

Well Come to Automation Testing

In this tutorial you will learn how to automate any website by Testing Framework. This tutorial base on the Selenium TesNg framework.

1)Set Up the default framework setup for any project

2)Standard architecture for Automation Framework

3) Standard scripting level

4) Slandered data driven architecture (Next Tutorials)

5) Test case Execution Strategy (Next Tutorials)

6) Customization Report (Next Tutorials)

7) Jenkins and Selenium Grids (Next Tutorials)

Note: Read documents thoroughly any. Documentation is still under process. This is not the final version .For ant concern reach out to me
@(mchowdhury891@outlook.com)

Installation of JDK

To install JDK you need to download JDK, Unzip the folder

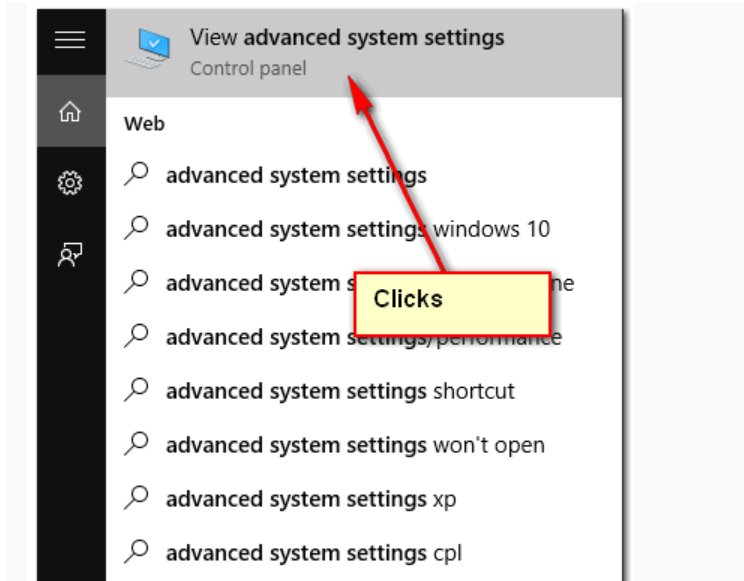
Tools

1. Windows 10
2. JDK 1.8

Don't be bored Take a break again continue To Next Page no:

1. Advanced System Settings

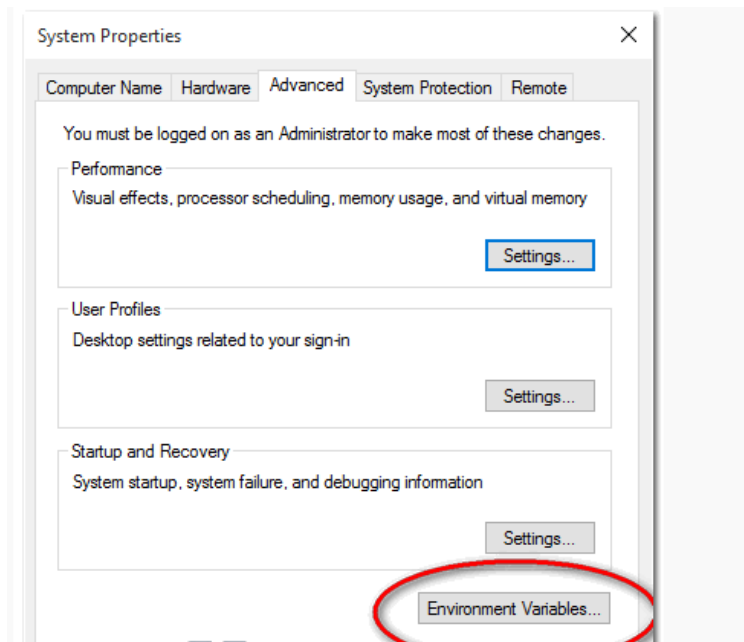
Type **advanced system settings** in the search box (beside the Windows start button), clicks **View advanced system settings**.



2. Environment Variables

Select **Advance** tab, clicks **Environment Variables**

Don't be bored Take a break again continue To Next Page no:



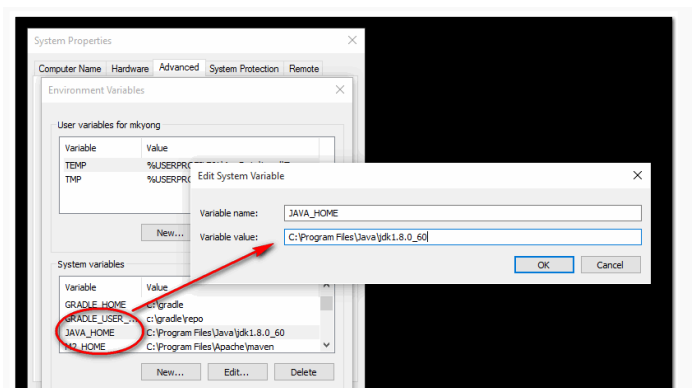
3. Add JAVA_HOME

In System variables, add a new **JAVA_HOME** variable and point it to the JDK installed folder.

Don't be bored

Take a break again continue

To Next Page no:



Note

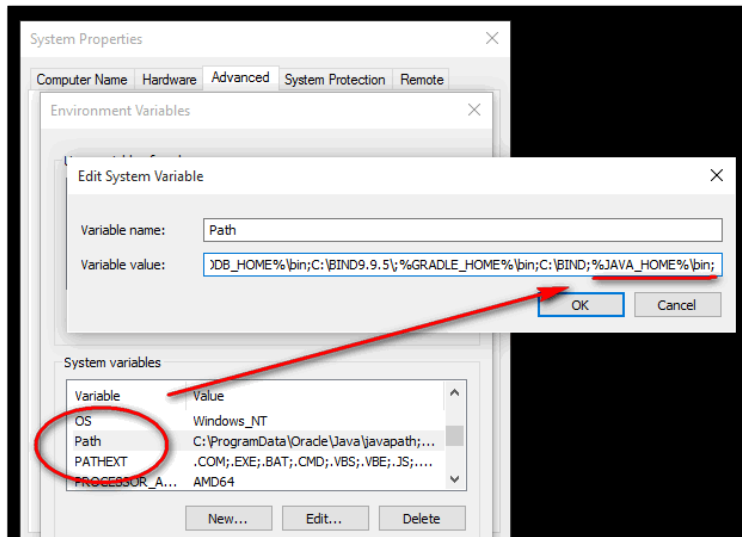
Don't include the `\bin` folder, just the JDK path. For example

3. **Correct** – `C:\Program Files\Java\jdk1.8.0_60`
4. **Wrong** – `C:\Program Files\Java\jdk1.8.0_60\bin`

4. Update PATH

In System variables, find `PATH`, click edit and append this `%JAVA_HOME%\bin` to the end.

Don't be bored Take a break again continue To Next Page no:



*P.S Puts the **JAVA_HOME\bin** in **PATH** make the Java's commands are accessible from everywhere.*

5. Test

Open a command prompt, type:

```
C:\Users\mjoy>java -version
java version "1.8.0_60" Java(TM) SE Runtime Environment (build 1.8.0_60-b27) Java HotSpot(TM) 64-Bit Server VM (build 25.60-b23, mixed mode)
```

Don't be bored Take a break again continue To Next Page no:

Don't be bored

Take a break again continue

To Next Page no:

.

Installation Maven for windows

To install [Apache Maven](#) on Windows, you just need to download the Maven's zip file, and Unzip it to the directory you wish to install, and configure the Windows environment variables.

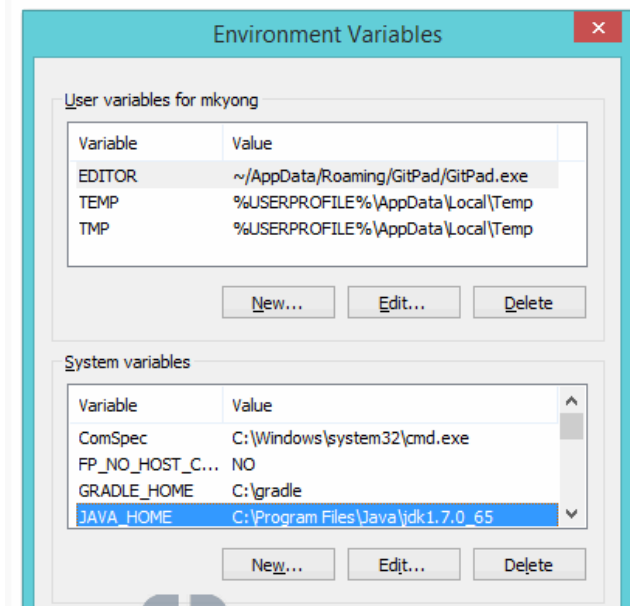
Tools Used :

1. JDK 1.7
2. Maven 3.2.2
3. Windows 8

1)JDK and JAVA_HOME installation

Make sure JDK is installed, and “**JAVA_HOME**” variable is added as Windows environment variable.

Don't be bored Take a break again continue To Next Page no:



2. Download Apache Maven

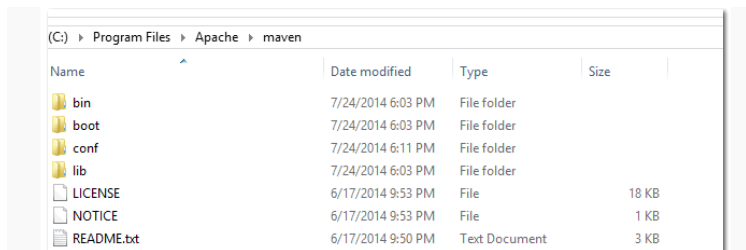
Visit [Maven official website](#), download the Maven zip file, for example : `apache-maven-3.2.2-bin.zip`. Unzip it to the folder you want to install Maven.

Assume you unzip to this folder – `C:\Program Files\Apache\maven`

Don't be bored

Take a break again continue

To Next Page no:



(C:) > Program Files > Apache > maven

Name	Date modified	Type	Size
bin	7/24/2014 6:03 PM	File folder	
boot	7/24/2014 6:03 PM	File folder	
conf	7/24/2014 6:11 PM	File folder	
lib	7/24/2014 6:03 PM	File folder	
LICENSE	6/17/2014 9:53 PM	File	18 KB
NOTICE	6/17/2014 9:53 PM	File	1 KB
README.txt	6/17/2014 9:50 PM	Text Document	3 KB

Note

That's all, just folders and files, installation is **NOT** required.

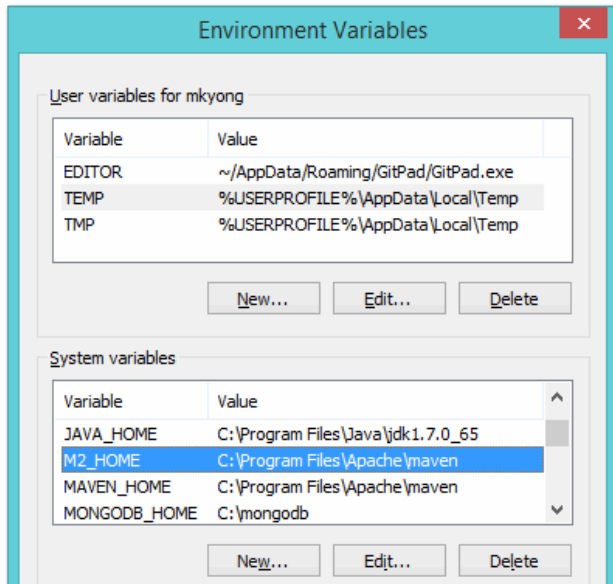
3. Add M2_HOME and MAVEN_HOME

Add both **M2_HOME** and **MAVEN_HOME** variables in the Windows environment, and point it to your Maven folder.

Don't be bored

Take a break again continue

To Next Page no:



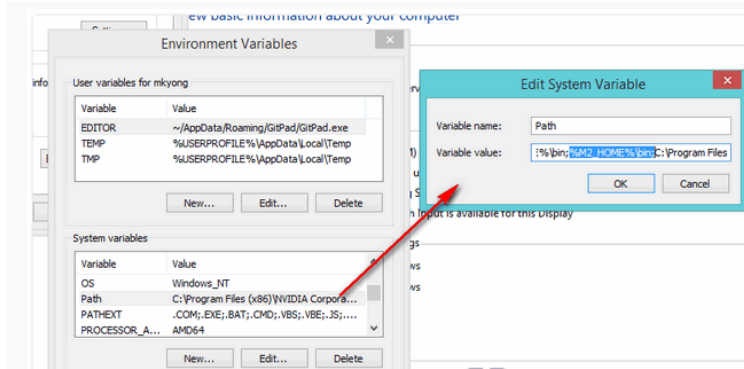
M2_HOME or MAVEN_HOME

Maven document said add M2_HOME only, but some programs still reference Maven folder with MAVEN_HOME, so, it's safer to add both

Don't be bored Take a break again continue To Next Page no:

4. Add To PATH

Update **PATH** variable, append Maven bin folder – **%M2_HOME%\bin**, so that you can run the Maven's command everywhere.



5. Verification

Done, to verify it, run **mvn -version** in the command prompt.

```
C:\Users\mjoy>mvn -version
Apache Maven 3.2.2 (45f7c06d68e745d05611f7fd14efb6594181933e; 2014-06-17T21:51:42+08:00)
Maven home: C:\Program Files\Apache\mavenJava
Java version: 1.7.0_65, vendor: Oracle Corporation
Java home: C:\Program Files\Java\jdk1.7.0_65\jre
Default locale: en_US, platform encoding: Cp1252
```

Don't be bored

Take a break again continue

To Next Page no:

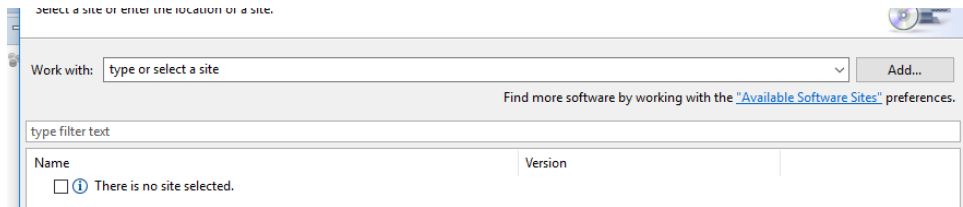
Maven Installation

Pre-Request: Maven Home valuable is already set Up in Environment variable (How to Install go to Page)

Steps

1) Open **Eclipse**

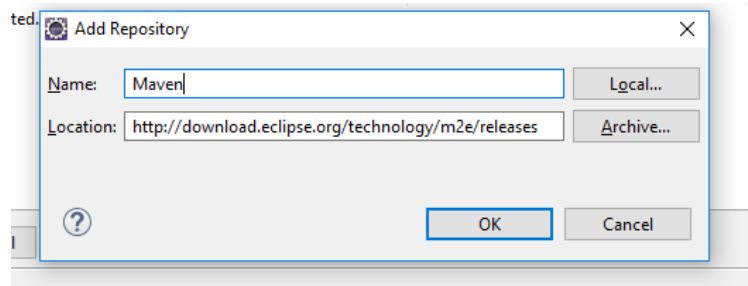
2) Click on Install new software under Help



3) Click on ADD

4) Enter Name: Maven

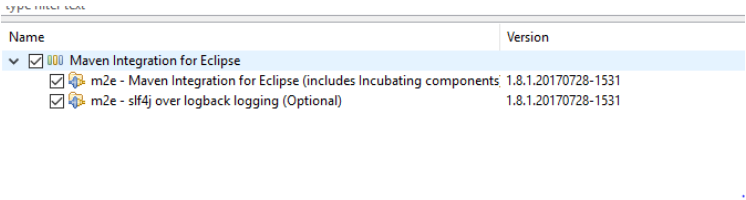
5) Enter URL: According to Release Note (Example-<http://download.eclipse.org/technology/m2e/releases>). You can check from <http://www.eclipse.org/m2e/>



6) Click on OK

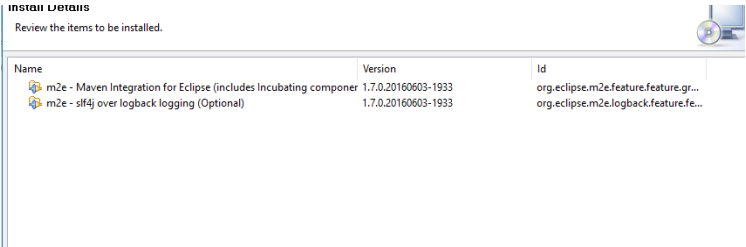
Don't be bored Take a break again continue To Next Page no:

7)Select the Maven Integration for Eclipse and Click Next



Note: It will calculate the dependency time may take some time

8) click Next after review the software which you are installing



9) Accept the terms and license agreement and click on Finish

Don't be bored Take a break again continue To Next Page no:

Eclipse Foundation Software User Agreement
April 9, 2014

Usage Of Content

THE ECLIPSE FOUNDATION MAKES AVAILABLE SOFTWARE, DOCUMENTATION, INFORMATION AND/OR OTHER MATERIALS FOR OPEN SOURCE PROJECTS (COLLECTIVELY "CONTENT").
USE OF THE CONTENT IS GOVERNED BY THE TERMS AND CONDITIONS OF THIS AGREEMENT AND/OR THE TERMS AND CONDITIONS OF LICENSE AGREEMENTS OR NOTICES INDICATED OR REFERENCED BELOW. BY USING THE CONTENT, YOU AGREE THAT YOUR USE OF THE CONTENT IS GOVERNED BY THIS AGREEMENT AND/OR THE TERMS AND CONDITIONS OF ANY APPLICABLE LICENSE AGREEMENTS OR NOTICES INDICATED OR REFERENCED BELOW. IF YOU DO NOT AGREE TO THE TERMS AND CONDITIONS OF THIS AGREEMENT AND THE TERMS AND CONDITIONS OF ANY APPLICABLE LICENSE AGREEMENTS OR NOTICES INDICATED OR REFERENCED BELOW, THEN YOU MAY NOT USE THE CONTENT.

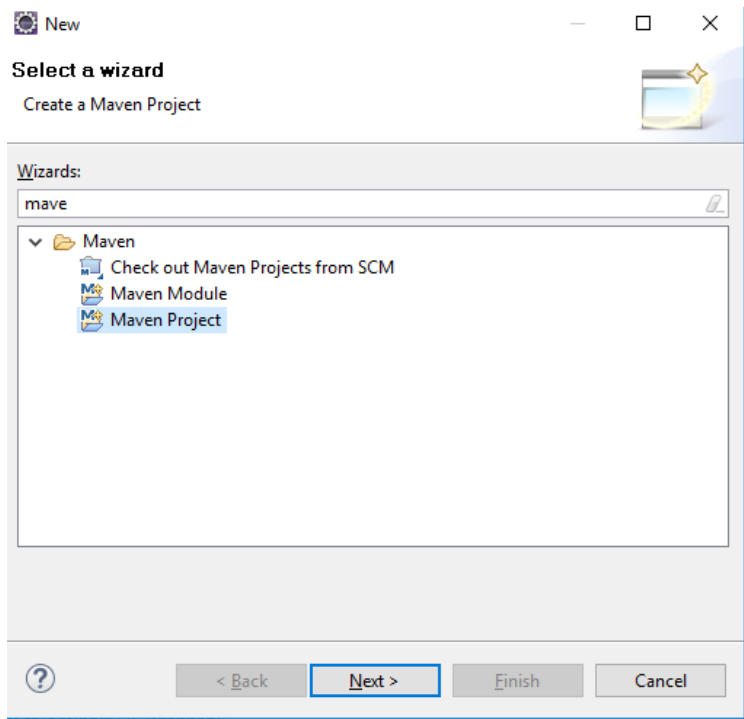
- ☒ I accept the terms of the license agreement
☐ I do not accept the terms of the license agreement

10)It will install maven and ask for restart the Eclipse. Restart Eclipse

11) Create a new project

File ->New ->Others->Maven->Maven Project

Don't be bored Take a break again continue To Next Page no:



12) Use default work location and Select the current location

13) Select quick start project and Click on Next

Don't be bored Take a break again continue To Next Page no:

Catalog: All Catalogs Configure...

Filter:

Group Id	Artifact Id	Version
org.apache.maven.archetypes	maven-archetype-plugin	1.2
org.apache.maven.archetypes	maven-archetype-plugin-site	1.1
org.apache.maven.archetypes	maven-archetype-portlet	1.0.1
org.apache.maven.archetypes	maven-archetype-profiles	1.0-alpha-4
org.apache.maven.archetypes	maven-archetype-quickstart	1.1
org.apache.maven.archetypes	maven-archetype-site	1.1
org.apache.maven.archetypes	maven-archetype-site-simple	1.1

An archetype which contains a sample Maven project.

☒ Show the last version of Archetype only ☐ Include snapshot archetypes Add Archetype...

► Advanced

? < Back Next > Finish Cancel

14) Enter Group ID: Example Automation, Artifact ID: Example Automation and keep other things remain same.

Don't be bored Take a break again continue To Next Page no:

New Maven project

Specify Archetype parameters

Group Id:

Automation

Artifact Id:

demo

Version:

0.0.1-SNAPSHOT

Package:

Automation.demo

Properties available from archetype:

Name	Value	

Add...

Remove

► Advanced

?

< Back

Next >

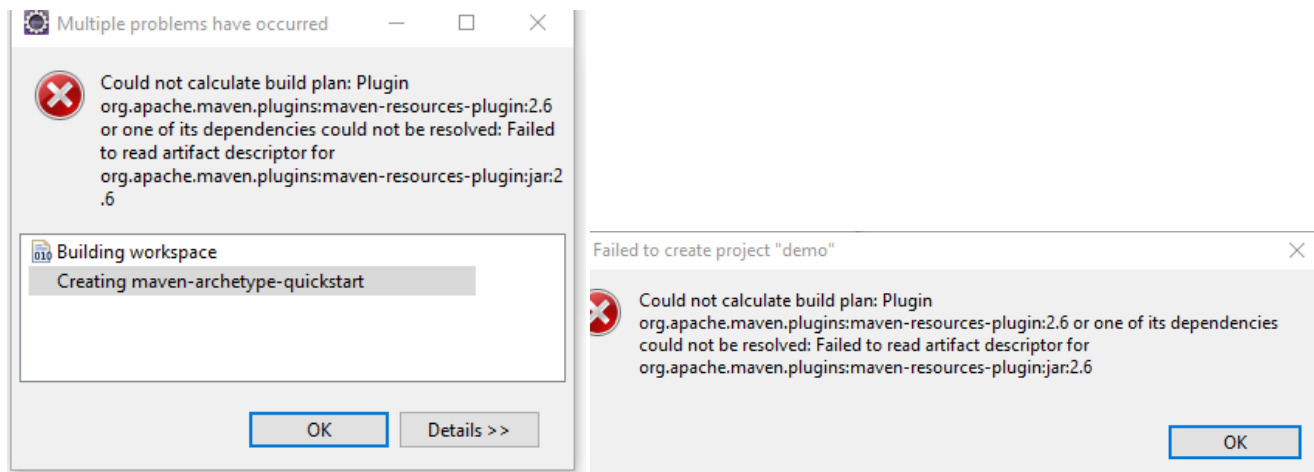
Finish

Cancel

15)If you are getting belllow error don't be afraid and keep patient and follow the steps as it is mention

Don't be bored Take a break again continue To Next Page no:

.



15) Open folder **conf** where you downloaded the Apache Maven (For My case: F:\selenium\apache-maven\apache-maven-3.5.0\conf)

16)Setting xml file

17)Add this Entry and save

- `<localRepository>F:\selenium\apache-maven\apache-maven-3.5.0\repository</localRepository>`

Don't be bored Take a break again continue To Next Page no:

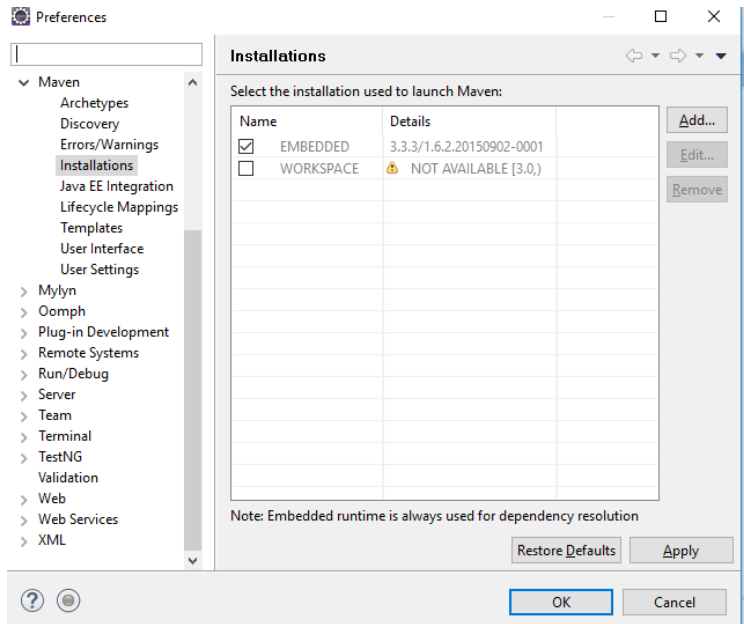
```

43 | Values (values used when the setting is not specified) are provided.
44 |
45 |-->
46 <settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
47         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
48         xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 http://maven.apache.org/xsd/se
49 <!-- localRepository
50 | The path to the local repository maven will use to store artifacts.
51 |
52 | Default: ${user.home}/.m2/repository
53 <localRepository>/path/to/local/rep</localRepository>
54 -->
55 <localRepository>F:\selenium\apache-maven\apache-maven-3.5.0repository</localRepository>
56 <!-- interactiveMode
57 | This will determine whether maven prompts you when it needs input. If set to false,
58 | maven will use a sensible default value, perhaps based on some other setting, for
59 | the parameter in question.
60 |
61 | Default: true
62 <interactiveMode>true</interactiveMode>
63 -->
64

```

Step 18) Open Window/Reference on Eclipse.

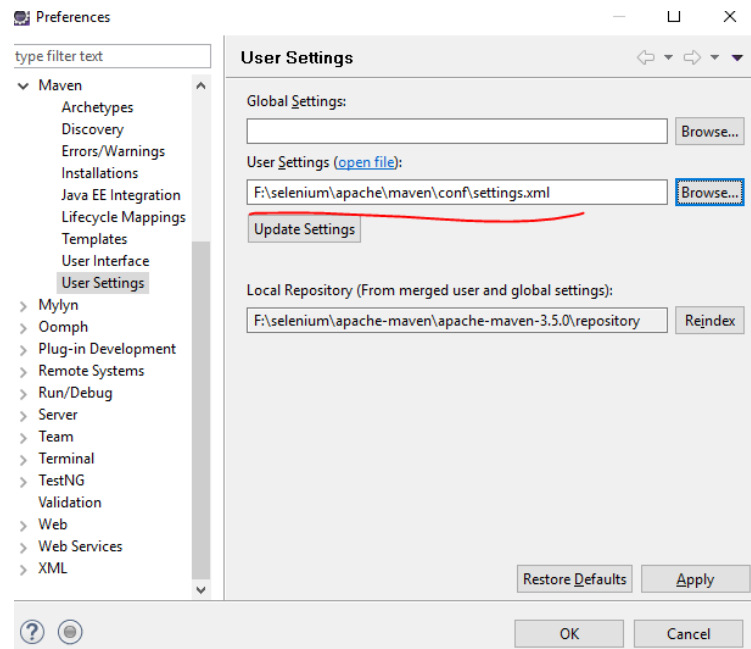
Don't be bored Take a break again continue To Next Page no:



- 19) Click on ADD
- 20) Enter the directory where maven is installed (This should be the MAVEN HOME)
- 21) Installation Name Will Auto populate Click on Finished
- 22) Select Apache Maven and click on Apply
- 23) Declare the position of the Maven configuration file which was changed last time.

Don't be bored Take a break again continue To Next Page no:

24)Click on User setting and upload the Setting XML file



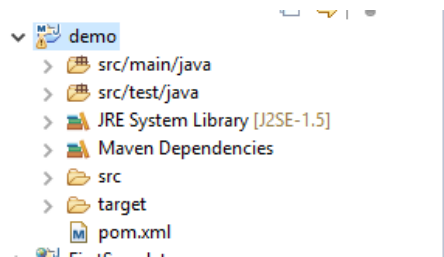
25)Click on Update and OK

26) Check Updating Dependency Maven. After updating update the project

Right Click on the Project ->Maven->Update Project

28)Below folder structure will Create

Don't be bored Take a break again continue To Next Page no:



Your Maven Installation is done. Now don't sits idle start working.

Over View of Maven Project

1)**prom.xml**: Impotent file in maven which will malintents all the dependency to the project all over the groups .and make the project in same dependency we use prom.xml file.

1.a) Adding Dependency to the project :

Steps

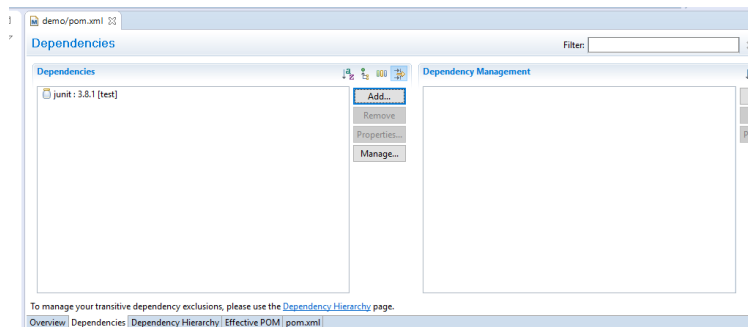
a) Open prom.xml

b) Navigate to dependency

Don't be bored

Take a break again continue

To Next Page no:



c)click on ADD

d)Enter Group ID and Artifact ID

Don't be bored Take a break again continue To Next Page no:

Select Dependency

Group Id: *

Artifact Id: *

Version: Scope: compile ▾

Enter groupId, artifactId or sha1 prefix or pattern (*):

⚠ Index downloads are disabled, search results may be incomplete.

Search Results:

✖ Artifact Id cannot be empty

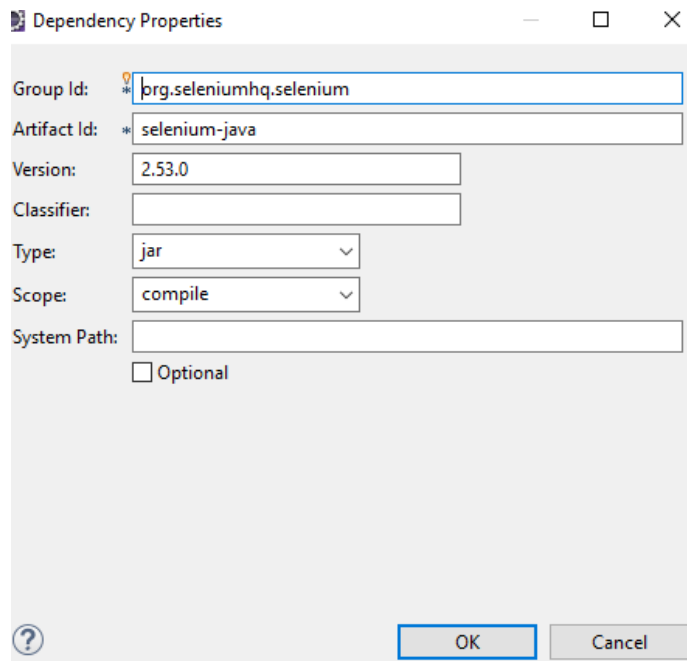
?

OK Cancel

Example: Adding Selenium Dependency

Search Dependency in google ()

Don't be bored Take a break again continue To Next Page no:



Dependency Properties

Group Id: * org.seleniumhq.selenium

Artifact Id: * selenium-java

Version: 2.53.0

Classifier:

Type: jar

Scope: compile

System Path:

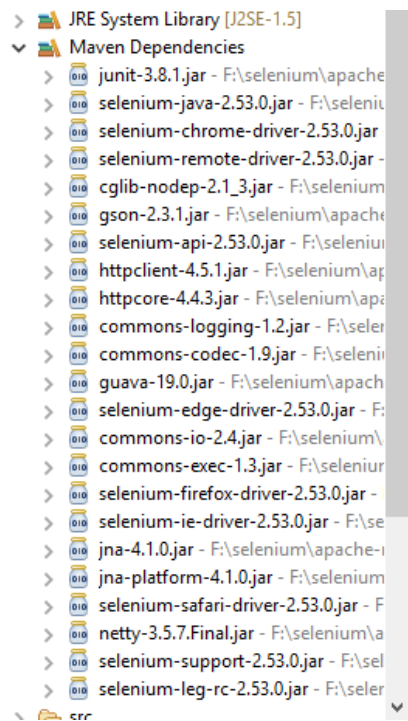
☐ Optional

OK Cancel

Click on OK and save the project. It will download selenium mentioned version

e) Check on Maven Dependency. It will add all the Jar it self

Don't be bored Take a break again continue To Next Page no:



2) **src/main/java**: Contains all the base java classes which will be called during the test case execution

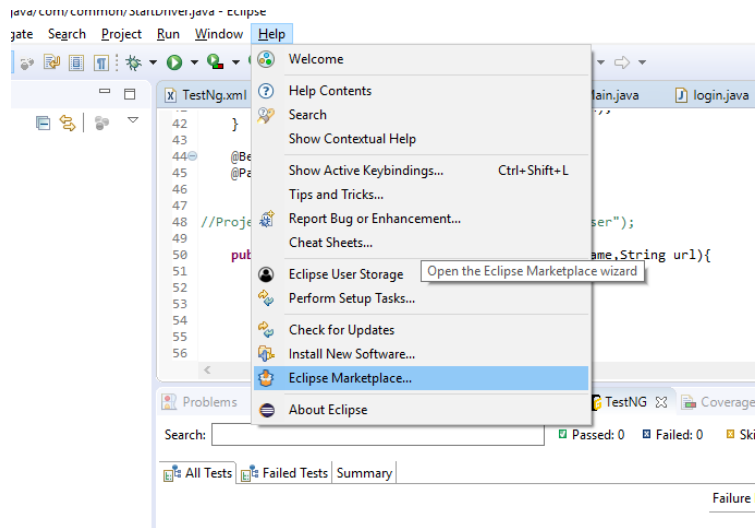
3) **src/test/java**: Contains all the Test classes for the automation

How to Install TestNg in Eclipse

Don't be bored Take a break again continue To Next Page no:

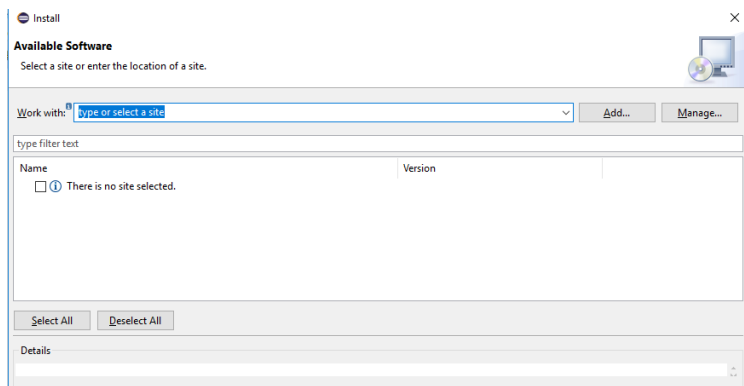
Step1)

Launch the Eclipse IDE and from Help menu, click “**Install New Software**”.

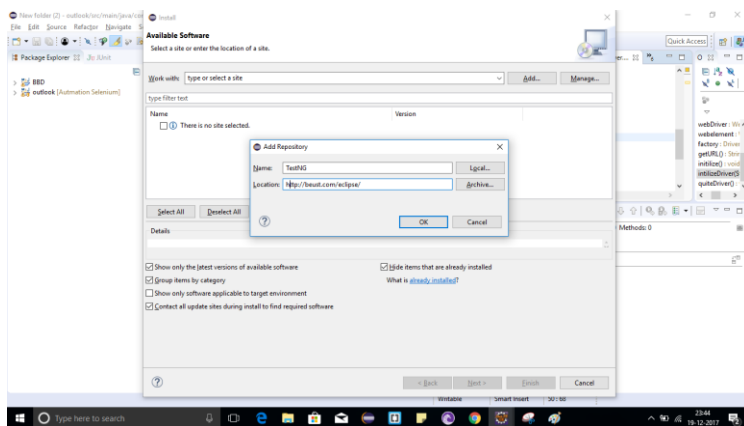


2) You will see a dialog window, click “**Add**” button.

Don't be bored Take a break again continue To Next Page no:

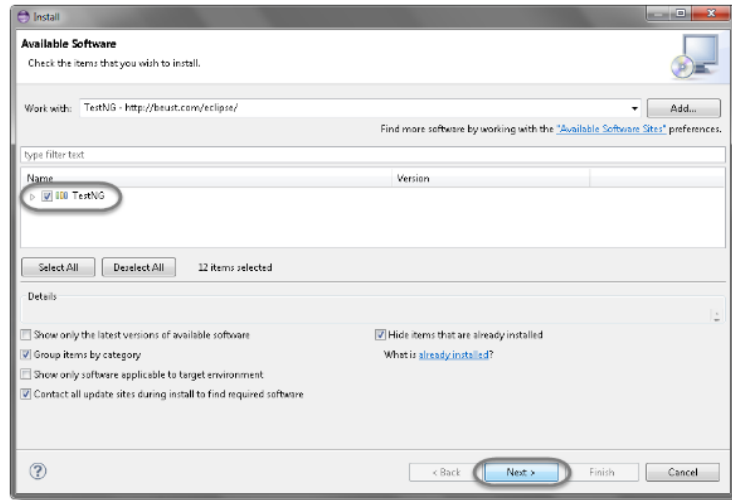


3) Type name as you wish, lets take **"TestNG"** and type **"http://beust.com/eclipse/"** as location. Click OK.



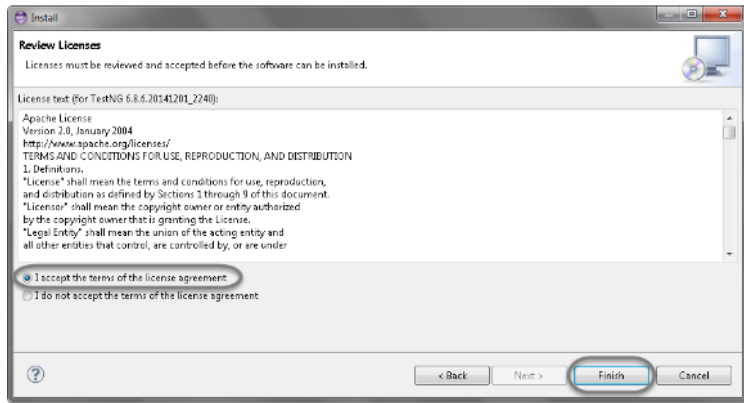
Don't be bored Take a break again continue To Next Page no:

4) You come back to the previous window but this time you must see TestNG option in the available software list. Just Click TestNG and press **"Next"** button.

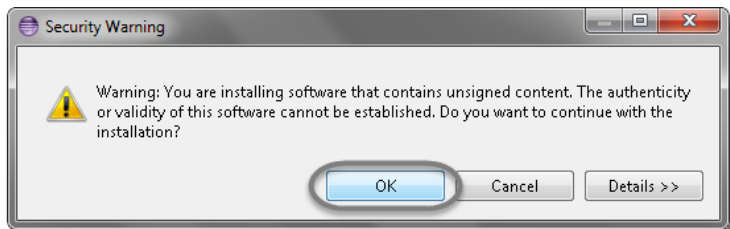


5) Click **"I accept the terms of the license agreement"** then click **Finish**.

Don't be bored Take a break again continue To Next Page no:



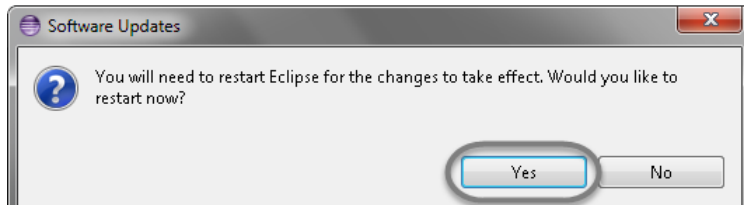
6) You may or may not encounter a Security warning, if in case you do just click **OK**.



7) Click **Next** again on the succeeding dialog box until it prompts you to Restart the Eclipse.

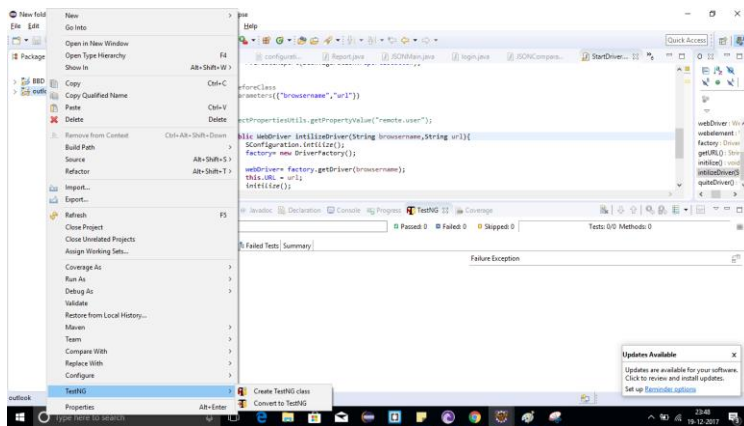
8) You are all done now, just Click **Yes**.

Don't be bored Take a break again continue To Next Page no:



9) Proceed with your workplace.

10) After restart, verify if TestNG was indeed successfully installed. Right click on you project and see if **TestNG** is displayed in the opened menu.



Selenium Framework

Don't be bored Take a break again continue To Next Page no:

Pre-Request: You have Installed Maven

1) Created Simple Maven Project In Eclipse (For Refence Got to Page -)

2) Add All the Dependency. Dependency are

A) Selenium-Java

B) Test Ng (<http://testng.org/doc/maven.html>)

C) Apache Poi (<https://mvnrepository.com/artifact/org.apache.poi/poi/3.16>)

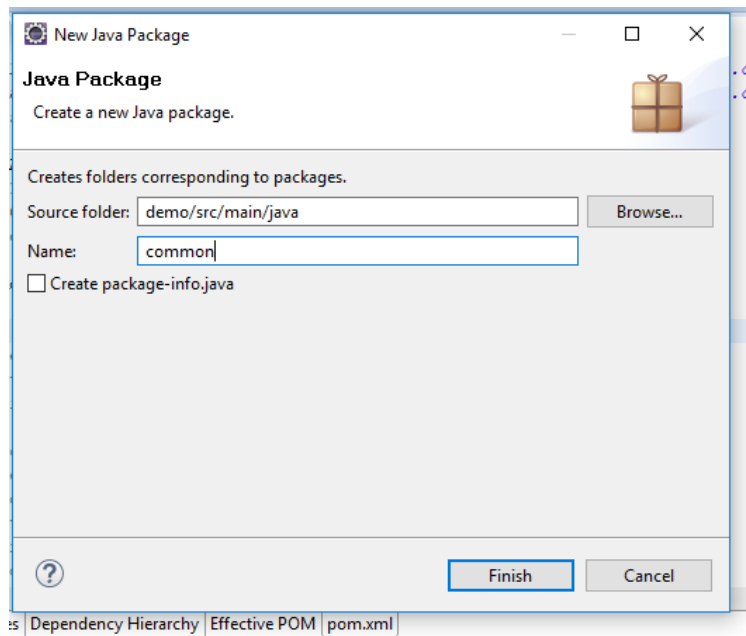
3) Create **src/main/java**: Contains all the base java classes which will be called during the teste case execution. Create all the common package which will be frequently and the methods which will be called during the any functionality

A) Creating a Common Package Name: common. Which will contain all the driver initiation,

Will Contains Class : DriverFactory, strartDriver, Constaints and BaseTest

Don't be bored Take a break again continue To Next Page no:

.



Class 1)

Constraint Class: Which will Store All the Constraints

Code:

/////

package common;

Don't be bored Take a break again continue To Next Page no:

```
public class Constraints {  
  
    public static final String IE_BROWSER = "iexplore";  
    public static final String FF_BROWSER = "Firefox";  
    public static final String CHROME_BROWSER = "chrome";  
    public static final String MISSING_PAGE_MSG = "page is missing";  
  
}
```

////

Class 2)

DriverFactory : Which Will contains and setup all the driver details

Note: As of now I don't have maven dependency for the Driver Binary location. Means we can add the driver in maven dependency later we can use it. Later will discuss on this. Currently I have downloaded my driver in my system and using the driver by setting the system **property**

Code :

////////

```
package common;
```

```
package common;
```

```
import org.openqa.selenium.WebDriver;
```

Don't be bored Take a break again continue To Next Page no:

.

Commented [MC1]: maven dependency for Driver need to give more details

```
import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.ie.InternetExplorerDriver;


public class DriverFactory {

    private static String BrowserDetails =null;

    public static String getBrowserDetails() {
        return BrowserDetails;
    }

    private WebDriver webDriver=null;

    public WebDriver getDriver(String driver){
        try{
            if(driver.equalsIgnoreCase(Constraints.CHROME_BROWSER)){
                System.setProperty("webdriver.chrome.driver", "F:\\selenium\\chromedriver_win32\\chromedriver.exe");

                webDriver= new ChromeDriver();
                BrowserDetails= Constraints.CHROME_BROWSER;
            }
        }
    }
}
```

Don't be bored Take a break again continue To Next Page no:

```
}  
else if (driver.equalsIgnoreCase(Constraints.FF_BROWSER)){  
  
System.setProperty("webdriver.gecko.driver", "F:\\selenium\\chromedriver_win32\\chromedriver.exe");  
  
    webDriver= new FirefoxDriver();  
    BrowserDetails= Constraints.FF_BROWSER;  
}  
  
else if (driver.equalsIgnoreCase(Constraints.IE_BROWSER)){  
    System.setProperty("webdriver.ie.driver", "F:\\selenium\\IEDriverServer_x64_2.52.0\\IEDriverServer.exe");  
  
    webDriver= new InternetExplorerDriver();  
    BrowserDetails= Constraints.IE_BROWSER;  
  
}  
  
else {
```

Don't be bored Take a break again continue To Next Page no:

```
System.setProperty("webdriver.gecko.driver","F:\\selenium\\geckodriver\\geckodriver.exe");
```

```
webDriver= new FirefoxDriver();
```

```
BrowserDetails= Constraints.FF_BROWSER;
```

```
}
```

```
return webDriver;
```

```
}
```

```
catch(Exception a){
```

```
System.out.println("Error oocured in Driver factory ");
```

```
System.out.println(a.getMessage());
```

```
return null;
```

```
}
```

```
}
```

```
}
```

```
///
```

Don't be bored Take a break again continue To Next Page no:

.

Class 3

StartDriver: This is a @BeforeClass means it will execute before the test cases start execution. It will take the browser and the URL and initiate the driver and open the page. It will be called by each test case to initiate the precondition.

It contains also @Afterclass, means it will execute the post condition after the test. Currently it is having only closing the browsers.

Code:

```
package common;
```

```
import org.openqa.selenium.WebDriver;
```

```
import org.testng.annotations.AfterClass;
```

```
import org.testng.annotations.BeforeClass;
```

```
import org.testng.annotations.Parameters;
```

```
public class StartDriver {
```

```
    private DriverFactory driver=null;
```

```
    protected WebDriver webDriver=null;
```

Don't be bored Take a break again continue To Next Page no:

.

```
public WebDriver getWebDriver() {  
    return webDriver;  
}  
  
protected String URL=null;  
  
public String getURL() {  
    return URL;  
}  
  
public void setURL(String uRL) {  
    URL = uRL;  
}  
  
@BeforeClass  
@Parameters({"browsername","url"})  
  
public void IntiateDriver(String browsername,String url){  
    System.out.println("Entering the Initiate Driver method ");  
    setURL(url);  
    driver =new DriverFactory();  
    webDriver=driver.getDriver(browsername);  
    webDriver.navigate().to(getURL());  
}
```

Don't be bored Take a break again continue To Next Page no:

```

        System.out.println("Driver is initiated "+webDriver.getTitle());
    }

    @AfterClass
    public void QuitDriver(){
        webDriver.close();
        webDriver.quit();

    }

}

```

Class 4

Base: It is base class for each and every object method present in java folder under src .it will initiate he XPATHs for all the page. Its having all the common methods which will be calling frequently by the methods for performing operation.main methods is **initlize()** which will initilize the XPATH file

Code:

```
package common;
```

```
import java.io.FileInputStream;
```

```
import java.io.FileNotFoundException;
```

```
import java.io.IOException;
```

Don't be bored Take a break again continue To Next Page no:

.


```
import java.util.Properties;
```

```
import org.openqa.selenium.By;
```

```
import org.openqa.selenium.WebDriver;
```

```
import org.openqa.selenium.WebElement;
```

```
import org.openqa.selenium.support.ui.ExpectedConditions;
```

```
import org.openqa.selenium.support.ui.WebDriverWait;
```

```
import org.testng.Assert;
```

```
public class Base {
```

```
    public Base(WebDriver driver, String url){
```

```
        setWebdriver(driver);
```

```
        setUrl(url);
```

```
        initlize();
```

```
    }
```

```
    protected WebElement webelement;
```

Don't be bored

Take a break again continue

To Next Page no:

.

```
protected Properties openpage;
```

```
public WebElement getWebelement() {  
    return webelement;  
}
```

```
/*public void setWebelement(WebElement webelement) {  
    this.webelement = webelement;  
}*/
```

```
protected static String url;
```

```
    public String getUrl() {  
        return url;  
    }
```

```
public void setUrl(String url) {  
    this.url = url;  
}
```

```
protected WebDriver webdriver;
```

Don't be bored Take a break again continue To Next Page no:

.

```

public WebDriver getWebdriver() {
    return webdriver;
}

public void setWebdriver(WebDriver webdriver) {
    this.webdriver = webdriver;
}

//*****
//Intilize property file
public void initlize(){

    try {
        openpage = new Properties();
        //openpage.load(new
FileInputStream(System.getProperty("F:\\selenium\\AutomationWorkspeace\\demo\\src\\test\\java\\XPath\\openPage.properties")));
        openpage.load(new FileInputStream(System.getProperty("user.dir").concat("/src/test/java/xpth/openPage.properties")));

    } catch (FileNotFoundException e) {
        System.out.println("Error occured in initilize method");
        System.out.println(e.getMessage());
    }
}

```

Don't be bored Take a break again continue To Next Page no:

```
} catch (IOException e) {  
    // TODO Auto-generated catch block  
    System.out.println("Error occured in initilize method");  
    System.out.println(e.getMessage());  
}  
  
}  
  
  
//*****  
  
// All common methods  
  
  
public void waitforElimentIsDisplay(String XPATH){  
    for (int seceond=0;seceond <=30;seceond++){  
  
        if(isPresent(By.xpath(XPATH),1)){  
            break;  
  
        }  
        else {
```

Don't be bored Take a break again continue To Next Page no:

```
        System.out.println("ELEMENT is Not present ");
        System.out.println("STuck in waitforElimentIsDisplay");
    }
}

}

public boolean isPresent(By by,int waitTime ){
    try{
        WebDriverWait wait = new WebDriverWait(webdriver,waitTime);
        wait.until(ExpectedConditions.visibilityOfElementLocated(by));
        webdriver.findElement(by).isDisplayed();

        return true;
    }
    catch(Exception e){
        System.out.println("Problem is in Preseent method");
        System.out.println(e.getMessage());
        return false;
    }
}
```

Don't be bored Take a break again continue To Next Page no:

```
}  
  
public WebElement getObject(String XPATH){  
  
    try {  
        return webdriver.findElement(By.xpath(XPATH));  
    } catch (Exception e) {  
        // TODO: handle exception  
  
        System.out.println("Error occurred in getObject");  
        System.out.println(e.getMessage());  
        return null;  
  
    }  
}
```

```
public void clickObject(String XPATH){  
    waitForElementIsDisplay(XPATH);  
    webelement= getObject(XPATH);  
    webelement.click();
```

Don't be bored Take a break again continue To Next Page no:

.

```
}
```

```
public boolean insertText(String XPATH,String strText){
```

```
try {
```

```
    waitForElimentIsDisplay(XPATH);
```

```
    webelement = getObejct(XPATH);
```

```
    if(webelement.isEnabled() && webelement.isDisplayed()){
```

```
        if(strText.isEmpty()){
```

```
            return true;
```

```
        }
```

```
    else{
```

```
        webelement.clear();
```

```
        webelement.sendKeys(strText);
```

```
        Thread.sleep(300);
```

```
        return true;
```

```
    }
```

Don't be bored

Take a break again continue

To Next Page no:

.

```
    }  
    else{  
  
        return false;  
    }  
  
} catch (Exception e) {  
    // TODO: handle exception  
  
    System.out.println("Error occurred in INsert method");  
    System.out.println(e.getMessage());  
  
    return false;  
}  
  
}  
  
/*public boolean asert(String value,String actual,String expected ){  
  
try {  
    switch(value){  
  
        Don't be bored      Take a break again continue      To Next Page no:  
        .
```



```
case "compareText":
    Assert.assertEquals(actual, expected);

    break;
case "containsTextTrue":
    Assert.assertTrue(isPresent(By.xpath(expected)));

    break;
}
} catch (Exception e) {
    // TODO: handle exception
}

}

*/

}

//*****END of Common package*****//
```

Don't be bored Take a break again continue To Next Page no:

.

4)

TestNg.xml: It is a test Ng xml which contains all the test cases which required to be run. Not only test cases you can call the multiple test suits which can contains test cases as well. It contains the base parameter like browser.URL and etc. The main function is it will search test case mention in the test case parameter in the src/test/java and contains the @test annotation. And before the Strat execute the Test class it will search @Beforeclasss and it will execute that first. After running the test cases, it will search for #AfterTest class and will run that class as a post condition

TextNg.xml code below (reminded that spelling is correct)

```
<suite name="basic">
<parameter name="browsername" value="iexplore"/>
<parameter name="url" value="http://www.google.com/" />
<test name ="openbrowser">
<classes>
<class name="common.Openpage"/>
</classes>
</test>
</suite>
```

Don't be bored Take a break again continue To Next Page no:

.

*****END OF TESTNG*****

Src/test/java: the folder contains all the test case class by model wise. Each test cases which needs to be run in testng.xml will be here with @Test annotation. And each class will extends the startDriver class to execute the preconditions and post conditions. And it will contains all the XPATH for each page (you can places and groups as a different folder).In each test method will have @Test annotation

Example given: Test cases written for Open Gmail from google starting page

Code:

```
package common;
```

```
import org.openqa.selenium.By;
```

```
import org.testng.annotations.Test;
```

```
import Gmail.basic;
```

```
public class Openpage extends StartDriver{
```

```
    public Openpage(){
```

```
        super();
```

```
    }
```

Don't be bored Take a break again continue To Next Page no:

.

@Test

```
public void TC_01(){
```

```
    Gmail.basic login = new basic(super.webDriver, URL);
```

```
    System.out.println("print");
```

```
    if(login.openGmail()){
```

```
        System.out.println("passed");
```

```
    }
```

```
    else{
```

```
        System.out.println("failed");
```

```
    }
```

```
}
```

```
/*public void openpage() {
```

```
    System.out.println("print");
```

Don't be bored Take a break again continue To Next Page no:

```
Base test= new Base(super.webDriver, URL);  
test.getWebdriver().findElement(By.linkText("Gmail")).click();
```

```
}*/
```

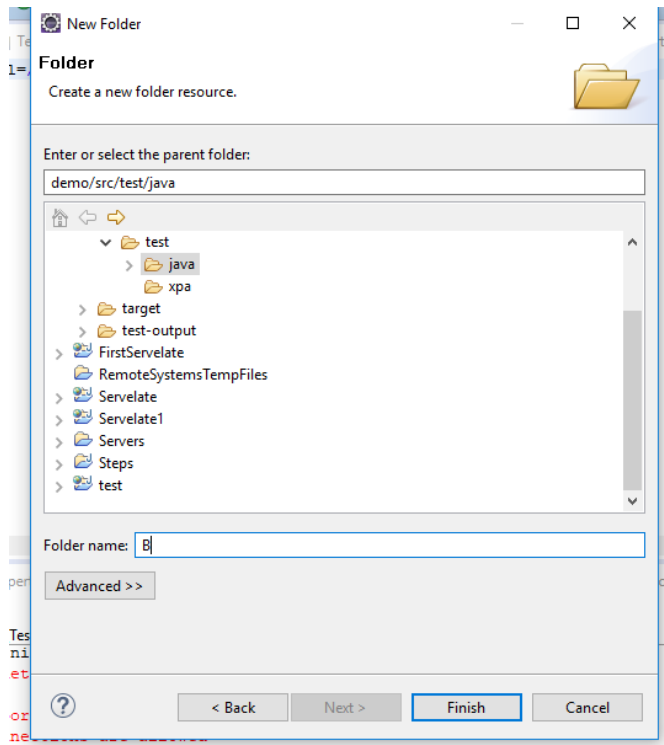
```
}
```

5)XPATH: Write Xpath for each element present in page.To take XPATH from the page follow the steps

1) Crate folder under src/test/jave

A) Right click on src/test/jave->New->Other->search folder->

Don't be bored Take a break again continue To Next Page no:



B) Select Normal folder

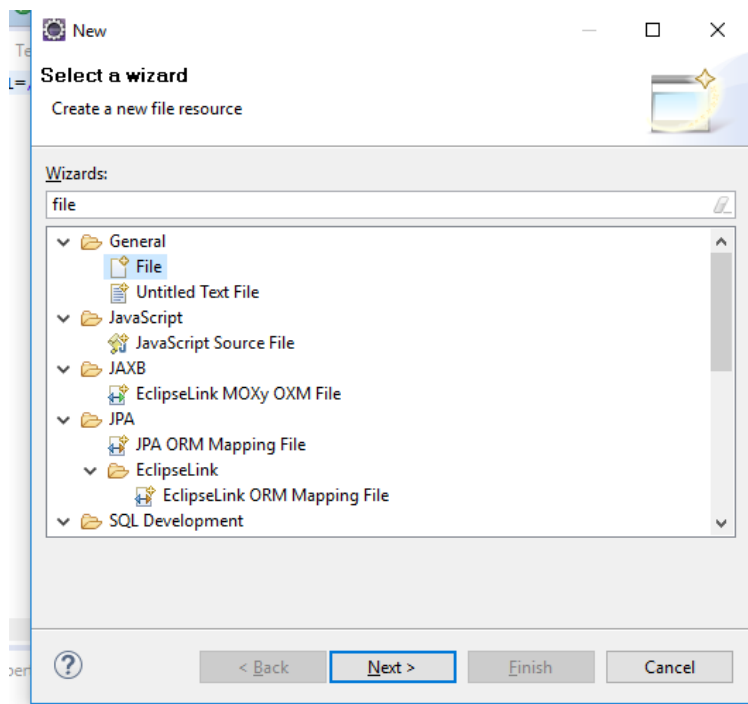
C) Click on next and enter name

D) Click finish

2) Create a property file

A) Right Click on newly created folder -> New -> others -> file -> Select File

Don't be bored Take a break again continue To Next Page no:



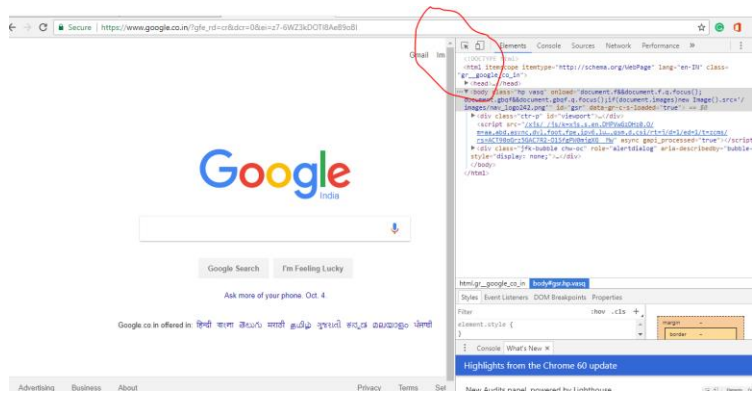
B) Enter name like Openpage.properties

3) Create XPATH on that properties file

A) open the page (<https://www.google.co.in/>) in chrome

Don't be bored Take a break again continue To Next Page no:

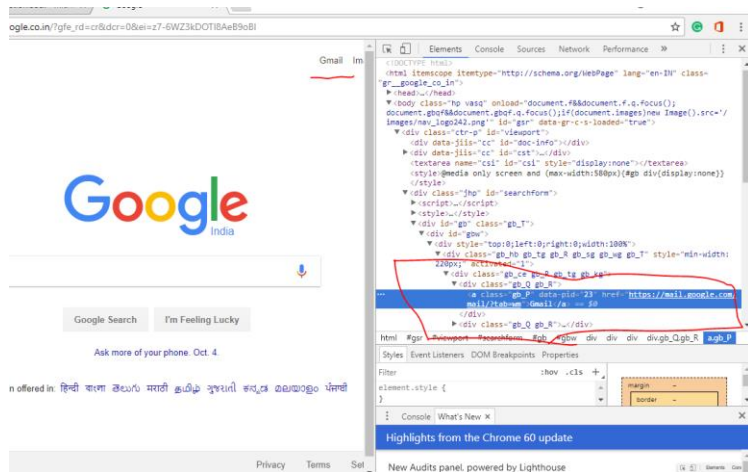
B) Click F12 for developer type



D) Click on inspect element

E) over the mouse on the page for which you want to take the XPATH (Example Gmail)

Don't be bored Take a break again continue To Next Page no:



F) go to the Element section, the selected element will show all the details

G) Right Click on it you will get the **Copy** ->**copy XPath**

H) In Property file create a variable and past the XPATH

Sample code for XPATH

click_gmail=//*[@id="gbw"]/div/div/div[1]/div[1]/a

Don't be bored Take a break again continue To Next Page no: