**Selenium Class**

**Code Optimization:**

Process of reducing the number of statements but still getting the same output is called as code optimization.

**Level1:**

int i =10;

int j = 20;

in k = i+j;

S.O.P(k);

O/P: 30

**Level2:**

int i =10;

int j = 20;

S.O.P(i+j);

O/P: 30

**Level3:**

int i =10;

S.O.P(i+20);

O/P: 30

**Level4:**

S.O.P(10+20);

O/P: 30

class PrintStream{

void println()

{

}

void println(String s)

{

}

void println(int i)

{

}

class System{

{

static final PrintStream out = new PrintStream();

}

PrintStream x = System.out;

x.println();

System.out.println();

class A{

String s = "abc";

}

class Demo{

PSVM(){

A a1 = new A();

String x = a1.s;

int i = x.lenght();

S.O.P(i);

}

}

class A{

String s = "abc";

}

class Demo{

PSVM(){

A a1 = new A();

String x = a1.s;

S.O.P(x.lenght(););

}

}

class A{

String s = "abc";

}

class Demo{

PSVM(){

A a1 = new A();

S.O.P(a1.lenght());

}

}

class A{

String s = "abc";

}

class Demo{

PSVM(){

S.O.P(new A().lenght());

}

}

System.out.println("C:\\JavaClass\\data");

Output: C:\JavaClass\data

Note: ‘\’ is escape sequence.

**Java Coding Standards:**

1. **CamelConvention:** Example: MyClass, LoginDemo
2. **pascalConvention:** Example: serachCity(), addNumber(), bankAccount
3. **Hungarian notation:** Example: strCityName, btnLogin, intAge

**Note:** it’s always better to write the java code with proper tab spacing, which is called as indentation.

**Note:** we should consider all the above points while reviewing the selenium script.

**Note:** Selenium webdriver is also called as Selenium2.

1. What are the languages supported by Selenium?

Ans: Java, C#, Ruby, Python, JavaScript(grid), perl, PHP

1. What is Selenium webdriver?

Ans: it’s a free open source web application automation tool.

Free: we can use selenium for commercial purpose without purchasing license.

OpenSource: we can download and customize the source code.

Automation: Any take done using a Sys/S/w is called as Automation.

**Launch Firefox browser:**

FirefoxDriver f = **new** FirefoxDriver();

**Architecture of Selenium WebDriver:** (After the name ‘I’ indicates Interface and ‘C’ indicates Class.)

SearchContext I

WebDriver I

**SearchContext** is Super most interface which extends **WebDriver** interface, all the abstract methods of these two interfaces are implemented in **RemoteWebDriver** class.

All the methods of RemoteWebDriver class are inherited in respective browser classes, such as: FF, Chrome, Opera, IE, Safari, so on.

In Selenium we use Up-Casting to achieve run-time polymorphism and we Up-Caste the object till WebDriver interface. (this is useful while doing browser compatibility testing)

As shown: WebDriver driver = new FirefoxDriver();

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

driver.get("http://127.0.0.1:8090/login.do");

driver.findElement(By.name("username")).sendKeys("user1");

driver.findElement(By.name("pwd")).sendKeys("user1");

driver.findElement(By.id("loginButton")).click();

driver.findElement(By.id("logoutLink")).click();

driver.close();

1. How Selenium works internally?

Ans: It performs action on the application by calling native methods of the browser. For this it uses JSON WIRE Protocol (**JSON : Java script object notation**)

1. Write a Selenium code to open the webpage.

Ans: WebDirver driver = new FirefoxDriver();

driver.get(<http://www.google.com>);

Note1: it is mandatory to specify complete URL of the application i,e. we should start from protocol such as http:// or https://

Note2: get() method not only enters URL in the address bar, but it also makes Selenium to wait till the page is completely loaded.

1. Write a script to close the brower.

Ans: driver.close();

1. Write a script to open the web page, without using get() method.

Ans: driver.navigate().to("http://www.google.com");

( OR Navigate n = new navigate(); and n.to(url) )

1. Write a script click on back, forward and refresh on web page.

Ans: WebDriver driver = **new** FirefoxDriver();

driver.navigate().back();

driver.navigate().forward();

driver.navigate().refresh();

driver.close();

1. Difference between get() and navigate() method.

Ans: get() used only to open web page.

navigate() can used to perform other operations.

1. Write a script to maximize the browser. (NO Minimize option)

Ans: WebDriver driver = **new** FirefoxDriver();

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

driver.close();

1. Write a script to resize and move the browser.

Ans: WebDriver driver = **new** FirefoxDriver();

Dimension d = **new** Dimension(420,600);

driver.manage().window().setSize(d);

Point newPoint = **new** Point(300, 500);

driver.manage().window().setPosition(newPoint);

1. Write a script to print URL & title and verify title.

Ans: WebDriver driver = **new** FirefoxDriver();

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

String url = driver.getCurrentUrl();

String title = driver.getTitle();

System.***out***.println(url);

System.***out***.println(title);

**if**(title.equals("Google"))

{

System.***out***.println("Pass");

}

**else**

{

System.***out***.println("Fail");

}

driver.close();

1. How do you debug Selenium code?

Ans: Process of executing the code line by line in order to detect the bug is called as debugging.

We also debug the code to understand the login.

To debug code we should perform following steps:

1. Insert Breakpoint in the required line. (Ctrl+Shif+B)
2. Start the execution in debugging mode (F11)
3. To execute each statement, press F5/F6.

**WebElement:** anything present in the web page is called as web element, such as textbox, button, link, etc.

Before performing any action in Selenium, we should identify web elements uniquely using its characteristics given by the application developer with the help of HTML.

1. HTML stands for Hypertext markup language,
2. We use predefined words within the angle brackets (< >) they are called as HTML tab.
3. HTML code is not case sensitive.
4. We can use notepad to create html files, while saving we need to give extension as .html
5. **Login page HTML page.**

<html>

<body>

UserName:<input type="text", value = "username"><br>

Password:<input type="pwd", value = "password"> <br>

<input type="button" value = "Login">

<input type="checkBox">Remember password

</body>

</html>

1. **Hyper link HTML page.**

<html>

<title>VISHNU Lifeskill Labs</title>

<body>

<a href = "http://askvishnudotcom.blogspot.com" title = "click here">Blogger</a>

</body>

<html>

1. **Multi select and Single select html page.**

<html>

<body>

<select multiple="true">

<option>idly</option>

<option>vada</option>

<option>poori</option>

<option>dosa</option>

</select>

<select>

<option>Tea</option>

<option>Coffee</option>

<option>Milk</option>

</select>

</body>

</html>

1. **Web page with table.** (tr 🡪 table row, th 🡪 table header, td 🡪 table data)

<html>

<body>

<table border = '1'>

<tbody>

<tr>

<th>Sl No</th>

<th>Subject</th>

<th>Cost</th>

</tr>

<tr>

<td>1</td>

<td>Java</td>

<td>Rs.5000</td>

</tr>

<tr>

<td>2</td>

<td>Selenium</td>

<td>Rs.10000</td>

</tr>

</tobody>

</body>

</html>

**Components of Web Elements:**

While developing the web page using html, developers will specify the following components.

1. HTML Tag: Any word which is present after the ‘<’ symbol can be a html tag.

Example: <html>, <body>, <input>,<Select …

1. Any word which is present after the HTML will the ‘<’ symbol is called as attribute, which contains ‘=’ symbol left side of the ‘=’ is called as property name, right side of the ‘=’ is called as property value.

Example: <type = “text” value= “admin”> <border = “P”>

1. Any word which is present after the ‘>’ symbol till the end of respective HTML tag is called as ‘text’ of the element.

<html>

<title>QSP</title>

<body>

<input type = “checkbox”> Remember

</body>

<html>

1. Text of the title is “QSP”.
2. Text of the input is not present
3. Text of the body is “Remember”
4. Text of the html is “QSP” and “Remember”

Note: for any given element HTML tag is mandatory attribute and text may or may not present.

**FireBug:** it is an add-on for mozilla firefox browser, which is used to inspect the element. (by looking into the source code of the web page)

Note: if the right click is disabled, click on Firebug icon and inspect the element.

Note: Right click is called as context clicking.

**Locators:**

Locators are HTML tags, attributes, text & expressions which are used to identify the web element.

In Selenium there are 8 locators available.

1. tagName
2. id
3. name
4. className
5. linkText
6. **partialLinkText**
7. **cssSelector**
8. **xpath**

Note: 1 🡪 tag , 2 to 5 🡪 attribute , 6 to 8 🡪 formula

All the above methods are static methods of “By” class

All these methods take string as an argument

All these methods return an object type of type “By”

**Sample web page:**

<html>

<body>

<a id = “i1” name = “n1” class = “c1” href = “http://www.google.com”> Google</a>

</body>

</html>

To perform action on the element we should write following statements.

1. Specify locator
2. Find the element using specified locator
3. Perform the action on the element which is found.

WebDrived driver = new FirefoxDriver();

//specify locator

By b = By.tagName(“a”);

//find element

WebElement element = driver.findElement(b);

//perform action

element.click();

**In single line we can write as:** (optimized code… run one by one by commenting other lines to confirm working or not)

System.*setProperty*("webdriver.chrome.driver", "C:\\chromedriver\_win32\\chromedriver.exe");

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

driver.get("C:\\GCVSEL\\GCVJAVA\\src\\SeleniumClass\\LinkPage.html");

driver.findElement(By.*tagName*("a")).click();

driver.findElement(By.*id*("i1")).click();

driver.findElement(By.*name*("n1")).click();

driver.findElement(By.*className*("c1")).click();

driver.findElement(By.*linkText*("Google")).click();

driver.findElement(By.*partialLinkText*("oogle")).click();

driver.findElement(By.*xpath*("//\*[@id='i1']")).click();

driver.findElement(By.*cssSelector*("a[id='i1']")).click(); //css 1

driver.findElement(By.*cssSelector*("#i1")).click(); //css 2

driver.close();

Both linkText() and partialLinkText() locators can be used only on hyperlinks, other locators can be used for other elements.

Note: to handle dynamic link we need to use patialLinkText() ex: Inbox(xx)

If specified locator is duplicate it returns first matching web element.

If the specified locator is not matching, it give NoSuchElementException.

WebElement is an interface in selenium.

To get web element present in the appn we use findElement() method in WebDriver interface.

**Selenium script to login to actiTime appn:**

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

driver.get("http://127.0.0.1:8090/login.do");

driver.findElement(By.*name*("username")).sendKeys("user1");

driver.findElement(By.*name*("pwd")).sendKeys("user1");

driver.findElement(By.*id*("loginButton")).click();

driver.findElement(By.*id*("logoutLink")).click();

driver.close();

Note1: if we can’t inspect the element, we can’t write findElement() code.

Note2: if are getting error for SendKeys method right click on java project, go to properties, click on Java compiler, select the latest version and click OK.

1. How do you enter values into the numberic fields.

Ans: driver.findElement(By.id("i1")).sendKeys("1234");

1. How do you login to the appn without using click method.

Ans:

driver.get("http://127.0.0.1:8090/login.do");

driver.findElement(By.*name*("username")).sendKeys("user1");

driver.findElement(By.*name*("pwd")).sendKeys("user1");

driver.findElement(By.*id*("loginButton")).sendKeys(Keys.***ENTER***);

Note: sendKeys() method is an example for overloading method it can take both string and keys as arguments.

1. Write a script to change the text present in textbox.

driver.findElement(By.*id*("UserName")).clear();

driver.findElement(By.*id*("UserName")).sendKeys("Vishnu");

1. Write a script to replace text present in text box without using clear()

driver.findElement(By.*id*("UserName")).sendKeys(Keys.***CONTROL***, 'a', Keys.***DELETE***)

driver.findElement(By.*id*("UserName")).sendKeys("Vishnu");

1. Write a script to enter to enter text into textbox without using **sendKeys()**

In order to enter the text into the text box without using **sendKeys()** we need to use JavaScript.

In order to execute JavaScript manually go to required we page, right click and select **Inspect Element** click on **Console** tab than click on **JavaScript(JS)** and type the JavaScript statement in the textbox available at the **bottom** of the browser (>>) and press enter.

In order to execute JavaScript form Selenium, we use **ExecuteScript** method of **JavaScriptExecutor** interface as shown below:

**For Manual: (JS entry)**

document.getElementById("username").value="Vishnu";

**For Automation:(JS entry)**

System.*setProperty*("webdriver.chrome.driver", "C:\\chromedriver\_win32\\chromedriver.exe");

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

driver.get("http://127.0.0.1:8090/login.do");

JavascriptExecutor j = (JavascriptExecutor) driver;

j.executeScript("document.getElementById(\"username\").value=\"Vishnu\";");

Thread.*sleep*(3000);

driver.close();

To execute the JavaScript on the current browser we should use JavaScriptExecutor to point to WebDriver, this can be done by **type casting.**

1. Write a script to scroll the web page.

Ans:

WebDriver driver = new FirefoxDriver();

driver.get("http://127.0.0.1:8090/login.do");

((JavaScriptExecutor)driver).executeScript("window.scrollTo(0,2000)");

Note1: we should import **import** org.openqa.selenium.JavascriptExecutor;

Note2: we can use JavaScript to enter text into text box even if its disabled or read-only.

**cssSelector:**

cssSelector stands for Cascade Style Sheet

cssSelector is an expression which has following syntax:

htmlTab[PropertyName = 'PropertyValue']

**Sample page:**

<html>

<body>

UN:<input type = "text">

PW:<input type="pwd">

</body>

</html>

In the above web page to identify the password field we can’t use id, name, className, linkTextg, partialLinkText, as they are not present, we can use tag name but it is duplicate with user name field.

In this case we can use cssSelector as shown below:

System.*setProperty*("webdriver.chrome.driver", "C:\\chromedriver\_win32\\chromedriver.exe");

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

driver.get("C:\\GCVSEL\\GCVJAVA\\src\\SeleniumClass\\cssSelectorAndxpathTestPage.html");

driver.findElement(By.*cssSelector*("input[type='pwd']")).sendKeys("VISHNU ROCK");

Thread.*sleep*(3000);

driver.close();

**xpath:**

xpath is a type of locator & it is an expression.

Xpath is the path of the element in the html tree.

1. While writing xpath expression we use ‘/’ forward slash, the first ‘/’ represents beginning of the html tree, which is called as root.
2. After every ‘/’ we should specify tag name of immediate child element only.
3. In xpath we can use index, by default it starts with 1.
4. The index changes only if there is any duplicate **siblings**. (same tag name under same parent).
5. If we don’t specify any index, than it matches with all the elements, but in selenium it performs action on 1st point.
6. In order to verify the xpath expression is correct or not we use **firepath**. In order to install in firefox 🡪 go to tools 🡪 add on search for firepath and install (note latest firefox don’t have this compatibility, they recommend to use firefox developer version.

Note: **Firepath** will be new tab in firebug window.

**Sample page:**

<html>

<body>

UN:<input type = "text">

PW:<input type="text">

</body>

</html>

In the above sample web page to identify the password field we can’t use **cssSelector** also, in such case we go for **xpath**. (Now type and text also same for UN and PW)

System.*setProperty*("webdriver.chrome.driver", "C:\\chromedriver\_win32\\chromedriver.exe");

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

driver.get("C:\\GCVSEL\\GCVJAVA\\src\\SeleniumClass\\xpathTestPage.html");

driver.findElement(By.*xpath*("//input[2]")).sendKeys("VISHNU ROCK on XPATH");

Thread.*sleep*(3000);

driver.close();

**There are two xpaths we can get:**

**Absolute xpath:** /html/body/input[2]

**Relative xpath:** //input[2]

You can get xpath into string as pass as below:

String xp = "/html/body/input[2]";

driver.findElement(By.*xpath*(xp)).sendKeys(" ROCK on XPATH");

**Sample web page to check xpath (absolute and relative paths)**

<html>

<body>

<div>

<input type = "text" value = "AA">

<input type = "text" value = "BB">

</div>

<div>

<input type = "text" value = "CC">

<input type = "text" value = "DD">

</div>

</body>

<html>

**Code to get text from web page:**

**System.*out*.println("Absolute xpath**:");

String A1 = driver.findElement(By.*xpath*("/html/body/div[1]/input[1]")).getAttribute("value");

String A2 = driver.findElement(By.*xpath*("/html/body/div[1]/input[2]")).getAttribute("value");

String A3 = driver.findElement(By.*xpath*("/html/body/div[2]/input[1]")).getAttribute("value");

String A4 = driver.findElement(By.*xpath*("/html/body/div[2]/input[2]")).getAttribute("value");

System.***out***.println(A1+" "+A2+" "+A3+" "+A4);

**System.*out*.println("Relative xpath:");**

String B1 = driver.findElement(By.*xpath*("//div[1]/input[1]")).getAttribute("value");

String B2 = driver.findElement(By.*xpath*("//div[1]/input[2]")).getAttribute("value");

String B3 = driver.findElement(By.*xpath*("//div[2]/input[1]")).getAttribute("value");

String B4 = driver.findElement(By.*xpath*("//div[2]/input[2]")).getAttribute("value");

System.***out***.println(A1+" "+A2+" "+A3+" "+A4);

**Absolute xpath:**

Writing complete path of the element. Starting from the root is called as absolute xpath.

**Relative xpath:**

The absolute xpath is very lengthy in order to reduce the length of xpath expression we can use relative xpath.

for relative path we use “//” [double slash] instead of “/” [single slash]

“//” – represents any child (decedents)

“/” – represents immediate child

**Absolute xpath**

/html/body/div/input

/html/body/div[1]/input

/html/body/div[1]/input[1]

/html/body/div[1]/input[2]

/html/body/div/input[1]

/html/body/div/input[2]

/html/body/div[2]/input[1]

/html/body/div[2]/input[2]

**Elements**

AA BB CC DD

AA BB

AA

BB

AA CC

BB DD

CC

DD

**Relative xpath**

//input

//div[1]/input

//div[1]/input[1]

//div[1]/input[2]

//input[1]

//input[2]

//div[2]/input[1]

//div[2]/input[2]

**Elements**

AA BB CC DD

AA BB

AA

BB

AA CC

BB DD

CC

DD

**Interview Questions:**

1. Diff b/w slash and double slash

Ans:

“//” – represents any child (decedents)

“/” – represents immediate child

1. xpath expression which matches with all the links in the web page.

Ans: //a

1. Match all the images in the web page.

Ans: //img

1. How do you driver the xpath expression if you don’t know the html tag name.

Ans: //\*

1. Difference between //a and //table//a

Ans: //a matches with all the links present in the web page. Where as //table//a matches with all the links present inside the table of a web page.

1. How do you use more then one attribute in xpath expression.

Ans: //input[@id='username' AND @placehodler='username']

Note: Don’t use the attribute if the value is empty or boolean.

**xpath by Attribute**

In xpath expression we can use property name and value, which is called as xpath by attribute.

xpath expression will identify hidden element also.

If element is hidden selenium will throw ElementNoVisibleException.

Which has following syntax:

//htmlTag[@PropertyName = ‘Property Value’]

ActiTime Appn Example:

//input[@id='username']

//input[@placeholder = 'username']

**xpath by text() function**

If the attribute is duplicate attribute, in such case we can identify the element using its text.

In order use this you should use following syntax:

//htmlTag[text() = 'text value']

Examples:

1. ActiTime user tab:

//div[@class = 'label'] 🡪 6 matches

//div[text() = 'users'] 🡪 1 match

1. Download page of selenium

//td[text() = 'Java']

**xpath by contains function**

If there is a space before or after property value or text value than xpath will not identify the element. [space by code (8ubs)] 🡪 normal keyboard space may work.

To handle this scenario we use contains function of xpath.

In order to use, you should use following syntax:

//htmlTag[contains(@property Name, ' property value ')]

//input[@value= 'Create Billing Type'] 🡪 Wrong

//input[@value= 'Create Billing Type '] 🡪 Wrong

//input[contains(@value, 'Create Billing type')] 🡪 Right

For text():

//htmlTag[contains(text(),'text value')]

**<a> delete </a>**

//a[text()='delete'] 🡪 Wrong

//a[text()='delete'] 🡪 Wrong

//a[contains(text(),'delete')] 🡪 Right

**What are the use cases of contains function?**

Ans:

1. We can identify the elements even if there is a space present before or after the value.
2. We can identify the dynamic elements.

Ex: //\*[contains(text(),'Inbox')] 🡪 here tag is keeps changing and value is keeps changing.

**Interview Questions:**

1. Write a script to copy and paste from one text box to other text box.

**Sample web page:**

<html>

<body>

UN1:<input type='text' value='FirstText'>

UN2:<input type='text' value=''>

</body>

</html>

**Script to copy/paste:**

driver.get("C:\\FirstToSecondCopyAndPasteTestWebPage.html");

driver.findElement(By.*xpath*("//input[@type='text']")).sendKeys(Keys.***CONTROL***,"ac");

driver.findElement(By.*xpath*("//input[2]")).sendKeys(Keys.***CONTROL***,"v");

1. Write a script to copy first text box text to another text box without using shortcuts.

driver.get("C:\\FirstToSecondCopyAndPasteTestWebPage.html");

String v = driver.findElement(By.*xpath*("//input[@type='text']")).getAttribute("value");

driver.findElement(By.*xpath*("//input[2]")).sendKeys(v);

1. Write a script to find co-ordinates and print the width and length of textbox.

driver.get("C:\\FirstToSecondCopyAndPasteTestWebPage.html");

WebElement ele = driver.findElement(By.*xpath*("//input[1]"));

Point p = ele.getLocation();

System.***out***.println(p.x);

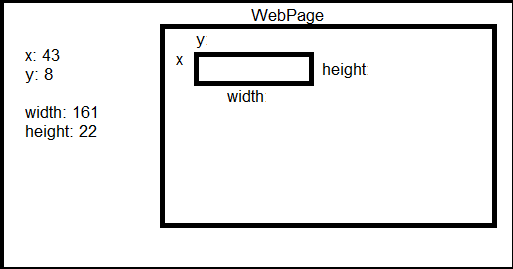
System.***out***.println(p.y);

Dimension d = ele.getSize();

System.***out***.println(d.getWidth());

System.***out***.println(d.getHeight());

Output:



1. Write a xpath expression to get all the checkboxes in the application.

//input[type = ‘checkbox’]

**Selenium script is:**

driver.get("C:\\CheckBoxesWebPage.html");

List<WebElement> ele = driver.findElements(By.*xpath*("//input[@type='checkbox']"));

System.***out***.println(ele.size());

**for**(**int** i=0;i<ele.size();i++)

{

System.***out***.println(ele.get(i).getAttribute("value"));

}

**Traversing in xpath:**

In xpath we can traverse from one note to another note which is called as traversing.

There are two types:

1. Forward traversing (FT)
2. Backward traversing (BT)
3. **Forward traversing:** navigating from parent node to child node is called as FT.

In order to do this, we use ‘/’ followed by tagname of immediate child.

Sample webpage:

<html>

<body>

<table id = 't1' border = '1'>

<tbody>

<tr>

<td>Subject</td>

<td>Cost</td>

</tr>

<tr>

<td>Java</td>

<td>500</td>

</tr>

</tbody>

</table>

</body>

</html>

1. **xpath navigation from table to subject.**

//xpath Traversing from table to subject.

String sub = driver.findElement(By.*xpath*("//table[@id='t1']/tbody/tr[1]/td[1]")).getText();

System.***out***.println(sub);

1. **Xpath navigation from table to java.**

//xpath Traversing from table to java.

String jav = driver.findElement(By.*xpath*("//table[@id='t1']/tbody/tr[2]/td[1]")).getText();

System.***out***.println(jav);

1. **Backward traversing:** navigating from child to parent node is called as BT.

In order to do this, we use ‘/’ followed by double dot (..)

//Backword Traversing

//xpath Traversing Subject to table

String tbl = driver.findElement(By.*xpath*("//td[text()='Subject']/../..")).getText();

System.***out***.println("Table FirstTime"+"\n"+tbl);

//xpath Traversing Java to table

String tbl1 = driver.findElement(By.*xpath*("//td[text()='Java']/../..")).getText();

System.***out***.println("Table SecondTime"+"\n"+tbl1);

Note: In backward traversing we don’t use indexing. [1],[2],[x],..

**Independent and Dependent Xpath:**

If we can’t use any of the previously discussed xpath, such as xpath by attribute or xpath by text() function etc,. we can use independent and dependent concept.

In order to use this concept we should follow below mentioned steps.

**Sample web page:**

<html>

<body>

<table id = 't1' border = '1'>

<tbody>

<tr>

<td>Subject</td>

<td>Cost</td>

</tr>

<tr>

<td>Java</td>

<td>500</td>

</tr>

</tbody>

</table>

</body>

</html>

1. For the given requirement: note down IE and DE. IE will be Unique, whereas DE is duplicate or dynamic.

Ex: Driver xpath to identify cost of Java

IE is Java

DE is Cost

1. Create above html tree which should contain **IE**, **Common Parent**(**CP**) and **DE**.
2. Inspect the IE & note down its code.
3. Place the mouse pointer on source code of IE than move the move pointer in the upward direction till both IE and DE highlighted, it will be the CP, add it to the tree.
4. Select the common parent, use arrow keys to navigate till DE & add it to the html tree.

<tr> CP

**IE:** <td>Java</td>

**DE:** <td>500</td>

1. Write xpath for IE

//Independent Xpath to get Java

String IE = driver.findElement(By.*xpath*("//td[text() = 'Java']")).getText();

System.***out***.println(IE);

1. Use the above xpath and driver xpath to get common parent (CP)

//Common parent xpath for Java

String CP = driver.findElement(By.*xpath*("//td[text()='Java']/..")).getText();

System.***out***.println(CP);

1. Use the above xpath and drive xpath to get DE.

//Dependent xpath to get 500

String DE = driver.findElement(By.*xpath*("//td[text()='Java']/../td[2]")).getText();

System.***out***.println(DE);

**IE and DE xpath script of Selenium website:**

//xpath to get IE and DE of Selenium website

//xpath to get IE of Ruby.

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

String RubyVer = driver.findElement(By.*xpath*("//td[text()='Ruby']")).getText();

System.***out***.println(RubyVer);

//xpath to get DE of Java Version.

String javaVer = driver.findElement(By.*xpath*("//td[text()='Ruby']/../../tr[1]/td[2]")).getText();

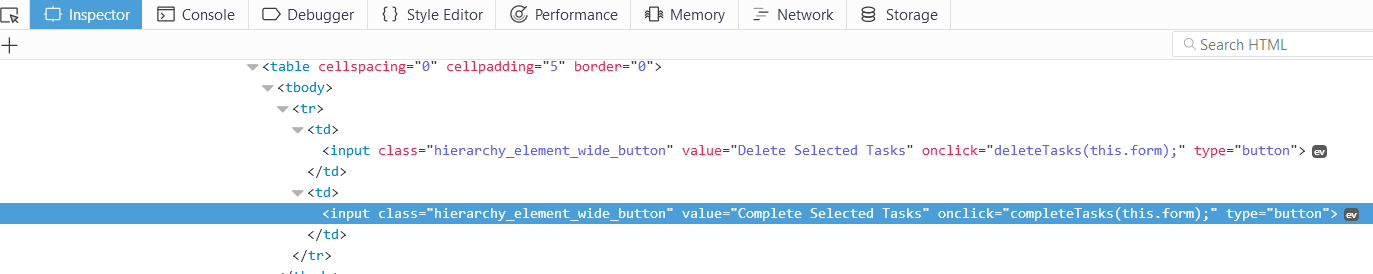
System.***out***.println(javaVer);

//xpath to get DE of JavaScript Version.

String jsVer = driver.findElement(By.*xpath*("//td[text()='Ruby']/../../tr[5]/td[2]")).getText();

System.***out***.println(jsVer);

**IE and DE xpath script of ActiTime website:**



driver.get("http://127.0.0.1:8090/login.do");

driver.findElement(By.*name*("username")).sendKeys("user1");

driver.findElement(By.*name*("pwd")).sendKeys("user1");

driver.findElement(By.*id*("loginButton")).click();

driver.findElement(By.*xpath*("//a[@class='content tasks']")).click();

//xpath getting DE 'Delete Selected Tasks' by using IE of 'Complete Selected Tasks'

String btn = driver.findElement(By.*xpath*("//input[@class='hierarchy\_element\_wide\_button']/../../td[1]/input[1]")).getAttribute("value");

System.***out***.println(btn);

driver.findElement(By.*xpath*("//a[@id='logoutLink']")).click();

driver.close();

**IMPQ**

1. **What is the diff between getAttribute() and getText()**

Ans:

getAttribute() method will return property value of the specified property name.

getText() method will return text of the element.

**Important Locators:**

1. id
2. name
3. linkText
4. xpath

Performance wise ID is better than xpath.

**Important Note:** xpath written in firefox mozilla may not work in internet explorer, in such cases you convert xpath expression into cssSelector because cssSelector is browser independent.

xpath

//input[@type = ‘text’]

//div[@class = ‘c1’]

//div[@id=’a1’]

//div/a

//div/..

//div[text() = ‘abc’

cssSelector

input[type=’text’]

div[class = ’c1’] or .c1 or div.c1

div[id=’a1’] or #a1 or div#a1

div>a

X (backward traverse not supported)

X (text() function not supported)

**Important Note:** xpath expression is not working and we cant convert into cssSelector in such cases we should handle them programmatically as shown below:

*If(browser.equal(Firefox)*

*{*

*driver.findElement(By.xpath(“FF\_xp”).click();*

*}*

*Else*

*{*

*driver.findElement(By.xpath(“IE\_xp”).click();*

*}*

**1. Write an xpath to match with all the links and images in the web page.**

//a | //img

Selenium script to get all links and images count from Amazon.com:

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

List<WebElement> li = driver.findElements(By.*xpath*("//a|//img"));

System.***out***.println(li.size());

**for**(**int** i=0;i<li.size();i++)

{

System.***out***.println(li.get(i).getText());

}

**Group Index xpath:**

There are some scenarios where we can’t use any of the xpath concepts such as attribute, text(), independent, dependent, etc., In such cases we can use Group Index (GI)

In order to use GI we should specify the complete xpath expression with the brackets ‘( )’ and then we should specify the index as shown below:

**Sample web page:**

<html>

<body>

<div>

A<input type='text' value='A'>

B<input type='text' value='B'>

</div>

<div>

C<input type='text' value= 'C'>

D<input type='text' value= 'D'>

</div>

</body>

</html>

**Sibling Index**

//input[1] 🡪 A C

//input[3] 🡪 Wrong

//input[4] 🡪 Wrong

**Group Index**  
(//input) [1] 🡪 A

(//input) [3] 🡪 C

(//input) [last()]

**Selenium code to get GI by number and by last():**

driver.get("C:\\GCVSEL\\GCVJAVA\\src\\SeleniumClass\\GroupIndexWebPage.html");

//GI: To get 2 input value

String GI = driver.findElement(By.*xpath*("(//input)[2]")).getAttribute("value");

System.***out***.println(GI);

//GI: To get last input value

String GIL = driver.findElement(By.*xpath*("(//input)[last()]")).getAttribute("value");

System.***out***.println(GIL);

//GI: To get first input value

String GIL\_1 = driver.findElement(By.*xpath*("(//input)[last()-1]")).getAttribute("value");

System.***out***.println(GIL\_1);

1. **xpath to identify help icon in ActiTime.**

//GI: to get 3rd element

String e = driver.findElement(By.*xpath*("(//div[@class='popup\_menu\_arrow'])[3]")).getAttribute("value");

System.***out***.println(e);

1. **Xpath to identify 5th link in the ActiTime.**

//GI: to get 5th link

String l = driver.findElement(By.*xpath*("(//a)[5]")).getText();

System.***out***.println(l);

1. **What is the diff between //a and (//a)[2]**

//GI: to get diff between //a links and (//a)[2] link

List<WebElement> ele1 = driver.findElements(By.*xpath*("//a"));

System.***out***.println("//a link count : "+ele1.size());

List<WebElement> ele2 = driver.findElements(By.*xpath*("(//a)[2]"));

System.***out***.println("(//a)[2] link count : "+ele2.size());

1. **Write the xpath to match with last link.**

//GI to get last link

String ll = driver.findElement(By.*xpath*("(//a)[last()]")).getText();

System.***out***.println(ll);

1. **Write the xpath to match with last() -1 (last but one) and last()-21 checkbox.**

//GI to get last but one (last()-1) link

String ll\_1 = driver.findElement(By.*xpath*("(//a)[last()-1]")).getText();

System.***out***.println(ll\_1);

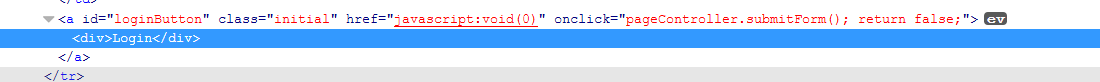
//GI to get last but one (last()-21) link

String ll\_21 = driver.findElement(By.*xpath*("(//a)[last()-21]")).getText();

System.***out***.println(ll\_21);

Note: If an element present inside an other element, try to identify the innermost element. If selenium can’t perform the action on innermost element than try to identify its parent element.

**Example ActiTime Login button:**



<a id=’loginButton’ class=……….>

<div>Login</div>

In this example 1st preference should be given for division <div> if this not working than we should use the link (<a>). But more importance should be given for performance of the code. i.e. in this example we can click on link as well as on the division, we should use xpath for division, but we use link, because it has **id**.

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

**Xpath: descendant, preceding and following sibling**

Create html file with below content:

<html>

<div id="books">

<table>

<tr><td class="title">Lord of the Rings</td>

<td class="author">JRR Tolkein</td></tr>

<tr><td class="title">The Hitch-Hikers Guide to the Galaxy</td>

<td class="author">Douglas Adams</td></tr>

</table>

</div>

</html>

xpath:

//td[text()='Douglas Adams']/parent::tr/parent::tbody/descendant::td

In the below example: C/D/E are descendants of A

A-->B-->C

A-->B-->D

A-->B-->E

-----------------------------------------------------------------------------------------------------------------------

https://www.calculator.net/conversion-calculator.html

xpath:

//option[text()='Micrometer']/preceding-sibling::option

xpath:

//option[text()='Micrometer']/following-sibling::option

in the below for "R"

preceding-sibling are "P" and "Q"

following-sibling are "S" and "T"

A-->B-->P

A-->B-->Q

A-->B-->R

A-->B-->S

A-->B-->T

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

**Synchronization:**

Process of matching selenium speed with the application speed is called as synchronization.

During run-time the find element method will try to locate the element. If it is present it performs action, if its not present it throws NoSuchElementException.

In order to handle this scenario, we should make the selenium to wait, this can be done in diff ways.

**Sonl1:** Using sleep method.

//Thread.sleep() method wait

driver.get("http://127.0.0.1:8090/login.do");

driver.findElement(By.*id*("username")).sendKeys("user1");

driver.findElement(By.*name*("pwd")).sendKeys("user1");

driver.findElement(By.*xpath*("//a[@id='loginButton']")).click();

Thread.*sleep*(20000);

driver.findElement(By.*xpath*("//a[@id='logoutLink']")).click();

driver.close();

if we write the above code without sleep statement we get NoSuchElementException because after clicking on login button, it will try to locate logout button which is not present. (when login delays loading main page). it will throw the exception.

To overcome with this issue, we have written the code wait for 20 seconds before clicking logout.

Note1: if the logout link appears after 20 seconds, still we get NoSuchElementException.

Note2: if the logout link appears in 5th second also, it wait till 20 seconds and then perform the action.

**Sonl2:** Using Exlicit wait. (WebDriverWait)

WebDriverWait is also called as explicit wait, in webdriver we should specify timeout in seconds as shown below:

//Explicit wait using WebDriverWait

driver.get("http://127.0.0.1:8090/login.do");

driver.findElement(By.*id*("username")).sendKeys("user1");

driver.findElement(By.*name*("pwd")).sendKeys("user1");

driver.findElement(By.*xpath*("//a[@id='loginButton']")).click();

WebDriverWait wait = **new** WebDriverWait(driver,20);

wait.until(ExpectedConditions.*elementToBeClickable*(By.*xpath*("//a[@id='logoutLink']")));

driver.findElement(By.*xpath*("//a[@id='logoutLink']")).click();

driver.close();

In the above code after clicking on login button, it will set the waiting period to 20 seconds and for every half second (1/2 sec) it will check whether logout is clickable or not. If the logout is clickable it immediately goes to next statement. If logout is not clickable even after 20 seconds it will throw TimeOutException.

The condition specified in WebDriverWait are called as **predicates**. Under frequency it which it checks the condition is called **polling**.

Note: we can control the polling using FluentWait (some methods are deprecated Selenium 3)

**Sonl3:** Using ImplicitlyWait. (WebDriverWait)

This method is used to set the time out which is used by every findElement method in the code.

Here if the element is not located even after the timeout we get NoSuchElementException. Generally, we specify this as a second statement in the code.

//Implicit wait using implicitlyWait()

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

driver.get("http://127.0.0.1:8090/login.do");

driver.findElement(By.*id*("username")).sendKeys("user1");

driver.findElement(By.*name*("pwd")).sendKeys("user1");

driver.findElement(By.*xpath*("//a[@id='loginButton']")).click();

driver.findElement(By.*xpath*("//a[@id='logoutLink']")).click();

driver.close();

**What is the difference between ImplicitlyWait and Explicit wait**.

**Explicit Wait (WebDriverWait)**

1. It can be used by any method.

2. TimeUnit is only in SECONDS.

3. After Timeout we get TimeOutException.

4. We must specify the predicates.

**ImplicitlyWait**

1. It is used by findElement and findElements methods only.

2. TimeUnit can be Days, Hours, Mins, and Seconds, Milliseconds, Microseconds, Nanoseconds .

3. After the timeout we get NoSuchElementException.

4. We don’t specify the condition or predicates.

1. Write a script to Login and Logout from the application without specifying the Timeout.

driver.get("http://127.0.0.1:8090/login.do");

driver.findElement(By.*id*("username")).sendKeys("user1");

driver.findElement(By.*name*("pwd")).sendKeys("user1");

driver.findElement(By.*xpath*("//a[@id='loginButton']")).click();

**while**(**true**)

{

**try** {

driver.findElement(By.*xpath*("//a[@id='logoutLink']")).click();

System.***out***.println("PASS");

**break**;

}

**catch**(NoSuchElementException ne){System.***out***.println("NSEE");}

}

driver.close();

1. Write a script to verify whether check box is selected or not.

driver.get("http://127.0.0.1:8090/login.do");

WebElement ele = driver.findElement(By.*xpath*("//input[@id='keepLoggedInCheckBox']"));

//Before CheckBox check

System.***out***.println("Before Select");

**if**(ele.isSelected())

{

System.***out***.println("ChekcBox is selected");

}

**else**

{

System.***out***.println("CheckBox is not selected");

}

System.***out***.println("After Select");

driver.findElement(By.*xpath*("//input[@id='keepLoggedInCheckBox']")).click();

//After CheckBox check

**if**(ele.isSelected())

{

System.***out***.println("ChekcBox is selected");

}

**else**

{

System.***out***.println("CheckBox is not selected");

}

driver.close();

There are 3 methods we have:

**if(ele.isSelected())**

**if(ele.isDisplayed())**

**if(ele.isEnabled())**

We can use isSelected() method to verify whether radio button isSelected or not.

If element is present in the source code and it is also visible in user interface than isDisplay will return **ture**.

If element is present in the source code but its not visible we get **false**.

If element is not present in the source code itself, than we get NoSuchElementException.

**Important Methods of WebElement:**

1. sendKeys()

2. click()

3. clear()

4. getAttribute()

5. getText()

6. isSelected()

7. isDisplayed()

8. isEnabled()

9. getLocation()

10. getSize()

11. submit()

**Handing Multiple Elements:**

In order to perform action on multiple elements we use **findElements()** method.

Which returns list of WebElement **List<WebElement>** we must import list from java.util.List package.

We generally use 2 methods:

size() to count no of objects present in the list.

get(i) to get web elements present in the specified index.

**Difference between findElement() and findElements()**

**findElements()**

1. Return type is List<WebElement>

2. If the locator is matching with multiple elements ‘n’ than it returns all the elements in the list with size index (0 to n-1)

3. if the locator is not matching with the any of the element than it returns empty list.

**findElement()**

1. Return type is WebElement.

2. If the locator is matching with multiple elements, it returns first element.

3. If the locator is not matching with any of the element than it throws NoSuchElementException.

**Note: Most of the cases we use xpath in findElements.**

**Selenium scrip on findElements():**

**Sample web page used: (Note: Link count getting 8..?!)**

<html>

<body>

<input type='checkbox' value ='one'>One<br>

<input type='checkbox' value ='two'>Tow<br>

<input type='checkbox' value = 'three'>Three<br>

<input type='checkbox' value = 'four'>Four<br>

<a href = 'aaa'>Link1<a/><br>

<a href = 'bbb'>Link2<a/><br>

<a href = 'ccc'>Link3<a/><br>

<a href = 'ddd'>Link4<a/><br>

</body>

</html>

1. **Write a script to count no of links present in the web page.**

//No of Links present in the web page.

List<WebElement> links = driver.findElements(By.*xpath*("//a"));

System.***out***.println("No of Links: "+links.size());

1. **Write a script to count no of checkboxes present in the web page.**

//No of Checkboxes present in the web page.

List<WebElement> checkboxes = driver.findElements(By.*xpath*("//input"));

System.***out***.println("No of CheckBoxes: "+checkboxes.size());

1. **Write a script to select all the checkboxs present in the web page.**

//Select all the checkboxes present in the web page.

**for**(**int** i=0; i<checkboxes.size(); i++)

{

checkboxes.get(i).click();

}

Thread.*sleep*(10000);

System.***out***.println("Select all the checkboxes present in the web page - PASS");

//Unselect all the checkboxes present in the web page.

**for**(**int** i=0; i<checkboxes.size(); i++)

{

checkboxes.get(i).click();

}

Thread.*sleep*(10000);

System.***out***.println("Unselect all the checkboxes present in the web page - PASS");

1. **Write a script to select first and last check box in the web page.**

//Check first and last checkboxes in the web page.

checkboxes.get(0).click();

checkboxes.get(checkboxes.size()-1).click();

Thread.*sleep*(10000);

System.***out***.println("Check first and last checkboxes in the web page - PASS");

//Unselect above selected checkboxes present in the web page.

checkboxes.get(0).click();

checkboxes.get(checkboxes.size()-1).click();

Thread.*sleep*(10000);

System.***out***.println("Unselect above selected checkboxes present in the web page - PASS")

//Check first and last checkbox in the web page using GroupIndexing.

driver.findElement(By.*xpath*("(//input)[1]")).click();

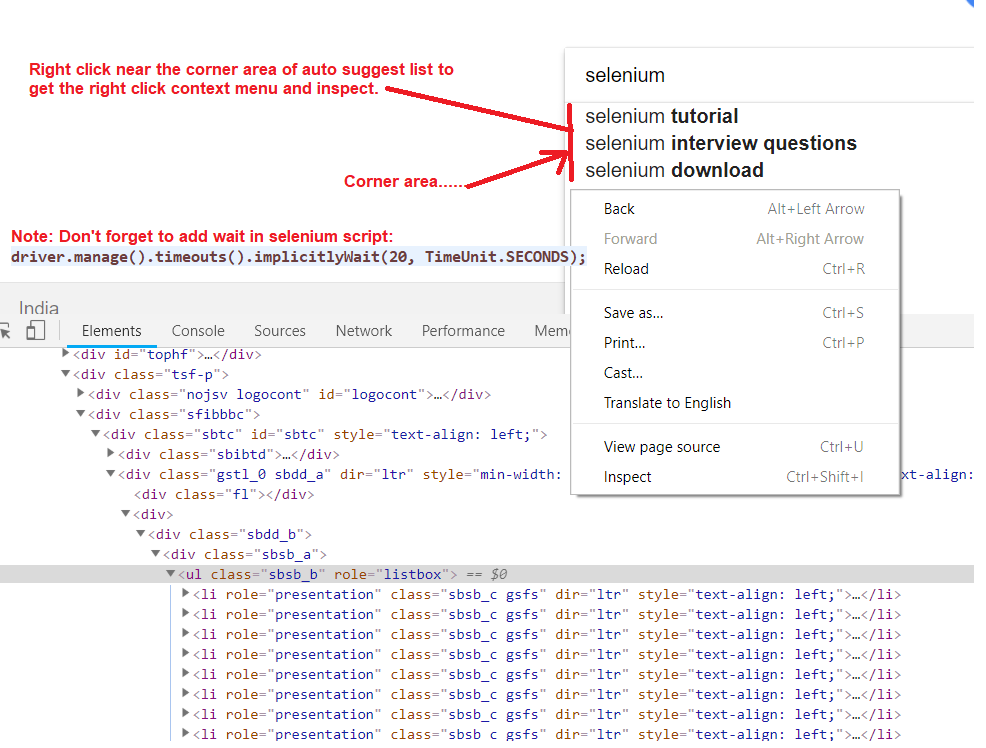
driver.findElement(By.*xpath*("(//input)[last()]")).click();

Thread.*sleep*(10000);

System.***out***.println("Check first and last checkbox in the web page using GroupIndexing - PASS");

1. **Write a script to perform below steps:**
   1. Go to google.com
   2. Search Selenium
   3. Get all the Auto Suggestions.
   4. Print the no of Auto Suggestions.
   5. Print the text of Auto Suggestions.
   6. Select the Auto Suggestion as it has ide

**To Inspect xpath: type required string (say: selenium) and right click on the corner area to get source.**



**Regarding xpath’s both will work:**

Xpath1: //div[@class='sbqs\_c']

Xpath2: //ul[@role='listbox']//li/descendant::div[@class='sbqs\_c']

Note: Don’t forget to add wait in selenium script to

**Script to get Auto Suggest list options and select in Google page:**

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

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

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

driver.findElement(By.*xpath*("//input[@id='lst-ib']")).sendKeys("selenium");

//Note: To find/inspect type right right click on the corner area to get source.

List<WebElement> ele = driver.findElements(By.*xpath*("//div[@class='sbqs\_c']"));

//List<WebElement> ele = driver.findElements(By.xpath("//ul[@role='listbox']//li/descendant::div[@class='sbqs\_c']"));

System.***out***.println(ele.size());

**for**(**int** i = 0; i<ele.size(); i++)

{

System.***out***.println(ele.get(i).getText());

**if**(ele.get(i).getText().contains("ide"))

{

ele.get(i).click();

}

}

Thread.*sleep*(5000);

driver.close(); //need to handle window handler before close

//getting exception org.openqa.selenium.StaleElementReferenceException

**Handing List box:**

In order to select required options in the list box, we should user **select method of Select class**, in order to do this we should follow steps:

1. Find the list box element.
2. Create the object of select class.
3. Select the required option using any one of the below methods:
   1. selectByVisibleText(strText)
   2. selectByIndex(int) 🡪 0,1,2,3….
   3. selectByValue(strValue)

If we specify invalid input for the above methods we get NoSuchElementException.

**Example:**

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

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

driver.get("https://www.plus2net.com/contactus.php");

WebElement ele = driver.findElement(By.*xpath*("//select[@class='form-control']"));

Select sel = **new** Select(ele);

sel.selectByValue("India");

Thread.*sleep*(3000);

sel.selectByIndex(21);

Thread.*sleep*(3000);

sel.selectByVisibleText("Albania");

Thread.*sleep*(3000);

driver.close();

1. **Write a script to perform the following steps:**
   1. Select country: India
   2. Select state: Karnataka
   3. Select city: Chitradurga

**IMP Note:** We can use **selectBy** method of select class to handle multi select list box.

If it’s a multi select list box, we can also use **deselect** (By/All) method.

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

driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);

driver.get("http://lab.iamrohit.in/php\_ajax\_country\_state\_city\_dropdown/");

WebElement country = driver.findElement(By.xpath("//select[@name='country']"));

Select selc = **new** Select(country);

selc.selectByVisibleText("India");

WebElement state = driver.findElement(By.xpath("//select[@name='state']"));

Select sels = **new** Select(state);

sels.selectByVisibleText("Karnataka");

WebElement city = driver.findElement(By.xpath("//select[@name='city']"));

Select selt = **new** Select(city);

selt.selectByVisibleText("Chitradurga");

driver.close();

1. **Muti Select and de-select test by using Select class methods:**

**Sample web page: MultiSelectListBoxWebPage.html**

<html>

<body>

<select id='s1' multiple = "true">

<option value = "" Select one </option>

<option value = "GBR">GBR</option>

<option value = "IND">IND</option>

<option value = "AUS">AUS</option>

<option value = "FRN">FRN</option>

<option value = "SWS">SWS</option>

<option value = "AMA">AMA</option>

<option value = "JPN">JPN</option>

</select>

</html>

</body>

Selenium script to select multi select and de-select by condition and all:

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

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

driver.get("C:\\GCVSEL\\GCVJAVA\\src\\SeleniumClass\\MultiSelectListBoxWebPage.html");

WebElement mlist = driver.findElement(By.*xpath*("//select[@id='s1']"));

Select msel = **new** Select(mlist);

msel.selectByIndex(0);

msel.selectByValue("IND");

msel.selectByVisibleText("AUS");

Thread.*sleep*(3000);

//deselect by checking condition isMulti select.

**if**(msel.isMultiple())

{

msel.deselectByIndex(1);

msel.deselectByValue("IND");

msel.deselectByVisibleText("AUS");

}

//Or we can use deselectAll to deselect all.

msel.deselectAll();

driver.close();

1. **Write a script to count no of options present in the list box.**
2. **Write a script to select all the options and deselect all the options.**
3. **Write a script to select and deselect all the options without using deselect.**
4. **Write a script to select and deselect all the options in reverse order.**

**Below is the code for above question:**

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

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

driver.get("C:\\GCVSEL\\GCVJAVA\\src\\SeleniumClass\\MultiSelectListBoxWebPage.html");

Select select = **new** Select(driver.findElement(By.*xpath*("//select[@id='s1']")));

//GetOptions and print count

List<WebElement> allopt = select.getOptions();

System.***out***.println(allopt.size());

//select and deselect all the options.

**for**(**int** i=0;i<allopt.size();i++)

{

select.selectByIndex(i);

}

Thread.*sleep*(3000);

select.deselectAll();

//select and deselect in reverse order and without usingn deselectAll

**int** count = allopt.size();

**for**(**int** i=count-1; i>=0; i--)

{

select.selectByIndex(i);

}

Thread.*sleep*(2000);

**for**(**int** i=count-1; i>=0; i--)

{

select.deselectByIndex(i);

}

Thread.*sleep*(2000);

driver.close();

1. **Write a script to print content of the list box.**
2. **Write a script to print all the list options.**

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

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

driver.get("C:\\GCVSEL\\GCVJAVA\\src\\SeleniumClass\\MultiSelectListBoxWebPage.html");

Select select = **new** Select(driver.findElement(By.*xpath*("//select[@id='s1']")));

//GetOptions and print count

List<WebElement> allopt = select.getOptions();

System.***out***.println(allopt.size());

//getText and print list

**for**(**int** i=0; i<allopt.size(); i++)

{

WebElement option = allopt.get(i);

System.***out***.println(option.getText());

}

driver.close();

1. **Write a script to select specified list option in the list box.**

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

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

driver.get("C:\\GCVSEL\\GCVJAVA\\src\\SeleniumClass\\MultiSelectListBoxWebPage.html");

Select ele = **new** Select(driver.findElement(By.*xpath*("//select[@id='s1']")));

List<WebElement> allopt = ele.getOptions();

**for**(**int** i=0; i<allopt.size(); i++)

{

**if**(allopt.get(i).getText().equals("IND"))

{

allopt.get(i).click();

**break**;

}

}

driver.close();

**Assignments:**

1. Write a script to print all the list items in sorted order.

**Hint:** get all the options present in the list box then get text of each option, store it in array list and use sort method of collections.

1. Write a script to verify whether the content of the list box is sorted or not.
2. Write a script to print all the selected options in the multiselect list box.

**Hint:** getAllSelectOptions() method of select class.

1. Write a script to verify whether **United States** is selected or not in [www.fatcow.com](http://www.fatcow.com)

**Hint:**

1. User **getFirstSelectedOption()** method of select class.’
2. Identify the option using xpath use findElement than use isSelected method.

IMP Note: If the list box is developed without using select html tag, in that case we get UnExpectedTagNameException in such case we can’t use select class. We should use the method available under WebElement interface such as: click(), clear(), sendKeys(), and we should write the appropriate code to select the required option.

**Example:**

WebElement list = driver.findElement(By.id(“X”));

list.clear();

list.sendKeys(“Moss”);

list.sendKeys(Keys.ENTER);

**Handing Dropdown Menu:**

Dropdown menu is an element on which if we more the mouse pointer it will display list of options. (sub-menus)

We use **move to element** method of **Actions** class whenever we use any method of **Actions** class we must always use **perform()** method.

**Example: to select menu item by using mouse event.**

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

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

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

WebElement menu = driver.findElement(By.*xpath*("//select[@id='countrySelect']"));

Actions act = **new** Actions(driver);

act.moveToElement(menu).perform();

driver.findElement(By.*xpath*("//option[@value='CAD']")).click();

Thread.*sleep*(2000);

driver.close();

1. How do you handle context menu.

Ans: Right click on any element is called context click. This can be done by using ContextClic() method of Actions class.

When we right click on any element it will display list of options, which is called as context menu.

To Select required option present in the context menu, we type the short cut such as: T for New Tab, W for new Window using sendKeys method of Actions class.

**Example:**

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

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

driver.get("http://127.0.0.1:8090/login.do");

WebElement ele = driver.findElement(By.*xpath*("//a[@id='licenseLink']"));

Actions act = **new** Actions(driver);

//Right click on specified element

act.contextClick(ele).perform();

act.sendKeys("Link").perform();

Thread.*sleep*(4000);

act.sendKeys(Keys.***ESCAPE***).perform();

//Left click on specified element

act.click().perform();

Thread.*sleep*(4000);

driver.quit();

**Assignment:**

1. Open Yatra.com
2. Move the mouse over customer support menu
3. Print all the displayed sub menus
4. Search whether sub-menu has contact us.
5. If its present select it.
6. **How do you Drag and Drop element in selenium?**

Ans: Using drag and drop method of actions class.

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

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

driver.get("https://jqueryui.com/droppable/");

**driver.switchTo().frame(0); //switching frame to select the element.**

WebElement source1 = driver.findElement(By.*xpath*("//div[@id='draggable']"));

WebElement target1 = driver.findElement(By.*xpath*("//div[@id='droppable']"));

String chk1 = driver.findElement(By.*xpath*("//div[@id='droppable']/p")).getText();

System.***out***.println("BeforeDrap: "+chk1);

Actions act = **new** Actions(driver);

act.dragAndDrop(source1, target1).perform();

String chk2 = driver.findElement(By.*xpath*("//div[@id='droppable']/p")).getText();

System.***out***.println("AfterDrap: "+chk2);

Thread.*sleep*(4000);

driver.close();

Note: Drag and Drop may not work if there is change in the pixel resolution in the application.

1. **How do you double click on the element?**

Ans: Using doubleClick() method.

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

driver.get("http://artoftesting.com/sampleSiteForSelenium.html");

WebElement ele = driver.findElement(By.*xpath*("//button[@id='dblClkBtn']"));

Actions act = **new** Actions(driver);

act.doubleClick(ele).perform();

Thread.*sleep*(2000);

driver.close();

**Actions Class Methods:**

1. moveToElement()
2. contextClick()
3. sendKeys()
4. dragAndDrop()
5. doubleClick()
6. perform()
7. **What is composite action how do you perform?**

Ans: performing more than one action at the same time is called as composite action, this can be done using Actions class before using perform() we should use build() method as shown below:

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

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

WebElement ele = driver.findElement(By.*xpath*("//a[@class='cta get-started']"));

Actions act = **new** Actions(drier);

//control+click opens page in new tab, where a only click opens page in same tab.

act.sendKeys(Keys.***CONTROL***).build().perform();

act.click(ele).build().perform();

//other example if case of composite operation. (not tested)

//act.sendKeys(Keys.CONTROL).click().build().perform();

//act.sendKeys(Keys.CONTROL,Keys.SHIFT).dragAndDrop(source,target).build().perform();

Thread.*sleep*(4000);

driver.quit();

**Handling Pop-up’s:**

**Note: Listeners features used to handle uncertain pop-up’s. Work on ???**

In selenium handling pop-up’s depends on the type of the pop-up.

In order to identify the popup properly always use the browser, which is opened by selenium.

In selenium we have following major pop-up’s:

1. Authentication pop-up
2. Hidden division pop-up
3. File upload pop-up
4. File download pop-up
5. Alert and Confirmation pop-up
6. Child browser pop-up
7. Window pop-up.
8. **Authentication pop-up**

Its also called as page on load pop-up.

1. Characteristics of this pop-up is displayed while loading the webpage.
2. We can move the pop-up.
3. Pop-up contain following fields.
   1. User name
   2. Password
   3. OK
   4. Cancel
4. We can’t inspect the pop-up.

**Soln:**

To handle authentication pop-up, we specify username & password in URL itself as shown below.

driver.get(“http://username:password@localhost/sm/automation/”);

If user name and pwd has special char (@) than above solution will not workl, in such cases we use 3rd party s/w such as **AutoIT.**

1. **Hidden division pop-up**

**Soln:**

This pop-up is developed using div HTML tag & initially it will be invisible hence its called as hidden division pop-up.

Sometimes developer creates there pop-ups using AJAX (Asynchronous Java Script and XML) to handle such type of pop-up also use a concept which is used to handle hidden division pop-ups.

**Characteristics:**

1. Can’t move the pop-up.
2. We can inspect
3. It will be colorful.

Note: Most of the calendar pop-up are hidden division pop-ups.

**Soln:**

Since we can inspect the pop-up, we handle it using findElement() method itself.

1. **Write a script to perform following steps:**
2. Open web site
3. Get the message displayed on hidden division pop-up and print.
4. Close the hidden division pop-up.

Selenium script:

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

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

driver.findElement(By.*xpath*("//input[@id='BE\_flight\_origin\_date']")).click();

WebElement ele = driver.findElement(By.*xpath*("//div[@class='active-month-holder']"));

System.***out***.println(ele.getText());

driver.findElement(By.*xpath*("//td[@id='27/09/2018']")).click();

Thread.*sleep*(3000);

driver.close();

1. **Write a script to perform the following.**
2. Open the login page of ActiTime
3. Enter valid user name and password.
4. Click on login
5. Create one task.
6. Click on tasks
7. Select all tasks
8. Click on delete.
9. Verify that hidden popup is displayed
10. If its displayed print the message present on popup.
11. Click on delete task.
12. Logout from the application.

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

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

//launch and login to actiTime.

driver.get("http://127.0.0.1:8090/login.do");

driver.findElement(By.*id*("username")).sendKeys("user1");

driver.findElement(By.*name*("pwd")).sendKeys("user1");

driver.findElement(By.*xpath*("//a[@id='loginButton']")).click();

//create task

driver.findElement(By.*xpath*("//a[@class='content tasks']")).click();

driver.findElement(By.*xpath*("//span[text()='Create Tasks']")).click();

driver.findElement(By.*xpath*("//select[@name='customerId']")).sendKeys("Vishnu");

driver.findElement(By.*xpath*("//select[@name='projectId']")).sendKeys("ROCKON");

driver.findElement(By.*xpath*("//input[@id='task[0].name']")).sendKeys("Task111");

driver.findElement(By.*xpath*("//input[@id='task[0].budgetedTimeStr']")).sendKeys("3:00");

driver.findElement(By.*xpath*("//img[@id='ext-gen7']")).click();

driver.findElement(By.*xpath*("//a[@class='x-date-date']")).sendKeys("Sep 27, 2018");

driver.findElement(By.*xpath*("//select[@id='task[0].billingType']")).sendKeys("Billable");

driver.findElement(By.*xpath*("//input[@id='task[0].markedToBeAddedToUserTasks']")).click();

driver.findElement(By.*xpath*("//input[@value='Create Tasks']")).click();

Thread.*sleep*(3000);

//select select ALL check box.

driver.findElement(By.*xpath*("//a[@class='listtblcolheadersmall']")).click();

//click on delete button.

driver.findElement(By.*xpath*("//input[@class='hierarchy\_element\_wide\_button']")).click();

Thread.*sleep*(3000);

//get Delete message popup.

String msg = driver.findElement(By.*xpath*("//td[@id='deleteAttention']")).getText();

System.***out***.println(msg);

//click on delete pop up.

driver.findElement(By.*xpath*("//input[@id='deleteButton']")).click();

Thread.*sleep*(3000);

//logout form actiTime and close the browser.

driver.findElement(By.*xpath*("//a[@id='logoutLink']")).click();

driver.close();

Note: To verify whether hidden division popup is displayed or not use isDisplayed() method for boundary element, that is the element which as div html tag with display:block attribute to get the complete text, we should use getText() method for boundary element only.

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

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

driver.findElement(By.*xpath*("//input[@id='BE\_flight\_origin\_date']")).click();

**try**{

WebElement ele = driver.findElement(By.*xpath*("//div[@class='active-month-holder']"));

**if**(ele.isDisplayed())

{

System.***out***.println(ele.getText());

}

**else**

{

System.***out***.println("Not displayed");

}

}

**catch**(NoSuchElementException e){

System.***out***.println("Not Displayed...");

}

driver.close();

**Write a script to perform following steps:**

1. Open Yatra.com
2. Specify Bengaluru in leaving from
3. Specify Goa in going to
4. Select a date
5. Click on flights.

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

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

WebElement source = driver.findElement(By.*xpath*("//input[@id='BE\_flight\_origin\_city']"));

source.sendKeys("Bangalore, India (BLR)");

source.sendKeys(Keys.***ENTER***);

WebElement destination = driver.findElement(By.*xpath*("//input[@id='BE\_flight\_arrival\_city']"));

destination.sendKeys("Goa, India (GOI)");

destination.sendKeys(Keys.***ENTER***);

driver.findElement(By.*xpath*("//input[@id='BE\_flight\_origin\_date']")).click();

driver.findElement(By.*xpath*("//td[@id='27/09/2018']")).click();

driver.findElement(By.*xpath*("//input[@id='BE\_flight\_flsearch\_btn']")).click();

driver.close();

Questions:

1. How selenium performs action on the application.

Ans: By calling Native Method of the browser.

2. How selenium works internally or which protocol selenium uses internally.

Ans: JSON Wire Protocol (JavaScriptObjectNotation)

3. If selenium opens a browser (MFF) fire bug will not be present why?

Ans: Selenium opens Firefox browser with factory settings. i.e no add-on’s, no history, & no cookies.

4. How do you open browser with add-on’s

Ans: System.setProperties(“webdriver.firefox.driver”,”defalut”);

WebDriver driver = new FirefoxDriver();

5. How do you delete cookies present in the browser.

Ans: diver.manage().deleteAllCookies();

6. How do you handle text box if its readonly or disabled.

Ans: Using JavaScriptExecuter

Put date value 25/12/2015 (for departure)

Thru JavaScript.

1. **File upload pop-up**

**Characteristics:**

1. There will be browse button clicking on which it will display the pop-up with title, FileUpload. (This popup is used to select the required filed)
2. We can move the popup but we can’t inspect it.
3. We can inspect the browse button and if we look at the source code html tag should be <input type=file …

**Soln:**

To handle file upload popup we use sendKeys() method, and for sendKeys() methods we specify absolute path of the file using ‘\\’ double backward slash as shown below:

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

driver.get("https://the-internet.herokuapp.com/upload");

WebElement ele1 = driver.findElement(By.*xpath*("//input[@id='file-upload']"));

ele1.sendKeys("C:\\Users\\vishn\\Documents\\TestFile.txt");

driver.findElement(By.*xpath*("//input[@id='file-submit']")).click();

WebElement ele2 = driver.findElement(By.*xpath*("//div[@id='uploaded-files']"));

String s = ele2.getText();

**if**(s.equals("TestFile.txt"))

{

System.***out***.println("File Upload Success");

}

**else**

{

System.***out***.println("File Upload Fail");

}

driver.close();

Note: Relative path is not allowed, forward ‘//’ slash not allowed, if path is invalid, file does not exists, file size is more than the limit, file format is not supported (Ex: .exe) in such cases application will display error message, after clicking on upload button that we should handle separately.

IMP: if we have file upload option which is not created using <input type=’file’ > like in gmail or any than sendKeys() method will not work. We should handle using **AutoIT.**

1. **File download pop-up**

**Characteristics:**

* 1. We can move the popup.
  2. We can’t inspect the popup.
  3. It will have open with (save file radio button along with do this automatically checkbox)
  4. OK and Cancel button.

**Soln:**

To handle file download popup we use setPreference() method of FirefoxProfile class.

1. Settings is called as preferences
2. Group of preferences is called as profile.
3. Every time selenium open firefox browser with default profile (factory settings)
4. To change these settings we use setPreference method of fireforProfile class it takes 2 arguments key and value.

In order to manually modify the settings.

1. Open the browser
2. Type about:confid in the address bar & enter
3. Click on button (be careful)
4. It will display all the preferences.
5. We must always change the settings before opening the browser. (WebDriver driver = new FirefoxDriver() )
6. All the settings their syntax and the usage is available in the following web page

<http://kb.mozillaazine.org/about:config_entries>

1. We should specify file format for which browse should automatically download it which is called MIME (multipurpose internet mail exchange) for this we can use following website:

<http://www.handenb.pdx.edu/DMKB/dict/tutorials/mime_typ.php>

Ex: //factory setting (NOTE: Below code not tested or practiced!!!!!!!!!!!!!)

FirefoxProfile profile = **new** FirefoxProfile();

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

String value = "appliation/zip";

//save the file to 0--> Desktop, 1-->Downloads, 2-->Other locaiton

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

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

WebDriver dreiver = **new** FirefoxDriverProfile();

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

String xp = "//dt[text()='Java]'/../td[4]/a";

driver.findElement(By.*xpath*(xp)).click();

1. **Alert and Confirmation pop-up**

**Characteristics:**

1. We can move the popup
2. We can’t inspect the popup
3. It will have OK button with warning symbol
4. If it has OK and Cancel button with question mark? Its called as confirmation popup.
5. It also called as JavaScript popup.

**Soln:**

To handle the alert pop up we should transfer the control to alert popup using **switchToAlert()** statement, if the alert is not present than we get **NoAlertException**. After transferring the control, we use following of alert class. **accept()** to click on OK button. **dismiss()** to click on Cancel or ‘X’ button.

Once alert popup is closed control will be transferred back to browser automatically. If we try to perform any action when the alert popup still exists than we get **UnhandledAlertException**.

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

driver.get("C:/GCVSEL/GCVJAVA/src/SeleniumClass/AlertWebPageOKCancel.html");

driver.findElement(By.*xpath*("//button[@onclick='myFunction()']")).click();

Thread.*sleep*(3000);

**Alert a = driver.switchTo().alert();**

System.***out***.println("First Alert: "+a.getText());

**a.accept();**

driver.findElement(By.*xpath*("//button[@onclick='myFunction()']")).click();

Thread.*sleep*(3000);

System.***out***.println("Second Alert: "+a.getText());

**a.dismiss();**

driver.close();

Interview Questions:

1. How do you handle multiple alert popups.

**Ans:** Browser will display only one alert popup at a time, it there are more than one popup at the same time, it makes popup may be other than alert popup or it can be combination of alert popup and another popup.

1. How do you get the tool tip text form the link.

Ans: get attribute title.

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

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

String tooltip = driver.findElement(By.*xpath*("//select[@class='nav-search-dropdown searchSelect']")).getAttribute("title");

System.***out***.println(tooltip);

driver.close();

1. Write a script to perform following menu.
   1. Open USA.com
   2. Get the tooltip.

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

WebElement ele = driver.findElement(By.*xpath*("//area[@href='/oregon-state.htm']"));

System.***out***.println(ele.getAttribute("title"));

1. **Child browser pop-up**

**Characteristics:**

1. We can move the popup.
2. We can inspect
3. It has address bar
4. It has min, max and close.
5. It has icon. (browser icon)

**Soln:**

To handle child browser popup we should use switch to window statement.

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

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

driver.get("https://wordpress.com/log-in?redirect\_to=http%3A%2F%2Fabodeqa.wordpress.com%2F");

WebDriverWait wait = **new** WebDriverWait(driver, 20);

wait.until(ExpectedConditions.*elementToBeClickable*(By.*xpath*("//span[text()='Continue with Google']")));

driver.findElement(By.*xpath*("//button[@class='social-buttons\_\_button button']")).click();

String mw = driver.getWindowHandle();

**for** (String cw : driver.getWindowHandles()) {

**if**(!cw.equals(mw)){

driver.switchTo().window(cw);

driver.findElement(By.*xpath*("//input[@type='email']")).sendKeys("VISHNU123");

Thread.*sleep*(3000);

driver.close();

}

}

driver.switchTo().window(mw);

driver.close();

Interview Questions:

1. What is a window handle?

**Ans:** It’s a unique alphanumeric string of a browser.

1. Difference between getWindowHandle() and getWindowHandles()

**Ans:**

getWindowHandle() **returns window handle of the current browser.**

getWindowHandles() windows handles of all browsers.

1. Write a script to print number of browsers opened by selenium and also print window handle of all the browsers.

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

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

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

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

**int** count = AW.size();

System.***out***.println(count);

**for**(String W : AW)

{

System.***out***.println(W);

}

driver.quit();

1. Write a script to print title of the browser and close all child browsers.

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

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

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

String mw = driver.getWindowHandle();

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

**int** count = AW.size();

System.***out***.println(count);

**for**(String W : AW)

{

**if**(!W.equals(mw))

{

driver.switchTo().window(W);

System.***out***.println(driver.getTitle());

driver.close();

System.***out***.println(W);

}

}

driver.switchTo().window(mw);

driver.close();

1. Write a script to close specified browsers.

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

driver.manage().timeouts().implicitlyWait(10, TimeUnit.***SECONDS***);

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

String mw = driver.getWindowHandle();

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

**int** count = AW.size();

System.***out***.println(count);

**for**(String W : AW)

{

**if**(!W.equals(mw))

{

driver.switchTo().window(W);

**if**(driver.getTitle().equals("Amazon"))

{

System.***out***.println(driver.getTitle());

System.***out***.println(W);

driver.close();

}

}

}

driver.quit();

6. Write a script close only parent browser.

**Ans:** driver.close();

7. Write a script to close all the child brosers.

**Ans: if**(!W.equals(mw))

{

driver.switchTo().window(W);

driver.close();

}

1. How do you handle new tab in selenium?

Ans: Not possible in selenium there is no method to switch to new tab.

Note: When we click on link manually and if it opens the page in new tab, than same page will be opened by the selenium in new browser forcefully, so that we can use child browser concept.

**Handling Frame:**

Web page present inside another web page is called a embedded web page. Developers create embedded web page using iframe or fromeset HTML tag.

**This Frame >** in the context menu displays when you right click on element.

If the element is present inside the frame than we should switch control into the frame, before performing any actions on the web element, using following statement.

driver.switchTo().frame(arg1);

Frame is overloaded method we can pass:

* Index of the frame.
* Name of the frame.
* Web Element of the frame.

In order to transfer the control back to the main page we should use following statement.

driver.switchTo().defaultContent();

driver.switchTo().defaultContent(); 🡪 bring control back to the Main window.

**driver.switchTo().parentFrame();** 🡪 bring control back to the parent frame. (use this when you have frame have child frames.)

1. Browser has only one frame and control is transferred to the frame, how do you switch back to the browser, tell me diff ways.

**Ans:**

* 1. driver.switchTo().defaultContent();
  2. driver.switchTo().parentFrame();
  3. driver.navigate().refresh();

**IMP Note:** In order to write xpath expression for the element which is inside the frame, right click on the element select inspect using chrome or Firefox and write xpath.

1. **Windows pop-up.**

If popup displayed on the application do not belongs to any of the first discussed popup category, it is called as window popup.

To handle window popup there is no solution in selenium hence we use their party tool: **AutoIT**. (by using doc or video download AutoIT and learn to edit and compile AutoIT)

**File Upload Using AutoIT: (and Download)**

Write two AutoIT script files using AutoIT editor:

*ControlFocus("Open","","Edit1")*

*ControlSetText("Open","","Edit1","C:\Users\vishn\Desktop\test.docx")*

*ControlClick("Open","","Button1")*

Save file with file name: **FileUpload**

Right click on the AutoIT file compile using Win64 compile option.

Take the path of the file and use at Runtime.getRuntime().exec(“Path”);

Similarly create one more file with fine name: **FileDelete** with below code:

*FileDelete("C:\Users\vishn\Downloads\wordtojpeg.zip")*

Right click on the AutoIT file compile using Win64 compile option.

Take the path of the file and use at Runtime.getRuntime().exec(“Path”);

**Write the below selenium script to upload and delete the file using AutoIT integration.**

**(Note: Below website take more time, use diff website and xpath to test the same)**

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

driver.get("https://wordtojpeg.com/");

driver.findElement(By.*xpath*("//div[@id='pick-files']")).click();

Thread.*sleep*(3000);

Runtime.*getRuntime*().exec("C:\\GCVSEL\\AutoIT\\FileUpload.exe");

WebDriverWait wait = **new** WebDriverWait(driver, 120);

wait.until(ExpectedConditions.*elementToBeClickable*(By.*xpath*("//button[@id='download-all']")));

**boolean** enable = driver.findElement(By.*xpath*("//button[@id='download-all']")).isEnabled();

**if**(enable)

{

System.***out***.println("FileUploadPass");

}

**else**

{

System.***out***.println("FileUploadFail");

}

driver.findElement(By.*xpath*("//button[@id='download-all']")).click();

Thread.*sleep*(6000);

Runtime.*getRuntime*().exec("C:\\GCVSEL\\AutoIT\\FileDelete.exe");

driver.close();

**File Upload Using Robot:**

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

driver.get("https://wordtojpeg.com/");

driver.findElement(By.*xpath*("//div[@id='pick-files']")).click();

//Using StringSelection to take path into clipboard.

**//Note: Using java.awt import for Robot.**

StringSelection stringSelection = **new** StringSelection("C:\\Users\\vishn\\Desktop\\test.docx");

Toolkit.*getDefaultToolkit*().getSystemClipboard().setContents(stringSelection, **null**);

//Using Robot with KeyEvents

Robot robot = **new** Robot();

robot.keyPress(KeyEvent.***VK\_CONTROL***);

robot.keyPress(KeyEvent.***VK\_V***);

robot.keyRelease(KeyEvent.***VK\_V***);

robot.keyRelease(KeyEvent.***VK\_CONTROL***);

robot.keyPress(KeyEvent.***VK\_ENTER***);

robot.keyRelease(KeyEvent.***VK\_ENTER***);

WebDriverWait wait = **new** WebDriverWait(driver, 120);

wait.until(ExpectedConditions.*elementToBeClickable*(By.*xpath*("//button[@id='download-all']")));

**boolean** b = driver.findElement(By.*xpath*("//button[@id='download-all']")).isEnabled();

**if**(b)

System.***out***.println("FileUpload Pass");

**else**

System.***out***.println("FileUpload Fail");

driver.close();

**File Download Using Robot: (Note: If PopUp Save Window to use can’t use this)**

driver.manage().timeouts().implicitlyWait(20, TimeUnit.***SECONDS***);

driver.get("https://wordtojpeg.com/");

driver.findElement(By.*xpath*("//div[@id='pick-files']")).click();

//Using StringSelection to take path into clipboard

StringSelection stringSelection = **new** StringSelection("C:\\Users\\vishn\\Desktop\\test.docx");

Toolkit.*getDefaultToolkit*().getSystemClipboard().setContents(stringSelection, **null**);

//Using Robot with KeyEvents

Robot robot = **new** Robot();

robot.keyPress(KeyEvent.***VK\_CONTROL***);

robot.keyPress(KeyEvent.***VK\_V***);

robot.keyRelease(KeyEvent.***VK\_V***);

robot.keyRelease(KeyEvent.***VK\_CONTROL***);

robot.keyPress(KeyEvent.***VK\_ENTER***);

robot.keyRelease(KeyEvent.***VK\_ENTER***);

Thread.*sleep*(12000);

driver.findElement(By.*xpath*("//button[@id='download-all']")).click();

//Uing left and etner keys we can download to default locatoin.

robot.keyPress(KeyEvent.***VK\_LEFT***);

robot.keyPress(KeyEvent.***VK\_ENTER***);

**Interview Question:**

1. How do you get first selected value in the dropdown list?

Ans: select.getFirstSelectOption().getTest()

1. Syntax to switch back from the frame?

Ans: driver.switchTo().defaultContent();

1. How do you right click on a link?

Ans: actions.contextClick().perform();

1. Which methods is used to check is selected?

Ans: isSelected()

1. Method which selects the option by the text displayed on the option?

Ans: selectByVisibleText();

Note: To check XPATH in chrome use Ctrl+F.

------------------------------------------------------------------------------------------

Run Firefox in Headless mode:

FirefoxBinary fb = **new** FirefoxBinary();

fb.addCommandLineOptions("--headless");

System.*setProperty*("webdriver.chrome.driver", "C:\\geckodriver\\geckodriver.exe");

FirefoxOptions fo = **new** FirefoxOptions();

fo.setBinary(fb);

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

driver.get("http://localhost:8090/login.do");

System.***out***.println(driver.getTitle());

------------------------------------------------------------------------------------------

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

**Framework Important Features**

**TestNG:**

Test Next Generation is a Unit Testing Framework, basically used by developers to automatically run all the unit test scripts.

In Selenium we use TestNG to perform the following steps:

1. Run all test classes.
2. Generate the result
3. Run Only failed test cases.

TestNG suite is an xml file, used to execute multiple TestNG classes. Below is the sample: **(Refer doc or video to install TestNG and testng.xml)**

**File: testing.xml**

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >

<suite name=*"MyActiTimePrj"* parallel = *"none"*>

<test name = *"ActiTime"*>

<classes>

<class name = *"com.testng.AirthmaticException"*> </class>

<class name = *"com.testng.TestNGActiTimeTest"*> </class>

<class name = *"com.testng.TestNGAnnotationsTest"*> </class>

<class name = *"com.testng.TestNGGoogleTest"*> </class>

<class name = *"com.testng.TestNGGroupingTest"*> </class>

<class name = *"com.testng.TestNGTest"*> </class>

</classes>

</test>

</suite>

**OR: File: testing.xml**

**<!-- BELOW IS ALTERNATIVE WAY TO DESIGN TESTNG.XML -->**

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >

<suite name=*"MyActiTimePrj"* parallel = *"none"*>

<test name = *"ActiTime"*>

<packages>

<package name = *"com.testng"*></package>

</packages>

</test>

</suite>

1. How do you re-run only failed test cases?

**Ans:** Using testng-failed.xml which will be present under test-output folder.

Note: During run-time if any test method fails, it automatically adds into testng-failed.xml

1. How do you fail the script explicitly?

**Ans:** using Assert.fail();

1. How do you compare actual and executed values in TestNG?

**Ans:** Using Assert class. It is also called as assertion.

**Note:** Assert class of testing has following methods:

Different TestNG Annotations and other methods:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**//Before and After annotations sequence**

@BeforeSuite

**public** **void** sysConfig()

{

System.***out***.println("@BeforeSuite: sysConfig");

}

@BeforeTest

**public** **void** testConfig()

{

System.***out***.println("@BeforeTest: testConfig");

}

@BeforeClass

**public** **void** setUp()

{

System.***out***.println("@BeforeClass: setUP");

}

@BeforeMethod

**public** **void** testSetup()

{

System.***out***.println("@BeforeMethod: testSetup");

}

**@Test(priority=1)**

**public void priorityTestMethod()**

**{**

**System.*out*.println("@Test: priorityTestMethod");**

**}**

@AfterMethod

**public** **void** clearTestSetup()

{

System.***out***.println("@AfterMethod: clearTestSetup");

}

@AfterClass

**public** **void** clearSetUp()

{

System.***out***.println("@AfterClass: clearSetUp");

}

@AfterTest

**public** **void** clearTestConfig()

{

System.***out***.println("@AfterTest: clearTestConfig");

}

@AfterSuite

**public** **void** clearSysConfig()

{

System.***out***.println("@AfterSuite: clearSysConfig");

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**//Test case with dependsOnMethod**

@Test

**public** **void** loginToApp()

{

System.***out***.println("@Test: loginToApp");

}

@Test(dependsOnMethods="loginToApp")

**public** **void** mainPage()

{

System.***out***.println("@Test: mainPage");

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**//Test case with expectedExceptions**

@Test(expectedExceptions=AirthmaticException.**class**)

**public** **void** excludeExceptionMethod()

{

**int** c = 10/0;

System.***out***.println("@Test: excludeExceptionMethod");

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**//Test case with Hard Assertion or Assert**

@Test

**public** **void** hardAssertBooleanTest()

{

System.***out***.println("@Test: HardAssertBoolenaTest");

**boolean** x = **true**;

Assert.*assertTrue*(x);

}

@Test

**public** **void** hardAssertEqualsTest()

{

String x = "Vishnu";

System.***out***.println("@Test: hardAssertEqualsTest");

Assert.*assertEquals*(x, "Vishnu", "HardAssert: Actual is not matching with expcted");

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**//Test case with SoftAssert**

@Test

**public** **void** softAssertTest()

{

String x = "VishnuGC";

SoftAssert softAssert = **new** SoftAssert();

softAssert.assertEquals(x, "Vish", "SoftAssesrt Fial First");

softAssert.assertEquals(x, "VishnuGC", "SoftAssesrt Pass after Faile");

**boolean** b = **false**;

softAssert.assertEquals(b, **true**);

softAssert.assertAll();

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**//Test case with invocationCount**

@Test(invocationCount=10)

**public** **void** invocationCount()

{

System.***out***.println("@Test: invocationCount10");

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**//Test case with invocationTimeout and Disabling testcase using enabled=false**

@Test(invocationTimeOut=2000,enabled=**false**)

**public** **void** invocationTimeOutTest()

{

System.***out***.println("@Test: invocationTimeOutTest");

**int** i=1;

**while**(i==1)

{

}

}

1. What is the limitation of assert statement how do you handle it.

Ans: If comparison fails it will not executed remaining statements of the current test method. In order to handle this, we can use SoftAssert which has non-static methods. Above is the Example.

Note: If we don’t call assertAll() it will not write results into the results window.

softAssert.assertEquals(b, **true**);

softAssert.assertAll();

Any statement written after assertAll(); will not execute if verification fails.

**Difference between Assert and SoftAssert**

**Assert**

1. if verification fails it will not execute remaining statements of current test method.

2. All the methods are static

3. We do not call assertALL method.

**softAssert()**

1. if verification fails also, it will execute remaining statements in the test method.

2. All the methods are non-static.

3. We must call assertAll();

1. Can we have more then one test method in TestNG class?

Ans: Yes, if multiple test methods present execution order will be in alphabetic order.

1. How do you control order of execution of test methods?

Ans: using priority @Test(priority=1) or @Test(priority=-1)

1. What is the default priority?

Ans: zero (0)

1. Can we use variable for priority value?

Ans: No.

1. Can we use negative number in priority?

Ans: Yes.

1. Can we use fraction number in priority?

Ans: No (0.2 or 0.34 not allowed)

1. Is it mandatory to specify priority in order?

Ans: No.

1. How do you create dependency in TestNG?

Ans:

**//Test case with dependsOnMethod**

@Test

**public** **void** loginToApp()

{

System.***out***.println("@Test: loginToApp");

}

@Test(dependsOnMethods={"loginToApp"})

**public** **void** mainPage()

{

System.***out***.println("@Test: mainPage");

}

Note: if two methods are dependent on each other we get **TestNGException** with the error: **CyclicDependencies”**

1. How do you disable execution of a test method?

Ans: @Test(enabled=false)

1. How do you execute test methods multiple times?

Ans: @Test(invocationCount=10) (Negative and fractions are not allowed)

1. How do you disable test method without using enabled=false?

Ans: @Test(invocationCount=0)

**DataProvider in TestNG:**

**POM Class Page Factory Definition:**

@FindBy(xpath="//select[@name='title']")

WebElement selectTitle;

@FindBy(id="first\_name")

WebElement firstName;

@FindBy(id="surname")

WebElement lastName;

@FindBy(name="client\_lookup")

WebElement companyName;

@FindBy(id="company\_position")

WebElement companyPosition;

@FindBy(xpath="//input[@type='submit' and @value='Save']")

WebElement saveBtn;

@FindBy(xpath="//td[@class='datacardtitle']")

WebElement savedContactInfoConfirmation;

**POM Class Page Factory Initialization:**

//Initialization

**public** NewContactPage() {

PageFactory.*initElements*(*driver*, **this**);

}

**POM Class validation Method:**

**public** String validateCreateContactInfo(String vtitle,String vfirstName,String vlastName,String vcompanyName,String vcompanyPosition) {

Select sel = **new** Select(*driver*.findElement(By.*name*("title")));

sel.selectByVisibleText(vtitle);

firstName.sendKeys(vfirstName);

lastName.sendKeys(vlastName);

companyName.sendKeys(vcompanyName);

companyPosition.sendKeys(vcompanyPosition);

saveBtn.click();

String vinfo = savedContactInfoConfirmation.getText();

**if** (vinfo.contains(vfirstName))

**return** vfirstName;

**else**

**return** "Invalid";

}

**TestClass @DataProvider Annocation:**

@DataProvider

**public** Object[][] getCRMTestData(){

Object data[][] = TestUtil.*getTestData*(sheetName);

**return** data;

**TestClass @Test Annotation and test Method:**

@Test(dataProvider="getCRMTestData")

**public** **void** verifyCreateContactInfo(String vtitle,String vfirstName,String vlastName,String vcompanyName,String vcompanyPosition) {

String x = newContactPage.validateCreateContactInfo(vtitle, vfirstName, vlastName, vcompanyName, vcompanyPosition);

Assert.*assertEquals*(x, vfirstName,"ContactInfoCreation Fail");

}

**Util Class Excel reading Method:**

//Excel sheet reading method.

**public** **static** Object[][] getTestData(String sheetName) {

FileInputStream file = **null**;

**try** {

file = **new** FileInputStream(*TESTDATA\_SHEET\_PATH*);

} **catch** (FileNotFoundException e) {

e.printStackTrace();

}

**try** {

*book* = WorkbookFactory.*create*(file);

} **catch** (InvalidFormatException e) {

e.printStackTrace();

} **catch** (IOException e) {

e.printStackTrace();

}

*sheet* = *book*.getSheet(sheetName

);

Object[][] data = **new** Object[*sheet*.getLastRowNum()][*sheet*.getRow(0).getLastCellNum()];

**for** (**int** i = 0; i < *sheet*.getLastRowNum(); i++) {

**for** (**int** k = 0; k < *sheet*.getRow(0).getLastCellNum(); k++) {

data[i][k] = *sheet*.getRow(i + 1).getCell(k).toString();

}

}

**return** data;

}

**CacheLookup:**

**import** org.openqa.selenium.support.CacheLookup;

@FindBy(xpath="//legend[text()='Contact Information']")

@CacheLookup

WebElement newContactInfoPage;

@FindBy(xpath="//select[@name='title']")

@CacheLookup

WebElement selectTitle;

This @CacheLookup used to increase the speed of execution. And it finds elements from memory cache pool. Usage: Need to use under page factory after declaration of @FindBy

It's good it will not read html every time to validate elements, it creates one cache memory for all the elements and read from there. But problem is, if web page gets refreshed, if DOM structure changed, it throws StaleElementException.

HTML: DOM

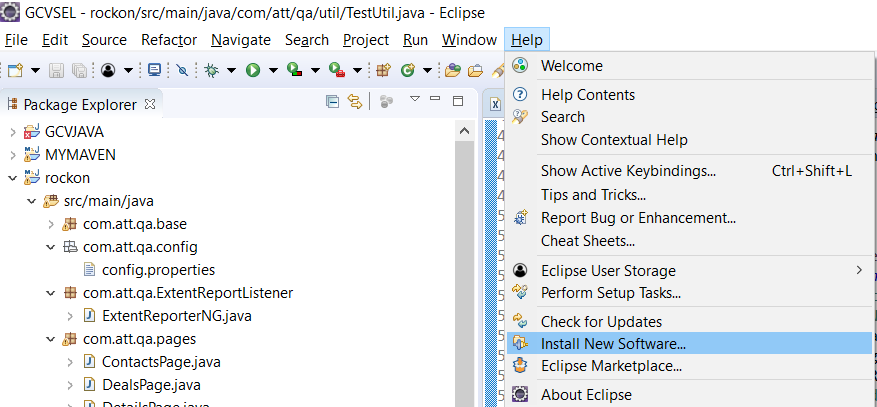
Application Looking pool for element instead of HTML: DOM

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

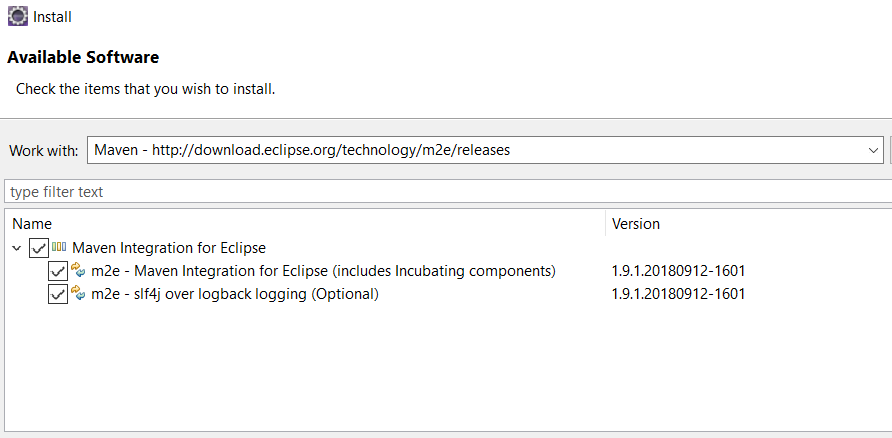
**Maven Setup and POM.xml Configuration:**

**Note: To Run multiple classes in Maven project, created testng.xml (testng.xml/testng\_smoke.xml/etc) under src/main/resources project folder and run thru testng.xml**

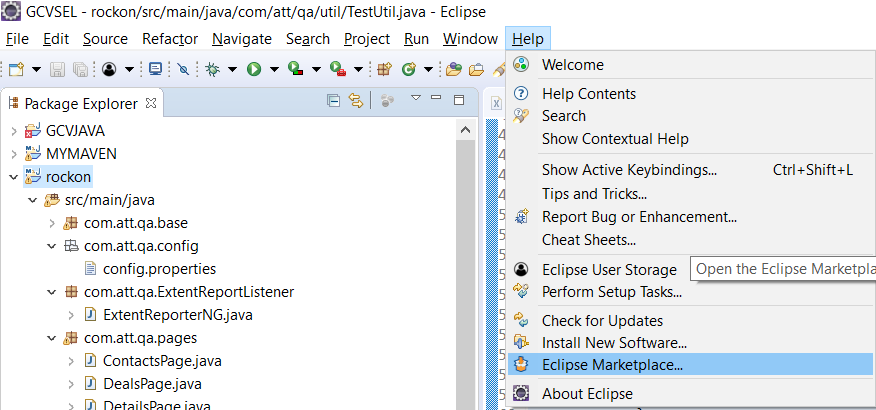
**Go to Help 🡪 Install New Software.**



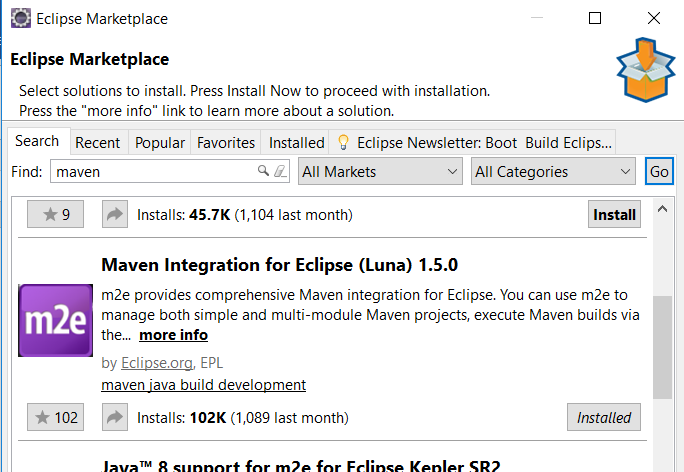
**Give name and URL as below by using Add option… Follow default install steps.**



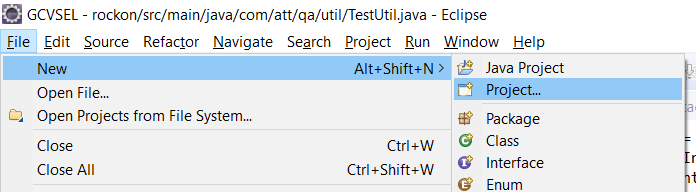
**Click on Help 🡪 eclipse Market place**



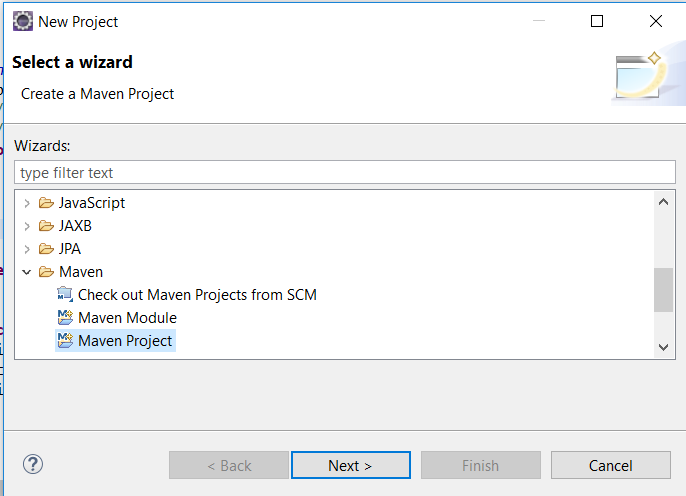
**Search for Maven and Install Maven after by using search result Install Maven Integration…**



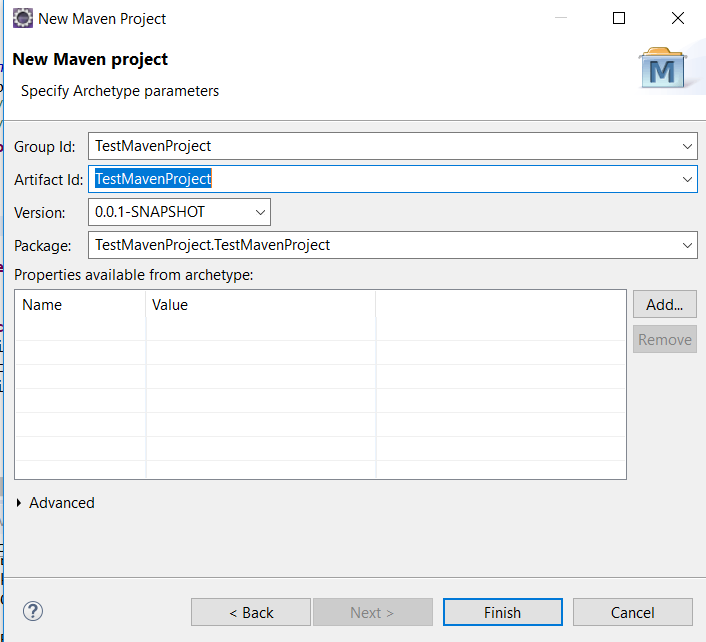
**Go to File 🡪 New 🡪 Project**



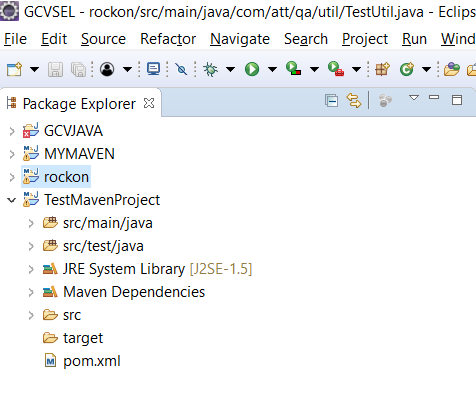
**Select Maven Project and continue by using defaults: (Till new Maven Project Naming window)**



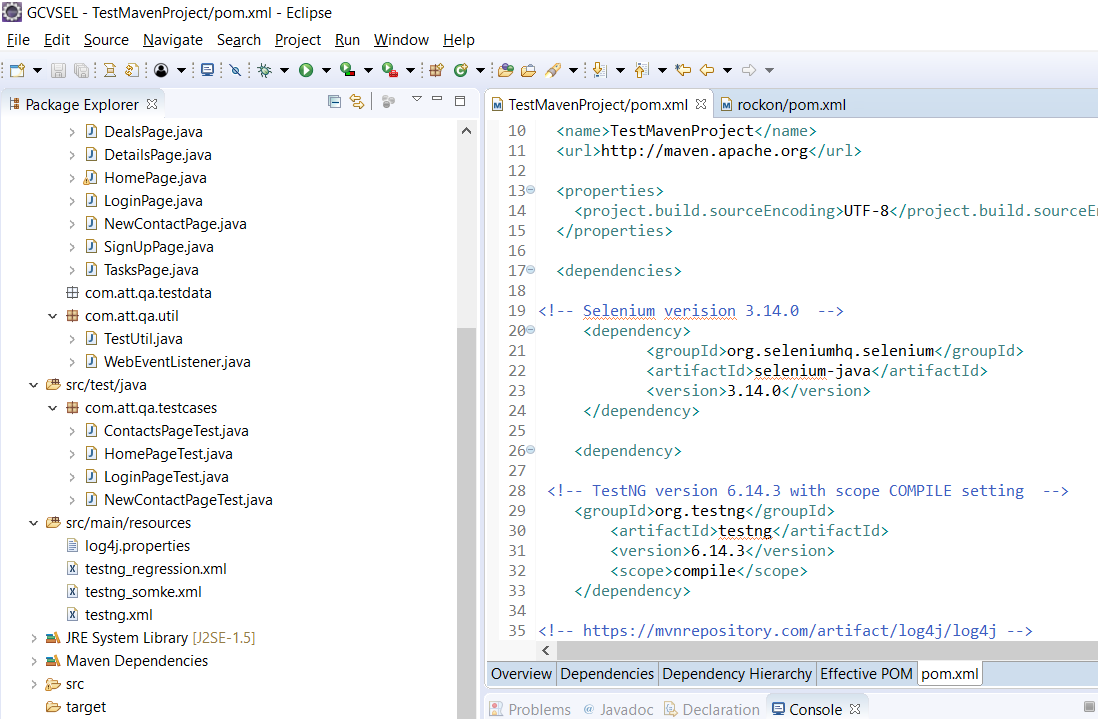
**Give GroupID and ArtifactID as show below:**



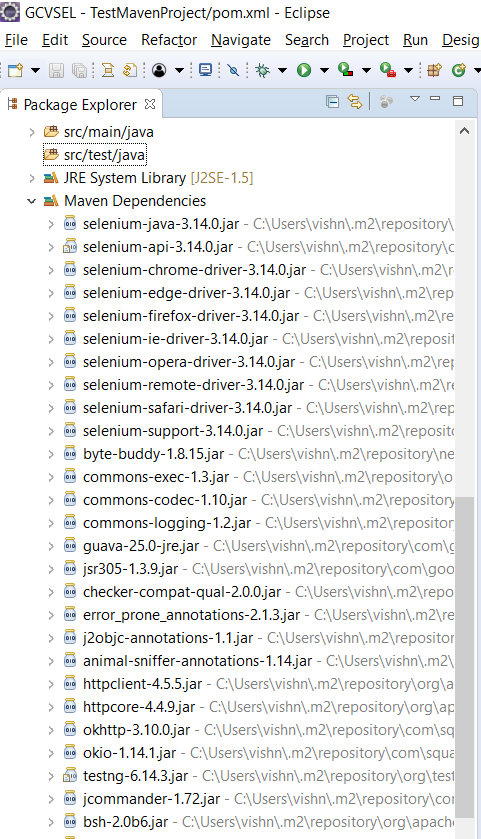
**It Creates Test Maven Project:**



**Open pom.xml file from the root directory of Maven project and remove Junit (if you need to use only for selenium) and add required packages:**



**After Saving pom.xml it automatically downloads given dependency jar files form the internet and under Maven Dependencies Folder as below:**



**Example: pom.xml:**

<https://docs.google.com/document/d/1ecjaZVStdKVHhVYJqp1pdrZpEqZ6ekM53tXBzCZyZh8/edit?usp=sharing>

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

**Listener Class: (To generate even logs)**

**Create WebEventListener class under below or any util in your maven project:**

**package** com.att.qa.util;

**Download class from below URL or google:**

**Example class name: public** **class** WebEventListener **extends** TestBase **implements** WebDriverEventListener

https://drive.google.com/file/d/1jnaCj\_GGwNaFwfCrMVbOak2WtesGx\_un/view?usp=sharing

**Listener defined in TestBase.java**

**public** **static** EventFiringWebDriver *e\_driver*;

**public** **static** WebEventListener *eventListener*;

//EvenListener declaration under initialization methods.

*e\_driver* = **new** EventFiringWebDriver(*driver*);

*eventListener* = **new** WebEventListener();

*e\_driver*.register(*eventListener*);

*driver* = *e\_driver*;

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

**Take Screenshot: methods and usage:**

**Write method under TestUtil.java**

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.FileNotFoundException;

**import** java.io.IOException;

**import** org.apache.commons.io.FileUtils;

**import** org.apache.poi.openxml4j.exceptions.InvalidFormatException;

**import** org.apache.poi.ss.usermodel.Sheet;

**import** org.apache.poi.ss.usermodel.Workbook;

**import** org.apache.poi.ss.usermodel.WorkbookFactory;

**import** org.openqa.selenium.OutputType;

**import** org.openqa.selenium.TakesScreenshot;

**public** **static** **void** takeScreenshotAtEndOfTest() **throws** IOException {

File scrFile = ((TakesScreenshot)*driver*).getScreenshotAs(OutputType.***FILE***);

String currentDir = System.*getProperty*("user.dir");

FileUtils.*copyFile*(scrFile, **new** File(currentDir +"\\screenshots\\"+ System.*currentTimeMillis*() + ".png"));

}

**onException methods calls takeScreenshotAtEndOfTest() under WebEventListener.java**

Note: if WebEventListener not created create and update method as below:

**public** **void** onException(Throwable error, WebDriver driver) {

System.***out***.println("Exception occured: " + error);

**try** {

TestUtil.takeScreenshotAtEndOfTest();

} **catch** (IOException e) {

e.printStackTrace();

}

}

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

**Extent Report:(For generating customized test report)**

**Update Extent Report dependency for pom.xml file:**

<!-- https://mvnrepository.com/artifact/com.relevantcodes/extentreports -->

<dependency>

<groupId>com.relevantcodes</groupId>

<artifactId>extentreports</artifactId>

<version>2.41.2</version>

</dependency>

**Update Extent Report listeners section in testng.xml file:**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >

<suite name=*"RockON Automation: Free CRM"*>

<listeners>

<listener

class-name = *"com.att.qa.ExtentReportListener.ExtentReporterNG"* />

</listeners>

<test name = *"Free CRM"*>

<classes>

<class name = *"com.att.qa.testcases.LoginPageTest"*> </class>

<class name = *"com.att.qa.testcases.HomePageTest"*> </class>

<class name = *"com.att.qa.testcases.ContactsPageTest"*> </class>

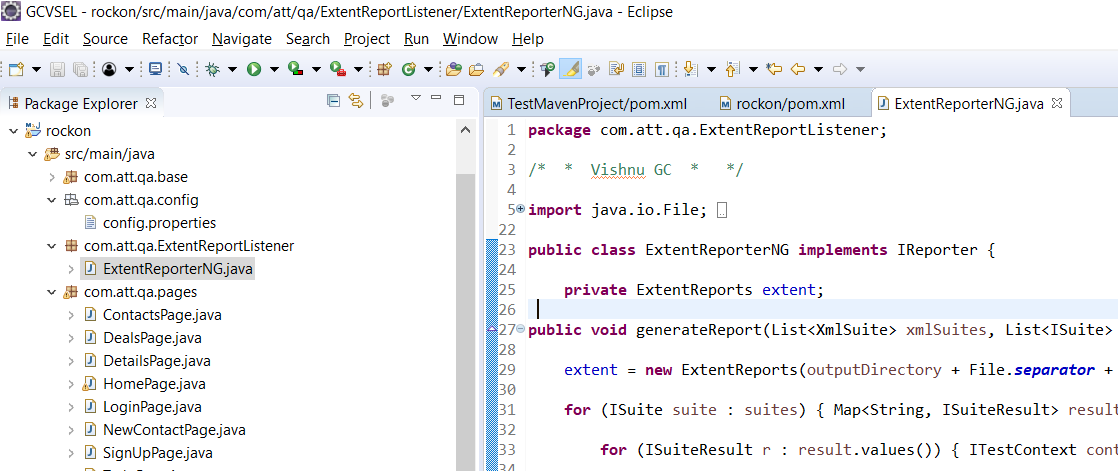
</classes>

</test>

</suite>

**Under the src.main create folder structure as shows and create class : ExtentReporterNG.class using below URL:**

<https://docs.google.com/document/d/19DlKIUdxYMJN8MdNl8EueJm7RNsMQfAf87rKJudoCsg/edit?usp=sharing>



$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

**Jenkins Install, Configure and Usage:**

**Refer the below link doc or YouTube:**

For Install.exe:

<https://jenkins.io/download/> download windows version and install.

For Password: C:\Program Files (x86)\Jenkins\secrets 🡪 get from this file: initialAdminPassword

Jenkin default URL: <http://localhost:8080/>

After that follow below WAR install doc or video to setup plugins, maven and testNG.

**For WAR install:**

This has explanation to integrate with Maven and TestNG. (refer other doc/video to integrate with GIT and other repositories)

<https://drive.google.com/file/d/1ZJ4ToCBDa23T1elyC06lDSEUtUfbwknW/view?usp=sharing>

<https://www.youtube.com/watch?v=lPxrPqSyCo8>

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

**GIT Install, Config and Usage:**

Using google download and install GIT software.

Using github url create new GIT account.

Using gitBash generate git private and public keys. (which is installed in your system)

**Ref Below video and URL:**

* https://www.youtube.com/watch?v=ZNER9rHjQ2Q
* https://help.github.com/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent/

**GIT PUSH:**

**Note:Open “GitBash”or “Command”prompt move to the Proj path and run below cmd’s..**

**Use below to FRESH Push code to GIT:**

* git init
* git add .
* git commit -m "Initial commit"
* git remote add origin youruser@yourserver.com:/path/to/my\_project.git
* git push origin master

**Use below to UPDATE Push code to GIT:**

* git status
* git add .
* git status
* git commit -m "Updated Home POM"
* git push origin master

**GIT PULL:**

**Existing project pull: (you stay in existing project path and do..)**

* Update something in git website repository.
* git pull
* see the changes in eclipse.

**New Project pull: (create new folder and stay in that folder and do..)**

* git init
* git clone <URL of git repo you want to pull>
* Example: git pull <https://github.com/vishnugit49/MYNEWREPO>

**GIT Commands from GITHUM website:**

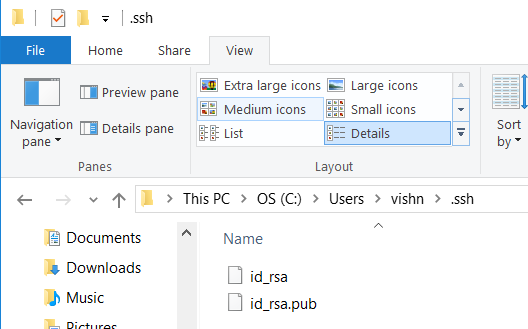
**…or create a new repository on the command line**

* echo "# ActiTimeMavenTestAutomation" >> README.md
* git init
* git add README.md
* git commit -m "first commit"
* git remote add origin https://github.com/vishnugit49/ActiTimeMavenTestAutomation.git
* git push -u origin master

**…or push an existing repository from the command line**

* git remote add origin https://github.com/vishnugit49/ActiTimeMavenTestAutomation.git
* git push -u origin master

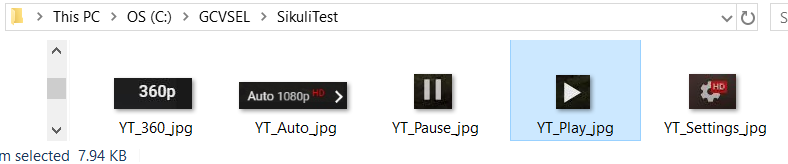
**GIT Key Gen path:**



$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

**Sikuli:**

1. Take Maven dependency form the website, update in pom.xml of Maven project.
2. Take screenshots of flash objects that need to click on store under Maven project. (example; YouTube play/pause in jpg or png formats)



1. Write below code to play with it…

driver.get("https://www.youtube.com/watch?v=MGC5V4WKDCk");

Screen s = **new** Screen();

Pattern playImg = **new** Pattern("YT\_Play\_jpg.JPG");

s.wait(playImg,3000);

Pattern pauseImg = **new** Pattern("YT\_Pause\_jpg.JPG");

s.wait(pauseImg,3000);

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$