


Selenium Interview FAQs

1. When u will get No such Element Exception?

Whenever your giving invalid location /id /name in such case driver couldn't able to find web Element locator again u have remove it run.

2. What are different Locators in Selenium?

In Selenium Web Driver, there are 8 different types of locators:

1. ID ----->Right click-----Inspect
2. Name ----> Right click-----Inspect
3. Css selector----->Inspect—click on Top  Symbol just put on mouse holder whatever web element is inspecting of button.
4. Xpath---->Right click----Inspect----click on ...dots-----copy-----copy xpath
5. LinkText ---->Whenever ur inspect the web element if the tag name is starts with <a, static hyperlink
6. PartialLinkText ----> whenever the text is change dynamically use partial link text. It is also to start a tag name as<a, Dynamic hyperlink
7. ClassName----->It is just like a id/name
8. TagName ---->Whenever u want to find group of similar web elements on a webpage .

- driver.findElement(By.cssSelector(""));
- driver.findElement(By.className(""));
- driver.findElement(By.id(""));
- driver.findElement(By.name(""));
- driver.findElement(By.linkText(""));
- driver.findElement(By.partialLinkText(""));
- driver.findElement(By.tagName(""));
- driver.findElement(By.xpath(""));

[Locators Example](#)

```

package Examples_Locators;
import CommonUtil.*;

public class TC01_Locators
{
    static WebDriver driver;

    @Test
    public void Test2() throws Exception {

        driver = TestBrowser.OpenChromeBrowser();

        String TestURL = "https://opensource-demo.orangehrmlive.com/";
        driver.get(TestURL);
        driver.findElement(By.cssSelector("input#txtUsername")).sendKeys("Admin");
        //driver.findElement(By.name("txtUsername")).sendKeys("Admin");
        //driver.findElement(By.id("txtPassword")).sendKeys("admin123");
        driver.findElement(By.xpath("//*[@id='txtUsername']")).sendKeys("admin123");
        driver.findElement(By.className("button")).click();
        driver.findElement(By.linkText("Admin")).click();
        driver.findElement(By.partialLinkText("Nationali")).click();
    }
}

```

If you want group of elements following syntax

```

List<WebElement> list1= driver.findElements(By.tagName("a"));
int count1=list1.size();

```

2. What is an XPath?

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

3. When you use these locators ID, Name, XPath, Or CSS Selector?

- ID & Name locators will be used when there are unique identifiers & unique names available on the web page.
- CSS Selector can be used for performance and when ID & Name locators are not unique.
- XPath is used when there are no preferred locators.

4. What is the difference between “/” and “//”

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

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

5. What are the ways you can customize the TestNG report?

Ans. You can customize TestNG report in two ways:

- Using ITestListener Interface
- Using IReporter Interface

6. What is the difference between driver.findElement vs driver.findElements ?

Suppose, we need to count the no. of links on the Google Page. As we know link is represented by tag ‘a’ in html. So, we will use the method ‘tagName’ to count the total no. of links.

```
List<WebElement> list1= driver.findElements(By.tagName("a"));  
int count1=list1.size();
```

```
driver.findElement(By.id("txtPassword")).sendKeys("admin123");  
driver.findElement(By.name("txtUsername")).sendKeys("Admin");
```

This is used to find the WebElement like Button, checkbox,radio button etc. on the page.The different locator (id,name,xpath etc.) are used within it to find the elements.

#	findElement	findElements
Definition	driver.findElement() is used to find a webElement on a webpage.	driver.findElements() is used to find a List of webElements matching the locator passed as parameter.
Syntax	WebElement element = driver.findElement(By locator);	List elements = driver.findElements(By locator);
For multiple matches	In case the same locator matches multiple webElements then findElement method returns the first web element found.	In case of multiple matches the findElements method returns a list of webElements. For interacting with a particular element we have find the particular element by its index e.g. elements.get(0).click(); will perform the click operation on the first element of the 'elements' list.
If no element is found	In case the locator passed to findElement() method leads to no element then NoSuchElementException is thrown.	In case the locator passed to findElements() method leads to no element then a List of 0 size is returned instead of an exception.

7. How to find number of hyperlinks in a web page ?

```
List<WebElement> list1= driver.findElements(By.tagName("a"));  
int count1=list1.size();
```

8. How to find number of IFrames in a web page ?

```
List<WebElement> list= driver.findElements(By.tagName("Iframe"));  
int count=list.size();
```

9. How do u handle dropdown in selenium?

1. we have to import the select class from open selenium support libraries.

Import the package org.openqa.selenium.support.ui.Select

2. for select class we can create a object

```
Select dropdown1= new Select(driver.findElementBy("location_country"));
```

- Select-----class name
- Dropdown1---object name

10. How many methods are there in dropdown?

In dropdown 3 methods are used i.e.

1. Select By visible Text---->Selects/deselects the option that displays the text matching the parameter.

Parameter: The exactly displayed text of a particular option.

selectByVisibleText() and deselectByVisibleText()

```
Ex1: Select dropdown1= new  
Select(driver.findElementBy("location_country"));  
Dropdown1.selectByVisibleText("India");
```

```
Ex: drpCountry.selectByVisibleText("ANTARCTICA");
```

- drpcountry is a obj name(write ur own name)

2. SelectBy Value---->Selects/deselects the option whose “value” attribute matches the specified parameter.

- **Parameter:** value of the “value” attribute
- Remember that not all drop-down options have the same text and “value”, like in the example below.

Syntax: **selectByValue()** and **deselectByValue()**

```
Ex: drpCountry.selectByValue("234");
```

Output:

```
<option value="10">ANGUILLA </option>
<option value="234">ANTARCTICA </option>
<option value="1">ANTIGUA AND BARBUDA </option>
```

Example 2: <option Value=”IN”>India</option>

```
Dropdown1.SelectByValue("IN");
```

Output: India

3. selectBy Index----->Selects/deselects the option at the given index.

- **Parameter:** the index of the option to be selected.

Syntax: **selectByIndex()** and **deselectByIndex()**

```
Ex 1: drpCountry.selectByIndex(0);
```

```
Ex 2: dropdown1.selectBy Index(2);
```

Output: Antlanta

11. What is Data provider?

1. Data provider provides input data to your test.
2. The test will be executed multiple times for different sets of input data.

12. Which approach is used in interview for the data provider.

i) method

//step1

```
@DataProvider(name = "Orange1")
```

```
publicstatic Object[][] Test1() throws Exception
```

@Test(dataProvider = "Orange1") //step2 u r test will be connected to dataProvider

```
publicvoid Test1(String TestURL,StringUserName,String Password,String Nationality) throws Exception
```

ii) @DataProvider(name="DataContainer")

```
public Object[] myDataProvider() {
```

```
    Object data[][]= new Object[2][4];
```

```
    // First student details
```

```
data[0][0]= "Mukesh";
```

```
data[0][1]= "Otwani";
```

```
data[0][2]= "Motwani@gmail.com";
```

```
data[0][3]= "Motwani@gmail.com";
```

```
    // Second student details
```

```
data[1][0]= "Amod";
```

```
data[1][1]= "Mahajan";
```

```
data[1][2]= "amahajan@hotmail.com";
```

```
data[1][3]= "amahajan@hotmail.com";
```

```
    return data;
```

```
}
```

2nd approach is written.

13. How to create a data provider obj array?

A: `obj data[]=new obj[2][4]-----2(rows),4(cols)`

U have to give input data

14.In selenium wt is the first line of code?

A: `WebDriver driver;`

1.`System.setProperty("webdriver.chrome.driver","C:\\chromedriver_win32\\chromedriver.exe");`

(Here am specifying the chrome driver location)

2.`driver =new ChromeDriver();`

(Here ur chrome browser window will be launched)

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

(ur window will be maximized)

4.`driver.get("https://opensource-demo.orangehrmlive.com/");`

(URL application)

15. How to invoke any URL application?

A:`driver.get(TestURL)`---we have to give application URL

16.How to call Static methods?

A: `classname.method name`

17. How to call Non static methods?

A: We have to declare the obj

`Obj.methodname`

We have to create the class

18. I want to know how many hyperlinks are there on the webpage?

```
List<WebElement> links1= driver.findElements(By.tagName("a"));  
int links_count= links1.size();
```

```
System.out.println("Number of hyperlinks on LoginPage : "+links_count);
```

19. What is static and dynamic?

A: Static: The hyperlink text is not yet all changing i.e. called as static[linktext](fixed).

Dynamic: The hyperlink text is changing everytime i.e. called as dynamic[partial linktext]

20. How to implement Browser compatiability testing?

A: if(Browser.equalsIgnore(chrome))

if(Browser.equalsIgnore(firefox))

21. How to get Test Data from TestNG Data Provider ?

A: **Data Provider provides the Test data to the test**

@DataProvider

```
@DataProvider(name = "TC01_OrangeHRM")
public static Object[][] Authentication1() throws Exception {
    return new Object[][] {
        { "Admin", "admin123", "Indian06" },
        { "Admin", "admin123", "Indian07" }
    };
}
```

@Test

```
@Test(dataProvider="TC01_OrangeHRM")
public void Login_Test(String UserName,String Password,String Nationality)
```

22. How to get Test Data from TestNG.xml Parameters ?

A:

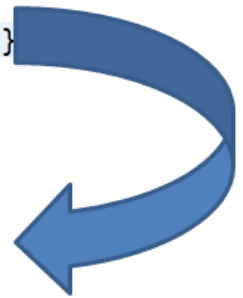
```
<parameter name=" Browser1" value="Chrome"/>
<parameter name=" UserName1" value="Admin"/>
<parameter name=" Password1" value="admin123"/>
<parameter name=" Nationality1" value="Indian555"/>
```

testng1.xml



```
@Parameters({"Browser1", "UserName1", "Password1", "Nationality1"})
```

```
@Test
public void Login_Test(String Browser,String UserName,String
Password,String Nationality)
```



22. How to get data from Excel sheet in data provider?

A: ExcelApiTest4 eat = new ExcelApiTest4();

Object[][] testObjArray =

eat.getTableArray("C://OrangeHRM6//TC01_Nationality1.xlsx", "Sheet1");

System.out.println(testObjArray.length);

return (testObjArray);

ExcelApiTest4 is the Excelutil methods.

23. How to get screen shot in Selenium?

```
//Convert web driver object to TakeScreenshot
TakesScreenshot scrShot =((TakesScreenshot)driver);

//Call getScreenshotAs method to create image file
File SrcFile=scrShot.getScreenshotAs(OutputType.FILE);

//Move image file to new destination
File DestFile=new File(filePath);

//Copy file at destination
FileUtils.copyFile(SrcFile, DestFile);
```

24. What is JUnit? Explain the various JUnit annotations.

A: JUnit is a Java-based testing framework from Apache that complements Selenium. Various JUnit Annotations are enumerated as follows:

- @After – Lets the system know that this method will be executed every time a test method achieves completion
- @AfterClass – Lets the system know that this method must be executed once after any of the test methods
- @Before – Lets the system know that this method will be executed just before every time a test method starts execution
- @BeforeClass – Lets the system know that this method must be executed once before any of the test methods start execution
- @Ignore – Lets the system know that this method shall be ignored i.e. it shall not be executed
- @Test – Lets the system know that this method is a test method. It is possible to have several test methods in a single test script.

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

- JUnit
- TestNG

26. What is the TestNG in Selenium?

A: TestNG is an automation testing framework in which NG stands for "Next Generation". TestNG is inspired from JUnit which uses the annotations (@). TestNG overcomes the disadvantages of JUnit and is designed to make end-to-end testing easy. ... The TestNG in Selenium provides an option, i.e., testng-failed.

27. What is emailable report in TestNG?

A: TestNG provides default html reports with basic information like No. of tests passed, failed, skipped and time taken to execute these tests in milliseconds. ... TestNG provides IReporter interface which we can implement and customize default testng emailable report.

```
@Parameters({"Browser1"})
@BeforeTest
public void Test1(String Browser) throws Exception
{

    if(Browser.equalsIgnoreCase("Chrome"))
    {
        driver = TestBrowser.OpenChromeBrowser();
    }

    if(Browser.equalsIgnoreCase("FireFox"))
    {
        driver = TestBrowser.OpenFirefoxBrowser();
    }
}
```

28. What is extent report in TestNG?

A: Extent Report is an open-source reporting library used to create visually attractive reports for Selenium tests using JUnit and TestNG. Extent reports produce HTML-based documents that offer several advantages like pie charts, graphs, screenshots addition, and test summary.

29. How to customize TestNG Report?

A: TestNG reporting is quite handy but still, sometimes we need some less data in reports or want to display reports in some other format like pdf, excel, etc. or want to change report's layout.

There can be two ways we can customize TestNG report

- Using ITestListener Interface:
- Using IReporter Interface:
-

30. Where can we apply Parallel Test execution in TestNG?

A:

1. Methods: This will run the parallel tests on all @Test methods in TestNG.
2. Tests: All the test cases present inside the <test> tag will run with this value.
3. Classes: All the test cases present inside the classes that exist in the XML will run in parallel.

Methods: If a TestNG Class contains multiple test methods.

```
<suitethread-count="3"parallel="methods"name="Suite">
<testthread-count="3"parallel="methods"name="Test">
<classes>
<classname="Day_014_ParallelMethods.AllTests"/>
</classes>
</test><!-- Test -->
</suite><!-- Suite -->
```

Tests: If a TestNG.xml contains multiple Tests.

```
<suitename="Suite"parallel="tests"thread-count="2">

<testname="Test1_MercuryTest">
<classes>
<classname="Day_015_ParallelTests.MercuryTest"/>
</classes>
</test>

<testname="Test2_OrangeHRMTest1">
<classes>
<classname="Day_015_ParallelTests.OrangeHRMTest"/>
</classes>
</test>

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

Classes: If a TestNG.xml contains multiple TestNG classes.

```
<testname="All_Test_Class_Files">
<classes>
<classname="Day_014_ParallelClasses.MercuryTest"/>
<classname="Day_014_ParallelClasses.OrangeHRMTest"/>
</classes>
</test>
```

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

31. How do you prioritize tests in TestNG?

A: Lower the priority number; higher is the priority of the test case method. Priority in TestNG contains only integer value. The value can be negative, zero, or positive. If a tester defines a priority in decimal in TestNG, it needs to **convert first to Integer** (through typecasting).

TestNG.xml --> Specific Module you can create Batch files.

1. Parallel Classes -----> TestNG.xml contains multiple TestNG classes,
2. Parallel Tests -----> TestNG.xml contains multiple Tests ,
3. Parallel Methods----->TestNG Class contains multiple test methods

4. Multiple Suites

```
PASSED: Test6_closebrowser-----> @Test(priority=6)
PASSED: Test4_Addnatialities----> @Test(priority=5)
PASSED: Test5_Logout-----> @Test(priority=-1)
PASSED: Test3_Login-----> @Test(priority=1)
PASSED: Test2_OpenOrangeHRM-----@Test(priority=0)
PASSED: Test1_OpenChromeBrowser--@Test(priority=0)
```

32. What are the types of waits available in Selenium WebDriver?

A:

1. Implicit Waits
2. Explicit Waits

3. Fluent Waits

1. Implicit Waits : It is a Global rule applicable for all the web Elements where ever it has performance issues , and driver will wait for the webelement as per the implicit time specified.

```
driver=TestBrowser.OpenChromeBrowser();  
driver.manage().timeouts().implicitlyWait(60, TimeUnit.SECONDS);
```

2. Explicit Waits : Some webElements can't be loaded with in the default implicit time , such specific web elements can be ehanded explicitly by using explicit wait.

```
findElement(By.id("welcome")).click();  
  
WebDriverWait wait= new WebDriverWait(driver,120);  
wait.until(ExpectedConditions.visibilityOfElementLocated(By.linkText("Logout")));
```

3. Fluent Waits: it is also used to handle specific webelements only on the webpage . The webdriver will look for the webelement at regular intervals as per the pooling frequency time.

```
// driver wait max 60 seconds for an element to be present on the page.  
// checks for the Web Element for every 5 seconds.  
WebElement wait3= new FluentWait<WebDriver>(driver)  
    .withTimeout(60, TimeUnit.SECONDS)  
    .pollingEvery(5,TimeUnit.SECONDS).  
    ignoring(NoSuchElementException.class)  
    .until(ExpectedConditions.presenceOfElementLocated(By.xpath("")));  
wait3.click();
```

33. What happen if you mix both implicit wait and explicit wait in a Selenium Script?
A: Implicit wait is defined only once in the code. It will remain same throughout the driver object instance.

Explicit wait is defined whenever it is necessary in the code. This wait will call at the time of execution. It is a conditional wait.

Explicit wait will overwrite the implicit wait where ever explicit wait is applied. So, Explicit Wait gets first preference then Implicit Wait.

34. What is frame?

A: Frame is used to call another html page in the current web page.

Frames can be handled in 3 ways

1. Switch into frame by Index:

```
List<WebElement> frames = driver.findElements(By.tagName("iframe"));
System.out.println("Number of frames: " + frames.size());
driver.switchTo().frame(frames.get(0));
driver.switchTo().defaultContent();
driver.switchTo().frame(frames.get(1));
driver.switchTo().defaultContent();
```

2. Switch into frame by name :

```
driver.switchTo().frame("Google_Contact Form");
```

3. Switch into frame by web element reference:

```
driver.switchTo().frame(driver.findElement(By.id("GoogleContactForm")));
```

35. How to switch or handle driver among the frames?

A:

```
driver.switchTo().frame(0); // By index
driver.switchTo().frame("Child Frame"); // By Frame Name
driver.switchTo().parentFrame(); // Back to Parent Frame
driver.switchTo().defaultContent(); // Default Fisrt Frame
```

36. Multiple windows

How to switch or handle driver among multiple windows?

```

@Test
public void MultipleWindows2() throws Exception
{
    WebDriver driver = new ChromeDriver();
    driver.get("http://yahoo.com");

    ((JavascriptExecutor)driver).executeScript("window.open()");
    ArrayList<String> tabs = new ArrayList<String>(driver.getWindowHandles());

    driver.switchTo().window(tabs.get(1));
    driver.get("http://google.com");
}

```

multiple windows:

```

ArrayList<String> tabs = new ArrayList<String>(driver.getWindowHandles());
driver.switchTo().window(tabs.get(0));
driver.switchTo().window(tabs.get(1));

```

37. What is Java script Executor?

A: JavaScriptExecutor in Selenium provides two methods through which we can **run JavaScript on the selected window or the current page**. This method executes JavaScript in the context of the currently selected window or frame in Selenium. The script will be executed as the body of an anonymous function.

In Selenium Web driver, locators like X Path, CSS, etc. are used to identify and perform operations on a web page.

In case, these locators do not work you can use Java Script Executor. You can use JavaScript Executor to perform an desired operation on a web element.

Selenium supports java Script Executor. There is no need for an extra plugin or add-on. You just need to import (**org.openqa.selenium.JavascriptExecutor**) in the script as to use Java Script Executor.

1. We can launch new tab
2. Scroll at particular web Element
3. Send Keys
4. Click

5. Highlight a web Element border

1. `((JavascriptExecutor)driver).executeScript("window.open()");`
2. `js.executeScript("arguments[0].scrollIntoView();", Connect_with_Us);`
3. `js.executeScript("arguments[0].setAttribute('value','admin123')", Password);`
4. `js.executeScript("arguments[0].click();", LoginButton);`
5. `js.executeScript("arguments[0].style.border='3px solid red'", UserName);`

38. What is an Action class and how many operations are performed?

A: Actions class is an ability provided by Selenium for handling keyboard and mouse events. In Selenium WebDriver, handling these events includes operations such as drag and drop, clicking on multiple elements with the control key, among others. These operations are performed using the advanced user interactions API. It mainly consists of Actions that are needed while performing these operations.

syntax:

```
Actions action = new Actions(driver);  
  
action.moveToElement(element).click().perform();
```

Action class methods

1. Mouse Hover action
2. Double click

3. Right Click
4. Drag and drop

Actions actions = new Actions(driver);

1. actions.moveToElement(Users).click().build().perform();
2. actions.doubleClick(Copy_Text_Button).perform();
3. actions.contextClick(RButton).perform();
4. actions.dragAndDrop(Source_Drag_Button, Target_Drag_Button).perform();

39. File upload and download?

File Upload:

1. We have to save file path temporary buffer in CTRL+C Action
C:\HTML Report\EMP_Photos\image2.jpg
 2. CTRL+V
 3. Enter key
- import Java.AWT.Robot;

File Download:

1. Users/Download
2. MoveFile(Src, Dest)

40. Advanced Xpaths

1. Absolute xpath-->Copy Full xpath--identifying the xpath from root node to current node is nothing but full xpath
2. Relative xpath-->Copy xpath---> identifying the xpath based on object properties is known as

41. How to write xpath manually?

A: 1. Take username webelement

```
<input name="txtUsername" id="txtUsername" type="text">
```

Here name/id/type-----obj property

2. < instead of we r writing // and also remove “ ” to ‘ ‘.

```
//input[@name='txtUsername']
```

3. suppose ur checking [name=txtUsername] the system is showing 1 of 3 with the name of txtUsername 3 webElemnets are there.

If i want add multiple conditions/filters in the xpath use **and** operator

```
//input[@name='txtUsername' and @id='txtUsername' and @type='text']
```

4. If i not specifying the tagname use *

*** means it may be anchor tag/div/input**

```
//*[ @name='txtUsername']
```

42. Why we can go with relative path?

A: The developer can changing something coding like id/name it will be impacted only specific webelement only it is not impacting all other webelemnets.

-----> In absolute xpath the developer is modifying one place all other dependency xpaths also impacted.

43. when i go with absolute xpath?

A: When there is no solution at all and ur not able to locate the webelements xpath by using any of the relative xpath/any of the other locate strategies.

In programming language absolute xpath is not recommended.

44. ExcelUtil_Apache_POI

1.TC01.xlsx --->org.apache.poi.XSSF--->Office 2007,office2010,2013,2016

```
import java.io.FileInputStream;
```

```
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
```

```
import org.apache.poi.xssf.usermodel.XSSFSheet;  
import org.apache.poi.xssf.usermodel.XSSFRow;  
import org.apache.poi.xssf.usermodel.XSSFCell;
```

```
public FileInputStream fis = null;  
public XSSFWorkbook workbook = null;  
public XSSFSheet sheet = null;  
public XSSFRow row = null;  
public XSSFCell cell = null;
```

```
    fis = new FileInputStream(xlFilePath); //excel File path  
    workbook = new XSSFWorkbook(fis); // workbook open  
    sheet = workbook.getSheet(sheetName); //Sheet1 open  
    row = sheet.getRow(rowNum); // 1st will be highlighted  
    cell = row.getCell(column); // 1th column will be highlited
```

```
String str6=cell.getStringCellValue();
```

2.TC02.xls---->org.apache.poi.HSSF---> Msoffice-97-2003

Excel Util Apache POI supporting libraries

org.apache.poi.xssf----->.xlsx

org.apache.poi.Hssf----->.xls

Datadriven Test

1. Data Provider
2. Excel Sheet
3. TestNG Parameters

45. Screenshot

A Screenshot in Selenium WebDriver is used for bug analysis. Selenium webdriver can automatically take screenshots during the execution. But if users need to capture a screenshot on their own, they need to use the TakeScreenshot method which notifies the WebDriver to take the screenshot and store it in Selenium.

Capture Screenshot using Selenium WebDriver

- Step 1) Convert web driver object to TakesScreenshotTakesScreenshotscrShot
=((TakesScreenshot)webdriver);
- Step 2) Call getScreenshotAs method to create image file
FileSrcFile=scrShot.getScreenshotAs(OutputType.FILE);
- Step 3) Copy file to Desired Location.

Syntax:

- `driver.get("https://opensource-demo.orangehrmlive.com/");`
- `TakesScreenshotscrShot1=((TakesScreenshot)driver);`
- `File SrcFile1=scrShot1.getScreenshotAs(OutputType.FILE);`
- `FileUtils.copyFile(SrcFile1,newFile("C:\\TC02_Login\\TC1_ScreenShot 1.jpg"));`

46. Alerts

A: An **Alert in Selenium** is a small message box which appears on screen to give the user some information or notification. It notifies the user with some specific information or error, asks for permission to perform certain tasks and it also provides warning messages as well.

Alert interface provides the below few methods which are widely used in [Selenium Webdriver](#).

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 stringToSend) // To send some data to alert box.

```
driver.switchTo().alert().sendKeys("Text");
```

```
// Switching to Alert
```

```
Alert alert = driver.switchTo().alert();
```

```
// Capturing alert message.
```

```
String alertMessage= driver.switchTo().alert().getText();
```

```
// Displaying alert message
```

```
System.out.println("ALERT DISPLAYED as : "+alertMessage);
```

```
Thread.sleep(5000);
```

```
// Accepting alert
```

```
alert.accept();
```

47.How to handle alerts?

```
driver.findElement(By.name("Alert Box")).click();
Alert alert= driver.switchTo().alert();
```

```
alert.accept();
alert.dismiss();
alert.getText();
alert.sendKeys("UserName");
```

```
driver.switchTo().alert().accept();
driver.switchTo().alert().dismiss();
driver.switchTo().alert().getText();
driver.switchTo().alert().sendKeys("UserName");
driver.switchTo().alert().sendKeys("Password");
```

48. Webtable alerts

WebTable - columns,rows ?

```
//*[@id="resultTable"]/thead/tr/th[2]
```

```
//*[@id="resultTable"]/thead/tr/th----->Number of columns 1 of 8
```

```
//*[@id="resultTable"]/tbody/tr[1]/td[2]
```

```
//*[@id="resultTable"]/tbody/tr[5]/td[2]
```

```
//*[@id="resultTable"]/tbody/tr/td[2]----->Number of rows 1 of 45
```

```
for(int i=1;i<=45;i++) //rows
```

```
{
```

```
for(int j=2;j<=8;j++) //column
```

```
{
```

```
    //*[@id="resultTable"]/tbody/tr[1]/td[2]
```

```
    //*[@id="resultTable"]/tbody/tr[1]/td[3]
```

```

        /**[@id="resultTable"]/tbody/tr[1]/td[4]

        -----

        -----

        /**[@id="resultTable"]/tbody/tr[1]/td[8]
        /**[@id="resultTable"]/tbody/tr[2]/td[2]
        /**[@id="resultTable"]/tbody/tr[2]/td[3]
        /**[@id="resultTable"]/tbody/tr[2]/td[4]

        -----

        -----

        /**[@id="resultTable"]/tbody/tr[2]/td[8]
        /**[@id="resultTable"]/tbody/tr[3]/td[2]
        /**[@id="resultTable"]/tbody/tr[3]/td[3]
        /**[@id="resultTable"]/tbody/tr[3]/td[4]

        -----

        -----

        /**[@id="resultTable"]/tbody/tr[2]/td[8]
        /**[@id="resultTable"]/tbody/tr[i]/td[j]
    }
}

```

49. What is Page Object Model

A: POM is nothing but writing a java file for each web page.

For each webpage ,write corresponding methods in a java file this method is known as POM .

Tc01 AddNationalities


```

@BeforeTest
public void TestSetup()throws Exception {

    driver = TestBrowser.OpenChromeBrowser();
    String TestURL = "https://opensource-demo.orangehrmlive.com/";
    driver.get(TestURL);

}

@Test
public void Book_OneWay_Flight() throws Exception {

    driver.manage().timeouts().implicitlyWait(60, TimeUnit.SECONDS);

    LoginPage L1 = new LoginPage();
    L1.LoginPage(driver);
    L1.Login();

    Nationalities N1= new Nationalities();
    //N1.Nationalities(driver);
    N1.AddNationality();

    HomePage H1= new HomePage();
    H1.HomePage(driver);
    H1.Logout();

}

@AfterTest
public void TestCloser()throws Exception {
    driver.quit();
}

```



Step1



Step2



Step3

For each page we r creating a java file and for that java file we r creating a onject.

Obj.method name we r calling.

50. What is the difference between POM and Page factory?

Page Object is a class that represents a web page and hold the functionality and members.

Page Factory is a way to initialize the web elements you want to interact with within the page object when you create an instance of it.

1. A Page Object Model is a test design pattern which says organize page objects as per pages in such a way that scripts and page objects can be differentiated easily. A Page Factory is one way of implementing PageObject Model which is inbuilt in selenium.
2. In POM, you define locators using '**By**' while in Page Factory, you use **FindBy** annotation to define page objects.

3. **Page Object Model** is a **design approach** while **PageFactory** is a **class** which **provides implementation of Page Object Model** design approach.

Advantages of Page Object Model Framework:

- **Code reusability** – We could achieve code reusability by writing the code once and use it in different tests.
- **Code maintainability** – There is a clean separation between test code and page specific code such as locators and layout which becomes very easy to maintain code. Code changes only on Page Object Classes when a UI change occurs. It enhances test maintenance and reduces code duplication.
- **Object Repository** – Each page will be defined as a java class. All the fields in the page will be defined in an interface as members. The class will then implement the interface.
- **Readability** – Improves readability due to clean separation between test code and page specific code