**Complex Websites: Travel , Shopping**

Selenium Features

* Selenium is open Source Automation Testing tool
* It is exclusively for Web Based applications.
* Selenium supports multiple browsers -   
  Chrome, Firefox, Internet Explorer, Safari
* Selenium works with Multiple Platforms   
  Windows, Apple OS X, Linux
* Selenium can be coded in multiple languages -   
   Java, C#, Python, Javascript, Python, php,Ruby

Selenium Webdriver



* After you trigger the Test, complete Selenium code (Client) which we have written will be converted to Json format
* Generated Json is sent to Browser Driver (Server) through http Protocol

Note: Each browser contains a separate browser driver

* . Browser drivers communicate with its respective browser and executes the commands by interpreting Json which It received on the browser.
* Browser Driver receives responses back from the browser and it sends Json response back to Client.

*Check if are on correct Website:*

*Validate title*

*Validate current URL*

*Some pages right click will be disabled like banking websites, so how to get Page Source?*

*Use driver.getPageSource()*

Go backwards in a website - driver.navigate().back()

Go forward in a website - driver.navigate().forward()

Driver.close Vs Driver.quit

Close closes current browser

Quit closes all open browsers opened by selenium

Locators Supported by WebDriver

ID,ClassName,Name,LinkText,Xpath,CSS

*ClassName : should not have spaces in class name .If this is the case you cannot use ClassName locator.although here you can use Xpath/CSS*

**Validate Xpath/CSS Accuracy:**

Always validate your xpath before you put it on your code

Go to browser console:

$x(“<you xpath>”) 🡪 XPATH

$(“<you xpath>”) 🡪 CSS

Generate your Xpath:

//tagname[@attribute=’value’]

Generate your CSS:

tagname[attribute=’value’]

or

tagname#id 🡪 use only when you have id

or

tagname.classname 🡪 use when you have classname

*You can skip tagname eg: #id*

*If your CSS selector is having multiple classes, then remove the spaces and add a . between each class*

*Eg:<a class=“ui-state-default ui-state-highlight”>*

*CSS Selector: .ui-state-defaul.ui-state-highlight*

Xpath and CSS using Regular Expressions:

//tagname[contains(@attribute,’value’)]

tagname[attribute\*=’value’]

Parent-Child Relationship Xpath

*If there’s a situation where your element cannot be distinctly idetified with the available attributes then you can use parent child relationship to go to desired element.*

You can go upwards in hierarchy until you can clearly identify parent.

Eg:

<parent xpath>/div/div[2]/input

Travel from parent to first child div.this child div has two child div.we want second child iv so we use div[2].this second child has the required input tag

Method-2:

With this method selenuim only searches child element in parents xpath scope.

<Parent Xpath><SPACE><Child Xpath>

SPECIAL CASE:

If there are 2 instances found of the desired xpath in browser console and you want to click on second instance then you can follow the below pattern.

(<XPATH\_OF\_DESIRED\_OBJECT>)[2]

Relative vs absolute xpath?

Basic difference is the way we traverse an element

Relative Xpath: *when we identify the desired element w/o help of its parents*

Absolute Xpath: *Requires the help of Parents*

Relative Xpath is more preferred over Absolute

*Traverse to Sibling from Sibling*

*Although you can use [2]/[3] to reach desired sibling from parent,this scenario is useful when attribute of parents is dynamic and one of the attribute of sibling is constant.So here you may need to traverse from sibling to sibling.*

*Very rarely occuring scenario.*

* Go till your sibling using your xpath then use

<sibling relative Xpath>/following-sibling::tagname[number]

*Parent to child using CSS:*

<Parent CSS Selector><space><tagname:nth-child(<number>)>

*Traverse to Parent from Child(Reverse Traverse)*

*This can be used when parent attributesare dynamic and one of the child’s attributes are static*

* <child relative Xpath>/parent::tagname[number]

*One major difference b/w Xpath and CSS is you cannot traverse in reverse using CSS.This is only possible using Xpath.*

*Identify Element based on Visible Text:*

This scenario will occur when code is not yet ready but on UI docs you see elements and the it Visible Text and you are asked to prepare automation Script

* //\*[text()=’<Your visible text including spaces>’]

Use this only in Extreme Cases!!

CSS Locators Identification

* CSS is 10 times faster than Xpath
* Class under class can be identified using CSS Selector only
* CSS is light-weight

Static Dropdowns-

* Dropdowns which have fixed number of values when clicked
* Handled using Select class

Dynamic Dropdowns-

The option in dropdown dynamically appear only when we click on dropdown.Until that instance no values for dropdown are present in html.

In these cases you use Xpath method 2 of parent-child relationship.

*Method-2:*

*With this method selenuim only searches child element in parents xpath scope.*

*Desired Xpath-> <Parent Xpath><SPACE><Child Xpath>*

Auto-Suggestive Dropdowns:

Based on your input , values appear on dropdown.

Moreover here keyboard events are important here.

Checkboxes:

Check if checkbox is selected or not 🡪 use isSelected() method

Count all checkboxes🡪 find out the common locators and prepare Xpath/CSS

Then use driver.findElements(<locator>).size()

*This method can be used for finding count of different elements present in a webpage.*

Calander UI:

isEnabled() not working in some websites

Workaround

Go to HTML, and check which attribute is changing when you enable/disable attribute.

Get that attribute using getAttribute() method and use if conditions.

**Java Alerts:**

Here you need to switch your driver

Use driver.switchTo().alert().accept() 🡪 Accepts the alert +ve outcome

Use driver.switchTo().alert().dismiss() 🡪 Cancels the alert -ve outcome

There are different methods for alerts, use can select the one you desire.

**Synchronization in Selenium:**

Impicit Wait 🡪 define wait time globally

Expiclit Wait 🡪 the wait is specific webelement

Thread.sleep 🡪 part of JAVA not Selenium, blindly will wait for time specified *(not preferred)*

Fluent Wait 🡪 explicit wait, with a difference that it checks element in DOM at specified intervals called polling

**Actions Class:**

Frames:

**Handling Calendar:**

Clicking on a particular date:

1. Find out how to unquely identify month
2. Find out common element in all calendar dates
3. Use findElements method to get all dates in a list
4. Iterate over list to get to desired date

**Handling Tables:**

Tables practice wesite:cricbuzz

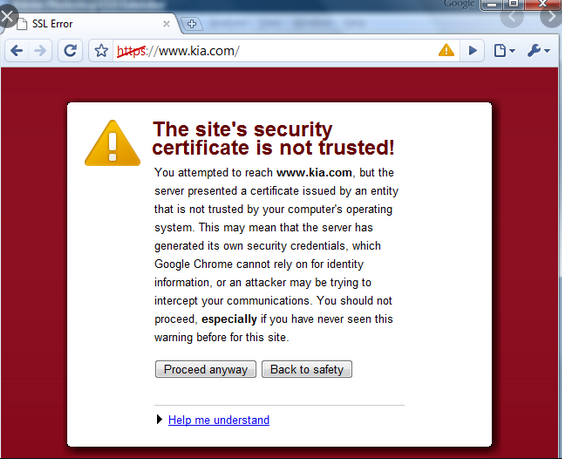
Parent child realtionship is important here

**Javascript Executor:**

Using Javascript DOM you can extract all hidden DOM,because Selenium cannot identify hidden elements

**Handling SSL Certifications:**

Error:



DesiredCapabilities ch= DesiredCapabilities.crome(); //general chrome profile

ch.acceptInsecureCerts();

ChromeOptions c = new ChromeOptions(); //local browser

c.merge(ch);

Wedriver driver = new ChromeDriver(c);

**Maximizing Browser and Deleting Cookies:**

driver.manage().window.maximize(); // maximizing browser

driver.manage().deleteAllCookies();

Frameworks

**Maven:**

Apache Maven is a software project management and build management tool for Java Frameworks.

1. Why Maven?

* Central repository to get dependencies
* Maintaining common structure across the organization
* Flexibility in Integrating with CI tools
* Plugins for Test framework execution.

Artifact: An artifact is a file, usually a JAR, that gets deployed to a Maven repository.

GroupId: groupId will identify your project uniquely across all projects

**Maven Commands In project folder on command line – sureFire Plugin:**

Maven Clean- mvn clean

Maven Compile- mvn compile

Maven Test – mvn test 🡪 ( by Default when you run test , clean and complie also runs)

**Executing testng.xml from Maven sureFire Plugin:**

Need to place xml file name in configuration tab inside sureFire Plugin dependencies in POM.xml

**Jenkins:**

CI Tool