Linkedln - www.linkedin.com/in/isha-chaudhary-51b9a4196
Ask Any Question - https://topmate.io/ishachaudharyy?utm_source=topmate&utm_medium=popup&utm_campaign=Page_Ready

Selenium Automation Q/A

Ques 1. How does Selenium work?

Ans. Selenium automates web browsers using WebDriver, which communicates with the browser. Here's a simple explanation of its architecture and an example:
Architecture:

Test Script: Written in languages like Java, Python, etc.
WebDriver: Acts as a bridge between the test script and the browser.
JSON Wire Protocol: Used by WebDriver to communicate with the browser driver.
Browser Driver: Specific to each browser (ChromeDriver for Chrome, GeckoDriver for Firefox).
Browser: Where the tests are executed.

Flow:
The test script sends commands to the WebDriver.
WebDriver translates these commands into JSON format and sends them using the JSON Wire Protocol to the Browser Driver.
The Browser Driver communicates with the browser to perform actions like clicking a button or entering text.
Results are sent back in the reverse order.

Ques 2. What are Key Differences Between Selenium 3 and Selenium 4?

Ans 1. Protocol:
Selenium 3: Uses the JSON Wire Protocol, leading to inconsistencies across browsers.
Selenium 4: Adopts the W3C WebDriver Protocol, ensuring better standardization and cross-browser compatibility.

2. ChromeDriver:
Selenium 3: Less optimized integration with ChromeDriver, potential performance issues.
Selenium 4: Improved integration with the latest ChromeDriver, offering enhanced performance and stability.

3. Methods and Features:
Selenium 3:Basic element locators.Limited window management.
Selenium 4:Relative Locators: Locate elements relative to other elements (above(), below(), toLeftOf(), toRightOf(), near()).
Chrome DevTools Protocol: Allows deeper browser interactions like capturing network requests and performance metrics.
New Window Management: Methods for handling multiple windows and tabs more efficiently (newWindow()).

Ques 3. What Test Automation Selenium Framework? Or Explain Test Automation Selenium Framework?

Ans. Definition: A Selenium test automation framework is a structured approach to automate web application testing efficiently. It combines tools, best practices, and design patterns to create effective and maintainable test scripts.

2. Key Components:
Programming Language: Java + Selenium – The main language for writing test scripts. Java is commonly used due to its industry-wide acceptance.
IDE: Eclipse – A popular Java IDE for coding and managing your test scripts.
Testing Framework: TestNG – Manages test execution, provides features like test grouping, parallel execution, and reporting.
Version Control: GIT – Tracks changes in test scripts and supports team collaboration.
Continuous Integration: Jenkins – Automates test execution and integrates with tools like Maven for continuous integration.

3. Key Features:
Page Object Model (POM): Separates page-specific logic into different classes for maintainability.
Test Base Class: Centralizes WebDriver setup and configuration.
Utility Class: Contains reusable functions like taking screenshots.
Property Files: Stores configurations like browser type and URLs.

Test Data: Managed through Excel sheets and Apache POI for data-driven testing.
TestNG: Organizes and runs tests, supports parallel and grouped test executions.
Maven: Manages project dependencies and builds.
Jenkins: Automates test runs and integrates with build pipelines.

Sample Answer
"Our Selenium framework uses Java for scripting and TestNG as the testing framework. We manage dependencies with Maven and use GitHub for version control. The framework is designed with a Data Driven approach and Page Object Model for maintainability. We read test data from Excel using Apache POI and generate detailed reports with Extent Reports. For logging, we use Log4j2 and capture screenshots for failed tests. We follow the DRY Principle to avoid redundancy, create page classes for common components, and use TestNG XML files to manage Test Suites like Sanity and Regression. Jenkins handles test execution and schedules the test suites based on their runtime, e.g., Sanity Suite runs daily and Regression Suites run nightly or weekly."

Ques 4. What are the top 5 advantages of Selenium?

Ans. Open Source: Selenium is free to use and has a large community for support and updates.
Cross-Browser Compatibility: Supports multiple browsers like Chrome, Firefox, Safari, and Internet Explorer.
Cross-Platform Support: Can run on various operating systems including Windows, macOS, and Linux.
Supports Multiple Programming Languages: Works with Java, C#, Python, Ruby, and more.
Integration with Other Tools: Easily integrates with tools like Maven, Jenkins, and Docker for CI/CD.

Ques 5. What are the top 5 advantages of TestNG?

Ans. Flexible Test Configuration: Allows creating and managing test configurations with ease.
Parallel Execution: Supports running tests in parallel, improving execution speed.
Detailed Reporting: Provides comprehensive and customizable test reports.
Annotations: Simplifies the creation of test cases with annotations like @BeforeMethod, @AfterMethod, etc.
Data-Driven Testing: Supports parameterization of test methods to run tests with multiple sets of data.

Ques 6. What are the different components of Selenium?

Ans.
Selenium IDE: A record and playback tool for creating test scripts without programming.
Selenium WebDriver: A tool for writing test scripts in various programming languages to interact with web browsers.
Selenium Grid: Enables running tests in parallel across different machines and browsers for distributed test execution.
Selenium RC (Remote Control): An older tool for writing automated web application UI tests.

Ques 7. Can we run parallel test cases without Selenium Grid? If yes, then why does Selenium Grid come into the picture?

Ans. Yes, parallel test cases can be run without Selenium Grid using frameworks like:
TestNG: Allows configuration for parallel test execution.
JUnit: Supports parallel execution with appropriate configurations.
Cucumber: Can be configured for parallel execution with plugins like Maven Surefire.

Ques 8. Why Selenium Grid?

Ans. Scalability: Run tests on multiple machines and browsers simultaneously.
Cross-Browser Testing: Test on different browser versions and operating systems.
Distributed Execution: Reduce test execution time by distributing tests across multiple nodes.

Ques 9. Key differences between Selenium Grid and TestNG

Ans. Purpose:

Selenium Grid: Focuses on distributed and parallel test execution across multiple environments.
TestNG: A testing framework that provides annotations, test configurations, and supports parallel execution within a single machine.

Parallel Execution:
Selenium Grid: Achieves parallel execution by distributing tests to different nodes/machines.
TestNG: Achieves parallel execution within the same machine or JVM.

Configuration:
Selenium Grid: Requires setting up hub and nodes.
TestNG: Configured through the testng.xml file.

Ques 10. Q. What are the types of annotations in TestNG?

Ans. 1. Pre-conditions:
These annotations set up the environment before the actual test runs. For example, @BeforeMethod runs before each test method to initialize test data or open a browser.
@BeforeSuite
@BeforeTest
@BeforeClass
@BeforeMethod
2. Test:
This is the actual test case logic, annotated with @Test, where the testing actions and assertions are performed.
@Test
3. Post-conditions:
These annotations clean up the environment after the test has run. For example, @AfterMethod runs after each test method to close the browser or reset data.
These annotations clean up the environment after the test has run. For example, @AfterMethod runs after each test method to close the browser or reset data.
@AfterMethod
@AfterClass
@AfterTest
@AfterSuite

Ques 11. What is the sequence of annotations?

Ans.
@BeforeSuite
@BeforeTest
@BeforeClass
@BeforeMethod
@Test
@AfterMethod
@AfterClass
@AfterTest
@AfterSuite

Ques 12. Difference between @BeforeMethod and @BeforeTest?

Ans
@BeforeMethod: Runs before each test method. This means it will execute before each individual test, ensuring that the setup specific to that test method is done properly.
@BeforeTest: Runs before any test method in the test class. This means it will execute once before any of the test methods in the class are run, making it suitable for setting up shared resources or configurations that all tests in the class will use.

Ques 13. Should test cases be dependent on one another? If 10 test cases are dependent on a single test case, what happens if that test case fails? What is the best approach to handle dependencies?

Ans. Dependency of Test Cases

Not a Good Practice - Making test cases dependent on one another can lead to a cascade of failures. For example, if Test Case A fails, and Test Cases B through K depend on A, then B through K will likely fail as well, making it difficult to determine the actual root cause of the issue.

Approach
1. Isolate Test Cases
Ensure each test case can run independently. This reduces the risk of cascading failures and makes it easier to identify issues.
Example: Instead of having a login test that all other tests depend on, include the login steps within each test case or use setup methods to handle the login.
2. Reusable Setup Methods
Avoid duplicating login tests by using reusable setup methods.
Example: Use the @BeforeMethod annotation in TestNG to perform the login once before each test method.
3. Mocking and Stubbing
Use mocks and stubs to isolate dependencies and create predictable test environments.
Example: If a test case depends on an external API, use a mock server to simulate the API responses instead of relying on the actual API.
4. Retry Logic
Implement retry logic for flaky tests to avoid false negatives due to intermittent issues.
Example: Configure your test framework to retry a test case a certain number of times before marking it as failed.
5. Setup and Teardown
Use setup and teardown methods to ensure each test starts with a clean state.
Example: Use the @BeforeMethod and @AfterMethod annotations in TestNG to reset the state before and after each test method.

Ques 14. What are Scenario and Scenario Outline in a Cucumber file?

Ans Scenario: Defines a single test case with specific steps.
Scenario Outline: Used for data-driven testing, allowing you to run the same scenario with different sets of data.

Ques 15. What is Given-When-Then in Cucumber? Can we use them randomly or do we have to use them in a series?

Ans. Given-When-Then:
Given: Describes the initial context or setup.
When: Describes the action or event.
Then: Describes the expected outcome or result.

Usage: Must be used in a series (Given-When-Then) to define the flow of a scenario properly.

Ques 16. What are Background and Hooks? What is the key difference between them?

Ans. Background: Defines steps that are common to all scenarios in a feature file and are executed before each scenario.
Hooks: Allow you to run code before or after each scenario or step (e.g., setup or teardown code).

Key Difference:
Background: Specific to a feature file.
Hooks: Can be used globally across multiple feature files.

Ques 17. What is XPath and How does it actually works?

Ans. XPath (XML Path Language) is a syntax used for navigating through elements and attributes in an XML document. In Selenium, it helps locate elements on a webpage.

How it works:

Path Expressions: XPath uses path expressions to select nodes or node sets in an XML document.
Absolute XPath: Starts from the root element and goes down to the desired element (e.g., /html/body/div).
Relative XPath: Starts from anywhere in the document, searching for the desired element (e.g., //div[@id='example']).

Functions: Supports various functions to find elements (e.g., contains(), text(), starts-with()).
Axes: Allows navigation to related nodes (e.g., parent, child, sibling).

Ques 18. What is the difference between relative and absolute XPath? Which is preferable and why? Any issues/disadvantages with absolute XPath?

Ans Relative XPath:
Starts from the middle of the HTML DOM structure.
Syntax: //tagname[@attribute='value']

Advantages:
More Flexible: Adapts better to changes in the DOM structure.
Easier to Write and Maintain: Shorter and simpler syntax.
Less Fragile: Less likely to break with minor changes to the HTML.

Absolute XPath:
Starts from the root node and navigates through each node to the target element.
Syntax: /html/body/div/div[2]/div[1]/a

Issues/Disadvantages:
Slow Performance: Navigates from the root to the required node, making it slower.
Highly Fragile: Any change in the node hierarchy can break the XPath.
Difficult to Maintain: Longer and more complex syntax, harder to update with changes in the DOM.

Ques 19. Why Prefer Relative XPath?

Ans. Flexibility: Can adapt to changes in the DOM structure without breaking.
Stability: Provides more robust and resilient locators compared to absolute XPath.
Performance: Faster as it doesn't need to traverse the entire DOM.

Ques 20. How to handle alerts or popups in Selenium?

Ans Switch to Alert: driver.switchTo().alert()
Get Text from Alert: alert.getText()
Accept Alert: alert.accept()
Dismiss Alert: alert.dismiss()

Ques 21. How to handle iframes in Selenium?

Ans Switch to iframe:
By Index: driver.switchTo().frame(index)
By Name or ID: driver.switchTo().frame("nameOrId")
Then after switching -
Perform Action: Locate elements and interact with them within the iframe, e.g., driver.findElement(By.id("elementID"))

Ques 22. How do you know an element is within an iframe?

Ans Use browser developer tools to inspect the page structure. If an element is inside an <iframe> tag, it needs to be accessed by switching to that iframe first.

Ques 23. How to handle drag and drop in Selenium?

Ans Action Class: Use the Actions class to build complex actions.
Click and Hold: Select the source element to drag.
Move to Element: Move to the target element where you want to drop.
Release: Release the source element to complete the drop.
Build and Perform: Combine all actions and execute them.

Code
```
WebElement fromElement = driver.findElement(By Locator of fromElement);
WebElement toElement = driver.findElement(By Locator of toElement);
Actions builder = new Actions(driver);
Action dragAndDrop = builder.clickAndHold(fromElement)
.moveToElement(toElement)
.release(toElement)
.build();
dragAndDrop.perform();
```

Ques 24. How to handle browser popup windows using window handles in Selenium?

Ans Get Window Handles: Retrieve all window handles.
Iterator: Use an iterator to navigate through the handles.
Parent Window: Store the handle of the main window.
Child Window: Store the handle of the popup window.
Switch to Child: Switch the driver's focus to the popup window.
Perform Actions: Execute the necessary actions on the popup window.
Close Child: Close the popup window after performing actions.
Switch to Parent: Switch back to the main window.

Code
```
Set<String> handles = driver.getWindowHandles();
Iterator<String> it = handles.iterator();
String parentWindowId = it.next();
String childWindowId = it.next();
driver.switchTo().window(childWindowId);
driver.close();
driver.switchTo().window(parentWindowId);
```

Ques 25. How is TestNG different from Selenium WebDriver?

Ans. TestNG: A testing framework providing annotations, test configurations, parallel execution, and reporting.
Selenium WebDriver: A tool for automating web browser interactions through code.

Ques 26. How can we disable or prevent a test case from running?

Ans @Test(enabled = false): Use this annotation to disable a specific test case in TestNG.

Ques 27. How can we make one test method dependent on others using TestNG?

Ans @Test(dependsOnMethods = "methodName"): Use this annotation to make a test method run only after specified methods.

Ques 28. How to parameterize TestNG tests using data providers?

Ans @DataProvider(name = "dataProviderName"): Create a method to provide test data.
@Test(dataProvider = "dataProviderName"): Use this annotation to link the test method with the data provider.

Ques 29. Differences between isDisplayed, isSelected, and isEnabled?

Ans isDisplayed: Checks if an element is visible on the web page.
isSelected: Checks if an element, like a checkbox or radio button, is selected.
isEnabled: Checks if an element is enabled and interactable.

Ques 30. Differences between get and navigate to?

Ans. get(): Loads a new web page in the current browser window.
navigate().to(): Can navigate to a URL and allows additional navigation options like back, forward, and refresh.

Ques 31. Differences between NoSuchElementException and ElementNotVisibleException?

Ans. NoSuchElementException: Thrown when an element cannot be found in the DOM.
ElementNotVisibleException: Thrown when an element is present in the DOM but not visible on the page.

Ques 32. What is StaleElementReferenceException?

Ans Thrown when an element is no longer present in the DOM, typically after a page refresh or dynamic content load.

Ques 33. What is the DOM?

Ans. Document Object Model (DOM): A programming interface for web documents, representing the structure of a web page as a tree of objects.

Ques 34. How to input text in a textbox using Send Keys?

Ans. Use the sendKeys method to input text into a textbox
WebElement element = driver.findElement(By.name("textboxName"));
element.sendKeys("text");

Ques 35. How to input text in a textbox without using Send Keys?

Ans. Use JavaScriptExecutor to set the value of a textbox
JavascriptExecutor js = (JavascriptExecutor) driver;
js.executeScript("document.getElementsByName('textboxName')[0].value='text';");

Ques 36. How to handle hidden elements?

Ans. Use JavaScriptExecutor to interact with hidden elements that are not visible on the page
JavascriptExecutor js = (JavascriptExecutor) driver;
js.executeScript("document.getElementById('hiddenElementId').click();");

Ques 37. Why do we typecast JavaScriptExecutor?

Ans Typecasting: Convert WebDriver to JavascriptExecutor to use the executeScript method.
JavascriptExecutor js = (JavascriptExecutor) driver;
Purpose: This typecasting is necessary because WebDriver does not have the executeScript method. By typecasting to JavascriptExecutor, you can execute JavaScript commands within the browser, allowing for more advanced interactions such as manipulating DOM elements directly, handling hidden elements, and performing actions that are not possible with standard WebDriver methods.

Ques 38. How do you upload a file in Selenium?

Ans. Using sendKeys method: Locate the file input element and use sendKeys to provide the file path.

WebElement uploadElement = driver.findElement(By.id("uploadfile"));
uploadElement.sendKeys("C:\\path\\to\\file.txt");
Ensure the file input element is present and interactable before using sendKeys.

Ques 39. How to resize the window size?

Ans Creating a Dimension object: Define the desired window size.

```
Dimension d = new Dimension(1024, 768);
driver.manage().window().setSize(d);
```

Resizing the window can be useful for testing responsive designs and ensuring UI consistency across different screen sizes.

Ques 40. How to highlight an element?

Ans Using JavaScriptExecutor: Change the element's background color.

```
public void flash(WebElement element, WebDriver driver) {
JavascriptExecutor js = (JavascriptExecutor) driver;
String bgColor = element.getCssValue("backgroundColor");
js.executeScript("arguments[0].style.backgroundColor = 'yellow'", element)
js.executeScript("arguments[0].style.backgroundColor = '" + bgColor + "'", element);
}
```

Highlighting elements is particularly useful for debugging and visually verifying that the correct elements are being interacted with during test execution.

Ques 41. Tell me about all OOP concepts used in automation frameworks.

Ans
Interfaces - Define methods that must be implemented by classes. They allow different classes to be used interchangeably if they implement the same interface, promoting flexibility and reusability.

Polymorphism
Overloading (Compile-time / Static / Early Binding)- Methods with the same name but different parameters within the same class.
Overriding (Runtime / Late Binding) - A subclass provides a specific implementation of a method already defined in its superclass.

Abstraction - Hiding complex implementation details and showing only the necessary features of an object. Achieved using abstract classes and interfaces.

Inheritance - A mechanism where one class inherits the properties and behaviors (methods) of another class, promoting code reusability.

Encapsulation - Bundling the data (variables) and methods (functions) that operate on the data into a single unit or class, and restricting direct access to some of the object's components.

These OOP principles help in building scalable, maintainable, and flexible automation frameworks.

Ques 42. What are the types of frameworks in Selenium?

Ans Keyword-Driven
Uses keywords to represent actions, making the test script simple and readable.
Data-Driven
Focuses on separating test data from test scripts, allowing testing with multiple data sets.
Hybrid
Combines the benefits of Keyword-Driven and Data-Driven frameworks.
POM with Cucumber
Page Object Model (POM) combined with Cucumber for behavior-driven development (BDD), making tests more structured and readable.

Linkedln - www.linkedin.com/in/isha-chaudhary-51b9a4196
Ask Any Question - https://topmate.io/ishachaudharyy?utm_source=topmate&utm_medium=popup&utm_campaign=Page_Ready

Ques 43. What are listeners?

Ans Special interfaces in TestNG and JUnit that allow you to alter the behavior of tests during execution. They monitor events like test start, success, failure, and provide hooks to take actions, such as logging or taking screenshots.

Ques 44. When do you use explicit wait? What are expected conditions?

Ans Explicit Wait - Used when you need to wait for a specific condition to occur before proceeding with the next step. It's useful for handling dynamic elements.

Expected Conditions- Include visibility of elements, element to be clickable, presence of elements, text to be present, and more.

Ques 45. What is the use of Desired Capabilities?

Ans Desired Capabilities: A way to configure the WebDriver to set browser properties, define browser versions, platform details, and other configurations to control test execution environment.

Ques 46. How to find broken links or images?

Ans Using Selenium WebDriver:
Get all links/images using findElements(By.tagName("a")) or findElements(By.tagName("img")).
Use HttpURLConnection in Java to send an HTTP request for each link/image URL.
Check the response code: if it's 404 or 500, the link or image is broken.

Ques 47. What can you do if the browser window is not fully maximized using driver.manage().window().maximize();?

Ans If the browser window doesn't fully maximize with driver.manage().window().maximize();, you can try the following solutions:
Use fullscreen()
driver.manage().window().fullscreen();
This will put the browser in fullscreen mode, similar to pressing F11 on the keyboard, covering the entire screen.
Or Manually Resize the Window

Ques 48. How to resize the window size in Selenium?

Ans Use driver.manage().window().setSize(new Dimension(width, height)); to resize the browser window.

Ques 49. How do you handle calendars in Selenium?

Ans Using JavaScript Executor
Why? Use JavaScript Executor when you need to directly set a date without interacting with the calendar UI.

```
JavascriptExecutor js = (JavascriptExecutor) driver;
js.executeScript("document.getElementById('datepicker').value='2024-07-02'");
```

Ques 50. What if interacting with the calendar UI is required to select a date?

Ans Using the Manual Method by Clicking
Why? Use the manual method when you need to navigate through the calendar by clicking "Next" or "Previous" until you reach the desired month and year.

```
while (!(month.equals("July") && year.equals("2024"))) {
driver.findElement(By.xpath("next button locator")).click();
}
driver.findElement(By.xpath("date locator")).click();
```

Linkedln - www.linkedin.com/in/isha-chaudhary-51b9a4196
Ask Any Question - https://topmate.io/ishachaudharyy?utm_source=topmate&utm_medium=popup&utm_campaign=Page_Ready

Ques 51. What if your calendar doesn't clearly display the full date instead it have separate dropdowns, and you need to select it?
Ans Using Dropdowns for Month and Year
Why? This method is ideal when the calendar provides dropdowns, allowing you to select the month and year directly.

```
Select selectMonth = new Select(monthDropdown);
selectMonth.selectByVisibleText("Apr");

Select selectYear = new Select(yearDropdown);
selectYear.selectByVisibleText("2024");

dateElement.click();
```

Thank you !