

Selenium Notes

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
 - **TestNG** - 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 browser

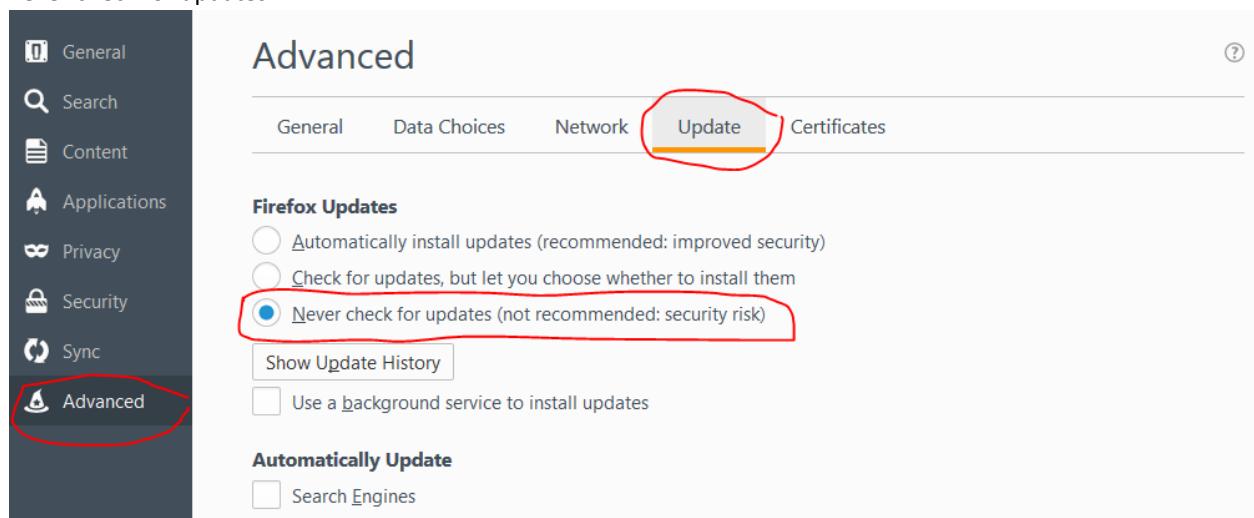
- the name of the driver executable is : **chromedriver.exe**
- Url to download : <https://chromedriver.storage.googleapis.com/index.html?path=2.31/>
- Stable version of chrome version is 62.0 (Use chromedriver.exe with version 2.33)

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>

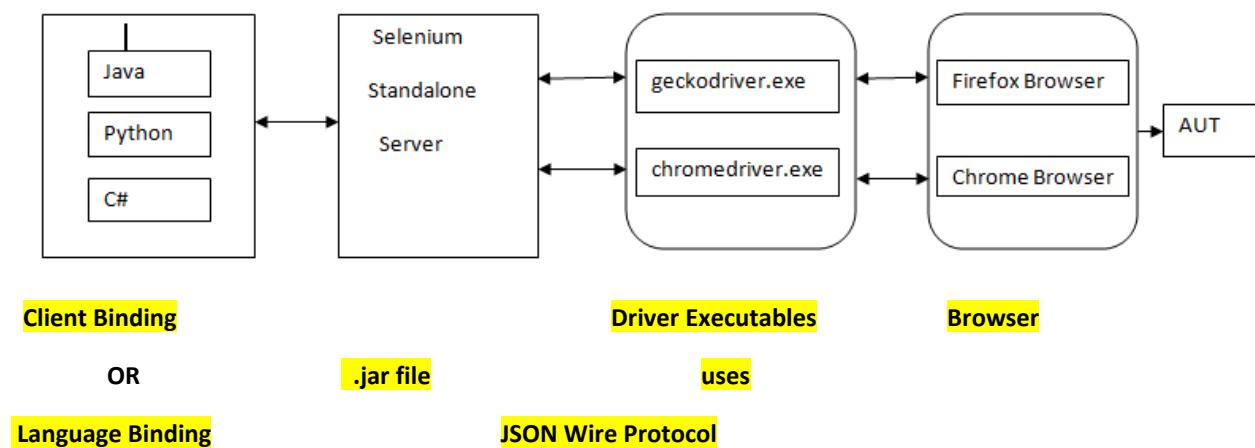
To download actiTIME application ,

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

6. Selenium Architecture - High Level ?

OR

How selenium performs automation testing on browser ?



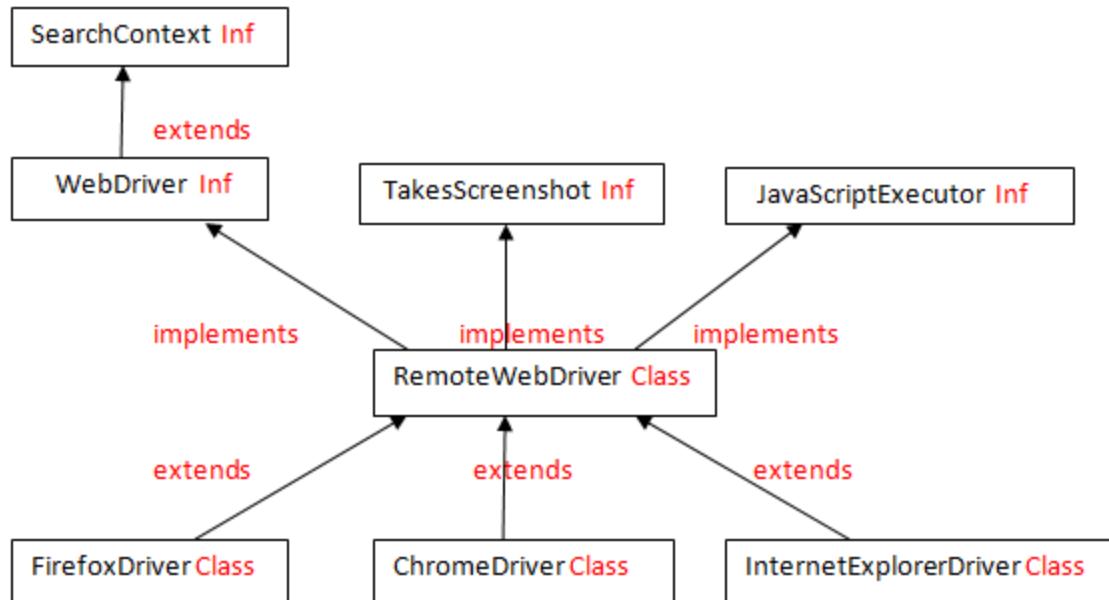
1. Since selenium supports multiple languages such as Java, Python, C# etc, we can develop automation scripts in all the supported languages. This is known as language binding or Client binding.
2. When we execute the selenium code, request goes to the Selenium Standalone Server (also known as Selenium WebDriver Server), which further processes 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.

The screenshot shows a GitHub repository page for SeleniumHQ/selenium. The top navigation bar includes 'Watch 992', 'Star 7,970', and 'Fork 3,275'. Below the bar, there are tabs for 'Code', 'Issues 342', 'Pull requests 125', 'Projects 1', 'Wiki', and 'Insights'. A red box highlights the breadcrumb navigation path: 'selenium / java / client / src / org / openqa / selenium /'. Below this, a commit list is shown, with the first commit by 'shs96c' circled in red.

Commit	Message	Date
shs96c	Fix the javadocs generation	Latest commit 85b8779 4 days ago
..		
chrome	Hide JSON processing behind our own APIs	11 days ago
edge	Hide JSON processing behind our own APIs	11 days ago
firefox	Code clean up to use braces and java 7 features	4 days ago
html5	For #401, apply a consistent copyright notice to the java/ tree.	3 years ago
ie	Merging capabilities to options should be fluent	11 days ago
interactions	Attempt to fix the build: varargs are never null	11 days ago
internal	Mark `Lock` for deletion	a month ago
io	Start using the skylark parser where possible	17 days ago
ison	Fix the iavadocs generation	4 days ago

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 dont use this for automation

Methods of WebElement interface :

1. `clear()`
2. `click()`
3. `getAttribute()`
4. `getCssValue()`
5. `getLocation()`
6. `getRect()`
7. `getSize()`
8. `getTagName()`
9. `getText()`
10. `isDisplayed()`
11. `isEnabled()`
12. `isSelected()`
13. `sendKeys()`
14. `submit()`

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/>

WebDriver is the name of the key interface against which tests should be written in Java, the implementing classes one should use are listed as below:

[ChromeDriver](#), [EventFiringWebDriver](#), [FirefoxDriver](#), [HtmlUnitDriver](#), [InternetExplorerDriver](#),
[PhantomJS](#)[Driver](#), [RemoteWebDriver](#), [SafariDriver](#)

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

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();

    }
}
```

}

7th Oct

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 typecasted 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.

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 navigate() methods.

```
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();
    }
}

```


Handling Mouse and Keyboard Operations

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 the web elements on the web page.
- We have 8 types of locators in Selenium using which findElement() methods identifies elements on the web page:
 1. id
 2. name
 3. tagName
 4. className
 5. linkText
 6. partialLinkText
 7. xpath
 8. cssSelector
- findElement() method returns the address of the web elements on the web page and the return type is WebElement.
- If the specified locators returns multiple elements, then findElement() method returns the address of the first matching element.
- If the specified locators returns No element, then findElement() method throws an exception called "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.className(""))
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.staticConcrete> methods

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 everytime 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”

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.

- 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 :

- In order to write cssSelector expression in chrome browser, we don't need any add-ons as such.
- 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.
- Press Ctrl+F and write the cssSelector expression, it will highlight the source code of the matching element.
- Place the cursor on the highlighted source code, it will highlight the corresponding element present on the web page.

cssSelector locator :

- It is one of the locator in Selenium using which we identify web elements on the web page.
- It stands for Cascading Style Sheet.
- 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.

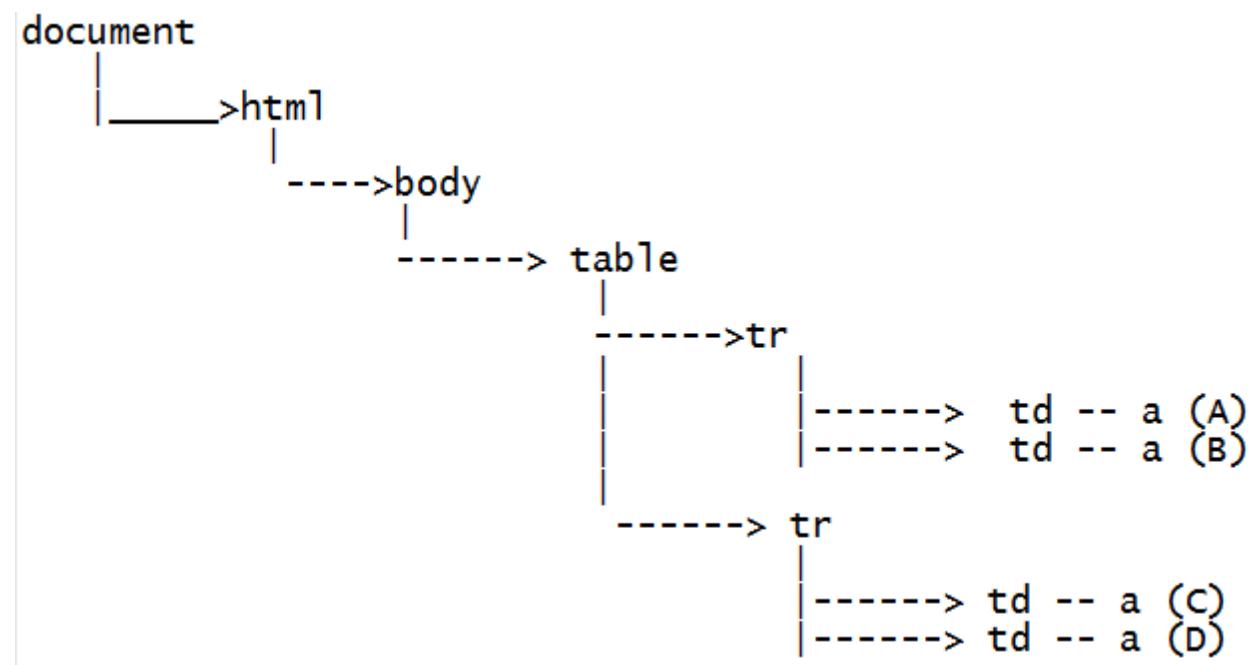
```
driver.findElement(By.xpath("html/body/a")).click();
```

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 (|)

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
	A
	B
	C
	D
	AB
	CD

	AC
	BD
	AD
	BC
	ABC
	ABD
	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
	A
	B
	C
	D
	AB
	CD
	AC
	BD
	AD
	BC
	ABC
	ABD

	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 :

- 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.
- 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 :

- Identify the independent element on the webpage and inspect the element to view the source code and then derive the xpath expression.
- 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.
- Use mouse pointer to navigate from common parent to the desired dependent element and derive the xpath of the dependent element.
- 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 :

Sample Html tree :

The screenshot shows a browser window with the URL `file:///D:/Ajit/Selenium/SeleniumBtm_7thSep17/webpages/GroupIndex.html`. Inside the browser, there is a 2x2 grid of input fields:

A	B
C	D

Below the browser is the FirePath interface. The top bar includes icons for file operations, a search bar, and tabs for Console, HTML, CSS, Script, DOM, Net, Cookies, and FirePath (which is selected). The main area shows the DOM tree:

```
<document>
  <html>
    <head>
    <body>
      <div>
        <input value="A" type="text"/>
        <input value="B" type="text"/>
        <br/>
      </div>
      <div>
        <input value="C" type="text"/>
        <input value="D" type="text"/>
      </div>
    </body>
  </html>
</document>
```

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

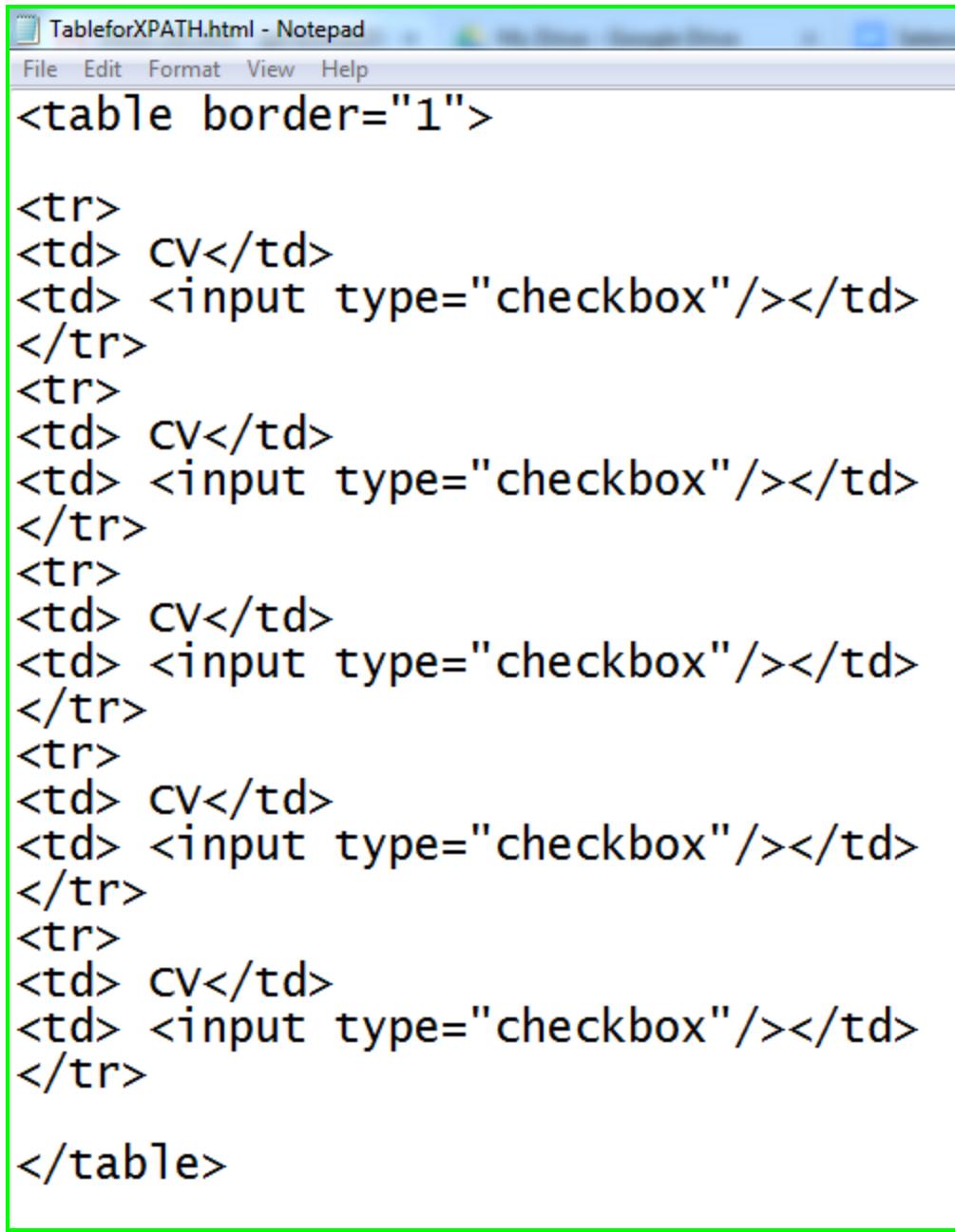
4. click on Settings
5. Click on the Types of Work link present in the window
6. click on the Set by Default link for a type of work called “testing”

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.



```
<table border="1">

<tr>
<td> CV</td>
<td> <input type="checkbox"/></td>
</tr>

</table>
```

Xpath expression using GroupIndex concept :

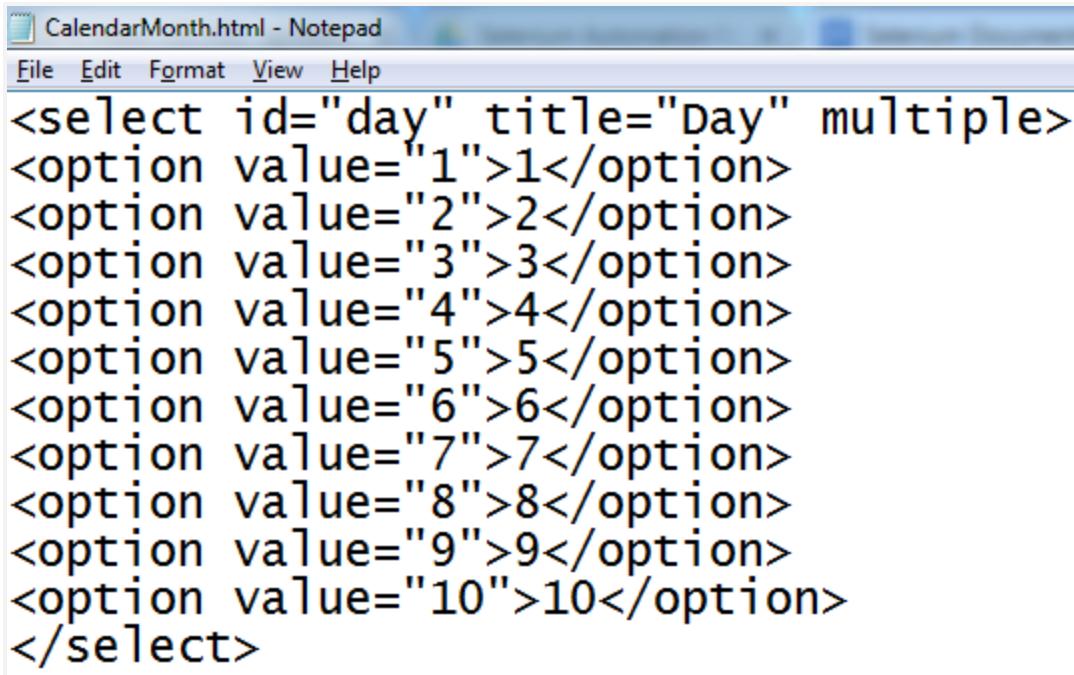
XPATH using Group Index	Matching Element
//input[@type='checkbox']	ABCDE
(//input[@type='checkbox'])[1]	A
(//input[@type='checkbox'])[-LAST()]	E
(//input[@type='checkbox'])[-POSITION()=3]	C
(//input[@type='checkbox'])[-POSITION()>=3]	CDE
(//input[@type='checkbox'])[-POSITION() < 3]	AB
(//input[@type='checkbox'])[-POSITION()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

Create a .html file with the below html code



The screenshot shows a Notepad window titled "CalendarMonth.html - Notepad". The menu bar includes File, Edit, Format, View, and Help. The main content area contains the following HTML code:

```
<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 “ > ”

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");

    }

}

}

```

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_Ycoordinate = unTB.getLocation().getY();
    System.out.println(username_Ycoordinate);
    WebElement pwdTB = driver.findElement(By.name("pass"));
    // get the y-coordinate of password field
    int password_Ycoordinate = pwdTB.getLocation().getY();
    System.out.println(password_Ycoordinate);
    //check whether the Y-coordinate of username and password field are same
    if (username_Ycoordinate==password_Ycoordinate) {
        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 :

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.DELETE); //  

        // 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_selectededorNot{  
  
    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 VerifyKeepMeLoggedInAsaCheckboxinActitime 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()

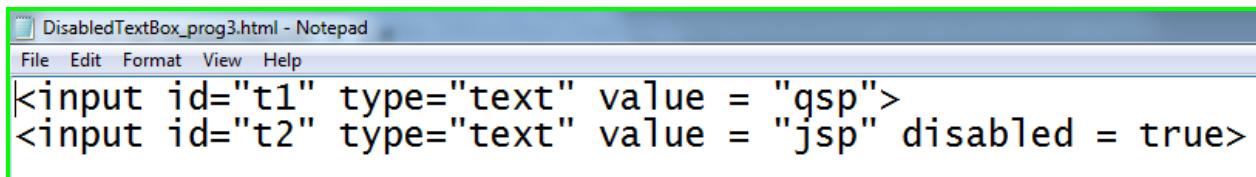
We use JavascriptExecutor when we fail to perform some actions using selenium.

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.



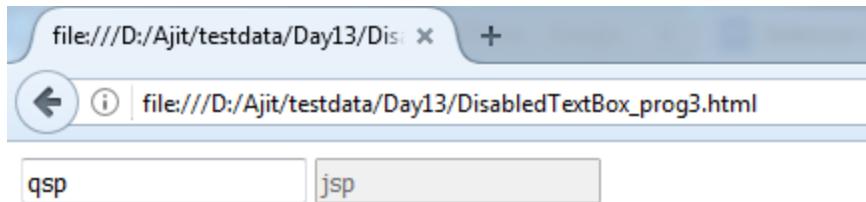
The screenshot shows a Notepad window titled "DisabledTextBox_prog3.html - Notepad". The menu bar includes File, Edit, Format, View, and Help. The main content area contains the following HTML code:

```

<input id="t1" type="text" value = "qsp">
<input id="t2" type="text" value = "jsp" disabled = true>

```

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();

}

*****

```

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.

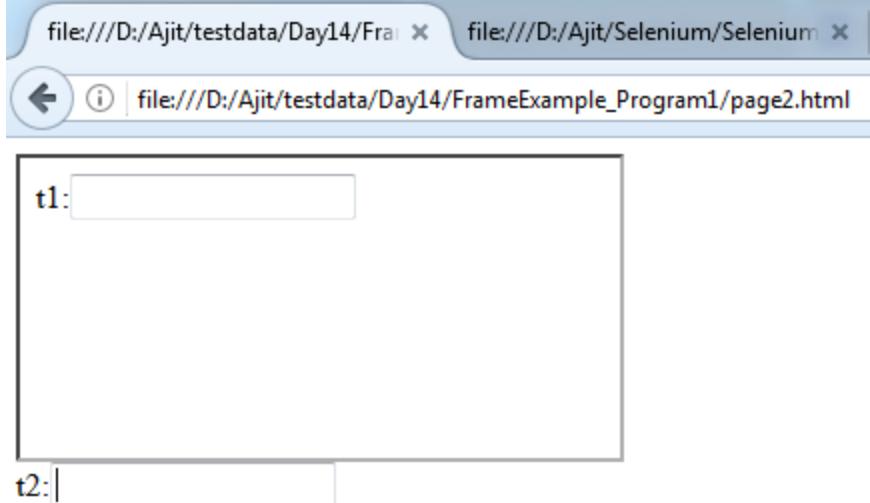
Create a sample webpage using the below html source code and save the file as Page1.html

```
t1:<input type="text" id="t1">
```

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

```
<iframe id="f1" name="n1" class="c1" src="Page1.html"></iframe><br>t2:<input type="text" id="t2">
```

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_Page2.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();

}

}

```

ACTIONS Class :

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 Applicationsss"

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();

    }

}

```

2. Hidden Division Popup

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 {

        Date d = new Date();

        String str = d.toString();

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

        String today = str2[2];

        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 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

```
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\\testdata\\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.mozilla.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>

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:

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.

How to handle Child browser pop up ?

- We handle child browser popup by using `getWindowHandle()` and `getWindowHandles()` methods of `WebDriver` interface.
- In Selenium, every browser will have an unique window handle id.
- In `firefox browser`, window handle is an `integer value`, whereas in `Chrome browser`, it is an `unique alpha numeric string`.

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();

}

}

```

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>

AutoIt Script Editor.(Customised version of SciTE with lots of additional coding tools for AutoIt)



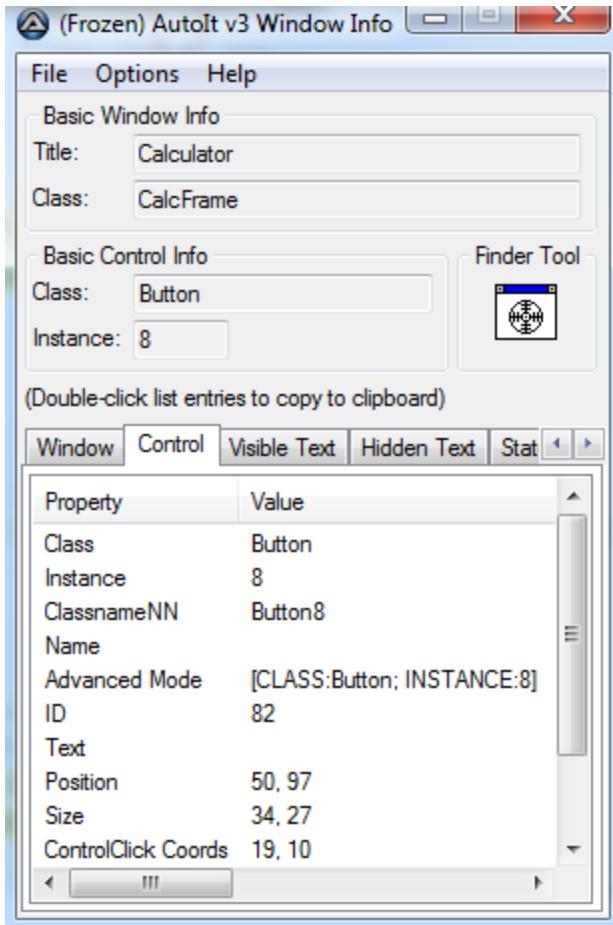
- Download the above Editor on your system.
- Double click on the Setup file.
- Follow the default instruction to install autoIT.

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

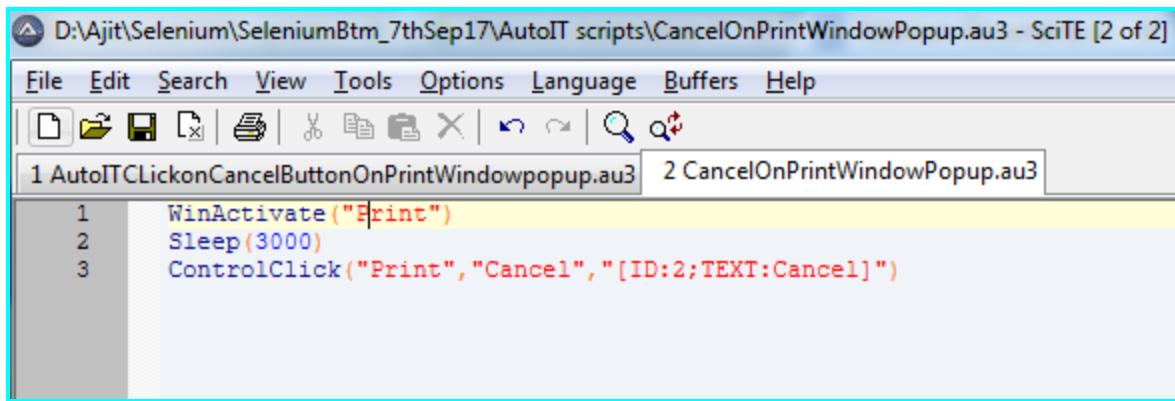
```
RunTime.getRuntime().exec("path of the compiled au3.exe file");
```

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.



```
1 AutoITClickonCancelButtonOnPrintWindowpopup.au3 2 CancelOnPrintWindowPopup.au3
1 WinActivate("Print")
2 Sleep(3000)
3 ControlClick("Print", "Cancel", "[ID:2;TEXT:Cancel]")

WinWait("Print", "Print Preview", 10)
ControlClick("Print", "Cancel", "[ID:2;TEXT:Cancel]")
```

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:

```

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 auto it script editor and write the below script.

Edit; INSTANCE:1]"), ControlSetText("File Upload", "", "[CLASS>Edit; INSTANCE:1]", "C:\Users\admin\Desktop\Test Yantra Sel batch.txt"), ControlClick("File Upload", "", "[CLASS/Button; INSTANCE:1]")."/>

```

1 WinActivate("File Upload")
2 Sleep(2000)
3 ControlFocus("File Upload", "", "[CLASS>Edit; INSTANCE:1]")
4 ControlSetText("File Upload", "", "[CLASS>Edit; INSTANCE:1]", "C:\Users\admin\Desktop\Test Yantra Sel batch.txt")
5 ControlClick("File Upload", "", "[CLASS/Button; INSTANCE:1]")
6

```

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

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 finds 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

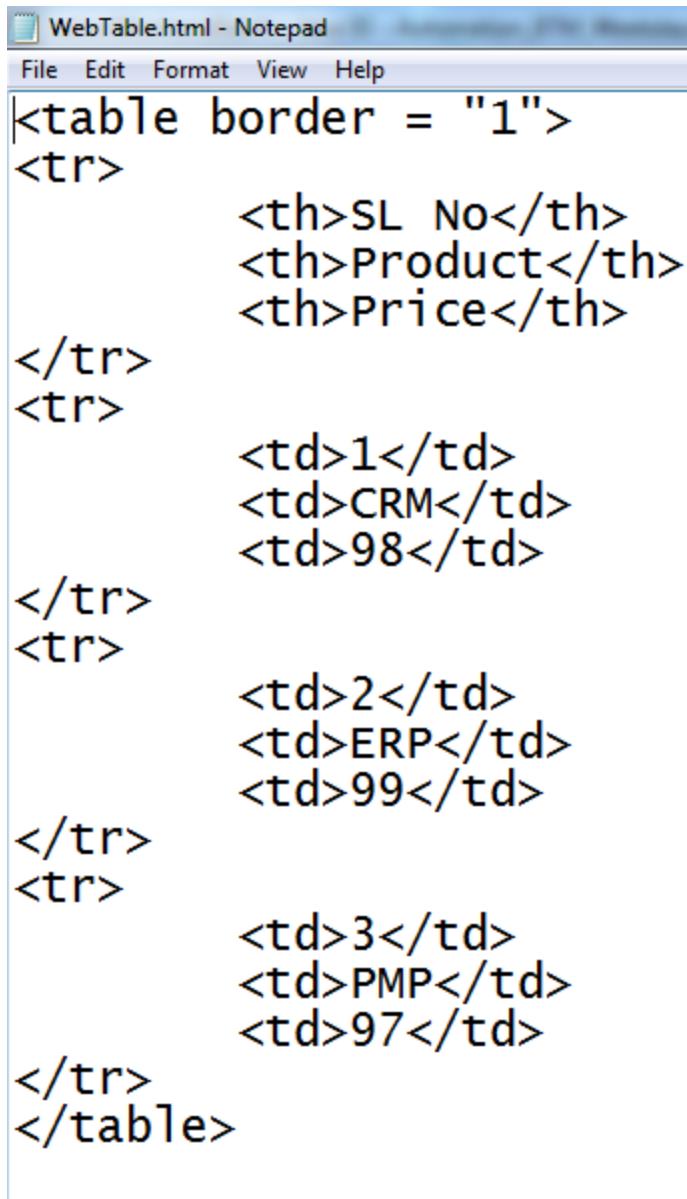
Selenium Code:

[Copy code here.](#)

WebTable :

Table present on the web page is called WebTable.

Create a webtable as shown below.



The screenshot shows a Notepad window titled "WebTable.html - Notepad". The menu bar includes File, Edit, Format, View, and Help. The main content area contains the following HTML code:

```
<table border = "1">
<tr>
    <th>SL No</th>
    <th>Product</th>
    <th>Price</th>
</tr>
<tr>
    <td>1</td>
    <td>CRM</td>
    <td>98</td>
</tr>
<tr>
    <td>2</td>
    <td>ERP</td>
    <td>99</td>
</tr>
<tr>
    <td>3</td>
    <td>PMP</td>
    <td>97</td>
</tr>
</table>
```

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

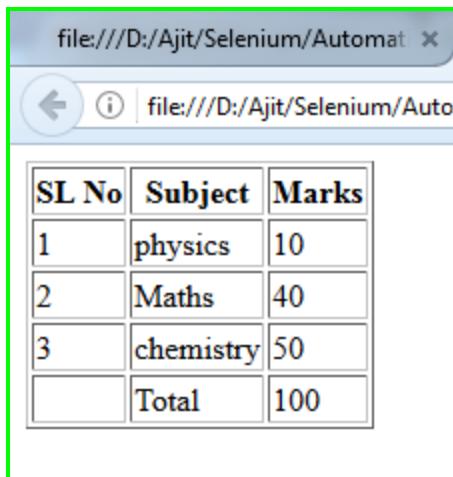
driver.close();

}
}

```

Assignment :

Write a script to verify that the sum of marks present in the below table is same as the Total marks.



A screenshot of a web browser window titled "file:///D:/Ajit/Selenium/Automat". The address bar shows the same URL. The main content is a table with the following data:

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 ?

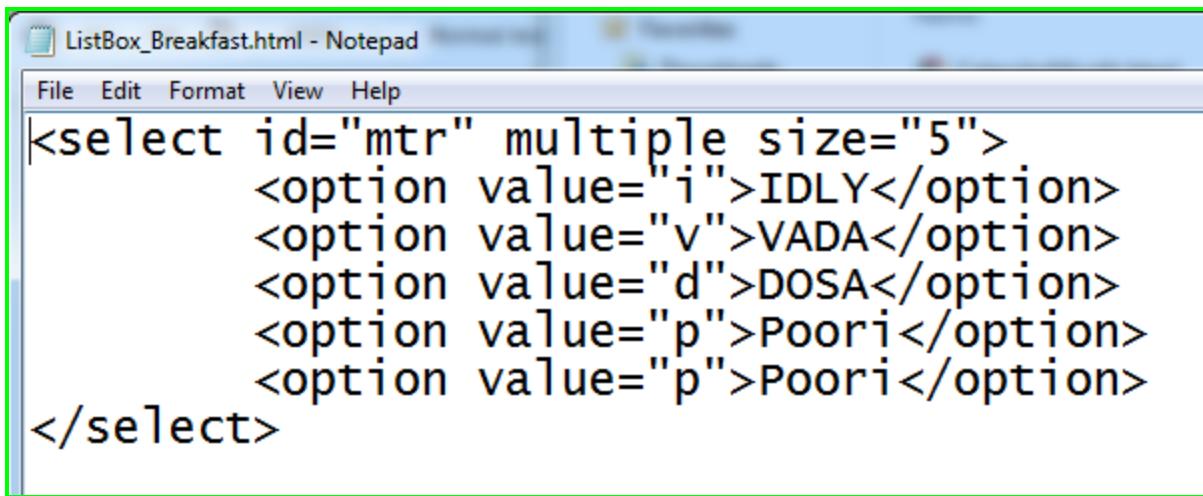
- In Selenium, we handle listbox using **Select** class.
- Select class is present in `org.openqa.selenium.support.ui` package.
- Select class has a parameterized constructor which accepts an argument of `WebElement` object (List box element)
- Following are the available methods of Select class
 - `selectByIndex()`
 - `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



The screenshot shows a Notepad window titled "ListBox_Breakfast.html - Notepad". The content of the file is an HTML snippet for a dropdown menu. It starts with a <select> tag, followed by five <option> tags with values "i", "v", "d", "p", and "p" respectively. The "p" values correspond to the words "Poori" and "Poori". Finally, there is a closing </select> tag. The entire code is highlighted with a green border.

```
<select id="mtr" multiple size="5">
    <option value="i">IDLY</option>
    <option value="v">VADA</option>
    <option value="d">DOSA</option>
    <option value="p">Poori</option>
    <option value="p">Poori</option>
</select>
```

Selenium Code :

```
public class ListBoxExample extends BaseClass{

    public static void main(String[] args) {

        driver.get("file:///D:/Ajit/Selenium/SeleniumBtm_7thSep17/webpages/ListBox_Breakfast.htm
l");

        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/ListBox_Breakfast.htm
l");

        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.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 -----");

        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 occurrence 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 occurrence 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**

- Easy readability of scripts
- Reduce or eliminate duplicacy
- Re-usability of code
- Reliability
- It is the object repository
- we achieve encapsulation using this pom framework.

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

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.

```
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 testing 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 @DataProviders

1. DataProvider is an annotation in testng using which we can create data bank.
2. This databank can be used across any testng class, thus achieving data parameterisation.

Executing the same script with multiple sets of data is called data parameterisation.

From Testng Class by creating our own data bank using DataProvider annotation

From Testng class

1. We use DataProvider annotation to create the data bank,

2. We utilise this data from any testng methods by using

"dataProvider" parameter as an argument to "Test" annotation.

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);

}

}
```

SELENIUM GRID

To run the same scripts on multiple browsers and multiple systems parallelly, 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-standalone-server.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 rules and guidelines which should be followed by all the automation engineers in the team while automating an 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 pages for all the page of our application under test.
2. In our application, we have 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 scripts.
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 this (which represent the current class object), 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 class. 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 :

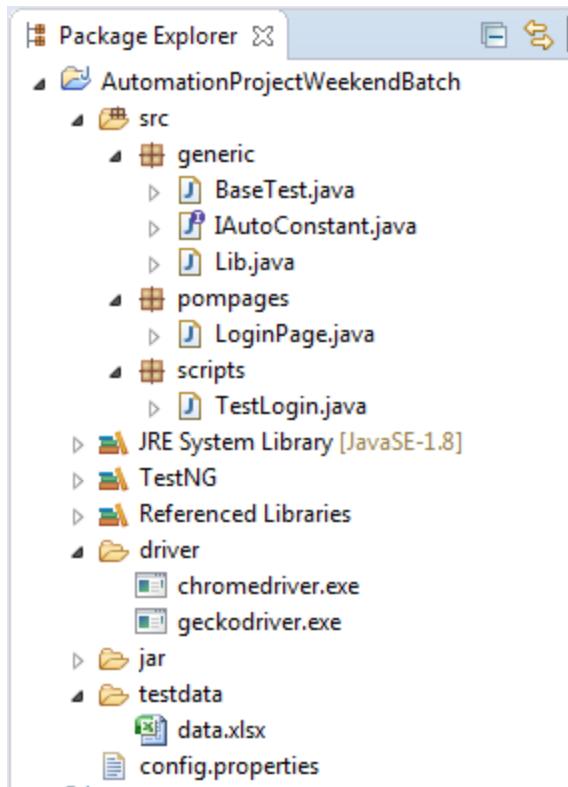
1. Module wise execution of test scripts is known as modular driven framework.
2. In our project, when ever we develop a test method while automating a test case, we tag it to some group based on the module name.
3. Now, if any test script fails during normal execution cycle, we log defects and once 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 kind of method driven framework as well.

In our way, the framework that we have implemented is a HYBRID framework.

Project Framework Folder Structure is mentioned below.



Config.Properties snapshot below :

The screenshot shows the Eclipse Properties editor for a Java resource. The tabs at the top are *BaseTest.java, *IAutoConst..., *Lib.java, and LoginPage.java. The configuration section displays the following properties:

```
1 URL=http://localhost:8080/login.do
2 ImplicitTimeOut=10
```

data.xlsx snapshot below:

A screenshot of Microsoft Word showing a table with two columns: "A" and "B". The first row contains "Username" and "Password". The second row contains "admin" and "manager". The table is selected, indicated by a black border around the second row.

	A	B
1	Username	Password
2	admin	manager
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		

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

MAVEN PROJECT :

Apache Maven is a software project management and build dependency management tool for Selenium/java frameworks.

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

POM.xml is the heart of Maven project.

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>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>test</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-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>

```

Solutions in case of maven issues

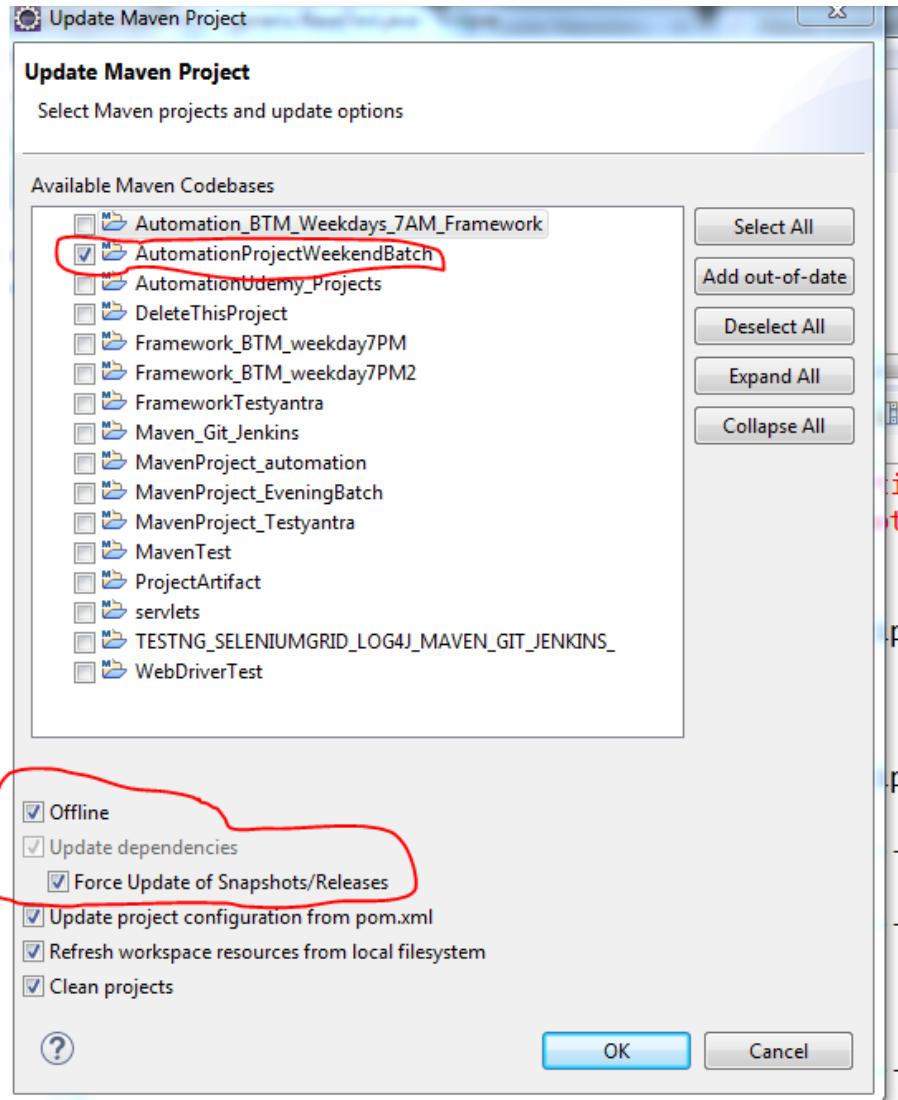
1. No compiler added to the build

Solution : Windows - preference -- Java - Installed JREs → navigate to location wherer your jdk1.8 is iintalled
(c--program files --java - JDK)

2. MOJOException

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

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.

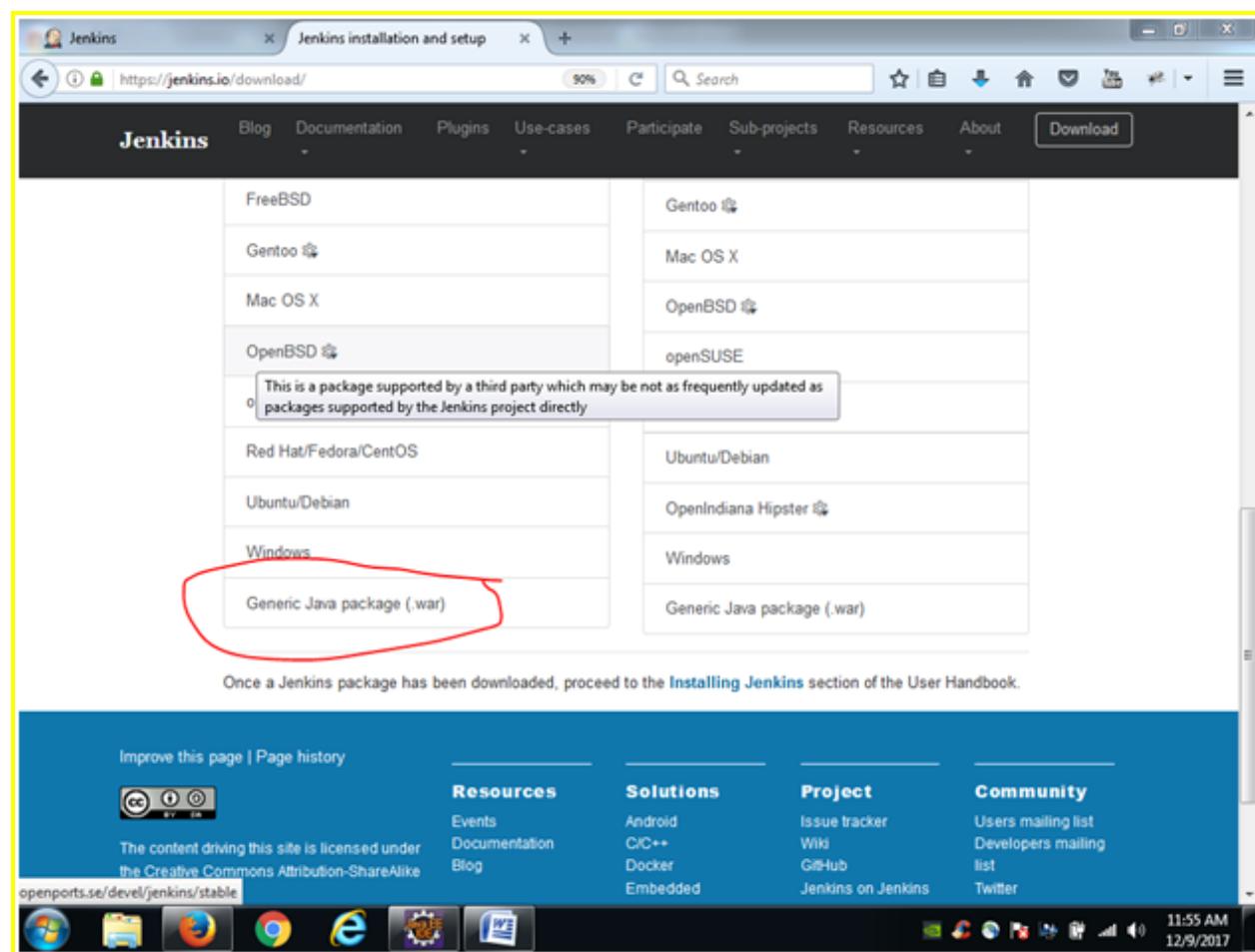
- Automatic kick off of automation suite execution as soon as the build gets deployed from DEV environment to QA environment.

How to download jenkins

URL :

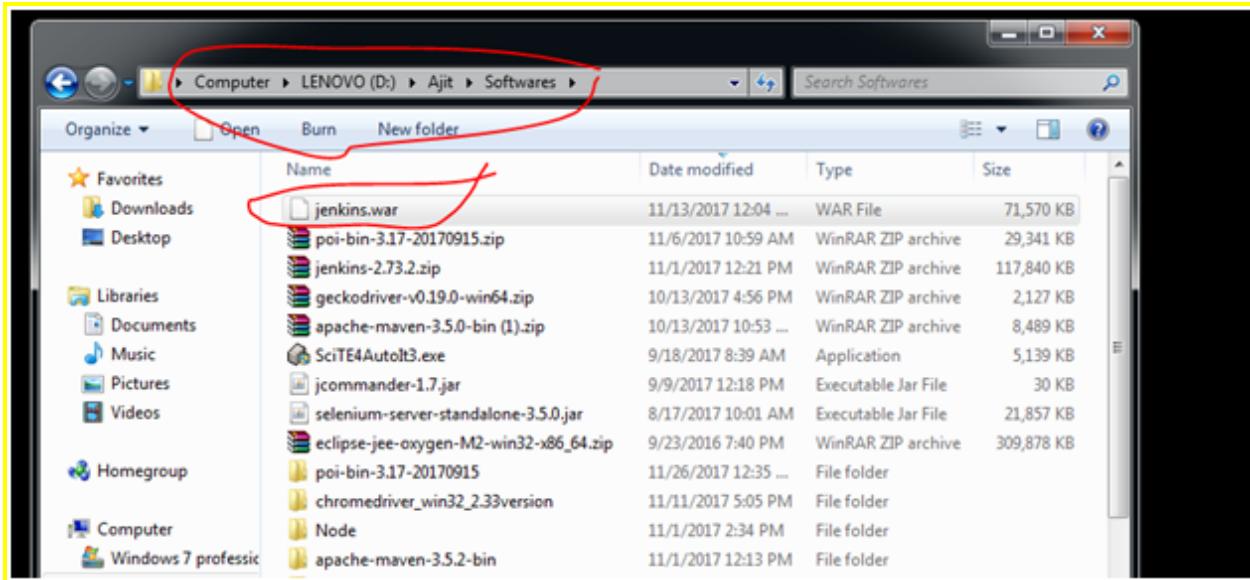
<https://jenkins.io/download/>

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

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\admin>d:
D:>\cd D:\Ajit\Softwares
D:\Ajit\Softwares>java -jar jenkins.war
```

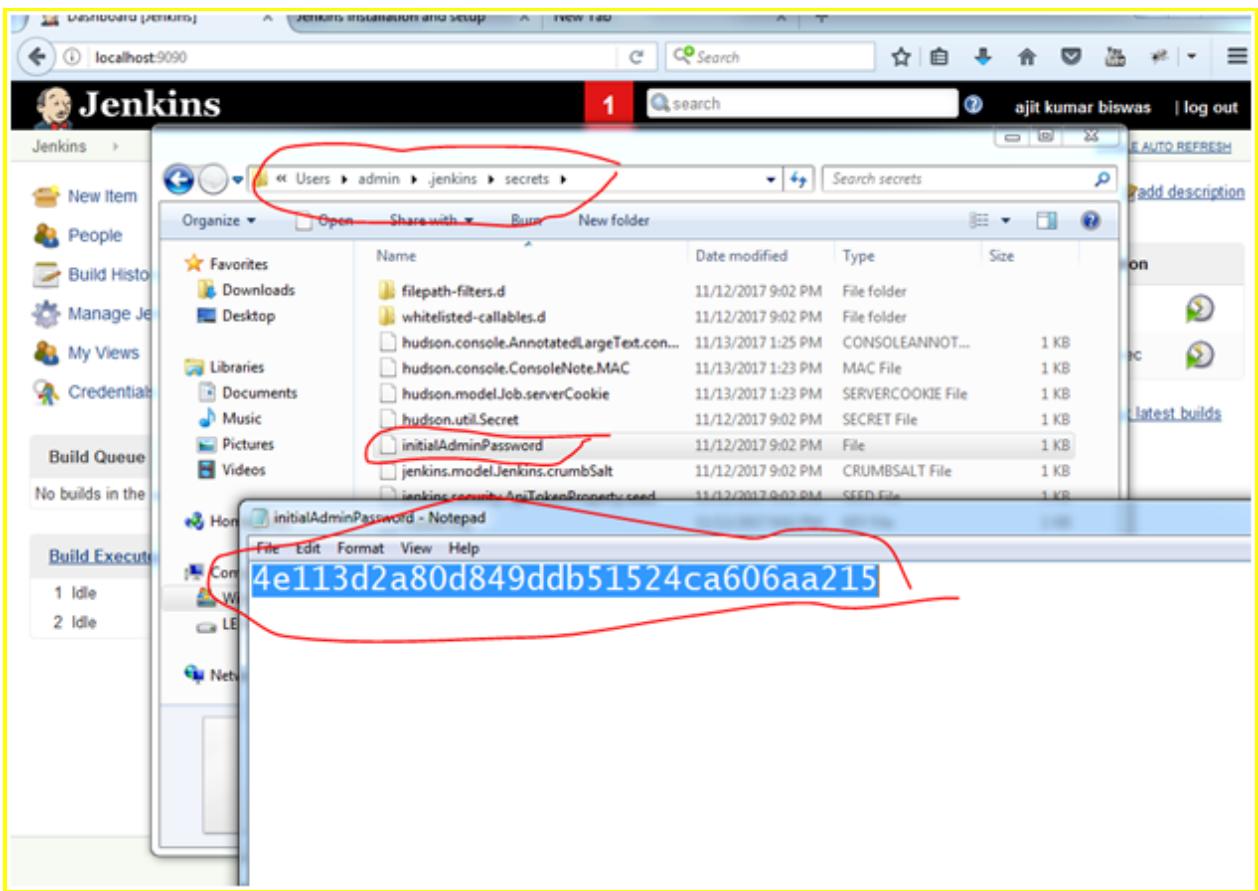
It will say – Jenkins Is up and running

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

The screenshot shows the Jenkins 'Manage Jenkins' page. On the left, there's a sidebar with links: People, Build History, **Manage Jenkins** (which is highlighted with a red oval), My Views, and Credentials. Below these are two expandable sections: 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 Idle, 2 Idle). The main content area is titled 'manage JENKINS'. It displays a message about a new Jenkins version (2.89.1) available for download. There are two buttons: 'Or Upgrade Automatically' (highlighted with a blue box) and 'Downgrade to 2.73.2'. Below this, there's a section for restoring the previous version with a 'Restore' button and a 'Downgrade to 2.73.2' button. The main list of management options includes:

- Configure System**: Configure global settings and paths.
- Configure Global Security**: Secure Jenkins; define who is allowed to access/use the system.
- Configure Credentials**: Configure the credential providers and types.
- Global Tool Configuration**: Configure tools, their locations and automatic installers. (This item is highlighted with a red oval).
- Reload Configuration from Disk**: Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.
- Manage Plugins**: Add, remove, disable or enable plugins that can extend the functionality of Jenkins. (**updates available**)
- System Information**: Displays various environmental information to assist trouble-shooting.
- System Log**: System log captures output from java.util.logging output related to Jenkins.
- Load Statistics**: Check your resource utilization and see if you need more computers for your builds.

Add JDK to jenkins

Global Tool Configuration [Jenkins] Jenkins installation and setup New Tab

localhost:8080/configureTools/ Search ajit kumar biswas | log out

Jenkins

1 search

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... (highlighted with a red box)

Git

Git installations

Git

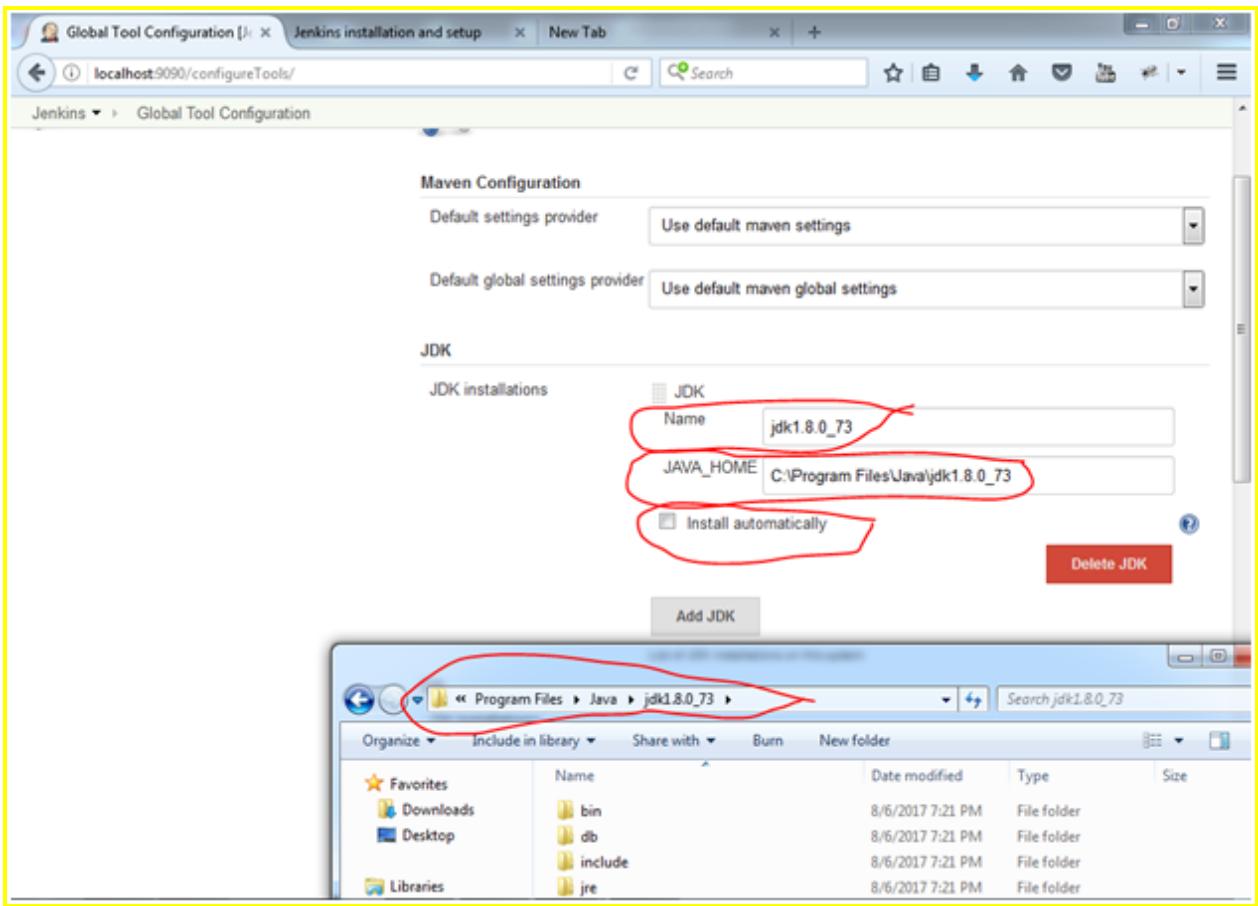
Name: Default

Path to Git executable: git.exe

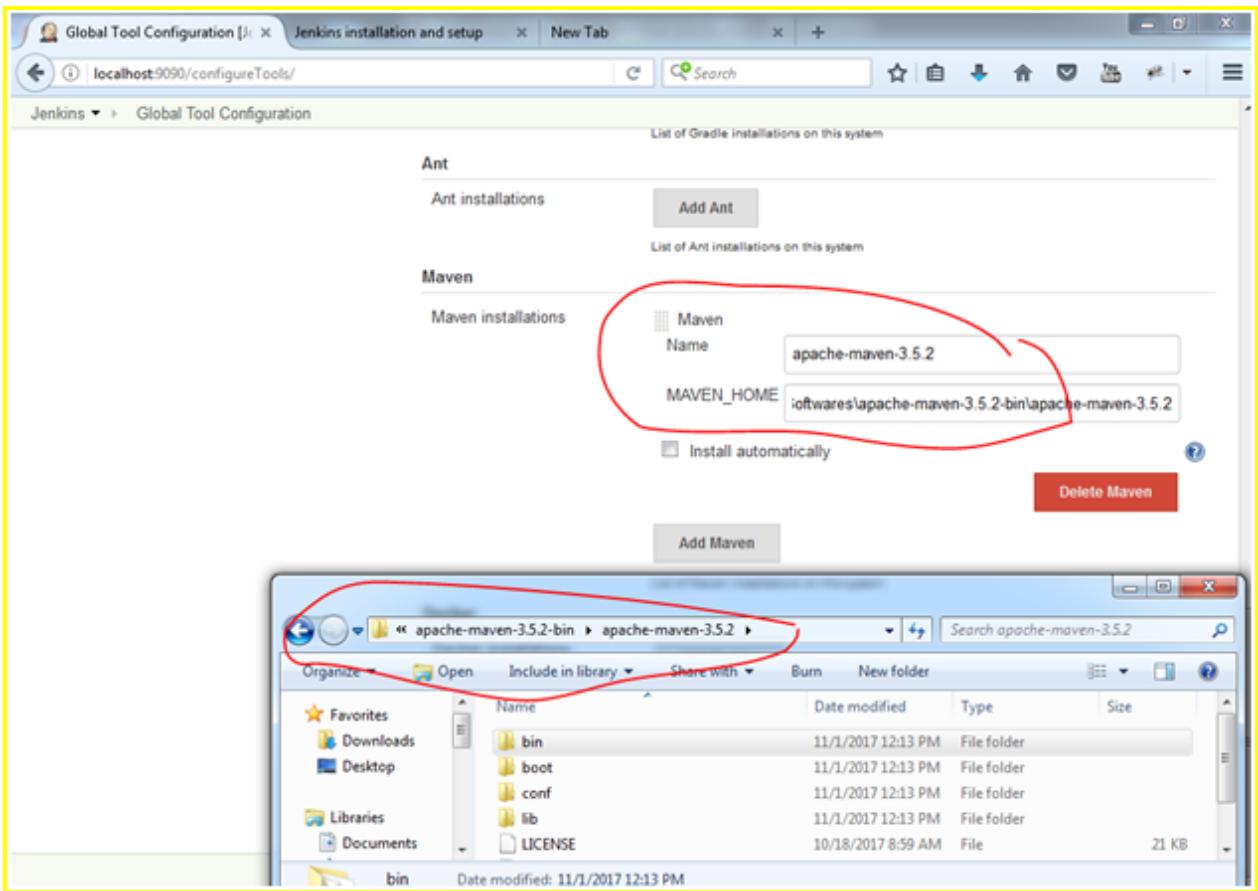
There's no such executable git.exe in PATH: D:/Ajit/Softwares/apache-maven-3.5.2-bin/apache-maven-3.5.2/bin, C:/ProgramData/Oracle/Java/javapath,

Save Apply

The screenshot shows the Jenkins Global Tool Configuration page. It has sections for Maven Configuration, JDK, and Git. The 'JDK' section contains a 'JDK installations...' button, which is circled in red. The 'Git' section shows a configuration for a 'Default' git installation with a path to 'git.exe'. A warning message indicates that 'git.exe' is not found in the specified PATH. At the bottom are 'Save' and 'Apply' buttons.



Add Maven Path as shown below.



Once you save, u get this page.

The screenshot shows the Jenkins Manage Jenkins interface. At the top, there are three tabs: 'Manage Jenkins [Jenkins]', 'Jenkins installation and setup', and 'New Tab'. The URL in the address bar is 'localhost:9090/manage'. The main header says 'Jenkins' and shows a user icon for 'ajit kumar biswas' with a 'log out' link. A 'ENABLE AUTO REFRESH' button is also present.

The left sidebar has links for 'New Item', 'People', 'Build History', 'Manage Jenkins' (which is selected), 'My Views', and 'Credentials'. Under 'Manage Jenkins', there are two collapsed sections: 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 Idle, 2 Idle).

The central content area is titled 'Manage Jenkins' and displays a message: 'New version of Jenkins (2.89.1) is available for [download \(changelog\)](#)'. Below this is a yellow warning icon with the text 'Or Upgrade Automatically'.

On the right, there's a 'Downgrade to 2.73.2' button. Below it is a list of management options:

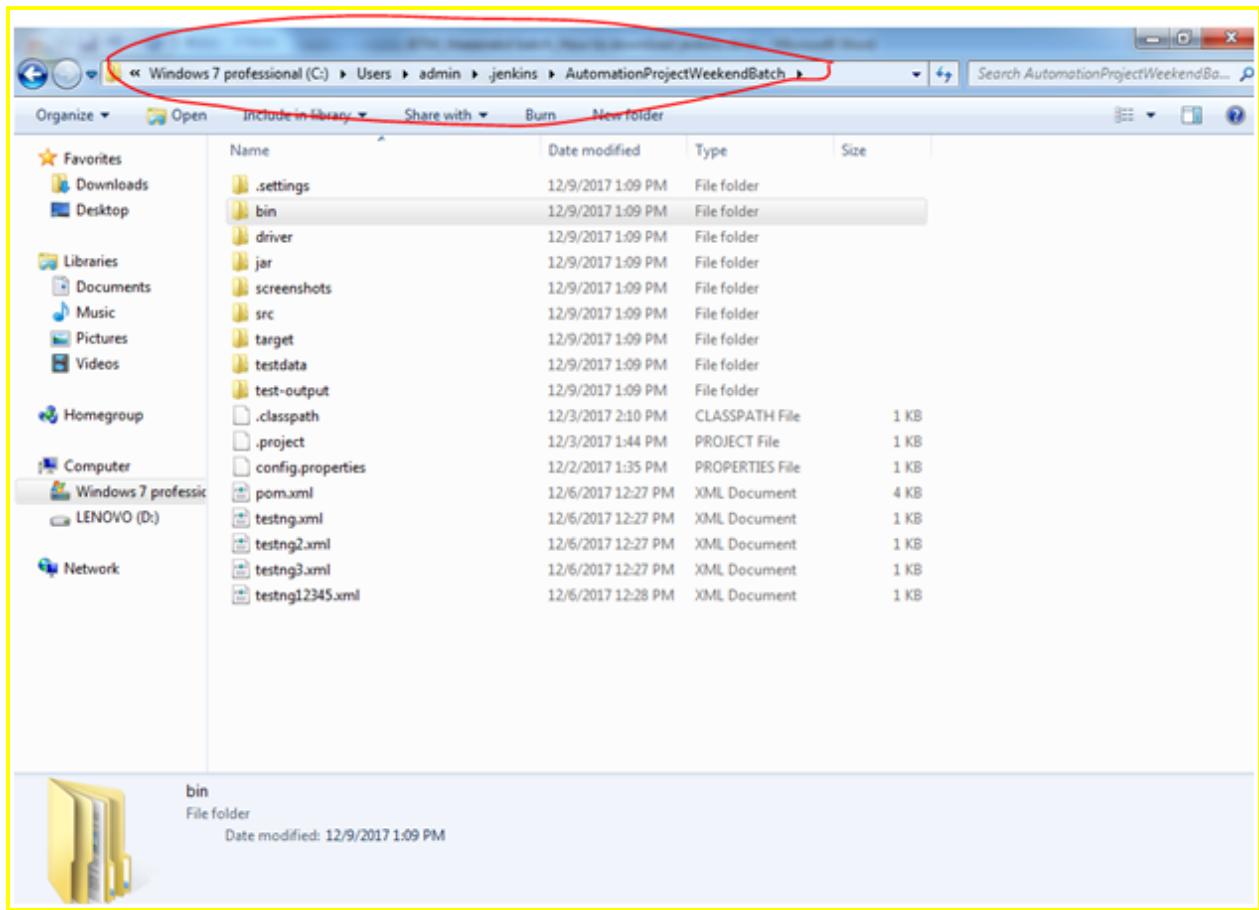
- Configure System**: Configure global settings and paths.
- Configure Global Security**: Secure Jenkins; define who is allowed to access/use the system.
- Configure Credentials**: Configure the credential providers and types.
- Global Tool Configuration**: Configure tools, their locations and automatic installers.
- Reload Configuration from Disk**: Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.
- Manage Plugins**: Add, remove, disable or enable plugins that can extend the functionality of Jenkins. **(updates available)**
- System Information**: Displays various environmental information to assist trouble-shooting.
- System Log**: View the system log.

Copy the project that you want to execute from Jenkins as shown below

A screenshot of a Windows 7 professional file explorer window. The path in the address bar is: Computer > Windows 7 professional (C:) > Users > admin > jenkins. The search bar contains 'jenkins'. The left sidebar shows 'Favorites' (Downloads, Desktop), 'Libraries' (Documents, Music, Pictures, Videos), 'Homegroup', 'Computer' (Windows 7 professional, LENOVO (D:)), and 'Network'. The main area is a file list with columns: Name, Date modified, Type, and Size. The 'Name' column lists several Jenkins-related folders and files. One folder, 'AutomationProjectWeekendBatch', is circled in red.

Name	Date modified	Type	Size
AutomationProjectWeekendBatch	12/9/2017 1:09 PM	File folder	
DeleteThisProject	12/9/2017 9:20 AM	File folder	
Framework_BTM_weekday7PM	12/8/2017 7:58 AM	File folder	
jobs	11/13/2017 1:10 PM	File folder	
logs	11/12/2017 9:02 PM	File folder	
nodes	11/12/2017 9:02 PM	File folder	
plugins	11/13/2017 12:43 ...	File folder	
secrets	11/13/2017 1:25 PM	File folder	
updates	12/9/2017 8:08 AM	File folder	
userContent	11/12/2017 9:02 PM	File folder	
users	11/12/2017 9:02 PM	File folder	
war	11/13/2017 12:10 ...	File folder	
workflow-libs	11/13/2017 12:40 ...	File folder	
config.xml	12/9/2017 8:24 AM	XML Document	2 KB
failed-boot-attempts.txt	12/9/2017 12:09 PM	Text Document	1 KB
hudson.model.UpdateCenter.xml	12/9/2017 8:24 AM	XML Document	1 KB
hudson.plugins.git.GitTool.xml	11/13/2017 1:07 PM	XML Document	1 KB
hudson.plugins.gradle.Gradle.xml	11/13/2017 1:07 PM	XML Document	1 KB
hudson.tasks.Ant.xml	11/13/2017 1:07 PM	XML Document	1 KB
hudson.tasks.Mailer.xml	11/13/2017 1:22 PM	XML Document	1 KB
hudson.tasks.Maven.xml	11/13/2017 1:07 PM	XML Document	1 KB
identity.key.enc	11/12/2017 9:02 PM	ENC File	2 KB

Copy the path till the project name



Create a new job by clicking on new ITEM

The screenshot shows the Jenkins Manage Jenkins interface. On the left, there's a sidebar with links like 'New Item', 'People', 'Build History', 'Manage Jenkins', 'My Views', and 'Credentials'. A red circle highlights the 'New Item' link. The main content area has a heading 'Manage Jenkins' and a message about a new version available for download. It includes a 'Configure System' section with various configuration options like 'Global Security', 'Credentials', 'Tool Configuration', and 'Reload Configuration from Disk'.

New Item

Manage Jenkins

New version of Jenkins (2.89.1) is available for [download \(changelog\)](#).

Or Upgrade Automatically

Configure System

Configure Global Security

Configure Credentials

Global Tool Configuration

Reload Configuration from Disk

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 ajit kumar biswas | log out

Jenkins

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.

GitHub Organization Scans a GitHub organization (or user account) for all repositories matching some defined markers.

OK

A screenshot of a web browser showing the Jenkins 'New Item' creation interface. The URL in the address bar is 'localhost:9090/view/all/newJob'. The page title is 'New Item [Jenkins]'. The main heading is 'Jenkins'. Below it is a navigation bar with 'Jenkins' and 'All'. A red circle highlights the 'Enter an item name' input field, which contains the text 'BTM Weeekend Job'. A red circle also highlights the 'Freestyle project' section, which describes it as the central feature of Jenkins for building projects using any SCM and build system. A third red circle highlights the 'GitHub Organization' section, which describes it as scanning a GitHub organization for repositories matching defined markers. The 'OK' button at the bottom of the 'GitHub Organization' section is also circled in red.

The screenshot shows the Jenkins configuration interface for a job named "BTM Weekend Job". A yellow box highlights the entire configuration page. A red circle highlights the "General" tab in the top navigation bar. Another red circle highlights the "Advanced..." button in the "General" section. The "Source Code Management" section is also visible.

General

Project name: BTM Weekend 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

Git

Save **Apply**

The screenshot shows the Jenkins job configuration page for 'BTM Weekend Job'. The 'General' tab is selected, indicated by a red circle. Other tabs include Source Code Management, Build Triggers, Build Environment, Build, and Post-build Actions. Under 'General', there are several configuration options:

- Quiet period
- Retry Count
- Block build when upstream project is building
- Block build when downstream project is building
- Use custom workspace
 - Directory: C:\Users\admin\jenkins\AutomationProjectWeekendBatch
 - Display Name: My Automation project path
- Keep the build logs of dependencies

Below the General tab, there are sections for Source Code Management (set to None), Build Triggers (disabled), and Post-build Actions (disabled).

how to schedule suite execution time.

The screenshot shows the 'Build Triggers' configuration section. It includes three trigger types:

- Trigger builds remotely (e.g., from scripts)
- Build after other projects are built
- Build periodically

The 'Build periodically' option is selected and highlighted with a red circle. A 'Schedule' field shows the cron expression '30 23 * * *'. A warning message at the bottom states:

⚠ 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 Weekend Job Config | Jenkins installation and setup | New Tab

localhost:9090/job/BTM Weekend Job/configure

Search

Jenkins > BTM Weekend 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

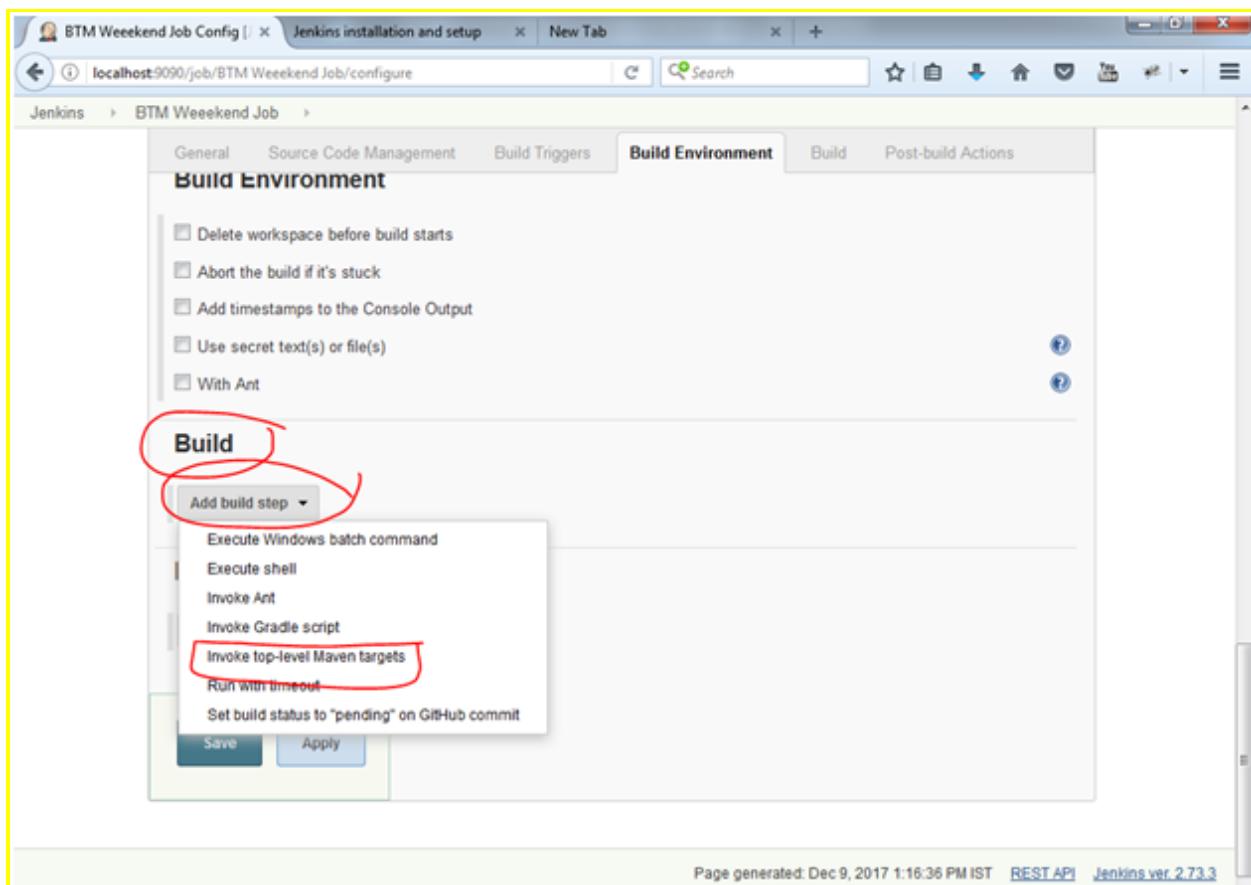
Build

Add build step ▾

- Execute Windows batch command
- Execute shell
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets**
- Run with timeout
- 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



The screenshot shows the Jenkins job configuration interface for a job named "BTM Weeekend Job". The "Build Environment" tab is selected. Under "Build", there is a step titled "Invoke top-level Maven targets" with a red oval highlighting the "Goals" field containing "clean install compile test". The "Post-build Actions" section is also visible.

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

Invoke top-level Maven targets

Maven Version: apache-maven-3.5.2

Goals: clean install compile test

Add build step ▾

Post-build Actions

Save Apply

Save the above

Install testing results plugin

The screenshot shows the Jenkins Manage Jenkins interface. On the left, there's a sidebar with links: New Item, People, Build History, **Manage Jenkins**, My Views, and Credentials. The 'Manage Jenkins' link is circled in red. Below the sidebar, there are two sections: 'Build Queue' (No builds in the queue) and 'Build Executor Status' (1 Idle, 2 Idle). The main content area is titled 'Manage Jenkins'. It displays a message about a new version (2.89.1) available for download. There are two buttons: 'Or Upgrade Automatically' (highlighted with a red circle) and 'Downgrade to 2.73.2'. Below these are several configuration links: 'Configure System', 'Configure Global Security', 'Configure Credentials', 'Global Tool Configuration', 'Reload Configuration from Disk', 'Manage Plugins' (highlighted with a red circle), 'System Information', and 'System Log'. The 'Manage Plugins' link has a tooltip: 'Add, remove, disable or enable plugins that can extend the functionality of Jenkins. (updates available)'.

Navigate to available tab and search Testng Result plugin and install ..

Select the job and click on configure..

Select below and save

My Automation project path × Jenkins installation and setup × New Tab

localhost:9090/job/BTM Weeekend Job/configure Search

Jenkins > My Automation project path >

General Source Code Management Build Triggers Build Environment Build Post-build Actions

Build

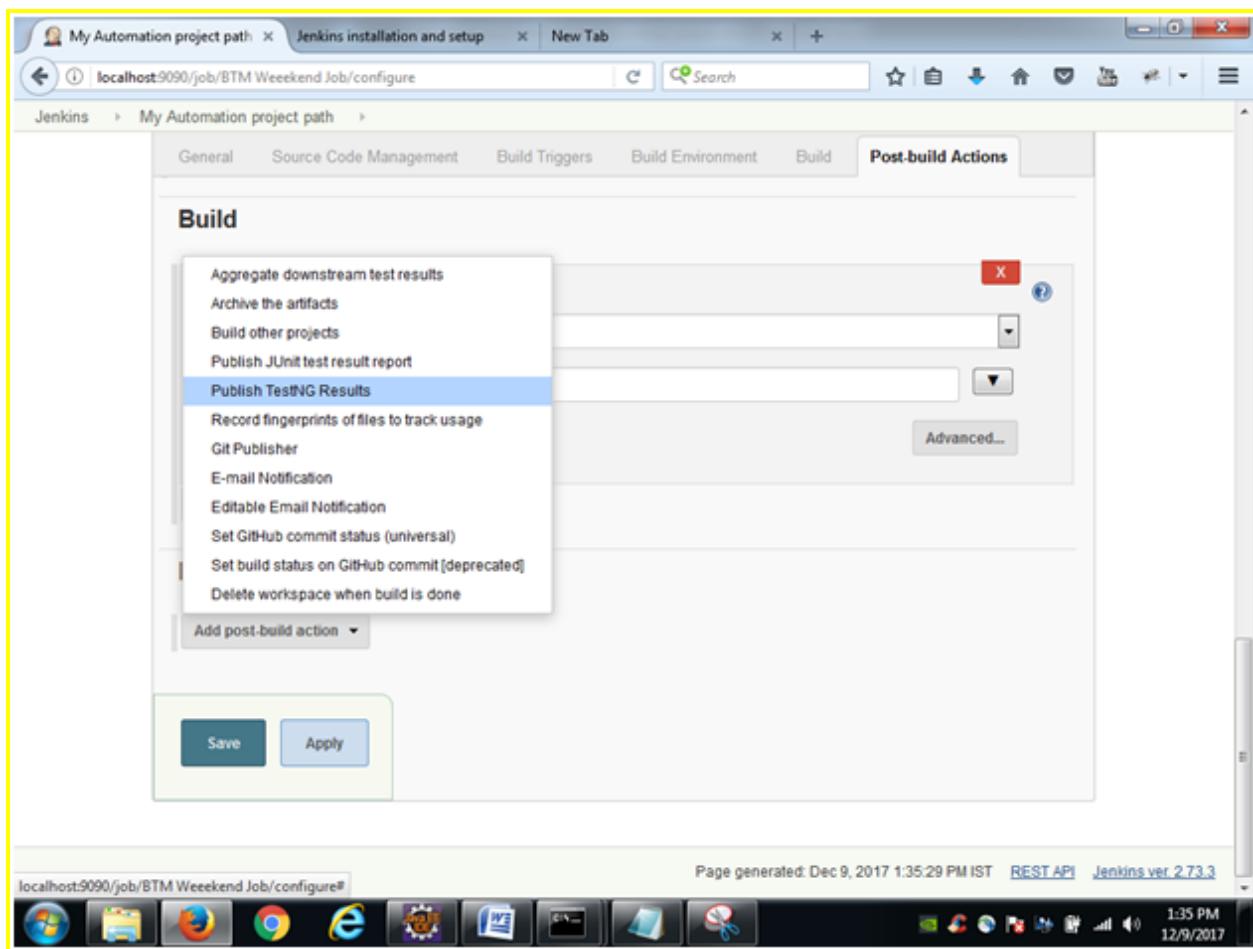
Aggregate downstream test results
Archive the artifacts
Build other projects
Publish JUnit test result report
Publish TestNG Results
Record fingerprints of files to track usage
Git Publisher
E-mail Notification
Editable Email Notification
Set GitHub commit status (universal)
Set build status on GitHub commit [deprecated]
Delete workspace when build is done

Add post-build action ▾

Save Apply

Page generated: Dec 9, 2017 1:35:29 PM IST REST API Jenkins ver. 2.73.3

localhost:9090/job/BTM Weeekend Job/configure 1:35 PM 12/9/2017



My Automation project path × Jenkins installation and setup × New Tab

localhost:9090/job/BTM Weeekend Job/configure Search

Jenkins > My Automation project path >

General Source Code Management Build Triggers Build Environment Build Post-build Actions

Build

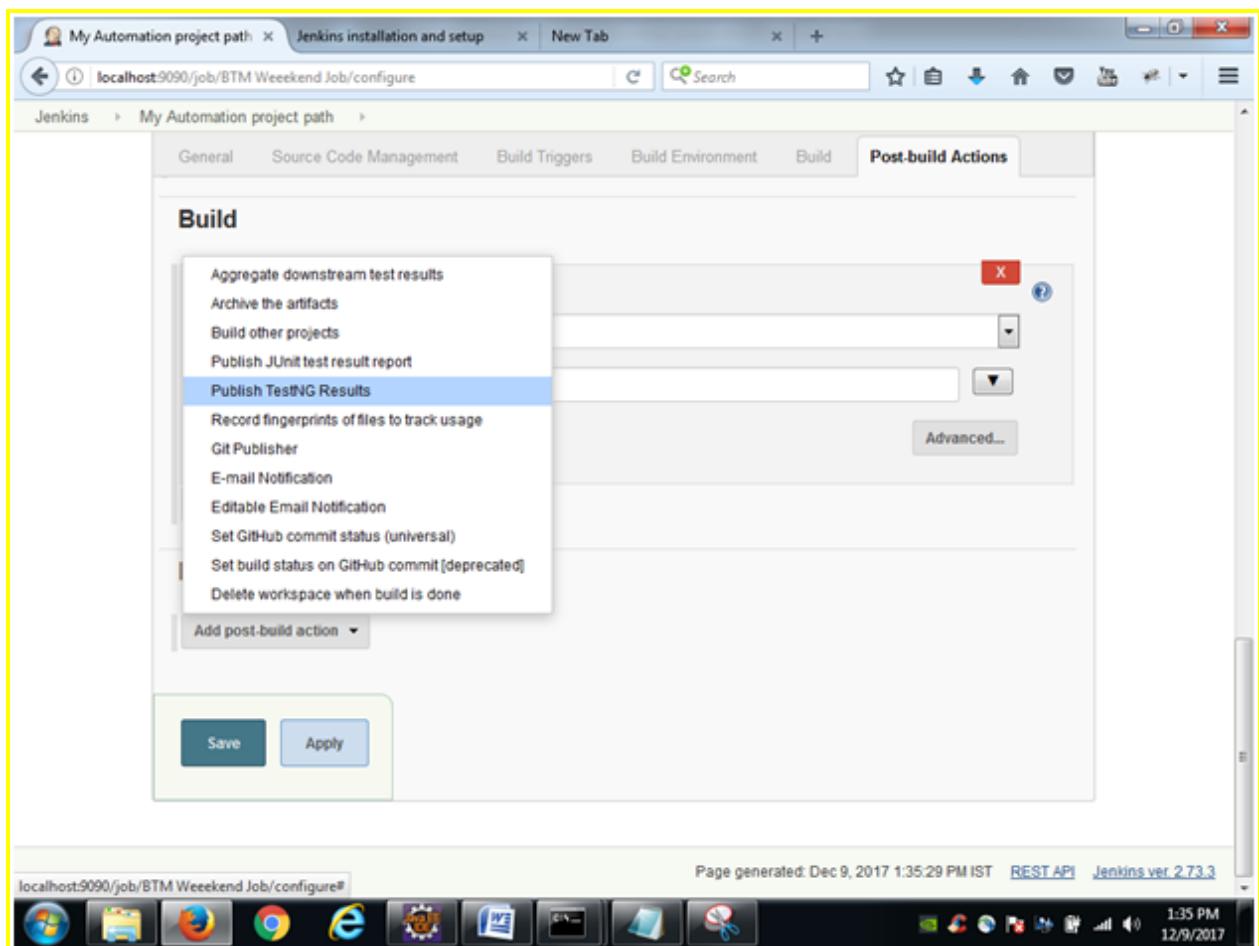
Aggregate downstream test results
Archive the artifacts
Build other projects
Publish JUnit test result report
Publish TestNG Results
Record fingerprints of files to track usage
Git Publisher
E-mail Notification
Editable Email Notification
Set GitHub commit status (universal)
Set build status on GitHub commit [deprecated]
Delete workspace when build is done

Add post-build action ▾

Save Apply

Page generated: Dec 9, 2017 1:35:29 PM IST REST API Jenkins ver. 2.73.3

localhost:9090/job/BTM Weeekend Job/configure 1:35 PM 12/9/2017



The screenshot shows the Jenkins Plugin Manager interface. A search bar at the top right contains the text "testng". Below it, there are tabs for "Updates", "Available", "Installed" (which is selected), and "Advanced". A filter bar also contains the text "testng". The main table lists four plugins:

Enabled	Name	Version	Previously installed version	Uninstall
<input type="checkbox"/>	bouncycastle API Plugin	2.16.2		Uninstall
<input type="checkbox"/>	JUnit Plugin	1.21		Uninstall
<input checked="" type="checkbox"/>	TestNG Results Plugin	1.14		Uninstall

A red oval highlights the "TestNG Results Plugin" row, which has a checked checkbox in the "Enabled" column. The description for this plugin states: "This plugin integrates [TestNG test reports](#) to Jenkins." At the bottom of the page, a footer bar displays the text "Page generated: Dec 9, 2017 1:31:52 PM IST [REST API](#) [Jenkins ver. 2.73.3](#)".

Click on Build now

The screenshot shows the Jenkins web interface for the 'My Automation project path' job. The left sidebar contains links: Back to Dashboard, Status, Changes, Workspace, Build Now (circled in red), Delete Project, Configure, Build History (with a 'trend' dropdown), a search bar, and RSS feeds for all and failures. The main content area displays the project name 'My Automation project path' and its description 'Project name: BTM Weekend Job'. It includes a 'Workspace' link, a 'Recent Changes' link, and a 'Permalinks' section. A 'Disable Project' button is visible in the top right. At the bottom, it says 'Page generated: Dec 9, 2017 1:36:38 PM IST REST API Jenkins ver. 2.73.3'.

Job is running and in progress as shown below.

Project My Automation project path

Project name: BTM Weeekend Job

Workspace

Recent Changes

Permalinks

Page generated: Dec 9, 2017 1:36:38 PM IST REST API Jenkins ver. 1.625

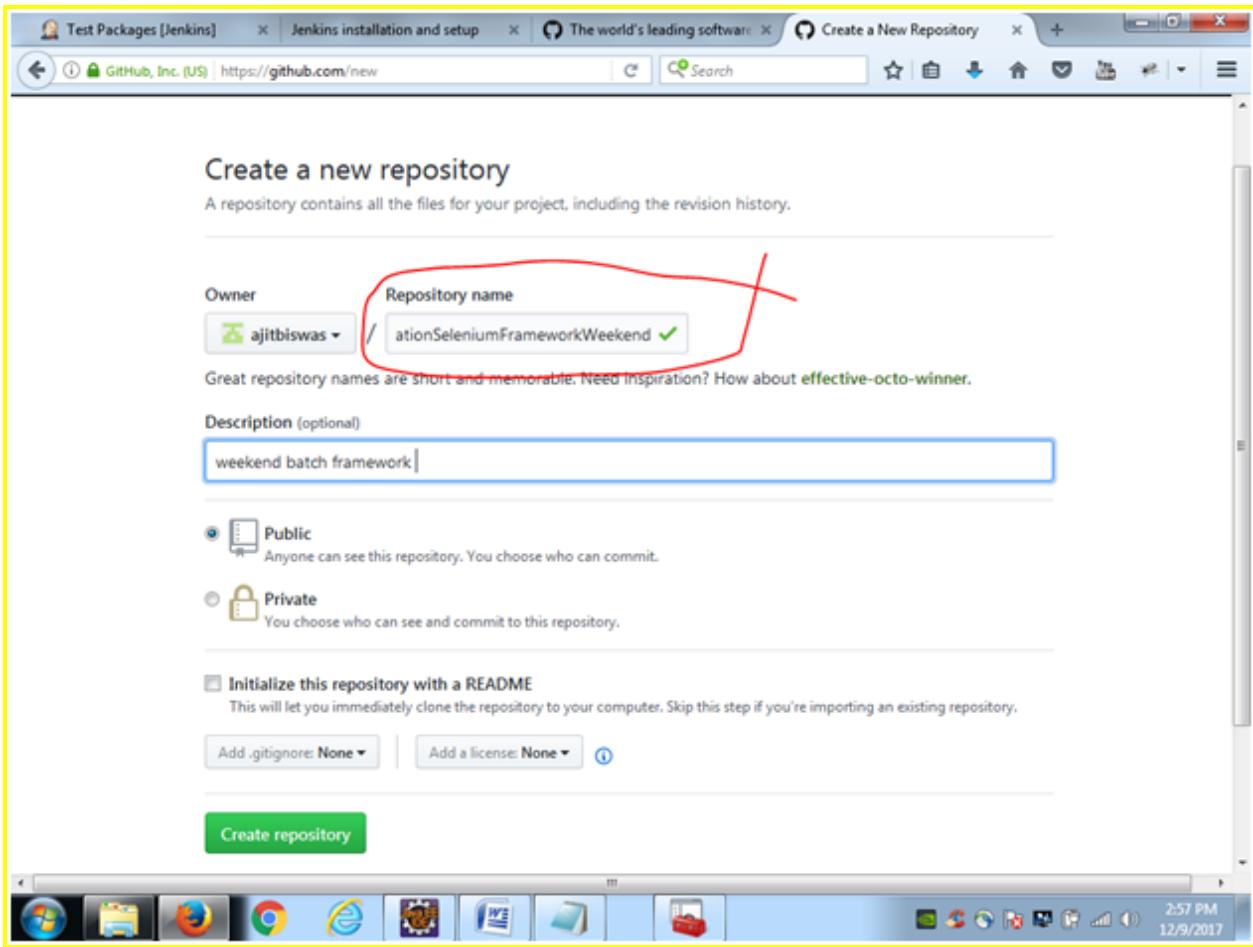
GITHUB SETUP

Go to

www.github.com

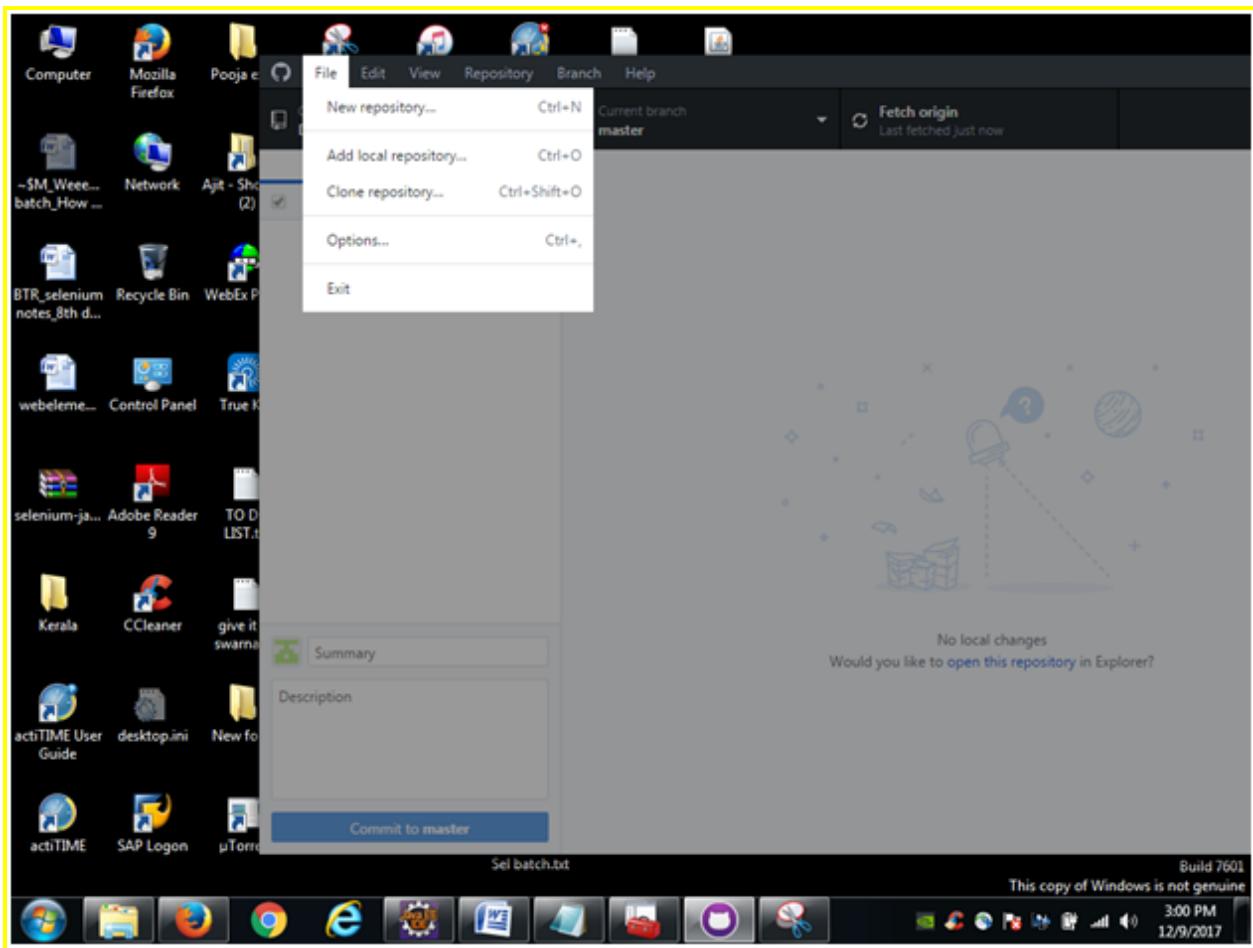
register and sign in

click on New Repository



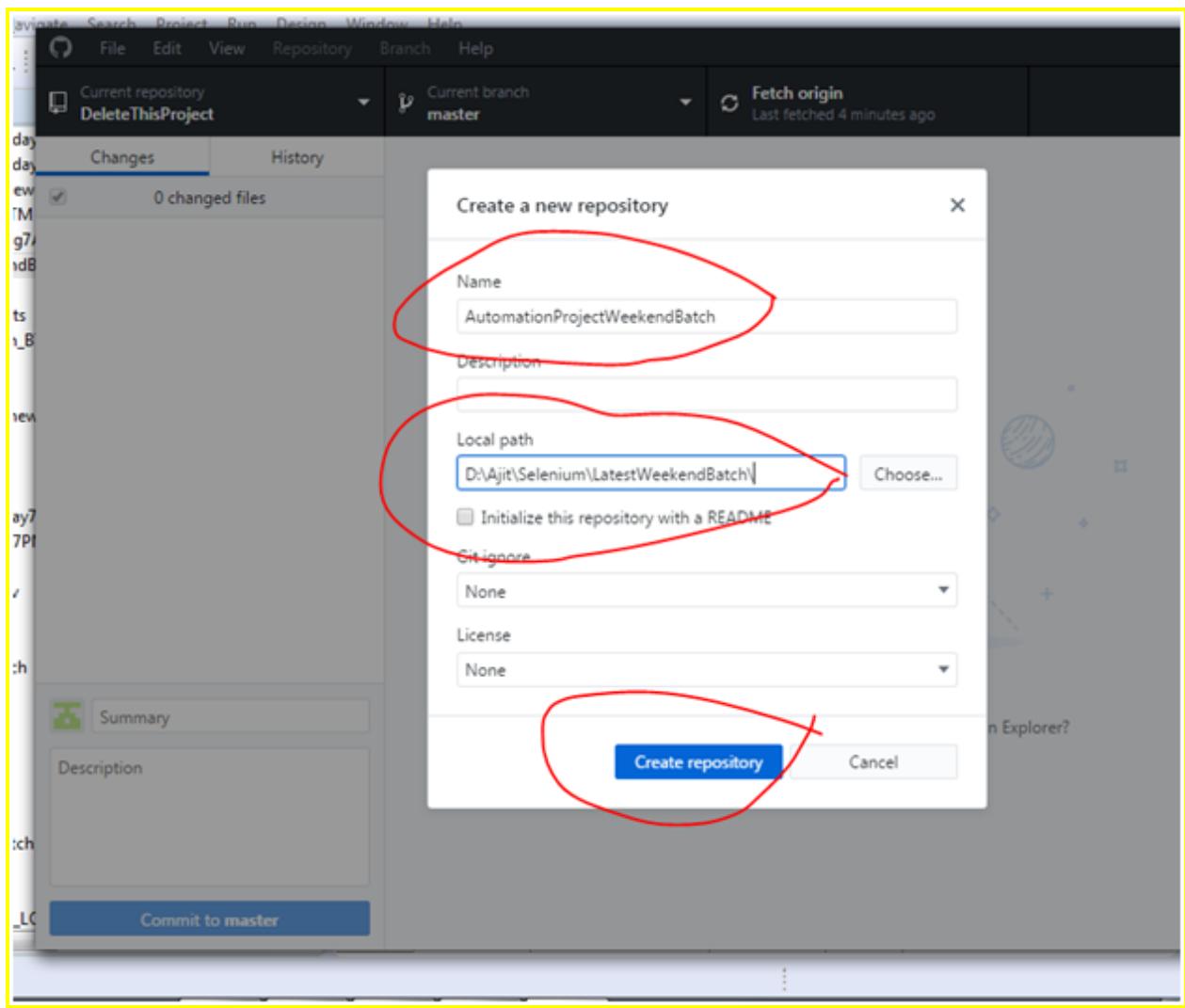
Now launch the **github desktop.exe**

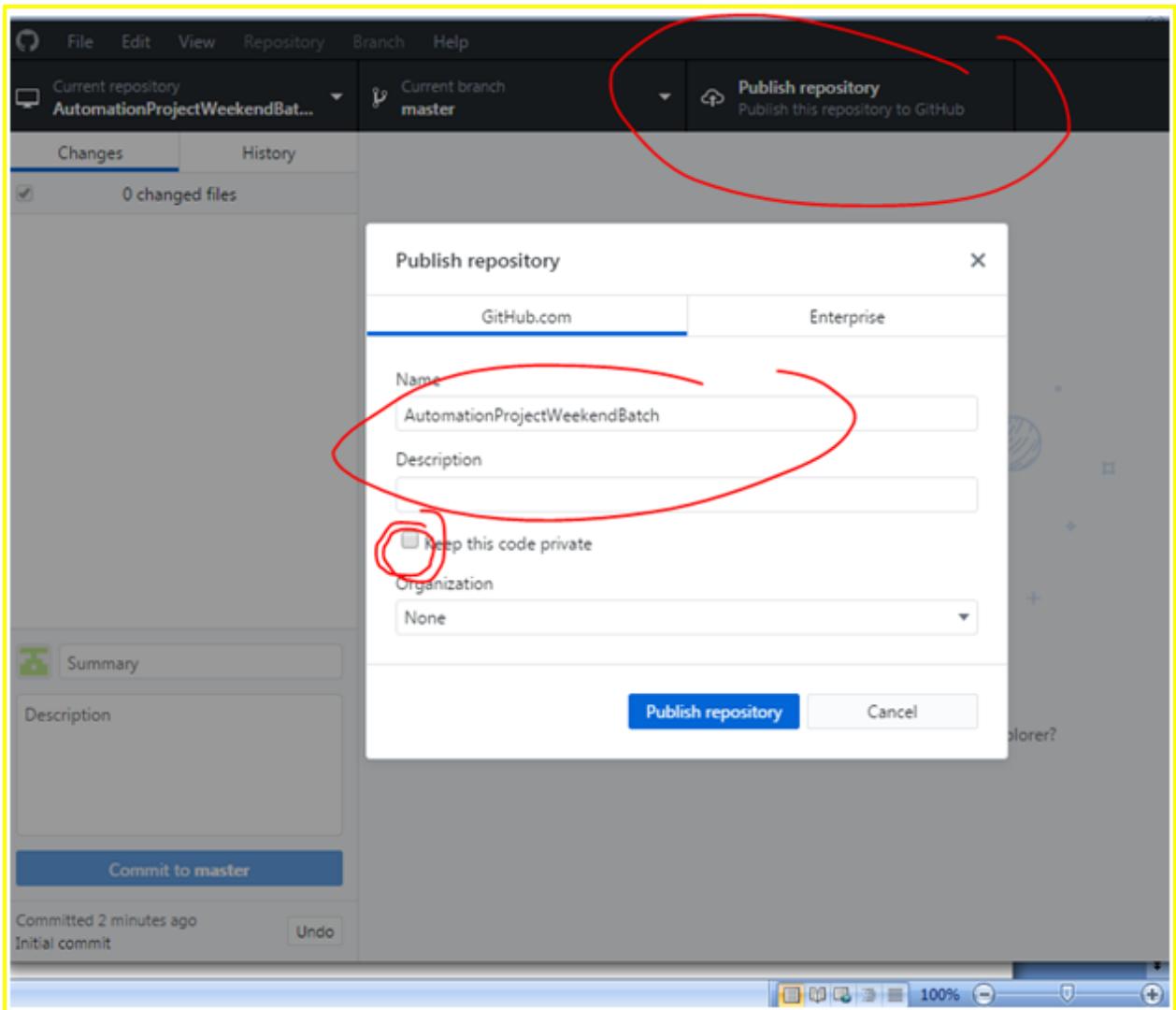
File – New Repository



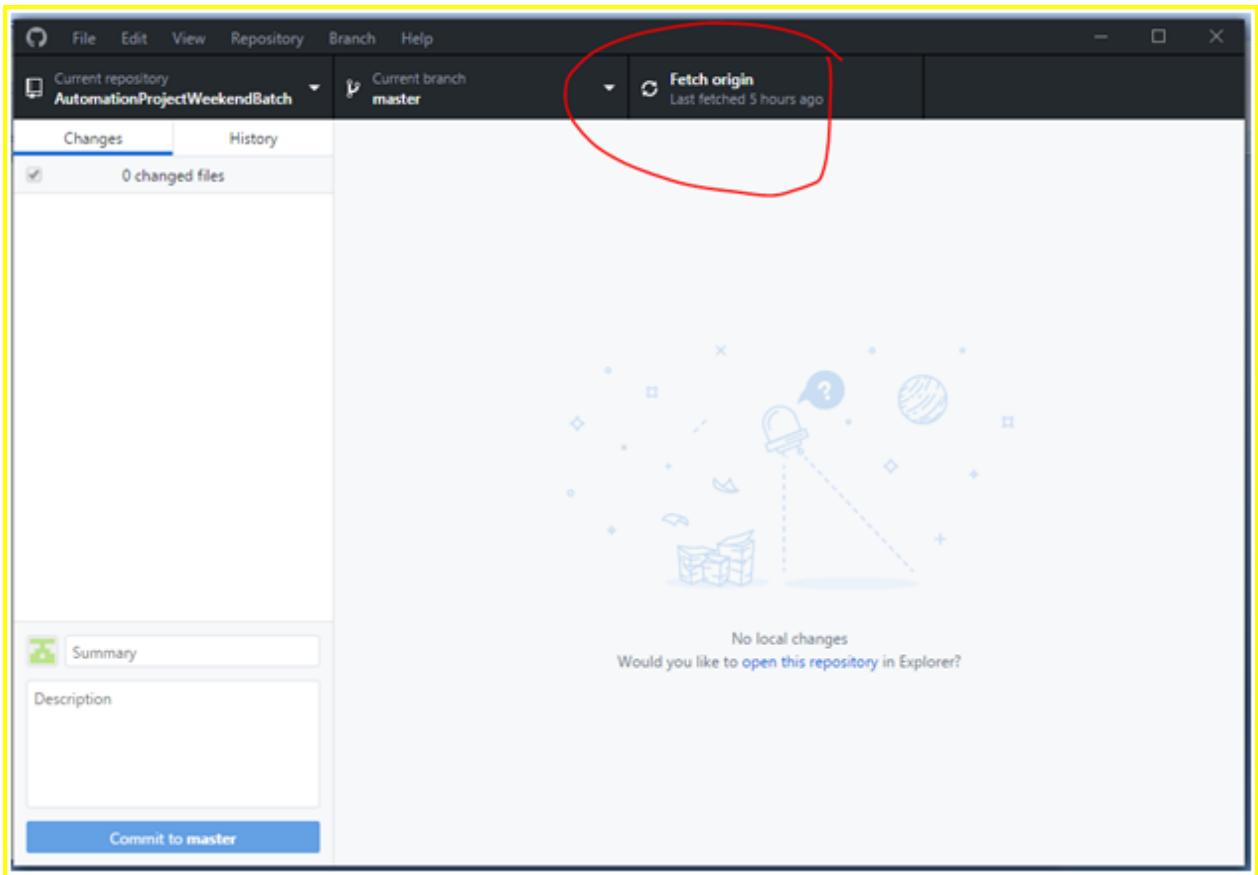
Enter project name which we want to upload under NAME text box

And Local path is the actual workspace where in our project is located.

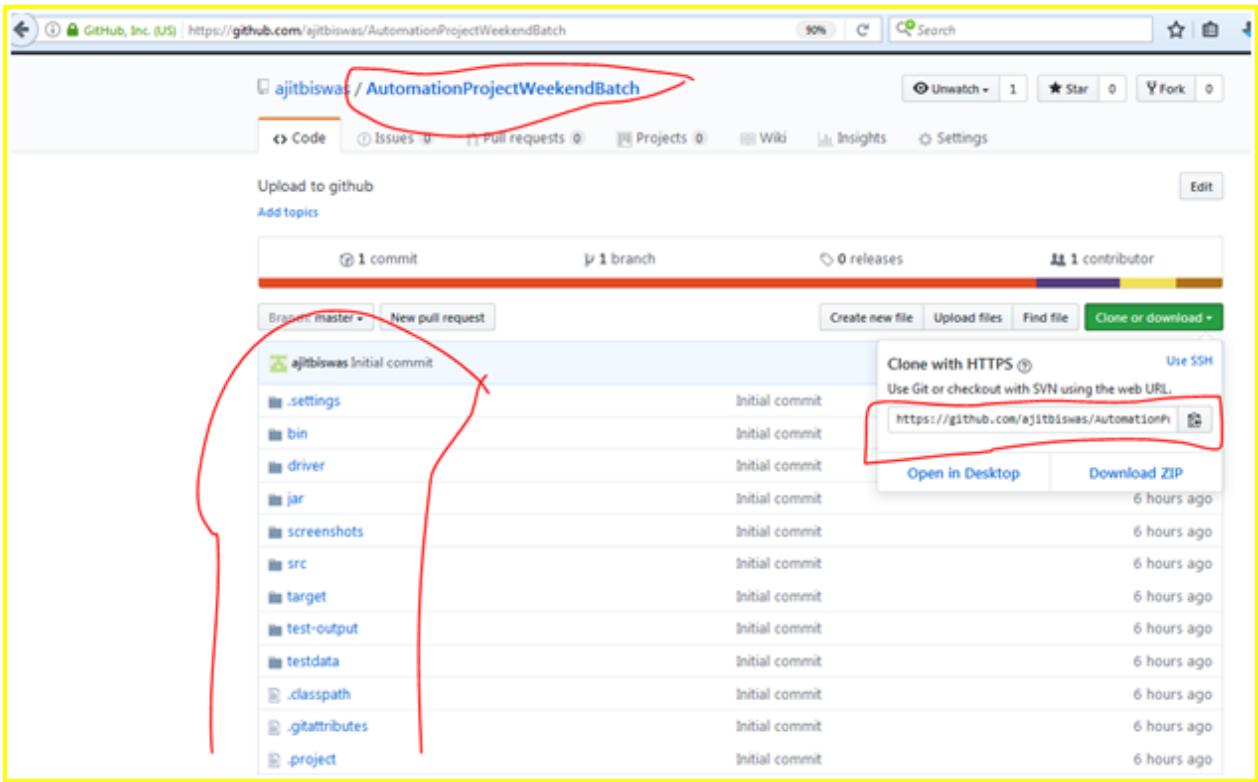




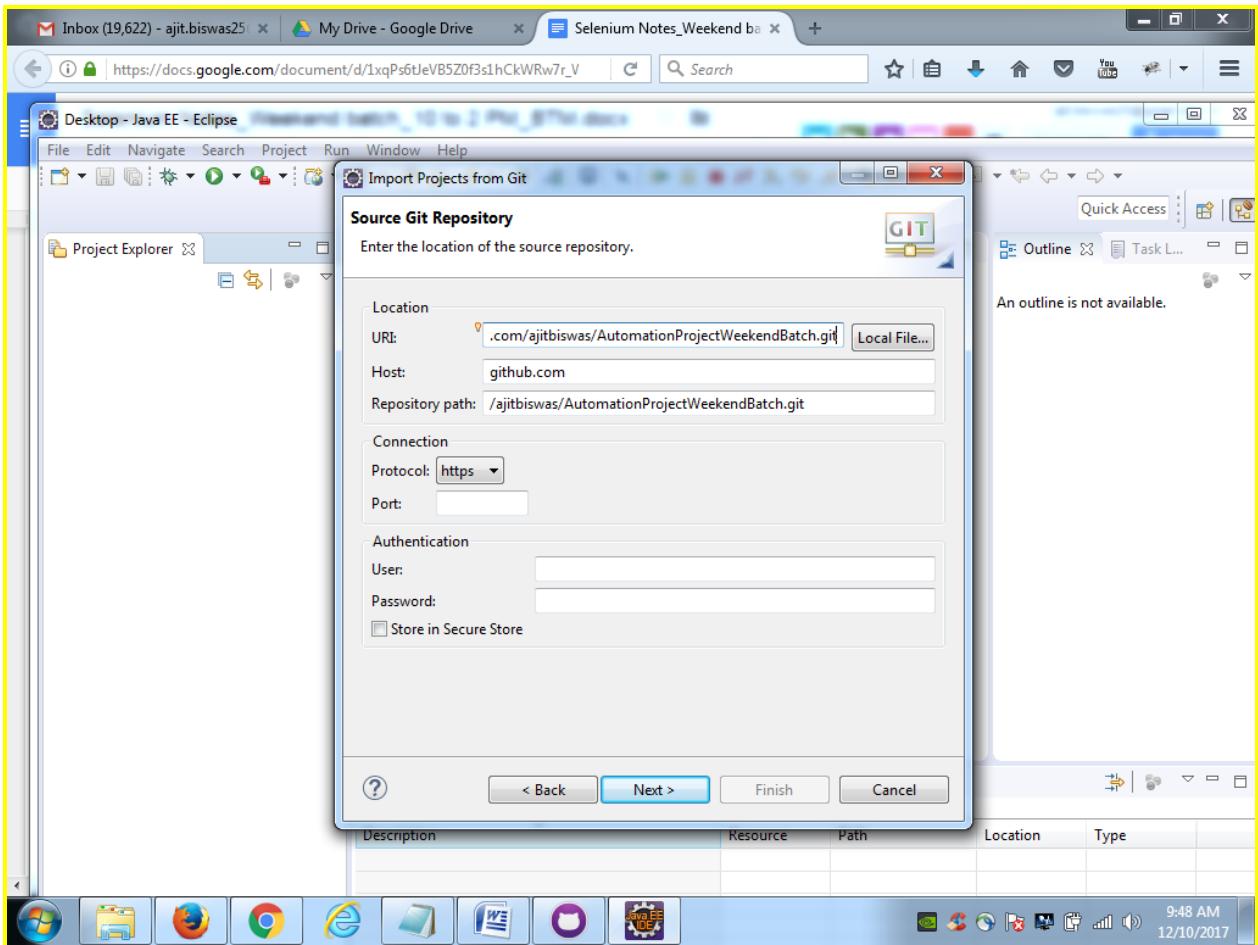
Once it is uploaded successfully, u will get something like this as shown below.



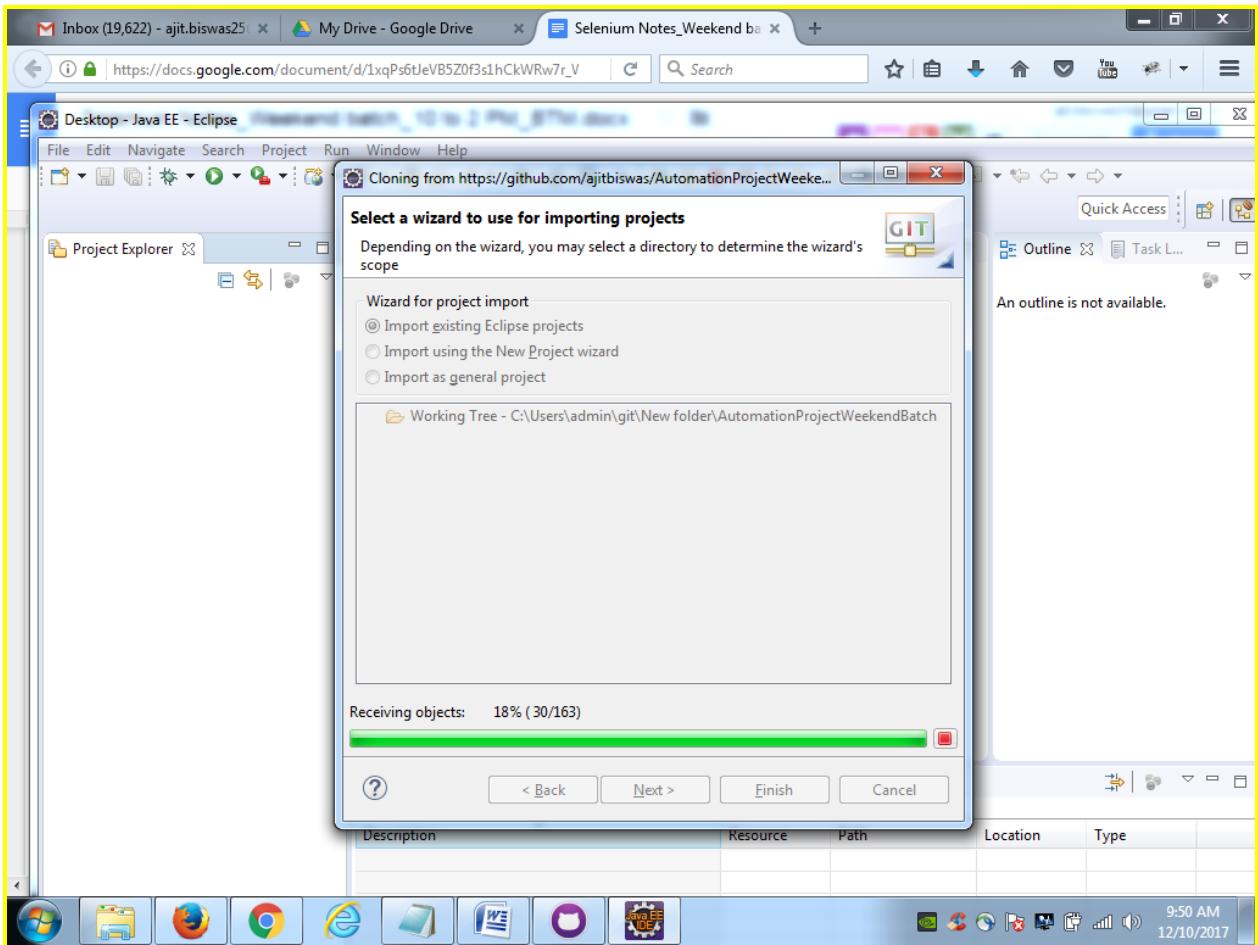
Now, go to the github , you will see that project is successfully uploaded to the central as shown below



Copy the URL above and import the project in Eclipse.

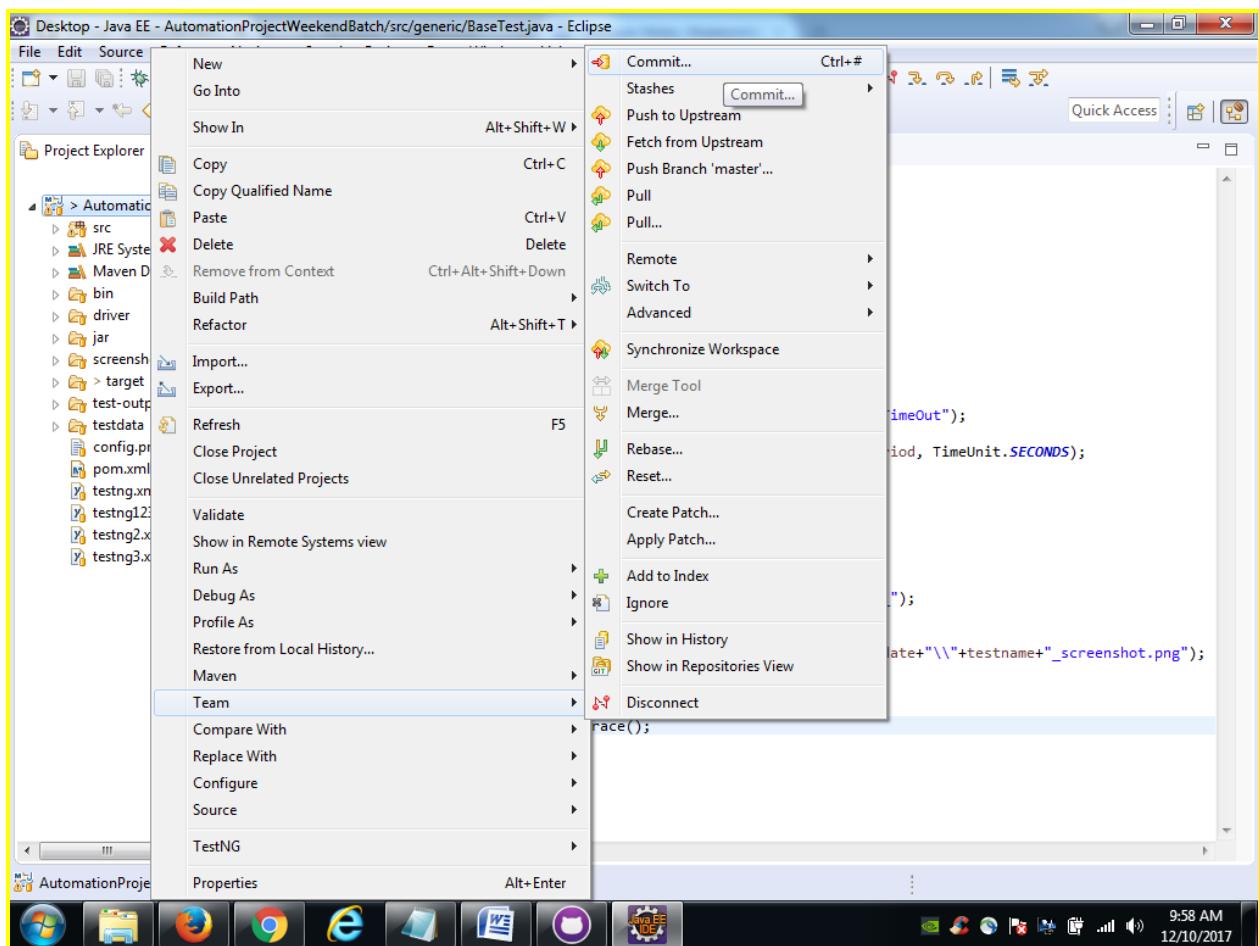


It will start downloading the project from Github to the local system in eclipse.

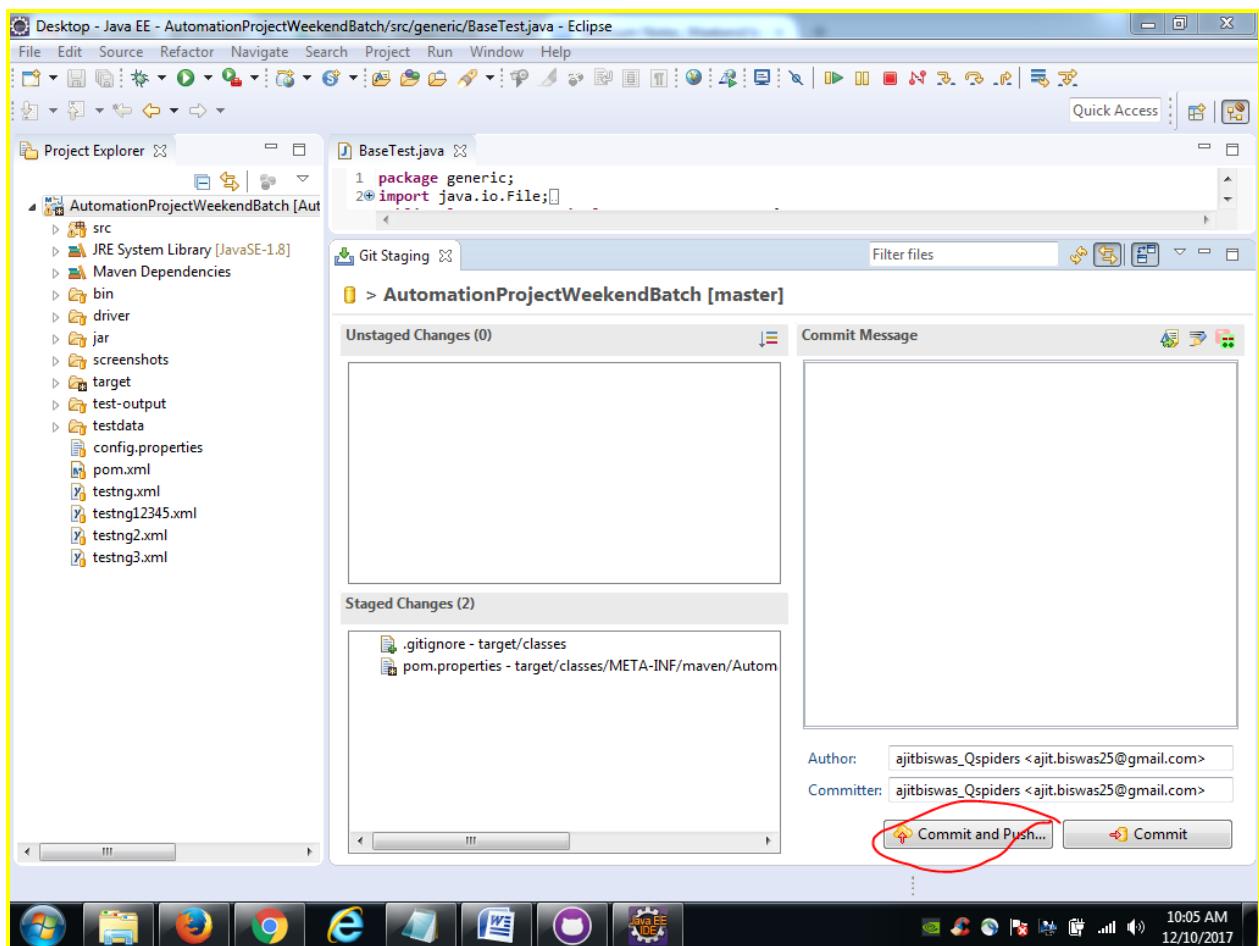


Once imported, we will do some changes and we will upload it back to github by using below navigation.

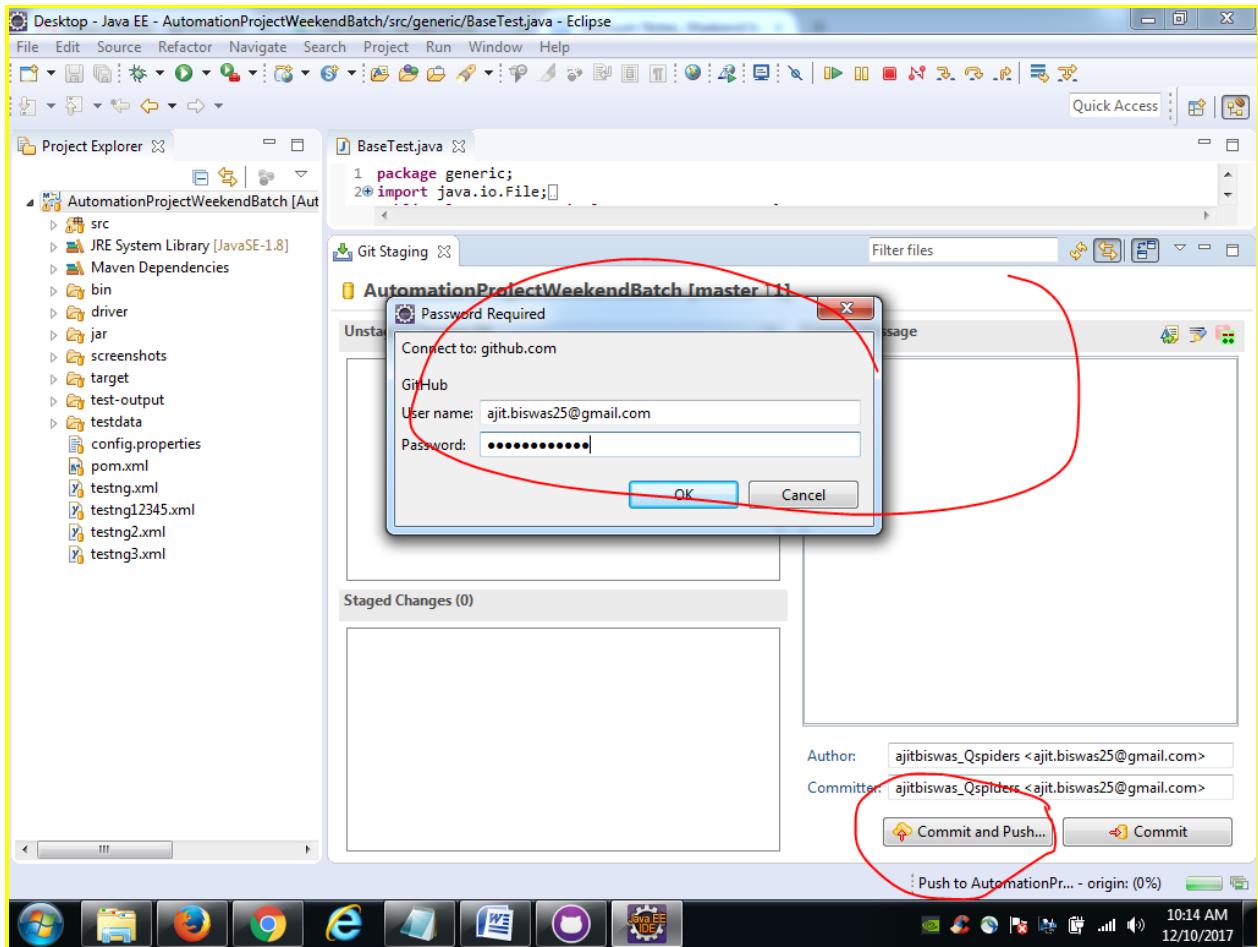
Right click on the project -- TEAM -- COMMIT



now commit and push



Now checkout the latest code from Github and get it in your local system.



LOG4J :

Paste the below settings in log4j2.xml file

```
<?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>

```

Database connection using Selenium

To be continued/.....

List of Exceptions

1. IllegalStateException [Java - Unchecked] (driver exe path not set)
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]
 6. JavascriptException[Selenium-Unchecked][on clicking on a button using submit() and the button don't have an attribute called type='submit']
 7. NoSuchFrameException[Unchecked - selenium] [when specified frame is not present on the webpage]
 8. NoSuchWindowException[Unchecked - selenium] no such window: target window already closed
 9. NoAlertPresentException [Selenium - unchecked] [When no alert is present]
 10. NoSuchElementException: Session ID is null [Unchecked - Selenium] when driver.quit is called and then you are trying to access any browser
-

URL for the automation framework

<https://github.com/ajitbiswas/AutomationProjectWeekendBatch.git>