Selenium Notes by Ajit Sir

Qpsiders - btm bangalore

Ph No - 9980600900

1. ***What is Selenium ?***

Selenium is an open source web automation tool.

Limitation of Selenium :

- It doesn't support windows based application directly. However, third party tool **(eg: AutoIt)** can be integrated with selenium to automate windows based applications.

Note :

1. Selenium community developed specific tool called **WINIUM** to automate windows based applications.
2. Selenium community also developed tools to test mobile applications,
    - **Selendroid** - it supports only Android platform
    - **Appium** - it supports Android platform, MAC, Windows etc.

Note :

All the selenium related resources and documents can be found on the below website.

http://www.seleniumhq.org

Here, hq stands for head quarter.

2***. Why Selenium is so popular and demanding ?***

Selenium is popular and demanding due to the following features.

1. it is an open source tool freely available on internet
2. No project cost involved
3. No licence required
4. Can be easily customized to integrate with other Test Management tools like ALM, Bugzilla etc.
5. It supports almost 13 different software languages
    - Java

- C#
- Ruby
- Python
- Perl
- Php
- Javascript
- Javascript (Node JS)
- Haskell
- R
- Dart
- TCL
- Objective - C

6. It supports almost all the browsers.(Firefox, Chrome, Internet Explorer etc) and hence, cross browser testing/compatibility testing can be performed using selenium.
7. It supports almost all the Operating System (MAC, Windows, LINUX etc) and hence, cross platform testing can also be performed.

---

3. What are the different flavours of Selenium ?

- **Selenium Core** (Developed by a company called **Thought Works** way back in 2004)
- **Selenium IDE** (supports only Mozilla Firefox - supports record and playback feature)
- **Selenium RC** (Remote Control - Version is 1.x) (Used for parallel execution of automation scripts on multiple remote systems)
- **Selenium WebDriver** (Version is 2.x and 3.x)

Note :

Selenium WebDriver version 3.x is no longer capable of running Selenium RC directly, rather it does through emulation and via an interface called WebDriverBackedSelenium.

But, **it does support Selenium Grid directly.**

Selenium Grid :

1. It is one of the component of selenium that is used to run automation scripts on multiple system simultaneously.

2. It is used to carry out compatibility testing on multiple browsers and platforms.

---

4. What are the key/Important topics of Selenium ?

- **Automation Framework -** guidelines and rules to write selenium code
- **GitHub** - Central Repository to store code
- **Maven** - build dependency tool for auto update of selenium version
- **Selenium Grid** - to test on multiple OS and browsers
- **Jenkins** - Continuous Integration
- T**estNG** - framework for generation of Test Reports and running multiple test scripts in one go

---

5. What are the Softwares required for Selenium ?

1. **Eclipse IDE** - Oxygen (Stable version)
2. **JDK 1.8**
3. **Selenium Server-Standalone**-3.7.1 (Stable version)

   (Download it from the given url : ***http://www.seleniumhq.org/download****)*

4. **Driver Executables**

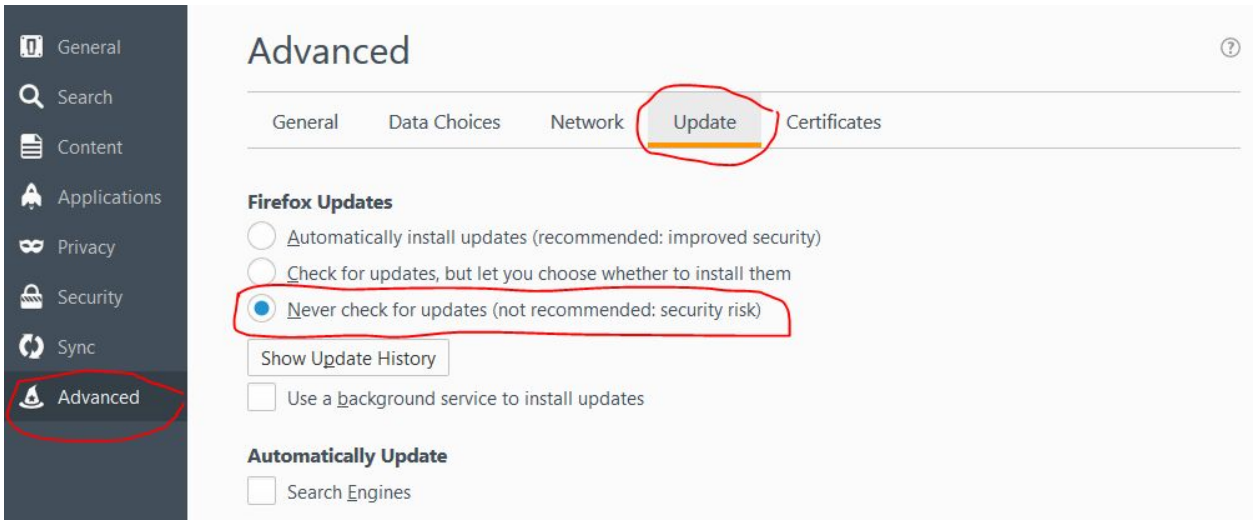
   - For Firefox Browser
     - the name of the driver executable is : **geckodriver.exe**
     - Url to download : https://github.com/mozilla/geckodriver/releases
     - Version **0.19** is recommended for firefox browser with version 56.0 (selenium jar - 3.7.1)
   - For Chrome brow
     - the name of the driver executable is : **chromedriver.exe**
     - Url to download : https://chromedriver.storage.googleapis.com/index.html?path=2.31/
     - Stable version of chrome version is 62.0 (Use chromedriver.exe with version 2.33)

5. Browsers:

Firefox (Version 57.0)

Chrome (Version 62.0)

Note : To stop auto update of firefox browser version, Make sure to disconnect the internet connection and then install 54.0 version, now go to Setting/Option in firefox browser and check the below checkbox - Never check for updates.



6. Application Under Test (AUT)

Application Name : actiTIME

Online url : https://demo.actitime.com/login.do

Offline url : https://localhost:8080/login.do

To download actiTIME application ,

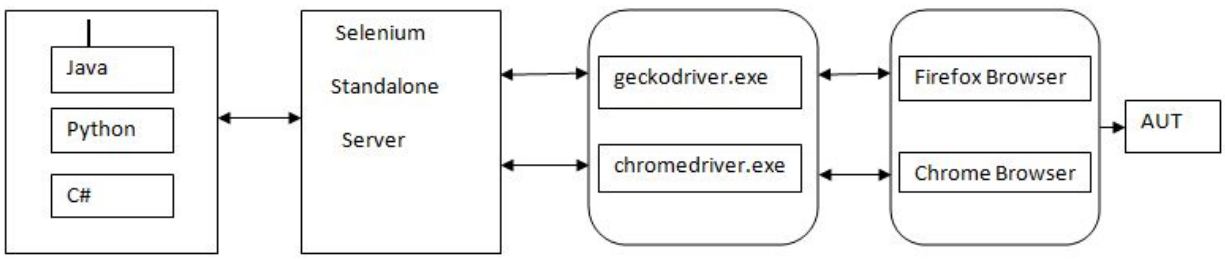
https://www.actitime.com/download.php

---

6. Selenium Architecture - High Level ?

OR

How selenium performs automation testing on browser ?



Client Binding **Driver Executables** **Browser**

OR **.jar file** **uses**

Language Binding **JSON Wire Protocol**

1. **Since selenium supports multiple languages such as Java, Python, C# etc, we can develop automation scripts in all the supported languages. This is known as language binding or Client binding.**
2. **When we execute the selenium code, request goes to the Selenium Standalone Server (also known as Selenium WebDriver Server), which further process the request based on the input received from the client binding and perform specific actions on the respective browsers using the browser specific driver executables,**

   Eg : geckodriver.exe for firefox browser and

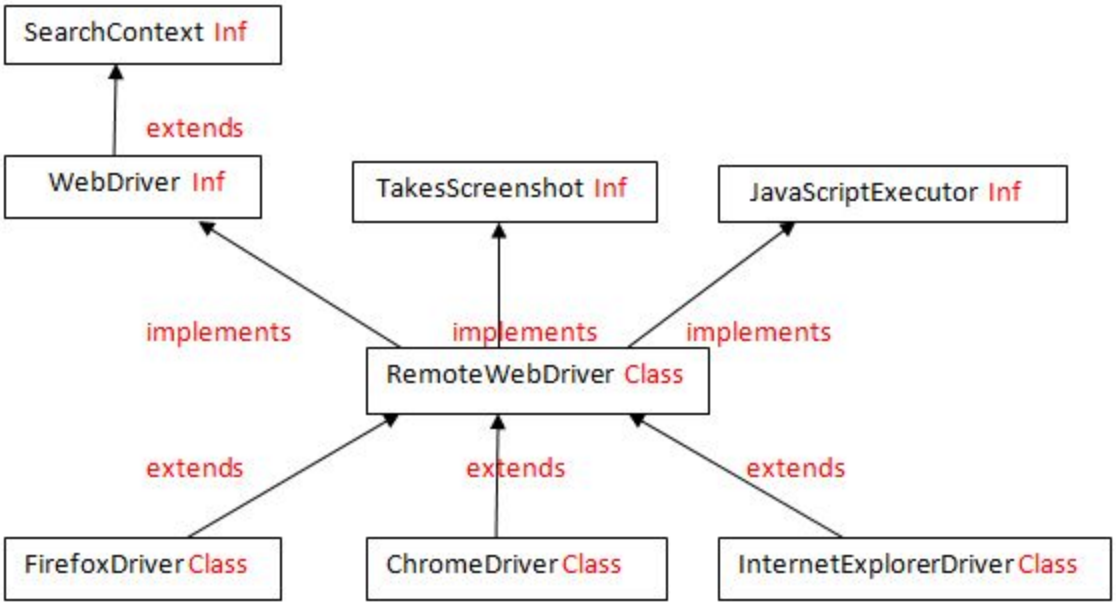
   chromedriver.exe for chrome browser and so on…

3. **Driver executables uses a protocol called JSON Wire protocol to communicate with related**

browsers. (JSON stands for Java Script Object Notation)

---

7. *Selenium Java Architecture - Detailed Level*



1. **SearchContext is the supermost interface present in selenium webdriver.**
2. **An interface called WebDriver extends SearchContext interface.**
3. **A total of 13 interfaces are available in selenium, which is implemented by a super most class called RemoteWebDriver**
4. **RemoteWebDriver is again extended by few browser specific child classes such as,**
    - **FirefoxDriver class to automate on firefox browser.**
    - **ChromeDriver class to automate on Chrome browser,**
    - **InternetExplorerDriver class to automate on IE and so on…….**

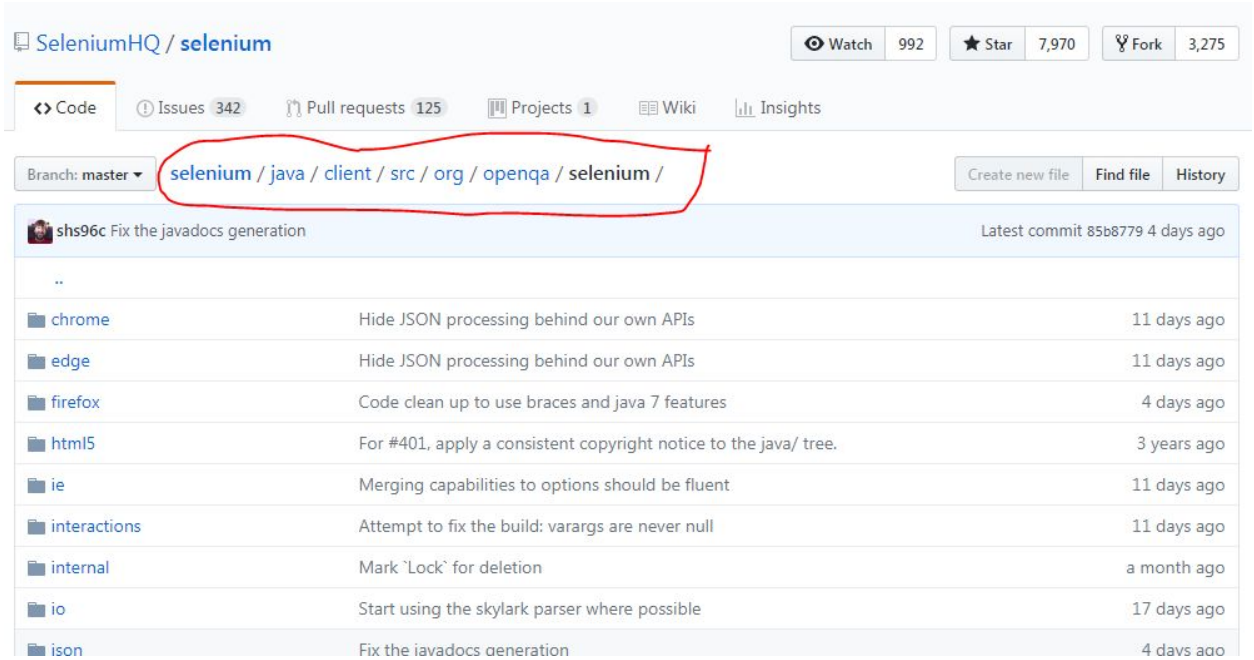
NOTE :

All the above mentioned **interfaces and classes** are present in a package called **"org.openqa.selenium".**

To view any information about Selenium interfaces, classes and methods, navigate to the below page.

https://github.com/SeleniumHQ/selenium/tree/master/java/client/src/org/openqa/selenium

Highlighted below in red is the navigation path.



---

8. *List down all the methods present in below interfaces of Selenium WebDriver.*

Methods of SearchContext interface :

1. **findElement()**
2. **findElements()**

Methods of WebDriver interface :

1. **close()**
2. **get()**
3. **getTitle()**
4. **getPageSource()**
5. **getCurrentUrl()**
6. **getWindowHandle()**

7. **getWindowHandles()**
8. **manage()**
9. **navigate()**
10. **quit()**
11. **switchTo()**

Methods of TakesScreenshot interface :

1. **getScreenshotAs(args)**

Methods of JavascriptExecutor interface :

1. **executeScript()**
2. **executeAsyncScript() - we don't use this for automation**

Methods of WebElement interface :

1. **clear()**

clear() --

i. clear() is a method present in WebElement interface.

ii. It is used to clear any value which is present in any element (eg: text box,text area etc)

2. **click()**
3. **getAttribute():-**

i. It is used to the get the value of the attribute in the form of string.

ii. As an argument to this method we pass the attribute name in the form of string.

ii. When we pass an attribute name which is not present in the html source code of an element as an argument to this method ,it returns an empty " "string.

4. **getCssValue()**
5. **getLocation()**

getLocation() method :

i. getLocation() is a method present in WebElement interface.

ii. It returns an instance of Point class.

iii. Point class has few non static methods like getX() and getY().

4. getX() method returns the coordinate of an element.

5. getY() method returns the y coordinate of an element.

6. **getRect()**
7. **getSize()**

1. getSize() is a method present in WebElement interface.

2. It returns an instance of Dimension class.

3. Dimension class has few non static methods like getHeight() and getWidth().

4. getHeight() method returns the height of an element.

5. getWidth() method returns the width of an element.

8. **getTagName() :-**

i. It is use to get the tagname of an element. It doesn't accept any argument.

9. **getText() :-**

i. It is use to get the text present in an element. It doesn't accept any argument.

ii. When there is no text present in the html source code of an element and we are using getText() method to retrieve the text of an element, it returns an empty string.

10. **isDisplayed()**

isDisplayed() -

1. isDisplayed() is a method present in WebElement interface.

2. It checks whether an element is present or not on the webpage

3. If the element is present on the webpage, it returns true.

4. And if the element is not present on the webpage, it returns false.

11. isEnabled()

isEnabled() -

1. isEnabled() is a method present in WebElement interface.

2. It checks whether an element is enabled or not on the webpage.

3. If the element is enabled on the webpage, it returns true.

4. And if the element is disabled on the webpage, it returns false.

12. isSelected()

isSelected() -

1. isSelected() is a method present in WebElement interface.

2. It checks whether an element is selected or not on the webpage.

3. If the element is selected on the webpage, it returns true.

4. And if the element is not selected on the webpage, it returns false.

13. sendKeys()

sendKeys() --

1. sendkeys() is a method present in WebElement interface.

2. It is used to enter any value in an element (eg: text box,text area etc)

14. submit()

We can use submit method to click on an element if the element is present inside a form(tag) and html source code of the element has an attribute called type = "submit". When both the conditions satisfied we use submit() method.

In Selenium we have total 13 interfaces. All these interfaces has abstract and non- static methods .

The following are the interfaces of Selenium webdriver.

1. **SearchContext**
2. **WebDriver**
3. **TakesScreenshot**
4. **JavascriptExecutor**
5. **Navigation**
6. **OutputType**
7. **WebElement**
8. **TargetLocator**
9. **Alert**
10. **Action**
11. **ExpectedConditions**
12. **Options**
13. **Timeouts**

9. *Why we upcast the browser related child class to WebDriver, and not RemoteWebDriver class (RemoteWebDriver being the super most class in selenium) ?*

Upcasting Example :

WebDriver driver = new FirefoxDriver();

- Converting a child class object to super type is called Upcasting.

- In selenium, we use upcasting so that we can execute the same script on any browser.
- In selenium, we can upcast browser object to RemoteWebDriver, WebDriver, TakesScreenshot , JavascriptExecutor etc, but a standard practice is to upcast to WebDriver interface.
- This is as per the Selenium coding standard set by the Selenium community. As a testimonial, navigate to the below selenium community site and check for the text as mentioned in the image below.

Url - http://www.seleniumhq.org/projects/webdriver/

---

10. Where did you use Upcasting in Selenium ?

WebDriver driver = new FirefoxDriver();

Explain the above statement..

1. WebDriver is an interface in Selenium that extends the supermost interface called SearchContext.
2. driver is the upcasted object or WebDriver interface reference variable.
3. " = " is an assignment operator.
4. new is a keyword using which object of the FirefoxDriver class is created.
5. FirefoxDriver() is the constructor of FirefoxDriver class which initialises the object and it will also launch the firefox browser.

---

11. Steps to install/integrate selenium server to the java project

1. Launch eclipse and go to package explorer [navigation path :- Window menu ⟶ Show View ⟶ Package Explorer]

2. Create a java project [File → New→ Java Project]
3. Right click on Java Project and add a new folder with name “driver” [File → New→ Folder]
4. copy **geckodriver.exe** file from your system and paste it into this driver folder
5. Similarly, create another folder with name **“jar”**and copy **Selenium Standalone Server.jar** file into this jar folder.
6. Expand the jar folder and right click on **Selenium Standalone Server.jar** file → select **Build Path** → select **Add to Build Path**
7. As soon as you add any .jar files to build path, a new folder will be available called **“Reference Libraries”** under the package explorer section and you can see the .jar file is added to this **“Reference Libraries”**
8. To remove the .jar file from the java build path, go to the Reference Libraries → select the .jar file → right click → select build path → Remove from build path.
9. Other way of adding .jar file to java build path is : right click on the project → build path → configure build path → Libraries tab → Add External jars → select the .jar file → Apply → ok

Interview Question:

How do you install selenium?

Selenium is an open source web automation tool that comes in the form of **.jar file**, we download this and attach it to the build path.

Selenium in order to communicate with browser it needs some extra file known as **driver executables**, we download this and set the path using System.setProperty(key, value)

That’s how we integrate or install Selenium.

How do you set the path of the driver executable ?

By using setProperty() method of System class,which takes 2 arguments, key and value.

As part of the value, we specify the path of the driver executable

and as part of key, we pass predefined key set by selenium community.

Line of code is ,

System.setProperty("webdriver.gecko.driver","path of geckodriver.exe")

System.setProperty("webdriver.chrome.driver","path of chromedriver.exe")

System.setProperty("webdriver.ie.driver","path of IEDriverServer.exe")

when do you get IllegalStateException ?

When Selenium Server try to communicate with the browser without using the relevant driver executables, we get this exception. In earlier version of selenium i.e 1.x and 2.x selenium server was able to perform automation without using driver executables, but 3.x series of selenium can't perform automation without relevant driver executable.

What is Method chaining ?

Calling a method on a reference of either a class or an interface, whose instance is returned by the immediate previous method is called method chaining.

eg. driver.navigate().to() :-

navigate is a method present in webdriver interface which returns an instance(reference) of Navigation interface, (Navigation is an interface of selenium),with this reference we call to() method which is present in Navigation interface, to() method is used to enter the url of any page, since it doesn't return anything so it is void.

12*. This program demonstrates Upcasting concept (FirefoxDriver class object to WebDriver interface) and accessing various methods of WebDriver interface*

package qspiders;

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
public class UpcastingToWebDriver_LaunchBrowser {
   public static void main(String[] args) throws InterruptedException  {
         //setting the path of the gecko driver executable
         System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");
         //Launch the firefox browser
         WebDriver driver = new FirefoxDriver();
         //Enter the url
         driver.get("http://www.google.com");
         //Get the title of the google page and print it on the console
         String title = driver.getTitle();
         System.out.println("the title of the page is :"+ title);
         //Get the URL of the google page and print it on the console
         String currentUrl = driver.getCurrentUrl();
         System.out.println("the URL of the page is :"+ currentUrl);
         //Get the source code  of the google page and print it on the console
         String pageSource = driver.getPageSource();
         System.out.println("the source code  of the page is :"+ pageSource);
         //Halt the program execution for 2 seconds
         Thread.sleep(2000);
         // Close the browser
         driver.close();
          }
```

}

****

Capturing Screenshot

****

Question : **How to capture screenshots in Selenium ?**

Answer : **We capture screenshots in Selenium using *getScreenshotAs()* method of *TakesScreenshot* interface.**

Steps to take screenshot:

1. Create an object of specific browser related class (eg : FirefoxDriver) and then upcast it to WebDriver object (eg : driver)
2. Typecast the same upcasted driver object to TakesScreenshot interface type.
3. Using the type casted object, we call getScreenshotAs(OutputType.FILE) which in turn returns the source file object.
4. Using the File IO operations (i.e FileUtils class), we store the screenshots to desired location in the project.

Program for Screenshot

```
package testpackage;
 import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.firefox.FirefoxDriver;

public class BaseClass {
```

```
	static {

		System.setProperty("webdriver.gecko.driver",
"./drivers/geckodriver.exe");
		System.setProperty("webdriver.chrome.driver",
"./drivers/chromedriver.exe");
		System.setProperty("webdriver.ie.driver",
"./drivers/IEDriverServer.exe");
}

	//public static WebDriver driver = new FirefoxDriver();
	public static WebDriver driver = new ChromeDriver();



}


package testpackage;
import java.io.File;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import com.google.common.io.Files;

public class Screenshot extends BaseClass {

	public static void main(String[] args) throws Exception {
```

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

                TakesScreenshot ts = (TakesScreenshot) driver;

                File srcFile = ts.getScreenshotAs(OutputType.FILE);

                File destFile = new File("F:/SCREENSHOT/Amazon.png");

                Files.copy(srcFile, destFile);

                driver.close();
        }
    }
```

How do we take screenshot of a page ?

We take the screenshot of a page by using getScreenshotAs() method of TakesScreenshot interface. It is a non- static method so we need a reference of TakesScreenshot interface to call it. There is no such method which returns an instance of TakesScreenshot interface, so we typecast our driver object to TakesScreenshot interface and using this object reference (ts) we call getScreenshotAs() method. As an argument to this method we pass (OutputType.File) to which format of file we want the screenshot type i.e File type. So, we specify the output type as File.

File srcFile = ts.getScreenshotAs(OutputType.FILE);

This line of code will store the screenshot in some temporary files or folder location, which is not a recommended place to store the file. Whenever the system becomes slow we tend to delete all temporary files and folders and if we store the screenshot in temporary location, we may lose the screenshot, we don't take screenshot to loose it so we want to store the

screenshot in some desired location based on our choice. We do this by creating an object of File class and as an argument to the constructor we pass the location (path) where we want to store the screenshot.

```
File destFile = new File("F:/SCREENSHOT/Amazon.png");
```

This line of code will create a blank file, it will not have the screenshot of the particular page. We copy the screenshot from temporary location and paste in our desired location. We do this by using the following line of code.

```
Files.copy(srcFile, destFile);
```

Files is a class from java, it has a static method 'copy' as an argument to this copy method we pass two file objects(srcFile, destFile). This method will copy the content of the screenshot from the temporary location(srcFile) and paste in the desired location(destFile). This is how we take the screenshot of a page.

Selenium Code :

```
package pack1;
import java.io.File;
import java.io.IOException;
import java.util.Date;
import org.apache.commons.io.FileUtils;
import org.openqa.selenium.OutputType;
import org.openqa.selenium.TakesScreenshot;

public class CaptureScreenshot_ActiTIMEPage extends BaseClass{
	public static void main(String[] args) throws IOException {
		//Creating an object of Date class
		Date d = new Date();
		//Printing the actual date
```

```
String date1 = d.toString();
System.out.println(date1);
//replacing the colon present in the date timestamp format to "_" using
replaceAll()
//method of String class
String date2 = date1.replaceAll(":", "_");
System.out.println(date2);
//Enter the url
driver.get("https://localhost:8080/login.do");

//Typecasting the driver object to TakesScreenshot interface type.
TakesScreenshot ts = (TakesScreenshot) driver;

//getting the source file using getScreenshotAs() method and storing in a file
File srcFile = ts.getScreenshotAs(OutputType.FILE);

/*Created a folder called "screenshot" in the project directory
Created another file by concatenating the date value  which has "_" in it
(Underscore is the accepted character while creating a file in the project )*/

File destFile = new File(".\\screenshot\\"+date2+"__actiTIMELoginPage.png");

/*copyFile() method is a static method present in FileUtils class of JAVA
storing the screenshot in the destination location*/

FileUtils.copyFile(srcFile, destFile);

//closing the browser
driver.close();
```

```
        }
}
```

Question : ***Why capturing screenshot of the web pages is important in the project ?***

Answer :

- **We capture screenshots in order to debug the failed test scripts.**
- **It actually helps the automation test engineer to find the exact root cause of the issue in the application at the earliest.**

Following are the possible scenarios after the script is failed:

- **Whenever an automation script is failed, we first manually execute the steps to check whether there is any issue in the application or the issue is with the script.**
- **If the script fails due to an issue in the script itself, we fix the script and re-run it till it is passed.**
- **If there is an issue in the application due to which the script is failed, then we log defect against the same issue. In this way, automation team gets credibility in the project.**

****

Handling Browser navigation

****

Question : ***How to navigate within the browser ?***

Answer : **Using navigation interface.**

How do you navigate with in the same browser ?

Ans:

1. By using Navigation interface.

2. Navigation interface has few non static methods, to access these methods, we need to create an object of Navigation interface.

3. But, we can't create an object of Navigation interface, so there is a method called navigate() from WebDriver interface, which returns an instance of Navigation interface.

4. Using this reference/instance, we can call methods like

to() --> which is used to navigate to any website.

back() --> it is used to navigate to the immediate previous page.

forward() -> it is used to navigate to the immediate next page.

refresh() --> it is used to refresh the current page.

This is how, we navigate with in the same browser

```
public class BrowserNavigationExample {
	public static void main(String[] args) throws InterruptedException {
		System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");
		WebDriver driver = new FirefoxDriver();
		//Enter the url
		driver.get("http://localhost:8080/login.do");
		driver.navigate().to("http://www.gmail.com");
		Thread.sleep(3000);
		driver.navigate().back();
		Thread.sleep(3000);
		driver.navigate().forward();
		Thread.sleep(3000);
		driver.navigate().refresh();
		driver.close();
	}
}
```

Q. Difference between to() and get() method .

to() - it is a non static method of Navigation interface.

it maintains browser history.

it doesn't wait for the page to be completely loaded, we have to explicitly delay the execution by thread.sleep(2000)/explicit wait().

get()- it is a non- static method of Webdriver interface.

it doesn't maintain browser history.

it waits till the page is completely loaded.

****

Handling Mouse and Keyboard Operations

****

How do you handle keyboard related operations ?

Ans:

1. By using Robot class present in java.awt package. awt- abstract window toolkit

2. Robot class has few non static methods, to access these methods,

we need to create an object of Robot class.

3.Using this reference/instance, we can call methods like

keyPress() --> which is used to press any key from the keyboard.

keyRelease --> it is used to release any key from keyboard.

KeyPress() and KeyRelease() methods accepts an argument where in we mention which exact key we want to press on.

All these keys are present in a class called KeyEvent present in java, which has static and final variables, which represents the KEY and it is of int type..

This is how, we handle key board related operations.

Scenario- Enter username in UNTB of actiTIME login page and delete the UN using

```
// CTRL+A+(DELETE/BACKSPACE)
Robot r = new Robot();
WebElement unTB = driver.findElement(By.id("username"));
unTB.sendKeys("admin");
Thread.sleep(3000);
r.keyPress(KeyEvent.VK_CONTROL);
r.keyPress(KeyEvent.VK_A);
r.keyPress(KeyEvent.VK_DELETE);
r.keyRelease(KeyEvent.VK_CONTROL);
r.keyRelease(KeyEvent.VK_A);
r.keyRelease(KeyEvent.VK_DELETE);
```

****

Question : ***How to handle mouse movement and keyboard Operations ?***

Answer :

- ***We handle mouse movement in Selenium using mouseMove() method of Robot Class.***
- ***Similarly, to handle keyboard operations, we use KeyPress() and KeyRelease() methods of Robot Class***

Selenium Code to demonstrate an example of Mouse movement and Keyboard operation :

```
package test;
```

```
import java.awt.AWTException;
import java.awt.Robot;
import java.awt.event.KeyEvent;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
public class Keyboard_Mouse_Operations {
	public static void main(String[] args) throws InterruptedException, AWTException {
		System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");
		//1. Launch the browser
		WebDriver driver = new FirefoxDriver();
		//2. enter the url -
		driver.navigate().to("http://localhost:8080/login.do");
		Thread.sleep(5000);
		//Creating an object of Robot Class
		Robot r = new Robot();
		//move the mouse by x and y coordinate
		r.mouseMove(300, 500);
		//press ALT key from keyboard
		r.keyPress(KeyEvent.VK_ALT);
		//press F key from keyboard
		r.keyPress(KeyEvent.VK_F);
		//Release F key from keyboard
		r.keyRelease(KeyEvent.VK_F);
		//Release Alt key from keyboard
```

```
r.keyRelease(KeyEvent.VK_ALT);

Thread.sleep(3000);

//Press W key from keyboard to open a new private window

r.keyPress(KeyEvent.VK_W);

//Release W key from keyboard

r.keyRelease(KeyEvent.VK_W);

Thread.sleep(3000);

// It will close only the current browser window

//driver.close();

// It will close all the browser windows opened by Selenium

driver.quit();

}}
```

****

Identification of WebElements using Locators

****

What is an WebElement ?

1. Any element present on a web page is called as web element.
2. Developers use HTML code to develop web pages.
3. For testing purpose, we can also create web page using HTML code.
4. In order to create a web page, we need to write HTML code in any text pad (eg : notepad and save the file with .html extension)

Create a sample web page using HTML as mentioned below.

```
<html>
```

```
<body>
    UN : <input type="text" id = "username" value = "admin">
    PWD: <input type="text" id= "pass" value = "manager">
    <a href="http://localhost:8080/login.do"> Click ActiTIME Link</a>
</body>
</html>
```

In the above HTML tree, every element should have one of the 3 options.

1. **Tagname (this is mandatory for all the elements)**
2. **Attributes (Optional)**
3. **Text (Optional)**

Example of Tagname in the above HTML Tree structure:

- **html,**
- **body,**
- **input,**
- **a**

Example of Attributes in the above HTML Tree structure:

- **type = "text"**
- **id = “username”**
- **value = "admin"**

Format : **attributeName = “attributeValue”**

Example of Text in the above HTML Tree structure:

- **Click ActiTIME link**

What are Locators ?

- **Locators are used to identify(locate) the web elements on the web page. Find Element and Find Elements methods use these locators to uniquely identify an element on a webpage. Locators are the static methods of By(Selenium) class.**

- **We have 8 types of locators in Selenium using which findElement()/findElements() methods identifies elements on the web page:**
    1. **id**
    2. **name**
    3. **tagName**
    4. **className**
    5. **linkText**
    6. **partialLinkText**
    7. **xpath**
    8. **cssSelector**

What is Factory Method:-

A method which returns the instance of the same class in which it is present is called as Factory method. All the locators are factory method which returns an instance of By class.

- **findElement() method based on specified locators start traversing in the html tree from the root node , the moment it found a matching element, it stops there and returns the address of the web element on the web page and the return type is WebElement.**
- **If the specified locators matches with multiple elements, then findElement() method returns the address of the first matching element.**
- **If the specified locators matches no element, then findElement() method throws an exception called "NoSuchElementException".**

NoSuchElementException:

When findElement() method based on a specified locator is unable to return the address of a matching element of a page, it throws NoSuchElementException.

Note :

In below Selenium code snippet,

```
WebDriver driver = new FirefoxDriver();
```

1. `driver.findElement(By.id(""));`
2. `driver.findElement(By.name(""));`
3. `driver.findElement(By.tagName(""));`
4. `driver.findElement(By.class(""))`
5. `driver.findElement(By.linkText(""))`
6. `driver.findElement(By.partialLinkText(""))`
7. `driver.findElement(By.xpath(""))`
8. `driver.findElement(By.cssSelector(""))`

1. (By is an abstract class and all the locators specified/highlighted above are static methods of By class and hence, we call directly by using <classname.static Concrete> methods

---

How selenium identifies object on webpage ?

Ans :

1. Selenium identifies objects on webpage by using findElement() method.

2. findElement() method internally uses any one of the 8 locators to identify objects on the webpage.

3. findElement() starts traversing in the html tree right from the root node and the moment, it finds the first matching elements, it returns the address of the same element.

4. Incase, findElement() fails to identify an object on the webpage using the specified locator, it throws NoSuchElementException

This is how, selenium identifies objects or elements on the webpage.

Why we don't use className and tagName locator to identify objects ?

Ans:

Multiple elements on the web page can have the same tag or can belong to the same class, and findElement() method always returns the address of first matching element. And our scenario may be to perform an action on any other elements and not on the first matching element. In this case, our purpose may not be served always and hence , we don't use either of them.

Below is the code to demonstrate the usage of locators in selenium while identifying the web elements on the web page:

```
package pack1;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.firefox.FirefoxDriver;


public class LocatorsExample{

   public static void main(String[] args) throws InterruptedException {

          System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

          WebDriver driver = new FirefoxDriver();
```

```
        // Enter the URL of your own created sample web page
        driver.get("file:///C:/Users/admin/Desktop/UN.html");
        // Used “id” locator to find USERNAME text box
        WebElement unTB = driver.findElement(By.id("user"));
        //Clear the existing value present in the text box
        unTB.clear();
        // Enter value into the USERNAME text box
        unTB.sendKeys("ajit.biswas@gmail.com");
        // Used “name” locator to find Password text box
        WebElement passTB = driver.findElement(By.name("n1"));
        //Clear the existing value present in the text box
        passTB.clear();
        //Halt the program execution for 2 seconds
        Thread.sleep(2000);
        // Enter value into the Password text box
        passTB.sendKeys("Qspiders123");
        // Find the address of ActiTIME Link and click
        driver.findElement(By.linkText("Click ActiTIME link")).click();
        Thread.sleep(2000);
}}
```

Important notes on LinkText and PartialLinkText locator

- **Out of all the locators, linkText and PartialLinkText are used to identify only the links present on the webpage.(elements whose tagname is “a” -- a stands for anchor)**
- **LinkText locator should be used when the text of the link is constant**

- **PartialLinkText locator should be used when certain part of the link text is getting changed every time the page is loaded. i.e for partially dynamically changing text, we use partialLinkText locator**
- **To handle those elements whose text changes completely, we can't use partialLInkText locator. It should be handled by another locator called "xpath"**
- **If we use try to use these 2 locators on other type of elements (except Links), then we get "NoSuchElementException"**

What is linkText and partialLinkText and when do we use them ?

Ans:

1. linkText and partialLinkText are the locators using which findElement method identifies objects on webpage.

2. These locators are used to identify link type of elements only.

When the text of the link is constant, we use linkText locator.

4 when the text of the link partially changes, then we use partialLinkText

locator.

Steps to install firebug and firepath add-ons in Firefox browser :

1. **We need to install firebug and firepath addons in firefox browser to write cssSelector expression and then evaluate whether the expression is correct or not.**
2. **To install firebug addon in firefox browser :**

   TOOLS -- > ADD-ONS → EXTENSIONS → search firebug -- > and click on Install -- Restart the browser.

3. To install firepath addon in firefox browser :

TOOLS -- > ADD-ONS → EXTENSIONS → search firepath -- > and click on Install -- Restart the browser.

Steps to write and evaluate cssSelector expression in firefox browser :

1. **Navigate to the web page -- > right click anywhere on the web page → select inspect element with firebug or Press F12 from keyboard.**
2. **Go to firepath tab and select CSS option.**
3. **Type the cssSelector expression and hit Enter key from the keyboard, it will highlight the corresponding matching element on the web page.**

Steps to write and evaluate cssSelector expression in Chrome browser :

1. **In order to write cssSelector expression in chrome browser, we don't need any add-ons as such.**
2. **Navigate to the web page -- > right click anywhere on the web page → Press F12 from keyboard or select inspect element, it will open the Developer tool section with Elements tab selected by default.**
3. **Press Ctrl+F and write the cssSelector expression, it will highlight the source code of the matching element.**
4. **Place the cursor on the highlighted source code, it will highlight the corresponding element present on the web page.**

cssSelector locator :

1. **It is one of the locator in Selenium using which we identify web elements on the web page.**
2. **It stands for Cascading Style Sheet.**
3. **The standard syntax for cssSelector expression is**

tagName[attributeName = 'attributeValue']

OR

here, tagName is not mandatory.

[attributeName = "attributeValue"]

Sample Element html code for Login button:

<input type="textbox" id= "ID123" class = "inputText" value="Login">

Following are the 4 different ways of writing cssSelector expression for above mentioned Login button :

CssSelector Expression using type as an attribute :: input[type='textbox']

Actual code to identify Login button using FindElement() method:

driver.findElement(By.cssSelector("input[type='textbox']"))

CssSelector Expression using id as an attribute : input[id='ID123']

Actual code to identify Login button using FindElement() method:

driver.findElement(By.cssSelector("input[id='ID123']"))

CssSelector Expression using class as an attribute : input[class='inputText']

Actual code to identify Login button using FindElement() method:

driver.findElement(By.cssSelector("input[class='inputText']"))

CssSelector Expression using value as an attribute : input[value='Login']

Actual code to identify Login button using FindElement() method:

driver.findElement(By.cssSelector("input[value='Login']"))

Important Note :

While deriving cssSelector expression, we can use either one attribute or multiple attributes till we found unique matching element on the web page.

eg : input[type='textbox'][id='ID123'][class='inputText'][value='Login']

4. CssSelector can also be written using ID and Class. Here, ID is represented by " # " and className is represented by dot operator(.)

Sample Element html code for Login button:

<input type=”textbox” id= “ID123” class = “inputText” value=”Login”>

4.1 CssSelector expression for the above Login button can be written using ID as

input#ID123 (syntax = Tagname#id)

OR

#ID123 (syntax = #id) [note : tagname is not mandatory]

Actual code to identify Login button using FindElement() method is below :

driver.findElement(By.cssSelector(“#ID123”))

4.2 CssSelector expression for the above Login button can be written using ID as shown below

input.inputText (syntax = tagname.classname)

OR

“.inputText” (syntax = .classname) [note : tagname is not mandatory]

Actual code to identify Login button using FindElement() method:

driver.findElement(By.cssSelector(“.inputText”))

Limitation of cssSelector :

1. **It does not support text i.e we can identify element using text of the element.**
2. **It does not support backward traversing.**
3. **It doesn’t support index**

XPATH :

1. **xpath is one of the locator in selenium using which we identify objects or elements on the web page and perform specific actions to carry out automation testing.**
2. **xpath is the path of an element in the html tree.**
3. **xpath are of 2 types.**

3.1) Absolute xpath

3.2) Relative xpath

Absolute xpath :

1. **It refers to the path of the element right from the root node to the destination element.**
2. **While writing the absolute xpath, We use single forward slash (/) to traverse through each immediate child element in the html tree.**
3. **In the below sample html tree,**

document

|_____html

|

---- body

|

------> a

Absolute xpath can be written in the following ways.

html/body/a

or

./html/body/a

(Note :- here, dot (.) refers to the current document or the current web page, using dot here is optional)

4. Using absolute xpath in selenium code as shown below.

driver.findElement(By.xpath("html/body/a")).click();

5. In xpath, if there are multiple siblings with same tagname, then the index starts from 1.

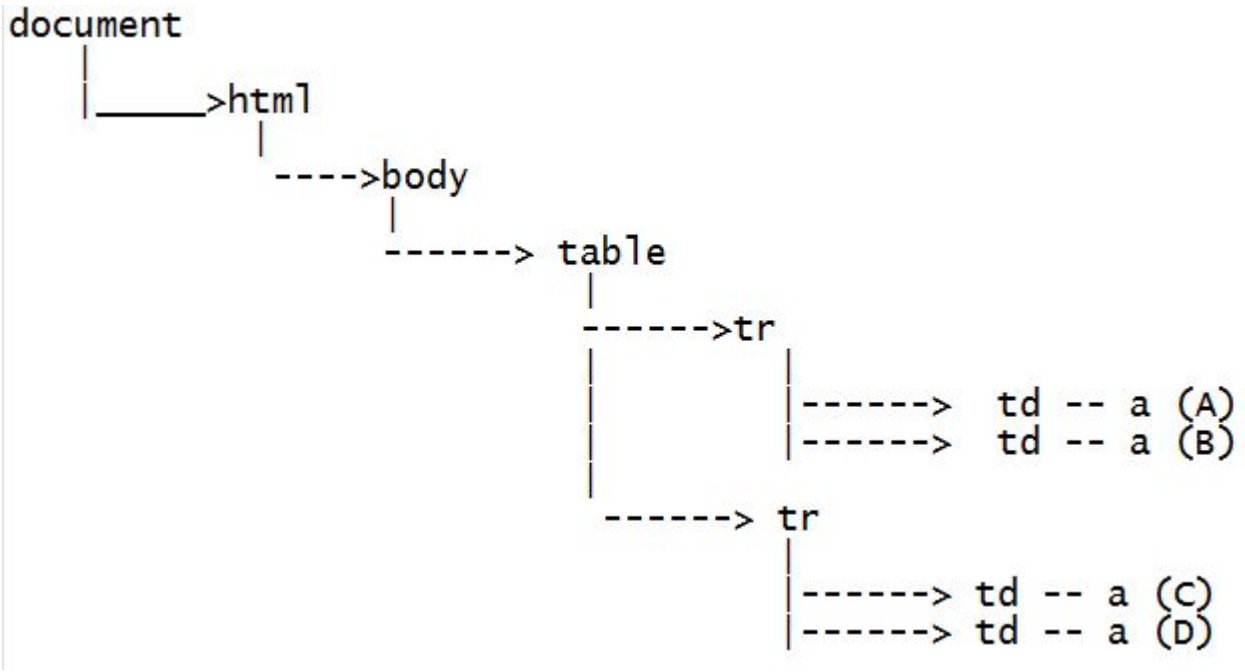
6.In case of multiple siblings with same tagname, if we don't use index, then it considers ALL the siblings.

7. We can join multiple xpath using pipeline operator (|)

Pipeline Operator / OR condition (|):

We use this operator to identify more than one element in different nodes of html tree. find element method after finding 1st matching element it stores the address of that element in some reference variable and start traversing from the root node to identify other matching elements. Once after identifying all the elements it consolidated all the address of matching elements and returns it.

Considering the below sample html tree, write Absolute xpath and Relative xpath expressions.



Fill in the table with Absolute xpath expressions using the sample html tree given above.

Absolute xpath expressions	Matching Element
html/body/table/tr[1]/td/a[1]	A
html/body/table/tr[1]/td/a[2]	B
html/body/table/tr[2]/td/a[1]	C
html/body/table/tr[2]/td/a[2]	D

html/body/table/tr[1]	**AB**
html/body/table/tr[2]	**CD**
html/body/table/tr[1]/td[1]|html/body/table/tr[2]/td[1]	**AC**
html/body/table/tr[1]/td[1]|html/body/table/tr[2]/td[2]	**BD**
html/body/table/tr[1]/td[1]|html/body/table/tr[2]/td[2]	**AD**
html/body/table/tr[1]/td[2]|html/body/table/tr[2]/td[1]	**BC**
html/body/table/tr[1]|html/body/table/tr[2]/td[1]	**ABC**
html/body/table/tr[1]|html/body/table/tr[2]/td[2]	**ABD**
html/body/table/tr[1]|html/body/table/tr[2]	**ABCD**

Relative xpath :

1. **In Absolute xpath, we write the path of the element right from the root node and hence, the expression for absolute xpath is lengthy.**
2. **In order to reduce the length of the expression, we go for Relative xpath.**
3. **In Relative xpath, we use double forward slash (//), which represents any descendant.**

Fill in the table with relative xpath expressions using the sample html tree given above.

Relative xpath expressions	**Matching Element**
//tr[1]//td[1]	**A**
//tr[1]//td[2]	**B**
//tr[2]//td[1]	**C**
//tr[2]//td[2]	**D**
//tr[1]//td	**AB**
//tr[2]//td	**CD**
//td[1]	**AC**

//td[2]	**BD**
//tr[1]//td[1] | //tr[2]//td[2]	**AD**
//tr[1]//td[2] | //tr[2]//td[1]	**BC**
//tr[1]//td | //tr[2]//td[1]	**ABC**
//tr[1]//td | //tr[2]//td[2]	**ABD**
//tr[1]//td | //tr[2]//td	**ABCD**

Interview questions :

1. **what is the difference between '/' and '//' ?**

Answer : "/" refers to the immediate child element in the html tree.

"//" refers to any element in the html tree. It also represent any descendant.

2. What are the types of xpath?

Ans: Absolute and Relative xpath.

3. Derive an xpath which matches all the links present on a web page ?

Ans : //a

4. Derive an xpath which matches all the image present on a web page ?

Ans : //img

5. Derive an xpath which matches all the links and images present on a web page ?

Ans : //a | //img

6. Derive an xpath which matches all the 2nd links present on a web page ?

//a[2]

7. Derive an xpath which matches all the links present inside a table present on a web page ?

//table//a

8. Difference between "//a"and "//table//a " ?

Ans : //a → refers to all the links present on the webpage.

//table//a → refers to all the links present within all the tables present on the webpage.

xpath by Attribute :

1. **xpath expression can be written using attribute of the web element.**
2. **Based on the situation, we would use either single attribute or multiple attributes to find an element on a web page.**
3. **Using single attribute in xpath expression, if it returns one matching element, then we would use single attribute only.**
4. **In case, by using single attribute in xpath expression, if it returns multiple matching elements on the web page, then we would go for using multiple attributes in the xpath expression till we get one matching element.**

xpath expression using Attribute :

1. using single attribute :

Syntax : //tagname[@attributeName = 'attributeValue']

//tagname[NOT(@attributeName = 'attributeValue')]

Sample application : actiTIME application

url : https://demo.actitime.com/login.do

Write xpath for below few elements on above actiTIME login page :

Web Element	xpath Expression
username textbox	//input[@id='username']
password textbox	//input[@name='pwd']
login button	//a[@id='loginButton']/div
checkbox	//input[@type='checkbox']
clock image	//td[@id='logoContainer']/div/img

Usage in selenium code :

driver.findElement(By.xpath("paste any xpath here from above table"))

2. Using multiple attribute :

xpath Syntax :

- //tagName[@AN1='AV1'][@AN2='AV2']
- //tagName[@AN1='AV1'] | //tagName[@AN2='AV2']

Element : View licence link

html code after inspecting the element using F12 key:

```
<a id="licenseLink" target="" href="javascript:void(0)"
onclick="openLicensePopup();">View License</a>
```

xpath expression using "href" and "onclick" attributes :

```
//a[@href='javascript:void(0)' and @onclick='openLicensePopup();']
```

Usage in selenium code :

```
driver.findElement(By.xpath("//a[@href='javascript:void(0)']
                            [@onclick='openLicensePopup();']"))
```

Assignment :

Write xpath expression for below 7 elements present on actiTIME login page

Elements :

1. UserName
2. Password
3. Login Button
4. Check box
5. Actitime Image
6. View Licence link
7. actiTIME Inc link

Use the below format (Sample example for actiTIME Inc Link):

html code for <actiTIME Inc.> :

```
<a href="http://www.actitime.com" target="_blank">actiTIME Inc.</a>
```

xpath syntax

//tagname[@AN1 = 'AV1']

1. using href attribute:

//a[@href = 'http://www.actitime.com']

2. using target attribute

//a[@target = '_blank']

Note: Use all the attributes of an element to write xpath expression

xpath expression using text() function :

1. In the html code of an element, if attribute is duplicate or attribute itself is not present, then use text() function to identify the element.
2. In order to use text() function, the element should have text in the element's html code.

Syntax :

//tagName[text()='text value of the element']

OR

//tagName[.='text value of the element']

Note : Instead of text(), we can use dot (.) , the problem here with using dot (.) is sometimes,

it returns the hidden element also present on the webpage. which might confuse the

user. So the best practice is to use text() instead of using dot.

xpath expression using text() function for below elements present on actiTIME login page.

Web Element	xpath Expression using text() function
login button	//div[text()='Login ']
actiTIME 2017.4 link	//nobr[text()='actiTIME 2017.4']
actiTIME Inc link	//a[text()='actiTIME Inc.']

xpath expression using contains() function :

1. In the html code of an element, if the attribute value or the text is changing partially, then use contains() function to identify the element.
2. In order to use contains() function, the element should have either attribute value or text value.

   Syntax :

   - //tagName[contains(@attributeName,'attributeValue')]
   - //tagName[contains(text(),'text value of the element')]

xpath expression using contains() function for below elements present on actiTIME login page.

Web Element	xpath Expression using contains() function	Using
actiTIME 2017.4 link	//nobr[contains(text(),'actiTIME 2017')] This will work for any version that starts with 2017 eg: 2017.1, 2017.2 etc	contains() with text()
Clock Image	//img[contains(@src,'timer')]	contains() with attribute

3. We use contains() function when the text value is very lengthy or the attribute value is very lengthy.

eg: xpath to identify error message present on actitime login page (Click on login button without entering username and password to get the error message)

//span[contains(text(),'invalid')]

Program to illustrate xpath by attributes, text() function, contains() function and their usages with attributes and text values.

```
public class XpathUsingAttribute_Actitime extends BaseClass{
	public static void main(String[] args) throws InterruptedException {
		System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");
		WebDriver driver = new FirefoxDriver();
		//Enter the url of actiTIME application
		driver.get("http://localhost:8080/login.do");
		//xpath using multiple attributes
		String xp = "//input[@class='textField'][ @id = 'username']";
		Thread.sleep(2000);
		//Enter admin into username text box
		driver.findElement(By.xpath(xp)).sendKeys("admin");
		Thread.sleep(2000);
		//find password element using xpath by attribute and enter manager in to password textbox.

		driver.findElement(By.xpath("//input[@name='pwd']")).sendKeys("manager");
		Thread.sleep(2000);
		//find an image on the web page whose attributes (src)contains a value called timer
		WebElement clock =
driver.findElement(By.xpath("//img[contains(@src,'timer')]"));
		//store the width value of the clock image into a variable called widthValue
```

```
        String widthValue = clock.getAttribute("width");

        //Print the width of the clock image

        System.out.println("the width is :"+widthValue);

        //Print the height of the clock image

        System.out.println("the height of the clock element is : "+
        clock.getAttribute("height"));

        //xpath using text() function

        driver.findElement(By.xpath("//div[text()='Login ']")).click();

        Thread.sleep(2000);

        //xpath using contains() function and text() function

        driver.findElement(By.xpath("//a[@id='loginButton']//div[contains(text(),'Login')
        ]")).click();

        Thread.sleep(2000);

        driver.close();

    }

}
```

xpath expression using starts-with() function :

1. We use starts-with() function to identify those elements whose text value starts with some specified value.

   *xpath using **contains()** function:*

   //[contains(text(),'actiTIME')] - this xpath will return 6 matching element on login page of actiTIME application.

   *xpath using **starts-with()** function,*

   //[starts-with(text(),'actiTIME')] - this xpath will return only 3 matching element on login page of actiTIME application as the text value of these 3 elements starts with “actiTIME” text

Handling completely dynamic links :

1. When the text value of the elements is completely changing, then we can't use functions like "contains()", "starts-with()" etc to handle those elements.
2. In such cases, we identify the dynamically changing element using the nearby unique element. We call this concept as independent dependent xpath.

Steps to derive xpath expression using Independent dependent concept :

1. Identify the independent element on the webpage and inspect the element to view the source code and then derive the xpath expression.
2. Place your cursor on the independent element source code and move the mouse pointer upward till it highlights both the independent and dependent elements which is the common parent element.

   Add /.. to the xpath of independent element already noted down in step 1 to get the xpath of common parent.

3. Use mouse pointer to navigate from common parent to the desired dependent element and derive the xpath of the dependent element.
4. Write the xpath from Independent element to Common parent and then write the xpath from Common parent to dependent element.

Example 1:

Write xpath to identify **version** of Java Language present in Selenium Download page.

//td[.='Java']/../td[2]

Example 2:

Write xpath to identify **Release data** of Java Language present in Selenium Download page.

//td[.='Java']/../td[3]

Example 3:

Write xpath to identify **Download link** of Java present in Selenium Download page.

//td[.='Java']/../td[4]/a

Note :

- In case, if the column number of **Download link** changes, then the above xpath will fail to identify the link as we are hard coding the column position as 4 in the above case.

In order to handle this, we will write xpath in such a way that it works irrespective of the column position as shown below.

//td[.='Java']/..//a[.='Download']

Program 1 :

Write a script to click on the download link of Java in Selenium website

Scenario :

1. Login in to Selenium official website

   Url : http://www.seleniumhq.org/download

2. Click on the Download link for Java language.

```
public class Independent_Dependent_Xpath_Seleniumsite_javaDownload{

    public static void main(String[] args) throws InterruptedException {

        System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

         WebDriver driver = new FirefoxDriver();

        // enter the url

         driver.get("http://www.seleniumhq.org/download/");
```

```
            Thread.sleep(3000);

            // xpath using independent and dependent concept

        driver.findElement(By.xpath("//td[.='Java']/..//a[.='Download']")).click();

    }

}
```

Group Index :

What is group index in xpath ?

Ans :

Retrieving one element from a group of matching elements by using index is called group index.

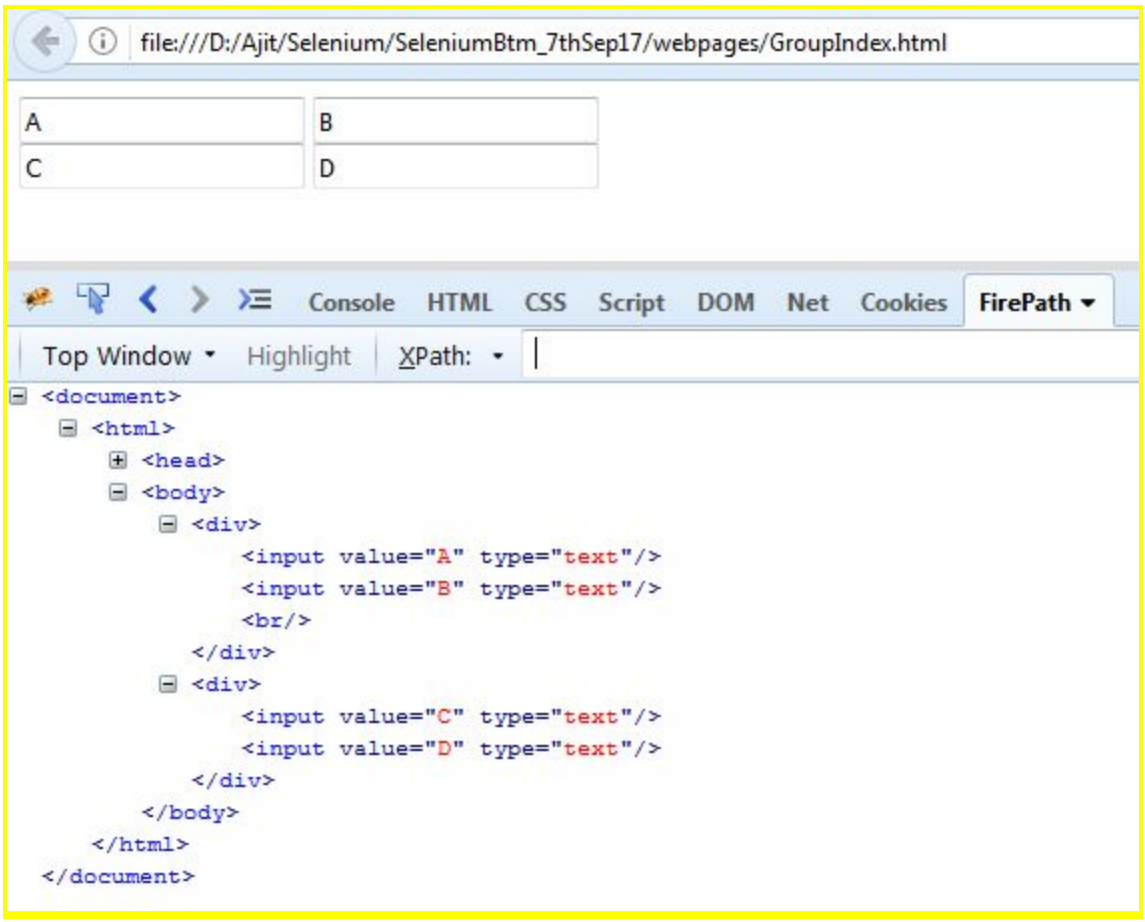
When do we go for group index ?

When the xpath expression matches with multiple elements, then we go for group index.

syntax :

(xpath expression)[index]

Sample Html tree :



xpaths using Group Index to identify the elements in the above sample tree:

xpath using Group Index	Matching Element
//input	ABCD
(//input)[1]	A
(//input)[2]	B
(//input)[3]	C
(//input)[4]	D
(//input)[last()]	D
(//input)[last()-1]	C
//input[1]	AC
(//input[1])[1]	A
(//input[1])[2]	C
(//input[1])[last()]	C
//input[2]	BD
(//input[2])[1]	B
(//input[2])[2]	D
(//input[2])[last()]	D

1. In Group index, we write xpath expression within the braces and then we write the index outside the braces.
2. Internally, it executes the xpath expression first and stores the result in an xpath array whose index starts with 1
3. last() is a function that is used to retrieve the last element present in the xpath array.

Program 2 :

Click on the **Set by default** link of **testing** present in **type of work (Setting tab)** of actiTIME application

Scenario :

3. Login in to actime application

   Url : http://localhost:8080/login.do

   UN - admin, PWD - manager

4. click on Settings

5. Click on the Types of Work link present in the window
6. click on the Set by Default link for a type of work called “testing”

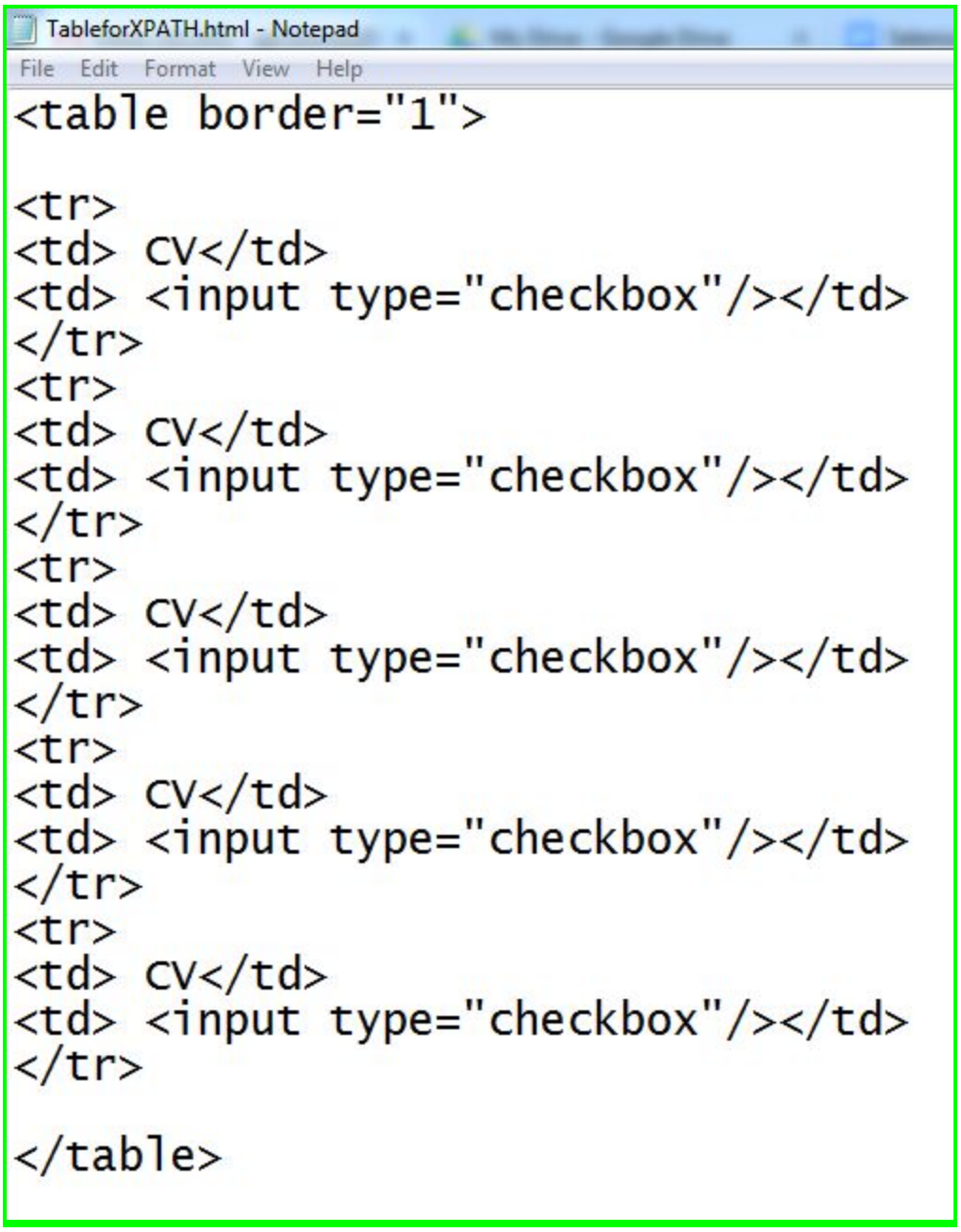
Use the below hints :

1. Use groupIndex concept to find **Setting** Element and
2. Independent and dependent concept to find **Set by Default** link

```
public class Xpaths_Independent_dependent_actitime_setbydefault {

   public static void main(String[] args) throws InterruptedException {

    System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

          WebDriver driver = new FirefoxDriver();

          driver.get("http://localhost:8080/login.do");

          driver.findElement(By.id("username")).sendKeys("admin");

          driver.findElement(By.name("pwd")).sendKeys("manager");

         //click on login button

          driver.findElement(By.xpath("//div[.='Login ']")).click();

          Thread.sleep(4000);

         //Click on settings tab on home page
         driver.findElement(By.xpath("(//div[@class='popup_menu_label'])[1]")).click();

          Thread.sleep(2000);

         //Click on Types of Work link

          driver.findElement(By.xpath("//a[.='Types of Work']")).click();
```

```
            Thread.sleep(4000);

            //Click on testing link present under Type of work column

            driver.findElement(By.xpath("//a[.='testing']/../..//a[.='set by default']")).click();

            driver.close();

        }

    }
```

Create a html file as shown below.



Xpath expression using GroupIndex concept :

XPATH using Group Index	Matching Element
//input[@type='checkbox']	ABCDE
(//input[@type='checkbox'])[1]	A
(//input[@type='checkbox'])[LAST()]	E
(//input[@type='checkbox'])[POSITION()=3]	C
(//input[@type='checkbox'])[POSITION()>=3]	CDE
(//input[@type='checkbox'])[POSITION() < 3]	AB
(//input[@type='checkbox'])[POSITION() = 1 OR Position() = last()]	AE

Xpath Axes :

1. In xpath, navigating from one element to another element is called ***traversing***.
2. In order to traverse from one element to another, we use xpath axes.
3. We have the following 6 xpath axes in selenium.
    - child
    - descendant
    - parent
    - ancestor
    - preceding-sibling
    - following-sibling

What are AXES in xpath ?

Axes are something using which xpath traverses in the html tree either in upward or downward direction.

There are 6 types of axes.

---

1. child (/)

a. Using this axes, xpath traverses to the immediate child element in the html tree.

b. This axes is represented by single forward slash (/)

c. We use this child axes in deriving the absolute xpath expression.

---

2. descendant (//)

a. Using this axes, xpath traverses to any child element in the html tree.

b. This axes is represented by double forward slash (//)

c. We use this descendant axes in deriving the relative xpath expression.

3.parent axes (/..)

a. Using this axes, xpath traverses to the immediate parent element in the html tree.

b. The shortcut of this axes is single forward slash (/..)

---

4. ancestor axes

a. Using this axes, xpath traverses to any parent element in the html tree.

---

5. preceding-sibling

using this axes, xpath traverses with in the siblings in upward direction.

---

6. following-sibling

using this axes, xpath traverses with in the siblings in downward direction.

Create a .html file with the below html code

CalendarMonth.html - Notepad

File Edit Format View Help

```
<select id="day" title="Day" multiple>
<option value="1">1</option>
<option value="2">2</option>
<option value="3">3</option>
<option value="4">4</option>
<option value="5">5</option>
<option value="6">6</option>
<option value="7">7</option>
<option value="8">8</option>
<option value="9">9</option>
<option value="10">10</option>
</select>
```

Following are the syntax to use all the xpath axes in Selenium.

child Axes:

eg : /html → can be written using ***child*** axes as → child::html

descendant Axes:

eg : //option[5] → can be written using ***descendant*** axes as → descendant::option[5]

parent Axes:

eg : //option[5]/.. → can be written using ***parent*** axes as → descendant::option[5]/parent::select

ancestor Axes:

eg : //option[5]/../.. → can be written using ancestor axes as →

descendant::option[5]/ancestor::body

preceding-sibling Axes:

eg : → xpath using ***preceding-sibling*** axes →

descendant::option[5]/preceding-sibling::option[1] - it will select 4 in the list box

following-sibling Axes:

eg : → xpath using ***following-sibling*** axes → descendant::option[5]/following-sibling::option[1] - it will select 6 in the list box

Following table illustrates a detailed level understanding of all the xpath axes :

xpath axes type	xpath using axes	xpath using shortcut	Matching Element (Months)
child	html/body/select/child::option[5]	html/body/select/option[5]	5
descendant	descendant::option[5]	//option[5]	5
parent	descendant::option[5]/..	//option[5]/..	It will highlight SELECT tag
ancestor	//option[5]/ancestor::html	no short cut available for ancestor axes	It will highlight HTML tag
preceding-sibling	//option[5]/preceding-sibling::option	No short cut available for preceding-sibling axes	1 2 3 4
	//option[5]/preceding-sibling::option[1]		4
	//option[5]/preceding-sibling::option[2]		3
	//option[5]/preceding-sibling::option[3]		2
	//option[5]/preceding-sibling::option[4]		1
	//option[5]/preceding-sibling::option[last()]		1
	//option[5]/preceding-sibling::option[last()-1]		2
	//option[5]/preceding-sibling::option[position()=1]		4
	//option[5]/preceding-sibling::option[position()=last()]		1
following-sibling	//option[5]/following-sibling::option[1]	No short cut available for preceding-sibling axes	6
	//option[5]/following-sibling::option[2]		7
	//option[5]/following-sibling::option[3]		8
	//option[5]/following-sibling::option[4]		9
	//option[5]/following-sibling::option[last()]		10
	//option[5]/following-sibling::option[position()=last()]		10

Difference between CssSelector and Xpath

CssSelector	Xpath

It is faster	It is slower
text() function is not supported	text() function is supported
backward traversing is not supported	backward traversing is supported
groupIndex is not supported	groupIndex is supported

Imp Note :

In CssSelector, we traverse through the element using this symbol " > "

Why Cssselector is faster than xpath ?

Cssselector is faster than xpath because cssselector traverse in the html tree only in the downward direction,

but xpath traverses in both upward and downward direction.

Q: Priority of using locators

1. id

2. name

3. linkText

4. cssSelector

5. xpath

Q1: When do we use ID locator ?

Ans : When the html source code have an id attribute, then we can use id locator. And what we pass as an argument to ID locator is the value of ID attribute.

Q2 : When do we use NAME locator ?

Ans : When the html source code have name attribute, then we can use name locator. And what we pass as an argument to Namelocator is the value of Name attribute.

Q3 : We have both id and name attribute, which locator do we prefer ?

Ans : We prefer ID over name locator, because ID is unique and Name can be duplicated.

Q4 : How do we handle dynamically changing objects on the webpage ?

Ans : By using xpath with contains function when the attribute value or the text partially changes.

If the text completely changes, we handle this scenario by using independent-dependent xpath concept.

----

what are the functions of xpath, u are aware of ?

1. **contains() - we use contains() function to handle elements where in the attribute value or the text of an element partially changes.**

syntax :

using attribute :

//tagname[contains(@attributeName,'constant part of the attribute value')]

----

text() - this function represents the text of an element.

----

last() - it retrieves the element present at the last index.

----

position() -

it starts from index 1 and it gets auto incremented till

the condition is true and it fetches all the matching elements present at

the given index.

----

starts-with() :

it checks whether the specified part of either the

attribute value or the text is present at the beginning of the given attribute value or text of an element.

----

ends-with function :

it checks whether the specified part of either the attribute value or the text is present at the end of the given attribute value or text of an element.

Note :

this function doesn't work with 3.x series of selenium. It has been deprecated in 3.x series.

It was working till 2.x series of selenium.

Interview Questions :

How do you ensure the required page is displayed or not ?

We can use following checkpoints to validate the required page is displayed or not.

1. using title of the page
2. using URL of the page
3. using any unique element on the page

Write a program to validate Actitime application home page using TITLE of the page

```
public class VerifyhomepageUsingTitle {
        public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");
         WebDriver driver = new FirefoxDriver();
         driver.get("http://localhost:8080/login.do");
          driver.findElement(By.id("username")).sendKeys("admin");
        driver.findElement(By.name("pwd")).sendKeys("manager");
        driver.findElement(By.xpath("//div[.='Login ']")).click();
        Thread.sleep(3000);
        String expectedTitle = "Enter Time";
        String actualTitle = driver.getTitle();
        //If actual title contains "Enter Time" text then home page is displayed.
        if (actualTitle.contains(expectedTitle)) {
        System.out.println("Home page is displayed");
        } else{
        System.out.println("Home page is NOT displayed");
        }
    }
```

}

*Write a program to validate Actitime application home page using **Current URL** of the page*

```
public class VerifyhomepageUsingUrl {
	public static void main(String[] args) throws InterruptedException {
	System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");
		WebDriver driver = new FirefoxDriver();
		 driver.get("http://localhost:8080/login.do");
		 driver.findElement(By.id("username")).sendKeys("admin");
		 driver.findElement(By.name("pwd")).sendKeys("manager");
		 driver.findElement(By.xpath("//div[.='Login ']")).click();
		 Thread.sleep(3000);
		 String expectedUrl = "submit";
		 String actualUrl = driver.getCurrentUrl();
		 if (actualUrl.contains(expectedUrl)) {
		 System.out.println("Home page is displayed");
		} else{
		System.out.println("Home page is NOT displayed");
		}
	}
}
```

*Write a program to validate Actitime application home page using **any UNIQUE element** on the page*

```
public class VerifyhomepageUsingUniqueElement {
	public static void main(String[] args) throws InterruptedException {
	System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");
```

```
        WebDriver driver = new FirefoxDriver();
        driver.get("http://localhost:8080/login.do");
        driver.findElement(By.id("username")).sendKeys("admin");
        driver.findElement(By.name("pwd")).sendKeys("manager");
        driver.findElement(By.xpath("//div[.='Login ']")).click();
        Thread.sleep(3000);
        WebElement logoutBtn = driver.findElement(By.xpath("//a[.='Logout']"));
        if (logoutBtn.isDisplayed()) {
        System.out.println("Home page is displayed");
        } else{
        System.out.println("Home page is NOT displayed");
        }
}
}
```

Interview Questions : WebElement interface methods ?

---

1. getLocation():

--> is a method of webelement interface which is used to get the position of an element on the webpage.

---> it returns an instance of Point class.

Point class has few methods like getX(), getY()..

--> getX() method returns the xco-ordinate of the given element in the form of int and hence the return type is integer.

```
        int eleObjXcor = eleObj.getLocation().getX();
```

--> getY() method returns the yco-ordinate of the given element in the form
of int and hence the return type is integer.

---

2. getSize():

--> is a method of webelement interface which is used to get the dimensions
of an element on the webpage.

---> it returns an instance of Dimension class.

Dimension class has few methods like getHeight(), getWidth()..

--> getHeight() method returns the height of the given element in the form
of int and hence the return type is integer.

```
int eleObjHeight = eleObj.getSize().getHeight();
```

--> getWidth() method returns the width of the given element in the form
of int and hence the return type is integer.

```
int eleObjWidth = eleObj.getSize().getWidth();
```

---

3. isDisplayed()

--> is a method of webelement interface which is used to check whether an
element is displayed or not on the webpage.

---> if the element is displayed, it returns true, and if it is not displayed, it returns false. and hence the return type is boolean.

---

4. isEnabled()

--> is a method of webelement interface which is used to check whether an
element is ENABLED or not on the webpage.

---> if the element is enabled,it returns true, and if it is disabled,
it returns false. and hence the return type is boolean.

---

5. sendKeys()

--> is a method of webelement interface which is used to enter a value in to a text box or a text area.

what is the return type of sendkeys ?

--> it returns nothing,it is void..

---

6. clear()

--> is a method of webelement interface which is used to clear a value from an element [element can be a text box or a text area etc.]

what is the return type of clear?

--> it returns nothing,it is void..

---

7. isSelected()

--> is a method of webelement interface which is used to check whether an element is Selected or not on the webpage.

---> if the element is selected, it returns true, and if it is not selected, it returns false. and hence the return type is boolean.

---

8. getAttribute(String)

--> is a method of webelement interface which returns the value of the specified attribute in the form of string.

-- > if the specified attribute name is found in the html source code of an element, it returns the value of that particular attribute.

--> In case, if the specified attribute name is not found, it returns an empty string object. (Note : it doesn't throw any exception)

---

9. getText()

--> is a method of webelement interface which returns the text of an element in the form of string and hence the return type is String. If no text is present it returns empty string.

---

10. getTagname()

--> is a method of webelement interface which returns the tagname of an element in the

form of string and hence the return type is String

---

11. click()

--> is a method of webelement interface which is used to click on any element.

what is the return type of click method ?

it doesn't return anything, it is void.

---

11. submit()

--> is a method of webelement interface which is used to click on any element.

--> In order to use submit() method, 2 preconditions need to be satisfied.

1. the element source code should have an attribute called type="submit"

2. the element must be present inside a form tag

eg:

<form>

<input type="submit">

</form>

---

12. getCssValue()

--> is a method of webelement interface.

--> it returns the value of the specified style related attribute in the form of string

--> the return type of this method is String

---

13. getRect() --

--> is a method of webelement interface.

--> it is used to get the x and y coordinates of elements and also to find the height and width of an element.

--> it has few methods like getPoint() and getDimension().

--> getPoint() method returns the instance of Point class.

```
int eleXcor = eleObj.getRect().getPoint().getX();
```

--> getDimension() method returns the instance of Dimension class.

```
int eleXcor = eleObj.getRect().getDimension().getWidth();
```

Write a program to validate Username and Password fields on Actitime login page are aligned or not ?

```
public class VerifyUNandPWDalignment extends BaseClass{
	public static void main(String[] args) {
		driver.get("http://localhost:8080/login.do");
		WebElement unTB = driver.findElement(By.id("username"));
		int un_x = unTB.getLocation().getX();
		int un_width = unTB.getSize().getWidth();
		int un_height = unTB.getSize().getHeight();
		WebElement pwTB = driver.findElement(By.name("pwd"));
		int pw_x = pwTB.getLocation().getX();
		int pw_width = pwTB.getSize().getWidth();
```

```
		int pw_height = pwTB.getSize().getHeight();
		if (un_x == pw_x && un_width==pw_width && un_height==pw_height) {
			System.out.println("Username and password text box are aligned");
		} else {
			System.out.println("Username and password text box are NOT aligned");
		}
	}
}
```

Assignment :

*Write a program to validate Username and Password fields on **Facebook login page** are aligned or not ?*

```
public class VerifyFB_UNandPWDfieldsAreAligned_intheSameRow {
	public static void main(String[] args) throws InterruptedException {
		System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");
		WebDriver driver = new FirefoxDriver();
		driver.get("https://www.facebook.com/");
		WebElement unTB = driver.findElement(By.id("email"));
		// get the y-coordinate of username field
		int username_Ycordinate = unTB.getLocation().getY();
		System.out.println(username_Ycordinate);
		WebElement pwdTB = driver.findElement(By.name("pass"));
		// get the y-coordinate of password field
		int password_Ycordinate = pwdTB.getLocation().getY();
		System.out.println(password_Ycordinate);
		//check whether the Y-coordinate of username and  password field are same
```

```
            if (username_Ycordinate==password_Ycordinate) {

            System.out.println("Both username and password fields are displayed in the
            same row");

            }else{

            System.out.println("username and password fields are NOT aligned in the same
            row");

            }

        }

}
```

*Write a program to validate **the height and width of Username and Password** fields on **Facebook login page** are same or not ?*

```
public class VerifyActime_UNandPassword_HeightandWidth {

        public static void main(String[] args) throws InterruptedException {

            System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

            WebDriver driver = new FirefoxDriver();

            driver.get("http://localhost:8080/login.do");

            //find the username field

            WebElement unTB = driver.findElement(By.id("username"));

            //store the height of username

            int username_height = unTB.getSize().getHeight();

            //store the width of username

            int username_width = unTB.getSize().getWidth();

            System.out.println(username_height);

            System.out.println(username_width);

            //find the password field
```

```
                WebElement pwdTB = driver.findElement(By.name("pwd"));
                //store the height of password
                int password_height = pwdTB.getSize().getHeight();
                //store the width of password
                int password_width = pwdTB.getSize().getWidth();
                System.out.println(password_height);
                System.out.println(password_width);
                //check the height and width of username and password fields are same
                if (username_height==password_height && username_width==password_width)
{
                System.out.println("Username and password fields are aligned”);
                }else{
                System.out.println("Username and password fields are NOT aligned");
                }

                }
        }
}
```

*Write a script to validate that **the username field** on **Facebook login page** is **smaller** than the **Mobile Number field** ?*

```
public class VerifyFB_Usernamefield_lessthanMobileNumberField {
        public static void main(String[] args) {
                System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");
                WebDriver driver = new FirefoxDriver();
                driver.get("https://www.facebook.com/");
                WebElement unTB = driver.findElement(By.id("email"));
```

```
        int username_width = unTB.getSize().getWidth();

        System.out.println(username_width);

        //Identify the mobile number text box

        WebElement mobileNumTB =
        driver.findElement(By.xpath("//input[contains(@aria-label,'Mobile number or
        email address')]"));

        int mobNumWidth = mobileNumTB.getSize().getWidth();

        System.out.println(mobNumWidth);

        //Compare the width of both username and mobilenumber text box

        if (username_width==mobNumWidth) {

        System.out.println("Size of Both username and password fields are same"
        +username_width+" = " + mobNumWidth);

        }else{

        System.out.println("Size of username and password fields are NOT same that is :
        " +username_width+" Not equals to " + mobNumWidth);

        }

    }

}
```

Interview Question :

What is ActiveElement method

1. ActiveElement method returns the address of the active element on the webpage.

2. The return type of activeELement() method is WebElement.

3. In case, there is no active element on the page, and we try to switch to an active element, we get this exception - NoSuchElementException.

Write a script to enter a text into the focussed element (eg : textbox).

```
public class EnterTextintoFocussedElement {

	public static void main(String[] args) throws InterruptedException {

		System.setProperty("webdriver.gecko.driver",
		".\\driver\\geckodriver.exe");

		WebDriver driver = new FirefoxDriver();

		driver.get("http://localhost:8080/login.do");

		//entering text into the focussed element

		driver.switchTo().activeElement().sendKeys("admin");

	}

}
```

How do you remove value present in username text box of Actitime application ?

Using **clear()** method of WebElement interface.

Selenium code :

```
public class RemoveValuefromText_usingClearMethod{
	public static void main(String[] args) throws InterruptedException {
		driver.get("http://localhost:8080/login.do");
		driver.findElement(By.id("username")).sendKeys("ajit");
		Thread.sleep(2000);
		String value = driver.findElement(By.id("username")).getAttribute("value");
		System.out.println("Value present inside the text box is : "+value);
		driver.findElement(By.id("username")).clear();
		Thread.sleep(2000);
		driver.findElement(By.id("username")).sendKeys("againEnteredAjit");
```

```
        Thread.sleep(2000);


        driver.findElement(By.id("username")).sendKeys(Keys.CONTROL+"a"+Keys.DELE
        TE); // this line will actually delete the value if there is no space in the text
        entered

        // if there is a space between two words in the username field, we have to use
        the below lines of code

        driver.findElement(By.id("username")).sendKeys(Keys.CONTROL+"a") ;

        driver.findElement(By.id("username")).sendKeys(Keys.DELETE);

        Thread.sleep(2000);

        }

}
```

How do you remove value present in username text box of Actitime application without using clear() method ?

Using **sendKeys()** method of WebElement interface.

Selenium code : *driver.findElement(By.id("username")).**sendKeys(Keys.CONTROL + "a" + Keys.DELETE);***

```
public class RemoveValuefromText_usingClearMethod{

    public static void main(String[] args) throws InterruptedException {

        driver.get("http://localhost:8080/login.do");

        driver.findElement(By.id("username")).sendKeys("ajit");

        Thread.sleep(2000);

        String value = driver.findElement(By.id("username")).getAttribute("value");

        System.out.println("Value present inside the text box is : "+value);

        driver.findElement(By.id("username")).clear();

        Thread.sleep(2000);

        driver.findElement(By.id("username")).sendKeys("againEnteredAjit");

        Thread.sleep(2000);
```

```
***driver.findElement(By.id("username")).sendKeys(Keys.CONTROL+"a"+Keys.DELETE);***

Thread.sleep(2000);

}
}
```

*Write a script to print the **tooltip text** of the checkbox present on the login page of Actitime application ?*

Using **getAttribute()** method of WebElement interface.

Selenium code below :

```
public class PrintTooltip_Actitime_RememberCheckbox {

    public static void main(String[] args) {

        System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

        WebDriver driver = new FirefoxDriver();

        driver.get("http://localhost:8080/login.do");

        //find the Keep me Logged in Checkbox

        WebElement Checkbox = driver.findElement(By.id("keepLoggedInCheckBox"));

        //get the tooltip text using getAttribute() method and store in a variable

        String tooltipText = Checkbox.getAttribute("title");

        System.out.println(tooltipText);

        driver.close();

    }}
```

Write a script to check "Keep me Logged in" checkbox on the login page of Actitime application is selected or not ?

Using **isSelected()** method of WebElement interface.

Selenium code below :

```
public class CheckBox_selectedorNot{

    public static void main(String[] args) {
```

```
		driver.get("http://localhost:8080/login.do");

		WebElement KeepMeLogIN_Checkbox =
		driver.findElement(By.name("remember"));

		//select the checkbox

		KeepMeLogIN_Checkbox.click();

		/Using the isSelected() method, it checks whether the checkbox is selected or

		not :   if it is already selected, it return true and if not selected, then it returns

		false/

		if (KeepMeLogIN_Checkbox.isSelected()) {

			System.out.println("Checkbox is selected");

		}else{

			System.out.println("Checkbox is NOT selected");

		}

	}

}
```

*Write a script to check **"Username"** textbox on the login page of Actitime application is enabled or not ?*

Using **isEnabled()** method of WebElement interface.

Selenium code below :

```
public class VerifyUNtextboxisEnabledinActitime {

	public static void main(String[] args) {

		driver.get("http://localhost:8080/login.do");

		WebElement UN = driver.findElement(By.id("username"));

		if (UN.isEnabled()) {

			System.out.println("Username text box is enabled");

		}else {
```

```
                System.out.println("Username text box is disabled");
            }
            driver.close();
        }
}
```

*Write a script to **print the version** of actitime on login page of Actitime application*

Using **getText()** method of WebElement interface.

Selenium code below :

```
public class PrintVersion_ActitimeLoginPage extends BaseClass{
            public static void main(String[] args) {
            driver.get("http://localhost:8080/login.do");
            String xpathforVersion = "//nobr[contains(text(),'actiTIME')]";
            String version = driver.findElement(By.xpath(xpathforVersion)).getText();
            System.out.println("Version of actitime on login page is : " + version);
        }
}
```

*Write a script to verify that **View License link** on login page of Actitime application is a link or not ?*

Using **getTagName()** method of WebElement interface.

Selenium code below :

```
public class VerifyViewLicense_isaLinkOnActitimepage extends BaseClass {
            public static void main(String[] args) {
            driver.get("http://localhost:8080/login.do");
            String tagName = driver.findElement(By.id("licenseLink")).getTagName();
            if (tagName.equals("a")) {
                System.out.println("View LIcence is a link");
```

```
        } else{
                System.out.println("View LIcence is NOT a link");
        }
        driver.close();
    }
}
```

*Write a script to verify that **KeepMeLoggedIn checkbox** on login page of Actitime application is a checkbox or not ?*

Using **getAttribute()** method of WebElement interface.

Selenium code below :

```
public class VerifyKeepMeLoggedInisaCheckboxinActitime extends BaseClass{
    public static void main(String[] args) {
    driver.get("http://localhost:8080/login.do");
    String elementType =
    driver.findElement(By.id("keepLoggedInCheckBox")).getAttribute("type");
    System.out.println(elementType);
    if (elementType.equalsIgnoreCase("checkbox")) {
            System.out.println("it is a checkbox");
    }else{
            System.out.println("it is NOT a checkbox");
    }}}
```

*Write a script to demonstrate different options to **click on a button or on a link (Or any element)***

Using the below methods of WebElement interface.

1. click()
2. sendkeys()
3. submit()

Selenium code below :

```
public class diffwaysofClickingonaButton{
        public static void main(String[] args) throws InterruptedException {
                System.setProperty("webdriver.gecko.driver", "./driver/geckodriver.exe");
                WebDriver driver = new FirefoxDriver();
                driver.get("https://demo.vtiger.com");
                String xp = "//button[.='Sign in']";
                //1. using click() method
                driver.findElement(By.xpath(xp)).click();
                //2. using sendkeys
                driver.findElement(By.xpath(xp)).sendKeys(Keys.ENTER);
                /3. using submit() method
                this method will work only and only if if the element has an attribute called type= 'submit'/
                driver.findElement(By.xpath(xp)).submit();
        }}
```

*Write a script to verify the **color of the error message** on Actitime login page when user clicks on Login button without entering username and password ?*

Using **getCssValue()** method of WebElement interface.

Selenium code below :

```
public class VerifyErrormessageonActimeloginpage {
        public static void main(String[] args) {
                System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");
                WebDriver driver = new FirefoxDriver();
                driver.get("http://localhost:8080/login.do");
                //click on Login button
                driver.findElement(By.xpath("//div[.='Login ']")).click();
```

```
//find the error message element
WebElement errMsg =

driver.findElement(By.xpath("//span[contains(.,'invalid')]"));
// get the text of the error message
String errtext = errMsg.getText();
//print the error message
System.out.println("error message is :"+errtext);
//get the value of color and store  in a variable
String c = errMsg.getCssValue("color");
//convert the color from string type to hexa form
String ColorasHex = Color.fromString(c).asHex();
System.out.println("hexadecimal format : "+ColorasHex);

if(ColorasHex.equals(“#ce0100”)){

System.out.println(“Error message is in red color”);

}else{

System.out.println(“Error message is in red color”);

}

//get the size of the font of error message
String fontSize = errMsg.getCssValue("font-size");
//get the weight of the font of error message
String fontWeight = errMsg.getCssValue("font-weight");
System.out.println("Size of the font is :" + fontSize);
System.out.println("Weight of the font is :" + fontWeight);
driver.close();
        }
}
```

JavascriptExecutor

It is one of the interface in selenium which has below 2 methods.

1. executeScript()
2. executeAsyncScript()

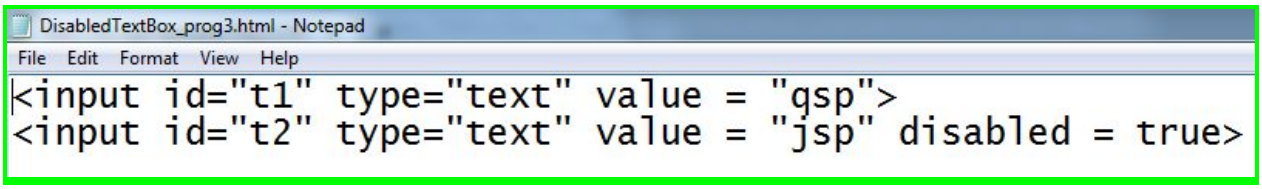
We use JavascriptExecutor when we fail to perform some actions using selenium.

*Write a script to **enter a text** in a **textbox** which is in **disabled mode** ?*

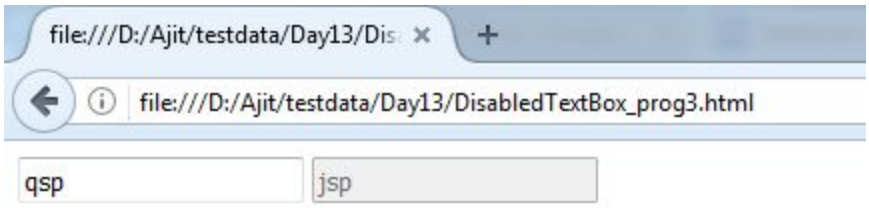
Using **sendKeys()** of **WebElement *interface***, *if we try to enter, we get **InvalidElementStateException***

Using **executeScript()** of ***JavascriptExecutor*** *interface, we can enter text in a disabled textbox.*

Create a sample webpage using the below html source code wherein the second textbox is disabled as shown below.



The webpage looks like this.



Selenium code below :

```
public class enterText_intoDisabledTextbox {

        public static void main(String[] args) throws InterruptedException {

                System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

                WebDriver driver = new FirefoxDriver();

                driver.get("file:///D:/Ajit/testdata/Day13/DisabledTextBox_prog3.html");

                //Typecast the driver object to JavascriptExecutor interface type
```

```
        JavascriptExecutor js = (JavascriptExecutor) driver;

        Thread.sleep(2000);

        //enter "admin" in first textbox using javascript

        js.executeScript("document.getElementById('t1').value='admin'");

        Thread.sleep(2000);

        //clear the value in second textbox using javascript

        js.executeScript("document.getElementById('t2').value=''");

        //enter "manager" in second textbox using javascript

        js.executeScript("document.getElementById('t2').value='manager'");

        //change the second text box to button type using Javascript

        js.executeScript("document.getElementById('t2').type='button'");
    }}
```

what are the usage of JavascriptExecutor ?

1. to scroll on the webpage.
2. to handle the disabled elements
3. to use as an alternate solution when selenium inbuilt methods (eg : clear(), click(), sendKeys()) doesn't work

In Selenium, we don't have any method to scroll up or down on the webpage, in such case, we can use **JavascriptExecutor.**

Steps to run javascript manually on browser webpage

1. Open the required page in the browser and press F12 from keyboard.
2. Navigate to Console tab, type the javascript statement and press Enter key

*Write a script to **scroll up and down** on Selenium official website*

Using executeScript() of *JavascriptExecutor interface*

Selenium code below :

```
public class ScrollUpandDown {

	public static void main(String[] args) throws InterruptedException {

		System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

		WebDriver driver = new FirefoxDriver();

		driver.get("http://seleniumhq.org/download");

		//typecasting driver object to JavascriptExecutor interface type

		JavascriptExecutor js = (JavascriptExecutor) driver;

		for (int i = 1; i < 10; i++) {

			//scroll down on the webpage

			js.executeScript("window.scrollBy(0, 1000)");

			Thread.sleep(3000);

		}

		for (int i = 1; i < 10; i++) {

			//scroll up on the webpage

			js.executeScript("window.scrollBy(0, -1000)");

			Thread.sleep(3000);

		}}}
```

*Write a script to **scroll down to a specific element (Applitool webelement)** on Selenium official website*

Using executeScript() of *JavascriptExecutor interface*

Selenium code below :

```
public class ScrollUpandDowntospecificElementonWebpage {

	public static void main(String[] args) throws InterruptedException {

		System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

		WebDriver driver = new FirefoxDriver();
```

```
        driver.get("http://seleniumhq.org/download");
        //click on the close icon  of the yellow color background pop up
        driver.findElement(By.id("close")).click();
        // find the Applitools element on the webpage
        WebElement ele =
        driver.findElement(By.xpath("//img[contains(@src,'applitools')]"));
        // get the X-coordinate and store in a variable
        int x = ele.getLocation().getX();
        // get the Y-coordinate and store in a variable
        int y = ele.getLocation().getY();
        JavascriptExecutor js = (JavascriptExecutor) driver;
        //Scroll to Applitools element's  x and y coordinate
        js.executeScript("window.scrollBy("+x+", "+y+")");
        Thread.sleep(3000);
    }}
```

Assignment : *Write a script to scroll down to the bottom of the page ?*

Using **executeScript() of *JavascriptExecutor*** *interface*

Selenium code below :

```
public class NavigatetoBottomofthePage {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");
        WebDriver driver = new FirefoxDriver();
        driver.get("http://www.seleniumhq.org/download/");
        driver.findElement(By.id("close")).click();
        //select an element which is present at the bottom of the page
        WebElement element = driver.findElement(By.id("footerLogo"));
```

```
        int x = element.getLocation().getX();

        int y = element.getLocation().getY();

        System.out.println("X coordinate is :"+x + " and Y coordinate is :"+ y);

        JavascriptExecutor js = (JavascriptExecutor) driver;

        js.executeScript("window.scrollBy("+x+","+y+")");

        Thread.sleep(3000);

        element.click();

    }}
```

Interview Question :

When do we go for Javascriptexecutor ?

Ans: When selenium conventional methods fails to perform an action on the webpage, we go for javascriptexecutor.

Q: How do you use javascriptExecutor ?

We typecast the driver object to javascriptexecutor interface, once we have the reference, we call executeScript() method, to which, as an argument, we pass the actual javascript statement.

Q: How do you get the javascript statement from the browser ?

Right click anywhere on the page, click on inspect , then navigate to console tab.

Write the lines of code :

```
JavascriptExecutor jse = (JavascriptExecutor) driver;

jse.executeScript("document.getElementById('un123').value='admin'");
```

****

HANDLING FRAMES

****

What is frame ?

1. Webpage present inside another webpage is called embedded webpage.
2. In order to create frame or embedded webpage, developer uses a tag called **iframe.**
3. In order to perform any operation on any element present inside a frame, we first have to switch the control to frame.
4. We switch to frame using the below statement

    driver.switchTo().frame(arg);

5. frame() is an overloaded method which accepts the following arguments.

    frame(index)

    frame(id)

    frame(name)

    frame(WebElement)

6. If the specified frame is not present, we get an exception called "NoSuchFrameException"

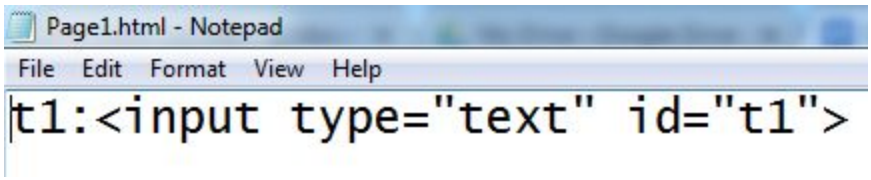
7. In order to exit from the frame, we use the following statements.

    driver.switchTo().defaultContent(); → it will take you to the main page

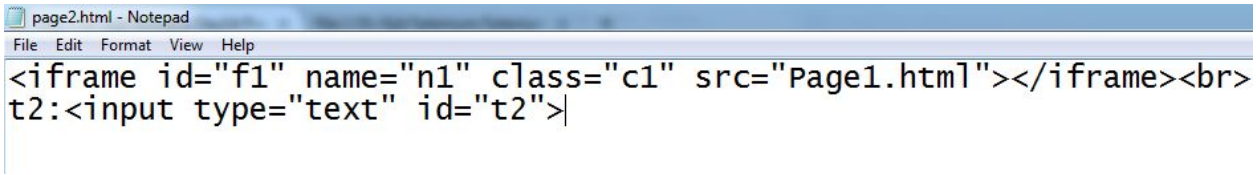
    driver.switchTo().parentFrame(); → it will take you to the immediate parent frame

8. Easiest way to verify that an element is present within a frame is to right click on the element and verify that **this frame** option is displayed in the context menu in Firefox browser.

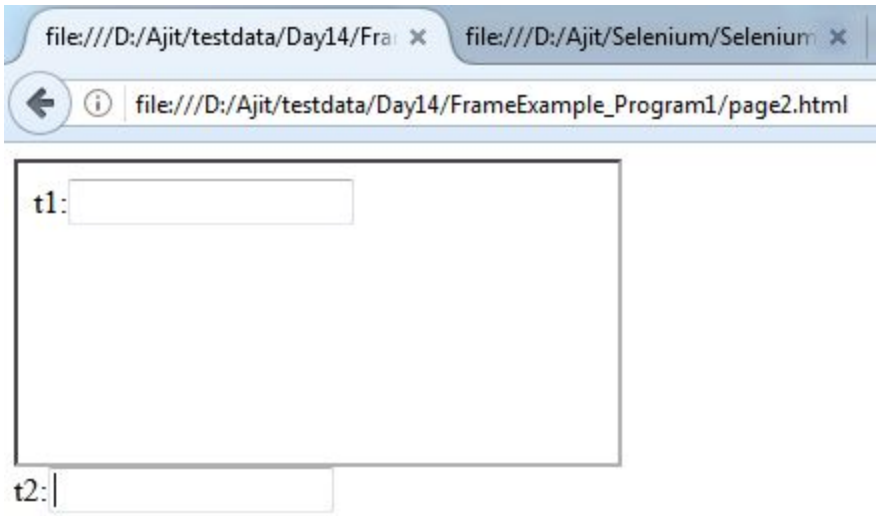
Create a sample webpage using the below html source code and save the file as Page1.html



Create another sample webpage using the below html source code and save the file as Page2.html



The webpage looks like this. Here, t1 is inside the frame and t2 is outside the frame on the webpage



Write a script to enter a text into an element which is present inside a frame ?

Selenium code:

```
public class Frame_Demo{

        public static void main(String[] args) {

                System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

                WebDriver driver = new FirefoxDriver();

                driver.get("file:///D:/Ajit/Selenium/SeleniumBtm_7thSep17/webpages/Frame_P
                age2.html");

                //using index of the frame - [ int value] [ index of frames starts with zero]

                driver.switchTo().frame(0);

                driver.findElement(By.id("t1")).sendKeys("a");

                driver.switchTo().defaultContent();

                driver.findElement(By.id("t2")).sendKeys("a");

                //using id attribute of the frame -string

                driver.switchTo().frame("f1");

                driver.findElement(By.id("t1")).sendKeys("b");
```

```
                driver.switchTo().defaultContent();
                driver.findElement(By.id("t2")).sendKeys("b");
                //using name attribute of the frame -string
                driver.switchTo().frame("n1");
                driver.findElement(By.id("t1")).sendKeys("c");
                driver.switchTo().defaultContent();
                driver.findElement(By.id("t2")).sendKeys("c");
                //using address of the frame -webelement
                WebElement f = driver.findElement(By.className("c1"));
                driver.switchTo().frame(f);
                driver.findElement(By.id("t1")).sendKeys("d");
                driver.switchTo().defaultContent();
                driver.findElement(By.id("t2")).sendKeys("d");
                driver.close();
        }
}
```

Q: Why the return type of frame() method is WEBDRIVER ?

frame() method returns the frame to which we are switching to, a frame is nothing but an html page, and in selenium, a page is always referred by driver object, here, driver is an object of Webdriver and hence the return type of frame() method is Webdriver.

Q: Why the return type of parentFrame() method is WEBDRIVER ?

parentFrame() method is used to switch to the immediate parent frame, a frame is nothing but an html page, and in selenium, a page is always referred by driver object, here, driver is an object of Webdriver and hence the return type of parentFrame() method is Webdriver.

Q: Why the return type of defaultContent() method is WEBDRIVER ?

defaultContent() method is used to switch to the default page or the main page, and in selenium, a page is always referred by driver object, here, driver is an object of Webdriver and hence the return type of defaultContent() method is Webdriver.

Q: Why the return type of window() method is WEBDRIVER ?

window() method is used to switch to a particular browser window, a browser window is nothing but a webpage, and in selenium, a page is always referred by driver object, here, driver is an object of Webdriver interface and hence the return type of window() method is Webdriver.

NoSuchFrameException:

When we try to switch to a frame by using any of the overloaded frame method such as by using frame(int), where we pass the index of the frame,but it does not match with the index of any frames on the webpage,

OR

when we pass any attribute value other than id or name

OR

when we pass the address of the frame which does not exist on the page, we get NoSuchFrameException.

ACTIONS Class :

→ Actions is a class present in selenium under a package called org.openqa.selenium.interactions.

→ Actions class is used to handle mouse related and keyboard related operations.

→ Actions class has same non-static methods, in order to use these methods, we have to create an object of the Actions class.

→ When we create an object of Actions class, we need to pass the driver object as an argument to the constructor of Actions class, so that we can instruct methods of Actions class to perform action on a particular page.

→ Then we can call few methods like :

1. moveToElement() :- this method is used to mouse hover on any element on the web page.
2. contextClick() :- this method is used to right click on any element on the web page

3. dragAndDrop() :- this method is used to drag an element from the source and drop it to any destination element.
4. doubleClick() :- this method is used to double click on any element on the web page
5. build() :- this method is used to combine multiple individual actions in to one single composite action.

→ For all the methods of Actions class, we need to explicitly call a method called perform(), until unless, we call perform() method, none of the Actions class methods will perform any action.

This is what Actions class is all about.

Interview Questions :

Why we need to pass driver object as an argument to Actions class constructor ?

We need to specify on which page, methods of Actions class to perform specific action, and hence, we pass driver

reference to Actions class constructor because driver object will be always pointing to the current page.

OR

Methods of Actions class is used to perform action on web elements present on the webpage.

And, On which page, we want methods of Actions class to perform specific action, we need to pass the reference of that particular page and in selenium, reference of any page will be always stored in driver object.

And this is why we pass driver object as an argument to Actions class constructor.

*How do you handle **Context Menu** in Selenium ?*

OR

*Write a script to right click on **"ActiTIME Inc."** link on actitime login page and then open it in new window ?*

*Using **contextClick()** method of Actions class*

Selenium code:

```
public class ContextClickusingActionsClass {

    //ContextClick does not work on firefox browser - pls do it on chromebrowser
```

```
public static void main(String[] args) throws AWTException, InterruptedException {
        System.setProperty("webdriver.chrome.driver", ".\\driver\\chromedriver.exe");
        //open the browser
        WebDriver driver = new ChromeDriver();
        //enter the url
        driver.get("http://localhost:8080/login.do");
        //find the ActiTIME Inc. link
        WebElement link = driver.findElement(By.linkText("actiTIME Inc."));
        //right click (context click) on actitime link
        Actions actions = new Actions(driver);
        actions.contextClick(link).perform();
        Thread.sleep(3000);
        //press 'w' from the keyboard for opening in a new window
        Robot r = new Robot();
        r.keyPress(KeyEvent.VK_W);
        r.keyRelease(KeyEvent.VK_W);
        //quit() method closes all the browsers opened by Selenium
        driver.quit();
}}
```

Imp Note :

Whenever we call any method of Actions class, we have to explicitly call perform() method of Actions class. Otherwise, it will not perform any action on the browser.

Assignment :

Automate the following scenario using contextClick() method of Actions Class.

Scenario Steps :

1. ***Login in to gmail***

2. ***Based on the subject of a mail, Right click on the mail***
3. ***Select Archive option***

Selenium Code:

```
public class gmail_contextClickDemo_mailArchive {

	public static void main(String[] args) throws InterruptedException {

		System.setProperty("webdriver.chrome.driver", ".\\driver\\chromedriver.exe");

		WebDriver driver = new ChromeDriver();

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

		//enter email id

		driver.findElement(By.xpath("//input[@type='email']")).sendKeys("enter your
		username");

		//click on Next button

		driver.findElement(By.xpath("//span[.='Next']")).click();

		Thread.sleep(3000);

		//enter password id

		driver.findElement(By.xpath("//input[@type='password']")).sendKeys("enter ");

		//click on Next button

		driver.findElement(By.xpath("//span[.='Next']")).click();

		Thread.sleep(10000);

		//Write xpath expression for the mail item based on a subject

		String xp = "(//b[contains(.,'Following Openings (for Bangalore)')])[2]";

		//get the address of the mail item which you want to archive

		WebElement mail = driver.findElement(By.xpath(xp));

		//print the subject of the mail

		System.out.println(mail.getText());

		//Creating an object of Actions class
```

```
        Actions actions = new Actions(driver);

        //using Actions class object and contextClick() method, right click on the mail item

        actions.contextClick(mail).perform();

        Thread.sleep(6000);

        //click on Archive to archive the mail

        driver.findElement(By.xpath("(//div[@class='J-N-JX aDE aDD'])[1]")).click();

    }}
```

Program :

*How do you **mouse hover** on any element on a web page ?*

*Answer : Using **moveToElement() of Actions** class*

Automate the following scenario using **moveToElement()** method of **Actions** Class.

Scenario Steps :

1. *Login in to http://www.actimind.com*
2. *Mouse hover on "About Company" menu*
3. *Click on Sub Menu - "Basic Facts"*

Selenium Code:

```
public class DropdownMenu {

    public static void main(String[] args) {

        System.setProperty("webdriver.chrome.driver", ".\\driver\\chromedriver.exe");

        //open the browser

        WebDriver driver = new ChromeDriver();

        driver.get("http://www.actimind.com/");

        //find the menu "About Company"

        String xp = "//span[.='About Company']";

        WebElement menu = driver.findElement(By.xpath(xp));
```

```
        //mouse hover on "About Company" menu
        Actions actions = new Actions(driver);
        actions.moveToElement(menu).perform();
        //click on submenu "Basic Facts"
        WebElement submenu = driver.findElement(By.linkText("Basic Facts"));
        submenu.click();
    }}
```

Scenario Steps :

4. *Login in to http://www.actimind.com*
5. *Mouse hover on "*AREAS OF EXPERTISE*" menu*
6. *Click on Sub Menu - "*Cloud Applicationss*"*

Selenium Code:

```
public class MouseHover{
    public static void main(String[] args) {
        driver.get("http://www.actimind.com/");
        Actions action = new Actions(driver);
        //movetoElement -  used for mouse hover
        //Mouse hover on "AREAS OF EXPERTISE" menu
        WebElement AreaOfExpertise =
driver.findElement(By.xpath("//a[contains(text(),'AREAS OF EXPERTISE')]"));
        action.moveToElement(AreaOfExpertise).perform();
        //Click  on "AREAS OF EXPERTISE" menu
        WebElement cloudApp = driver.findElement(By.linkText("Cloud Applicationss"));
        action.moveToElement(cloudApp).click().perform();
        //composite multiple actions can be achieved using the below statement
```

```
//action.moveToElement(AreaOfExpertise).moveToElement(cloudApp).click().build().perform();
        }}
```

Program :

*How do you **mouse hover** on any element on a web page ?*

*Answer : Using **moveToElement() of Actions** class*

Automate the following scenario using **moveToElement()** method of **Actions** Class.

Scenario Steps :

1. *Login in to **http://www.istqb.in***
2. *mouse hover on Foundation tab*
3. *mouse hover on Enrollment*
4. *mouse hover on Corporate Enrollment*
5. *click on Corporate Enrollment*

Selenium Code:

```
public class DropdownMenu {
        public static void main(String[] args) {
                System.setProperty("webdriver.chrome.driver", ".\\driver\\chromedriver.exe");
                //open the browser
                WebDriver driver = new ChromeDriver();
                driver.get("http://www.istqb.in/");
                WebElement foundation =
driver.findElement(By.xpath("//span[.='FOUNDATION']"));
                Actions actions = new Actions(driver);
                //mouse hover on Foundation tab
                actions.moveToElement(foundation).perform();
                Thread.sleep(3000);
```

```
WebElement enrollment =
driver.findElement(By.xpath("(//span[text()='ENROLLMENT'])[1]"));

//mouse hover on Enrollment

actions.moveToElement(enrollment).perform();

Thread.sleep(3000);

WebElement corporateEnrol =
driver.findElement(By.xpath("//span[text()='CORPORATE ENROLLMENT']"));

//mouse hover on Corporate Enrollment

actions.moveToElement(corporateEnrol).perform();

Thread.sleep(3000);

//click on Corporate Enrollment

driver.findElement(By.xpath("//span[text()='ONLINE ENROLLMENT']")).click();

driver.close();

}}
```

Program :

*How do you handle **DRAG and DROP** feature on a web page ?*

*Answer : Using **dragAndDrop()** method of **Actions** class*

Selenium Code:

```
public class DragAndDropExample {

    public static void main(String[] args) throws InterruptedException {

        System.setProperty("webdriver.chrome.driver", ".\\driver\\chromedriver.exe");

        WebDriver driver = new ChromeDriver();

        driver.get("http://www.dhtmlgoodies.com/submitted-scripts/i-google-like-drag-
        drop/index.html");

        String xp1 = "//h1[.='Block 1']";

        WebElement block1 = driver.findElement(By.xpath(xp1));

        String xp2 = "//h1[.='Block 3']";
```

```
        WebElement block3 = driver.findElement(By.xpath(xp2));

        Actions actions = new Actions(driver);

        // drag block 1 element and drop it on block 2 element

        actions.dragAndDrop(block1, block3).perform();

    }

}
```

Program :

*How do you handle **DRAG and DROP** feature on a web page ?*

***Answer :** Using **dragAndDropBy()** method of **Actions** class*

//hint - first find out the x-coordinate and height of block 3 and then add 10 points to it and then do it

Selenium Code:

```
public class DragAndDropbyOffset_Example {

    public static void main(String[] args) throws InterruptedException {

        System.setProperty("webdriver.chrome.driver", ".\\driver\\chromedriver.exe");

        WebDriver driver = new ChromeDriver();

        driver.get("http://www.dhtmlgoodies.com/submitted-scripts/i-google-like-drag-drop/index.html");

        //write xpath for Block 1

        String xp1 = "//h1[.='Block 1']";

        WebElement block1 = driver.findElement(By.xpath(xp1));

        //write xpath for Block 3

        String xp2 = "//h1[.='Block 3']";

        WebElement block3 = driver.findElement(By.xpath(xp2));

        //Create an object of Actions class and pass driver object as an argument
```

```
Actions actions = new Actions(driver);

//call the dragAndDropBy() method of Actions class

actions.dragAndDropBy(block1, block3.getLocation().getX()+10,
block3.getSize().getHeight()+10).perform();
}}
```

****

HANDLING POP UP

****

In selenium, pop up are categorized into following types.

1. Javascript Popup
2. Hidden Division Popup
3. File Upload popup
4. File download popup
5. Child browser popup
6. Window popup

1. *Javascript Pop up :*

This pop up is subdivided into below mentioned 3 pop ups.

1. **Alert pop up**
2. **Confirmation pop up**
3. **Prompt pop up**

1. Alert Pop up :

Characteristics features :

- **We can't inspect this pop up.**
- **We can't move this kind pop up.**
- **This pop up will have white color background with black color font.**
- **This pop up will have only one "OK" button**

How to handle Alert pop up

In order to handle the alert pop up, we first have to switch to alert window using the below statement.

driver.switchTo().alert();

After transferring the control to alert window, we can use the following methods of

"Alert" interface.

getText() → to get the text present on the alert window.

accept() / dismiss() → to click on OK button on the alert window.

2. Confirmation Pop up :

Characteristics features :

- **We can't inspect this pop up.**
- **We can't move this kind pop up.**
- **This pop up will have white color background with black color font.**
- **This pop up will have two buttons :- "OK" button and "Cancel" button.**

How to handle Prompt Alert pop up

- **In order to handle the alert pop up, we first have to switch to alert window**

  using the below statement.

driver.switchTo().alert();

- **After transferring the control to alert window, we can use the following methods of "Alert" interface.**

  getText() → to get the text present on the alert window.

  sendKeys() → to enter a text in the textbox on the alert window.

  accept() → to click on "OK" button on the alert window.

  dismiss() → to click on "Cancel" button on the alert window.

Selenium Code : to handle prompt alert popup on browser

```
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
public class Alert_Promptpopup {
    public static void main(String[] args) throws InterruptedException {
        System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");
        WebDriver driver = new FirefoxDriver();
        //Enter the url
        driver.get("http://www.tizag.com/javascriptT/javascriptprompt.php");
        //find this button : "Say my name"
        driver.findElement(By.xpath("//input[@value='Say my name!']")).click();
        Thread.sleep(2000);
```

```
            //Switch to alert pop up
            Alert alert = driver.switchTo().alert();
            Thread.sleep(2000);
            //print the text present on the alert pop up
            System.out.println(alert.getText());
            Thread.sleep(2000);
            //enter your name in the text box present on the alert pop up
            alert.sendKeys("ajit");
            Thread.sleep(2000);
            //click on OK button
            alert.accept();
            Thread.sleep(2000);
            //print the text present on the second alert pop up
            System.out.println(alert.getText());
            //click on Cancel button
            alert.dismiss();
        }
}
```

What are the Alert popup exceptions that you are aware of ?

1. NoAlertPresentException :

When the alert pop up is physically not available on the page, and we still try to either switch to the alert or try to perform any action on alert popup, we get this exception.

2. UnhandledAlertException :

We get this exception, when we perform any action on the webpage without handling the alert popup.

---

2. Hidden Division Popup

- **Any pop up which will be in hidden mode as soon as we navigate to the page, the moment we perform action ,pop up will appear and then it goes off . This is all about Hidden Division Pop up**
- **One of the examples for Hidden Division Pop up is Calendar pop up.**
- **If we are able to inspect the element present on the webpage then we can handle Hidden Division Pop up by using:**

driver.findElement(By.locator-name(" ")).click();

- **Here, click ( ) is the action that we are performing on webpage.**

If we are not able to inspect the element present on the webpage, then we go for some other option

How to handle geo location and notification in chrome

```
public class HiddenDivisionPopup_CalendarPopup_cleartrip_selectTodaysDate extends BaseClass {

    public static void main(String[] args) throws InterruptedException {

        System.setProperty("webdriver.chrome.driver", "./driver/chromedriver.exe");

        ChromeOptions option = new ChromeOptions();

        option.addArguments("--disable-notifications");

        option.addArguments("--disable-geolocation");

        option.addArguments("--ignore-certificate-errors");

        WebDriver driver = new ChromeDriver(option);

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

        Thread.sleep(3000);
```

```
                driver.findElement(By.xpath("(//input[@placeholder='Pick a
                date'])[1]")).click();

                Thread.sleep(3000);

                driver.findElement(By.linkText("24")).click();

        }

}
```

How to resize the browser window?

Browser window can be resized by using dimension class and DesiredCapabilities class.

Using dimension class:-

```
        Dimension dim = new Dimension(int,int);

          driver.manage().window().setSize(dim);
```

Using DesiredCapabilities class:- (deprecated) in current version

```
        ChromeOptions options = new ChromeOptions();

        options.addArguments("window-size=int,int");

        DesiredCapabilities cap=DesiredCapabilities.chrome();

        cap.setCapability(ChromeOptions.CAPABILITY, options);

        WebDriver driver = new ChromeDriver(cap);
```

How to handle geo location and notification in Firefox Browser ?

```
public class
Day15_Program2_HiddenDivisionPopup_CalendarPopup_cleartrip_selectTodaysDate extends
BaseClass {

        public static void main(String[] args) throws InterruptedException {

                Date d = new Date();

                String str = d.toString();
```

```
		String[] str2 = str.split(" ");

		String today = str2[2];

			System.setProperty("webdriver.gecko.driver",
"./driver/geckodriver.exe");

			DesiredCapabilities cap = DesiredCapabilities.firefox();

			FirefoxProfile profile = new FirefoxProfile();

			profile.setPreference("geo.enabled", false);

			cap.setCapability(FirefoxDriver.PROFILE, profile);

			WebDriver driver = new FirefoxDriver(cap);

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

			Thread.sleep(3000);

			driver.findElement(By.xpath("(//input[@placeholder='Pick a
			date'])[1]")).click();

			Thread.sleep(3000);

			driver.findElement(By.linkText("24")).click();

		}

}
```

3. File Upload Pop up:

What is file upload popup ?

A popup using which, we can upload a file from local system to destination server is called file upload popup.

How do we handle file upload popup ?

There are multiple ways of handling file upload popup.

1. Using any third party tool which can handle window based applications.

one such tool we have is called AutoIT.

2. by using sendkeys() method of webelement interface.Here, what we do is.

we pass the path of the file that we want to upload as an argument to sendkeys method. This is some kind of short cut approach to handle file upload popup. We can use this shortcut, only and only if the html source code of the upload button or browse button have an attribute called TYPE = "FILE"

eg:

html source code of upload button.

<input type="file" name = "upload">

uploadBtnObj.sendKeys("path of the file to be uploaded").

Don't tell this in interview :

we can handle this file upload popup using Robot class as well.

Program to demonstrate File Upload Pop up:

```
package test;

import java.awt.AWTException;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

public class FileUploadPopup_Demo  {

        public static void main(String[] args) throws InterruptedException, AWTException {

                System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

                WebDriver driver = new FirefoxDriver();


driver.get("http://nervgh.github.io/pages/angular-file-upload/examples/simple");

                Thread.sleep(2000);

                driver.findElement(By.xpath("//input[@multiple='']")).sendKeys("D:\\Ajit\\tes
                tdata\\Absolute xpath examples.xlsx");
```

```
Thread.sleep(2000);

driver.findElement(By.xpath("//button[@ng-click=\"item.upload()\"]")).click();

Thread.sleep(2000);

driver.close();
```

}}

4. File Download Pop up

Characteristic features:

- **We can move this popup but we can't inspect it.**
- **This pop up will have 2 radio buttons : Open with and Save File**

How to handle File Download pop up:

- **In Google Chrome browser, when we click on Download link of Java language present on Selenium official website, it doesn't show any file download pop up on the screen, instead, it automatically starts downloading the file in default location on the system. (i.e downloads folder)**
- **But, in firefox browser, on clicking on the same download link, we get a file download pop up on the screen. In order to handle this pop up, we use setPreference() method of FirefoxProfile class.**
- **setPreference() is used to change the settings of Firefox browser.**
- **setPreference() method is an overloaded method which takes 2 parameters (KEY, VALUE).**

  "*Key" will always be a String,*

  "Value" can be either String or int or boolean

- **For more information on Key , we can refer the following websites.**

  http://kb.mozillazine.org/About:config_entries#Browser

Following example demonstrates how to use key and value with setPreference() method.

```
FirefoxProfile profile = new FirefoxProfile();

// If the file type is .zip, then don't display the popup, instead, download it directly.

String key = "browser.helperApps.neverAsk.saveToDisk";

String value = "application/zip";

profile.setPreference(key, value);

// 0 - save to desktop, 1 - save to downloads folder (default value),

// 2 - save the downloaded file to other folders in the system

profile.setPreference("browser.download.folderList", 2);

profile.setPreference("browser.download.dir", "D:\\");
```

In the above example, "application/zip" refers to MIME types. (Multi purpose Internet Mail Extension), which says what kind of file you want to download.

For a detailed level information on MIME types (or the type of file to be downloaded), visit the following website.

https://www.freeformatter.com/mime-types-list.html

Interview Questions :

How do you handle file download popup ?

Ans :

File download popup is a window based application, which is not supported by selenium.

In order to handle such popup, we will have to use any third party tool that can handle window based applications. One such tool we have used in our project is AUTOIT, using which we have handled window based popups.

Instead of using AUTOIT, we can handle window based popup using FirefoxProfile class, this class we use to do any profile related settings in firefox browser.

FirefoxProfile class has a non static method called setPreference, which take 2 arguments - KEY and VALUE, Key is always string, and value can be either an int or boolean or string.

As part of the key and value, we specify some profile related settings - like ,, if we are downloading any .zip file, don't show the popup, Instead directly start downloading.

And finally, we pass this profile object to FirefoxDriver class constructor.

But, FirefoxDriver class has no constructor defined which takes an argument of FirefoxProfile class object.

And hence, we create an object of another class called FirefoxOptions, to which we assign the profile object, which has the modified settings.

And finally, we pass this option object to FirefoxDriver class constructor.

So, during the script execution, when it launch the firefox browser, it will get the instruction like , if user is trying to download any type of file, don't show the popup.

In this way, we have handled file download popup in our project.

Program : Write a script to download the selenium-java present on selenium official website without opening the file download pop up and save it to specific folder in any drive in your system.

Selenium Code:

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

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.openqa.selenium.firefox.FirefoxProfile;

import org.openqa.selenium.remote.DesiredCapabilities;

public class FileDownload {

        public static void main(String[] args) throws InterruptedException {

                System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

                //Create an object of FirefoxProfile class

                FirefoxProfile profile = new FirefoxProfile();

                //Set the Key so that it will not show the file download pop up on the screen

                String key = "browser.helperApps.neverAsk.saveToDisk";
```

```
//Set the type of file which you want to download

String value = "application/zip";

//using setPreference() method, change the setting

profile.setPreference(key, value);

// 0 - save to desktop, 1 - save to download folder( default), 2 - save to any other //location

profile.setPreference("browser.download.folderList", 2);

//save the file to the given folder location

profile.setPreference("browser.download.dir", "D:\\Ajit\\Others");

//Use DesiredCapabilities class to modify the firefox settings as shown below

DesiredCapabilities cap = DesiredCapabilities.firefox();

cap.setCapability(FirefoxDriver.PROFILE, profile);

//Launch the firefox browser with the above modified settings

WebDriver driver = new FirefoxDriver(cap);

//Enter selenium official website url

driver.get("http://www.seleniumhq.org/download/");

//Use following-sibling axes in Xpath to find the download link for selenium java

driver.findElement(By.xpath("//td[text()='Java']//following-sibling::td[3]/a")).click();

Thread.sleep(3000);

}}
```

Note : After the script is executed, verify that the file is downloaded in the specified folder location.

How do you download in Chrome Browser where in you will not get the file download pop up ?

```
package qspiders;
```

```
import java.util.HashMap;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;
import org.openqa.selenium.remote.DesiredCapabilities;

public class FileDownloadInChromeBrowser {
        public static void main(String[] args) throws InterruptedException {

System.setProperty("webdriver.chrome.driver","./driver/chromedriver.exe");
                //Create Hashmap object and assign the profile settings
                HashMap<String, Object> chromePrefs = new HashMap<String,
Object>();
                chromePrefs.put("profile.default_content_settings.popups", 0);
                chromePrefs.put("download.default_directory", "D:\\");
                //Assign this chromePrefs object with ChromeOptions object
                ChromeOptions options = new ChromeOptions();
                options.setExperimentalOption("prefs", chromePrefs);
                //Create Capability object and assign the option object
                DesiredCapabilities cap = DesiredCapabilities.chrome();
                cap.setCapability(ChromeOptions.CAPABILITY, options);
                WebDriver driver = new ChromeDriver(cap);
                driver.get("http://www.seleniumhq.org/download/");
                Thread.sleep(3000);
                String xp = "//td[.='Java']/following-sibling::td/a[.='Download']";
                driver.findElement(By.xpath(xp)).click();
```

```
        }
}
```

Child Browser Pop up:

How do you handle multiple browser windows using selenium ?

OR

How do you handle child browser popup in selenium ?

Ans :

1. We handle multiple browser windows using getWindowHandles() method of WebDriver interface.

2. getWindowHandles() method returns the window handle id of all the browsers launched by selenium in the form of Set of String.

3. If we want to perform any action on specific browser window, we first have to switch to that particular window.

4. And how we switch to a particular window is by using driver.switchTo().window().

As an argument to window() method, we pass the unique window id of the browser window to which we want to switch to.

And once our driver control is on any browser window, we can perform any action on any elements.

This is how we handle multiple browser windows in selenium.

Note:·

NoSuchSessionException:-

When we are trying to switch to a window by using windowHandle Id of a browser window and all the windows are already been closed by the quit() method, we get NoSuchSessionException.

NoSuchWindowException:-

when we are trying to switch to a browser window using window handle Id and it does not match with window handle Id of any browser window launched by the selenium, then we get no such window exception.

- **Characteristic features :**
  - **We can move this pop up.**
  - **We can also inspect it.**
  - **this pop up is very colorful and will have both minimise and maximise buttons.**

Difference between getWindowHandle() and getWindowHandles() ?

- **getWindowHandle() returns the window handle id of the current browser window.**
- **getWindowHandles() returns the window handle id of all the browser windows.**

Program to print the window handle of a browser window ?

Selenium Code :

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

import org.openqa.selenium.firefox.FirefoxDriver;

public class Print_windowHandle {

        public static void main(String[] args) throws InterruptedException {

                System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

                WebDriver driver = new FirefoxDriver();

                driver.get("http://localhost:8080/login.do");

                //get the window handle id of the browser

                String windowHandle = driver.getWindowHandle();
```

```
            System.out.println(windowHandle);

        }}
```

Program to print the window handle id of browser ?

Selenium Code :

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

import org.openqa.selenium.firefox.FirefoxDriver;

public class Print_windowHandle {

        public static void main(String[] args) throws InterruptedException {

            System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

            WebDriver driver = new FirefoxDriver();

            driver.get("http://localhost:8080/login.do");

            //get the window handle id of the browser

            String windowHandle = driver.getWindowHandle();

            System.out.println(windowHandle);

        }}
```

Program :

Scenario :

Write a script to automate the following scenarios:

1. **Count the number of browser windows opened by selenium**

2. Print the window handle of all the browser windows

3. Print the title of all the browser windows ?

4. Close all the browser windows.

Selenium Code :

```
public class ChildBrowserPopUp extends BaseClass{

    public static void main(String[] args) {

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

        //using getWindowHandles(), get a set of window handle IDs

        Set<String> allWindowHandles = driver.getWindowHandles();

        //using size(), get the count of total number of browser windows

        int count = allWindowHandles.size();

        System.out.println("Number of browser windows opened on the system is : "+
        count);

        for (String windowHandle : allWindowHandles) {

            //switch to each browser window

            driver.switchTo().window(windowHandle);

            String title = driver.getTitle();

            //print the window handle id of each browser window

            System.out.println("Window handle id of page -->"+ title +" -->  is :
            "+windowHandle);

            //close all the browsers one by one

            driver.close();

        }
```

```
                    /*Instead of using driver.close(), we can use driver.quit() to close all the browsers at once*/

                    //driver.quit();

}}
```

Program :

Write a script to close only the main browser window and not the child browser windows.

Selenium Code :

```
public class CloseMainBrowserOnly extends BaseClass{

        public static void main(String[] args) {

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

                //get the window handle id of the parent browser window

                String parentWindowhandleID = driver.getWindowHandle();

                Set<String> allWindowHandles = driver.getWindowHandles();

                int count = allWindowHandles.size();

                System.out.println("Number of browser windows opened on the system is : "+ count);

                for (String windowHandle : allWindowHandles) {

                        //switch to each browser window

                        driver.switchTo().window(windowHandle);

                        /* compare the window id with the Parent browser window id, if both are equal, then only close the main browser window.*/

                        if (windowHandle.equals(parentWindowhandleID)) {

                                driver.close();
```

```
                System.out.println("Main Browser window with title -->"+ title +" --> is
                closed");

                }}}}
```

Program :

Write a script to close all the child browser windows except the main browser.

Selenium Code :

```
public class CloseALLChildbrowsersONLY extends BaseClass{

        public static void main(String[] args) {

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

                //get the window handle id of the parent browser window

                String parentWindowhandleID = driver.getWindowHandle();

                Set<String> allWindowHandles = driver.getWindowHandles();

                int count = allWindowHandles.size();

                System.out.println("Number of browser windows opened on the system is : "+
                count);

                for (String windowHandle : allWindowHandles) {

                        //switch to each browser window

                        driver.switchTo().window(windowHandle);

                        String title = driver.getTitle();

                        /* compare the window id of all the browsers with the Parent browser
                        window id, if it is not equal, then only close the browser windows.*/

                        if (!windowHandle.equals(parentWindowhandleID)) {

                                driver.close();
```

```
                    System.out.println("Child Browser window with title -->"+ title
                    +" --> is closed");

                }}}}
```

Program :

Write a script to close the specified browser window ?

Selenium Code :

```
public class CloseAnySpecifiedBrowser extends BaseClass{

        public static void main(String[] args) {

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

                //Set the expected title of the browser window which you want to close

                String expected_title = "Tech Mahindra";

                Set<String> allWindowHandles = driver.getWindowHandles();

                int count = allWindowHandles.size();

                System.out.println("Number of browser windows opened on the system is : "+
                count);

                for (String windowHandle : allWindowHandles) {

                        //switch to each browser window

                        driver.switchTo().window(windowHandle);

                        String actual_title = driver.getTitle();

                        //Checks whether the actual title contains the specified expected title

                        if (actual_title.contains(expected_title)) {

                                driver.close();

                        System.out.println("Specified Browser window with title -->"+
                        actual_title +" --> is closed");
```

```
                }
            }
        }
}
```

Program :

Write a script to navigate between multiple tabs and perform some action on each tabs ?

Selenium Code :

```
public class HandleTabs_using_getWindowHandles extends BaseClass {
    public static void main(String[] args) {
        //enter actitime login url
        driver.get("http://localhost:8080/login.do");
        //get the window handle id of the parent browser window
        String parentwindowHandle = driver.getWindowHandle();
        //enter username
        driver.findElement(By.id("username")).sendKeys("admin");
        //enter password
        driver.findElement(By.name("pwd")).sendKeys("manager");
        //click on actiTIME INC link
        driver.findElement(By.xpath("//a[text()='actiTIME Inc.']")).click();
        //get the number of windows currently opened on the system
        Set<String> allwhs = driver.getWindowHandles();
        //switch to all the browser windows
        for (String wh : allwhs) {
```

```
                driver.switchTo().window(wh);
            }
            //get the title of the tab
            String childtitle = driver.getTitle();
            System.out.println("Title of the child tab is :"+ childtitle);
            //close the child tab
            driver.close();
            //switch back to the main browser window
            driver.switchTo().window(parentwindowHandle);
            //close the main browser window
            driver.findElement(By.xpath("//div[text()='Login ']")).click();
            //closing the parent window
            driver.close();
        }
```

Q1. What is GetWindowHandle() method ?

- **This is a method present in WebDriver interface.**
- **It returns the window handle id of the current browser window in the form of String.**

Q2. What is getWindowHandles() method ?

- **This is a method present in WebDriver interface.**

- **It returns the window handle id of all the browser windows launched by selenium in the form of Set<String>.**

Q3. Why the return type of getWindowHandles method is Set<String> ?

Window handle id are unique and hence it is SET and why String, because these ids are in the form of String and hence the return type of getWindowHandles method is set<String>.

Q4. What is Window Handle Id ?

Every browser window has a unique ID, using which selenium identifies a browser window uniquely. Using this window handle id, we perform action on any particular window.

Window Pop Up:

In Selenium, if the pop up displayed on the application doesn't belong to the following types,

- **JavaScript popup,**
- **Hidden Division pop up,**
- **File Upload pop up,**
- **File Download pop up,**
- **Child Browser pop up,**

then it belongs to a category called WINDOW POP UP

Characteristic features of Window pop up:

- **We can move some of the window popups and some of them, we can't.**

- **We can't inspect this pop up.**

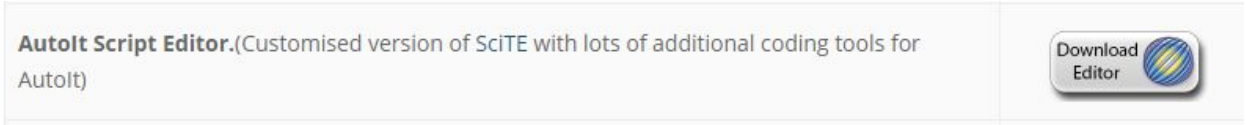
How to handle Window Popup ?

- **In selenium, there is no option to handle window pop up, hence, we have to use some third party tool like AUTOIT to handle this kind of pop up. We can also use ROBOT class to handle this pop up.**
- **But, by using ROBOT class, we can't achieve much functionalities, as it has limited option eg: we can't identify the object properties present on the window pop up.**
- **Hence, we use another third party automation tool called AUTO IT.**

What is AUTO IT ?

- **It is a open source window based automation tool.**
- **It can be downloaded from below mentioned site :**

  https://www.autoitscript.com/site/autoit/downloads



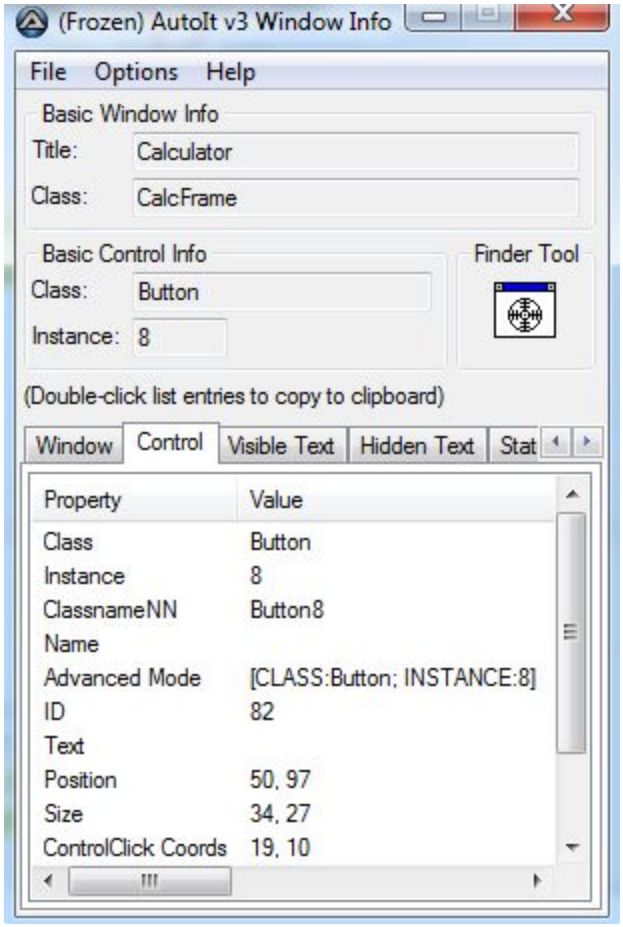
- **Download the above Editor on your system.**
- **Double click on the Setup file.**
- **Follow the default instruction to install autoIT.**

How Auto IT identifies objects on window popup ?

- **Elements present on window pop up are known as *CONTROLS*.**
- **In order to inspect these controls, AutoIT uses AutoIT Window Info".**
- **In order to open "AutoIT Window Info", navigate to the below path.**

  ***Go to Start → All Programs → AutoIT V3 → Select AutoIT Window Info*.**

- **As a result, the below window opens up.**



- **In the above image, drag the *"Finder Tool"* option and drop it on any element/control present on the window pop up for which you want to identify the properties.**
- **It will display the properties of the same controls such as *Class, Name, ID and Text.***
- **These properties are also known as CONTROL ID, using which AutoIT locates elements/controls on window pop up.**
- **General syntax for using single Control ID is :**

  [Control ID : Value]

- **We can use multiple Control IDs as well using semicolon as the delimiter to identify the controls using below syntax.**

[Control ID 1: Value1 ; Control ID 2 : Value2 ; Control ID 3 ; Value3]

Steps to write and execute AutoIT script :

- **Navigate to the below path and open the Editor to write the autoIT script**

  Go to Start → All Programs → AutoIt → Select *SciTE Script Editor*

- **Write the autoIT script and save the file with .au3 extension**
- **Go to Tool → Select Compile and compile script. As a result, it will generate an .exe file**
- **Navigate to the folder location where .exe file is located and double click on this .exe file to execute the autoIT script.**
- **We can also execute the script from eclipse by using RunTime class of Java**

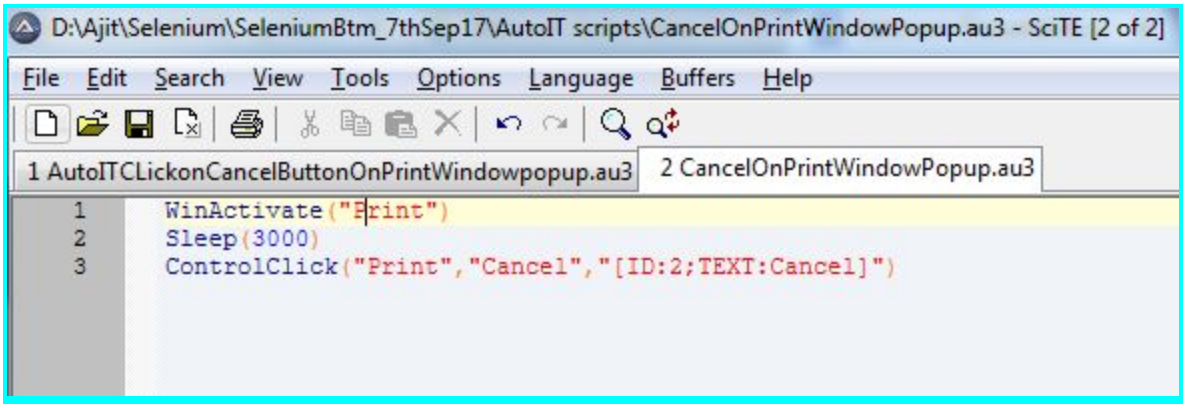
RunTime.getRunTime.exec("path of the compiled au3.exe file");

Automate the following scenario using AutoIT :

1. **Navigate to actiTIME login page.**
2. **By default, username text box will be active.**
3. **Press Control + P using Robot class and ensure the print window popup is displayed**
4. **On the Print window, click on Cancel button by using AutoIT**

Selenium Code:

Write the below lines of code in AutoIT editor, save with .au3 extension.



Go to tools → select compile and as a result, .exe file gets generated.

Now, write the below selenium code to run the .exe file

```
public class AutoIT_Example {

    public static void main(String[] args) throws InterruptedException, AWTException,
    IOException {

        System.setProperty("webdriver.gecko.driver", ".\\driver\\geckodriver.exe");

        WebDriver driver = new FirefoxDriver();

        driver.get("http://localhost:8080/login.do");

        Thread.sleep(3000);

        //Press Control + P from keyboard using Robot class

        Robot r = new Robot();

        r.keyPress(KeyEvent.VK_CONTROL);

        r.keyPress(KeyEvent.VK_P);

        r.keyRelease(KeyEvent.VK_P);

        r.keyRelease(KeyEvent.VK_CONTROL);

        //Using Runtime class, to run the .exe file

        Runtime run = Runtime.getRuntime();

        run.exec("D:\\Ajit\\Selenium\\SeleniumBtm_7thSep17\\AutoIT
        scripts\\CancelOnPrintWindowPopup.exe");

        //close the browser

        driver.close();
```

}}

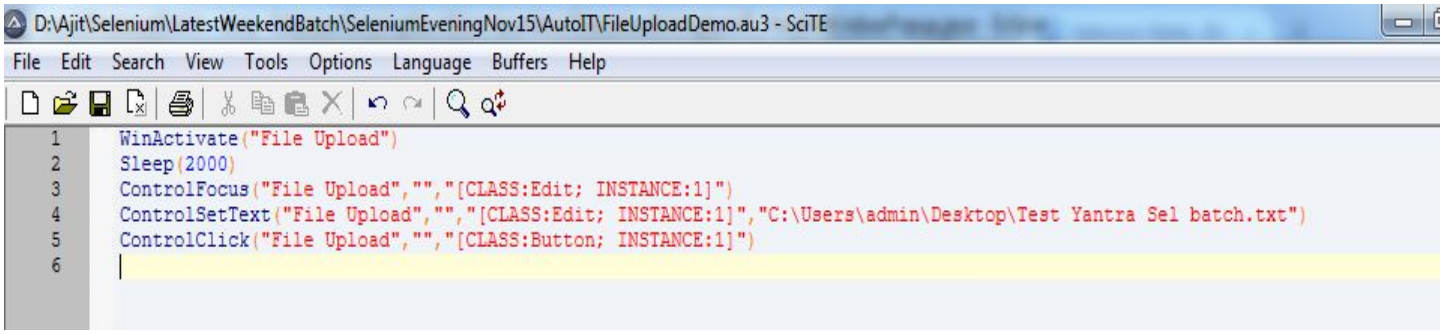
How to upload a file using AutoIT?

Code below:

*FileUploadUsingAutoIT_HandleWindowPopup.java

```
1  package qspiders;
2  import java.io.IOException;
3  import org.openqa.selenium.By;
4  public class FileUploadUsingAutoIT_HandleWindowPopup extends BaseClass{
5      public static void main(String[] args) throws IOException, InterruptedException {
6
7          driver.get("http://nervgh.github.io/pages/angular-file-upload/examples/simple/");
8          driver.findElement(By.xpath("(//input[@uploader='uploader'])[2]")).click();
9          Thread.sleep(2000);
10         Runtime.getRuntime().exec(".\\AutoIT\\FileUploadDemo.exe");
11     }
12 }
13
```

Open the autoit script editor and write the below script.



Summary of the different popups in selenium and how to handle those is mentioned below.

Popup Type	Solution
Javascript - Alert	driver.switchTo().alert() **Methods:** accept(), dismiss(), getText(), sendKeys()
Hidden Division pop up	findElement()
File Upload pop up	BrowseButton.sendKeys("Absolute path of the file")
File Download pop up	FirefoxProfile.setPreference(Key, Value)
Child Browser pop up	driver.getWindowHandles(); driver.switchTo().window("window handle ID")
Window pop up	Robot Class / Auto IT tool

How do you handle window based application using selenium ?

Ans:

How to handle Window based popup/ window based applications ?

Ans :Selenium can't handle window based popup and hence, in order to handle such popup, we have to use any third party tool which can handle window based popups. One such tool, we have is AUTOIT. using which, window based applications can be handled. What we do here is, we develop autoit scripts based on our requirement and then we save the file with .au3 extension. This file, we further compile to get an .exe version. How we compile is -- Tools -- compile..It will generate a compiled autoit script (.exe) in the same location where .au3 file is present.

This .exe file we execute by using exec() method of RunTime class.

The line of code is

Runtime.getRuntime().exec("./autoIT/printpopup.exe");

what we pass as an argument to exec() method is the path of the autoit compiled script.

When the script executes, it performs action on the window based popup. and this is how it handles window based applications.

What is findElements() ?

- **findElements() method is present in SearchContext interface, the super most interface in Selenium.**
- **findElements() identifies the elements on the webpage based on the locators used.**
- **It returns a list of webElements if it find the matching element.**
- **If it does not find any matching web element on the web page, it returns an empty list.**

Program : Write a script to find the total number of links, number of visible links and number of hidden links present on actitime login page.

Selenium Code :

```
public class findElements_Example extends BaseClass {
        public static void main(String[] args) throws InterruptedException {
                driver.get("http://localhost:8080/login.do");
                //findElements() method returns list of web element
                List<WebElement> allLinks = driver.findElements(By.tagName("a"));
                //get the total number of link elements
```

```
		int totalLinks = allLinks.size();
		System.out.println("total number of links present on the web page is : "+totalLinks);
		int visibleLinkCount = 0;
		int hiddenLinkCount = 0;
		//using foreach loop, iterate through all the links
		for (WebElement link : allLinks) {
			//if the link is displayed, then print the text of the link
			if (link.isDisplayed()) {
				visibleLinkCount++;
				System.out.println(visibleLinkCount+" --> "+link.getText());
			}else{
				hiddenLinkCount++;
			}
		}
		System.out.println("Total number of visible links :" + visibleLinkCount);
		System.out.println("Total number of hidden links :" + hiddenLinkCount);
		driver.close();
}}
```

Assignment :

Automate the following scenario

- **Login in to actitime**
- **click on Tasks**

- **Count the total number of checkbox present on the page**
- **Select all the checkbox**
- **Deselect all the checkboxes in reverse order**
- **Select first and last checkbox**

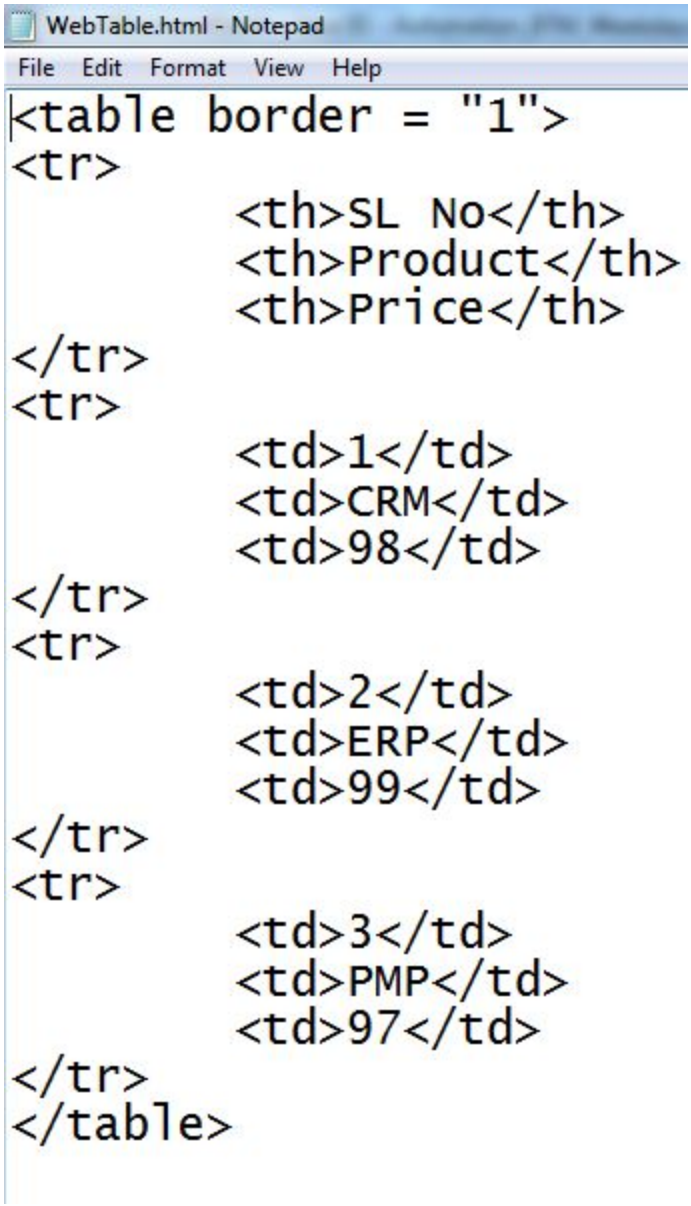
Selenium Code:

Copy code here.

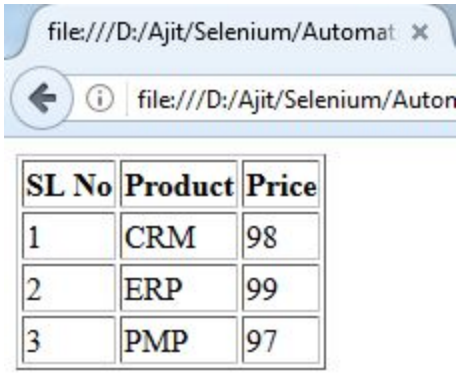
WebTable :

Table present on the web page is called WebTable.

Create a webtable as shown below.



The webpage looks like this as shown below.



SL No	Product	Price
1	CRM	98
2	ERP	99
3	PMP	97

Program :

In the below webtable, find the following scenarios :

- **print the total number of ROWS present**
- **print the total number of COLUMNS present**
- **print the total number of CELLS present**
- **print ONLY the NUMERIC values present**
- **Count the TOTAL number of NUMERIC values present**
- **print the SUM of all the numeric values in the table**

Selenium Code:

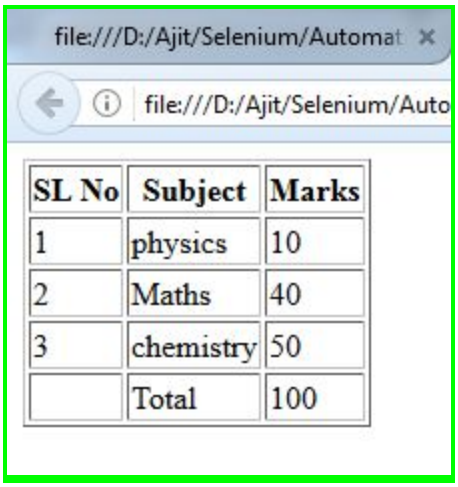
```
public class WebTable_Example extends BaseClass{

        public static void main(String[] args) {


                driver.get("D:\Ajit\Selenium\SeleniumBtm_7thSep17\webpages\WebTable.html");

                //Count Total number of rows present in the table

                List<WebElement> allRows = driver.findElements(By.xpath("//tr"));
```

```
int totalRows = allRows.size();
System.out.println("total number of rows present in the table is :"+
totalRows);
//count total number of columns
List<WebElement> allColumns = driver.findElements(By.xpath("//th"));
int totalColumns = allColumns.size();
System.out.println("Total number of columns in the table is :" + totalColumns);
//Count number of cells present in the table
List<WebElement> allCells = driver.findElements(By.xpath("//th|//td"));
int totalCells = allCells.size();
System.out.println("Total number of cells present in the table is :" + totalCells);
//Print ONLY the numbers
int countNumberValue = 0;
int sum=0;
for (WebElement cell : allCells) {
        String cellValue = cell.getText();
        try{
                int number = Integer.parseInt(cellValue);
                System.out.print(" "+number);
                countNumberValue++;
                sum = sum+number;
        }catch (Exception e) {
        }
}
System.out.println("Total count of numeric values is :"+countNumberValue);
```

```
            System.out.println("Total sum of all the numeric values is :"+sum);
            //close the browser
            driver.close();
        }
}
```

Assignment :

Write a script to verify that the sum of marks present in the below table is same as the Total marks.



SL No	Subject	Marks
1	physics	10
2	Maths	40
3	chemistry	50
	Total	100

HTML code to create the sample webpage is below.

WebTable_StudentMarks.html - Notepad

File Edit Format View Help

```
<table border = "1">
<tr>
        <th>SL No</th>
        <th>Subject</th>
        <th>Marks</th>
</tr>
<tr>
        <td>1</td>
        <td>physics</td>
        <td>10</td>
</tr>
<tr>
        <td>2</td>
        <td>Maths</td>
        <td>40</td>
</tr>
<tr>
        <td>3</td>
        <td>chemistry</td>
        <td>50</td>
</tr>
<tr>
        <td></td>
        <td>Total</td>
        <td>100</td>
</tr>
</table>
```

Selenium Code :

copy the code here.

How to handle Auto Suggestion list box ?

Answer : Using findElements() method

Program :

Automate the following scenario :

- **Navigate to google page**
- **Enter Selenium in google search text box**
- **Print the list of auto suggestion values**
- **Click on a specified link (Selenium Interview Questions) displayed in the dropdown**

Selenium Code :

```
public class AutosuggestionEx_GoogleSearch extends BaseClass{
        public static void main(String[] args) throws InterruptedException {
                driver.get("http://www.google.com");
                //Enter Selenium in google search text box
                driver.findElement(By.id("lst-ib")).sendKeys("selenium");
                Thread.sleep(2000);
                List<WebElement> allOptions =
                driver.findElements(By.xpath("//*[contains(text(),'selenium')]"));
                int count = allOptions.size();
                System.out.println("Number of values present in the dropdown is : " + count);
                String expectedValue="selenium interview questions";
                //Print all the auto suggestion values
```

```
                for (WebElement option : allOptions) {
                        String text = option.getText();
                        System.out.println("   "+text);
                //Click on Java Interview Questions
                if (text.equalsIgnoreCase(expectedValue)) {
                                option.click();
                                break;
                        }
}
```

How to select List Box ?

How do you handle list box ?

LIst box can be handled by 2 ways.

1 by using findElements() method.

2. by using select class. We can use select class only and only if the list box is develop using select tagname.

Select class has multiple non static methods, in order to call them, we need to create an object of select class.

When we create an object of Select class, we pass the reference of the list box on which we want to perform actions as an argument to the select class constructor.

using this reference variable, we call methods like,

-- getOptions() -- this method returns the address of all the options present in the list box in the form of list of webelement.

-- getAllSelectedOptions() -- this method returns the address of all the selected options from the list box in the form of list of webelement.

If none of the elements are selected in the list box, it returns an empty list object.

-- getFirstSelectedOption() -- this method returns the address of the first selected option in the list box.

If none of the elements are selected in the list box, it throws NoSuchElementException, because this method internally calls findElement method.

In order to select any option in the list box, we have 3 methods like,

-- selectByIndex() -- this method is used to select any option in the list box by using the index. Index of element in the list box starts from 0.

If specified index doesn't match with any element in the list box, it throws NoSuchElementException.

-- selectByValue() -- this method is used to select any option in the list box by using the value attribute.

If specified value doesn't match with any element in the list box, it throws NoSuchElementException.

-- selectByVisibleText() -- this method is used to select any option in the list box by using the text of the element.

If specified text doesn't match with any element text in the list box, it throws NoSuchElementException.

If the list box is of type multi select, we can also deselect any option in the list box which is already selected By using few methods like,

--deselectByIndex(), deselectByValue(), deselectByVisibleText() and deselectAll().

if the list box is not of type multi select, and if we try to call any of the deselect methods, we get an exception called UnsupportedOperationException.

In order to check whether the list box is single select or multi select, we can use isMultiple() method.

This method returns true if the list box is a multi select list box. and it returns false, if it is single select list box.

This is how we handle list box using SELECT class.

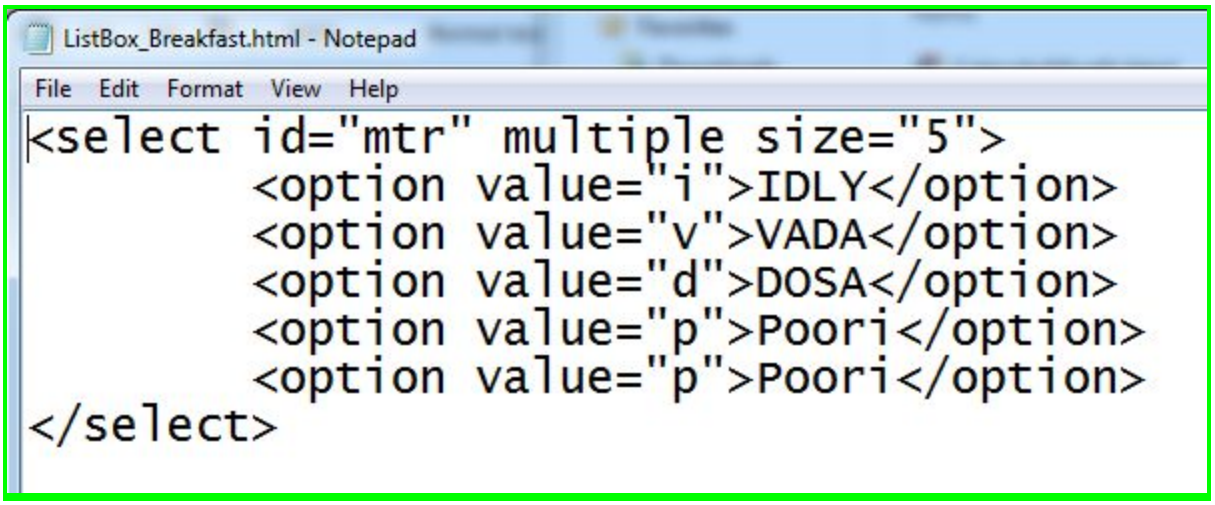
- **In Selenium, we handle listbox using Select class and findElements() method.**
- **Select class is present in org.openqa.selenium.support.ui package.**
- **Select class has a parameterized constructor which accepts an argument of WebElement object (List box element)**
- **Following are the available methods of Select class**
  - ➔ **selectByIndex()**
  - ➔ **selectByValue()**
  - ➔ **selectByVisibleText()**
  - ➔ **deSelectByIndex()**
  - ➔ **deSelectByValue()**
  - ➔ **deSelectByVisibleText()**
  - ➔ **isMultiple()**
  - ➔ **getOptions()**
  - ➔ **getAllSelectedOptions()**
  - ➔ **getFirstSelectedOption()**
  - ➔ **deSelectAll()**

- **We can use the following deSelect() methods only on multi select listbox. If we try to use it on single select list box, then it throws *UnsupportedOperationException***
  - ➔ **deSelectByIndex()**
  - ➔ **deSelectByValue()**
  - ➔ **deSelectByVisibleText()**
  - ➔ **deSelectAll()**

Program:

Write a script to select few elements in the list box.

Create a sample webpage



Selenium Code :

```
public class ListBoxExample extends BaseClass{

        public static void main(String[] args) {

                driver.get("file:///D:/Ajit/Selenium/SeleniumBtm_7thSep17/webpages/ListBo
                x_Breakfast.html");

                WebElement list = driver.findElement(By.id("mtr"));

                //Create an object of Select class and pass the address of list box as an
argument

                Select s = new Select(list);

                //getOptions() method returns a list of all the elements of the list box

                List<WebElement> options = s.getOptions();

                int size = options.size();

                System.out.println("Number of elements present inside the listbox is : "+ size);
```

```
		//Print all the elements present in the list box
		for (WebElement webElement : options) {
			String text = webElement.getText();
			System.out.println(text);
		}
		//selectByIndex() selects an element based on the Index, here index starts with
0
		s.selectByIndex(0);
		//selectByValue() method selects an element based on its value attribute.
		s.selectByValue("v");
		/*selectByVisibleText() method selects an element based on the actual text
		that is visible to the user. For instance, if there are multiple Poori present inside
		the listbox , it will select all the Poori elements.*/
		s.selectByVisibleText("Poori");
		System.out.println("************Print all selected options***********");
		List<WebElement> allSelectedOptions = s.getAllSelectedOptions();
		int size2 = allSelectedOptions.size();
		System.out.println("Number of items that is selected in the list box is :
"+size2);
		System.out.println(" Selected items are printed below ");
		for (WebElement webElement : allSelectedOptions) {
			System.out.println(webElement.getText());
		}
		System.out.println("check whether it is a multiple select listbox or not");
		boolean multiple = s.isMultiple();
		System.out.println(multiple +" yes , it is multi select");
```

```
            if (multiple) {
                //Print the first selected option in the list box
                WebElement firstSelectedOption = s.getFirstSelectedOption();

                System.out.println(firstSelectedOption.getText()+" is the first selected
                item in the list box");
                //deselect the item present in 0th index.
                s.deselectByIndex(0);
                //Print the first selected option in the list box
                WebElement firstSelectedOption1 = s.getFirstSelectedOption();
                System.out.println(firstSelectedOption1.getText()+" is the first selected
                item");
                //deselect an item which has an attribute called value and its value is
"v"
                s.deselectByValue("v");
                //Print the first selected option in the list box
                WebElement firstSelectedOption2 = s.getFirstSelectedOption();
                System.out.println(firstSelectedOption2.getText()+" is the first selected
                item");
                s.deselectByVisibleText("Poori");
            }
        }
}
```

Program:

Write a script to print the content of the list box in sorted order.

Selenium Code :

```
public class PrintListValues_SortedOrder extends BaseClass{

        public static void main(String[] args) throws InterruptedException {
        driver.get("file:///D:/Ajit/Selenium/SeleniumBtm_7thSep17/webpages/ListBo
        x_Breakfast.html");

        WebElement listElement = driver.findElement(By.id("mtr"));

        Select s = new Select(listElement);

        List<WebElement> allOptions = s.getOptions();

        int count = allOptions.size();

        System.out.println(count);

        System.out.println("-----print the values in the list ----");

        ArrayList<String> list = new ArrayList<String>();

        for (WebElement option : allOptions) {

                String text = option.getText();

                System.out.println(text);

                list.add(text);

        }

        Collections.sort(list);

        System.out.println("-----print the value in sorted order----");

        for (String value : list) {

                System.out.println(value);
```

}}}

Program:

Write a script to print the UNIQUE content of the list box.

Hint : Use HashSet<>

Selenium Code :

```
public class printUniqueElementinthelistbox  extends BaseClass{

public static void main(String[] args) throws InterruptedException {


driver.get("file:///D:/Ajit/Selenium/SeleniumBtm_7thSep17/webpages/ListBox_Breakfast.ht
ml");
                WebElement listElement = driver.findElement(By.id("mtr"));

                Select s = new Select(listElement);

                List<WebElement> allOptions = s.getOptions();

                int count = allOptions.size();

                System.out.println(count);

                System.out.println("-----print the values in the list ----");

                HashSet<String> allElements = new HashSet<String>();

                for (WebElement option : allOptions) {

                        String text = option.getText();

                        System.out.println(text);

                        allElements.add(text);

                }

                System.out.println(allElements);

        }
```

}

Program:

Write a script to print the UNIQUE content of the list box in SORTED order.

Hint : Use TreeSet<>

Selenium Code :

```
public class printUniqueElement_Sorted  extends BaseClass{

public static void main(String[] args) throws InterruptedException {


driver.get("file:///D:/Ajit/Selenium/SeleniumBtm_7thSep17/webpages/ListBox_Breakfast.html");
                WebElement listElement = driver.findElement(By.id("mtr"));
                Select s = new Select(listElement);
                List<WebElement> allOptions = s.getOptions();
                int count = allOptions.size();
                System.out.println(count);
                System.out.println("-----print the values in the list ----");
                TreeSet<String> allElements = new TreeSet<String>();
                for (WebElement option : allOptions) {
                        String text = option.getText();
                        System.out.println(text);
                        allElements.add(text);
                }
                System.out.println(allElements);
        }
}
```

Program:

Write a script to check whether listbox has duplicate or not ?

Selenium Code :

```
public class checklisthasDUPLICATEvalues_HashSet  extends BaseClass{

   public static void main(String[] args) {


                driver.get("file:///D:/Ajit/Selenium/AutomationByBhanuSir_BTM/testdataFile
                s/ListBox_Breakfast.html");

      WebElement listbox = driver.findElement(By.id("mtr"));

      Select s = new Select(listbox);

      List<WebElement> allOptions = s.getOptions();

      int count1 = allOptions.size();

      System.out.println("Number of elements in the list is :" +count1);

      HashSet<String> allElementText = new HashSet<String>();

      for (int i = 0; i < count1; i++) {

         String text = allOptions.get(i).getText();

         System.out.println(text);

         allElementText.add(text);

      }

      int count2 = allElementText.size();

      System.out.println("Number of elements in the hashset is :" +count2);

      if (count1==count2) {

         System.out.println("list box has NO duplicate values");

      }

      else{

         System.out.println("list box has  duplicate values");

      }
```

```
    System.out.println(allElementText);

    driver.close();

  }}}
```

Program:

Write a script to print the duplicate item in the list ?

Selenium Code :

```
public class PrinttheDUPLICATEItem_intheList_HashSet  extends BaseClass{

  public static void main(String[] args) {


            driver.get("file:///D:/Ajit/Selenium/AutomationByBhanuSir_BTM/testdataFile
            s/ListBox_Breakfast.html");

    WebElement listbox = driver.findElement(By.id("mtr"));

    Select s = new Select(listbox);

    List<WebElement> allOptions = s.getOptions();

    int count1 = allOptions.size();

    System.out.println("Number of elements in the list is :" +count1);

    HashSet<String> allElementText = new HashSet<String>();

    for (int i = 0; i < count1; i++) {

      String text = allOptions.get(i).getText();

                  /*allElementText.add(text) returns true if the element is not already
                  added, and it returns false if the same element is trying to be added
                  twice.  */

      if (!allElementText.add(text)) {

        System.out.println(text +" is the duplicate item in the list box");

      }

    }

    System.out.println(allElementText.size());
```

```
      // it will print all the unique values in the HashSet object
      System.out.println(allElementText);
      driver.close();
   }
}
```

Program :

Print the number of occurrance of Poori in the list box.

Selenium Code

```
package qspiders;
import java.util.HashMap;
import java.util.List;
import java.util.Set;

import org.openqa.selenium.By;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.ui.Select;
public class HashMapExample_printtheOcuuranceOfPoori extends BaseClass{
        public static void main(String[] args) {

driver.get("file:///D:/Ajit/Selenium/SeleniumBtm_7thSep17/webpages/ListBox_Breakfast.html"
);
                WebElement list = driver.findElement(By.id("mtr"));
                Select  s = new Select(list);
                List<WebElement> allElements = s.getOptions();

                HashMap<String, Integer> hashMapObj = new HashMap<String, Integer>();
```

```
                for (WebElement element : allElements) {
                        String text = element.getText();
                        if (hashMapObj.containsKey(text)) {
                                Integer value = hashMapObj.get(text);
                                value++;
                                hashMapObj.put(text, value);
                        }else{
                                hashMapObj.put(text, 1);
                        }
                }
                Set<String> allKeys = hashMapObj.keySet();
                for (String key : allKeys) {
                        Integer value = hashMapObj.get(key);
                        System.out.println(key +" -->"+ value);

                        if (value>1) {
                                System.out.println("Occurance of " + key + " is :" + value);
}}}}
```

Program:

Write a script to check whether listbox has duplicate or not ?

Selenium Code :

```
public class checklisthasDUPLICATEvalues_HashSet  extends BaseClass{
   public static void main(String[] args) {
```

```
            driver.get("file:///D:/Ajit/Selenium/AutomationByBhanuSir_BTM/testdataFile
            s/ListBox_Breakfast.html");
    WebElement listbox = driver.findElement(By.id("mtr"));
    Select s = new Select(listbox);
    List<WebElement> allOptions = s.getOptions();
    int count1 = allOptions.size();
    System.out.println("Number of elements in the list is :" +count1);
    HashSet<String> allElementText = new HashSet<String>();
    for (int i = 0; i < count1; i++) {
       String text = allOptions.get(i).getText();
       System.out.println(text);
       allElementText.add(text);
    }
    int count2 = allElementText.size();
    System.out.println("Number of elements in the hashset is :" +count2);
    if (count1==count2) {
       System.out.println("list box has NO duplicate values");
    }
    else{
       System.out.println("list box has  duplicate values");
    }
    System.out.println(allElementText);
    driver.close();
  }}}
```

Program:

Write a script to print the duplicate item in the list ?

Selenium Code :

```
public class PrinttheDUPLICATEItem_intheList_HashSet  extends BaseClass{

   public static void main(String[] args) {


                driver.get("file:///D:/Ajit/Selenium/AutomationByBhanuSir_BTM/testdataFile
                s/ListBox_Breakfast.html");

      WebElement listbox = driver.findElement(By.id("mtr"));

      Select s = new Select(listbox);

      List<WebElement> allOptions = s.getOptions();

      int count1 = allOptions.size();

      System.out.println("Number of elements in the list is :" +count1);

      HashSet<String> allElementText = new HashSet<String>();

      for (int i = 0; i < count1; i++) {

         String text = allOptions.get(i).getText();

                     /*allElementText.add(text) returns true if the element is not already
                     added, and it returns false if the same element is trying to be added
                     twice.  */

         if (!allElementText.add(text)) {

            System.out.println(text +" is the duplicate item in the list box");

         }

      }

      System.out.println(allElementText.size());

      // it will print all the unique values in the HashSet object

      System.out.println(allElementText);

      driver.close();

   }}
```

Program :

Print the number of occurrance of Poori in the list box.

Selenium Code

```
package qspiders;

import java.util.HashMap;

import java.util.List;

import java.util.Set;


import org.openqa.selenium.By;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.ui.Select;

public class HashMapExample_printtheOcuuranceOfPoori extends BaseClass{

	public static void main(String[] args) {


driver.get("file:///D:/Ajit/Selenium/SeleniumBtm_7thSep17/webpages/ListBox_Breakfast.html"
);

		WebElement list = driver.findElement(By.id("mtr"));

		Select  s = new Select(list);

		List<WebElement> allElements = s.getOptions();


		HashMap<String, Integer> hashMapObj = new HashMap<String, Integer>();

		for (WebElement element : allElements) {

			String text = element.getText();

			if (hashMapObj.containsKey(text)) {

				Integer value = hashMapObj.get(text);
```

```
                    value++;
                    hashMapObj.put(text, value);
                }else{
                    hashMapObj.put(text, 1);
                }
            }
            Set<String> allKeys = hashMapObj.keySet();
            for (String key : allKeys) {
                Integer value = hashMapObj.get(key);
                System.out.println(key +" -->"+ value);

                if (value>1) {
                    System.out.println("Occurance of " + key + " is :" + value);
}}}}
```

Page Object Model - Framework Design pattern

Definition :

Page object model is a page factory design pattern. We implement this model in our automation framework because of the following advantages listed below.

Advantages of POM :

- **Easy to Maintain/low maintenance**
- **Easy readability of scripts**
- **Reduce or eliminate duplicacy**
- **Re-usability of code**

- **Reliability**
- **It is the object repository of our project**

When do you get StaleElementReferenceException ?

An exception where in the address of an element is no longer fresh on the webpage due to a recent refresh of the webpage and we trying to perform an action on the same element using the same old address, we get this exception.

Pom class for Actitime Login page.

```
package pages;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

public class LoginPage {

        //Declaration

        @FindBy(id="username")

        private WebElement unTB;

        @FindBy(name="pwd")

        private WebElement pwTB;

        @FindBy(xpath="//div[.='Login ']")
```

```
        private WebElement loginBtn;

        //Initialisation

        public LoginPage(WebDriver driver){

                PageFactory.initElements(driver, this);

        }

        //Utilisation

        public void setUsername(String un){

                unTB.sendKeys(un);

        }

        public void setPassword(String pw){

                pwTB.sendKeys(pw);

        }

        public void clickLogin(){

                loginBtn.click();

}}
```

TESTNG Framework

Testng is a framework that we implement in our Selenium automation framework for following advantages.

- **Data Driven testing can be achieved (Data parameterisation is possible using testng)**
- **we can execute the test scripts in batch (i.e multiple test scripts at one go)**

- **we can also execute selected test scripts based on priority.**
- **we can execute the test case group wise or module wise**
- **generation of Automatic HTML reports**
- **We can integrate testng with Maven as well**
- **Due to certain annotations that testng provides, it has become so powerful**

What is Test Method and Test Class.

A method which is declared with test annotation(@Test) is called as Test method. A class with atleast 1 test method is called as Test class. We can have more than 1 test method in a Test class but in real time we use only 1 test method in a Test class.

If a test method fails , then it is considered that the Test class is failed which also means that the testscript got failed.

How to log in TestNG Report.

We use Reporter.log to print any customize message in html report. Reporter is a class from TestNG which has a static method log which accepts an argument of string type.

Reporter.log("customize message");

Reporter class has few static overloaded methods which are used to log report in different places. If we want to print the customize message in console and html report we use the following line of code.

Reporter.log("any message",true/false).

It accept two arguments, first is of String type and second is of boolean type. Default value is false. If it is true then it will print on eclipse console, if it is false it will print only in html report.

Write a program to check the sequence in which the annotations of Testng class gets executed.

Selenium code below :

```
package testngpackage;

import org.testng.annotations.BeforeMethod;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeClass;

import org.testng.Reporter;

import org.testng.annotations.AfterClass;

import org.testng.annotations.BeforeTest;

import org.testng.annotations.AfterTest;

import org.testng.annotations.BeforeSuite;

import org.testng.annotations.AfterSuite;

public class BaseTestNg {

 @BeforeMethod

  public void beforeMethod() {

  Reporter.log("beforeMethod", true);

  }

 @AfterMethod
```

```
public void afterMethod() {
Reporter.log("afterMethod", true);
}
@BeforeClass
public void beforeClass() {
Reporter.log("beforeClass", true);
}
@AfterClass
public void afterClass() {
        Reporter.log("afterClass", true);
}
@BeforeTest
public void beforeTest() {
        Reporter.log("beforeTest", true);
}
@AfterTest
public void afterTest() {
        Reporter.log("afterTest", true);
}
@BeforeSuite
```

```
 public void beforeSuite() {

        Reporter.log("beforeSuite", true);

}

@AfterSuite

 public void afterSuite() {

        Reporter.log("afterSuite", true);

}}
```

Demonstration on few parameters of @Test annotation as shown below.

How do you tag a test case to a particular module or a group ?

By using groups attribute of Test annotation. what do we pass here is the name of the module or a group to which we want a test method to tag to.

we can tag a particular test method to a group of modules as well by passing the list of modules in a string array.

Syntax :

```
@Test(groups={"module 1","module 2"})

public void createUser(){

}
```

How to execute test cases module wise or group wise ?

By using testng suite file.

By using the below syntax,

<groups>

<run>

<include name = "module name or group name">

<exclude name = "module name or group name">

</run>

</groups>

Summary :

which ever modules or groups are part of the include statement, those module specific or group specific test cases will be executed. If anything is specified with exclude statement, will not be executed.

This is how we execute test cases based on a module or a group.

@Test annotation attributes :

The default order of execution of multiple test methods is alphabetical order.

Priority :

It is an attribute of test annotation, using which, the order of execution

of test methods can be prioritized.

eg:

```
@Test(priority=1)
public void createUser(){
        System.out.println("createUser...");
}
```

How does priority works ?

The least the value, the highest the priority, and the highest the value, the least the priority.

Default value of priority is 0

Multiple test methods with same priority takes default order of execution (alphabetical order)

InvocationCount Attribute:

How do you execute a test method multiple times ?

By Using invocationCount attribute of Test annotation.

InvocationCount is used to invoke or run a test method for the specified number of times.

Default value of invocationCount is 1

syntax :

```
@Test(invocationCount=4)

public void editUser(){
System.out.println("editUser...");
}
```

Note:

If we specify invocationCount as 0, the test method will not be consider for execution (It will be skipped but will not be updated in the

report/console)

HOw do you skip a test method from execution without using invocationCount ?

By using Enabled attribute of Test annotation.

syntax :

```
@Test(invocationCount=4, enabled=false)
```

```
   public void editUser(){
        System.out.println("editUser...");
   }
```

Enabled=false will not execute the test method. It takes precedence over invocationCount.

Default value of Enabled is true.

How to set methods dependency on other methods?

Using dependsOnMethods attribute of Test annotation.

Syntax :

```
public class Demo {
   @Test(priority=1)
   public void createUser(){
        System.out.println("createUser...user created successfully");
```

```
        Assert.fail()}


   @Test(priority=3, dependsOnMethods="createUser")
   public void deleteUser(){
        System.out.println("deleteUser...");
   }
}
```

Here, createuser method will be failed and deleteUser method will be skipped. (In this case, skipped count will be updated on the report or console)

What is TestNg Exception - Cyclic Dependencies ?

When two test methods are dependent on each other, it results in to Testng

Exception.

Eg:

```
public class Demo {
```

```
    @Test(priority=1, dependsOnMethods="deleteUser")
    public void createUser(){
            System.out.println("createUser...user created successufuly");
    }

    @Test(priority=3, dependsOnMethods="createUser")
    public void deleteUser(){
            System.out.println("deleteUser...");
    }
}
```

@Test annotation attributes :

The default order of execution of multiple test methods is alphabetical order.

What is a Priority in Testng ?

It is an attribute of test annotation, using which, the order of execution of test methods can be prioritized.

eg:

```
@Test(priority=1)
public void createUser(){
        System.out.println("createUser...");
}
```

How does priority works ?

The least the value, the highest the priority, and the highest the value, the least the priority.

Default value of priority is 0

Multiple test methods with same priority takes default order of execution (alphabetical order)

What is InvocationCount Attribute ?

How do you execute a test method multiple times ?

By Using invocationCount attribute of Test annotation.

InvocationCount is used to invoke or run a test method for the specified number of times.

Default value of invocationCount is 1

syntax :

```
        @Test(invocationCount=4)
public void editUser(){
System.out.println("editUser...");
```

```
}
```

Note:

If we specify invocationCount as 0, the test method will not be consider for execution (It will be skipped but will not be updated in the report/console)

How do you skip a test method from execution without using invocationCount ?

By using Enabled attribute of Test annotation.

syntax :

```
@Test(invocationCount=4, enabled=false)
public void editUser(){
        System.out.println("editUser...");
}
```

Enabled=false will not execute the test method. It takes precedence over invocationCount.

Default value of Enabled is true.

Enabled attribute of test annotation will have highest priority out of all other attributes of test annotations.

How to set methods dependency on other methods?

Using dependsOnMethods attribute of Test annotation.

Syntax :

```
public class Demo {
```

```
   @Test(priority=1)
   public void createUser(){
         System.out.println("createUser...user created   successfully");
Assert.fail();
   }
   @Test(priority=3, dependsOnMethods="createUser")
   public void deleteUser(){
         System.out.println("deleteUser...");
   }
}
```

Here, createuser method will be failed and deleteUser method will be skipped. (In this case, skipped count will be updated on the report or console)

What is TestNg Exception - Cyclic Dependencies ?

When two test methods are dependent on each other, it results in to Testng Exception.

Eg:

```
public class Demo {
   @Test(priority=1, dependsOnMethods="deleteUser")
   public void createUser(){
         System.out.println("createUser...user created successfully");
```

```
    }

    @Test(priority=3, dependsOnMethods="createUser")

    public void deleteUser(){

            System.out.println("deleteUser...");

    }

}

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

package demotest;

import org.testng.Reporter;

import org.testng.annotations.Test;

public class DemoA {

        @Test(priority=1, groups={"user", "smoke"})

        public void CreateUser(){

                Reporter.log("CreateUser", true);

        }

        @Test(priority=2, invocationCount=1, enabled=true, groups={"user"})

        public void editUser(){

                Reporter.log("editUser", true);

        }

        @Test(priority=3, groups={"user"})
```

```
	public void deleteUser(){
		Reporter.log("deleteUser", true);
	}
	@Test(priority=1, groups={"product", "smoke"})
	public void createProduct(){
		Reporter.log("createProduct", true);
	}
	@Test(priority=2, invocationCount=1, enabled=true, groups={"product"})
	public void editProduct(){
		Reporter.log("editProduct", true);
	}
	@Test(priority=3, groups={"product"})
	public void deleteProduct(){
		Reporter.log("deleteProduct", true);
	}
}
```

Convert the above testng class to create testng.xml suite file as shown below

How to convert a testng class to testng.xml suite file ?

Right click on the testng class → TestNg ---> Convert to testng

Below xml file is created with the following data.

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

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name="Suite">

 <test name="Test">

 <groups>

 <run>

 <include name="user"></include>

 <exclude name="user"></exclude>

 </run>

  </groups>

  <classes>

      <class name="testngpackage.DemoA"/>

  </classes>

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

</suite> <!-- Suite →
```

Launch Multiple browser using Testng.xml suite file parameters

We can create multiple test blocks to work on multiple browsers in automation project.

Example is shown below.

```
package testngpackage;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.IOException;

import java.util.Properties;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.testng.Reporter;

import org.testng.annotations.Parameters;

import org.testng.annotations.Test;

public class LaunchFirefoxAndChromeTogether {

        static{

        System.setProperty("webdriver.gecko.driver", "./driver/geckodriver.exe");

        System.setProperty("webdriver.chrome.driver", "./driver/chromedriver.exe");

        }
```

```
	WebDriver driver;
	@Test
	@Parameters({"browser"})
	public void loginFFandCHROME(String browser) throws InterruptedException,
IOException{
		//Reporter.log(browser, true);
		if (browser.equals("firefox")) {
			driver = new FirefoxDriver();
		} else {
			driver = new ChromeDriver();
		}
		FileInputStream configPath = new FileInputStream(".\\config.properties");
		Properties prop = new Properties();
		prop.load(configPath);
		String url = prop.getProperty("URL");
		driver.get(url);
		WebElement un = driver.findElement(By.id("username"));
		for (int i = 0; i < 10; i++) {
			un.sendKeys("admin" + i);
			Thread.sleep(2000);
```

```
                un.clear();
            }
            driver.close();
        }
}
```

Use the below testng.xml suite file

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Suite" parallel="tests">
 <test name="TestFirefox">
 <parameter name="browser" value="firefox"></parameter>
  <classes>
   <class name="testngpackage.LaunchFirefoxAndChromeTogether"/>
  </classes>
 </test> <!-- Test -->
 <test name="TestChrome">
 <parameter name="browser" value="chrome"></parameter>
  <classes>
   <class name="testngpackage.LaunchFirefoxAndChromeTogether"/>
```

</classes>

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

</suite> <!-- Suite ⟶

Parameterisation using @DataProvider

How do you achieve data parameterisation using DataProvider annotation ?

Ans:

1. Using DataProvider annotation, we can create our own set of data.

2. In order to create data, we create an object of 2 dimensional array wherein we specify the row and column.

Here, row represents the number of times the test method should iterate and column represents the number of parameters that we should pass as an argument to the test method.

3. These databank can be utilised across any testng classes by using an attribute of Test annotation called dataProvider. Once we have the access to data bank, we can use the data in our script.

This is how we achieve data parameterisation using DataProvider annotation.

Note: A test method can't access data from multiple data banks at the same time.

code :

```
package scripts;

import org.testng.Reporter;

import org.testng.annotations.DataProvider;
```

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

public class DataProviderExample{

   @DataProvider

   public Object[][] dataBank(){

         Object[][] data = new Object[2][2];

         data[0][0] = "admin1";

         data[0][1] = "manager1";

         data[1][0] = "admin2";

         data[1][1] = "manager2";

         return data;

   }

   @Test(dataProvider="dataBank")

   public void useDataBank(String un, String pwd){

         Reporter.log(un + " --> "+ pwd, true);

   }

   }
```

Why we don't use DataProvider in real time ?

1. **Data creation using DataProvider annotation is time consuming.**
2. **Modification is a tough job and hence maintenance is very high**
3. **We can't access data from multiple data banks.**

Data Parameterization using TestNG Suite File:

Executing the test scripts with multiple set of data by taking the data from external source is known as data parameterization.

We have multiple approaches to achieve this.

Data Parameterization by using testng suite file:

- In testing suite file, what we do is, we declare and initialize the parameters using parameter tag.

-Once the parameters are declared and initialized, we utilize these parameters from any testNG class by using @Parameters annotation.

-As an argument to this Parameters annotation, what we pass is the parameter name which is declared in the suite file.

-- We can access multiple parameters as well by using String array.

-And then by creating local variables to the test method, we access these parameters value and utilize them in our scripts.

This is how ,we achieve data parameterization using testng suite file.

What are testng listeners ?

In testng framework, listeners keep on listening to the execution status of suite, test blocks, and testng classes. And based on a predefined event, it performs specific actions.

for eg :

In our project, if we need to connect to databases and perform some database validations, then before starting the suite execution, we can connect to database and once the entire suite is executed, we can flush out the unwanted data from the database.

code :

```
public class TestngListeners implements ISuiteListener {

   @Override

   public void onStart(ISuite suite) {

         //code to connect to database

         Reporter.log("Suite started here.."+ suite.getName(), true);

   }

   @Override

   public void onFinish(ISuite suite) {

         //code to flush out the unwanted data from database

         Reporter.log("Suite finished here.."+ suite.getName(),true);

   }
```

Data parameterisation from Excel file:

· Data parameterisation from Excel file can be done by using Apache poi related jar file which has a class called WorkbookFactory.

· We call static method of this WorkbookFactory called create( ) wherein we pass reference of FileInputStream class as an argument ,this returns an instance of Workbook interface.

· Using this Workbook reference we call getSheet( ) method , where in we pass sheet no. as an argument which specifies from which sheet we are accessing the data.

· Then to get to the particular row we use getRow( ) method and to get to the particular column we use getCell( ) method and

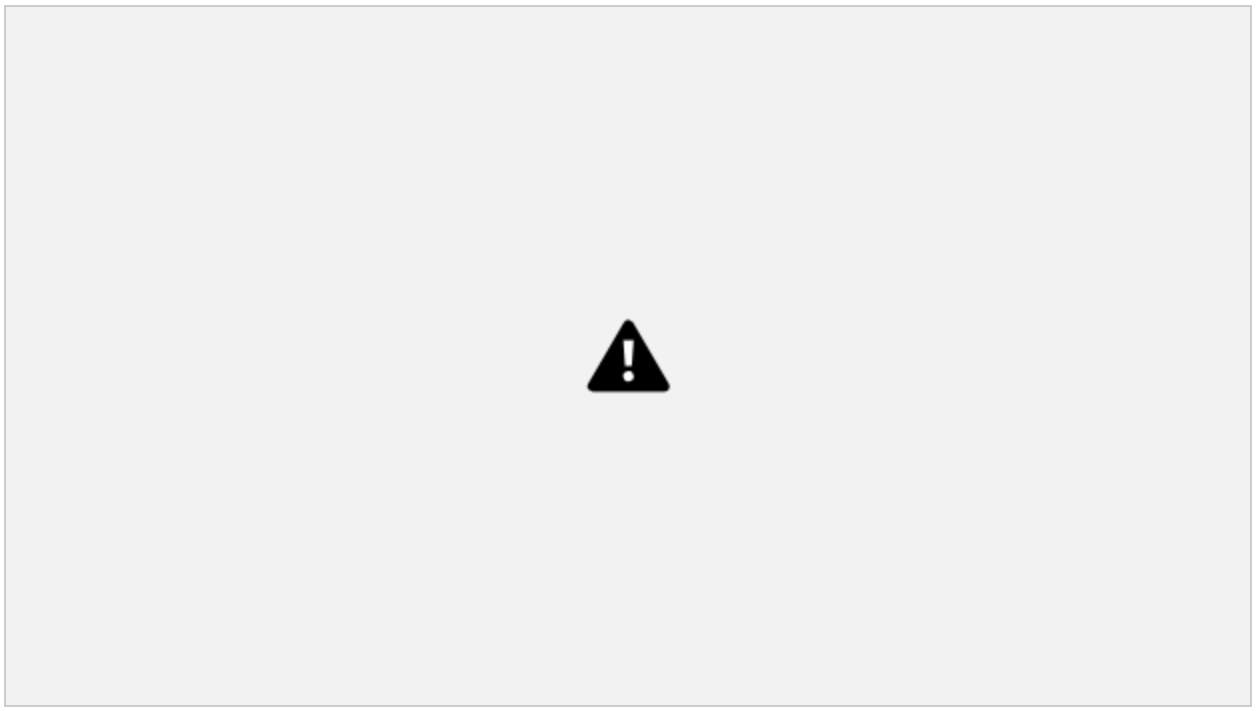
· To get actual content of particular cell we use toString( ) method.

This is how we access data from Excel sheet

How to install Apache POI ?

GO to below url:

https://poi.apache.org/download.html#POI-4.0.0



Data parameterisation from property file:

1) A file with . properties extension is called property file where data is stored in the form of key-value pair.

2) Data parameterisation from property file can be done by using Properties class which has a non static method called load( ).

3)We use load( ) method to a load property file which takes reference of FileInputStream class as an argument and

4) To access data from property file we use get property( ) method which takes property name(key) as an argument.

Q. How do you handle validation ? OR

Q. how do you handle failed test cases?

Ans. is Testng Assertion , to validate the results we use assertions.

Assertion are of two types

1. **Assert/ Hard Assert : It is the normal assert which is used to do validations in the Testng class. If Assert is fails, than none of the code gets executed after the assert statement. We use Assert class , in this we have some static methods like fail(), assertEquals(actual value, expected value), this assertEquals() method takes 2 arguments both can be String type, int type or Object type. [so we can directly call these methods by using class name Assert.fail().}**
2. **Soft Assert : If we want to continue the test execution even after the assert statements fails than we have to use soft assert. To create a soft assert, we have to create an object of SoftAssert Class because all methods of this class are non static. some methods are assertEquals(... ,...) ,assertAll().**

   SoftAssert assert= new SoftAssert();

   assert.assertEquals(10,20);

assert.assertEquals(“driver.getTitle”, “Expected value ”);

assert. assertAll();// this is the mandatory method of soft assert class, it should be return in last or last executable statement otherwise it will give Compilation error(unreachable code).

SELENIUM GRID

To run the same scripts on multiple browsers and multiple systems parallely, we use Selenium Grid.

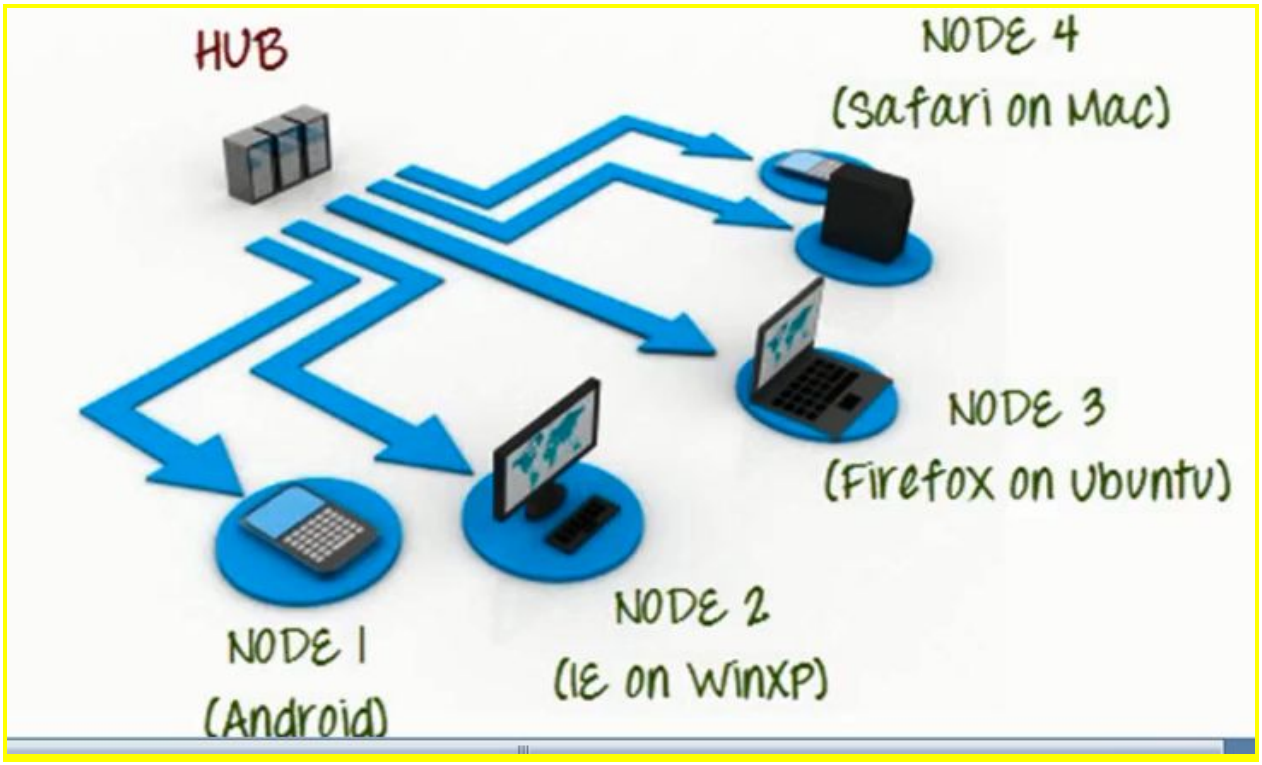
Here, there will be 2 types of system.

1. **HUB**
2. **NODE.**

Node is the remote system on which you run the automation scripts.

In node system, JDK and Browser should be installed and we should also know the ip address.

It Is used for Cross browser compatibility testing and cross platform testing on multiple Operating systems.



Steps to setup NODE system :

1. **Download selenium server jar file and browser specific driver executables files such as chromedriver.exe and geckodriver.exe files in to a folder.**
2. **In the same folder, create a batch file with .bat extension and write the following command.**

   java -Dwebdriver.gecko.driver=geckodriver.exe -Dwebdriver.chrome.driver=chromedriver.exe -jar selenium-server-standalone-3.141.59.jar

   3.Double click on the Run.bat file and it should display the following message in the command prompt window.

   Selenium server is up and running.

HUB :

It is centralised system where the script is present. It is also used to control the execution.

We run the scripts from HUB and it will connect to remote system called NODE.

It will open the browser and perform action in the node and the result will be stored in the HUB machine.

Steps to set up HUB:

1. **Hub will have all the softwares which is required for a typical selenium machine.**
2. **Update the selenium code to execute the scripts in remote system as shown below.**

To work on selenium grid, we have to create an object of RemoteWebDriver class which accepts 2 arguments, both are object type. First argument is an object of URL class and second argument is an object of DesiredCapabilities class.

```
package demotest;

import java.net.MalformedURLException;

import java.net.URL;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.remote.DesiredCapabilities;

import org.openqa.selenium.remote.RemoteWebDriver;

import org.testng.annotations.Parameters;

import org.testng.annotations.Test;

public class SeleniumGridDemo {
```

```
	@Test
	@Parameters({"node","browser"})
	public void LaunchFireFoxAndChrome(String node, String browser) throws
MalformedURLException{
		URL whichSystem = new URL(node);
		DesiredCapabilities whichbrowser = new DesiredCapabilities();
		whichbrowser.setBrowserName(browser);
		WebDriver driver = new RemoteWebDriver(whichSystem, whichbrowser);
	}}
```

Update the testng.xml suite file to run in multiple browsers and multiple systems.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">
<suite name="Suite" parallel="tests">
 <test name="TestFirefox">
 <parameter name="node" value="http://localhost:4444/wd/hub"></parameter>
 <parameter name="browser" value="firefox"></parameter>
   <classes>
    <class name="testngpackage.SeleniumGridExample"/>
```

```
  </classes>
 </test> <!-- Test -->
 <test name="TestChrome">
 <parameter name="node" value="http://localhost:4444/wd/hub"></parameter>
 <parameter name="browser" value="chrome"></parameter>
  <classes>
   <class name="testngpackage.SeleniumGridExample"/>
  </classes>
 </test> <!-- Test -->
</suite> <!-- Suite -->
```

SELENIUM FRAMEWORK

Framework is set of instructions or guidelines which should be followed by all the automation engineers in the team while automating an web application.

There are 3 major status as mentioned below.

1. **Automation Framework Design**
2. **Automation Framework Implementation**
3. **Automation Framework Execution**

Questions : Which framework have you developed/implemented in your project ?

We have implemented Hybrid driven framework, which is a combination of POM driven framework, testng framework, data driven framework, method driven framework and modular driven framework.

POM Driven Framework :

1. **In POM driven framework, we have created POM classes for all the page of our application under test.**
2. **In our application, we had 20 pages in total and for all these 20 pages, we have created 20 pom classes. All these pom classes, we have created in a single package called pompages.**
3. **In each POM class, we declared the elements present on that particular page using @FindBy annotation. As an argument to FindBy annotation, we can use any one of the locator using which, element can be uniquely identified on the web page.**
4. **Once elements are declared, we initialise all the elements declared above using PageFactory.initElements() method, which accepts 2 arguments - both are of type object. First argument is WebDriver driver object and second argument is the current class object which is referred by this keyword , we write this statement inside the constructor, so that when the object of this pom class is created from another class, it will invoke the constructor and initialise all the elements which are declared in the pom class.**
5. **Once elements are declared and initialized, we utilise all the elements by creating respective setter methods. This is what we have done on a high level inside a pom class.**

TESTNG FRAMEWORK :

1. **Based on the number of test cases, we will create that many number of Testng classes. In our project, we had close to 678 regression test cases and we have developed 678 testng class with one test method in each class.**

2. **In test method, we create object of respective POM class and using this reference variable, we keep calling the relevant method of pom class based on the manual test steps. This is how, we have automated our scripts using testng framework.**

Data Driven Framework :

1. **Executing the same scripts with multiple set of data is called data parameterisation. We used Excel file to get data from external source and utilised it in the scripts.**
2. **Using apache poi related jar, we implemented this data driven technique to achieve data parameterisation in our framework. Hence, our framework is also a data driven framework.**

Modular Driven Framework :

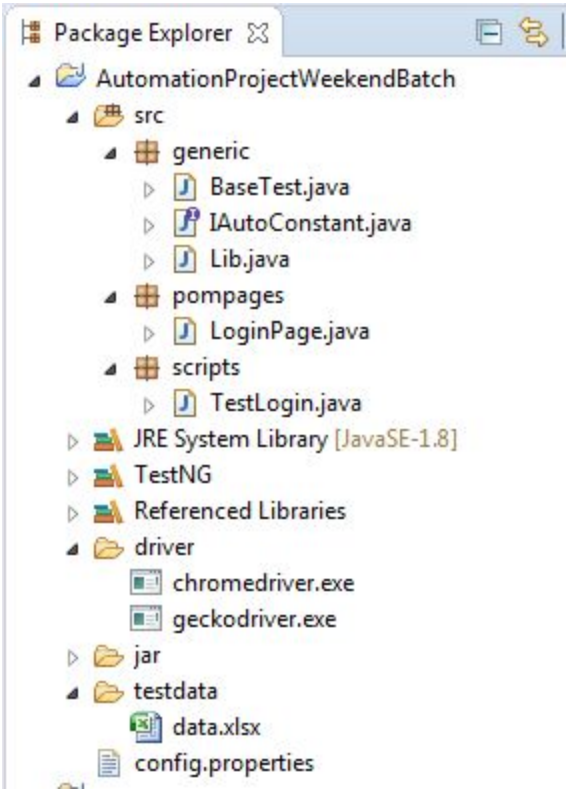
1. **Module wise execution of test scripts is known as modular driven framework.**
2. **In our project, when ever we develop a test method while automating a test case, we tag it to some group based on the module name.**
3. **Now, if any test script fails during normal execution cycle, we log defects and once developer fix the issue, we ensure that all the related test scripts of this particular module are executed and passed. This process of execution of module wise test scripts is known as modular driven framework.**

Method Driven Framework :

1. **We created few generic methods to access data from external sources like Excel file, config file, etc.**
2. **And furthermore, based on the manual regression test case steps, we call the relevant method of pom class from the testng class, In this way , our framework is a method driven framework as well.**

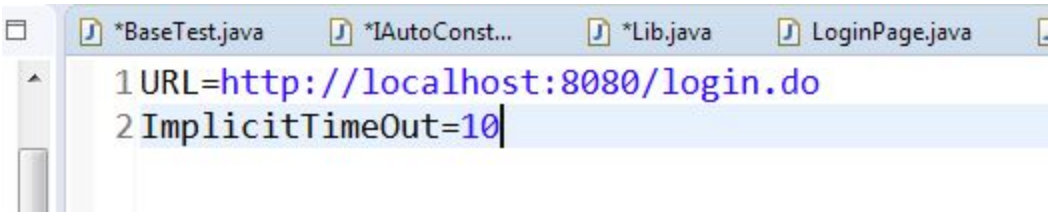
In this way, the framework that we have implemented is a HYBRID framework.

Project Framework Folder Structure is mentioned below.

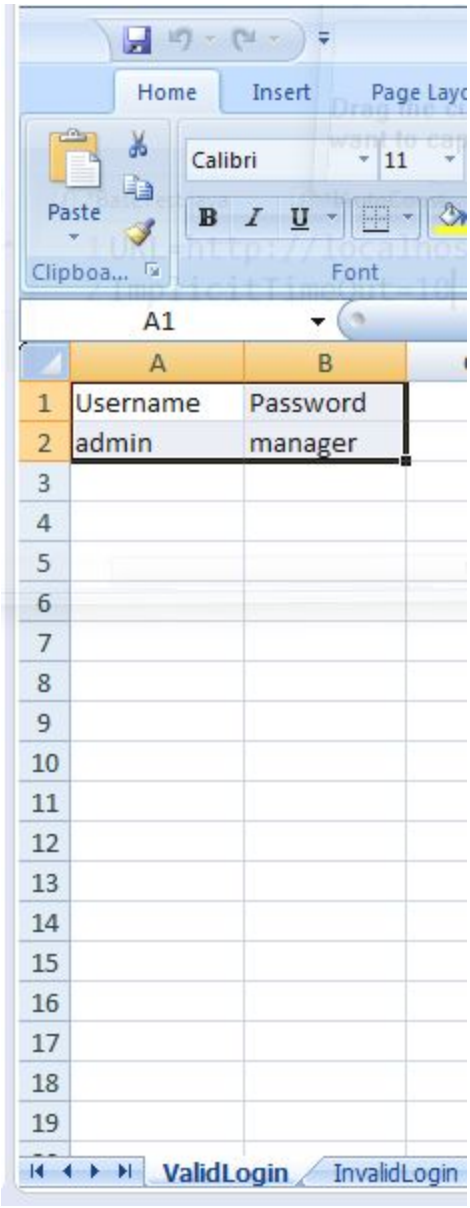


_____

Config.Properties snapshot below :



data.xlsx snapshot below:



BaseTest.java Code

---

package generic;

import java.util.concurrent.TimeUnit;

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.firefox.FirefoxDriver;
import org.testng.annotations.AfterMethod;
import org.testng.annotations.BeforeMethod;
public class BaseTest implements IAutoConstant{
	public static WebDriver driver;
	static{
		System.setProperty(GECKO_KEY, GECKO_VALUE);
		System.setProperty(CHROME_KEY, CHROME_VALUE);
	}
	@BeforeMethod
	public void openApplication(){
		 driver = new FirefoxDriver();
		 String url = Lib.getProperty(CONFIG_PATH, "URL");
		driver.get(url);
		String ITO = Lib.getProperty(CONFIG_PATH, "ImplicitTimeOut");
		int timeoutPeriod = Integer.parseInt(ITO);
		driver.manage().timeouts().implicitlyWait(timeoutPeriod, TimeUnit.SECONDS);
	}
	@AfterMethod
```

```
	public void closeApplication(){
		driver.close();
	}
}
```

IAutoConstant Interface code below

```
package generic;
public interface IAutoConstant {
	String CONFIG_PATH = ".\\config.properties";
	String EXCEL_PATH = ".\\testdata\\data.xlsx";
	String GECKO_KEY = "webdriver.gecko.driver";
	String GECKO_VALUE = ".\\driver\\geckodriver.exe";
	String CHROME_KEY = "webdriver.chrome.driver";
	String CHROME_VALUE = ".\\driver\\chromedriver.exe";
}
```

Lib.java class file to create all project related generic functions.

```
package generic;
import java.io.FileInputStream;
import java.util.Properties;
```

```
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.usermodel.WorkbookFactory;
public class Lib implements IAutoConstant{
	public static Workbook wb;
	public static String getProperty(String CONFIG_PATH, String key){
		String property = "";
		Properties prop = new Properties();
		try {
			prop.load(new FileInputStream(CONFIG_PATH));
			property = prop.getProperty(key);
		} catch (Exception e) {
		}
		return property;
	}
	public static int getRowCount(String EXCEL_PATH, String sheet){
		int rowCount = 0;
		try {
			wb = WorkbookFactory.create(new FileInputStream(EXCEL_PATH));
			rowCount = wb.getSheet(sheet).getLastRowNum();
		} catch (Exception e) {
```

```
            }

            return rowCount;

    }

    public static String getCellValue(String EXCEL_PATH, String sheet, int row, int column){

            String value = "";

            try {

                    wb = WorkbookFactory.create(new FileInputStream(EXCEL_PATH));

                    value = wb.getSheet(sheet).getRow(row).getCell(column).toString();

            } catch (Exception e) {

            }

            return value;

    }
```

---

pompackage - LoginPage.java

```
package pompages;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.support.FindBy;

import org.openqa.selenium.support.PageFactory;

public class LoginPage {
```

```
//declaration
@FindBy(id="username")
private WebElement unTB;
@FindBy(name="pwd")
private WebElement pwTB;
@FindBy(xpath="//div[.='Login ']")
private WebElement loginBtn;
//initialisation
public LoginPage(WebDriver driver){
        PageFactory.initElements(driver, this);
}
//Utilisation
public void setUsername(String un){
        unTB.sendKeys(un);
}
public void setPassword(String pw){
        pwTB.sendKeys(pw);
}
public void clickLogin(){
        loginBtn.click();
```

```
	}
}
```

TestNg Class - TestLogin

```
package scripts;
import org.testng.annotations.Test;
import generic.BaseTest;
import generic.Lib;
import pompages.LoginPage;
public class TestLogin extends BaseTest{
	@Test
	public void testLogin(){
		LoginPage l = new LoginPage(driver);
		String un = Lib.getCellValue(EXCEL_PATH, "ValidLogin", 1, 0);
		String pw = Lib.getCellValue(EXCEL_PATH, "ValidLogin", 1, 1);
		l.setUsername(un);
		l.setPassword(pw);
		l.clickLogin();
	}}
```

Take Screenshots when a test method is failed

In BaseTest.java file, write below code

```
package generic;

import java.io.File;

import java.io.IOException;

import java.util.Date;

import java.util.concurrent.TimeUnit;

import org.apache.commons.io.FileUtils;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.firefox.FirefoxDriver;

import org.testng.annotations.AfterMethod;

import org.testng.annotations.BeforeMethod;

public class BaseTest implements IAutoConstant{

	public static WebDriver driver;

	static{

		System.setProperty(GECKO_KEY, GECKO_VALUE);

		System.setProperty(CHROME_KEY, CHROME_VALUE);

	}
```

```
	@BeforeMethod
	public void openApplication(){
		driver = new FirefoxDriver();
		String url = Lib.getProperty(CONFIG_PATH, "URL");
		driver.get(url);
		String ITO = Lib.getProperty(CONFIG_PATH, "ImplicitTimeOut");
		int timeoutPeriod = Integer.parseInt(ITO);
		driver.manage().timeouts().implicitlyWait(timeoutPeriod, TimeUnit.SECONDS);
	}
	@AfterMethod
	public void closeApplication(){
		driver.close();
	}
	public void takeScreenshot(String testname){
		Date d = new Date();
		String currentdate = d.toString().replaceAll(":", "_");
		TakesScreenshot ts = (TakesScreenshot) driver;
		File srcFile = ts.getScreenshotAs(OutputType.FILE);
		File destFile = new
File(".\\screenshots\\"+currentdate+"\\"+testname+"_screenshot.png");
```

```
            try {
                    FileUtils.copyFile(srcFile, destFile);
            } catch (IOException e) {
                    e.printStackTrace();
            }
    }
    }
```

Create a class called TestListener.java

```
package generic;
import org.testng.ITestContext;
import org.testng.ITestListener;
import org.testng.ITestResult;
public class TestngListeners implements ITestListener {
    BaseTest b = new BaseTest();
    @Override
    public void onTestStart(ITestResult result) {
        // TODO Auto-generated method stub
    }
```

```
@Override
public void onTestSuccess(ITestResult result) {
	// TODO Auto-generated method stub
}
@Override
public void onTestFailure(ITestResult result) {
	String testmethodName = result.getName();
	b.takeScreenshot("TestValidLogin");
}
@Override
public void onTestSkipped(ITestResult result) {
	// TODO Auto-generated method stub
}
@Override
public void onTestFailedButWithinSuccessPercentage(ITestResult result) {
	// TODO Auto-generated method stub
}
@Override
public void onStart(ITestContext context) {
	// TODO Auto-generated method stub
```

```
	}

	@Override

	public void onFinish(ITestContext context) {

		// TODO Auto-generated method stub

	}

}
```

Create testng.xml suite file as shown below.

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

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd">

<suite name="Suite">

<listeners>

<listener class-name="generic.TestngListeners"></listener>

</listeners>

  <test name="Test">

    <classes>

      <class name="scripts.TestLogin"/>

    </classes>

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

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

Update the TestLogin.java class as shown below

```
package scripts;

import org.testng.annotations.Test;

import org.testng.asserts.SoftAssert;


import generic.BaseTest;

import generic.Lib;

import pompages.LoginPage;

public class TestLogin extends BaseTest{

	@Test

	public void testLogin(){

		LoginPage l = new LoginPage(driver);

		String un = Lib.getCellValue(EXCEL_PATH, "ValidLogin", 1, 0);

		String pw = Lib.getCellValue(EXCEL_PATH, "ValidLogin", 1, 1);

		String expectedTitle = Lib.getCellValue(EXCEL_PATH, "ValidLogin", 1, 2);

		l.setUsername(un);

		l.setPassword(pw);

		l.clickLogin();

		String actualtitle = driver.getTitle();
```

```
        SoftAssert s = new SoftAssert();

        s.assertEquals(actualtitle, expectedTitle);

        s.assertAll();

    }

}
```

Run the suite.xml file

SYNCHRONIZATION ISSUE

The selenium scripts works much faster than the application loading speed and hence we find lot of synchronization issue while executing our script. To avoid such synchronisation issue in our test execution, we use some kind of wait to delay the selenium script execution. There are two ways to handle the synchronisation issue.

1. Static wait 2. Dynamic wait.

Static Wait- In static wait we actually delay the execution by using Thread.sleep(), as an argument to sleep method we pass some duration in seconds, milliseconds to stop the execution. If application is loading in 2 secs and we are using a static wait of 4 or 5 secs , we are wasting 3 secs of time in each and every instant which is not a good choice so we always prefer dynamic wait of handling synchronisation issue.

Dynamic wait is categorised into two types

1. Implicit wait 2. Explicit wait

IMPLICIT WAIT -

It is the maximum time period wherein findElement() and FindElements() methods will wait before it actually throws an exception when in the process of identifying elements on the page.

Line of code to set the implicit wait time period for 10 seconds is below:

driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

Note :

→ If findElement() method is able to identify objects on the webpage within the specified implicit time period, it returns the address of the first matching element.

→ And, if findElement() method fails to identify object on the webpage with in the specified implicit wait period, then findElement method throw NoSuchElementException, where as

findElements method doesn't throw any exception, instead, it returns an empty object.

Polling Period/ Pooling Period

It is a time interval in which findElement() and FindElements() methods will check at an interval of every 0.5 sec (default value) whether the element is loaded on the webpage or not.

If the element is loaded, it will return the address and if it is not found, then it will wait for another 0.5 sec and this process continues till it reaches the maximum implicit time period.

If still the element is not found, then it throws NoSuchElementException or empty list object.

We declare implicit wait only once throughout the project. It works only with findElement() and findElements() methods.

EXPLICIT WAIT

→ It is the maximum time period where in WebDriver will wait for a condition to be satisfied on the webpage.

Explicit wait is also known as WebDriver wait. Here we are directing the webdriver to wait explicitly for a specific amount of time when a transition is happening between the pages of the application. (example → i.e from Login page to Homepage). During this transient period we are not performing any action until the page is completely loaded. Like this we have a scenario when we have to wait for an alert to be pop up on the page. Based on the requirement we use explicit wait in our script execution.

We create an object of WebDriverWait class and as an argument to the constructor we pass driver reference and the timeunit and then we pass some expected conditions to be satisfied, until the condition is satisfied, it will wait for the specified explicit time period before throwing Timeoutexception.

```
WebDriverWait  wait = new   WebDriverWait(driver, 10);

wait.until(ExpectedConditions.titleIs("Actual title of the page"));
```

alertIsPresent(), elementToBeClickable(), elementToBeLocated() etc.

Explicit wait is used based on a condition, we can use whenever we want any condition to be satisfied.. It works with any kind of elements on a page.

TIMEOUT EXCEPTION

WebDriver throws this exception if the expected condition is not satisfied within the specified explicit wait time.

Explicit wait- polling period

It is the time interval of every 500 milliseconds, where in webdriver checks for the condition to be satisfied, if the condition is satisfied, it continue with further execution, and if the condition is not satisfied, it waits for another 500 milliseconds and then it checks again for the condition to be satisfied, this process continues till the maximum explicit time period is reached. and if still, the condition is not satisfied, webdriver will throw an exception called TimeOut Exception.

Interview question

Find out the differences between Implicit and Explicit wait.

MAVEN PROJECT :

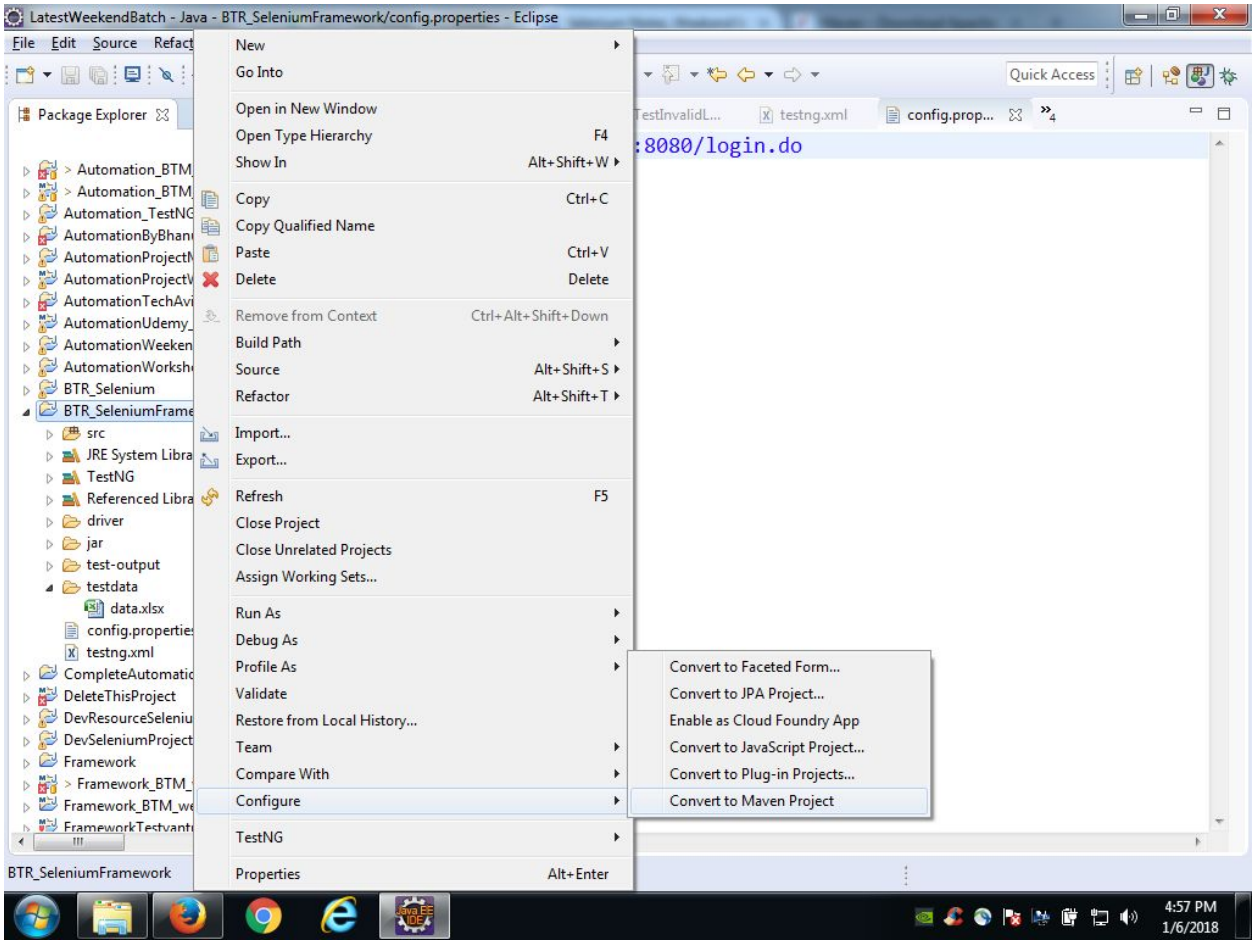
Apache Maven is a project management tool OR build dependency tool OR Build configuration tool.

Why Maven ?

- Central repository to get dependencies
- maintaining common structure across the organization
- Flexibility in integrating with CI tools like JENKINS
- Plugins for Test framework execution

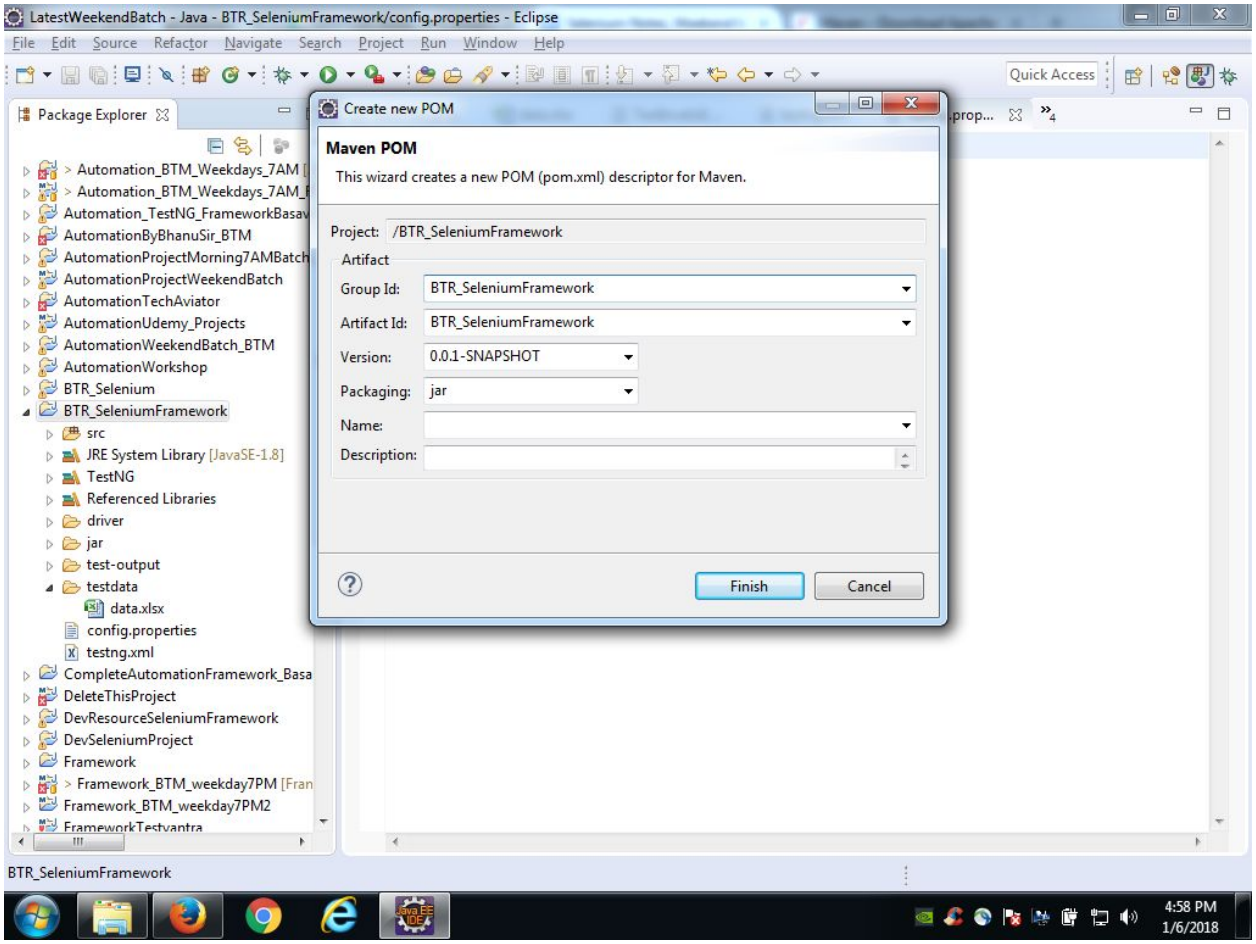
Convert your java project in to maven project like this

Right click on the project -- configure -- convert to maven project

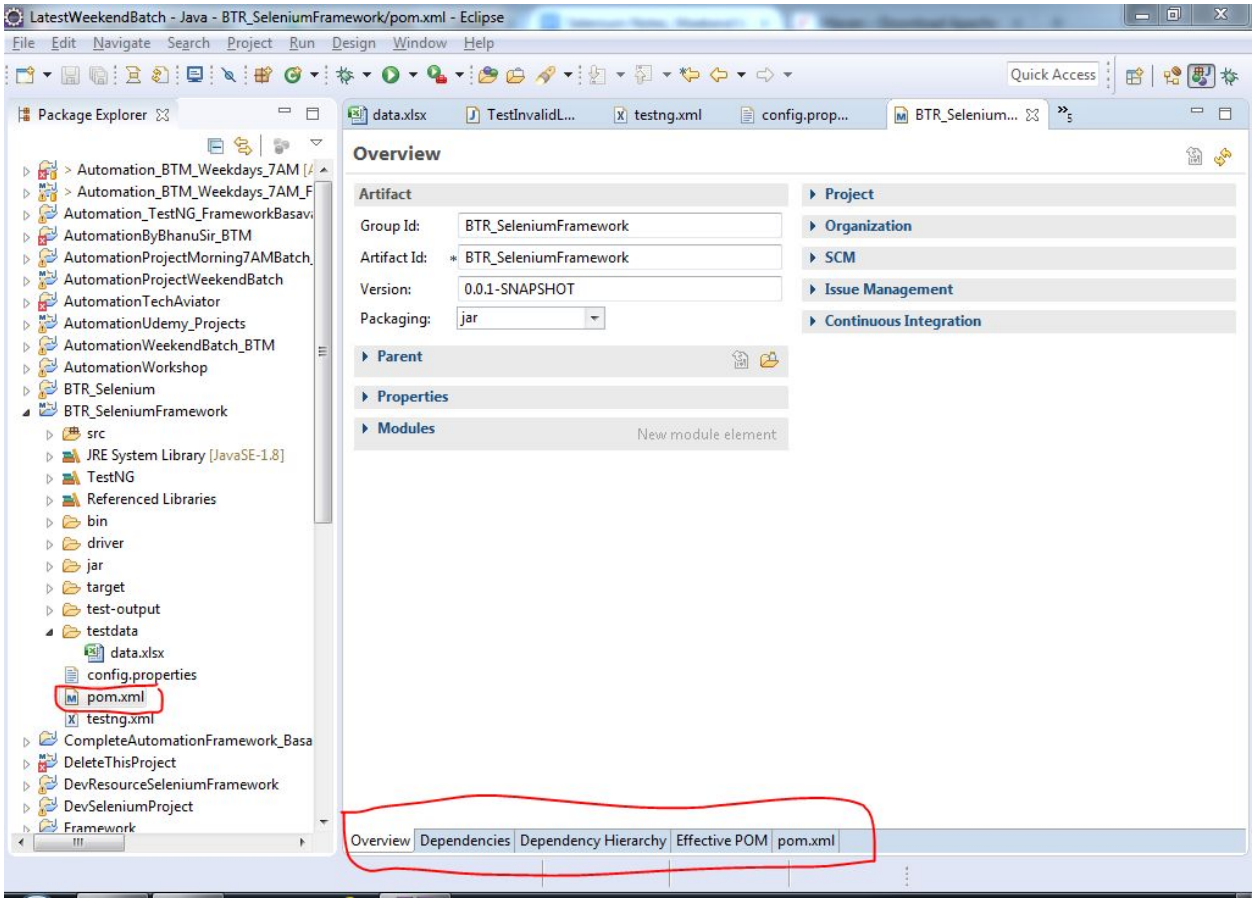


leave it as it is..

no change is required in the below snapshot. Just click on Finish button.



It creates a pom.xml file as shown below.

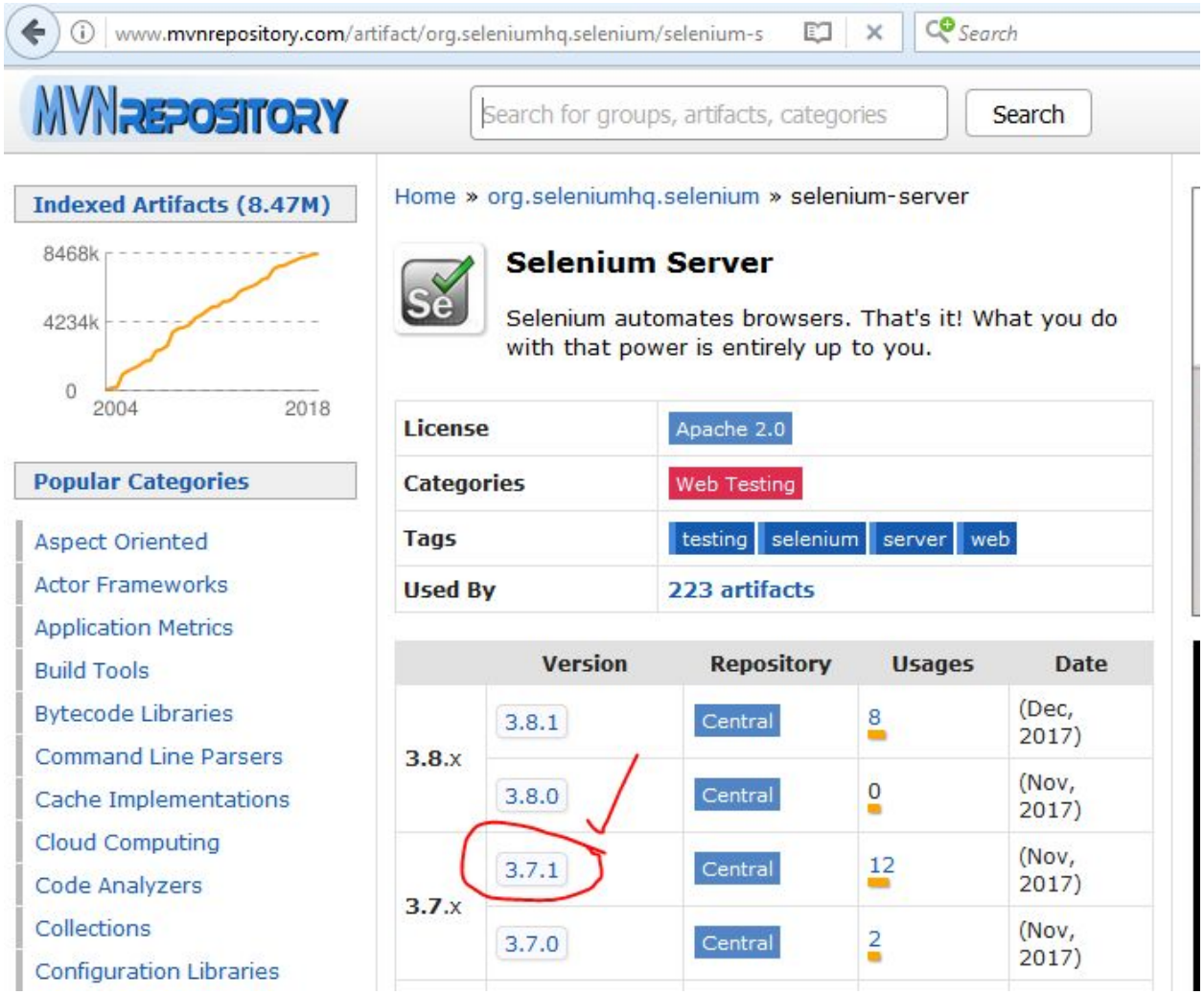


Go to 2nd tab which is Dependencies tab and add dependency related information from below url

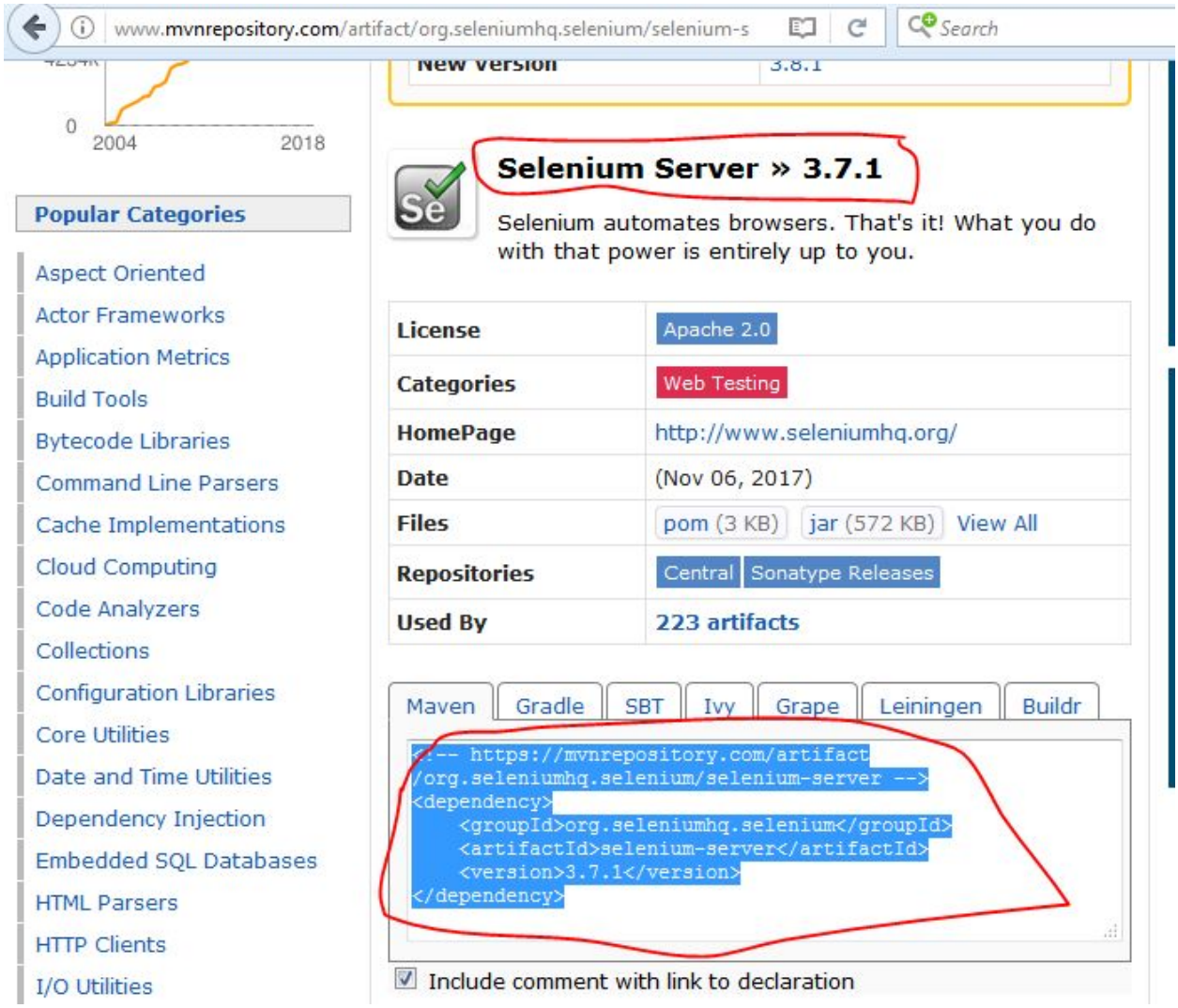
http://www.mvnrepository.com

Search selenium server as shown below

CLick on 3.7.1 link as shown below

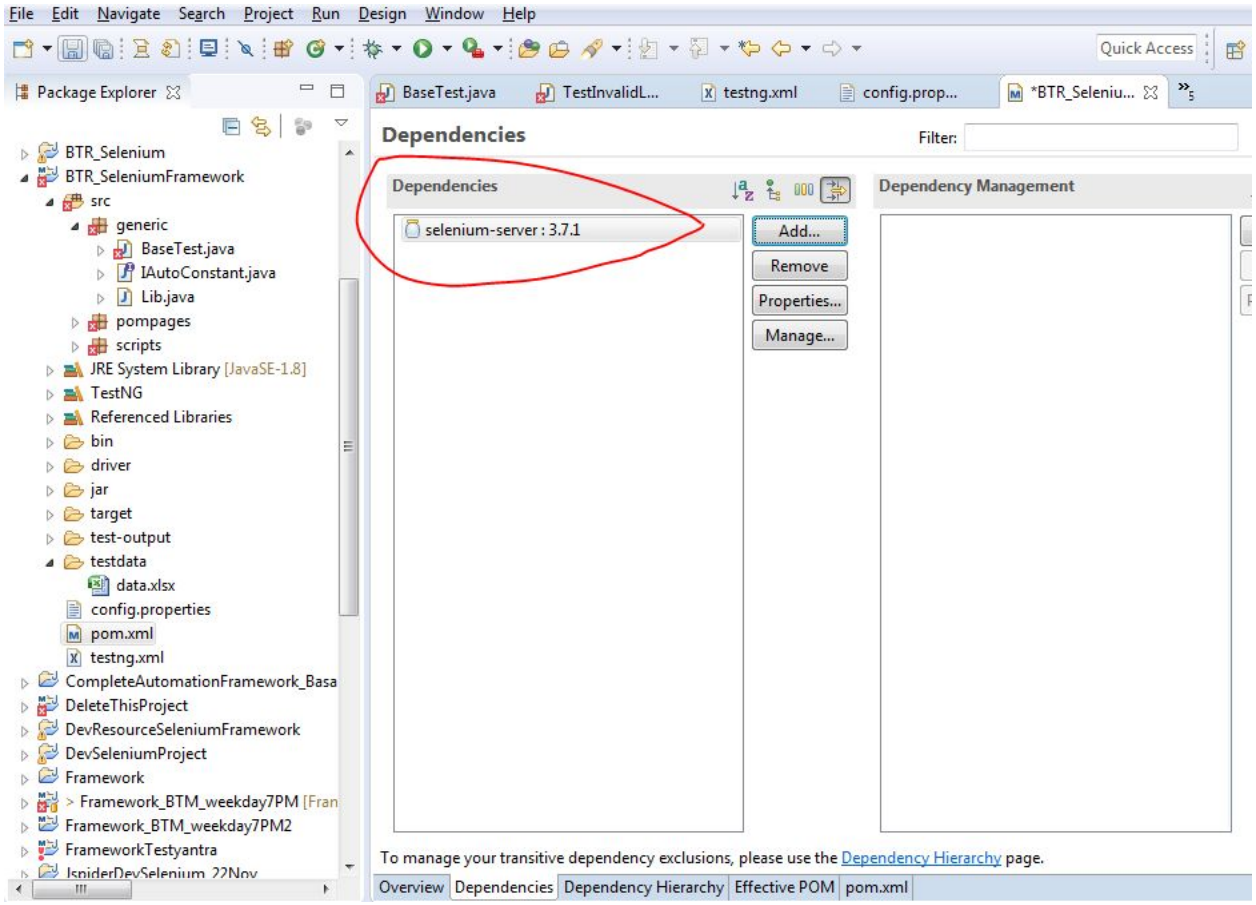


copy the below dependency information and add it in the pom.xml file



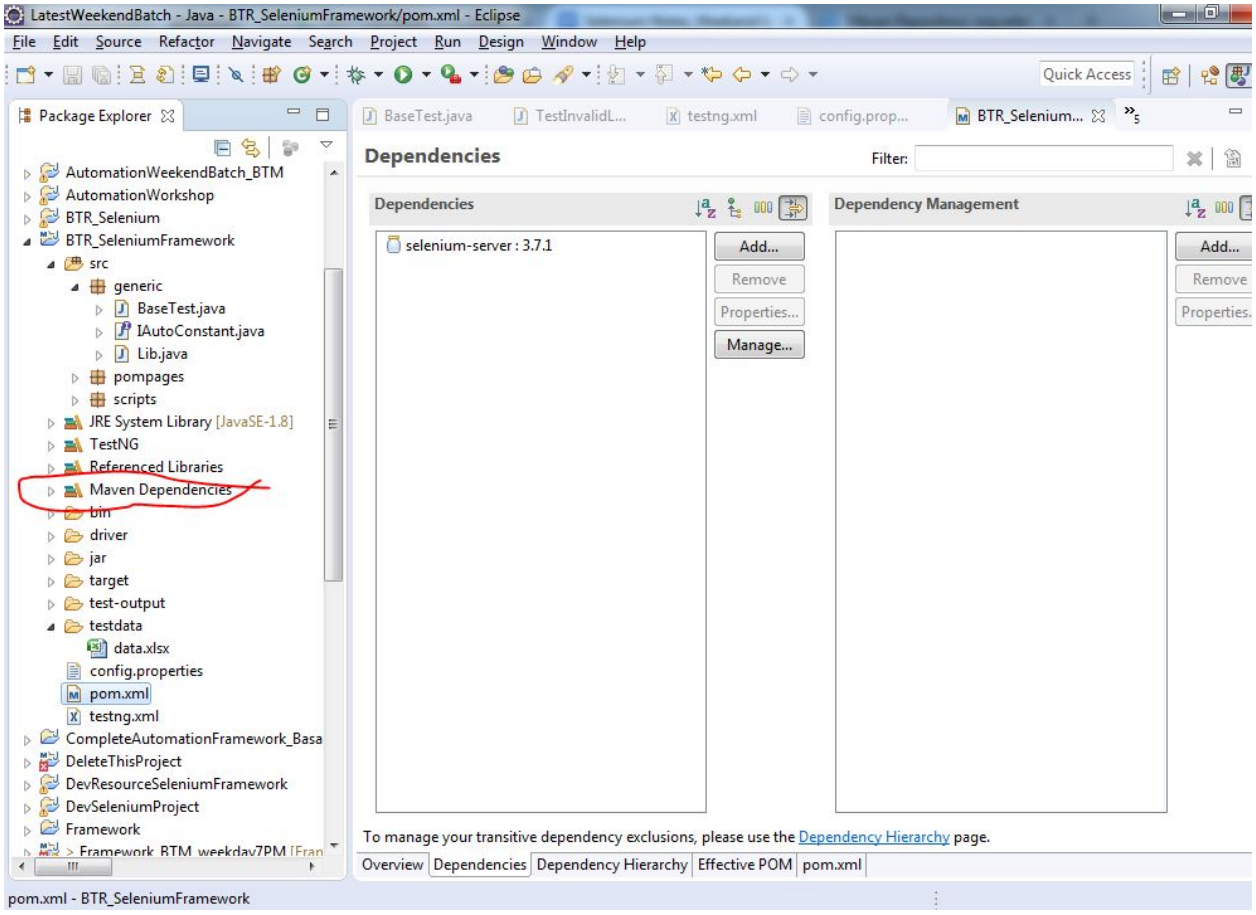
how do you add to pom.xml file ?

click on OK button and you see something like this.

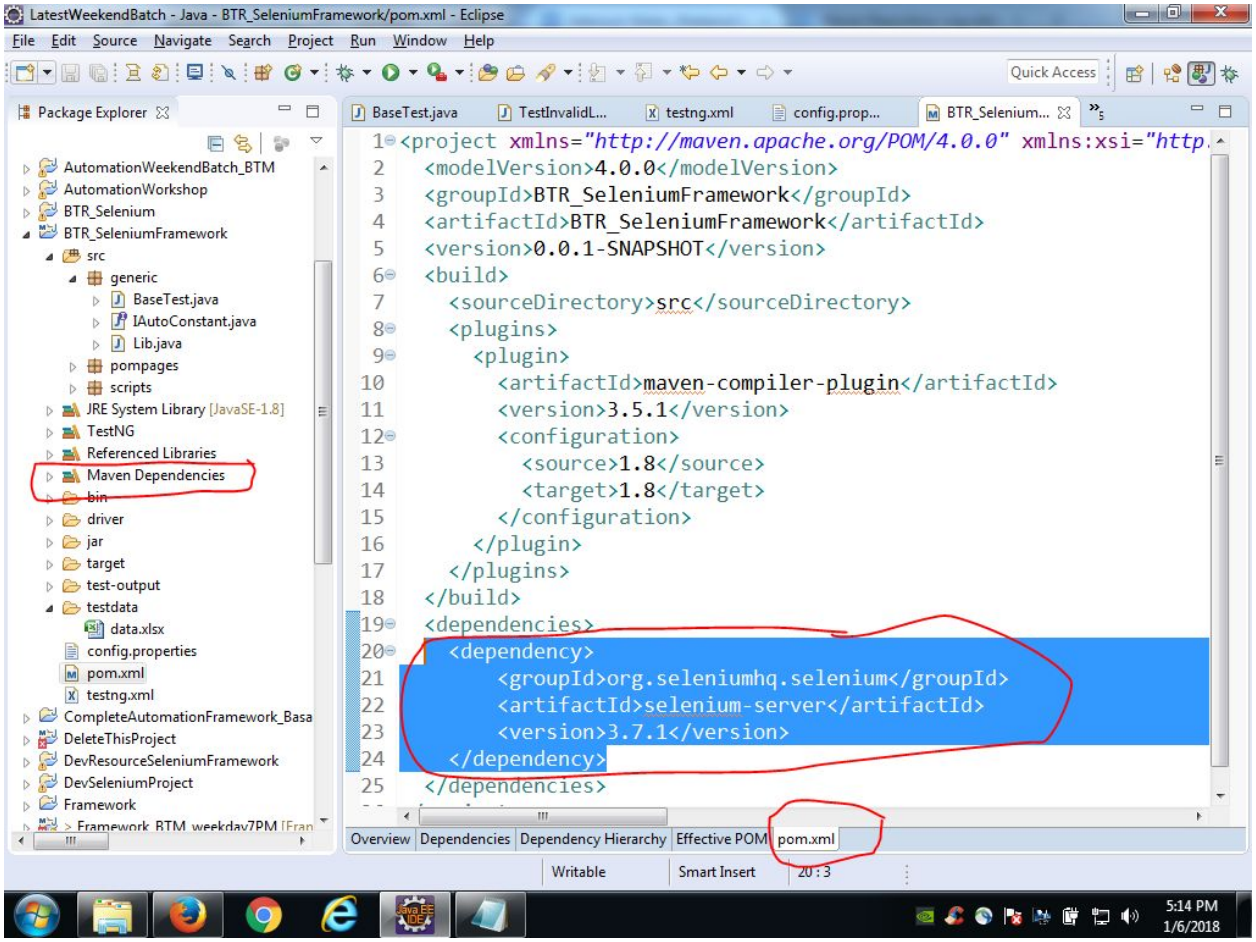


now save the project -- control + S - in order to build the dependency jar files to the project

As a result, Maven dependency file is created as shown below

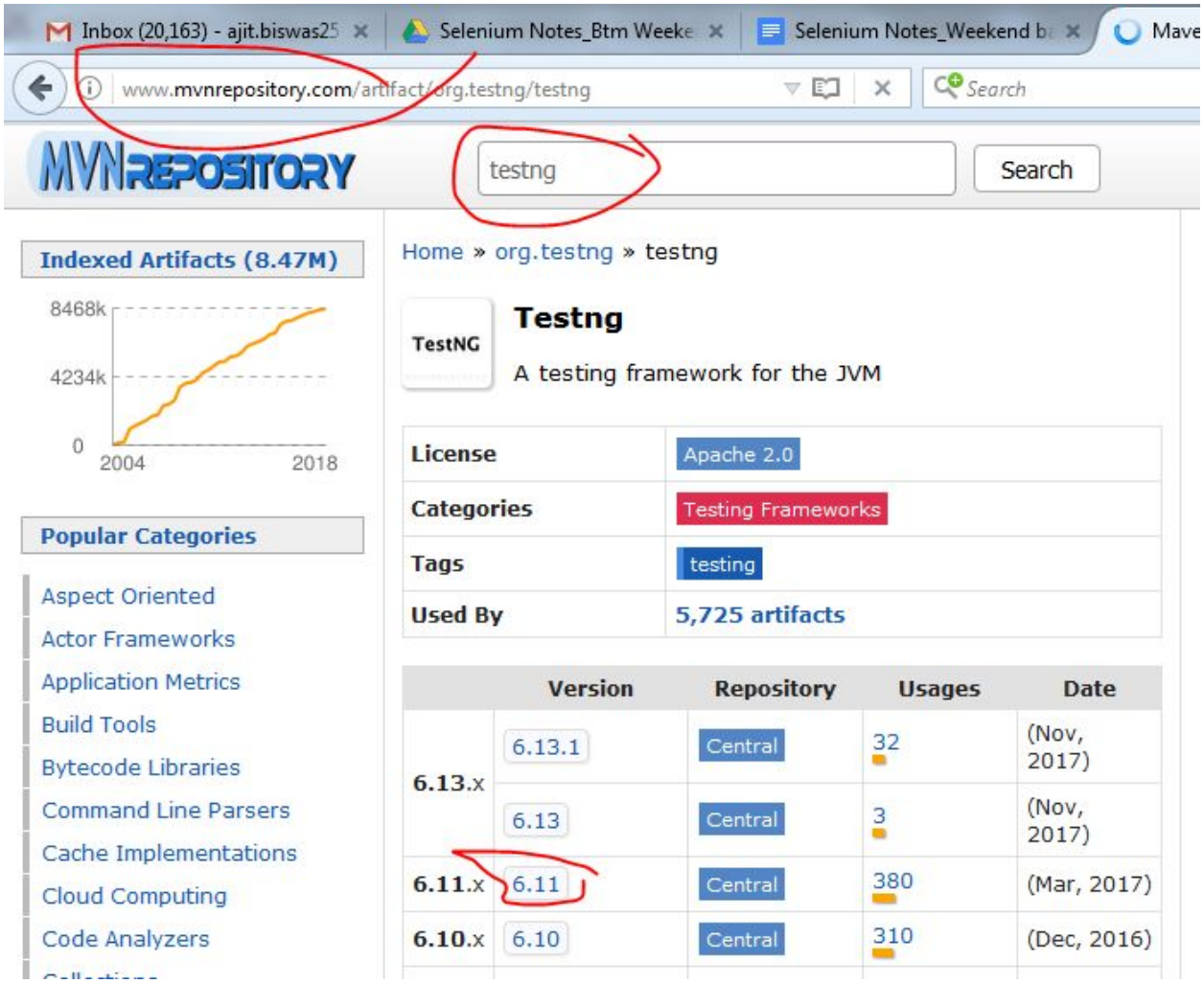


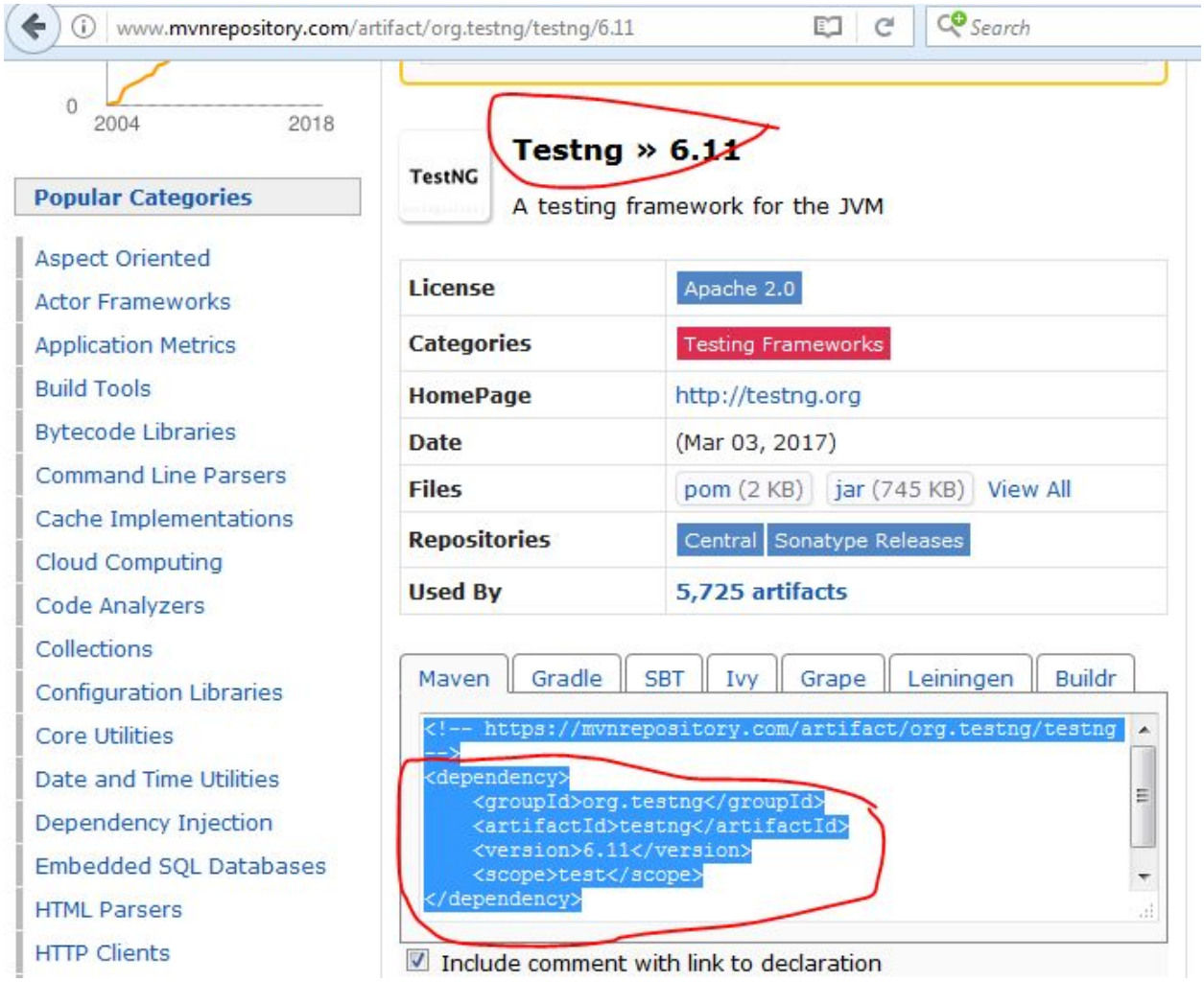
Verify that the selenium server dependency is added to the pom.xml tab as shown below.



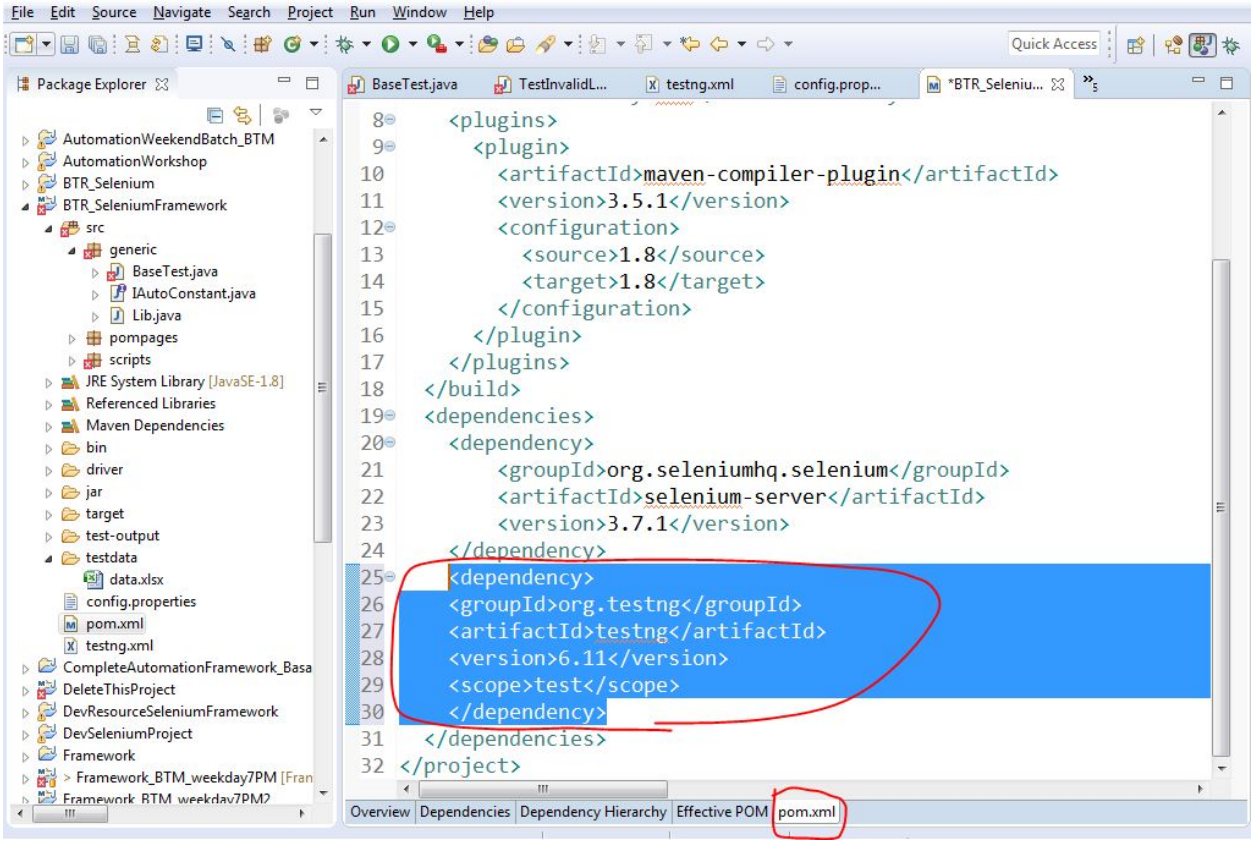
Now , add testng related dependency jar files to the pom.xml file.

How ?


Inbox (20,163) - ajit.biswas25
Selenium Notes_Btm Weeke
Selenium Notes_Weekend b
Mave
www.mvnrepository.com/artifact/org.testng/testng
Search
MVNREPOSITORY
testng
Search
Indexed Artifacts (8.47M)
8468k
4234k
0
2004
2018
Popular Categories
Aspect Oriented
Actor Frameworks
Application Metrics
Build Tools
Bytecode Libraries
Command Line Parsers
Cache Implementations
Cloud Computing
Code Analyzers
Home » org.testng » testng
TestNG
Testng
A testing framework for the JVM
License
Apache 2.0
Categories
Testing Frameworks
Tags
testing
Used By
5,725 artifacts
Version
Repository
Usages
Date
6.13.x
6.13.1
Central
32
(Nov, 2017)
6.13
Central
3
(Nov, 2017)
6.11.x
6.11
Central
380
(Mar, 2017)
6.10.x
6.10
Central
310
(Dec, 2016)



Now add testng related dependency information to the pom.xml file as shown below



We can also add browser specific driver executables using pom.xml file as shown below.

We need to add this dependency to the pom.xml file.

And then we need to add this line of code before launching the browser.

```java
package generic;
import java.util.concurrent.TimeUnit;
public class BaseTest implements IAutoConstant{
    public WebDriver driver;
    static{
        WebDriverManager.iedriver().setup();
        WebDriverManager.chromedriver().setup();
        WebDriverManager.firefoxdriver().setup();
    }
        @BeforeMethod
    public void setUp(){
        //Launch the browser
        driver = new InternetExplorerDriver();
        driver = new FirefoxDriver();
        driver = new ChromeDriver();
        //Enter the uri
        driver.get(Lib.getPropertyValue("URL"));
        //Set the implicit wait time period
        String timeout = Lib.getPropertyValue("IMPLICIT_TIME_OUT");
driver.manage().timeouts().implicitlyWait(Long.parseLong(timeout), TimeUnit.SECONDS);
    }
    @AfterMethod
    public void tearDown(ITestResult res){
        if( ITestResult.FAILURE == res.getStatus()){
            Lib.captureScreenshot(driver, res.getName());
        }
```

POM.XML file dependencies

_____

<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>AutomationProjectWeekendBatch</groupId>

<artifactId>AutomationProjectWeekendBatch</artifactId>

<version>0.0.1-SNAPSHOT</version>

```
<build>
  <sourceDirectory>src</sourceDirectory>
  <plugins>
    <plugin>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.5.1</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>2.20.1</version>
      <configuration>
        <suiteXmlFiles>
          <suiteXmlFile>testng.xml</suiteXmlFile>
        </suiteXmlFiles>
      </configuration>
```

```
    </plugin>
   </plugins>
  </build>
  <dependencies>
        <dependency>
   <groupId>io.github.bonigarcia</groupId>
   <artifactId>webdrivermanager</artifactId>
   <version>3.6.1</version>
</dependency>
        <dependency>
                <groupId>org.seleniumhq.selenium</groupId>
                <artifactId>selenium-server</artifactId>
                <version>3.7.1</version>
        </dependency>
        <dependency>
   <groupId>org.testng</groupId>
   <artifactId>testng</artifactId>
   <version>6.11</version>
   <scope>compile</scope>
</dependency>
```

```
        <dependency>
    <groupId>org.apache.poi</groupId>
    <artifactId>poi</artifactId>
    <version>3.17</version>
</dependency>
<dependency>
    <groupId>org.apache.poi</groupId>
    <artifactId>poi-ooxml</artifactId>
    <version>3.17</version>
</dependency>
<dependency>
    <groupId>org.apache.poi</groupId>
    <artifactId>poi-scratchpad</artifactId>
    <version>3.17</version>
</dependency>
<dependency>
    <groupId>org.apache.poi</groupId>
    <artifactId>poi-ooxml-schemas</artifactId>
    <version>3.17</version>
</dependency>
```

```
<dependency>
        <groupId>org.apache.poi</groupId>
        <artifactId>poi-examples</artifactId>
        <version>3.17</version>
</dependency>
<dependency>
   <groupId>org.apache.poi</groupId>
   <artifactId>poi-excelant</artifactId>
   <version>3.17</version>
</dependency>
<dependency>
   <groupId>commons-codec</groupId>
   <artifactId>commons-codec</artifactId>
   <version>1.11</version>
</dependency>
<dependency>
   <groupId>commons-io</groupId>
   <artifactId>commons-io</artifactId>
   <version>2.6</version>
</dependency>
```

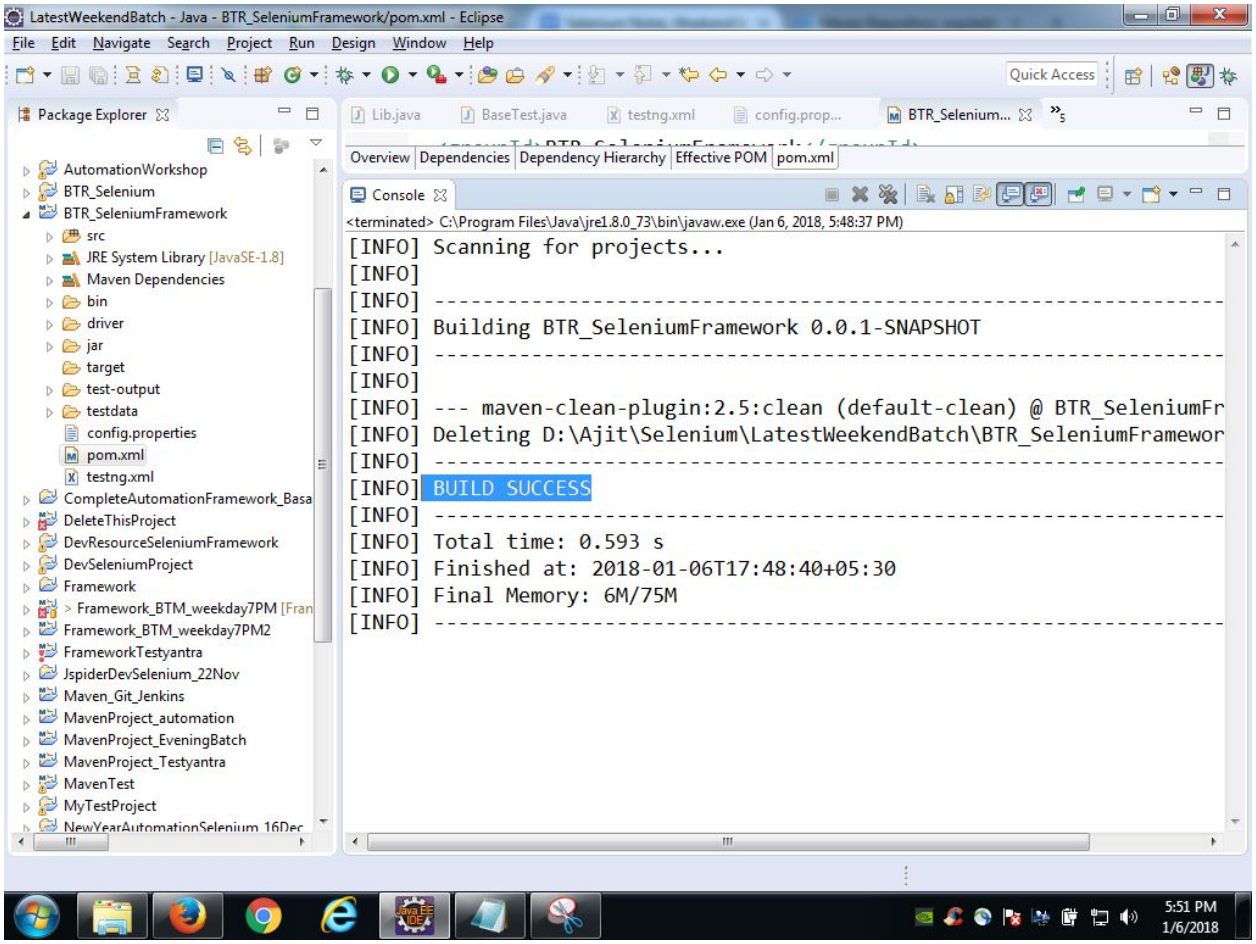
```
<dependency>
    <groupId>commons-logging</groupId>
    <artifactId>commons-logging-api</artifactId>
    <version>1.1</version>
</dependency>
<dependency>
    <groupId>com.github.virtuald</groupId>
    <artifactId>curvesapi</artifactId>
    <version>1.05</version>
</dependency>
<dependency>
    <groupId>org.apache.xmlbeans</groupId>
    <artifactId>xmlbeans</artifactId>
    <version>2.6.0</version>
</dependency>
<dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-core</artifactId>
    <version>2.9.1</version>
</dependency>
```

```
<dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-api</artifactId>
    <version>2.9.1</version>
</dependency>
  </dependencies>
</project>
```

How to run testng.xml file from pom.xml ?

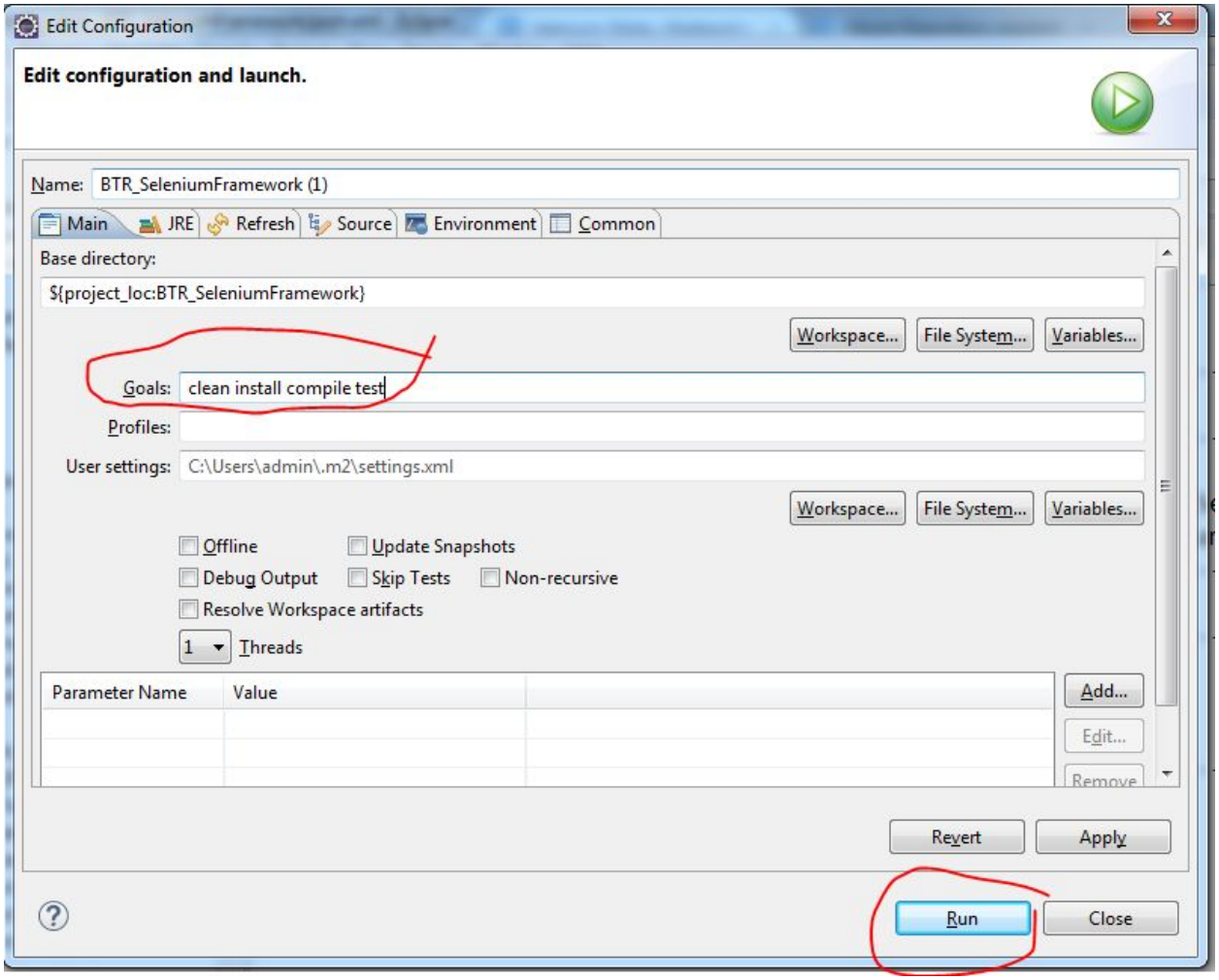
Right click on pom.xml file and run as maven clean for the first time.

u get the build success message as shown bleow.



now run using below command, **clean install compile test**

Run as maven build (2nd option)



Solutions in case of maven issues

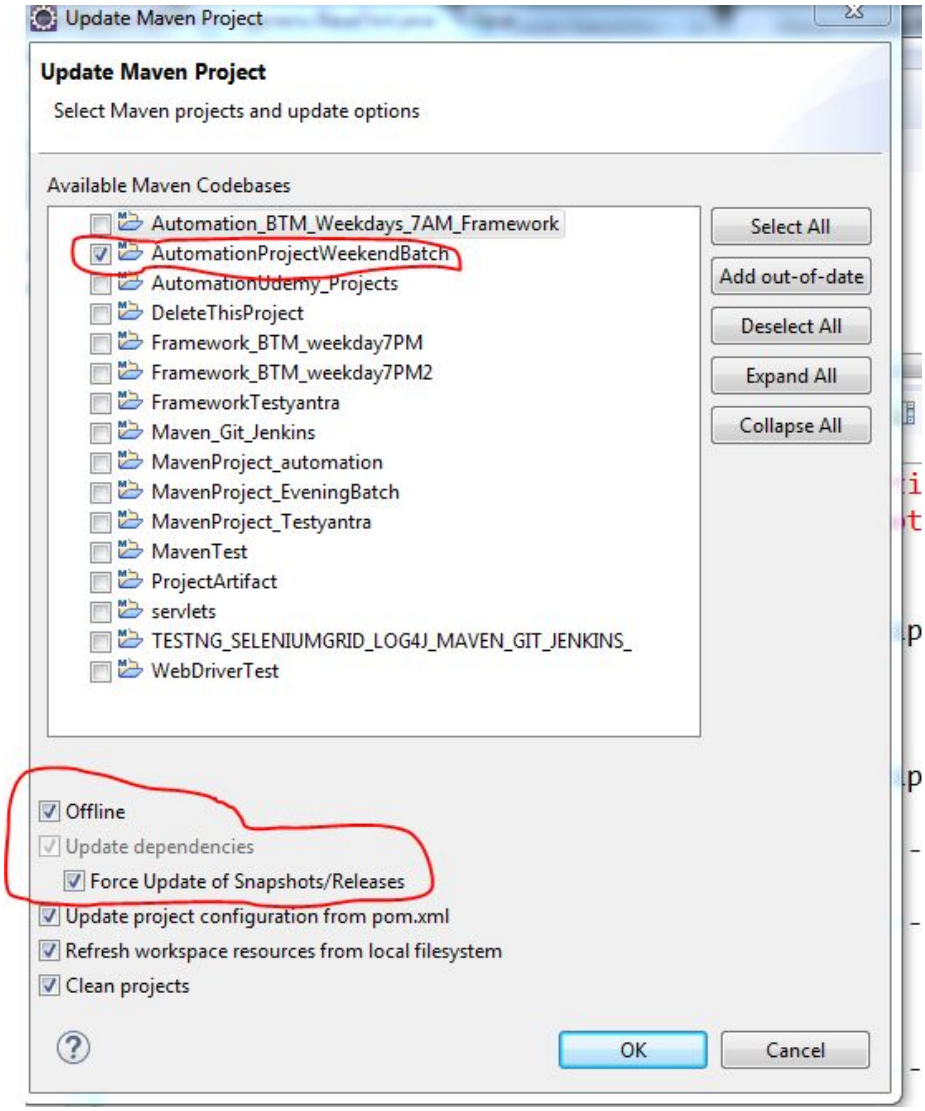
1. No compiler added to the build

Solution : Windows - preference -- Java - Installed JREs →search--> navigate to location where your jdk1.8 is installed (c--program files --java - JDK)> select bin folder> apply> close.

2. MoJoFailureException

Right click on the project → maven → update project → Select the project and select the highlighted options:

- offline checkbox
- force update of snapshots



Note : Still if it doesn't work, delete the .m2 folder from the below location.

C://users/admin/.m2

Q1 : Who is responsible to build the maven project ? OR

How maven works ?

Maven compiler plugin is responsible for building the maven project. This compiler plugin utilises all the dependency files to build the project. How it works is : First, compiler checks for the availability of the dependency file in maven local repository (that is .m2 folder). If the file is not present in .m2 folder, what it does is : it goes to the respective website specified in the pom.xml file and downloads the files in .m2 folder for further use. And in case, if the dependency file is already present in .m2 folder, it doesn't go to the website, instead, it utilises the resources from the maven local repo only to build the project. This is how maven works.

Q2 : How maven executes the test suite ?

By using a plugin called Maven-Surefire-plugin. This plugin is responsible for executing the maven project. All we have to let the surefire plugin know is the path of the suite files which we want to execute.

Q3: What are the commands or goals to execute maven project ?

The command to execute the maven project is : clean install compile test

clean -- this command/goal is used to clean any previous history or reports.

install - this command is used to install any resources if required to build the project.

compile - this command is responsible to build the project.

test - this command is used to trigger the execution of maven project.

JENKINS

It is a continuous integration tool widely used by software project.

Advantages of Jenkins :

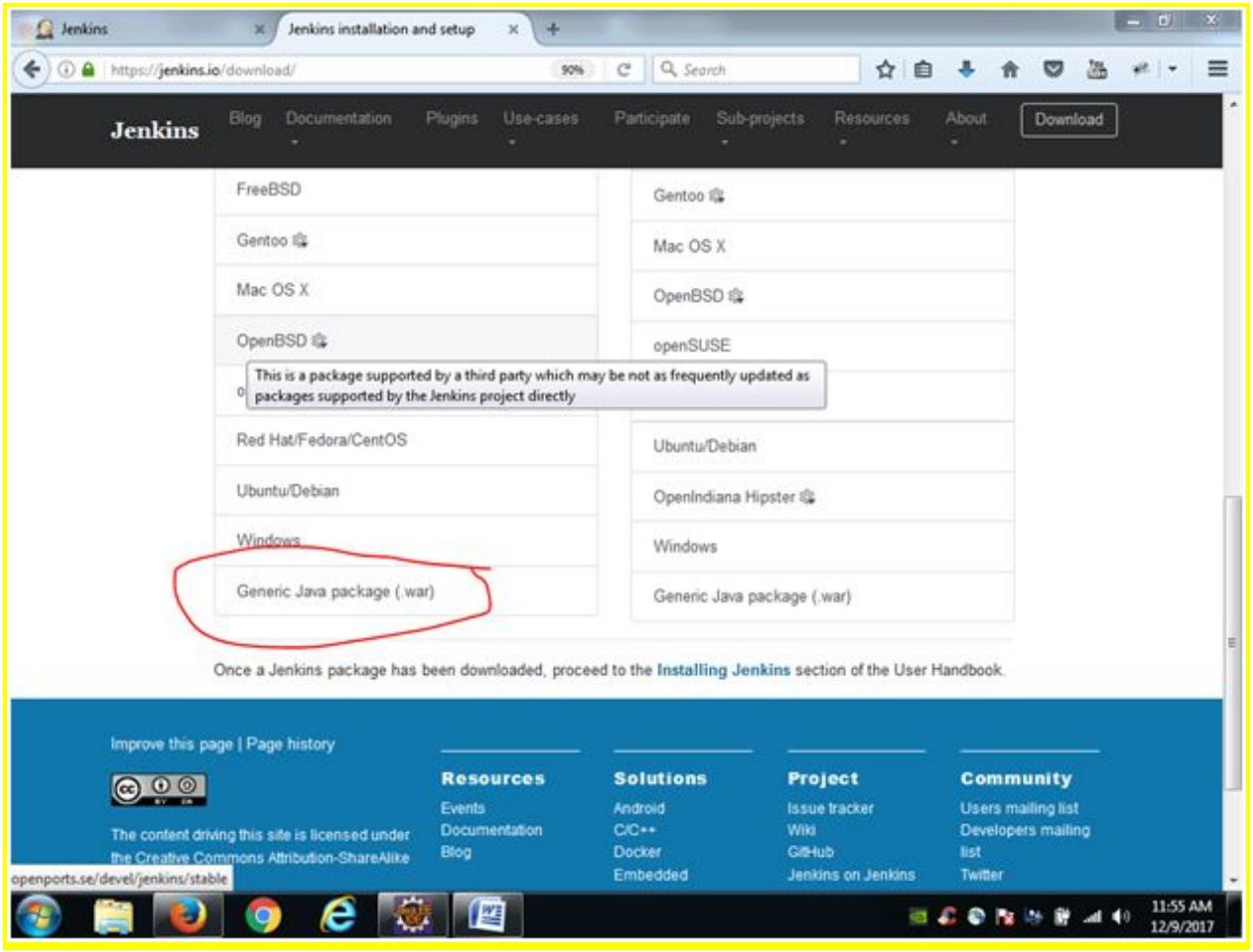
1. **We can schedule the time for automation framework suite execution.**
2. **Automatic kick off of automation suite execution as soon as the build gets deployed from DEV environment to QA environment.**

How to download jenkins

URL :

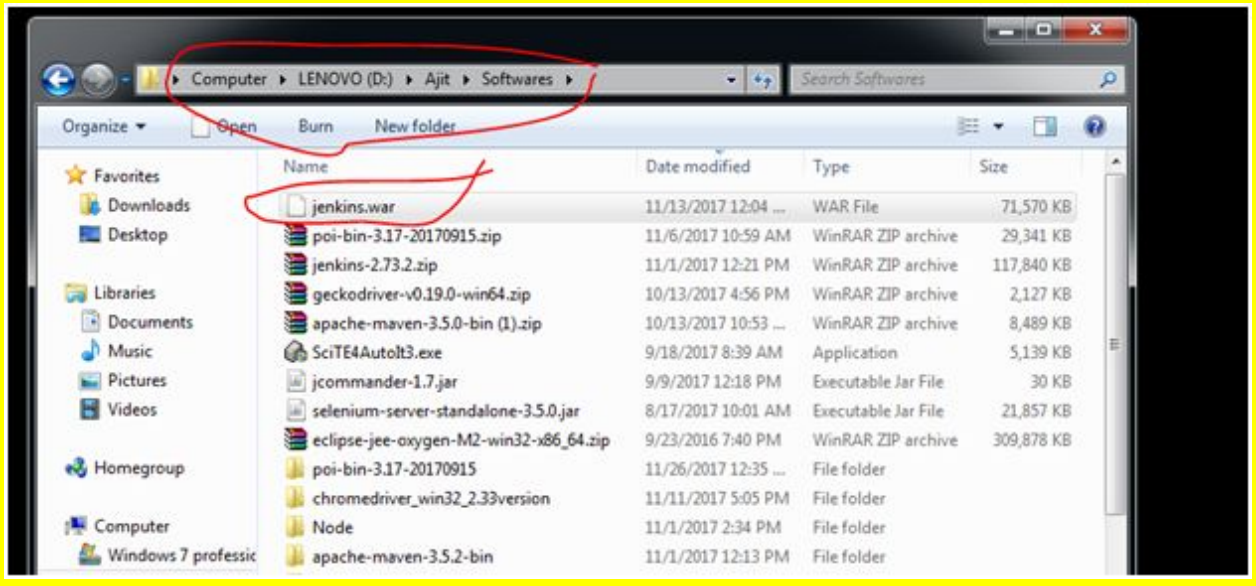
https://jenkins.io/download/

click on the link highlighted below.



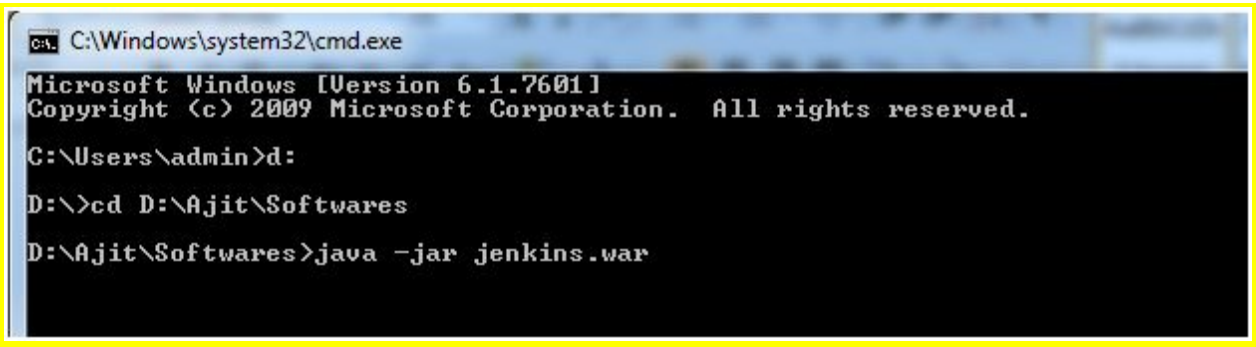
How to start the Jenkins server from Command prompt ?

Navigate to the below location where ur jenkins.war is placed.



Open the command prompt

And type the below command



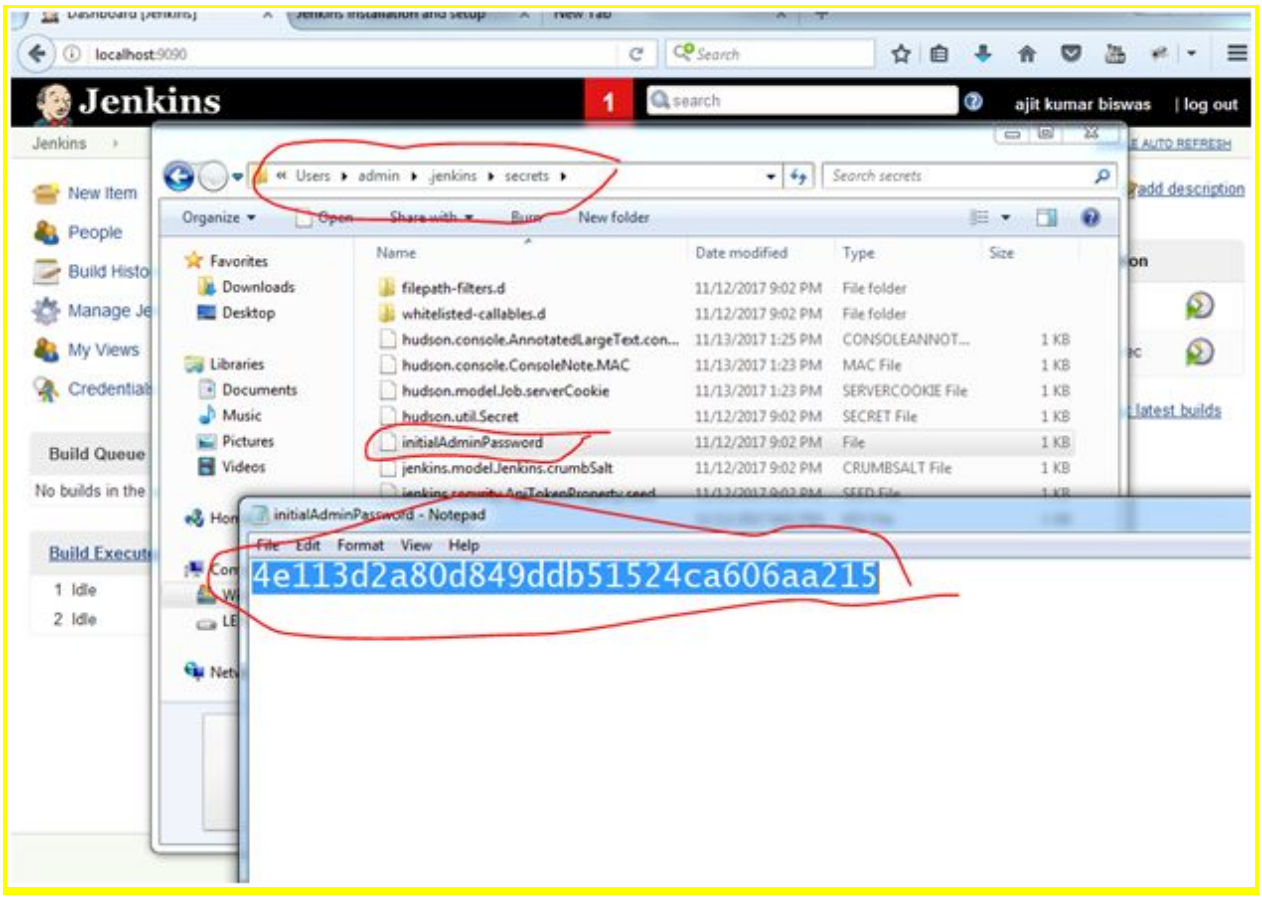
It will say – Jenkins Is up and running

Now open the jenkins dashboard.

Login to Jenkins using the below username and password

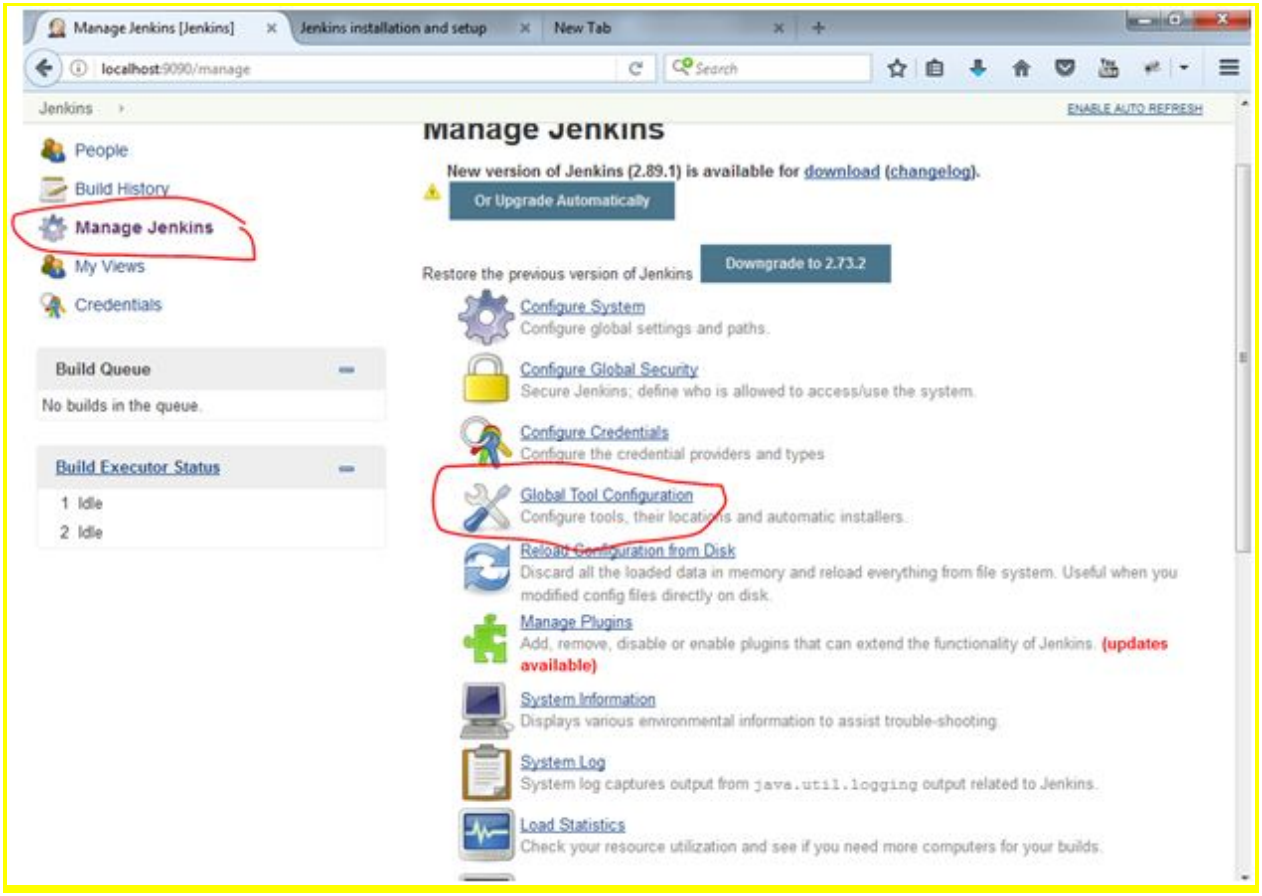
Username : admin

Password is below;

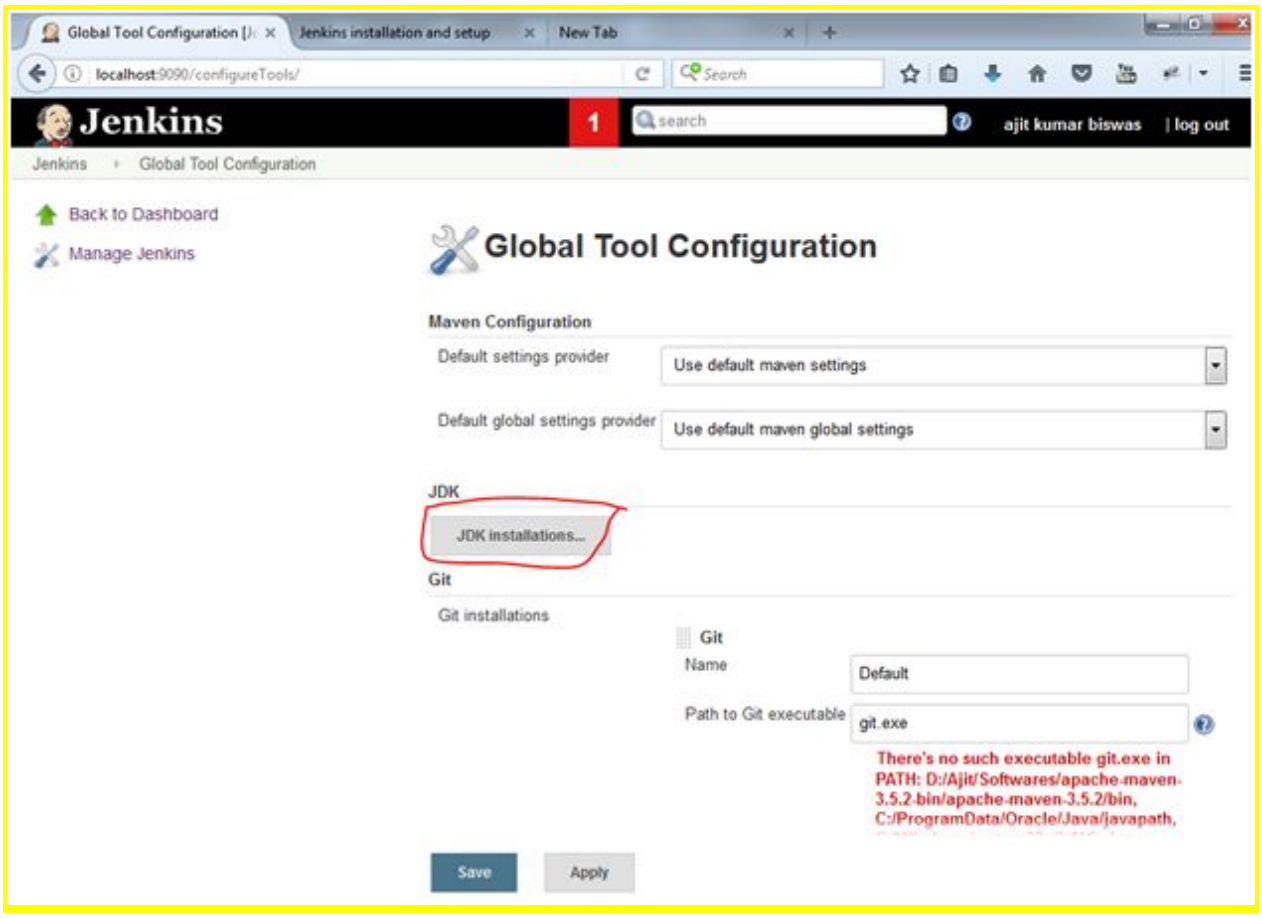


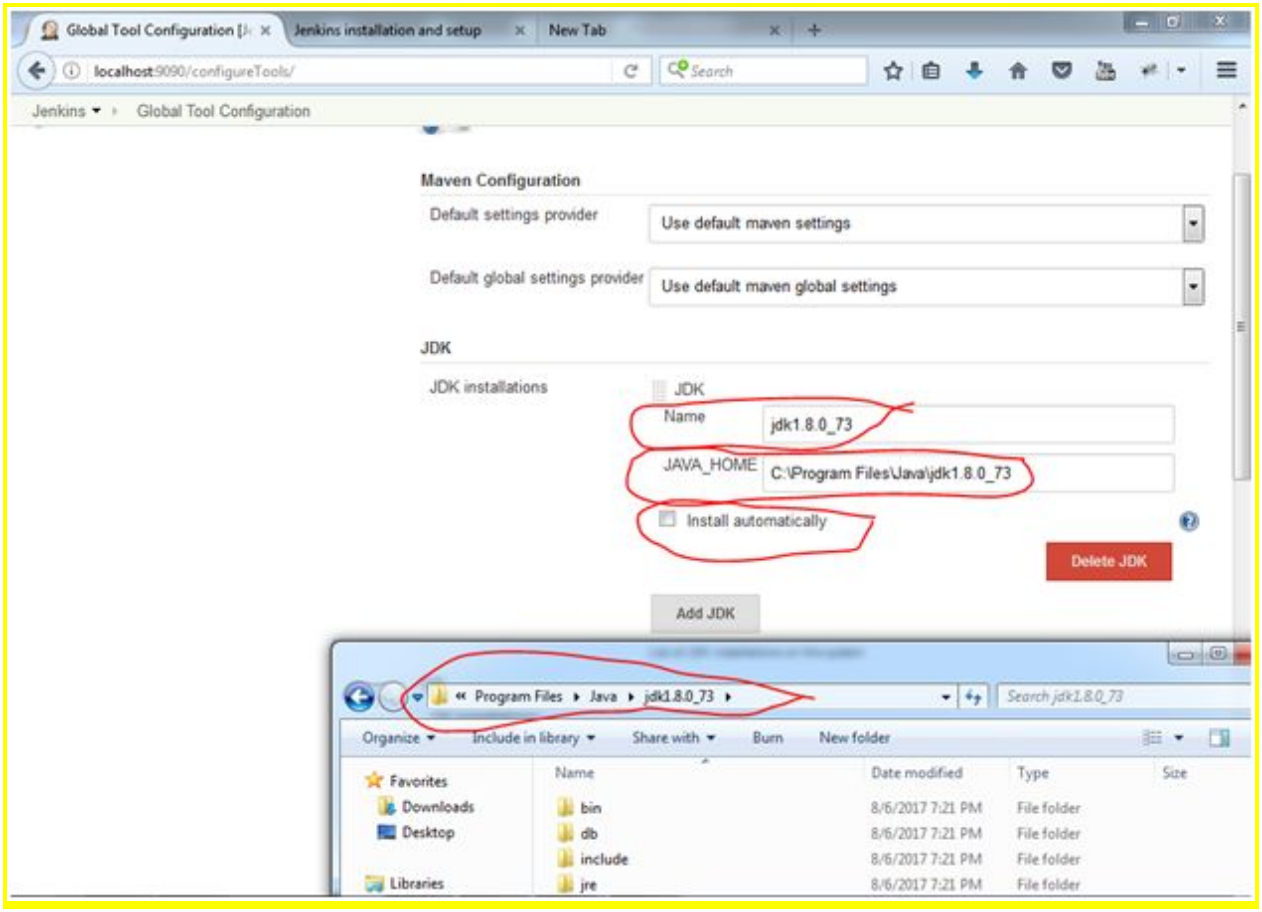
Jenkins dashboard page.

Click on Manage Jenkins and global tool configuration



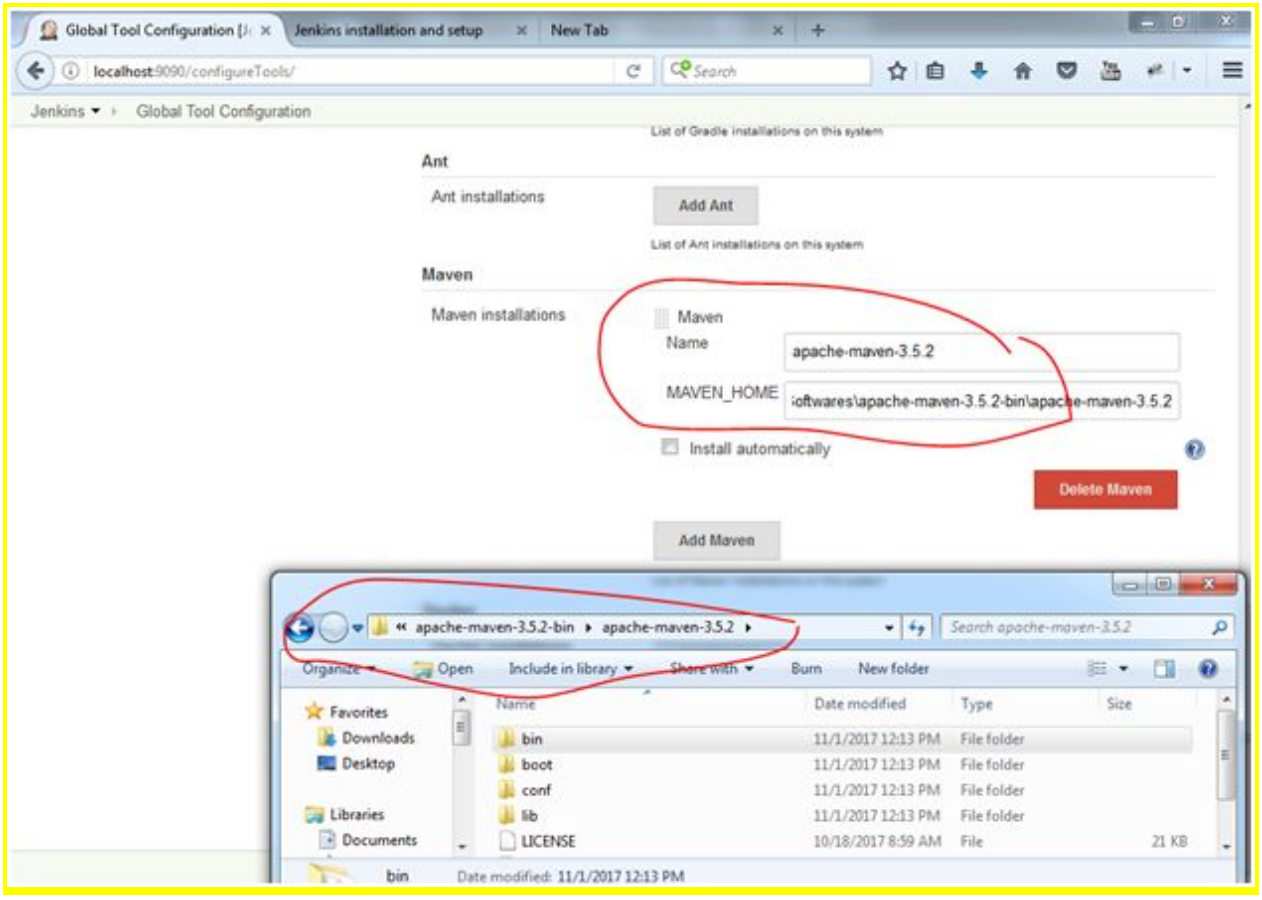
Add JDK to jenkins


Global Tool Configuration
Jenkins installation and setup
New Tab
localhost:9090/configureTools/
Search
Jenkins
1
search
ajit kumar biswas
| log out
Jenkins
Global Tool Configuration
Back to Dashboard
Manage Jenkins
Global Tool Configuration
Maven Configuration
Default settings provider
Use default maven settings
Default global settings provider
Use default maven global settings
JDK
JDK installations...
Git
Git installations
Git
Name
Default
Path to Git executable
git.exe
There's no such executable git.exe in PATH: D:/Ajit/Softwares/apache-maven-3.5.2-bin/apache-maven-3.5.2/bin, C:/ProgramData/Oracle/Java/javapath,
Save
Apply

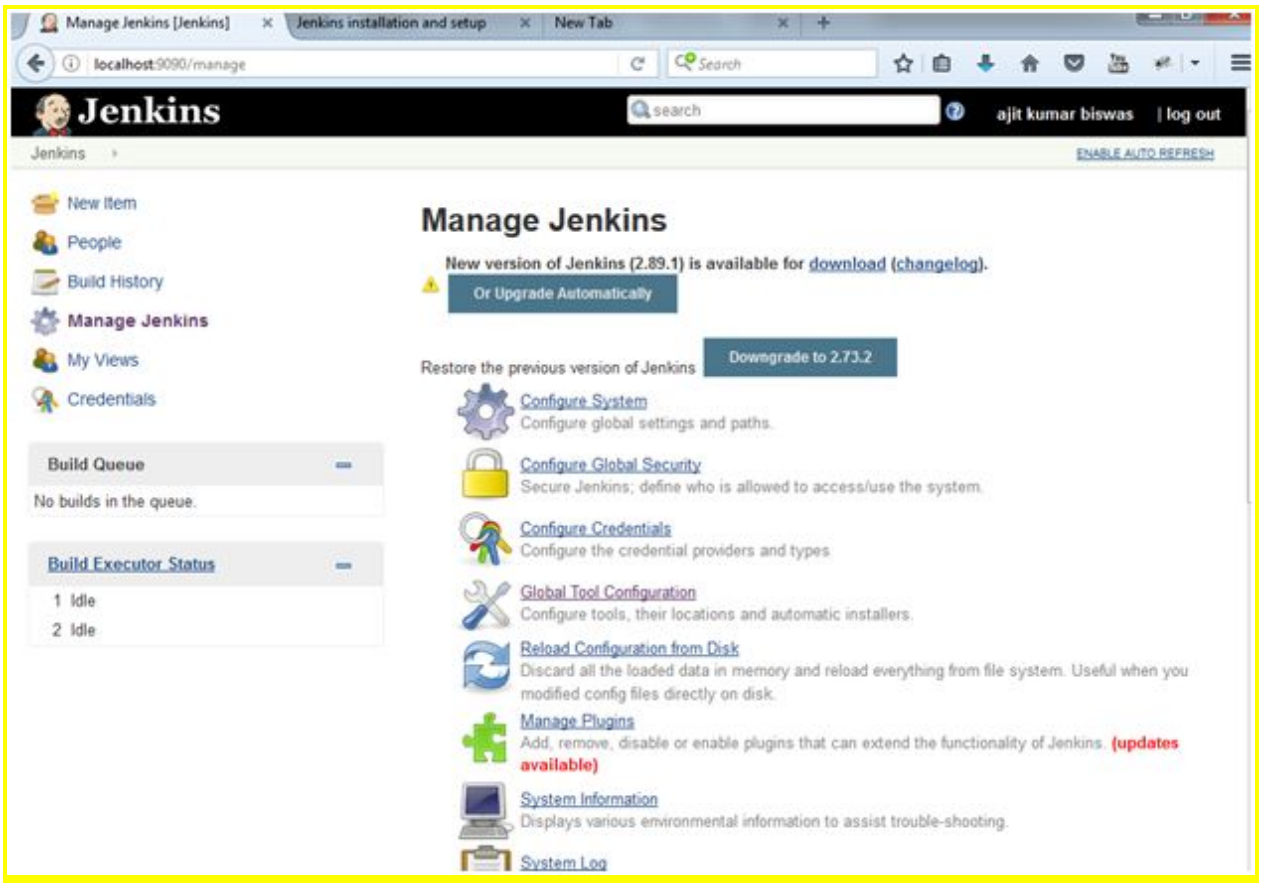


Add Maven Path as shown below.

Precondition : you need to download apache maven jar files

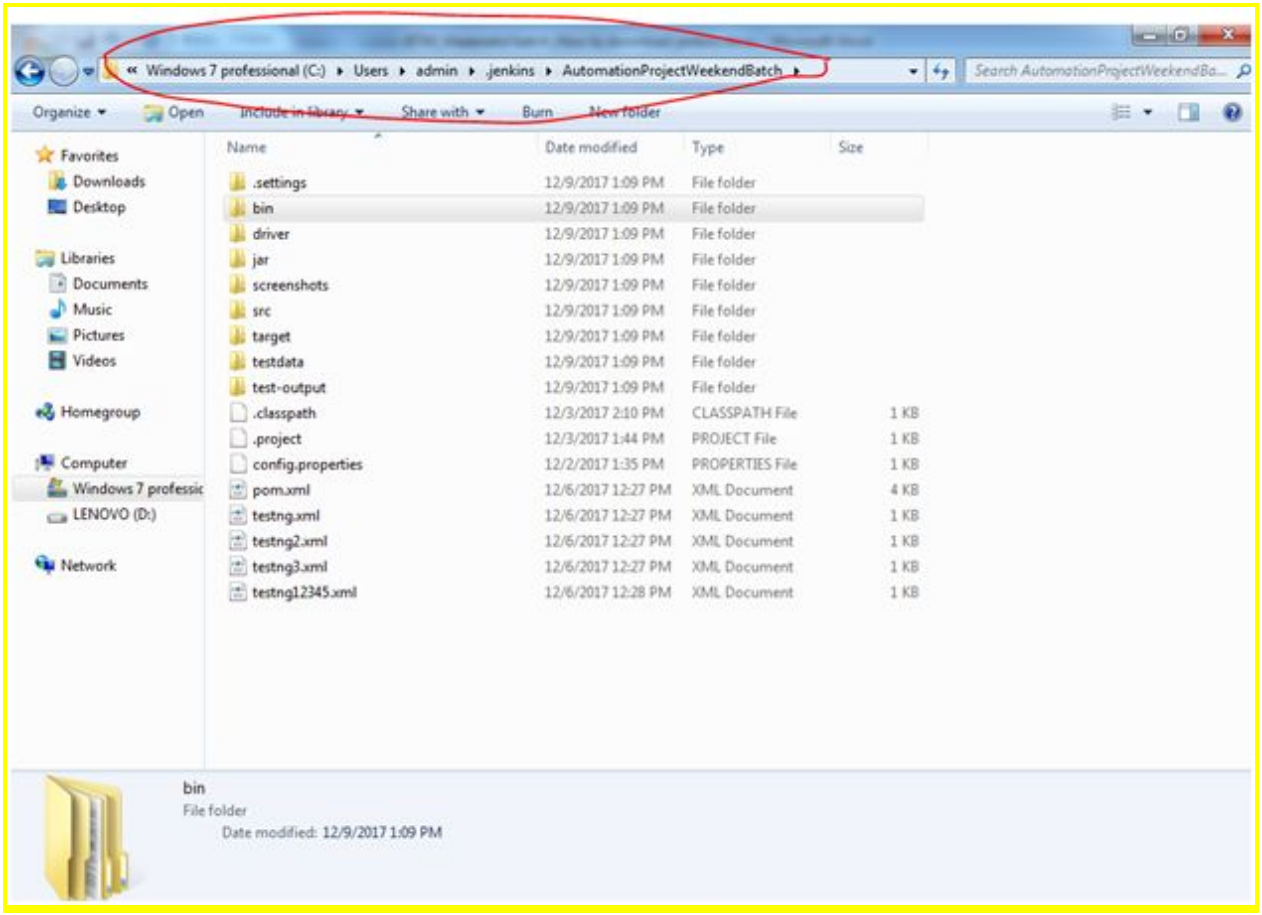


Once you save, u get this page.

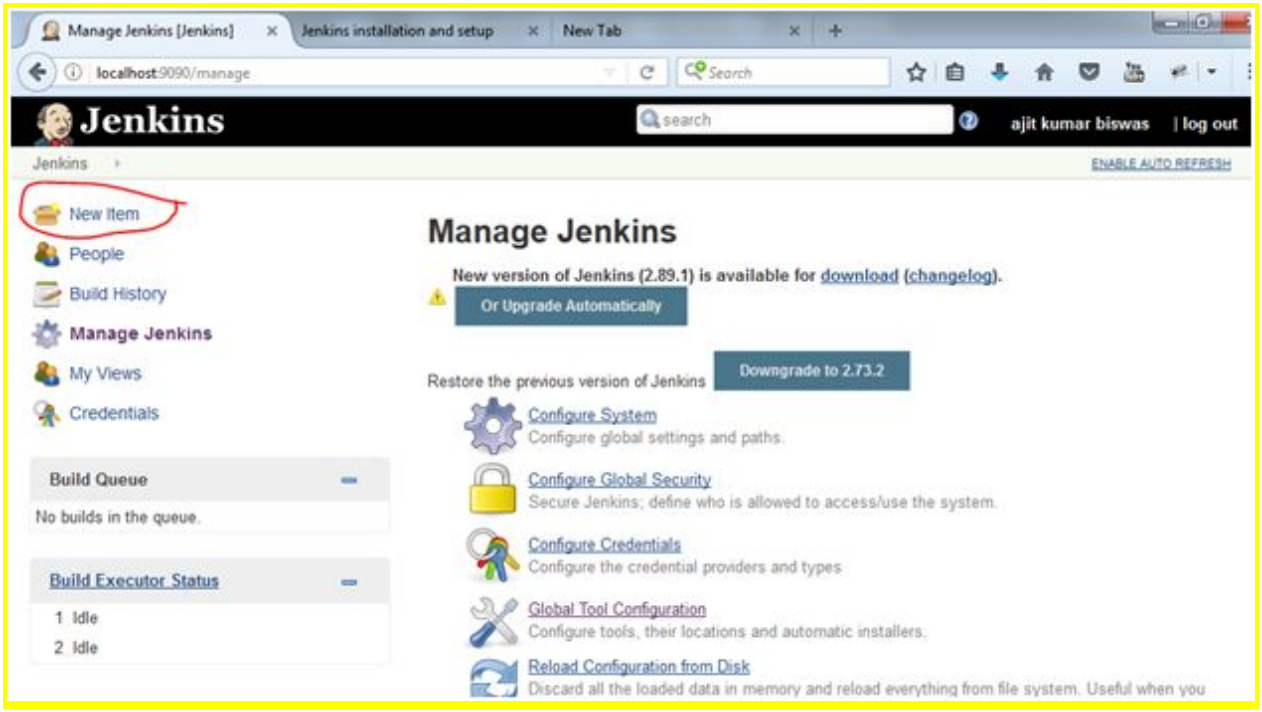


Copy the project that you want to execute from Jenkins as shown below

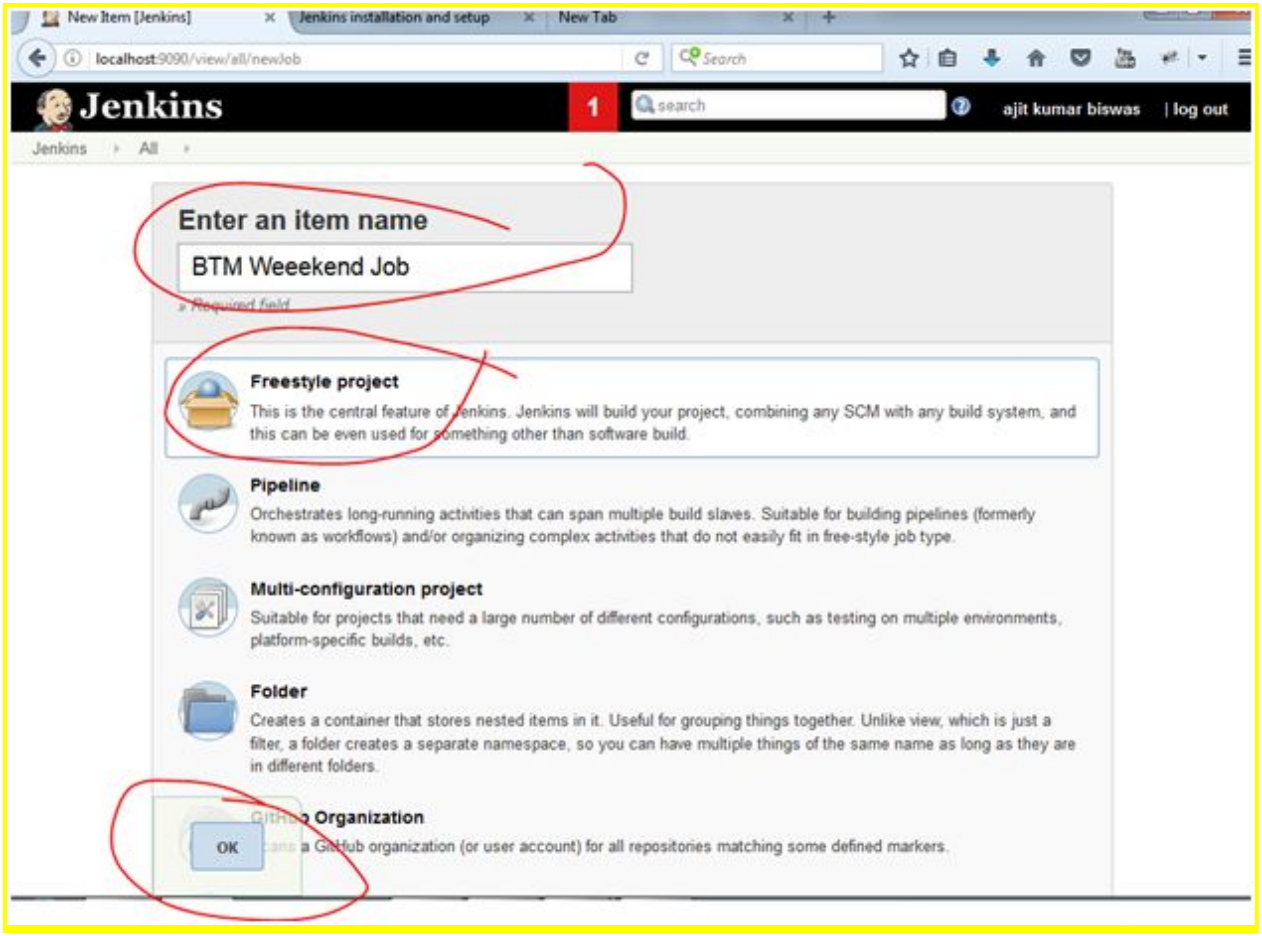
Copy the path till the project name

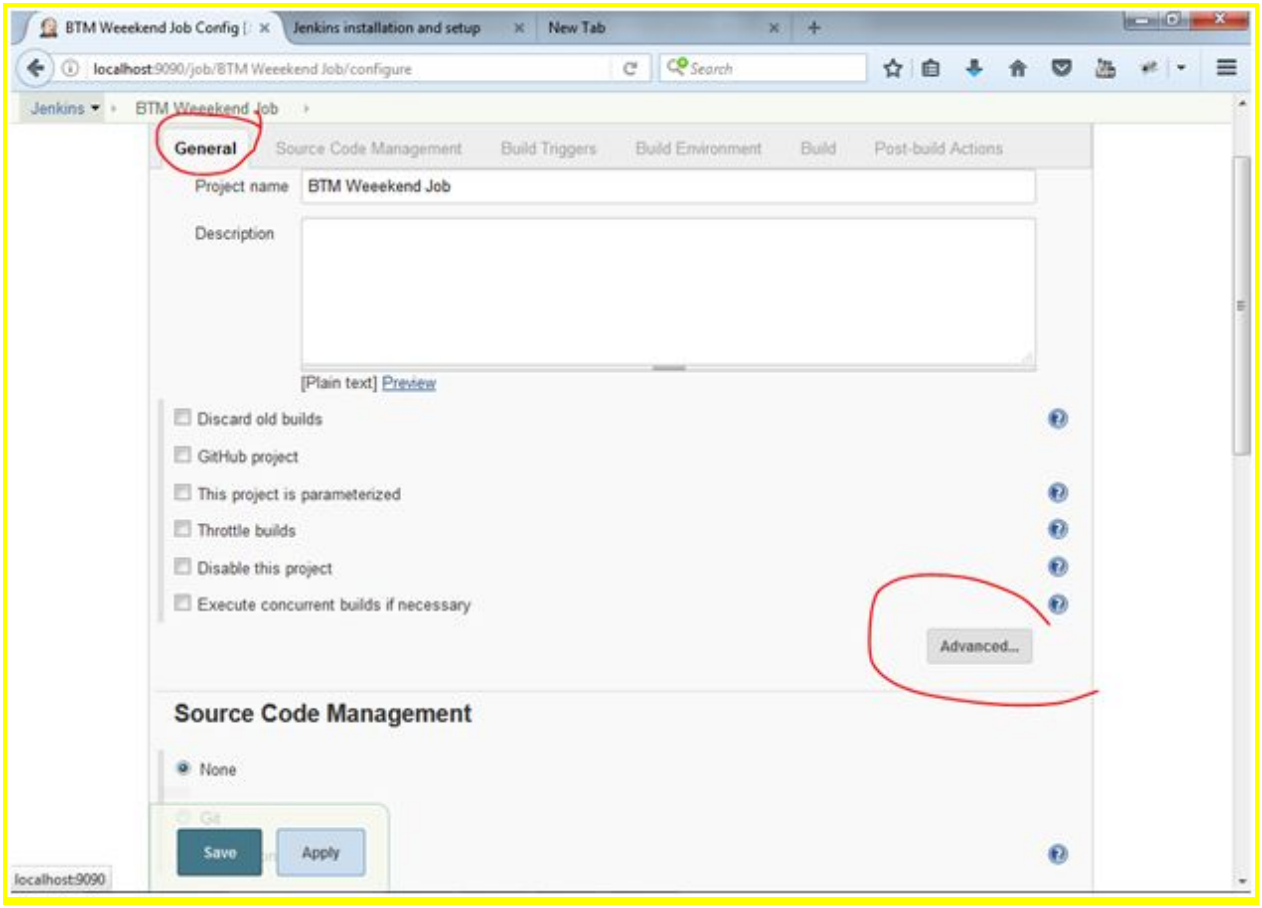


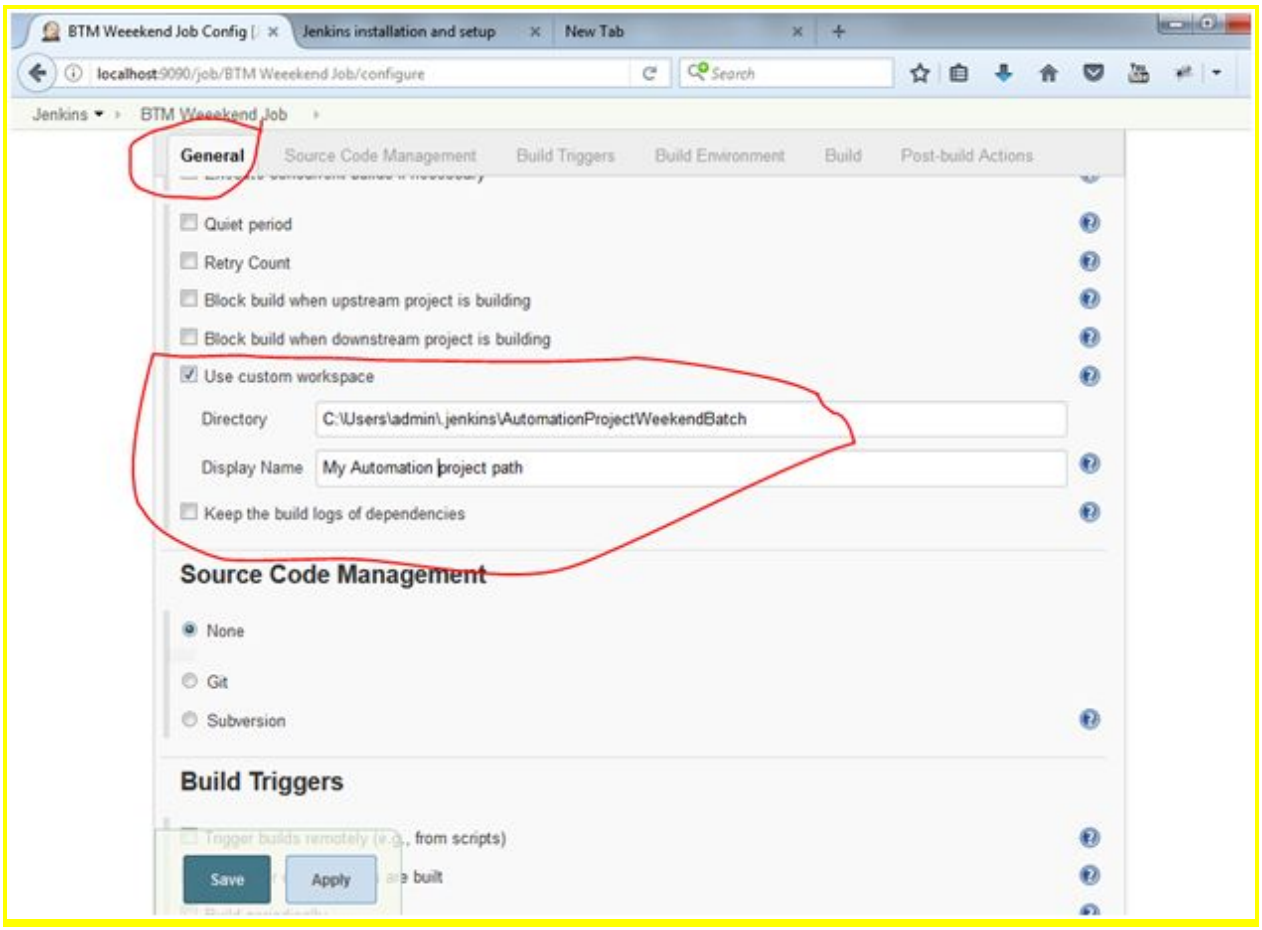
Create a new job by clicking on new ITEM



Enter name and click on free style project and click on OK button

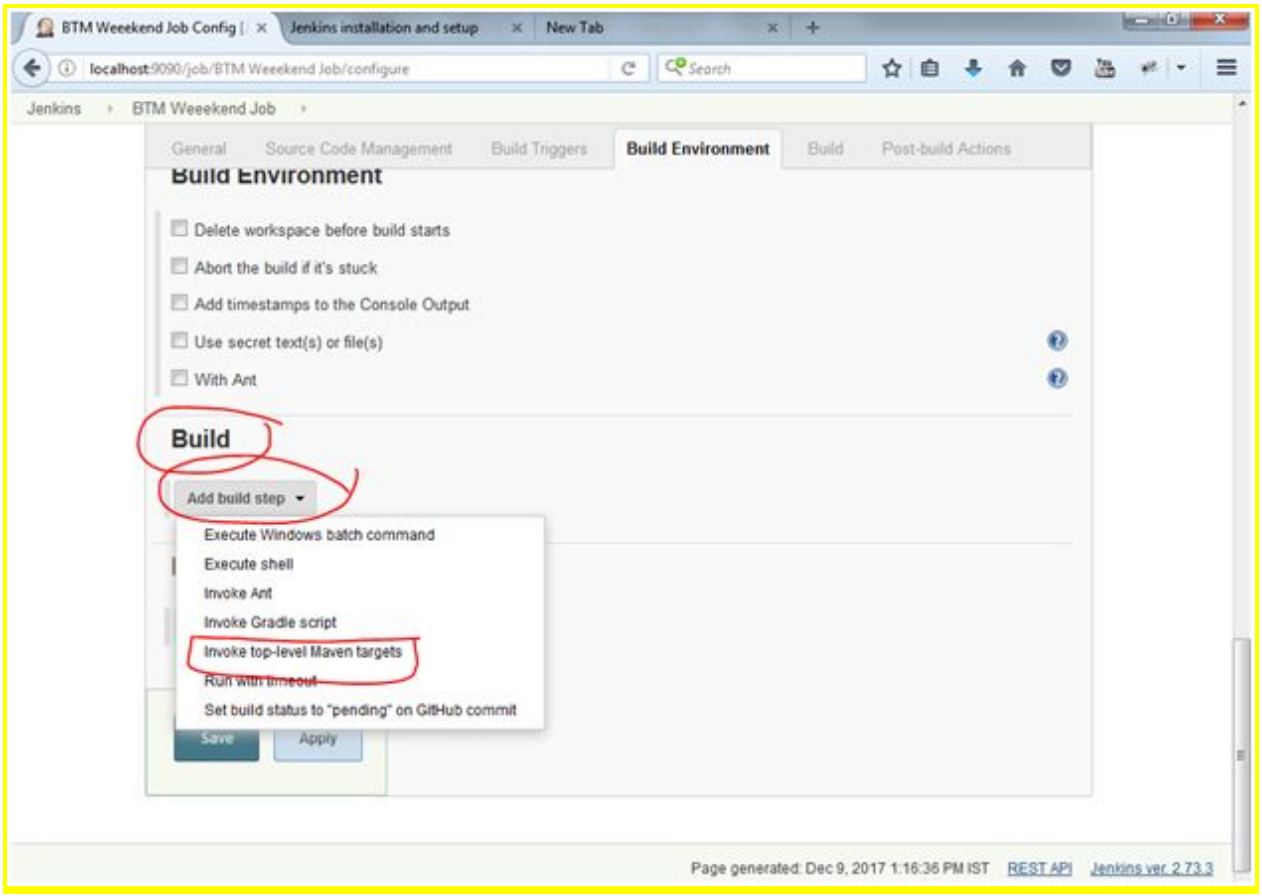

New Item [Jenkins]
Jenkins installation and setup
New Tab
localhost:9090/view/all/newJob
Search
Jenkins
1
search
ajit kumar biswas
| log out
Jenkins
All
Enter an item name
BTM Weeekend Job
» Required field
Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
Pipeline
Orchestrates long-running activities that can span multiple build slaves. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
Organization
a GitHub organization (or user account) for all repositories matching some defined markers.
OK

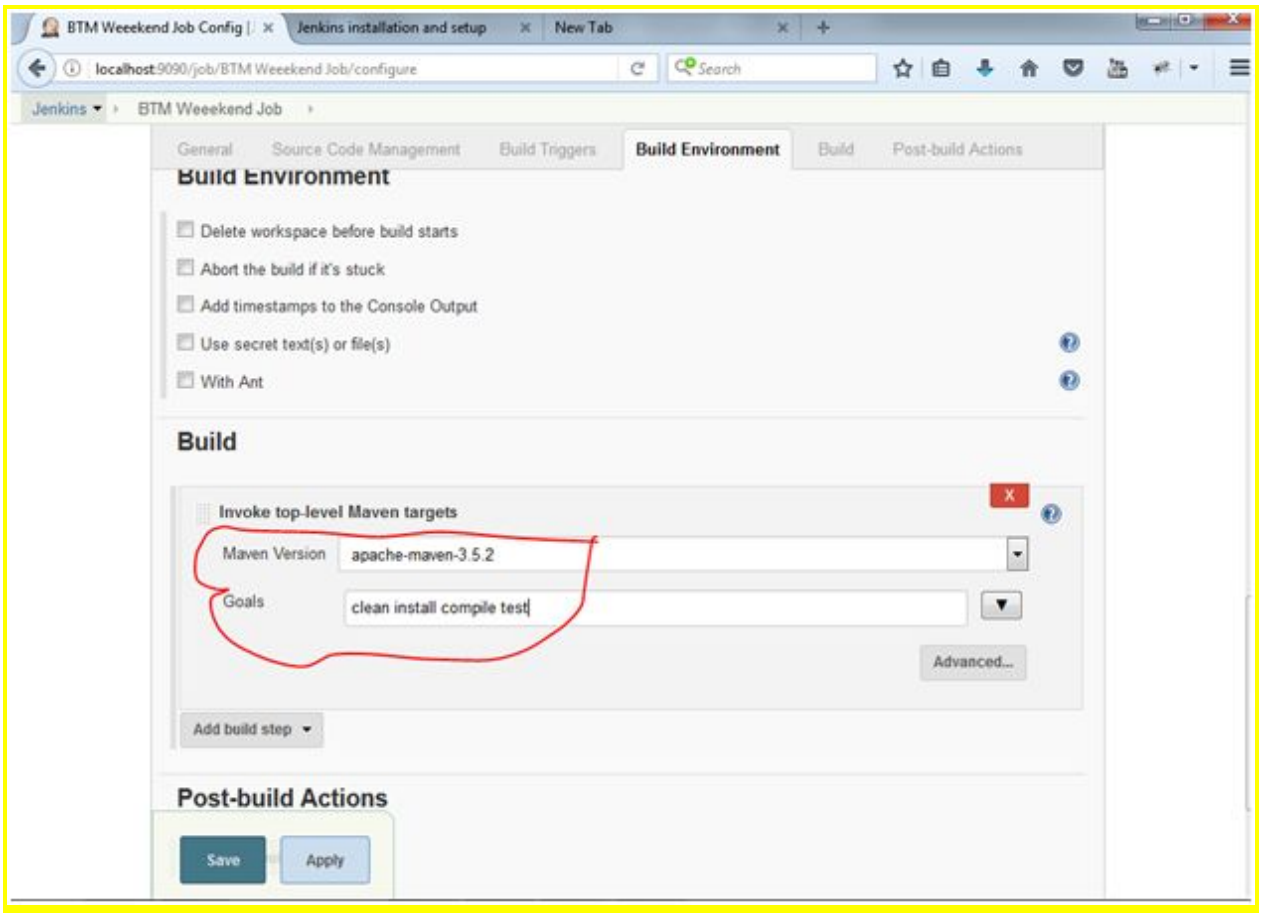

BTM Weeekend Job Config
Jenkins installation and setup
New Tab
localhost:9090/job/BTM Weeekend Job/configure
Search
Jenkins
BTM Weeekend Job
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Project name
BTM Weeekend Job
Description
[Plain text] Preview
Discard old builds
GitHub project
This project is parameterized
Throttle builds
Disable this project
Execute concurrent builds if necessary
Advanced...
Source Code Management
None
Save
Apply
localhost:9090



how to schedule suite execution time.

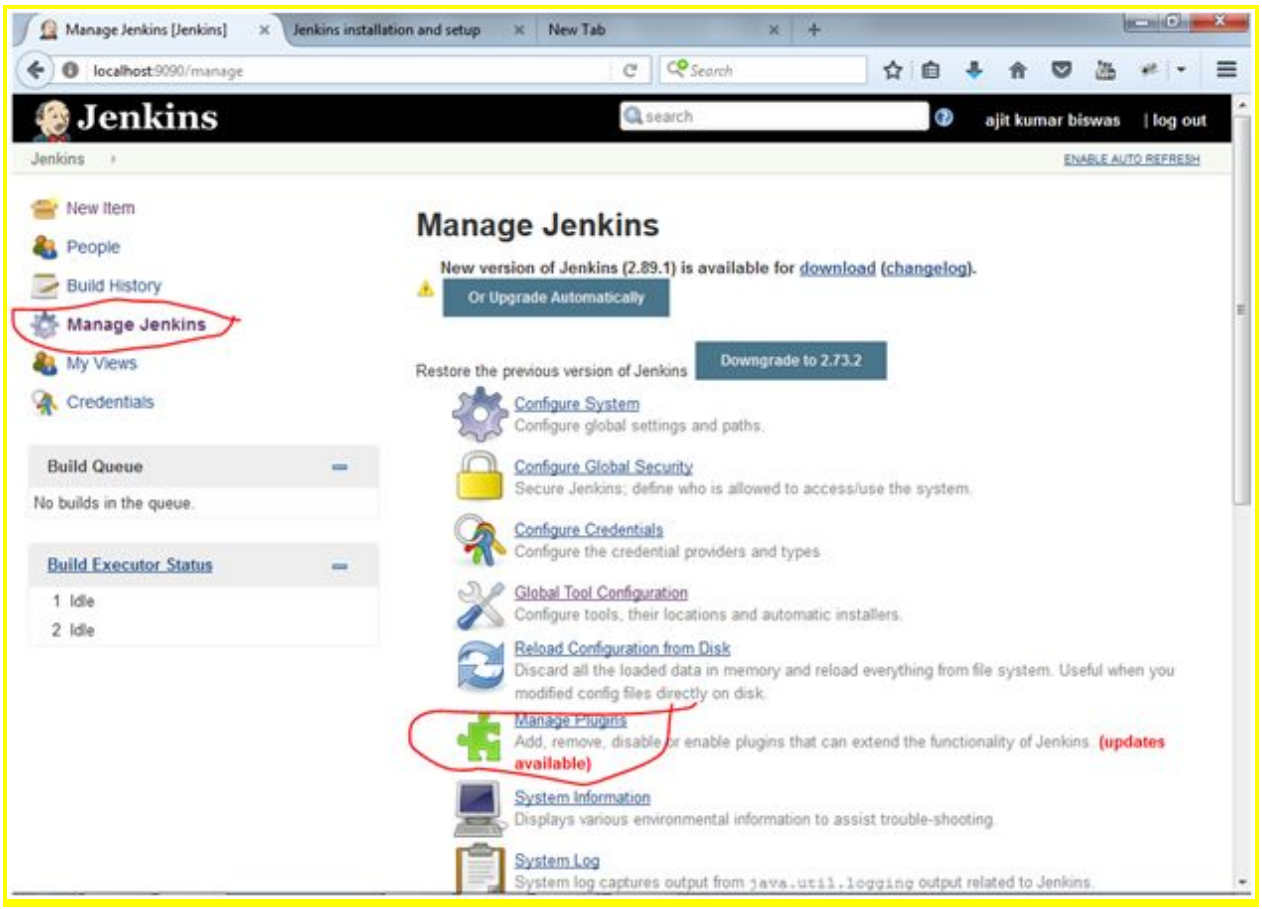

Build Triggers
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Build periodically
Schedule
30 23 * * *
Spread load evenly by using 'H 23 * * *' rather than '30 23 * * *'
Would last have run at Friday, December 8, 2017 11:30:22 PM IST; would next run at Saturday, December 9, 2017 11:30:22 PM IST.


BTM Weeekend Job Config
Jenkins installation and setup
New Tab
localhost:9090/job/BTM Weeekend Job/configure
Search
Jenkins
BTM Weeekend Job
General
Source Code Management
Build Triggers
Build Environment
Build
Post-build Actions
Build Environment
Delete workspace before build starts
Abort the build if it's stuck
Add timestamps to the Console Output
Use secret text(s) or file(s)
With Ant
Build
Add build step
Execute Windows batch command
Execute shell
Invoke Ant
Invoke Gradle script
Invoke top-level Maven targets
Set build status to "pending" on GitHub commit
Save
Apply
Page generated: Dec 9, 2017 1:16:36 PM IST
REST API
Jenkins ver. 2.73.3

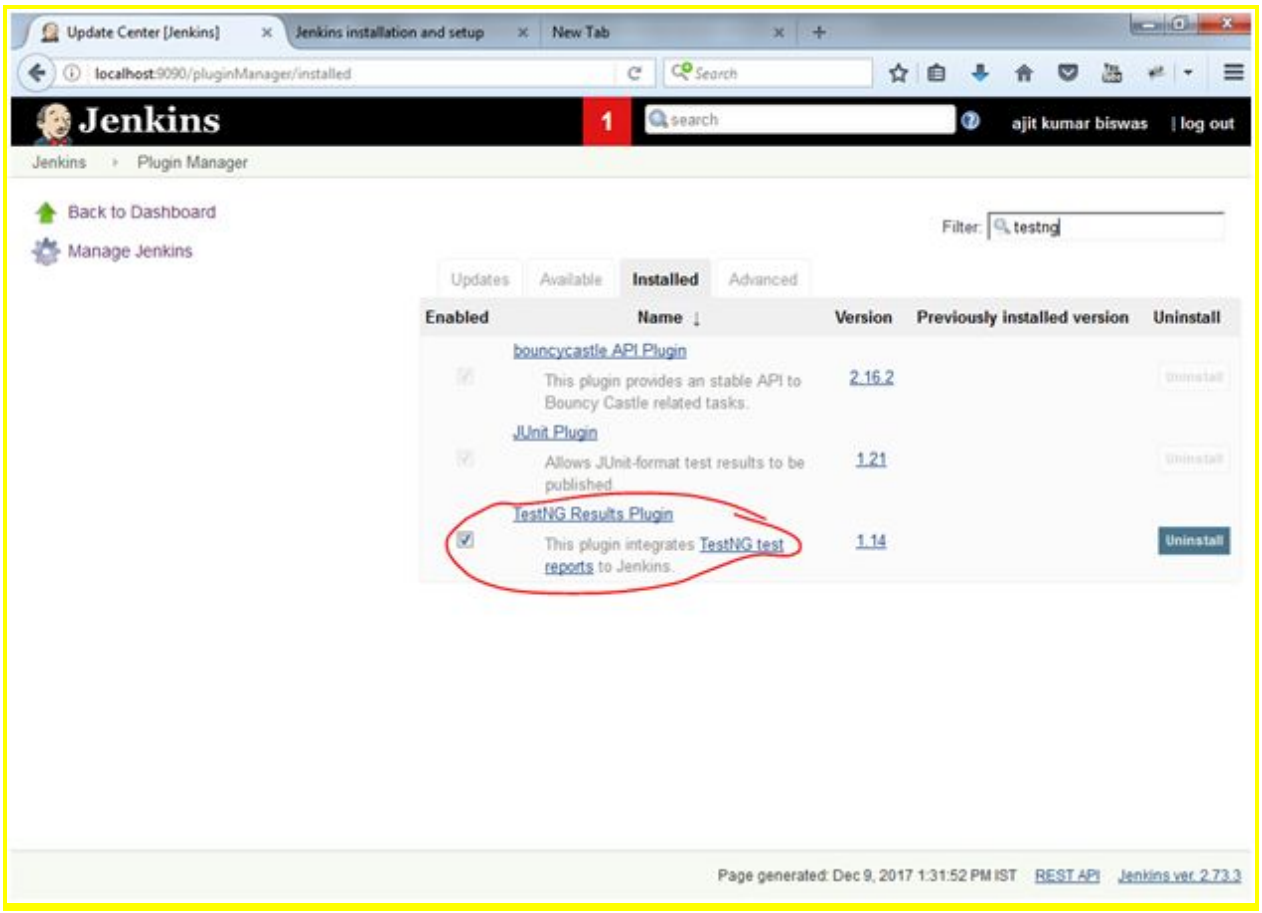


Save the above

Install testng results plugin

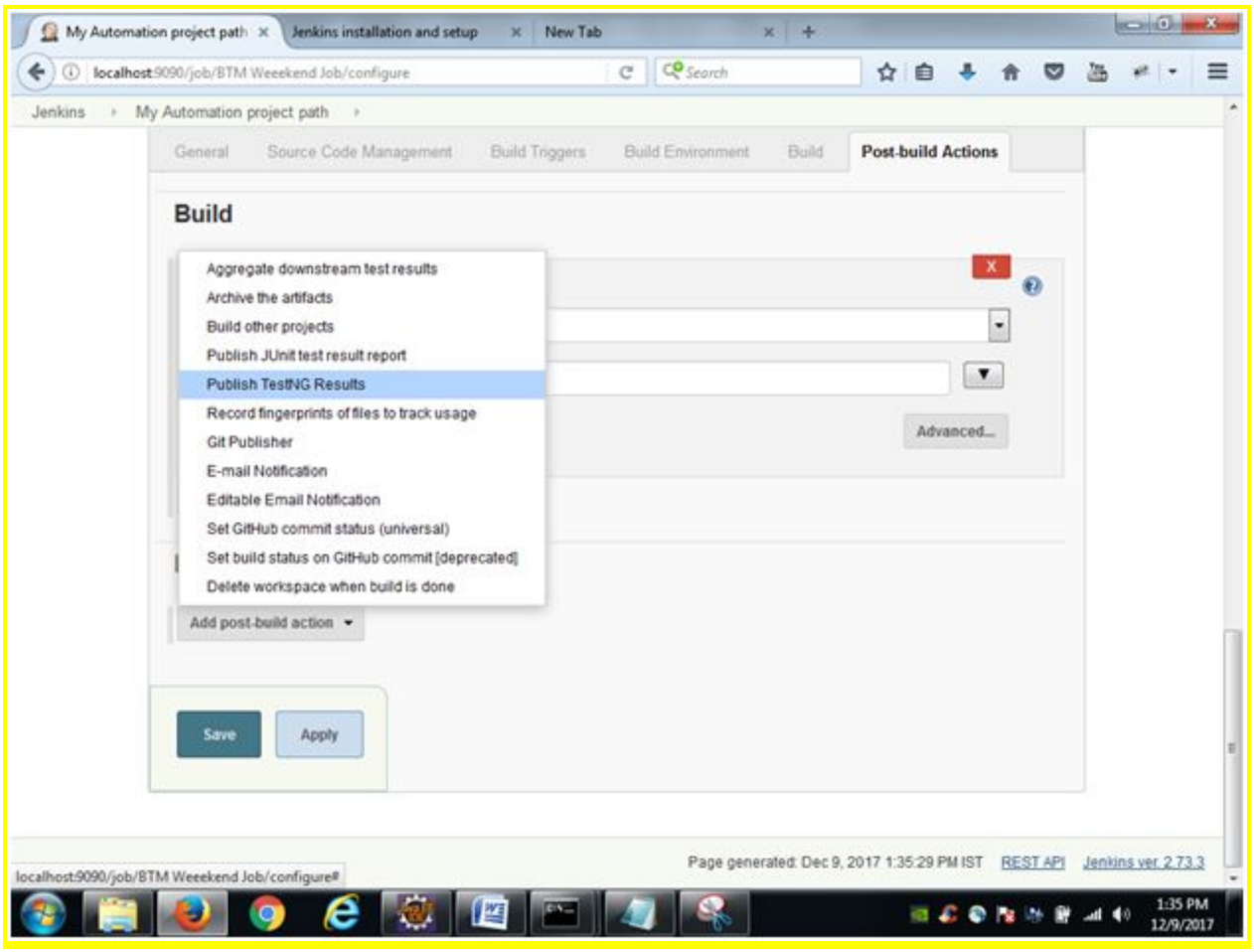


Navigate to available tab and search Testng Result plugin and install ..

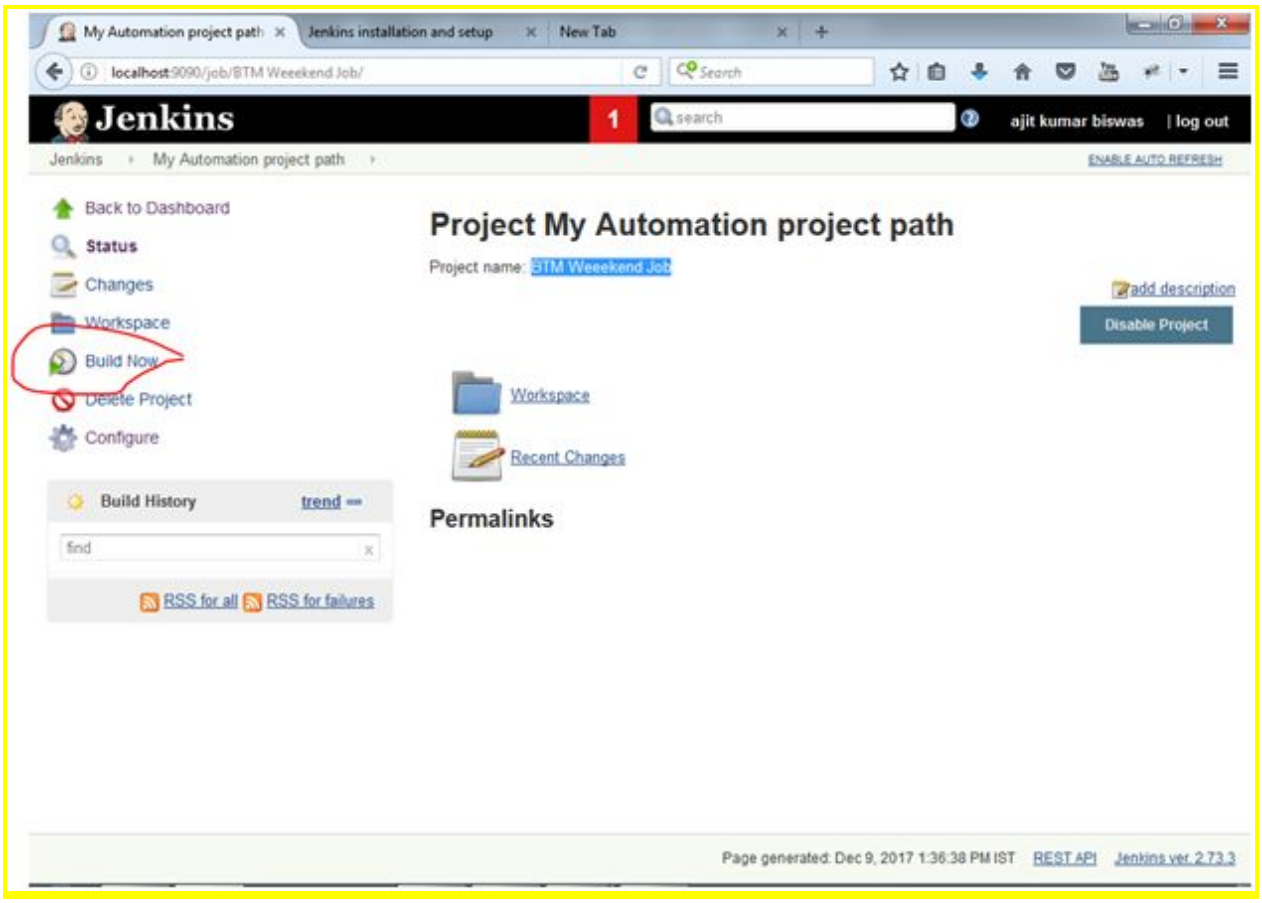


Select the same job and click on configure..

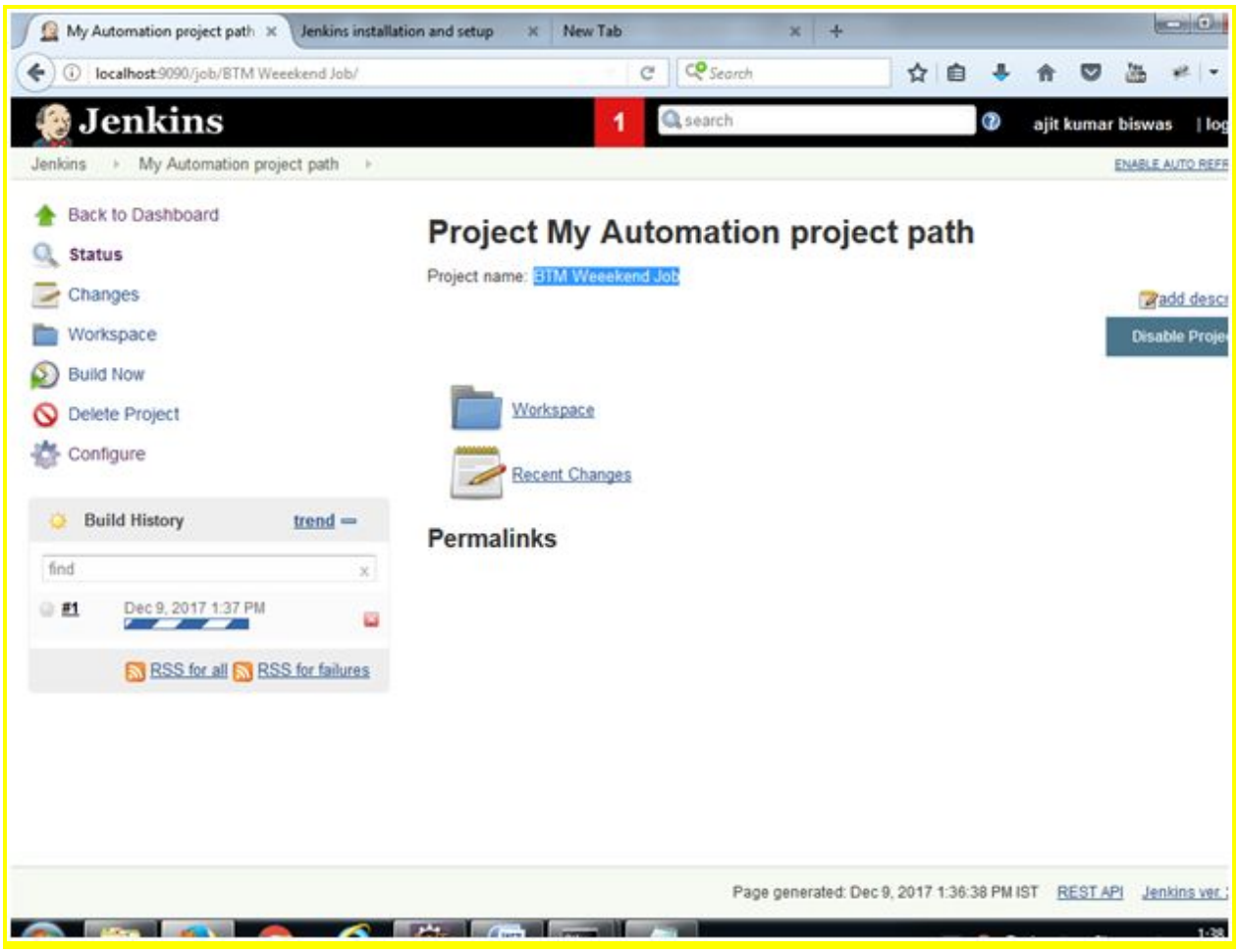
Select below and save



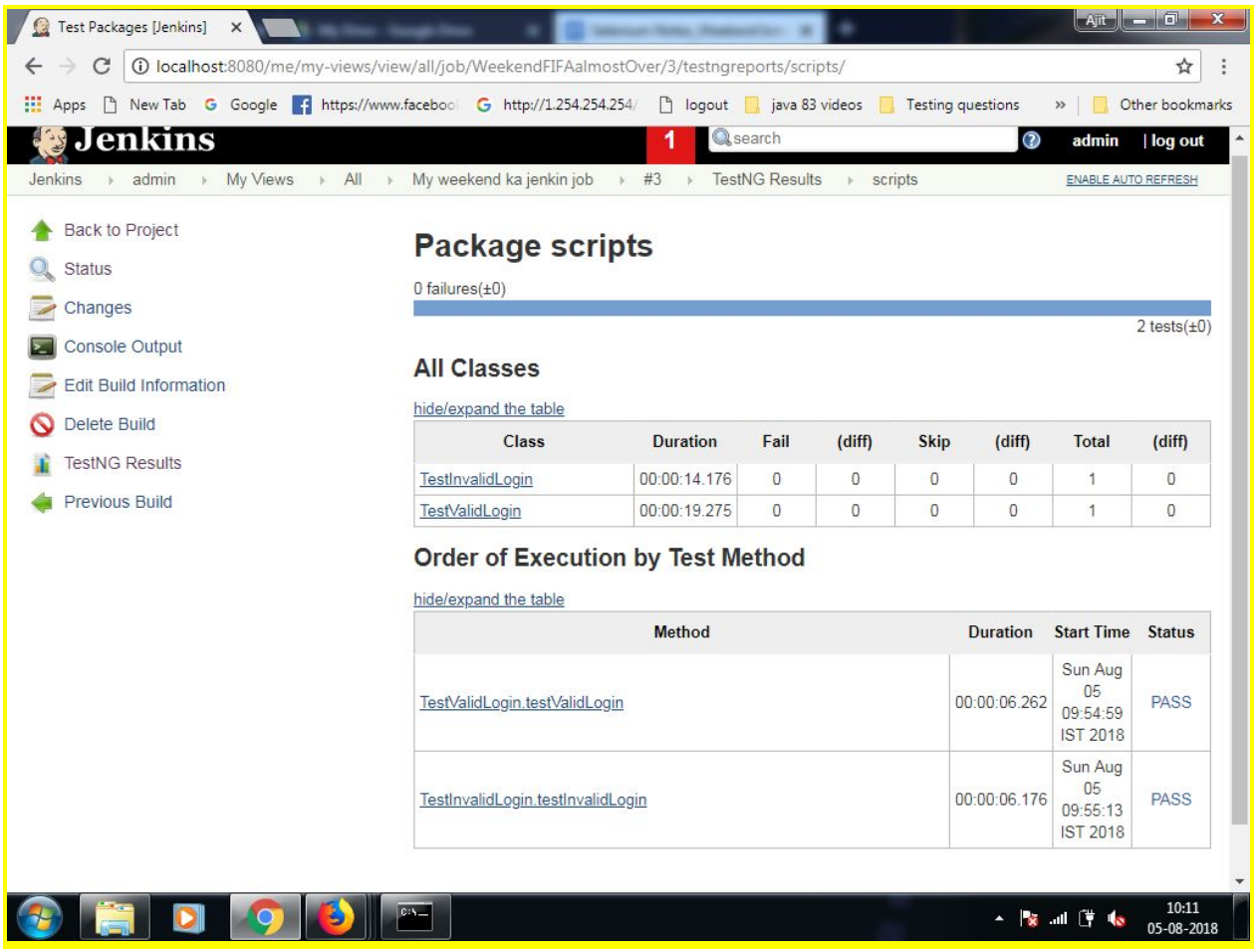
Click on Build now



Job is running and in progress as shown below.



it will generate the report in Testng format as shown below.

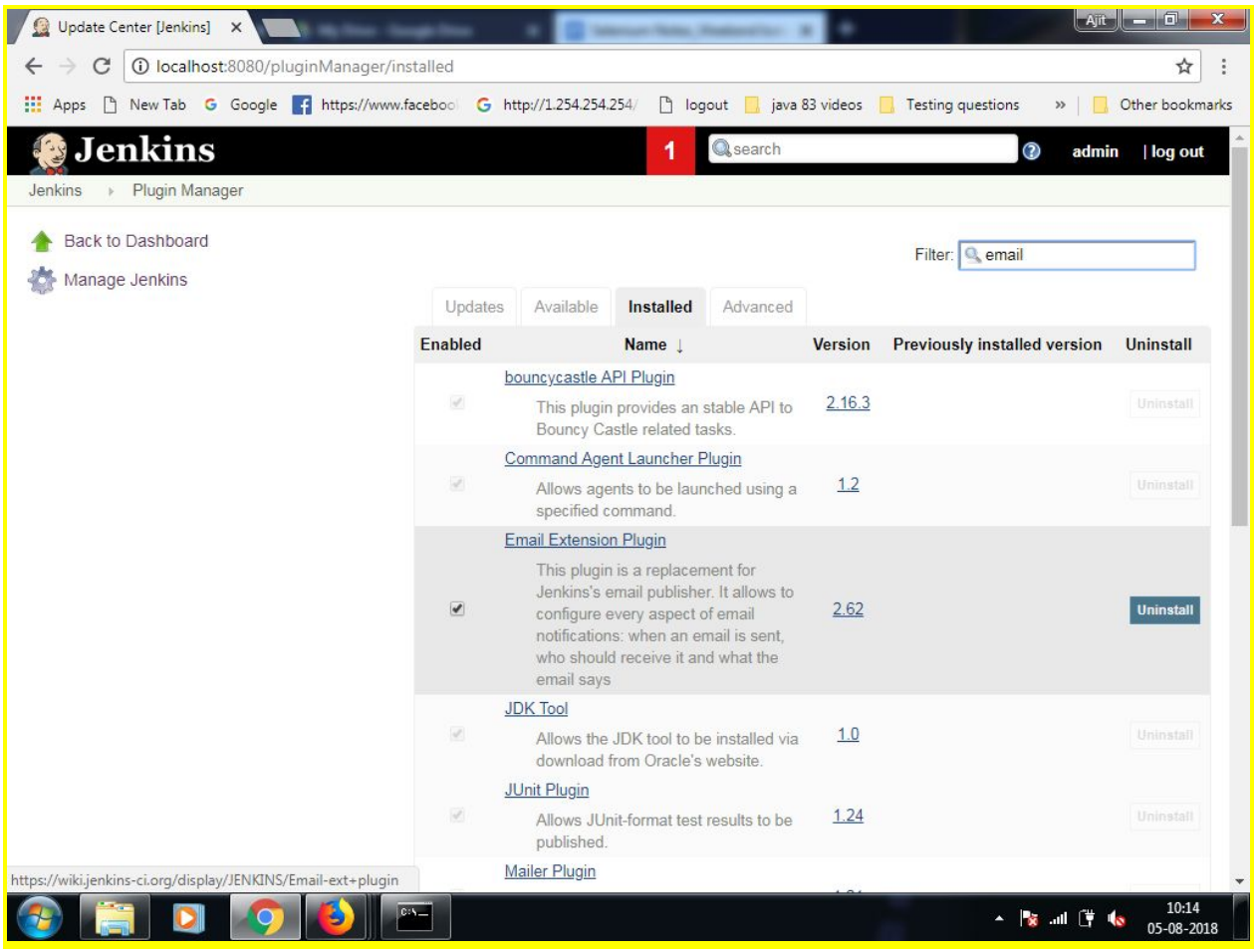


Next..

How do you send testng report via email ?

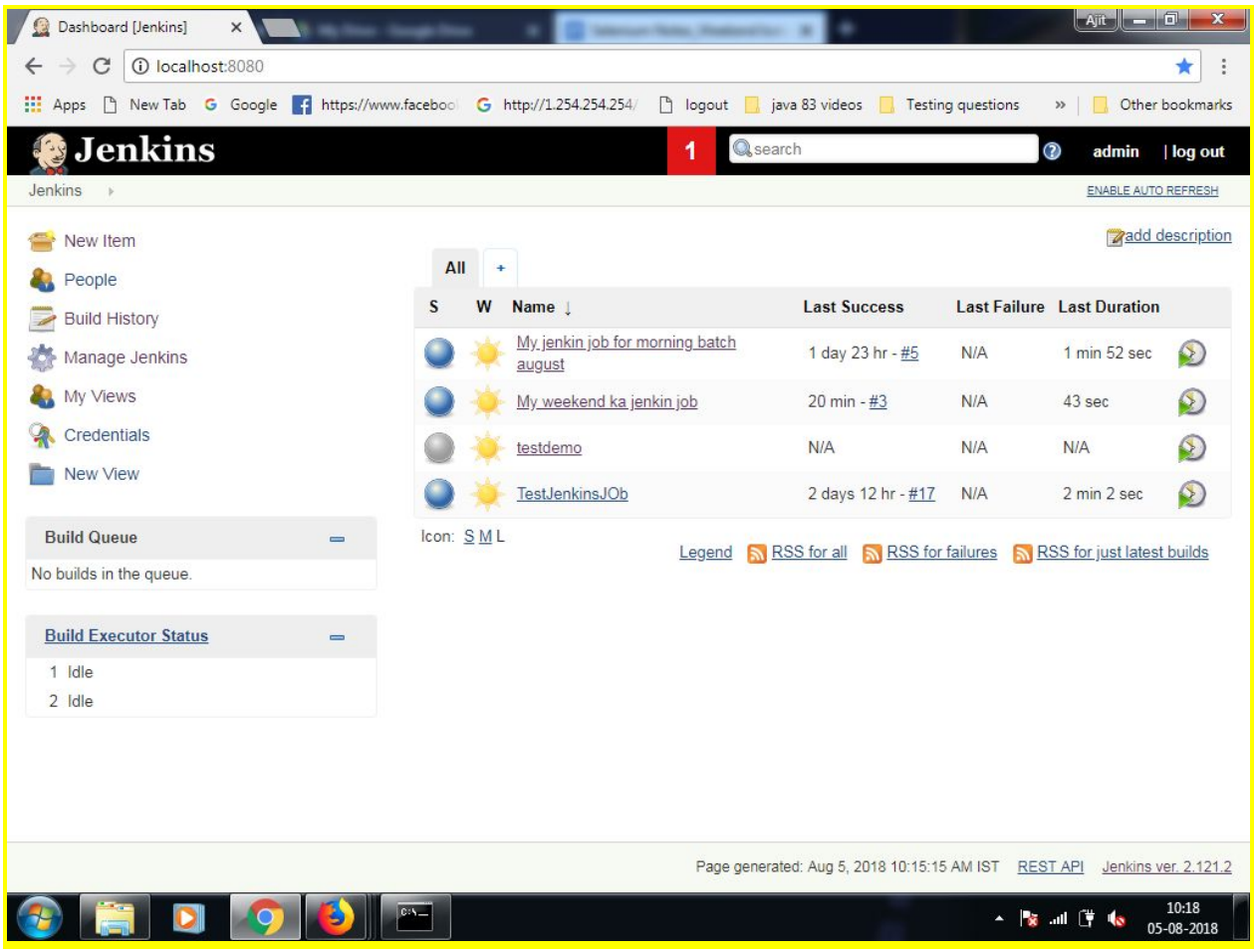
Step 1 : Install this plugin : Email Extension Plugin

Since the plugin is already installed, you can see the plugin under INSTALLED tab as shown below.

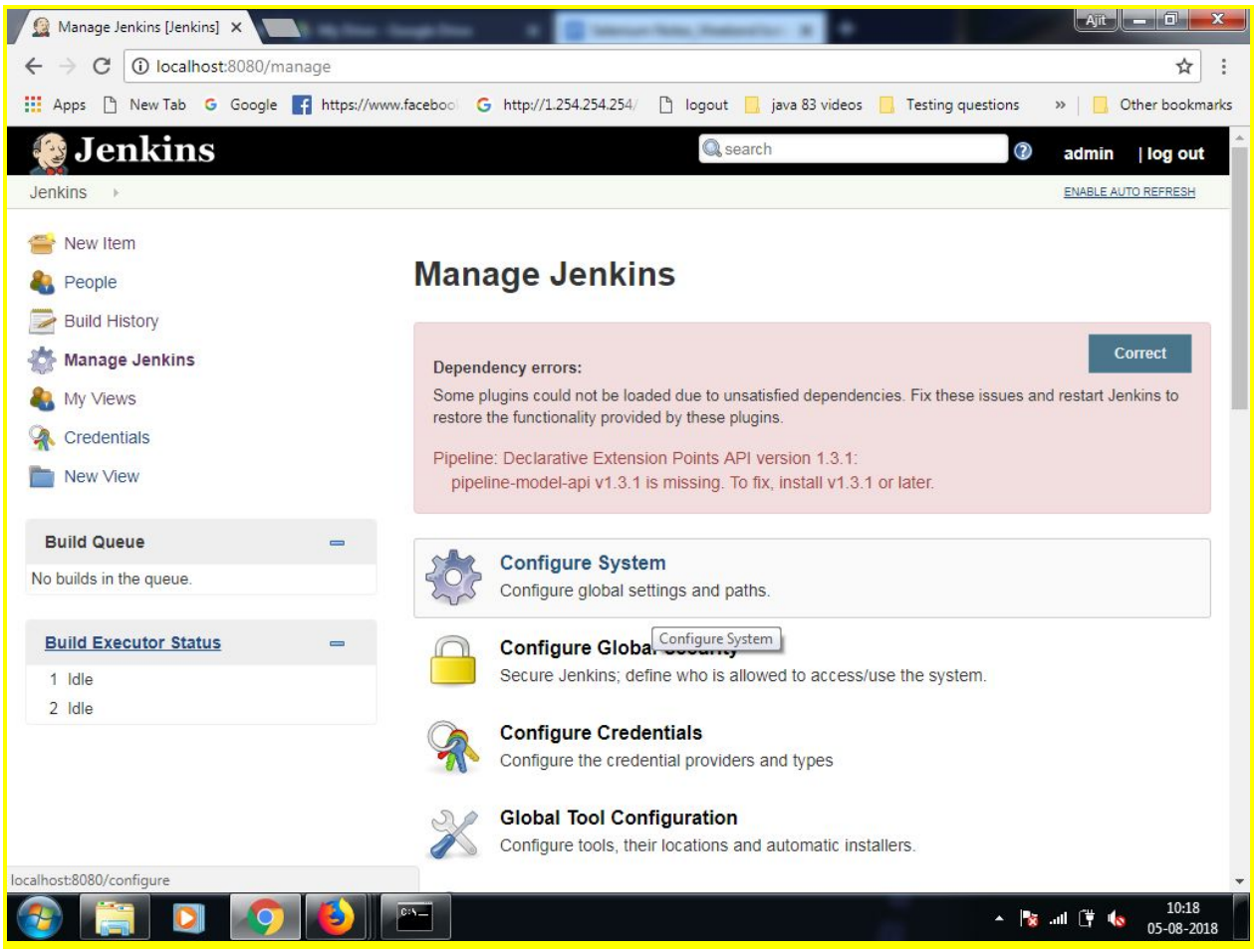


Step 2 : Add gmail server information to jenkins.

click on Manage Jenkins

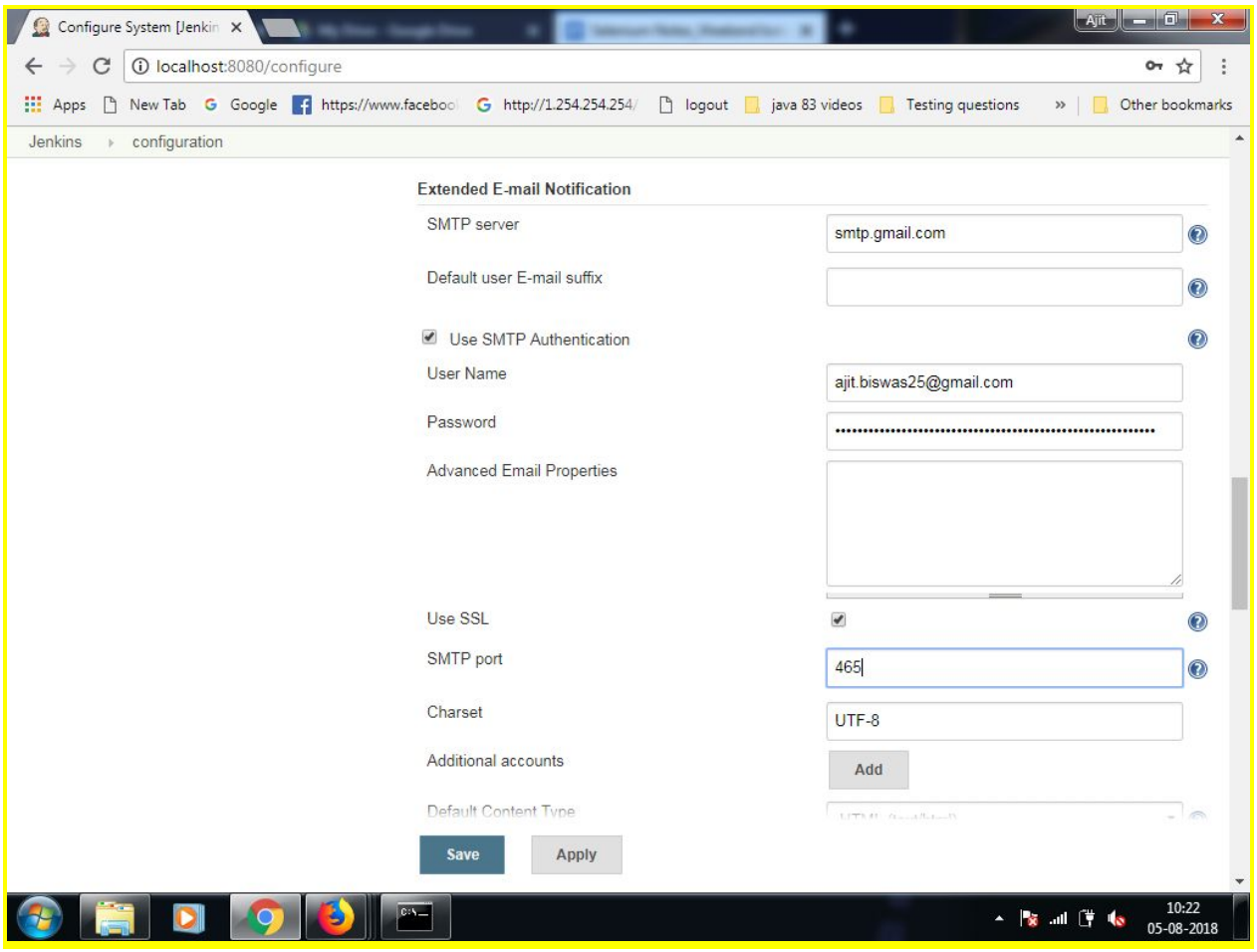


CLick on CONFIGURE SYSTEM



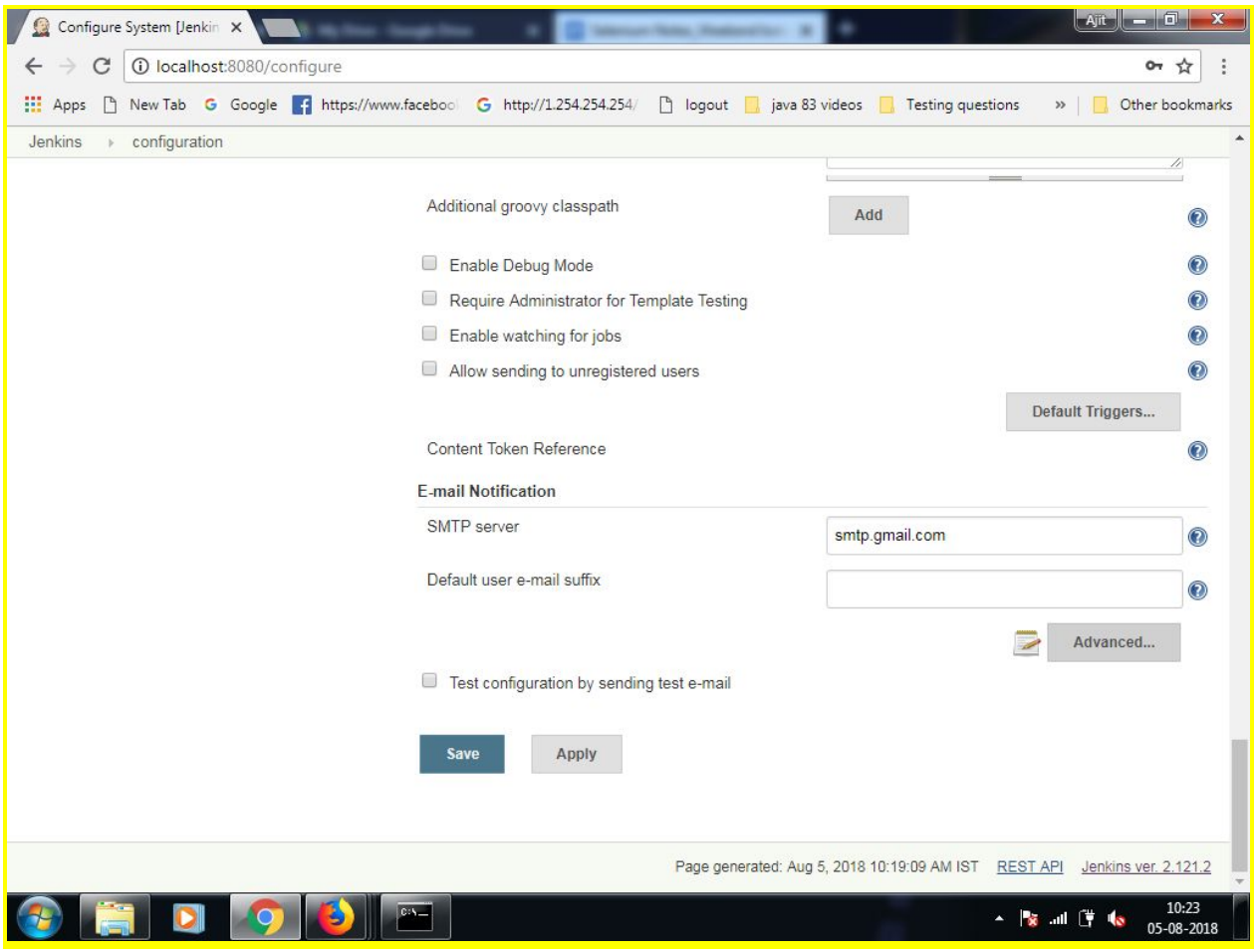
Step 3:

Add the below details under Extended Email notification as shown below.



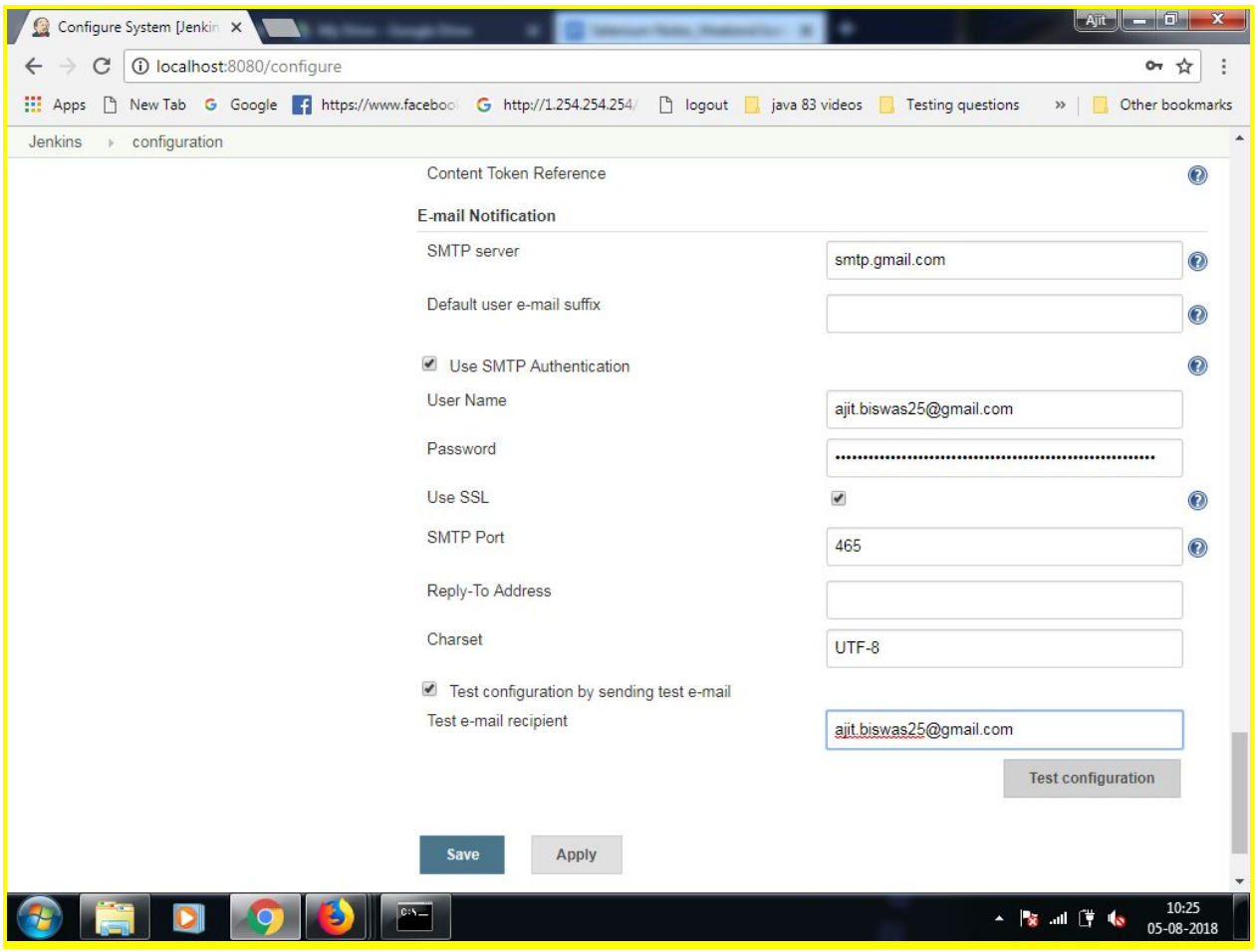
Step 4 :

Under EMAIL NOTIFICATION, click on ADVANCE button.



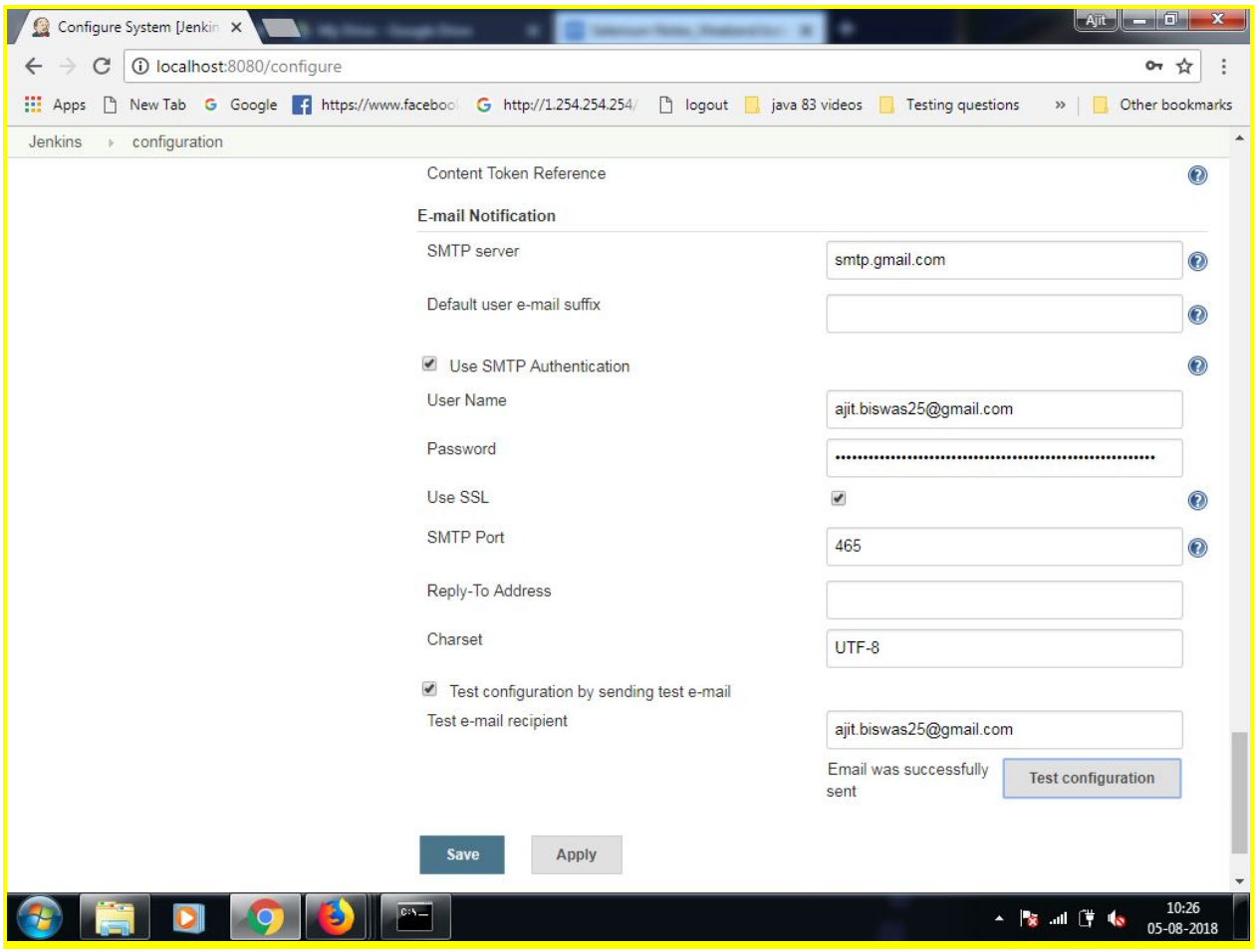
Fill in all the below details

and click on TEST CONFIGURATION



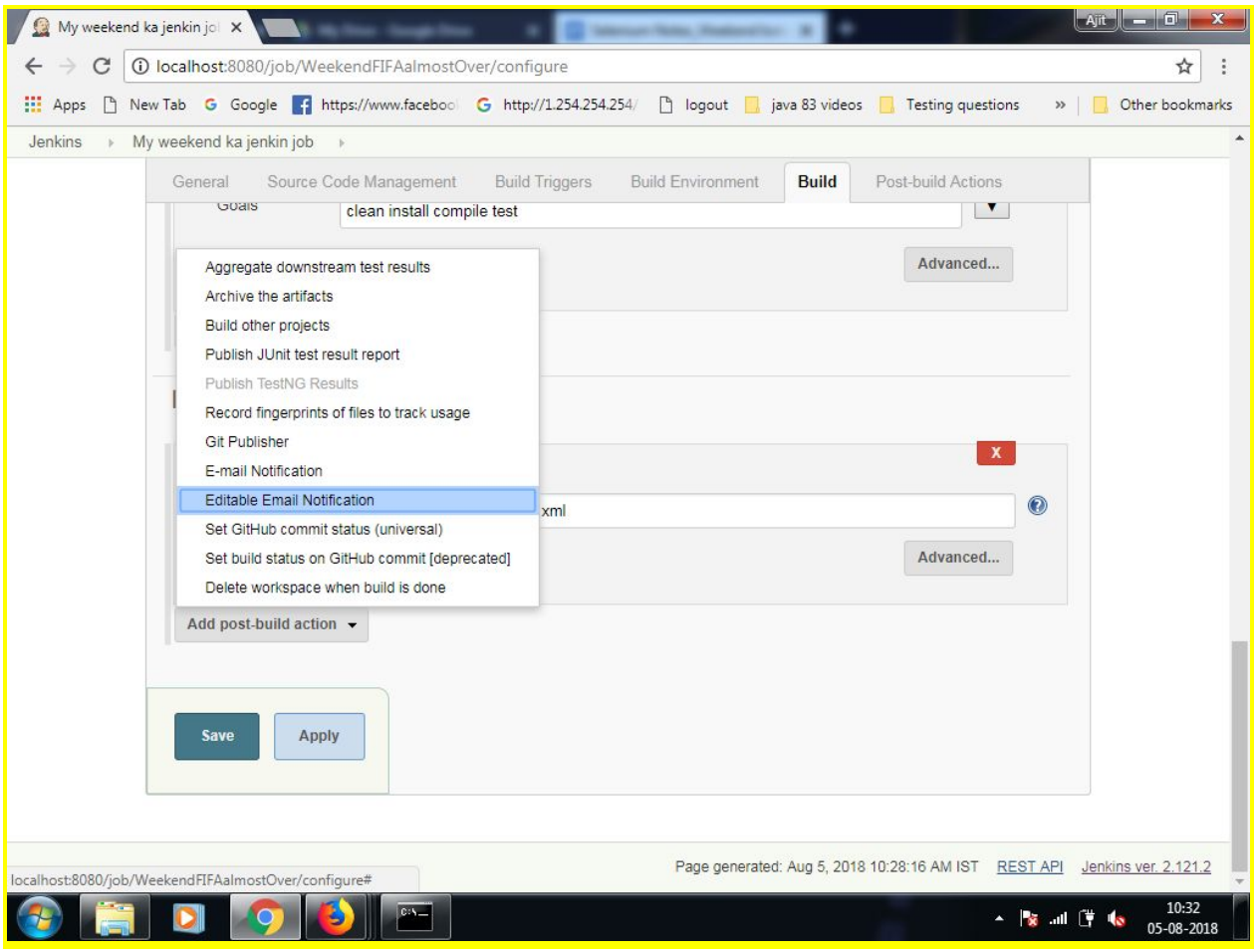
Check that the email is successfully sent out to the gmail inbox.

CLick on SAVE button.

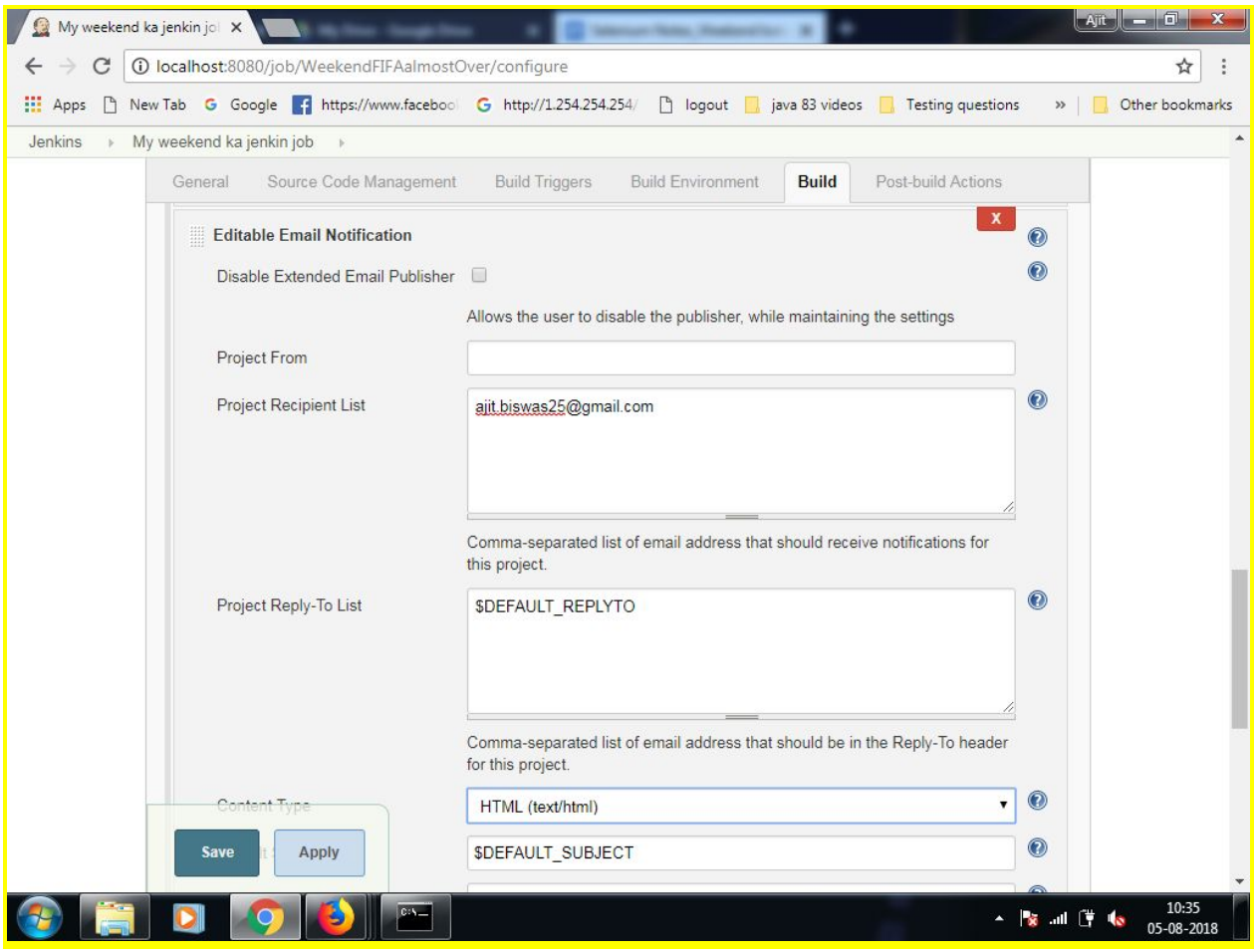


Step 5 : Now go the project related job, CLICK on CONFIGURE and add the mail recipients as shown below.

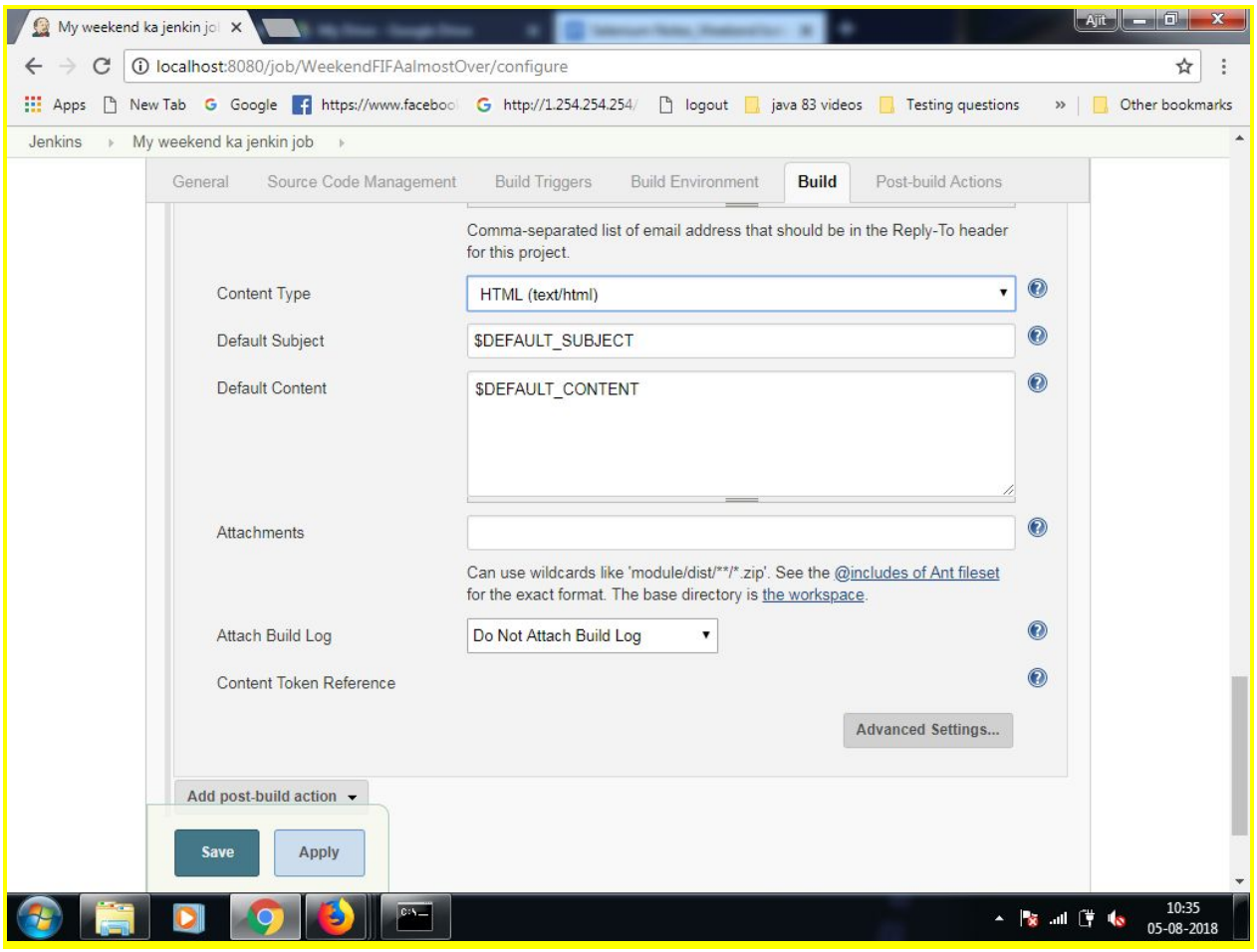
Navigate to POst Build Actions and Select Editable Email Notification -- This is used to send the testng execution status report in the mail body once the build is executed and finished successfully.



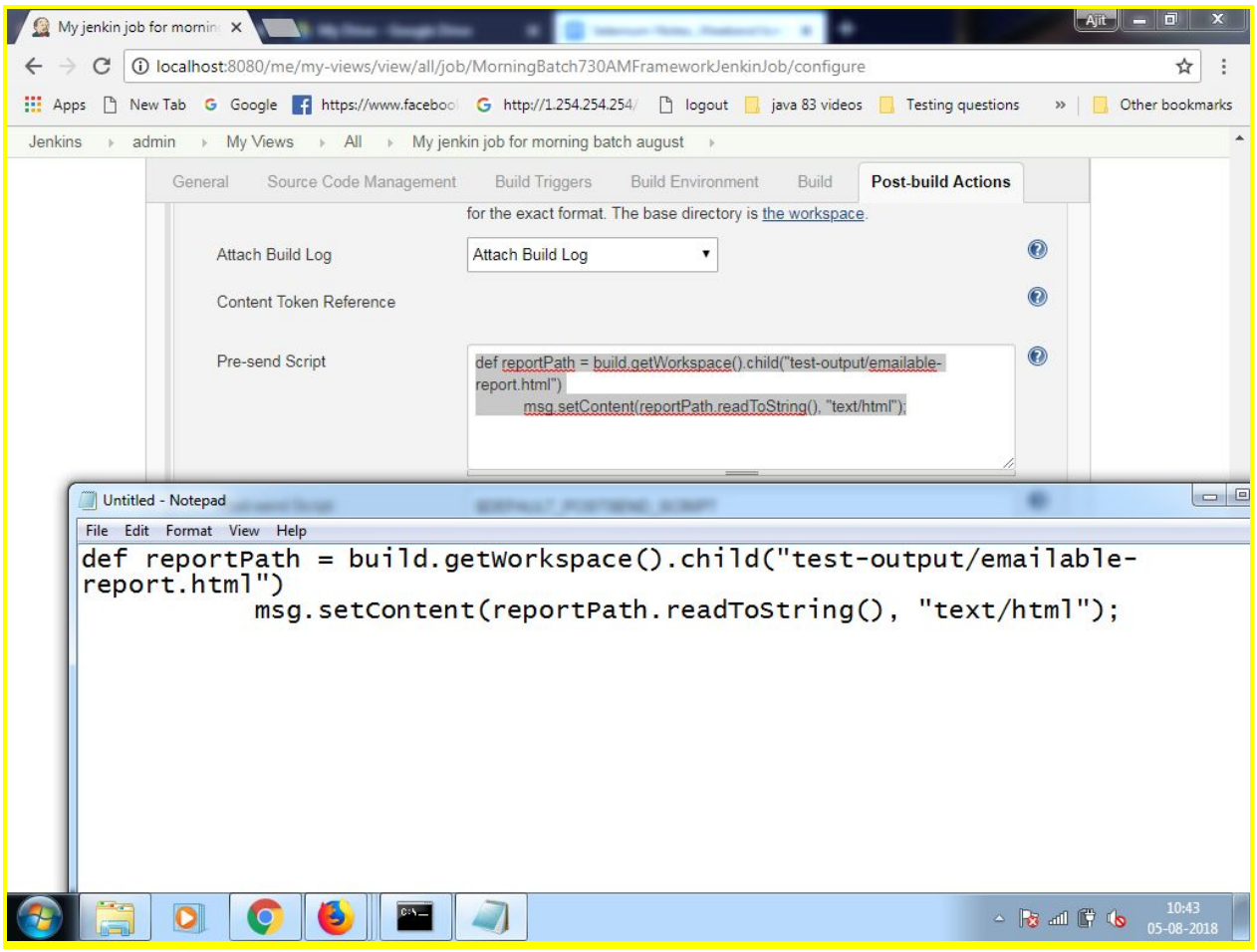
Add the below information



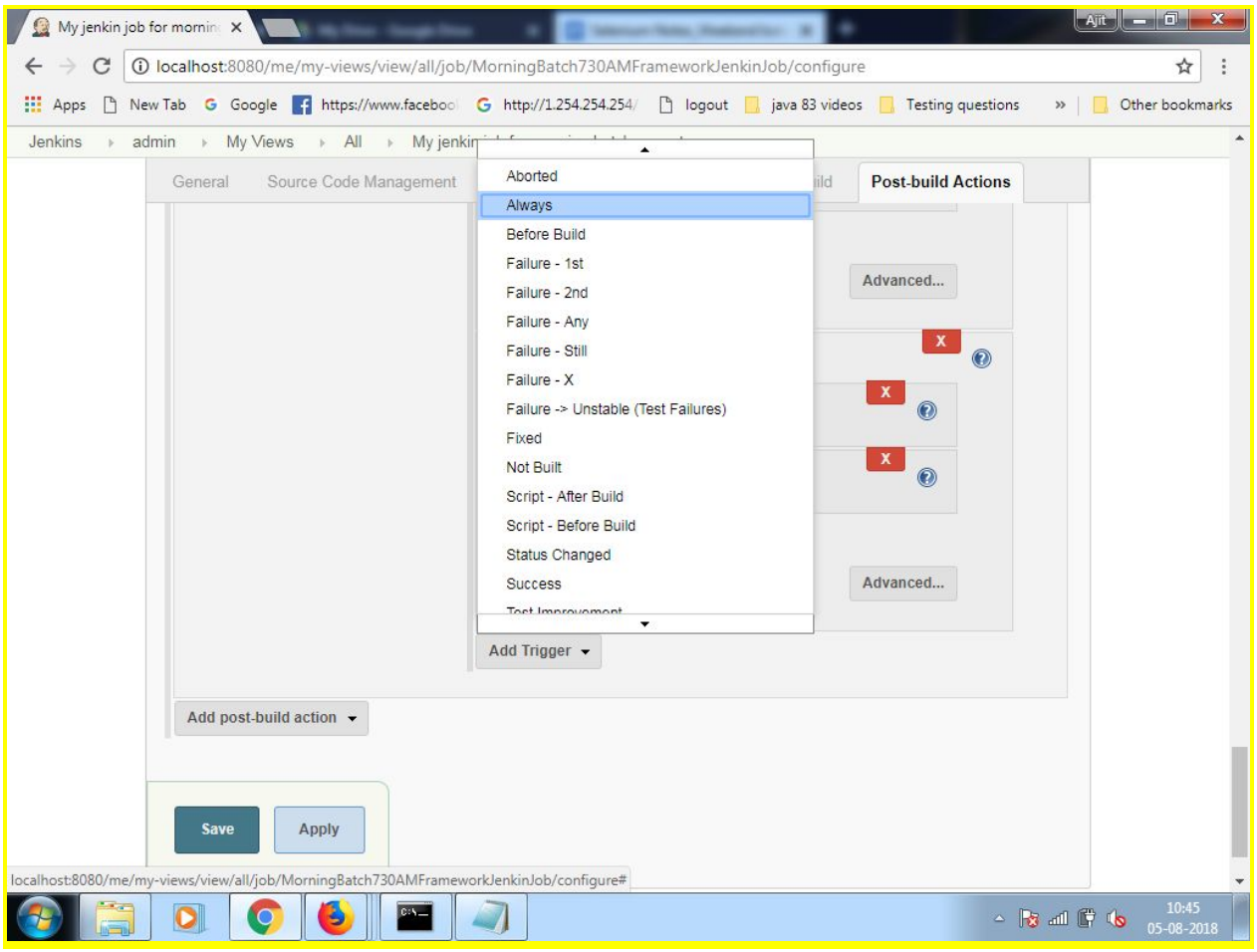
select the CONTENT TYPE and click on ADVANCE SETTING



On this page, add the below script under Pre Send script text area.

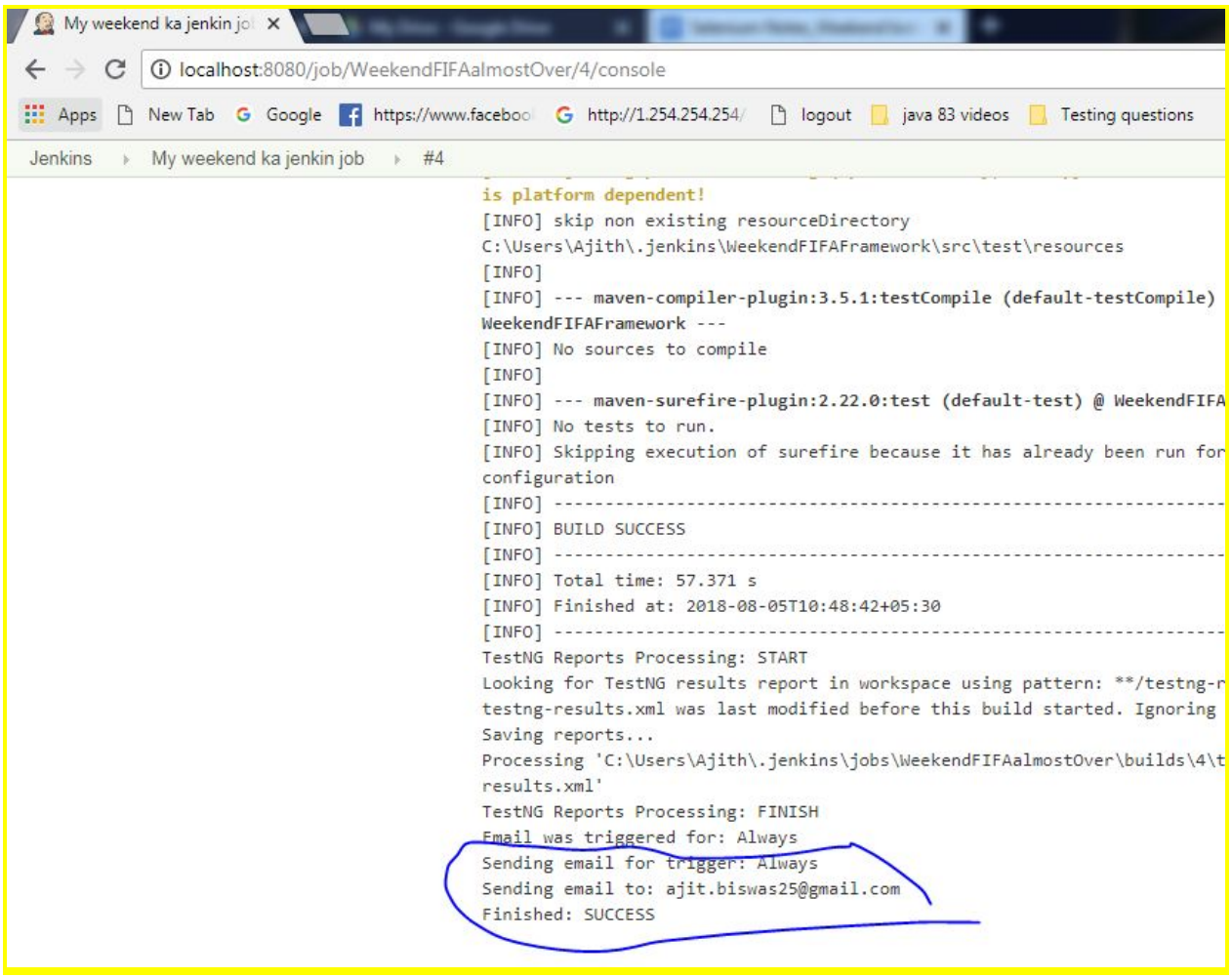


Next step is select Always under Add Triggers drop down

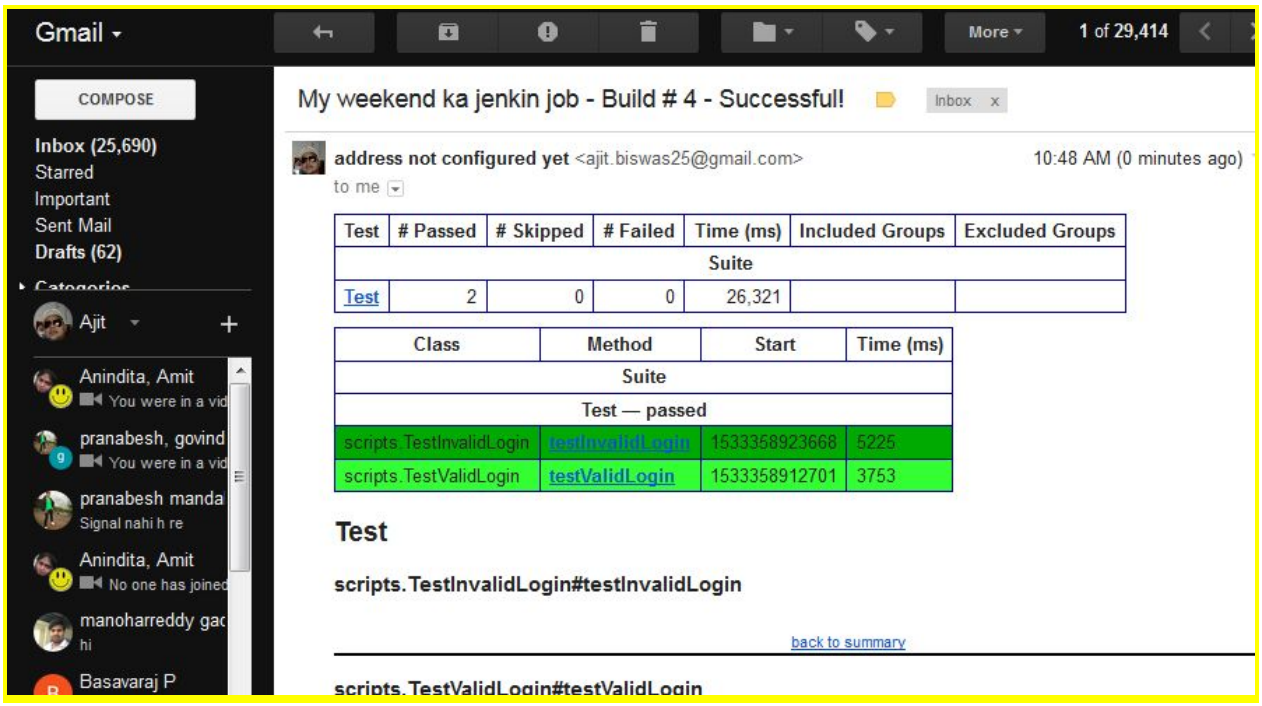


Once the above configuration is done, then build the project by clicking on Build now

check the build execution status as shown below.



Mail recived successfuly in gmail inbox..



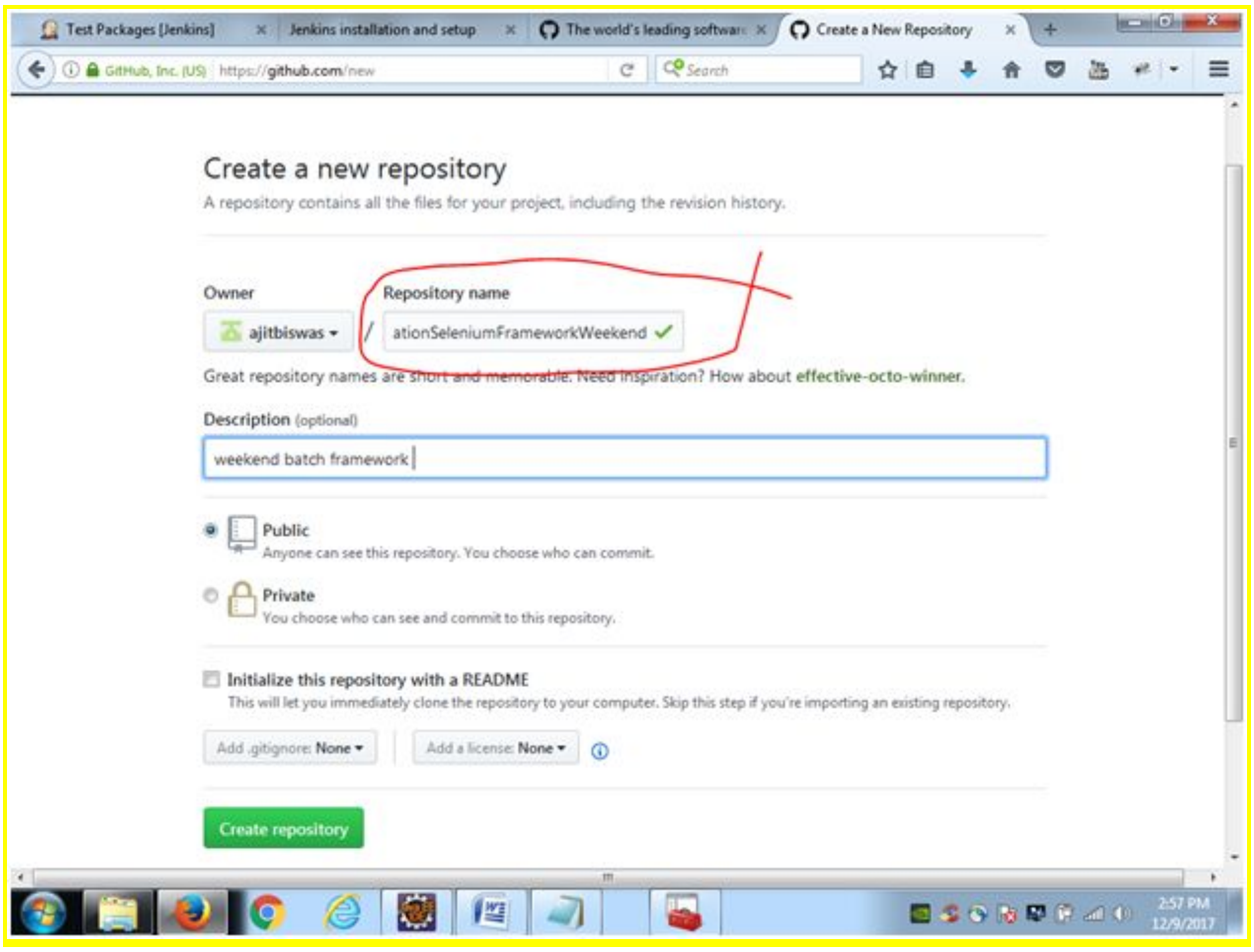
GITHUB SETUP

Go to

www.github.com

register and sign in

click on New Repository

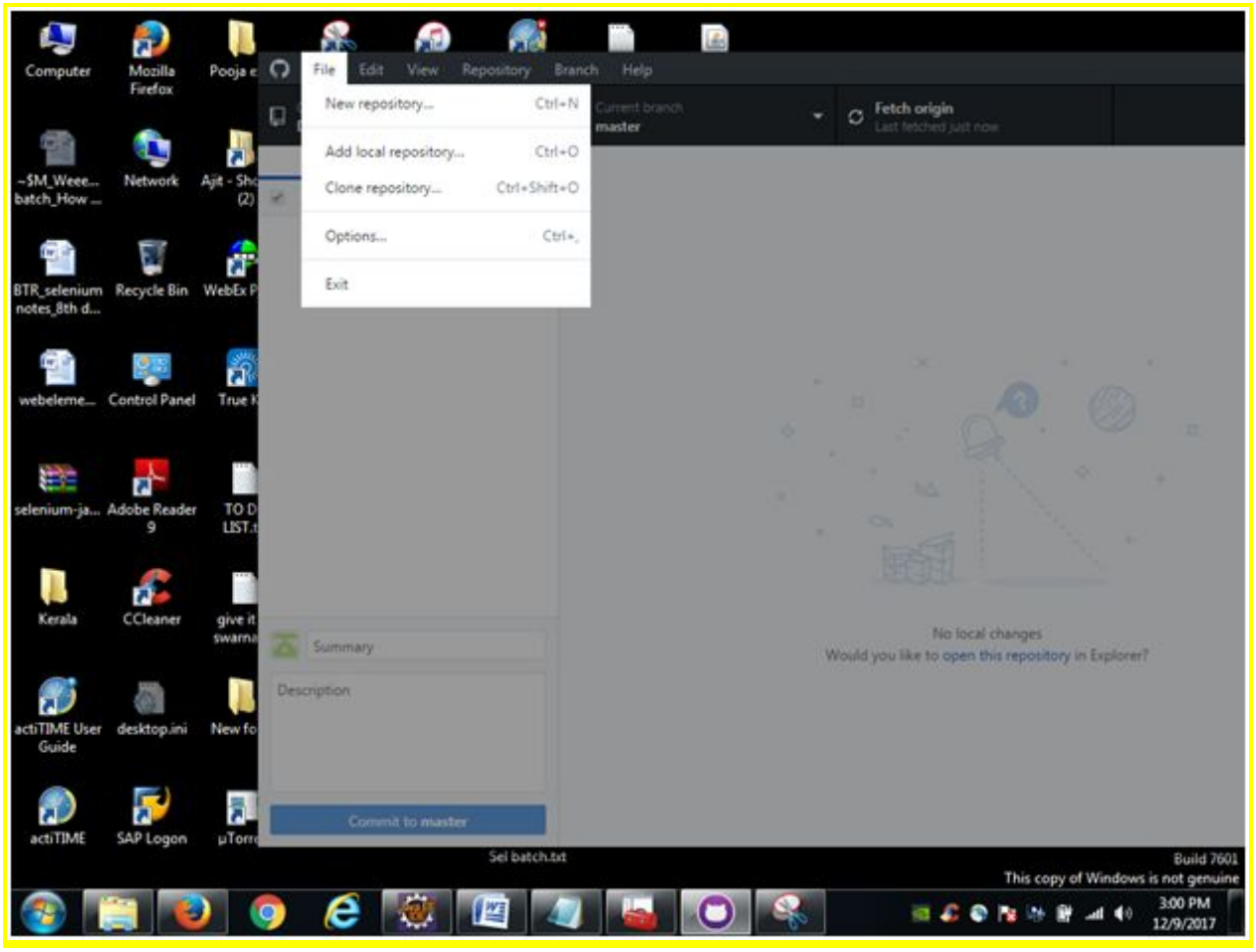


Now, download github desktop from the following url :

https://desktop.github.com/

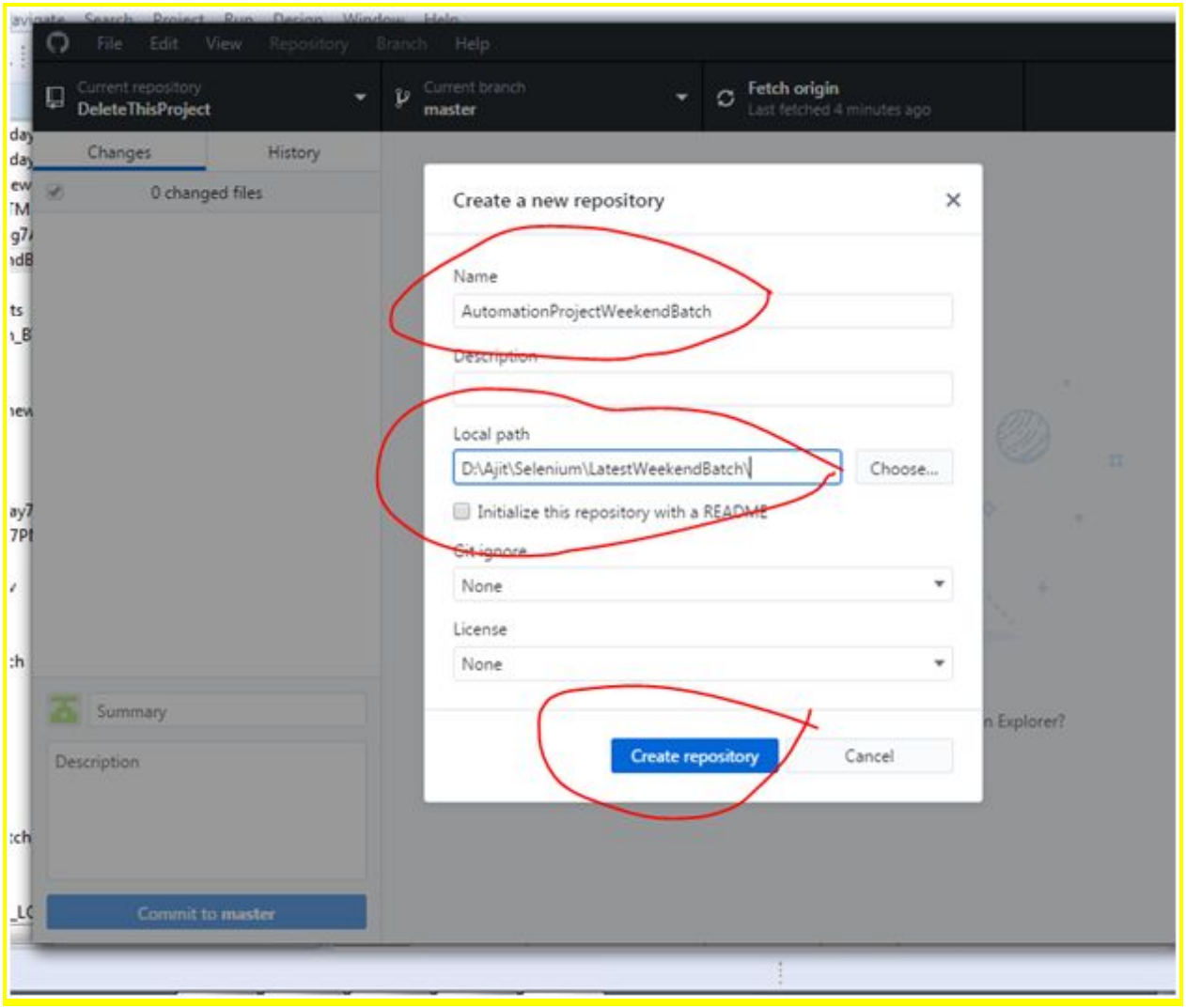
Now launch the github desktop.exe

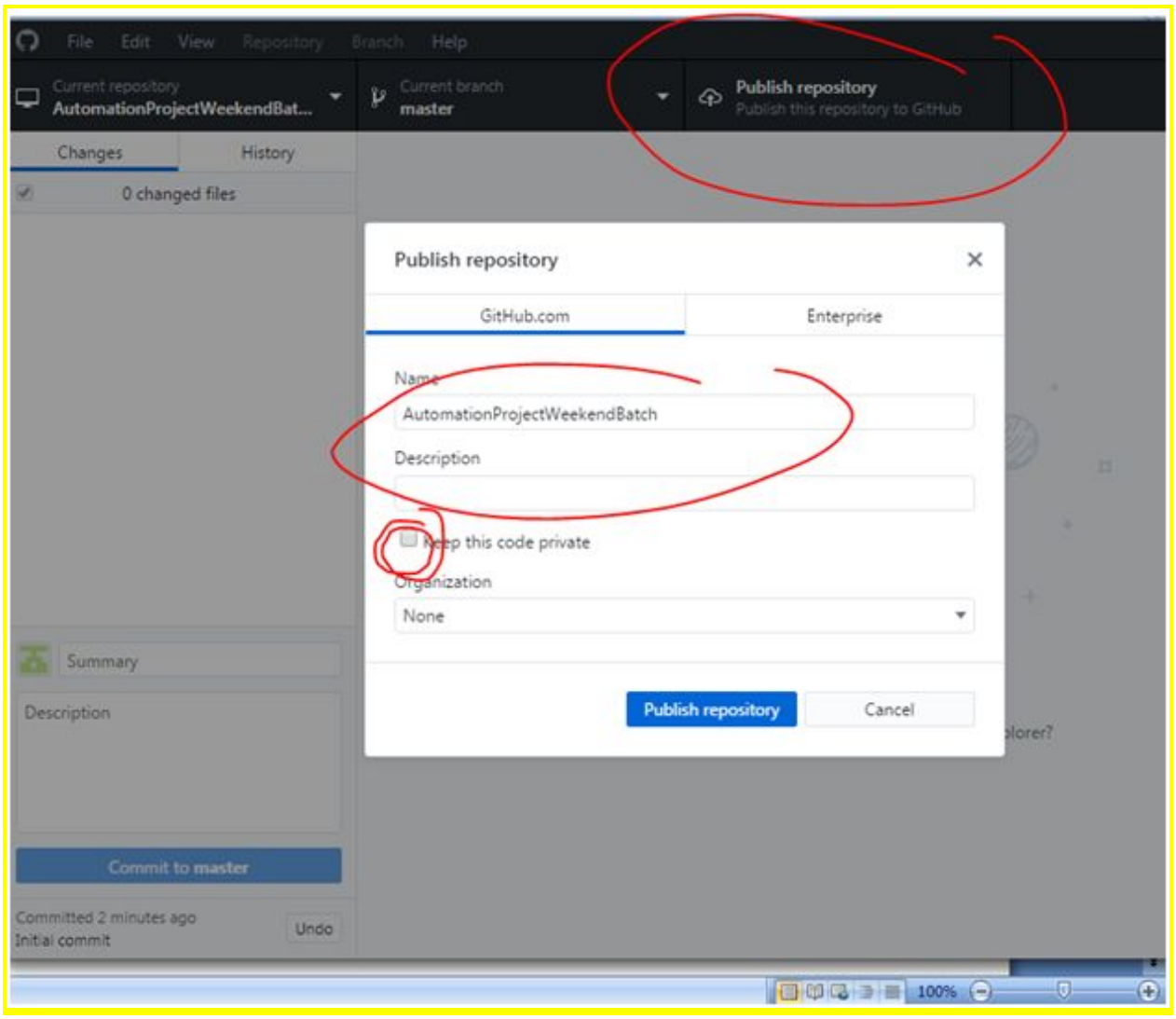
File – New Repository



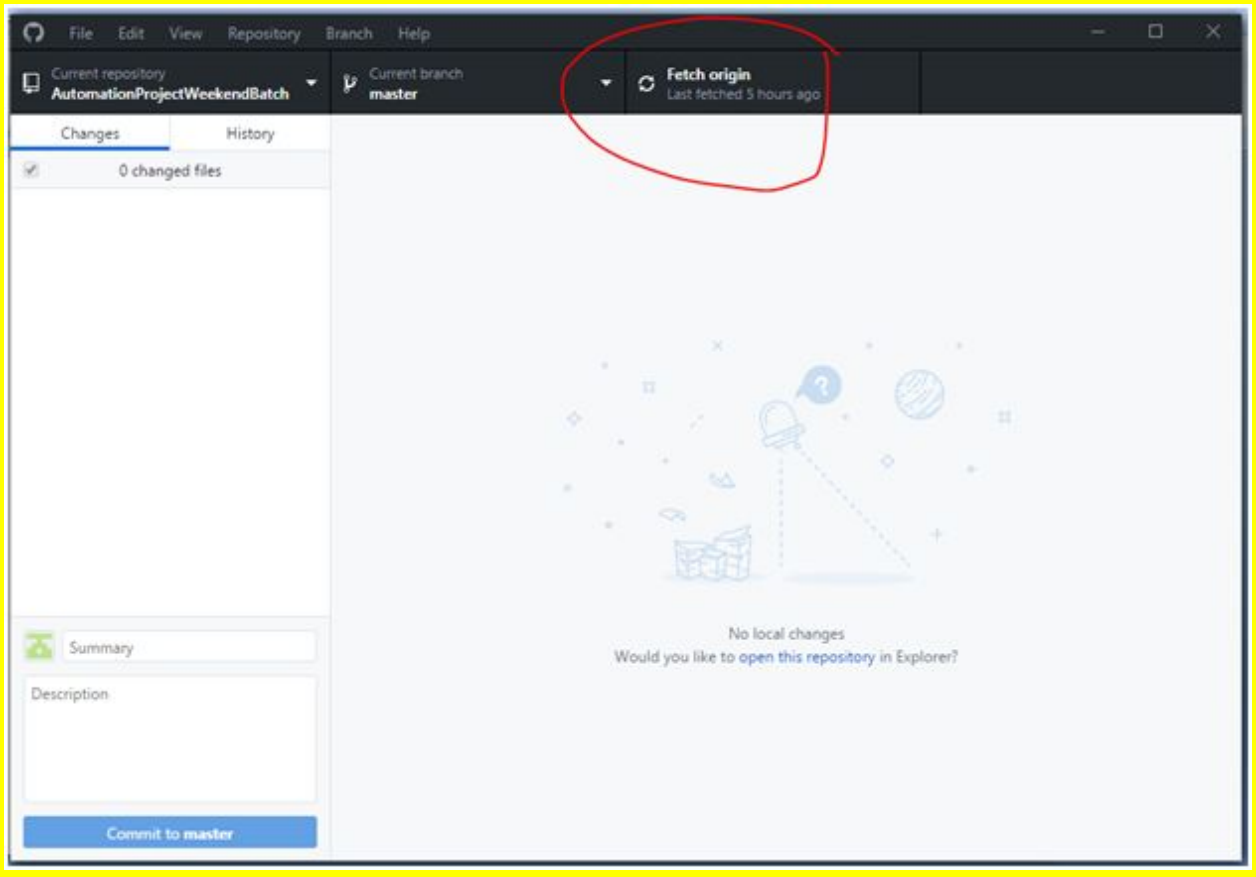
Enter project name which we want to upload under NAME text box

And Local path is the actual workspace where in our project is located.

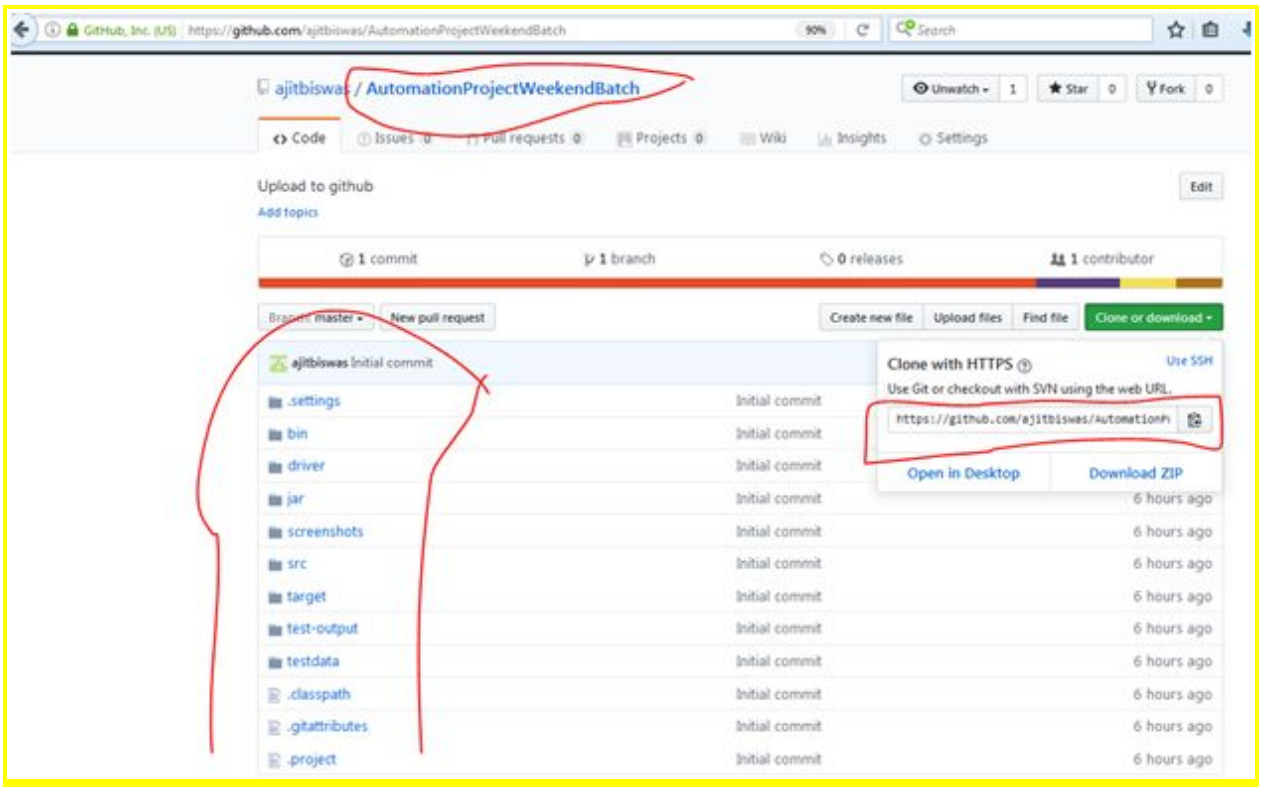

File Edit View Repository Branch Help
Current repository
DeleteThisProject
Current branch
master
Fetch origin
Last fetched 4 minutes ago
Changes
History
0 changed files
Summary
Description
Commit to master
Create a new repository
Name
AutomationProjectWeekendBatch
Description
Local path
D:\Ajit\Selenium\LatestWeekendBatch\
Choose...
Initialize this repository with a README
Git ignore
None
License
None
Create repository
Cancel



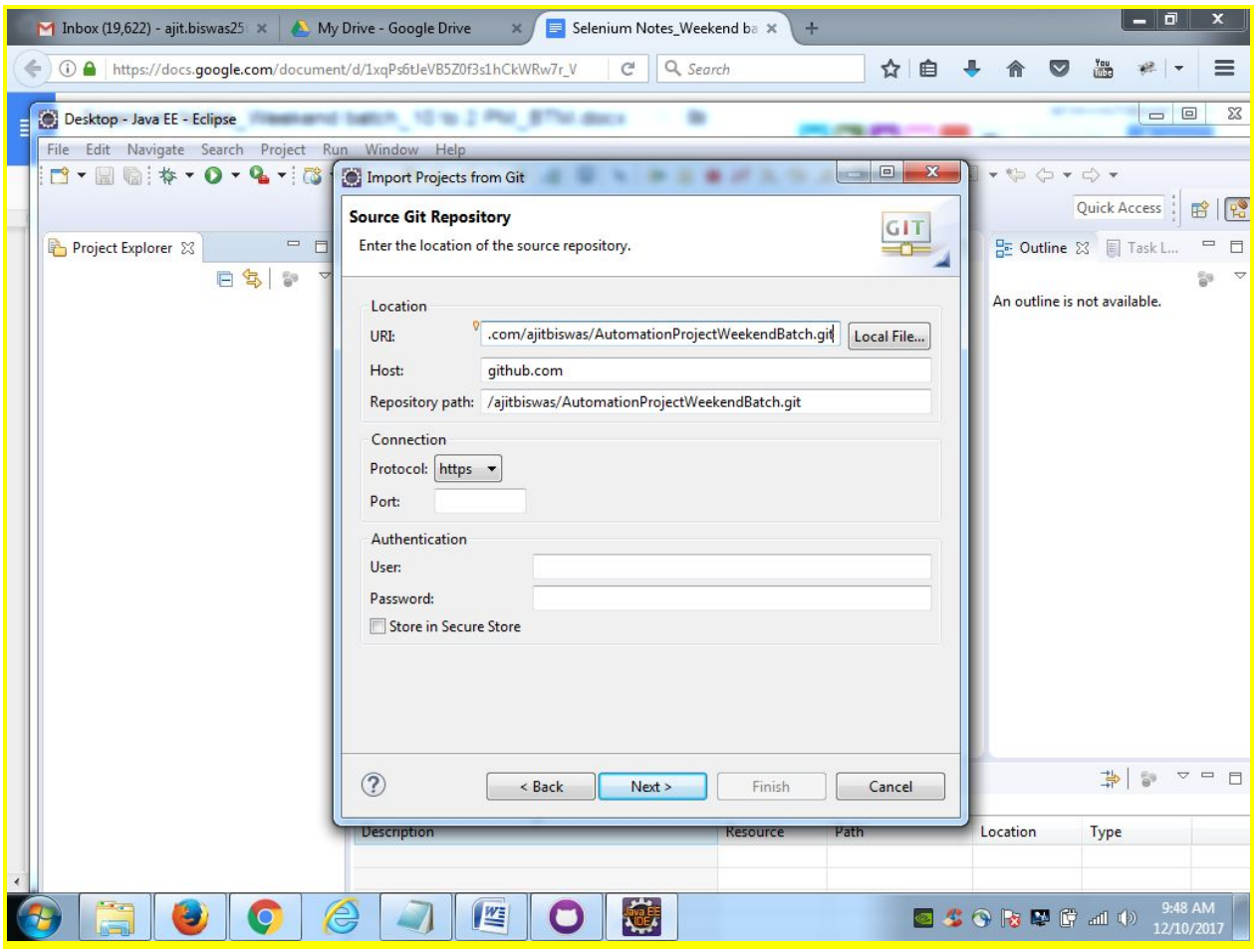
Once it is uploaded successfully, u will get something like this as shown below.



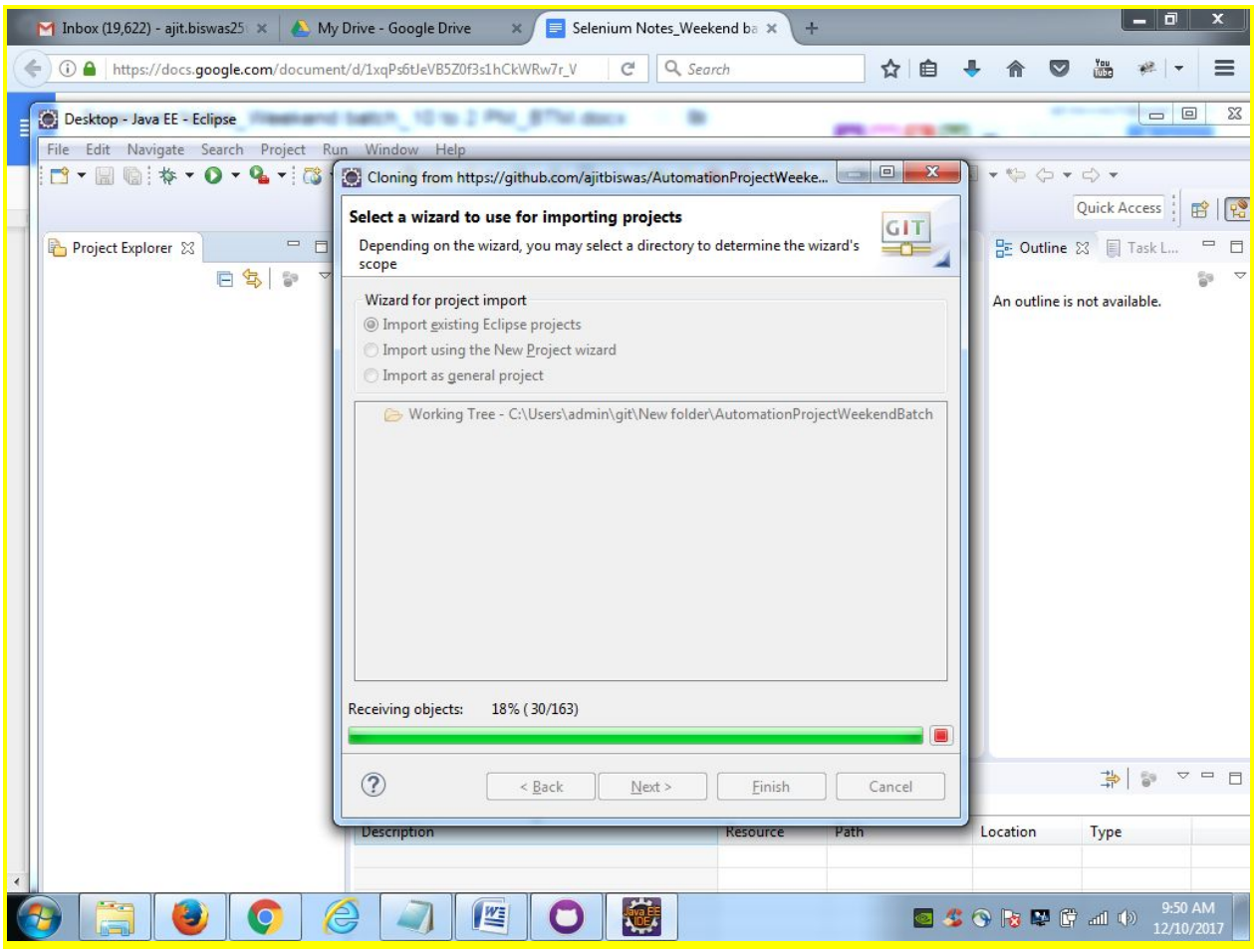
Now, go to the github , you will see that project is successfully uploaded to the central as shown below



Copy the URI above and import the project in Elcipse.

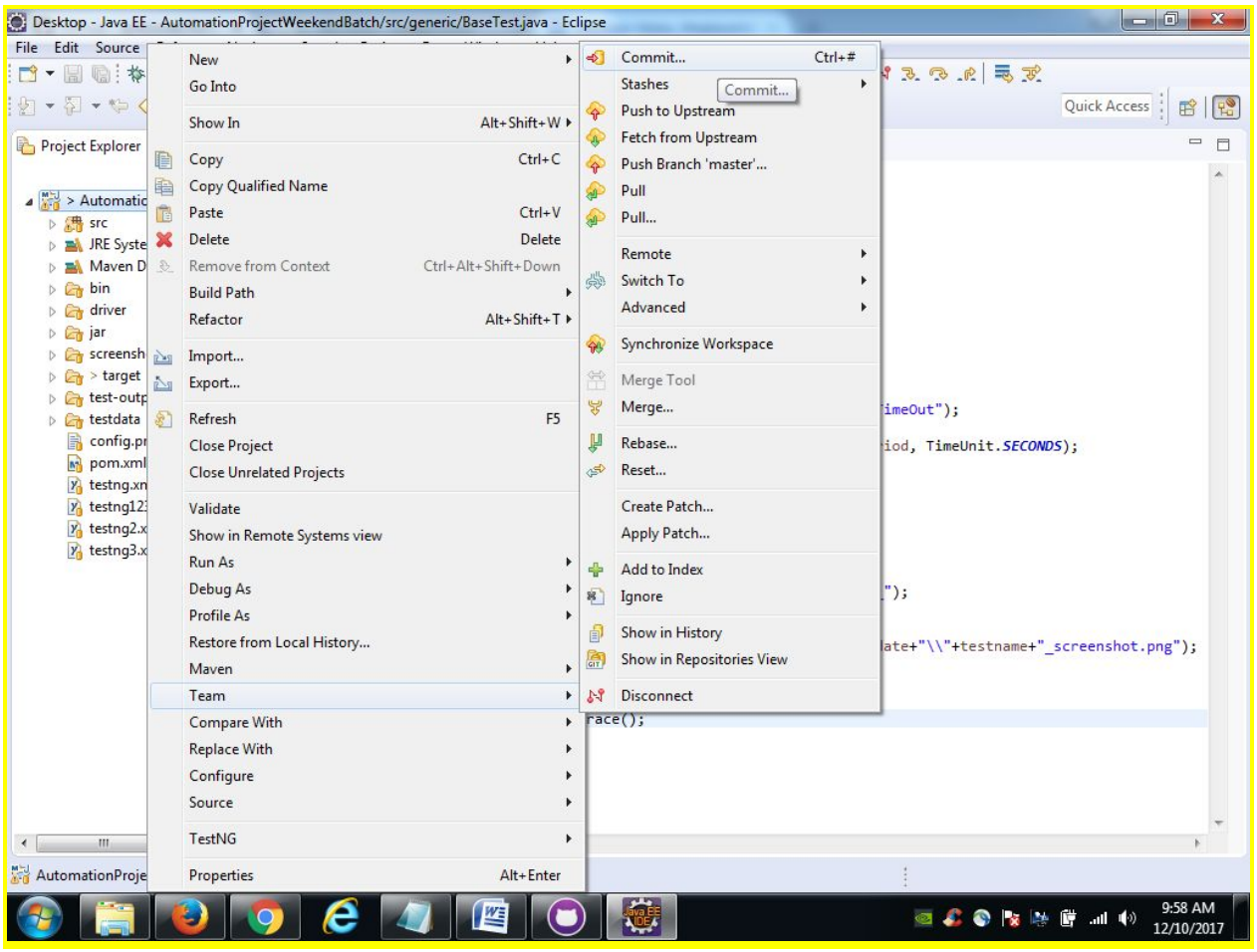


It will start downloading the project from GIthub to the local system in eclipse.

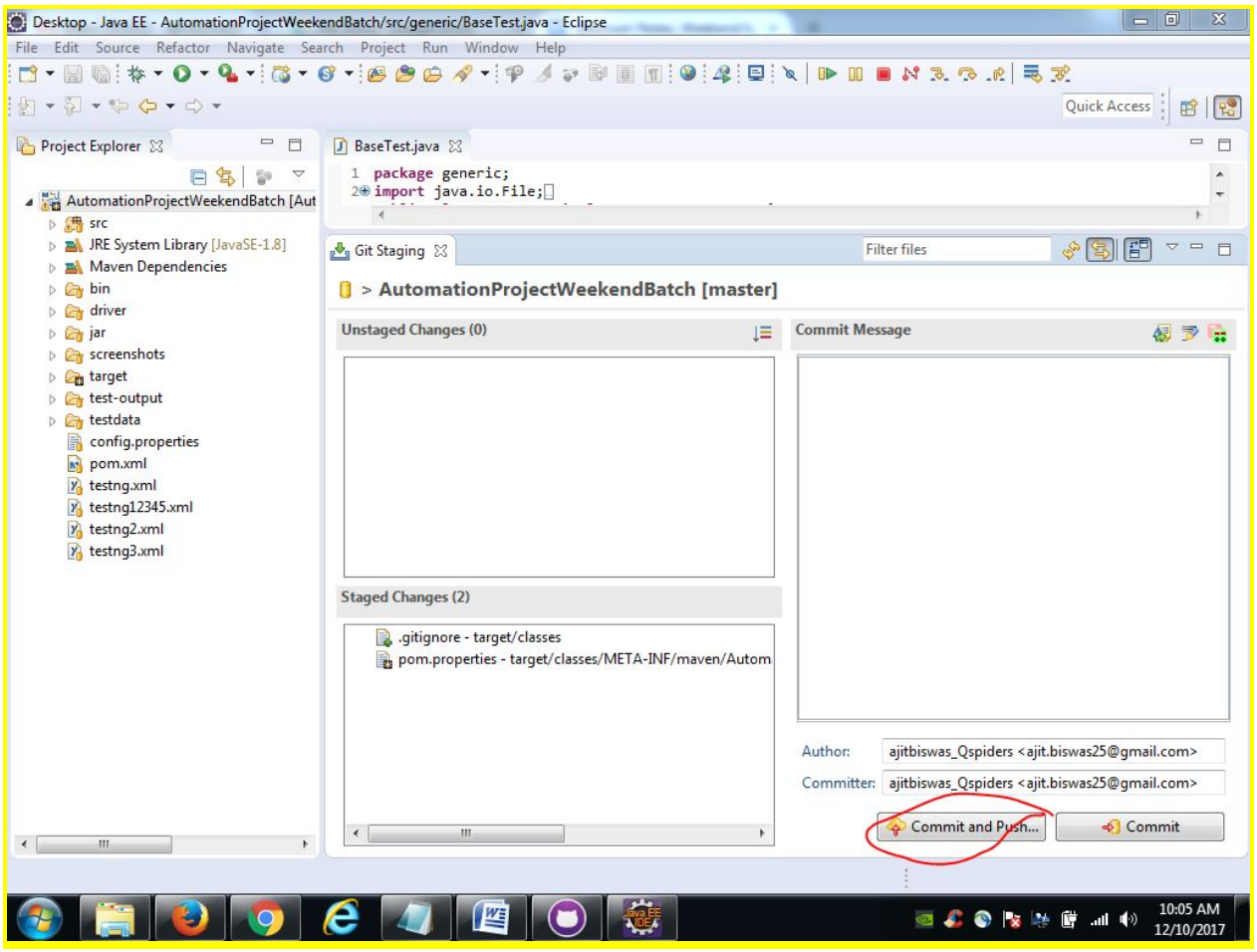


Once imported, we will do some changes and we will upload it back to github by using below navigation.

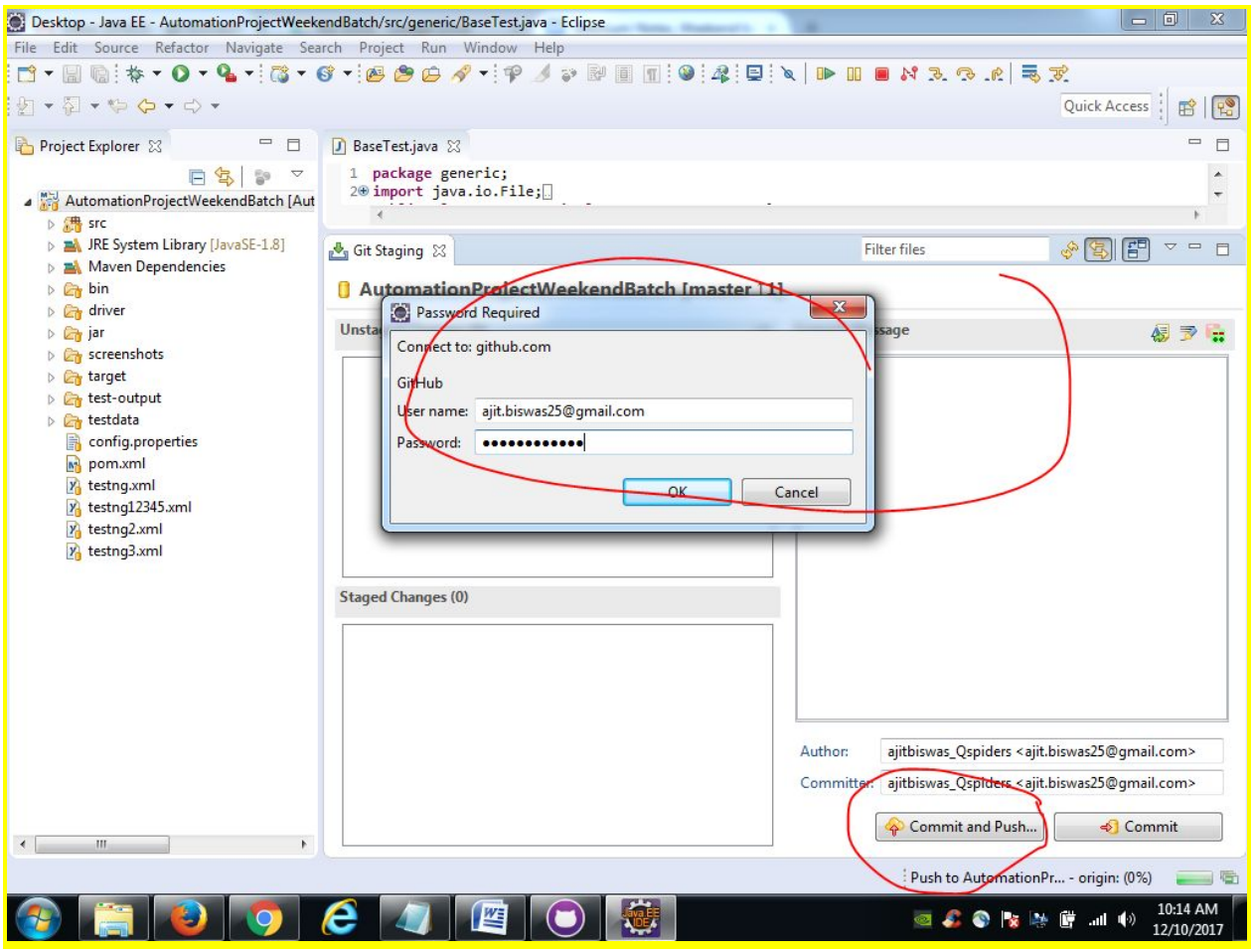
Right click on the project -- TEAM -- COMMIT



now commit and push



Now checkout the latest code from Github and get it in your local system.



LOG4J :

Log4j

1. Add these 2 dependencies to pom.xml

```
<dependency>
  <groupId>org.apache.logging.log4j</groupId>
  <artifactId>log4j-api</artifactId>
  <version>2.9.1</version>
```

```
</dependency>

<dependency>

   <groupId>org.apache.logging.log4j</groupId>

   <artifactId>log4j-core</artifactId>

   <version>2.9.1</version>

</dependency>
```

2. Create testng class and create an instance of LOGGER interface

```
static Logger log =
LogManager.getLogger(TestLogin_MySQLDatabaseusingSelenium.class.getName());
```

3. Create a log4j2.xml right under the src folder and do the following setting .

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

<Configuration status="WARN">

<Properties>

<Property name="basePath">./logs</Property>

</Properties>

    <Appenders>

        <RollingFile name="File" fileName="${basePath}/prints.log"
filePattern="${basePath}/prints-%d{yyyy-MM-dd}.log">

        <PatternLayout pattern="%d{HH:mm:ss.SSS} [%t] %-5level %logger{36} - %msg%n"/>

    <SizeBasedTriggeringPolicy size="500" />

        </RollingFile>
```

```
        <Console name="Console" target="SYSTEM_OUT">
    <PatternLayout pattern="%d{HH:mm:ss.SSS} [%t] %-5level %logger{36} - %msg%n"/>
        </Console>
 </Appenders>
 <Loggers>
        <Root level="trace">
        <AppenderRef ref="File"/>
        </Root>
 </Loggers>
</Configuration>
```

Important point : paste this file under the src folder in the project

Create a folder called logs under the projects and execute the script

---

Database connection using Selenium

```
package scripts;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
```

```
import org.apache.logging.log4j.LogManager;

import org.apache.logging.log4j.Logger;

import org.testng.annotations.Test;

import org.testng.asserts.SoftAssert;

import generic.BaseTest;

import generic.Lib;

import pompages.LoginPage;

public class TestLogin_MySQLDatabaseusingSelenium extends BaseTest{

   static Logger log =
LogManager.getLogger(TestLogin_MySQLDatabaseusingSelenium.class.getName());

   @Test

   public void testLogin() throws InterruptedException, SQLException{

         log.debug("Creating an object of LoginPage pom class");

         LoginPage l = new LoginPage(driver);

         log.info("Loginpom object created successfully");

         log.error("object creation failed");

         try {

               Class.forName("com.mysql.jdbc.Driver");

               Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306", "root", "root");

               Statement stmt = con.createStatement();
```

```
            ResultSet rs = stmt.executeQuery("SELECT * FROM `ACTITIME`.`USERS`");
            while (rs.next()) {
                String username = rs.getString(1); // 1 refers to the first column
                String password = rs.getString(2);
                l.setUsername(username);
                l.setPassword(password);
                l.clickLogin();
            }
        } catch (ClassNotFoundException e) {
            log.fatal("Database Connection not established...");
        }
        Thread.sleep(10000);
        String actualtitle = driver.getTitle();
        SoftAssert s = new SoftAssert();
        //s.assertEquals(actualtitle, "actiTIME - Enter Time-Track");
        s.assertAll();
    }
}
```

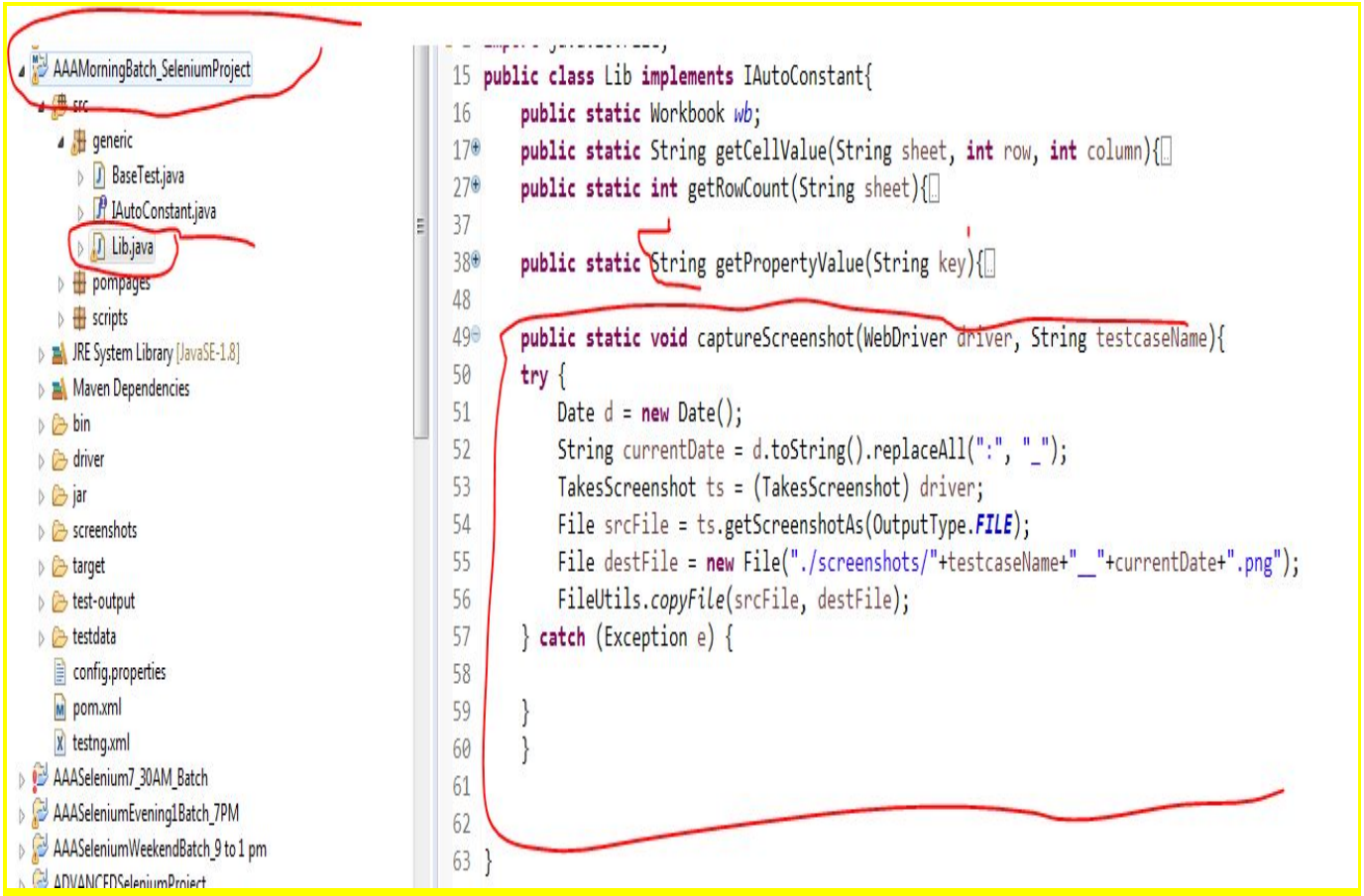
How to take snapshot for failed test cases in selenium ?

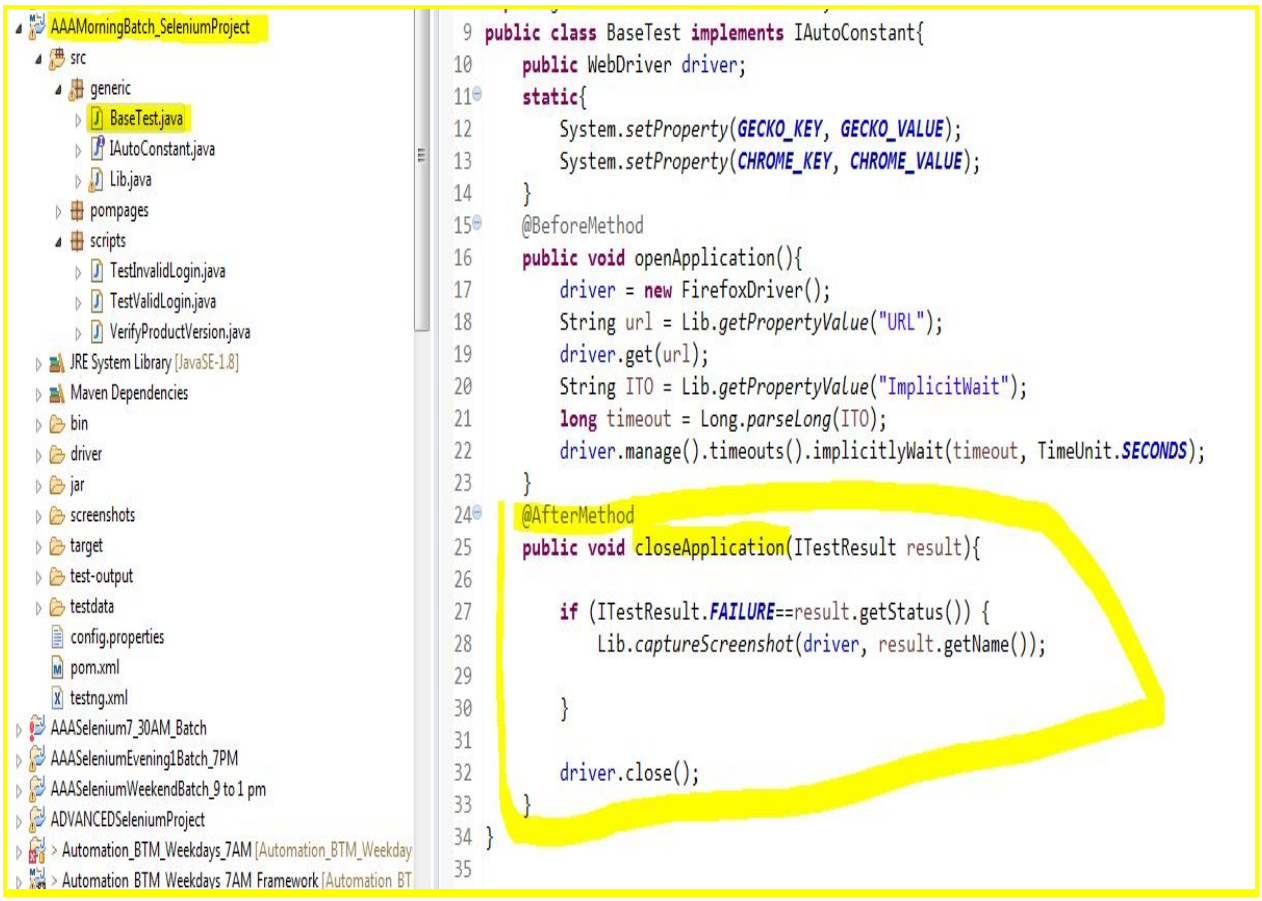
ITestResult

- It is an Interface which keep all information about the test case which we executed
- We will capture some information from this like

TestCase execution status

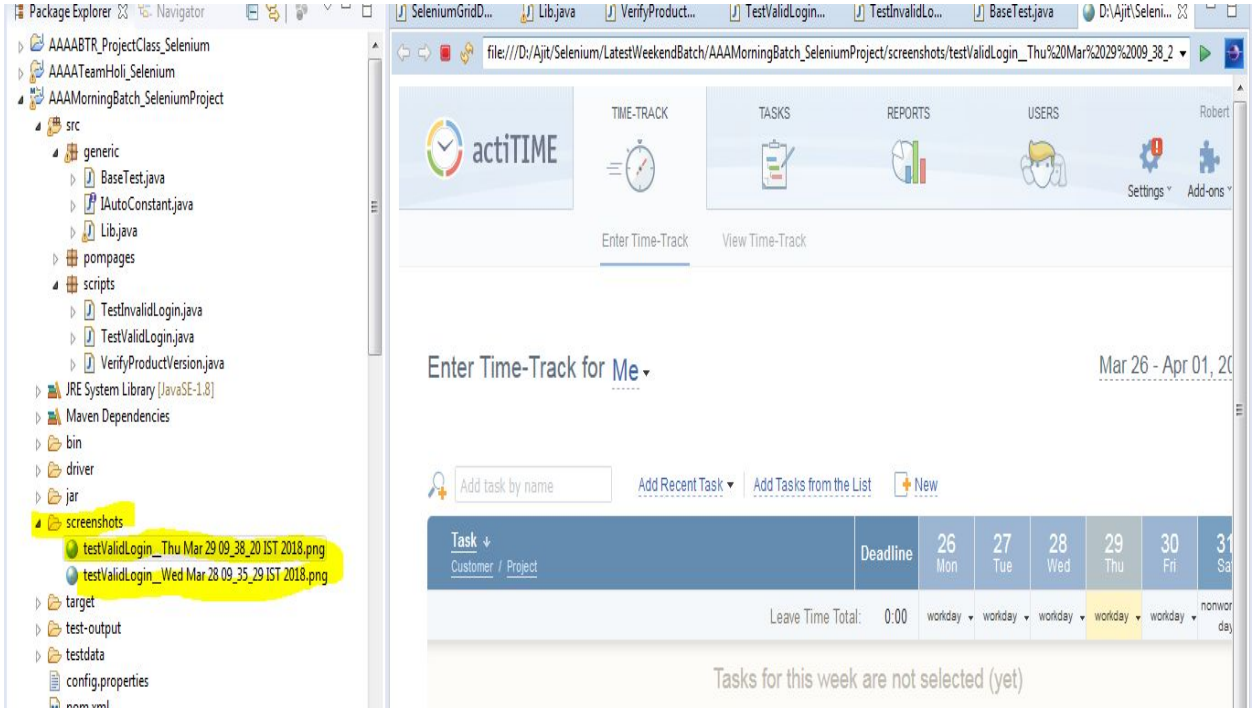
Testcase name





After a script is executed, it captured the screenshots as shown below.

****



Thread.sleep():

The method Thread.sleep throws InterruptedException. We have to handle InterruptedException in our threads because we are calling methods which throw InterruptedException.

Thread.sleep() method can be used to pause the execution of current thread for specified time in milliseconds. ...

It has overloaded methods. One overloaded method is sleep(long millis, int nanos) that can be used to pause the execution of current thread for specified milliseconds and nanoseconds.

List of Exceptions

****

1. IllegalStateException [Java - Unchecked] (driver exe path not set)-

   When Selenium Server tries to communicate with the browser without using relevant driver executables, we get this exception. We avoid this exception by setting the path of relevant driver executable.

2. InterruptedException [Java - Checked] (Thread.sleep)

3. IOException[Java-Checked][File handling scenario]

4. AWTException [Abstract Window Toolkit] [java - checked] [While handling Robot object]

5. NoSuchElementException[Selenium - unchecked][unable to locate the element]-

findElement() method using specified locators fails to uniquely identify a matching element on an webpage, it throws no such element exception.

6. Javascript Exception[Selenium-Unchecked][on clicking on a button using submit() and the button don't have an attribute called type='submit']

7. InvalidElementStateException[Selenium- unchecked]- when element is disabled, and we use sendkeys, or clear or click methods, to perform any operation on the disabled element.

8. NoSuchFrameException[Unchecked - selenium] [when specified frame is not present on the webpage]

9. NoSuchWindowException[Unchecked - selenium] no such window: target window already closed

10. NoAlertPresentException [Selenium - unchecked] [When no alert is present]

11. **NoSuchSessionException**: Session ID is null [Unchecked - Selenium] when driver.quit is called and then you are trying to access any browser

12. **ElementNotVisibleException** :- Element not visible but present in DOM, we get this exception.

URL for the automation framework

https://github.com/ajitbiswas/NewYearBatch_WeekendBatch10AM