Java Concepts:

1. Loop:
   a. For each
   b. Regular for loop
2. Constructor
3. Inheritance
4. Runtime Polymorphism
5. Upcasting and Down casting
6. Encapsulation:
   a. Private
   b. Getter and setter
7. String Class

a. == operator
b. Equals operator
c. equalsIgnoreCase

8. Arrays:
   a. 1D
   b. 2D
9. Collections:
   a. List
   b. Set
   c. Map
10. Exception
    a. Checked and Unchecked
    b. Throw, Throws and Throwable
    c. Try and Catch

d. Final, Finally and Finalize

11. File handling:

a. File
b. FileInputStream
c. FileOutputStream

12. Thread:

a. Thread.sleep(ms)

Introduction:

1. **What is Automation?**

   Ans: The process of converting manual testcases into automation test scripts is called Automation.

   Or

   When one Software is testing another software in order to check whether actual and expected results are matching or not, then this process is called Automation.

2. **When do we go for Automation?**

   Ans:

   a. When there are regression testing in scenarios, instead of executing old manual test cases, we will write automation scripts and we will do Automation Testing using automation tool.

   Note:

   i. Manual test cases are based documents for automation testing.

   ii. We will start automation only after the build is stable.

   b. High quality of product is to be delivered.

   c. In order to save the execution time.

   d. Competition in the market.

   e. Long-term project.

   f. More repetitive tasks.

3. **What is an Automation Script?**

   Ans: It is a program which is written against the manual test case and it includes verifications or assertions.

   It can be written in any programming language provided that the automation tool supports that programming language.

4. **What is an Automation Tool?**

   Ans: It is a software through which we can write the automation script and can run those on the Application Under Test (AUT) or Software Under Test (SUT).

5. **What are the advantages of Automation Testing?**

Ans: The advantages to Automation Testing are as follows:

a. Faster Execution.
b. Accuracy of the result is more.
c. We can reduce cost.

6. **What are the disadvantages of Automation Testing?**

Ans: The disadvantages to Automation Testing are as follows:

i. **We cannot convert all the regression scenarios into Automation Scripts.**
   a. OTP related test cases.
   b. Captcha related test cases.
   c. Audio related test cases.
   d. Screen Transition
   e. Network Errors
ii. **Programming skills are required.**
iii. **We cannot perform ADHOC testing.**
iv. **Small changes in the UI may have a big impact.**

7. **What are the different Automation tools available in the market?**

Ans:

I. **Functional Testing Tools: UI Testing**
   a. Selenium
   b. QTP: Quick Test Professional
   c. TestComplete
   d. SAHI
II. **Performance Testing Tools: Stress or Load Testing**
   a. JMeter
   b. Load Runner
III. **Back-end Testing Tool**
   a. Rest API
   b. Postman
   c. Rest Assured

Selenium:

It is an Open-Source test automation tool which is used to test Web Application and Mobile Application.

Open Source: Free to use and modify

Test Automation Tool: A Software which is used to test another software.

Web Application: Application which is opened through web browser via internet.

Mobile Application: Application which is designed to run on mobile devises such as smartphones or tablet computer.

Note:

1. ***To test Web Application, we should use Selenium WebDriver.***
2. ***To Test Mobile Application, we should use Appium.***
3. ***Selenium tool is not having any user interface or it can be said Selenium is a headless tool.***

1. **What are the advantages of Selenium Tool?**

Ans:

1. ***Open source***: Selenium is an Open source which means it is free to download and used for both professional and business purposes.

   It also means the source code is open for

a. Viewing (that's why we can easily debug).
b. Customization (that's why we can enhance this tool).
c. Distribution (that's why we can distribute our enhancement done to the tool).
d. Integrate with 3rd party tool if required.

2. ***Multiple Programming Language***: Selenium supports multiple programming language. As a result:

a. It gives flexibility for various language developers to use the selenium tool i.e., they are not required to learn any other language, just with the language developers already knows is enough to use selenium tool.
b. It gives flexibility to choose the development language as the automation tool language as well.

   This provides an opportunity for the automation testers to take help from developer if there are any critical coding issues.

c. Selenium supports 11+ programming language:

i. Java
ii. C#
iii. Python
iv. Java Script
v. PHP

vi. Perl
vii. Ruby
viii. Haskell
ix. Dart
x. Go
xi. Kotlin

3. ***Multiple OS:*** Selenium supports multiple OS. It supports all the major OS's like:

i. Windows
ii. Mac
iii. Linux

It helps us to perform OS specific browser compatibility testing in desktop web apps.

Note: Appium also supports Android and iOS mobile operating system. So that in Appium we can write some common automation scripts for both OS.

4. ***Multiple Browser:*** Selenium supports multiple browsers. It supports 6 major browsers such as:

i. Chrome
ii. Opera
iii. Firefox
iv. IE
v. MS Edge
vi. Safari

And 2 headless/transparent/GUI less browsers, which are:

a. PhantomJS
b. HTMLunit

This helps to perform browser compatibility testing.

Therefore, selenium helps us to perform cross browser testing or browser compatibility testing (mandatorily) in all the browser, since customer may use any of the browser to open the application.

Note: Appium supports chrome mobile web browser testing on android and safari web browser testing on iOS.

2. **What is the disadvantages of selenium webdriver?**
   Ans: The disadvantage of selenium is that we can automate only web applications and mobile applications and not stand alone application.

3. **What are the differences between Selenium and QTP?**

   Ans:

ja	**QTP(Quick Test Professional)**
. Open Source	Licensed
. It supports all the major OS's like Windows, Mac and Linux	It supports only windows
. It supports 6 major browsers.	IE and MS Edge
. Selenium supports 11+ programming languages	VB scripting and Java Script
. we can automate only web applications and mobile applications.	All kinds of app

4. What are the different flavors of selenium?
   Ans:

i. Selenium Core
ii. Selenium RC (Remote Control)
iii. Selenium IDE → Record and Playback tool
iv. Selenium WebDriver (3.141.59→ 2018)
v. Selenium Grid→Runs test on remote machines.
vi. Appium

Selenium Core[2004]: It is the first version of Selenium and supported only Java programming language.
There was no IDE supported for Selenium core so the script Where are getting executed in command prompt.

Selenium RC[2006]: It is the next version of Selenium which came into the market in 2006.

It was supporting few more languages like Java Ruby Python etc.
There was proxy server issue in Selenium RC.

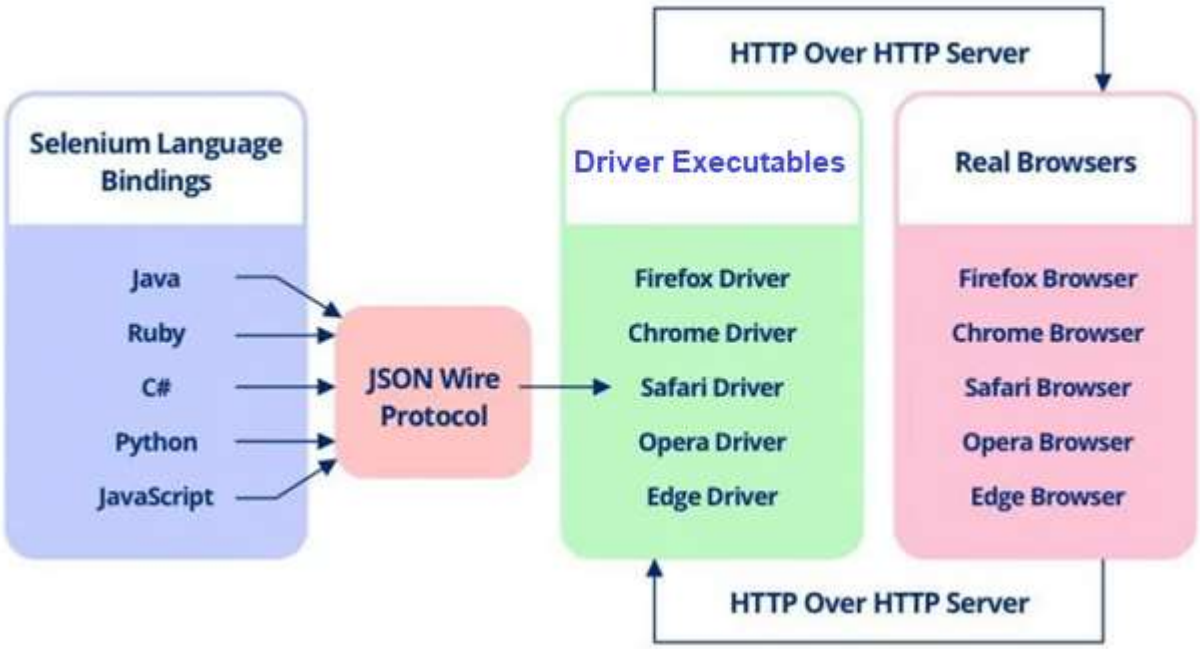
Selenium IDE: It was a record and playback tool.
It was supporting only Chrome and Firefox.
Debugging the code is difficult in Selenium IDE.

Selenium WebDriver: To the market in 2007.
It supports all the programming language and all the browser.

Selenium Grid: It is used to run the test in remote machines.

Appium: This is used to test mobile applications.

Selenium WebDriver Architecture in General:



5. Explain the Selenium WebDriver Architecture.
Or
Explain how Selenium WebDriver works.

Ans: Selenium WebDriver architecture contains majorly 4 components:

i. Selenium Client Bindings/Language Bindings
ii. JSON Wire Protocol
iii. Driver Executable Software
iv. Browser and AUT.

Client Bindings are the libraries provided by the selenium developers with respect to the different programming language selenium supports. For Java we have a client binding called **selenium-server-standalone.jar** file. Client bindings are also called as language bindings or client libraries.

JSON Wire Protocol is a way through which the client (automation script) can communicate with the server. Automation Script will be converted into standard protocol i.e., JSON Wire Protocol which can be understood by the driver executables software and based on that automation will be performed.

Here JSON stands for Java Script Object Notation which is a standard file format which is used for data exchange between client and server or between two application which may be developed using same or different technology.

Before JSON, XML was used more for data exchange.

JSON is light weight when compared with XML.

JSON is full of key value pairs. It is called JSON because it uses the java script object creation syntax.

A Sample XML code with tag:

<os>windows</os>

<name>Akash</name>

A Sample JSON code with tag:

{

os : “windows”,

name : “Akash”

}

Driver Executable Software are the software which are provided by browser developers to support automation in their own respective browser.

In every driver executable software there are 2 components:

i. **Server:** to receive the request from the client (automation script) in the form of JSON wire protocol, process it and perform requested action on the AUT on browser.
ii. **Browser native API’s**: These are browser specific functionalities to perform automation on their own browser.

6. How to configure selenium WebDriver in eclipse IDE?

i. JDK 1.8 and above
ii. Eclipse photon and above 2020 or 2021
iii. Create a java project in eclipse
iv. Create a package
v. Create “hello world” program and run the program.

vi. Create 2 folders by name
   a. Drivers: To store driver executable software.
   b. Jars: To store all the .jar files inside the project.
vii. Copy paste all the driver executables software's and selenium jar files into the drivers and jars folder respectively.
viii. Right click on selenium.jar file
ix. Click on Build path
x. Click on Add to Build path.

7. Steps to download Java Client Bindings:

i. Go to www.selenium.dev
ii. Click on the download link
iii. Click on latest stable version

8. Steps to download driver executable software for Google Chrome:

i. Check the browser version
ii. Go to https://www.selenium.dev/downloads/
iii. Scroll down to browser section and expand it.
iv. Click on documentation link under Chrome
v. Click on latest stable version
vi. Click chromedriver.win32.zip
vii. Extract the content of the file
viii. Find a file chromedriver.exe

9. Steps to download driver executable software for firefox:

Assignment

10. Steps to download driver executable software for IE:

Assignment

11. Launching browser in Selenium WebDriver

Ans: To launch the empty web browser in Selenium WebDriver we should perform Two Steps:

i. We should set the driver executable paqath
ii. We should create the object of browser class

I. How to set the driver executable path?

Ans: We need to use browser specific key and the driver executable path in a method called setProperty().

Selenium developers have provided standard set of keys for all the 6 browser Selenium supports. We should use only those keys. Keys are case sensitive and we must use the lowercase letters only.

They are:

1. Chrome→ "webdriver.chrome.driver"
2. Firefox→ "webdriver.gecko.driver"
3. IE→ "webdriver.ie.driver"
4. Edge→"webdriver.edge.driver"
5. Opera→"webdriver.opera.driver"
6. Safari: → no need of any key, just we need to select "allow remote automation" in the develop menu of safari browser.

We can use relative path or absolute path while providing the driver executable path

Relative path means the path which starts from current project.

Absolute path means the path which starts from the drive (C://...)

The standard practice for load is always using relative path that is from the project because project portability will be easy.

setProperty() method is a static method of system class. It takes key-value pairs as input in the string form. **The Key represents which browser to be launched and value represent the driver executable path for the browser.**

Exampleusage: System.setProperty("webdriver.chrome.driver", "./drivers/chromedriver.exe");

Note: If there is any mistake while pass setting then you will get exception called **Illegal StateException** with the proper error message.

Illegal state exception is one of the unchecked exception of Selenium, which we encounter when the path of the driver executable in incorrect or driver executable is not present in the path provided.

We May also encounter this exception if the key of setProperty method is not correct.

II. Creating object of browser class:

For all the six browsers which Selenium supports, there are 6 browsers related concrete class which can be used for launching the browsers. They are:

i. Chrome driver
ii. Firefox driver
iii. Opera driver
iv. Internet Explorer driver
v. Edge driver
vi. Safari driver

Before using these classes we should import them from Selenium built in packages **org.openqa.selenium.browsername.browserClassName.**

org is the domain

openqa is the main branch for all the open-source software

selenium is the project name under open QA

Browser name like **chrome** is the module name

Browser class name like **ChromeDriver** is the class name.

Example usage: Chrome driver = new ChromeDriver();

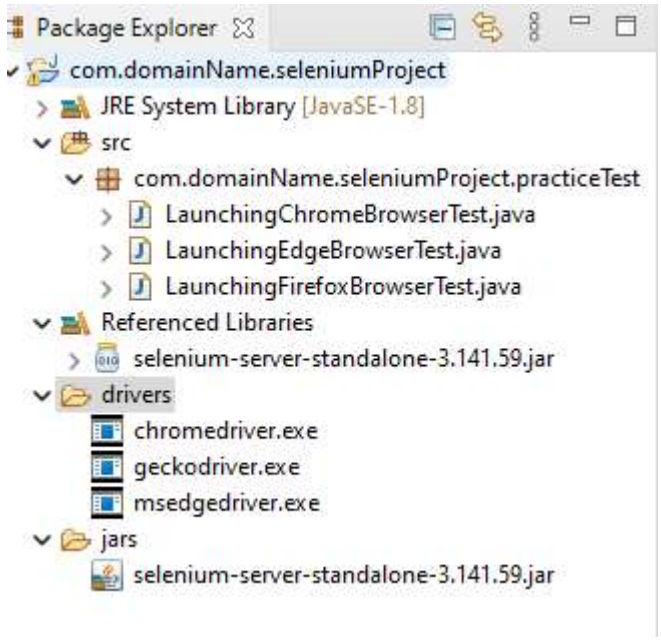
Whenever we create object for any browser related class in Selenium WebDriver the constructor of the browser related classes will perform two important operation they are:

1. Start the driver executable software or starts the server of the browser

2. Launch the empty Browser

In selenium webdriver creating the object of a browser related class means we are launching the empty Browser

Java Project Snippet:



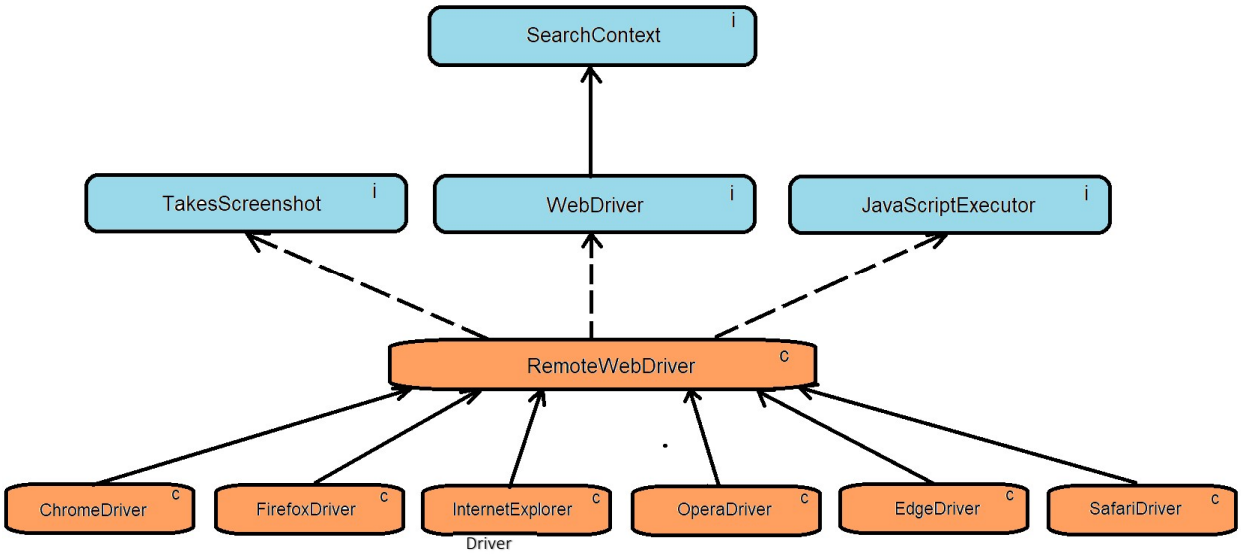
Selenium WebDriver Architecture w.r.t. Java Language Binding:

Or

Explain the inheritance hierarchy of Selenium Hierarchy?

Or

Explain Selenium WebDriver Hierarchy w.r.t. java client bindings?



In this hierarchy there are four major interfaces:

1. SearchContext

2. TakesScreenshot

3. WebDriver

4. JavaScriptExecutor

And here we also have six browser related classes along with one major implementation class called RemoteWebDriver.

1. SearchContext: It is an interface which provides a way to search the elements in the webpage. Generally, before performing any actions on the elements in the web page we should first search it using the methods of SearchContext interface.

It provides two abstract methods, they are:

i. findElement() [Pascal case]: It is used to find single element in a web page.

ii. findElements(): It is used to find multiple elements in a web page.

2. TakesScreenshot: It is an interface which provides a way to take the screenshot in Selenium WebDriver. We can take two types of screenshot:

i. Single element screenshot.

ii. Entire web page screenshot.

It provides one abstract method known as getScreenShotAs().

3. JavaScriptExecutor: It is an interface which provides a way to write JavaScript (Live Script) programs in selenium WebDriver. There are two types of JavaScript one is synchronous JavaScript and another one is asynchronous JavaScript.

We can write both the types of JavaScript programs.

This interface provides two abstract method,

i. executeScript(): This method is used to run synchronous JavaScript.

ii. executeAsyncScript(): This method is used to run asynchronous JavaScript.

4. RemoteWebDriver: It is a concrete class of Selenium WebDriver and it is known to be one of the major implementation classes because it implements all these above major interfaces and also many other interfaces as well.

5. WebDriver: It is the core interface of Selenium WebDriver which is used to control the web browser, like maximizing the browser perform browser history navigations or navigating to URL of the application under test(AUT) and many more.

Without this interface method we can't start automation. So that's one of the reasons the name Selenium WebDriver.

It provides 11 browsers controlling abstract methods they are:

i. get()
ii. getTitle()
iii. getCurrentUrl()
iv. getPageSource()
v. close()
vi. quit()
vii. navigate()
viii. manage()
ix. getWindowHandles()
x. getWindowHandle()
xi. switchTo()

Note: +2 methods from searchContext, they are findElement() and findElements().

WebDriver Methods in detail:

1. get():

- This method is used to navigate to the main URL of the Application under Test(AUT).
- This method takes the URL as the input in the string format.
- The URL should be fully qualified (fully qualified URL main URL with the protocol i.e. http, https file etc.,.
- This method will wait for the entire page to load.
- The return type of this method is void

Signature: Public void get(string URL){

(java instructions)

}

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

Note: Main URL means you are in which is used to launch the AUT.

Q. What if protocol is not mentioned in get()?

Ans: Throw **InvalidArgumentException**. It is one of the unchecked exception of Selenium.

2. getCurrentUrl():

- This method will give the URL of the current webpage.
- This method is used to verify the URL of the webpage.
- This method returns the URL in the form of string.
- This method doesn't take any argument.

Signature: public String getCurrentUrl(){

(java instructions)

}

Usage: String actualUrl= driver.getCurrentUrl();

3. getTitle():

- This method will give the title of the current webpage.
- To verify the web page title, we can use this method.
- This method returns the title in the form of string.
- This method doesn't accept any argument.

Signature: Public string getTitle(){

}

Usage: String actualTitle= driver.getTitle();

Page Source: Any web page will have a backend source code

The major building block of any web page in browser are:

a. HTML

b. CSS

c. JavaScript.

a. HTML: It stands for hypertext markup language which is used to display the data in the structured format.

b. CSS: It stands for cascading style sheets which is used to decorate the data displayed in the browser.

c. JavaScript: It is used to perform client side validation.

Client side validation means verifying whether the formats are correct or not before sending the request to the web server. But doing so we can avoid invalid request to server.

(JavaScript or LiveScript was developed by Brendan Eich at Netscape.)

Whenever a request is sent to the web server then a web server will respond with the three building blocks i.e., HTML, CSS, JavaScript.

4. **getPageSource():**
   - This method is used to get the source of the current web page.
   - This method is used to verify whether a current web page contains certain information or not.
   - This method returns the source code in the string format.
   - This method doesn't accept any argument.

Signature: public String getPageSource(){

}

Usage: String source= driver.getPageSource();

5. **close():**
   - This method is used to close the current Browser.
   - Initially as the driver control will be in the parent Browser window, this method will close the parent Browser.
   - Generally, it closes the browser window whenever the driver control is available.
   - At a time it can close only one browser window.
   - This method takes no argument.
   - The return type is void.

Signature: public void close(){

}

Usage: driver.close();

Note: In a group of Windows or tabs during automation, by default the driver control will be in parent browser window, which means the window from which we launch the main URL of AUT.

6. **quit():**

- This method is used to close all the browser windows (both parent and child browser windows).
- After closing all the browser Windows it will stop the server(driver executable software).
- This method receives no argument and the return type is void.
- This method is best to use quit() method in post condition instead of close() method.

Signature: public void quit(){

}

Usage: driver.quit();

Q. What is the difference between close() and quit() method?

close()	quit()
. It will close the current Window	It will close all the browser Windows
. It will not stop the server	It will stop the server
. It is not used in the post condition	It is always used in the post condition
. It can close only one browser at a time	It will close multiple browser Windows

7. **navigate():**

- This method is used to perform browser history navigation like back, forward and perform browser refresh and also it is used to navigate to the sub-URL of the application.
- This method returns Navigation type of object.

Navigation is an interface and it contains four abstract methods which are implemented in the RemoteNavigation concrete class.

navigate() method internally creates the object of RemoteNavigation concrete class and return it in the form of Navigation interface(upcasting while returning).

a. **back():** What is the void method which is used to navigate back to the previous visited page.
b. **forward():** It is the void method which is used to navigate to the next page which is stored in the history.
c. **refresh():** It is used to refresh the webpage.
d. **to(String URL):** It is used to navigate to the sub URL of the application under test as a standard practice.
e. **to(URL url):** It is also used to navigate to the server URL of the web page Under test but the argument with it takes is of URL type.

We need to create the object of URL and pass the URL in the parameter of the constructor, and then we have to use the reference variable as a parameter to the to() method.

```
URL url = new URL("https://www.google.com/")

driver.to(url);
```

Signature: public Navigation navaigate(){

}

Usage 1: Navigation nav = driver.navaigate();

```
nav.back();

nav.forward();

nav.refresh();

nav.to("URL");
```

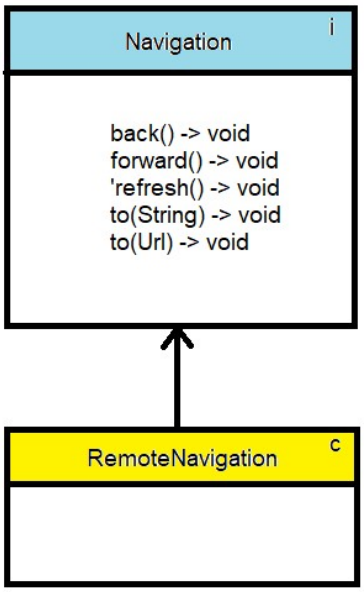
Usage 2: driver.navigate().refresh();

Note: There is no difference between get() method and to() method, both of them are same in terms of internal logics. However, the usage is different. In industries as a standard practice get() method is used to navigate to the main URL and the to() method is used to navigate to sub-URL.

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

get()	navigate()
1. Is used to navigate to the main URL	It is used to navigate to the sub-URL and along with that we can perform browser history navigation
2. Return type is void	Return type is navigation interface
3. Standard practice it is used to navigate to main URL of application under test (AUT)	Standard practice it is used to navigate to sub URL of application under test (AUT)

Inheritance Hierarchy of Navigation Interface:



8. **manage():**

- This method is used to perform:

a. Window related operations

b. Script Timeout related operation.

c. Cookie related operation.

a. To perform **window related operations** we need to use driver.manage().window() method.

This window() method will return the **Window** type of object.

Window is an interface that contains 6 abstract methods to perform 6 window-related operations. They are:

i. maximize(): It is used to maximize the browser window.

ii. fullScreen(): It is used to open the browser window in full screen.

iii. getSize(): It is used to get the current dimensions of the current browser window.

This method returns **Dimension** type of object.

Dimension is a concrete class of Selenium package this object contains two important methods which are: getWidth() and getHeight().

These methods provide the width and the height of the current browser window.

iv. setSize(): This method is used to modify the dimensions of the current browser window.

It receives dimension class object as argument so we should pass dimension class object with width and height to it.

Return type is void.

v. getPosition(): This method is used to get the current coordinates of the browser window.

It returns Point type of object.

Point in the concrete class of Selenium package.

It has two important methods which are getX() and getY() methods which returns the X and Y coordinates of the browser window.

vi. setPosition(): This method is used to move the browser window to a new

X and Y coordinates.

It receives point class obsssss as input.

We should create a Point class object with new X and Y coordinates value and supply it as argument so that the browser moves to that target co-ordinates.

The return type is void.

Negative value of x and y coordinates will move the browser window left and up respectively to the given pixel.

Usage: Window wind = driver.manage().window();

wind.maximize();

wind.fullscreen();

Dimension currentDim = wind.getSize();

int width = currentDim.getWidth();

int height= currentDim.getHeight();

Point currentPosition = wind.getSize();

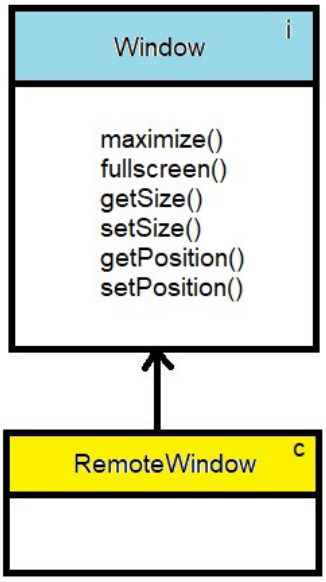
int startX= currentPosition.getX();

int startY= currentPosition.getY();

wind.setSize(new Dimension(500, 600));

wind.setPosition(new Point(200, 70));

Inheritance Hierarchy of Window Interface:



Notes:

As a standard practice we should always maximize the browser after launching it using maximize() method, as lot of features might not be visible when the window is not maximized.

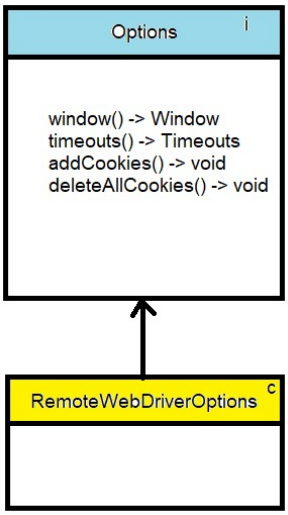
There is no method to **minimize** the browser (in selenium 3.141.59), instead we can use some keywords shortcuts which is present in **Robot Class**

Whenever we want test the applications feature in various different dimensions then these window related methods are used.

Q. What is return type of manage() method?

Ans: Options interface type of object.

Inheritance Hierarchy of Options Interface:



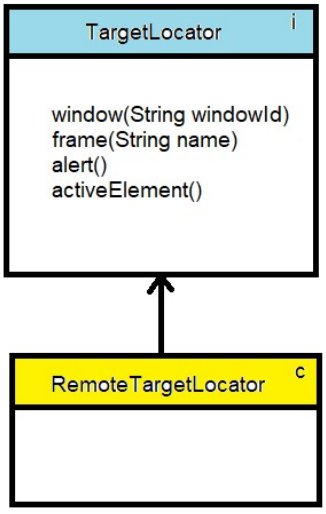
9. **switchTo():** This methos is used to transfer the driver control to target areas like:

a. Browser Window
b. Frame
c. Alert Pop-Up
d. Active Element

This method doesn't take any argument.

It returns **TargetLocator** interface type of object.

Inheritance Hierarchy of TargetLocator interface:



10. getWindowHandle(): This method is used to get the current window-Id.

While calling getWindowHandle() method wherever the driver control is available that window id is called as the **current window Id.**

As initially the driver control will be in the parent window, if we call this method the initially it will return the parent window Id.

The return type of this method is String.

This method receives no argument.

Signature: public String getWindowHandle(){
}

Usage: String currentWindowId = driver.getWindowHandle();

NoSuchWindowException: It is one of the unchecked exception of Selenium which is thrown when we are trying to perform automation on the browser window which is already closed and was having the driver control.

NoSuchSession Exception: It is one of the unchecked exception of selenium which will be thrown if the selenium server has already been stopped or not started at all and we are trying to perform automation.

Generally, this occurs if we have terminated the session using quit() method and after that we are trying to perform automation.

11.getWindowHandles(): Any Web browser will have a unique id which is called as window id or windowhandle.

Whenever we are performing automation with group of windows then by default the driver control will be in the parent browser window. So, to perform any automation on other browser windows we need to transfer the driver control to the desired browser window and only then we can perform automation on it.

To transfer the driver control we need to know the window id of the target browser window.

To capture the window ids for all the browser currently opened from selenium server (driver executable software) we can use this getWindowHandles() method.

getWindowHandles() will return all the window-id's in the Set<String> form.

To perform action on a particular window we should follow 3 steps:

a. Capture the Window-Id.
b. Transfer the control to browser using window Id.
c. Perform Automation on it.

Signature: public Set<String> getWindowHandles(){

}

Usage: Set <String> allWindowIds = driver.getWindowHandles();

Q. How to perform cross browser testing in Selenium WebDriver?

Ans: Testing the application on multiple browser is called as Cross Browser Testing.

It is also called **browser compatibility testing**.

There are two ways to achieve cross browser testing in selenium Webdriver with respect to java language bindings.

1. Implementing Runtime Polymorphism concept of Java.
2. By the use of third party tool called TestNG (Test Next Generation).

Implementing Runtime Polymorphism concept of Java.

```
public class ClassA {
public class DriverClass {

    public static void main(String[] args) {

        System.setProperty("webdriver.chrome.driver", "./drivers/chromedriver.exe");
        System.setProperty("webdriver.gecko.driver", "./drivers/geckodriver.exe");
        System.setProperty("webdriver.edge.driver", "./drivers/msedgedriver.exe");

        ClassA.test(new ChromeDriver());
        System.out.println("Application has been tested on chrome browser");
        ClassA.test(new FirefoxDriver());
        System.out.println("Application has been tested on firefox browser");
        ClassA.test(new EdgeDriver());
        System.out.println("Application has been tested on edge browser");

    }

}
```

Here, in the above program in ClassA the WebDriver (driver) reference does not know which browser to launch. In DriverClass when we call the static method test() from ClassA and pass the object creation statement as a parameter to the test() method, then the driver reference come to know which particular browser to launch.

This is an example of cross browser testing using runtime polymorphism.

Q. Explain WebDriver driver = new ChromeDriver();

Ans: This statement is to used launch the chrome browser. This is an **Up-casting** statement wherein ChromeDriver object is up-casted to WebDriver interface type.

Here, **WebDriver** is an interface.

driver is the reference variable.

= is the assignment operator.

new is a keyword

ChromeDriver() is the constructor which will start the chrome server and launches the empty browser.

Q. Why we have to up-cast browser object to WebDriver reference only, and why not any other interface or why not RemoteWebDriver class or why not store the browser object to the same browser reference?

Or

Why we write WebDriver driver = new ChromeDriver;

1. It is the standard recommendation given by selenium developers because according to Object Oriented Analysis and Design (OOAD) principle, we should always program to interface
   Program to interface means we should store the object of the class to utmost possible interface reference.
2. We will not up-cast driver classes to RemoteWebDriver class due to this OOAD principle.
3. In Selenium WebDriver, WebDriver interface is the utmost possible interface because all the major browser controlling methods available in it, without which we cannot start automation.
   For example without get() method we cannot navigate to the main URL of AUT.

In other words if we up-cast the browser object to other interfaces like SearchContext, TakesScreenshot or JavaScriptExecutor, then we will not have the access to the major browser controlling methods like get(), navigate() etc..
So we have to cast the browser object to WebDriver interface only.

4. We don't store the browser object into the same browser reference because that is again against the OOAD principle.

HTML:

- HTML stands for Hyper Text Mark-up Language.
- It is used to display data in the browser in the structured format.
- It is nested language (It is in the tree- structure).
- It is not case-sensitive (It is case insensitive).
- All html programs should be stored as .html as an extension
  For example: FileName.html
- All the program elements are known as tags.
- The tag will have the symbol < >.
- The root or the main tag of html is <html>
- <html> tag is having 2 child tags

a. <head>
b. <body>

1. **What is a tag?**
   Ans: The program element which is having the symbol < > is called a tag.

2. **What is a tagName?**
   Ans: The first word within a tag is called as tagName.

   <a href = “https://www.google.com/>
   <img src=”https://static.xx.fbcdn.net/rsrc.php/y8/r/dF5SId3UHWd.svg”>

3. **What is an attribute?**
   Ans: The key value pair in a tag is called as attribute.

   <a href = “https://www.google.com/>
   Note: One tag can have zero or more attribute.

4. **What is an Attribute name?**
   Ans: The key of an attribute is called as attribute name.
   <input type = “text”>
   <button type = “submit”>
   Here type is the Attribute name.

5. **What is an Attribute value?**
   Ans: The value of an attribute is called as attribute value.

<input type = “text”>
<button type = “submit”>
Here “text” and “submit” is the attribute value.
Note: Attribute values are not mandatory.

6. **What is a text?**
   Ans: Anything which is between opening and closing tag is called as text.
   Google
   Selenium

7. **What is link text?**
   Ans: Any text between anchor tags is called as link text.
   Google

Sample Html Code:

```
<html>
<head>
        <title> Selenium Demo </title>
</head>
<body>
        <h1> Demo Page For HTML </h1>
        <span>Username</span>
        <input type = "text" id = "123" name = "abhishek"><br>
        <span>Password</span>
        <input type = "text" id = "345" name = "arun"><br>
        <input type = "checkbox">
        <span> Remember Me </span><br>
        <input type = "radio">
        <span> Male </span>
        <input type = "radio">
        <span> Female </span>
        <input type = "radio">
        <span> Others </span><br>
        <button type = "submit">Login</button>
        <a href = "https://www.qspiders.com/"> Goto Qspiders </a>
</body>
</html>
```

WebElement:

Web-Element with respect to Browser: Anything which is displayed in webpage is called as Web-Element, such as:

1. Button
2. Checkbox
3. RadioButton
4. Link
5. Images
6. Drop-down menu
7. Drop-down Listbox
8. Text Field
9. Password Field
10. Test Areas

WebElement with respect to OOPS: WebElement is an interface provided by selenium which has different abstract methods to perform actions on the web-element in the webpage or to retrieve some information of the WebElement in webpage.

WebElement interface has 14 abstract method through which we can perform actions or retrieve information from the desired web-element of the webpage.

This WebElement interface is implemented by a concrete class called RemoteWebElement.

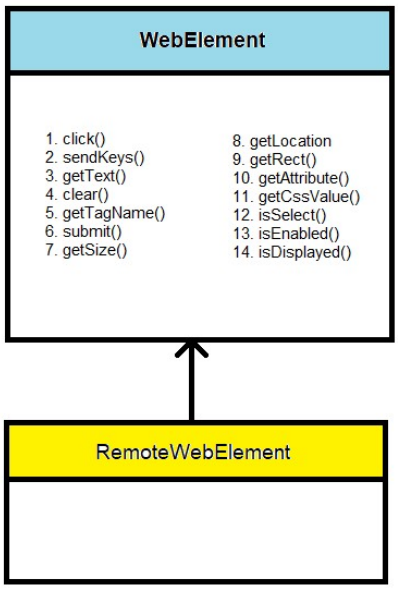
Whenever we need to perform any action on any web-element we should first locate the web-element using findElement() method by providing the appropriate locator strategy and then use the WebElement interface methods to perform actions like click(), clear(), sendKeys() etc.

WebElement Methods:

1. click()
2. sendKeys()
3. getText()s
4. clear()
5. getTagName()
6. submit()
7. getSize()
8. getLocation()

9. getRect()
10. getAttribute()
11. getCssValue
12. isSelect()
13. isEnabled()
14. isDisplayed()

WebElement Hierarchy:



How to perform any actions on the Web-Elements or How to retrieve any information of Web-Element?

Ans:

1. Find the element or search the element.
2. Perform the action on it or retrieve the information from it.

Locators:

Locators are the search criteria through which we can identify the target web-element or web-elements from the webpage to perform some actions.

Selenium supports 8 locator strategies which are:

1. Id
2. Name

3. Class
4. Link text
5. Partial link text
6. Tag name
7. Css selector
8. Xpath

Selenium tool provides 8 locator methods to identify the desired elements in the webpage, which are:

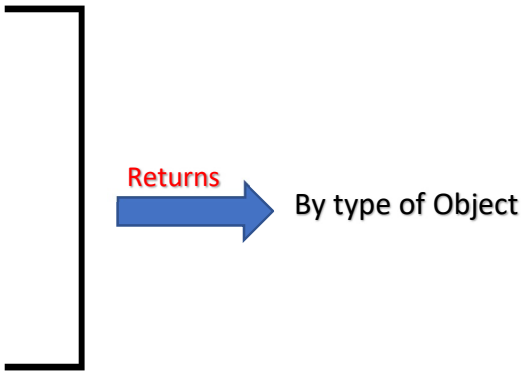
1. id()
2. name()
3. classname()
4. linkText()
5. partialLinkText()
6. tagName()
7. cssSelector()
8. xpath()

By:

- By is an abstract class in Selenium
- It provides eight methods to search the element in the web page.
- All these eight methods are called as locator methods.
- All these methods are Static.
- The argument given for this locator methods are called as **selector**.
- These located methods are used as an argument to findElement() and findElement() methods.

The 8 locators methods are as follows:

1. By.id("id attribute value")
2. By.name("name attribute value")
3. By.className("class attribute value")
4. By.linkText("full link text")
5. By.partialLinkText("partial or full link text")
6. By.tagName("tag name")
7. By.cssSelector("css expression")
8. By.xpath("xpath expression")



Q. What is findElement()?

- It is one of the methods of SearchContext interface which is used to find the element in the webpage so that we can perform some action on it.
- The argument of findElement() method can be any one of the locator method of By class. Based on the given locator strategy this method will identify the web elements and return the address of it.
- The return type of findElement() is WebElement interface type of object.
- This method will search for the element from the beginning of the HTML code till it find the desired element.
- If the given locator strategy is matching with multiple elements, then it will return the first matching element.
- If the given locator strategy is not matching with any element in the webpage, then it will through NoSuchElementException.

Signature: public WebElement findElement(By by){
}
Usage: driver.findElement(By.id(loginButton")).click();

Sample html code with duplicate elements:

```
<html>
<head>
<title>DemoSite</title>
</head>
<body>
<div>
<input type="text" id="abc">
<input type="text" id="abc">
</div>
<div>
<input type="text" id="abc">
<input type="text" id="abc">
</div>
</body>
</html>
```

CSS:
CSS stands for cascading style sheets. Which is used to decorate or apply some styling information to the web elements.
For example: Is used to add background color for a paragraph of to add mouse hovering in color of a button etc.

CSS Selector: It is a query language used in CSS to identify single element or group of elements to apply styling information.
It is generally written in the form of expression.

Why we should use CSS selector in Selenium?
Whenever the source code of the target element is not having ID name and class attribute and if the element is not a link then we will use CSS selector.
Another main reason of using CSS Selector is to identify Shadow Dom element.
Whenever the ID name and class a duplicate then also we can use CSS selector.

Syntax: tagName[attributeName = "attributeValue"]

Example:
```
driver.findElement(By.cssSelector("input[title=\"Search\"]"
)).sendKeys("TestYantra");
```

Q. Can we use id, name and class attribute in the CSS selector expression?
Ans: Yes, we can use any attribute name.

Shortcuts to use id and class attribute name:
a. id→#
   Syntax: tagName#nameAttributeValue
b. class→.
   Syntax: tagName.classAttributeValue

InvalidSelectorException: It is one of the unchecked exceptions of selenium which will be thrown if the CSS or xpath expression is not correct.

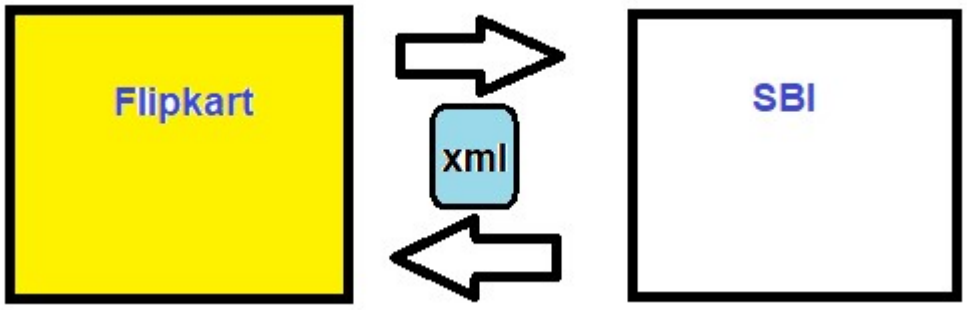
NoSuchElementException: It is one of the unchecked exception of selenium which will be thrown if the given locator strategy is not matching with any element in the webpage currently.

Q. What is the drawback of CSS Selector?
Ans: CSS Selector doesn't support text() function.
OR
We cannot identify the target based on its text because CSS selector doesn't support text() function.
We cannot perform backward traversing in CSS Selector.

Q. What is the advantage of CSS Selector?
Ans:
1. It is faster compared to xpaths
2. Only CSS Selector can be used to identify the shadow DOM element.
3. All browsers support CSS Selector.

Xpath

- X stands for xml.
- XML stands for Extensible Markup Language.
- XML is used for data exchange between two application or between client and server.



Q. What is xpath?
Ans: Xpath stands for xml path language. It is the path of the element in the HTML tree or XML tree. It was designed as a query language to identify the elements in the xml tree. It can be used in HTML tree also to identify the path of any element.

Q. Why we should use xpath?
- If the source code of the element is not having any attribute and if its having any text, then to identify that target element we should use xpath.
- To identify the completely dynamic element we will use xpath.
- To identify the elements uniquely if there are duplicated then we will use xpath.

Q. What are the types xpaths?

Ans: There are two types of xpaths:

1. Absolute xpath.
2. Relative xpath.

1. **Absolute Xpath:** The xpath which starts from the root of the document is called as absolute xpath. Here we will use "." which means **current document** and "/" which means **immediate child.**
   Whenever the xpath matches with the multiple element then all those element stored in the xpath array
   (xpath array index starts from 1).

   Q. What are the drawbacks of absolute xpath?
   Ans:
   1. Very time consuming
   2. Very lengthy
   3. If the position of the element changes the index has to be changes which requires lot maintenance effort.
      So, in real time absolute xpaths are not used.

2. **Relative Xpath:** The xpath which starts from any element of the document is called as relative xpath. Here we will begin the xpath with "//" which means **any child or descendant.**
   Here we will use the combination of both / and // symbols
   Example: //div[1]/input[2]

   Types of Relative xpath:
   1. Xpath by attribute
   2. Xpath by text() function
   3. Xpath by contains()
   4. Independent and dependant xpath
   5. Xpath by group index
   6. Xpath by Axes

Q. What is the advantage of Relative xpath?

Ans: It is shorter and less time consuming compared to relative xpath.

Q. What is the drawback of Relative xpath?

Ans: While writing relative xpaths we might use index to identify the element uniquely, but if the position of the element changes in the future then we have to also change the index.
This task requires more maintenance effort.

Xpath by Attribute:
It is one of the type of relative xpath which is used to identify the target element using its attribute
It can be used in two scenarios :

1. When the source code of the target element is not having id, name and class attributes and if it is not a link
2. While identifying completely dynamic elements.

Syntax: //tagname[@attributeName = “attributeValue”]
//button[@type = “submit”]

Drawbacks:

1. If attributes are not there for the target element we cannot identify it.
2. If attribute is same for multiple element, then we can’t identify our target element uniquely.

Xpath by text():
It is a type of relative xpath which is used to identify the target element based on the text.
Syntax: //tagname[text() = ‘textValue’]
//span[text()=”Qspiders”]
Or
//tagname[. = ‘textValue’]
//span[.=’Qspiders”’]
Drawbacks:

1. If the text is partially dynamic then we need to frequently change the xpath according to the dynamic text which involves too much of maintenance effort.
2. Whenever the target element’s text is having non-breakable spaces then we can’t identify the element using text().
3. When the target element’s text is very lengthy then its not suggestable to use text() as the length of the code will also increase.

Xpath by contains():

It is a type of xpath which is used to identify the target element based on its text or based on its attribute.
Here the text and attribute value can be partial or complete.
Whenever the target element text is partially dynamic or if it contains no-breakable spaces or if the text is very lengthy, then we can use conatains().
Whenever the target element attribute value is partially dynamic or if it is very lengthy, and also if you want to verify whether attribute value contains certain text then we can use contains().

Syntax:
Conatains by text():
//tagName[contains(text(), 'textValue')]
Or
//tagName[contains(., 'textValue')]

Example:
//tagName[contains(text(), 'SignIn')]
//tagName[contains(., 'SignIn')]

Contains by Attribute:
//tagName[contains(@attributeName, 'attributeValue')]

Example:
//tagName[contains(@placeholder, 'username')]

Drawbacks:
1. Whenever the target element text is completely dynamic text then we can't use xpath by conayains().
2. Whenever the target element attribute value is matching with multiple attributes, even with partial values then we cannot use contains().

Independent and dependant xpath:
It is a type of relative xpath which is used to identify the:
a. Completely dynamic element.
b. Element uniquely if there are duplicates.

It is step by step procedure to be followed in which we will perform forward and backward traversing.
Backward Traversing: This means traversing from child to parent.
For backward traversing we should use the symbol **"/.."**
Independent : Fixed Element

Dependent : Dynamic Element

Steps to follow:

1. Identify which one is fixed element and which is dynamic element.
2. Write the xpath for fixed element (make sure it is identified uniquely).
3. Identify the common parent element for both fixed and dynamic element and perform backward traversing using /..
4. Write the xpath for dynamic element.
5. Combine all the xpath together.

Drawbacks:

Sometimes the fixed element itself might be duplicate. In such cases we cannot use independent and dependent xpath.

Xpath by axes:

It is a type of relative xpath which is used to identify the target element based on the relationship between between other element.
It is used to identify the completely dynamic/duplicate elements.

Syntax: /axesName::tagName[predicate]

Axes Names:

1. parent
2. child
3. descendant
4. ancestor
5. following
6. following-sibling
7. preceding
8. preceding-sibling

Dynamic Xpath: It is a type of xpath which is used to identify the completely dynamic element or duplicate elements such as cost of a mobile or add to compare checkbox in E-Commerce application.
There are two types:

1. Independent and Dependent xpath.
2. Xpath axes.

Q. What are the differences between CSS Selector and Xpath?

CSS Selector	Xpath
1. It is faster	It is slow
2. Css Selector doesn't support text()	It supports text()
3. It doesn't support backward traversing	It supports both backward and forward traversing
4. It is used to identify shadow DOM elements	Xpath doesn't support shadow DOM element
5. We can handle <svg> elements using Css Selector	Xpath doesn't identify <svg> element

Xpath by group index:
It is a type of relative xpath which is used to identify the target element uniquely in case of duplicates.
Sometimes, if we cannot identify the target element uniquely with any of the locator strategies then we should use it.

Syntax: (xpath)[index]

//a → All the Links in a webpage
(//a)[1] → First link in a webpage
(//a)[last()]→ Last link of a webpage
(//a)[last()-1] → Second last link
(//a)[position()=5] → Fifth Link in a webpage
(//a)[position() mod 2 = 0] → Even Links in a webpage
(//a)[position() mod 2 != 0] → Odd Links in a webpage
//text() → Only text

Predicate Function:

1. text()
2. contains()
3. last()
4. position()
5. starts-with(text(), 'tv')]
   starts-with(@attributeName, 'attributeValue')]

WebElement Methods in Detail:

1. **click():**

- It is used to click on the target element like, ratio button, checkbox, link etc.
- It will perform two operation:
  a. Scroll the target element to the visible area of the webpage.
  b. It will click on the target element.
- It will not wait for the next page to load.
- It will throw **ElementClickInterceptedException** in case if the target element is obscured by another element.
- The return type is void and it doesn't accept any arguments.

  Signature: public void click(){
  }

  Usage: element.click()

  ElementClickInterceptedException: It is one of the unchecked exception of selenium which is thrown whenever the target element is obscured by another web-element.

  There are three ways to handle it.

  a. First handle the element which is obscuring the target element.
  b. Use JavaScript click
  c. Use delay

```
public static void main(String[] args) {
	System.setProperty("webdriver.chrome.driver", "./drivers/chromedriver.exe");
		WebDriver driver = new ChromeDriver();
		driver.manage().window().maximize();
		driver.get("https://www.makemytrip.com/");
		driver.findElement(By.xpath("//span[text()='ExploreInternational
Flights']")).click();
	}
```

2. **clear():**

- It is used to remove the existing contents from the text field.
- If there are no contents in a text field, then it doesn't throw any exception
- Generally, before we enter any text into the text fields its always a best practice to remove the existing text and then enter new text.
- The return type of this method is void and it doesn't accept any arguments.

  Signature: public void clear(){
  }

  Usage: element.clear();

3. **sendKeys():**
   - It is used to perform 2 operations
     a. Entering the text into the text-fields
     b. To perform keyboard simulations like using special keys of the keyboard. For eg: ENTER, DELETE, etc.
   - It will not remove the existing text; instead existing text is present it will concatenate the entered text along with it.
     So it's a best practice to call the clear() method before using this method.
   - The argument of this method is Char Sequence…
   - The Return type is void.

   Signature: public void sendKeys(){
   }

   Usage: element.sendKeys("Text"); or element.sendKeys(Keys.ENTER);

   Q. What is CharSequence?
   Ans: It is an interface in java, which is a parent for all the string related classes like, String, StringBuilder, StringBuffer. All the string related classes inherits from the CharSequence Interface.

   Q. How can we perform keyboard simulation using sendKeys()?
   Ans: In selenium tool there is an Enum called Keys, which contains all the special keys of the keyboard like ENTER, DELETE, BACKSPACE, CONTROL, etc., using these we can perform keyboard simulations.
   In some scenarios we may have to test whether we can login to the application pressing the Enter Key on the Login Button.

   #WAS to login to the application using enter key.
   #WAS to remove the content from the text field with using clear() method.
   #WAS to copy paste the text from one textfield to another.

   Note: If the given value is null for sendKeys method then it throws IllegalArgumentException which is one of the Unchecked Exception of Java.

4. **getText():**
   - It is used to get the text of the target element.
   - It will give both normal text and also link text.
   - If we have to verify the text of the target element according to the test case then we should use this method.
   - The return type of this method is String.
   - If the target element is not having text, then it returns empty string.

   Signature: public String getText(){
   }
   Usage: String actualText = element.getText();

Script to verify the error message on wrong username input:

```
public static void main(String[] args) throws InterruptedException {
	System.setProperty("webdriver.chrome.driver", "./drivers/chromedriver.exe");
	WebDriver driver = new ChromeDriver();
	driver.manage().window().maximize();
	driver.get("https://demo.actitime.com/login.do");
	driver.findElement(By.id("username")).sendKeys("dsaf");
	driver.findElement(By.xpath("//div[text()='Login ']")).click();
	Thread.sleep(3000);
	String actualText = driver.findElement(By.xpath("//span[text()='Username or
Password is invalid. Please try again.']")).getText();
	String expectedText = "Username or Password is invalid";

	if(actualText.contains(expectedText)) {
		System.out.println("Pass: The error message is verified");
		}
	else System.out.println("Fail: The error message is not verified");

		driver.quit();
		}
```

5. **isEnabled():**
   - It is used to verify whether the target element is enabled or not, based on some conditions given in the test case.
   - The return type is Boolean.
   - If the target element is enabled then it returns true or false.

   Signature: public Boolean isEnabled(){
   }
   Usage: if(element.isEnabled()){
   }

else{
}

#WAS to verify whether the button is disabled or not.

Note: isEnabled() method works properly only for <input> and <button> elements. This method will return unexpected value or incorrect value for other elements.

ElementNotInteractableException: It is one of the unchecked exception of selenium which will be thrown when the target element is not ready or cannot perform the desired action.

6. **isSelected():** It is used to verify whether the radio button, checkbox or option in a Dropdown Listbox is selected or not.
   It returns true if the target element is selected, otherwise it returns false.
   The return type is Boolean
   It doesn't take any argument.

   Signature: public Boolean isSelected(){
   }
   Usage: if(element.isSelected()){
   }
   else{
   }

Script to verify whether the current month is selected by default in Create Account Page of Facebook.

```
public static void main(String[] args) throws InterruptedException {
	System.setProperty("webdriver.chrome.driver", "./drivers/chromedriver.exe");
	WebDriver driver = new ChromeDriver();
	driver.manage().window().maximize();
	driver.get("https://www.facebook.com/");
	driver.findElement(By.linkText("Create New Account")).click();
	Thread.sleep(3000);
	WebElement element = driver.findElement(By.xpath("//option[text()='Jun']"));
	if(element.isSelected()) {
		System.out.println("Pass: The month of June is selected by default");
	}
	else System.out.println("Fail: The month of June is not selected by default");

	driver.quit();
		}
```

7. isDisplayed():

- It is used to verify whether any target element is displayed in the web-page or not.
- The return type is Boolean
- If it returns true, then it means:
  a. Target element is loaded in the webpage and
  b. Target element is displayed in the webpage
- If it returns false, it means:
  a. Target element source code is available in the webpage and
  b. Target element is not displayed in the webpage currently.
- It doesn't take any argument.

Signature: public Boolean isDisplayed(){
}

Usage: if(element. isDisplayed()){
}
else{
}

8. submit():

- It is used to submit a form.
- This method can be used on any element within the form, which means the target element should be a descendant of the <form> tag.
  Note: It can be used on text field, radio button, button etc.
- This method submits the form to web server bypassing the client side validation (any kind of JavaScript code)
- To verify the error messages from the server side we can use this method.
- Sometimes this method is used as an alternative for click() method, if it is not working
- The return type is void and it doesn't take any argument.

Signature: public void submit(){
}
Usage: element.submit();

Difference between click() and submit():

click()	submit()
1. It can be used to click on any element	It can be used to click on only <form> element
2. It will perform the client side validation and then send the form to the server	It will bypass the client side validation and send the form directly to the server.
3. It can be used to verify the client side error messages	It is used to verify server side error messages.
4. It will not wait for the next page to load.	It waits for the next page to load.

9. getSize():

- It is used to get the dimension of the target web-element.
- It returns Dimension class type of object.
- The Dimension class has 2 methods
  a. getHeight(): It returns height of the target web-element in **int** form
  b. getWidth(): It returns width of the target web-element in **int** form
  This 2 methods are used to get the height and width of a particular web-element.
- It doesn't take any argument

Signature: public Dimension getSize(){
}

Usage: Dimension dim = element.getSize();
int width = dim.getWidth();
int height- = dim.getHeight();

10. getLocation():

- It is used to get the X and Y co-ordinates of the target web-element.
- It returns Point class type of object.
- The Point class has 2 methods
  a. getX(): It returns startX of the target web-element in **int** form.
  b. getY(): It returns startY of the target web-element in **int** form
  This 2 methods are used to get the height and width of a particular web-element.
- It doesn't take any argument

Signature: public Point getLocation(){

}

Usage: Point p = element.getLocation();
int startX = p.getX();
int startY = p.getY();

11. getRect():

- It is used to get both dimension and co-ordinate of a target web-element.
- It retuns Rectangle class type of object.
- Rectangle class has 4 methods:
  a. getWidth().
  b. getHeight().
  c. getX.
  d. getY.

Signature: public Rectangle getRect(){
}

Usage: Rectangle rect = element.getRect();
int width = rect.getWidth();
int height = rect.getHeight();
int startX = rect.getX();
int startY = rect.getY();

12. getAttribute(Attribiute Name):

- It is used to get the value associated with given attribute name of the target web-element.
- The argument and return type of this method is String.
- If the given attribute name is wrong or not present in the source code of the target web element, then this method returns null.
- If the given attribute name is not having any value, then it returns empty String.
- It is used to verify whether the tool-tip text or attribute text of an image or it can be also used to verify whether a text field contains value or not.

Signature: public String getAttribute(String atributeName){
}

Usage: String attributeValue = element.getAttribute("attributeName");

Note:

- title attribute is used to create the tool-tip text
- alt attribute is used to create alternate text of an image.

13. getCssValue():

- **I**t is used to verify the styling information of the target web element like color, background color, front, size, front-weight, front family etc.
- The Argument and return type of this method is string.
- This method takes the CSS property name as the String argument and returns the property value associated in the String form.
- If the property name is invalid or if the property name is not having any value then returns empty string.

Signature: public String getCssValue(String CssPropertyName){
}
Usage: String cssPropertyValue = element.getCssValue("CssPropertyName");

RGBA stands for Red, Green, Blue and Alpha
RGBA highest value is 255 and lowest value is 0
Alpha is used to represent whether color is transparent or opaque
0→Transparent
1→ Opaque
0.5→Semi-Transparent

14. getTagName():

- It is used to get the tagname of the target web element.
- The return type is String
- The target element tagname will be returned in the String form
- Whenever we need to verify the tagname of the target element, then we will use this method.
- Generally, this method is used to verify element tag before performing the desired action.
  For example: If we are creating any library method to enter data into a text field then we can use this method to verify whether the tag is <input> tag or not.

- It doesn't accept any argument.

Signature: public String getTagName(){
}
Usage: String tagName = element.getTagName();

findElements():

- It is one of the methods of SearchContext
- It is used to identify multiple elements in the webpage.
- The return type is List<WebElement>
- If the given strategy is matching with one or more elements, then it stores it in the List <WebElement> and returns the same.
- If the given strategy is not matching with any elements of the webpage then it returns empty list.

- We use this method whenever we need to:
  1. Test Auto-suggestion.
  2. Test default Auto-suggestions.
  3. Order of a Menu Container.

Signature: public List <WenElement> findElements(By by){
}
Usage: List<WebElement> element = driver.findElements(any Locator Strategy)

Active Element:

- It is one of the methods of TargetLocator interface.
- The return type is WebElement
- It is used to identify the current active element in the web page
- **Active Element** means the element which is currently having focus.
- Whenever we need to verify the currently focused element in the webpage like if some mandatory field is to be verified then we can use this method.
- It doesn't accept any argument.

Signature: public WebElement activeElement(){
}
Usage: driver.switchTo().activeElement().click();

Synchronization:

Synchronization is a process of matching two different events according to requirement of the system to get the desired outcome.

In Automation synchronization is a process of matching the application loading speed with the automation script execution speed.

Always application loading speed is slow when compared with automation script execution speed.
Application loading speed will be slow due to below reasons:

- Internet Connection is slow
- Webserver is busy.
- Poor business logic in webserver

Types of Synchronization:

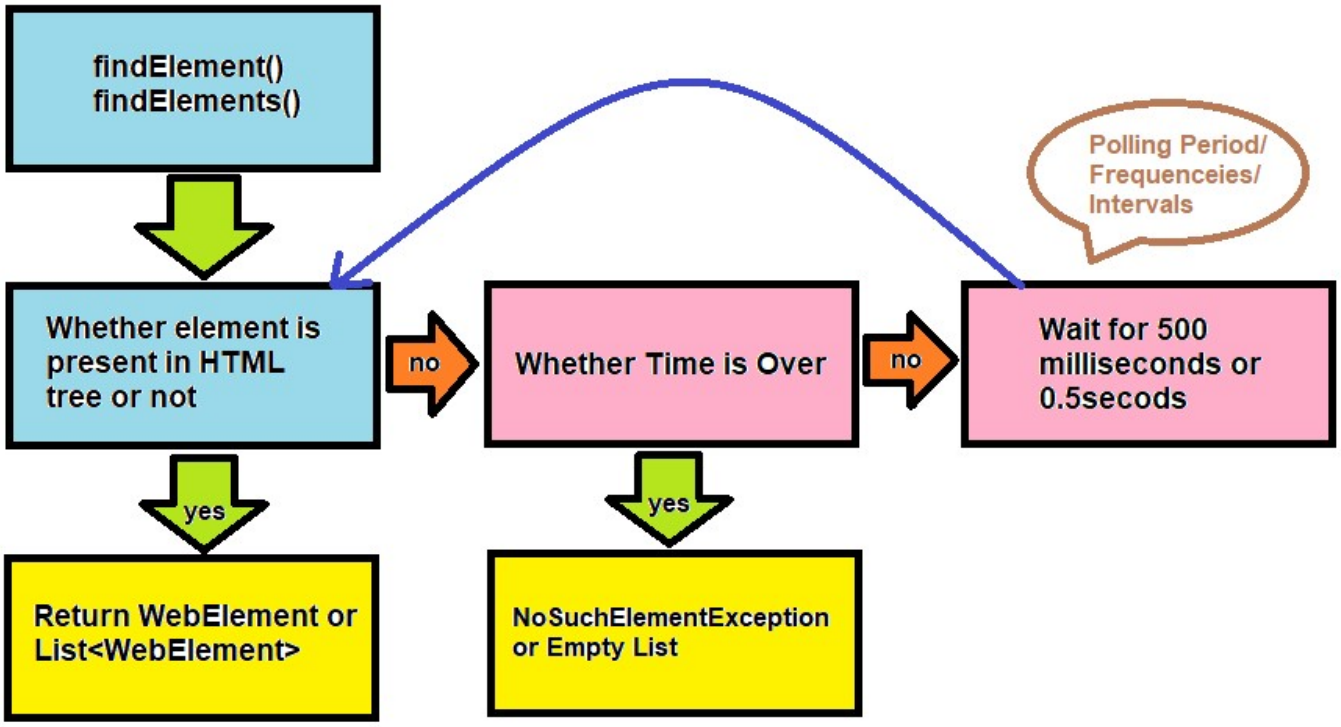
1. Implicit Wait
2. Explicit Wait
3. Fluent Wait
4. Thread.sleep
5. Custom Wait.

Advantage of Synchronization:

1. We can save time.
2. Avoid NoSuchElementException.

3. We can get proper test outcomes.

Implicit Wait:



- It is one of the intelligent waits in selenium WebDriver, which helps us to save time while we wait for the web element to load on the DOM or HTML tree.
- Here we provide the time delay for searching elements on the webpage.
- If the element is found anytime within the specified time delay, then the code doesn't wait for the remaining duration, rather it returns the found element. Therefore, it is called as intelligent wait.
- It uses 500 milliseconds (0.5 seconds) of polling period or search frequency.
- If the element is not found within the specified delay, then we will get NoSuchElementException or empty List<>.
- We should use implicit wait method on the Timeouts object. It receives two arguments one is the time delay and another is the TimeUnit.

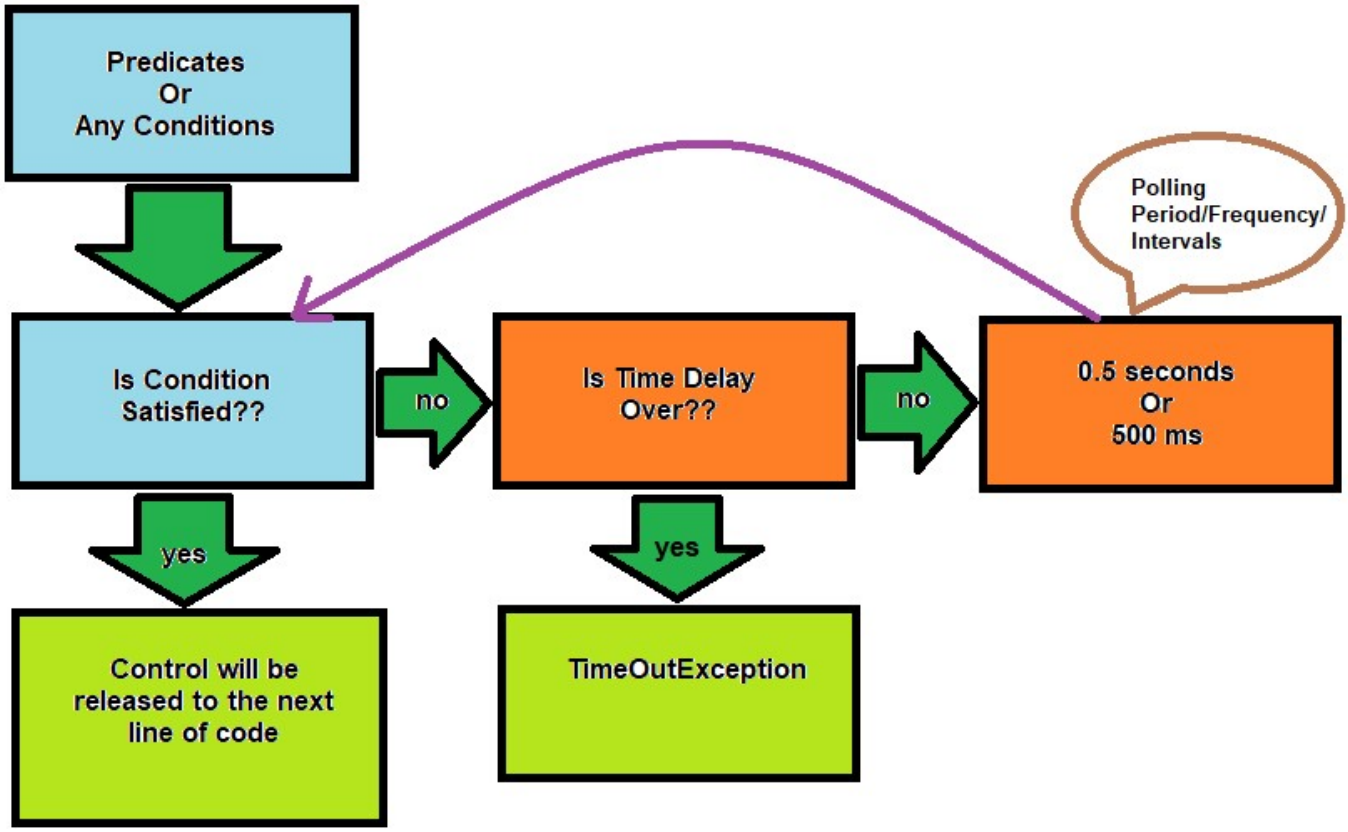
Signature: public TimeOuts implicitlyWait(Long timeDelay, TimeUnit){
}

Usage: driver.manage().timeouts().implicitlywait(20, TimeUnit.SECONDS)

Q. What are the different TimeUnits allowed for implicitly wait?
Ans: There are 7:

1. SECONDS
2. MILLISECONDS
3. MICROSECONDS
4. NANOSECONDS
5. MINUTES
6. HOURS
7. DAYS

Explicit Wait:



- It is one of the intelligent wait in the selenium WebDriver which helps us to save time while waiting for:
  1. Expected title of the webpage to load.
  2. Expected URL of the webpage to load.
  3. Element to reach to the clickable state.
  4. Image to visible on the webpage.
  5. Alert to be present on the webpage and so on...
- Here we provide time delay for the expected conditions to be successful.

- If the expected condition is satisfied anytime within the specified time delay, then the code doesn't wait for the remaining duration. It releases the control to the next line of the automation script.
- It uses 500ms of polling period or search frequency.

How to apply explicit wait in selenium WebDriver code?
Ans:

1. Create the object of WebDriver wait.
2. Pass the WebDriver reference and time delay for the constructor.
3. Call the until() method.
4. For until() method supply the predicate finctions.

Expected Conditions:

It is a concrete class of selenium which contains approximately 25 static methods which are also called predicate functions.

Eg: titleIs(), titleContains(), urlToBe(), urlContains(), alertIsPresent(), visibilityOfElement(), elementToBeClickable() etc…

Implicit Wait	Explicit Wait
1. It is applicable only for findElement() and findElements()	It can be used for any condition.
2. It throws NoSuchElementException if the time delay is over	It throws TimeOutException if the time delay is over
3. We can use TimeUnits like SECONDS, MINUTES, HOURS, DAYS etc..	Only seconds is allowed (In selenium4 it is allowed)
4. Here we don't have to specify condition explicitly	Here we have to specify condition explicitly
5. This will be applied once in the script	We need to apply this again and again depending on the various conditions

Actions Class:

Actions Class is the concrete class of selenium which is used to perform various keyboard and mouse gestures like moving the cursor to an element or dragging and dropping an element on to the destination element.

This class has approximately 21 methods to perform the keyboard and mouse gestures. They are:

1. moveToElement(WebElement)
2. moveToElement(WebElement, xOffset, yOffset)
3. moveByOffset(xOffset, yOffset)
4. perform()
5. click()
6. click(WebElement)
7. doubleClick()
8. doubleClick(WebElement)
9. contextClick()
10. contextClick(WebElement)
11. dragAndDrop(WebElement Source, WebElement destination)
12. dragAndDrop(WebElement source, xOffset, yOffset)
13. sendKeys(character sequence)
14. sendKeys(WebElement, character sequence)
15. keyDown(character Sequence)
16. keyUp(character Sequence)
17. clickAndHold(WebElement)
18. release()
19. release(WebElement)
20. build()

How to use Actions Class?

1. Create an object of Actions Class and pass the driver reference to the constructor of the class.
2. Store the address of the object into a reference variable.
   Eg: Actions action = new Actions(driver)
3. Use the reference variable to call all the non-static methods of actions class.
4. Use method chaining to perform composite action on an element.
   Eg: action.moveByOffset(200, 100).click().perform();
5. Use perform method, otherwise the action will not be performed.

ActionClass methods in Details:

1. **moveToElement(WebElement target):**
   - This method is used to handle dropdown menu or the menu which will open when you do mouse hovering. Using this method, we can hover pointer on top of a given target element.
   - It will move the mouse pointer to the mid-point of the given target element.

   Signature: public Actions moveToElement(WebElement target){

   }

   Usage: action.moveToElement(target).perform();

2. **moveToElement(WenElement target, xOffset, yOffset):**
   - This method is used to move the mouse pointer to the particular coordinate of the given target element.
     For eg: To rate the food in Zomato we can move from the particular coordinate to the rating element and provide ratings.
   - Here Xoffset and Yoffset means the x and y coordinates of the element.
   - To move to the right side of the element we should give x value (for example 100) and y value should be 0, so that from the midpoint of that element mouse will move to 100 pixels right.

   Signature:
   public Actions moveToElement(WebElement target, int Xoffset, int Yoffset){

   }

   Usage: action.moveToElement(target, 100, 0);

3. **moveByOffset(Xoffset, Yoffset):**
   - This method is used to move the cursor to a particular coordinate of the webpage.
   - If we want to do double click or a right click on the same coordinate on the webpage we can use this method.

   Signature:
   public Actions moveByOffset(int Xoffset, int Yoffset){

   }
   Usage: action.moveByOffset(200, 10).perform();

4. **perform():**
   - This method is used to execute single action or composite actions.
   - If this method is not called, none of the actions will be executed.

   Signature:
   public Actions perform(){

   }

   Usage: action.moveToElement(target).perform();

5. **click():**
   - In the series of composite actions, at some point we need to click on an element than we can use this method.
   - Or else it will click on a location where current mouse pointer is available.

   Signature: public Actions click(){

   }
   Usage: action.moveToElement(WebElement target).click(){

   }

6. **click(WebElement target):**
   - **In the series of composit actions to click on a WebElement we can use this method**

   Signature: public Actions click(WebElement target){

   }
   Usage: action.click(target){

   }

7. **contextClick():**
   - It is used to perform right click on the location where the mouse pointer is available.
   - The menu which will open is called as context menu.
   - Whenever we have some operation to be done using context menu then we can use this method.

For example: Some of the application will have their own context menu in this case it will be useful.

Signature: public Actions contextClick(){

}

Usage: action. contextClick (){

}

8. **contextClick(WebElement target):**
   - It is used to right click on any web-element

   Signature: public Action contextClick(WebElement target){

   }

   Usage: action. contextClick (target);

9. **dragAndDrop(WebElement source, WebElement Destination):**
   - This method is used to place an element from one location to another location. Here, location is dependent on the given web element. In some of the applications for simplicity purpose like a shortcut developer will provide this option to the end user. To test such scenarios, we will use this method.
   - **Signature:**
     public Action dragAndDrop(WebElement source, WebElement Destination){

     }

   - **Usage**: action. dragAndDrop(source, destination).perform();

10. **dragAndDrop(WebElement target, int Xoffset, int Yoffset):**
    - This method is used to move the given element to a given coordinate of the webpage
    - **Signature:**
      public Action dragAndDrop(WebElement target, int Xoffset, int Yoffset){

      }

    - **Usage:** action.dragAndDrop(target, 100, 50).perform();

11. clickAndHold():

- This method is used to click and hold the mouse cursor on the mouse pointer on the current mouse location.
- Signature: public Actions clickAndHold(){
  }
- Usage: action.moveToElement(target).clickAndHold().perform();

12.clickAndHold(WebElement target):

- This method is used to click and hold the mouse cursor on the mouse pointer on the current mouse location.
- **Signature:** public Actions clickAndHold(){
  }
- **Usage:** action. clickAndHold().perform();

13.release():

- This method is used to perform the release of mouse event.
- Whenever we need to drag and drop then we can use this method
- **Signature:** public Actions release(){
  }
- **Usage:** action. release().perform();

14. release(WebElement target):

- This method is used to release the mouse keypress event.
- While performing dragAndDrop option we can use this method.
- **Signature:** public Actions release(WebElement target){
  }
- **Usage:** action. release(WebElement target).perform();

15. doubleClick():

- This method is used to doubleClick on the location where mouse is available.
- **Signature:** public Actions doubleClick(){
  }
- **Usage:** action. moveToElement(target).doubleClick ().perform();

16. doubleClick(WenElement target):

- This method is used to doubleClick on on a particular WebElement
- **Signature:** public Actions doubleClick(WebElement target){
  }

- **Usage:** action.doubleClick (WebElement target).perform();

17. sendKeys(Character Sequence… args):

- This method is used to perform keyboard actions on the webpage.
- This method is not specific to any WebElement.
- **Signature:** public Actions sendKeys(Character Sequence… keys){
  }
- **Usage:** action. sendKeys(Keys.PAGE_DOWN).perform();

18.sendKeys(WebElement target, Character Sequence… args):

- In the series of composite action to enter any data into any data into any web element we can use this method.
- **Signature:** public Actions sendKeys(WebElement target, Character Sequence… keys){
  }
- **Usage:** action. sendKeys(Keys.PAGE_DOWN).perform();

19. keyDown(Character Sequence):

- This method is used to perform key press event
- This method can be used on three modifier keys i.e. Keys.CONTROL, Keys.SHIFT, Keys.ALT.
  (Modifier means the key which modifies the functionality of other keys. For Example: Control + C → To copy)
- Whenever we need to perform a control click operation, we can use this method.
- After calling this method key will remain in the presses state. So, KeyUP() method should be called to release the specific key.
  - **Signature:** public Actions KeyDown(Character Sequence){
    }
  - **Usage:** action. KeyDown(Keys.CONTROL).perform();

20. keyUp(Character Sequence):

- This method is used to perform key release event. This method is opposite to keyDown() method.
  - **Signature:** public Actions KeyUp(Character Sequence){
    }
  - **Usage:** action. KeyUp(Keys.CONTROL).perform();

21. build():

- This method is used to build all the composite actions which are stored.
- In the earlier versions of selenium before executing all the actions using perform method we had to call build method explicitly.

- In the latest versions of selenium perform method itself internally call build method, to build all the composite actions and later execute it. So no need to call build method explicitly.
- This method is the only method of Actions Class whose return type is Action.
- **Signature:** public Action KeyUp(Character Sequence){
  }
- **Usage:** action.click().build().perform();

Select Class:

It is the concrete class of selenium which has lot of non-static methods to handle both single select and multi-select list-box.
Select class methods will work only if the list box is created using select tag.
Eg: <select>
</select>

How to use Select Class methods?

Ans:

1. Identify the list box using findElement().
2. Create an object of Select class and pass the reference of the list box element in to the constructor.
3. Store the address in a reference variable.
4. Use the reference variable to call various methods of Select Class.
   Eg:
   WebElement selectElement = driver.findElement(By.id("oldSelectMenu"));
   Select select=new Select(selectElement);
   select.selectByIndex(2);

Methods of Select Class:

1. isMultiple()
2. getOptions()
3. selectByVisibleText()
4. selectByIndex()
5. selectByValue()
6. deSelectByVisibleText()
7. deSelectByIndex()
8. deSelectByValue()
9. deSelectAll()
10. getAllSelectedOptions()
11. getFirstSelectedOption()

isMultiple():

- This method is used to verify whether the list box is a single-select list box or multi-select list box.
- It returns true if the list box is multi-select, and false otherwise.
- Some deselect methods can be only used on multi-select listbox.
  Eg: select.isMultiple();

getOptions():

- This method is used to get all the options available in the listbox.
- We can do some operations on all elements by using this method like select and deselect.
- This method will return all the options in the form of list type collection.
  Eg: select.getOptions();

SelectByVisibleText(String text):

- This method is used to select the options present in the list-box on the value displayed in the UI.

SelectByIndex(int index):

- This method is used to select the options of a list box by using its index.
- Every option in the list box will be automatically assigned with the index and the index starts from zero

SelectByValue(String value):

- This method is used to select the options present in the list box by passing the value attribute name as an argument of the method.
- **Example :** Red
  select.selectByValue(“red”);

deSelectAll():

- This method is used to deselect all the selected options.

getAllSelectedOptions():

- This method will return all the options which are currently selected.
- It returns all of the web element in the list type of collection.

Example: List<WebElement> allOptions = select.getAllSelectedOptions();

getFirstSelectedOptions():

- This method is used to get the options which is selected as the first options.

JavaScriptExecutor:

It is an interface which provides the mechanism to write the java script code in selenium webdriver. It has two abstract methods.

1. executeScript() : It is used to write synchronous java script code.
2. executeAsynScript() : It is used to execute asynchronous java script code.

Synchronous means one task at a time or single threaded and **Asynchronous** means multi task at a time or multiple threaded task.
These two methods are implemented in RemoteWebDriver class. To use these methods we should cast the WebDriver object to JavaScriptExecutor type i.e.
JavaScriptExecutor js = (JavaScriptExecutor) driver;

The executeScript() method receives two argument, they are:

1. Java script code in string form.
2. Generic type which acts as input to java script code.

Few Window object methods:

1. **window.scrollBy(xPixel, yPixel):**
   - This method is used to scroll the window horizontally or vertically based on the given distance.
   - To Scroll down 500 pixels from current operation
     Window.scrollBy(0, 500)
   - To scroll up 500 pixels from current operation
     Window.scrollBy(0, -500)
2. **window.scrollTo(xCoord, yCoord)**
   - This method is used to scroll the window to a particular coordinate from any position.
   - To scroll the window to the extreme bottom of the webpage, window.scrollTo(0, document.body.scrollHeight);
   - To scroll the window to the extreme top of the webpage, windowscrollTo(0, -document.body.scrollHeight).
3. **Element.scrollIntoView(Boolean):**

- This method is used to move the element to the visible area of the page. If the Boolean value is true it will take the top of the element to the top of the webpage and if Boolean value is false then it moves the bottom of the element to the bottom of the webpage.

4. **Element.click():** This is used to click on the given element. It can be enabled or disabled element or hidden element.
5. **Element.value:** This is the var args or attribute which is used to set the value of the input element.
   argument[0].value='Qspiders'

Q. Why we should use javascript code in selenium WebDriver?
Ans: Sometimes selenium webdriver commands will not work as expected. For example click() may not work on the identified element. So in that case we can use java script code as a work-around.
And also whenever there is no direct method to perform some operation on the webpage, we can use javascript code in selenium webdriver.

For example: There is no direct method to scroll the window, so here we can use java script code.

Q. List out all the operations which can be performed using JavaScriptExecutor.
1. Window scrolling in all direction
2. Performing action on the disabled element:
   - Clicking on disabled link/button.
   - Entering data into the disabled text field.
3. Performing actions on the hidden elements like entering data into the hidden text field.
4. To scroll the window or element to the visible area.

Handling Frames:

Q. What is a frame?
Ans: A Frame is nothing but webpage inside a webpage or it is called as embedded webpage. Sometimes, whenever developer want to integrate third party- application into the site then this concept will be used.

Q. Which HTML tag is used to create a frame?
Ans: ifram tag
Example: <iframe></iframe>

Q. How to handle frames?

Ans: Whenever we need to perform action on the element within the frame we should pass the driver control to the frame and perform the action.

Initially the driver control will be in the default content area of the parent window so we should switch the driver control to the frame.

There are 3 ways to switch the driver control to the frame:

1. Through frame index
2. Through name and id of a frame.
3. Through frame WebElement.

Through frame index:

- In group of frames to identify a particular frame we can use the position of it using index, that is using **driver.switchTo().frame(0);**
- Here index starts from zero.
- This is the fastest version of switching to frame.
- This is the best approach when there is only one frame in the webpage.
- If multiple frames are there, then this approach is not the best approach because position of the frame might change.

Through name and id of a frame:

- In the group of frames, we can identify the frame based on id or name attribute value.
- If name or id is available for a frame then we can use this method that is driver.switchTo().frame("idAttributeValue");
  driver.switchTo().frame("nameAttributeValue");
- Given a string value for this method it will first search whether the frame is having this value under name attribute and if name is not there it will search whether this value is under id attribute or not.

Through frame WebElement:

In the group of frameif the desired frame is not having id or name then we should go for this version.

Here first we should identify the frame using findElement() or findElements() then switch the control to it by supplying the target frame element by using driver.switchTo().frame(target WebElement).

Q. How to switch the control back to the default or parent window or frame?

Ans:

- Generally, if the page is refreshed the driver control will automatically switch back to the default content area. In that case no need to explicitly switch back to the parent frame.
- By using two methods, they are
  a. driver.switchTo().defaultContent()
  b. driver.switchTo().parentFrame()

Data Driven Testing:

1. **What is Data driven testing or parameterization?**

   Ans: Data Driven is a testing strategy in which the automation scripts are written independent of the test data rather than hard coding of the data in the automation script

   Here test data will be stored in the external resource like excel sheet, property file, data base, json file, csv file, etc.

2. Why Data Driven testing?

   As per the rule of the automation data shouldn't not hardcoded(fixed) with in a test scripts, because data modification & maintenance is tedious job when you want to run the test with different data, instead we should get the data from external resource like xlsx, .properties file , db , XML, JSON, CMD Line Data

3. What is Advantages of Data driven testing
   1. Maintenance of the test data is easy
   2. Modification of the test data in external recourse is easy
   3. Cross browser /platform testing is easy (means change the browser in property File)
   4. Running test scripts in different Environment is easy
   5. Running test scripts in different credentials is easy
   6. We can create the test data prior the Suite execution (we can also get the data from testData team)
   7. Rerunning same test Script with multiple time with different data is easy

1. **What is Properties File?**

Properties is java feature file where we can store the data in from of key & values pair,

Key & value data type should be always string .

2. **How to read data from properties File?**
   - Get the java representation Object of the Physical file using "FileInputSteam
   - Create a Object of "Properties" class & load all the keys
   - Read the data using getProperty("Key")

Note : properties file light weight & faster in execution compare to Excel

Excel Sheet: Apache POI:

Apache-POI is the open source software package which contains the libraries to read the data from the excel sheet with .xlsx or xls format.

Steps to configure Apache poi in the project:

- **Go to https://poi.apache.org/download.html and download** poi-bin-5.2.2-20220312.zip file.
- **Extract the file and add all the (19 jars) to the project**
- **Build path for the all the jar files.**

What is Workbook, Sheet, Row, Cell in apache-poi hierarchy?

Ans: All these are interfaces in the interface hierarchy of apache poi library.

Workbook is an object representation of physical excel file,

Sheet is an object representation of a excel sheet,

Row is an object representation of a row in the excel sheet,

And Cell is an object representation of a cell in the excel sheet.

How to read data from Excel Sheet?

Ans:

- Create an Object of FileInputStream class and pass the file path to the constructor of the class.
- Create Workbook object using create() method of WorkbookFactory class.

- Call getSheet(), getRow(), and getCell method to navigate to particular cell in the excel sheet.
- And finally get the required data using:
  1. getNumericCellValue()
  2. getBooleanCellValue()
  3. getStringCellValue()
  4. getLocalDateTimeCellValue()

Pop Ups:

What is a Pop up?

Ans: It is a GUI window which opens on top of the browser window. It is basically used to give same vital information to the end user and also to take some vital information from the end user.

It is also used by many websites to grab the attention of the end user(especially the Ads).

In automation we should take care of testing the popups, like verifying the expected behaviors of the popups are working fine or not, pop ups texts are working as per the requirement or not.

What are the strategies to handle a popup?

Ans:

1. Check whether we can inspect the popup or not. If we can inspect than we can handle the pop up using findElement(), findElements() and Actions Class etc.
2. If the pop up cannot be inspected than we should investigate whether any library method is provided within Selenium WebDriver. If provided use those to handle the popup.
3. If not try to avoid the popup.
4. Or use Robot class methods to handle the popup.
5. Or use open-source third party tools, one them being AutoIt.

List of all the pop ups:

1. Java script pop up
2. Hidden Division pop up
3. Child Browser pop up
4. File Upload pop up
5. File Download pop up
6. Notification pop up
7. Authentication pop up
8. Print pop up

Java script pop up:

The pop up which is created using java script language is called as Java Script Pop Up.

It will appear at the top and middle of the webpage.

It can have ok and cancel button and also a text field.

There are 3 types of Java Script Pop up:

1. Alert Pop up: It will have only ok button
2. Confirmation Pop up: It will have both ok and cancel button
3. Prompt Pop up: It will have a text field along with ok and cancel button.

We cannot inspect this pop up.
Selenium library methods are present for it. So we can use those methods to handle this popup.
For example: driver.switchTo().alert().accept()

Hidden Division Popup:
The popup which is created using the html code is called as the hidden division popup.
This popup will appear anywhere on the webpage.
We can inspect this popup, and we can use findElement() and findElements() methods to identify the elements of the popup (such as close button) and handle it.

This popup is colorful and usually it is used for advertisement purpose.

Child Browser Popup:
Sometimes when the user opens a webpage along with the webpage or when he clicks on a link, another browser window opens, this browser window is known as child browser or child browser popup.

We cannot inspect the pop up, so we have to look for selenium library methods.

To handle this popup we can use getWindowHandle() and getWindowHandles() methods to get the window IDs of the browser window and iterate through the window ids to switch the driver control to the required window and perform the required action.
For example:

```
driver.get("https://demo.actitime.com/login.do");
String parentWindowId = driver.getWindowHandle();
driver.findElement(By.linkText("actiTIME Inc.")).click();
Set<String> allWindowIDs = driver.getWindowHandles();
for(String windowID:allWindowIDs) {
```

```
            driver.switchTo().window(windowID);
            if(!windowID.equals(parentWindowId)) {
                  driver.findElement(By.linkText("Try Free")).click();
                  break;
            }
      }
      driver.switchTo().window(parentWindowId);
      driver.findElement(By.id("keepLoggedInCheckBox")).click();
```

File Upload Popup:

- This is the pop up which will open whenever user needs to upload a file.
- Generally clicking on the upload button on the webpage will generate this pop up
- It is one of the operating system level pop ups
- We cannot inspect it, in order to handle this we should go f0or third party tool or Robot Class
- We can maximize and close it.

How to handle File Upload Popup?

As File Upload Pop up is a GUI level pop up, that is it is not part of the webpage, we need to use some third party tool which can automate standalone applications. Here we are using autoIt tool to do the same.

Steps to use autoIt in our scripts are as follows:

1. Download and install autoIt.
2. Open autoIt tool in C drive → program file (x86) folder → autoIt folder →Scite Folder.
3. Write autoIt script on the editor of the tool.
4. Save the file with .au3 extension
5. Compile the autoIt script and .exe file will generate.
6. Pass the control to the autoIt tool from the IDE using Runtime.getRuntime.exec("path of the .exe file").
   exec() method takes path of the .exe file in String format.

File Download popup: This is a popup which will open whenever user needs to download some online content. It is one of the OS level pop ups.
This pop up is not inspectable.
We can use autoIt or Robot class to handle it.

Notification Popup:

This is a popup which will open in the browser when we load it and use to take permission from the end user for their website to show some notifications.,
It will have close, allow and block button.

We can handle it using robot class or AutoIt
We can avoid it using the following code for Chrome:

```
ChromeOptions option = new ChromeOptions();
option.addArguments("--disable-notifications");

System.setProperty("webdriver.chrome.driver", "./drivers/chromedriver.exe");
WebDriver driver=new ChromeDriver(option);
driver.manage().window().maximize();
driver.manage().timeouts().implicitlyWait(20, TimeUnit.SECONDS);
driver.get("https://www.yatra.com/");
```

Authentication popup:
This is the popup which will appear whenever we are working with some gateway application and only certain user are allowed to use this application.
We cannot inspect this pop up, so we have to use either Robot class or AutoIt to handle it.
We can also handle it by pass the username and password in the url.
For eg: http://admin:admin@the-internet.herokuapp.com/basic_auth

Print popup/Window popup:
This is a popup which will appear on the browser when the user is trying to take out a print out of any document.
This a OS level popup, so cannot inspect it.
We need to use Robot class or AutoIt to handle it.

POM (Page Object Model)/Object Repository/Element Repository

What is Design Patern?

Ans: Design patterns represent the best practices used by experienced object-oriented software developers. Design patterns are solutions to general problems that software developers faced during software development. These solutions were obtained by trial and error by numerous software developers over quite a substantial period of time.

What is POM (Page Object Model)/Object Repository/Element Repository?

Ans: POM is a design pattern which is used in automation to write the test scripts independent of the element identification. Automation rule says, we shouldn't hard code the element identification statements within the test script. Therefore, we create separate classes for each page of the Application Under Test (AUT) and identify all the required elements in those classes using @FindBy, @FindBys, and @FindAll annotations.

Advantage of POM:

1. Reduces effort in maintenance of code
2. Reusability of the code
3. Readaptibility of the code
4. We can handle StaleElementReferenceException
5. We can avoid manipulation of element identification.

Procedures to be followed while building a POM Class:

1. Create a java class for each page of the application. Name the java class with suffix "Page". For eg: LoginPage, HomePage etc..
2. Store the WebElement of each webpage into their respective page classes. For eg: All Login Page web element should be stored in LoginPage.java class.
3. We should identify the web element using @FindBy annotation.
4. All the web elements of each class should be initialized with PageFactory.initElement() method.
5. The variable which holds the web element should be declared with "private" access specifier.
6. Create the parameterized constructor for each page class which accepts the same WebDriver reference.
7. Create business logic or action methods to perform a full fledged action like login action, create customer, add to cart etc.

What is the difference between POM and PageFactory?
Ans: POM is the design pattern and PageFactorty is a class which is used within POM. We can say PageFactory is a subset POM.

PageFactory is a class which we will use to initialize the WebElement of the page which are created using @FindBy annotation (We get NullPointerException otherwise). This has a static method called initElements which will accept two inputs one is WebDriver reference and another is the page class object.
PageFactory.initElement(driver, PageObject Reference).

What is the drawback of @FindBy annotation?
We can't write dynamic xpaths here. Instead we should use findElement() and findElements() methods.

How to design POM Class?
1. Create a class for the specific page with Page as suffix, eg: LoginPage.java.
2. Identify all the elements on the page using @FindBy. Provide variable name and make the variable as private.
3. Generate getters method to provide the access to the vraiables.
4. Provide a parameterized constructor and provide WebDriver driver as a parameter.
5. Provide PagaFactory.initElement(driver, this) in the constructor.
6. Write Business Logic or Generic methods for a repetitive task on a particular page.

TestNG

1. TestNG is an open source unit testing framework or unit testing tool where NG stands for Next Generation
2. TestNG is a unit testing TDD (Test Driven Development) framework, which supports java and .net.
3. TestNG was developed as an additional plugin for Eclipse
4. In case of automation, TestNG will be used to develop all the scripts using TestNG annotations and achieve batch execution without any manual interaction.
5. TestNG will be used to handle all framework component & help us to run all the test scripts in batch / parallel / group without any manual intervention.
6. TestNG.xml is main controller of the selenium framework, where we start the execution.
7. TestNG is used by automation engineers because of its features and advantages

Advantage:

- Generate Reports
- Batch Execution of Test Cases
- Group Execution of Test Cases
- Distributed parallel execution of Test Cases
- Perform Parameterization
- Perform Assertion
- Add preconditions and post condition
- Add dependency of test cases
- Prioritize the test cases
- Disabled the test cases.
- Provides different annotations

What is annotation?

Annotation is the java template which is used to give information or metadata to compiler, developer and runtime environment.

Annotation can be used on top of variable, methods or classes(if allowed).

Annotation always start with @symbol
@Test
@BeforeMethod

@AfterMethod
@BeforeClass
@AfterClass
@BeforeTest
@AfterTest
@BeforeSuite
@AfterSuite
@parameters
@dataProvider
@Listener

Installation steps of TestNG:

- Go to Eclipse Window Click on help option-> Eclipse Marketplace
- Write TestNG in find edit box and click on go button.
- Find TestNG for eclipse division and click on install button
- Click on confirm button and I accept the terms and conditions and click on finish
- In order to verify the TestNG installation- →Go to windows →Show view →others →expand java folder, TestNG symbol will be present

@Test:

1. It is a core annotation of testing
2. Whenever we execute TestNG class, java Compiler/Interpreter always looks for @Test Annotation method to start the execution.
3. Without @Test , TestNG class will not be executed, @test annotation method act like main method in TestNG
4. In one TestNG class we can have multiple @test methods, but each test method should have @Test annotation before method signature.
5. Annotation method return type should be “void” and access specifier should be public, but method name can be anything (we have provide manual testNameTest).
6. As per the Rule TestNG class Name & TestNG method Name should end With “Test”.
7. In one class we can keep multiple @test annotation , but in real time we are going maintain maximum 10 to 15 @test script , because maintenance will be easy.

Priority

Whenever we execute TestNG class, by default all the test methods will be executed based on Alphabetical Order , in order the change the Order of Execution , we go for priority.

DependsOnMethod :

Its help us to check the dependent test case is pass or fail, If dependent test-script get pass, execution will continue If dependent test-script get fail, skip the all other test script which is dependent on first test

Invocation Count:
Same test-script executed multiple times with same test data

@BeforeMethod and @AfterMethod
1. Before method annotations will be executed, before executing each @Test method in a class
2. After method annotation will be executed, after executing each @Test in a class
3. BeforeMethod & AfterMethod will not be executed, without @test annotation method, because they are configuration method
4. In order to implement similar pre- condition for all the test cases like “LOGIN code ” we go for @ BeforeMethod
5. In order to implement similar post-condition for all the test cases like “LOGOUT code ” we go for @ AfterMethod

@BeforeClass & @AfterClass
1. BeforeClass Annotation method will be executed, before Executing first @test in a class
2. AfterClass Annotation method will be executed, after executing all/last test-case within a class
3. BeforeClass & AfterClass annotations will be executed only once in an entire class execution.
4. It will be used to develop global configuration like launch browser, object initialization NOTE: As per the Rule of the Automation, there should not be any dependency between two test cases, every test case should be unique (it means every test should have fresh login & logout code).

Batch Execution
- Collection of multiple test scripts together is called a batch, to execute multiple test scripts through the xml in a single click is called batch execution.
- In order to achieve batch execution, we go for Testng.xml configuration file
- TestNG xml file always start with suite xml tag followed by and
- In one xml file we can invoke N-number of testng classes, but all the classes should be present within a project.

How to create testNG xml file automatically through eclipse?

- Select all the testNG classes or packages -> right click-> select➔ testNG ➔and click on “Convert to testing” and➔ click on finish.
- Automatically you will get the testng.xml file with in the project
- In order to edit xml File➔ double click on testing.xml➔ click on “Source”

Grouping Execution:

- Collection of similar test scripts across the testing classes is called grouping Execution
- In order achieve grouping execution , each & every test script should have group name, group name will be written along with annotation
- In grouping execution, all configure annotation should have group name , other wise those annotation will not participate in grouping execution like

@BeforeSuite @BeforeClass , @BeforeMethode etc
EG :

```
@BeforeSuite(groups = {"smokeTest","regressionTest"})
public void configBS() {
        System.out.println("======Execute BeforeSuite============");
}
```

Smoke Test:

```
@Test(groups={"smokeTest"})
public void createCustomerTest()
{
        System.out.println("execute createCustomerTest");
}
```

```
@Test(groups={"regressionTest")
public void modifyCustomerTest()
{
        System.out.println("execute modifyCustomerTest");
}
```

➔ In order to invoke grouping execution should declare Group Key in testing.xml file & group keep should be declared before <test>, after <suite> tag

Smoke Test:

```
<suite name="Suite">
    <groups>
        <run>
          <include name="smokeTest"/>
        </run>
    </groups>
 <test name="Test">
  <classes>
   <class name="pac1.ProjectAndCustomerTest"/>
    <class name="pac2.ReportTest"/>
```

```
  </classes>
 </test>
</suite>
```

➔We can invoke multiple groupkey in one XML File

```
<suite name="Suite">
      <groups>
              <run>
                 <include name="smokeTest"/>
                    <include name="regressionTest"/>
              </run>
      </groups>
 <test name="Test">
  <classes>
   <class name="pac1.ProjectAndCustomerTest"/>
    <class name="pac2.ReportTest"/>

  </classes>
 </test>
```

Parallel Execution:

1. **Distributed Parallel execution:**

- Distribute the test case across the multiple test runner & execute each <test>/Test runner in parallel is called Distributed Parallel execution
- we reduce the suite execution time, so that we can get the result early
- in order the archive parallel execution we should enable parallel=”tests” & thread-count=5 in <suite> , then create multiple test runner & distribute the testcase
- maximum thread count is 5
- Thread count should be same as number of <test> testRunner

2 Cross browser parallel execution / Browser Compatibility testing

- Execute same set of testcase in different Browser parallel is called cross browser testing
- To achieve cross browser testing we should we <parameter> in XML file & @parameters annotation inside the testScript

1. **<parameter > is used to specify the browser data for each <test>**

EG :

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Suite" parallel= "tests" thread-count="2">
  <test name="Test-Runner-Chrome">
   <parameter name="BROWSER" value="chrome"/>
    <classes>
       <class name="com.vtiger.comcast.organizationtest.CreateOrganization"/>
      <class name="com.vtiger.comcast.organizationtest.SearchOrgTest"/>

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

  <test name="Test-Runner-Firefox">
  <parameter name="BROWSER" value="firefox"/>
    <classes>
    <class name="com.vtiger.comcast.organizationtest.CreateOrganization"/>
      <class name="com.vtiger.comcast.organizationtest.SearchOrgTest"/>
    </classes>
  </test> <!-- Test -->
</suite> <!-- Suite -->
```

2. @parameters annotation will be used to receive the data (Eg : browser , platform etc) from the XML file to test Scripts [Like BaseClass]

Assertion / CheckPoint

- Assertion is a feature available in TestNG used to validate test scripts expected results
- As per the Rule of the automation every expected result should be verified with Assert statements, because java "if else "statement will not have capability to fail the testNG test scripts
- There are 2 types of Assertions in TESTNG
    1. Hard Assert
    2. Soft Assert

Hard Assertion	**Soft Assertion**
All methods are static in nature	All methods are non-static in nature
It does not allow further execution of test if the line containing hard assert gets failed.	Next steps would be executed even if the line containing soft assertion gets failed.
Whole test case gets failed if at least 1 hard assert fails.	AssertAll() extra lines of code are required to track the fail status.
To verify mandatory fields we go for hard assert	To verify non mandatory fields we go for soft assert

HardAssert

Whenever hardAssert method fails, testNG generate AssertError exception & stop the current test execution & continue execution with remaining test

```
@Test
public void createCustomerTest(){
        System.out.println("step_1");
        System.out.println("step_2");
        Assert.assertEquals("A", "B");
        System.out.println("step_3");
        System.out.println("step_4");
}
@Test
public void modifyCustomerTest(){
        System.out.println("============================");
        System.out.println("step_1");
        System.out.println("step_2");
        System.out.println("step_3");
                                }
```

Out Put

```
step_1
step_2
============================
step_1
step_2
step_3
PASSED: modifyCustomerTest
FAILED: createCustomerTest
java.lang.AssertionError: expected [B] but found [A]
```

Soft Assert

Whenever softAssert mtd fails , testNG Generate AssertError exception & continue execution with remaining steps of same testScript

```
public void createCustomerTest(){
        System.out.println("step_1");
        System.out.println("step_2");
        SoftAssert s = new SoftAssert();
        s.assertEquals("A", "B");
        System.out.println("step_3");
        s.assertEquals("X", "Y");
        System.out.println("step_4");
        s.assertAll();
}
@Test
public void modifyCustomerTest(){
```

```
        System.out.println("==============================");
        System.out.println("step_1");
        System.out.println("step_2");
        System.out.println("step_3");
        System.out.println("step_4");
}
```

Output

```
step_1
step_2
step_3
step_4
==============================
step_1
step_2
step_3
step_4
PASSED: modifyCustomerTest
FAILED: createCustomerTest
java.lang.AssertionError: The following asserts failed:
expected [B] but found [A],
        expected [Y] but found [X]
```

Advantages of Assertion:

- It's used to fail the TESTNG test scripts
- It's used for test scripts validation
- It's generate "AssertErrorException" & reason of the failure + failed line number whenever test is failed
- We can compare any 2 primitive variable or array or Collection or MAP in single line

Framework:

- Framework can be defined as a set of rules and regulations or best practices which we can follow in a systematic way to achieve the desired result.
- It can be also defined as a set of guidelines to be followed while automating the testcases.
- While developing test scripts framework provides a well-organized structure of components which can be reused multiple times in different test scripts, reducing test script development time, and execution time, achieving code reusability.
- The other advantages of using framework are, code becomes more readable, code modification becomes easy as we need not change the required code in every test cases, instead we go to particular class of a specific package to achieve this.
- Therefore, we can say that using framework makes the code altogether an optimized one.

Types of Framework:

1. Data driven framework
2. Method driven framework
3. Modular driven framework
4. Hybrid framework

Components of Framework:

1. Test Data
2. Resources
3. Generic Utility
4. POM Repository
5. Test Scripts
6. pom.xml File
7. Screenshots
8. Reports
9. Build Management Tool/Maven
10. Jenkins

Maven:

It is a Build Management Tool which is used by the developers to create the build and it is used by the test engineers to create the framework, add the dependencies and to execute it from the command prompt and Jenkins.

Advantages:

1. It provides folder structure
2. We can handle dependencies and jar files
3. We can execute code from command prompt
4. It supports CI/CD (Jenkins)

Maven Commnads:

mvn clean : It will clear all the older reports

mvn validate : It will validate entire framework (check for missing jar to download from global repo)

mvn compile : It will compile the entire framework.

mvn test : It will execute the framework or test cases.

Github

It's a distributed cloud decentralized repository where we can maintain our sourceCode / Automation Framwork / CRS doc /build of the application in one place

There are 2 Software in GitREpository

1. **Git HUB** : Cloud based repository(software) , which is used maintain the source code in one place , in order to use it just create an account with https://github.com
2. **Git [Git client]** : it's a software should installed in client machine , which is used to communicate to GITHUB
   EG : Git client Software EGit , GitDeskTop , GitBash

Advantages of GitHud Cloud

1. Since its cloud based repository , no need have maintenance team to maintain the Software / HardWare
2. Cloud means pay rent for what you use
3. Cloud software always access via internet
4. Cloud System / sever physically not present within the Organization, but present virtually
5. Initial investment is not required for Software/ Physical location

6. Scale UP / Scale Down is easy
7. File Share between the team members is easier
8. Jenkins Always get the latest framework from the Git for batch Execution

Why Git is Decentralized Repository?

Git is Decentralized Repository because, in Git before pushing any Code to git Hub, we have to commit the code to local repository first, make sure code is working in Local Repository then push Code to GITHUB(Global Repo)

There are Three stages in Git , start with "Working Diretory"➔Local Repo➔Global Repo

How to create an account in github?

- Go to Google> search for github login
- Click on first name, navigate to https://github.com/login
- Sign-up

How to create a repository in github?

- Login to github
- Go to + dropdown menu and click on new repository
- Give repository name Actitime_OCM32_Framework
- Select checkbox that says "initialize this repository with README
- Click on create repository
- Go to clone or download dropdown and copy the git repository url https://github.com/qspidersseleniumoar/Acttime_OCM32_Framework.git
- username: qspiders.selenium@gmail.com
- password: Selenium1-2

How to share existing framework in eclipse to git?

- Select the project>Right Click>Team>Share Project
- Status of the project should be [NO-HEAD]

How to transfer the framework from working directory to local repository?

- Select the proj.>right click>team>click on commit
- Go to Git staging window, drag all the files from unstaged area to stage area
- Write commit message and click on commit

How to transfer the framework from local repository to global repository?

- Select proj.>team>click on push branch 'master'
- provide url, username and password and follow the procedure

How to get the framework from global repository to local system?

Pre-condition: make sure you have git url, username and password

- Go to eclipse>File>import
- Expand Git folder>Click on project from git
- Click on clone uri>provide username and password, click next and finish