1) what is webElement? when should we prefer the web Element? (Document Object Model)

webElement is an interface. webElement represents DOM element (HTML element). Selenium Webdriver encapsulates a simple form element as an object of the webElement. The Execution of findElement & findElements method returns. webElement or List of webElement.

+ findElement():- finds a single webElement and returns it as a webElement object.

* findElements():- finds elements in a particular location using Locators.

⇒ If we supposed to do more than one action on the same element then we should prefer the webElement.

⇒ A webElement in this case, A Selenium webElement is essentially an HTML element on a website. HTML document Consists of HTML elements. Each HTML element Consist of a start tag and End tag. The Content lies blw the tags.

Syntax:- <starttag> Content </endtag>

⇒ Each webElement is represented in Selenium via the webElement Interface - which is used by Selenium to interact visible and invisible elements on the webpage.

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

2) Diff blw findElement()

*This Command is used for finding a particular element on a webpage, it is used to return an object of the first found element by the locator.

Eg:-
```
WebElement element = driver.
findElement(By.name("Q"));
```

findElements()

*This Command is used for finding all the elements in a webpage. Specified by the locator value. The return type of this Command is list of all the matching webElements

Eg:-
```
List<WebElement> elementList =
driver.findElements(By.tagName("p"));
```

3) Difference blw close and quit Methods?

close()	Quit()
close() method shall close the browser which is in focus.	Quit() method closes all the browsers.
close() method closes the active WebDriver instance. - driver.close()	Quit() method closes all the active webdriver instances. - driver.Quit().

4) Difference blw getwindowHandle() & getwindowHandles()?

Get Windaw Handle()	Getwindow Handles().
It fetches the handle of the webpage which is in focus.	It stores the set of handles for all the webpages opened Simultaneously
* It returns the window handle of current focused window/tab	* It returns all windows/tabs handles launched/opened by same driver instance including parent and child window.
* return type of getWindowHandle() is string.	* return type of getWindowHandles() is Set<String>
String CurrentWindow = driver.get Window Handle();	Set <String> all windows = driver. get Window Handles();

5) Difference between WebDriver wait & Fluent wait.

WebDriver Wait	Fluent wait
1) WebDriver Wait class is an extension of Fluent wait class. It doesn't have its own methods.	1) Fluent wait is an implementation of the wait interface that may have its timeout & polling interval configured on the fly.
2) Web drive wait is applied to. Certain element with defined. expected conditions and time. This wait is only applied to the specified element. This wait can also throw an exception when an element is not found.	2) Fluent wait is a type of Explicit wait where you can define polling and ignore the exception to continue with script execution in case an element is not found, Here we can set pooling time which is not possible in webdrive wait

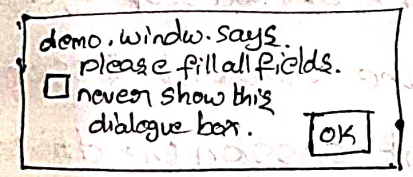
1) WebDriverWait (WebDriver driver, long timeOut in seconds).

2) FluentWait <WebDriver>
(driver).withTimeOut(30, SECONDS).pollingEvery(5, SECONDS).
.ignoring(NoSuchElementException.class);

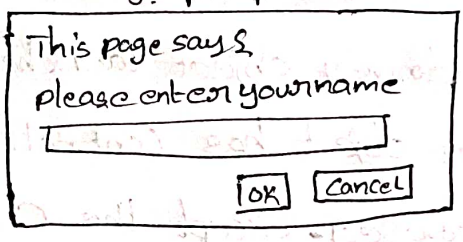
6) How to Handle the Alerts?

An alert in Selenium is a small messagebox which appears on screen to give the user some information or notification. It notifies the user with some specific information or error, and asks permisson to perform certain tasks. It also provides warning messages as well.

1) Sample Alert
The simple alert class in Selenium displays some information or warning on the screen.

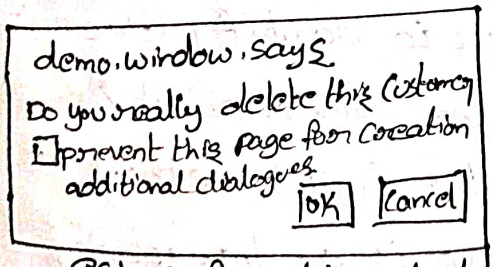


eg:- Simple Alert.

eg:- promptAlert


2) promptAlert :- It asks some input from the user and Selenium Webdrive can enter the text using SendKeys().

3) ConfirmationAlert:- This alert asks permisson to do some type of operation.



eg:- Confirmation Alert.

How to handle Alert is Selenium WebDriver.
Alert interface provides the below few methods which are widely used in Selenium Webdrives:-

1) Void dismiss() To click on the 'Cancel' button of the alert.

driver.SwitchTO().alert().dismiss();

2) Void accept() // TO click on the 'Ok' button of the alert.

driver.SwitchTO().alert().accept();

3) String getText() //To capture the alert message

driver.SwitchTO().alert().getText();

4) Void sendkeys(String stringTo Send) // To send some data to alert box

driver.SwitchTO().alert().sendkeys("Text");

6) How to Handle the alert?

once the driver object launches the browser by default the driver object has Controll on the Current Session of the browser. Except the Current session the driver object Cannot identify any windows.

In this Case if javascript alerts occured on top of Current Session of the browser by default Selenium is not able to identify the alert or the elements which are available in the alert. In order to identify elements which are available in alert, once the driver object has got the Controll on alert based on the Switch method it always returns alert object [Alert is an Interface]

7) List out the methods alert Supported the Alert object?

1) Accept :- It performs the click operation on Ok Button which is Available on the Alert.

2) Dismiss :- (It wont click on the Cancel button, It closes the Alert).

3) getText :- (In order to capture Text Content available on Alert we can use getText method).

4) SendKeys (If Alert containg Text field by using SendKeys method we can Enter the Text Content).

7) List out the Methods supported by the driver object? (15)

(a) get():- This method opens the specified URL in the Current browser window

driver.get("www.google.com");

(b) getCurrentUrl():- This method is used to get String url of the Current webpage loaded in the opened browser

driver.getCurrentUrl();

(c) getPageSource():-

This method is used to get the page Source Code of the Current Loaded webpage

driver.getPageSource();

(d) getTitle():-

This method is used to get the title of the Current webpage

driver.getTitle();

(e) getTagName():-

This method is used to get the tagName of the webElement returned by the findElement(By) method

String tagName = driver.findElement(By.id("DownloadButton")).getTagName();

(f) getAttribute():-

It is used to fetch or get the value of the attribute of the webElement

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

String attributeValue = element.getAttribute("id");

Syntax:- element.getAttribute();

(g). getCssValue:-

The getCSSvalue() method accepts nothing as a parameter & return a String value. used to fetch the value of Css property of the given WebElement

element.getCssValue();

(h) getSize():-

This method is used to fetch the width and height of the rendered element.

```
element.getSize();
```

(i) findElement():-

It returns first matching webElement on the current webpage

(j) findElements:- It returns all the matching webelements on the current webpage

(k) close() → This method is used to close only the browser window that webdriver is currently controlling.

(l) Quit():- This method is used to close all the windows opened by the Webdriver.

List:- Type of methods in driver object.

1) get()
2) getCurrentURL()
3) getPageSource()
4) getText()
5) getTagName()
6) findElement()
7) findElements()
8) getAttribute()
9) getCSSValue()
10) getSize()
11) getwindowHandle()
12) getwindowHandles()
13) ~~findElement()~~ close()
14) ~~findElements()~~ Quit()
15) manage()

8) List out the methods supported by the webElement object?

1) click()
2) sendkeys()
3) Submit()
4) isDisplayed()
5) isSelected()
6) isEnabled()
7) getLocation()

webElement:

8) clear()
9) getText()
10) getattribute()

1) Click() :- allows us to do the click action on button, link, checkbox or radiobutton

2) SendKeys() :- allows the tester to type the content element. Send keys ("") in to the editable field.

3) Submit() :- this command is usefull when it comes to interesting with forms on a webpage.

4) isDisplayed() :- Verifies if Certain element is present & Displayed If element is displayed, then the value returned is true. If not returns NoSuchElementFoundException

5) isSelected() :- It is used to determine if the element is selected.

6) isEnabled() :- Verifies if an element is Enabled on the web. page. element is enabled returns true value,

9) List out the methods Supported by the Action class object?

Action class is an ability provided by Selenium for handling keyboard and mouse events.

methods under Actions class :-

1) sendKeys() — Keyboard used to send the series of keys to the webElement

2) keyDown() :- perform keypress. without release

3) keyUp() :- perform key release

Mouseactions

1) click() :-

2) DoubleClick() :- perform double click on the element.

3) ClickandHold() : perform long click on the mouse without releasing it

4) DragandDrop() : drags the element from one point & drops to another

5) moveToElement() : Shifts the mousepointer to the centre of the element

6) ContextClick() : perform right-click on the mouse

10) List out any 10 Exception classes available in Selenium?

* NoSuchElementException
* NoSuchWindowException
* NoSuchFrameException
* NoAlertPresentException
* InvalidSelectorException
* ElementNotVisibleException
* ElementNotSelectableException
* TimeoutException
* NoSuchSessionException
* StaleElementException.

11) Describe How to Execute Java Script in Selenium?

JavaScript Executor is an interface that helps to execute JavaScript through Selenium Webdriver

JavaScript Executor provides two methods

1) ExecuteScript
2) ExecuteAsyncScript (To run javascript on the selected window or Current page)

Selenium Supports JavaScript Executor. There is no need for an Extra plugin or add-on, we just need to import.

[org.openqa.selenium.JavaScriptExecutor]

```
JavaScriptExecutor js = (JavaScriptExecutor) driver;
js.executeScript(Script, Arguments);
```

Syntax.

Script = This is the javascript that need to Execute.

Arguments = It is the arguments to the Script. This is optional.

* Creating the JavaScript Executor interface object by Typecasting.

a) How to get started with Java Script Executor

1) import the package
2) create reference
3) Call JavaScriptExecutor Methods.

12) Explain Page Object Model (pom) of writing Test Scripts & Describe its advantages?

The Page Object Model eliminates FindElement Method. The POM works based on the principle of encapsulation. Here the webelements have declared as private based on the Getter method we are returning the webElement.

In POM the webElements have derived based on the ID and Name attribute of the Element in the Dom structure. Suppose in the HTML Dom structure if any Elements has attribute ID or name based on it respective values we can Create an webElement

If the Element in the HTML Dom's structure if they dont have ID and Attribute the webElement can be Created based on @FindBy annotations, This can be acheived by "pageFactor.initElement(aBrowser, this);".

initElement is a static Method it accepts driver object & a Current class as a parameter. based on this Execution obj the above mentained approach we can Create the webElement-

Advantages of Page-object-Model :-

The main advantage of POM is it provides reusability of the webElements. It reduces the Code duplication & improves test maintainance. Make Code readable (methods get more realistic names). The Code becomes less and optimized.

13) How to Handle the Frames?

In order to work on Identify GUI elements in a frame by using Selenium approach we have to switch to that respective Frame.

=> we can handle Frames in an application :-

1) By using Index of the frame

SwitchTO().frame(frameNumber)

2) By using Name of the frame

SwitchTO().frame(frameName)

3) By using WebElement of the frame.

SwitchTo().frame(frame WebElement)

15) How to Select the items from the dropdown? Explain?

1) SelectByIndex (int index)

2) SelectByValue (String value)

3) SelectByVisibleText (String Text)

1) SelectByIndex

This method selects the dropdown option by its index number. we provide an integer value as the index number as an argument

Syntax:- SelectByIndex (int args)

2) SelectByValue

This method select the dropdown option by its value. we provide string value as an argument.

Syntax:- SelectByValue (String arg)

Eg:- se.SelectByvalue (6.)

3) SelectByVisibleText

This method select the dropdown option by its Text

Syntax:- SelectByVisibleText (String arg)

se.SelectByVisibleText ("white");

16) How do you display the items available in droppown?

Using getoptions()

Select class provides the following methods to get the options of a dropdown

1) getoptions() 2) getFirstSelectedOption()

3) getselectedOptions()

1) getoptionsS()

They are used to get all the options in a dropdown or multiselect box

Syntax:- getOptions(); List<WebElement>

This method returns all the options of the dropdown as a List of WebElement

Ex:- Select se = new Select (driver.findElement(By.id("old Select menu")));

// Get all the options of the dropdown.

List<WebElement> options = se.getOptions();

for(WebElement options1 : options)

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

17) Explain the object model of JDBC to fetch the records from database?

note:- Class.forName("oracle.jdbc.driver.oracleDriver");

Class.forName executes the staticBlock available in a Specific Source file.

note:- Class.forname always accepts parameter as Complete package Name with class name.

step1:

Class.forName("oracle.jdbc.driver.oracle Drive");

By using class.forName we can load oracle Driver.

step2:-

Connection conn = null;

we have to specify reference variable for Connection class

Step3:-

conn = DriverManager.getConnection(ConnectionString, UserName, password);

In order to allocate the memory for the Collection object we should use DriverManager.getConnection there the Drive Manager is a class, getConnection is a Static Method. It accepts 3 parameters. Those are.

* Connection String * Username * password.

Step 4:-

Statement Stmt = Conn.CreateStatement();

Based on the Connection object we can access the Create Statement Method. This Method returns Statement object

Step 5:-

String Query = "Select * from dept";

Design the Query to get the expected results from the DataBase table.

Step 6:-

ResultSet rs = stmt.executeQuery(Query);

Based on the statement object we can access ExecuteQuery this method accepts SQL Query as a parameter and it return Results set object.

Step 7:-

rs.next()

Based on the ResultSet object we can access method Next. The Next method return true if the record is available in resultsset. otherwise it returns false. Inorder to read the data from each Specific Column we can use Method as follows.

rs.getString(column name).

18) Describe Stale Element reference Exception?

Stale Element means an old element or no longer available element. Assume there is an element that is found on a webpage referenced as a webElement in webDriver. If the DOM changes then the WebElement goes stale. If we try to interact with an element which is staled, then the Stale Element Reference Exception is thrown.

A staleElement Reference Exception is a webDrive error that is thrown in following two cases.

Case(i):- The reference webelement has been deleted Completly :- Assume that there is an element in a web page and the webpage that the web element was part of has been refreshed or the Selenium Script navigated way to another webpage. This causes any subsequent interaction with the element to fail with the Stale element reference Exception.

Case(ii):- The referenced element is no longer attached to the Dom :- Assume a document node is removed from the DOM. it element reference will be invalidated. This causes any subsequent interaction with the element to fail with the Stale element reference Exception.

To overCome staleElementException in Selenium we have 4 ways :-

Solution 1 :- Refreshing the webpage.

You could refresh the page and try again for the same Element

Sample Code to overCome this issue.

```
driver.navigate().refresh();
driver.findElement(By.Xpath("Xpath here")).click();
```

Solution 2:- Using Try Catch Block

If an element is not attached to DOM then you could try using 'try-catch Block' within 'for-loop'.

Eg:= //using for loop it tries for 3 times.
// If the element is located for the first time then it breaks from the for loop and come out of the loop.

```
for (int i=0; i<=2; i++)
{ try{ driver.findElement(By.xpath("xpath here")).click();
    breaks;
  } catch(Exception e)
  { System.out.println(e.getmessage());
  }
}
```

Solution 3:- Using Expected Conditions.refreshed.

Use Expected Conditions.refreshed to avoid Stale Element Reference Exception and retrieve the element again. This method updates the element by redrawing it and we can access the referenced element

```
wait.until(ExpectedConditions.presenceOfElementLocated(By.id("table")));
wait.until(ExpectedConditions.refreshed(ExpectedConditions.Stale-
    ness of("table")));
```

Solution 4: Using POM (page object model)

We can avoid Stale Element Exception using POM. In POM we ~~can~~ use initElements() method which loads the element but it won't initialize elements. initElements() takes latest address. It initializes during run time when we try to perform any action on an element. This process is also known as "Lazy Initialization".

19) Describe the Annotation Execution Summary in TestNG?

TestNG Annotation is a piece of code which is inserted a programm or business logic used to controll the flow of execution of test methods.

Hierarchy of the TestNG Annotations:

@BeforeSuite
  @BeforeTest
    @Before class
      @Before Method
        @Test.
      @After Method.
    @After class
  @After Test
@ After Suite.

1) @BeforeSuite:- The method specified in the @BeforeSuite Annotation it executes first before execution of the whole TestNG Suite (or) all the test methods in the Suite.

2) @AfterSuite:- The methods specified in @AfterSuite Annotation it executes last after execution of the whole Test-NG Suite (or) will run after ~~all~~ the execution of all the test methods in Suite.

3) @BeforeTest:- The @BeforeTest annotated method will be executed before the execution of all the test methods of available classes belonging to that folder.

4) @AfterTest:- The @AfterTest annotated method will be executed after the execution of all the test methods of available classes belonging to that folder.

5) @Beforeclass:- The methods specified in @Beforeclass Annotation it executes first with respect to each SourceFile level (.Java)

6) @After class:- The methods specified in @After class Annotation it executes after execution of all the TestNG test Cases which are available in the respective (or) available Sourcefile

7) @BeforeMethod :- The method available in the @BeforeMethod it executes before execution of each TestNG test cases.

8) @AfterMethod :- The method available Specified in the @AfterMethod annotation it executes after execution of each TestNG test cases.

9) @Test :- The method Specified in the @Test Annotation indicates actual Test-Case in the TestNG file.

note :- we can have a number of TestNG Test-cases in a Single Source file.

10) @BeforeGroups :- The @BeforeGroups annotated method run only once for a group before the execution of all test cases belonging to that group.

11) @AfterGroups :- The @AfterGroups annotated method run only once for a group after the execution of all test cases belonging to that group.

* Benifits of using TestNG Annotation :-

* TestNG Annotations In this case, you do not need to extend any test classes.
* TestNG Annotations are strongly typed, i.e., Errors are detected at the Compile Time.
* you can pass the Additional parameters to the TestNG Annotation.
* TestNG Annotations Based on you requirements, you can access the test methods, i.e, it has no predefined pattern or format.

20) what is Log4j? what are the loggers options are Available?

Log4j is used to track the logs. Log4j is a tool to help the programmer output log statements to a variety of output targets

Methods and Description.

1) public void debug (Object message)
   it prints messages with the level Level.DEBUG
2) public void error (Object message)
   it prints messages with the level Level.ERROR.
3) public void fatal (Object message)
   it prints messages with the level Level.FATAL
4) public void info (Object message)
   " " " " " " " Level.INFO
5) public void warn (Object message)
   " " " " " " " Level.WARN
6) public void trace (Object message)
   " " " " " " " Level.TRACE

- Log4j has 3 main Components.

Loggers: Responsible for Capturing Logging information

appenders: Responsible for publishing logging information to Various preferred destination.

Layouts: Responsible for formatting logging information in different styles.

- Log4j is an API. It is a Apache product the main purpose of Log4j to track the logs during the execution of testscripts.

The same Log4j is used for the javaprogramming to print output in a file.

21) Difference between Hard Assertion & Soft Assertion?

Hard Assertion	Soft Assertion.
1) It does not allow further execution of test if the line containing hard asserts gets failed.	1) Next steps would be executed even if the line containing soft asserts gets failed.
2) Whole test case gets failed if atleast 1 hard assert fails	2) Extra lines of code are reqd. to track the fail status When used in raw format, failed test step would also yield in pass test script.
3) Hard Assert is a type of TestNG Assertion which throws Exception immediately if the assert statement fails and moves to the next testcase and if there is any code in the current test case after assert statement it will not execute the statement.	3) Soft Assertion are the type of Assertion when an assertion fails and continue with the next step after the assert statement This is used when the test requires multiple assertions to be executed and we want all of the assertions/ code to be executed before failing/skipping the tests

Assert.assertEquals => Assert.assertEquals(actual, expected);
Assert.assertNotEquals => Assert.assertnotEquals(actual, expected);
Assert.assertTrue.
Assert.assertFalse
Assert.assertNULL
Assert.assertNOTNULL
Assert.assertSame
Assert.assertNOTSame

AssertEquals()

Assertion Equals() is a method used to Compare the actual and expected results. If both the actual and expected results are same, then the assertion pass with no exception and the testCase is marked as "passed."

Assert NOTEquals()

AssertNotEquals() is a method used to Compare the actual and expected results. if both the actual and expected results are not the Same, then the assertion pass with no exception and test Case is marked as "passed."

22) Difference blw @parameters and @dataprovider ?

Parameter	Dataprovider
Parameter pass the parameters Just once per execution in TestNG	Dataprovider pass the different parameters on a single test in single execution.

These are two ways to pass the parameters in TestNG.

* TestNG parameters
* TestNG dataproviders.

Parameter values are hardcoded in the text Configuration file. testing.xml	They donot need to be hardcoded any where you can compute them on the fly.

23) How to handled the Tabbed Browser/Popup Browser?

By default selenium does not support child browser or tabbed browsers which are opened from the current session

Inorder to handle child browsers or tabbed browse selenium supports the following two methods.

(a) getwindowHandle() (b) getwindowHandles()

(a) getwindowHandle()

The getwindowHandle() returns window handle of parent browser in string representation.

(b) getwindowHandles()

This method return window handles of the both parent and child Browser in Set() of string representation.

24) Difference b/w X-path & Css Allocator?

X-path	Css Allocator.
X-path is a locator it was derived from XML (Extensible Mark up Language)	Css stands for Cascading Style sheet, The main purpouse of Css to apply styling on the HTML page
xpath can travels both in forward and backward direction in DOM Structure.	The Css. Can travels in forward direction only.
Xpath can locate element by text	Css ~~path~~ cannot locate element by text
Xpath is slower in Internet Explorer (I.E)	Css selector is faster in all browsers.
Xpath is slower in terms of performance and speed.	Css has better performance and speed than Xpath.
X-path is used to select nodes (or) node-sets in an XML document	Css is used to provide styling to HTML elements - Like color, font, background, image, boundary, shadow.
Xpath Allows very precise locator.	Allows for selection of elements by their Surrounding Context

* What is xpath axes and what are the options available in xpath axes.

⇒ * XPath axes are those axes that are used to search for the multiple nodes in the XML document from the current node context.

* These methods are mainly used when the web element is not identified with the help of ID, name, class name, link text etc.. locators

The options available in xpath axes are -

following-sibling
following
preceding-sibling
preceding
ancestor
descendant.

* What is Group in testng how to execute specific group.

⇒ The specific module related testcases or the testcases which belongs to specific function or the testcases which belongs to a particular feature, in this way we can group the testng testcases.

How to execute specific group –

```
<?xml version = "1.0" encoding = "UTF-8"?>
< suite name = " Groups of Tests ">
    < test name = " Tests " >
        < groups >
            < run >
                < include name = "regression" ></in
            </run>
        </groups>
        < classes >

        < class name = "com.sg testing.groupsdemo..
                                    Testcases" ></clas
        </classes >
    </test>
</suite>
```

* what are the different approaches we can run testng tests in a sequential order.

⇒ ① enabled = false

If you apply this condition into a testng testcase it ignore the execution of that specific test case.

```
@Test(enabled = false)
public void modifyUser()
{
System.out.println("The user demo1 has modified");
}
```

② Priority -

Which test cases have the lower priority those test cases executes first.

```
@Test(priority = 1)
public void createUser()
{
System.out.println("The user DemoUser1 has created");
}
```

③ Depends on methods -

By using Depends on methods condition we can run tests in sequential order.

```
@Test(dependsOnMethods = {"createProduct"})
public void modifyProduct()
{
SOP("The product demoproduct1 has modified");
```

Describe the POI object model to read or write content into the file.

⇒ The POI API is used to interacting with excell file in terms of read content or write content.

object model -

Step 1 :

```
FileInputStream fin = new FileInputStream("E:\\EXCEL\\Test.xlsx");
```

or

```
FileOutputStream fout = new FileOutputStream("E:\\EXCEL\\Test.xlsx");
```

Step 2 :

```
Workbook wb = null;
wb = new XSSFWorkBook(fin);
```

Step 3 :

```
Sheet sh = wb.getSheet("Sheet1");
Sheet sh = wb.creatSheet("Sheet2");
```

Step 4 :

```
Row row = sh.getRow(rownum);
Row row = sh.createRow(rownum);
```

Step 5 :

```
Cell cell = row.getCell(cellnum);
Cell cell = row.createCell(cellnum);
```

---

cell object -> Read content.

or

cell object --> write content.

* How to create object dynamically

=> To create an object dynamically from the class name, you need to use a reflection. If the fully-qualified name of a class is available, it is possible to get the corresponding class using the static method Class.forName().

```
Class.forName("com.sgtesting.reflection.Sample");
```