**Selenium Basic Interview Questions for Freshers**

**1. What is Selenium testing?**

[Selenium](https://intellipaat.com/blog/selenium-projects/) testing is a tool used for automating web applications to verify their functionality. It deals with browsers, clicks buttons, fills forms, and checks if the site works correctly.

***Enroll today in our***[***Selenium Course***](https://intellipaat.com/selenium-training/)***to know more about Selenium!***

**2. Is Selenium 2.0 different from Selenium 3.0? If so, how?**

Selenium 3.0 represents an enhanced iteration of Selenium 2.0, introducing numerous enhancements and novel functionalities. A significant distinction lies in the inclusion of the WebDriver [API](https://intellipaat.com/blog/java-api/) as an integral part of Selenium 3.0, eliminating the need for separate downloads and installations as required in Selenium 2.0.

Moreover, Selenium 3.0 exhibits various notable modifications, encompassing enhanced browser compatibility, improved management of browser plugins and extensions, and fortified security features. Furthermore, this version of Selenium demonstrates heightened compatibility with contemporary web browser editions and operating systems.

**3. Mention some of the popular tools used for Automation testing.**

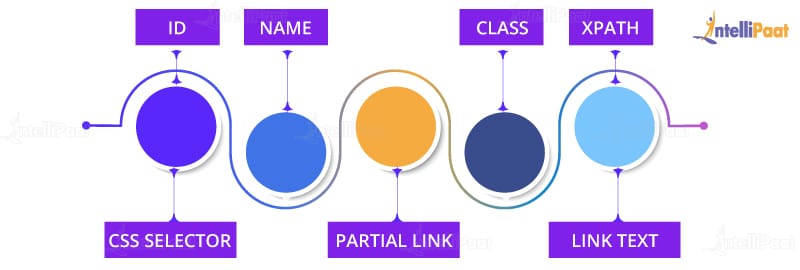
There are several popular tools used for automation testing, including;

* Selenium
* [Appium](https://intellipaat.com/blog/appium-tutorial/)
* TestComplete
* Katalon Studio
* Ranorex
* HP Unified Functional Testing (UFT)
* Apache JMeter, and many more.

These tools help automate the testing process, improve efficiency, and reduce the risk of human error.

**4. What is a Locator? How can you find elements in Selenium?**

Selenium uses locators to find and match the elements of a web page that it needs to interact with. There are different types of [Selenium locators](https://intellipaat.com/blog/locators-in-selenium/) to identify various web elements on a web page:



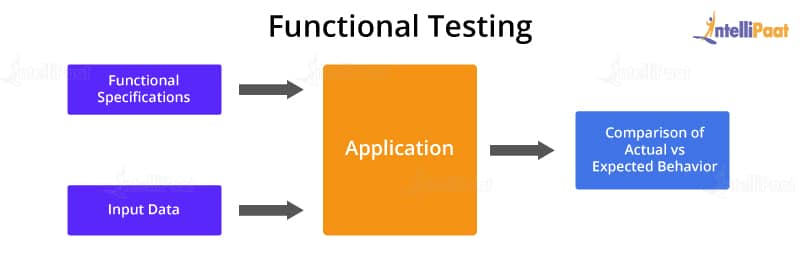
* ID
* Name
* [Class](https://intellipaat.com/blog/select-class-in-selenium/)
* Partial Link
* [XPath](https://intellipaat.com/blog/xpath-in-selenium/)
* [CSS Selector](https://intellipaat.com/blog/css-selector-in-selenium/)
* Link Text

**5. What are the test types supported by Selenium?**

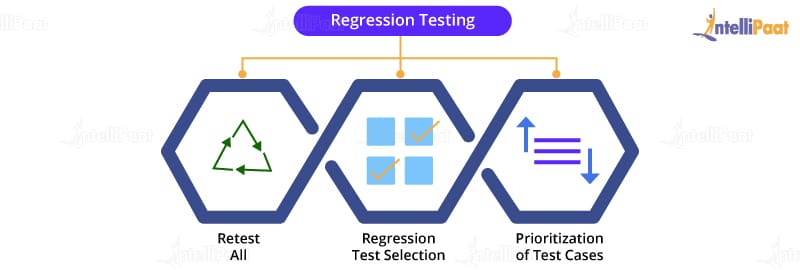
For testing web-based applications, Selenium can be used.

The test types supported by Selenium are:

* **Functional testing**: It verifies if each function of a software application performs in accordance with specific requirements. This testing primarily involves black-box testing, and it is not concerned with the source code of the application.



* [**Regression**](https://intellipaat.com/blog/what-is-regression/)**testing**: It is nothing but a full or partial selection of the already executed test cases to be re-executed to ensure whether the existing functionalities work fine.



**Get 100% Hike!**

Master Most in Demand Skills Now !

Top of Form



By providing your contact details, you agree to our [Terms of Use](https://intellipaat.com/terms-of-use/) & [Privacy Policy](https://intellipaat.com/terms-of-use-and-privacy-policy/)



Bottom of Form

**6. What is XPath?**

While DOM is the recognized standard way for navigating through an HTML element tree, [**XPath**](https://intellipaat.com/blog/xpath-in-selenium/) is the navigation tool used to locate a web element based on its XML path.

XML stands for ‘Extensible Markup Language’ and is used to store, organize, and transport arbitrary data. It stores data in a key-value pair that is very much similar to HTML tags. Both being markup languages and falling under the same umbrella, XPath can be used to locate HTML elements.

The fundamental concept behind locating elements using XPath is traversing between various elements across the entire page and thus enabling a user to find an element with the reference of another element.

**7. Explain the difference between single slash and double slash in XPath.**

* **Single slash (/)**: Single slash is used to create an XPath with an absolute path. In this case, the XPath would start selection from the document’s start node.
* **Double slash (//)**: Double slash is used to create an XPath with a relative path. In this case, the [XPath](https://intellipaat.com/blog/xpath-in-selenium/) would start selection from anywhere within the document.

***Also, check out the blog on***[***Manual testing vs. Automation testing***](https://intellipaat.com/blog/automation-vs-manual-testing/)***.***

**8. Why should you use Selenium for test automation?**

**Selenium should be used for test automation as it:**

* Is a free and open-source tool
* Has a large user base and community support
* Has cross-browser compatibility (Firefox, Chrome, Internet Explorer, Safari, etc.)
* Has great platform compatibility (Windows, Mac OS, Linux, etc.)
* Supports multiple programming languages ([Java](https://intellipaat.com/blog/tutorial/java-tutorial/), C#, Ruby, Python, Perl, etc.)
* Has fresh and regular repository developments
* Supports distributed testing

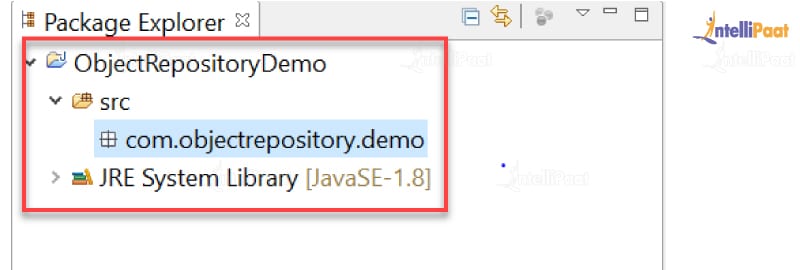
**9. Does Selenium have any technical limitations? If so, what are those limitations?**

Yes, Selenium has a few limitations:

* Testing of only web applications is possible using Selenium.
* Testing of mobile applications or desktop applications is not possible.
* [Captcha](https://intellipaat.com/blog/what-is-captcha/) and barcode readers cannot be tested using Selenium.
* A third-party tool like **[TestNG](https://intellipaat.com/blog/tutorial/selenium-tutorial/testng-in-selenium/" \t "_blank)** or JUnit should be used to generate reports.
* As Selenium is a free tool, there is no ready vendor support through which users can find various helping communities.
* Prior programming language knowledge is expected from users.

**10. What is an object repository?**

An [object](https://intellipaat.com/blog/object-oriented-database/) repository allows testers to accumulate web elements of the application under test (AUT), along with their locator values, in one or more centralized locations as restricted to hard-coding them within the test scripts.



Watch this video on Selenium Automation Testing Interview Questions and Answers:

**11. What is Selenium?**

Selenium is a popular open-source software that is used to automate web-based applications. It is a set of multiple software tools, and each tool has a different approach to automated testing.

**Selenium has four major components, namely:**

* Selenium Integrated Development Environment
* Selenium Remote Control
* Selenium WebDriver
* S[elenium Grid](https://intellipaat.com/blog/selenium-grid/)

**12. What is the difference between type keys and type commands?**

TypeKeys() will trigger JavaScript events, while type() won’t. TypesKeys collects different value attributes using JavaScript. Whereas, the type commands imitate an actual user typing.

**95%** learner satisfaction score post completion of the program\*

**500%** salary hike received by a working professional post completion of the course\*

Fresher earned **30 LPA** salary package on completion of the course\*

**53%** of learners received 50% and above salary hike post completion of the program\*

**85%** of the learners achieved their training objectives within 9 months of course completion\*

**95%** learner satisfaction score post completion of the program\*

**500%** salary hike received by a working professional post completion of the course\*

Fresher earned **30 LPA** salary package on completion of the course\*

**53%** of learners received 50% and above salary hike post completion of the program\*

**85%** of the learners achieved their training objectives within 9 months of course completion\*

**95%** learner satisfaction score post completion of the program\*

**13. What are the advantages of Selenium?**

* Selenium is a purely open-source and portable automation testing tool.
* It supports different languages such as C#, PHP, Java, Perl, [Python](https://intellipaat.com/blog/tutorial/python-tutorial/what-is-python/), JS, and Groovy.
* It also supports different OS, including Windows, Linux, UNIX, and Mac OS.
* It provides powerful methods such as Xpath, DOM, and CSS to locate elements.
* Since it is an open-source tool, developers can customize the code. Also, the developer community is supported by Google.

**14. Define automation testing, and list down its advantages.**

[Automation testing](https://intellipaat.com/blog/what-is-automation-testing/), also known as test automation, involves using tools to automate the testing process, writing and executing test cases without human intervention. It empowers us to develop scripts that can be executed repeatedly and generate comprehensive test reports for the application.

Its advantages are given below:

* It helps with the performance and functional testing of an application.
* It makes the execution of repeated test cases easy.
* It facilitates the concurrent execution of multiple test cases.
* It boosts the accuracy and efficiency of the application by cutting down the chances of human error.
* It efficiently executes tests across an extensive test matrix.
* It saves time and money by reducing the burden of arbitrary tasks.

**15. What are the significant changes/upgrades made to various Selenium versions?**

Selenium’s first version included only three sets of tools: Selenium IDE, Selenium RC, and Selenium Grid. There was no WebDriver included in the first version. Later, [Selenium WebDriver](https://intellipaat.com/blog/what-is-selenium-webdriver/) was introduced and hence included in Selenium V2. However, as WebDriver got included, the use of Selenium RC was discouraged with time and has not been much in use since. Selenium 3 is in use. There are some newly added [**Selenium features**](https://intellipaat.com/blog/tutorial/selenium-tutorial/features-of-selenium/) such as the IDE and WebDriver. Selenium 4 is the latest released version.

**16. What is an exception test in Selenium?**

An exception test is a test that looks forward to an exception being thrown inside a test class. It anticipates the @Test annotation followed by the expected exception name. For example, **@Test(expectedException = NoSuchElementException.class)** is an exception test for missing elements in Selenium.

**19. What are the different types of annotations used in Selenium? Explain the JUnit annotation linked to Selenium.**

In Java, a special form of syntactic metadata can be added to the Java source code, which is known as ‘annotations’. Variables, parameters, packages, methods, and classes are annotated. Some of the [JUnit](https://intellipaat.com/blog/what-is-junit/) annotations are:

* Test
* Before
* After
* Ignore
* BeforeClass
* AfterClass
* RunWith

JUnit annotations linked to Selenium are:

JUnit AnnotationsJUnit Annotations

* @Test: The @Test annotation finds a method to be a test method. When used before a test method, it is mentioned as ‘@Test’; it informs the JUnit framework that the following method is a test method.
* @Before: @Before annotation serves the purpose of identifying the method that should be executed prior to running the test method. Its intended use is to establish and configure the test environment before conducting the actual test.
* @After: The “@After” annotation is utilized as a post-execution method following the execution of the test method. This annotation performs teardown operations, such as deleting temporary data, restoring default values, [cleaning](https://intellipaat.com/blog/what-is-data-cleaning-in-data-science/) up the test environment, and other relevant tasks.
* @BeforeClass: The @BeforeClass method is used only once before the start of all tests. Basically, this is used to perform cumbersome activities, like connecting to a database.
* @AfterClass: The @AfterClass method is used only once after executing all tests. This is used to carry out clean-up activities, like disconnecting from a database.

***Download the***[***Selenium Cheat Sheet***](https://intellipaat.com/blog/tutorial/selenium-tutorial/selenium-cheat-sheet/)***and use it whenever required, especially during your interviews.***

**20. Why do testers choose Selenium over QTP?**

Selenium is more widely used than QTP since:

* Selenium is an open-source tool, whereas QTP is a profitable tool.
* Selenium is used specifically for testing web-based applications, while QTP can be used for testing client–server applications too.
* Selenium supports multiple browsers like Firefox, IE, Opera, Safari, etc., and has multiple operating systems compatibility too. Selenium-supported OS platforms are Windows, Mac, [Linux](https://intellipaat.com/blog/kali-linux-tutorial/), etc. On the other hand, QTP is limited to Internet Explorer on Windows.
* Selenium supports multi-programming language compatibility. Languages supported by Selenium are Python, Ruby, Perl, etc. But, QTP supports only VBScript.

***You can find more on Selenium by visiting our***[***Selenium Community***](https://intellipaat.com/community/selenium)***!***

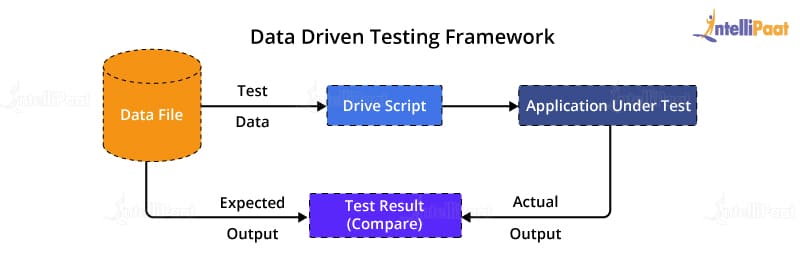
**21. What are the four elements that you have to pass in Selenium?**

Four parameters that need to be passed in Selenium are:

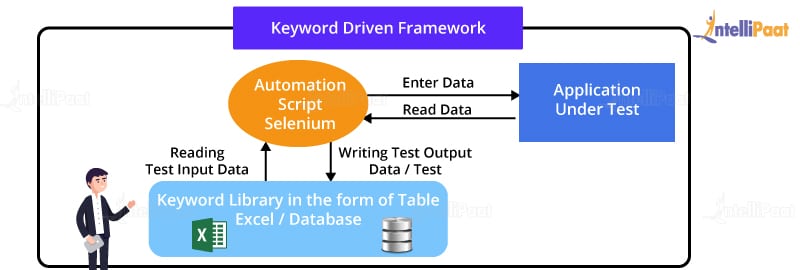
* Host
* Port number
* Browser
* URL

**23. What are data-driven frameworks and keyword-driven frameworks?**

A data-driven framework in Selenium is an approach to separating a ‘dataset’ from the actual ‘test case’ (code). This framework is completely dependent on the input test data. The test data is inserted from external sources, such as an Excel file, a CSV file, or any database. It also allows us to easily control how much data needs to be tested. We can easily increase the number of test [parameters](https://intellipaat.com/blog/parameters-in-tableau/) by adding more username and password fields to the Excel file (or other sources).



A keyword-driven framework is an extension of the data-driven testing framework in the sense that it not only isolates the test data from the scripts but also keeps the particular section of the code belonging to the test script in an external data file. These sets of code are known as keywords, and hence the framework is so named. Keywords are self-guiding and work based on what actions need to be performed on the application.



**24. How will you use Selenium to upload a file?**

If the file is on the same machine or on a mapped network drive, it is really straightforward: We have to just type the ‘path’ of the file in the FileUpload control.

Example:

|  |  |
| --- | --- |
| 1  2  3 | driver = webdriver.Firefox()  element = driver.find\_element\_by\_id("fileUpload")  element.send\_keys("C:myfile.txt") |

**34. How do you set the test case priority in TestNG?**

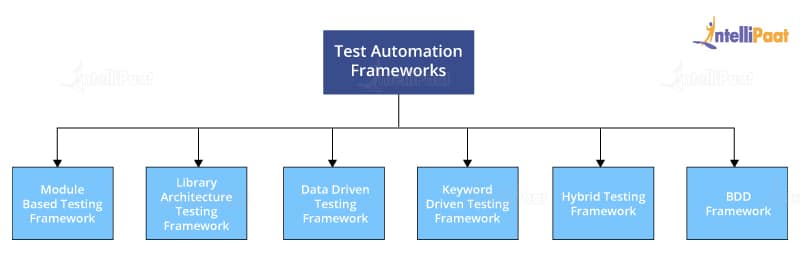
Setting Priority in TestNG:

**Test Execution Sequence:**

* Method1
* Method2
* Method3

**35. What are the different types of frameworks?**

The various categories of frameworks are outlined as follows:



* **Module-based testing framework:** This framework divides the entire application under test (AUT) into a number of logical and isolated modules. For each module, we create a separate and independent test script. Thus, when these test scripts are taken together, it builds a larger test script representing more than one module.
* **Library architecture testing framework:** Instead of dividing AUT into test scripts, with this framework, we segregate the application into functions or rather common functions that can be used by the other parts of the application as well. Thus, we create a common library constituting common functions for AUT. Therefore, these libraries can be called from the test scripts whenever required.
* **Data-driven testing framework:** The data-driven testing framework helps us segregate the test script logic and the test data from each other. It lets us store the test data into an external database. The data is conventionally stored in ‘key–value’ pairs. Keys can be used to access and populate the data within the test scripts.
* **Keyword-driven testing framework:** The keyword-driven testing framework is an extension to the data-driven testing framework in the sense that it not only segregates the test data from the scripts but also keeps a certain set of codes belonging to the test script in an external data file.
* **Hybrid testing framework:** A hybrid testing framework is a combination of more than one of the above-mentioned frameworks. The best thing about such a setup is that it leverages the benefits of all kinds of associated frameworks.
* **Behavior-driven development framework:** The behavior-driven development framework allows the automation of functional validations in an easily readable and understandable format for Business Analysts, Developers, Testers, etc.

**49. What is the use of the ‘ExpectedConditions’ class in Selenium?**

The ‘ExpectedConditions’ class in Selenium is a utility class that provides a set of predefined conditions that can be used to wait for a certain condition to be met before proceeding with the next step in a test. These conditions can be used in conjunction with the ‘WebDriverWait’ class to wait for elements to be visible, clickable, or to have a certain text, among other things.

For example,

'ExpectedConditions.visibilityOfElementLocated(By.id("elementId"))'

this can be used to wait for an element with a specific ID to be visible on the page. The ‘WebDriverWait’ class can then be used to wait for this condition to be met before interacting with the element. This helps to ensure that the element is present and visible before the test interacts with it, which can help prevent errors and improve the stability of the test.

**51. How can you perform mouse hover actions in Selenium?**

To perform a mouse hover action in Selenium, we can use the [Actions class](https://intellipaat.com/blog/action-class-in-selenium/). We first need to create an object of the Actions class and then use the moveToElement() method to move the mouse to the element we want to hover over. We can then perform the required action on the element using click() or other similar methods.

**52. What is Cucumber?**

[Cucumber](https://intellipaat.com/blog/cucumber-selenium/) is a BDD testing tool that enables developers to write test cases in a natural language format that can be understood by non-technical stakeholders. It supports multiple programming languages, including Java, Ruby, and Python, and can be integrated with various testing frameworks, including Selenium WebDriver.

It allows developers to write executable specifications in a readable format and automate the testing process. It enables developers to collaborate and communicate effectively with the team, including product owners, developers, and testers.

**53. How do you perform Cross browser testing using Selenium?**

Cross browser testing is a critical aspect of software testing that ensures the compatibility and functionality of a [web application](https://intellipaat.com/blog/how-to-test-web-application-using-selenium/) across different web browsers. Selenium, a popular open-source automation testing framework, provides a comprehensive solution for performing cross browser testing efficiently and effectively.

This process involves creating and executing test scripts using Selenium WebDriver to verify the consistency of the application’s behavior across multiple browsers.

To perform cross browser testing using Selenium, the following steps can be followed:

* **Test Environment Setup:**Start by setting up the necessary testing environment, including installing the required browsers (e.g., Chrome, Firefox, Safari, Internet Explorer) and the corresponding browser drivers (e.g., [ChromeDriver](https://intellipaat.com/blog/chromedriver-in-selenium/" \t "_blank), GeckoDriver).
* **Choose the Browsers:** Determine the target browsers based on the application’s target audience and market share. Consider both popular browsers and any specific browsers relevant to your user base.
* **Create Test Scripts:** Develop test scripts using Selenium WebDriver and the programming language of your choice (e.g., Java, Python, C#). The test scripts should encompass a wide range of test cases, covering various features and functionalities of the application.
* **Implement Cross Browser Testing:** Within the test scripts, configure the desired browser capabilities and instantiate the WebDriver accordingly. This allows you to run the same tests across different browsers seamlessly.
* **Execute the Test Scripts:** Run the test scripts on each targeted browser individually, using the appropriate browser driver. Selenium WebDriver will launch the browsers, navigate to the application, and execute the defined test cases.
* **Validate Results:** Compare the expected results with the actual results obtained from each browser. Identify any inconsistencies, discrepancies, or browser-specific issues and address them accordingly.
* **Reporting and Analysis:** Generate detailed test reports and analyze the results. This includes documenting any discrepancies or failures encountered during cross browser testing.

By performing cross browser testing using Selenium, you can identify and resolve any browser-specific issues, ensuring consistent functionality and a seamless user experience across different browsers. It is essential to regularly update the browser versions and drivers to stay compatible with the latest updates and changes in the browser landscape.

**54. Can you explain how to run Selenium tests in parallel?**

Running Selenium tests in parallel can improve the test execution time and reduce the overall test suite duration. The candidate should know how to set up a Selenium grid or use a cloud-based testing service to run tests in parallel on multiple machines or browsers.

**55. Can you explain how you would perform drag-and-drop operations in Selenium WebDriver?**

Performing drag-and-drop operations in Selenium WebDriver can be done using the ‘Actions’ class. The ‘Actions’ class provides several methods for simulating mouse and keyboard [actions](https://intellipaat.com/blog/action-class-in-selenium/), including drag-and-drop.

Here are the steps to perform drag-and-drop operations in Selenium WebDriver:

First, you need to instantiate an instance of the ‘Actions’ class by passing in the ‘WebDriver’ instance as a parameter.

Next, you need to locate the source and target elements on the page. These are the elements that you want to drag and drop.

Once you have located the source and target elements, you can use the ‘drag and drop (WebElement source, WebElement target)’ method of the ‘Actions’ class to simulate a drag-and-drop operation.

You can use the ‘build()’ method and ‘perform()’ method of the ‘Actions’ class to build the drag-and-drop action, and then perform it on the elements.

Here’s an example of how you would perform a drag-and-drop operation using Selenium WebDriver:

|  |  |
| --- | --- |
| 1  2  3  4 | WebElement sourceElement = driver.findElement(By.id("source"));  WebElement targetElement = driver.findElement(By.id("target"));  Actions actions = new Actions(driver);  actions.dragAndDrop(sourceElement, targetElement).build().perform(); |

**56. How do you handle dynamic web elements in Selenium?**

Dynamic web elements are those elements that change continuously on a web page. You can handle dynamic web elements using various methods such as:

* Using regular expressions in locators
* Using XPath functions like contains, starts-with, ends-with
* Using [CSS](https://intellipaat.com/blog/css-selector-in-selenium/) selectors

**57. Can you explain how you would automate the login functionality of a web application using Selenium WebDriver?**

To automate the login functionality of a web application using [Selenium WebDriver](https://intellipaat.com/blog/what-is-selenium-webdriver/), we first need to identify the username and password fields on the login page using locators such as ID, name, or CSS selector.

Once we have located the elements, we can use the sendKeys() method to enter the username and password into the respective fields. We can then use the click() method to click the login button to submit the login form.

**58. How do you handle dynamic dropdowns in Selenium WebDriver?**

Dynamic dropdowns are those where the options change based on the selection made in another dropdown. To handle dynamic dropdowns in Selenium WebDriver, we first need to locate the parent dropdown and select an option from it. Then, we need to wait for the child dropdown to populate with new options before locating and selecting an option from it.

**59. Can you explain how you would automate the testing of a file upload functionality using Selenium WebDriver?**

In order to automate the testing of file upload functionality with Selenium WebDriver, the initial step involves locating the file upload button on the web page using locators such as ID, name, or CSS selector.

Once the upload button is successfully located, the sendKeys() method can be utilized to input the file path of the desired file that needs to be uploaded. Alternatively, the AutoIT tool can be employed to handle the file upload dialog box, as Selenium WebDriver lacks the inherent capability to interact with it.

**60. Can you explain how you would automate the testing of a responsive web application using Selenium WebDriver?**

The browser window resize function may be used to enlarge the browser window to various sizes in order to evaluate how the program works on various screen sizes during automated testing of a responsive web application using Selenium WebDriver. The browser development tools may also be used to simulate other devices and evaluate how the application operates on them.

**63. How would you test a web application that uses a single-page application (SPA) framework like Angular or React?**

To ensure the proper functioning, user-friendliness, and optimal experience of a web application developed using a single-page application (SPA) framework like [Angular](https://intellipaat.com/blog/what-is-angular/) or React, a comprehensive testing approach is essential. The following explanation outlines the detailed methodology for testing a web application built with a SPA framework:

* Understand the SPA [Architecture](https://intellipaat.com/blog/selenium-architecture/): Familiarize yourself with the underlying architecture and concepts of the SPA framework, such as Angular or React. Gain knowledge about components, routing, state management, and asynchronous data handling.
* Identify Test Scenarios: Analyze the application’s functionality and identify key test scenarios. This includes testing navigation, form submissions, data fetching, dynamic rendering, and user interactions.
* Unit Testing: Begin with unit testing to test individual components, services, and utilities. Utilize testing frameworks specific to the SPA framework, such as Jasmine or Jest, to write and execute unit tests. Focus on testing component rendering, event handling, and data manipulation.
* Integration Testing: Perform integration testing to ensure the correct collaboration between components, services, and modules within the application. Test interactions between components, API integrations, and third-party library integrations. Utilize testing frameworks like Karma or Enzyme to execute integration tests.
* [End-to-End Testing:](https://intellipaat.com/blog/end-to-end-testing/) Conduct end-to-end (E2E) testing to simulate user interactions and test the entire application flow. Use tools like Selenium WebDriver or Cypress to write automated E2E tests. Test navigation, form submissions, user authentication, and other critical user flows.
* Accessibility Testing: Ensure the web application adheres to accessibility guidelines by performing accessibility testing. Verify that the application is accessible to users with disabilities and meets WCAG (Web Content Accessibility Guidelines) standards. Use tools like Axe or Lighthouse for automated accessibility testing.
* Compatibility Testing: Test the application across different browsers, devices, and operating systems. Verify that the SPA functions correctly and appears consistent across popular browsers like Chrome, Firefox, Safari, and Edge. Use browser testing tools or cloud-based testing platforms to perform cross-browser and cross-device testing.
* Security Testing: Assess the application for potential security vulnerabilities. Conduct security testing, including penetration testing, to identify and mitigate security risks. Validate inputs, test against common vulnerabilities, and ensure secure communication.
* Continuous Testing and Automation: Establish a resilient test automation framework to enable continuous testing practices. Automate repetitive tests, including unit tests, integration tests, and end-to-end (E2E) tests. Seamlessly integrate testing into the Continuous Integration/Continuous Deployment (CI/CD) pipeline to facilitate faster feedback loops and ensure the consistent delivery of high-quality software.

**66. How would you handle a scenario where a web application uses third-party APIs or services?**

Web applications often use third-party APIs or services, and testing them requires a different approach. A hybrid framework in Selenium combines both keyword-driven and data-driven approaches to test automation.

To implement it, you need to define a set of [keywords](https://intellipaat.com/blog/tutorial/c-tutorial/keywords-comments/) that represent the actions and assertions you want to perform on the application. Then, you create test scripts that use these keywords to interact with the application and verify its behavior.

**67. How would you handle a scenario where a web application has dynamic URLs?**

Dynamic URLs can be challenging to test, as they change frequently. When dealing with a web application that has dynamic URLs, it is important to implement a consistent and logical URL structure that is easy for both users and search engines to understand.

This can be achieved by using descriptive keywords in the URL and implementing URL rewriting techniques to ensure that the URLs remain clean and user-friendly. Additionally, using canonical tags and setting up proper redirects can help avoid issues with duplicate content and broken links.

**68. What is BDD (Behavior Driven Development)?**

Behavior Driven Development (BDD) is an Agile software development approach that focuses on defining the behavior of a software application in a way that is understandable by all stakeholders, including developers, testers, and business analysts.

It is based on the principles of Test Driven Development (TDD), but it emphasizes collaboration and communication between team members. It uses natural language descriptions of features, scenarios, and acceptance criteria to define the expected behavior of the software.

**69. How would you handle a scenario where a web application uses complex data structures like trees or graphs?**

When working on web applications that rely on complex data structures like trees or graphs, it’s essential to grasp their fundamental data model fully and how it impacts application functionality. To reach optimal performance levels, efficient algorithms, and [data structures](https://intellipaat.com/blog/what-is-data-structure/) for manipulating or traversing these structures must also be utilized in addition to rigorous testing procedures that protect data integrity while mitigating potential memory allocation or resource issues.

**71. How would you optimize your Selenium scripts for speed and performance?**

Optimizing Selenium scripts for speed and performance is key to decreasing execution time. There are various steps you can take to do so; one approach might be reducing page refreshes/waits in favor of conditional waits that trigger by specific page elements.

Headless browsers or cloud-based services can also help reduce test runtime by simultaneously running tests in parallel, shortening overall run times. Another technique to reduce data transfer between browser and server by employing efficient [selectors](https://intellipaat.com/blog/css-selector-in-selenium/) and eliminating unnecessary requests can help limit data transfers between browser and server; additionally, optimizing test environments to reduce unnecessary logging can boost Selenium script performance as well.

**72. How would you handle a scenario where a web application has multiple languages and requires localization testing?**

Localization testing is essential when your web app serves users from multiple languages. To effectively conduct localization testing for multiple-lingual web apps, several steps should be taken.

First and foremost, an application should support localization with language packs or resource files which can be tested to ensure accurate and consistent translations across languages. Furthermore, its user interface and layout should also be rigorously scrutinized against various character sets and formats of different languages to ensure compatibility.

Furthermore, testing should cover scenarios like date/time formats, currency symbols, and regional variations.

**76. How do you ensure the stability and reliability of your Selenium tests?**

Selenium tests are susceptible to fragility and potential failure caused by factors like varying browser versions, network disruptions, or modifications in the application being tested. To guarantee the steadfastness and dependability of Selenium tests, it is essential to employ resilient and sustainable test code capable of accommodating dynamic changes in the application. This encompasses utilizing efficient selectors, modularizing the test code, and minimizing the reliance on hardcoded values. By adhering to these practices, the stability and reliability of Selenium tests can be enhanced, ensuring smoother test execution even in the face of evolving conditions or alterations in the application under examination.

**78. How do you implement a Hybrid Framework in Selenium?**

A Hybrid Framework is a combination of different design patterns and approaches, such as Data-Driven, Keyword-Driven, or Behavior-Driven Testing, to improve the modularity, reusability, and maintainability of Selenium tests.

To implement a Hybrid Framework in Selenium, you would first identify the different types of tests required, such as functional, regression, or performance testing.

Then, you would create modular test scripts using a combination of data-driven and keyword-driven approaches.

These scripts can be managed through a test management tool and executed using a test runner. Additionally, implementing a reporting mechanism can provide valuable insights into the test results and aid in debugging and analysis.

**80. How do you integrate Selenium with other testing tools, such as Jenkins or TestNG?**

Selenium can be integrated with other testing tools and frameworks to improve the automation workflow and generate useful reports and metrics. Integrating Selenium with other testing tools such as [Jenkins](https://intellipaat.com/blog/jenkins-docker/) or TestNG involves configuring the testing tools to work with Selenium.

This can be achieved by adding Selenium libraries to the project and setting up test configurations to run Selenium tests. Additionally, integrating with a continuous integration tool like Jenkins allows for automated testing and reporting, while TestNG provides advanced testing features such as grouping and parallel execution.

**81. How do you implement performance testing in Selenium?**

Performance testing involves measuring the web application’s response time, throughput, and scalability under different loads and stress levels. To implement performance testing in Selenium, you would use load testing tools such as JMeter or LoadRunner to simulate heavy user traffic and measure the performance of the web application.

Selenium [scripts](https://intellipaat.com/blog/tutorial/cognos-tm1/advanced-techniques-for-ti-scripting/) can be integrated with these tools to simulate real user behavior and generate load on the application. Additionally, using performance profiling tools can help identify bottlenecks and areas for optimization in the application

**83. Can you explain how to handle and automate complex web elements, such as iframes, pop-ups, or drag-and-drop elements?**

To handle and automate complex web elements such as iframes, pop-ups, or drag-and-drop elements in Selenium, first identify the element using appropriate locators such as ID or XPath.

For iframes, switch to the frame using the driver’s switchTo() method. For pop-ups, use the Alert class to handle the pop-up window. For drag-and-drop elements, make use of the Actions class to perform drag-and-drop operations.

**84. What is a DesiredCapabilities class in Selenium?**

The DesiredCapabilities class is a part of the Selenium WebDriver API and is used to configure the capabilities of a WebDriver instance. It is essentially a set of key-value pairs that specify the various properties of the browser or device that the WebDriver will use to run the tests.

Some examples of the properties that can be set using DesiredCapabilities include; browser type, version, platform, proxy settings, and browser language.

It is particularly useful for cross-browser testing, where you may need to run the same tests on different browsers or platforms. By setting the capabilities using this class, you can ensure that the WebDriver will use the correct settings for each browser/platform, and your tests will run consistently across all environments.

**86. What is Gherkin?**

Gherkin is a simple, natural language syntax used to describe the behavior of a software application in a way that is understandable by non-technical stakeholders. It is used in conjunction with [Cucumber](https://intellipaat.com/blog/cucumber-selenium/) to define the acceptance criteria and scenarios for a software feature.

It uses keywords, such as Given, When, and Then, to describe the steps of a scenario in a clear and concise manner. Gherkin allows developers to write executable specifications that can be easily understood and reviewed by the team.

**92. What is Selenium WebDriver?**

Selenium WebDriver is the most popular component of the Selenium framework. It is a powerful tool that allows you to automate web browsers. With Selenium WebDriver, you can write automation scripts in various programming languages and execute them on different browsers like Chrome, Firefox, Safari, and more.

**93. How many types of WebDriver APIs are available in Selenium?**

The following is a list of WebDriver APIs:

* AndroidDriver
* ChromeDriver
* EventFiringWebDriver
* FirefoxDriver
* HTMLUnitDriver
* InternetExplorerDriver
* iPhoneDriver
* iPhoneSimulatorDriver
* RemoteWebDriver

**94. How do you launch the web browser using WebDriver?**

The following syntax can be used to launch the browser corresponding to the system’s operating system:

WebDriver driver = new FirefoxDriver();

Or

WebDriver driver = new InternetExplorerDriver();

Or

WebDriver driver = new ChromeDriver();

**95. What are the different types of waits available in WebDriver?**

There are two types of waits available in WebDriver: Explicit and Implicit WaitExplicit and Implicit Wait

* Implicit wait: These waits are used to provide a default waiting time (say, 30 seconds) between the consecutive test steps across the entire test script. Hence, the subsequent test step would only be executed when the 30 seconds are over after executing the previous test step.
* Explicit wait: These waits are used to halt the execution until a particular condition is met or the maximum time has elapsed.

Explicit waits are instantiated for a particular instance only, whereas implicit waits are not.

**96. How do you handle a frame in WebDriver?**

An iframe (an acronym for ‘inline frame’) is used to insert another document within the current HTML document. Selecting iframe by ID:

driver.switchTo().frame(“ID of the frame“);

Locating iframe using the tagName:

driver.switchTo().frame(driver.findElements(By.tagName(“iframe”).get(0));

Locating iframe using index:

* frame(index)

driver.switchTo().frame(0);

* frame(“Name of the Frame”)

driver.switchTo().frame(“name of the frame”);

* frame(WebElement element)
* Select Parent Window

driver.switchTo().defaultContent();

**97. For the database testing in Selenium WebDriver, what API is required?**

For the [database testing](https://intellipaat.com/blog/interview-question/database-testing-interview-questions/) in Selenium WebDriver, we need the JDBC (Java Database Connectivity) API. It allows us to execute SQL statements.

**98. Can WebDriver test mobile applications?**

No, WebDriver is a testing tool used for web-based applications. So, we cannot test mobile applications with Selenium WebDriver.

**99. How do you handle synchronization issues in Selenium WebDriver?**

Synchronization issues can occur when the script moves too quickly, and the web page is not ready for the next step. To handle synchronization issues in Selenium WebDriver, we can use different types of waits like Implicit Wait, Explicit Wait, or [Fluent Wait](https://intellipaat.com/blog/fluent-wait-in-selenium/). We can also use the Thread.sleep() method to pause the execution of the script for a specified amount of time. However, this method should be used sparingly as it can make the script slower and less efficient.

**100. How do you find broken links in Selenium WebDriver?**

We can detect whether the given links are broken or not by using the following process:

1. First, accumulate all the links present on a web page using the anchor tag. For each tag, use the attribute ‘href’ value to obtain the hyperlink.
2. Send HTTP requests for each link and verify the HTTP response code
3. Based on the HTTP response code, determine if the link is valid or broken. Then, use the driver.get() method to navigate to a URL, which will respond with a status of 200 – OK (200 – OK indicates that the link is working). If we get any other status, then it indicates that the link is broken
4. Repeat the same process for all the links captured

**103. Which year was selenium created?**

1. **2004**
2. **2005**
3. **2006**
4. **2001**

2004

**104. Who was the creator of selenium?**

1. **Dan Cuellar**
2. **Jason Huggins**
3. **Rossmanith Gmbh**

Jason Huggins

**105. Which of the following browsers supports selenium?**

1. **Google Chrome**
2. **Safari**
3. **Mozilla Firefox**
4. **Internet Explorer**
5. **All of the above**

All of the above

**106. What do you mean by open-source software?**

1. **Open-source software is software that is circulated across the world with its source code which also means it is available for use, and modification.**
2. **Open-source software is software that is easy to use and where the code is stored in a public repository also it comes with a distributed license.**
3. **All of the above**

All of the above

**107. Which of the following is not the alternative to selenium?**

1. **Cucumber**
2. **Cypress**
3. **Puppeteer**
4. **Mocha**

Mocha

**111. You are testing an e-commerce website, and during checkout, the 'Place Order' button is not responding consistently. How would you approach debugging and resolving this issue using Selenium?**

There are several approach for debugging and resolving the issue using Selenium are:

1. Optimize Element Identification: Ensure ‘Place Order’ button has a unique and SEO-friendly identifier, such as a relevant ID or class.
2. Efficient Waiting: Implement precise waits with WebDriverWait to enhance user experience and optimize loading times.
3. Robust Error Handling: Incorporate try-catch blocks with clear error messages for better [SEO](https://intellipaat.com/blog/seo-tutorial/) diagnostics.
4. Detailed Logging for Analysis: Use detailed logging statements to aid developers in analyzing and improving the checkout process.
5. SEO-friendly Naming: Name identifiers and variables descriptively, enhancing code readability and SEO friendliness.
6. Browser DevTools Insights: Leverage browser developer tools for a thorough analysis of the button element, ensuring a seamless user experience.
7. Stay Updated with Latest Technologies: Regularly update Selenium and browser drivers to align with the latest SEO-friendly practices and ensure compatibility.
8. Prioritize User Focus: Confirm browser window focus for improved user interactions and SEO performance.

**112. The application you are testing involves uploading files. Explain how you would automate the testing of file uploads using Selenium WebDriver.**

We can automate the testing of file uploads using Selenium WebDriver by these mentions approach:

1. Optimize File Input Element Identification: Choose SEO-friendly locators (like ID, XPath, or CSS) for the file input element.
2. Streamlined Interaction with `send\_keys()`: Utilize `send\_keys()` method for a seamless and SEO-friendly simulation of file selection, providing the file path.
3. Efficient Wait Strategies for Upload Completion: Implement explicit waits to optimize user experience and ensure timely file upload completion.
4. SEO-friendly Verification of Upload Success: Verify upload success using SEO-friendly messages or elements indicating a successful upload.
5. Graceful Handling of Popups (if any): Ensure a smooth transition by gracefully switching to and handling confirmation popups, if presented.
6. Detailed Logging and SEO-friendly Reporting: Incorporate detailed logging statements for transparent progress tracking and integrate with SEO-friendly reporting tools for comprehensive logs.

**113. While running your Selenium tests, you encounter intermittent failures due to dynamic elements. Describe strategies to handle dynamic elements effectively and maintain test stability.**

Strategies to handle dynamic elements effectively and maintain test stability are:

1. Strategic Element Waits: Implement SEO-friendly explicit waits using WebDriverWait to ensure stability when dealing with dynamic elements.
2. Dynamic XPath/CSS Selectors: Use SEO-friendly dynamic XPath or CSS selectors, focusing on relative paths to enhance test resilience against dynamic content.
3. Stable Parent Elements: Identify SEO-friendly stable parent elements and locate dynamic elements relative to them for increased resilience.
4. Retry Mechanism for Robustness: Enhance test robustness with an SEO-friendly retry mechanism, intelligently rerunning steps in case of intermittent dynamic element issues.
5. Smart Waiting Strategies: Implement SEO-friendly smart waiting strategies, considering combinations of elements, to ensure tests are adaptive to dynamic changes.
6. Page Object Model (POM): Leverage the SEO-friendly Page Object Model for structured organization of locators and methods, promoting readability and maintainability.
7. Dynamic Locator Updates: Dynamically update SEO-friendly locators during runtime to adapt to evolving application changes.
8. Capture SEO-friendly Screenshots and Logs: Capture SEO-friendly screenshots and detailed logs during failures to facilitate debugging and SEO-friendly issue resolution.
9. Regular SEO-friendly Maintenance: Regularly review and update test scripts with SEO-friendly practices to accommodate application changes, ensuring ongoing effectiveness against dynamic elements.

**Selenium Tester Roles and Responsibilities**

|  |  |
| --- | --- |
| Job Role | Description |
| Test Automation Engineer | Develop and implement automated test scripts using Selenium. Execute automated test suites and analyze the result. |
| Quality Assurance (QA) Engineer | Design and execute manual test cases. Report and tackle the defects using a big tracking system. |
| Test Lead/Manager | Oversee the entire testing process and ensure that it aligns with the project. Develop and implement test plans and strategies. Assign tasks to the testing team. |
| Automation Architect | Design and implement the overall test automation framework. Evaluate and select appropriate tools and technologies for testing. |
| Test Analyst | Analyze requirements, identify test scenarios, create and execute manual test cases. |

**Role:** Senior Quality Engineer Level 1- Automation Testing Selenium

Responsibilities

* 1. Maintain software quality through an automated QE strategy.
  2. Collaborate with the project and business to develop detailed automated scripts.
  3. Ability to handle a team of  2-5 people.
  4. Skill to enforce defects or other processes in the team.
  5. Ability to handle low and medium complexity applications and have used at least one of the estimation techniques.

Skill Required:

* 1. Experience with QE for distributed and highly scalable systems.
  2. Good understanding of OOPS concepts and strong programming skills in Java, Groovy, or JavaScript.
  3. Familiarity with the process of test automation tool selection and test approach.
  4. Experience in designing and developing automation frameworks and the creation of scripts using best industry practices, such as the Page object model.
  5. Understanding of all the aspects of Quality Engineering.
  6. Understanding of SOAP and REST principles.
  7. Experience in creating test pipeline CI/CD.