INTRODUCTION

WHEN WE WILL GO FOR AUTOMATION??

- When cost make sense.
- When time need to be saved.
- When you are using repetitive tests.
- When quality is the matter of concern.
- When you need to run multiple test at once.
- As and when we get a new build we go for regression testing which is repetitive in nature. After New build has got released led to go for Automation.

WHAT IS AUTOMATION SCRIPT?

It is a step-by-step procedure of converting manual test case steps into automation scripts with a predefined expected result along with test data by using appropriate automation tools.

WHAT IS TEST SCRIPT?

The program which returned manual testcase.

WHEN WE WRITE THE TEST SCRIPT?

1. Selenium Automation Tool.
    a. Series of test cases will be converted to test scripts in automation tool.
    b. When a new build comes in the staging server we will run the test script all together on to the application installed in staging server and it will gives us the report back.
    c. Report gives summary of extension such as total test cases passed, failed, time taken and skipped.
    d. Report can be shared and it can be analyzed with customer and by ourselves (retrospective meeting.)
    e. Faster execution will be possible by automation.
    f. To see the report, we can give access to customer by Jenkins (C1/CD Tool) which gives the live execution and full visibility.
    g. Customer can see the real time execution but which we can't see in manual testing.
    h. Customer felt more trust worthy for this test report.

WHAT IS SELENIUM?

It is an open source tool which is used to validate your web application on any browser across all the platforms.

WHAT ARE THE ADVANTAGES OF SELENIUM?

- It is open source tool.
- Supports Multi Programming language like-JAVA, C#, PHYTON, RUBBY, PERAL, JAVASCRIPT, PHP, HASKELL.
- Supports Multi browser -SAFARI, CHROME, EDGE, IE, FIREFOX, OPERA.
- Multi OS support-

- For web application like-WINDOWS, LINUX, MAC OS, UNIUX.
- For web mobile like-ANDROID, IOs.

WHAT ARE THE DISADVANTAGES OF SELENIUM?

- ❖ We cannot automate standalone application.
- ❖ If there are any issues in the tool you will not get immediate support.

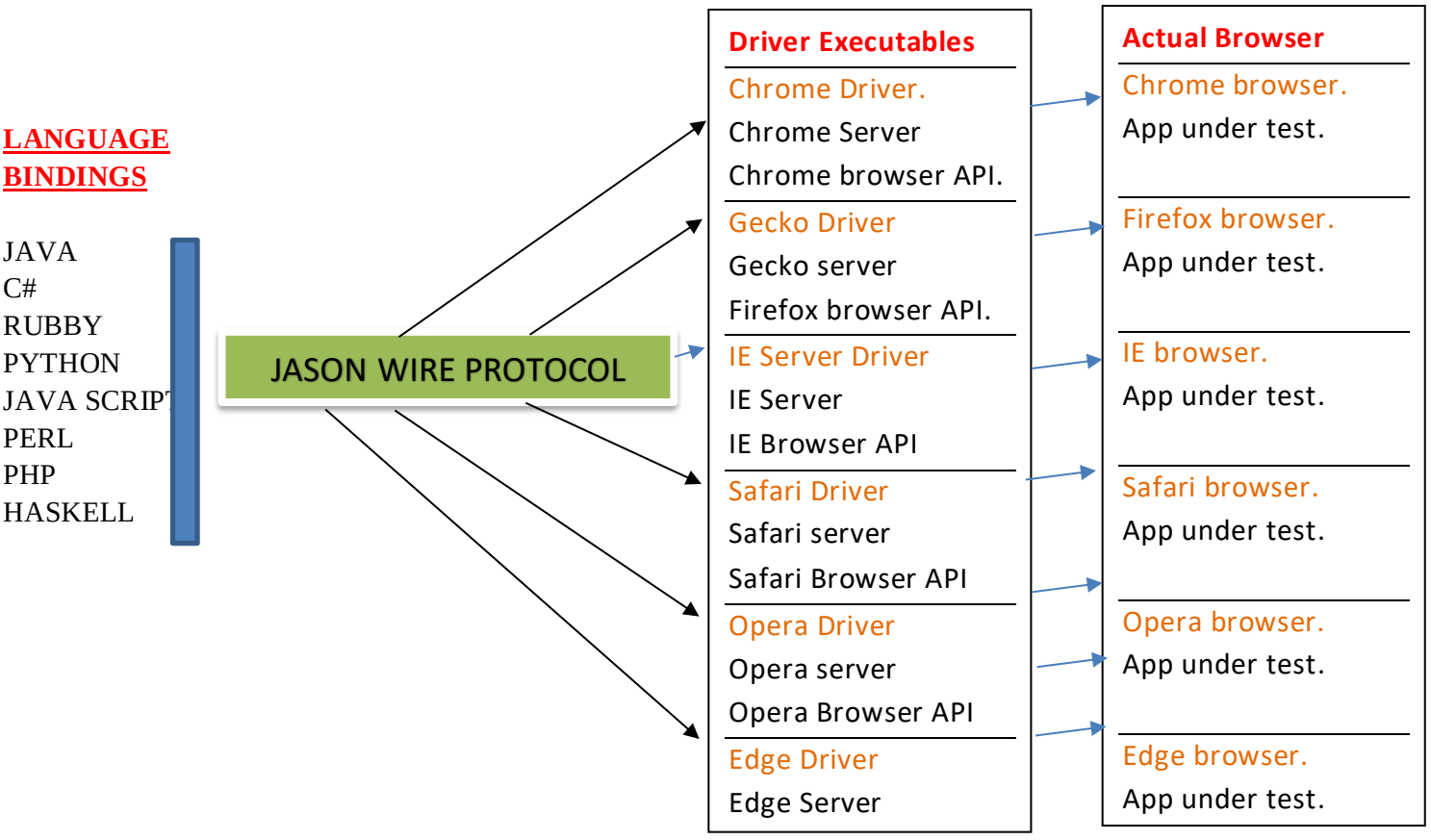
DIFFERENCE BETWEEN QTP AND SELENIUM?

QTP	SELENIUM
It is licensed	It is an open source tool
Supports only windows Os	Supports all major Os
Supports only VB scripting	Supports all major programming language.
Supports all types of application	Support web and mobile application.

WHAT ARE THE TESTCASES WHICH CAN NOT BE AUTOMATED?

-OTP (one-time password)
-user interface.
-Network errors.
-Video and Audio.
-Security Features like Bio-metric, Face ID, Captcha, Auto 2 steps Authenticator.
-100% regression test cases cannot be automated.

GENERIC SELINUM ARCHITECTURE



INTRODUCTION TO SELENIUM GENERIC ARCHITECTURE: -

- ❖ It helps us to understand how the control flow happens inside the selenium tool, science it does not have any UI (user interface).

CLIENT BINDINGS (LANGUAGE BINDING)

-It is language specific selenium libraries which contains all the functionalities of the tools by obelizing the syntax of the compatible language.

- Client= user selected programming language.
- Bindings = selenium inbuilt functions available with respect to the selected programming language.
- Here we know java – so we will have java selenium rich libraries available as tool to perform different actions on the web application which is under testing.
- Here client bindings are available in **.jar** format (jar means java arched).
- Example Java selenium (here we accessing selenium libraries w.r.t java syntax Semitics).

JASON WIRE PROTOCOL

- Json can be abbreviated as JAVA SCRIPT OBJECT NOTATION.
- Its way of text representation of the data (paring), which in turn relatively easy to read and write for the user and as well as for the s/w.
- Here text-based representation is nothing but key and value pair.
- EXAMPLE- {
  COMPUTERS -DELL
  MOBILE - REDMI
  }
- This give the seamless comfort for the structure data to exchange it over the network typically between Server and Web application.

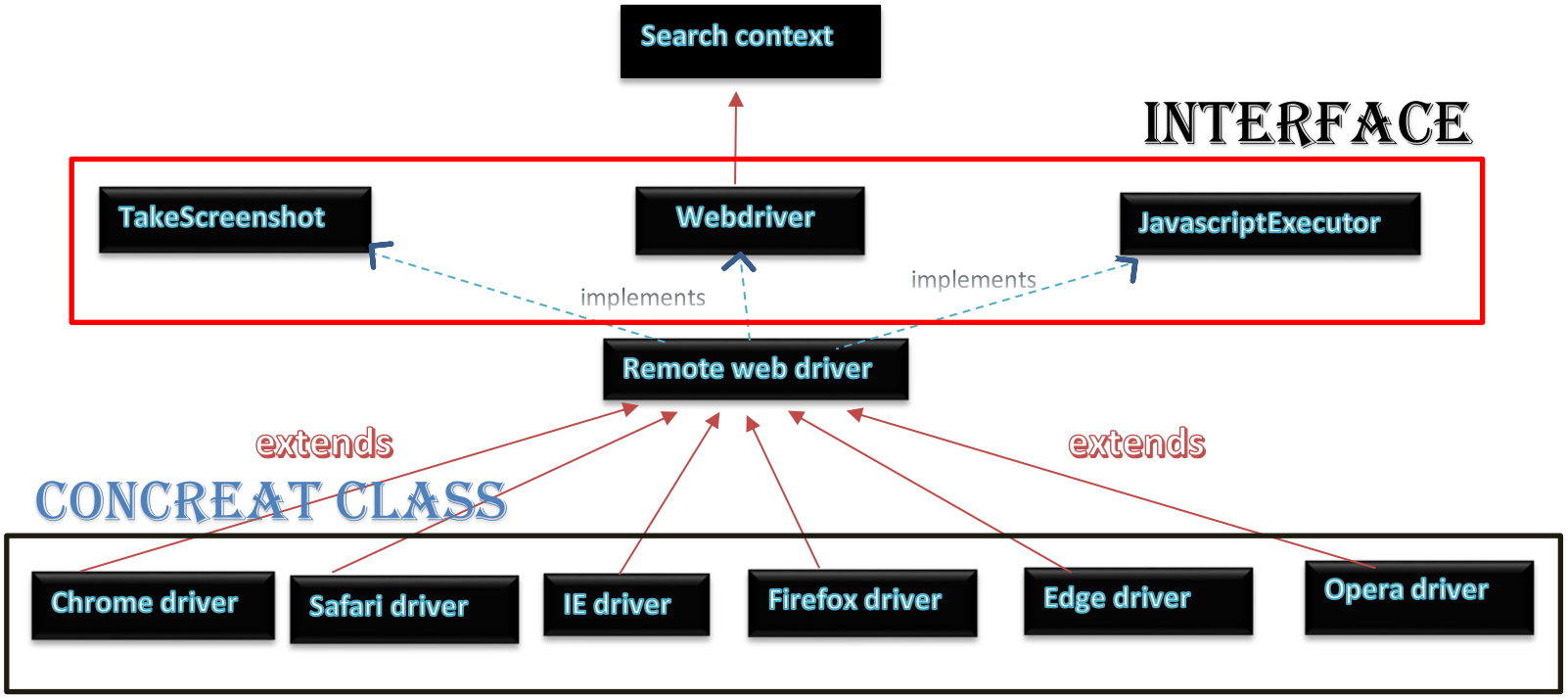
DRIVER EXECUTABLE

- It is nothing but the executable s/w of the particular browser.
- It will act like a many server to trigger the actual browser in the local computer.
- It is always available in an executable format (.exe).
- It will help us to communicate the request to the specific browser, the intern will enable actual browser to perform some action on the target web app which is under testing.

ACTUAL BROWSER

- These are the real browsers or s/w with user interface (UI), where we can access the web app.
- It needs the proper internet connection to process the request and response with respective server of the web app.

SELENIUM WEB DRIVER ARCHITECTURE: -



- Dotted line represents implement keyword.
- Normal line represents extended keyword.

SEARCH CONTEXT-

- It provides search mechanism to search the elements in the web page.
- It is an interface which contains two abstract methods.
- 1)findElement()- to search single element.
- 2)findElements()- to search multiple elements.

TakeScreenShot (Interface)

- It provides a way to take the page or element screenshot of the app under testing.
- It is an interface which contains only one abstract method that is getScreenshotAs().

JavaScriptExecutor(interface)

- It is an interface which contains two abstract methods.
- It provides a way to write java script programs in selenium.
- 1) executeScript() – to write synchronous javascript program.
- 2) executeAsynScript() – to write asynchronous javascript program.

WebDriver (Interface(i))

- It provides a way to control the browser.
- It is an ***interface*** which has 11 abstract methods.
- 1) get().
- 2) getTitle().
- 3) getCurrentUrl().

- 4) getPageSourse().
- 5) getWindowHandle().
- 6) getWindowHandles().
- 7) close().
- 8) quit().
- 9) navigate().
- 10) manage().
- 11)switchTo()
- This is the core interface of selenium without which we cannot start automation because it contains main browser controlling methods.
- WebDriver is an object representing the browser, which can be used to control different browsers.

RemoteWebDriver(Concrete Class): -

- It is an implementation ***concrete class*** for lot of interfaces in selenium and also it is used mainly to run the automations scripts from the host computer to the remote computer.
  (Selenium Grid).

BROWSER RELATED CLASSES-

- Browser related classes are used to perform –
- 1) starting their own servers (Driver Executable s/w).
- 2)Launching their own respective browsers.
- 3) Appling some browser level settings.
- All the 6 browser related classes are concrete classes, here the advantage of this is object can be created or else implementations can be done.

INHERITANCE-

- Browser specific classes – remote web driver- Hierarchical Inheritance – Multiple child having common parent.
- Remote web driver – interfaces (TakesScreenshot/WebDriver/JavascriptExecutor)-Multiple inheritance – RemoteWebDriver Concrete class is providing implementations for all the 3 interfaces.
- RemoteWebDriver – WebDriver – SearchContext – Multi level inheritance.

LAUNCHING EMPTY CHROME BROWSER

STEP 01: To step the driver Executable Path

- System.setProperty(String Key , String value) – Method.
- Here System is the java.
- This method will return string.
- This method will take 2 inputs/parameters.
- Key represents – Which browser?
- Value represents – Which Browser?
  Ex:
  System.setProperty("webdriver.chrome.driver","./drivers.chromedriver.exe");

  It's the best practice to store the driver software within the project directory and provide the relative path.
  DOT represents the current project.

KEYS FOR DIFFERENT BROWSERS

"webdriver.chrome.ddriver"-Google Chrome
"webdriver.gecko.driver"- firefox
"webdriver.ie.driver"- InternetExplorer
"webdriver.edge.driver"- Microsoft Edge
"webdriver.opere.driver" – opera Browser
"webdriver.safari.driver" – safari browser

Step 02: Instantiating the browser specific class

- We need to create the object for the specific class to instanciate.
- ChromeDriver – it is the built in class available in selenium library.
- It is a concreat class having lot of non static methods through which we can perform automation on the browser.
- ChromeDriver driver = new ChromeDriver(); →Object creation statement.
- NOTE: To use the browser related classes we should import them from the selenium package i.e.org.openqa.selenium.chrome.*(browser);
- If we had not set the path of the driver executable file properly then we will get IllegalStateException. Of java.lang package. It is an unchecked Exception.

Unchecked Exception

Take viv notes and complete

1. ClassClassException
2. InputMismatchException
3. ArrayIndexOutOfBoundsException
4. AirthematicException
5. NullPointerException

6. NumberFormatException
7. NoSuchElementException
8. MissingResourseException
9. IllegalStateException

Checked Exception

- The exceptions which are checked during compile time is known as Checked Exception
- Try catch block or by using throws keyword we can handle checked Exception.

  IO exception
  1. ClassNotFound
  2. InvocationException
  3. FileNotFoundException

Web URL

- Web uniform location (URL): uniquely identified the specific web resource inside the web application.
- Every web application should have its unique address in the form of URL.
- URL is one and only way to access web application via browser.
- http://localhost:8080/login.
- https://google.com/search?q=iphone11.
- Maximum no of character in the URL is 200.
- http://172.217.160.142/search?q=iphone11.

PROTOCOL

- Protocol is common language where two applications exchange information with each other.
- When an application wants to communicate with each other (in our case browser & server), There need to be a common language where both applications can understand each other.
- This language is known as protocol, where protocol has set of rules and instruction.
- Browser always sends a request & receive response via http protocol hence it is called http request / http response.
- It is an optional information and case insensitive.

TYPES OF PROTOCOL

- http
- https
- ftp
- smtp,etc

DOMAIN

- Uniquely identify the specific computer in the network in which web application is present.
- Domain name might be computer name or IP address.
- It's a mandatory information and case sensitive.
- .com

- .gov.in
- .org
- .edu

PORT: -

- Uniquely identify the specific s/w or application inside the computer.
- In the URL this an optional information.
- http Tomcat 8080
- http Joss 80
- http WebLogic 123
- https Tomcat 8443
- https jboss 443
- https WebLogic 456

PATH: -

- Uniquely identify the specific web resource inside the application Server (web application).
- W2 1e Know web application is a collection of web resources, so its's full of paths i.e. web resources at the web application side.

FARGAMENT ID-

- Identify the specific fragment/section in the webpage.
- It's an optional information in the URL.

WEBDRIVER

get()-

signature: public void get(String URL){
}.
Usage: driver.get("Fully qualified main URL")

- it is used to navigate main URL of the application.
- URL should be passed in the string form.
- The URL should be fully qualified (with its protocol).

getTitle(): -

signature: : public String getTitle(){
}.
Usage: String CapturedTitle = driver,getTitle();

- This method is used to get the webpage title.
- It returns the webpage title in the string form.
- This method will be used to verify the title of any webpage of the application under testing.

contains(CharSequence…ch): -

- contains() – method.
- It will check whether the condition is true/not.

- Equals/not equals
- Return type of this method is Boolean.
- Used to compare two strings.

4)getCurrent(): -

signature: public void getCurrentUr(){
}.
Usage: currentUrl=driver.currentUrl().

- It is used to fetch the current webpage URL.
- It gives the URL in the string form.
- It is used to verify the URL of the webpage.

getPageSource()

signature: public String getPAgeSource(){
}.
Usage: pageSource=getPageSource().

- It will fetch the web page source.
- It will be used to verify whether the webpage source contains certain text or elements etc.
- It return the resource in the form of String.

WEBPAGE

- The major building block of any webpage is HTML, CSS, javascript.
- Every browser will understand javascript, Html and CSS by default.
- Hypertext Markup Language (html) – it is used to display the elements in the structured format.
- Cascading Style Sheets (CSS) – It is used to add styling to the web elements.
- Javascript (Live Script)- It is programming language, it will perform client-side validation (email id format/credit card number).
- Through JavaScript we can do machine learning, science it's a live script.

getWindowHandle(): -

signature:
public String getWindowHandel()
Usage:
String windowIdUnderControl=driver.getWindowHandel().

- This method is used to get the window id.
- Current Window ID mean, the current window on which the driver control is available.
- Initially the driver control will always lie in the parent window and will get the same window id.

- It gives the window id in the string form.

getWindowHandles(): -

signature:

```
public String getWindowHandels()
```

Usage:

```
Set <String> allWindowIDs=driver.getWindowHandels().
```

- This method is used to get the multiple window IDs.
- Sometimes while we are automating we might come across multiple windows, by using its window ID we can transfer driver control and we can perform different actions on windows.
- When we call this method, it captures all the window IDs and stores it in the set <string>.
- Iterate using for each loop over this set to capture each and every window ID iteratively.

quit(): -

signature:

```
public void quit(){
                    }
```

Usage:

```
driver.quit().
```

- It will close the browser window which are open from the current selenium secession.
- After closing all the browser window, it will stop the server (driver executable file).
- It always the best practice to use this method as forced condition
- You can check whether it stops the server or not in task manager.

close(): -

signature:

```
public void close(){
        }
```

Usage:

```
driver.quit().
```

- This method will close the current browser window.
- NOTE: close() will only close the current browser window among the group of windows. As initially the driver will be pointing to parent window, so close() will have control to close only that parent window.
- NOTE: we can also transfer the driver control to other windows. In such scenarios depending on where the driver control is available, that browser window will get closed.

Close()	Quit()
It will close the current browser window.	will close all the browser windows.
Close() will not stop the server.	quit() will stop the server.
We should not use close() as post condition	We should use quit() as the post condition.

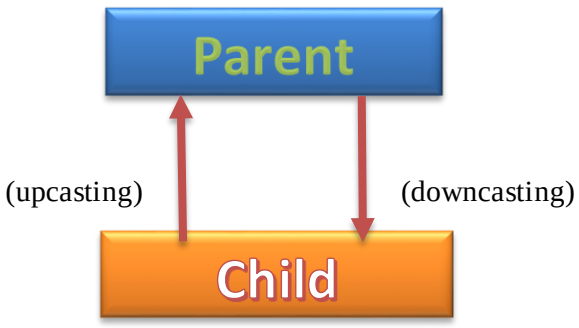
Navigate(): -

signature:
public Navigation navigate()

- Navigate() is used to perform browser history navigation like back, forward, refresh.
- Navigate() is also used to navigate to the sub URL of any web application.
- It returns navigation type object.
- Navigation is an interface in selenium which contains 5 Abstract methods.
- RemoteNavigation is the concrete class, it will provide implementation for all the 5 methods
- back() – void
- forward() – void
- refresh() – void
- to(String URL)- void
- to(URL url)- void

navigate(): -

- navigation is an inner interface of WebDriver.
- Remote navigation is an inner concrete class of RemoteWebDriver Class.
- If return type of any method is parent type then that method can return the child objects as well, so while returning auto upcasting will happen.
- Public Fruit getFruit()
  {
  Return new Apple();
  }



- EX: attachment, Here getFruit() Is returning Fruit type object. So, here fruit is the parent and Apple is the child. Auto Upcasting will happen while Fruit type object is returned.
- Public Navigation navigate()
  {
  Return new RemoteNavigation();
  }
- Navigate() internally creates the object of RemoteWebDriver class and returns it in the upcasted form yo its parent Navigation.

back(): -

- Signature : public void back()
- Navigates to immediate Previous page in the same session.

- Usage: - driver.navigate().back();

forward(): -

- Signature: public void forward(){}
- Navigates to immediate next page which we loaded.
- Usage: - driver.navigate().forward();

Refresh(): -

- Signature: public void refresh()
- Reloads the same webpage one more time.
- Usage: driver.navigate().refresh();

to(String url): -

Signature:

```
public void to(String url)
```

Usage:

```
driver.navigate().to("sub URL");
```

- This method is used to navigate to the sub url of the application get() method will be used.
- If any test case doesn't mention how to any sub page of the application then we can directly go to the webpage by using the url of the page using navigate().to()method.
- But internally coding of get() and navigate().to() is same, but only usage is different.

What is the difference between get() and navigate().to()??

There is no difference between these two methods implementations. We can find the difference in the usage. As a best practice get() is used to navigate to main url and navigate().to() is used to navigate to sub url.

Get()	Navigate()
It is used to navigate to main url of the application i.e. we can use it only for url navigation.	It is used to navigate to the sub url of the application and also we can perform browser history navigation's like back, forward and refresh.
The return type is void	the return type is navigation type object

Navigate().to() (Ex: Method overloading): -

- This method is used to navigate to sub url of the application as the best practice.
- Generally to navigate to the main url of the application get() will be used.
- Whenever any testcase does not mention how to navigate to any sub page of the application then directly we can go to the web page by using the url of that page.

NOTE: internally coding of get() and navigate().to() is same but only usage is different.

- Public void to(String URL) {}.
- Public void to(URL url) {}.

to(URL url): -

- This method accept URL concrete class object as an input.
- URL class concrete class of java.netpackage.
- Using URL class we can compose the url's is different ways.

- URL constructor throws a checked the url's is different ways.
- URL constructor throws a checked exception called as MalformedURLException.
- MalformedURLException is a checked exception of java.netpackage, which will occur whenever the protocol is not specified while URL creation.
- As it is checked exception we can handle it during compile time itself either by using try, catch block or by using throws keyword.
- URL- Uniform Resource Locator.
- Dynamically composed URL will be used here by using URL Class we have multiple Constructors.

Java Concepts

- Constructor overloading – same name different argument under the same class object creation.
- Here java URL Class provides different constructor and we witness constructor overloading.

Manage(): -

- Signature: public Option manager()
- Usage opt=driver.manage();
- Manage() used to perform the following operations.
    1. Window related operations.
    2. Timeouts related operations.
    3. Cookies related operation.

Window related operations.

- The following are the operations we will perform under this: -
    1. Maximize the browser window – maximize().
    2. fullScreen mode of the browser window- fullScreen().
    3. Get current size of the window – getSize().
    4. Get current position of the window – getPosition().
    5. Move the window – setPosition().
    6. Re-size the window – setSize();
    7. To minimize the browser window - minimize().

maximize().

- Signature: public void maximize().
- Usage: driver.manage().window().maximize().
- It is used to maximize the browser window.
- It will be used as the precondition whenever we used write automation script.

fullScreen().

- Signature: public void fullScreen().
- Usage: driver.manage().window().fullScreen().
- It is used to open the browser window in full screen mode.

getSize(): -

- Signature: public Dimension getSize().

- getSize() used to get the current dimension of the browser window.
- i.e. the height and width of the browser window.
- This method returns the Dimension class object.
- Dimension is the concreate class of the selenium package which contains two method getWidth() and getHeight().
- Usage:
- Dimension size=driver.manage().window().getSize();
- Int height=size.getHeight();
- Int width=size.getWidth();

Minimize(): -

- Signature: public void minimize().
- Usage: driver.manage().window().minimize().
- It is used to minimize the browser window.
- It will be used as the post condition in the automation script.

setSize(): -

- Signature: public Dimension setSize(Dimension targetSize).
- Usage:
- Dimension targetSize=new Dimension(int height, int width);
- Dimension size=driver.manage().window().setSize(targetSize);
- This method is used to re-size the browser window
- This method accepts Dimension class object as input.
- So, we should create the dimension class object with the target width and height and later we should call the setSize() and supply the created Dimension type object as an input.

getPosition(): -

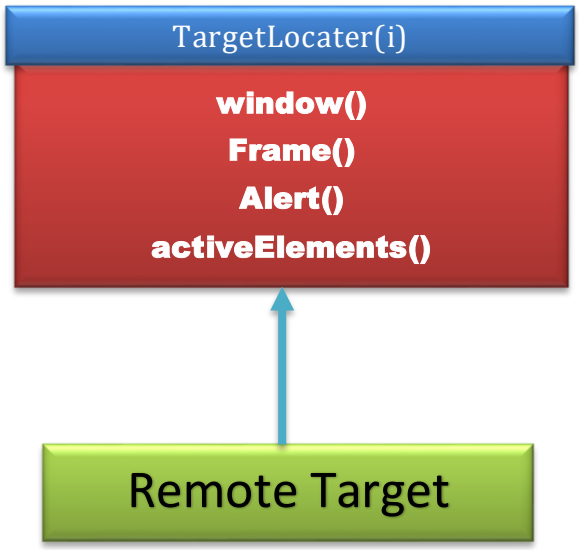
- Signature: public point getPosition().
- Usage:
- Point position= driver.manage().window().getPosition();
- Int startx=position.getx();
- Int startY=position.gety();
- getPosition() is used to get the browser window X and Y coordinates.
- getPosition() returns the point class object through which we can get the X and Y coordinates by calling getX() and getY().
- getX() will return the startX value and getY() will return the startY value.

setPosition(): -

- Signature: public void setPosition(Point targetPosition)
- Usage:
- Point targetPosition=new Point(int x, int y);
- Driver.manage().window().setPosition(targetPosition);
- This method is used to move the browser window.
- setPosition() accepts point class object as input.
- So, first we should create the point class object with target x and y coordinates and later we should call setPosition() and supply the point object as input.

switchTo(): -

- Signature: public TargetLocater switchTo()
- Usage: driver.switchTO().window(String windowID);



1. This method is used to transfer the driver control to –
    - Window
    - Alert pop-up
    - Frame
    - Active element of the webPage
2. This method returns the target locater type object.
3. To transfer the webdriver control to the window(), we should use window method and pass the window ID to it.

OOAD(OBJECT ORIENTED ANALYSIS AND DESIGN): -

Why WebDriver driver = ChromeDriver();

We should always program with the at most possible interface , according to the developers of selenium store the object in the remote webdriver reference.

- We should not store in SearchContext, TakeScreenShot & JavaScript Executor.
- We should not store the object in class reference, so here we should not store in browser class ref
- And RemoteWebDriver class ref.
- So we should not store the object in SearchContext, TakeScreenShot, and JavaScript Executor interface ref because all the method in WebDriver type ref object will not be accessible.
- Here the utmost possible interface is WebDriver.
- In selenium document, the selenium developers have also suggested that all browser objects should be stored in WebDriver ref because of OOAD principle and also WebDriver ref because of OOAD principle and also WebDriver is the core interface of the Selenium.

HTML(HYPER TEXT MARKUP LANGUAGE): -

```
<html>
        </head>
         <body>
                  <input type=”text”>
                  <button type=”submit”>click Me!!</button>
                  <a href=”https://www.google.com”>Google</a>
                  <input type=”radio”>
                  <input type=”password”>
                  <input type=”checkbox”>
                 <span>Mohan</span>
        </body>
</html>
```

INTRODUCTION

- Hypertext Mark-up language.
- It is used to display the content in the structured format.
- It is not case sensitive.
- All the codes are considered as tags.
- Tag will be represented with angular brackets [].
- It is nested language / it has tree structured.
- All the HTML program will be saved with Filename.html extension.
- The root tag of HTML program is <html>.
- Majority of the tags will have closing tag.

What is a tag??

1. The angular bracket in html program is called as tag.
2. All the html program elements are called as tags (something like tokens in java).

What is a tag name??

- The first word within the tag is called tag name.
- Ex: <html><input>,<a>.

What is an Attribute??

1. The key and value pairs within the tag is known as attribute
2. A tag can have zero or more attributes.
3. Ex: <input type=“text” placeholder=”enter”>
4. Ex: <a href=” https://www.google.com”>

What is an Attribute name??

The key of an attribute is called Attribute name.

Ex:<div id=”login-layer” name=” tag”>

What is Attribute value??

The value of the attribute is called Attribute value.
Some attributes may or may not have values.
Ex: <input type=”text” placeholder=” enter” value””>

What is text??

Any thing which comes in between opening and closing tag is considered text.
Ex:Hello.
welcome to india .

What is link text??

Anything which comes in between opening and closing tag associated with href attribute within <a> is called as link text.
Ex: Google.

What is normal text??

Anything which comes in between opening and closing of any tag except <a>tag is called as Normal Text.
Ex: Hello.

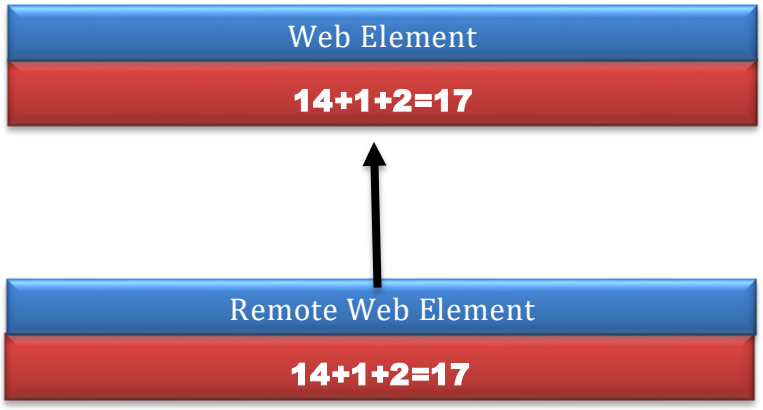
How to perform action on any element within the webpage??

- Search / locate the element.
- Perform action on that element.

What is Web Element??

- Generally anything on the webpage viewport area is considered as webelement.
- Viewport area means visible area of the web page.
- Ex:- Button, radio Button, Check Box, image, link, Text Field, etc.
- From OOPS perspective webelement is an interface which contains 14 Abstract Methods. These 14 Abstract Methods are also called as action methods which can be perform action on the web element or to retrieve some information of the web element.
- All the 14 Abstract Methods are implemented in remote web element concreate class.
    1. click().
    2. clear().
    3. sendKeys().
    4. Submit().
    5. getSize().
    6. getLocation().
    7. getRect().
    8. getTagName().
    9. getCssValue().
    10. getAttribute().
    11. isEnabled().

12. isSelected().
13. isDisplayed().
14. getText().



How to inspect HTML source code??

1. Right click on target element.
2. Click on inspect.
3. Press f12 key.
4. Click on element tag.
5. Click on mouse pointer icon.
6. Place the mouse cursor on the target element and click on it.

POSSIBLITY NO 2

1. C lick on setting by …on top right of the window.
2. Click on more tools.
3. Click on developers tools.
4. Click on elements tab.
5. Click on mouse pointer icon.
6. Place the mouser curser on the target element and click on it.

POSSIBLITY NO 3

1. Press CTRL + Shift + i.
2. Click on elements tab.
3. Click on the mouse pointer icon on top left of source code.
4. Place mouse curser.

LOCATERS

INTRODUCTION

- Locaters are the search criteria through which we can find the elements in the web page.
- Locaters are also called as selectors.
- Selenium provides 8 locaters strategies to find the element in the web page they are:
    1. id
    2. name
    3. className
    4. linkText
    5. partialLinkText
    6. tagName
    7. cssSelector
    8. xpath

NOTE:

1. Here id, name and class are the attribute names in the source code of any web element.
2. cssSelector and xpath are the Expression which can be used to fin the element.
3. Selenium developers provided a class called By in which all the locaters related methods are available.
4. By is an abstract class which contains 8 static methods which can be used to find the element in the web page.

```
By abstract class By
{
    Public static By name(String nameValue)
    {
        =========
        =========
    }
    Public static By tagName(String tagname)
    {
        =========
        =========
    }
    Public static By linkText(String linkText)
    {
        =========
        =========
    }
    Public static By partialLinkText(String linkText)
    {
        =========
        =========
    }
    Public static By cssSelector(String expression)
```

```
    {
        =========
        =========
    }
    Public static By xpath(String expression)
    {
        =========
        =========
    }
}
```

- To use all the static methods we should use it in findElement() or findElements() as shown below.
- By.id("idvalue");
- By.name("namevalue");
- By.className("classvalue");
- By.tagName("tagname");
- By.linkText("fulllinktext");
- By.partialLinkText("full / partial linktext");
- By.cssSelector("cssExpression");
- By.xpath("xpath Expression");

What is findElement()….??

- It is an abstract method of SearchContext interface which is used to
- If the element is found, it will return the address of that element or in other words if the element is found it will return the WebElement interface type object using where we can perform some actions on it.
- WebElement element = driver. findElement(By.id("LoginButton");
- If the element is not found with the given locater strategy then findElement() will throw NoSuchElementException.
- If the given locater strategy is matching with the multiple elements then findElements() will return the first matching element.
- Signature: public WebElement element= driver.findElement(By.("username"));
- Element.clear():
- Element.sendKeys("admin");
- Ex:→ driver.findElement(By.id("LoginButton")).click();
-

NoSuchElementException

- It is an unchecked exception of selenium which will be thrown:
- If the locater strategy is incorrect.
- If the element is not loaded (due to slow internet connection)

NOTE

- Whenever the browser object to webdriver interface type reference then we can access only web driver interface methods and its parents i.e., SearchContextInterface .
- Webdriver driver = new WebDriver();
- We are achieving this by auto upcasting.
- Other methods will be available in the browser object due to inheritance but not accessible through WebDriver reference.
- ChromeDriver driver = new ChromeDriver();
- So to access them we need to downcast to their interface type.

PRIORITY OF LOCATERS BY USING findElement()

- Id
- Name
- linkText
- partialLinkText
- cssSelector
- xpath

NOTE:-

- generally By.tagName() and By.className() are used to identify multiple elements because of this the chances of duplicates are more.
- By.classNAme() won't identify the element if the attribute value is having a space and it gives InvalidSelectorException.

NOTE: By.linkText()

- This locaters strategy will be used to identify the <a> Since the link is large enough I its length that will be represented with the name of the link, we call it as linkText.

NOTE: By.partialLinkText()

- Here also link will be stored within then<a> because the name of the link is dynamic in nature this leads to the usage of By.partialLinkText() locater.
- Ex:- in Gmail application, Inbox(31),Inbox(41)

NOTE: By.tagName()

- This input is more suitable for findElements(), because w.r.t tagName there are lot of duplicates will be their.

NOTE: By.cssSelector()

Css stands for (CASCADING STYLE SHEETS) it is a language which is used to apply styling to the web elements like :

1. Decorating the group of text with the same color, font face, style, etc.
2. Applying the background color for group of buttons.
3. Applying the mouse overring color for the group of elements.
4. Css is embedded along with html.

5. CssSelector is a query language which is used in CSS to identify group of elements in html on which we can apply common styling.
6. As in automation also we need to identify the web elements by using CSS selector.
7. Whenever the source code of the web element does not have id, name and class attributes and if the web element is not link text then to search that element we should use css selector.

- Syntax:→ tagName[attName='attValue']
    - Ex: input[type='submit']
      Button[id='btn1']
      Input[class='textField']
- Shortcut syntax to id in cssSelector: - tagName#idvalue
    - Ex:- button#btn1
      Shortcut syntax to class in CSS selector: -
      tagName.classvalue
- Ex03:input.textfield

DISADVANTAGE OF cssSelectors

- cssSelector does not support text function (i.e., if the source code of the WebElement is not having any attributes just the text then we cannot use cssSelector)
- usage Ex: -
- driver.findElement(By.cssSelector("input['text']")).sendKeys("admin");
- we cannot perform backward traversing using cssSelector.

ADVANTAGE OF CSS SELECTOR

- it is faster compare to xpath.
- We can identify shadow doc elements using cssSelector.

What is an InvalidSelectorExcption???

- It is an unchecked exception of selenium which will occur when syntax of cssSelector / xpath is incorrect.
- NOTE: it a best practice to verify the css and xpath expressions in the browser itself before using it in the automation script.

1. To verify the expressions in chrome:
    - Inspect the target element.
    - CTRL+f in the elements tab.
    - Then write the expression and verify in the search string, selector and x-path textbox.
2. To verify the expression in the firefox:
    - Inspect the target element.
    - write the expression and verify in the search HTML text field.

XPATH(XML path)

1. Xml can be abbreviated as extensible markup language.
2. XML is like HTML language but it is used as a standard format for data exchange between client and server and between two different applications.

3. Xpath is a query language which is used in XML to identify the XML elements. As the XML and the HTML are having same program elements we can use xpath in html also.
4. Xpath is the path or address of the web element in the HTML tree structure.

WHY XPATH…??

1. If the source code of the element not having any attributes, but just the text, we should use xpath.
2. If the elements are completely dynamic oe if there are duplicate elements we should use xpath.
3. If the source code of the element is not having id, name or class attributes and if their was no link then we should use xpath.

TYPES OF XPATH

Based on where the xpath starts, it is classified into two types: -

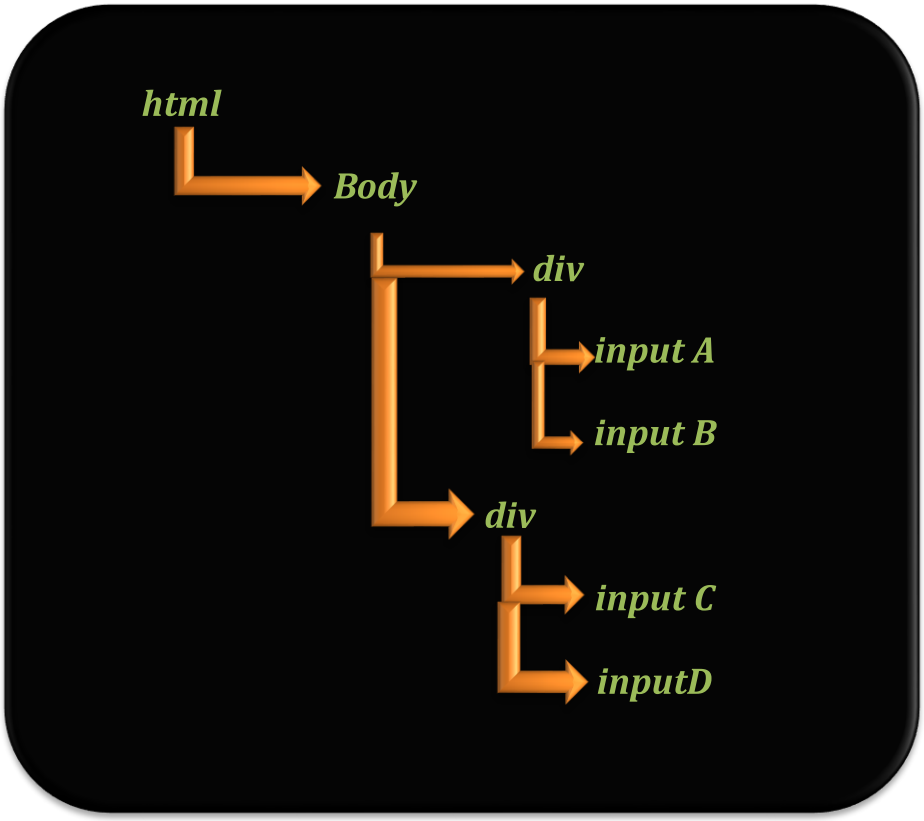
1. **Absolute xpath**
2. **Relative xpath**

Absolute xpath

1. If the xpath expression starts from the root of the HTML doc then it is called as absolute spath.
2. Here we use two symbols DOT(.) and Forward Slash(/).
3. Dot(.)means current document or current webpage.
4. Forward Slash(/) means immediate child.
5. If the xpath matches with the multiple elements then the xpath will store those elements in the array

```
<html>
    <head>
        <title>For Absolute Path</title>
    </head>
    <body>
        <div>
        <input type="text" value="A">
        <input type="text" value="B">
        <div>
        <div>
        <input type="text" value="C">
        <input type="text" value="D">
        </div>
    </body>
</html>
```

structure which is xpath ArrayAlways xpath Array index starts from 1.



A →html/body/div[1]/input[1]

B →html/body/div[1]/input[1]

C →html/body/div[2]/input[1]

D →html/body/div[2]/input[1]

AB →html/body/div[1]/input

CD →html/body/div[1]/input

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

NOTE: ./ is optional while writing absolute xpath.

DISADVANTAGES OF THE ABSOLUTE XPATH:

1. It is very length and time consuming because as we write the path from the root of the document.
2. Here there are more chances of using index. So, if the position of the element changes then we need to modify the index used in the xpath again and again which needs more maintains effort.
3. So for above two reason absolute xpath is not used in real time.

RELATIVE XPATH.

- If the xpath starts from any child of the document or webpage then it is called as relative xpath.

TYPES OF RELATIVE XPATH

- Core Relative Xpath.
- Xpath by attribute.
- Xpath by text function
- Xpath by contains functions.
- Independent and Dependent Xpath.
- Xpath by axis.
- Xpath by Group Index.

Core Relative Xpath: -

- Here we use '//' which means any child or child of any parent.
- Generally the relative xpath starts from // and in between it may contain the combination of // and /.
- Compared to absolute xpath relative xpath is shorter. So it will consume much time to write.

```
<html>
    <head>
        <title>Relative xpath</title>
    </head>
    <body>
        <input type="text"value="A">
        <input type="text"value="A">
        <input type="button"value="A">
    </body>
</html>
```

A-----//input[1]

B-----//input[2]

C-----//input[3]

AB-----//input[1]//input[2]

BC-----//input[2]//input[3]

ABC-----//input

- Here also we may need to use index to identify the elements uniquely, in such cases the relative xpath should not be as their will be too much of maintenance efforts if the position of the element's changes.
- In real time this type of xpath used to identify multiply elements.
    - Ex: Identify all the links of webpage, images.
- ***NOTE***: IT IS ALSO USED TO IDENTIFY THE FIRST ELEMENT OR LAST ELEMENT OF THE WEBPAGE.

DRAWBACKS

- Here also we have to use the index to identify the element uniquely, in such case correlatively xpath should not be used as their was too much of maintenance effort is needed.

!!!!!!!!!!!!!!!!INTERVIEW QUESTIONS!!!!!!!!!!!!!!

WHAT IS THE DIFFERENCE BETWEEN '/' AND '//'.....??

- / - immediate child.
- // - any child or child of any parent.

Write any xpath to identify all the links in the webpage...??

- //a.

What is the difference between //a and //table//a...??

- //a – all the links of all the tables (all the links of all the tables).
- //table//a – only the links of all the tables (all the links of all the tables).

What an xpath to identify all the images present in the webpage...??

- //img.

What is the difference between //img and //div//img....??

- //img → it is the total images available in the web page.
- //div//img → Total images available among all the div tags.

Write an xpath to identify all the links and images present in the webpage??

- //a|//img.

What is damage between //a and //a[1]...?

- //a→ all the links of webpage.
- //a[1] → will identify all 1st links of any parent.

XPATH By attribute: -

- Writing a relative xpath by using attribute is called as xpath by attribute.
- General syntax:
  //tagName[@attributeName='attributeValue']
- @ is compulsory while writing xpath by attribute.
- <button type="submit">sign in

Xpath by using AND/NOT functions.

- //input[@type='text' and @value='A']
- //input[@type='text' and @value='B']
- //input[not(@type='button')]
- //input[not(@type='text' and @value='A')]
- While writing xpath we can use multiple attributes together by using and ,OR, Not function

Drawbacks of xpath by attribute

- If the source code of the WebElement is not having any attributes then we can not use xpath by attribute.
- Ex:→<button>Register</button>
- If the attributes are having duplicates then we cannot use xpath by attribute.
- Ex:→most of the time the class attribute within any tag contains duplicates .

Xpath by text()

- <html>
  <body>
  Fruit

  Fruit

  </body>

  </html>

What is xpath by text()…????

The process of deriving the address of the web element by using its text is known as xpath by text().

Why we go for xpath path by text()…???

- If there are 0 attributes but there are only text is available for that web element then we should use xpath by text function.
- Example: <button>Submit</button>

Syntax: //tagName[text()='CompleteTextValue']

in the above syntax text value is :-

1. Case sensitive
2. Should be the complete text
3. It allows both the normal and link text.

Syntax ://tagName[.='textValue']

- Dot function will identify both the tag text and also sub tag text.
- Xpath to identify both the fruit and apple in the current web page is
  //span[.='Fruit']
- Text function will identify only tag text
  //span[text()='Fruit']

Drawbacks of xpath by text()

- If the tagtext contains unbreakable space the xpath by text() will not work.
- If the tagtext of the WebElement is partially dynamic then we should not use xpath by text(), because every time when the partially dynamic text changes we need to modify the xpath which involves more maintenance effort.
- **Example**: - actiTIME 2020 online

Partial Dynamic text

<nobr> actiTIME 2020 online</nobr>

//nobr[text()='actiTIME 2020 online']

  - Here though the above xpath works fine now, when the developer changes the year to 2021 then the xpath will not work. So we should not use xpath by text function in the above scenario.
  - If the text of the target WebElement is very lengthy the by default xpath will be lengthier. In such cases if we have to same give the same partial text, text() will not work because it needs complete text. So it is not the best practice to use next function in such cases.
  - **Example: - //td[text()='Shree Dharmasthala Manjunatheshwara Engineering College,All rights Reserved'].**
  - The above xpath is very lengthy

Xpath by contains()

- Writing the xpath to identify the target WebElement based on its partial text and partial attribute value Is called as xpath by contains function.

Types of Xpath by contains()

- Contains by text
- Contains by attribute

Contains by text(): -

Syntax: -01) //tagName[contains(text(),'partialTextValue')]

- Text function will select the current tag text value

Syntax: -02) //tagName[contains(.,'partialTextValue')]

- .(Dot) will select all similar tags having the same text value.

Contains by partialAttributeValue

- Writing the xpath using partial attribute value can be considered as xpath by contains function using attribute.
- **Syntax: -01)**: - //tagName[contains(@attributeName,'partialAttributeValue')]

WHY OR WHEN WE SHOULD USE XPATH BY CONTAINS FUNCTION BY USING ATTRIBUTE..??

- Whenever the attribute value is partially dynamic.
- When the attribute value is very lengthy.

Drawbacks of x-path by contains function

- If the text of the target WebElement is completely dynamic then we should not use contains function because if the text changes then we need to modify the x-path which involved more maintenance effort.

Note: - Completely dynamic text means the text which changes completely always.

https://www.irctc.com/Contact.html

observe the table data such as phno , mailId of the different trains.

Independent/dependent X-path

It is the procedure to identify completely dynamic/duplicate elements using other types of x-path and also by using backward Traversing (/..).

Why Independent/dependent X-path..???

- To identify completely dynamic/duplicate elements we will use Independent/dependent x-path.
- Here Independent means fixed element and Dependent means dynamic element.

Steps to write Independent/dependent X-path

1. Create a question or scenario
2. Identify which element is fixed and which is dynamic
3. Write xpath to identify the fixed element.
4. Update the xpath to identify the immediate common parent by using backward Traversing (/..).
5. Update the xpath to identify the dynamic element.

Example: - 01) https://www.irctc.com/Contact.html

1. Check for the Buddhist train helpline no
2. Step 01) Write an xpath for Buddhist train helpline no
3. Step 02) identify which element is fixed and which element dynamic. Buddhist train is the fixed element. Help line no is Dynamic.
4. Step 03) Write the xpath to identify the fixed element??

a. //td[text()=' Buddhist train']

5. Update the xpath to identify the immediate common parent by using backward Traversing (/..).
   a. //td[text()=' Buddhist train']/..

Update the xpath to identify the dynamic element

- //td[text()=' Buddhist train']/../td[@data-title='helpline']

Here the xpath written for the WebElement itself is having duplicate's. That means the address itself has got duplicated because of similarity In the fixed element which we have chosen for the corresponding dynamic element.

This is the clear situation / scenario where we cannot identify the element by using independent and dependent x-path approach.

Even if we identify, it is not reliable.

X-path by group index

Syntax: (xpath)[indexValue]

What is X-path by group index..??

Writing relative x-path to identify unique element among group of duplicates using is called as X-path by group index.

Why we should use X-path by group index..??

1. Identify the REDMI 9i Sport (Metallic Blue, 64 GB)price of Rs 8485/-
2. Go to flip kart website and search for REDMI 9i Sport (Metallic Blue, 64 GB)price of Rs 8485/-

X-PATH-→ //div[text()='REDMI 9i Sport (Metallic Blue, 64 GB)']/../.. //div[text()='REDMI 9i Sport (Metallic Blue, 64 GB)']

Whenever there is no way to identify the element uniquely using any of the xpath we should use X-path by group index

http://money.rediff.com/gainers

identify previous price of the tata teleservices in the table.

(//a[contains(text(),"Tata Teleservices(M')")]/../..//td)[4]

Note:

- We should group by index only if necessary and we should keep it as last option, because of the position of the element changes then we need to modify the xpath which involves more maintenance effort.
- But whenever the scenario it self is demanding index then we must use, as shown below.

Write an x-path to identify the 1st link in the webpage..?

- (//a)[1]

Write an x-path to identify the last link in the webpage..?

- (//a) [last()]

Write an x-path to identify the even positive links w.r.t indexing..?

- (//a) [Position()mod 2=0]

Write an x-path to identify all the links which are starting form 4.1 in selenium WebSite

- (//a) [Starts-with(text(),'4.1')]

What are X-path predicate?

A predicate means condition. In X-path to avoid duplicates or index, We will write the X-path with some condition, they are called as X-path predicates.

Ex: //d[@data-label='helpline']

(X-path predicate/condition)

- In above Ex we are giving condition to identify the target <td> tag element. So here [@data-label='helpline'] is considered as predicate.
- Generally we can say predicate function or predicates as functions or the condition.
- These are the functions which we can use in X-path more frequently.
  1. text()
  2. Contains()
  3. Last()
  4. Position()
  5. Starts-with()

Note: -

- There is no first(0 in X-path.
- There is no ends-with() in X-path.

❖ Syntax for start-with function

Syntax for start-with()➔//tagName[starts-with(text(),'StringTextValue')]
Syntax for start-with()➔//tagName[starts-with(.,'StringTextValue')]
Syntax for start-with()➔//tagName[starts-with(@attributeName,'StringAttributeValue')]

❖ X-path by axes

```
<table> ➔Ancestor
    <tr>➔parent
        <td>Java</td>
----------<td>Selenium</td>----------
        <td>SQL</td>
    </tr>
    <tr>
        <td>
            <div>Automation</div>
        </td>
    </tr>
</table>
```

- Axes is plural form of Axis

Types of x-path by Axes

- Parent ➔/..

- Child➔ /
- Descendent ➔//
- Ancestor➔
- Following
- Following-sibling
- Preceding
- Preceding sibling
- Syntax===➔fixedElementxpath/axisName::tagName[predicate/condition]

What is x-path by axes..??

- The x-path by axis represents the relationship of the dynamic elements and fixed element.
- It is used to identify completely dynamic elements or duplicates based on the relationship.
- Here we can use various levels of backward and forward traversing in a simple way by using various axes names.

Parent axes: -

- This axes will identify the parent element from the current fixed element.

Note: -it is same as using /.. operator.

Child axes: -

- These axes will identify the child element from the fixed element.

Note: -it is same as using / operator.

Descendant: -

- This axes will identify all the descendants i.e. child/grandchild and so on from the context/fixed element.

Note: -it is same as using // operator.

Ancestor axes: -

- This axes will identify all the parent, grandparent and so on from the context/fixed element.

Note: -it is same as using /.. operator.

Following axes: -

- This axes will identify all the element which are appearing before context/fixed element.

Following-Sibling axes: -

- This axes will identify only the sibling of the fixed element which comes after context/fixed element.

Preceding-sibling axes: -

- This axes will identify only the sibling of the fixed element which comes before context/fixed element.

Preceding axes: -

- This axes will identify all the element which are appearing before the context/fixed element.

List of the forward axes: -

- Child

- Descendent
- Following
- Following-sibling

List of the reverse axes: -

- Parent
- Ancestor
- Preceding
- Preceding-sibling

Note: -

1. we can use * (symbol) in the place of tagName in the X-path in such case it will identify target element with any tagName based on the given condition.
   Ex: //*[@id='LoginButton']
   Here the above xpath identify the element with any tagName which
2. The above kind of xpath generally used in identifying the mobile web elements in Appium(mobile testing) because all the tagname's will be same most of the time.
3. Some graphics related tagname's will not be identified through X-path directly, in such case we can use.
   Example: - //svg[@id='users']
   SVG➔Scalable Vector graphics
4. Here SVG is graphics related tag through which x-path will not identify, but if we use * in place of SVG tagName then x-path will identify.

Example for x-path by axes

- In some scenarios using the index will be the only option as shown in the below example.
- URL: - https://money.rediff.com/gainers.

Write an x-path to identify national fittings previous closure.=======➔try this at home

- //a[contains(text(),'National Fittings')]/../following-sibling::td[2]

- Here we have to use the index since td tag has no attributes and also text is completely dynamic.
- This makes you to use x-path by index without any second thought.

!!!!!!!!!!!!!!!!INTERVIEW QUESTIONS!!!!!!!!!!!!!!!

What is WebElement..?

What is findElement..?

What are locaters and explain all the locater type..?

What is by class..?

What is no such element exception..?

What is css selector and why we should use css selector..?

What is x-path and why we should use x-path..?

What are the different types of x-path..?

Write the syntax of relative x-path with an example..?

what are the difference b/w css selector and x-path..?

what are the priorities of locaters..?

WEBELEMENT

Click()

- It is used to click on the target WebElement.
- Target WebElement can be :-
  1)Button
  2)RadioButton
  3)textField
  4)checkbox
  5)link..etc.

it performance 2 operations.

1. It will scroll the target WebElement to the visible or viewport area of webpage to same extend
2. It clicks on the target WebElement.
3. if it is not able to click on target WebElement then it will through elementClickInterceptedException(it is an unchecked selenium exception)
4. We will get the above exception whenever the target element is obscured or hidden by some other web element.
5. It will not receive any argument and the return type of click method is void.

- Signature: - public void click()

```
ele.click();
```

What is element click intercepted exception..?

- It is an unchecked exception of selenium which will be thrown by click() when ever the target element is obscured or hidden by another element.
- There Two ways to handle this exception
  1)Handel the element which is obscuring the target element.

2)using the java script executor.

Clear()

Signature➔public void clear()

Usage➔ele.class();

- It is used to remove the content from the textField.

sendKeys()

Signature➔Public void sendKeys(CharSequence…keysToSend)

Usage➔ele.sendKeys(“admin”);

Ele.sendKeys(“KeysEnter”);

- It is used to perform two operations.
  1. To enter the data into the text field.
  2. To perform some keyboard simulations.
- It will append the data without removing the existing content, so it is the best practice to use clear method using sendKeys.
- If the value is new it will throw “IllegalArgumetException”
- To perform keyboard simulation like pressing enter key on the WebElement or using some short keys to copy paste, we have to use the built in enum of selenium called keys this keys enum contains all the keyboard keys as the constants.
- Enum Keys{

  ENTER;
  BACKSPACE;
  DELETE;
  TAB;

  }

- IF the keyboard action cannot perform on the target WebElement then it will throw element Noninterpretable Exception
- Enum stands for Enumeration and it is one of the java templates.
- Note: 1)CharSequence is an interface in java which was implemented by string and string Builder classes

  2)Keys enum is also implemented by CharSequence interface of java.

  3)So for this reason sendKeys() can supply string as an input, StringBuilder as an input and keys enum as an input.(auto upcasting)

  4)IllegalArgumetException is unchecked exception of java.

What is element not interactable exception..?

- It is an unchecked exception which will be thrown when the target element cannot perform desire operation or keyboard simulation.

- Example: -> upon the target element if it is not able to perform keyboard simulation. In such cases to avoid this exceptions we should take the parent tab.

getText() -27/05/2022

Public String getText()

ele.getText();

This Method is used get that of WebElement.
It is used to It is used get text of WebElement both the normal to & link text.
It is used to verify the text of the target element.
It returns String.

Note:

- getText() Will return both the tagtext & Subtagtext. This behavior of getText() is sometimes an advantage Sometime its an disadvantage.
- Advantage in calendar popup.

getAttribute()

Signature:-

Public String getAttribute(String attribute name)

Usage

String attributeValue = ele.gatAttribute("placeholder")

- This method will return the value of the given attribute name of target WebElement.
- The argument of the method is String type and it will the string.
- If the given attribute name is not there with target element then it will return null.
- It is used to verify the attribute value of the WebElement like
  1. Verifying tool tip text.
  2. Verifying Alternative text.
  3. To verify the placeholder text.
  4. To v erify the textField is empty (or) not.

GetCssValue()

Signature: -

Public string getCssValue(String property name)

Usage:-

String propertyValue=ele.getCssValue("property value");

- This method is used verify the styling information of the target WebElement.
- This method accepts the Css property name as an input in the String form and returns the value associated with that property in String form itself.
- If the property name is not having any value in such case it returns empty String . so by using this method first verify whether the written value is empty or not .

Ex:-

1. Verifying the background color of the button.
2. Verifying the color of the text.

3. Verifying the boldness of the text.

Note:-

1. To see the Css property and values of the target WebElement:-
2. Inspect the target element.
3. Go to the computed tab, which is at the right side of the browser window.

Note:-

1. color is selenium class through which we can convert given rgb value into hexadecimal code.
2. Form String (rgb value) static method of color class which returns the color class type object itself.
3. asHex() is the nonstatic method of color class which returns the hexadecimal code for the supplied rgb value in form of string.

Note:-

- It is always but practice to compose the color is the hexadecimal color code format instead of the comparing him rgb.

Note:-

1. Rgb are primary color through which any color can be created.
2. Highest value is 255.
3. Ex: -(0,0,0) is prefect black.
4. (255,255,255) is perfect white.
   Hexadecimal value color code is the number used to r4epresent the rgb value.
   #000000 Black
   #FFFFFF
   Hexadecimal value contains o to 9 and ABCDEF. This combination is nothing but hexadecimal value.

Property name : font-weight
100 Thin(hairline)
200 Extralight(ultra light)
300 light
400 normal
500 Medium
600 SemiBold(Semi bold)
700 Bold
800 ExtraBold(ultraBold)
900 Block(heavy)

isEnabled()

Signature:-

```
Public Boolean isEnabled()
```

Usage: -

```
If(ele.isEnabled())
{
}
else
{
}
```

- IsEnabled() used to verify weather target WebElement is Enabled (or) not.
- This return type of this method Boolean, if it is return's true then target element enabled other wise it is disabled.

Note:-

1. IsEnabled() works properly only on input elements i.e,<input>tag and <button>tag
2. If it is called upon any other element then it will return always true through the element is enabled or not.

3. So in such cases we should find the proper work around.

IsSelected()

Signature:-

```
Public Boolean isSelected()
```

Usage:-

```
If(ele.isSelected()){
}
else
{
}
```

- This method is used to verify the status of radio button or check box
- If you use it on other elements then it will give you unpredicted o/p
- The return type of this method is Boolean. It returns Boolean true if it is Selected, false otherwise.

Note:-
isSelected() give unpredicted results if target element is not a radio button, checkbox and option of dropdown list box.
If it is yes on same other element in that case it will return false irrespective of weather target element is selected or not selected.

isDisplayed()

Signature: - public Boolean isDisplayed()

```
Is(ele.isDIsplayed()){
        }
Else{
`       }
```

- It is used to check the target WebElement is displayed(visible) or not.
- Return type is Boolean
- If it is return's true: element is present and it is displayed.
- If it is return's false: element is present but not displayed.

Note:-
This method can be used to verify both the element is present in DO< and weather it isDisplayed or not.

getSize()

Signature: -

```
Public Dimension getSize()
```

Usage: -

```
Dimension eleDimension=ele.getSize();
Int eleHeight=eleDimension.getheight();
Int elewidth=eleDimension.getwidth();
```

- This methos used to get the size of target WebElement.
- It returns dimension class object.
- Using the dimension class object, we can get the height and width of the target WebElement, i.e We have to call get height() and getWidth() or dimension object.

getLocation(): -

Signature: -

```
Public Dimension getLocation()
```

Usage: -

```
Point eleposition = ele.getLocation();
Int  start x = eleposition.getx();
Int start y = eleposition.gety();
```

- This method is used to get the co-ordinates of the target WebElement.
- It represents the point class object.
- Using point class object we can get x and y co-ordinates of the target WebElement, i.e, we have to call getX() and getY() of the point class type object.

getRect(): -

Signature: -

```
Public Rectangle getRect()
```

Usage: -

```
Rectangle Rect = ele.getRect ();
        Int eleHeight=eleRect.getHeight();
        Int eleWidth=eleRect.getWidth();
        Int ele StartX=eleRect.getX();
        Int ele StartY=eleRect.getY();
```

- This method is used to get both size and co-ordinates of the target WebElement.
- It returns rectangle class object.
- Using rectangle object we can the size and co-ordinates y.e, by calling getHeight(), getWidth(), getX(), getY().

Note:

- Dimension, point and rectangle are concrete classes of selenium library. We should always import it from selenium package before using if it in the code.
- getX and getY always returns the starting x co-ordinates value i.e, startY, by using this we can calculate end point.

!!!!!!!!!!!!!!!INTERVIEW QUESTIONS!!!!!!!!!!!!!!

WHAT IS USE OF GetSize(),GetLocation, GetSize()..?

1. All the method used to verify the arrangement of target WebElement like verification of the left, right and top alignment of the webpage.
2. Sometimes there might be issues with alignment while testing on different browsers.

Submit(): -

- It is used to click on the submit button.
- Submit button means:
    - The button element should contain type=”submit”.
    - The <button> should be the sub tag of the <form>.
    - <form>

```
<button> type=”submit”</button>
</form>
```

- If the method button is called on any element apart from submit button then it will throw JavaScriptException/the action won't be performed.
- This method will click on the submit button and if there is page redirection then it will wait until the next page has got loaded.
- JavaScriptException is an unchecked exception of selenium which will be thrown when there is mistake in the program/script and also when submit () is not used on the submit button.

What is the difference b/w click() and Submit()..??

Submit()	Click()
Submit() can be used to click on only submit button present within the form tag.	Whereas, Click() is used to click on any web element
Submit() will wait until the next page has got loaded.	where as click() does not wait for the next page to load.

getTagName()

Signature: public String getTagName()

Usage: ele.getTagName();

```
If(ele.getTagName().equals("input"))
{
        ele.clear();
        ele.sendkeys();
        ele.submit();
}
```

- This method is used to get the tag name of the target WebElement.
- It is used to verify the TagName of the target WebElement.
- It returns the tagName in String form.
- It is generally used in library methods to verify the tagName of the target WebElement to perform the desired task.

What are the different ways of clicking on a button..??

- Click();
- Submit();
- SendKeys(keys.Enter);
- JavaScriptClick();

What are the different ways of removing the text from the text field..??

- Clear();
- By using sendKeys() with keyboard shortcuts like:
  - keys.CTRL+"A"
  - keys.DELETE.

Tips to write the test Scripts..??

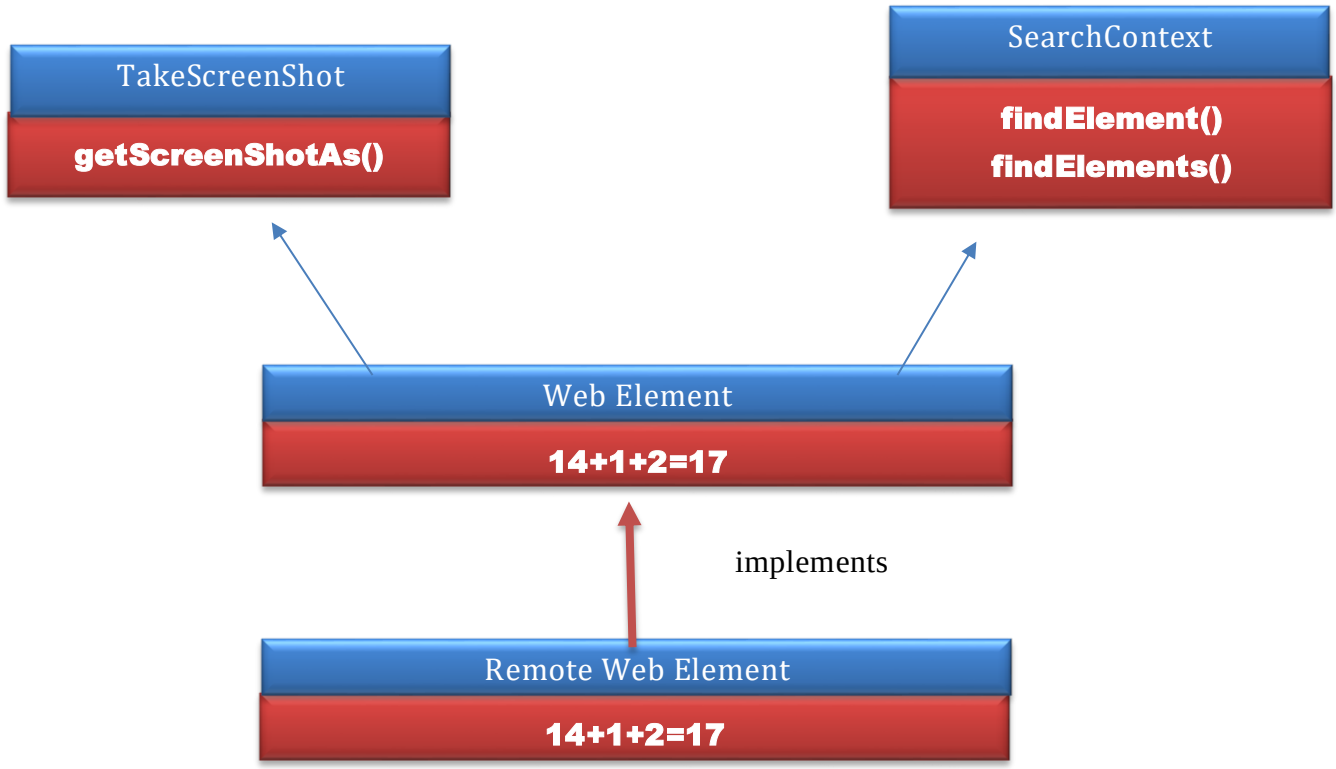
- execute the test case manually, write down dew important points to be considered while writing automation scripts.

- Always write an automation script according to manual test case i.e. there should not be any deviation in the automation script.
- Always write an automation script horizontally against manual test case. Horizontally means we should write verifications also.
- We should include the logs in the automation scripts which helps in analyzing the script output in a better way.

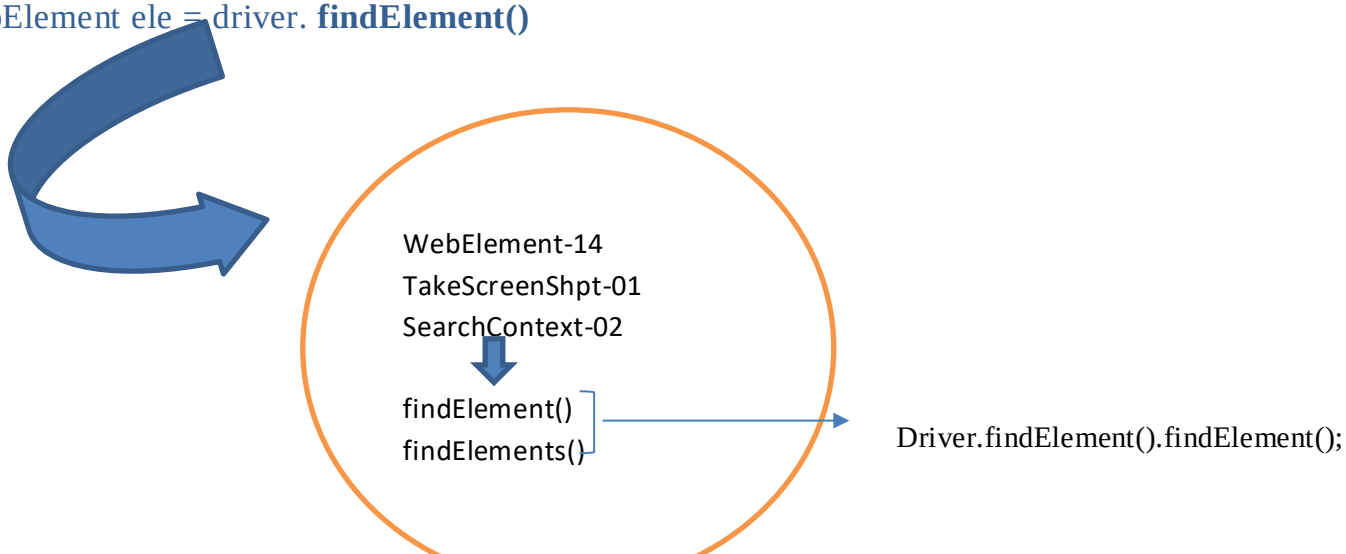
How do you write the log statements/what are the plugins you are aware of to write the log statements..??

- Log4j.
- testEng-repoter.
- Extend-reports. ----->comes under advance selenium.

HANDLING FORMS, TABLES, DIVISIONS USING CHILD ELEMENT.



WebElement ele = driver. **findElement()**



There are two implementations for find element and findElements:

1. Remote WebDriver implementation
2. Remote WebElement implementation

Remote WebDriver implementation

- It is to identify parent element in Webpage
- The search for target element begins from the beginning of the Webpage or html tree.

Remote WebElement implementation

- It is to identify parent element in webpage.
- The search for target element begins from the beginning of the webpage or html tree.
- Here search for the target element begins from the beginning of the webpage or html tree.
- The RemoteWebElement implementation
    - It is to identify the element within particular area or the WebPage or within the particular parent element.

Example:

```
WebElement form = driver.findElement();
WebElement formEle = form.findElement();
List<WebElement>form Elements = form.findElement();
```

- Here the search criteria for the target element begins and confined within the <form>/form elements, not from the beginning of the webpage or html tree.

Complete hierarchy of WebElement

- We can confined the search criteria to a particular form, table or division using the implementation of findElement() and findElements() or RemoteWebElements.
- First we identify the form element or table element or element under which we have group of elements using the find element method of RemoteWebDriver implementation.
- Then keeping the identified form or table and div element we will perform the search only on that identified parent element by using implementations of RemoteWebElements.

Note: -

- Search happens only within the parent element.
- use . at the beginning of the x-path for identifying child elements or else no need to use // i.e. just use the tagName with the predicate (if required).

SYNCRONIZATION

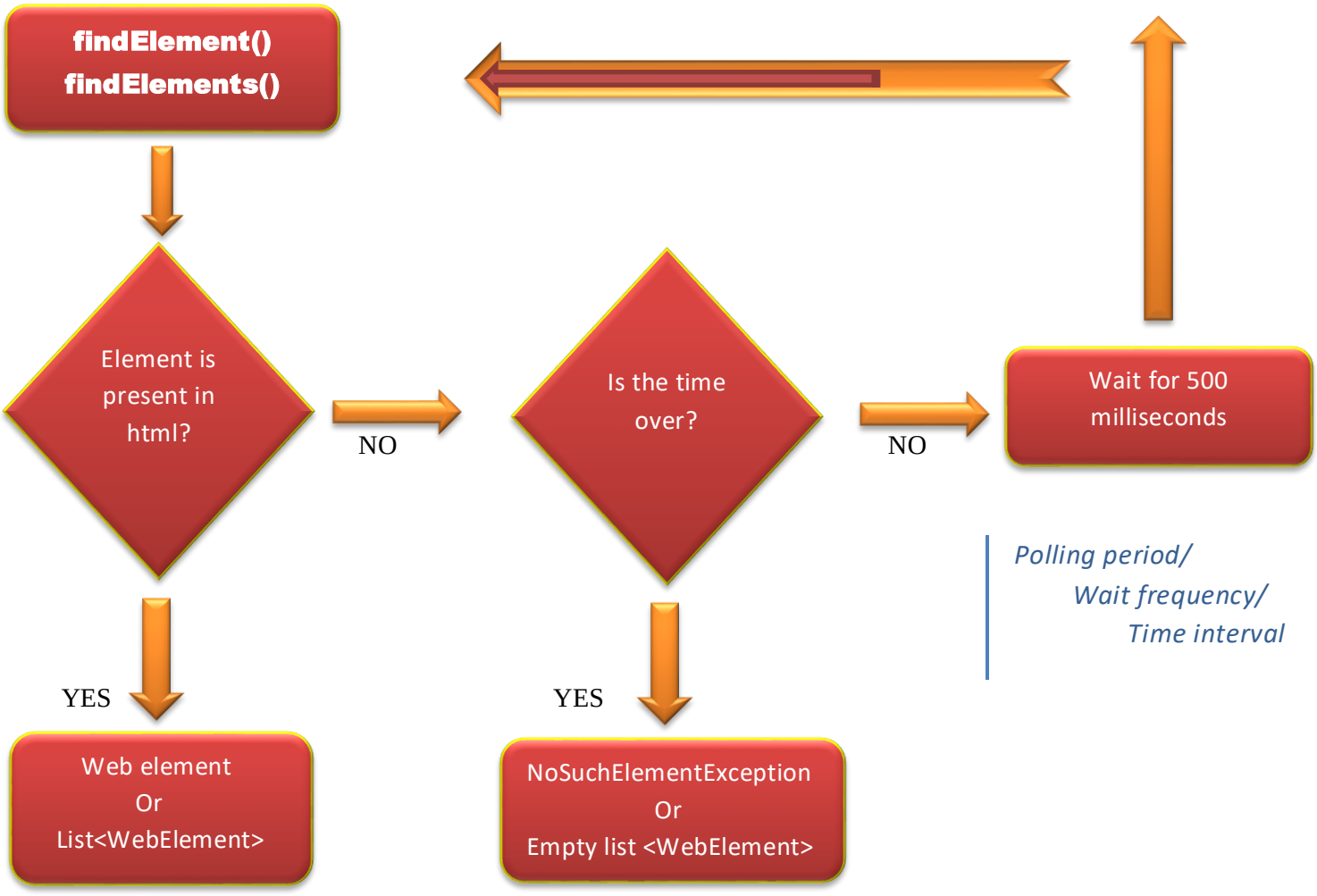
WHAT IS SYNCRONIZATION…??

- Synchronization is a process of matching two different events according to the requirement of the system, to get the desired outcome.
- In automation, we had to match the application loading speed and the automation script execution speed to get the proper results.
- The process is called as Synchronization in automation.
- Generally, the automation script execution speed is faster than the application loading speed. So we have to apply delays, in the automation script to match the application loading speed.
- There are 5 ways to achieve Synchronization.
    - Implicit wait.
    - Explicit wait.
    - Fluent wait.
    - Thread. Sleep wait.
    - pageLoadTimeOut.

Note:

- application loading speed will be slow due to
    - slow internet connection.
    - Poor business logics written in the webserver.

What is implicit wait..??



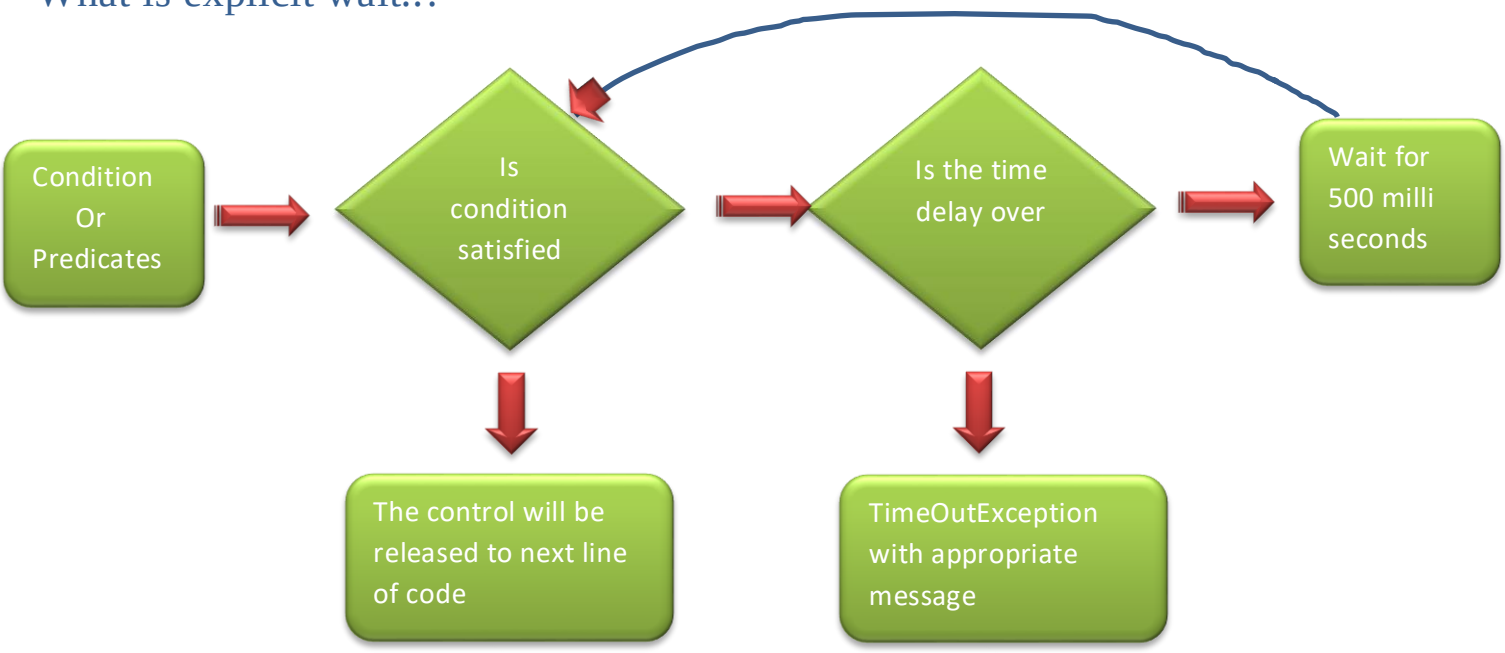
1. it is one of the intelligent waits in the selenium WebDriver. Which helps us to save time while we wait for the WebElement to load on the WebPage or html or Dom.
2. here we provide the time delay for searching elements in the WebPage.
3. If the element is formed any time within the specified time delay then the code does not wait for the remaining duration. It returns the found WebElement. So we call it as intelligent wait.
4. It uses 500 milliseconds of polling period/the search frequency.
5. if the element is not found within the specified time delay the we will get NoSuchElementException (By findElement()) or empty list of WebElement (By findElements()).

The following are the time units we can apply:

1. ofSeconds(long seconds);
2. ofMillis(long millis);
3. ofNanos(long nanos);
4. ofMinutes(long minutes)
5. ofHours(long Hours)
6. ofDays(long days)

Note:- this implicit timeout is applicable for each and every call of findElement() and findElements().So generally it will be applied at the beginning of the automation script as precondition.

What is explicit wait..?



It is one of the intelligent waits in selenium WebDriver which helps us to save time while we wait for:

- Expected title of the webpage to load.
- Expected URL of the webpage to load.
- to reach the clickable state.
- Image to be visible on the webpage.
- Alert to be present on the webpage and many more.

- Here we provide the time delay for the expected conditions to be successful.
- If the expected condition is satisfied anytime within the specified time delay then the load Does not wait for the remaining duration. It releases the control to the next line of automation script.
- It uses 500 milliseconds or the search frequency/polling period.
- If the given condition is not successful within the specified time delay then they will get TimeOutException with the appropriate error message.

Note:

1. Here expected conditions are called as predicates/predicates functions.
2. TimeOutException is an unchecked exception of selenium thrown when the explicit wait fails.

Steps to apply explicit wait in the automation script:

1. Create the object of WebDriverWait class by using WebDriver reference.
2. Call the until().
3. Supply the predicate function as an argument for the until method depending on your expected conditions to wait.

Create the object of WebDriverWait class by using WebDriver reference.

- There is a build in class called WebDriverWait in selenium package.
- It is a concreate class.
- The constructor of WebDriverWait process 2 arguments:
- WebDriver reference: -
    - This references the browser and application on which explicit wait has to be applied.
- The time delay in seconds: -
    - The time delay for explicit wait conditions.
    - Example: WebDriver wait = new WebDriver+(driver, Duration of Seconds(10));

Until()

- Until() is nonstatic method of webdriver wait class, whenever we have we have multiple conditions the we can this method again and again.
- For each and every until() invocation the overall time delay is applied.
- Example: wait until(ExpectedConditions title("Google"));
- ExpectedConditions is a concreate class of selenium and it is having lot of static methods which are also known as predicate function.
- Depending on the conditions we have to use the various static methods of this class.

Note:

- If the static method which are not there as the condition specified than we can create our own predicate function by using practice to create reference variable name as wait so that the code will be more readable.

Difference b/w implicit and explicate wait .

Implicit wait	Explicit wait
• **We need not specify waiting condition(built-in)**	• We need specify waiting condition.
• **We can handle the synchronization of all the invocation of find element()**	• We can handle the synchronization of any statement but only one at a time.
• **After specified time we will get NoSuchElementException**	• After specified time we will get TimeOutException

Where we get Thread. Sleep..??

1. Waiting for the animation to appear on the screen.
2. Popups which are not inspectable (file upload popup).
3. If implicit and explicit wait are not working as expected.

When we go fluent wait..??

When we have to set user define search frequency or polling period.

TAKESCREENSHOT (TAKESCREENSHOT ())

1. We need to take the screenshot for two reasons:
    a. for test case failure analysis i.e. is on failure of test script, we need to screenshot for better failure analysis.
    b. For adding it in the defect report which give more information for the developer about the defect.
2. We can take 2 types screenshot in selenium:
    a. Webpage ScreenShot
    b. WebElement ScreenShot.

Steps to take web page screen shot

1. Downcast the webdriver object into TakeScreenShot interface.
2. Call getScreenShotAs() with a target output type as FILE.
3. Store the temporary ScreenShot returned by getScreenShotAs() in the FILE class with the reference variable.
4. Create the destination file class object with the destination file path.
5. Copy the temporary file class object given by get ScreenShotAs() into the destination file class object using copy file static method of file util class.

getScreenShotAs()

Signature: - Public ___getScreenShot
Usage : - File tempFile = ts.getScreenShotAs(outputType.FILE)

- ❖ It is an abstract method o Takes ScreenShot of interface which is implement in.
- ❖ Remote WebDriver for taking page screenshot.

- ❖ Remote WebElement for taking web Element ScreenShot.
- ❖ This method will take ScreenShot based on the Target type specified as an argument to it.
- ❖ It will store the ScreenShot in the temporary locater of the local system and detects the ScreenShot after end of the program. (i.e. They are calling the detect on exit method.)
- ❖ By default, the ScreenShot Extension will be .png if the Target type given is FILE.
- ❖ It takes ScreenShot of only the visible area of the webpage.

Note: -

- Here the return type depends on the argument.
- Ex: - if the argument is FILE then return type is file class object.
- It will take current browser window ScreenShot.
- Always remember that it will shift focus on the target window automatically and then takes the ScreenShot.

Output type: -

Output type is an interface which contains 3 final variables which is used to specify the target screenshot type for getScreenShot as method, the final variables are:

- FILE- Datatype of this final variable is File.
  - Usage: - File tempFile = ts.getScreenshotAs(OutputType.FILE);
- BASE64- Datatype of this final variable is String.
  - Usage: - Base64 tempFile = ts.getScreenshotAs(OutputType.BASE64);
- BYTES- Datatype of this final variable is Binary.
  - Usage: -Bytes tempFile = ts.getScreenshotAs(OutputType.BYTES);

FileUtils

- It is a class name.
  - Signature: - public static void copyFile(Filesrc,Filedest)
    - Usage: -try{

```
FileUtils.copyFile(tempFile,destFile)
}
Catch(IOException e)
{
        e.printStackTrace();
}
```

1. It's a concreate class of commons-IO library which is used to perform various File-IO Operations.
2. In automation the scree shot taken by getScreenShotAs() is temporally, so to copy the temporally file we have use FileUtils class.
3. It is having lot of static methods, one among them is copyFile().
4. copyFile() takes two arguments of file class type.
5. First argument is the source file and the second argument is the destination file.
6. If the destination folder does not exist, it will create that folder then it will takes the screen shot.
7. It throws IOException in case of any folder access issues/some input/output issues while copying the source to the destination file. As IOException is one of the checked exceptions, we need to Handel it either with try catch block/with throws keyboard.

8. You should import this class from the commons-IO package.

Steps to download and configure commons-IO in eclipse.

1. commons-IO is open source 3rd party library from apache which contains lot of libraries to perform FileIO related operations.
2. It comes as a jar file in the maven repository (a cloud platform which contains most of the public java args(jar)).
    1. Go to https://mvnrepositary.com
    2. Enter commons-io in the search field and click on search button
    3. Click on 1st apache common's-io.
    4. Click on the latest version numbered link.
    5. Click on the jar in the files section.
    6. Click on the keep button (if asked).
    7. Copy paste this jar file in the jars folder of eclipse folder.
    8. Right-click on it.
    9. Click on build path and click on add to build path.

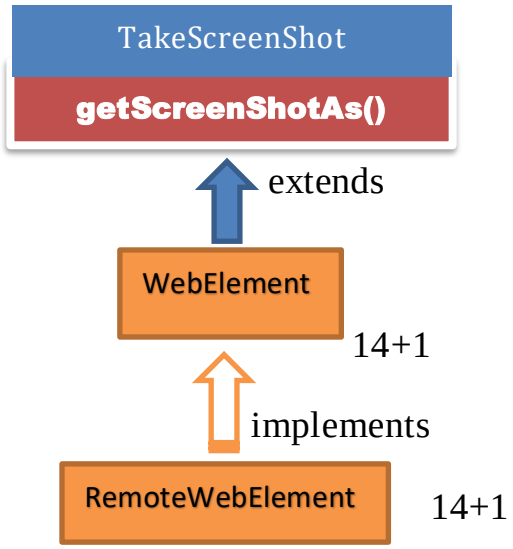
Creating unique file name for the screenshots.

- Whenever we are running automation scripts in real time we run them many times in a day/week so if the files name are not unique then the screenshot will be overridden to the latest execution. So we have to create a unique file name for the screen shot.
- To create unique file name for the screen shot we have to use the type stamp is nothing but the system date and time.
- If we capture the system date and time and add it to the screen shot file then we can retain the screen shot for all the executions.
- So to capture the system date and time we need to use the LocalDateTime concreate class of java.time package.
- It contains the static method called now which returns the same class object using which we can get the date and time.
- We have to call the toString method on the LocalDateTime object which returns the date and time in string form.
- But the date and time string contains ':' which is not a valid file name in the operating system, so we have to replace the special char to some other valid char like '-'.
    a. Signature:- public static LocalDateTime now()
        i. Usage: - String timestamp = LocalDateTime.now().toString replace(':' , '-');

Note:

i. As a standard practice all the Screenshots should be stored in project directly inside the folder called Bug Shots.
ii. Only the visible of the webpage will be taken in the screenshot.
iii. Current browser window screenshot will be taken, so to take other browser screen shot we have to transfer the control to that particular window.

WebElement ScreenShot



File webElementScreeshot = driver.findElement(By by).getScreenshotAs(OutputType.FILE);

Steps to take the WebElement ScreenShot

1. Identify the target WebElement using driver.findElement();
2. Call getScreenshotAs() on the target WebElement with target type as File.
3. Store the temporary ScreenShot returned by the getScreenshotAs() in the file class reference variable.
4. Create the destination file class object with the destination file path.
5. Copy the temporary file class object given by getScreenshotAs() into the destination file class object by using copy file method of file Utils class.

Note : there is no need to downcast the webdriver to take screenshot interface to take the WebElement screenshot.

FRAMES

WHAT IS A FRAME...?

- A frame is nothing but an imbedded webpage. Imbedded webpage means webpage inside another webpage or html inside another html.

HOW WEB DEVELOPER CREATES A FRAME...?

- Developer will use the html tag called <iframe> to create a frame.

```
<html>
    <body>
        <iframe> id=”i1” name=”n1” data-tit=”dd1”>
        <html>
            <head>
                <title>Frame o1</title>
            </head>
            <body></body>
        </html>
        </iframe id=”i2” name=”n2” data-tit=”dd2”>
            <html>
                <head>
                    <title>Frame 02</title>
                </head>
                <body></body>
            </html>
        </iframe>
    </body>
</html>
```