**Where have you actually used Java Object Oriented Concepts in Selenium Automation Framework? – Highly Important Question**

**#1. ABSTRACTION**

* Abstraction is the process of hiding the implementation details and showing the functionality to the users.

Let’s see an example of data abstraction in Selenium Automation Framework.

* In Page Object Model design pattern, **we write locators and the methods in a Page object Class.**
* We utilize these locators **in tests** but we can’t see the implementation of the methods. Literally we hide the implementations of the locators from the tests.
* In Java, abstraction is achieved by interfaces and abstract classes. Using interfaces, we can achieve 100% abstraction.

**2. INTERFACE**

* WebDriver is an Interface.
* JavaScript Executor
* TakeScreenshot

**WebDriver driver = new ChromeDriver();**

* Here, we are initializing Chrome browser using Selenium WebDriver.
* It means we are creating a reference variable (driver) of the interface (WebDriver) and creating an Object.
* Here **WebDriver is an Interface** as mentioned earlier and **Chromedriver is a class.**

**#3. INHERITANCE**

* The mechanism in Java by which one class acquires the properties (instance variables) and functionalities of another class is known as Inheritance.
* We create a **Base Class i**n the Automation Framework to initialize **WebDriver interface, WebDriver waits, Property files, Excels, etc., in the Base Class**.
* We **extend the Base Class** in other classes such as **Tests and Utility Class**.
* **Here we extend one class** (Base Class like WebDriver Interface) into other class (like Tests, Utility Class) is known as Inheritance.

**4. POLYMORPHISM**

* Polymorphism allows us to perform a **task in multiple ways.**
* Combination of overloading and overriding is known as Polymorphism. We will see both overloading and overriding below.

**#1. METHOD OVERLOADING**

* A class having multiple methods with same name but different parameters is called Method Overloading
* We use **Implicit wait in Selenium**
* **Implicit wait** is **an example of overloading**.
* In Implicit wait we use different time stamps such as SECONDS, MINUTES, HOURS etc.,
* **Action** class in TestNG is also an example of overloading.
* **Assert class in TestNG** is also an example of overloading.

**2. METHOD OVERRIDING**

* We use a method which was already implemented in another class by changing its parameters. To understand this you need to understand Overriding in Java.
* Declaring a method in child class which is already present in the parent class is called Method Overriding.
* Examples are **get and navigate methods of different drivers in Selenium .**

**5. ENCAPSULATION**

* Encapsulation is a mechanism **of binding code and data (variables) together in a single unit.**
* **All the classes in a framework are an example of Encapsulation**. We use access modifiers like Private,public to encapsulate
* **In POM classes, we declare the data members using @FindBy and initialization of data members will be done using Constructor**

**Challenges Faced In Selenium Automation**

**Captcha or OTP Handling**

**Limitations to Cross-browser testing**

* Sometimes our web application may not respond the same in different browsers, and there might be a possibility that our website works fine on Chrome but not on Firefox.

**Handling Dynamic Elements**

**Popup and Alert Handling**

* Different types of popup and alerts occur while interacting with a web application, theare is a difficulty in handling the issue of native pop-ups.
* We can handle some type of Popup and Alerts, but not all types,
* Examples of such notifications can be; “Allow/Decline camera access,” “Allow/Decline microphone access,” etc. These are browser level notifications that are handled in different ways depending upon the browsers.

**Problem of test flakiness**

* Tests are said to be flaky when they don’t give consistent results each time that they are run. They pass sometimes and fail the rest of the time. Flakiness in the test happens due to many reasons.
* For example, a UI element being validated is dynamic and takes some time to appear on the screen. It sometimes appears on time and sometimes it takes longer and by that time the test reverts it as a failure.

**Timeout or Sync Issue**

**Page Loading Issues**

* Some of the web pages in a web app are user-specific and load different elements depending on the user. Some features even appear based on the user’s previous activity. So some times selenium tests may not perform properly.

**Integration with different software**

* Since Selenium is open-source and we all are using many open source software like TestNG, Maven, Jenkins, Extent Reports, etc. so integration between these tools is sometimes a very challenging task.

**Smart locators**

* As we all know that locators are the core part of any scripting and we need to keep on enhancing our xpath and css for script stability, because if xpath and css are not proper then chances are very high that script might fail in upcoming releases.

**Waits in Selenium:**

**Synchronisation in automation or Selenium:**

* In Selenium, synchronization ensures that the code and applications execute in more efficiently to carry out the expected operations.

**How to achieve synchronization in the Selenium Web driver?**

* We can achieve synchronization in selenium by Using waits

**Types of Synchronizations in Selenium**

1. Thread.Sleep
2. Explicit Waits
3. Implicit Wait
4. Fluent Wait

**1)Why do we get exceptions when interacting with web elements**

**Reasons**

1.Element we try to findout is NOT present in DOM

2.Element is present but NOT visible in DOM

3.Element is present but NOT interactable

**2) How to resolve by exceptions?**

By using different waits like thread.sleep

**Thread sleep:**

* This method **will stop the execution for the specified duration of time**, irrespective of whether the element is found or not on the web page.

**Implicit wait:**

* We can **set the timeout for a specific amount of time**
* If finding web element is successful before the duration specified, it moves on to the next line of code execution, thereby **reducing the time of script execution**
* If finding web element is NOT successful .it will throwing the **NoSuchElementException.**

The key point to note here is, unlike Thread.sleep**(), it does not wait for the complete duration** of time.

**Implicit wait – When it is applicable ? –**

During initialization to driver,quit() it is applicable

**ADV of IMPLICIT WAIT**

* Better than thread.sleep

**Method name**

driver.manage(),timeout().implicitwait(30,TiMEUNIT.seconds);

**Problem in using Implicit wait?**

* Implicit wait will check that IF element is present in DOM or NOT,If present it will click

**IT wont worry about whether the element is INTERACTABLE or VISIBLE ,So it will throw exceptions**

* It will **wait for ALL the elements**
* We cannot **give conditions** like wait for the element to be clickable or visible

**What is Polling?**

* Every 500 miliseconds selenium will check element is present or NOT
* If present it will click if NOT it will wait another 500 miliseconds
* This will continue till the max given time is reached

**Explicit Wait:**

* **Explicit Wait** will command the **webdriver to wait until several conditions are met**

**How to use Explicit wait:**

1.Create webdriverwait instance(object)

2.wait until (condition are met)

**Some important conditions are:**  
1.elementTobeClickable()

2.elementTobeSelected()

3.presenceofElmentLocated()

4.textTobePresentInElement()

5.alertIsPresent()

**Syntax:**

WebDriverWait wait = new WebDriverWait(driver,30);

wait.until(ExpectedConditions.elementTobeClickable(By.xpath("//div[contains(text(),'COMPOSE')]")));

// click on the compose button as soon as the "compose" button is visible

**Adv of explicit wait:**

* It will wait until certain conditions are satisfied ,wherease implicit wait cannot wait with conditions like element Is clikable or visible or selectable

**Disadv of explicit wait:**

* If there is a network issue and if your application performs slow then implicit wait is better

**Can we mix implicit and explicit wait together?**

**YES ,we can use it wont throw error ,**

BUT don’t use it is not good practice bcz it will take too much time for execution (implicit wait + explicit wait) TOO much time

**Fluent Wait:**

* Fluent Wait gives  **maximum amount of time** untill **certain conditions are met**
* **Fluent Wait looks for a web element repeatedly at regular intervals** until timeout happens or until the element is found.

**Syntax**

Wait wait = new FluentWait(WebDriver reference)

.withTimeout(timeout, SECONDS)

.pollingEvery(timeout, SECONDS)

.ignoring(Exception.class);

WebElement foo=wait.until(new Function<WebDriver, WebElement>() {

public WebElement applyy(WebDriver driver) {

return driver.findElement(By.id("foo"));

}

});

**Difference between GET and Navigate Methods?**

| **sl.no.** | **get()** | **navigate()** |
| --- | --- | --- |
| 1 | It is responsible for loading the page and waits until the page has finished loading. | It is only responsible for redirecting the page ,It will NOT wait for the the page to finish loading. |
| 2 | It cannot Store the history of the browser. | It Stores the browser history |
| 3. | We cannot navigate Forward and backward | We can navigate Forward, Backward and also refresh the page in navigate |

**Difference ways to take screenshot?**

**Steps to take Screenshot using TakesScreenshot interface:**

**----------------------------------------------------------------------------------------------**

1. Using Selenium's **TakesScreenshot Interface,create object**

2. call **GetScreenshotAs** method and store it in a file as **Sourcefile**

3. By using java.Io.File save the file in any location as **Destinationfile**

4. copy the screenshot in destinationFile (image) file, **using FileUtils.(commons-io)** or we can use Selenium's **FileHandler** as well to copy the file.

Note – Whenever there is a limitation for taking screenshot,

Example – We need to take screenshot after Alert box ,it will throw UnhandledAlertexception

To overcome this ,we wil use ROBOT class for taking screenshot

Also Takesscreenshot will NOT capture complete screenshot whereas by using ROBOT class we can do that

TakesScreenshot screenshot = (TakesScreenshot) driver;

File sourcefile = screenshot.getScreenshotAs(OutputType.***FILE***);

File destinationfile = **new** File("C:\\Users\\1576486\\Documents\\TestFiles");

FileHandler.*copy*(sourcefile, destinationfile);

**Steps to take Screenshot using Robot class:**

**-----------------------------------------------------------------------**

1) Create an object of Robot class.

2) Get the screen size using getscreensize()method from Toolkit and store.

3) Create Object of Rectangle class and pass the screen size.

4) Use createScreenCapture(rectangle) of Robot class and capture screenshot and store as source

5) By using java.Io.File save the file in any location as Destinationfile

6) By using ImageIO.write pass (sourcefile,format,destinationfile)

Robot robo = **new** Robot();

Dimension screensize=Toolkit.*getDefaultToolkit*().getScreenSize();

Rectangle rectangle = **new** Rectangle(screensize);

BufferedImage source=robo.createScreenCapture(rectangle);

File destinationfile2 = **new** File("C:\\Users\\1576486\\Documents\\Selenium\\robot.png");

ImageIO.*write*(source, "png", destinationfile2);

**Different ways to refresh a browser:**

1.Using navigate.Refresh() command

2.driver.getCurrentUrl()

3.Javascript Executor: location.reload or history.go(0)

4.Press F5 using Robot class

**1.** Using navigate.Refresh() command

driver.navigate().to("https://www.google.co.in/");

driver.findElement(By.*name*("q")).sendKeys("Nazeer");

Thread.*sleep*(2000);

driver.navigate().refresh();

2.driver.getCurrentUrl()

driver.get("https://www.google.co.in/");

driver.findElement(By.*name*("q")).sendKeys("Nazeer");

driver.get(driver.getCurrentUrl());

* In this method,refresh is happened by loading the get URL again using getcurrentURL

3. Javascript Executor Using location.reload method or history.go(0)

JavascriptExecutor executor = (JavascriptExecutor) driver;

executor.executeScript("location.reload()");

4.Press F5 using Robot class

* In this method ,we will use Robot class will create object
* Usekeypress method to select F5

**Different ways to maximise the browser?**

1.**Using Selelnium built-In Maximise method**

driver.manage().window().maximise()

2.**Using Setsize method by providing Dimension in Dimension class**

Dimension dimension = **new** Dimension(1366,768);

driver.manage().window().setSize(dimension);

Here we giving the screen dimension in Dimension class

**3.Chrome options**

ChromeOptions chromeoptions = **new** ChromeOptions();

chromeoptions.addArguments("--start-maximsed");

**Difference between Sleep() and SetSpeed()?**

* setSpeed will apply wait before every Selenium operation.
* setSpeed is deprecated in webDriver
* Thread. sleep() will set up wait only for once for the particular selenium operation when called

**Example :**

* Setspeed(2000) for 3 operations
* It will wait for 6000 miliseonds
* If you give threadsleep of 2000 miliseconds for particular operation ,it will wait only for 2000 seconds

**Can we launch browser without system.setproperty()?**

Yes. We can do it By 2 ways

1.By using **Webdriver manager,**we can avoid System.setproperty()method to launch browser

2.By adding  **driver location in System environment variables** ,**Restart the system** then no need of System.setproperty()method to launch browser

**Handling authentication Popup?**

**Problem :** We cannot Inspect and Sendkeys in these pages

**Solution1** : We can use third-party tools like AutoiT

**Solution2:**We can directly send credentials via URL

[http://username:password@url.com](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbm1DN1BvTTFud1JNdTlRNlJLczlmcEVIMFRGQXxBQ3Jtc0tsLVBidFBBSHZsb3puajBQbjRJMGVUYnBoc2x6b1gwMEpsZ2VGY0dIV21FVmhabWxEOURfUzZZeXVKTVlVWC1neUVKSWNsNjJIOXgtTFRabHdRWURoSkp6NGdDOFA0T0N4ZFk1RDdjOVNvU2hZRHppRQ&q=http%3A%2F%2Fusername%3Apassword%40url.com) //

**//\*[@id='search']//following::h3//following::div/cite**

**Note : Id you will find only when you inspect element**

**Different ways to give Press Enter key ??**

**1.By using Sendkeys + Keys.Enter**

WebElement searchbox = driver.findElement(By.*name*("q"));

//searchbox.sendKeys("Trichy" + Keys.ENTER);

**2.By using Submit method**

WebElement searchbox = driver.findElement(By.*name*("q"));

searchbox.sendKeys("Trichy");

**3.By using Robot class**

Robot robot = **new** Robot();

robot.keyPress(KeyEvent.***VK\_ENTER***);

robot.keyRelease(KeyEvent.***VK\_ENTER***);

**4.By using \n in the double quotes of sendkeys**

searchbox.sendKeys("Trichy \n");

**How to find element without find element?**

**1.By Switch to Active element method**

driver.switchTo().activeElement().sendKeys("Nazeer");

2.By using FindBy method in Pagefactory

**How to rerun the failed test cases? - Important Question**

**3 ways we can do it**

* **Manual Run :**Whenever test case failed,testing-failed.xml file will be created, so we can re-run the failed test cases again and again
* **Using IRetryAnalyser Interface :** Create Retry class implements IRetryAnalyser and override the method,Go to the test case class @test mention (retryanalyser = Retry.class)
* **Testcase with** (retryanalyser = Retry.class) only run re-run multiple times

**Here Limitation is that for all the test case we need to mentiond @test**(retryanalyser = Retry.class) To overcome this problem Using IAnnotationTransformer Interface with IRetryanalyser

* **Using IAnnotationTransformer** : Create a class called transformer implements IAnnotationTransformer with IRetrytransformer ,Overide the method with @test annotation with mapping on SetRetyAnalyser (Retry class)

**How to Scroll a webpage?**

* **We have to use JavaScript Executor to scroll-up**
* **We can also use Robot class**

**Javascript function to scroll down:**

1.Window.scroll (x,y) or Window,scrollBy(x,y)

**Javascript function to scroll Up**

Window.scroll (x,-y)

**Scrolling to Bottom of a Page:**

Window.scrollTo(0, document.body.scrollHeight)

**Scrolling to Top of a page:**

Window,scrollTo(0,0)

**Scroll to a particular position where an element is present:**

“argument[0].scrollintoView(true);”,element)

**Using Robot class**

Robot robot = new Robot();

robot.Keypress(KeyEvent.VK\_Pagedown);

robot.Keyrelease(KeyEvent.VK\_Pagedown);

1.Scroll to Particular position and revert back to original position

driver.get("https://echoecho.com/htmlforms11.htm");

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

JavascriptExecutor executor = (JavascriptExecutor) driver;

executor.executeScript("window.scroll(0,450)", "");

Thread.*sleep*(3000);

executor.executeScript("window.scroll(0,-450)", "");

2.Go to the Bottom and Top of the page

// 2.Go to the Bottom and Top of the page

JavascriptExecutor executor = (JavascriptExecutor) driver;

executor.executeScript("window.scrollTo(0,document.body.scrollHeight)", "");

Thread.*sleep*(3000);

executor.executeScript("window.scroll(0,0)", "");

3.Go to Specific element

WebElement link = driver.findElement(By.*linkText*("online tool"));

executor.executeScript("arguments[0].scrollIntoView(true);", link);

**Using Robot Class scroll Down and Up:**

Robot robot = **new** Robot();

robot.keyPress(KeyEvent.***VK\_PAGE\_DOWN***);

robot.keyRelease(KeyEvent.***VK\_PAGE\_DOWN***);

Thread.*sleep*(3000);

robot.keyPress(KeyEvent.***VK\_PAGE\_UP***);

robot.keyRelease(KeyEvent.***VK\_PAGE\_UP***);

**How to verify whether a link is underlined?**

* Use getCSSvalue method to get the method CSS ,then print it using Syso
* After that mousehover using Action class
* Then use getCSSValue again and Print it ,output will be underlined

driver.get("https://www.google.co.in/");

WebElement tamil=driver.findElement(By.*xpath*("//\*[@id=\"SIvCob\"]/a[5]"));

String beforehovering = tamil.getCssValue("text-decoration");

System.***out***.println("Before hovering" + beforehovering);

Actions actions = **new** Actions(driver);

actions.moveToElement(tamil);

actions.perform();

String afterhovering = tamil.getCssValue("text-decoration");

System.***out***.println("Before hovering" + afterhovering);

**Assert vs Verify?**

* Both used to validate the element is present or NOT
* When assert failed ,it will stop the code execution
* When verify failed it wont stop the code execution,program will continue
* Verify is called softassertion,We use Softassert class
* Assert is called Hardassertion,We use Assertclass

**How to send keys without sendkeys method?**

1.Most used and Simple way is Using Sendkeys method

2.Using Javascriptexecutor Interface

3.Using Robot class

2.Using Javascriptexecutor Interface by using document.getElementsBy.Value method

JavascriptExecutor executor = (JavascriptExecutor) driver;

executor.executeScript("document.getElementsByName('q')[0].value = 'Duckling'", "");

Using Javascriptexecutor Interface by using argument.value method

executor.executeScript("arguments[0].value='Riya'", searchbox);

Using robot class

driver.switchTo().activeElement();

Robot robot = **new** Robot();

robot.keyPress(KeyEvent.***VK\_R***);

robot.keyRelease(KeyEvent.***VK\_R***);

robot.keyPress(KeyEvent.***VK\_I***);

robot.keyRelease(KeyEvent.***VK\_I***);

robot.keyPress(KeyEvent.***VK\_Y***);

robot.keyRelease(KeyEvent.***VK\_Y***);

robot.keyPress(KeyEvent.***VK\_A***);

robot.keyRelease(KeyEvent.***VK\_A***);

**How to select all the checkboxes in the page?**

* First wen need to find the xpath for all checkboxes like here as //input[@type='checkbox'
* Store it in as LIST
* Then use for each loop to click on all the checkboxes

List<WebElement> allboxes =driver.findElements(By.*xpath*("//input[@type='checkbox']"));

**for** (WebElement webElement : allboxes) {

webElement.click();

}

**TestNG Annotations and Hierarchy of Annotations:**

**Hierarchy:**

beforeSuite

beforeTest

beforeClass

beforeMethod

In test

afterMethod

afterClass

afterTest

afterSuite

**When 2 @tests there means**

beforeSuite

beforeTest

beforeClass

beforeMethod

In test 1

afterMethod

beforeMethod

In test 2

afterMethod

afterClass

afterTest

afterSuite

**How to execute a test multiple times with TestNG ?**

* Using invocationCount=2 ,it will give output 2 times

**@Test(invocationCount=3)**

**public void f() {**

**System.*out*.println("In test");**

**}**

**What is Timeout in testng? How to set timeouts for a testcase in TestNG?**

* Timeout will give testcase certain time to execute,if failed test case failed to execute on the given time test case will fail

@Test(timeOut=2000)

**public** **void** f() {

System.***out***.println("In test");

}

**How to handle exceptions in TestNG:**

* We can use expectedExceptions,so that even if exception occurs,testcase will pass

**How to use?**

* By giving the exception class to expectedExceptions

@Test(timeOut=2000,expectedExceptions = ThreadTimeoutException.class )

public void testing() throws InterruptedException {

Thread.*sleep*(3000);

System.*out*.println("In test");

}

**Significance of Always Run method:**

**AlwaysRun :** This is used to make sure a **method always runs** even if the parameters on which the method dependson is fails.

Example – When you have 2 methods,If second method is dependson First method,then if first method fail means second method wont run,BUT by giving Always run=true,we will run the second method

@Test(timeOut=2000 )

public void Myfather() throws InterruptedException {

Thread.sleep(3000);

System.out.println("Myfather");

}

@Test(dependsOnMethods="Myfather",alwaysRun=true)

public void Me() throws InterruptedException {

System.out.println("Me");

}

}

**How to Download Browser drivers automatically?**

* WebdriverManager is a library to automate management of the webdrivers
* It helps to download webdriver with current browser version

To use simply call **WebDriverManager.chromedriver().setup()**

* **Copy the webdrivermanager dependency and paste in POM of maven**

**How to hit a URL without using Get or Navigate method:**

* We can use JavascriptExecutor interface
* By using **Window.location** \’url’\ method of javascriptexecutor
* String url = [www.google.com](http://www.google.com)

JavascriptExecutor executor = (JavascriptExecutor) driver

Executor.execturescript(**Window.location =** ’url’\)

**Handling Pagination:**

1.String type Arraylist is created to store the names

2.Find the Xpath of the pagination list and Store as list

3.With size method of findelements we can get the size of the list

4.Find the nextbutton and store it and

5.Use if condition for executing pagination is greater than 1 means run

6.we can use another if else loop to stop the loop when it have disabled message in class

Use Do while to loop to iterate if pagination is available

**How To Use JavaScriptExecutor in Selenium WebDriver?**

[**https://www.lambdatest.com/blog/how-to-use-javascriptexecutor-in-selenium-webdriver/**](https://www.lambdatest.com/blog/how-to-use-javascriptexecutor-in-selenium-webdriver/)

**What is JavaScriptExecutor?**

* JavaScriptExecutor is an interface which helps to execute JavaScript from Webdriver

**What will you if different web locator such as ID, Name, CSS selector,Xpath, link text, partial linkt ext is not working?**

* We can use Javascriptexecutor Interface to work on the webElement

If you handling a tricky Xpath ,You can use **Javascriptexecutor**

**What will you do if click() method may not work on your browsers?**

* We can use Javascriptexecutor Interface

**Types of JavascriptExecutor consists of two methods**

1. **executeScript method**
2. **executeAsyncScript method**

**Uses of** **JavascriptExecutor:**

* To Refresh a Browser
* To Scroll to a webpage
* To Send keys without using sendkeys method
* **To hit a URL without suing Get or Navigate method**
* **If all locators are NOT working**
* **For tricky xpath**

**Additional uses**

* To **Click on a Button**
* **To Handle Checkbox** by passing the value as true or false
* **To generate Alert Pop window** in Selenium Webdriver
* **To get the innertext of the entire webpage** in Selenium
* To get the **Title** of our webpage
* To get the **domain** name
* To get the **URL** of a webpage
* To get the **Height and Width** of a web page
* **To find a hidden element in selenium** using JavaScriptExecutor
* **To navigate to a different page using Javascript**
* **Adding an element in DOM**

**What is Robot Class?**

* Robot Class is used in Selenium to control mouse and keyboard events which interacts with **Window OS download-popups and print-pop-ups**
* Selenium Webdriver cannot handle these pop-ups/applications, so in Java version 1.3, robot class was introduced which can handle OS pop-ups/applications

**Benefits of Robot Class**

* Robot Class can **simulate Keyboard and Mouse Event**
* Robot Class can help **in upload/download of files** when using selenium web driver
* Robot Class can easily **be integrated with automation framework** (keyword, data-driven or hybrid)

**Difference Between Gettext and GetAttributes?**

* **getText() method** returns the innerText of an element. Which is visible
* **getAttribute() method** returns the text contained by the HTML attributes.

**How to Make Implement Modularization in Automation?**

* **Create Separate functions for all the functionality** (This will make your project modular which will help to understand,readable,make changes in future)
* **Add comments wherever applicable**
* **Keep classes small**
* **Name methods carefully**

**Explain different types of Selenium commands.**

* **Actions:** These are the commands **interacting directly with web applications**.
* **Accessors:** These are the commands which allow **users to store values** to a user-defined variable.
* **Assertions:** They enable a comparison of the current state of the application with its expected state.

**What is meant by a locator and name a few different types of locators present in Selenium.**

* **A locator is an address** for uniquely identifying web elements within a web page.

There are different types of locators present in Selenium to identify web elements uniquely and accurately like:

* ID
* ClassName
* Name
* TagName
* LinkText
* PartialLinkText
* Xpath
* CSS Selector
* DOM

### What is meant by an exception test in Selenium?

If you have used expectedExceptions attribute with @Test method then Whenever Exceptions thrown In @Test method, It will compare It with expectedExceptions attribute's value.

If both match then It will be handled by TestNG.

That means your @Test method will be pass and remaining part of that method will be skipped

**/expectedExceptions attribute with @Test method will handle NoSuchElementException**

@Test(**expectedExceptions = NoSuchElementException.class**)

public void testCaseOne\_Test\_One() {

System.out.println("Executing testCaseOne\_Test\_One.");

**//This element Is not available on page so It will throw NoSuchElementException.**

driver.findElement(By.xpath("//input[@id='10']")).click();

**How to handle window pop-ups?**

If ALERT POP-UP WE CAN USE alert ,accept or dismiss methods

Or third party tools like Auto it

Also lately Robot class introduced to handle window based popups

**Selenium** – Selenium Automates Web browser

**Selenium IDE** used for?

* Selenium IDE (Integrated Development Environment) is primarily a **record/run tool** that a test case developer uses to develop Selenium Test cases.

**Selenium WebDriver**

* It is an open source tool for automated testing of webapplications

**Selenium Web Driver in Java:**

* **Selenium WebDriver is an Interface which contains different methods (eg., *get(), getTitle(), close() etc.,* ).**

**What are all the browsers Selenium webdriver supports?**

* Selenium WebDriver is that it supports all the major browsers like Firefox, Google Chrome, Apple Safari, IE, Edge, and Opera. **Every browser has a specific WebDriver for executing automation** scripts.

**Selenium Grid**

* Selenium Grid allows us to run tests in parallel on multiple machines,

**ChromeDriver** is a separate executable file used by Selenium WebDriver uses to control Chrome.

The main purpose of the ChromeDriver is to launch Google Chrome

**setProperty()** method sets the system property indicated by the specified key.

**Selenium Architecture**

A diagram of a webdriver

Description automatically generated

**1. Selenium Client Libraries Binding**

* Testers use the language which they are comfortable with to write automation scripts ,Here we going to write with Java,so we need Java libraries

**2. JSON Wire Protocol**

* JavaScript Object Notation (JSON) Wire Protocol facilitates the capability of transferring the data between the Server and Client on the web. It is a REST API

**3. WebDrivers**

* Drivers are useful for working with browsers and running the automation script

**4. Browsers**

* The best part about the Selenium WebDriver is that it supports all the major browsers like Firefox, Google Chrome, Apple Safari, IE, Edge, and Opera.
* Every browser has a specific WebDriver for executing automation scripts.

**How Selenium WebDriver works Internally?**

WebDriver driver = **new** ChromeDriver(); //

Driver.get("https://www.data-flair.training");

After running the code, it will launch the Chrome browser this time

**WebDriver** is an interface and all the methods which are declared in Webdriver interface are implemented by respective **driver class**.

**new** ChromeDriver(); // Object is created by using new keyword

**General information:**

* Selenium WebDriver is an Interface which contains different methods (eg., get(), getTitle(), close() etc., ).
* All the third party Browser vendors implement these methods in addition to their browser specific methods.
* Selenium developers don’t know how all these browsers work. So Selenium developers just declare methods whatever they required and leave the implementation part to the browser developers.

A diagram of a web driver

Description automatically generated

driver.get method will navigate to a page given by the URL.

**Questions:**

**What are the advantages/benefits of Automation Testing?**

This is one of the common interview questions. Some of the advantages of Automation Testing include:

* **Save Money and Time.**
* **Automation Testing increases the accuracy** as there is no chance for human errors.
* **Code Reusability.** Create once and execute multiple times with Less or no maintenance.
* **Easy Reporting.** Automatically generates the reports after the execution of the test cases.
* **Easy for Compatibility Testing** by enabling the parallel execution in combination with OS and browser environments.
* **More Reliable, Powerful and Versatile.**
* **Automation Tools allows us to integrate with the Jenkins, GitHub etc.,**
* **Mostly used for Regression Testing.**

**How many Test cases have you automated per day?**

This is one of the tricky Selenium interview questions. Actually, it depends on the complexity and length of the Test case scenario

**.When the complexity is limited**, you can **automate two to five test cases** and one or fewer test cases when the complexity is high in a day.

**What do you mean by Selenese?**

Selenese which is called the Selenium command is the set of the selenium commands that run your test cases. For example, open (URL) is a Selenium command which opens the specified URL in the specific browser.,

**Actions:** We can use for interactions on the elements like mouse and keyboard used for intreaction

**Accessors**: These are used for storing the values in the variables.

**Assertions:** Used to compare and check the expected value with actual values

**What is the difference between Absolute path and Relative Path?**

**Absolute Path:**

* This path starts from the root node and ends with the desired elements node.
* It starts with a single slash (/)

Example: /html.body/div/td/input

**Relative Path:**

* This path starts from any node in between and to the desired elements node.
* It starts with a double slash (//)

Example: //input/example[@id=name]

**What is the difference between findElement() and findElements()?**

**findElement() :**

* It is a locating mechanism to find the **ONE element** within the current page and **return a single element.**

**Example:**

WebElement loginLink = driver.findElement(By.linkText("Login"));

**findElements():**

* It is a **locating mechanism** to **find all the elements** within the current page and **return the list of all elements.**

**Example:**

List<WebElement> listOfElements = driver.findElements(By.xpath("//div"));

**What is the major difference between driver.close() and driver.quit()?**

**driver.close():**

* It’s a command which **closes the current window**
* For example, if we have multiple browser windows open then, by using this command we can close the window with which the focus is.

**driver.quit():**

* This command **closes all the browser windows which are open.**
* For example, if we have multiple browser windows open then, this command can close all the windows at once.

**Can Selenium handle windows-based pop-up?**

* As we already know that Selenium is an automation testing tool that can work with only web applications. Therefore, the pop-up in windows cannot be handled by Selenium. But, by integrating with the third-party tools we can overcome this problem.

**What is an Object Repository and Explain how we can create the Object Repository in Selenium?**

* C**ollection of web elements and locators**  are called **Object Repository.**
* The only thing is the locators are stored in **a centralized location** than hard coding them in the test scripts.

**What are the different kinds of frameworks?**

**Data-driven framework:**

* Test data is generated from external files like excel, CSV, XML, etc.
* We can use Apache POI for data driven framework

**Keyword-driven framework:**

* It uses keyword,to perform a set of Actions
* No programming knowledge is needed for this one

**Module-based testing framework:**

* Independent test scripts are written for each module.
* These scripts are then clubbed together as a whole.

**Behaviour driven testing framework:**

* The test scripts are written in a way which non-technical people will understand the features

**Hybrid testing framework:**

* Combination of more than Keyword and Datadriven frameworks by leveraging the benefits of each is called a Hybrid testing framework

**What is Exception?**

* Exception” is an unusual event that disrupts the normal flow of program execution.

**What is Exception Handling in Selenium?**

* Exception handling is a mechanism **that detects and resolves runtime exception errors**

**Types of Exception:**

**Checked Exceptions:** These exceptions are handled while writing the code itself before the compile

**Unchecked Exceptions:** These exceptions get thrown at run time

**11 Common Exceptions in Selenium WebDriver**

**ElementNotSelectableException:** An element in disable state which is **Not be selectable** **in spite of being present in the DOM**

**ElementNotInteractableException:** An element in disable state where it **cannot be interacted** with (can not be clicked or able to send keys) **in spite of it being present in the DOM**

**ElementNotVisibleException:** **In spite of the element being present in the DOM,** it is **not visible** (can not be interactive).

**NoSuchElementException:** Webdriver is not able to find the elements during runtime, i.e., **the FindBy method cannot find a particular component**

**NoSuchFrameException**: Webdriver attempts **to switch to an invalid frame,** which is unavailable

**NoSuchWindowException:** Webdriver attempts to **switch to an invalid window,** which is unavailable

**NoAlertPresentException:** Webdriver attempts to **switch to an invalid alert**, which is unavailable

**StaleElementReferenceException:** The referenced element is no longer present on the DOM page. For example, the item belongs to a different frame than the current one or the user has navigated away to another page

**SessionNotFoundException:** Webdriver throws immediately after ‘quitting’ the browser

**TimeoutException:** The command did not complete in the specified time.

For example, the element didn’t display at the specified time. This is especially encountered when working with waits

**WebDriverException:** Webdriver throws immediately after ‘closing’ the browser

**What is the difference between the getwindowhandle() and getwindowhandles()?**

* **getwindowhandle():** This is used to get the address of the **current window** and **returns the data type of String.**
* **getwindowhandles():** This is used to get **the address of all the open windows** and **returns the data type of Set<String>.**

**What are the different types of Annotations used in Selenium?**

@Before Suite, @Test, @After Suite

**How can you debug the tests in the Selenium IDE?**

1.insert the breakpoints where we need to debug and execute the test step by step.

2.Later, Run the test case.

3.After this, the execution stops at the point where the breakpoint is kept.

4.You can click on the Run button to continue the execution of all the commands sequentially.

**How can you handle working with multiple windows in Selenium?**

* We can use getwindowhandle
* getwindowhandles
* switchTo windows
* selectWindow() command used to switch among the working windows.

H**ow can the message in the alert box be retrieved?**

storeAlert() command can be used to retrieve the message from the alert pop-up and store it in a variable.

**How can you verify the specific position of a web element?**

verifyElementPositionLeft and verifyElementPositionTop commands are used.

**When do you use AutoIT?**

As we already know that Selenium is used only to automate web applications. But if we want to handle or manage or maintain the GUI, HTML pop-ups then, we need the use of AutoIT.

**Can you mention why do you need the Session handling in Selenium?**

* Session handling is very important.
* There can be an issue where while running a particular test script, another test script can be triggered within the same host and same type of browser. This is the most important part and the reason why we need the session handling.

**Can you automate CAPTCHA?**

* The answer is No.
* We cannot automate CAPTCHA in Selenium.
* The concept of CAPTCHA is to ensure that the bots or automated test scripts should not have access to sensitive information, and that is why we cannot automate it. The captcha should be manually typed in order to continue the flow of execution of the test suite.

**How do we launch different browsers in Selenium?**

* A driver instance should be created for a particular browser which we need to work upon.
* WebDriver driver = new FireFoxDriver();
* WebDriver driver = new ChromeDriver();
* Here, the WebDriver is the interface where the Firefox, chrome and all other browser driver implementations are made.

**What are the challenges and Limitations of the Selenium Web Driver?**

* Not compatible to test the Windows or Desktop applications.
* Also, cannot be used to test the Mobile applications.
* No Built-in reporting and it must be integrated with the other tools to obtain the reports.
* Support for Image Testing is limited.
* The user should have prior programming knowledge.
* Since it is open-source, there is no vendor support.

**What are the Testing types that are supported by Selenium?**

* Regression Testing and Functional Testing are majorly supported by Selenium.
* Regression Testing:
* Re-testing:
* Functional Testing:

**Which API is used for database testing in the Selenium web driver?**

JDBC (Java Database connectivity) is used for database testing in Selenium. This allows us to write the SQL (Structured Query Language) queries and execute them.

**Can you write a small code snippet to launch a Chrome browser in Web Driver?**

public class ChromeBrowserLaunchDemo {

public static void main(String[] args) {

//Creating a driver object referencing WebDriver interface

WebDriver driver;

//Setting the webdriver.chrome.driver property to its executable's location

System.setProperty("webdriver.chrome.driver", "/lib/chromeDriver/chromedriver.exe");

//Instantiating driver object

driver = newChromeDriver();

//Using get() method to open a webpage

driver.get("http://javatpoint.com");

//Closing the browser

Driver.quit();

}

}

**Can you mention the challenges you face when handling the Ajax calls in Selenium Web driver?**

* Pause command in Ajax calls is not conventional and reliable in handling. Long pauses do cause the test case execution increases the time of execution. We can use the waitForCondition instead to test the AJAX applications.
* Encoding and Serialization in the AJAX applications make it difficult in developing the automation test case scenarios.

**Can you explain how can you handle colours in web driver?**

* We can use the command getCssValue(arg0) to fetch the colors by sending the color as the argument.

**Why is selenium so popular?**

* Transparency
* Platform Independent
* Fosters Continuous Integration Efforts
* Visibility in End-to-End Testing
* Reduces Turnaround Time
* Integration With Other Tools

**Is Selenium a framework?**

Yes, selenium is a software testing framework for web applications that allows you to automate web app testing. With languages like Java, Python, Ruby, C#

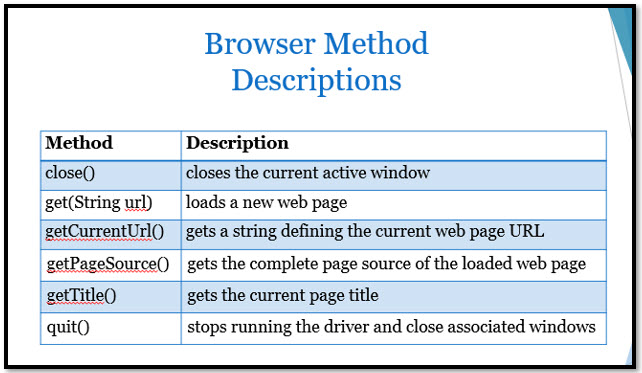
**Which language is best for selenium?**

**Python and Ruby** are the best binding languages for selenium

**Selenium Commands or Selenium Methods**

**1.Browser Methods**

* Browser Methods perform actions on a browser.



**2.** [**WebElement Methods**](https://blog.testproject.io/2020/03/05/selenium-webelement-methods/)

**What Is a Web Element?**

* Anything that is present on the web page is a WebElement such as text box, button, etc.

Deeper Answer:

* A Web Element is sometimes called an element.
* HTML stands for HyperText Markup Language which instructs the browser how to display content.
* HTML element contains a start tag, end tag and content between both tags

In alphabetical order, the following 16 methods carry out actions on the web page via WebElement Interface:

* clear
* click
* findElement
* findElements
* getAttribute
* getCssValue
* getLocation
* getRect
* getSize
* getTagName
* getText
* isDisplayed
* isEnabled
* isSelected
* sendKeys
* submit

**click()**

* The click() method is used to click an element.

**sendKeys()**

* we attempt to log into Any application by typing a Username and Password
* By Using sendkeys we can give Username and password

**getText()**

* The getText() method returns visible text of an element.

Example : If you try to login it throws error message, to get the error msg we can use get text method

**String errorMessage = driver.findElement(By.id("tp-message-error")).getText();**

**System.out.println("What Is The Error Message? " + errorMessage);**

Console shows “What Is The Error Message? Invalid username or password”.

**getAttribute()**

* The getAttribute() method returns an attribute’s current value or if there isn’t a value it will be NULL

**Example** : For Login Username and password,you need to find the default value of password, Then we can use attribute value and print it

**String** holderPassword = driver.findElement(By.id("password")).getAttribute("placeholder");

System.out.println("Which Sign In Credential Is Incorrect? " + holderPassword);

**clear()**

* clear() method clears a value from the text entry field

**Example** :If you enter username in Username field.Using clear method you can clear the text field to empty

**isDisplayed()**

* isDisplayed() method returns a boolean value by determining if an element is displayed or not displayed.
* If element is displayed then Boolean value is true
* If not Boolean value is false

The following code snippet implements isDisplayed() method is used to find the Signup for Free message id displayed or NOT

WebElement linkSignUpForFree = driver.findElement(By.id("tp-signup-link"));

**boolean** isSignUpForFreeLinkDisplayed = linkSignUpForFree.isDisplayed();

System.out.println("Is The Sign Up For Free Link Displayed? " + isSignUpForFreeLinkDisplayed);

linkSignUpForFree.click();

**isEnabled()**

* isEnabled() method returns true if an element is enabled and false if an element is disabled

**Example**

After clicking the Sign up for free link, the next page shows a disabled Sign Up button. Let’s verify the button is disabled on the Create your free account page:

**boolean** isSignUpButtonEnabled = driver.findElement(By.id("tp-sign-up")).isEnabled();

System.out.println("Is The Sign Up Button Enabled? " + isSignUpButtonEnabled);

**3.Navigation Methods (Chapter 3)**

**navigate().to()**

There are 2 navigate().to() methods in the Navigation Methods Category

navigate().refresh()

navigate().refresh() method refreshes the current page thereby reloading all WebElements

@Test

public void demoNavigationMethods() {

driver.navigate().to("https://blog.testproject.io/");

System.out.println("To Title: " + driver.getTitle());

driver.findElement(By.id("menu-item-7501")).click();

driver.findElement(By.id("username")).sendKeys("Rex.Jones@Test4Success.org");

driver.navigate().refresh();

System.out.println("Refresh Title: " + driver.getTitle());

}

**navigate().forward()**

* navigate().forward() method **moves forward one page in our browser’s history.**

**navigate().back()**

* The navigate().back() method **moves back a single page in our browser’s history.**

**Automation Testing Life Cycle:**

1. Determining The Scope Of Automation test
2. Selecting The Right Tool For Automation
3. Test Plan + Test Design + Test Strategy
4. Setting Up The Test Environment
5. Automation Test Script Development + Execution
6. Analysis + Generation Of Test Reports

**Determining The Scope Of Test Automation**

In this particular stage, the following things should be taken care of without a failure.

* Which modules of the applications can be automated and which not?
* Which test can be automated and how to automate them?
* Factors like cost, team size and expertise should also be considered.

**Selection of the Right Tool for Automation**

* It is the second phase of the ATLM process**. Identifying the** **right automation testing tool** is critical for the automation testing life cycle since automation testing is highly dependent on the tool used.
* The developers need to consider **budgetary constraints**, **the familiarity of the team with the automation tool used,** along **resources available with the team.**
* test engineer should define and evaluate the criteria for a pilot test for the automation tool.
* Testing personnel should then evaluate it based on the different criteria stipulated by the test engineer.

# **What Is TDD?**

* TDD stands for Test Driven Development.
* In this software development technique, we create the test cases first and then write the code which is based on test cases
* TDD, takes more time for development however, they tend to find very few defects
* TDD also helps in achieving high test coverage of about 90-100%.
* The most challenging thing for developers following TDD is to write their test cases before writing the code.

**What Is BDD?**

* BDD stands for Behavior Driven Development.
* BDD is an extension to TDD where instead of writing the **test cases**, we start by writing a **behavior.** Later, we develop the code which is required for our application to perform the behavior.
* BDD is considered a best practice when it comes to automated testing as it focuses on the behavior of the application

**What is Page Object Model (POM) ?**

* Page Object Model (POM) is a design pattern
* As per pom, a class is maintained for every web page.
* Every webpage has a separate class for functionality and object repositories
* Separate classes for every individual test.

Advantage of this model is that it reduces **code duplication** and **improves test maintenance.**

**What is Page Factory in Selenium?**

* **Another Use of Page Factory:** Page factory used for **initialization of Page objects**
* We use **initElements** method **to initialize the page objects**
* Page Factory is **an inbuilt concept** of **Page Object Model framework**
* It is also used to initialize Page class elements **without using** “**FindElement/s.”**
* In page factory ,we use **@FindBy** annotation to **find WebElement**

**Write code for two dimensional array(just one line syntax) or 3 dimensional array?**

**Two dimensional array:**

int[][] twoD\_arr = new int[10][20];

**Three dimensional array:**

int[][][] threeD\_arr = new int[10][20][30];

**How will you make sure web page is loaded using selenium and webdriver?**

1. selenium.waitForPageToLoad("5000");

Below is the Explicit wait and Webdriver specific code to achieve the same objective.

WebDriverWait check = new WebDriverWait(driver, 100);

1. WebElement elem = driver.findElement(By.id("myInvisibleElement"));
2. elem.click();
3. wait.until(ExpectedConditions.visibilityOf(elem));

**Can we execute a program without constructor ?**

**No.** You need object in a class,object needs constructor to initialize it,so without constructor we cannot execute a program

**Which are the file types used for data source for different frameworks?**

* CSV
* Excel
* XML

**If I am a particular page then I goes back to different page,then again if I want to came back to particular page means,so which method I should use?**

* driver.manage.**navigate()**
* driver.manage.**back()**

**How can you handle dynamic web elements in Selenium**? (IMPORTANT QUESTION)

* Solution : 1 :Using different types of waits – Sometime the synchronization is the root cause ,using waits we can fix
* Solution : 2 : Using Customized Xpath like single attribute,multiple attribute ,using contains or stars-with or etcs
* Solution 3 : Using CSS Selectors – Somecases CSS will be better option
* Solution 4 : Still we are facing difficulties means ,we can use JAVASCRIPT EXECUTOR Interface

**Use Relative XPath using contains or starts with text**

* If you observe a pattern in the attribute values like ID or Class of the web element.

For eg consider the HTML snippet:

<input type="submit" id=" submit\_334350" value="Subscribe">

This Subscribe button on the page has an ID with a dynamically changing number in it (‘334350’). This keeps on changing every time you refresh the page. But it always starts with submit. So you can use a relative XPath as given below to identify the web element:

XPath - //input[starts-with(@id, ‘submit\_’)]

Now, consider another example:

<input type="submit" id=" 1002-subscribe" value="Subscribe">

In this case you can write XPath as:

XPath - //input[contains(@id, ‘subscribe’)]

**Identify by index - If there is 2 elements with same locator value**

For example there may be two submit buttons with id starting with ‘Submit’.

In this case you can use findElements method and locate the element using the index.

driver.findElements(By.xpath(//\*[contains(@id, ‘submit’).get(0).click();

* Here get(0) is used to get the first web element with matching XPath.

**Use Multiple attributes to locate an element**

To identify a particular element you can use multiple attributes if a single attribute is not enough to identify your web element uniquely.

Xpath- //button[starts-with(@id, 'save') and contains(@class,'publish')]

* We can also use CSS selectors,
* and in some cases, switch to alternative locators when dealing with dynamic content.

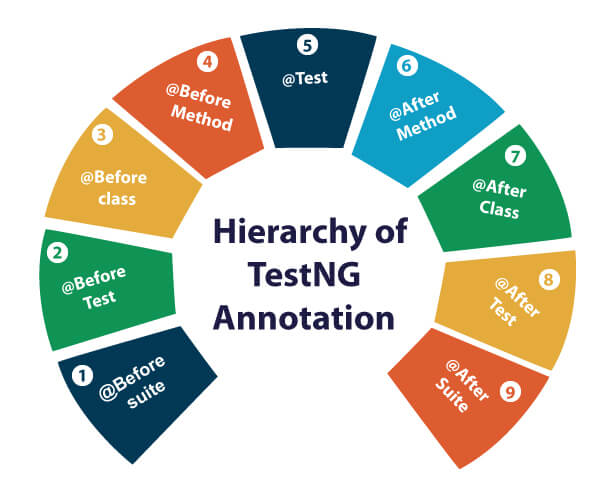
What is Framework?

It is an execution to run and maintain Automated tests

TestNG

* TestNG is an open source automated Testing framework
* TestNG, NG stands for "Next Generation".
* It is used in Functional,unit ,integration and end to end testing
* **TestNG provides you full control over the execution of the test cases. Due to this reason, TestNG is also known as a testing framework.**

## Hierarchy of the TestNG Annotations:



* @BeforeSuite
* @BeforeTest
* @BeforeClass
* @BeforeMethod
* @Test
* @AfterMethod
* @AfterClass
* @AfterTest
* @AfterSuite

Sample Testcase with @Test annotation

1. **public** **class** test
2. {
3. @Test
4. **public** **void** test1() // First test case.
5. {
6. System.out.println("test1");
7. }
8. @Test
9. **public** **void** test2() // Second test case.
10. {
11. System.out.println("test2");
12. }}

**Priority:**

* Priority is an attribute that helps the **users define the order** in which they want the test cases to be executed.
* @Test(priority =0)

PublicclassPrioritylearn {  
  
  
@Test(priority=0)  
public void startthecar()

{  
System.out.println("Start the car");  
}  
  
@Test(priority - 1)  
public void firstgear() {  
System.out.println("firstgear");  
}

**Dependencies:**

public class Dependecies {

@Test(enabled = true)

public void highschool()

{

System.out.println("High School");

}

@Test(dependsOnMethods ="highschool")

public void highersecondary()

{

System.out.println("Higher secondary");

}

@Test (dependsOnMethods = "highersecondary")

public void engineering()

{

System.out.println("Engineering");

}

**Dependency in TestNG:**

* It allows a test method (@Test) to depends on another methods.
* Dependency allows us to make one test method dependent on one or multiple other test methods.
* We define by giving @Test(dependsOnMethods ="highschool")

**How to write Test suites ?**

@BeforeSuite

public void launchbrowser ()

{

starttime = System.currentTimeMillis();

System.setProperty("webdriver.chrome.driver", "C:\\Users\\NAZEER\\Desktop\\chromedriver.exe");

driver = new ChromeDriver();

}

@Test

public void openchrome()

{

driver.get("http://www.google.co.in");

}

@Test

public void bing()

{

driver.get("http://www.bing.com");

}

@Test

public void openyahoo()

{

driver.get("http://www.yahoo.com");

}

@AfterSuite

public void closebrowser()

{

driver.quit();

endtime = System.currentTimeMillis();

long totaltime=endtime -starttime;

System.out.println("Totaltime" + totaltime);

}

**How to skip or exclude test cases::**

* **We can skip test cases by using** @Test(enabled =false )

public class Skiptestcase {

@Test(priority=0)

public void startthecar()

{

System.out.println("Car started");

}

@Test(priority=4,enabled =false )

public void turnonthemusic()

{

System.out.println("start music");

}

@Test(priority=1)

public void firstgear()

{

System.out.println("1st gear");

}

@Test(priority=2)

public void secondgear()

{

System.out.println("2nd gear");

}

@Test(priority=3)

public void thirdgear()

{

System.out.println("thirdgear");

}

**Why we need a test suite in XML?**

* Imagine if we have 200 test cases ,we cannot run each class file (which is test cases) for 200 classes
* To make it simpler,we are creating a suite and we are adding test cases in test suite

**How to create XML file?**

New > XML or other then XML> XML File

Name the file name as **TestNG.xml**

1.<suite>

</suite>

Then inside suite add test

<test>

</test>

Then inside test add classes

Add individual class inside the classes

<class>

</class>

**It will be like**

**<suite>**

**<test>**

**<classes>**

**<class>**

**</class>**

**</classes>**

**</test>**

**</suite>**

**How to add class name in xml?**

Packagename.classname like below

<classes>

<class name="testng.Groupsexample"></class>

</classes>

Note – If u select CTRL + on classname it will open as link

**<suite name = “ Test Suite”>**

**<test name = “Test case”>**

**<classes>**

<class name="testng.Groupsexample">

</class>

**</classes>**

**</test>**

**</suite>**

**Practical code:**

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

**<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd" >**

**<suite name ="Testsuite" verbose ="2" parallel="methods" thread-count="2" >**

**<listeners>**

**<listener class-name="testng.Listernersexample"></listener>**

**</listeners>**

**<test name = "Test cases">**

**<parameter name="Name" value="Nazeer"></parameter>**

**<groups>**

**<run>**

**<include name="Moto"></include>**

**<include name="Apple"></include>**

**</run>**

**</groups>**

**<classes>**

**<class name="testng.Groupsexample"></class>**

**</classes>**

**</test>**

**</suite>**

**How to view test report?**

A screenshot of a computer

Description automatically generated

Right on Index.html under testoutput > open with >web browser

Note:testputput will only generate after we run the script

If u add verbose = 2 it will have more details

<suite name ="Testsuite" verbose ="2" parallel="methods" thread-count="2" >

**Assertion:**

public class Assertexample {

String name = "Nazeer";

boolean value = false;

**@Test**

public void checkequal()

{

//System.out.println("Nazeer");

Assert.assertEquals(name, "Nazeer");

}

**@Test**

public void checknotequal()

{

//System.out.println("Nazeer");

Assert.assertNotEquals(name, "Nazeers");

}

**@Test**

public void Asserttrue()

{

//System.out.println("Nazeer");

Assert.assertTrue(value, "This is true");

}

**@Test**

public void Assertfalse()

{

//System.out.println("Nazeer");

Assert.assertFalse(value,"This is true");

}

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

Cucumber

Cucumber with Java – Build Automation Framework with Minimal code

**What is Behavior-driven development (BDD)?**

* Behavior-driven development (BDD) is an Agile software development methodology in which an application is designed and documented around **the behavior of the**

**Advantages of Cucumber Software**

* It is helpful to involve business stakeholders who can’t easily read code
* Cucumber Testing tool focuses on end-user experience
* Style of writing tests allow for easier reuse of code in the tests
* Quick and easy set up and execution
* Cucumber test tool is an efficient tool for testing

What is Framework?

Whenever Automation scripts are developed, We need to create an Execution system called framework,Inorder to to run and maintain Automated tests

What is Cucumber?

* Cucumber is a tool which implements BDD framework
* Since It implements BDD ,Cucumber enables you to write test cases that anyone can easily understand

What Cucumber does?

Data driven, Parameterization, Execution controls, Hooks, Reports, Automation

When you say automated tests, what type of Automation testcases does cucumber Support?

Any Test (Web, Mobile, API, Unit Testing) which is written in Java/Ruby supported by Cucumber

Cucumber Terminologies:

What is Gherkin?   
 It is a Business Readable, Domain Specific Language that describe **software's behavior using given ,when and then keywords**

Keywords Used in Cucumber: Scenario, Feature, Feature file, Scenario outline, Step Definition

Scenarios:

* In Cucumber Testcases are represented as Scenarios.

Scenarios contain Steps which are equivalent to test Steps).

* **Given**: Preconditions are mentioned in the Given keyword
* **When**: describe the user action.
* **Then**: observe the expected output.
* **And:** addition to the previous Steps ,represents positive statements
* **But:** addition to previous Steps and represent negative statements.

Feature and Feature File:

* Feature represents Business requirement.
* Feature File acts as a Test Suite which consists of all Scenarios.

**Parametrization: Sending Data from Feature File to Step Definition**

**What is Parameterizing?**

* Parameterization is the process of providing parameters to your Cucumber scenarios, allowing you to run the same scenario with different sets of data
* In Cucumber ,We can achieve parameterization using Scenario Outline and Examples keyword

**Different types of Data Driven Testing (Sending data via Feature File):**

1.Sending values directly and receiving it via regular expression

2.Data tables without header

3.Data tables with header

4.Using Scenario Outline and Example keyword

**Cucumber Options:**

1. **Features :** Feature will have feature location
2. **Glue :** Glue will have Step definition file location
3. **Dry run**
4. **Strict**
5. **Tags**
6. **Monochrome**
7. **format**

**3)dryRun**

* **DryRun True** will check the script is correct or not without executing the script
* It saves time when there is huge script, we need to check script correct or not,then we will use DryRun ,If anything wrong in script we will correct and re-run the script
* DryRun false will run execute the script normally

@RunWith(Cucumber.**class**)

@CucumberOptions(features= "src/test/java/featurefiles/PMTlogin.feature" , glue="stepdefinition",stepNotifications = **true**,

dryRun = **true**)

**4)strict**

* Strict will check the script is correct or not by executing the script
* Strict=true will execute the script and check,if failed it will throw failure,test case will fail
* Strict = false will skip the execution of script,if failed it will skip the failure,test case will pass

**Difference between dryRun and strict?**

* **DryRun True** will check the script are correct or not without executing the script
* Strict will check the script correct or not by executing the script

**Monochrome:**

* To convert the console message into human readable messages
* Monochrome=true will convert the non-readable messages in console into readable messages
* Monochrome=false will show the non-readable messages in console without conversion

**Format**

* Format is deprecated ,use plugin instead of format
* **Format is used for report generation**
* **Report can be generated using html,json and xml**

**Tags:**

In Cucumber, tags are used to group the testcases like smoke , regression

Simply tag =grouping

**Tag purposes:**

* By default, Cucumber executes all the scenarios inside the feature file, but inorder to add or skip any specific scenario ,we can declare scenarios within a tag.
* If we have many scenarios in the feature file , to keep them in **one group**, we use **tag**s in Cucumber, so that we can generate report on the specific group tag

**Syntax**

@TestName

Scenario: Mention the Scenario

tags={"Smoke test"}

**Example:**

Suppose, a feature file of an application contains 100 test scenarios, and when we test this application through Cucumber testing each time 100 test scenarios will get executed unnecessarily. And due to that, system performance is getting low.

To overcome this problem, we can use a tag

tags={"Smoke test"}

tags ={""Regression test}

To overcome this problem, we can use a tag

**How to ignore tags in Cucumber testing?**

* when we need to skip a test, then we can use Special symbol "~" within the tag

**Syntax:**

tags={"@SmokeTest", "~@RegressionTest"}

tags={"~smoke test"}

Note : Tag can be given on Feature file level or Scenario or Scenario outline level

* If we need to run all the scenarios of feature file ,we can use feature file tage
* If we need to run specific scanrios in a feature file,then scenario tag we can run

**How to club Add and OR tag?**

**OR operator**

When we need to test an application , if the application has failed in the first test, then the next test should be checked.

Example:

tags= {"@SmokeTest, @RegressionTest"}

In the above example, OR operator executes all the tagged tests i.e., @SmokeTest, @RegressionTest.

Tags={“@smoketest @Regressiontest””}

**AND Operator**

when we need to test an application like this, if the application has passed in the first test, then jump to check the next test, but if it gets failed, then testing should be terminated.

Example:

tags= {"@SmokeTest", "@RegressionTest"}

In the above example, AND operator executes the first test, if the result of this test is passed then it will jump to check the next test. But if the result is failed then testing will be terminated.

**tags={“@smoketest”, “Regression test”}**

**Cucumber Hooks:**

**Why cucumber hooks?**

* Used to set things up before executing the scenario
* And also after executing the scenario
* It is similar to TestNG annotations but only 2 annotations

Simply hooks = setting up before and After scenario

We have 2 annotations:

**@Before and @After**

**Example:**

@Before : Launch the browser

@After : Close the browser

**Order of before and After cucumber hooks:**

**Similar to TestNG priority,In cucumber we have Order for the Before and After execution**

**public** **class** Thanoshook {

@Before(order=1)

**public** **void** beforesnapping() {

System.***out***.println("Thanos collecting the infinity stones");

}

@After(order=1)

**public** **void** aftersnapping() {

System.***out***.println("After killing everyone,taking rest");

}

@Before(order=0)

**public** **void** beforebeforesnapping() {

System.***out***.println("Thanos collecting the infinity stones");

}

@After(order=0)

**public** **void** afteraftersnapping() {

System.***out***.println("After killing everyone,taking rest");

}

**Before annotation order follows ascending order (0,1,2)**

**After annotation order follows descending order(2,1,0)**

**Tagged Hooks:**

* If prerequisites are the **same** for all the scenarios ,we can use **Hooks**
* If prerequisites **are different** some scenarios? We can use **Tagged Hooks**
* Combination of tagging and Hooks is called Tagged Hooks
* It can be Feature level tags and Scenario level tags

**Syntax for tagged hooks:**

Hook(“Tag”)

@Before("@Firsthook")

Hook.java file

@Before

**public** **void** beforescenario() {

System.***out***.println("Run Before EVERY scenario");

}

@After

**public** **void** afterscenario() {

System.***out***.println("Run after first Every scenario");

}

@Before("@Firsthook")

**public** **void** beforefirst() {

System.***out***.println("Run Before first scenario");

}

@After("@Firsthook")

**public** **void** afterfirst() {

System.***out***.println("Run After first scenario");

}

Feature file

*@TaggedHooks*

**Feature:** Test Tagged Hooks

*@Firsthook*

**Scenario:** First Scenario

*Given* First step

*When* second step

*Then* third step

*@Secondhook*

**Scenario:** Second Scenario

*Given* First step

*When* second step

*Then* third step

*@thirdhook*

**Scenario:** Third Scenario

*Given* First step

*When* second step

*Then* third step

**public** **class** Taggedhookstepdef {

@Given("First step")

**public** **void** first\_step() {

System.***out***.println("First step");

}

@When("second step")

**public** **void** second\_step() {

System.***out***.println("Second step");

}

@Then("third step")

**public** **void** third\_step() {

System.***out***.println("Third step");

}

}

**Background in Cucumber:**

* It represents list of **pre-conditions for all the scenarios**
* It can contain one or more Given steps (Given represents pre-condition)
* It allows you to add some context(info) to the scenarios in the feature file

Example – Feature file can contain hooks like @Before test ,@Before firsthook but when non-technical person sees feature file he see only Scenarios and Given,When,then, So he will miss “Hook”.inorder to resolve this problem we use “background” to add some context or information about hooks .

Difference between Hook and Background

* Hooks have Precondition (@Before) and Postcondition (@after)
* Background will only have pre-conditions
* Multiple Precondition (@Before) and Postcondition (@after) possible for a scenario in HOOK
* Only one single of background steps in Feature

|  |  |  |
| --- | --- | --- |
| **Slno** | **Hooks** | **Background** |
| 1 | Hooks have Precondition (@Before) and Postcondition (@after) | Background will only have pre-conditions |
| 2 | Multiple Precondition (@Before) and Postcondition (@after) possible for a scenario in Feature file | Only one single background steps per Feature is allowed |
| 3 | We cannot add some context for scenarios feature file | We can add some context for scenarios in feature file |

How to implement multiple background step in Background?

* We need to divide the feature files into different category

What if I have both hooks and Background?

* A background will run after each scenario but only after before hooks

**How to write Xpath?**

**Why do we need Xpath?**

* Locators like Id,Name may be written or may NOT written by developer
* Similarly for linktext and partial linktext used only for links
* Class name and CSS locator may not be available for the element
* So a best solution we have Xpath

**What is Staitc xpath?**

* This Xpath value will NOT be changed,So we will use browser plugin to copy the xpath

**When we need to write Xpath?**

* When the content are dynamic.Which mean the xpath values are NOT unique

**What is Xpath?**

* Xpath is a XML Path
* Xpath is one of the strongest and flexible location strategy,Can be used to locate any web element

**How Xpath handles HTML WebElement?**

* Even though Xpath is a XML element,Both XML and HTML parent is SGML(standarised markup language),so we can use Xpath in HTML element

**What is Dynamic Xpath?**

* Values of Dynamix Xpath is NOT unique

**Types of Xpath:**

1.Absolute Xpath :

* Start with Root node,start with single slash
* Performance will be slow
* Path can be changed

2.Relative Xpath :

* Start with any node,start with double slash
* Performance will be Faster
* Path wont be changed most of the time

**Identification strategies:**

a) Locating Elements with Unknown Element and Known Attribute

b) Locating Elements with known Element and Attributes

c) Locating Elements with Known Visible Text (exact match)

d) Locating Elements when part of the visible text (partial match)

e) Locating Elements with Multiple Attributes

f) Locating elements when starting visible text is known

g) Locating Elements with Dynamic Attribute values

Graphical user interface, website

Description automatically generated

Sample site: https://opensource-demo.orangehrmlive.com/

**Relative Xpath syntax:**

**//Element[@attribute=’value’]**

**If we don’t the element put \***

**//\*[@attribute=’value’]**

**a) Locating Elements with Unknown Element and Known Attribute**

* **Here we know only attribute,but don’t know element ,so we put element as \***

**Syntax:**

//\*[@attribute=’value’]

**By ID**

//\*[@id='txtUsername']

**Or By Name**

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

**b) Locating Elements with known Element and Attributes**

* **Here we know both element and attribute**

**syntax:**

//input[@attribute=’value’]

Input is the element

**By ID**

//Input[@id='txtUsername']

**Or By Name**

//Input[@name='txtUsername']

**c) Locating Elements with Known Visible Text (exact match)**

* Used for locating elements containing exact text within an element
* Normally we will use it for LINKS

**Syntax:**

//elementName[text()=’exact text’]

Orange HRM link :

//\*[text()='OrangeHRM, Inc']

**d) Locating Elements when part of the visible text (partial match)**

* Used for locating elements with partial text
* Normally we will use it for LINKS

**Syntax:**

//elementName[contains (text(),’part of the text)’]

//a[contains(text(),'OrangeHRM')]

**e) Locating Elements with Multiple Attributes**

* To locate an element with multiple attributes

**Syntax:**

//\*[@attribute1=’value1’] [@attribute2=’value2’] [@attribute3=’value3’]

**Example for Login button in Orange HRm:**

//\*[@type='submit'][@name='Submit'][@class='button']

**f) Locating elements when starting visible text is known**

* **To locate element with starting text**

**Syntax:**

//elementName[starts-with (text(),’starting text)’]

Example:

//\*[starts-with(text(),'OrangeHRM')]

**g) Locating Elements with Dynamic Attribute values**

* When attribute values changes dynamically, we can use contains or starting text or ending text

**Syntax:**

//elementName[contains (@attributeName,’part of the text)’]

//elementName[starts-with ((@attributeName,’starting text)’]

**Example**

//\*[contains(@href,’Orange')]

**h)Normalise-space method**: If an element has spaces in its text or in the value of any attribute,then we can use normalize-space() method

**Syntax:**

**//tagname[normalize-space(@attribute/ function) = 'value']**

**Example:**

//strong[normalizespace( text())='JAVA BASICS']

Can be used in below 2 ways:

//tagName[normalise-space()='value']

//tagName[normalise-space(text='value')]

**Locating Elements relative to known element:**

1.Locating a parent element

2.Locating a child element

3.Locating ancestors of a known element

4.Locating the following elements

5.Locating preceding element

6.Locating the following sibling

7.Locating preceding sibling

**1.Locating a parent element**

* It is used to find the parent of current xpath

**Syntax:** //<knowXpath>/parent::elementName

**Example** : ////\*[@id='divUsername']/parent::form

Here form we the parent so form,if not we can use \*

**2.Locating a child element**

* It is used to find the child of current xpath

**Syntax:** //<knowXpath>/child::elementName

**Example:** //\*[@id='divUsername']/child::\*

Above example have 2 child,inorder to identify unique element we can use unique keywork of particular element here it is input

Xpath should be unique

**//\*[@id='divUsername']/child::input**

**3.Locating ancestors of a known element**

* It is used to find the forefather of current xpath

**Syntax:** //<knowXpath>/ancestor::elementName

**Example:** //\*[@id='divUsername']/ancestor::\*

**4.Locating the following elements**

* It is used for following nodes after the current xpath
* If more than 1 following node are there, mention the node by its index position

Syntax: //<knowXpath>/following::input

Example:

//\*[@id='divUsername']/following::input

To mention by position

//\*[@id='divUsername']/following::input[2]

**5.Locating preceding element**

* It is used for preceding before the current xpath(i.e Nodes before the current xpath)

Syntax: //<knowXpath>/precending::input

Example:

//\*[@id='divUsername']/preceding::input

**6.Locating the following sibling**

* It is used for selecting the siblings by following of the current xpath

Syntax: //<knowXpath>/precending::input

Example:

//\*[@id='divPassword']/following-sibling::\*

**7.Locating preceding sibling**

* It is used for selecting the siblings by preceding of the current xpath

**Syntax:** //<knowXpath>/precending-sibiling::input

Example:

//\*[@id='divPassword']/ precending-sibiling::\*

**Always use relative xpath :**

With known attributes

**For Dynamic element :**

* Use contains,Start with or end for customized xpath

**For Links:**

* Use text methods

**Introduction to Testing?**

**What is Software testing?**

**Software testing i**s the process of verifying and validating a software application to check whether it is working as expected.

**What is Verification?**

* Verification is the process of checking whether the software satisfies the **business specifications**
* Verification is **static** testing.
* Verification means **Are we building the product right?**
* In simpler terms, verification means the **business team perception of the final product.**

**What is Validation?**

* Validation is the process of checking whether the software product **satisfies User expectations** of the product
* Validation is **dynamic** testing.
* Validation means **Are we building the right product?**
* In Simpler terms, validation means the User **perception** of the product

**Real-life Example:** Imagine yourself going to a restaurant blueberry pancakes. Waiter brings the order food

You will verify by looking:

Does the food look like what pancakes typically appear to be?

Are the blueberries to be seen?

Do they smell right?

On the other hand, when you need to be absolutely sure about whether the food is as you expected: You will have to eat it.

**Verification** is all when you are yet to eat but are checking on a few things by reviewing the subjects.

**Validation** is when you actually eat the product to see if it is right.

**Difference between Verification and Validation:**

|  |  |  |
| --- | --- | --- |
| **Slno** | **Verification** | **Validation** |
| 1 | Verification is the process of checking whether the software meets the business specifications without any bugs. | Validation is the process of checking whether the software product meets customer expectations of the product |
| 2 | Verification means Are we building the product right? | Validation means Are we building the right product? |
| 3 | Verification is static testing | Validation is dynamic testing. |
| 4 | This is done without executing the software | This is done with executing the software. |
| 5 | Evaluates the intermediary products to check whether it meets the specific requirements of the particular phase. | Evaluates the final product to check whether it meets the business needs. |
| 6 | Examples include Requirement reviews, inspection (Document preparation,) and Code-walkthrough. | Example includes all types of testing like smoke, regression, functional, systems and UAT. |

**7 Principles of Testing:-**

(i) All the test should meet the customer requirements

(ii) To make our software testing should be performed by a third party

(iii) Exhaustive testing is not possible. As we need the optimal amount of testing based on the risk assessment of the application.

(iv) All the test to be conducted should be planned before implementing it

(v) It follows the Pareto rule(80/20 rule) which states that 80% of errors come from 20% of program components.

(vi) Start testing with small parts and extend it to large parts.

**Types of testing:**

### #1) Alpha Testing

Alpha Testing will be conducted at the developer’s site. An in-house virtual user environment can be created for this type of testing.

**#2) Acceptance Testing**

An Acceptance Test is performed by the client and it verifies whether the end to end flow of the system is as per the business requirements or not and if it is as per the needs of the end-user.

Client accepts the software only when all the features and functionalities work as expected. This is the last phase of testing, after which the software goes into production. This is also called User Acceptance Testing (UAT).

**#3) Ad-hoc Testing**

* The objective of this testing is to find the defects and break the application by executing the application or randomly
* Ad-hoc Testing is an informal way of finding defects and can be performed by anyone in the project.
* It is difficult to identify defects without a test case but sometimes it is possible that defects found during ad-hoc testing might not have been identified using the existing test cases.

**#4) Accessibility Testing**

* The aim of Accessibility Testing is to determine whether the software or application is accessible for disabled people or not.
* Here, disability means deafness, color blindness, mentally disabled, blind, old age and other disabled groups. Various checks are performed such as font size for visually disabled, color and contrast for color blindness, etc.

**#5) Beta Testing**

* Beta Testing is a formal type of Software Testing which is carried out by the customer. It is performed in the Real Environment before releasing the product to the market for the actual end-users.

**#6) Back-end Testing**

* Whenever an input or data is entered on the front-end application, it is stored in the database and the testing of such database is known as Database Testing or Backend Testing.
* There are different databases like SQL Server, MySQL, and Oracle, etc. Database Testing involves testing of table structure, schema, stored procedure, data structure and so on.

**#7) Browser Compatibility Testing**

* Browser Compatibility Testing is performed for web applications and ensures that the software can run with a combination of different browsers and operating systems. This type of testing also validates whether a web application runs on all versions of all browsers or not.

**#8) Backward Compatibility Testing**

It is a type of testing which validates whether the newly developed software or updated software works well with the older version of the environment or not.

**#9) Black Box Testing**

* Internal system design is not considered in this type of testing. Tests are based on the requirements and functionality.

**#10) Black Box Testing**

Boundary Value Testing is performed to check if defects exist at boundary values. Boundary Value Testing is used for testing a different range of numbers. There is an upper and lower boundary for each range and testing is performed on these boundary values.

**#11) Compatibility Testing**

This is a testing type in which it validates how software behaves and runs in a different environment, web servers, hardware, and network environment.

Compatibility testing ensures that software can run on a different configuration, different databases, different browsers, and their versions. Compatibility testing is performed by the testing team.

**#12) Component Testing**

* This is mostly performed by developers after the completion of unit testing.
* Component Testing involves testing of multiple functionalities as a single code and its objective is to identify if any defect exists after connecting those multiple functionalities with each other.

**#15) End-to-End Testing**

Similar to system testing, End-to-End Testing involves testing of a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems if appropriate.

**#16) Equivalence Partitioning**

It is a testing technique and a type of Black Box Testing. During this Equivalence Partitioning, a set of groups are selected and a few values or numbers are picked up for testing. It is understood that all values from that group generate the same output.

The aim of this testing is to remove redundant test cases within a specific group which generate the same output but not any defect.

Suppose the application accepts values between -10 and +10, then using equivalence partitioning the values picked for testing are zero, one positive value, and one negative value. So the Equivalence Partitioning for this testing is -10 to -1, 0, and 1 to 10.

**#18) Exploratory Testing**

Exploratory Testing is informal testing performed by the testing team. The objective of this testing is to explore the application and look for defects that exist in the application.

**#20) Functional Testing**

This type of testing ignores the internal parts and focuses only on the output to check if it is as per the requirement or not.

This is a black-box type testing that is geared towards the functional requirements of an application. For detailed information about Functional Testing, you can check it here.

**#21) Graphical User Interface (GUI) Testing**

The objective of this GUI Testing is to validate the GUI as per the business requirement. The expected GUI of the application is mentioned in the Detailed Design Document and GUI mockup screens.

GUI Testing includes the size of the buttons and input fields present on the screen, alignment of all text, tables, and content in the tables.

It also validates the menu of the application, after selecting different menu and menu items, it validates that the page does not fluctuate and the alignment remains the same after hovering the mouse on the menu or sub-menu.

**#34) Regression Testing**

Testing an application as a whole for the modification of any module or functionality is termed as Regression Testing.

It is difficult to cover all the systems in Regression Testing, so typically Automation Testing Tools are used for these types of testing.

**#38) Smoke Testing**

Whenever a new build is provided by the development team, then the Software Testing team validates the build and ensures that no major issue exists.

The testing team will ensure that the build is stable and a detailed level of testing will be carried out further. Smoke Testing checks for no show stopper defects exist in the build which will prevent the testing team from testing the application in detail.

If the testers find that the major critical functionality is broken down at the initial stage itself then the testing team can reject the build and inform accordingly to the development team. Smoke Testing is carried out to a detailed level of any Functional or Regression Testing.

**Test your testing skill**

### Exercise #1: Finding Defects

As a software tester, what do you do? Of course, testing the software, you would say………Okay, can you find out defects on the page shown below?



**Test Plan:**

* A Test Plan is a detailed document that describes

1. In Scope
2. Out of scope
3. Approach
4. Test strategy
5. Objectives
6. Schedule
7. Estimation
8. Deliverables
9. Entry criteria
10. Exit criteria
11. Resources (required to perform software testing )

* Test plan serves as a blueprint to conduct software testing activities which is minutely monitored and controlled by the team lead or test manager.
* Test Plan will be prepared by Team Lead or Test Manager
* Test Plan document is Dynamic
* Test Plan prepared at Project level

Example: The Test Plan gives information about who is going to test at what time.

For Example, Module 1 is going to be tested by “X tester”. If tester Y replaces X for some reason, the test plan has to be updated.

**Test Strategy:**

* Test Strategy is a set of guidelines that explain the **test design** and determine **how testing needs to be done.**
* Test Strategy document is Static
* Test Strategy document prepared by Project Manager at Organizational level
* Purpose of the test strategy is to define the testing approach, testing types, test environments, and tools to be used for testing
* Test Strategy document at organizational level

Example: A Test Strategy includes details like “Individual modules are to be tested by the test team members”. In this case, who tests it does not matter – so it’s generic and the change in the team member does not have to be updated, keeping it static.

### Test Plan Vs Test Strategy

| **TEST PLAN** | **TEST STRATEGY** |
| --- | --- |
| It is derived from software requirement specification(SRS). | It is derived from the Business Requirement document(BRS). |
| It is prepared by the test lead or manager. | It is developed by the project manager or the Business analyst. |
| Test plan id, features to be tested, test techniques, testing tasks, features pass or fail criteria, test deliverables, responsibilities, and schedule, etc. are the components of the test plan. | Objectives and scope, documentation formats, test processes,  team reporting structure, client communication strategy, etc. are the components of test strategy. |
| If there is a new feature or a change in the requirement that is happened then the test plan document gets updated. | Test strategy maintains the standards while preparing the document.  It is also called as Static document. |
| We can prepare the test plan individually. | In smaller projects, test strategy is often found as a section of a test plan. |
| We can prepare a Test plan at the project level. | We can use Test strategy at multiple projects. |
| It describes how to test , when to test, who will test and what to test. | It describes what type of technique to follow and which module to test. |
| We can describe about the specifications by using a Test Plan. | Test strategy describes about the general approaches. |
| Test Plan will change over the course of the project. | Test Strategy usually will not change once approved. |
| Test plan is written after requirement sign off. | Test strategy is made before the test plan. |
| Test plans can be of different types. There will be a master test plan and separate test plan for different types of testing like system test plan, performance test plan, etc. | There will be only one test strategy document for a project. |
| Test plan should be clear and concise. | Test strategy provides overall guidance for the project in hand. |

**Software testing i**s the process of verifying and validating a software application to check whether it is working as expected.

**Test Scenarios :** Highest level classification of test cases

**Test case:** A test case is a document that has a set o**f conditions or actions** that are performed on the software application in order to verify the expected functionality of the feature.

**Positive test cases** ensure that users can perform appropriate actions when using valid data.

**Negative test cases** are performed to try to “break” the software by performing invalid (or unacceptable) actions, or by using invalid data.

**Manual Testing** – This is the oldest type of software testing where the testers manually execute test cases without using any test automation tools.

**Automation Testing** – This is the process of using the assistance of tools, scripts, and software to perform test cases by repeating pre-defined actions.

**What is quality control? Is it similar to Quality Assurance?** Quality control is a product-oriented approach of running a program to determine if it has any defects, as well as making sure that the software meets all of the requirements put forth by the stakeholders.

**Different types of manual testing are?**

* Black Box Testing
* White Box Testing
* Unit Testing
* System Testing
* Integration Testing
* Acceptance Testing

**Alpha Testing** – It is a type of software testing performed to identify bugs before releasing the product to real users or to the public. Alpha Testing is a type of user acceptance testing.

**Beta Testing** – It is performed by real users of the software application in a real environment. Beta Testing is also a type of user acceptance testing.

**Unit testing –** It is a way of testing the smallest piece of code referred to as a unit that can be logically isolated in a system. It is mainly focused on the functional correctness of the standalone module.

**Integration Testing –** It is a level of software testing where individual units are combined and tested to verify if they are working as they intend to when integrated. The main aim here is to test the interface between the modules.

**System Testing** – In system testing all the components of the software are tested as a whole in order to ensure that the overall product meets the requirements specified. There are dozens of types of system testing, including usability testing, regression testing, and functional testing.

**API testing** is a type of software testing where application programming interfaces (APIs) are tested to determine if they meet expectations for functionality, reliability, performance, and security.

### Q20. What are the phases involved in Software Testing Life Cycle?

The different phases involved in the [software testing life cycle](https://www.edureka.co/blog/software-testing-life-cycle/#stlc) are:

|  |  |
| --- | --- |
| **Requirement Analysis** | QA team understands the requirement in terms of what we will testing & figure out the testable requirements. |
| **Test Planning** | In this phase, the test strategy is defined. Objective & the scope of the project is determined. |
| **Test Case Development** | Here, detailed test cases are defined and developed. The testing team also prepares the test data for testing. |
| **Test Environment Setup** | It is a setup of software and hardware for the testing teams to execute test cases. |
| **Test Execution** | It is the process of executing the code and comparing the expected and actual results. |
| **Test Cycle Closure** | It involves calling out the testing team member meeting & evaluating cycle completion criteria based on test coverage, quality, cost, time, critical business objectives, and software. |

**Test Estimation:**

* Test Estimation is a management activity which approximates how long a Task would take to complete

**List of Software Test Estimation Techniques**

* Work Breakdown Structure
* 3-Point Estimation Technique
* Wideband Delphi technique
* Function Point/Testing Point Analysis
* Use – Case Point Method
* Percentage distribution
* Ad-hoc method

**Work Breakdown Structure**

**Step1) Divide the whole project task into subtasks**

In this technique, a complex project is divided into modules. The modules are divided into sub-modules. Each sub-module is further divided into functionality. It means divide the whole project task into the smallest tasks.

**Step 2) Allocate each task to team member**

**Step 3) Effort Estimation For Tasks**

There are 2 techniques which you can apply to estimate the effort for tasks

* Functional Point Method
* Three Point Estimation

**Step 4) Validate the estimation**

Once you create an aggregate estimate for all the tasks mentioned in the WBS, you need to forward it to the management board, who will review and approve it.

The management board will review and discuss your estimation plan with you. You may explain them your estimation logically and reasonably so that they can approve your estimation plan

**Function Point Method**

* In this method, the Test Manager estimates Size, Duration, and Cost for the tasks

**Step A) Estimate size for the task**

Based on the complex of software functions, the Test Manger has to give enough weightage to each functional point. For example

Group Weightage

Complex 5

Medium 3

Simple 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Module Name** | **Applicable Roles** | **Description** | **Weightage** |
| 1. | Balance Enquiry | Manager  Customer | **Customer:**A customer can have multiple bank accounts. He can view balance of his accounts only  **Manager:**A manager can view balance of all the customers who come under his supervision | 3 |
| 2. | Fund Transfer | Manager  Customer | **Customer:**A customer can have transfer funds from his “own” account to any destination account.  **Manager:**A manager can transfer funds from any source bank account to destination account | 5 |

**STEP B) Estimate duration for the task**

Graphical user interface, text, application, Word

Description automatically generated

|  |  |  |
| --- | --- | --- |
| **Weightage** | **# of Function Points** | **Total** |
| Complex | 5 | 3 | 15 |
| Medium | 3 | 5 | 15 |
| Simple | 1 | 4 | 4 |
| **Function Total Points** | | | **34** |
| **Estimate define per point** | | | **5** |
| **Total Estimated Effort (Person Hours)** | | | **170** |

**STEP C) Estimate the cost for the tasks**

Suppose, on average your team salary is $5 per hour. The time required for “Create Test Specs” task is 170 hours. Accordingly, the cost for the task is 5\*170= $850

**METHOD 2) Three Point Estimation**

A picture containing diagram

Description automatically generated

**Maven**

**What is Maven?**

* Apache Maven is a powerful project management, Based on the concept of a project object model (POM)
* Maven used for **Maintaining proper structure**
* it can **manage dependencies** on our behalf
* Maven Can **build code** for us
* Maven can **manage documentation** from a central piece of information

**Project Object Model:**

* All POM files require the project element and three mandatory fields: **groupId, artifactId, version.**

**Maven Repository:**

* Maven  **repository is a directory where all the project jars**, library jar, plugins are stored and can be used by Maven easily.
* Example – If it is dependency jar,we can retrieve dependeny using dependency tag in MVN repository

**Maven Dependencies:**

* It will be a tough task to install the dependent jars, dependent transient jars manually, sometime issue may occur, and it will take time to do that
* By using maven dependencies, we can easily do the dependencies like jars, library can be added

**How do we add dependencies?**

* Simply by copying required dependency from MVN central repository, then by adding dependency in pom xml

**Maven Build Process:**

* It takes time and may cause error issues in manually building the code by extracting the jars so, By using maven build we can easily do build process with less time
* When creating build in Maven, make sure manifest, add the plugins tag with main class, add class,classpath lib mentioned in pom xml

**Maven Documentation:**

* Add maven site plugin in POM xml ,set goals in Configuration as “site” and Run ,The document will be created in target folder

**Maven – Build life cycle:**

**1.Default - Handles your project deployment**

**1.Validate –** validation will check project correctness

**2.compile -** compile the source code of the project

**3.test** - test the compiled source code using a suitable unit testing framework.

**4.package** - take the compiled code and package it in its distributable format, such as a JAR.

**5.verify** - run any checks on results against the requirement

**6.install** - install the package into the local repository, for use as a dependency in other projects locally

7. **deploy** - done in the build environment, copies the final package to the remote repository for sharing with other developers and projects.

**2.Clean -**clean goal deletes the output of a build by deleting the build directory

**3.Site -** The Site Plugin is used to generate a site for the project.

**1. List the differences between ANT and Maven.**

|  |  |
| --- | --- |
| **Ant** | **Maven** |
| Ant doesn’t have formal conventions, so we need to provide information about the project structure in the build.xml file. | Maven has a convention to place source code, compiled code, etc. So we don’t need to provide information about the project structure in pom.xml file. |
| Ant is procedural, you need to provide information about what to do and when to do through code. You need to provide order. | Maven is declarative, everything you define in the pom.xml file. |
| There is no life cycle in Ant. | There is a life cycle in Maven. |
| Ant is a toolbox. | Maven is a framework. |
| It is mainly a build tool. | It is mainly a project management tool. |
| The ant scripts are not reusable. | The maven plugins are reusable. |
| It is less preferred than Maven. | It is more preferred than Ant. |

**NOTE**: This is a common Maven Interview Question that you must know.

**2. What is Maven?**

*[Maven](https://maven.apache.org/" \t "_blank)*is a project management and comprehension tool. Maven provides developers a complete build lifecycle framework. The development team is easily able to automate the project’s build infrastructure in almost no time as Maven uses a standard directory layout and a default build lifecycle.

## **3. What are the main features of Maven?**

Some of the main features of Maven are:

* + **Simple to use**: Maven provides easy project settings that are based on genuine practices.
  + **Fast**: You can receive a fresh project or module that began in fewer seconds in Maven.
  + **Easy to learn:** Maven usage and commands are pretty easy to learn across all projects. Therefore ramp-up time for new developers coming onto a project is very less.
  + **Dependency management:** Maven provides superior dependency management including automatic updates and transitive dependencies.
  + **Multiple Projects**: You can easily work on multiple projects at the same time by using Maven.
  + **Huge Library:** Maven has a large and growing repository of libraries and metadata to use out of the box.

## **4. What does the build tool?**

* + Generates source code (if the auto-generated code is used)
  + Generates documentation from source code
  + Compiles source code
  + Packages compiled code into a JAR or ZIP file
  + Installs the packaged code in the local repository, server repository, or central repository

## **5. Why should one use Maven?**

* It aids in setting up the project very quickly and it avoids complicated build files like build.xml. the pom.xml file is at the core of Maven.POM.xml is a collection of dependencies of your Java Project which one can specify to Maven. After this Maven will download all of them from the internet and store it to some repository i.e. local repository, central repository, and remote repository.
* It helps to bundle all the jars in your package i.e. in your War file or Ear file because all of them will be stored in the repository. So next time wherever you install this application that repository will be used for any dependencies lookup. In this way, your deployment file will be very light.

## **6. Mention the steps for installing Maven on windows.**

Maven can be downloaded and installed on windows, linux, and MAC OS platforms. To install Maven on windows, you need to perform the following steps:

* + Download Maven and extract it.
  + Add JAVA\_HOME and MAVEN\_HOME in the list of environment variables.
  + Add the environment path in Maven variable.
  + The last step is the verification of Maven by checking its version.

## **7. What does it mean when you say Maven uses Convention over Configuration?**

* + In the case of Configuration, developers have to create the build processes manually and they have to specify each and every configuration in detail. But, Maven uses convention where the developers need not create the build processes manually.
  + Also, for convention, users do not need to specify the configuration in detail. Once a developer creates a project in Maven then Maven will automatically create a structure. Developers just have to place the files appropriately. There is no need to specify any configuration details in pom.xml file.

**NOTE**: This is an important Maven Interview Question that you must know.

## **8. What are the aspects Maven manages?**

Maven provides developers ways to manage following −

* + Builds
  + Documentation
  + Reporting
  + Dependencies
  + SCMs
  + Releases
  + Distribution
  + mailing list

## **9. How do you know the version of mvn you are using?**

Type the following command – mvn -version

## **10. What is POM?**

POM stands for Project Object Model. In Maven, it is a fundamental Unit of Work and it is an XML file. You can find it in the base directory of the project. It consists of information about the project and various configuration details used by Maven to build the project(s).

## **11. What information does POM contain?**

POM contains the following configuration information −

* project dependencies
* plugins
* goals
* build profiles
* project version
* developers
* mailing list

## **12. What is Maven artifact?**

* An artifact is a file, normally a JAR that gets deployed to a Maven repository. A Maven build creates one or more artifacts, such as a compiled JAR and a source JAR.
* Each artifact consists of a group ID, an artifact ID, and a version string. The three together uniquely identify the artifact. A project’s dependencies are specified as artifacts.

## **13. What is the Maven Build lifecycle?**

A Build Lifecycle can be defined as a well-defined sequence of phases. It clearly defines the order in which the goals are to be executed. Each build phase contains a sequence of goals. If one life cycle is executed, all build phases in that life cycle are executed. If a build phase is executed, all build phases before it in the pre-defined sequence of build phases are executed.

**NOTE**: This is another important Maven Interview Question that you must know.

## **14. Name the 3 build lifecycle of Maven.**

The three build lifecycles are −

**clean**: cleans up artifacts created by prior builds.

**default**: used to build the application.

**site**: generates site documentation for the project.

## **15. What is the command to build your Maven site?**

Type the command − mvn site

## **16. What would the command mvn clean do?**

This command deletes the target directory with all the build data before starting the build process.

## **17. What are the different phases of a Maven Build Lifecycle?**

Following are the phases of Maven build lifecycle −

**validate** − validate the project and check if everything is correct and all necessary information is available.

**compile** − this phase compiles the source code of your project.

**test** − tests the compiled source code by using a suitable unit testing framework. These tests should not require the code to be packaged or deployed

**package** − takes the compiled code and packages it in its distributable format.

**integration-test** − processes and deploys the package if possible into an environment where integration tests can be run.

**verify** − runs any checks to verify the package is valid and meets the required quality criteria.

**install** − installation of the package into the local repository. This is done to use it as a dependency in other projects locally.

**deploy** − done in an integration environment or release environment. Here the final package is copied to the remote repository for sharing with other developers and projects.

## **18. What is a goal in Maven terminology?**

A goal represents a specific task that contributes to the building and managing of a project. It is bound to zero or more build phases. A goal that is not bound to any build phase could be executed outside of the build lifecycle by invocating it directly.

## **19. What would the command mvn clean dependency:copy-dependencies package do?**

This command will clean the project, copy the dependencies and package the project (executing all phases up to package).

## **20. What phases does a Clean Lifecycle consist of?**

The clean lifecycle consists of the following phases −

pre-clean

clean

post-clean

## **21. What phases does a Site Lifecycle consist of?**

The phases in Site Lifecycle are −

pre-site

site

post-site

site-deploy

## **22. What is Build Profile?**

A Build profile is a set of configuration values that can be used to set or override default values of Maven build. Using a build profile, you can customize build for different environments such as Production and the Development environments.

## **23. What are different types of Build Profiles?**

Build profiles are of three types −

**Per Project** − Defined in the project pom.xml file.

**Per-User** − Defined in Maven settings XML file &#40;%USER\_HOME%/.m2/settings.xml&#41;

**Global** −Defined in Maven global settings xml file &#40;%M2\_HOME%/conf/settings.xml&#41;

## **24. How can you activate profiles?**

A Maven Build Profile can be activated in the following ways −

* Explicitly using command console input.
* Through maven settings.
* Based on environment variables (User/System variables).
* OS Settings (for example, Windows family).
* Present/missing files.

## **25. What is a Maven Repository?**

A repository is a place i.e. a directory where all the project jars, library jar, plugins or any other project specific artifacts are stored and this can be used by Maven easily.

## **26. What types of Maven repository?**

Maven repositories are of three types –

**Local:** Maven local repository is a folder location that is present on your machine. It is created when you run any maven command for the first time. Maven local repository is a location where you can find your project’s all dependencies (library jars, plugin jars etc).

**Central:** It is a repository provided by the Maven community. It contains a huge collection of commonly used libraries. When Maven does not find any dependency in local repository, it starts searching in central repository using the following URL: http://repo1.maven.org/maven2/.

**Remote:** Sometimes, Maven is not able to find a mentioned dependency in the central repository as well then it stops the build process and an output error message is displayed on the console. To avoid such a situation, Maven provides the idea of Remote Repository which is nothing but the developer’s own custom repository containing required libraries or other project jars.

## **27. Can you tell me the default location of your local repository?**

~/m2./repository.

## **28. Tell me the command to install JAR file in local repository.**

mvn install

## **29. Is there a particular sequence in which Maven searches for dependency libraries?**

Following is the search pattern –

* Search for dependency in the local repository, if not found, move to step 2 else do the further processing.
* Search for dependency in the central repository first, if not found and the remote repository is mentioned then move to step 4 else it is downloaded to the local repository for future reference.
* If a remote repository has not been mentioned, Maven simply stops the processing and throws an error (Unable to find dependency).
* Search for dependency in the remote repository first, if found it is downloaded to the local repository for future reference otherwise Maven is expected to stop processing and throws an error.

## **30. What are the uses of Maven Plugins?**

Maven Plugins are used to −

* create jar file.
* create war file.
* compile code files.
* unit testing of code.
* create project documentation.
* create project reports.

## **31. What are the types of Maven Plugins?**

Maven provides the following two types of Plugins −

**Build plugins** −They come into picture during the build and should be configured in the <build/> element of pom.xml

**Reporting plugins** −They get executed during the site generation and they should be configured in the <reporting/> element of the pom.xml

## **32. When does Maven use the External Dependency concept?**

Maven dependency management uses the concept of Maven Repositories (Local, Central, Remote). Suppose dependency is not present in any of remote repositories and central repository then in such case Maven uses the concept of External Dependency.

## **33. What are the things that you must define for each external dependency?**

External dependencies (library jar location) can be configured in pom.xml in same way as other dependencies are configured.

* First, specify groupId the same as the name of the library.
* Then specify artifactId the same as the name of the library.
* Thirdly, specify scope as a system.
* Lastly, specify the system path relative to the project location.

## **34. What is Archetype?**

An archetype is a Maven plugin whose task is to create a project structure as per its template.

## **35. What is the command to create a new project based on an archetype?**

Type the following command − mvn archetype:generate

## **36. What is SNAPSHOT in Maven?**

SNAPSHOT can be defined as a special version that indicates a current development copy. Unlike the regular versions, Maven checks for a new SNAPSHOT version in its remote repository. Maven does it for every build.

## **37. What is the difference between Snapshot and Version?**

* In the case of Version, if Maven once downloads the mentioned version say data-service:1.0, it will never try to download a newer 1.0 available in the repository. To download the updated code, the data-service version is then upgraded to 1.1.
* In the case of SNAPSHOT, Maven will automatically fetch the latest SNAPSHOT (data-service:1.0-SNAPSHOT) every time the team builds its project.

## **38. What is a transitive dependency in Maven?**

Transitive dependency is nothing but to avoid the need to discover and specify the libraries that your own dependencies require, and including them automatically.

**39. What does dependency management mean with respect to transitive dependency?**

It simply means to directly specify the versions of artifacts to be used when they are encountered in transitive dependencies. For example project C can include B as a dependency in its dependencyManagement section and directly control which version of B is to be used when it is ever referenced.

## **40. How Maven handles and determines what version of dependency will be used when multiple versions of an artifact are found?**

If you find two dependency versions at the same depth in the dependency tree, then you use the first declared dependency. This is nothing but dependency mediation.

## **41. What is the dependency scope? Name all the dependency scope.**

Dependency scope typically includes dependencies as per the current stage of the build. The various Dependency scopes are −

**compile** − This scope indicates that dependency is available in the classpath of the project. It is the default scope.

**provided** − This indicates that the dependency is to be provided by JDK or web-Server/Container at runtime.

**runtime** − This scope tells that you dont need dependency is for compilation but you need it for for execution.

**test** − This scope states that the dependency is only available for the test compilation and execution phases.

**system** − This scope indicates that you must provide the system path.

**import** − This scope is only used when the dependency is of type pom. This scope tells that the specified POM should be replaced with the dependencies in the POM’s <dependencyManagement> section.

## **42. What is the minimal set of information for matching dependency reference against a dependencyManagement section?**

{groupId,artifactId,type,classifier}.

## **43. Is it possible to refer a property defined in your pom.xml file?**

* To refer a property that is defined in your pom.xml, the property name uses the names of the XML elements that define the value, with “pom” being allowed as an alias for the project element i.e root.
* So ${pom.name} points to the name of the project, ${pom.version} refers to the project version, ${pom.build.finalName} refers to the final name of the file created during the built project packaging etc.

**NOTE**: This is a frequently asked Maven Interview Question that you must know.

## **44. What are the default values for the packaging element? If there is no packaging element defined? What is the default value for that?**

The valid packaging values include jar, war, ear, and pom. If no packaging value has been specified, then by default it will be a jar.

## **45. What is the use of the execution element in pom file?**

## **The <execution> element contains the information required for the execution of a plugin.**

## **46. What is a project’s fully qualified artifact name?**

<groupId>:<artifactId>:<version>

## **47. If you fail to define any information, where does your pom inherits that information from?**

All POMs are inherited from a parent despite explicitly defined or not. This base POM is called Super POM and it contains values that are inherited by default.

## **48. How profiles are specified in Maven?**

Profiles are specified by making use of a subset of the elements that are available in the POM itself.

## **49. What are the elements in POM that a profile can freely modify when specified in the POM?**

<repositories>, <pluginRepositories>,<dependencies>, <plugins> ,<properties>, <modules><reporting>,<dependencyManagement>,<distributionManagement>

## **50. What are the benefit of storing JARS/external dependencies in the local repository instead of a remote one?**

It uses less storage and also makes checking out a project quicker, without the need for versioning the JAR files.

## **51. What is a system dependency?**

Dependency with scope system is always available and is not looked up in the repository, they are usually used to tell Maven about dependencies that are provided by the JDK or the VM. Thus, system dependencies are especially useful for resolving dependencies on artifacts that JDK normally provides.

## **52. What is the use of optional dependency?**

You can mark any transitive dependency as optional using the “optional” element. As an example, A depends upon B and B depends upon C. Now B marked C as optional. Then A will not use C.

## **53. What is dependency exclusion?**

You can exclude any transitive dependency using the “exclusion” element. Suppose if A depends upon B and B depends upon C then A can be marked C as excluded.

## **54. How to run the clean plugin automatically during the build?**

For this, you just need to place the clean plugin inside the execution tag in pom.xml file.

## **55. What is the meaning of the message “You cannot have two plugin executions with the same or missing elements”?**

It simply means that you have executed a plugin multiple times with the same <id>. To correct this you just need to provide each <execution> with a unique <id>.

**Log4j**

**1.What is logging?**

Logging is the process of writing log messages during the execution of a program to a central place.

**2.Why do we need logs?**

* Console will run only the current logs ,so when closed all the logs will be lost,So we need logs

**3.Log4j components:**

a) **Logger ->**

* Logger is a class inside org.apache.log4j.\*package
* Every class need to initialize the objects
* We have to initialize one logger objects for each java class
* We have logger class methods to indicate log levels
* Logger class methods : debug,Info ,warn,error,fatal

**b) Appender**

* To write messages into a file or DB or SMTP

Several types of appenders are there ,they are

1. JDBCAppender
2. FileAppender – RollingFileAppender
3. ConsoleAppender

**c) Layout**

* To define the formatting in which logs wil print in a repository

1. Pattern Layout
2. SimpleLayout
3. XMLLayout
4. HTMLLayout

**4. Log4j Priority Levels:**

Debug > info > warn > error > fatal

**5. Different ways to configure log4j:**

**a) Basic configurator**

follows this pattern - “%-4r [%t] %-5p %c %x - %m%n”

Basic configurator is suitable for tutorials and temporary purpose

**b) Property File Configurator**

**c) XML configurator (DOMConfigurator) - course stopped at 10.50**

**Jmeter**

**Test Plan:**

* Series of steps that Jmeter is going to execute when the run button is pressed.
* A complete test plan consist of one or more thread groups,logic controllers,sample generating controllers,listeners,Timers,Assertions and Configuration elements

1.Adding Elements

2.Removing Elements

3.Saving the Test Plan

4.Running a Test Plan

5.Stoppinga test (Stop – Immediate shutdown , Shutdown – Graceful shutdown)

6.Logging the Info and errors

**Thread group:**

* Beginning point of any test plan
* Thread group element controls the **number of threads or users** Jmeter will use to execute the test
* **All controllers and samplers** must be under a thread group
* Other elements Listeners may be placed directly under the Test plan,so that wll be applicable to thread group as well which is under the test plan

**Main properties of Thread Group:**

* Set the number of threads(simulation of number of concurrent users)
* Set the number of times to execute the test(otherwords LOOPS)
* Set the ramp-up period(Time take to threads up and running = Ramp up period / Thread)
* Example – If 10 threads and rampup duration as 100 seconds given,then all the thread running in 100/10=10 seconds (which means every 10 seconds 1 thread will executed)

**Controllers:**

**1.Logic Controllers :** **A logic controller in JMeter is a component that lets you customize how JMeter sends requests**

1. Simple Controller : Used simply save the transaction in container or folder
2. Recording Controller - Used for recording the requests
3. Transaction Controller – Used to save the transaction in container by transaction by transaction also it gives performance meterics on parent level
4. Module Controller - Used to re-run the already ran requests or samplers
5. Interleave Controller – One sampler is executed per iteration ,it starts from top then to bottom (Example – If we have 5 samplers ,for first iteration ,1st sampler will executed ,then in second interation ,second sampler will be executed rest of the samplers will be skiiped)
6. Runtime Controller ; Runtime controller controls the execution of sampler/request at a given time
7. Random Controller : it picks one of the children samplers randomly and runs it

**2.Samplers**

**Samplers:**

* Samplers are used to Send request to a server and wait for the response
* If multiple samplers are present in the test tree,they are processed according to the order they are present

Samplers =====è Send Request to Server and Wait for the Response

* Examples samplers like HTTP Request etc
* ***FTP Request***
* ***HTTP Request***
* ***Java Request***
* ***SMTP Sampler***
* ***JDBC Request***
* ***SOAP/XML RPC-Request***

**Listeners**

* Whenever Jmeter runs there will be some information obtained, by using listeners we can access the information
* Listeners can be added in Test Plan or Thread group level
* **Listeners** allows us to view and analyze the Sampler request and response in the form of tables, graphs, trees etc.

Few Example listeners of Jmeter are

**Aggregate Graph** – Generate Bar graph

**Aggregate Report** – Generate Table rows

**Assertion Results** – It will tell you the assertion applied on the sampler (don’t add assertion on load test as it will consume more resources of cpu and memory)

**View Results in Table** – Generate reports in Row and columns (don’t add assertion on load test as it will consume more resources of cpu and memory)

**View Results Tree** - View Results Tree displays a tree consists of all the Sampler responses along with their requests.

Backend Listener

BeanShell Listener

**ConnectTime -** Represents the time to establish connection to the server.

**Latency Time**: JMeter measures the latency from just **before sending the request to just after the first response has been received.** (Connect time is included while calculating Latency Time)

**Elapsed time**: JMeter measures the elapsed time from just before sending the request to just after the last response has been received

**Aggrregate Report:**

**Average Time** = Total number of samples/No of samples

**Median** is value separating higher half and lower half of the table data set

If odd number of sample Is there ,then middle value as median,

Eg {1,2,4,6,9} = odd number data set median is4

If even number take sum number which is in middle

(group the sample in ascending order)

Eg {1,2,4,6}> even data set

Average of middle 2 number is the median here 2+4=6,6/3=3

**Throughput** : Number of transactions per second

**Assertions:**

* Assertion in JMeter is used **to validate response of the request, that you have sent to the server.**
* Assertion is a process where you verify expected result with the actual result of the request at run time.

**1: Response Assertion:**

* Response Assertion can be used to **add and compare pattern strings** against one or many values of server response

. For Example, when you send a request to the URL: https://www.google.com and get the server response. Here, you can verify this response by using Response Assertion.

You can insert "<title>Google</title>" as a "Pattern to Test" field value in Response Assertion. If response doesn’t contain this string, it will fail the sampler.

**2.Size Assertion**

* Size assertion used to verify response contains **expected number of bytes or NOT**

**3.Duration Assertion**

* Duration assertion is used to verify response received at specified time or not.If the response received longer than the mentioned time it will get failed

**4.XML Assertion**

* XML assertion is used to verify response data compromises of correct XML document or NOT

**5.HTML Assertion**

* HTML assertion is used to verify response data compromises of correct HTML syntax or NOT

**6.Xpath Assertion**

* Tests a response of particular request which will validate the xpath with expected result

**PreProcessors executes before executing any sampler request**

1.BeanShell PreProcessor - For Randaom token creation and validation before sending request

2.JSR223 PreProcessor - Similar to Beanshell preprocessor additional we can use java,java script,grovvy programing language to script

**Post Processor** executes when Sampler request execution is done. There are various Post processors that are being used in JMeter.

* Regular Expression Extractor
* CSS/JQuery Extractor
* XPath Extractor
* BeanShell Post Processor
* Result Status Action Handler
* JSR223 Post Processor
* JDBC Post Processor
* JSON Path Post Processor
* Boundary Extractor
* BSF Post Processor
* Debug Post Processor

**Timers :** It is used to delay the execution of the samplers

1.Constant Timers

2.Gausian RandomTimers

Example – if we run 10 loop sampler ,for the second loop if we added constant timers ,the execution get delayed

Runing jmeter test via cmd line and gennerate HTML report

**Create report from an existing CSV file**

================================

Jmeter –g “location of csv file” -o “location of result file”

Jmeter –g testresults.csv -o Desktoppath

To create a report while testing:

========================

Jmeter –n –t “location of jmeter script” -l “location of result fi;e”

-e –o “location of output folder”

**Record and Run play:**

Run with Firefox browser

* Create a HTTPS Test Recorder in Jmeter
* Enter the Port number here as 8888
* Target Controller should be Testplan > HTTPS test recorder
* Add Suggested excludes to remove unwanted png,gif files
* Set manual proxy and port as 8888 in Firefox connecrtion settings
* For HTTPs connection certificate needs to install

**Run the Jmeter test plan in non gui mode:**

Step 1 : Go to the bin folder > cmd >jmeter -n -t demotest.jmx

jmeter -n -t <testplan name .jmx>

To monitor result below cmd used

Step 1 : Go to the bin folder > cmd >jmeter -n -t demotest.jmx -l C:\Result.jtl

Importance of Blazemeter Cloud execution

1.Prerequiste – create a test plan in jmeter

2.go to jmeter test

3.Uplaod the already created jmx file into blazemeter

4.Results will populate

Jmeter Distributed Mode

Your java profiler tool – helps for server monitoring

**Keyword Driven**

**Keyword Driven Framework:**

* Keyword Driven Framework is a type of Functional Automation Testing Framework
* Also called Table-Driven testing or Action Word based testing.

**Why keyword driven is Needed?**

* The Keyword Driven approach in automation is to separate the coding from the test case & test step. This helps a non technical person to understand the automation very well.
* Example – A Manual tester can create automation script by using Keyword driven framework

**Structure of Keyword driven framework:**

The basic working of the Keyword Driven Framework is to divide the Test Case into **4 different parts**.

**1.Test case**

a.Test Steps

* It is a very small description of the Action going to perform on Test Object.

**2.Test Data**

* Data can be any value which is needed by the Object to perform any action, like Username value for Username field

**3.Action (What to do)**

* It is the name of the action, which is going to perform on any Object such as click, open browser, input etc.

**4.Test Object**

* It is the name of the Web Page object/element, like Username & Password

**Creating Project Setup:**

1.Create Maven project (add dependencies in POM.xml) – just add selenium dependency as of now

Also add maven plugin in pom to update JRE to 1.8

Go to maven > Update Project

========================================================================================= **Page Object Model :**

**Why Page object model?**

Start with problem statement – Application undergoes changes always, So code we which worked earlier may not run again due to changes, In order to **withstand these changes with minimal change ,**we need **Page Object model,Using this we achieve easier maintenance**

Page Object Model in Selenium – Page Object model is an object design pattern in Selenium, where web pages are represented as classes, and the various elements on the page are defined as variables on the class.

**Test case ===> extract ===> how to find element**

**Features of Page Object Framework:**

* **Reusable**
* **Organised**
* **Manageble**
* **Understandable**
* **Scalable**
* **Cost Effective**
* **Rules**
* **Standard**

**Aim : Page object is created to find the elements**

**Steps:**

1.Created **test cases** with class file and package - Login functionality - Logintestcase

2.Create a package and class for **Objects** – In this Approach we will create separate pages for each elements ,example – for Logintestcase

we have login elements like username,password and submit,So in this Loginpageobjects willl have username,password and submit

3.Normally create method for each element like public void methodname()

{

} in Objects page

4.Paste the driver (WebDriver driver) element in Objects inside the method body

5.Create NEW Objects in test case and Call the method with driver ( loginpageobjects.username(driver)

Now driver shows error ,Pass the Webdriver driver inside the method bracket

6.Now when we try to enter usernmae(driver).sendkeys value is NOT coming bcz of VOID in methods in Object page

Change the void to WebElement so it will return the webElement

7.Now we have Objects for Objectspage is created on Logintestcase page,it is not good practice to have one class dependent on another class ,so we going to make it as Static ,So by using Static we can use classname without creating objects

8.Now we can remove the objects created in Logintestcase page

9.Just type the Object class name.method example – LoginPageobjects.username(driver).sendkeys(“”);

**Can we find element without using web element?**

YES.Using **Page Factory** we can do that

Page factory available as a Inbuilt to selenium

**Enhancing POM using Page Factory**

1.Create variables example public static WebElement username

2.Enter @Findby (name=”password”)name or Id or anything and pass the locators value which we got from element

3.Now go to testcase enter objects.username.sendkeys for all the elements

4.Now it throws error bcz we need driver

5.Use pagefactory class of selenium,use initelement

why this pagefactory class is benefit it simply the code,we no need to enter driver.findbyelement each time for each elements instead simply we will have findby for all the elements specified in objectclass

Simply we can call it in test case like below

**5) What is Page Factory or Page Factory Class?**

A) Page Factory is an inbuilt page object model concept for Selenium Web Driver, but it is much optimized. Page Factory can be used in any kind of framework such as Data Driven, Modular or Keyword Driven. Page Factory gives more focus on how the code is being structured to get the best benefit out of it.

**InitElement Method** – Wil execute the mentioned driver and class and , the class we mentioned have all the web elements.

**@Findby Annotation –** This method belongs to Page factory,Using this we will give selector with element details ,it works similar to findelement By

@Findby (xpath or id or name = “”)

Public static Webelement login;

Pagefactory.initElments(driver,class file”)

**Can we skip findby annotations and execute script?**

Yes .we can do by directlry passing ID or Name on webelement with pagefactory

Example public static Webelement txtusername;

**Best Practice:** Use pagefactory with findby annotations

**Practical Industry Standard Design:**

* Project
* PageObjects
* CommonActions
* Config file
* TestNG XML
* Log4j file

**2) What is a page object model in selenium?**

A) Page Object Model in Selenium – [Page Object model](https://www.selenium.dev/documentation/en/guidelines_and_recommendations/page_object_models/) is an object design pattern in Selenium, where web pages are represented as classes, and the various elements on the page are defined as variables on the class.

**3) Is Page object model a framework?**

A) Page Object Model is a Design Pattern which has become popular in Selenium Test Automation. Page object model (POM) can be used in any kind of framework such as modular, data-driven, keyword driven, hybrid framework etc.

**4) What is the Page Factory Class?**

A) The Page Factory Class is an extension to the Page Object design pattern. It is used to initialize the elements of the Page Object or instantiate the Page Objects itself. Annotations for elements can also be created (and recommended) as the describing properties may not always be descriptive enough to tell one object from the other.

**5) What is Page Factory?**

A) Page Factory is an inbuilt page object model concept for Selenium Web Driver, but it is much optimized. Page Factory can be used in any kind of framework such as Data Driven, Modular or Keyword Driven. Page Factory gives more focus on how the code is being structured to get the best benefit out of it.

**6) What is the difference between Page Object Model (POM) and Page Factory?**

A) 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.

**7) What is Test Class?**

A) Test Class – In Test Class, we will write an actual selenium test script. Here, we call Page Action and mentioned actions to be  
performed on Web Pages. For each page, we can write our own test class for better code readability. We can write test cases in @Test annotation.

**8) What is Page Action Class?**

A) Page Action Class – In Page Action Class, we can write all web pages action as per the pages and functionality. Under Page Action component, for each page in the application, we have corresponding Page class.

**9) What is Page Factory Class?**

A) Page Factory class is nothing but Object Repository in other term. For each web page, it has its own Page Object definitions. Each web element should uniquely get identified and should be defined at class level. We will use Find By annotation and will define web element so that we will be able to perform actions on them.

**10) Can you write sample code for Page Factory Class?**

A) Page Factory Class

@FindBy(xpath=”.//\*[@id=’Email’]”)  
publicWebElementgmailUserIDWebEdit;

**11) Can you write sample code for Page Action Class?**

A) Page Action Class  
publicclassPageActions\_Login  
{  
WebDriver driver;  
PageObjects\_Loginpo; // Create Instance to Page Factory  
class  
publicPageActions\_Login(WebDriver driver)  
{  
this.driver = driver; // set webDriver for current Page  
Action  
}  
}

**12) What are the advantages of using page object pattern?**

A) ADVANTAGES OF USING PAGE OBJECT PATTERN  
Easy to maintain.  
Easy readability of scripts – since the test scripts, functions and locators are in different classes it is easy to walk through the code.  
Eliminate redundancy – no duplicity of functions or locators.  
Re-usability of code – a locator or function can be reused in the tests.  
Reliability.  
Test coverage is more since the tests are written program wise.  
Performance of each test can be known.  
The changes is to be made only in Page Factory class if any locator changes – no need to

**Q.1) Why pick Selenium over other automated testing tools?**

Selenium is open-source. It is very easy to adapt compared to other automated tools in the market. Due to this reason, many companies pick up Selenium automated testing over other traditional methods.

**Q.2) List some benefits of Selenium over tools like TestComplete and QTP. Also, what are the disadvantages?**

Selenium does not require a license, unlike TestComplete and QTP, being easy on pockets. The online community offers massive support. The release cycles are smaller, and the feedback is prompt compared to TestComplete and QTP. Further, Selenium works on Mac, Linux, and Windows as well.

On the contrary, Selenium requires a developer to have high coding skills. Whereas QTP and TestComplete require low to moderate level of coding skills, respectively.

**Q.3) What changes have occurred in the various Selenium version upgrades?**

In the first version of Selenium, Selenium v1, it only comprised three suites of tools, which are Selenium IDE, RC, and Grid. The Webdriver was missing. It was only in the second version of Selenium, Selenium v2 that the Webdriver was introduced. Once this was done, Selenium RC was no longer in use.

You can find them in the market, but the support isn’t available. The next version of Selenium is Selenium v3. It consists of Webdriver, IDE, and the Grid. It is currently in use. A newer version, Selenium v4, is also now available.

Selenium IDE is mainly for recording and playing back. The Webdriver is for testing the dynamic web applications using a programming interface. The Grid is used for employing tests in remote host machines.

You must use the IDE for recording and playback of tests. A WebDriver is used for testing active web applications using a programming interface, and the Grid is employed for deploying tests in isolated host machines.

**Q.4) What are the various exceptions in Selenium WebDriver?**

Just like any other programming language, you can find exceptions in Selenium as well. You can find the following exceptions in Selenium WebDriver:

TimeoutException: You get this exception when a command does not perform an action in the specified time.

NoSuchElementException: You get this exception when it cannot find an element with the given attributes on a web page.

ElementNotVisibleException: You get this exception when an element is available in the document object model, but it is not seen on the web page.

StaleElementException: You get this exception when an element is not attached to the document object model or is deleted.

**Q.5) Explain Selenium exception test**

The exception you expect to be thrown inside a test class is an exception test. If you write a test case intending it to throw an exception, you must use the @test annotation and also mention it in the parameters that which exception would be thrown. For instance,

@Test(expectedException = NoSuchElementException.class)

**Q.6) Is there a need for an excel sheet in a project? Is yes, how?**

Excel sheets are used as a data source during testing. Further, it also stores the data set while executing data-driven testing. When excel sheets are used as a data source, it can store:

Application URL: Developers can mention the environment URL under which the testing is executed. For example, testing environment, development environment, QA environment, production environment, or staging environment.

User name and password information: Excel sheets can keep safe the access credentials like the username of a password of various environments. Developers can encrypt and store these details for security reasons.

Test cases: Developers can make a table wherein one column write the test case name and the other which says to be executed or not.

If you are going to use excel sheets for DataDriven Test, you can easily store the information for various duplications to be executed during the tests. For instance, all the data that needs to be written in a text box for testing on a web page can be stored in the excel sheets.

**Q.7) What is POM? List its advantages?**

POM stands for Page Object Model. It is a design pattern for creating an Object Repository for web UI elements. Every single web page in the application must have its own corresponding page class, which is in charge of searching the WebElements in that page and then execute operations on them.

The advantages of using the Page object model are:

* It makes the code readable by letting developers separate operations and UI flows from verification.
* Several tests can use the same Object Repository because it is independent of Test Cases.
* The code becomes reusable.

**Q.8) What is a Page Factory?**

Page Factory offers an enhanced method to execute the Page Object Model by efficiently using the memory, and the execution is done using object-oriented design.

|  |  |
| --- | --- |
| **POM Implementation** | |
| With Page Factory | Without Page Factory |
| Uses By() | Uses @FindBy() |
| No imports are required | Imports Page factory |
| No cache storage | Cache lookup is faster |

 Page Factory initializes the elements of the Page Object or instantiates the Page Objects itself. Annotations for elements can also be produced. It is, in fact, a better way as the describing properties may not be expressive enough to differentiate one object from another

If POM is used without a page factory, instead of having to use ‘FindElements,’ @FindBy is used to look for WebElement, and initElements is used to initialize web elements from the Page Factory class.

@FindBy can accept attributes like tagName, name, partialLinkText , linkText, id, className , css, and xpath.

**Q.9) How do you achieve synchronization in WebDriver? Or, tell us about the different types of wait statements Selenium Web Driver?**

You can find two wait statements in Selenium web driver, namely, Implicit Wait and Explicit Wait.

Implicit wait commands the WebDriver to wait for a little by polling the DOM. It is present for the complete life of the web driver instance, once the implicit wait is declared. The pre-set value is zero. If you set it more than zero, then the behavior will poll the DOM on a regular basis based on the driver implementation.

Explicit wait commands the execution to wait for a little till a condition is attained like:

* elementToBeClickable
* presenceOfElementLocated
* elementToBeSelected

**10) What is the use of JavaScriptExecutor?**

You can execute JavaScript through Selenium Websriver using JavaScriptExecutor. It is an interface that offers this mechanism. It gives methods like “executescript” and “executeAsyncScript” to run JavaScript in the condition of the currently chosen frame or window. An example of that is:

JavascriptExecutor js = (JavascriptExecutor) driver;

js.executeScript(Script,Arguments);

**Q.11) Which function lets you scroll down a page using JavaScript in Selenium?**

The function window.scrollBy() helps you scroll down the page using JavaScript in Selenium. For instance:

((JavascriptExecutor) driver).executeScript(“window.scrollBy(0,500”);

**Q.12) How do you handle mouse and keyboard actions using Selenium?**

Special mouse and keyboard actions are handled using Advanced User Interactions API. It comprises of the Actions and the Action Classes that are required for performing these events. The most used mouse and keyboard events are given by Action class are:

dragAndDrop(): This event performs click-and-hold at the position of the source element, moves.

source, target(): Moves to the position of the target element and releases the mouse.

clickAndHold(): It clicks the current location of the mouse.

**Q.13) What are various types of Selenium frameworks?**

The various types of Selenium frameworks are:

* Keyword Driven Framework: In this framework, the operations and instructions are written in a separate file like Excel.
* Data-Driven Framework: In this framework, full test data is taken from some external source files like an XML, Excel, CSV, or some other database table.
* Hybrid Framework: This framework is a blend of both the Keyword Driven framework and the Data-Driven framework.

**Q.14) Name a few files that serve as a data source for various Selenium frameworks.**

They can be an XML, Excel, CSV, or even a Text file.

**Q.15) What is Selenese?**

Selenese is the group of selenium commands to test a web application. Developers can use Assertions, Actions, and Accessors. Assertions are used as checkpoints. Actions are for running operations, and Accessors are used to store the value of a variable.

**Q.16) What is the major difference between a Page Factory and Page Object Model (POM)?**

A common selenium interview question. A page factory is a method to initialize web elements within the page object on the creation of the instance. On the other hand, the page object model is a class that states the web page and holds its functionalities.

**Q.17) Does Selenium support handling window pop-ups?**

No. Selenium does not support handling pop-ups. An alert, which is a pop-up window, displays a warning message on the screen. You can achieve this by using a few methods like:

Void dismiss(): When the cancel button is clicked in the alert box, this method is called.

Void accept(): When the ‘OK’ button of the alert is clicked, this method is called.

String getText(): If you want to capture the alert message, you must call this method.

Void sendKeys(String stringToSed): If you want to send some information to the alert box, you must call this method.

**Q.18) Explain Robot class**

A Robot class gives control over the keyboard and mouse devices.

The methods comprise:

* KeyPress(): Called on the event where you want to press a key.
* KeyRelease(): Called in the event to release the pressed key.
* MouseMove(): Called in the event when you have to move the mouse pointer in the X and Y coordinates.
* MousePress(): Called in the event when you press the left button of the mouse.
* MouseMove(): Called in the event of releasing the pressed button of the mouse.

**Q.19) How to handle many windows in Selenium?**

The window handle is a special identifier that has the address of all the windows. It serves as a pointer to a window returning the value in the string.

* get.windowhandle(): It gets the current window handle.
* get.windowhandles(): Gets the handles of all the windows opened.
* switch to: Helps in switching across the windows.
* set: Sets the window handles, which is in the form of a string.
* action: helps to execute certain actions on the windows.

**Q.20) What are Listeners?**

The interface that changes the behavior of the system is called listeners in Selenium. They enable customizations of logs and reports. They are of two kinds: TestNG listeners and Webdriver listeners.

**Q.21) Explain Assert and Verify commands**

Assert: An assertion is used to differentiate between the real result and the expected result.

Verify: The test executions aren’t paused no matter if the verify condition is true or false.

**Q.22) How does one navigate back and forth on a webpage?**

It is one of the most common selenium interview questions.

You can use the below methods to navigate back and forth.

driver.navigate.forward

driver.manage.navigate

driver.manage.back

driver.navigate.to(“url”)

**Q.23) How to send ALT/SHIFT/CONTROL key in Selenium WebDriver?**

Typically using the keys like ALT, Shift, or Control, we combine them with other keys to activate a function. We cannot just click them alone. We need to define two methods for the purpose of holding onto these keys while the following keys are

pressed: keyUp(modifier\_key) and keyDown(modifier\_key)

Parameters: Modifier\_key (keys.ALT or Keys.SHIFT or Keys.CONTROL)

Objective: The purpose is it performs a modifier keypress without releasing the modifier key. Following interactions may assume it’s kept pressed.

Parameters: Modifier\_key (keys.ALT or Keys.SHIFT or Keys.CONTROL)

Objective: The purpose is it performs a key release.

So, with a mix of these two methods, we can capture the special function of a particular key.

**Q.24) How do we take screenshots in Selenium WebDriver?**

The TakeScreenshot function helps to take a screenshot in Selenium Webdriver. Further, you can save the screenshot taking by using getScreenshotAs() method.

File scrFile = ((TakeScreenshot)driver).getScreenshotAs(outputType.FILE);

**Q. 25) Can we set the size of the browser window using Selenium? If yes, how?**

Yes. If you wish to maximize the size of browser window, you need to use the code

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

If you wish to resize the current window to a specific dimension, you must use the setSize() method. For instance:

System.out.println(driver.manage().window().getSize());

Dimension d = new Dimension(420,600);

driver.manage().window().setSize(d);

If you wish to set the window to a particular size, you must use window.resizeTo() method.

For instance:

((JavascriptExecutor)driver).executeScript(“window.resizeTo(1024, 768);”);

**Q.26) How to select a value from the dropdown? How to handle a dropdown?**

You most likely will be asked about a question about dropdown and selection of values as it is a little tricky and technical as well.

The most crucial detail you must know is that to work with a dropdown in Selenium, it is important to use of the html tag: ‘select’. You cannot handle dropdowns without using the select tag. Have a look at the code below:

<select id=”mySelect”>

<option value=”option1″>Cakes</option>

<option value=”option2″>Chocolates</option>

<option value=”option3″> Candies</option>

</select>

In the above code, an HTML ‘select’ tag is used to define a dropdown element. The ID of the select tag here is myselect. We have given three options in the dropdown: Cakes, Chocolates, and Candies. You can see that each of these choices has an attached value attribute. For instance, for Cake, the value is Option1, for Chocolates its Option2, and for Candies, it is Option3.

To choose a value, you need to:

Identify the ‘select’ html element by using the findelement()

Example:

WebElement mySelectElement = driver.findElement(By.id(“mySelect”));

Select dropdown = new Select(mySelectElement);

Pick up an option from that dropdown element.

To pick an option from that dropdown, there are three ways:

dropdown.selectByVisibleText(“Chocolates”); → Choosing an option by the text that is seen.

dropdown.selectByIndex(“1”); → Choosing an option using the Index number of that option.

dropdown.selectByValue(“option2”); → Choosing an option using the value of that option.

Note that in all the cases, the option “Chocolates” is selected from the dropdown. Points 1 and 3 are obvious and point two; we say “1” because the indexing starts from zero.

**Q.27) How do you hop to a new tab which opens up after you click on a link?**

On clicking a link on a web page, you need to use the switchTo() command to change the focus of the Webdriver. Example: driver.switchTo().window();

where ‘windowName’ is the name of the window, you want to switch your focus to.

If you do not have the name of the window, you can use the driver.getWindowHandle() command to fetch the name of all the windows that were initiated by the WebDriver. Remember, it will never give you the names of those windows which Webdriver did not initiate.

On getting the name, you need to run through a loop to get to that window. Here is an example:

String handle= driver.getWindowHandle();

for (String handle : driver.getWindowHandles())

{

driver.switchTo().window(handle);

}

**Q.28) How can one upload a file in Selenium WebDriver?**

The command element.send\_keys(file path) is used to upload a file in Selenium Webdriver. But before you that, you must use the html tag: ‘input’ where the attribute type should be ‘file’. Here is an example to understand it better:

<input type=”file” name=”my\_uploaded\_file” size=”50″ class=”pole\_plik”>

element = driver.find\_element\_by\_id(”my\_uploaded\_file”)

element.send\_keys(“C:myfile.txt”)

**Q.29) What is the importance of testng.xml?**

If you are interviewing for Selenium, you surely know the importance of testing. Selenium does not support the generation of the report as well as test case management. We, therefore, use the TestNG framework with Selenium. It is way advanced compared to Junit, and it is easier to implement annotations making TestNG framework the choice with Selenium Webdriver.

You can define the test suites and grouping of test classes in TestNG, by taking commands from the testing.xml file. It is represented in an XML file and not in a test suite within the testing source code because the suite is a feature of execution. A test suite is a group or collection of test cases.

The testng.xml file should contain the name of all the methods and classes that you wish to execute as a portion of that execution flow.

Some of the advantages of using testng.xml file are:

* It lets the execution of multiple test cases from multiple classes
* It lets the execution of test cases in groups, where a single test can belong to multiple groups.
* It lets parallel execution.

**Q. 30) Explain DataProviders in TestNG. Is it possible to call a single data provider method for multiple functions and classes?**

One of the advanced selenium interview questions. DataProvider is a feature of TestNG, enabling developers to write DataDriven tests. It supports DataDriven testing, meaning that the same test method can run multiple times with different data-sets. DataProvider is just a method of passing parameters to the test method.

@DataProvider is a method for providing data for a test method. The annotated method must give back an Object[] where each Object[] can be allocated to the parameter list of the test method.

Yes. It possible to call a single data provider method for multiple functions and classes. The same DataProvider can be used in several functions and classes by declaring DataProvider in a separate class and then using it again in multiple classes.

**Q. 31) What are the features of TestNG?**

* Before and after annotations
* XML based test configuration
* Multithreaded execution
* Open API
* Better reporting
* Data-Driven testing
* Dependent Groups.
* Dependent methods

**SQL**

**SQL: SQL is a standard language for storing, manipulating and retrieving data in databases.**

**What is SQL?**

* SQL stands for Structured Query Language
* SQL lets you access and manipulate databases
* SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987

**What Can SQL do?**

* SQL can execute queries against a database
* SQL can retrieve data from a database
* SQL can insert records in a database
* SQL can update records in a database
* SQL can delete records from a database
* SQL can create new databases
* SQL can create new tables in a database
* SQL can create views in a database
* SQL can set permissions on tables, procedures, and views

**What is RDBMS?**

* RDBMS stands for Relational Database Management System.
* **RDBMS is the basis for SQL,** and for all modern database systems such as MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access.
* The data in RDBMS is stored in database objects called tables.
* A table is a collection of related data entries and it consists of columns and rows

**What is Database Tables?**

* A database most often contains one or more tables. Each table is identified by a name (e.g. "Customers" or "Orders"). Tables contain records (rows) with data.

**What is SQL Statements?**

* Most of the actions you need to perform on a database are done with SQL statements.

**SQL SELECT Statement**

* The SELECT statement is used to select data from a database.

**Example :**

SELECT column1, column2, ...

FROM table\_name;

Here, column1, column2, ... are the field names of the table you want to select data from. If you want to select all the fields available in the table

SELECT \* FROM table\_name;

**SQL SELECT DISTINCT Statement**

* SELECT DISTINCT statement is used to return only distinct (different) values.

**SELECT DISTINCT Examples**

The following SQL statement selects only the DISTINCT values from the "Country" column in the "Customers" table:

**Example**

SELECT DISTINCT Country FROM Customers;

The following SQL statement lists the number of different (distinct) customer countries:

Example

SELECT COUNT(DISTINCT Country) FROM Customers;

**SQL WHERE Clause**

* WHERE clause is used to **filter** records.
* It is used to extract only those records that fulfill a specified condition.

**Example**

SELECT \* FROM Customers WHERE Country='Mexico';

**SQL AND, OR and NOT Operators**

* The AND operator displays a record if all the conditions separated by AND are TRUE.
* The OR operator displays a record if any of the conditions separated by OR is TRUE.
* The NOT operator displays a record if the condition(s) is NOT TRUE.

Example : <https://www.w3schools.com/sql/sql_and_or.asp>

**SQL INSERT INTO Statement**

* INSERT INTO statement is used to insert new records in a table.

**SQL ORDER BY Keyword**

* The ORDER BY keyword is used to sort the result-set in ascending or descending order.
* The ORDER BY keyword sorts the records in ascending order by default. To sort the records in descending order, use the DESC keyword.

**SQL NULL Values**

**What is a NULL Value?**

* A field with a NULL value is a field with no value.
* If a field in a table is optional, it is possible to insert a new record or update a record without adding a value to this field. Then, the field will be saved with a NULL value.

**SQL UPDATE Statement**

* The UPDATE statement is used to modify the existing records in a table.

**SQL DELETE Statement**

* The DELETE statement is used to delete existing records in a table.

**SQL MIN() and MAX() Functions**

* The MIN() function returns the smallest value of the selected column.
* The MAX() function returns the largest value of the selected column.

**SQL LIKE Operator**

* The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.

**Inner Join or Equi Join or Simple Join:**

* Match data of one table with another table
* If matched result will display

Example – Let us compare Employee data and Office table, from employee table we need employee number ,firstname, lastname ,email id and from office table we need city and state

So the query will be

Select employee number,first name.last name ,email id ,city state from employees

**Inner join** Offices on employees.officecode =offices.officecode



**Left join or Outer Join:**

* All records which is matches condition will display
* Remaining records on the left side of the table will also display

A diagram of a table and two circles

Description automatically generated

Example – Select employeenumber, firstname ,lastname,email ,city from employees

Left join Offices On employees.officeCode = offices.OfficeCode;

**Right Join**

* All records which is matches condition will display
* Remaining records on the Right side side of the table will also display
* A green and black circles with black text

  Description automatically generated

Sysntax SELECT column\_name(s) FROM table1 RIGHT JOIN table2 ON table1.column\_name = table2.column\_name;

**Self Join:**

* Same as Inner join display matching records
* Join is applied for 1 one table that itself

Example

Select A.firstname as “**Emp name**” B.Firstname as “**Manager Name**” from employee A Inner join employee B ON A.reportsTo =B.employeeNumber

Select first name first name from employee inner Join employee

So to diferentiate Select A.first name B.first name from employee A Innter Join employee B

Employee A and Employee to diferentiate tables

A.firstname is denoting that from Table A Firstname

**Full Join**

* In full join, all the records form both the tables are merged and selected irrespective of the condition mentioned with the join having met or not. (here 1+2+3)

**Syntax of Full Join:**

SELECT column-names FROM table-name1 FULL JOIN table-name2 ON column-name1 = column-name2 WHERE condition

**Union:**

* The UNION operator is used to combine the result-set of two or more SELECT statements.

SELECT City FROM Customers

UNION

SELECT City FROM Suppliers ORDER BY City;

**Group By :**The GROUP BY statement groups rows that have the same values into summary rows, like

**SQL Injection**

* SQL injection is a code injection technique that might destroy your database.
* SQL injection is one of the most common web hacking techniques.
* SQL injection is the placement of malicious code in SQL statements, via web page input.

**SQL PRIMARY KEY Constraint**

* The PRIMARY KEY constraint uniquely identifies each record in a table.
* Primary keys must contain UNIQUE values, and cannot contain NULL values.
* A table can have only ONE primary key; and in the table, this primary key can consist of single or multiple columns (fields).

**SQL FOREIGN KEY Constraint**

* The FOREIGN KEY constraint is used to prevent actions that would destroy links between tables.

**Data Driven Framework:**

**Why data driven framework?**

* When we try to execute 4 cases for login functionality like Invalid username, invalid password, invalid username/password,valid username/password ,we need to create 4 different class and bunch of codes ,just the differences in code is only the test data ,it will cause more memory, more time, not optizmed way of writing code ,so in order to overcome this we have data driven framework

**Problem1: only 1 test shown in testng report**

We write 1 test tag with 4 cases ,so only 1 test is showing in report,

So resolve this,we neeed to add 4 test tag in testng xml,so in test it will show as 4 tests

**Problem2: we created code with direct test data,this is not best practice,to resolve this we will create**

Parameters tag in testng xml and parameter annnotation and we pass the parameters

**Problem 3 : code have number of lines still huge?So by using TestNG parameter ,our problem is not solved**

**Solution : Use Data provider take value from excel sheet using JXL or Apache POI jars**

**By using Testng – we can do by parameters or Data providers**

**1.Steps with Data provider annotation with JXL Jars:**

1.File Location ->

2.Get Workbook

3.Get Sheet

4.Get Rows

5.Get Columns

6.Cell Iterate and get cell values

Note – 97-2003 Format

**2.Steps with Apache POI with normal java project:**

**Pre-requisite: Download Apache POI Jars**

1.Read the file location

2.Create Object for workbook

3.Go to the sheet to be worked on

4.Row iteration and get row values

5.Cell iteration and get cell values

HSSF(Class) -----> A Microsoft Excel 2003 file - 97-2003 Formet

XSSF(Class)-------> Microsoft Excel 2007 file or later - Excel workbook

XSSFWorkbook and HSSFWorkbook ------> Excel Workbook

HSSFSheet and XSSFSheet are classes which act as an Excel worksheet

Row defines as Excel row

Cell defines and Excel cell addressed in reference to a row

Practical steps:

1.Create a Main class

2. **Read the file location** - Create method to read excel - For reading filelocation we need : FIleInputStream java class - here object is created for Fileinput here anmed excel

3**. Create Object for workbook -** (Need Apache.poi.usermodel.XSSWorkbook jars)

Object is created with FileInputstream object

**4. Go to the sheet to be worked on - Created a variable with Sheet Interface anf get sheetAt**

**5.Sheet iterator will get row values readed**

**=================================================================================**

**Excel > JXL or POI**

**TestNG we can do 2 types of data driven testing**

**1.Parameters - We should avoid this bcz number of code is very high**

**2.Data Providers annotation**

**Extent Report:**

**Report Tool available in Market:**

* Extent Report
* Allure
* Grafana
* Kibana

**Extent Report:**

* Extent Reports is an **open-source reporting library** used for test automation
* It can be easily integrated with major testing frameworks like Junit and TestNG, etc.
* Once an automated test script runs successfully, testers need to generate a test execution report. While TestNG does provide a default report, they do not provide the details.So we use Extent report

**Advantages of using Extent Reports**

* Customizable HTML report with stepwise and pie chart representation.
* Displays the time taken for test case execution within the report.
* Each test step can be associated with a screenshot.
* Multiple test case runs within a single suite can be tracked easily.
* Can be easily integrated with TestNG and JUnit frameworks.

**How to Select Dropdown in Selenium?**

1.By using JavaScriptExecutor class

js.executeScript("arguments[0].value='highestprice'", dd);

2.By using sendKeys

3.By using Actions Class

4.By storing all the options in List and iterating through it

5.By creating Custom Locator and without iterating the List

**What are all the return types in Selenium?**

FindElement = WebElement

FindElements = List<WebElement>

Gethandle = String

Gethandles = Set<String>

Isdisplayed = Boolean

Isselected = Boolean

Isenabled = Boolean

Getattribute = String

**How to download a file in selenium?**

1. Sendkeys method ( Will be applicable ONLY when we have input as tag)
2. Robot Class withString selection classs and Toolkit classs
3. Auto IT third party tool

Source: https://www.softwaretestinghelp.com/file-upload-in-selenium/

**How to handle Tooltip in selenium?**

1. Using getattributes method
2. Using actions class

Synchronization in automation or Selenium?

How to achieve synchronization in the Selenium Web driver?

Different types of synchronization?

Why do we get exceptions when interacting with web elements?

How to resolve by exceptions?

What is Thead.sleep?

Implicit wait – When it is applicable ? –

Problem in using Implicit wait?

What is Explicit Wait:

How to use Explicit wait:

Adv of explicit wait:

Disadv of explicit wait:

Can we mix implicit and explicit wait together?

What is Fluent Wait?

What is Polling?

Difference between GET and Navigate Methods?

Difference ways to take screenshot?

Different ways to refresh a browser:

Different ways to maximise the browser?

Difference between Sleep() and SetSpeed()?

Can we launch browser without system.setproperty()?

Handling authentication Popup?

Different ways to give Press Enter key ??

How to find element without find element?

How to rerun the failed test cases?

How to Scroll a webpage?

How to verify whether a link is underlined?

Assert vs Verify?

How to send keys without sendkeys method?

How to select all the checkboxes in the page?

How to execute a test multiple times with TestNG ?

What is Timeout in testng? How to set timeouts for a testcase in TestNG?

How to handle exceptions in TestNG:

How to Download Browser drivers automatically?

How to hit a URL without using Get or Navigate method:

What is JavaScriptExecutor?

What will you if different web locator such as ID, Name, CSS selector,Xpath, link text, partial linkt ext is not working?

What will you do if click() method may not work on your browsers?

Types of JavascriptExecutor consists of two methods

Uses of JavascriptExecutor?

What is Robot Class?

Benefits of Robot Class?

Difference Between Gettext and GetAttributes?

How to Make Implement Modularization in Automation?