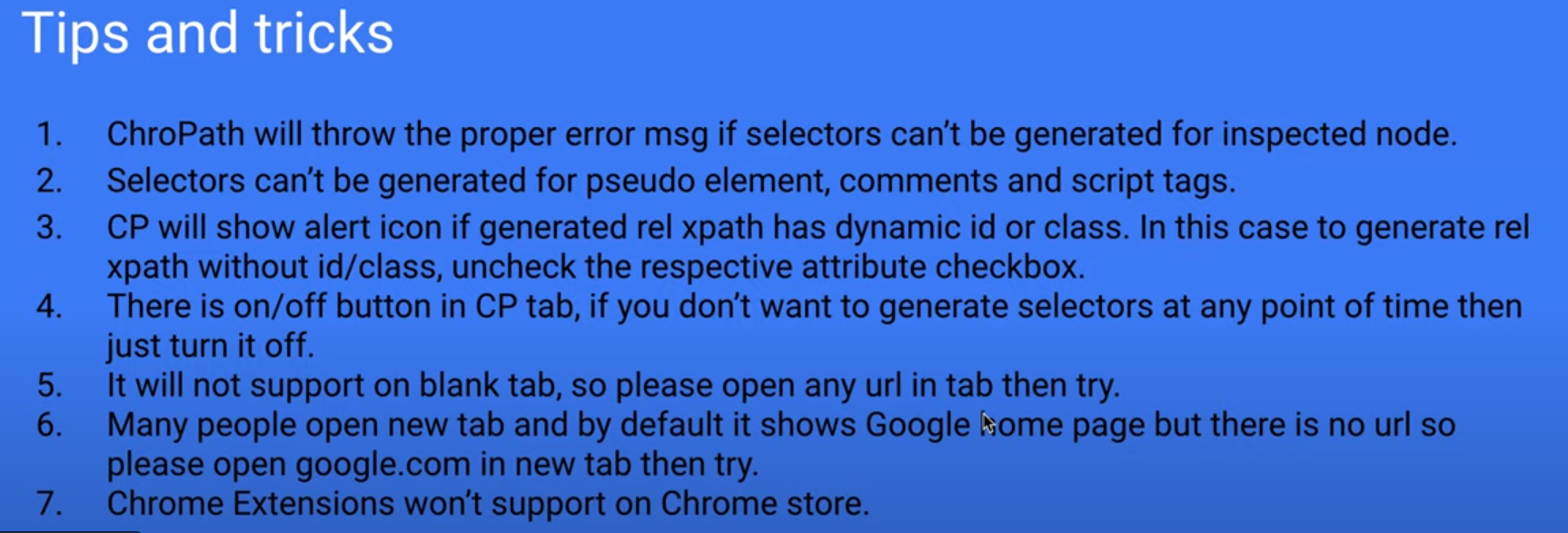
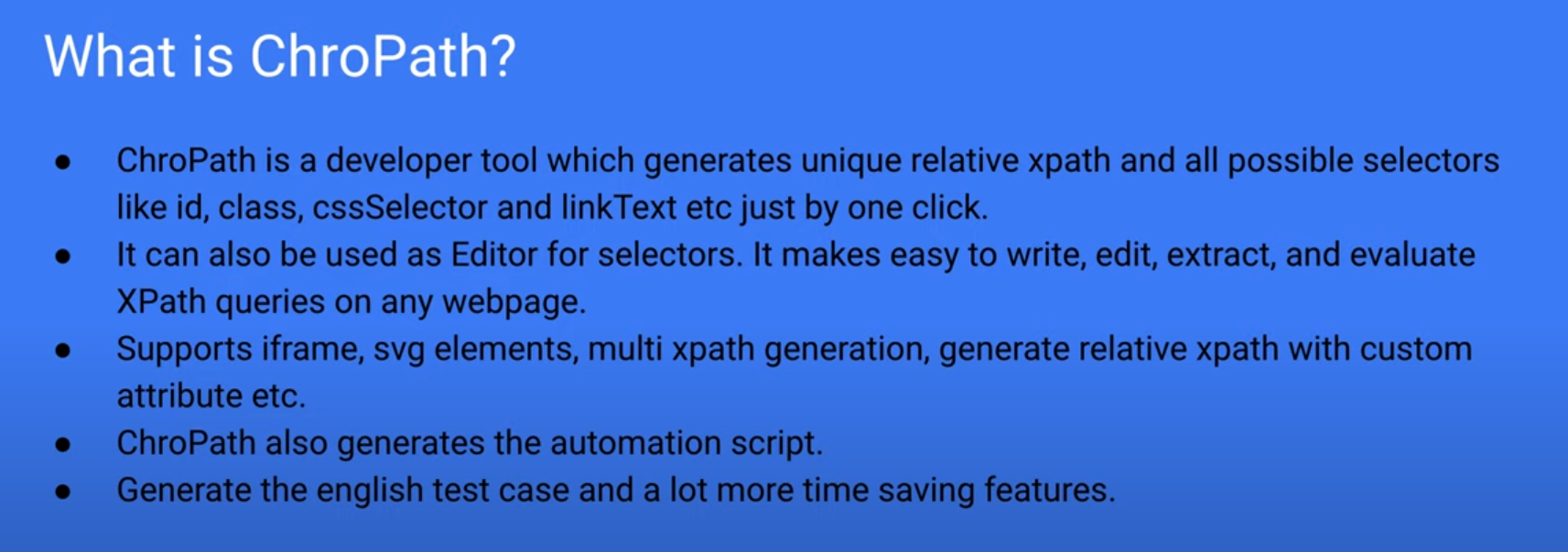
**Selenium Documents**



**Why you chose Selenium WebDriver for your automation?**

In my project we were working on web based application and for the same reason selenium is the best option and have multiple advantages like

1. Selenium is pure open source, freeware and portable tool.   
2. Selenium supports variety of languages that include Java, Perl, Python, C#, Ruby, Groovy, Java Script, and VB Script. etc.   
3. Selenium supports many operating systems like Windows, Macintosh, Linux, Unix etc.   
4. Selenium supports many browsers like Internet explorer, Chrome, Firefox, Opera, Safari etc.   
5. Selenium can be integrated with ANT or Maven kind of framework for source code compilation.   
6. Selenium can be integrated with TestNG testing framework for testing our applications and generating reports.   
7. Selenium can be integrated with Jenkins or Hudson for continuous integration.   
8. Selenium can be integrated with other open source tools for supporting other features.   
9. Selenium can be used for Android, IPhone, Blackberry etc. based application testing.   
10. Selenium supports very less CPU and RAM consumption for script execution.   
11. Selenium comes with different component to provide support to its parent which is Selenium IDE, Selenium Grid and Selenium Remote Control (RC).

**Selenium alone can perform full web based automation?**

Selenium can perform web based automation but not completely (silver light application-you tube, captha, graph) and for strong and clean framework we need a TestNG , POM ,sometimes support from JavaScript as well.

**Pain of an automation tester-**

When I used to create automation script

* My script used to work fine sometimes and sometimes it used to fail.
* When I run the script in my local machine it works fine but when I run in remote or another browser it fails.
* Scripts work fine in one browser and it fails in other browsers.
* Scripts work fine for current release, but it fails when new release come for my applications.
* Earlier I used to create scripts without any framework so even if small changes in the application then I used to make the changes in each script.

**Best practice to overcome this pain.**

* Start writing your own xpath using xpath methods.
* Start using a design/Framework-You can directly start writing your automation script without any framework that fine but maintenance task will be a big headache once test case size will grow even if a small change in the application then you have to modify all script.
* You can start using any framework, but it should have below characteristic. -

1. Reusable 2- Easy to use 3- Easy to maintain 4- Robust in nature

* Execute your script multiple times -It may take some time but your script will be highly stable and you will get good ROI from your script.
* Use Smart wait in your script-90 % automation script failure reason is Sync issue and locator changes only.

**IE browser** is very much concerned about security you will find a couple of Issues with IE browser

If you are using Selenium 3 then to work with Firefox browser, you need to use separate driver which will interact with Firefox browser.

**How to execute Selenium Webdriver in Chrome Browser?**

Your test case will fail and you will get IllegalStateException which says we need to specify the chrome driver path where it resides. If you notice Selenium also gives a very meaningful message that we need to add some chrome variable also while running the script.

**Variable name is – webdriver.chrome.driver**

In Java to set variable we use setProperty method of System class so let us add the same in our program.

**System.setProperty("webdriver.chrome.driver", "path of the exe file\\chromedriver.exe");**

# **How to remove Disable Developer Mode Extension in Selenium?**

ChromeOptions options = new ChromeOptions();

options.addArguments("--disable-extensions");

WebDriver driver = new ChromeDriver(options);

**Some key point while working with IE Browser in Selenium.**

* IE browser is slow as compared to other browsers.
* Your browser zooming level should be set to 100 % otherwise, you will get an exception.
* You have to check your security setting also in IE. While running IE browser in Selenium your all zones should be either enabled or disabled. If not, then again you will get an exception and your test cases will fail.

## Challenges with IE browser in Selenium Webdriver

You will get the different type of Exception while working with IE Browser

**Issues 1-**

### openqa.selenium.NoSuchWindowException

This is a common issue with Selenium and you can avoid this by doing some IE setting, which we are going to discuss now.

**Issue 2-**

### sendKeys works very slow it takes 1-2 second to type each character.

This is a known issue with Selenium and it only happens once you work with IE 64 bit driver.

Solution- You can download IE Driver 32 bit and start using it, even you are working with 64 bit OS this 32 bit IE driver works every time.

**Issue 3-**

### **Unexpected error launching Internet Explorer. Protected Mode must be set to the same value**

When I started working with IE this was the first exception, which I used to get, and I was sure that this related to some browser setting.

**Issue 4-**

**Unexpected error launching Internet Explorer. Browser zoom level was set to 0%**

By the name itself, you can see that we have to set the zoom level to 100 % to make it work.

**Issue 5:**

**Handle Untrusted SSL certificate error in IE browser in different ways  
Solution: IE is the product of Microsoft and IE is much worried about security so when you start working with some https application you will get a untrusted certificate.**

**Selenium has so many ways to handle this, but we will see 2 ways which work all the time for me.**

**First:**Open the application for which SSL certificate is coming so use below code after passing the URL.  
driver.get(“ur app URL”);  
driver.navigate().to(“javascript:document.getElementById(‘overridelink’).click()”);  
// you can use your code now

**Second:**

You can handle this certificate using Desired Capabilities as well.

**Absolute xpath** = complete address of particular element from parent node to child node. Not Recommended -Performance issue, not reliable, can be change any time.

Example - div/table/tbody/tr/td/a

**Relative xpath** = Can be start from anywhere in the DOM.

//div[@class='dropdown']//button[@type='button' and @class='btn btn' and @id='dropdownMenuButton']

**++parent & preceding-sibling:**

//a[text()='test2 test2']//parent::td[@class='datalistrow']//preceding-sibling::td[@class='datalistrow']//input

==============================================================================

**Handle Dynamic Search Using Dynamic XPath - Google Search:**

Steps:

1. Google Search – pick a value from suggestions

2. Search using Xpath

3. Get the total count of suggestions

4. Use if condition to match the value

5. Click on matched value

double slace (//) - means total number of child elements

descendant - number of childs(child of child, child of child....)

//ul[@role='rolelist']//li/descendant::div[text()='testing'] - return the list

than iterate using for loop-check condition-click the element

==============================================================================

**How to handle dynamic WebTable In Selenium**

There are two ways of handling WebTable:

Method – 1:

• Iterate row and column and get the cell value using for loop

• Get total rows and iterate table

• Put if(string matches) then select the respective element

• Lengthy method

Method – 2:

• Using custom XPath

• Using parent and preceding-sibling tags

• No need to write for loop

• No full iteration of table

• Single line statement

• More dynamic

• Efficient and fast

==============================================================================

**Steps to handle calendar in Selenium Webdriver:**

1- Click on calendar

2- Get all td of tables using findElements method?

3- using for loop get text of all elements

4- Using if else condition we will check specific date

5- If date is matched then click and break the loop.

6- Handle NoSuchElementException in case of (31st day)

**Handle Calendar/Date Picker using JavaScriptExecutor:**

+Use JavaScriptExecutor to inject JavaScript into DatePicker/Calendar field.

+Very Fast approach

+Direct injection of date into DOM - HTML

Selenium cannot change any HTML Dom structure but JavaScript can do

public static void selectDateByJS(WebElement element, WebDriver driver, String dateValue)

{

JavascriptExecutor js = ((JavascriptExecutor) driver);

js.executeScript("arguments[0].setAttribute('value','"+dateValue+"');", element);

}

==============================================================================

**What is HTML DOM - Document Object Model? Why we get StaleElementReferenceException in Selenium.**

• DOM is an API Interface provided by browser (kind of platform and provided by specific browser)

• When a web page is loaded, browser creates a DOM of the page at the run time.

DOM is divided page wise (every page has separate DOM)

Although most of the DOM structure will be same in all the browser but these are provided by specific browser.

With the DOM, JavaScript gets all the power it needs to create dynamic HTML:

• JavaScript can change all the HTML elements in the page

• JavaScript can change all the HTML attributes in the page

• JavaScript can change all the CSS styles in the page

• JavaScript can remove existing HTML elements and attributes

• JavaScript can add new HTML elements and attributes

• JavaScript can react to all existing HTML events in the page

• JavaScript can create new HTML events in the page

==============================================================================

**Selenium** - have 4 flavour/tool- IDE, RC, WebDriver and Grid

Because of the JavaScript injection to the browser and because of security reason RC is deprecated officially, most of the browser are not allowing JS injection to their sites, So WebDriver came into picture.

Selenium RC - API

Selenium WebDriver - API, well-designed object-oriented API, and Selenium-WebDriver makes direct calls to the browser using each browser’s native support for automation. Again, it drives the browser directly using the browser’s built in support for automation.

**Selenium WebDriver**: still runs Selenium 1’s (Selenium RC) interface for backwards compatibility.

**Selenium IDE** - (Integrated Development Environment) is a prototyping tool for building test scripts. It is a Firefox/Chrome plugin and provides an easy-to-use interface for developing automated tests. Selenium IDE has a recording feature, which records user actions as they are performed and then exports them as a reusable script in one of many programming languages that can be later executed.

**Selenium Grid**- Selenium Grid allows you to run your tests in parallel, different tests can be run at the same time on different remote machines. Improve the Performance and reduce the Time for execution.

**WebDriver + Selenium-Server**

You may, or may not, need the Selenium Server, depending on how you intend to use Selenium-WebDriver. If your browser and tests will all run on the same machine, and your tests only use the WebDriver API then you do not need to run the Selenium-Server, WebDriver will run the browser directly.

There are some reasons though to use the Selenium-Server with Selenium-WebDriver.

* You are using Selenium-Grid to distribute your tests over multiple machines or virtual machines (VMs).
* You want to connect to a remote machine that has a particular browser version that is not on your current machine.
* You are not using the Java bindings (i.e. Python, C#, or Ruby) and would like to use HtmlUnitDriver

**HtmlUnitDriver**

This is currently the fastest and most lightweight implementation of WebDriver. As the name suggests, this is based on HtmlUnit. HtmlUnit is a java based implementation of a Web Browser without a GUI. For any language binding (other than java) the Selenium Server is required to use this driver.

==============================================================================

WARNING: Do not mix implicit and explicit waits. Doing so can cause unpredictable wait times.

For example, setting an implicit wait of 10 seconds and an explicit wait of 15 seconds, could cause a timeout to occur after 20 seconds. WebDriverWait by default calls the ExpectedCondition every 500 milliseconds until it returns successfully. A successful return value for the ExpectedCondition function type is a Boolean value of true, or a non-null object.

**Implicit Waits :**

An implicit wait is used to tell WebDriver to poll the DOM for a certain amount of time when trying to find an element or elements if they are not immediately available. The default timeout is 0. Once set, the implicit wait is applicable for life of the WebDriver object instance.

==============================================================================

When working with IE browser -

* Zoom level should be=100% else SessionNotFoundException will appear
* Protected mode should be same for all, otherwise SessionNotFoundException will appear
* xpath changes in IE
* sendkeys performance is very slow (64 bit version is worst for IE don’t use- it will type character very slow-sendkeys performance will worst for it)

==============================================================================

**Difference b/w CSS and XPATH**

* CSS is faster and the reason is in XPATH-XPATH engine is evaluate your XPATH and locate element but CSS does not have any engine and it directly identify an object in DOM.
* CSS will not change with Browser but XPATH may change in IE

**How can we find CSS?**

* tagname[attribute='value']
* tagname[attribute='value'][attribute2='value2']
* #id
* .classname -if spaces is present in classname than remove those spaces and give dot on that places
* input[id^='customer'][name='log'] - startwith example
* input[id$='Login'] - endwith example
* input[id\*='Email'] - contains example

**How to write XPATH**

* // tagname[@attribute=’value1’]
  + // a [@href=’http://www.google.com’]
  + //input[@id=’name’ OR @class='abc']
  + //input[@name=’username’ AND @class='abc']
  + //img[@alt=’sometext’]
  + //tagname[@attribute1=’value1’][attribute2=’value2’]
  + //a[@id=’id1’][@name=’namevalue1’]
  + //tagname[contains(@attribute,’value1’)]
  + //a[contains(@href,’abc’)][text()='abc']
  + //tagname[starts-with(@attribute-name,’’)]
* Xpath/following::again-ur-regular-path
  + //input[@id=’’]/following::input[1]
  + //a[@href=’’]/following::a[1]
  + //img[@src=’’]/following::img[1]
* preceding node
  + Xpath/preceding::again-ur-regular-path
  + //input[@id=’’]/ preceding::input[1]
  + //a[@href=’’]/ preceding::a[1]
  + //img[@src=’’]/ preceding::img[1]
* /html/head/body/div/input
* //parent-xpath/absolute xpath
  + //input[@id=’section’]/div/input

If 1 XPATH matches with 2 or more than 2 node it will always perform operation on 1st matching node only

==============================================================================When dealing with Bootstrap Login/popup/window (Flipkart login or redbus login etc.)- remember most of the time window will come under Frame so 1st we need to switch to Frame and then we can perform operations and if it is normal window than we can directly perform operations.

After switching, it is now a separate window /alert-you can directly perform any operations

==============================================================================

Bootstrap Dropdown- The bootstrap dropdown is enhanced part of dropdown where you will deal with UL and LI tag of HTML.

To handle this kind of drop-down we must use findElements method and then we can run a for loop to get specific elements.

==============================================================================

Alert - Alert is an Interface

will use switchTo() method for- Alert/Frame/windowHandle

Note– alert is separate window so before using accept() or dismiss() methods we have to switch to alert window using switchTo() method

==============================================================================

Now consider a scenario where alert window comes when certain condition true for this we can create method which will check if alert window present then only it will execute otherwise it will skip this part

**public** **static** **void** handleAlert(WebDriver ldriver) {

**if** (isAlertPresent(ldriver)) {

Alert alert = ldriver.switchTo().alert();

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

alert.accept();

}

}

Important point- If alert in not present in the window and still we try to switchTo alert window then Selenium will throw NoAlertPresentException which will terminate your program so better you should use exception handle also in your script.

**public** **static** **boolean** isAlertPresent(WebDriver driver) {

**try** {

driver.switchTo().alert();

**return** **true**;

} **catch** (NoAlertPresentException ex) {

**return** **false**;

}

}

==============================================================================

**Frames :**

By default, selenium will focus on parent window

* you need to perform any operation on Frame than you need to switchTo that frame and perform operation and after completion come back to parent window

**How to identify Frames?**

* Source code (iframe/ frameset)

Ways to switch to frame-index/id/webelement

if we don’t have any id to find iframe- than so many attribute will be there to find (you will get iframe id or attribute in HTML tag)

**How to handle?**

* switch to frame-perform operation-switch back to parent window

**How to find no of frames?**

* driver.findelements(By.tagname("iframe")).size();
* driver.switchTo().frame(indexnumber);
* driver.switchTo().frame(“framename”);
* driver.switchTo().defaultContent();

Now if you want to find with the help of webelement

my\_frame=driver.findElement(By.xpath("//iframe[@title='selenium\_news']"));

driver.switchTo().frame(my\_frame);

perform operation on frame now and than come back to parent window

==============================================================================

What is JavaScript

* JavaScript is one of the programming languages of the Web.
* JavascriptExecutor is an Interface which is available in package org.openqa.selenium.JavascriptExecutor;
* Inside this Interface we have some predefined method called executeScript()- so whatever script you will pass as a String It will be executed by JavascriptExecutor.
* Note- This is the most common question in the interview that how to type in Selenium without using the sendKeys method.

Program 1- How to type in Selenium without using sendKeys() method

(JavascriptExecutor)driver.executeScript("document.getElementById('some id').value='mukesh';");

Program 2- How to click in Selenium if button or radio button is disable

(JavascriptExecutor)driver.executeScript("document.getElementById('enter your element id').click();");

Program 3- How to uncheck checkbox in Selenium if checkbox is disable

(JavascriptExecutor)driver.executeScript("document.getElementById('enter element id').checked=false;");

Program 4.How to Scroll Page in Selenium Webdriver using Java Script ?

Selenium handle scrolling page automatically but if want to scroll page using JavaScript.

(JavascriptExecutor)driver.executeScript("scroll(0,400)");

Program 5. How to highlight elements Selenium Webdriver using JavaScript ?

(JavascriptExecutor)driver.executeScript("arguments[0].setAttribute('style', 'background: yellow; border: 2px solid red;');", element);

==============================================================================

**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.

**What are the benefits of Automation Testing**?

• Saves time and money. Automation testing is faster in execution and reliable.

• 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.

• 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 type of tests have you automated?**

Our 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 as well. If same test case needs to execute with multiple data sets, test case needs to execute in multiple browser/OS/Environments than these are good candidates for automation.

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

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

**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?**

Here we need to clearly explain each component of Framework.

**Why do you prefer Selenium Automation Tool?**

• Free and open source

• Have large user base and helping communities

• Cross browser compatibility

• Platform compatibility

• Multiple programming languages support

**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.

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,

1. Selenium IDE (Integrated Development Environment)
2. Selenium RC (Remote Control) – selenium 1
3. Selenium WebDriver – selenium 2 & 3 & 4
4. Selenium Grid

**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 achieve distributed test execution

**What is a hub in Selenium Grid?**

A hub is a server or a central point that conrols 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 are the types of WebDriver APIs available in Selenium?**

• Firefox Driver

• Gecko Driver

• InternetExplorer Driver

• Chrome Driver

• HTMLUNIT Driver

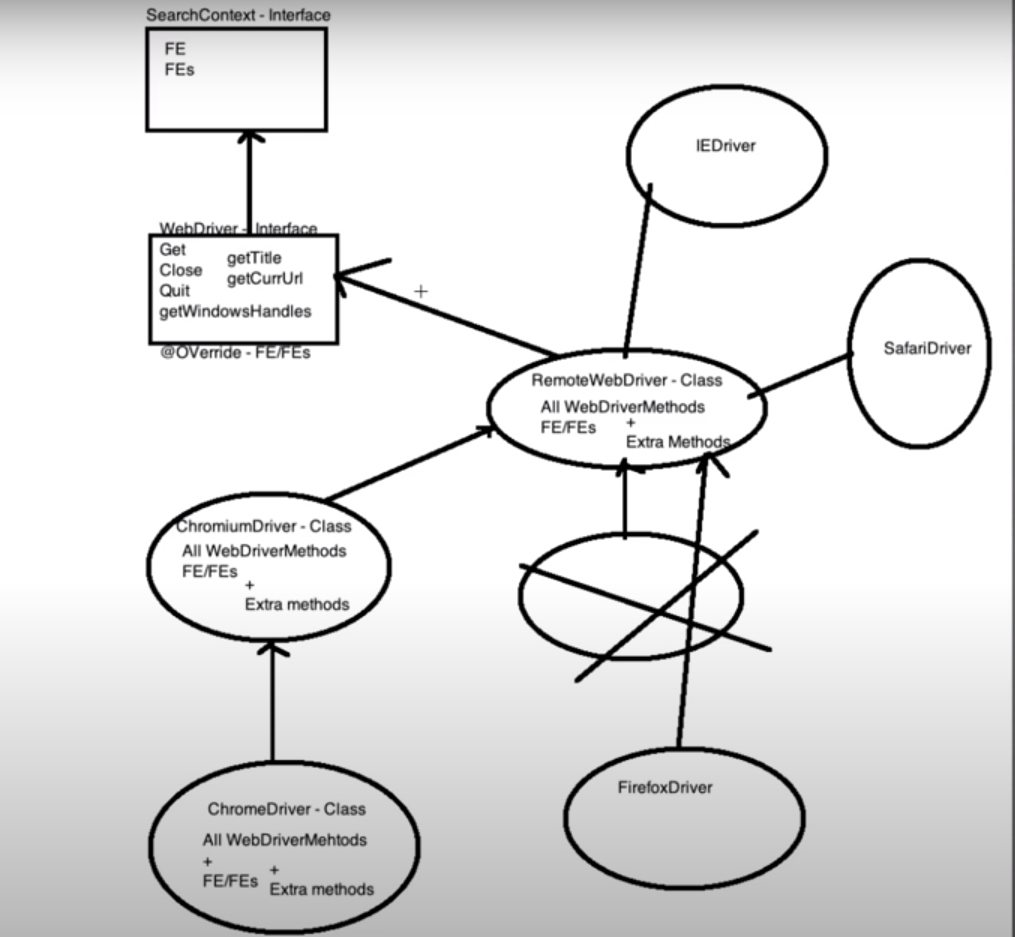
• Opera Driver

• Safari Driver

• Android Driver

• iPhone Driver

• 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 Programming Languages supported by Selenium WebDiver?**

• Java

• C#

• Python

• Ruby

• Perl

• PHP

**What are the Operating Systems supported by Selenium WebDriver?**

• Windows

• Linux

• Apple

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

• JUnit

• TestNG

• Cucumber

• Jbehave

**What are the Locators available in Selenium?**

Different types of locators are:

1 ID

2 ClassName

3 Name

4 TagName – a, input, img,iframe,table

5 LinkText

6 PartialLinkText

7 XPath

8 CSS Selector

**What is XPath?**

XPath is used to locate the elements. Using XPath, we could navigate through elements and attributes in an XML document to locate web elements such as textbox, button, checkbox, Image etc. in a web page.

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

Single Slash “/” – Single slash is used to create absolute path i.e. the XPath would be created to start selection from the document node/start node.

Double Slash “//” – Double slash is used to create relative (Dynamic ) path i.e. the XPath would be created to start selection from anywhere within the document.

**What is the difference between Absolute Path 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']

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

Assert: 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 TestNG?**

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.and at the end we need to call SoftAssert.assertAll() function.

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

**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, logo, image, text, element

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

* SearchContext

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

WebDriver driver = new FirefoxDriver();

This is dynamic polymorphism.

Easily we can do cross browser testing by just changing classname.

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.,

**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

**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 and Fluent Waits.

• Implicit Waits

• Explicit Waits

• Fluent Waits

• PageLoadTimeOut

• Thread.sleep() – static wait, this comes from Java

**What is Fluent Wait in Selenium WebDriver?**

FluentWait can define the maximum amount of time to wait for a specific condition and frequency (polling time) with which to check the condition before throwing an “ElementNotVisibleException” exception.

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

By using getAttribute(value);

**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();

**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 WebDriver Scripts?**

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

**What is the 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, it also stored browser history so we can navigate back and forward, used to launch external url.

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

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.

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

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

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

driver.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

**What is the difference between driver.getWindowHandle() and driver.getWindowHandles() in Selenium WebDriver?**

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.

**What is the difference between driver.close() and driver.quit() methods?**

Purpose of these two methods (driver.close and driver.quit) is almost same. Both allow us to close a browser but still, there is a difference.

driver.close(): To close current WebDriver instance, if will use driver instance again will get : NoSuchSessionException: invalid session id

driver.quit(): To close all the opened WebDriver instances if will use driver instance again will get: NoSuchSessionException: Session ID is null. Using WebDriver after calling quit()?

**What is the difference between driver.findElement() and driver.findElements() commands?**

The difference between driver.findElement() and driver.findElements() commands is-

• findElement() returns a single WebElement (found first and if more than one element is available it return first one) based on the locator passed as parameter. Whereas findElements() returns a list of WebElements, all satisfying the locator value passed.

• Syntax of findElement()-WebElement textbox = driver.findElement(By.id(“textBoxLocator”));

• Syntax of findElements()-List <WebElement> elements = element.findElements(By.id(“value”));

• Another difference between the two is- if no element is found then findElement() throws NoSuchElementException whereas findElements() returns a list of 0 elements(empty list).

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

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();

**How to select a value in a dropdown?**

By using Select class

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

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

dropdown.selectByVisibleText(Text);

dropdown.selectByIndex(Index);

dropdown.selectByValue(Value);

**How to mouse hover on a web element using WebDriver?**

By using Actions class

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

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

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

**How can we handle web based/alert pop-up?**

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

**How can we handle windows/desktop 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.

**How to handle hidden elements in Selenium WebDriver?**

It is one of the most important selenium interview questions

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

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

JavascriptExecutor JS = (JavascriptExecutor) webdriver;

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

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

**How do you read test data from excels?**

Test data can efficiently be read from excel using Apache POI API.

**List some scenarios which we cannot automate using Selenium WebDriver?**

1. Bitmap comparison is not possible using Selenium WebDriver

2. Automating Captcha is not possible using Selenium WebDriver

3. We cannot read bar code using Selenium WebDriver

4. Windows OS based pop ups

5. Third party calendars/element

6. Image

7. Word/PDF

**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.

**How can you use the Recovery Scenario/exception handling in Selenium WebDriver?**

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

**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.

Practical Example.

Browser Button – type =“file”

SendKeys (c:\\test\\naveen.jpg);

**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?**

Run Java Program using Command Prompt

cd c

c: javac A.java

c: java A.java

**How to connect a Database in selenium?**

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

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

**public** **class** ConnectToMySQL {

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

Connection conn = **null**;

Statement stmt = **null**;

ResultSet resultSet = **null**;

Class.*forName*("com.mysql.jdbc.Driver");

conn = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/testdb", "root", "root");

stmt = conn.createStatement();

resultSet = stmt.executeQuery("select \* from sampletable");

**while** (resultSet.next()) {

System.***out***.println(resultSet.getString(1) + " " + resultSet.getString(2) + " " + resultSet.getString(3)

+ " " + resultSet.getString(4) + " " + resultSet.getString(5));

}

**if** (resultSet != **null**) {

**try** {

resultSet.close();

} **catch** (Exception e) {

}

}

**if** (stmt != **null**) {

**try** {

stmt.close();

} **catch** (Exception e) {

}

}

**if** (conn != **null**) {

**try** {

conn.close();

} **catch** (Exception e) {

}

}

}

}

**How to Resize Browser Window Using Selenium WebDriver**?

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

Dimension d = **new** Dimension(480,620);

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

**How To Perform Right Click Action (Context Click) In Selenium WebDriver?**

All the advance activities are performed by **Actions** class like: mouse movement, right click, double click, drag and drop, click and hold, release, keyboard events

WebElement element=driver.findElement(By.id("abc"));

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

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

action.doubleClick(element);

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

action.dragAndDrop(source, target);

==============================================================================

**What are the different ceremonies in Agile Scrum?**

Sprint (or iteration) is the heart of Scrum. It calls for below ceremonies that bring structure to each sprint:

• **Sprint Planning**: Held at the start of each sprint to define the Sprint Backlog (importing stories from the Product/Release backlog), i.e. items that can be completed in the current sprint. As you might have guessed, the Product Owner drives Sprint Planning as in which stories are highest in priority.

• **Daily Scrum**: Presided over by the Scrum Master, Daily Scrum is a 15-minute stand-up meeting to synchronize the work of team members, i.e. what’s done on the prior day, what needs to be done today, and identify any impediments. It is also a means to track Sprint progress.

• **Sprint Review**: Held at the end of each sprint to demonstrate the added functionality. The goal is to get feedback from the product owner and other stakeholders to ensure that the delivered increment met the business need and to revise the Product Backlog based on the feedback. Also Dev/QA team can provide demo to the PO or Stack holder.

• **Sprint Retrospective**: Held at the end of each sprint to reflect on the completed sprint and identify opportunities to improve in the next – what went well, what did not and what can be improved. It allows the team to focus on its overall performance and identify strategies for continuous improvement.

==============================================================================

**how to get text from hidden elements?**

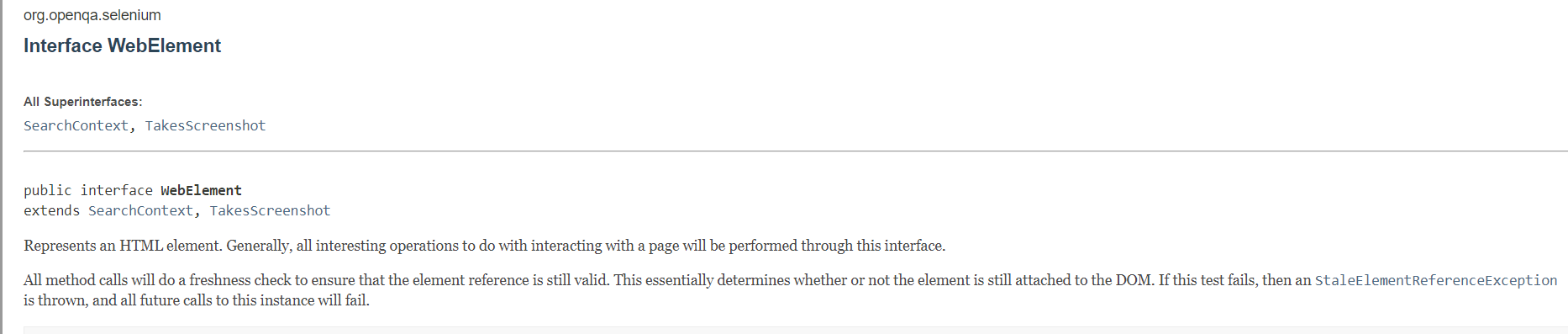
String script = "return arguments[0].innerHTML";

As defined in WebDriver spec, Selenium WebDriver will only interact with visible elements, therefore the text of an invisible element will always be returned as an empty string.

However, in some cases, one may find it useful to get the hidden text, which can be retrieved from element's textContent, innerText or innerHTML attribute, by calling element.attribute('attributeName') or injecting JavaScript like return arguments[0].attributeName.

**What is WebElement & explain all the Mtds available in WebElement**

WebElement is an interface.



**Give one example of method overloading concept used in Selenium?**

* Switching to frames

**What are some common exceptions in Selenium Webdriver?**

• ElementNotVisibleException : Although an element is present in the DOM, it is not visible (cannot be interacted with). E.g. Hidden Elements – defined in HTML using type=”hidden”.

• ElementNotSelectableException : Although an element is present in the DOM, it may be disabled (cannot be clicked/selected).

• InvalidSelectorException : Selector used to find an element does not return a WebElement. Say XPath expression is used which is either syntactically invalid or does not select WebElement.

• NoSuchElementException : WebDriver is unable to identify the elements during run time, i.e. FindBy method can’t find the element.

• NoSuchFrameException : WebDriver is switching to an invalid frame, which is not available.

• NoAlertPresentException : WebDriver is switching to an invalid alert, which is not available.

• NoSuchWindowException : WebDriver is switching to an invalid window, which is not available.

• StaleElementReferenceException : The referenced element is no longer present on the DOM page (reference to an element is now Stale). E.g. The Element belongs to a different frame than the current one OR the user has navigated away to another page.

• SessionNotFoundException : The WebDriver is performing the action immediately after ‘quitting’ the browser.

• TimeoutException : The command did not complete in enough time. E.g. the element didn’t display in the specified time. Encountered when working with waits.

• WebDriverException : The WebDriver is performing the action immediately after ‘closing’ the browser.

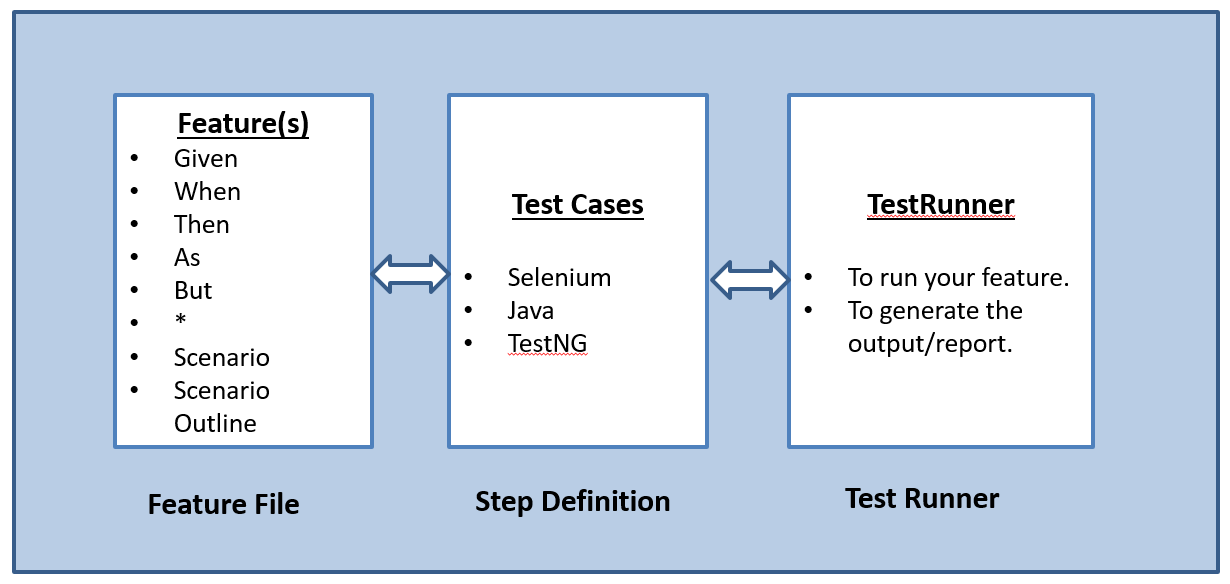
==============================================================================

**Explain Cucumber shortly.**

Ans: Cucumber is a Framework that is based on Behaviour Driven Development (BDD) methodology.

The main aim of Behaviour Driven Development framework is to make various project roles such as Business Analysts, Quality Assurance, Developers etc., understand the application without diving deep into the technical aspects.

**Cucumber Framework Architecture:**



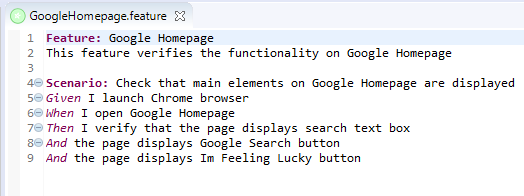
**what language is used by Cucumber?**

Ans: Gherkin is the language that is used by the Cucumber tool. It is a simple English representation of the application behaviour. Gherkin language uses several keywords to describe the behaviour of application such as Feature, Scenario, Scenario Outline, Given, When, Then etc.

**What is meant by a feature file?**

Ans: A feature file must provide a high-level description of an Application Under Test (AUT). The first line of the feature file must start with the keyword ‘Feature’ following the description of the application under test.

A feature file may include multiple scenarios within the same file. A feature file has the extension .feature.

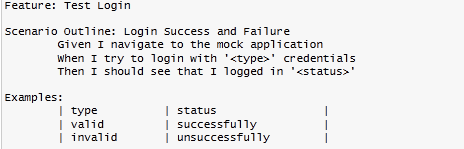


**What are the various Gherkin keywords that are used in Cucumber for writing a scenario?**

Ans: Mentioned below are the keywords that are used for writing a scenario:

* Given
* When
* Then
* And

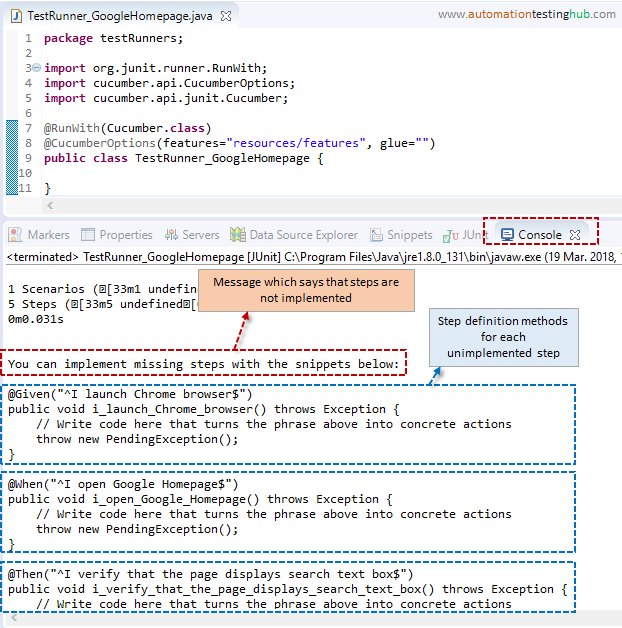
**What is the purpose of Scenario Outline in Cucumber?**

Ans: Scenario outline is a way of parameterization of scenarios. This is ideally used when the same scenario needs to be executed for multiple sets of data, however, the test steps remain the same. Scenario Outline must be followed by the keyword ‘Examples’, which specify the set of values for each parameter. 

**What programming language is used by Cucumber?**

Ans: Cucumber tool provides support for multiple programming languages such as Java, .Net, Ruby etc. It can also be integrated with multiple tools such as Selenium, Capybara etc.

**What is the purpose of Step Definition file in Cucumber?**

Ans: A step definition file in Cucumber is used to segregate the feature files from the underlying code. Each step of the feature file can be mapped to a corresponding method on the Step Definition file.While feature files are written in an easily understandable language such as Gherkin, Step Definition files are written in programming languages such as Java, .Net, Ruby etc. 

**What are the major advantages of Cucumber framework?**

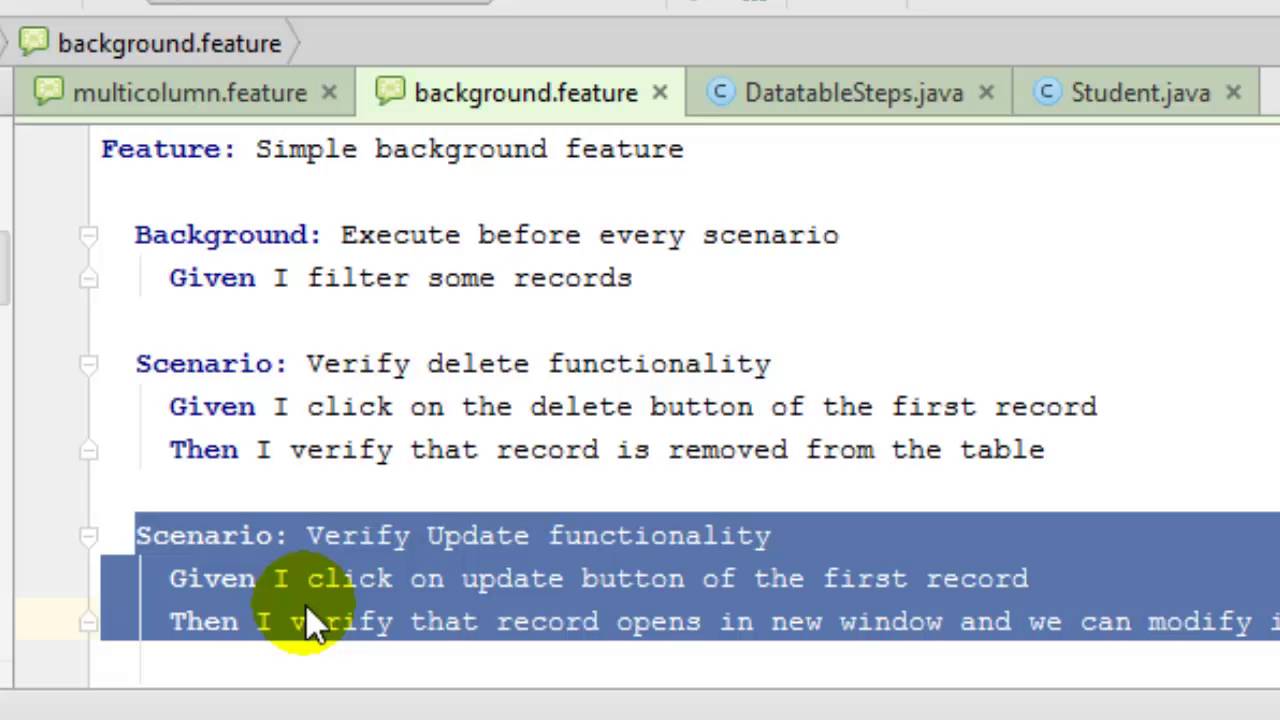
Ans: Given below are the advantages of Cucumber Gherkin framework that make Cucumber an ideal choice for rapidly evolving agile methodology in today’s corporate world.

* Cucumber is an open source framework.
* Plain Text representation makes it easier for non-technical users to understand the scenarios.
* It bridges the communication gap between various project stakeholders such as Business Analysts, Developers, and Quality Assurance personnel.
* Automation test cases developed using the Cucumber tool are easier to maintain and understand as well.
* Easy to integrate with other tools such as Selenium.

**What is the limit for the maximum number of scenarios that can be included in the feature file?**

Ans: A feature file can contain a maximum of 10 scenarios, but the number can vary from project to project and from one organization to another. But it is generally advisable to limit the number of scenarios included in the feature file.

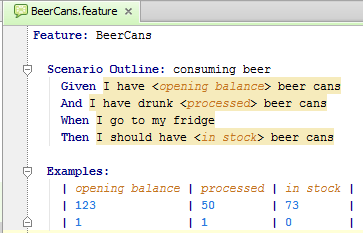
**What is the use of Background keyword in Cucumber?**

Ans: Background keyword is used to group multiple given statements into a single group. This is generally used when the same set of given statements are repeated in each scenario of the feature file. 

**What symbol is used for parameterization in Cucumber?**

Ans: Pipe symbol (|) is used to specify one or more parameter values in a feature file.

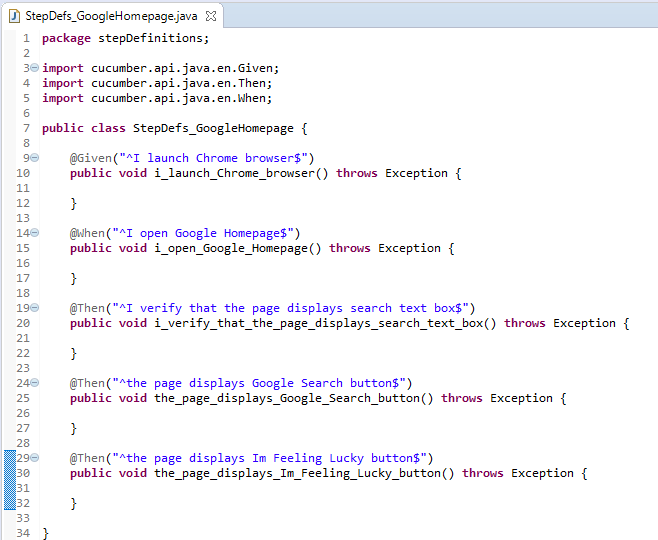
**What is the purpose of Examples keyword in Cucumber?**

Ans: Examples keyword is used to specify values for each parameter used in the scenario. Scenario Outline keyword must always be followed by the keyword Examples. 

**What is the file extension for a feature file?**

Ans: File Extension for a feature file is .feature. A feature file is ideally written in a notepad file and is saved with the extension feature.

Q #17) Provide an example of step definition file in Cucumber.



**What is the purpose of Cucumber Options tag?**

Ans: Cucumber Options tag is used to provide a link between the feature files and step definition files. Each step of the feature file is mapped to a corresponding method on the step definition file.Below is the syntax of Cucumber Options tag:

@CucumberOptions(features="Features",glue={"StepDefinition"})

**How can Cucumber be integrated with Selenium WebDriver?**

Ans: Cucumber can be integrated with Selenium webdriver by downloading the necessary JAR files.

Given below are the list of JAR files that are to be downloaded for using Cucumber with Selenium web driver:

cucumber-core-1.2.2.jar

cucumber-java-1.2.2.jar

cucumber-junit-1.2.2.jar

cucumber-jvm-deps-1.0.3.jar

cucumber-reporting-0.1.0.jar

gherkin-2.12.2.jar

**Explain the purpose of keywords that are used for writing a scenario in Cucumber.**

Ans: “Given” keyword is used to specify a precondition for the scenario. “When” a keyword is used to specify an operation to be performed. “Then” keyword is used to specify the expected result of a performed action. “And” keyword is used to join one or more statements together into a single statement.

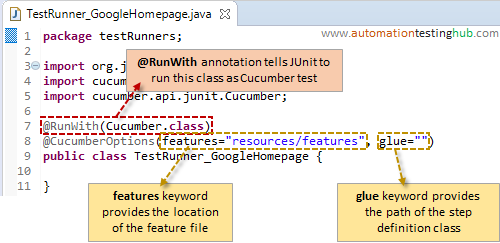
**What is the name of the plugin that is used to integrate Eclipse with Cucumber?**

Ans: Cucumber Natural Plugin is the plugin that is used to integrate Eclipse with Cucumber.

**What is the meaning of TestRunner class in Cucumber?**

Ans: TestRunner class is used to provide the link between feature file and step definition file. Below is the sample representation of how TestRunner class will look like. A TestRunner class is generally an empty class with no class definition.

**Provide an example of TestRunner class in Cucumber.**



**What is the starting point of execution for feature files?**

Ans: When integrated with Selenium, the starting point of execution must be from TestRunner class.

**Should any code be written within TestRunner class?**

Ans: No code should be written under the TestRunner class. It should include the tags @RunWith and @CucumberOptions.

**What is the maximum number of steps that are to be written within a scenario?**

Ans: The maximum number of steps to be written in a scenario is 3-4 steps.

==============================================================================

**How do I explain Java-Selenium automation project in the interview?**

We can start as mentioned below.

1. We are using Page Object model with Page Factory implementation.

2. We have maintained a page class for every page in our application and a test class to maintain test for that pages. e.g. Product listing page, Add to cart page, Payment page, Invoice generation page.

3. We have maintained separate package for page and page test e.g.

4. We also have a base page class for common functions use by all the pages/tests.

5. We also have utility package to maintain common functions related to frame/dropdown/waits/ etc

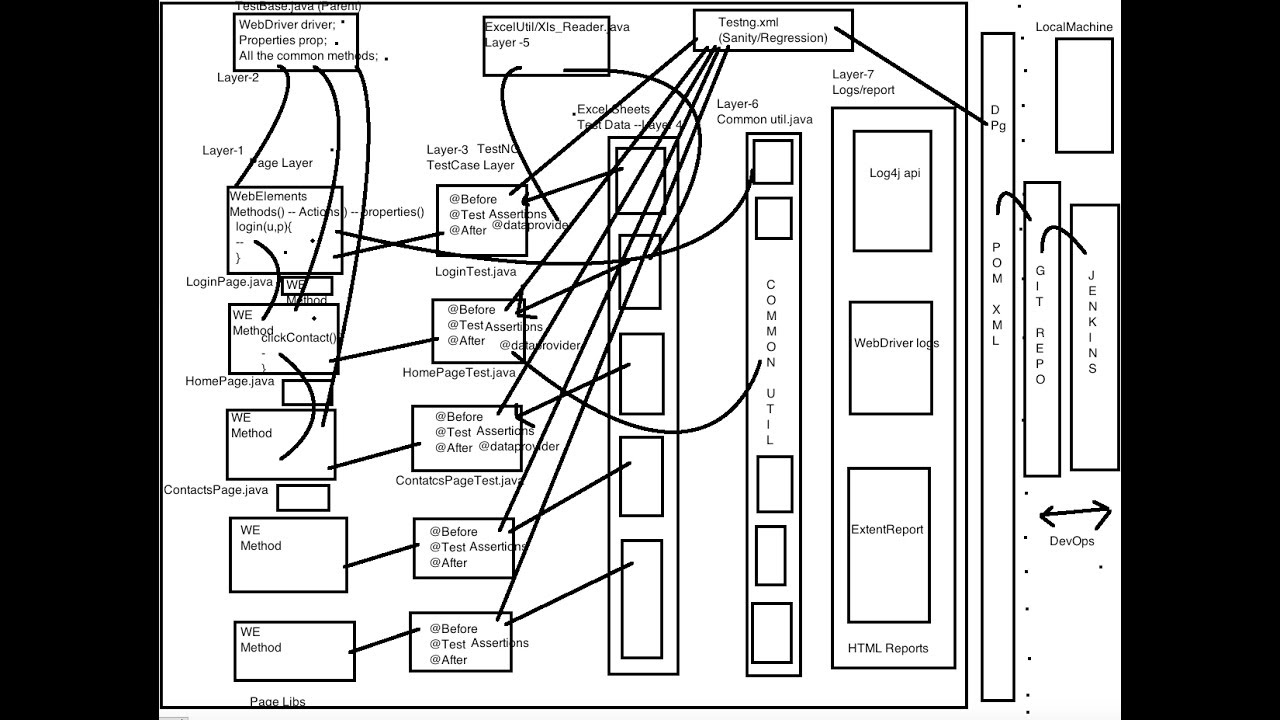
6. We are using standardized maven project for build, execution & dependency management.

7. For handling data driven cases we are passing data using java properties file/xls file.

8. For tests management we are using testNG framework.

10. We are using Extent Report for reporting and logging purpose. It is a third-party library and it is easily available at maven central repo.

We maintain our automation code into client repository using a version control tool git bash on windows system.



**Used TestNG with Selenium? What are some of the annotations used?**

A. Annotations are lines of code that help in passing the configuration information. They are always preceded by @ symbol.

@BeforeSuite : Once, before all Tests in the suite.

@BeforeTest : Once, Prior to the first Test case in the TestNG file.

@BeforeClass : Once before the first Test method in the current class.

@BeforeMethod : Before each Test method.

@Test: Actual Test case, the business logic.

@AfterMethod : After each Test method. @AfterClass : Once after all the Test methods in the current class.

@AfterTest : Once, after all Test cases in the TestNG file.

@AfterSuite : Once, after all Tests in the suite.

@BeforeGroups : The list of groups that this method will run before.

@AfterGroups : The list of groups that this method will run after.

@Parameters : to pass parameters in Test methods.

@DataProvider : Marks a method as supplying data for a Test method.

@Factory : Marks a method as a factory that returns objects that will be used by TestNG as Test classes.

@Listeners : Defines listeners on a Test class, helpful for logging purpose.

==============================================================================

**Find the position of the X and Y co-ordinate .**

driver.manage().window().getPosition().getX();

driver.manage().window().getPosition().getY();

**Explain the different exceptions in Selenium WebDriver.**

An exception in Java in an event, which can disturb our program flow or you, can say it can terminate our program.

In Java, we can handle exception using try- catch block. Try can have multiple catch blocks.

Try must have catch block or finally block.

Note- Make Sure Parent Exception should come in last catch block otherwise we will face code, not reachable error.

**illegalstateexception** - when you are working with 3rd party browser that time you need to set the location of the driver else you get this exception.

**NoSuchElementException** - This exception occurs when WebDriver is unable to identify the elements during run time. Due to wrong selector or selector which is not exist.

**ElementNotVisibleException** - This Exception occurs when the element presence is in DOM, it is not visible

Example:-

Hidden Elements, which has presence in DOM and it is not visible. Visibility means the height and width should be greater than zero. Hidden Elements are defined in HTML using of type=”hidden”.

**Reasons for ElementNotVisibleException**

**Reason 1- Duplicated XPATH**

While writing xpath for your application, you might have taken xpath that is matching with more than 1 element, in this case, Selenium will throw Element, not the visible exception.

**Reason 2-** If you are trying to access some particular element on Webpage that is not currently visible, in this case also you will get the Element, not visible exception.

**First Solution:** Try to write unique XPATH that matches with a single element only.

**Second Solution:** Use Explicit wait feature of Selenium and wait till the element is not visible. Once it is visible then you can perform your operations.

**Third solution:**

int ok\_size=driver.findElements(By.xpath("//button[text()='OK']")).size();

driver.findElements(By.xpath("//button[text()='OK']")).get(ok\_size-1).click();

**NoSuchFrameException** - This Exception occurs when the driver is switching to an invalid frame, which is not available.

**NoAlertPresentException** - This Exception occurs when the driver is switching to an invalid Alert, which is not available.

**NoSuchWindowException** - This Exception occurs when the driver is switching to an invalid Window, which is not available.

**SessionNotFoundException** - This Exception occurs when the driver is performing the action after immediately closing/quitting the browser.

**InvalidStateException**- when there is button but we want to type or there is a texbox(disable) but we want to write that time will get this exception.

**StaleElementReferenceException** - This Exception occurs when the Element is removed from the DOM or no longer present in the DOM.

**How to solve stale element reference exception in selenium**

**Solution 1–**

You can refresh/reload the page and again try for the same element.

Example- If you are trying to click on link and getting the exception then try in below format

*driver*.navigate().refresh();

*driver*.findElement(By.*id*("abc"));

**Solution 2-**

Sometimes it takes the time to attach element on Dom so you can retry using for loop and try catch.

**for**(**int** i=0;i<=2;i++)

{

**try**{

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

**break**;

}

**catch**(Exception e){

e.getMessage();

}

**Solution 3: try to find element in catch block as well**

**try** {

*driver*.findElement(By.*id*("abc"));

Thread.*sleep*(1000);

}

**catch** (Exception e) {

*driver*.findElement(By.*id*("abc"));

}

**Solution 4 : Explicit wait**

new WebDriverWait(driver, 10).until(ExpectedConditions.visibilityOf(element));

**WebDriverException: f.QueryInterface is not a function**

While passing URL in Selenium we have to pass protocol as well so that Selenium can understand which protocol to use while communicating.

There are so many protocols available some of them are

FTP, SMTP, HTTP etc.

Solution:

While passing URL we have to mention HTTP also.

driver.get(“http://www.learn-automation.com”);

**ElementNotClickableException:**

**The reason for the element is not clickable at point(x,y) exception.**

Some of my observation was

It mostly happens in Chrome so if you are mostly working with Firefox or IE then you will not be getting this exception.

Chrome does not calculate the exact location of element

Chrome always click in the middle of Element.

Sometimes you will get this exception due to Sync issue also.

WebElement elementToClick = driver.findElement(By.xpath("Your xpath"));

((JavascriptExecutor)driver).executeScript("window.scrollTo(0,"+elementToClick.getLocation().y+")");

elementToClick.click();

OR

WebElement elementToClick = driver.findElement(By.xpath("Your xpath"));

((JavascriptExecutor)driver).executeScript("window.scrollTo(0,"+elementToClick.getLocation().x+")");

elementToClick.click();

**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)**

This is used for Negative test cases.

**How can you redirect browsing from a browser through some proxy?**

Selenium provides a PROXY class to redirect browsing from a proxy. Look at the example below:

|  |  |
| --- | --- |
|  | String PROXY = “199.201.125.147:8080”;  Proxy proxy = new Proxy();  proxy.setHTTPProxy(Proxy)   .setFtpProxy(Proxy)   .setSslProxy(Proxy)  DesiredCapabilities cap = new DesiredCapabilities();  cap.setCapability(CapabilityType.PROXY, proxy);  WebDriver driver = new FirefoxDriver(cap); |

**What is POM (Page Object Model)? What are its advantages?**

1- It is design pattern in which will help you to maintain the code and code duplication, which is a crucial thing in Test automation.

2- You can store all locators and respective methods in the separate class and Call them from the test in which you have to use. So the benefit from this will be if any changes in Page then you do not have to modify the test simply modify the respective page.

3- You can create a layer between your test script and application page, which you have to automate.

4- In other words, it will behave as Object repository where all locators are saved.

**What is Page Factory?**

Page Factory gives an optimized way to implement Page Object Model. When we say it is optimized, it refers to the fact that the memory utilization is very good and also the implementation is done in an object oriented manner.

Page Factory is used to initialize the elements of the Page Object or instantiate the Page Objects itself. Annotations for elements can also be created (and recommended) as the describing properties may not always be descriptive enough to differentiate one object from the other.

The concept of separating the Page Object Repository and Test Methods is followed here also. Instead of having to use ‘FindElements’, we use annotations like: **@FindBy** to find WebElement, and **initElements** method to initialize web elements from the Page Factory class.

**@FindBy** can accept **tagName**, **partialLinkText**, **name**, **linkText**, **id**, **css**, **className**& **xpath**as attributes.

**What are the different types of WAIT statements in Selenium WebDriver? *Or the question can be framed like this:* How do you achieve synchronization in WebDriver?**

There are basically two types of wait statements: **Implicit Wait** and **Explicit Wait**.

Implicit wait instructs the WebDriver to wait for some time by polling the DOM. Once you have declared implicit wait, it will be available for the entire life of the WebDriver instance. By default, the value will be 0. If you set a longer default, then the behavior will poll the DOM on a periodic basis depending on the browser/ driver implementation.

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

**Write a code to wait for a particular element to be visible on a page. Write a code to wait for an alert to appear.**

We can write a code such that we specify the XPath of the web element that needs to be visible on the page and then ask the WebDriver to wait for a specified time. Look at the sample piece of code below:

|  |  |
| --- | --- |
|  | **WebDriverWait wait=new WebDriverWait(driver, 20);**  **Element = wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath( “<xpath”)));** |

Similarly, we can write another piece of code asking the WebDriver to wait until an error appears like this:

|  |  |
| --- | --- |
|  | **WebDriverWait wait=new WebDriverWait(driver, 20);**  **Element = wait.until(ExpectedConditions.alertIsPresent());** |

**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); |

**Which files can be used as data source for different frameworks?**

Some of the file types of the dataset can be: excel, xml, text, csv, etc.

**18.How to send ALT/SHIFT/CONTROL key in Selenium WebDriver?**

**When we generally use ALT/SHIFT/CONTROL keys, we hold onto those keys and click other buttons to achieve the special functionality. So it is not enough just to specify keys.ALT or keys.SHIFT or keys.CONTROL functions.**

**For the purpose of holding onto these keys while subsequent keys are pressed, we need to define two more methods: keyDown(modifier\_key) and keyUp(modifier\_key)**

**Parameters: Modifier\_key (keys.ALT or Keys.SHIFT or Keys.CONTROL)**

**Purpose: Performs a modifier key press and does not release the modifier key. Subsequent interactions may assume it’s kept pressed.**

**Parameters: Modifier\_key (keys.ALT or Keys.SHIFT or Keys.CONTROL)**

**Purpose: Performs a key release.**

**Hence with a combination of these two methods, we can capture the special function of a particular key.**

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

Action seriesOfActions = builder.moveToElement(txtUerName).click()

.keyDown(txtUserName, Keys.SHIFT)

.sendKeys(txtUserName, “hello”)

.keyUp(txtUserName, Keys.SHIFT)

.doubleClick(txtUserName)

.contextClick()

.build();

seriesOfActions.perform();

## }

## How to Capture screenshot in selenium for failed test cases

1-If script has some issue (some locator has been changed or application has some changes)- In this case, we need to maintain our Selenium script.

2-Due to application issue- In this case, we need to inform to respective point of contact (Manual Tester or Developer)

Here I will be using two new topics which will help us to achieve the same.

1-We will use ITestResult Interface which will provide us the test case execution status and test case name.

Please refer official doc for **ITestResult**

2- @AfterMethod is another annotation of TestNG which will execute after every test execution whether test case pass or fail @AfterMethod will always execute.

## Program to Capture screenshot in selenium for failed test cases

// It will execute after every test execution

@AfterMethod

**public** **void** tearDown(ITestResult result) {

**if** (ITestResult.FAILURE == result.getStatus()) {

**try** {

TakesScreenshot ts = (TakesScreenshot) driver;

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

FileUtils.copyFile(source, **new** File("./Screenshots/" + result.getName() + ".png"));

} **catch** (Exception e) {

System.***out***.println("Exception while taking screenshot " + e.getMessage());

}

}

## ****How to set the size of browser window using Selenium?****

To maximize the size of browser window, you can use the following piece of code:  
**driver.manage().window().maximize();**

To resize the current window to a particular dimension, you can use the **setSize()** method. Check out the below piece of code:

|  |  |
| --- | --- |
|  | System.out.println(driver.manage().window().getSize());  Dimension d = new Dimension(420,600);  driver.manage().window().setSize(d); |

To set the window to a particular size, use **window.resizeTo()** method. Check the below piece of code:

|  |  |
| --- | --- |
|  | **((JavascriptExecutor)driver).executeScript("window.resizeTo(1024, 768);");** |

## ****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.

**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.

**public** **class** WindowHandle3 {

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

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

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

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

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

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

String parentWindow = driver.getWindowHandle();

Set<String> s1 = driver.getWindowHandles();

**for** (String child : s1) {

**if** (!parentWindow.equals(child)) {

driver.switchTo().window(child);

driver.close();

}

}

ArrayList<String>list=**new** ArrayList<>(s1);

## list.get(1);

## }

## }

## The reason why we use Set in window handling is Set cannot contains duplicate but list can.

## ****Explain how**** ****you will login into any site if it is showing any authentication popup for username and password?****

Since there will be popup for logging in, we need to use the explicit command and verify if the alert is actually present. Only if the alert is present, we need to pass the username and password credentials. The sample code for using the explicit wait command and verifying the alert is below:

|  |  |
| --- | --- |
|  | WebDriverWait wait = new WebDriverWait(driver, 10);  Alert alert = wait.until(ExpectedConditions.alertIsPresent());  alert.authenticateUsing(new UserAndPassword(\*\*username\*\*, \*\*password\*\*)); |

## ****Explain how can you find broken links in a page using Selenium WebDriver**?**

This is a trick question which the interviewer will present to you. He can provide a situation where in there are 20 links in a web page, and we have to verify which of those 20 links are working and how many are not working (broken).

Since you need to verify the working of every link, the workaround is that, you need to send http requests to all of the links on the web page and analyze the response. Whenever you use driver.get() method to navigate to a URL, it will respond with a status of **200 – OK**. 200 – OK denotes that the link is working and it has been obtained. If any other status is obtained, then it is an indication that the link is broken.

But how will you do that?

First, we have to use the anchor tags <a> to determine the different hyperlinks on the web page.

For each <a> tag, we can use the attribute ‘href’ value to obtain the hyperlinks and then analyze the response received for each hyperlink when used in **driver.get()** method.

find broken links using selenium it means we need to check the link which is pointing to wrong URL or invalid URL.

 404 page not found an issue in most of the application which is called **broken link.**

**Approach 1-**

Manual Process- Go to each link and verify the link is working or not.

**Approach 2-**

Smart work- Write a code which will check all the link and will verify the status as well.

## ****What is the significance of testng.xml?****

Since Selenium does not support report generation and test case management, we use TestNG framework with Selenium. TestNG is much more advanced than JUnit, and it makes implementing annotations easy. That is the reason TestNG framework is used with Selenium WebDriver.

But have you wondered where to define the test suites and grouping of test classes in TestNG?

It is by taking instructions from the testng.xml file. We cannot define a test suite in testing source code, instead it is represented in an XML file, because suite is the feature of execution. The test suite, that I am talking about is basically a collection of test cases.

So for executing the test cases in a suite, i.e a group of test cases, you have to create a testng.xml file which contains the name of all the classes and methods that you want to execute as a part of that execution flow.

Other advantages of using testng.xml file are:

* It allows execution of multiple test cases from multiple classes
* It allows parallel execution
* It allows execution of test cases in groups, where a single test can belong to multiple groups

## ****What is parameterization in TestNG? How to pass parameters using testng.xml?****

Parameterization is the technique of defining values in testng.xml file and sending them as parameters to the test class. This technique is especially useful when we need to pass multiple login credentials of various test environments. Take a look at the code below, in which “myName” is annotated as a parameter.

**public** **class** ParameterizedTest1 {

@Test

@Parameters("myName")

**public** **void** parameterTest(String myName) {

System.***out***.println("Parameterized value is : " + myName);

}

}

To pass parameters using testng.xml file, we need to use ‘parameters’ tag. Look at the below code for example:

|  |  |
| --- | --- |
|  | <?xml version="1.0" encoding="UTF-8"?>  <!DOCTYPE suite SYSTEM "http://testng.org/testng-1.0.dtd" >  <suite name=”CustomSuite">  <test name=”CustomTest”>  <parameter name="myName" value=”John"/>  <classes>  <class name="ParameterizedTest1" />  </classes>  </test>  </suite> |

## ****Explain DataProviders in TestNG using an example. Can I call a single data provider method for multiple functions and classes**?**

DataProvider is a TestNG feature, which enables us to write DataDriven tests. When we say, it supports DataDriven testing, then it becomes obvious that the same test method can run multiple times with different data-sets. DataProvider is in fact another way of passing parameters to the test method.

**@DataProvider** marks a method as supplying data for a test method. The annotated method must return an Object[] where each Object[] can be assigned to parameter list of the test method.

To use the DataProvider feature in your tests, you have to declare a method annotated by **@DataProvider** and then use the said method in the test method using the ‘dataProvider‘ attribute in the Test annotation.

As far as the second part of the question is concerned, Yes, the same DataProvider can be used in multiple functions and classes by declaring DataProvider in separate class and then reusing it in multiple classes.

## ****How to skip a method or a code block in TestNG?****

If you want to skip a particular test method, then you can set the ‘enabled’ parameter in test annotation to false. @Test(enabled = false)

By default, the value of ‘enabled’ parameter will be true. Hence it is not necessary to define the annotation as true while defining it.

## ****What is soft assertion in Selenium? How can you mark a test case as failed by using soft assertion?****

Soft Assertions are customized error handlers provided by TestNG. Soft Assertions do not throw exceptions when assertion fails, and they simply continue to the next test step. They are commonly used when we want to perform multiple assertions.

To mark a test as failed with soft assertions, call **assertAll()** method at the end of the test.

**What is priority feature in TestNG? In addition, how we can use this?**

Ans:- If you want the @Test methods to be executed in your order, then you need to use the ‘priority‘ .

Like First we need to execute a test case "Registration" before login. In order to achive, we use need to add annotation as @Test(priority=??).

Sequence of execution is in ascending order of value. “0”is having highest priority so it will be executed first. There is no need for your values to be consecutive

Example:- @Test (priority=0)

Note : Priority can be negative and the execution will be from Negative to Positive.

**How to execute multiple test cases in Selenium?**

Ans:- using TestNG.xml

**How to execute parallel test cases in Selenium?**

Ans:- There are two ways

1> using DataProvideranotation :- @DataProvider(parallel=true)

2> using testng.xml :- <suite name="My suite" parallel="methods" >

**How to execute only failed test cases in Selenium?**

Ans:-

Every time tests fail in a suite, TestNG creates a file called testng-failed.xml in the output directory.

This XML file contains the necessary information to rerun only these methods that failed, utlizing this we can easily rerun the failed tests without having to run the entire test suite.

## ****Explain what is Group Test in TestNG?****

In TestNG, methods can be categorized into groups. When a particular group is being executed, all the methods in that group will be executed.  We can execute a group by parameterizing it’s name in group attribute of**@Test** annotation. Example: @Test(groups={“xxx”})

|  |  |
| --- | --- |
|  | @Test(groups={“Car”})  public void drive(){  system.out.println(“Driving the vehicle”);  }  @Test(groups={“Car”})  public void changeGear() {  system.out.println("Change Gears”);  }  @Test(groups={“Car”})  public void accelerate(){  system.out.println(“Accelerating”);  } |

## ****How does TestNG allow you to state dependencies? Explain it with an example.****

**Dependency**is a feature in TestNG that allows a test method to depend on a single or a group of test methods. Method dependency only works if the “depend-on-method” is part of the same class or any of the inherited base classes (i.e. while extending a class). Syntax:  
**@Test(dependsOnMethods = { “initEnvironmentTest” })**

|  |  |
| --- | --- |
|  | @Test(groups={“Car”})  public void drive(){  system.out.println(“Driving the vehicle”);  }    @Test(dependsOnMethods={“drive”},groups={cars})  public void changeGear() {  system.out.println("Change Gears”);  }  @Test(dependsOnMethods={“changeGear”},groups={“Car”})  public void accelerate(){  system.out.println(“Accelerating”);  } |

## ****Explain what does @Test(invocationCount=?) and @Test(threadPoolSize=?) indicate.****

**@Test(invocationCount=?)** is a parameter that indicates the number of times this method should be invoked.  
**@Test(threadPoolSize=?)** is used for executing suites in parallel. Each suite can be run in a separate thread.

To specify how many times @Test method should be invoked from different threads, you can use the attribute**threadPoolSize** along with **invocationCount**. Example:

|  |  |
| --- | --- |
|  | @Test(threadPoolSize = 3, invocationCount = 10)  public void testServer() { |

}

# **How to Automate Radio button and Checkbox in Selenium webdriver.**

Before performing click action, sometimes we need to verify some activity as well, take some example

* You need to verify whether radio button or checkbox is enabled.
* You need to verify whether radio button or checkbox is Displayed on UI or not.
* You need to verify whether checkbox and radio button is default selected or not.

Above validations are must use in script because automation is all about validation only.

we can easily [***verify***](http://learn-automation.com/capture-error-message-in-selenium/)this using some predefined method in Selenium.

|  |
| --- |
| isDisplayed();  isEnabled();  isSelected(); |

# **How to Handle Dropdown in Selenium WebDriver**

WebElement month\_dropdown=driver.findElement(By.id("month"));

Select month=**new** Select(month\_dropdown);

month.selectByIndex(4);

WebElement month\_dropdown=driver.findElement(By.id("month"));

Select month=**new** Select(month\_dropdown);

month.selectByValue(“5”);

WebElement month\_dropdown=driver.findElement(By.id("month"));

Select month=**new** Select(month\_dropdown);

month.selectByVisibleText("Aug");

**Get Selected option from Dropdown.**

WebElement month\_dropdown=driver.findElement(By.id("month"));

Select month=**new** Select(month\_dropdown);

WebElement first\_value=month.getFirstSelectedOption();

String value=first\_value.getText()

**Get All option from dropdown**

WebElement month\_dropdown=driver.findElement(By.id("month"));

Select month=**new** Select(month\_dropdown);

List<WebElement> dropdown=month.getOptions();

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

{

String drop\_down\_values=dropdown.get(i).getText();

System.***out***.println("dropdown values are "+drop\_down\_values);

}

Explanation- getOptions() is a method of Select class which will return List of WebElement then we can iterate using for loop or iterator and using getText() method we can extract values.

getAllOptions().size()-1 which will always select last value from Dropdown.

# **How to Handle Bootstrap Dropdown in Selenium WebDriver**

**First one – What is bootstrap dropdown**

**The second one- How to Select values from the bootstrap dropdown.**

The bootstrap dropdown is enhanced part of dropdown where you will deal with UL and LI tag of HTML and to handle this kind of drop-down we have to use findElements method and then we can run a for loop to get specific elements.

// Dropdown items are coming in <a> tag so below xpath will get all

List<WebElement> list = *driver*.findElementsByXPath("//ul[@class='dropdown-menu']//li/a");

**for** (WebElement ele : list)

{

System.***out***.println("Values " + ele.getAttribute("innerHTML"));

**if** (ele.getAttribute("innerHTML").contains("JavaScript")) {

ele.click();

**break**;

}

}

}

You can also select the values directly using [xpath](http://learn-automation.com/write-dynamic-css-selector-in-selenium/)  and [CSS](http://learn-automation.com/write-dynamic-css-selector-in-selenium/) but that approach is not recommended because direct xpath might change.

In above approach, we can pass a parameter directly so based on test data it will select the values from the list.

# Upload file in Selenium webdriver using Robot class

**There are multiple ways to upload files in Selenium Webdriver.**

* You can directly use sendKeys(“File path”) method but sometime this methods does not work.
* We can use Robot class to upload files in Selenium.
* *We can upload files using AutoIT as well.*

***Robot class is a separate class in Java which will allow us to perform multiple tasks based on our requirement. It generally will throw AWT exception so we need to deal with it accordingly.***

Using Robot class, we can simulate keyboard event in Selenium.

To use keyboard events you have to use to a method of Robot class.

## Robot Class in Selenium Webdriver

keyPress()

keyRelease()

Each key has to be press and release respectively-

  Robot r=new Robot();

  r.keyPress(KeyEvent.VK\_ENTER);

  r.keyRelease(KeyEvent.VK\_ENTER);

**Actions Class-**

For all advance activity in Selenium Webdriver, we can perform easily using Actions class like Drag and Drop, mouse hover, right click, Click and Hold and so on.  
We have predefined method called **dragAndDrop(source, destination)** which is a method of Actions class.

Approach- Find the xpath of the Source  and find the xpath of destination.

Both source and destination in form of WebElement.

Note- Any method of Actions class we need to call perform () method otherwise we will get an exception. If we have series of action in our script using Actions class then we have to call build().perform() method.

Actions act=new Actions(driver);

 act.contextClick(driver.findElement(By.linkText("Gujarati"))).sendKeys(Keys.ARROW\_DOWN).sendKeys(Keys.ARROW\_DOWN).sendKeys(Keys.ENTER).build().perform();

## Limitation of screenshots by Selenium.

1- When any alert comes on screen and if you call screenshot method then it will fail because the alert is windows activity.

2- When running cross browser testing if need to verify that test is running on which browser then you won’t be able to verify because it captures only web view par

### **What is Base Class in Selenium**

* Base class in the main class which will take care of Browser setup, loading [configuration file](http://learn-automation.com/object-repository-in-selenium-webdriver/)and other reusable methods like [screenshot](http://learn-automation.com/how-to-capture-screenshot-for-failed-test-cases-in-selenium-webdriver/), handling [sync issues](http://learn-automation.com/best-way-to-handle-synchronization-in-selenium-webdriver/)and many more.
* Using Base class we can avoid code duplication.
* Reuse code as much we can.

### How Base class works in Selenium

1-When we create base class and if TestCases extends BaseClass then we can use all the methods of Baseclass.

2- Before calling actual @Test Base class methods will get executed and Depends on annotations it will call the respective methods.

3- We can extend this class in all test cases and we can call custom methods as well directly.

==============================================================================

Selenium always opens new profile/fresh instance of browser by default along with no addons/extensions. So there is no need to clear history separately.

**is it not possible to set zooming level 100% and security settings in ie using selenium?**

DesiredCapabilities caps = DesiredCapabilities.internetExplorer();

caps.setCapability(“EnableNativeEvents”, **false**);

caps.setCapability(“ignoreZoomSetting”, **true**);

WebDriver driver = **new** InternetExplorerDriver(caps);

==============================================================================

DesiredCapabilities cap=DesiredCapabilities.chrome();

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

System.*setProperty*("webdriver.chrome.driver","Chrome driver path");

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

DesiredCapabilities cap = DesiredCapabilities.internetExplorer();

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

System.*setProperty*("webdriver.ie.driver", "IE driver path");

WebDriver driver = newInternetExplorerDriver(cap);

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

FirefoxProfile profile=**new** FirefoxProfile()

profile.setAcceptUntrustedCertificates(**true**);

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

text() function always looks for exact matching of text while contains(text(),”) doesn’t look for exact matching text.

**I have following scenario :**

**Click on + Add button then it will add new row in the table.Now here whatever new row added in the table then its hard to find xpath. Second thing i want to add multiple dynamic xpath and want to enter text.**

If you are able to locate + button using xpath then after click on button, try dynamic xpath like //button[text()=’+’]/following::tr[1] to get first row. In similar fashion, you can proceed with other rows. This is just an example to navigate. Actual xpath on your application might differs.

**When the dropdown tag element is not a ‘select’ but a ‘button’**

Selenium by default supports only select tag. If you are seeing other than select then you need to go for some workaround.If we are using drop down but having div class instead of select class what needs to be done

In this case you can use findElements() method which will return list of elements then you can select the item.

**If we have multiple dropdown and all dropdown is depend on previous one selection dropdown. How we will execute second one or how to write the script in webdriver.**

Ex:- one dropdrown for Mobile Brand name. Second dropdown for Mobile Model so mobile model name depend on Brand name. ?

generally, use explicit wait to handle this scenario

**I actually finding different scenario like drag from local machine and drop in website. such as go to and drag a object from machine and drop on specified area. is there any method available in action or other class?? without using any third party tool.**

Actions class comes from Selenium APIs’ and selenium is built for performing actions on webpage only. Therefore, usage of third party tool is must if you want to drag and drop an item from your machine to webpage.

Selenium works with browsers only so if you want to work with local drive then you have to check other tools like Sikuli and AutoIT.

**What is difference between build() and perform() method?**

build() is called when you have to compile more than one action in a single step or in simple words it like adding sequence of actions to buffer and later we use perform() to execute it.

If you have single operation then its ok but when you have multiple operation then build().perform() is must.

**Default timeout is ZERO. Default polling time is 250 millisecond if implicit wait given.**

==============================================================================

My code works fine with implicit wait on firefox. When I run the same code in chrome, I get No such element exception. I increased the implicit wait time to 100 seconds but still it throws no such element exception as soon as it logs in before waiting for 100secs (implicit wait time). The only way I could get my code work in chrome is by adding explicit wait on each element. Can you please let me know why is implicit wait not working in chrome in my case? Is there any other solution than adding explicit wait to each element to all my scripts?

If you are working with cross browsers, then sometime xpath will change. Try to use CSS which will remain same for all browsers.

==============================================================================

I am just wandering how your code is working without using System.SetProperty(“webdriver.chrome.driver”, “chromedriver.exe path”).

Yes, we can do if you set the path in Env variable then You don’t have to define the path in every script.

==============================================================================

**I have below doubts:**

**1. If we do not use any types of wait statements in program, what is default time for which webdriver will search for an element?**

**2. Is there any timeout for web page loading in selenium?**

**3. Can we increase wait time at run time means can we make webdriver to wait for some extra time to find out en element?**

1- by default wait is zero and if some wait are given then by default polling is 250 mili second.

2- Yes page load timeout is present if you want to set page load timeout. If no timeout given then page will wait until full page loaded.

3- At run time we can’t change the time out.

**How to get the webelemnt height & width**

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

element.getRect().getHeight();

element.getRect().getWidth();

element.getSize().getHeight();

element.getSize().getWidth();

**What is Synchronization.**

It is a mechanism which involves more than one component to work parallel with Each other.

Generally, in Test Automation, we have two components

1. Application Under Test

2. Test Automation Tool.

Both these components will have their own speed. We should write our scripts in such a way that both the components should move with same and desired speed, so that we will not encounter "Element Not Found" errors which will consume time again in debugging.

Synchronization can be classified into two categories:

1. Unconditional

2. Conditional Synchronization

Unconditional :

In this we just specify timeout value only. We will make the tool to wait until certain amount of time and then proceed further.

Examples: Wait() and Thread.Sleep();

The main disadvantage for the above statements are, there is a chance of unnecessary waiting time even though the application is ready.

The advantages are like in a situation where we interact for third party systems like interfaces, it is not possible to write a condition or check for a condition. Here in this situations, we have to make the application to wait for certain amount of time by specifying the timeout value.

Conditional Synchronization:

We specify a condition along with timeout value, so that tool waits to check for the condition and then come out if nothing happens. It is very important to set the timeout value in conditional synchronization, because the tool should proceed further instead of making the tool to wait for a particular condition to satisfy.

**Difference between WebDriverlistner and TestNG listner? Usage of the same?**

Ans:- You must have always craved for more logs from the Webdriver so that you can debug your scripts or may be log more information about your tests.

Here is your answer to it, WebDriverEventListner and TestNgListner(ITestListener Interface.)

In TestNg you can use Listeners in Annotation.

WebDriverEventListener – This is an interface, which have some predefined methods so we will implement all of these methods.

Difference:- TestNG Listener are triggered at test level such as before starting test after the test or when test fails etc;

whereas WebDriver Listener are triggered at component level such as before click, after click etc

**What is log4j And How to generate log files in Selenium?**

Ans:-Log file is just simple file, which keep track of the record or event or info when any event happens or any software run. This whole process known as logging.

We can create log file as simple log file as well as HTML format.

Log4j is open source project from Apache which help us achive above mentioned things.

**Q> What are profiles in selenium? (Dautche bank, 3I infotech)**

Ans :- This feature is available only with firefox.

If you want to open an instance of firefox which has your plugins & bookmarks then one can use firefox profile .

FirefoxProfile profile = new FirefoxProfile(new File("D:\\Selenium Profile"));

WebDriver driver = new FirefoxDriver(profile);

**Q> How would you automate charts/graphs/tool tips? (NTT data,JP Morgan, Mphasis )**

Ans:-Many web applications contains charts to show different data In chart view. One of the chart they are using Is Pie chart. It Is very Important for us to know how to **automate** these **pie charts In selenium WebDriver** test. Few of the charts are very easy to automate but It Is very hard to automate some charts as we are unable to locate parts of pie chart.

@Test

public void pieChart(){

**//Locate pie chart elements based on different colors.**

WebElementVioleteColor = driver.findElement(By.xpath("//\*[contains(@class,'yui3-svgSvgPieSlice')][@fill='#66007f']"));

WebElementGreenColor = driver.findElement(By.xpath("//\*[contains(@class,'yui3-svgSvgPieSlice')][@fill='#295454']"));

WebElementGreyColor = driver.findElement(By.xpath("//\*[contains(@class,'yui3-svgSvgPieSlice')][@fill='#e8cdb7']"));

WebElementLightVioleteColor = driver.findElement(By.xpath("//\*[contains(@class,'yui3-svgSvgPieSlice')][@fill='#996ab2']"));

WebElementBrownColor = driver.findElement(By.xpath("//\*[contains(@class,'yui3-svgSvgPieSlice')][@fill='#a86f41']"));

**//locate tooltip pie chart.**

WebElement ToolTip = driver.findElement(By.xpath("//div[contains(@id,'\_tooltip')]"));

**//Click on pie chart parts and get tooltip values.**

System.out.println("-X-X-X-X-X-X-X-X- Violete Part -X-X-X-X-X-X-X-X-");

VioleteColor.click();

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

System.out.println();

System.out.println("-X-X-X-X-X-X-X-X- Grey Part -X-X-X-X-X-X-X-X-");

GreyColor.click();

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

System.out.println();

System.out.println("-X-X-X-X-X-X-X-X- Light Violete Part -X-X-X-X-X-X-X-X-");

LightVioleteColor.click();

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

System.out.println();

System.out.println("-X-X-X-X-X-X-X-X- Green Part -X-X-X-X-X-X-X-X-");

GreenColor.click();

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

System.out.println();

System.out.println("-X-X-X-X-X-X-X-X- Brown Part -X-X-X-X-X-X-X-X-");

BrownColor.click();

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

}

}

**Q>How to find the number of browsers opened by selenium?**

Ans :- driver.getWindowHandles()

Q> how to use open new tab in existing browser (One network, media ocean, PTC)

Ans :-

driver.findElement(By.tagName("body")).sendKeys(Keys.CONTROL,"t");

or

driver.navigate().to(urlToOpen);

Actions action =newActions(driver);

// close the newly opened tab

action.keyDown(Keys.CONTROL).sendKeys("w").perform();

// switch back to original

action.keyDown(Keys.CONTROL).sendKeys("1").perform();

 Ctrl+t combination to open a new tab, Ctrl+w to close it, and to switch back to original tab I used Ctrl+1 (the first tab).

Q>Difference between click and clickandwait statements? (Sapient )

Ans :- click command only execute the command and then proceed to next command of the script.

"clickAndWait" command is some different than "click" command. It will wait for page to load means next command will be not executed until page not loaded successfully. You need to use this command when your   
are clicking somewhere on page of software web application and page start reloading content.

Q>How to run the selenium webdriver scripts in the background? Write a sample code to perform that action. (Accenture, CGI, Lloyds)

Ans :- To achieve this one has to use either HtmlUnitDriver or PhantomJS

**How do I know when a page has finished loading?**

Ans :- Selenium does this by default for us.

**how to find out xpath or view dom when test case is already running.**

Ans :- You will have to create firefoxprofile and add the firebug and firepath plugins into that.

**Framework Interview Questions**

**Question :- What framework you have used in your current Project? (HCL, Synechron, TCS, Capgemini, KPIT, TechM, Persistent, Zensar)**

Ans :- We have used TestNG as a unit testing framework with Data driven approach & design pattern used is Page Object Model

**Q> What is singleton design pattern? (GE, Avaya, Mindtree, HCL)**

Ans :-

in Java then you must have used new keyword. This new keyword create an Object of class whenever required. But there are some scenarios where you don’t want to create individual Object for different purpose.**Singleton Pattern** ensures that one and only one Object is instantiated for a given class. Whenever object of given class is required, only single(No more than one object) Object get returned

we will create a private Constructor and also a static method to create object of the same class.

class JBT {

private static JBT instance = null;

/\*

\* As private constructor is used so can not create object of this class

\* directly. Except by using static method of same class.

\*/

private JBT() {

}

/\*

\* This method will be used to get instance of JBT class. This method will

\* check if there is aready an object of class create or not if not then it

\* will create an Obect of JBT class and return the same else it will return

\* the existing Object.

\*/

static JBT createInstance() {

if (instance == null)

return new JBT();

else

return instance;

}

inti;

}

**Q>can you achieve Singleton design pattern using final? (NTT data, deutsche bank)**

Ans :- No

**Q>What did you keep in excel and .properties? (Amazon, CG)**

Ans :- In Excel we keep all the data, such as multiple combination of user names and password.

In properties file we keep xpaths, URL, default user name and password etc.

**Q>What are components of framework? (Media ocean, neeyamo, L &T)**

**Or**

**Question :- Give me few points about what is the use of Framework (jUnit or TestNg)**

Ans :-

1. Reusability of code
2. Maximum coverage
3. Recovery scenario
4. Low cost maintenance
5. Minimal manual intervention
6. Easy Reporting

**Q>How have you implemented OOPS in your project? Explain with examples (Tieto, Fundtech)**

Ans :- please refer to our designed framework

**Question :- Have you tested in different browsers? If yes how? (Xnet, Wipro, TechM)**

ANS: yes, this can be done in 2 ways,

1) we can pass the browser names in Frame Work it selfthrough CONFIG.Properties file and

2) By using Grid we can test our application in multiple Browsers at a time. This is called Parallel and First one is Sequential

**Question:- what are the issues you have faced while doing cross browser testing?**

Ans:- There is no issues with Firefox , But with IE and Chrome there is an CERTIFICATION ERROR's.

Means if we are attempting to test an https sites these errors will occer. So we have to override those errors with Desired Capabilities class

2) This is easy but you should write, the Chrome and IE is not independent browsers youknow these will come up with a driverexe file you have to download these driverexe file according urosbit(64/32),

if ur system is 32 you have to download 32 bit driverexefile.. and you have to specify the driver exe file Path in our program then only it will work, other wise it will display an error.

**Question:- Once a framework is developed will you do any changes to the framework?**

ANS: Yes. There could be few changes related to reports. There could be few enhancement or changes in methods. The structure is same.

**Question :- How frequently you execute your script? (In agile)**

ANS: Every Day Dry Runs will be there, in order to confirm that all will be working fine.

**Question:- what are the components of framework?**

Ans:- 1)General Frame work components: Properties files, configuration files, Data sheets, Testbase class, Utilities Class, Test cases, Reports, Logs

**Question:- how to read the property files**

Ans:-

**Question:- How to read the excel files which API have you used**

Ans:- Apache POI

**Note:**

Webdriver is as interface and we can’t create object of it.means can’t create constructor as well.

driver can handle one window at a time.

if after clicking any object element new window is opening than driver can not handle new window for that we need to switch to other one

Frames-kind of HTML document which are embeded on other HTML document

**How to capture color , height , width , font –size of the Element**

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

element.getAttribute("id");

element.getAttribute("name");

element.getCssValue("border");

element.getCssValue("font");

element.getCssValue("background");

**How to get the Location of the Webelement**

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

Point p=element.getLocation();

p.getX();

p.getY();

**Difference between Java Program and TestNg Script?**

Ans- When we execute Java program and TestNG script then functionality wise nothing will change because our script is performing the same functionality but using TestNG you will get some additional functionality.

Some benefit of TestNG Script:

1- For even single test case you will get 3 reports. These reports generated by TestNG

2-You can check execution time i.e. How much time test case has taken

3-Parallel execution

**How we can write TestNG Script?**

Ans- To execute TestNG script we don’t have to write the separate class. We can use simple java class but here we will not write public static void main(String []args) because we are not going to execute this from JVM.

TestNG works with Annotations and annotation can be represented by @ symbol

@Test- is this the main annotation from where TestRunner will start execution.

In other words, you can say @Test in entry point

**Can you tell about this , where the Test NG Program will run if not in JVM**

“we will not write public static void main(String []args) because we are not going to execute this from JVM.”

Annotations of TestNG are itself a class which contains the definition of main() method due to which annotations drives compilation forward.

**how to run only skipped test cases in selenium web driver?**

rerun test cases also can be run by testng-failed.xml file.

**I want to run some method in sequential order and some in parallel in testng.how would u configure the testng.xml?**

In this case you can create 2 xml file. One will run in parallel and one in sequence.

you can with single xml as well but for this-You can convert all script in testng.xml file and you can trigger direct from eclipse or ant or pom.

**Importance of Reporting in Selenium?**

1- Reports helps you to identify the status of the test case (Pass/Fail/Skip).

2- Using reports we calculate time taken by each test case that helps to calculate ROI(Return on Investment).

3- You can share automation reports with your team and clients as well to share the status of testing progress etc.

**Does selenium support report generation?**

Ans- No Selenium will only help you to automate your web application. If you want to generate reports then we can use Third party tools that we can integrate and can generate reports.

**Do we need to write some script or any additional code for reports?**

Ans- No we do not have to write any additional code for report generation. We need to refresh our project and we will get 1 additional folder (default suite) inside that folder we will get all the reports. but you can with ExtentReport with as well.

One good feature about TestNG for every test case it creates 3 different type of reports.

refresh and you will get 3 reports-

index.html

Default test.html

emailable-report.html

**How to create a report for Selenium Parallel Test execution.**

I have two test in testng.xml file. Each test calls 50methods(@Test).

One test runs in chrome browser and another one runs in firefox browser.

Which one is the best report generation tool and how can i achieve it.

In this case,You can continue with default testng report.

**What is Cross browser testing?**

Cross browser, testing refers to testing the application in multiple browsers like IE, Chrome, Firefox so that we can test our application effectively.

Example- Suppose if you have 20 test cases that you have to execute manually, so it is not a big deal right we can execute in 1 day or 2 days. However, if the same test cases you have to execute in five browsers it means 100 test cases then probably you will take one week or more than one week to do the same and it will be quite boring as well.

**Note-** Your script can fail due to so many reasons some of them are

1- Some locator has been changed in application because the application is getting new feature- so in this case you need to modify your script in other words you have to refine your script. You cannot avoid maintenance of test script you always have to maintain your scripts

2- Either functionality has been broken- in this case, you have to raise a defect and assign to the respective person.

Why test fail?

Application issue/down

Server is not responding

Network issue

Scripting issue

Environment issue

**How to run failed TC?**

test output-suitname-TestNG will generate testng-failed.xml- you can run this one only

OR

Consider you have 10 test cases in Selenium and you have created a testng.xml to execute all test cases.

Now you create another 20 test cases which belong to the same project, but they belong to different module so again we created testng1.xml for the same.

If the same process goes on, then at last you must be having couple of xml files which will be having all the testcases.

Now the real problem is if you want to execute all xml in one shot then you cannot do because Eclipse will allow to execute only one xml.

In this case we can take help of a TestNG class which allow us to execute multiple xml in one shot.

@Test

**public** **void** test() {

TestNG runner = **new** TestNG();

List<String> list = **new** ArrayList<String>();

list.add("testng.xml");

runner.setTestSuites(list);

runner.run();

}

**Note:** I got a case. For suppose I have 100 test cases, in that 100 I want to execute only 20 test cases. Is there any way to skip the 80 test cases, rather using enable=false, for all the 80 test cases. Grouping of test case will work for you.

Create group of test cases then decide which group to run which do not has to run via XML file(testng.xml).

Best and easy way to Group test cases in selenium

For Example- If we have 200 test case out of these some are end to end test cases, some are the functional test case and some are regression or smoke test cases so if you don’t categorize these test cases these all test case will come under one common category.

@Test(groups={"Smoke"})

@Test(groups={"group1","group2"})

After grouping in Selenium we can specify the include and exclude in testng.xml.

<include> – It tells testng.xml that which group we need to execute.

<exclude>- It tells testng.xml which group we have to Skip

public class TestGroupDemo {

@Test(groups={"Smoke"})

public void login(){

System.out.println("Login done");

System.out.println("Smoke Scenario passed");

}

@Test(groups={"Regression"})

public void register(){

System.out.println("Registration done");

}

}

<suite name="Suite" parallel="none">

<test name="Test">

<groups>

<run>

<include name="Smoke" />

<exclude name="Regression"/>

</run>

</groups>

<classes>

<class name="testngDemo.TestGroupDemo"/>

</classes>

</test> <!-- Test -->

</suite> <!-- Suite -->

**Can we group the different methods from different classes and run them together.**

Let suppose: We have 3 classes and in each class we have 3 methods. So can we group as: 2 methods from class 1, 2 method from class 2 and 1 methods from class 3?

Ans-Yes

**Benefits of Running parallel execution in selenium**

It saves execution effort.

We can cover a number of tests.

We can perform cross-browser testing as well which will make the application more stable.

If you are running scripts parallelly then it will help you to increase ROI (return of Investment)

We need to understand some topics before proceeding further.

TestNG internally handles threading concepts which will allow us to run the test in multiple threads.

Each thread will be assigned to individual test so if you have fewer threads than Test will get same thread if free.

You need to separate machine for parallel execution or machine with good resources which can handle multiple browsers at one time.

Thread count

You need to also consider thread-count in TestNG which will create threads based on our requirement.

I have some use case which will make your concept clearer.

Scenario 1- If you have three test cases and thread count is only 1 then the single thread will execute all test which is of no use.

Scenario 2- If you have three test cases and thread count is two then you will notice parallel execution because one thread will execute one test and one thread will execute two.

Scenario 3- If you have three test cases and thread count is three then each thread will execute an individual test and you will notice execution time will be less.

Listeners-allow to customize your Logs and report of TestNG

Listen to certain event and behave accordingly

ItestListner-class level, suite level

there are 5 ways of logging now i have learned so far from the site.

Can you please correct me if i am wrong somewhere or anything to be added. Its like a summary

1) Reporter.log

A simple print statement

2) testNG

The most easiest way since you just have to extend the interface and you get advance logging

3) WebDriverEventliStener

This a little complex since you have yo create 2 classes . One extending other class and use different object than driver but logging details are very good liked

you click on which link, you navigated where

4) Log4j

external library, it has 4 different things under it and has its own syntax and classes

5) Extent report