QSpiders Global
Java_Selenium

INSTRUCTOR: Aman Singh
Subject: Java Selenium

Manual Testing: Testing the software or application manually is called manual testing

Automation Testing: Testing the software or application with the help of any automation tool is called as Automation testing.
Verifying the functionality of any application with the help any automation Tool is called as Automation Testing.
Converting the Manual Test Case into Automation test Script with the help Any Automation Tool is called as Automation testing.

Why we are going for Automation?
To reduce the human effort in the repetitive task we are switching in Automation.
To reduce the time in the repetitive task we are switching in Automation.
To maintain the accuracy and consistency in the repetitive task we are going in Automation.
To sustain in the market competition.
If the product size is big then regression test case will increase to avoid that we are going in Automation.

When We are going for Automation?
If the product or application is functionally stable then we are going for Automation.

Is 100% automation is possible or not?
No 100% automation is not possible

What are things we can't Automate?
Captcha/OTP
Bar code
Gaming Application
Audio And Video Files

Automation Testing Tool present in the market:
Selenium, QTP/UFT, Winium, Appium, Fireflink, Selendriod etc.
Winium is used to automate desktop Application.[Auto IT]
Appium is used to automate mobile application
fireflink is used to perform manual testing, automation testing, api Testing and mobile

testing.

Selenium:

It is free and open source.(See the source code of this particular tool).
It supports many languages.(Java , C# , .net , python , Ruby).
It is script based automation tool.

QTP [Quick Test Profession] [Mercury]

It is paid license tool.
It will support only one language i.e.: VB Script (Visual Basic)
it is Ui based Automation tool.

What is Selenium? Who developed Selenium?
Selenium was developed by **Jason Huggins in 2004** When he was working in Thought Works.
Selenium is an Automation Tool which is used to automate Web-application.

Applications:

Standalone application
(No need internet, no need of server, no database required)
Client-server application
(installation is required, internet is required, server is required, database required)
Web-application.
Internet
Browser
URL

Components/flavours of selenium:
Selenium Core
Selenium RC [Remote Control]
Selenium IDE [Integrated Development Environment]
Selenium Grid
Selenium WebDriver

Selenium Core:
It is the first component of selenium.
It was using java language only.
There is no need of IDE
you can run your test script in command prompt.

Selenium RC - Remote Control
It uses the languages like java, Ruby, c# etc.
If you want to automate your test script you have to run in command prompt
there is no IDE

The main issue with selenium RC is proxy server issue.

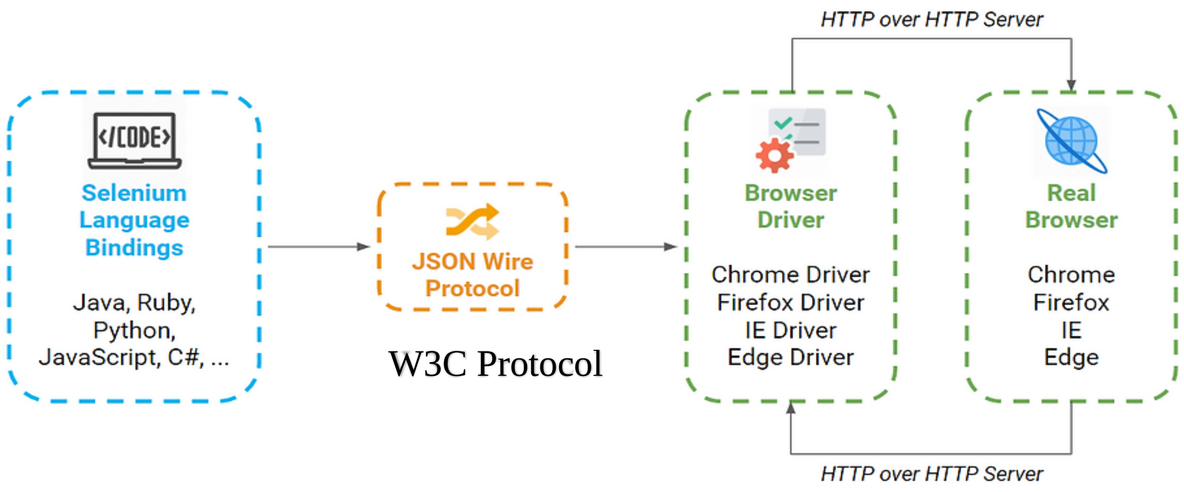
Selenium IDE:
Selenium IDE is record and playback tool.
we can use selenium ide in chrome, Firefox and Microsoft edge.
It does not need any programming language
Debugging is not that much easy.

Selenium Grid:
It is used for compatibility testing or cross browser testing.
we can perform parallel execution with the help of selenium grid.

Selenium WebDriver:
Selenium WebDriver is a script-based automation tool which is used to automate web applications.

Advantages:
It supports multiple programming language.
It supports multiple operating system [Mac, Windows, Linux]
It supports multiple browsers like chrome, Microsoft edge, Firefox, opera mini etc.

Selenium WebDriver Architecture:



Explanation:

- Script Written in any programming language should be bind with selenium this step is called as language binding/client binding.
- Since browser understands only http request. the script is transferred through JSON(W3C) wire protocol which will convert our script into browser understandable language i.e http request.
- Each browser vendors provide their own drivers to communicate with browser.
- When a browser driver receives request, that will be send to browsers in the form of http request.
- Browser performs specified action and it will send back the response in the form of http response.

Installation Steps:

- Create a new java project in eclipse
- Right click on java project and create 2 folders, name it as jars and drivers
- Steps to install jars and drivers:
- Search selenium.dev in chrome.
- Click on downloads in top nav bar.

To download jar file:

- Find Previous releases and click on releases.
- Click on selenium version 3.141.59.
- Click on selenium-server-standalone-3.141.59 jar
- To download driver file
- In the same page click on browsers.
- Click on documentation of respective browser.
- Click on respective browser version.
- Download zip file.(eg: chromedriver_win32.zip)
- Extract the zip file
- Copy the jar file to jars folder in eclipse and add to build path.
- Copy driver.exe file to drivers folder in eclipse.

To Launch an Empty Browser:
Step1:
Set the property of a driver executable file.
We have a method called setProperty(String key,String value) which is present in System class.
System-->It is an inbuilt final class present in java.lang package.
setProperty-->It is a static method present in System class
It takes 2 arguments(key and value in form of String)
Key =Specifies which browser we want to launch(given by browser vendors).
Value=Specifies path of driver executable file.
Step 2:
Create an instance of browser specific class.

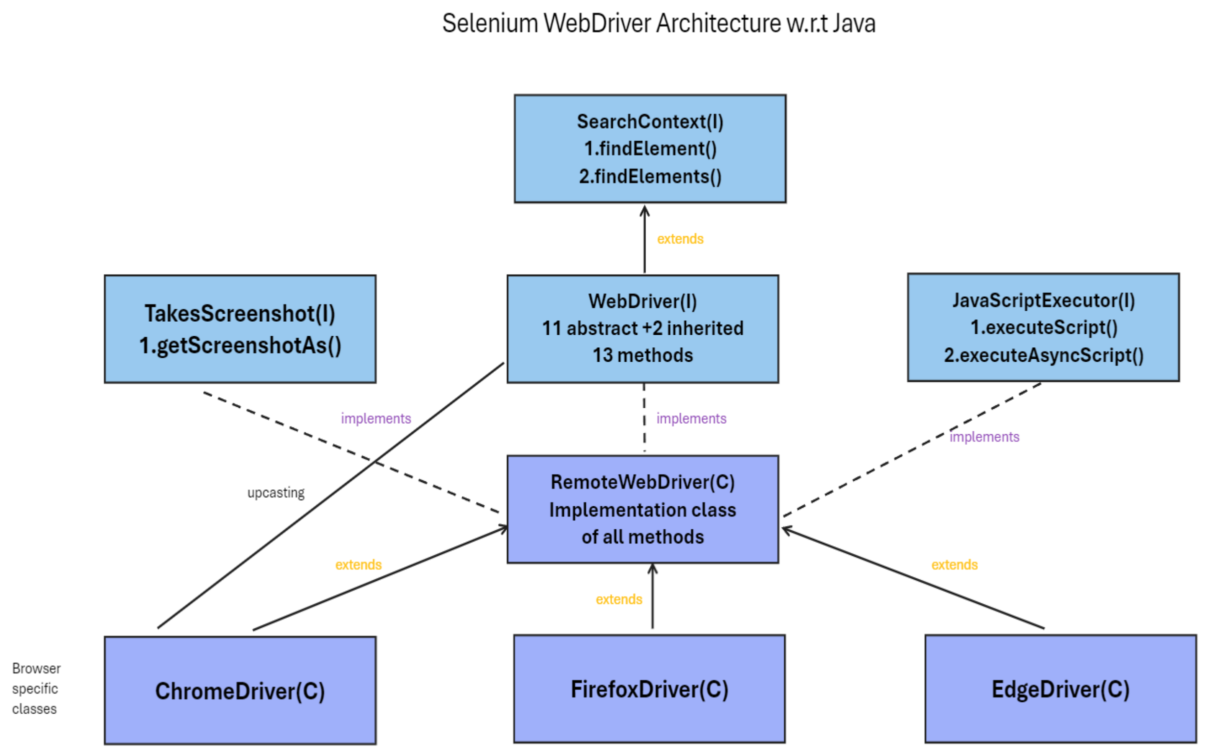
How To launch the browser In Previous Versions:

```
//how to launch the browser in previous versions
        System.setProperty("webdriver.chrome.driver", "./driver/chromedriver.exe");
 //launch the chromeBrowser
         WebDriver driver = new ChromeDriver();
```

How To launch the browser In Updated Versions:

```
//launch the chromeBrowser
  WebDriver driver = new ChromeDriver();
```

Selenium Webdriver Architecture w.r.t java



Explanation:

SearchContext<I>:

- It provides a mechanism to search web element in webpage.
- It provides 2 abstract methods

- findElement(By by)
- findElements(By by)

WebDriver<I>:

- It provides mechanism to perform browser related actions.
- It provides 11 abstract methods(direct) and 2 inherited methods.

TakesScreenshot<I>:

- It provides mechanism to take a screenshot of entire web page.
- It provides 1 abstract method.
- **getScreenshotAs()**

JavaScriptExecutor<I>:

- It provides a mechanism to write a script so that browser understands(JS).
- It provides 2 abstract methods.
- executeScript()
- executeAsyncScript()

  Note: All the implementation of these methods is given in RemoteWebDriver class.
- Respective browser specified classes extends RemoteWebDriver class.

  Best Way to launch the Browser:

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

  If you want to launch Firefox Browser:

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

  If you want to launch Edge Browser:

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

WebDriver Methods:

1. get(String url):

- It is used to navigate to an application.
- Return type is void.

Syntax:

```
//to launch the browser
WebDriver driver = new ChromeDriver();
//to access the web application
driver.get("https://qspidersglobal.com/");
```

2. manage()-->

- Return type is Options(I) type object ref

- It is used for managing a) window()
  b) timeouts()

a)window()--> Return type is Window(I) type object ref
fullScreen()-->used to make browser fullscreen.
maximize()-->used to maximize the browser window.
setSize(Dimension)-->to set size of the browser.
getSize()-->to get height and width of the webpage.
setPosition(Point)-->to set the cursor position.
getPosition()-->to get the cursor position.

How to maximize the browser:

```
//how to maximize the window
 Options opt = driver.manage();
                    Window win = opt.window();
                    win.maximize();
```

Optimized Code:

```
driver.manage().window().maximize();
```

3. getTitle():

- It is used to capture the title of the Web-page.
- return type is String.

Syntax:

```
//launch the browser
WebDriver driver = new ChromeDriver();
//maximize the window
driver.manage().window().maximize();
//launch the web application
driver.get("https://www.amazon.com");
//use getTitle()
String title = driver.getTitle();
System.out.println(title);
```

4.getCurrentUrl()------String

- It is used to fetch or capture the url of current webpage
- return type is String.

Syntax:

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.facebook.com/");
//fetch the source code
String url = driver.getCurrentUrl();
System.out.println(url);
```

5.getPageSource()-----String

- It is used to capture the source code of that particular webpage.
- return type is string

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.facebook.com/");
//fetch the source code
String sourceCode = driver.getPageSource();
System.out.println(sourceCode);
```

6.close()--void

- It is used to close only parent browser.
- return type is void.

Syntax:

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.facebook.com/");
//fetch the source code
String sourceCode = driver.getPageSource();
System.out.println(sourceCode);
//close the browser
driver.close();
```

7. quit()---void

- it is used to close the parent as well as child browser.
- Return type is void.

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://omayo.blogspot.com/");
```

```
//open a window pop up
driver.findElement(By.linkText("Open a popup window")).click();
//quit method
driver.quit();
```

8.switchTo() -------TargetLocator(I)

- it is used to switch to active element, child browser, frames and alert popups.
- Return type is TargetLocator(I).

Syntax:

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get(“https://www.google.com”);
driver.switchTo().activeElement().sendKeys(“Poha”,Keys.ENTER);
```

Question: launch the web application without using get() method?

9.navigate()----- Navigation(I)

- it is used to navigate to a particular web application.
- we can perform various operation:

- back
- forward
- refresh

Syntax:

```
WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("https://www.flipkart.com");
        Thread.sleep(2000);
        // launch the web application by using navigate method
        driver.navigate().to("https://www.amazon.com");
        Thread.sleep(2000);
        // to go back
        driver.navigate().back();
        Thread.sleep(2000);
        // to go forward
        driver.navigate().forward();
        Thread.sleep(2000);
        // to refresh the browser
        driver.navigate().refresh();
        Thread.sleep(2000);
        // close the browser
```

```
driver.close();
```

10.findElement()------ WebElement

11.findElements()----- list<WebElement>

12.getWindowHandle()----String

13.GetWindowHandles()------set<String>

How to set the size of browser?

We can set the size of our browser with the help of **setSize(Dimension target)** which is present inside Window(I)

Syntax:

```
WebDriver driver = new ChromeDriver();
driver.manage().window().setSize(new Dimension(int height, int width));
```

Note: Dimension is a class which is present inside **org.openqa.selenium** package.

How to set the position of browser?

We can set the size of our browser with the help of **setPosition(Point targetSize)** which is present inside Window(I)

Syntax:

```
WebDriver driver = new ChromeDriver();
driver.manage().window().setPoint(new Point(int x, int y));
```

Note: Point is a class which is present inside **org.openqa.selenium** package.

Difference between get(String Url) and navigate()?

get(String url)	navigate()
• 1. It is used to launch the Web application. • 2. It will accept String url as an argument. • 3.We cannot perform backward, forward andRefresh operation in get() • 4.It does not take advantage of browser history.	• 1.It is used to navigate the web application. • 2. It takes the help of to(String url) method which will accept String url as an arguments • 3. With the help of navigate() we can perform operation such as back, forward and refresh. • 4.It takes advantages of browser history.

What is HTML?

HTML: - Hyper Text Markup Language
It is used to develop a web page.
It is made up of tags.
Core Structure of HTML

```
<html>
        <head>
                <title>…..</title>
        </head>
        <body>
        </body>
</html>
```

Tags:

- Paired tag
- Unpaired tag/self closing tag

Tag name:
The very first word after "<" symbol, until space is called as tagname.
ex: <div …..> </div>

Paired tag:

- Tag which has both opening and closing tags is considered as Paired tag.
- They can have child tags , plain text and attributes.

Ex: <p></p>, <h1></h1>, <body></body>   etc.

Unpaired tag:

- Unpaired tags are opened tag and do not have to be closed.
- They are also called as self closing tags.

Ex: <br>, <img> etc.

Attributes:

- It provides additional information about the tag.
- Any key value pair inside opening tag is considered as attribute.

  - Attribute name:- key of an attribute.
  - Attribute value:- value of an attribute.

Ex: <input type="text" id="username"> </input>

Visible Text: The text which is present in b/w the tags is considered as visible text.

Ex: <button type="button">Login</button>

Note: Here **Login** Text is considered as a visible text.

Locators

1. Locators is used to identify or locate the web element present on the webpage

2. Locators are the static methods of By class.

In By class we have 8 static methods i.e Locators only.

We Have 8 type of locators:

- tagName(String tagName)
- id()
- className()
- name()
- linkText()
- partialLinkText()
- cssSelector()

- xpath()

1.tagName()- This method is used to identify the location of particular web element by using tag Name.

tags ----html, head, input.

Syntax: WebDriver driver=new ChromeDriver();

WebElement SearchTextField=driver.findElement(By.tagName(input));

1.sendKeys(String argument): This method is used to pass the value into the particular web element.

- This method is used to pass the value into the particular webelement.
- The return type is void.

Disadvantage of tagName() : if you want to identify the second or nth web element you can't do it with the help of tag name locator

- **2. id(String id attribute value) :** this method is used to identify the location of particular web element by using id attribute value**.**
- **id always unique.**

Syntax: Attribute name--id="small-searchterms"--**Attribute value**

WebElement searchTextField=driver.findElement(By.id("small-searchterms"));

searchTextField.senKeys("books");

3. className(String class Attribute value)- this method is used to identify the location of particular web element by using class attribute value.

class="button-1 search-box-button"

WebElement searchTextField=driver.findElement(By.id("small-searchterms"));

searchTextField.senKeys("books");

WebElement searchButton=driver.findElement(By.className("search-box-button"));

searchButton.click();

InvalidSelectorException : if you pass more than one class name inside the className locator then you will get this exception.

4. name(String name Attribute value): it is used to identify the location of particular web element by using name attribute value**.**

- name will be duplicate.

Syntax:

WebElement username=driver.finElement(By.name("username"));

username.sendKeys("7894561");

5. linkText(String linkText) : this method is used to identify the location of particular web element by using link Text.

- if that particular link is developed by <a> tag then only you can use linkText locator.

Ex: **<a href="https://www.instagram.com">Instagram</a>**

6. partialLinkText(String partialLinkText): this method is used to identify the location of particular web element by using partial link text.

- when the link have dynamic value.
- this locator will work only if <a> is there.

7.cssSelector ()= this method is used to identify the location of particular web element by using css expression.

- we can use css selector for id, name and class Attribute value also.

Syntax:

[AttributeName = 'AttributeValue']

Or

tagName[AttributeName='AttributeValue']

- **Shortcuts in cssSelector:**

  1 ^: this symbol is used to match the starting character of any attribute value.

  Syntax:

  [AttributeName^='Starting character of you attribute value']

  2. *: this symbol is used to match any character of attribute value.

  Syntax:

  [AttributeName*='any character of you attribute value']

- **3 $:** this symbol is used to match ending character of attribute value.

  Syntax:

  [AttributeName$='ending character of you attribute value']

- **Disadvantage ---** we can not use visible text inside css selector then we have to go with xpath.

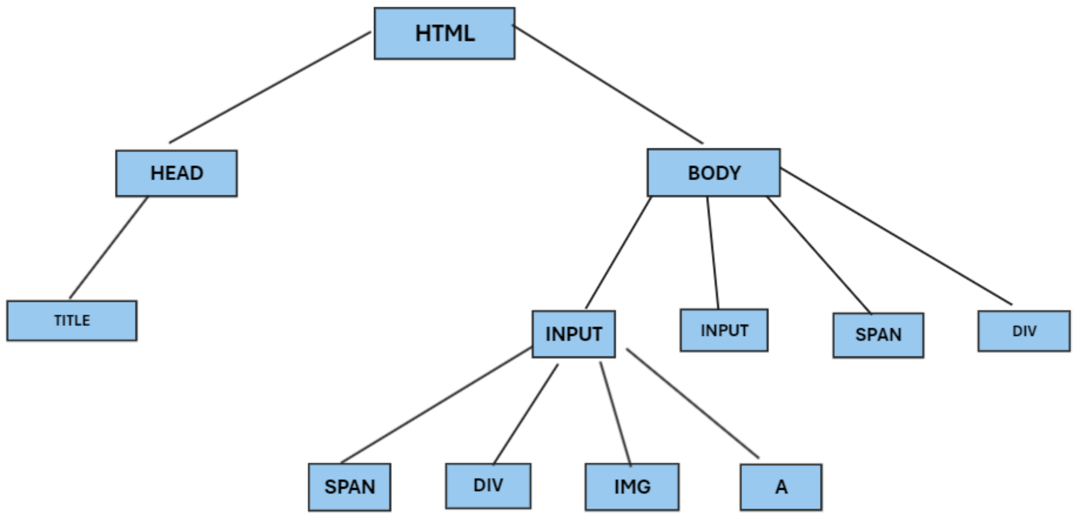
8. Xpath: It is path of web element / HTML Document / Webpage.

We have two types of xpath:

Absolute xpath: It is complete path of an web element.

- It is path which will start from root of the web element(html) and goes till desired web element.

- **/:** we can jump from parent to immediate child.



- **Absolute xpath for input tag:** /HTML/BODY/INPUT[1]

2.Releative xpath: It is path which will start from that particular tagname and it will go till desired web element.

- **//:** We can jump from parent to any child.

Releative Xpath for IMG: //IMG

Advanced Cases of Releative xpath:

1 xpath by Attribute:

//tagName[@AttributeName='AttributeValue']

Example**:** **<input type="text" name=" username"></input>**

//input[@name='username']

@: It is a JavaScript function which will search inside attribute only.

2. Xpath by text:

//tagName [text ()='complete visible text']

Example:

<button type=”button”>Login</input>

//input[text()=’Login’]

text() : It is also a javaScript function which will search inside text only.

3. xpath by contains attribute:

//tagName[contains(@AttributName,'Partial Attribute value')]

4.xpath by contains visible text

//tagName[contains(text(),'partial visible text')]

5.xpath by multiple attribute

//tagName[@AN1='AV1' and @AN2='AV2']

6. xpath by axes :

- **ancestor:** if you want to jump from child to immediate parent or any parent then you can go with ancestor**.**

  //tagName[]/ancestor::tagName[]

- **descendant:**if you want to jump from parent to immediate child or any child then we can go with descendant.

  //tagName[]/descendant::tagName[]

- **following-sibling**: if you want to jump to immediate sibling or any sibling then we can go with following-sibling.

  //tagName[]/following-sibling::tagName[]

- **preceding-sibling**: if you want to jump previous immediate sibling or any previous sibling then we can go with preceding-sibling.

  //tagName[]/preceding-sibling::tagName[]

7. xpath by dependant-independant{Dynamic xpath}:

Step1: Identify the static and dynamic web element.
product name -static
product price-dynamic
Step 2: Identify the static web element.

```
//div[text()='Apple iPhone 15 (Green, 128 GB)']
```

Step 3: Go to the comman parent which will highlight both static and dynamic value.

```
//div[text()='Apple iPhone 15 (Green, 128 GB)']/ancestor::div[@class='yKfJKb row']
```

Step 4: From comman parent you have to jump to the dynamic value.

```
//div[text()='Apple iPhone 15 (Green, 128 GB)']/ancestor::div[@class='yKfJKb
row']/descendant::div[@class='Nx9bqj _4b5DiR']
```

8. **xpath by index:**

```
<input type="text"></input>--username
<input type="text"></input>-- password


(//input[@type="text"]) [2]
```

SearchContext(I) :

Abstract method---

findElement()--------WebElement

- It is used to identify the single web element.
- The return type of this method is Web element.
- If your find element is not able to identify that particular element then you find element will throw one exception
  i.e: NoSuchElement Exception.
- Your find element will return always first matching web element. if you are passing multiple web element.

findElements():---------List<WebElement>

- It is used to identify the multiple webelement.
- The return of this method is list<WebElement>.
- If your findElements() is not able to identify the particular web element then it will not throw any exception it will return **empty list.**
- In findElements() we have to use some looping statement:
  for loop
  forEach loop

WebElement Interface Methods :

1.sendKeys(String value):

- It is used to pass the value inside the particular web element.
- The return type of this method is void.

Syntax:

```
 WebElement username = driver.findElement(By.name(“username”));
                        username.sendKeys(“admin”);
```

2.click():

- It is used to perform click action on any web element.
- Return type is void.

Syntax:

```
WebElement button = driver.findElement(By.id(“loginButton”));
button.click();
```

3.getText():

- It is used to fetch the particular text of particular web element.
- The return type is string.

Syntax:

```
WebElement button = driver.findElement(By.id(“loginButton”));
String text=button.getText();
System.out.println(text);
```

4.clear():

- It is used to clear the present value inside the particular web element.
- the return type is void.

Syntax:

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.flipkart.com");
// identify the search
driver.findElement(By.name("q")).sendKeys("iphone");

Thread.sleep(2000);

// clear the value
driver.findElement(By.name("q")).clear();

// pass the laptop value
driver.findElement(By.name("q")).sendKeys("laptop", Keys.ENTER);
```

5.getAttribute(String Attribute name):

- It is used to fetch the attribute value of particular web element.
- return type is string.

6.getTagName():

- It is used to capture the tag name of that particular web element
- return type is string.

Example :

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.get("https://www.facebook.com/");
Thread.sleep(2000);
WebElement username =
driver.findElement(By.xpath("//input[@name='email']"));
System.out.println(username.getTagName());
String av = username.getAttribute("class");
System.out.println(av);
```

7.getCssValue() :

- Is is used to capture the css property of particular webelement. like color, font-size etc.
- The return type is String.

8.getSize()-

- It is used to capture the dimension(Height, width) of that particular webelement.
- The return type is Dimension.

9. getLocation() –

- It is used to capture X, y co-ordinate of that particular web element.
- The return type is Point.

10.getRect()---

- if you want to capture height, width, x and y coordinates separately.
- The return type is Rectangle.

11.isEnabled() :

- This method is used to check whether that particular web element is enabled or not.
- The return type is boolean.

12. isSelected():

- This method is used to check whether that particular web element is selected or not
- The return is boolean.

13. isDisplayed():

- This method is used to check whether that particular web element is displayed or not.
- The return type is boolean.

14. submit():

- It is used to click on submit button.

Note: If we have type=”Submit” attribute is there on that particular web element then only submit method will work.

Synchronization Handling:

To get Proper automation test script the web application speed and test script speed should properly match this process is called as Synchronization Handling.

To Handle the synchronization, we have two ways:

1. Static wait
2.Dynamic wait

In Static wait we have one wait:

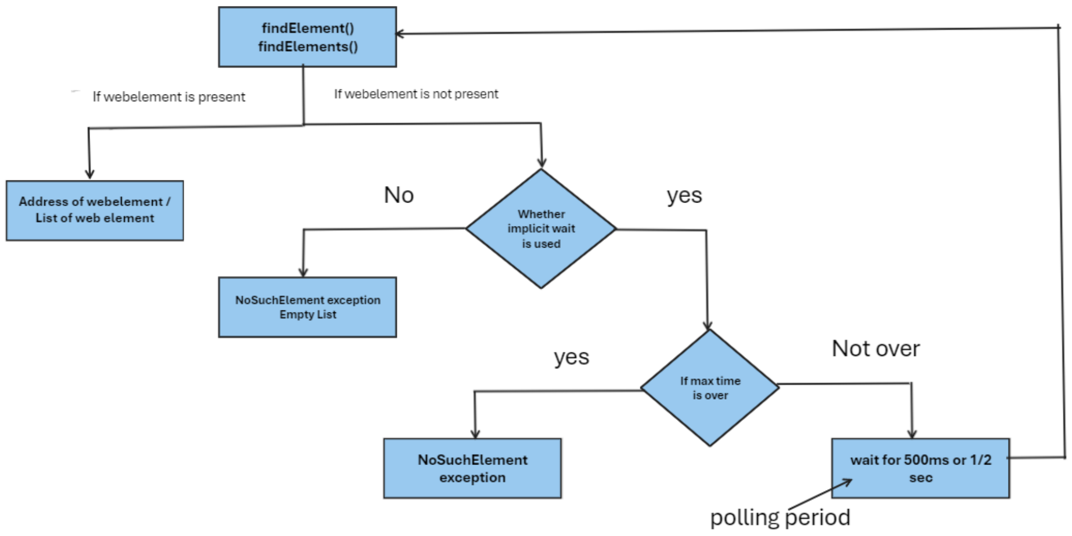
Thread.sleep(time seconds): It is a java wait. It will stop the execution for a particular period of time because of that it is also dead wait.

Drawbacks of Thread.sleep():
1. It means the code become more lengthy.
2. It reduces the execution speed.
3. We can't trust on this wait 100% percent.

In Dynamic wait we have two types:

1. implicit wait
2. explicit wait

Implicit wait:


findElement()
findElements()
If webelement is present
If webelement is not present
Address of webelement / List of web element
No
Whether implicit wait is used
yes
NoSuchElement exception Empty List
yes
If max time is over
Not over
NoSuchElement exception
wait for 500ms or 1/2 sec
polling period

- When the control comes to findElement() or findElements() it tries to search for the webelement or webelements.
- If the webelement or webelements are present on web page it will give address of webelement or webelements
- But if the webelements or webelements are not present on web page the presence of implicit wait will be checked.
- If implicit wait is not used then we will get NoSuchElementException or

Empty List.

- But implicit wait is used, then it will check whether the max time is over or not.
- If max time is over it will give NoSuchElementException or Empty List.
- But if max time is not over then, it will wait for 500ms or ½ sec. which is known as polling period & control will again go to findElement() or findElements().

Syntax:

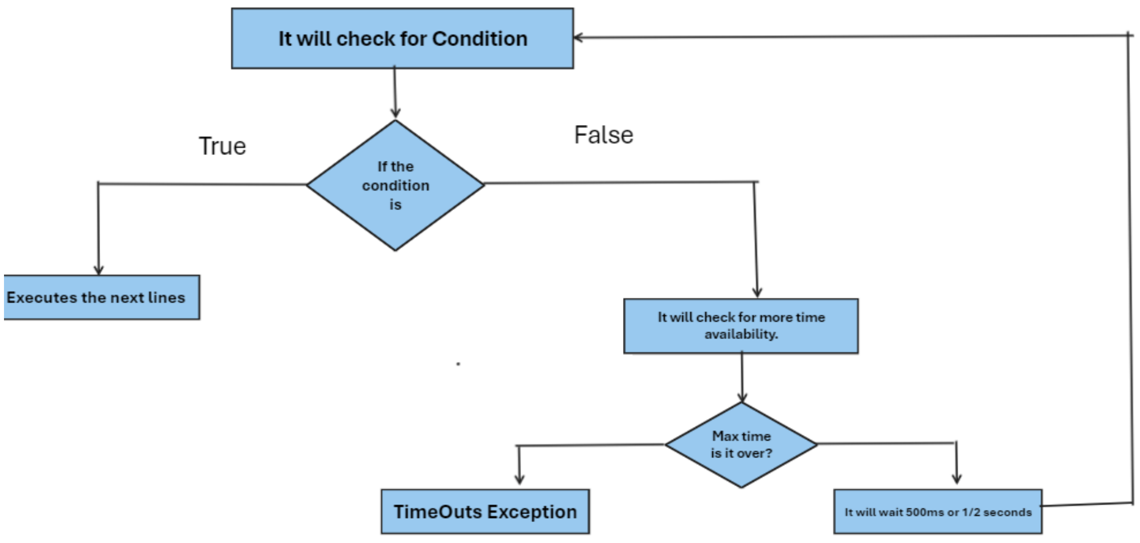
driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));

Note: Implicit wait can only be used for findElement() and findElements().

Disadvantages:

It can only be used for findElement() or findElements().

Explicit wait:


It will check for Condition
True
False
If the condition is
Executes the next lines
It will check for more time availability.
Max time is it over?
TimeOuts Exception
It will wait 500ms or 1/2 seconds

- If the condition is true the next line of the code will be executed and if the condition is false then if will check for maximum time.
- If max time is over, it will give **Timeouts Exception** but if the maximum time is not over, it will wait for ½ sec and again retries for condition to become true.

Syntax:

```
WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(20));
wait.until(ExpectedConditions.elementToBeClickable(WebElement element));
```

Note: Explicit wait works for condition we can use explicit wait for any method including findElement() & findElements().

Question: Difference between Implicit wait and explicit wait.

Implicit wait	Explicit wait
1. It is used to synchronize findElement() & findElements().	1. It is used to synchronize any methods.
2.No need to mention any condition	2.We should mention the condition as per the situation.
3.In implicit wait if the element / list of elements is not found with max time, it gives NoSuchElementException.	3.In explicit wait if the element / list of elements are not found with max time, it gives TimeOutException.
4. Implicit wait is kind of global wait, because we can mention implicit wait once at the top of the code which works for every findElement() and findElements().	4. Implicit wait is kind of local wait, because we have to mention it before every Synchronization issue.
5. We don’t Have to create any object.	5. We don’t Have to create any object.

Handling Dropdowns:

- Dropdown:
  Dropdown is a list of different web elements present in a single area.

  Types of Dropdown:

  Static Dropdown:

- Static dropdown are designed with the help of select tag.
- Options will be given with the help of option tag.
- The options in static dropdown are fixed.

Static dropdown have two types:

1. single-select Dropdown
2. multi-select Dropdown

Dynamic Dropdown:

- In dynamic dropdown options are not fixed.
- Dynamic dropdowns are designed with the help of tag such as div, li, span, div.

Static Dropdown:

1. Single select dropdown:

- The options present in single select dropdown are not scrollable.
- The options which is selected in a single select dropdown can not be deselected.
- In single select dropdown we can select only one option at a time.
- By default one option will be selected in single select dropdown.

2. Multi select dropdown:

- In multi select dropdown the options can be scrolled.
- We can select multiple options in multiselect dropdown.
- In multiselect dropdown we can deselect the options
- No option will be selected by default.

Note: To design a multi select dropdown we have to use an attribute called as multiple.
<select id=”name” multiple>

**********************Selection Methods**************************************

1. selectByIndex(int index) ------ void
2. selectByValue(String value attribute)-----void
3.selectByVisibleText(String visible text)---------void

***********************Deselection Methods***********************************

5. deselectByIndex(int index) ------ void
6. deselectByValue(String value attribute)-----void
7. deselectByVisibleText(String visible text)---------void
8. deselectAll()-------void

***********************Operational Method************************************

9. isMultiple() ------boolean
10. getOptions()------List<WebElement>
11. getFirstSelectedOption------WebElement
12. getAllSelectedOptions()-----List<WebElement>
13. getWrappedElement()---------WebElements

To handle the dropdown we need to create the object of Select class.

How to use Select class:

Step:1 : Identify the dropdown and store it in a variable.

EG → WebElement dropdown= driver.findElement(By.id("dropdown"));

Step 2 : Create an object of Select class

Select sel = new Select (WebElement dropdownElement);

Step 3 : Use the following methods to select options from the dropdown

1. selectByIndex(int index)

This method is used to select the option by using index value of the option from the dropdown. The Return type is void.

Example : WebElement dropdown= driver.findElement(By.id("dropdown"));
sel.selectByIndex(1);

2. selectByValue(String Value)

This method is used to select the option by using value attribute of the option from the dropdown.
The Return type is void.

Example : WebElement dropdown= driver.findElement(By.id("dropdown"));
sel.selectByValue("v4");

3. selectByVisibleText(String visibleText)

This method is used to select the option by using the text of the option from the dropdown.
The Return type is void.

Example : WebElement dropdown= driver.findElement(By.id("dropdown"));
sel.selectByVisibleText ("Karnataka");

4. deselectByIndex(int index)

This method is used to deselect the option by using index value of the option from the dropdown.
The Return type is void.

Example : WebElement dropdown= driver.findElement(By.id("dropdown"));
sel.deselectByIndex(1);

5. deselectByValue(String Value)

This method is used to deselect the option by using value attribute of the option from the dropdown.
The Return type is void.

Example : WebElement dropdown= driver.findElement(By.id("dropdown"));
sel.deselectByValue("v4");

6. deselectByVisibleText(String visibleText)

This method is used to deselect the option by using the text of the option from the dropdown.
The Return type is void.

Example : WebElement dropdown= driver.findElement(By.id("dropdown"));
sel.deselectByVisibleText ("Karnataka");

7. deselectAll() :

This method is used to deselect all the selected options from the dropdown.
The Return type is void.

Example:

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
driver.get("https://demoapps.qspiders.com/ui/dropdown/multiSelect?sublist=1");

// identify the multi-select dropdown
WebElement multiSelectDropdown = driver.findElement(By.id("select-multiple-native"));

// create the object of select class
Select s = new Select(multiSelectDropdown);

// selecting the first three options
for (int i = 0; i <= 3; i++) {
s.selectByIndex(i);
Thread.sleep(2000);

}
// deselect all the options
s.deselectAll();
```

8. isMultiple() :

This method is used to check whether the dropdown is single select or multi-select
If it is a multi-select dropdown then the output will be true.
Return type is boolean.

Example :
```
WebElement dropdown= driver.findElement(By.id(“dropdown”));
boolean status=Dropdown.isMultiple();
sopln(status);
```

9. getOptions() :

This method is used to get all the options from the dropdown.
Return type is List<WebElement>;

Example :

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
driver.get("https://demoapps.qspiders.com/ui/dropdown/multiSelect?sublist=1");

// identify the multi-select dropdown
WebElement multiSelectDropdown = driver.findElement(By.id("select-multiple-native"));

// create the object of select class
Select s = new Select(multiSelectDropdown);

boolean value = s.isMultiple();
System.out.println(value);


//print all the options
List<WebElement> allOptions = s.getOptions();

//			for(WebElement option:allOptions)
//			{
//				System.out.println(option.getText());
//			}

for(int i=0;i<allOptions.size();i++)
{
System.out.println(allOptions.get(i).getText());
}

}
```

10. getFirstSelectedOption() :

This method is used to get the First Selected Option from among all the Selected Options.
Return type is WebElement.

Example:

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
driver.get("https://demoapps.qspiders.com/ui/dropdown/multiSelect?sublist=1");

// identify the multi-select dropdown
WebElement multiSelectDropdown = driver.findElement(By.id("select-multiple-native"));

// create the object of select class
Select s = new Select(multiSelectDropdown);


for(int i=0;i<3;i++)
{
s.selectByIndex(i);
}

WebElement firstSelectedOptions = s.getFirstSelectedOption();
System.out.println(firstSelectedOptions.getText());
```

11. getAllSelectedOptions() :

This method is used to get all the selected options from the dropdown.
Return type is List<WebElement>.

Example :

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
driver.get("https://demoapps.qspiders.com/ui/dropdown/multiSelect?sublist=1");

// identify the multi-select dropdown
WebElement multiSelectDropdown = driver.findElement(By.id("select-multiple-native"));

// create the object of select class
Select s = new Select(multiSelectDropdown);


for(int i=0;i<3;i++)
{
s.selectByIndex(i);
}
```

```
List<WebElement> allSelectedOptions = s.getAllSelectedOptions();

for(WebElement options:allSelectedOptions)
{
System.err.println(options.getText());
}
```

12. getWrappedElement():

This method is used to get all the options from the dropdown together.
Return type is WebElement.

Example :

```
WebDriver driver = new ChromeDriver();
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
driver.get("https://demoapps.qspiders.com/ui/dropdown/multiSelect?sublist=1");

// identify the multi-select dropdown
WebElement multiSelectDropdown = driver.findElement(By.id("select-multiple-native"));

// create the object of select class
Select s = new Select(multiSelectDropdown);

WebElement allOptions = s.getWrappedElement();

System.out.println(allOptions.getText());
```

Actions Class:

To perform mouse related actions we will be using Actions class and its methods.

Actions class has following methods:

1.moveToElement(WebElement element)
2.contextClick(Webelement element)
3.doubleClick(Webelement element)
4.dragAndDrop(Webelement source, Webelement target)
5.clickAndHold(Webelement element)
6.release()
7.perform()
8.click(webelement element)

1.moveToElement():

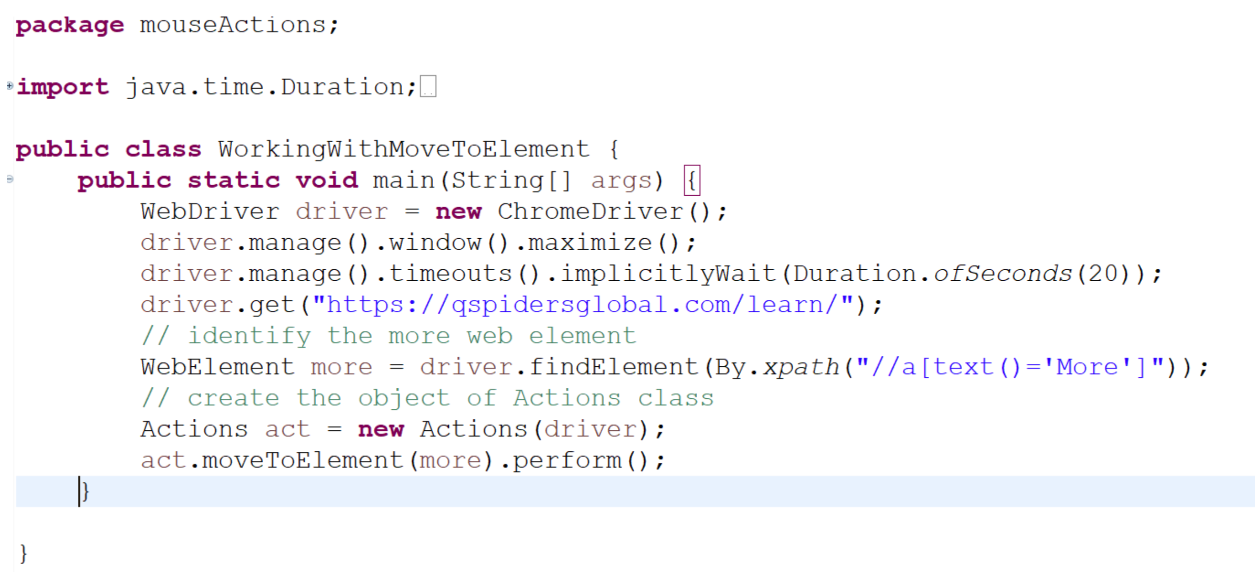
- It is used to perform the mouse hovering actions.
- Return type is void.

2. perform():

- It is used to perform that particular action.
- Return type is void.

Note: It is a compulsory method and if you are not using this method that particular action will not work.

Example:

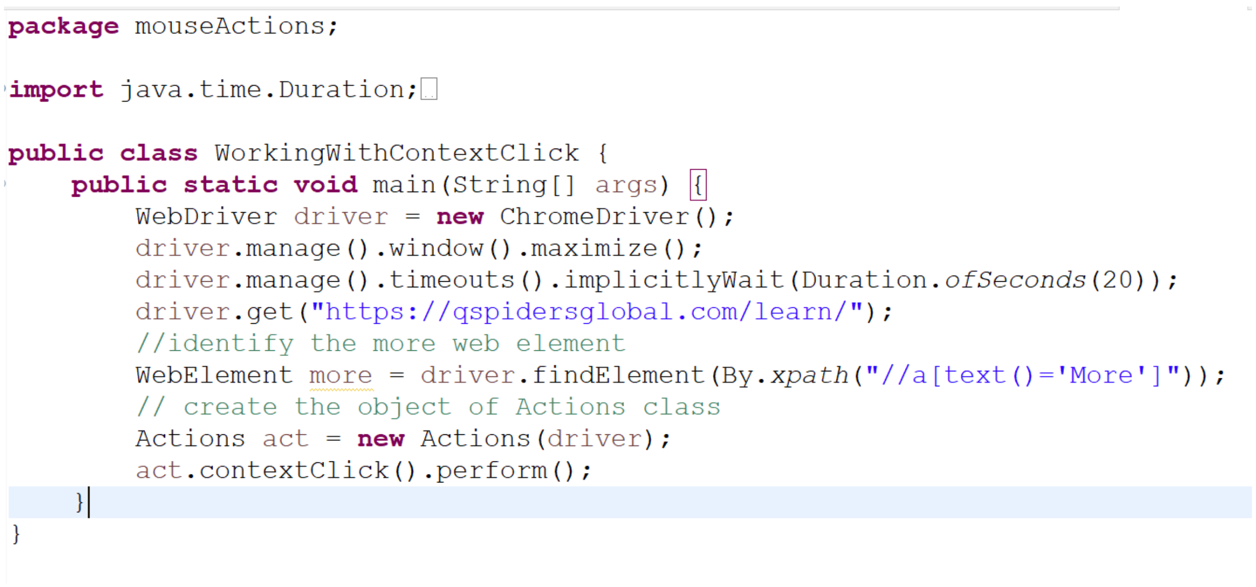


```java
package mouseActions;

import java.time.Duration;

public class WorkingWithMoveToElement {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://qspidersglobal.com/learn/");
        // identify the more web element
        WebElement more = driver.findElement(By.xpath("//a[text()='More']"));
        // create the object of Actions class
        Actions act = new Actions(driver);
        act.moveToElement(more).perform();
    }

}
```

3.contextClick():

- It is used to perform right click action.
- Return type is void.

Example:

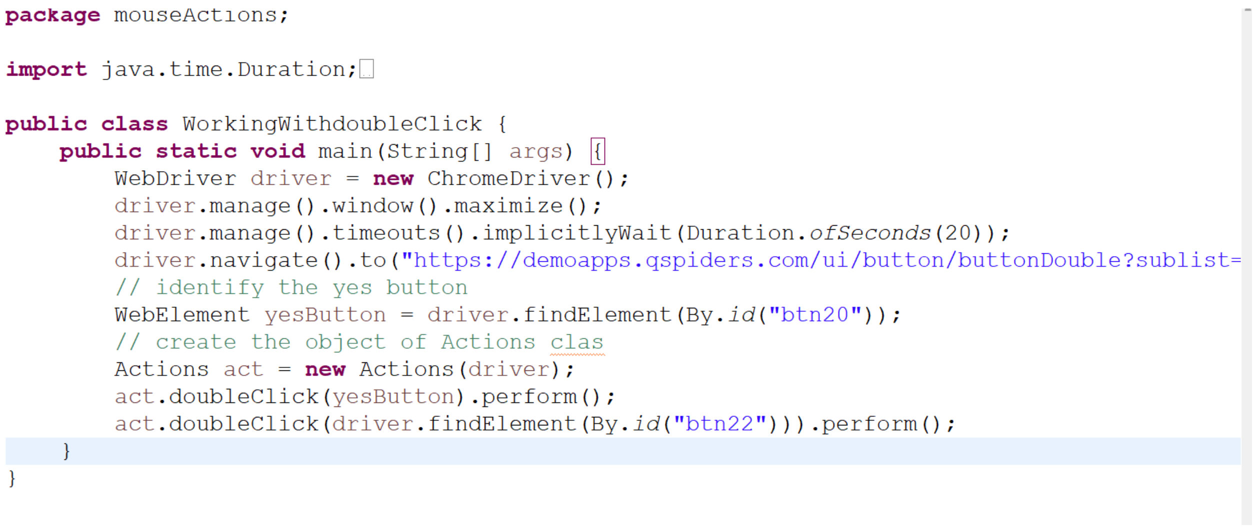


```
package mouseActions;

import java.time.Duration;

public class WorkingWithContextClick {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://qspidersglobal.com/learn/");
        //identify the more web element
        WebElement more = driver.findElement(By.xpath("//a[text()='More']"));
        // create the object of Actions class
        Actions act = new Actions(driver);
        act.contextClick().perform();
    }
}
```

4.doubleClick():

- It is used to perform double click action.
- Return type is void.

Example:



```
package mouseActions;

import java.time.Duration;

public class WorkingWithdoubleClick {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.navigate().to("https://demoapps.qspiders.com/ui/button/buttonDouble?sublist=
        // identify the yes button
        WebElement yesButton = driver.findElement(By.id("btn20"));
        // create the object of Actions clas
        Actions act = new Actions(driver);
        act.doubleClick(yesButton).perform();
        act.doubleClick(driver.findElement(By.id("btn22"))).perform();
    }
}
```

5.dragAndDrop():

- It is used to perform drag and drop operation.
- Return type is void.

```
package mouseActions;

import java.time.Duration;

public class WorkingWithDragAndDrop {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.navigate().to("https://demoapps.qspiders.com/ui/dragDrop/dragToCorrect?subli
        WebElement drag = driver.findElement(By.xpath("//div[text()='Mobile Charger']"));
        WebElement drop = driver.findElement(By.xpath("//div[text()='Mobile Accessories']")
        Actions act = new Actions(driver);
        act.dragAndDrop(drag, drop).perform();
```

6. clickAndHold():

- It is used to perform click and hold Action.
- Return type is void.

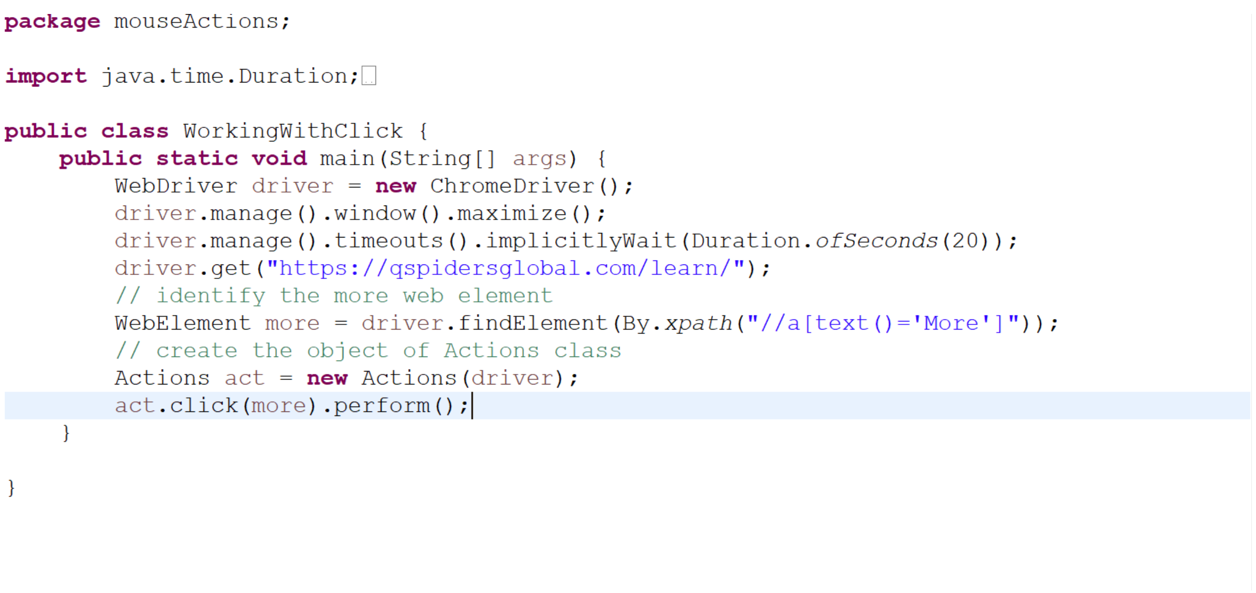
7. release():

- It is used to perform release operation.
- Return type is void.

```
package mouseActions;

import java.time.Duration;

public class WorkingWithClickAndHold {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.navigate().to("https://demoapps.qspiders.com/ui/clickHold?sublist=0");
        WebElement circle = driver.findElement(By.id("circle"));
        Actions act = new Actions(driver);
        act.clickAndHold(circle).perform();
        try {
            Thread.sleep(5000);
        } catch (InterruptedException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        act.release().perform();
    }

}
```

8.Click():

- It is used to perform click action
- Return type is void.



```
package mouseActions;

import java.time.Duration;

public class WorkingWithClick {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://qspidersglobal.com/learn/");
        // identify the more web element
        WebElement more = driver.findElement(By.xpath("//a[text()='More']"));
        // create the object of Actions class
        Actions act = new Actions(driver);
        act.click(more).perform();
    }

}
```

How to handle keyboard Action:

We can handle keyboard action by two ways:

- Keys Enum
- Robot class

Keys Enum:

Keys is Enum.
We can use this keys Enum inside the sendKeys() only.

Disadvantage:
1. We can use this only inside sendKeys().
2. We cannot deal with alphabets.

Example:

```
package KeyboardActions;

import java.time.Duration;

public class WorkingWithKeysEnum {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://opensource-demo.orangehrmlive.com/web/index.php/auth/login");
        driver.findElement(By.xpath("//input[@name='username']")).
        sendKeys("Admin",Keys.TAB,"admin123",Keys.TAB,Keys.ENTER);
    }

}
```

Robot class:

It is java class which is present inside java.awt package.
It have two non-static method

1.keypress(Keys);
2.keyRelease(Keys);

if you want to inspect any webpage then we have one shortcut key = ctrl+shift+i

```
package KeyboardActions;

import java.awt.AWTException;

public class WorkingWithRobotClass {
    public static void main(String[] args) throws AWTException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://www.myntra.com");
        Robot r = new Robot();
        // to press the key
        r.keyPress(KeyEvent.VK_CONTROL);
        r.keyPress(KeyEvent.VK_SHIFT);
        r.keyPress(KeyEvent.VK_I);
        // to release the key
        r.keyRelease(KeyEvent.VK_CONTROL);
        r.keyRelease(KeyEvent.VK_SHIFT);
        r.keyRelease(KeyEvent.VK_I);

    }

}
```

Github/Source control tool/Code management tool/Project management tool:

Github is a distributed cloud de-centralized repository where we can store our source code or automation framework/ client requirement/ test cases in a single area or place.

Why we are calling it is decentralized repository?
we cannot push our source code directly from ecllipse to global repository(Github) so before that we have to create a local copy/ repository and from that local repository we can push the code into global repository i.e GitHub.

We have two github commuinity in the market:

- Github software
- git client

github software:
It is a software which is used to store the source code or Automation framework we can store in one place.

git [Git-Client]:It is a s/w which should be installed in the system. which is used to commuincate with github software.

Steps to push Automation Scripts in GitHub Repository:

1. Open Browser and navigate to url: https://github.com/
2. Click on Sign in Button
3. Enter username, password and click on Sign in button
4. Click on New Button
5. Enter Repository name and click on Create repository button
6. Copy The URL of the current Web page
7. Now open eclipse
8. Click on magnifying glass at top right corner or press ctrl+3
9. Type: git re in search box and click on second option- Git Repositories
10. Click on Clone a Git repository
11. Click on next
12. Click on next
13. Click on finish
14. Right click on wcsm8seleniumproject in package explorer (your workspace)
15. Mouse hover to Team option
16. Click on share project
17. Click on Repository dropdown and select the first option
18. Click on Finish

19. Again, Right click on wcsm8seleniumproject in package explorer (your workspace)
20. Mouse hover to Team option
21. Click on commit (first option)
22. Double click and maximize the Git stagging window
23. Click on any one option on unstaged changes
24. And press ctrl+a on keyboard
25. Now drag and drop everything in Staged Changes
26. Add a commit message in commit message section ex: commit
27. Click on commit and push button
28. Enter username (GitHub Username) and in password text box give the authorization (GitHub Token). To generate a token, follow the bellow steps: How to create a GitHub Token.
29. Now click on login button
30. Click on close button

You successfully pushed all your code in GitHub repository

How To create GitHub Token:

1. Login GitHub account.
2. Click on Profile Dropdown.
3. Click on second last option i.e: Settings
4. Scroll Down and click on Developer settings
5. Click on Personal access tokens
6. Click Tokens (classic)
7. Click on Generate new token dropdown
8. Click on Generate new token (classic)
9. If it is asking for password then enter password and click on confirm
10. Give a note in Note TextBox forex: Note
11. Click on Expiration Dropdown
12. Click on No expiration
13. Select all check boxes given below
14. Click on Generate token button
15. Make sure to copy the generated token and save in a Notepad file.
16. Now you have successfully created a Token For Github

How to take the screenshot with script :

TakesScreenshot(I):

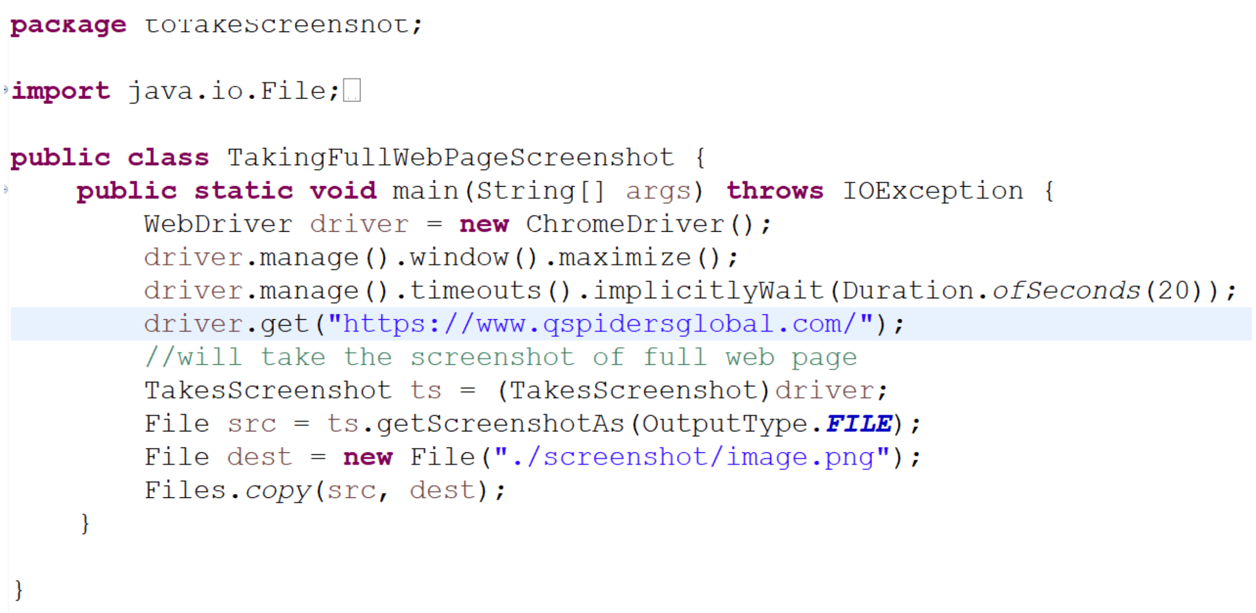
- Which is used to taking screenshot.
- It have one abstract method i.e : getScreenshotAs()

- the implementation of this method is present inside RemoteWebDriver class

We can take the screenshot by two ways:

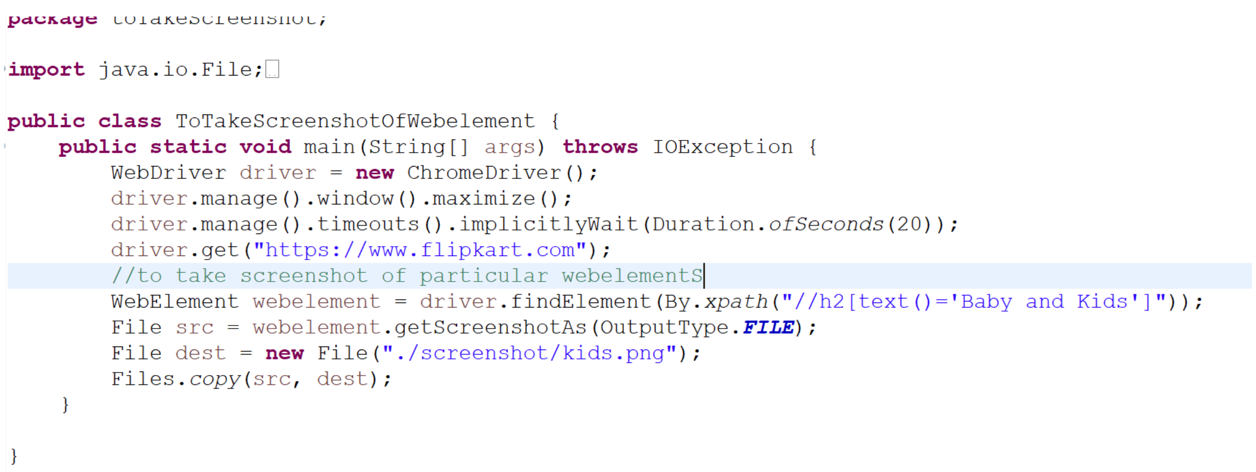
1. we can take the screenshot of full web page.
2. we can take the screenshot of single web element.

How to take the screenshot of full webpage:



```
package toTakeScreenshot;

import java.io.File;

public class TakingFullWebPageScreenshot {
    public static void main(String[] args) throws IOException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://www.qspidersglobal.com/");
        //will take the screenshot of full web page
        TakesScreenshot ts = (TakesScreenshot)driver;
        File src = ts.getScreenshotAs(OutputType.FILE);
        File dest = new File("./screenshot/image.png");
        Files.copy(src, dest);
    }

}
```

How to take the screenshot of particular web element:



```
package toTakeScreenshot;

import java.io.File;

public class ToTakeScreenshotOfWebelement {
    public static void main(String[] args) throws IOException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://www.flipkart.com");
        //to take screenshot of particular webelementS
        WebElement webelement = driver.findElement(By.xpath("//h2[text()='Baby and Kids']"));
        File src = webelement.getScreenshotAs(OutputType.FILE);
        File dest = new File("./screenshot/kids.png");
        Files.copy(src, dest);
    }

}
```

JavaScriptExecutor(I):

which has 2 abstract methods :
executeScript()
executeAsyncScript()

Implemenation of these methods are present inside RemoteWebDriver Class.

JavaSciptExecutor js = (JavaScriptExecutor)driver;
js.executeScript(JavaScript code);

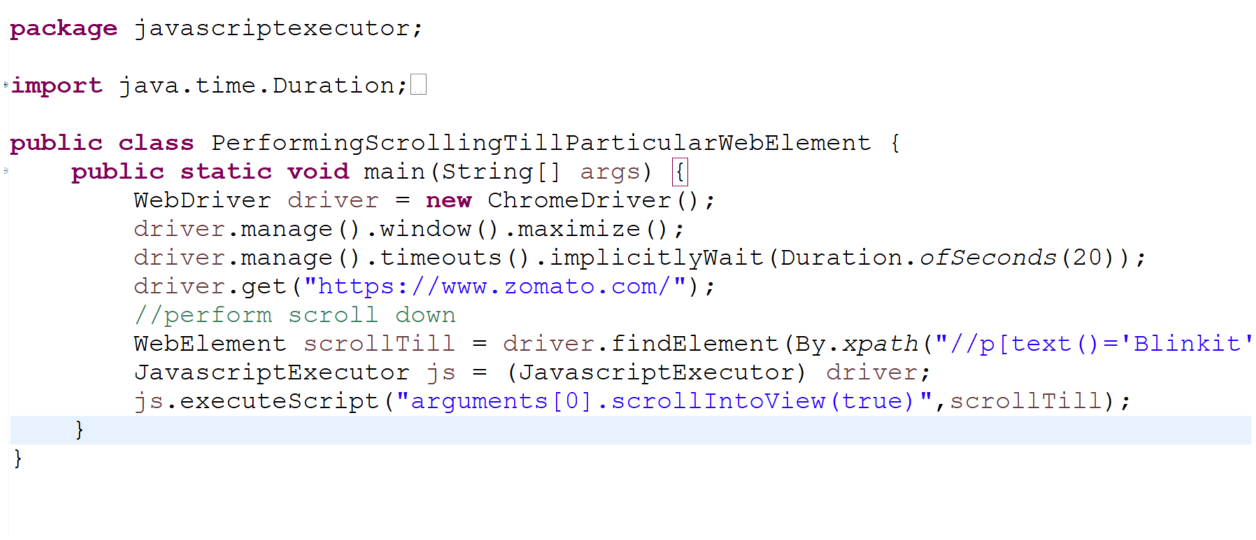
What is the use of javaScriptExecutor?

1.We can perform scrolling actions.
2. We can handle the disable web element.
3.We can handle the hidden web element.

How to perform scrolling actions by using javascriptexecutor:

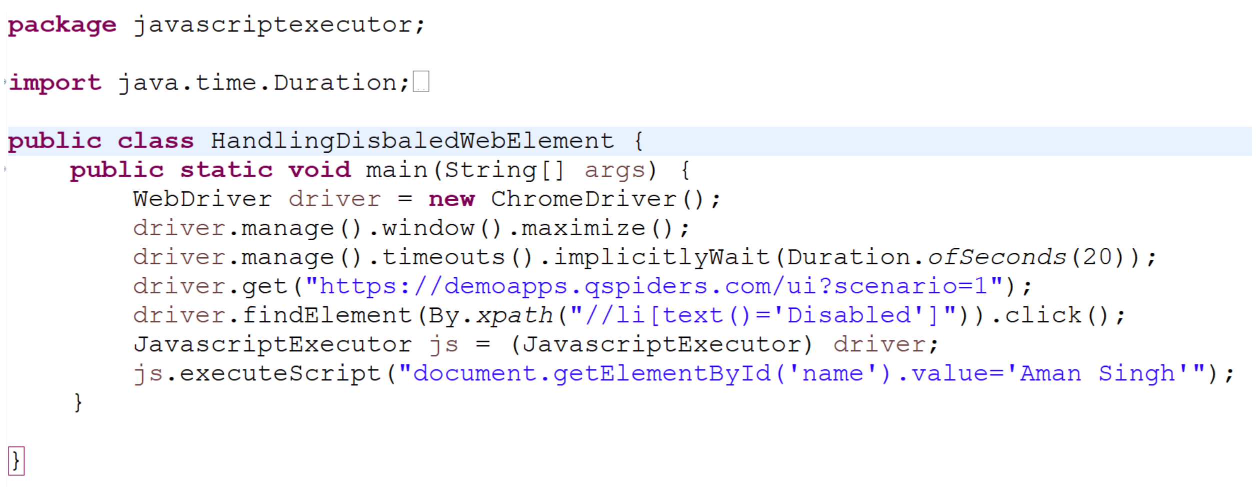
```
package javascriptexecutor;

import java.time.Duration;

public class PerformingScrollingActions {
    public static void main(String[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://www.zomato.com/");
        //perform scroll down
        JavascriptExecutor js = (JavascriptExecutor) driver;
        js.executeScript("window.scrollBy(0,1500)");
        Thread.sleep(2000);
        //performing scroll up
        js.executeScript("window.scrollBy(0,-1500)");


    }

}
```

How to perform scrolling actions till particular web element:



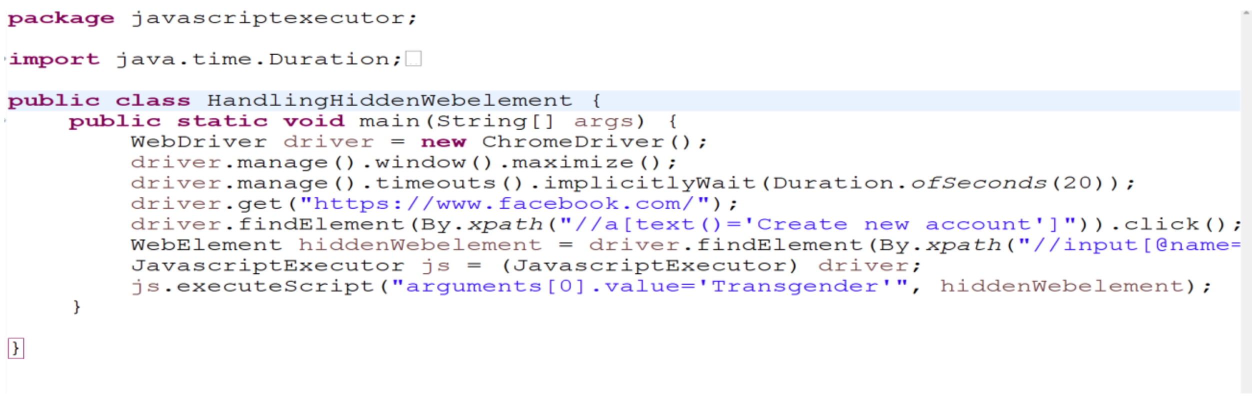
```
package javascriptexecutor;

import java.time.Duration;

public class PerformingScrollingTillParticularWebElement {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://www.zomato.com/");
        //perform scroll down
        WebElement scrollTill = driver.findElement(By.xpath("//p[text()='Blinkit'
        JavascriptExecutor js = (JavascriptExecutor) driver;
        js.executeScript("arguments[0].scrollIntoView(true)",scrollTill);
    }
}
```

How to handle Disabled Web element:



```
package javascriptexecutor;

import java.time.Duration;

public class HandlingDisbaledWebElement {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://demoapps.qspiders.com/ui?scenario=1");
        driver.findElement(By.xpath("//li[text()='Disabled']")).click();
        JavascriptExecutor js = (JavascriptExecutor) driver;
        js.executeScript("document.getElementById('name').value='Aman Singh'");
    }

}
```

How to handle Hidden web element:



```
package javascriptexecutor;

import java.time.Duration;

public class HandlingHiddenWebelement {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://www.facebook.com/");
        driver.findElement(By.xpath("//a[text()='Create new account']")).click();
        WebElement hiddenWebelement = driver.findElement(By.xpath("//input[@name=
        JavascriptExecutor js = (JavascriptExecutor) driver;
        js.executeScript("arguments[0].value='Transgender'", hiddenWebelement);
    }

}
```

Popups:
It is a obstacle which we are getting during the execution.

We have two Types of popup:

Web-Based Popup: The popup which we are getting in web application that particular popup we can consider as a web based popup.
1.JavaScript popup:
Confirmation popup
alert popup
prompt popup
2.notification popup
3.Authentication pop up
4.Hidden-division popup

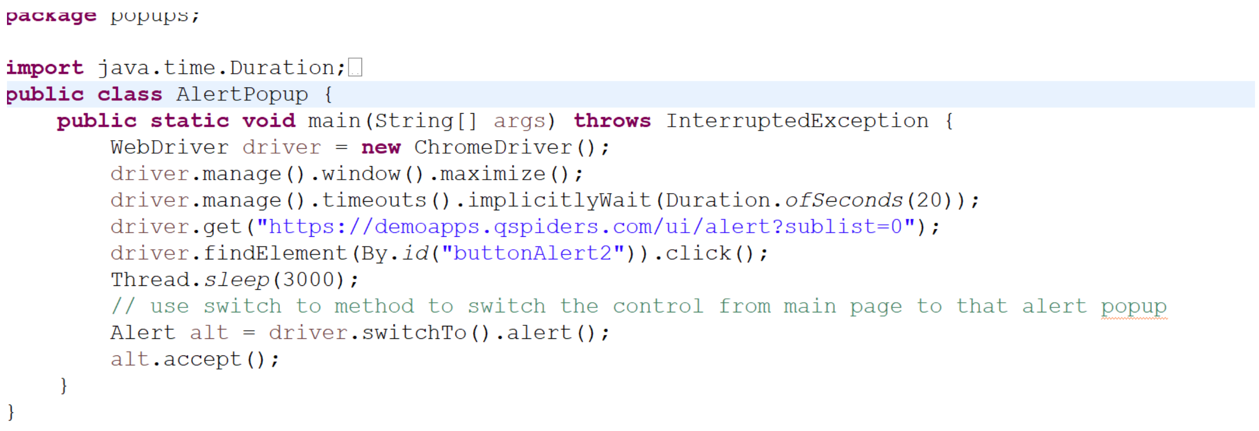
Window based Popup: The popup which we are getting in desktop application that type of popup we can consider as a window based popup.

File Upload Popup
File Download Popup

1. JavaScript popup:
That popup is developed by using java script language that why we are calling it is java script popup.

1.alert popup: that popup which is alerting you are you sure you sure you want do it.
We can not inspect popup
We can not drag this popup.

Syntax:
Alert alt=driver.switchTo().alert();
alt.accept();



```
package popups;

import java.time.Duration;
public class AlertPopup {
    public static void main(String[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://demoapps.qspiders.com/ui/alert?sublist=0");
        driver.findElement(By.id("buttonAlert2")).click();
        Thread.sleep(3000);
        // use switch to method to switch the control from main page to that alert popup
        Alert alt = driver.switchTo().alert();
        alt.accept();
    }
}
```

Confirmation Popup:

```
package popups;
import java.time.Duration;

public class ConfirmationPopup {
    public static void main(String[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://demoapps.qspiders.com/ui/alert/confirm?sublist=1");
        driver.findElement(By.id("buttonAlert5")).click();
        Thread.sleep(3000);
        //use switch to method to switch the control from main page to that alert popup
        Alert alt = driver.switchTo().alert();
        //alt.accept();
        alt.dismiss();


    }

}
```

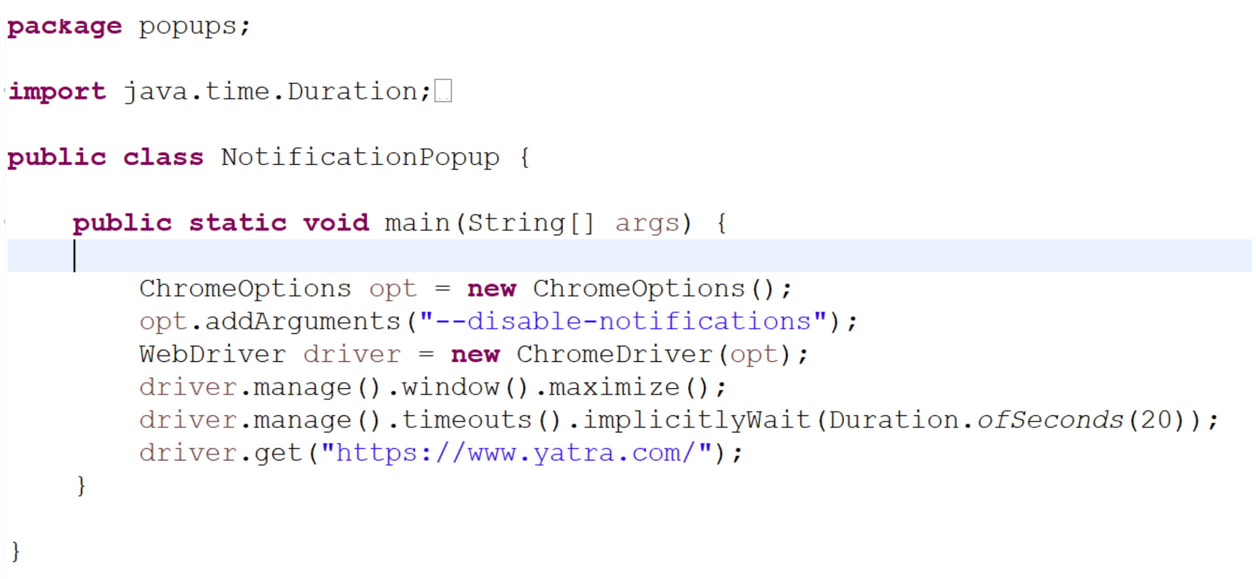
Prompt Popup:

```
package popups;

import java.time.Duration;

public class PromptPopup {
    public static void main(String[] args) throws InterruptedException {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://demoapps.qspiders.com/ui/alert/prompt?sublist=2");
        driver.findElement(By.id("buttonAlert1")).click();
        Thread.sleep(3000);
        //use switch to method to switch the control from main page to that alert popup
        Alert alt = driver.switchTo().alert();
        //use sendKeys()
        System.out.println(alt.getText());
        alt.sendKeys("yes");
        alt.accept();
    }
```

In Alert interface we have 4 types of methods:
accept(): if we want to click on ok button in alert popup we can use this method
dismiss():if you want to click on cancel
getText():fetching the text present in alert popup
sendKeys(): if you want to pass any value inside propmpt popup.

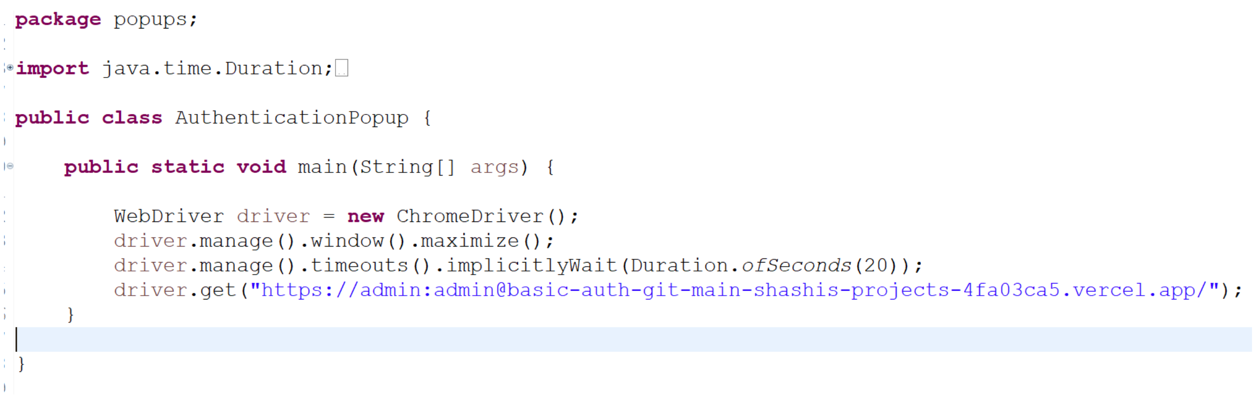
2.notification popup/ browser control popup/ permission popup:

The popup which will ask for allow and block.



```
package popups;

import java.time.Duration;

public class NotificationPopup {

    public static void main(String[] args) {

        ChromeOptions opt = new ChromeOptions();
        opt.addArguments("--disable-notifications");
        WebDriver driver = new ChromeDriver(opt);
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://www.yatra.com/");
    }

}
```

3.Authentication Popup:



```
package popups;

import java.time.Duration;

public class AuthenticationPopup {

    public static void main(String[] args) {

        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://admin:admin@basic-auth-git-main-shashis-projects-4fa03ca5.vercel.app/");
    }

}
```

How to handle child window or Window Handling?

The new tab or new browser window is considered as a child popup.
By default the driver control is present on the parent browser itself if you want to switch your driver control form one window to another window we have to switch window

How to switch to another window:

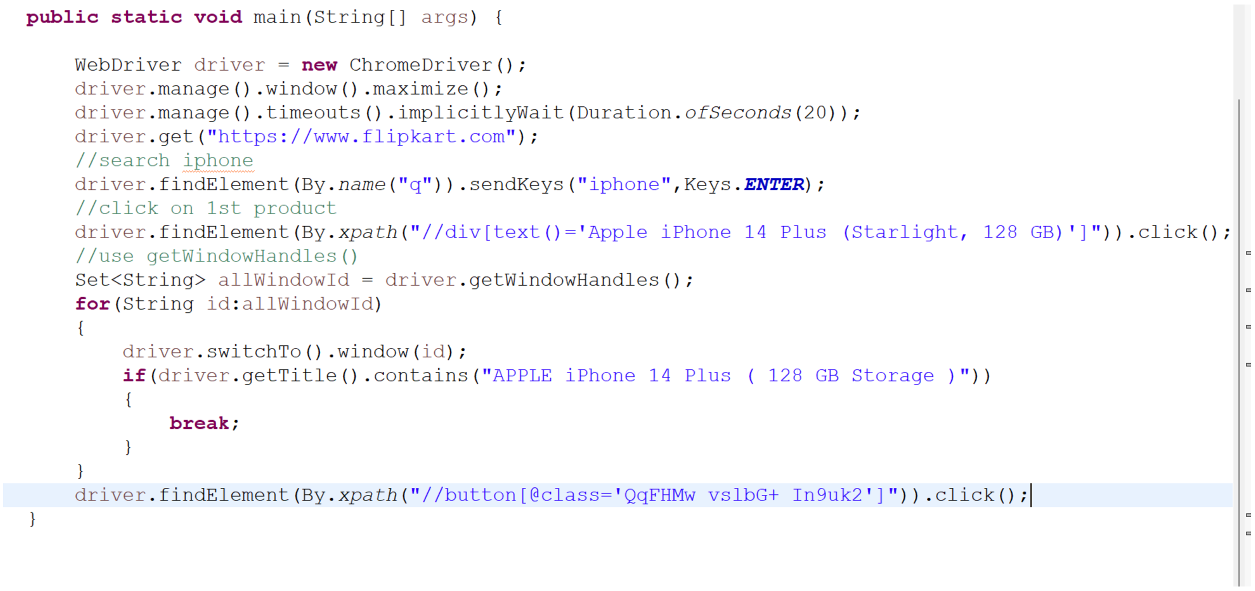
driver.switchTo().window(Session id);

Session id:
every web page have one session id.

How to get the session id or window id:
getWindowHandle():String
getWindowHandles():set<string>

getWindowHandle(): It is used to give the parent window id.

getWindowHandles(): It is used to give parent as well as child window id.



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

    WebDriver driver = new ChromeDriver();
    driver.manage().window().maximize();
    driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
    driver.get("https://www.flipkart.com");
    //search iphone
    driver.findElement(By.name("q")).sendKeys("iphone",Keys.ENTER);
    //click on 1st product
    driver.findElement(By.xpath("//div[text()='Apple iPhone 14 Plus (Starlight, 128 GB)']")).click();
    //use getWindowHandles()
    Set<String> allWindowId = driver.getWindowHandles();
    for(String id:allWindowId)
    {
        driver.switchTo().window(id);
        if(driver.getTitle().contains("APPLE iPhone 14 Plus ( 128 GB Storage )"))
        {
            break;
        }
    }
    driver.findElement(By.xpath("//button[@class='QqFHMw vslbG+ In9uk2']")).click();
}
```

Hidden Division Popup:

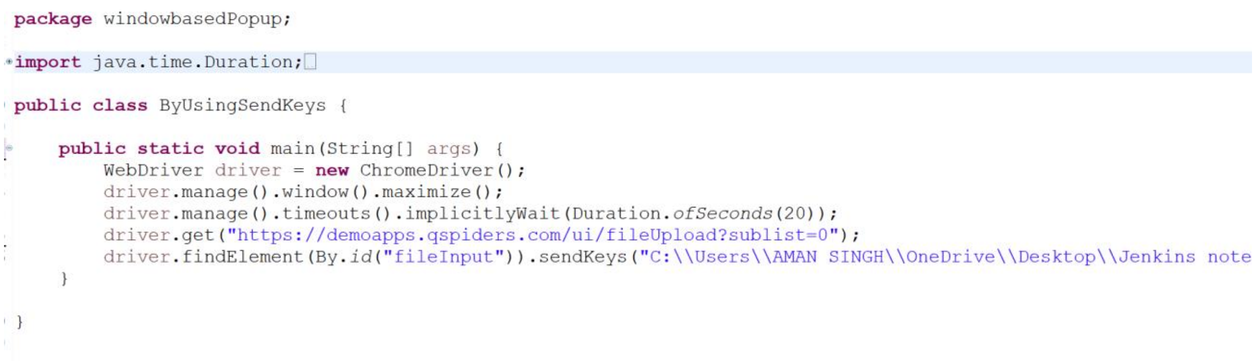
- These popup are colorful in nature.
- We can inspect this popup.
- If we can inspect this popup so directly we can handle by selenium webdriver.

Window Based Popup:

File Upload Popup:

We can handle file upload popup by two ways.

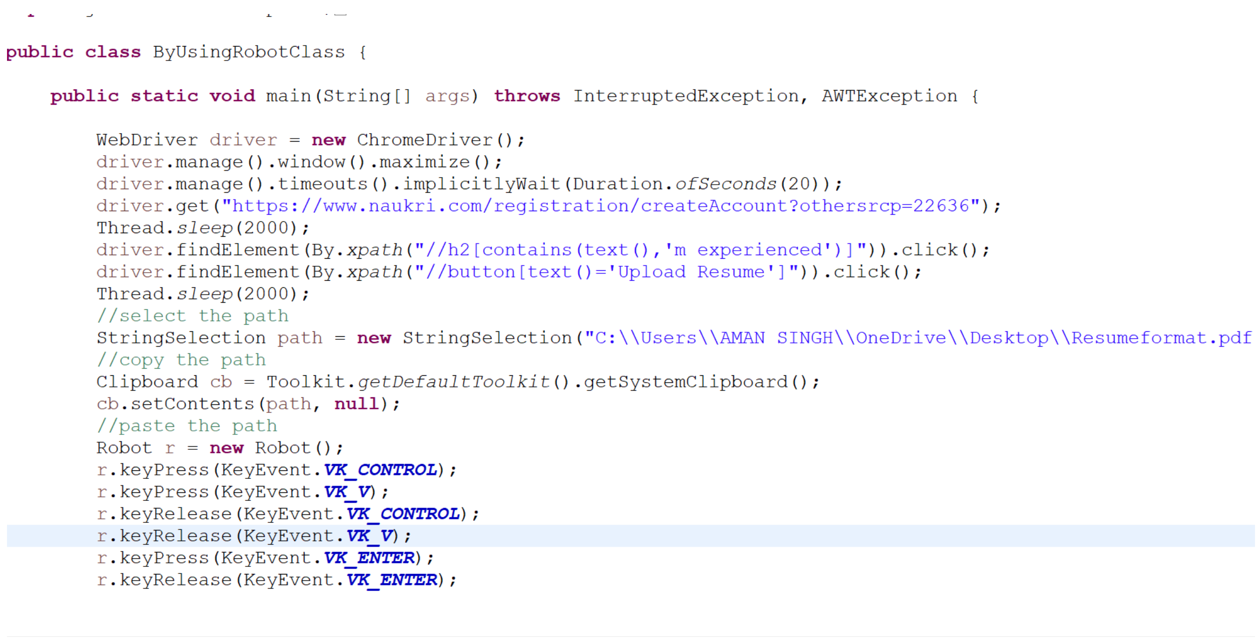
1. By Using sendKeys():



```
package windowbasedPopup;

import java.time.Duration;

public class ByUsingSendKeys {

    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://demoapps.qspiders.com/ui/fileUpload?sublist=0");
        driver.findElement(By.id("fileInput")).sendKeys("C:\\Users\\AMAN SINGH\\OneDrive\\Desktop\\Jenkins note
    }

}
```

Note: If we have **type="File"** attribute in that web element then only sendkeys() will work.

2. By Using Robot Class:



```
public class ByUsingRobotClass {

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

        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://www.naukri.com/registration/createAccount?othersrcp=22636");
        Thread.sleep(2000);
        driver.findElement(By.xpath("//h2[contains(text(),'m experienced')]")).click();
        driver.findElement(By.xpath("//button[text()='Upload Resume']")).click();
        Thread.sleep(2000);
        //select the path
        StringSelection path = new StringSelection("C:\\Users\\AMAN SINGH\\OneDrive\\Desktop\\Resumeformat.pdf
        //copy the path
        Clipboard cb = Toolkit.getDefaultToolkit().getSystemClipboard();
        cb.setContents(path, null);
        //paste the path
        Robot r = new Robot();
        r.keyPress(KeyEvent.VK_CONTROL);
        r.keyPress(KeyEvent.VK_V);
        r.keyRelease(KeyEvent.VK_CONTROL);
        r.keyRelease(KeyEvent.VK_V);
        r.keyPress(KeyEvent.VK_ENTER);
        r.keyRelease(KeyEvent.VK_ENTER);
```

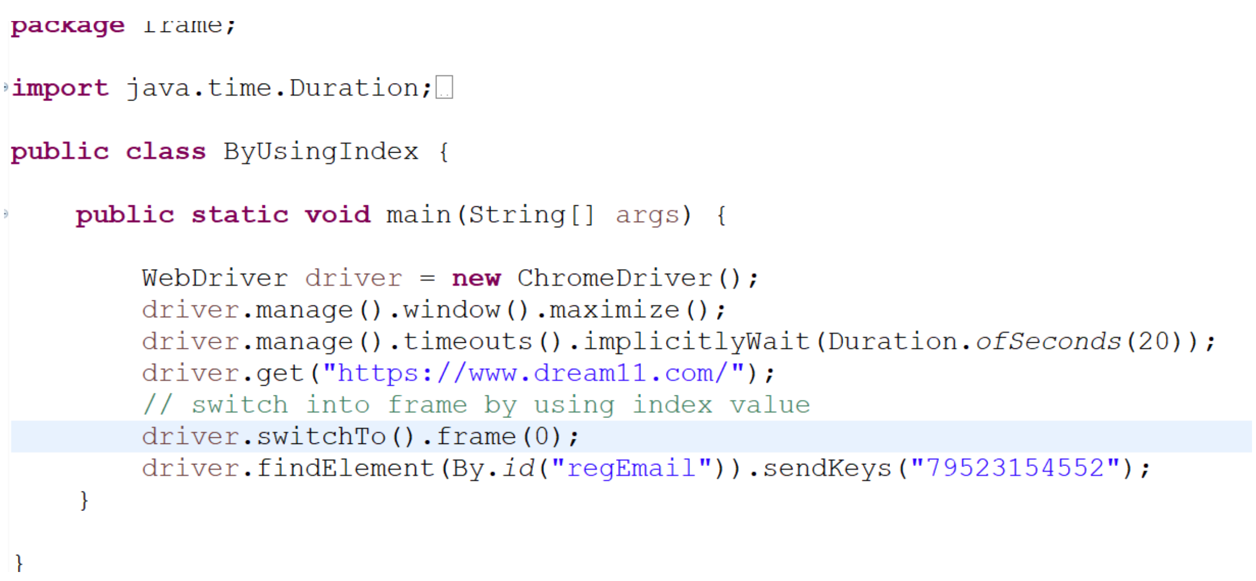
Frames:

- A webpage inside another webpage is called as frames.
- If you want to check how many frames are present inside this webpage so for that we have to write the xpath i.e: //iframe

We can Handle the frames by three ways:

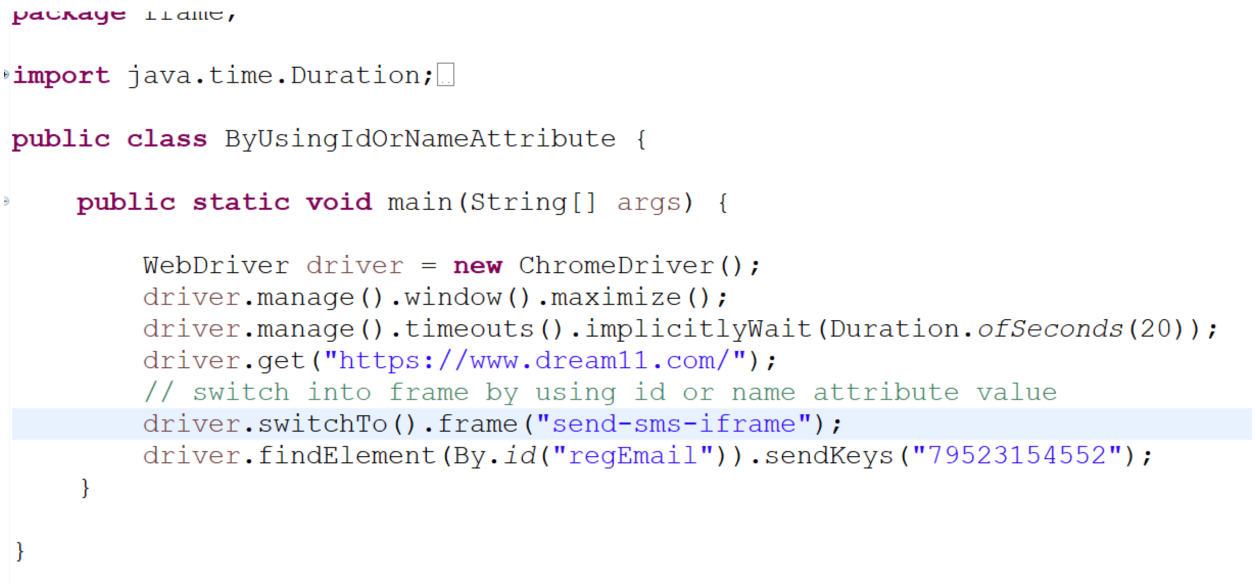
driver.switchTo().frame(int index);
driver.switchTo().frame(id or name attribute value);
driver.switchTo().frame(webelement wb);

First Way:



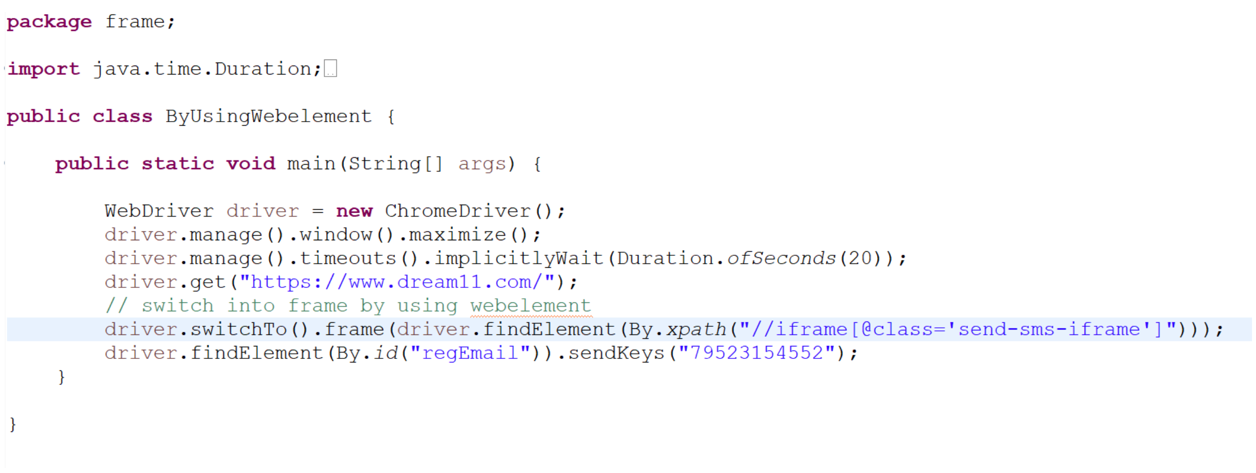
```
package frame;

import java.time.Duration;

public class ByUsingIndex {

    public static void main(String[] args) {

        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://www.dream11.com/");
        // switch into frame by using index value
        driver.switchTo().frame(0);
        driver.findElement(By.id("regEmail")).sendKeys("79523154552");
    }

}
```

Second Way:



```
package frame;

import java.time.Duration;

public class ByUsingIdOrNameAttribute {

    public static void main(String[] args) {

        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://www.dream11.com/");
        // switch into frame by using id or name attribute value
        driver.switchTo().frame("send-sms-iframe");
        driver.findElement(By.id("regEmail")).sendKeys("79523154552");
    }

}
```

Third Way:



```
package frame;

import java.time.Duration;

public class ByUsingWebelement {

    public static void main(String[] args) {

        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("https://www.dream11.com/");
        // switch into frame by using webelement
        driver.switchTo().frame(driver.findElement(By.xpath("//iframe[@class='send-sms-iframe']")));
        driver.findElement(By.id("regEmail")).sendKeys("79523154552");
    }

}
```

Automation Framework:

Automation Framework is well-defined structure which means execution, modification and creation of the test script become more easy.

Automation Framework is a set of instructions which followed by every organization or company to make automation engineer work easy.

Three stages in framework:

Design: Those who have 10 or 15 years of experience Example: Team lead, Project manager.
Implementation: Senior Automation Engineer(4-5 years experience)
Execution: Freshers

Types of framework:

1. Data-Driven Framework
2. Keyword-Driven Framework
3. Modular Driven Framework
4. TestNG
5. Method Driven Framework
6. Hybrid Framework

Data-Driven Framework:

Storing the test data into external resources like json, xml, excel, properties and database and fetching that data into test script is called as data- driven framework.

Advantages:

- Maintenance of test data is very easy.
- Modification of test data is very easy.
- It will give security to your test data.

How to fetch the data from properties files:

```
package datadrivenframework;

import java.io.FileInputStream;□

public class FetchingDataFromPropertyFile {
    public static void main(String[] args) throws IOException {
        // create the object of fileInputStream class and pass the path
        FileInputStream fis = new FileInputStream("./src/test/resources/testData/CommanData.pr
        // create the object the properties class
        Properties prop = new Properties();
        // to load all the keys inside test script
        prop.load(fis);
        // to get the property
        String username = prop.getProperty("username");
        System.out.println(username);
        String password = prop.getProperty("password");
        System.out.println(password);
    }

}
```

Disadvantages: if I want to store 100 test data inside property file then we need 100 keys for that so to avoid that we are storing the data into excel file.

Note: If you have jDK in your laptop or system then only you can fetch the data from property file.

How to fetch the data from excel:

In selenium we cannot fetch the data from excel file.
If you want to fetch the data from excel file we need third party tool i.e : Apache poi
Apache poi is third party tool which will help to fetch the data from excel file.

1. Apache POI Common
2. Apache POI(poi-ooxml-schemas)

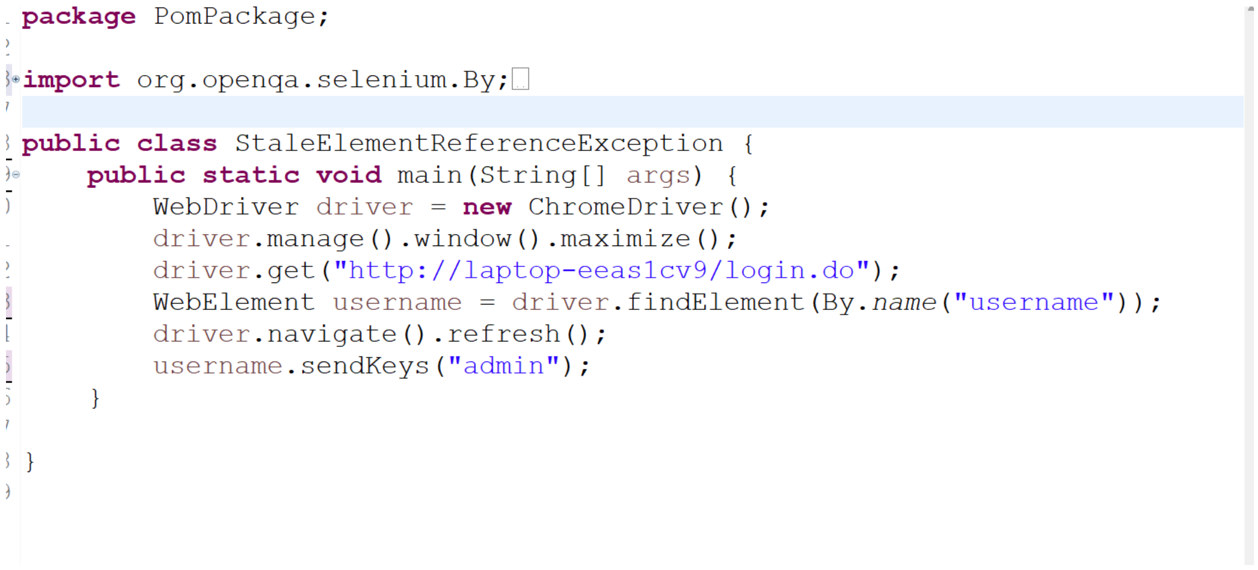
```
package datadrivenframework;

import java.io.FileInputStream;

public class FetchingTheDataFromExcelFile {
    public static void main(String[] args) throws EncryptedDocumentException, IOException {
    //create the object of fileInputStream class
    FileInputStream fis = new FileInputStream("./src/test/resources/testData/LoginCred.xlsx");
    //open the workbook in a readable mode
    Workbook book = WorkbookFactory.create(fis);
    //get the sheet
    Sheet sheet = book.getSheet("LoginCred");
    //get the row
    Row row = sheet.getRow(1);
    //create a cell
    Cell cell = row.getCell(0);
    //convert into string
    String value = cell.getStringCellValue();
    System.out.println(value);
    }
}
```

How to write the data into excel:

```
import java.io.FileInputStream;

public class WritingDataIntoExcel {
    public static void main(String[] args) throws EncryptedDocumentException, IOException {
    //create the object of fileInputStream class
    FileInputStream fis = new FileInputStream("./src/test/resources/testData/LoginCred.xlsx");
    //open the workbook in a readable mode
    Workbook book = WorkbookFactory.create(fis);
    //get the sheet
    Sheet sheet = book.getSheet("LoginCred");
    //get the row
    Row row = sheet.getRow(1);
    //create a cell
    Cell cell = row.createCell(2);
    //set the value
    cell.setCellValue("Pass");
    //open the workbook in write able mode
    FileOutputStream fos = new FileOutputStream("./src/test/resources/testData/LoginCred.xlsx"
    book.write(fos);
    book.close();
    System.out.println("Data  sent successfully");
```

Stale Element Reference Exception:

After Identifying the web element if the webpage gets refresh then we are getting **stale Element Reference** because the address will get old if the webpage get refreshed.



```
package PomPackage;

import org.openqa.selenium.By;

public class StaleElementReferenceException {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.get("http://laptop-eeas1cv9/login.do");
        WebElement username = driver.findElement(By.name("username"));
        driver.navigate().refresh();
        username.sendKeys("admin");
    }

}
```

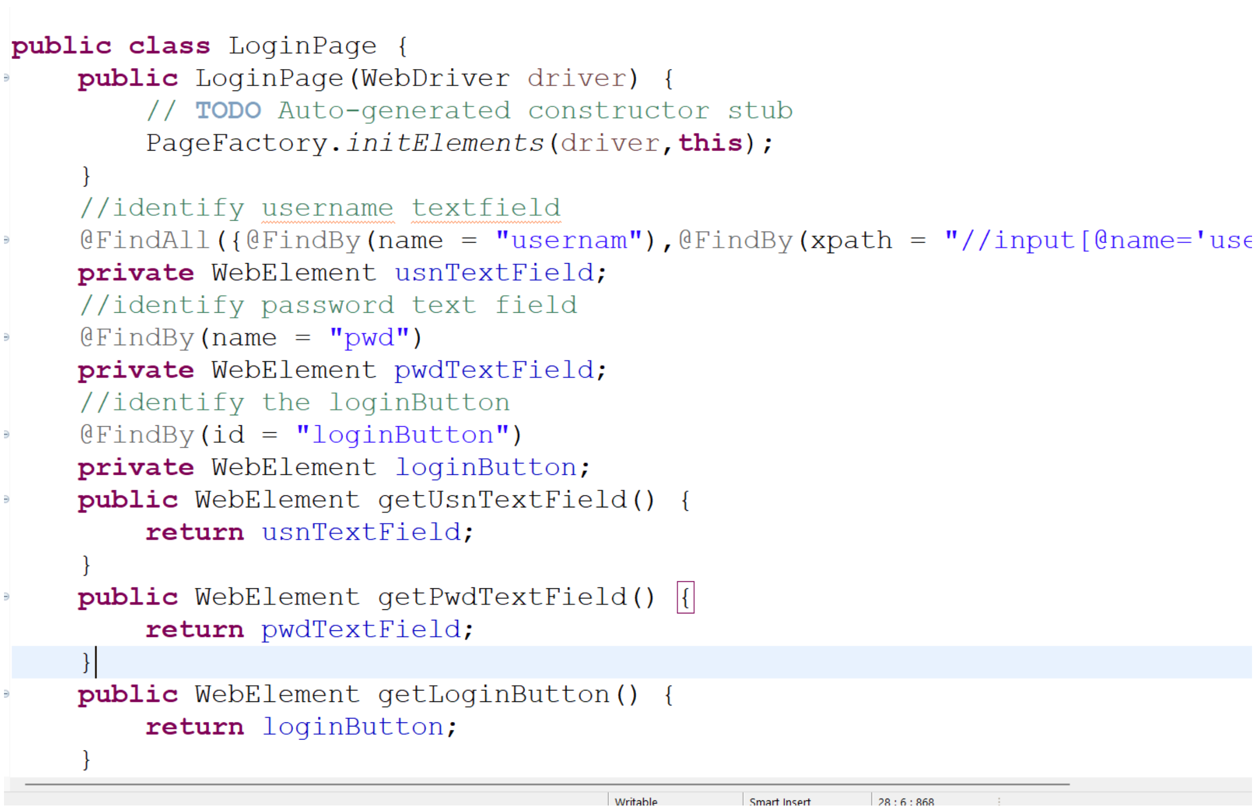
POM: Page Object Model

- It is a java-designed pattern which is introduced by Google.
- It is centralized repository where we can store our web elements as well as business operation.

Advantages:

- Maintenance of web element is very easy.
- Modification of the web element is very easy.
- We can avoid stale Element reference exception.
- We can achieve data hiding
- Auto healing

Auto Healing: According to the automation rule we have to give more than one locators to each and every web element.



```
public class LoginPage {
    public LoginPage(WebDriver driver) {
        // TODO Auto-generated constructor stub
        PageFactory.initElements(driver,this);
    }
    //identify username textfield
    @FindAll({@FindBy(name = "usernam"),@FindBy(xpath = "//input[@name='use
    private WebElement usnTextField;
    //identify password text field
    @FindBy(name = "pwd")
    private WebElement pwdTextField;
    //identify the loginButton
    @FindBy(id = "loginButton")
    private WebElement loginButton;
    public WebElement getUsnTextField() {
        return usnTextField;
    }
    public WebElement getPwdTextField() {
        return pwdTextField;
    }
    public WebElement getLoginButton() {
        return loginButton;
    }
```

Test Ng:

TestNg stands for Test Next Generation.
TestNg is Unit testing tool.
It is the combination of Junit and Nunit.

Junit: if java developer wants to do unit testing then they using Junit tool.
Nunit : if .net developer wants to do unit testing.

Both java developer and .net developer can use this tool.
We are using testng to design the framework.

Advantages:
In testng there are some annotations which will help to create the script easily.
we can perform batch Execution.
In Testng we are getting Readymade execution report.
In testng we can perform parrallel execution.
In testng we can perform cross browser testing
In testng we can perform group execution.
In testng we have on feature i.e : Listener
We can use parameterization.
We can use data-provider.

In Testng we are using assertion.

Annotations:
it is set of instruction which will tell to compiler execute me.

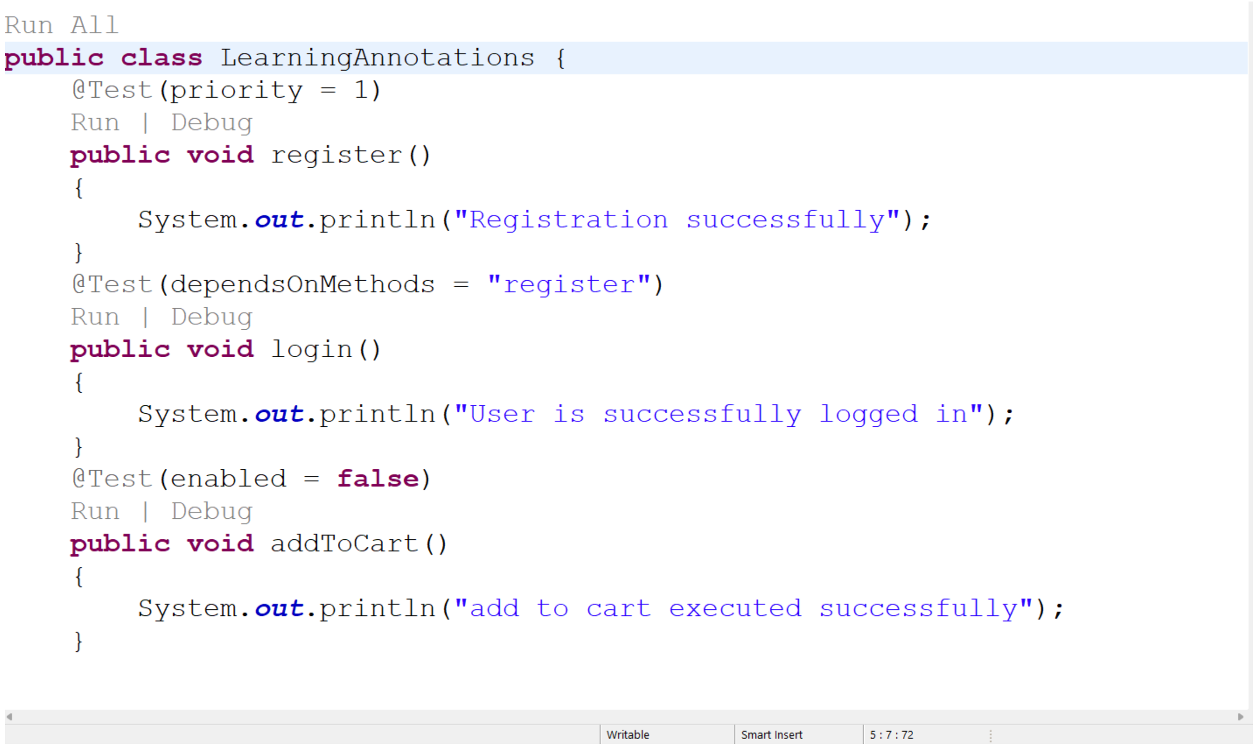
@test:

It acts like a main method.
it is a annotation which will give the instruction to the compiler i am the main method execute me.
we can use more than 1 @test inside our testng class.
if you are using @test in your class that class we will call TestNg class
If you are using 2 @test in your script so it will considered as a 2 test script.

Note: if you running more than one @test in your class so by default it will execute in alphabatical order.

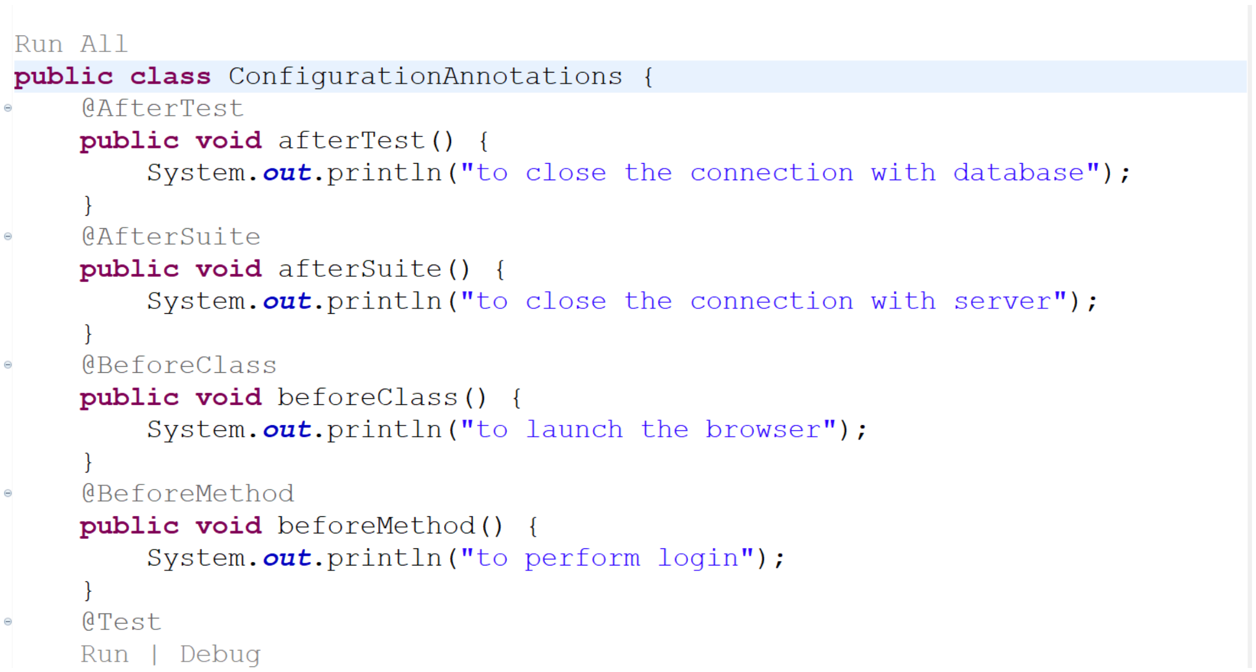
Helping attributes in Testng /Flags:

1.Priority: if i want to manage the order of execution then we can go with priority attribute.
2.enabled=false : if i don't want to execute any test script then we can go with attribute.
3.invocation Count : if you want to execute your test script more than one time.
4.DependsonMethods: if you want to create the dependency between two test scipts then are we going for this attribute

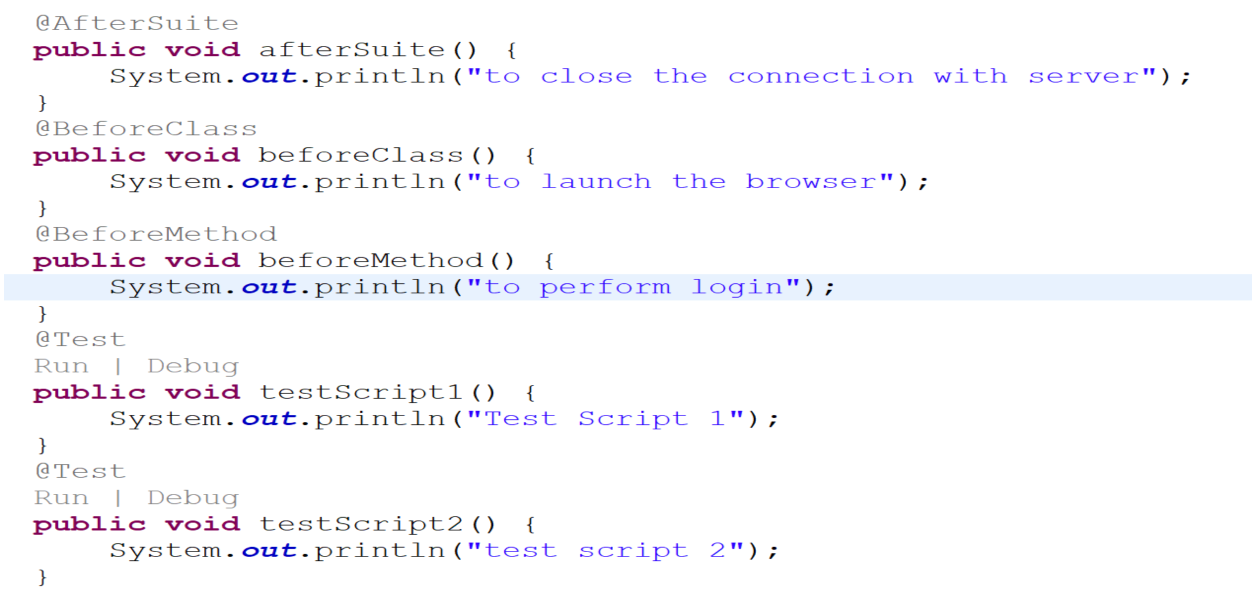

```
Run All
public class LearningAnnotations {
    @Test(priority = 1)
    Run | Debug
    public void register()
    {
        System.out.println("Registration successfully");
    }
    @Test(dependsOnMethods = "register")
    Run | Debug
    public void login()
    {
        System.out.println("User is successfully logged in");
    }
    @Test(enabled = false)
    Run | Debug
    public void addToCart()
    {
        System.out.println("add to cart executed successfully");
    }
```

Configuration Annotation Testng:

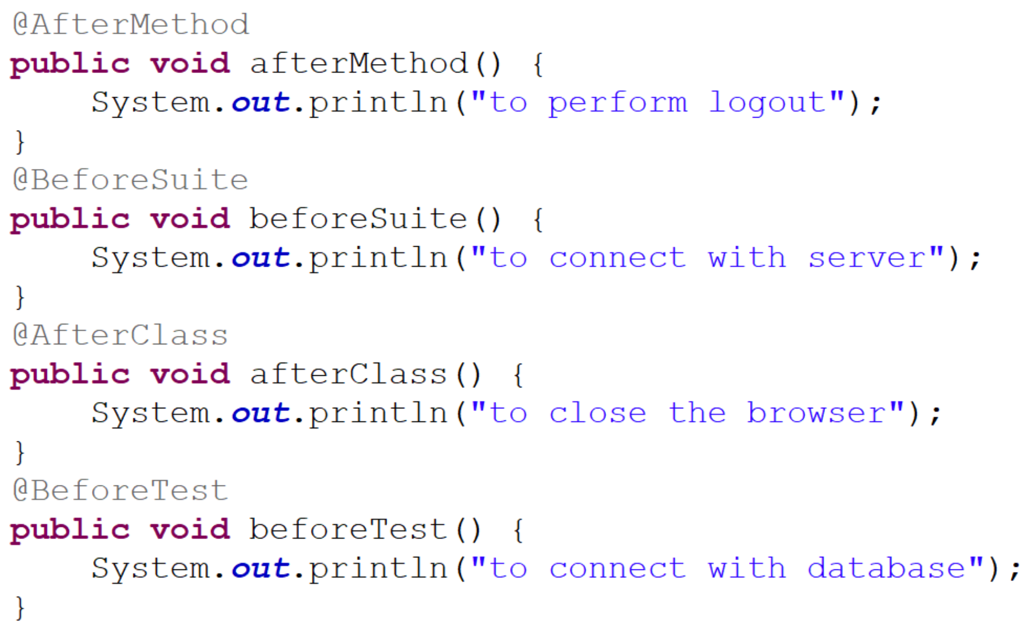
@beforeSuite--------Connect the server
@beforeTest---------Connect the database
@beforeClass--------Launching the browser
@beforeMethod------Login
@test-----------Test Script
@test----------Test Scrpit 2
@AfterMethod---Logout
@AfterClass---Close the browser
@AfterTest----Close the connection with database.
@AfterSuite---close the connection with server.



```
Run All
public class ConfigurationAnnotations {
    @AfterTest
    public void afterTest() {
        System.out.println("to close the connection with database");
    }
    @AfterSuite
    public void afterSuite() {
        System.out.println("to close the connection with server");
    }
    @BeforeClass
    public void beforeClass() {
        System.out.println("to launch the browser");
    }
    @BeforeMethod
    public void beforeMethod() {
        System.out.println("to perform login");
    }
    @Test
    Run | Debug
```



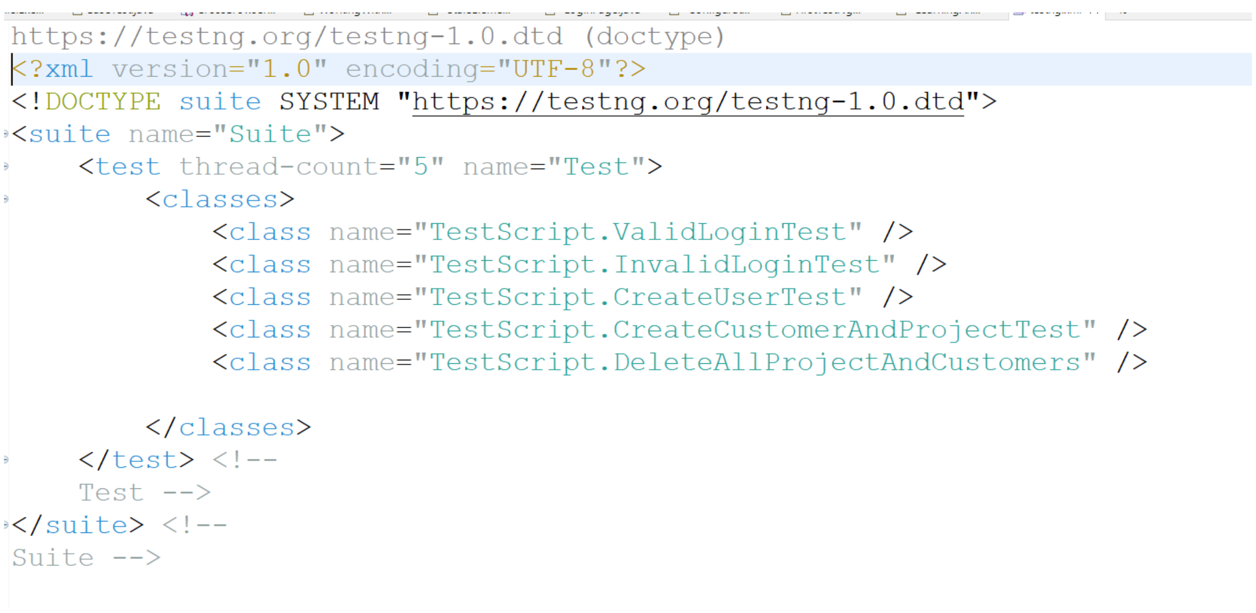
```
@AfterSuite
public void afterSuite() {
     System.out.println("to close the connection with server");
}
@BeforeClass
public void beforeClass() {
     System.out.println("to launch the browser");
}
@BeforeMethod
public void beforeMethod() {
     System.out.println("to perform login");
}
@Test
Run | Debug
public void testScript1() {
     System.out.println("Test Script 1");
}
@Test
Run | Debug
public void testScript2() {
     System.out.println("test script 2");
}
```



```
@AfterMethod
public void afterMethod() {
     System.out.println("to perform logout");
}
@BeforeSuite
public void beforeSuite() {
     System.out.println("to connect with server");
}
@AfterClass
public void afterClass() {
     System.out.println("to close the browser");
}
@BeforeTest
public void beforeTest() {
     System.out.println("to connect with database");
}
```

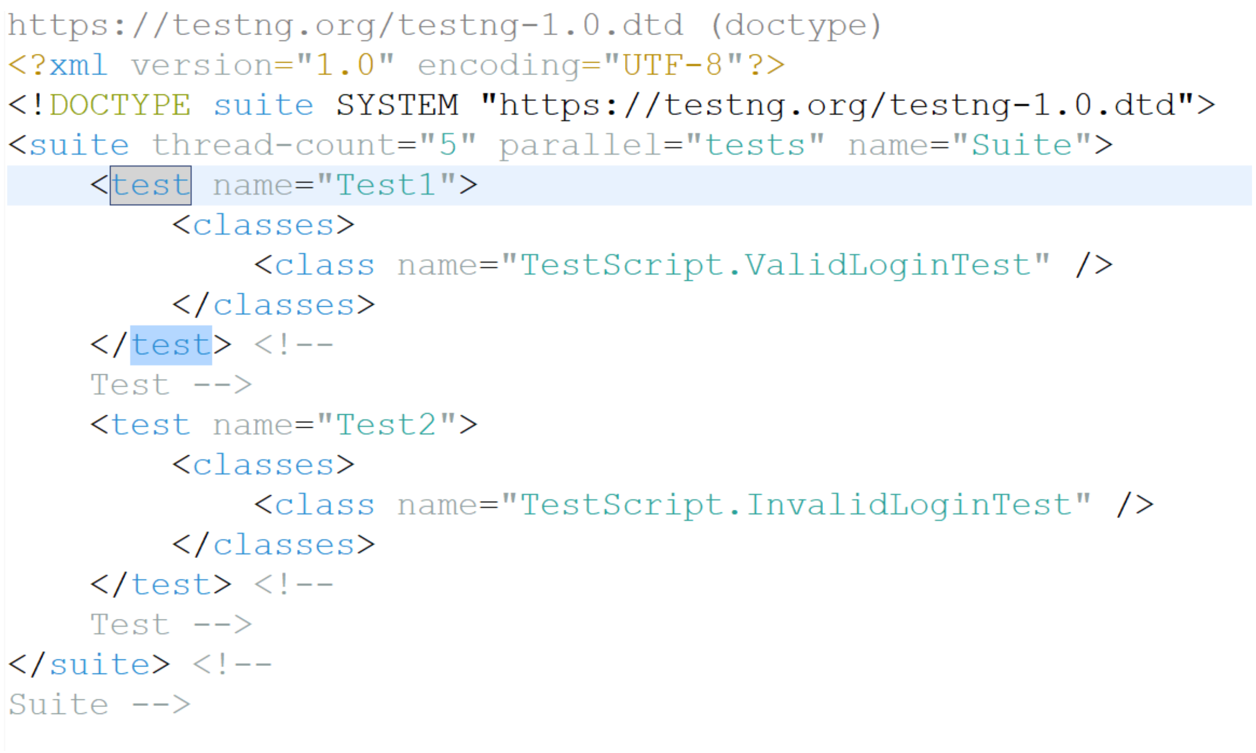
Batch Execution:

Running the test scripts one by one automatically.
If we want to perform batch execution we need Testng.xml File



```
https://testng.org/testng-1.0.dtd (doctype)
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite">
    <test thread-count="5" name="Test">
        <classes>
            <class name="TestScript.ValidLoginTest" />
            <class name="TestScript.InvalidLoginTest" />
            <class name="TestScript.CreateUserTest" />
            <class name="TestScript.CreateCustomerAndProjectTest" />
            <class name="TestScript.DeleteAllProjectAndCustomers" />

        </classes>
    </test> <!--
    Test -->
</suite> <!--
Suite -->
```

Parrallel Execution:



```
https://testng.org/testng-1.0.dtd (doctype)
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite thread-count="5" parallel="tests" name="Suite">
    <test name="Test1">
        <classes>
            <class name="TestScript.ValidLoginTest" />
        </classes>
    </test> <!--
    Test -->
    <test name="Test2">
        <classes>
            <class name="TestScript.InvalidLoginTest" />
        </classes>
    </test> <!--
    Test -->
</suite> <!--
Suite -->
```

Cross Browser Execution:

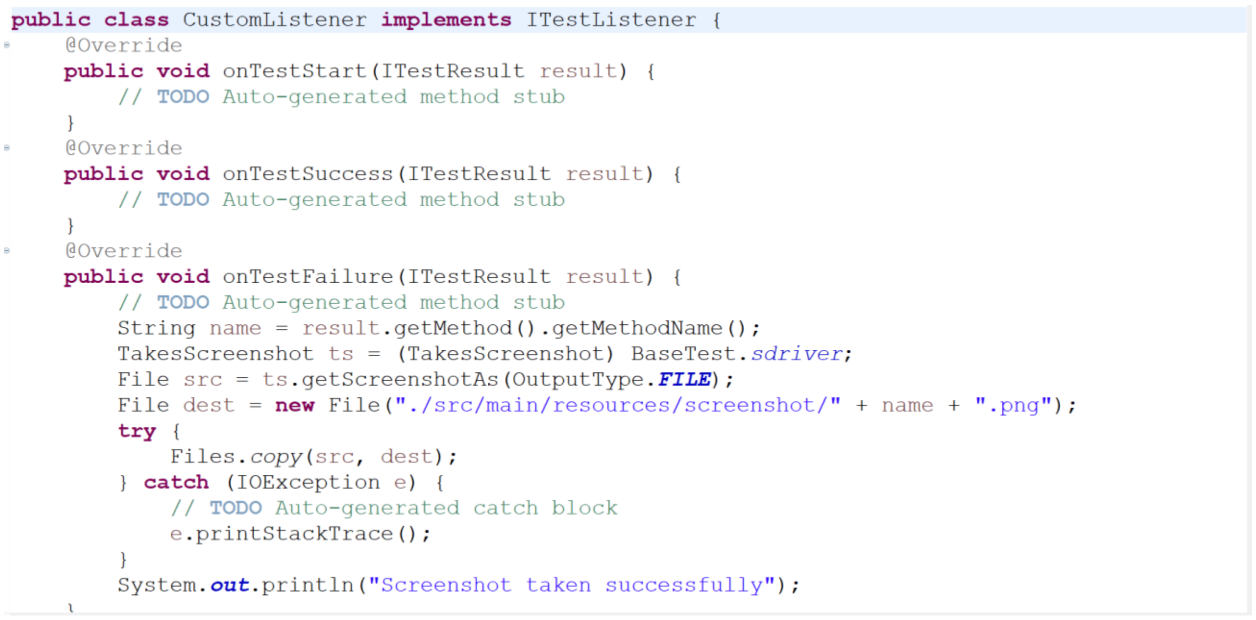
```
https://testng.org/testng-1.0.dtd (doctype)
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite thread-count="10" parallel="tests" name="Suite">
<test name="ChromeTest">
    <parameter name="browser" value="chrome"></parameter>
    <classes>
        <class name="TestScript.ValidLoginTest" />
    </classes>
</test> <!--
    Test -->
<test name="EdgeTest">
    <parameter name="browser" value="edge"></parameter>
    <classes>
        <class name="TestScript.ValidLoginTest" />
    </classes>
</test> <!--
    Test -->
</suite> <!--
Suite -->
```

Group Execution:

```
https://testng.org/testng-1.0.dtd (doctype)
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite">
    <groups>
        <run>
            <include name="RT"></include>
        </run>
    </groups>
    <test thread-count="5" name="Test">
        <classes>
            <class name="TestScript.CreateCustomerAndProjectTest" />
            <class name="TestScript.CreateUserTest" />
            <class name="TestScript.InvalidLoginTest" />
            <class name="TestScript.ValidLoginTest" />
            <class name="TestScript.DeleteAllProjectAndCustomers" />
        </classes>
    </test> <!--
    Test -->
</suite> <!--
Suite -->
```

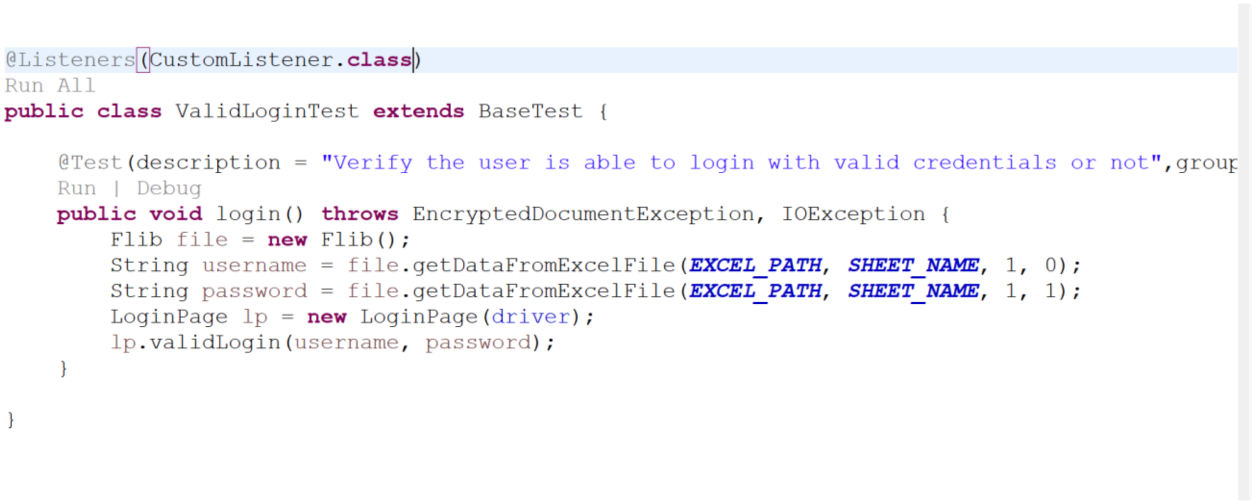
Listener feature: it is a feature which will monitor your test script it that test get failed it will take the screenshot automatically.

Implementation of Listener Feature:



```java
public class CustomListener implements ITestListener {
    @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) {
        // TODO Auto-generated method stub
        String name = result.getMethod().getMethodName();
        TakesScreenshot ts = (TakesScreenshot) BaseTest.sdriver;
        File src = ts.getScreenshotAs(OutputType.FILE);
        File dest = new File("./src/main/resources/screenshot/" + name + ".png");
        try {
            Files.copy(src, dest);
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        System.out.println("Screenshot taken successfully");
```

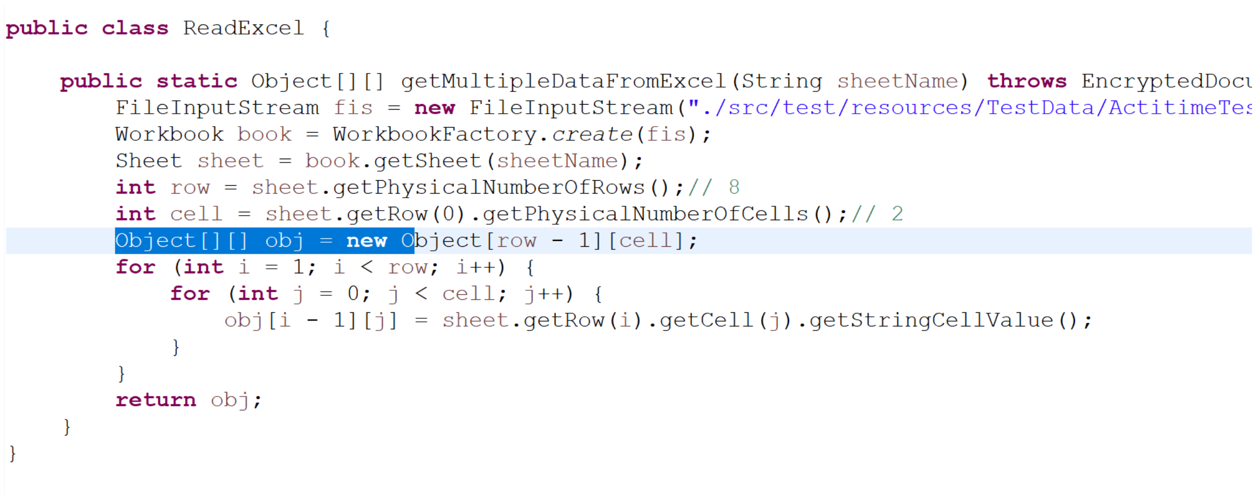
If you want to use this in test script you have to add one annotation:



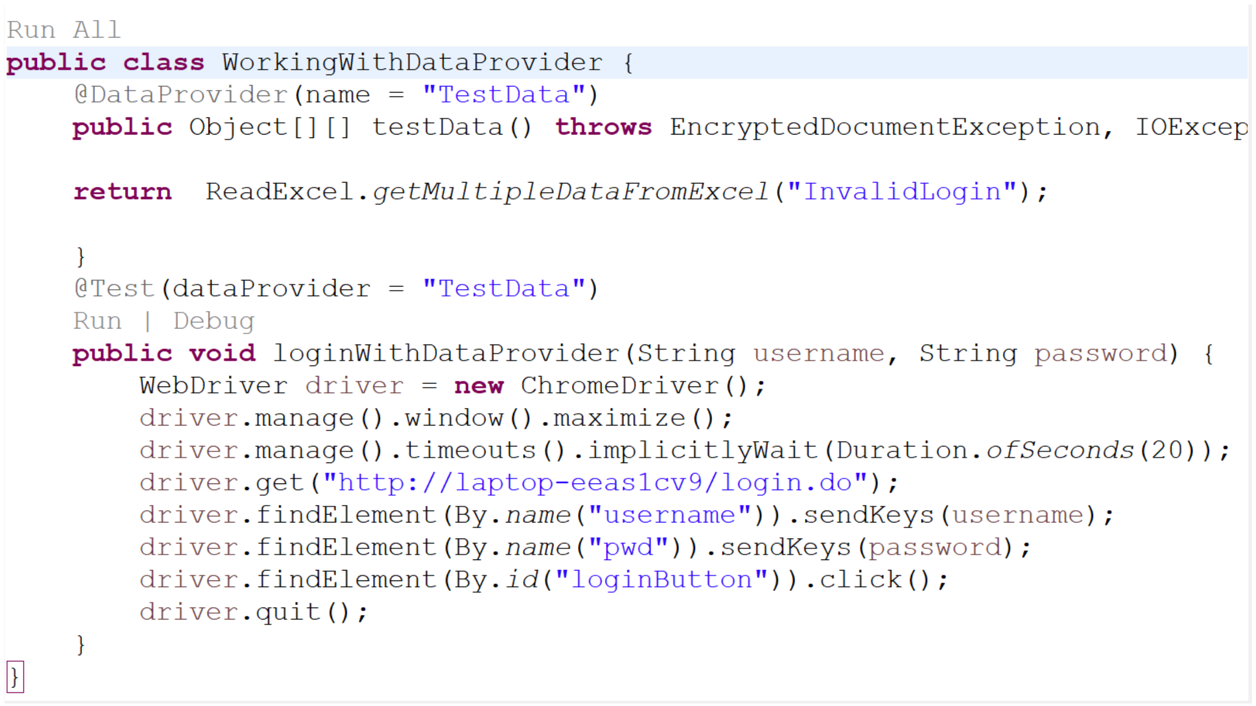
```java
@Listeners(CustomListener.class)
Run All
public class ValidLoginTest extends BaseTest {

    @Test(description = "Verify the user is able to login with valid credentials or not",group
    Run | Debug
    public void login() throws EncryptedDocumentException, IOException {
        Flib file = new Flib();
        String username = file.getDataFromExcelFile(EXCEL_PATH, SHEET_NAME, 1, 0);
        String password = file.getDataFromExcelFile(EXCEL_PATH, SHEET_NAME, 1, 1);
        LoginPage lp = new LoginPage(driver);
        lp.validLogin(username, password);
    }

}
```

Data provider:

Whenever we have to test the same module with different set of data then we have one feature in testing i.e: dataProvider



```java
public class ReadExcel {

    public static Object[][] getMultipleDataFromExcel(String sheetName) throws EncryptedDocu
        FileInputStream fis = new FileInputStream("./src/test/resources/TestData/ActitimeTes
        Workbook book = WorkbookFactory.create(fis);
        Sheet sheet = book.getSheet(sheetName);
        int row = sheet.getPhysicalNumberOfRows();// 8
        int cell = sheet.getRow(0).getPhysicalNumberOfCells();// 2
        Object[][] obj = new Object[row - 1][cell];
        for (int i = 1; i < row; i++) {
            for (int j = 0; j < cell; j++) {
                obj[i - 1][j] = sheet.getRow(i).getCell(j).getStringCellValue();
            }
        }
        return obj;
    }
}
```



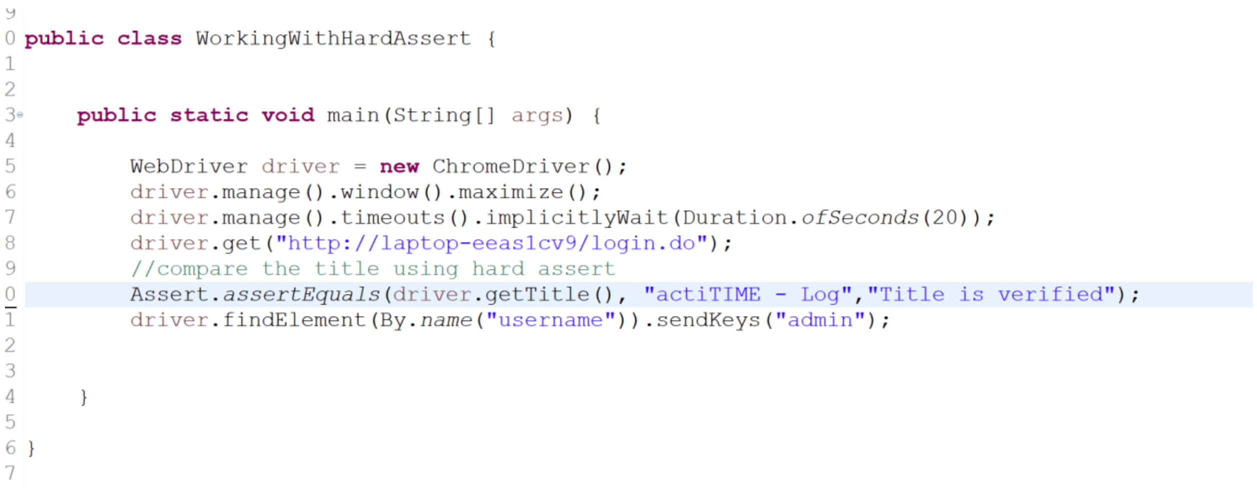
```java
Run All
public class WorkingWithDataProvider {
    @DataProvider(name = "TestData")
    public Object[][] testData() throws EncryptedDocumentException, IOExcep

    return  ReadExcel.getMultipleDataFromExcel("InvalidLogin");

    }
    @Test(dataProvider = "TestData")
    Run | Debug
    public void loginWithDataProvider(String username, String password) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("http://laptop-eeas1cv9/login.do");
        driver.findElement(By.name("username")).sendKeys(username);
        driver.findElement(By.name("pwd")).sendKeys(password);
        driver.findElement(By.id("loginButton")).click();
        driver.quit();
    }
}
```

Assertion:

Normally we are doing validation by using if else but in testing we have on e concept i.e assertion

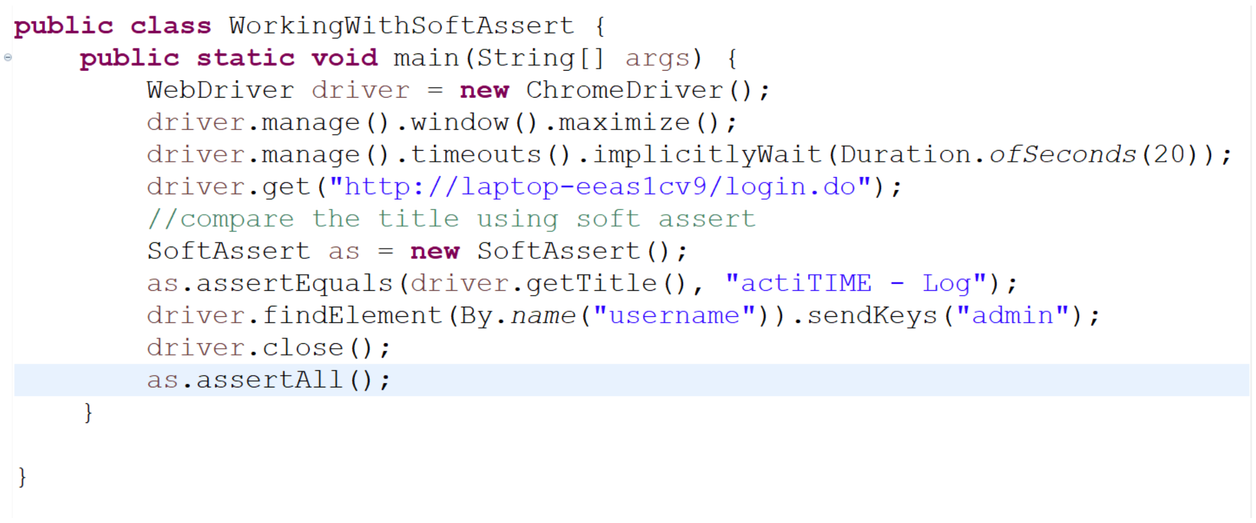
We Have to two types of assert:

Hard Assert:
We have static methods.
if hard assert get failed then it will terminate the execution.



```java
public class WorkingWithHardAssert {


    public static void main(String[] args) {

        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("http://laptop-eeas1cv9/login.do");
        //compare the title using hard assert
        Assert.assertEquals(driver.getTitle(), "actiTIME - Log","Title is verified");
        driver.findElement(By.name("username")).sendKeys("admin");


    }

}
```

Soft Assert:
we have non-static methods
If softassert get failed it will not stop the execution it will continue.
In softassert we have one compulsory method : assertAll();
This assertAll() is the last statement in test script.



```java
public class WorkingWithSoftAssert {
    public static void main(String[] args) {
        WebDriver driver = new ChromeDriver();
        driver.manage().window().maximize();
        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(20));
        driver.get("http://laptop-eeas1cv9/login.do");
        //compare the title using soft assert
        SoftAssert as = new SoftAssert();
        as.assertEquals(driver.getTitle(), "actiTIME - Log");
        driver.findElement(By.name("username")).sendKeys("admin");
        driver.close();
        as.assertAll();
    }

}
```