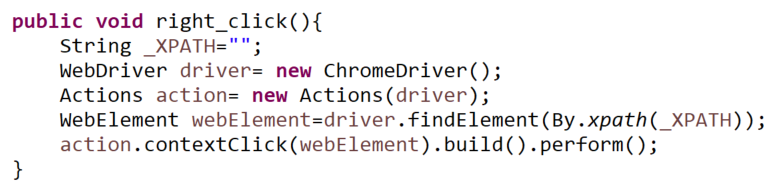
**Test Automation Bootcamp**

**Selenium & Other APIs Questions**

1. **Why do we initialize the driver as ‘WebDriver driver = new ChromeDriver();’? Why not initialize it as ‘ChromeDriver driver = new ChromeDriver();?**
   1. **We could initialize “ChromeDriver driver= new ChromeDriver();” but its limited to Chrome Web Browser only. So to be able to work with all supported browser drivers we initialize using the super class WebDriver interface as “WebDriver driver= new ChromeDriver();”.**
2. **What are the 3 elements required for adding a dependency to the pom.xml file?** 
   1. In Maven Project POM.xml file is an XML file that contains information about the project and configuration, so to add a dependency the following 3 information are required:
      1. **GroupId**
      2. **ArtifactID**
      3. **Version**
3. **The ID attribute value for an input element is “element\_ID”. Write the CSS selector to locate this element.**
   1. **Input#element\_ID**
4. **Give an example of how you would implement index values within a CSS Selector locator strategy.**
   1. **When we have a child element while locating elements on the DOM, we apply index to locate elements in reference to the parent element.**
   2. **Example: on amazon page**

driver.findElement(By.cssSelector("overview-list li:nth-of-child(3)"));

1. **Explain the difference between implicit waits and explicit waits.**
   1. **Implicit wait applies to all elements on the DOM, and If the WebElement is not available immediately the WebDriver wait a certain amount of time when trying find element then exception will be thrown after specified amount of time.**
   2. **Explicit wait applies to the web element explicitly when an element is not available immediately it will wait until the specified wait time and proceed to the next step if the condition becomes true otherwise will throw exception.**
2. **Explain the different types of Xpaths.**
   1. **There are 2 types of XPath locaters to navigate through the element in the DOM.**
      1. **Absolute XPATH: is a method of locating the web element in the DOM reference starting from the root to the desired element, begins with single slash “/” , It has disadvantages when there is a design change and elements are moved we have to update our script.**
      2. **Relative XPATH: Simply start locating the desired element from a particular location in the DOM, it begins with double forward slash “//”, It’s the most preferred way of locating elements for test cases.**
3. **Write the code to perform a right-click on an element using WebDriver.** 
   1. ****
4. **What are the benefits of Automation Testing?**
   1. **Multiple execution of testcases with one script.**
   2. **Saves amount of time to perform test execution.**
   3. **Increased efficiency and faster project completion.**
   4. **Avoids human intervention and error during execution.**
   5. **Ensures consistency and improves accuracy.**
   6. **Reusability of automation scripts**
   7. **Higher test coverage.**
5. **Which interface have you worked with to capture screenshots?** 
   1. **The Interface “TakesScreenshot” which Capture the screenshot and store it in the specified location.**
6. **Provide an example of when you’ve utilized the findElements method.**
   1. **When a WebElement locator returns a list of WebElements we use** findElements**() method returns all the elements matching with the locator specified as a list of WebElements.**

**Example: List webElements= driver.findElements(By.xpath("XPATH"));**

1. **How do you handle working with multiple tabs or windows in a browser?** 
   1. **We get the windows handle using driver.getWindowsHandles(); and perform Action using Actions class and then we utilize the windows handle to go back and forth using** switchTo() **method.**
      1. **To get multiple windows: Actions.keydown(Keys.SHIFT).click(EebElement).build().perform();**
      2. **To get multiple tabs:**

**Actions.keydown(Keys.CONTROL).click(EebElement).build().perform();**

1. **What is the purpose of the .properties files?**
   1. **The purpose of .property file is to store a property resources as a String, that can give as flexibility in managing variables which are changing often with out changing the code in our script.** 
      1. **Test data that can be used by test cases classes as a global variable**
      2. **Creating an object repository when we use Page Object Model frame work,**
      3. **Storing Object Configuration variables.**
2. **Explain the 5 elements of the log4j2.xml configuration file shown below:**
   1. **1: shows the maximum file size in which the log4j logger transaction are recorded.**
   2. **2: Shows the Appenders reference where the log output is to be written on the console or the dedicated log file, in this case it’s on the *File*.**
   3. **3: shows a setting when each test cases are executed the logger applies to the root level and if a class level logger additivity is set to false (by default its true), the duplicate logs will be removed from being recorded.**
   4. **4: shows the root level logger configuration, the transaction that are recorded during runtime either (info, error or warning), appender name (file, Counsel).**
   5. **5: shows the setting for logging level either of trace, warn… that needs to be recorded in the logging file or console explicitly for the test case “listeners class”**



1. **What is the JavaScriptExecutor and when should it be used?**
   1. **JavaScriptExecutor is an Interface that provides a way to automate a user interaction even when page is not essentially loaded completely, or elements are placed in a way that the direct interaction is blocked.**
   2. **Java Script Executor is used in a replacement of WebDriver when WebDriver is unable to perform some Web Element actions, like Highlighting, Un-highlighting, and as a substitute of WebDriver actions like, clicking, sending keys to the input, scrolling till element is visible.**
2. **What are the APIs, Interfaces, and Classes used to Read data from Excel?** 
   1. **The Apache POI in Selenium is a widely used API for selenium data driven testing.**
      1. **API: ApatchePOI**
      2. **Interfaces: XSSF, HSSF Workbook, Sheet, Row, Cell**
      3. **Classes: WorkbookFactory.**
3. **What are some advantages of the Page Object Model Design Pattern?**
   1. **Easy Maintenance and reusability.**
   2. **More Script readability**
   3. **Low Redundancy, reduce duplication of code.**
4. **If you want a method to be executed before any test method is executed, which TestNG annotation would you use?** 
   1. **@BeforeMethod annotation can help us to execute methods before a Test Method.**
5. **What are some examples of TestNG Assert methods?**
   1. **Assert.fail(“String message”);**
   2. **Assert.assertEquals(“Value1”,”value2 ”);**
   3. **Assert.assertTrue(“Boolean value1”, “Boolean value2”);**
   4. **Assert.assertNotEquals(“value1”, “value2”);**
6. **How do you prioritize your test cases in TestNG?**
   1. **We can achieve prioritizing the test case in TestNG in 2 ways:**
      1. **Establishes the priority of the test method (lower priority # will execute first)**

**@Test(priority=0), default @Test when no priority is specified.**

**@Test(priority=1),…..**

* + 1. **Writing the test cases in an alphabetical orderly manner so that by default the test cases (@Test) will be assigned priority of (0) and JVM will execute alphabetically.**

1. **How have you implemented the OOPs concepts within the Automation Frameworks you have designed?** 
   1. **Encapsulation: Encapsulation is a mechanism of binding code and data together in a single unit. when we have a private object in a base class, and we implement public accessor method that can allow other classes to use the private object through public accessor method getters and setters. Also, we declare data members using @FindBy page factory, and initialization of data members will be done using Constructor to utilize those in methods.**
   2. **Inheritance: We create a Base Class in the Framework to initialize WebDriver interface, WebDriver waits, Property files, Excels, extent report in the Base Class. Then We extend the Base Class in other classes such as Tests and Utility Class using inheritance.**
   3. **Polymorphism: when we create a constructor in the base class and perform different implementations or when we use Assert methods with different parameters, when we use implicit wait with different time stamps.**
   4. **Abstraction: Abstraction is the methodology of hiding the implementation of internal details and showing the functionality to the users. we write locators (such as id, name, xpath etc.,) in a Page Class. We utilize these locators in tests, but we can’t see these locators in the tests. Because they are hidden in the object repository. We can achieve 100% abstraction and multiple inheritance in Java with Interface.**
2. **How do you control the execution of multiple tests at the same time in TestNG?** 
   1. **We use TestNG XML file to control the execution of test cases at the same time using parallel execution, more tests are executed in parallel, hence reducing the total execution time.**
3. **What are some of the different TestNG Annotations?**
   1. **@BeforeSuit**
   2. **@BeforeClass**
   3. **@BeforeGroup**
   4. **@BeforeMethod**
   5. **@BeforeTest**
   6. **@Test**
   7. **@AfterTest**
   8. **@AfterMethod**
   9. **@AfterGroup**
   10. **@AfterSuit**
   11. **@AfterClass**
   12. **@DataProvider**
   13. **@Parameters**
   14. **@PageFactory**
   15. **@Listeners**
4. **TestNG’s DataProvider will return the test data in what form?** 
   1. **2D Array List of Objects**
5. **What type of Assertions are available within TestNG?**
   1. **Soft Assertion and Hard Assertion.**
      1. **Soft assertion: asserts the given value, collects errors and proceed to the next line of code, without throwing exceptions.**
      2. **Hard assertion: asserts the given values and stops the execution if the result is false, by throwing Assert Exception and continues to the @Test next method.**
6. **Explain the pom.xml file**
   1. **pom.xml file is a maven-java project used to manage all the dependencies API library needed for the project.**