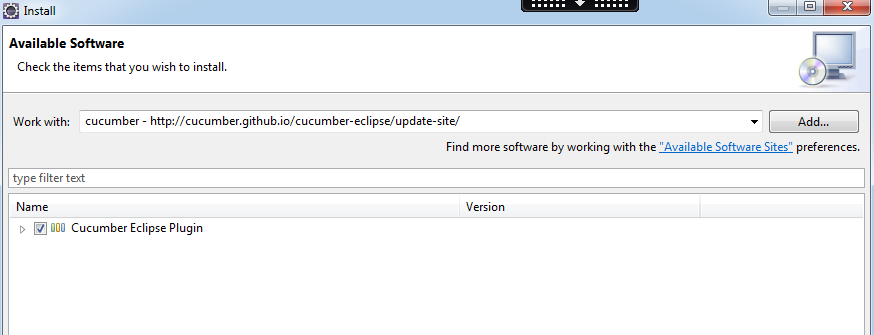


After open eclipse, provide any workspace system path like - C:\Users\narendra.modi\Desktop\Workspace

1. Install cucumber(<http://cucumber.github.io/cucumber-eclipse/update-site/> ), TestNG(optional) and maven plugin(It may be already installed) from **Install new Software** or **Eclipse Marketplace** present under **Help** in eclipse

**Install new Software**

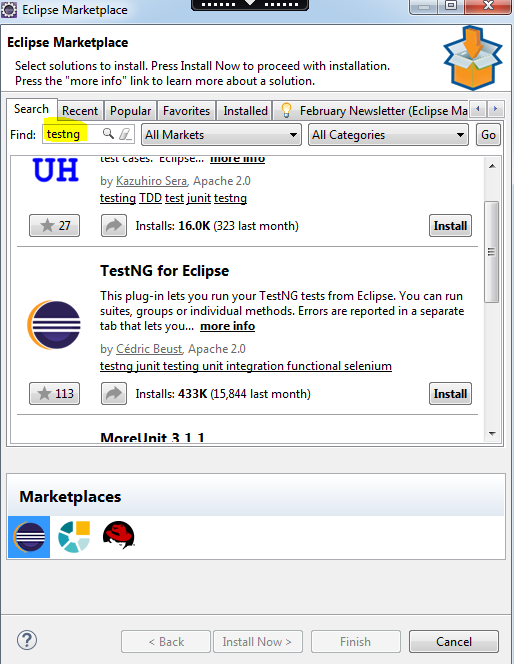
Enter URL and click on add

1. Select below option
2. And click on finish project.

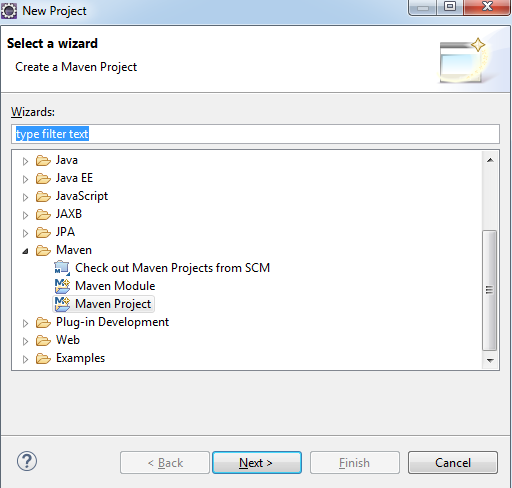
OR

**Eclipse Marketplace**

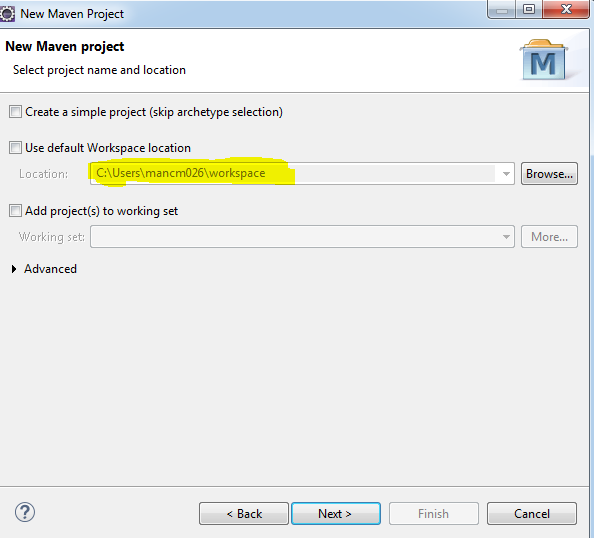
Find by keyword like



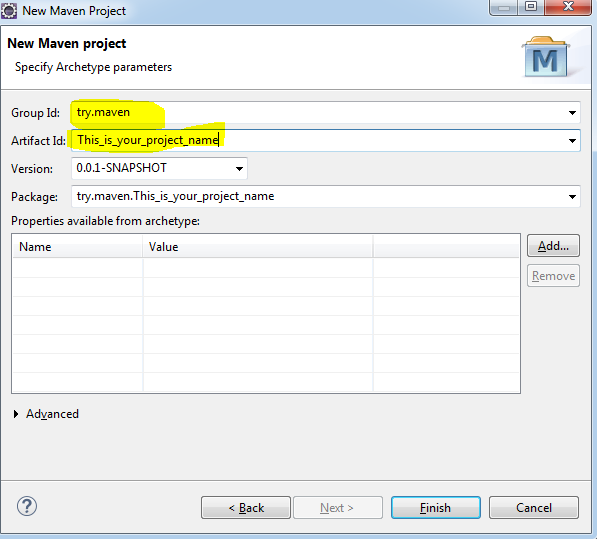
1. Create new Maven project(File 🡪 Project 🡪 Maven project), new package and delete app.java from Main and test package folder



Provide your workspace path



**Enter project name and any group ID**



1. Open POM file and add below dependencies

*Java jar files*

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.11</version>

<scope>test</scope>

</dependency>

*Cucumber jar files*

<dependency>

<groupId>info.cukes</groupId>

<artifactId>cucumber-java</artifactId>

<version>1.2.3</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>info.cukes</groupId>

<artifactId>cucumber-picocontainer</artifactId>

<version>1.2.3</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>info.cukes</groupId>

<artifactId>cucumber-junit</artifactId>

<version>1.2.3</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>info.cukes</groupId>

<artifactId>gherkin</artifactId>

<version>2.12.2</version>

</dependency>

*Selenium jar files*

<dependency>

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

<artifactId>selenium-java</artifactId>

<version>2.53.0</version>

</dependency>

***Optional*** *- Excel sheet jar files*

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi</artifactId>

<version>3.15-beta2</version>

</dependency>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi-scratchpad</artifactId>

<version>3.15-beta2</version>

</dependency>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi-ooxml</artifactId>

<version>3.15-beta2</version>

</dependency>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi-ooxml-schemas</artifactId>

<version>3.15-beta2</version>

</dependency>

1. Save and run as Maven test, Verify you receives a build success message.
2. Create RunnerTest.java class in test package and note name must be “RunnerTest” only because Jenkin read names contains **test** for running maven project.
   1. Add below two lines before class line

@RunWith(Cucumber.**class**)

@CucumberOptions(

plugin = {"pretty", "json:target/cucumber-report.json", "html:target/cucumber-htmlreport"},

features = {"src/test/features"}

)



1. Place feature file in “src/test/features” this folder because we mentioned its path in above step.
   1. Use colon with feature and scenario name without any space.

Example “Feature:” and “Scenario:”

* 1. Extension of feature file must be feature



1. Refresh project in eclipse to reflect feature file and run feature file as cucumber feature.

On running it, missing steps will be generated in console as methods name. Copy and paste missing steps in your java class and put your selenium code under methods.

Note: Do not include **public static void main** in it, new class created in above step.

# GitHub

## Download GitHub

* + GitHub Desktop = <https://desktop.github.com>
  + Git for Windows = <https://git-scm.com/downloads>

GitHub repository must be created to perform below steps. If not then you can create your own repository only if you have admin access. (github.com)

## To commit code in Github

1. Open maven project in window explorer. Right click and select option “Git bash here”.

Perform below steps on the very first time

* In the Bash console, type git init <enter>
* Include Maven files into Git stage area by typing git add . <enter>
* Commit stage are by typing git commit -m ‘Initial commit‘ <enter>
* Connect now your local Git to your GitHub repository by typing

git remote add origin https://github.com/<<Organization>>/<<Repository>>.git <enter>

-- Or --

git remote add origin git@github.<<client GitHub enterprise name>>.com:<<Organization>>/<<Repository>>.git <enter>

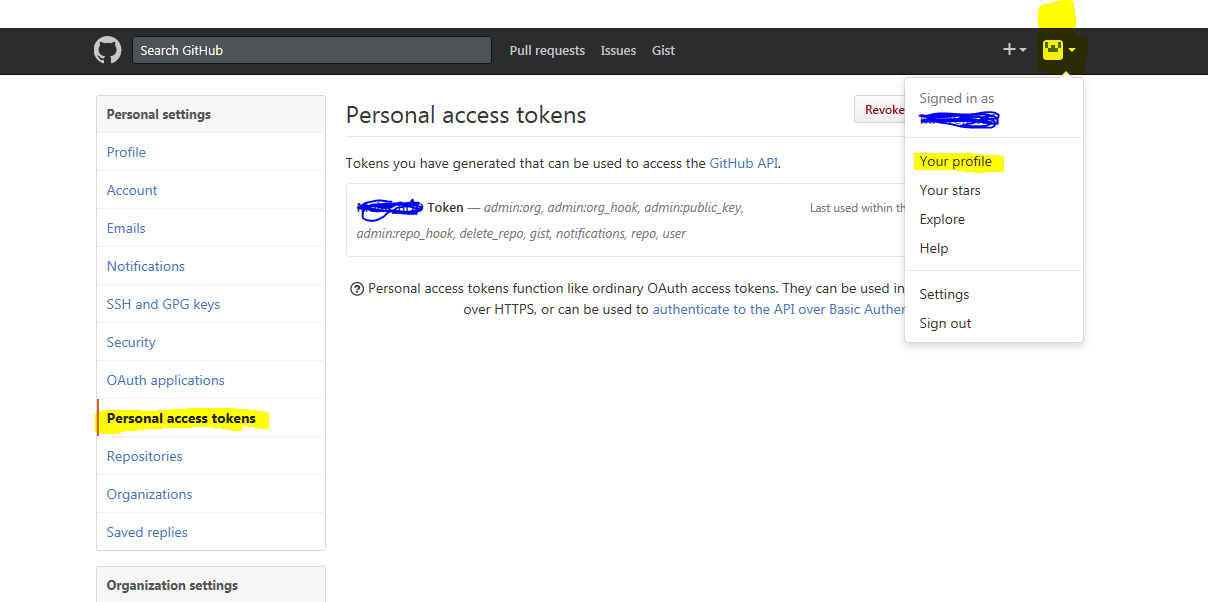
* Finally push the code from your local machine to the remote repository through the command git push -u origin master <enter>
* Close Git Bash window by typing exit <enter>



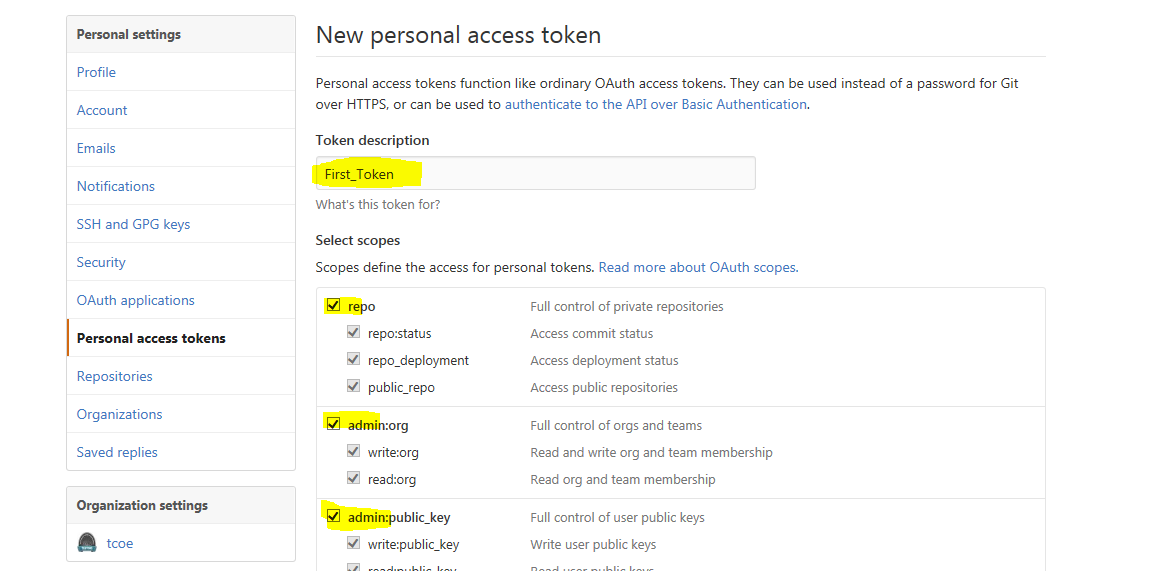
1. Verify your code got pushed in github.com

Generate Token – Token will be used as a password while commit your changes in git

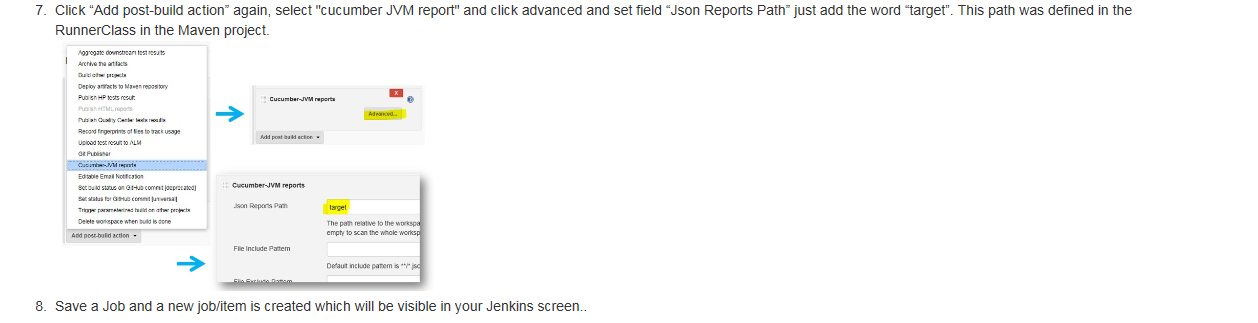
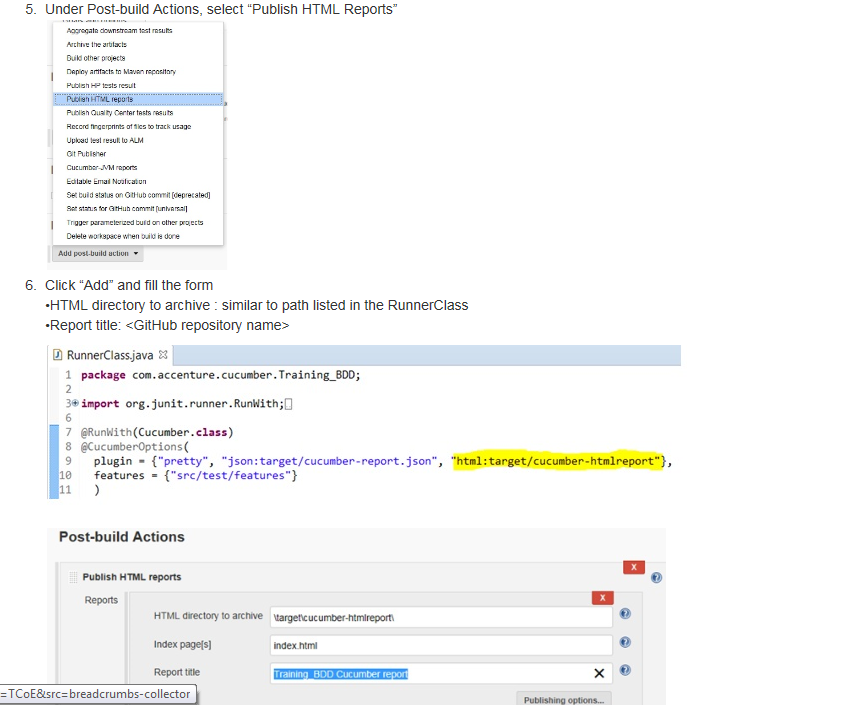
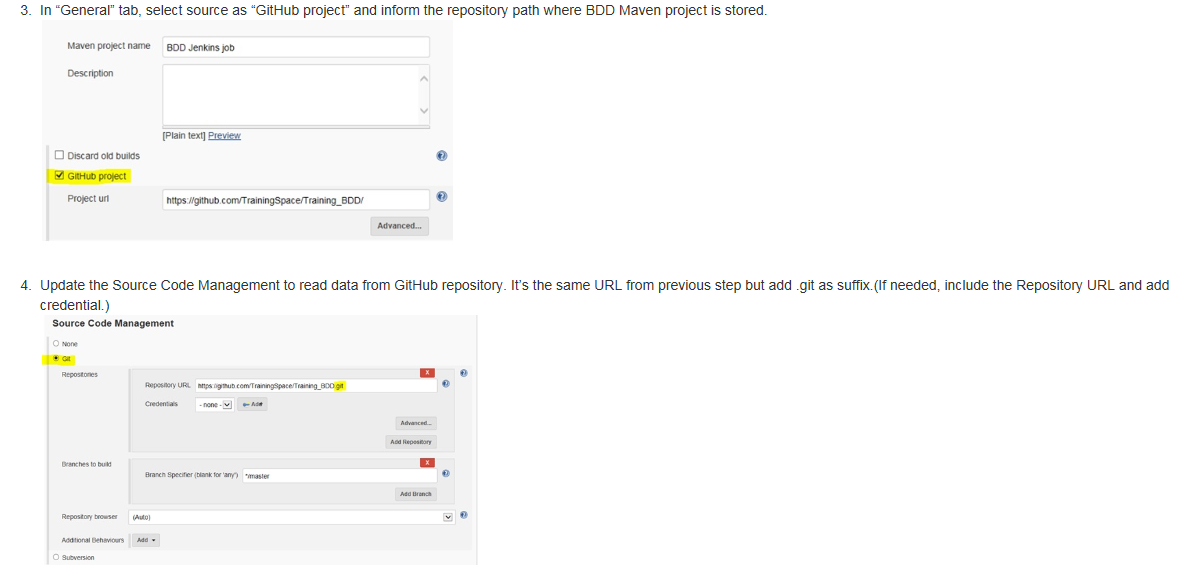
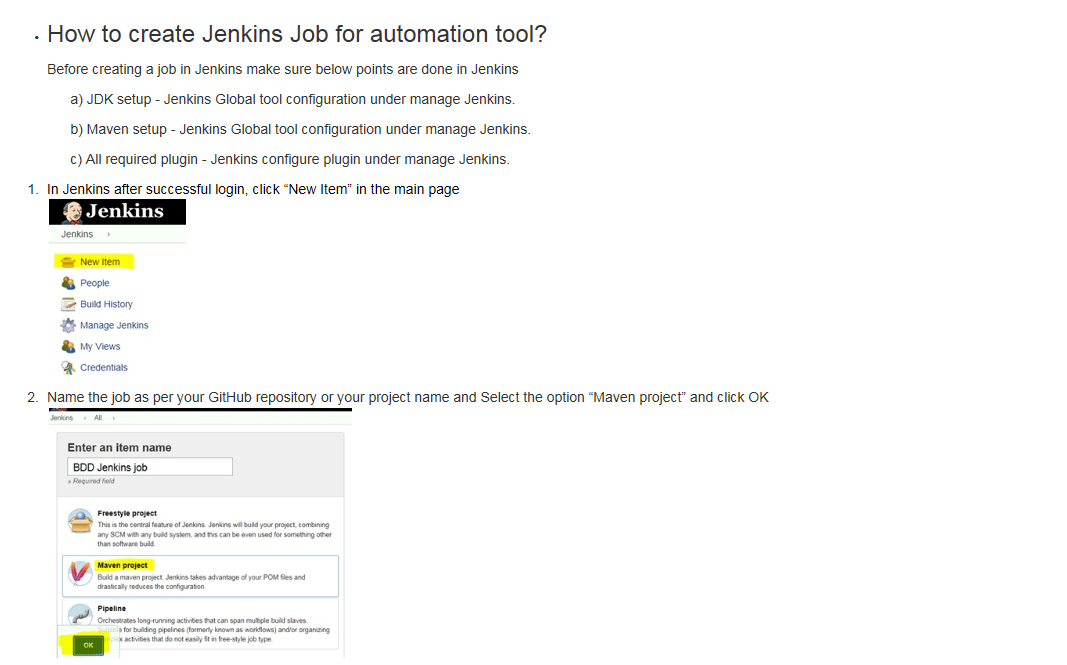
– Click on view profile 🡪 your profile 🡪 personal access token 🡪 generate new token



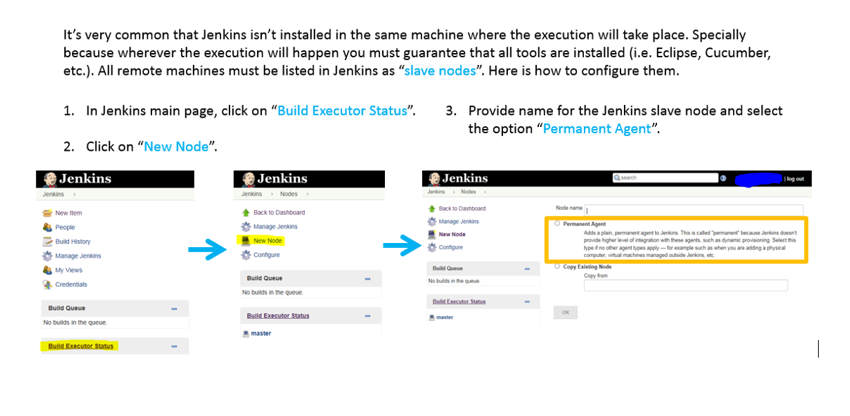
Add any description and select all checkbox present and click generate token

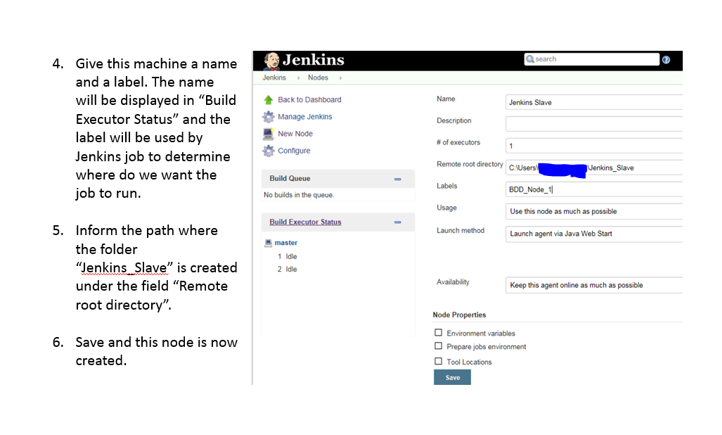
**Note: -** Save generated token as this will be used later on many times while commit your code 

# Jenkins

Create Jenkins job for new project

Optional (Not Mandatory) **Jenkins slave** setup

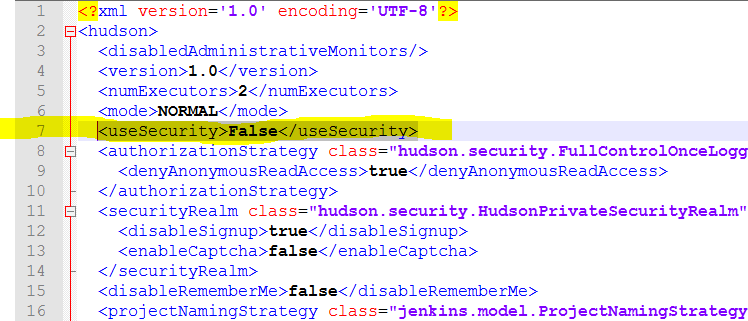




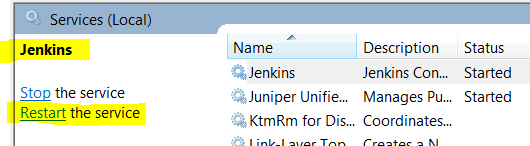
# Optional

**You can also set up your own Jenkins on your local by using below steps**

1. Download Jenkins <https://jenkins.io/download>
2. If after your installation Jenkins initiates in the login page and you just don’t have credentials – somehow it may never ask you to actually define one – then here is a workaround to bypass security but you should adjust it later.
   * Through Win Explorer, navigate where Jenkins is installed.
   * Open Notepad as administrator.
   * Open the file config.xml
   * Update the tag <useSecurity>False</useSecurity>

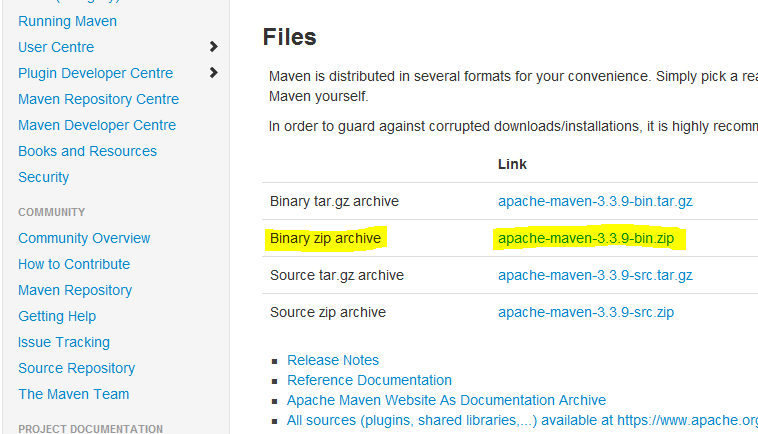


* + Save the file.
  + Through Win main button, search for Services.

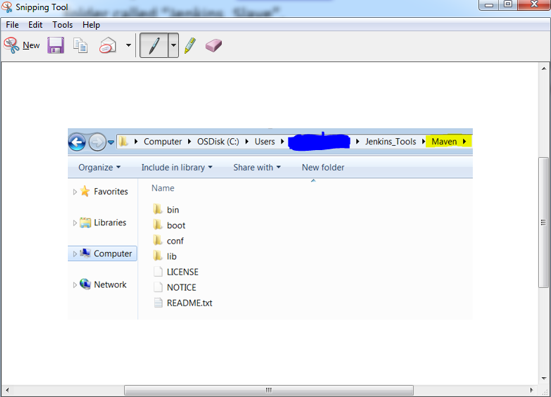


* + Locate Jenkins and restart the service.

1. Open any browser and call the URL <http://localhost:8080/>
2. Download Maven binary zip archive from <https://maven.apache.org/download.cgi>

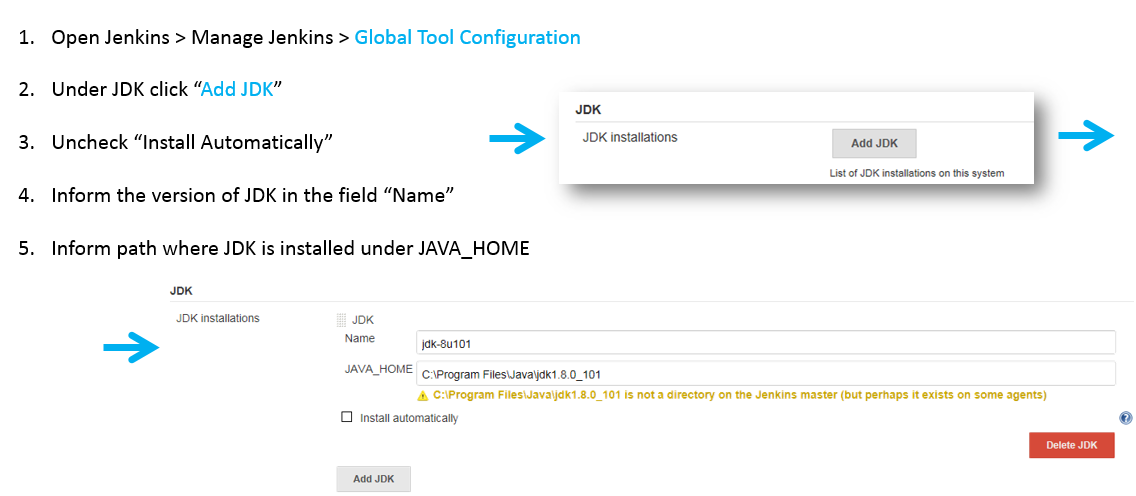


* + Create a new folder wherever you have enough repository available. Name this folder “Jenkins\_Tools”.
  + NOTE: If your machine will become a node for remote execution then create a folder called “Jenkins\_Slave”.
  + Copy Maven into this folder and unzip it.
  + SUGGESTION: Rename the folder where Maven is hosted to a generic name like “Maven” without the version number. By doing so you avoid having to change Jenkins configurations once/if Maven version ever changes.

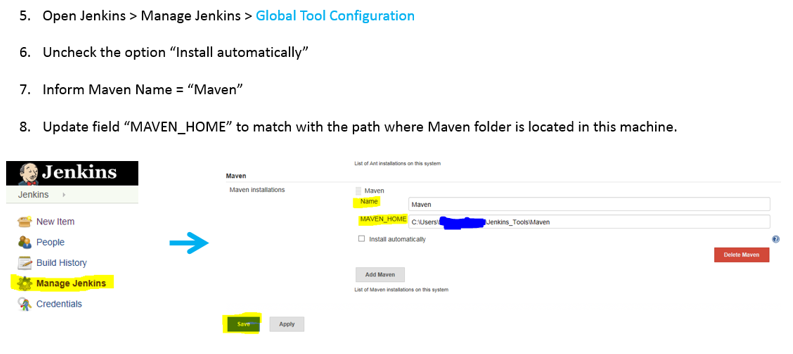


1. Setup Jenkins

* JDK

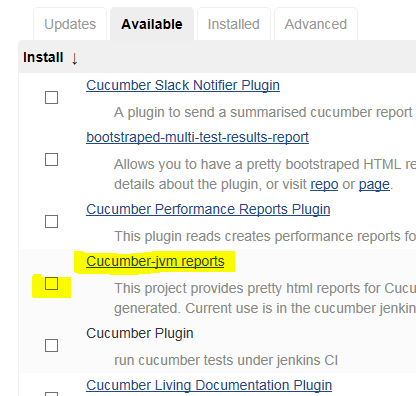


* Maven



1. Add all plugins from Manage Jenkins 🡪 Configure plugin

* Cucumber-JVM reports plugins, click on “Install without restart”



* Similarly HTML Publisher plugin and envinjector for environment variables