# **Ajax Controls**

## How to work with Ajax Controls in WebDriver?

AJAX stands for Asynchronous JavaScript and XML. AJAX allows the web page to retrieve small amounts of data from the server without reloading the entire page. AJAX sends HTTP requests from the client to server and then process the server’s response without reloading the entire page. To handle AJAX controls, wait commands may not work. It’s just because the actual page is not going to refresh

To test Ajax application, different wait methods should be applied

* ThreadSleep
* Implicit Wait
* Explicit Wait
* WebdriverWait
* Fluent Wait

# **Alerts or Pop-Up**

## What Is Alert?

Class.

## How can we handle web based Alerts or pop-up?

To handle alerts popups we need to do switch to the alert window and call Selenium WebDriver Alert API methods.

There is couple of method are available

**Dismiss():** It will allow you to cancel button action.

**Accept():** It will allow you to Ok or Accept button action.

**GetText():** It will allow you to get text from pop up window

**SendKeys:** This Method simulates Keystrokes in the alert window

## How can we handle windows based pop up?

Selenium doesn’t support windows based applications. It is an automation testing tool which supports only web application testing. We could handle windows based popups in Selenium using some third party tools such as AutoIT, SIKULI, Robot class etc.

AutoIt will work only windows base application.

**Example:**

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

ControlSetText("Open","","Edit1",$CmdLine[1])

Sleep(2000)

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

## How to upload a file in Selenium WebDriver?

There are two cases which are majorly used to upload a file in Selenium WebDriver such as using SendKeys Method and using AutoIT Script.

Browser Button – type =“file”

SendKeys (c:\\test\\naveen.jpg);

# **Command or Method Related**

## How do I launch a batch file in a Selenium WebDriver project?

If throw automation then following code

Runtime.getRuntime.**exec**("path of the batch file");

If it’s not throw automation then we can use command

## How to disable the Browser level Notifications in Selenium?

ChromeOptions options = new ChromeOptions();  
options.addArguments(“–disable-notifications”);  
WebDriver driver =new ChromeDriver(options);

## How to Login Into Any Site If It Is Showing Any Authentication Pop-Up For Username And Password?

To do this we pass username and password with the URL

http://username:password@url

e.g. [http://admin:admin123@xyz.com](http://admin:admin123@xyz.com/)

## What are the different exceptions you have faced in Selenium WebDriver?

* + WebDriverException
  + TimeoutException
  + NoAlertPresentException
  + NoSuchWindowException
  + NoSuchElementException
  + StaleElementReferenceException
  + IllegalStateException

## ElementNotVisibleException:

This is occurs if the element is not visible but still there inside the DOM

In spite of the element being present in the DOM, it is not visible (can not

be interactive). For example, elements defined in HTML with *type =”hidden”*

## ElementNotSelectableException:

This is occurs when an element is disabled (cannot be clicked/selected) in spite of being present in the DOM

## NoSuchElementException:

This is occurs when an element is not present on the Webpage Webdriver is not able to determine the elements during runtime, i.e., the *FindBy* method cannot find a particular component

## NoSuchFrameException:

Webdriver attempts to switch to an invalid frame, which is unavailable

## NoAlertPresentException:

Webdriver is trying to switch to an invalid alert, which is unavailable

## NoSuchWindowException:

Webdriver is trying to switch to an invalid window, which is unavailable

## StaleElementReferenceException:

The referenced element is no longer present on the DOM page (a reference to a component is now Stale). For example, the item belongs to a different frame than

The current one or the user has navigated away to another page

## SessionNotFoundException:

Webdriver is acting immediately after ‘quitting’ the browser

## TimeoutException:

The command did not complete in the specified time. For example, the element didn’t display at the specified time. This is especially encountered when working with waits

## WebDriverException:

Webdriver is acting immediately after ‘closing’ the browser

## How to launch a browser using Selenium WebDriver?

To launch Firefox Driver:

WebDriver driver = new FirefoxDriver();

To launch Chrome Driver:

WebDriver driver = new ChromeDriver();

To launch Internet Explorer Driver:

WebDriver driver = new InternetExplorerDriver();

## How to input text in the text box using Selenium WebDriver?

By using sendKeys() method

WebDriver driver = new FirefoxDriver();

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

driver.findElement(By.xpath("xpath")).sendKeys("test");

## How to clear or Reset the text in the text box using Selenium WebDriver?

By using clear() method

WebDriver driver = new FirefoxDriver();

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

driver.findElement(By.xpath("xpath\_of\_element1")).sendKeys("Software Testing Material Website");

driver.findElement(By.xpath("xpath\_of\_element1")).clear();

## How to get a text of a web element?

By using getText() method

## How to get an attribute value using Selenium WebDriver?

By using getAttribute(value);

## What is getAttribute and getText?

getAttribute:

Its Return any attribute value form Dom

GetText:

It return only text element form element

## How to click on a hyperlink using Selenium WebDriver?

We use click() method in Selenium to click on the hyperlink

driver.findElement(By.linkText(“Software Testing Material Website”)).click();

## How to submit a form using Selenium WebDriver?

We use “submit” method on element to submit a form

driver.findElement(By.id("form\_1")).submit();

Alternatively, you can use click method on the element which does form submission

## How to press ENTER key on text box In Selenium WebDriver?

To press ENTER key using Selenium WebDriver,

We need to use Selenium Enum Keys with its constant ENTER.

driver.findElement(By.xpath("xpath")).sendKeys(Keys.ENTER);

## Is Selenium Server needed to run Selenium Web Driver Scripts?

When we are distributing our Selenium Web Driver scripts to execute using Selenium Grid, we need to use Selenium Server.

## **What happens if I run this command?**

## Is It Mandatory to Prefix the URL with HTTP or HTTPS while Calling the Web Driver Get () Method?

Yes We Need It if we don’t Provide Script will throw an error.

Driver. Get (“www.softwaretestingmaterial.com”);

An exception is thrown. We need to pass HTTP protocol within driver. Get () method.

Driver. Get ("http://www.softwaretestingmaterial.com");

.if we don’t pass www not a problem.

## What is the alternative to driver.Get () method to open an URL using Selenium Web Driver?

Alternative method to driver.get(“url”) method is driver.navigate.to(“url”)

## Can I navigate back and forth in a browser in Selenium WebDriver?

## What are the different types of navigation commands?

We use Navigate interface to do navigate back and forth in a browser. It has

methods to move back, forward as well as to refresh a page.

**navigate().forward();** – to navigate to the next web page with reference to the browser’s history

**navigate().back();** – takes back to the previous webpage with reference to the browser’s history

**navigate().refresh();** – to refresh the current web page thereby reloading all the web elements

**navigate().to(“url”);** – to launch a new web browser window and navigate to the specified URL

## How to fetch the current page URL in Selenium?

To fetch the current page URL, we use getCurrentURL()

driver.getCurrentUrl();

## How can we maximize browser window in Selenium?

To maximize browser window in selenium we use maximize () method. This method maximizes the current window if it is not already maximized

Driver. Manage ().window ().maximize ();

## How to delete cookies in Selenium?

To delete cookies we use deleteAllCookies() method

driver.manage().deleteAllCookies();

## What are the ways to refresh a browser using Selenium WebDriver?

There are multiple ways to refresh a page in selenium

* Using driver.navigate().refresh()
* Using driver.get(“URL”) on the current URL or using driver.getCurrentUrl().
* Using driver.navigate().to(“URL”) on the current URL or driver.navigate().to(driver.getCurrentUrl());.
* Using sendKeys(Keys.F5) on any textbox on the webpage.

## How to select a value in a dropdown?

By using Select class

WebElement mySelectElement = driver.findElement(By.name("dropdown"));

Select dropdown = new Select (mySelectElement);

dropdown.selectByVisibleText(Text);

dropdown.selectByIndex(Index);

dropdown.selectByValue(Value);

## How to capture Screenshot in Selenium WebDriver?

By using TakesScreenshot Interface

File scrFile = (TakesScreenshot)*driver*).getScreenshotAs(OutputType.***FILE***);

## How to handle hidden elements in Selenium WebDriver?

We can handle hidden elements by using JavaScript executor

(JavascriptExecutor)driver)).executeScript("document.getElementsByClassName(ElementLocator).click();");

## How can you find Broken Links in a page using Selenium WebDriver?

## How to find more than one web element in the list?

List <WebElement> eleList = driver.findElements(By.xpath("xpath"));

int listSize = eleList.size();

for (int i=0; i<listSize; i++)

{

    links.get(i).click();

   driver.navigate().back();

}

## How do you read test data from excel?

Test data can efficiently be read from excel using JXL or POI API. POI API has many advantages than JXL.

FileInputStream objFile = **new** FileInputStream(excellFilePath);

XSSFWorkbook WorkBook = **new** XSSFWorkbook(objFile);

## Is it possible to automate the captcha using Selenium?

No, it’s not possible to automate captcha and bar code reader.

## List some scenarios which we cannot automate using Selenium Web Driver?

* Bitmap comparison Is not possible using Selenium Web Driver
* Automating Captcha is not possible using Selenium Web Driver
* We cannot read bar code using Selenium WebDriver
* windows OS based pop ups
* third party calendars/element
* Image
* Word/PDF

## How can you use the Recovery Scenario in Selenium Web Driver?

By using “Try Catch Block” within Selenium Web Driver Java tests.

Try {

     driver.get("www.xyz.com");

} catch(Exception e){

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

}

## How to download a file in Selenium WebDriver?

By using AutoIT script, we could download a file in Selenium WebDriver.

## How to run Selenium WebDriver Test from the command line?

Class A{

}

cd c

c: javac A.java

c: java A.java

java org.testng.TestNG C:\Users \Desktop\ \workspace\testing\testng.xml

## How to Resize Browser Window Using Selenium Web Driver?

To resize the browser window to particular dimensions, we use ‘Dimension’ class to resize the browser window.

         //Create object of Dimensions class

         Dimension d = new Dimension (480,620);

         //Resize the current window to the given dimension

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

## How to Perform Right Click Action (Context Click) In Selenium Web Driver?

We use Actions class in Selenium Web Driver to do Right-Click (Context Click) action.

          action.contextClick (driver.findElement(By.xpath()).build().perform();

## How to Perform Double Click Action in Selenium Web Driver?

We use Actions class to do Double click action in selenium.

## How to Perform Drag and Drop Action in Selenium Web Driver?

We use Actions class to do drag and Drop Action

## How To Highlight Element Using Selenium Web Driver?

By using JavaScript Executor interface, we could highlight the specified element

## How to perform keyboard operation in selenium?

By using Action class

## How do you select a sub menu from a Main Menu

We can select a sub menu by using Actions Class with move to element method

## What does the annotation @ Find by do in Selenium

@Find by annotation is used to identify the web elements through its attributes

## What is the role of set Speed () and Sleep () methods in Selenium?

The role of set Speed () and Sleep () in Selenium is to delay the speed of execution.

## Can we handle colors in Web Driver?

Yes, we can handle colors in Web Driver using the getCssValue(arg0) function. It will help in getting the color by sending the 'color' string as an argument.

## What are the three arguments that can be taken into consideration for Switching?

The three arguments that can be taken into consideration for Switching includes:

* **A number:** This will select the number by its zero-based index.
* **A name or ID:** This will select a frame by its name or ID.
* **Previously found WebElement:** This will help in using the previously located WebElement to select a frame.

# **CSS Selector**

What is the syntax of finding elements by class using CSS Selector?

By .class Name we can select all the elements belonging to a

Particular class e.g. ‘.red’ will select all elements having class ‘red’.

## What is the syntax of finding elements by id using CSS Selector?

By #id Value we can select all the elements belonging to a particular class e.g. ‘#userId’ will select the element having an id – userId.

## **How can we select elements by their attribute value using the CSS Selector?**

Using [attribute=value] we can select all the elements belonging to a particular class e.g. ‘[type=small]’ will select the element having attribute type of value ‘small’.

## **How can we move to nth-child element using CSS selector?**

Using: nth-child (n) we can move to the nth child element

e.g

div:nth-child(2) will locate 2nd div element of its parent.

# **DataBase**

## How to connect a Database in selenium?

As we all know Selenium Web Driver is a tool to automate User Interface. We could only interact with Browser using Selenium Web Driver.

We use JDBC Driver to connect the Database in Selenium (While using Java Programming Language).

Using DriverManager.getConnection() method

String url="jdbc: odbc: MakeConnection";

DriverManager.*getConnection*(url, *UserId*, *UserPassword*);

# **Difference**

## Difference in getWindowHandle() and getWindowHandles()?

getWindowHandle() return a String but getWindowHandles() return a collections of String

driver.getWindowHandle() – It returns a handle of the current page (a unique identifier)

driver.getWindowHandles() – It returns a set of handles of the all the pages available.

Handles meaning Its Id

## Difference between driver.get() and driver.navigate.to(“url”)?

driver.get(): To open an URL and it will wait till the whole page gets loaded

driver.navigate.get(): To navigate to an URL and It will not wait till the whole page gets loaded

## Difference between driver. Close () and driver. Quit () methods?

**Close ():**

driver.close():  To close current Web Driver instance or close current browser.

If after close driver if we call driver exception error with invalid session id

**Quit():**

driver.quit():  To close all the opened Web Driver instances or Browser.

If after quit driver if we call driver exception error with session id is null.

## Difference between find Element () and find Elements () commands?

**Find Element () vs Find Elements ()**

* findElement() returns a single WebElement (found first) based on the locator passed as parameter. Whereas findElements() returns a list of WebElements, all satisfying the locator value passed.
* findElement()

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

findElements()

List <WebElement> elements =Element.findElements(By.id(“value”));

* if no element is found then findElement() throws NoSuchElementException whereas findElements() returns a list of 0 elements or empty list.

## **Difference between XPath and CSS selectors?**

The fundamental difference between XPath and CSS selector is using XPaths we can traverse up in the document i.e. we can move to parent elements and child element on the other hand

CSS selector we can only move forward its not able to go back to parent.

## Difference between Selenium,QTP and TestComplete?

***Selenium vs HP QTP vs TestComplete***

|  |  |  |  |
| --- | --- | --- | --- |
| **Features** | **Selenium** | **HP QTP** | **TestComplete** |
| **License** | Open Source | Required | Required |
| **Cost** | Free | High | High |
| **Customer support** | Yes; Open source community | Yes | Yes |
| **Release Cycles/ Development Sprints** | Smaller release cycles with immediate feedback | Smaller release cycles | Agility only |
| **Coding skills** | Very High | Low | High |
| **Environment support** | Windows, Linux, Mac | Only Windows | Windows only (7, Vista, Server 2008 or later OS) |
| **Language support** | Language support | VB Script | VB Script, JS Script, Delphi Script, C++ & C# |

## Difference between Absolute and Relative Path?

Absolute XPath starts from the root node and ends with desired descendant element’s node. It starts with top HTML node and ends with input node. It starts with a single forward slash (/) as shown below.

/html/body/div[3]/div[1]/form/table/tbody/tr[1]/td/input

Relative XPath starts from any node in between the HTML page to the current element’s node (last node of the element). It starts with a single forward slash (//) as shown below.

//input[@id='email']

# **Excel**

**Why and how will you use an Excel Sheet in your project?**

The reason we use Excel sheets is because it can be used as data source for tests. An excel sheet can also be used to store the data set while performing DataDriven Testing. These are the two main reasons for using Excel sheets.

When you use the excel sheet as **data source**, you can store the following:

* **Application URL for all environments**: You can specify the URL of the environment in which you want to do the testing like: development environment or testing environment or QA environment or staging environment or production/ pre-production environment.
* **User name and password credentials of different environments:**You can store the access credentials of the different applications/ environments in the excel sheet. You can store them in encoded format and whenever you want to use them, you can decode them instead of leaving it plain and unprotected.
* **Test cases to be executed**: You can list down the entire set of test cases in a column and in the next column, you can specify either Yes or No which indicates if you want that particular test case to be executed or ignored.

When you use the excel sheet for **DataDriven Test**, you can store the data for different iterations to be performed in the tests. For example while testing a web page, the different sets of input data that needs to be passed to the test box can be stored in the excel sheet.

# **Framework**

## What is a Framework?

A framework defines a set of rules or best practices which we can follow in a systematic way to achieve the desired results. There are different types of automation frameworks and the most common ones are:

* Data Driven Testing Framework
* Keyword Driven Testing Framework
* Hybrid Testing Framework

## Have you created any Framework?

If you are a beginner: No, I didn’t get a chance to create a framework. I have used the framework which is already available.

If you are an experienced tester: Yes, I have created a framework.  Or I have involved in the creation of the framework.

## Can you explain the Framework which you have used in your Selenium Project?

## Tools:

We are using Selenium WebDriver.Version 3.141.59

## ****Language:****

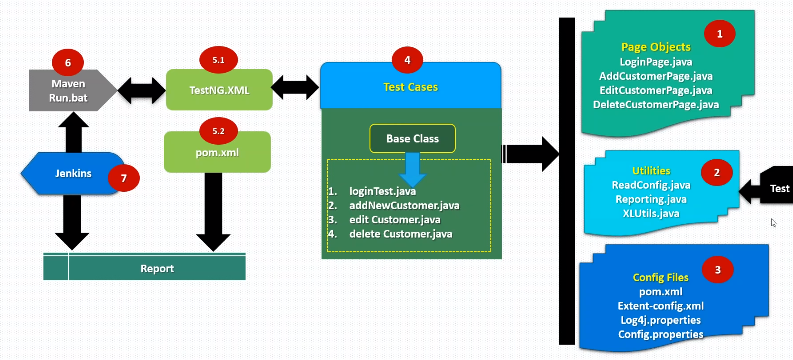
In our Selenium Project we are using Java language. Even though Selenium supports multiple languages, we are using Java language is just because most of the automation developers have knowledge on Selenium with Java.

## ****Type of Framework:****

In our project, we are using [Data-driven Framework](https://www.softwaretestingmaterial.com/data-driven-framework-selenium-webdriver/) by using [Page Object Model design pattern](https://www.softwaretestingmaterial.com/page-object-model/) with Page Factory.

**There are mainly three parts on this section**

* Implementation
* Execution
* Maintenance.



## Implementation:

* Create Maven Project.
* Update pom.xml.
* Create Page Object Class
* Create Basic Test Case.
* Add Log to Test Case.
* Read common values from properties file.
* Run Test Case on desired browser.
* Add extent report
* Create DD Test Case
* Adding new test case.

## Execution:

* Run test case with Maven pom.xml to GITHUB
* Run test cases through Maven CLI(Command Line Interface)
* Run test cases using run.bat
* Run test cases using Jenkins(using bat file)

## Maintenance:

* Creating repository in GITHUB
* Commit the project code in local repository.
* Push the project code to GITHUB remote repository from local GIT repository

## Implementation:

Create Project

First create Maven project. And folder structure as per the maven project, all the tests are kept in the ‘***src/test/java*** ‘and remaining files (such as config.properties, element locators (POM classes), utility files, test data, etc.,) kept under ‘***src/main/java***‘.

## Create Package and Folder Structure:

**Package:**

Under ***src/main/java* folder Create Separate Package**

* **Common**
* **Page Object**
* **Test Data.**

**Folder**

* Configuration Folder
* Create Driver Folder.
* Log Folder.
* Screenshots Folder

## ****Update pom.xml file:****

Before Dependencies keep same then add all required Dependencies under Dependencies tag

* Selenium Java
* TestNg
* For excel
* Poi and poi-ooxml
* Extent Report
* Common Language
* Commons IO
* Log4j
* Log4j-Core
* Html unitdriver

**Update Some Plugin**

* Compiler-Plugin
* Surefire-Plugin
* Source-plugin

## ****Common Packages:****

**Under Common Package have following class:**

## ****Base Class:****

Base Class (TestBase.java) deals with all the common functions used by all the pages. This class is responsible for loading the configurations from properties files, Initializing the Web Driver, Implicit Waits, Extent Reports and also to create the object of FileInputStream which is responsible for pointing towards the file from which the data should be read.

## ****Utility Class (AKA Functions Class):****

Utility class (TestUtil.java) stores and handles the functions (The code which is repetitive in nature such as waits, actions, capturing screenshots, accessing excels, sending email etc.,) which can be commonly used across the entire framework. The reason behind creating utility class is to achieve reusability. This class extends the TestBase class to inherit the properties of TestBase in TestUtil.

**Extent Report Class:**

**Listeners Class**

**Page Object:**

Under Page Object Package As per the Page Object Model, we have maintained a class for every web page. Each web page has a separate class and that class holds the functionality and members of that web page. Separate classes for every individual test.

## Create Test Case:

Create Simple Test Case.

## ****Add Log to Test Case:****

* Add dependency to pom file.
* Add log4j.properties

It’s part of home directory so go to home dir. Under main resource provide log4j xml file.

## ****Run test case on desired browser:****

## ****Extent Reports:****

For the reporting purpose, we are using Extent Reports. It generates beautiful HTML reports. We use the extent reports for maintaining logs and also to include the screenshots of failed test cases in the Extent Report.

* In order to do extends reports first add jar to pom.xml
* Create extent-config.xml
* Add listeners (Reporting.java)
* Add entry in xml file on testng on listeners

## ****Log4J:****

Now we can add log4j setup baseclass and logger

## ****Test Data:****

Apache poi jars

XLS file (Test data)

Under Test Data package we have **config.properties** stores the information that remains static throughout the framework such as browser specific information, application URL, screenshots path etc.

Configuration file does not contain any quote or double quote

To get home directory. /

Under Test Data package we have controller.xlsx which contains the entire historical test data by using ‘controller.xlsx’, we pass test data and handle data driven testing. We use [Apache POI](https://www.softwaretestingmaterial.com/handling-excel-files-using-apache-poi/) to handle excel sheets.

Driver Folder:

Under driver folder we have all driver example chrome driver, ff, ie.

## ****Screenshots Folder:****

Under Screenshots folder Screenshots will be captured and stored in a separate file name base on test case and also the screenshots of failed test cases will be added in the extent reports.

## ****TestNG:****

Using TestNG for Assertions, Grouping and Parallel execution.

Here you could find [TestNG Complete Tutorial](https://www.softwaretestingmaterial.com/testng-tutorial/) and also you could

find [TestNG Interview Questions](https://www.softwaretestingmaterial.com/testng-interview-questions/)

## ****Maven:****

Using Maven for build, execution and dependency purpose. Integrating the TestNG dependency in POM.xml file and running this POM.xml file using Jenkins.

## ****Version Control Tool:****

We use Git as a repository to store our test scripts.

## ****Jenkins:****

By using Jenkins CI (Continuous Integration) Tool, we execute test cases on daily basis and also for nightly execution based on the schedule.

 Test Result will be sent to the peers using Jenkins.

## How to integrate HPALM/HPQC with Selenium?

Bumblebee's JUnit and TestNG solution allows you to easily integrate Selenium WebDriver tests with [HP ALM](https://saas.hpe.com/en-us/software/application-lifecycle-management), map them to requirements, and automatically upload tests, and test results to HP ALM Testplan, TestLab, and Requirements Module.

# **Frame Handle**

## How to switch between frames in Selenium?

By using the following code, we could switch between frames.

driver.switchTo().frame();

## How to go back to main menu from Inner most frames directly?

driver.switchTo().default Content();

## How to switch from inner frames to parent frame one by one?

driver.switchTo().parentFrame() ;

## Can we switch from one frame to another frame through automation?

No we cannot switch from one frame to another frame directly.

# **JavaScript Executor**

## What is the use of JavaScriptExecutor?

**JavaScriptExecutor** is an interface which provides a mechanism to execute Javascript through the Selenium Web Driver. It provides “**execute script**” and “**executeAsyncScript**” methods, to run JavaScript in the context of the currently selected frame or window. An example of that is:

JavascriptExecutor objjs=((JavascriptExecutor)*driver*);

objjs.executeScript("arguments[0].click()",ExpElement);

## Explain the uses of JavaScript Executor in Selenium.

* To locate an element if normal locators do not work.
* To pass values instead of send Keys ().
* To generate user defined alert window.
* To capture scrape data (url, title, domain etc).
* To navigate to a url.
* To Scroll the page.
* To open a new Tab.

## How to input text in the text box without calling the send Keys ()?

We Can Use Java Script Executor to enter value

// to initialize js object

JavascriptExecutor JS = (JavascriptExecutor)driver;

// To enter username

JS.executeScript("document.getElementById(‘User').value=test.com'");

## How to read a JavaScript variable in Selenium WebDriver?

By using JavascriptExecutor

// To initialize the JS object.

JavascriptExecutor JS = (JavascriptExecutor) webdriver;

// To get the site title.

String title = (String)JS.executeScript("return document.title");

System.out.println("Title of the webpage : " + title);

## How to scroll down to a particular element?

To scroll down to a particular element on a web page, we can use the function **scrollIntoView()**. Example:

|  |  |
| --- | --- |
|  | ((JavascriptExecutor) driver).executeScript("arguments[0].scrollIntoView();", element); |

00)");

## How to Scroll Web Page Down or UP Using Selenium Web Driver?

JavaScript scrollBy() method scrolls the document by the specified number of pixels.

|  |  |
| --- | --- |
| 1 | ((JavascriptExecutor) driver).executeScript("window.scrollBy(0,500)"); |

C

# **Locators**

## What are the Locators available in Selenium?

Different types of locators are:

* ID –  Unique for every web element
* ClassName –  Uses the Class name attribute
* Name –  Same as id although it is not unique
* TagName –  Uses HTML tags to locate web element
* LinkText –  Uses Anchor text to locate web elements
* PartialLinkText –  Uses Anchor partial text to locate web elements
* XPath –  Search Elements in the DOM, Reliable but slow
* CSS Selector –  Uses Class Name Attribute

## Which of the Id, Name, X Path or CSS Selector Should You Use?

If any unique Id or Name available first choice will be that if not then go to CSS Selector as it faster than XPath.

When none of this available then we can go XPath.

# **Mouse Over**

## How to mouse over on a web element using Web Driver?

By using Actions class

WebElement ele = driver.findElement(By.xpath("xpath"));

//Create object 'action' of an Actions class

Actions action = new Actions (driver);

//Mouseover on an element

action.moveToElement(ele).build().perform();

Actions class always pass driver.

Build is collected and perform means is execute.

# **Page Object Model**

## What is Page Object Model?

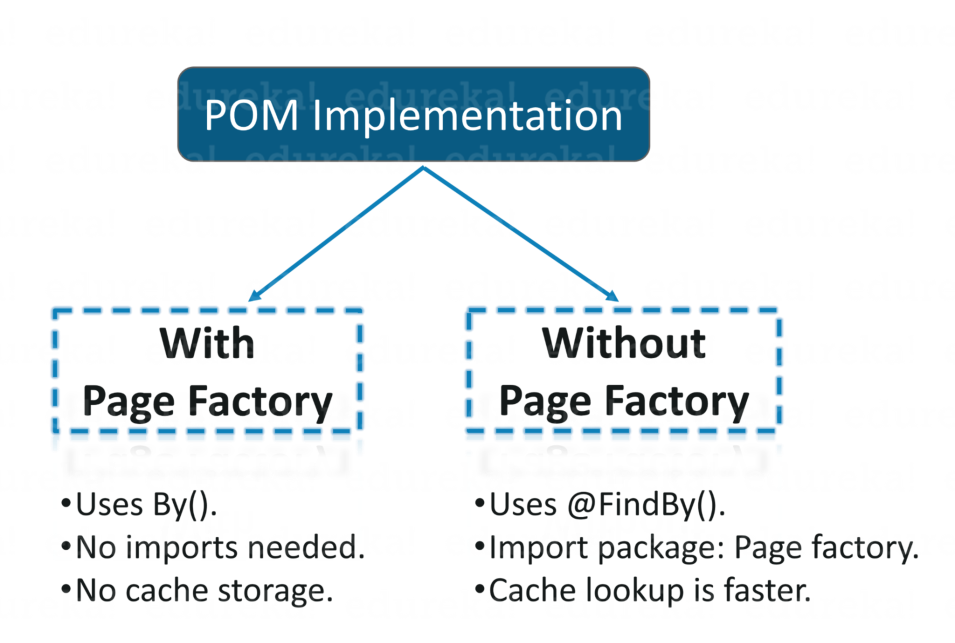
**Page Object is a class in (POM)** corresponding to a web page. POM is a design pattern, popularly used in test automation that creates **Object Repository** for web UI elements. The advantage of the model is that it reduces code duplication and improves test maintenance.

## ****What the Advantages of the Page Object Model:****

* According to the **Page Object Model**, you should keep the tests and element locators separately. This will keep the code clean and easy to understand and maintain.
* The Page Object approach makes automation framework in a testing programmer friendly, more durable and comprehensive.
* Another important advantage is our Page Object Repository is Independent of Automation Tests. If you keep a separate repository for page objects, it helps us to use this repository for different purposes with different frameworks like you will be able to integrate this repository with other tools like **JUnit**/**NUnit**/**PhpUnit**as well as with **[TestNG](https://www.edureka.co/blog/selenium-webdriver-tutorial" \t "_blank)**/**Cucumber**/etc.
* Test cases become short and optimized as you will be able to reuse page object methods in the **POM**
* **POM**is best applicable for the applications which contain multiple pages. Each of which has fields which can be uniquely referenced with respect to the page.
* So these are a few of the advantages that make POM as unique and easy to work with for automation testers.

## ****What is Page Factory?****

Page Factory is a method to set up the web element with in the page object. Page Factory is an inbuilt Page Object Model concept for [Selenium WebDriver](https://www.edureka.co/blog/selenium-tutorial) but it is much optimized. Here, you follow the concept of separation of Page Object Repository and Test Methods.



# **Selenium Testing**

## What is Selenium IDE?

Selenium IDE (Integrated Development Environment) is a Firefox plugin. It is the simplest framework in the Selenium Suite. It allows us to record and playback the scripts. Even though we can create scripts using Selenium IDE, we need to use Selenium RC or Selenium WebDriver to write more advanced and robust test cases.

## What is Selenese?

Selenese is the language which is used to write test scripts in Selenium IDE.

There is couple of command using in Selenese

**Actions:**

It can alter the state of an application

**Accessors:**

This Command monitors the state of command

**Assertions**:

This Command allowing to checkpoint or verification point.

## Which is the only browser that supports Selenium IDE to be used?

Firefox Currently Google Chrome plugins Also Support

## What is Selenium RC?

Selenium RC AKA Selenium 1. Selenium RC was the main Selenium project for a long time before the WebDriver merge brought up Selenium 2. Selenium 1 is still actively supported (in maintenance mode). It relies on JavaScript for automation. It supports Java, Javascript, Ruby, PHP, Python, and Perl and C #. It supports almost every browser out there.

# **Selenium Grid**

## What is Selenium Grid?

Selenium Grid is a tool used together with Selenium RC to run tests on different machines against different browsers in parallel. That is, running multiple tests at the same time against different machines running different browsers and operating systems.

In simple words, it is used to distribute your test execution on multiple platforms and environments concurrently.

## When do you use Selenium Grid?

Selenium Grid can be used to execute same or different test scripts on multiple platforms and browsers concurrently so as to achieve distributed test execution

## What are the advantages of Selenium Grid?

It allows running test cases in parallel thereby saving test execution time.

It allows multi-browser testing

It allows us to execute test cases on multi-platform

## What is a hub in Selenium Grid?

A hub is a server or a central point that controls the test executions on different machines.

## What is a node in Selenium Grid?

Node is the machine which is attached to the hub. There can be multiple nodes in Selenium Grid.

## What is the command to bind a node to Selenium Grid?

Before Run Command Download Standalone jar file then run bellow command

**java -jar servername.jar -role node -hub ipaddress -port 5566/grid/register**

make sure node computer has java

make sure you define webdriver

goto where you have chromedriver

then

java -Dwebdriver.chrome.driver="path chromedriver"-jar servername.jar -role webdriver -hub ipaddress -port 5566

## Which of Java, C-Sharp, and Or Ruby Can We use Selenium Grid?

Everything we can use.

## What is the difference between MaxSessions and Max Instances properties of Selenium Grid?

## MaxInstances:

**It** is the number of browser instances that can run on the remote machine.

browserName=firefox,virson=59,maxinstances=3,platform=WINDOWS

browserName=Internetexplorer,virson=11,maxinstances=3,platform=WINDOWS

It means we can run 3 instances of both FF and IE at the same time so total 6 browsers can run parallel.

# **Screenshot**

## How to capture Screenshot in Selenium Web Driver?

By using TakesScreenshot Interface

String filePath=*createFolderFile*(methodName,".png");

File scrFile = ((TakesScreenshot)*driver*).getScreenshotAs(OutputType.***FILE***);

//The below method will save the screen shot in d drive with test method name

FileUtils.*copyFile*(scrFile, **new** File(filePath));

System.***out***.println("\*\*\*\*\*\*\*\* Placed where screen shot in \*\*\* => "+filePath);

# **SSL Certificate**

**What is SSL Certificate?**

SSL (Secure Sockets Layer) is a standard security protocol for establishing a secure connection between the server and the client which is a browser.

SSL (Secure Socket Layer) Certificate ensures secure transformation of data across the server and client application using strong encryption standard or digital signature. One has to install an SSL certificate or a code signing certificate.

## Benefits of SSL Certificate

There are number of benefits of using SSL certificate like,

* One can increase their users' and customer's trust in order to enhance the business' growth rapidly
* These certificates help to secure online transactions and customers sensitive information like credit-card/debit-card data, etc.
* Signing certificate tends to get a maximum number of downloads and good reviews from users.

SSL-secured websites begin with **https://**and you can see a lock icon or green address bar if the connection is securely established.

For example, if you want to do some transaction via net banking or want to purchase a[Mobile](https://www.guru99.com/mobile-testing.html)phone through e-commerce site such as Flipkart or Amazon.

## What happens between the Web Browser and Server?

* A browser tries to connect with a website secured with SSL. The browser requests the webserver to identify itself
* The server sends the browser a copy of its SSL certificate
* The browser verifies whether the SSL certificate is genuine. If so, it sends a message to the server
* The server sends back a digitally signed acknowledgment to start an SSL encrypted session
* The encrypted data is shared between the server and the browser

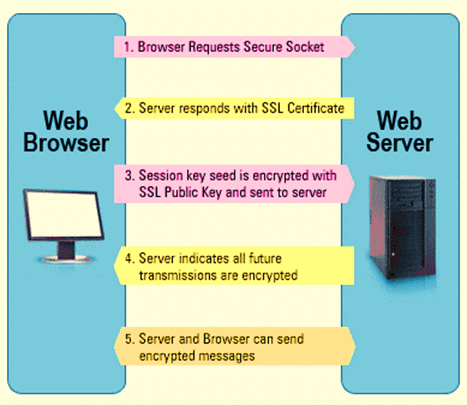
In doing so, you need to transmit sensitive information such as credit card numbers or login credentials and that has to transmit securely so that it cannot be hacked or intercept.

For example

1. Type **https://netbanking.hdfcbank.com/netbanking/** .
2. Hit Enter.
3. You will see a green address bar in the browser as below :-

[SSL Certificate Error Handling in Selenium](https://www.guru99.com/images/3-2016/032816_1141_SSLCertific1.png)

**How Does the SSL Certificate Create a Secure Connection?**

[](https://www.guru99.com/images/3-2016/032816_1141_SSLCertific2.png)

* **Browser** sends HTTPS request to the server.
* Now Server must provide some identification to Browser to prove that it is trusted. This can be done by sending a copy of its SSL certificate to the browser.
* **Each Browser has its own list of Trusted CA's.**Browser checks the certificate root against its list of trusted CAs and that the certificate is unexpired, unrevoked, and that the common name is valid for the website that it is connecting to.
* If the browser trusts the certificate, an encrypted session is created between the server and the browser.
* Server and Browser can send encrypted messages

**Types of SSL Certificates**

Browser and the server use SSL Certificate mechanism to be able to establish a secure connection. This connection involves verification of three types of certificates.

* Root
* Intermediate
* Server Certificate

**Process of getting SSL Certificate**

The process of getting SSL certificate includes below steps:-

* First, you must create CSR (create a Certificate Signing Request) request.
* CSR request creates CSR data file, which is sent to SSL certificate issuer known as CA (Certificate Authority).
* The CA uses the CSR data files to create SSL certificate for your server.
* After receiving the SSL certificate, you have to install it on your server.
* An intermediate certificate is also needed to be installed which ties yours SSL certificate with CA's root certificate.

The below image represent all the three certificate- **Root, Intermediate, and Server Certificate.**

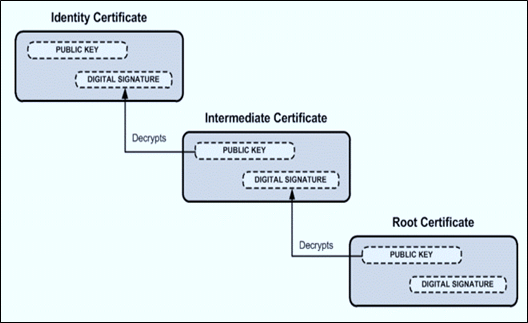
**How SSL certificates are verified**

SSL works through a combination of programs and encryption/decryption routine that exist on the web server computer and web server browser.

SSL certificate basically contains below information.

1. Subject which is the identity of the website owner.
2. Validity information- a public and a private key.

The Private and public key are two uniquely related cryptographic keys (numbers). Whatever is encrypted by a public key may only be decrypted by a private key.

[](https://www.guru99.com/images/3-2016/032816_1141_SSLCertific4.png)

When a secure connection is not established between the server and client due to the certificate, following SSL certificate error will be manifested.

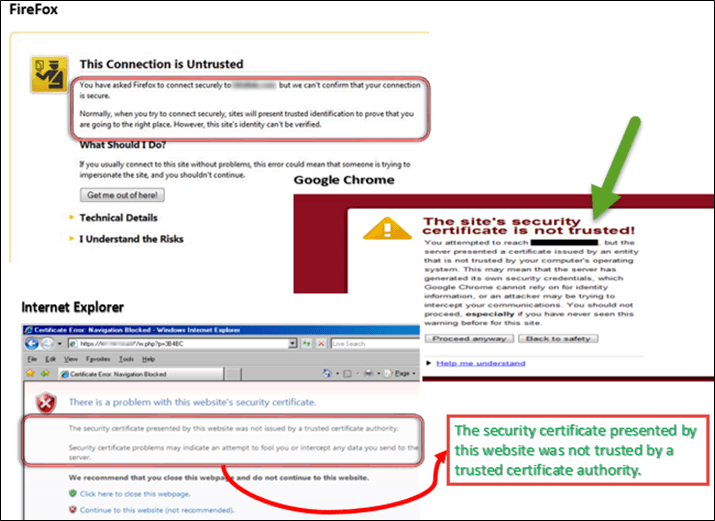
**Types of SSL Certificate Error**

Suppose you type some https request in the browser and get a message such as "This connection is Untrusted" or the "The site's security certificate is not trusted" depending upon the browser you are using. Then such error is subject to SSL certificate error.

Now, if the browser is unable to establish a secured connection with the requested certificate, then the browser will throw "Untrusted Connection" exception as below and ask the user to take appropriate action.

The types of error you likely to see due to certificate in different browsers may be somewhat like this

* **FireFox** - This connection is untrusted
* **Google Chrome** -This site security is not trusted
* **Internet Explorer ( IE)** - This security certificate presented by this website was not trusted by a trusted certificate authority (CA)

[](https://www.guru99.com/images/3-2016/032816_1141_SSLCertific5.png)

## How to handle SSL Certificate Error using Selenium Webdriver

Suppose we have written some test scripts and while executing the script, we caught in the situation as "Untrusted Connection" above then how do we handle the exception purely through automation.

In such case, we have to adjust our script in such a way that it will take care of SSL Exception by itself.

The scripts need to be modified according to the type of browser instance we are using. These when desired capabilities comes in picture.

Desired Capabilities is used to configure the driver instance of Selenium Webdriver. Through Desired Capabilities, one can configure all driver instance like ChromeDriver, FirefoxDriver, and Internet Explorer.

As of now we don't have any specific URL to create the above scenario, but I am

providing steps that we can add in the Selenium Script to handle the above situation "Untrusted Connection."

**SSL Certificate Error Handling in Firefox**

For handling SSL certificate error in Firefox, we need to use desired capabilities of Selenium Webdriver and follow the following steps.

**Step 1**:

First we need to create a new firefox profile say "**myProfile**". You can refer google to learn "How to create" firefox profile. It is simple and easy.

**Step 2**:

Now access myProfile in the script as below and create the FirefoxProfile object.

FirefoxProfile objff=new FireFoxProfile();

**Step 3**:

Now we need to set "**setAcceptUntrustedCertificates**" and "**setAssumeUntrustedCertificateIssuer**" properties in the Fire Fox profile.

objff.setAcceptUntrustedCertificates(true)

objff.setAssumeUntrustedCertificateIssuer(false)

**Step 4**: Now use the FireFox profile in the FireFox driver object.

WebDriver driver = new FirefoxDriver (ffProfile)

**Note**: "setAcceptUntrustedCertificates" and "setAssumeUntrustedCertificateIssuer**"** are capabilities to handle the certificate errors in web browsers.

**SSL Certificate Error Handling in Chrome**

For handling SSL error in Chrome, we need to use desired capabilities of Selenium Webdriver. The below code will help to accept all the SSL certificate in chrome, and the user will not receive any SSL certificate related error using this code.

We need to create instance of DesiredCapabilities class as below:-

DesiredCapabilities objd = DesiredCapabilities.chrome ()

objd.setCapability (CapabilityType.ACCEPT\_SSL\_CERTS, true)

WebDriver driver = new ChromeDriver (handlSSLErr);

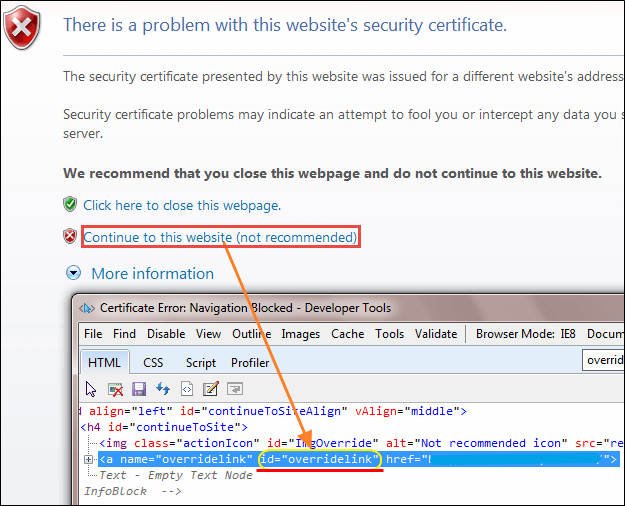
**SSL Certificate Error Handling in IE**

Unlike handling SSL certificates in Chrome browser and Firefox, in IE, you may have to handle it using javascript.

To handle SSL certificate in IE, you can handle this situation in two ways,

1. In this, you will click the link "**Continue to this website (not recommended)".** In the following we will see how to handle SSL error in IE.

Observe SSL certificate error in IE browser you will find "Continue to this website (not recommended)" link.This link has ID "override link".You can view the ID in HTML mode using F12.

[](https://www.guru99.com/images/3-2016/032816_1141_SSLCertific6.png)

Click on the link using driver.navigate() method with[JavaScript](https://www.guru99.com/interactive-javascript-tutorials.html)as below :-

driver.navigate ().to ("javascript:document.getElementById('overridelink').click()");

1. The second method is quite similar to chrome SSL Handling code

DesiredCapabilities capabilities = new DesiredCapabilities();

capabilities.setCapability(CapabilityType.ACCEPT\_SSL\_CERTS, true);

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

WebDriver driver = new InternetExplorerDriver(capabilities);

# **Testing Question**

## What version of selenium do you use and which is the latest version?

Selenium 3.141.59 Java 1.8

## What is Automation Testing?

Automation testing is the process of testing the software using an automation tool to find the defects. In this process, executing the test scripts and generating the results are performed automatically by automation tools. Some most popular tools to do automation testing are HP QTP/UFT, Selenium WebDriver, etc.,

## What are your roles and responsibilities for automation?

Automation framework design and implementation according to project Structure. Creating, Enhancing, Debugging and Running **Test** Cases. Organizing, monitoring defect management process Handling changes and conducting regression tests. Find solutions for issues related to object identity issues and error handling.

## What are the factors should consider before doing automation?

* **It Constant?**Is the sequence remains the same every time
* **How repetitive it is?**If it is not executed frequently, no point in automating
* **Is it cost effective?** How much effort is needed to automate? As a thumb rule, Automation effort should not exceed the effort needed to execute the task for a month manually…
* **Infrastructure**: don’t buy new scenario automation software just for automation of few test cases. Identify lots of scenarios before investing right automation software
* **Trouble shooting**: Make sure the automation errors are logged somewhere in detail for ease of troubleshooting. Other managing it will be a nightmare
* **Runnable**: Make sure you design it to rerun from point of failure.
* **Modularized and parameterized:** Under all the automation requirements before designing. Combine similar things and address it through parameterization; different things with modularization;
* **Granularity**: Automate at the lowest level so that if you want to execute the low level can be executed individually.

## What are the benefits of Automation Testing?

* Saves time and money. Automation testing is faster in execution.
* Reusability of code. Create one time and execute multiple times with less or no maintenance.
* Easy reporting. It generates automatic reports after test execution.
* Easy for compatibility testing. It enables parallel execution in the combination of different OS and browser environments.
* Low-cost maintenance. It is cheaper compared to manual testing in a long run.
* Automated testing is more reliable.
* Automated testing is more powerful and versatile.
* It is mostly used for regression testing. Supports execution of repeated test cases.
* Minimal manual intervention. Test scripts can be run unattended.
* Maximum coverage. It helps to increase the test coverage.

## What are the benefits of Selenium Automation Testing?

* Saves time and money. Automation testing is faster in execution.
* Since it’s an OSS, so we don’t have to bear any licensing cost for using it.
* It’s Works on All Major Browser such as Google Chrome, FF, IE, IOS, and Safari.
* We Can Run Same Script on Multiple browsers.
* Have large user base and helping communities
* Cross browser compatibility
* Platform compatibility
* Multiple programming languages support
* CI Support
* Free Help Support
* Tester Friendly.
* Third Party Integration Support example TestNg, Jenkins, AutoIt
* Reusability of code. Create one time and execute multiple times with fewer or no maintenance.
* Easy reporting. It generates automatic reports after test execution.
* Easy for compatibility testing. It enables parallel execution in the combination of different OS and browser environments.
* Low-cost maintenance. It is cheaper compared to manual testing in a long run.
* Automated testing is more reliable.
* Automated testing is more powerful and versatile.
* It is mostly used for regression testing. Supports execution of repeated test cases.
* Minimal manual intervention. Test scripts can be run unattended.
* Maximum coverage. It helps to increase the test coverage.

## What are all the disadvantages (limitations) of Selenium?

* Supports Web-based applications only.
* Difficult to use takes more time to create Test cases.
* No reliable Technical Support from anybody.
* Difficult to Setup Test Environment when it compares to Vendor Tools like UFT, RFT, SilkTest.
* Limited support for Image Testing.
* New features may not work properly.
* No Test Tool integration for Test Management.
* No Built-in Reporting facility.
* Does Not Support Windows Base Application Directly.
* Can’t be Automated Barcode or Captcha.
* Not 100% perfect for handling dynamic web elements.
* Poses challenges while processing popups or frames.
* Lacks of Good Reporting.

## What is automation test life cycle?

* Determining The Scope Of Test Automation
* Selecting The Right Tool For Automation
* Test Plan + Test Design + Test Strategy
* Setting Up The Test Environment
* Automation Test Script Development + Execution
* Analysis + Generation Of Test Reports

## ****What is the strategy for Automation Test Plan?****

* The strategy for Automation Test Plan
* Preparation of Automation Test Plan
* Recording the scenario
* Error handler incorporation
* Script enhancement by inserting check points and looping constructs
* Debugging the script and fixing the issues
* Rerunning the script
* Reporting the result

## **What type of tests have you automated?**

Our main focus is to automate test cases to do Regression testing, Smoke testing, and Sanity testing. Sometimes based on the project and the test time estimation, we do focus on End to End testing.

## **What type of tests can you automated with Selenium?**

* Functional Testing.
* Integration Testing.
* Regression testing.
* Smoke testing.
* Acceptance Testing.
* Sanity testing
* End to End testing.
* Cross-Browser Test.

## How many test cases you have automated per day?

It depends on Test case scenario complexity and length. I did automate 2-5 test scenarios per day when the complexity is limited. Sometimes just 1 or fewer test scenarios in a day when the complexity is high.

## When will you stop testing?

**Time, Budget and Extent of Testing.**

The most common approach is to stop when either Time / Budget is exhausted or all test scenarios are executed. However, with this approach, we will be compromising on the quality of testing and this will not give enough confidence about the software; how?

## How can you test Mouse?

* Click all the buttons on your **mouse** and check if they light up on the **mouse**

 Illustration.

* Point your **mouse** cursor at the **mouse** illustration and then spin the scroll wheel on your **mouse** up and down.
* Check if the arrows on the illustration also light up.

## What is the use of cookies and cache?

**Cookie** is used to store information to track different characteristics related to user, while **cache** is used to make the loading of web pages faster.

**Cookies** stores information such as user preferences, while **cache** will keep resource files such as audio, video or flash files.

## How to handle cookies in selenium?

By using getCookies();

## What is Selenium?

Selenium is an open source (free) automated testing suite to test web applications. It supports different platforms and browsers. It has gained a lot of popularity in terms of web-based automated testing and giving a great competition to the famous commercial tool HP QTP (Quick Test Professional) AKA HP UFT (Unified Functional Testing).

Selenium is a set of different software tools. Each tool has a different approach in supporting web based automation testing.

It has four components namely,

* Selenium IDE (Integrated Development Environment)
* Selenium RC (Remote Control) – selenium 1
* Selenium WebDriver – selenium 2 & 3
* Selenium Grid

## What are the Programming Languages supported by Selenium WebDiver?

* Java
* C#
* Python
* Ruby
* Perl
* PHP

## What are the Operating Systems supported by Selenium?

* Windows
* Linux
* Apple
* Android
* IOS

## What is exception test in Selenium?

An exception test is an exception that you expect will be thrown inside a test class. If you have written a test case in such way that it should throw an exception, then you can use the **@Test** annotation and specify which exception you will be expecting by mentioning it in the parameters. Take a look at the example below: **@Test(expectedException = NoSuchElementException.class)**

Do note the syntax, where the exception is suffixed with .class

## Can You Use Selenium for Testing Rest API or Web Service?

No.

## Why do you prefer Selenium Automation Tool?

* Free and open source
* Have large user base and helping communities
* Platform compatibility
* Multiple programming languages support

## Why do we use packages in Java?

**Packages** are used in **Java** in order to prevent naming conflicts, to control access, to make searching/locating and usage of classes, interfaces, enumerations and annotations easier, etc.

## How often do you run your automation test?

It depends on Dev. and client Requirements and also Time Slot. Approximately your automation suite should be used or run at least 15 to 20 times for separate builds (General assumption. depends on specific application complexity) to have good ROI.

## What is the time taken to run all your scripts?

There is no guaranty for time its depends on how big script some script take sort time some are long

# **Validate or Verify**

## What is the difference between Assert and Verify in Selenium?

**Assert:**

In simple words, if the assert condition is true then the program control will

Execute the next test step but if the condition is false, the execution will stop and further test step will not be executed.

**Verify:**

In simple words, there won’t be any halt in the test execution even though the verify condition is true or false.

## What are Soft Assert and Hard Assert in Selenium?

**Soft Assert:**

Soft Assert collects errors during *@Test* Soft Assert does not throw an exception when an assert fails and would continue with the next step after the assert statement

**Hard Assert:**

Hard Assert throws an AssertException immediately when an assert statement fails and test suite continues with next *@Test*

## What are the verification points available in Selenium?

In Selenium IDE, we use Selenese Verify and Assert Commands as

Verification points.

In Selenium WebDriver, there is no built-in features for verification points. It totally depends on our coding style. Some of the Verification points are

To check for page title

To check for certain text

To check for certain element (text box, button, drop down, etc.)

## How do you check if an Object is Present on Multiple Pages?

We Can Use IsElementPresent() Method command to verify the object on all pages

## How to find whether an element is displayed on the web page?

## How do you check for the Presence of a web element after Successful Page Load?

WebDriver facilitates the user with the following methods to check the visibility of the web elements. These web elements can be buttons, drop boxes, checkboxes, radio buttons, labels etc.

* **isDisplayed()**
* boolean elePresent = driver.findElement(By.xpath("xpath")).isDisplayed();
* **isSelected()**
* boolean eleSelected= driver.findElement(By.xpath("xpath")).isSelected();
* **isEnabled()**
* boolean eleEnabled= driver.findElement(By.xpath("xpath")).isEnabled();

# **WebDriver**

## What is WebDriver?

WebDriver is a web automation framework that allows you to execute your tests against different browsers, not just Firefox, Chrome (unlike Selenium IDE).

WebDriver also enables you to **use a programming language** in creating your test scripts (not possible in Selenium IDE).

You can now use **conditional operations** like if-then-else or switch-case. You can also perform looping like do-while.

## What is Selenium WebDriver?

Selenium WebDriver AKA Selenium 2 is a browser automation framework that accepts commands and sends them to a browser. It is implemented through a browser-specific driver. It controls the browser by directly communicating with it. Selenium WebDriver supports Java, C#, PHP, Python, Perl, Ruby.

## Why we can't create instance of WebDriver directly?

## What are the different types of WebDriver APIs available or support in Selenium?

* Firefox Driver()
* Gecko Driver(a.k.a. Marinetto)
* InternetExplorerDriver()
* ChromeDriver()
* HTMLUNITDriver()
* Opera Driver
* SafariDriver()
* AndroidDriver()
* IPhone Driver 🡺IOSDriver().
* EventFiringWebDriver

## Which WebDriver implementation claims to be the fastest?

The fastest implementation of WebDriver is the HTMLUnitDriver. It is because the HTMLUnitDriver does not execute tests in the browser.

## What are the Open-source Frameworks supported by Selenium WebDriver?

* JUnit
* TestNG
* Cucumber
* JBheave

## What Are step to create a simple WebDriver Script?

* Lunch The Browser (FF,Chrome) by creating the WebDriver Reference pointing to ChromeDriver/FireFox Driver class.
* Open The Website [www.google.com](http://www.google.com) using the .get() method
* Wait for page to Load
* Maximize the window
* Display a message on the console that webpage load successfully
* Close The Browser

**Code:**

**import** java.util.concurrent.TimeUnit;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

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

**public** **class** CssSelect {

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

// **TODO** Auto-generated method stub

System.*setProperty*("webdriver.chrome.driver","C:\\Users\\nyc\\eclipse-workspace\\Criglist\\Driver\\chromedriver.exe");

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

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

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

driver.manage().timeouts().pageLoadTimeout(5,TimeUnit.***SECONDS***);

driver.manage().timeouts().implicitlyWait(4,TimeUnit.***SECONDS***);

driver.findElement(By.*xpath*("//a[contains(@href,'mypage')]/div")).click();

driver.findElement(By.*cssSelector*("input[type='search']")).sendKeys("iphon

e");

driver.quit();

}

}

## What does FirefoxDriver Mean, A Class or an Interface?

FirefoxDriver is a Java class, and it implements the WebDriver interface.

## **What is supper class of chromedriver?**

Remotewebdriver.

## **What is the super interface of WebDriver?**

The SearchContext acts as the super for the webdriver.

It is the external interface which has only two methods:findelement() and findElements()

## What are the search context interfaces?

FindElement.

FindElements.

## In Selenium, what is WebDriver - a class or interface?

WebDriver is an Interface in Selenium

## Explain the line of code Webdriver driver = new FirefoxDriver(); ?

* WebDriver is an Interface in Selenium
* FirefoxDriver is a Class in Selenium
* The “driver” is a variable of type WebDriver Interface
* In this statement, we have created an object for FirefoxDriver class using new keyword and assigned the created object to the reference variable of WebDriver interface.

## Why do create a reference variable ‘driver’ of type WebDriver not the Actual Browser type?

It is because we could use the same WebDriver variable to hold the object of any browser. Such as Chrome,IE,FF etc.

WebDriver driver = new FirefoxDriver();

## Why we creating WebDriver driver = new FirefoxDriver() instead of FirefoxDriver driver = new FirefoxDriver();

If we create a reference variable driver of type WebDriver then we could use the same driver variable to work with any browser of our choice such as IEDriver, SafariDriver etc.,

But if we create a

FirefoxDriver driver = new FirefoxDriver();

Then we can use only that particular browser.

## What is Object Repository in Selenium WebDriver?

Object Repository is used to store element locator values in a centralized location instead of hard coding them within the scripts. We do create a property file (.properties) to store all the element locators and these property files act as an object repository in Selenium WebDriver.

# **Wait Time**

## What are the types of waits available in Selenium WebDriver?

In Selenium we could see three types of waits such as:

* + Implicit Waits
  + Explicit Waits
  + Fluent Waits
  + PageLoadTimeOut
  + Thread. Sleep.

## Which class in responsible for wait time?

Webdriverwait.

## What is Implicit Wait in Selenium WebDriver?

Implicit waits tell to the WebDriver to wait for a certain amount of time before it throws an exception. Once we set the time, WebDriver will wait for the element based on the time we set before it throws an exception. The default setting is 0 (zero). We need to set some wait time to make WebDriver to wait for the required time.

**Example Code:**

driver.manage().timeouts().implicitlyWait(4,TimeUnit.***SECONDS***);

## What is Explicit Wait in Selenium WebDriver?

Explicit Wait is applied on a certain element with defined expected condition and time. This wait is only applied to the specified element. This wait can also throw an exception when an element is not found.

**Example Code:**

WebDriverWait ExWait=**new** WebDriverWait(*driver*,timeOut);

Boolean waitStatus=ExWait.ignoring(StaleElementReferenceException.**class**).until( ExpectedConditions.*textToBePresentInElementValue*(ExpwebElement,ExPText));

Some Common Use Method is

* elementToBeClickable
* elementToBeSelected
* presenceOfElementLocated

## What is Fluent Wait in Selenium WebDriver?

FluentWait can define the maximum amount of time to wait for a specific

Condition and frequency with which to check the condition before throwing

an “ElementNotVisibleException” exception.

**Example Code:**

Wait wait = **new** FluentWait<WebDriver>(driver)

.~~withTimeout~~(50, TimeUnit.***SECONDS***)

.~~pollingEvery~~(3, TimeUnit.***SECONDS***)

.ignoring(NoSuchElementException.**class**);

## How to pause a test execution for 5 seconds at a specific point?

By using java.lang.Thread.sleep(long milliseconds)

method we could pause the execution for a specific time. To pause 5 seconds, we need to pass parameter as 5000 (5 seconds)

Thread.sleep(5000)

## How do you achieve synchronization in WebDriver?

Explicit wait instructs the execution to wait for some time until some condition is achieved. Some of those conditions to be attained are:

* elementToBeClickable
* elementToBeSelected
* presenceOfElementLocated

# **WindowHandle**

## Why Set<Iterator> you have used for handling window handles?

Iterator using because it can hold all set of window that open and easily we can go for next window

## What is the return type of driver.getwindowhandles?

Set String

## How to handle Multiple Browser window Popup In Selenium?

Using GetWindowhandles() Method we can handle multiple pop up.

String PareantWindow=*driver*.getWindowHandle();

System.***out***.println("No. of tabs: " + PareantWindow);

Set<String> objWhandles = *driver*.getWindowHandles();

**int** TotalWindow=objWhandles.size();

**for**(String child:objWhandles)

{

**if**(!PareantWindow.equalsIgnoreCase(child))

{

*driver*.switchTo().window(child).close();

System.***out***.println("No. of tabs: " +TotalWindow);

}

}

*driver*.switchTo().window(PareantWindow);

}

## How can you get the value from Set?

Using iterator or copy the set to list of array then get the value.

## How to switch to a new window (new tab) which opens up after you click on a link?

If you click on a link in a web page, then for changing the WebDriver’s focus/ reference to the new window we need to use the **switchTo()** command. Look at the below example to switch to a new window:  
**driver.switchTo().window();**

Here, ‘windowName’ is the name of the window you want to switch your reference to.

In case you do not know the name of the window, then you can use the **driver.getWindowHandle()** command to get the name of all the windows that were initiated by the WebDriver. Note that it will not return the window names of browser windows which are not initiated by your WebDriver.

Once you have the name of the window, then you can use an enhanced for loop to switch to that window. Look at the piece of code below.

# **XPath**

## How do handle dynamic elements during run time?

There is multiple ways we can handle this scenario

* Absolute Path method
* Use Relative XPath using contains or starts with text
* Identify by index
* Use Multiple attributes to locate an element

## How to handle dynamic xpath in selenium?

By using methods such as, text(),Contains(), parent,child,following-siblings,ansestors etc

## What is an XPath? How Does It work?

XPath is used to locate the elements. Using XPath,

It works by navigating through the DOM elements and attributes to locate the target object for example textbox, button, and checkbox, Image etc., in a web page.

## XPath Syntax:

//TagName[@attributeName=’attributeValue’]

## What is the difference between “/” and “//”

**Single Slash “/”:**

Single slash is used to create XPath with absolute path i.e. the

XPath would be created to start selection from the document node/start node to target element. Normally its start from parent node to target Node.

**Double Slash “//”:**

Double slash is used to create XPath with relative path i.e. the XPath would be created to start selection from anywhere within the document.

## How do you locate an element By Partially Comparing its Attribute in XPath?

XPath Support Contains function so its allows partially matching of attribute Value.

It helps when the attribute use dynamic values while having some fixed part

Example:

//TagName[contains(@attributeName,”partial Value”)]

## How do you locate element Base on The text Xpath?

XPath Support Text function so its allows partially or full matching of text value.

It helps when the attribute use dynamic text values while having some fixed part

Example:

//TagName[text(),”partial Value”)]

//TagName[text()=”Text Value”)]

## How Do You Access Parent Node With XPath?

We Can Use the double dot(“..”) to point to the parent of any node using XPath

Example

//\*[@id='fakebox-input']/../..

## How do you get to the nth Sub-element Using XPath?

We can modify the XPath expression to get the nth element in the

Following way

//\*[@id='ntp-contents']/div[2]

## Write the syntax for to check the xpath in console?

$x[“Value”]