**Q #1) What is Automation Testing?**

Automation testing or Test Automation is a process of automating the manual process to test the application/system under test. Automation testing involves the use of a separate testing tool which lets you create test scripts which can be executed repeatedly and doesn’t require any manual intervention.

**Q #2) What are the benefits of Automation Testing?**

Benefits of Automation testing are:

1. Supports execution of repeated test cases
2. Aids in testing a large test matrix
3. Enables parallel execution
4. Encourages unattended execution
5. Improves accuracy thereby reducing human-generated errors
6. Saves time and money

**Q #3) Why should Selenium be selected as a test tool?**

Selenium

1. is a free and open source
2. have a large user base and helping communities
3. have cross Browser compatibility (Firefox, Chrome, Internet Explorer, Safari etc.)
4. have great platform compatibility (Windows, Mac OS, Linux etc.)
5. supports multiple programming languages (Java, C#, Ruby, Python, Pearl etc.)
6. has fresh and regular repository developments
7. supports distributed testing

**Q #4) What is Selenium? What are the different Selenium components?**

Selenium is one of the most popular automated testing suites. Selenium is designed in a way to support and encourage automation testing of functional aspects of web-based applications and a wide range of browsers and platforms. Due to its existence in the open source community, it has become one of the most accepted tools amongst the testing professionals.

Selenium is not just a single tool or a utility, rather a package of several testing tools and for the same reason, it is referred to as a Suite. Each of these tools is designed to cater different testing and test environment requirements.

The suite package constitutes the following sets of tools:

* [**Selenium Integrated Development Environment (IDE)**](https://www.softwaretestinghelp.com/selenium-ide-download-and-installation-selenium-tutorial-2/) – Selenium IDE is a record and playback tool. It is distributed as a Firefox Plugin.
* **Selenium Remote Control (RC)** – Selenium RC is a server that allows a user to create test scripts in the desired programming language. It also allows executing test scripts within the large spectrum of browsers.
* [**Selenium WebDriver**](https://www.softwaretestinghelp.com/selenium-webdriver-selenium-tutorial-8/) – WebDriver is a different tool altogether that has various advantages over Selenium RC. WebDriver directly communicates with the web browser and uses its native compatibility to automate.
* [**Selenium Grid**](https://www.softwaretestinghelp.com/selenium-grid-selenium-tutorial-29/) – Selenium Grid is used to distribute your test execution on multiple platforms and environments concurrently.

**Q #5) What are the testing types that can be supported by Selenium?**

Selenium supports the following types of testing:

1. Functional Testing
2. Regression Testing

**Q #6) What are the limitations of Selenium?**

Following are the limitations of Selenium:

* Selenium supports testing of only web-based applications
* Mobile applications cannot be tested using Selenium
* Captcha and Barcode readers cannot be tested using Selenium
* Reports can only be generated using third-party tools like TestNG or JUnit.
* As Selenium is a free tool, thus there is no ready vendor support through the user can find numerous helping communities.

**Q #8) When should I use Selenium IDE?**

Selenium IDE is the simplest and easiest of all the tools within the Selenium Package. Its record and playback feature makes it exceptionally easy to learn with minimal acquaintances to any programming language. Selenium IDE is an ideal tool for a naïve user.

**Q #9) What is Selenese?**

Selenese is the language which is used to write test scripts in Selenium IDE.

**Q #10)** **What are the different types of locators in Selenium?**

The locator can be termed as an address that identifies a web element uniquely within the webpage. Thus, to identify web elements accurately and precisely we have [different types of locators in Selenium](https://www.softwaretestinghelp.com/using-selenium-xpath-and-other-locators-selenium-tutorial-5/):

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

**Q #11)** **What is the difference between assert and verify commands?**

**Assert:** Assert command checks whether the given condition is true or false. Let’s say we assert whether the given element is present on the web page or not. If the condition is true then the program control will execute the next test step but if the condition is false, the execution would stop and no further test would be executed.

**Verify:** Verify command also checks whether the given condition is true or false. Irrespective of the condition being true or false, the program execution doesn’t halt i.e. any failure during verification would not stop the execution and all the test steps would be executed.

**Q #12) What is an XPath?**

[XPath](https://www.softwaretestinghelp.com/using-selenium-xpath-and-other-locators-selenium-tutorial-5/) is 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 which is very much similar to HTML tags. Both being markup languages and since they fall under the same umbrella, XPath can be used to locate HTML elements.

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

**Q #13) What is the difference between “/” and “//” in Xpath?**

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

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

**Q #14) What is Same origin policy and how it can be handled?**

The problem of same origin policy disallows to access the DOM of a document from an origin that is different from the origin we are trying to access the document.

Origin is a sequential combination of scheme, host, and port of the URL. For example, for a URL https://www.softwaretestinghelp.com/resources/, the origin is a combination of http, softwaretestinghelp.com, 80 correspondingly.

Thus the Selenium Core (JavaScript Program) cannot access the elements from an origin that is different from where it was launched. For Example, if I have launched the JavaScript Program from “https://www.softwaretestinghelp.com”, then I would be able to access the pages within the same domain such as “https://www.softwaretestinghelp.com/resources” or “https://www.softwaretestinghelp.com/istqb-free-updates/”. The other domains like google.com, seleniumhq.org would no more be accessible.

So, In order to handle the same origin policy, Selenium Remote Control was introduced.

**Q #15)** **When should I use Selenium Grid?**

Selenium Grid can be used to execute same or different test scripts on multiple platforms and browsers concurrently so as to achieve distributed test execution, testing under different environments and saving execution time remarkably.

**Q #16) What do we mean by Selenium 1 and Selenium 2?**

Selenium RC and WebDriver, in a combination, are popularly known as Selenium 2. Selenium RC alone is also referred to as Selenium 1.

**Q #17) Which is the latest Selenium tool?**

WebDriver

**Q #18) How do I launch the browser using WebDriver?**

The following syntax can be used to launch Browser:  
*WebDriver driver =* ***new*** *FirefoxDriver();*  
*WebDriver driver =* ***new*** *ChromeDriver();*  
*WebDriver driver =* ***new*** *InternetExplorerDriver();*

**Q #19) What are the different types of Drivers available in WebDriver?**

The different drivers available in WebDriver are:

* FirefoxDriver
* InternetExplorerDriver
* ChromeDriver
* SafariDriver
* OperaDriver
* AndroidDriver
* IPhoneDriver
* HtmlUnitDriver

**Q #20) What are the different types of waits available in WebDriver?**

There are two [types of waits available in WebDriver](https://www.softwaretestinghelp.com/selenium-webdriver-waits-selenium-tutorial-15/):

1. Implicit Wait
2. Explicit Wait

**Implicit Wait:** Implicit waits are used to provide a default waiting time (say 30 seconds) between each consecutive test step/command across the entire test script. Thus, the subsequent test step would only execute when the 30 seconds have elapsed after executing the previous test step/command.

**Explicit Wait:** Explicit waits are used to halt the execution till the time a particular condition is met or the maximum time has elapsed. Unlike Implicit waits, explicit waits are applied for a particular instance only.

**Q #21)** **How to type in a textbox using Selenium?**

The user can use sendKeys(“String to be entered”) to enter the string in the textbox.

**Syntax:**  
*WebElement username = drv.findElement(By.id(“Email”));*  
*// entering username*  
*username.sendKeys(“sth”);*

**Q #22)** **How can you find if an element in displayed on the screen?**

WebDriver facilitates the user with the following methods to check the visibility of the web elements. These web elements can be buttons, drop boxes, checkboxes, radio buttons, labels etc.

1. isDisplayed()
2. isSelected()
3. isEnabled()

**Syntax:**

**isDisplayed():**  
***boolean*** *buttonPresence = driver.findElement(By.id(“gbqfba”)).isDisplayed();*

**isSelected():**  
***boolean*** *buttonSelected = driver.findElement(By.id(“gbqfba”)).isSelected();*

**isEnabled():**  
***boolean*** *searchIconEnabled = driver.findElement(By.id(“gbqfb”)).isEnabled();*

**Q #23)** **How can we get a text of a web element?**

Get command is used to retrieve the inner text of the specified web element. The command doesn’t require any parameter but returns a string value. It is also one of the extensively used commands for verification of messages, labels, errors etc displayed on the web pages.

**Syntax:**  
*String Text = driver.findElement(By.id(“Text”)).getText();*

**Q #24) How to select value in a dropdown?**

The value in the dropdown can be selected using WebDriver’s Select class.

**Syntax:**

**selectByValue:**  
*Select selectByValue =* ***new*** *Select(driver.findElement(By.id(“SelectID\_One”)));*  
*selectByValue.selectByValue(“greenvalue”);*

**selectByVisibleText:**  
*Select selectByVisibleText =* ***new*** *Select (driver.findElement(By.id(“SelectID\_Two”)));*  
*selectByVisibleText.selectByVisibleText(“Lime”);*

**selectByIndex:**  
*Select selectByIndex =* ***new*** *Select(driver.findElement(By.id(“SelectID\_Three”)));*  
*selectByIndex.selectByIndex(2);*

**Q #25) What are the different types of navigation commands?**

Following are the [navigation commands](https://www.softwaretestinghelp.com/selenium-webdriver-waits-selenium-tutorial-15/):  
**navigate().back()** – The above command requires no parameters and takes back the user to the previous webpage in the web browser’s history.

**Sample code:**  
*driver.navigate().back();*

**navigate().forward()** – This command lets the user to navigate to the next web page with reference to the browser’s history.

**Sample code:**  
*driver.navigate().forward();*

**navigate().refresh()** – This command lets the user to refresh the current web page there by reloading all the web elements.

**Sample code:**  
*driver.navigate().refresh();*

**navigate().to()** – This command lets the user to launch a new web browser window and navigate to the specified URL.

**Sample code:**  
*driver.navigate().to(“https://google.com”);*

**Q #26) How to click on a hyper link using linkText?**

*driver.findElement(By.linkText(“Google”)).click();*

The command finds the element using link text and then click on that element and thus the user would be re-directed to the corresponding page.

The above-mentioned link can also be accessed by using the following command.

*driver.findElement(By.partialLinkText(“Goo”)).click();*

The above command finds the element based on the substring of the link provided in the parenthesis and thus partialLinkText() finds the web element with the specified substring and then clicks on it.

**Q #27)** **How to** [**handle frame in WebDriver**](https://www.softwaretestinghelp.com/selenium-tutorial-18/)**?**

An inline frame acronym as iframe is used to insert another document within the current HTML document or simply a web page into a web page by enabling nesting.

**Select iframe by id**  
*driver.switchTo().frame(“ID of the frame“);*

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

**Locating iframe using index**

**frame(index)**  
*driver.switchTo().frame(0);*

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

**frame(WebElement element)**  
**Select Parent Window**  
*driver.switchTo().defaultContent();*

**Q #28) When do we use findElement() and findElements()?**

**findElement():** findElement() is used to find the first element in the current web page matching to the specified locator value. Take a note that only first matching element would be fetched.

**Syntax:**

*WebElement element = driver.findElements(By.xpath(“//div[@id='example']//ul//li”));*  
**findElements():** findElements() is used to find all the elements in the current web page matching to the specified locator value. Take a note that all the matching elements would be fetched and stored in the list of WebElements.

**Syntax:**  
*List <WebElement> elementList = driver.findElements(By.xpath(“//div[@id='example']//ul//li”));*

**Q #29)** **How to find more than one web element in the list?**

At times, we may come across elements of the same type like multiple hyperlinks, images etc arranged in an ordered or unordered list. Thus, it makes absolute sense to deal with such elements by a single piece of code and this can be done using WebElement List.

**Sample Code**

|  |
| --- |
| // Storing the list  List <WebElement> elementList = driver.findElements(By.xpath("//div[@id='example']//ul//li"));  // Fetching the size of the list  int listSize = elementList.size();  for (int i=0; i<listSize; i++)  {  // Clicking on each service provider link  serviceProviderLinks.get(i).click();  // Navigating back to the previous page that stores link to service providers  driver.navigate().back();  } |

**Q #30) What is the difference between driver.close() and driver.quit command?**

**close()**: WebDriver’s close() method closes the web browser window that the user is currently working on or we can also say the window that is being currently accessed by the WebDriver. The command neither requires any parameter nor does it return any value.

**quit()**: Unlike close() method, quit() method closes down all the windows that the program has opened. Same as close() method, the command neither requires any parameter nor does is return any value.

**Q #31) Can Selenium handle windows based pop up?**

Selenium is an automation testing tool which supports only web application testing. Therefore, windows pop up cannot be handled using Selenium.

**Q #32) How can we handle web-based pop-up?**

WebDriver offers the users a very efficient way to [handle these pop-ups using Alert interface](https://www.softwaretestinghelp.com/handle-alerts-popups-selenium-webdriver-selenium-tutorial-16/). There are the four methods that we would be using along with the Alert interface.

* void dismiss() – The dismiss() method clicks on the “Cancel” button as soon as the pop-up window appears.
* void accept() – The accept() method clicks on the “Ok” button as soon as the pop-up window appears.
* String getText() – The getText() method returns the text displayed on the alert box.
* void sendKeys(String stringToSend) – The sendKeys() method enters the specified string pattern into the alert box.

**Syntax:**  
*// accepting javascript alert*   
*Alert alert = driver.switchTo().alert();*  
*alert.accept();*

**Q #33) How can we handle windows based pop up?**

Selenium is an automation testing tool which supports only web application testing, that means, it doesn’t support testing of windows based applications. However Selenium alone can’t help the situation but along with some third-party intervention, this problem can be overcome. There are several third-party tools available for handling window based pop-ups along with the selenium like AutoIT, Robot class etc.

**Q #34) How to assert the title of the web page?**

*//verify the title of the web page*  
*assertTrue(“The title of the window is incorrect.”,driver.getTitle().equals(“Title of the page”));*

**Q #35) How to mouse hover on a web element using WebDriver?**

WebDriver offers a wide range of interaction utilities that the user can exploit to automate mouse and keyboard events. Action Interface is one such utility which simulates the single user interactions.

Thus, In the following scenario, we have used Action Interface to mouse hover on a drop down which then opens a list of options.

**Sample Code:**

|  |
| --- |
| // Instantiating Action Interface  Actions actions=new Actions(driver);  // howering on the dropdown  actions.moveToElement(driver.findElement(By.id("id of the dropdown"))).perform();  // Clicking on one of the items in the list options  WebElement subLinkOption=driver.findElement(By.id("id of the sub link"));  subLinkOption.click(); |

**Q #36) How to retrieve CSS properties of an element?**

The values of the css properties can be retrieved using a get() method:

**Syntax:**  
*driver.findElement(By.id(“id“)).getCssValue(“name of css attribute”);*  
*driver.findElement(By.id(“id“)).getCssValue(“font-size”);*

**Q #37) How to capture screenshot in WebDriver?**

|  |
| --- |
| import org.junit.After;  import org.junit.Before;  import org.junit.Test;  import java.io.File;  import java.io.IOException;  import org.apache.commons.io.FileUtils;  import org.openqa.selenium.OutputType;  import org.openqa.selenium.TakesScreenshot;  import org.openqa.selenium.WebDriver;  import org.openqa.selenium.firefox.FirefoxDriver;    public class CaptureScreenshot {  WebDriver driver;  @Before  public void setUp() throws Exception {  driver = new FirefoxDriver();  driver.get("https://google.com");  }  @After  public void tearDown() throws Exception {  driver.quit();  }    @Test  public void test() throws IOException {  // Code to capture the screenshot  File scrFile = ((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);  // Code to copy the screenshot in the desired location  FileUtils.copyFile(scrFile, new File("C:\\CaptureScreenshot\\google.jpg"))  }  } |

**Q #38) What is Junit?**

[Junit](https://www.softwaretestinghelp.com/selenium-junit-framework-selenium-tutorial-11/) is a unit testing framework introduced by Apache. Junit is based on Java.

**Q #39) What are Junit annotations?**

Following are the JUnit Annotations:

* **@Test:** Annotation lets the system know that the method annotated as @Test is a test method. There can be multiple test methods in a single test script.
* **@Before:** Method annotated as @Before lets the system know that this method shall be executed every time before each of the test methods.
* **@After:** Method annotated as @After lets the system know that this method shall be executed every time after each of the test method.
* **@BeforeClass:** Method annotated as @BeforeClass lets the system know that this method shall be executed once before any of the test methods.
* **@AfterClass:** Method annotated as @AfterClass lets the system know that this method shall be executed once after any of the test methods.
* **@Ignore:** Method annotated as @Ignore lets the system know that this method shall not be executed.

**Q #40)** **What is TestNG and how is it better than Junit?**

[TestNG](https://www.softwaretestinghelp.com/testng-framework-selenium-tutorial-12/) is an advanced framework designed in a way to leverage the benefits by both the developers and testers. With the commencement of the frameworks, JUnit gained enormous popularity across the Java applications, Java developers and Java testers with remarkably increasing the code quality. Despite being easy to use and straightforward, JUnit has its own limitations which give rise to the need of bringing TestNG into the picture. TestNG is an open source framework which is distributed under the Apache Software License and is readily available for download.

TestNG with WebDriver provides an efficient and effective test result format that can, in turn, be shared with the stakeholders to have a glimpse on the product’s/application’s health thereby eliminating the drawback of WebDriver’s incapability to generate test reports. TestNG has an inbuilt exception handling mechanism which lets the program to run without terminating unexpectedly.

There are various advantages that make TestNG superior to JUnit. Some of them are:

* Added advance and easy annotations
* Execution patterns can set
* Concurrent execution of test scripts
* Test case dependencies can be set

**Q #41)** **How to set test case priority in TestNG?**

**Setting Priority in TestNG**

**Code Snippet**

|  |
| --- |
| package TestNG;  import org.testng.annotations.\*;  public class SettingPriority {  @Test(priority=0)  public void method1() {  }  @Test(priority=1)  public void method2() {  }  @Test(priority=2)  public void method3() {  }  } |

**Test Execution Sequence:**

1. Method1
2. Method2
3. Method3

**Q #42) What is a framework?**

The framework is a constructive blend of various guidelines, coding standards, concepts, processes, practices, project hierarchies, modularity, reporting mechanism, test data injections etc. to pillar automation testing.

**Q #43)** **What are the advantages of the Automation framework?**

**The advantage of** [**Test Automation framework**](https://www.softwaretestinghelp.com/test-automation-frameworks-selenium-tutorial-20/)

* Reusability of code
* Maximum coverage
* Recovery scenario
* Low-cost maintenance
* Minimal manual intervention
* Easy Reporting

**Q #44) What are the different types of frameworks?**

**Below are the different types of frameworks:**

1. **Module Based Testing Framework:** The framework divides the entire “Application Under Test” into the number of logical and isolated modules. For each module, we create a separate and independent test script. Thus, when these test scripts have taken together builds a larger test script representing more than one module.
2. **Library Architecture Testing Framework:** The basic fundamental behind the framework is to determine the common steps and group them into functions under a library and call those functions in the test scripts whenever required.
3. Data Driven Testing Framework: Data Driven Testing Framework helps the user segregate the test script logic and the test data from each other. It lets the user store the test data into an external database. The data is conventionally stored in “Key-Value” pairs. Thus, the key can be used to access and populate the data within the test scripts.
4. **Keyword Driven Testing Framework:** The Keyword Driven testing framework is an extension to Data-driven Testing Framework in a sense that it not only segregates the test data from the scripts, it also keeps the certain set of code belonging to the test script into an external data file.
5. **Hybrid Testing Framework:** Hybrid Testing Framework is a combination of more than one above mentioned frameworks. The best thing about such a setup is that it leverages the benefits of all kinds of associated frameworks.
6. **Behavior Driven Development Framework:** Behavior Driven Development framework allows automation of functional validations in an easily readable and understandable format to Business Analysts, Developers, Testers, etc.

**Q #45) How can I read test data from excels?**

Test data can efficiently be read from excel using JXL or POI API. [See detailed tutorial here](https://www.softwaretestinghelp.com/selenium-framework-design-selenium-tutorial-21/).

**Q #46) What is the difference between POI and jxl jar?**

| **#** | **JXL jar** | **POI jar** |
| --- | --- | --- |
| 1 | JXL supports “.xls” format i.e. binary based format. JXL doesn’t support Excel 2007 and “.xlsx” format i.e. XML based format | POI jar supports all of these formats |
| 2 | JXL API was last updated in the year 2009 | POI is regularly updated and released |
| 3 | The JXL documentation is not as comprehensive as that of POI | POI has a well prepared and highly comprehensive documentation |
| 4 | JXL API doesn’t support rich text formatting | POI API supports rich text formatting |
| 5 | JXL API is faster than POI API | POI API is slower than JXL API |

**Q #47)** **What is the difference between Selenium and QTP?**

| **Feature** | **Selenium** | **Quick Test Professional (QTP)** |
| --- | --- | --- |
| Browser Compatibility | Selenium supports almost all the popular browsers like Firefox, Chrome, Safari, Internet Explorer, Opera etc | QTP supports Internet Explorer, Firefox and Chrome. QTP only supports Windows Operating System |
| Distribution | Selenium is distributed as an open source tool and is freely available | QTP is distributed as a licensed tool and is commercialized |
| Application under Test | Selenium supports testing of only web based applications | QTP supports testing of both the web based application and windows based application |
| Object Repository | Object Repository needs to be created as a separate entity | QTP automatically creates and maintains Object Repository |
| Language Support | Selenium supports multiple programming languages like Java, C#, Ruby, Python, Perl etc | QTP supports only VB Script |
| Vendor Support | As Selenium is a free tool, user would not get the vendor’s support in troubleshooting issues | Users can easily get the vendor’s support in case of any issue |

**Q #48) Can WebDriver test Mobile applications?**

WebDriver cannot test Mobile applications. WebDriver is a web-based testing tool, therefore applications on the mobile browsers can be tested.

**Q #49) Can captcha be automated?**

No, captcha and barcode reader cannot be automated.

**Q #50) What is Object Repository? How can we create an Object Repository in Selenium?**

Object Repository is a term used to refer to the collection of web elements belonging to Application Under Test (AUT) along with their locator values. Thus, whenever the element is required within the script, the locator value can be populated from the Object Repository. Object Repository is used to store locators in a centralized location instead of hardcoding them within the scripts.

In Selenium, objects can be stored in an excel sheet which can be populated inside the script whenever required.

Resource:-- <https://www.softwaretestinghelp.com/selenium-interview-questions-answers/>

**Question: Can you explain the difference between assert and verify commands in Selenium?**

**Answer**: Both assert and verify commands are responsible for checking whether the given condition is true or false. However, the main distinction between the two lies what each of them does after the condition checking is complete.

If the condition comes out to be false in the case of a verify command, then the execution stops and no further tests will be executed. However, if the condition is true then the program control will continue executing the next test step.

Verify command, on the other hand, does not care about the result of the condition checking. Whether it is true or false, the program execution continues and all the test steps will be completed.

**Question: What do you understand by XPath in Selenium? Can you tell the difference between “/” and “//” in XPath?**

**Answer**: XPath is a type of locator in Selenium that is used to locate a web element based on its XML path. XML denotes Extensible Markup Language, which is used for storing, organizing, and transporting arbitrary data. Much like HTML tags, XML stores data in a key-value pair.

Since HTML and XML both are markup languages, XPath can be used for locating HTML elements on a webpage. The underlying principle of XPath is traversing between several elements across the entire webpage and allowing them to find an element with the reference of some other element.

The single slash i.e. ‘/’ is used to create XPath with the absolute path, while the double slash i.e. ‘//’ is used for creating XPath with the relative path.

In the absolute path, the created XPath will start selection from the document node or the start node. However, in the relative path, the created XPath can start selection from anywhere within the entire web document.

**Question: How will you launch the browser using WebDriver?**

**Answer**: The syntax used for launching Google Chrome, Mozilla Firefox, and Internet Explorer using WebDriver is respectively,

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

**Question: Please explain how to find if an element is displayed on the screen.**

**Answer**: The WebDriver component of the Selenium suite facilitates checking the visibility of web elements, which can be buttons, checkboxes, drop boxes, labels, radio buttons, et cetera. WebDriver allows doing so with the following three methods:

* isDisplayed()  
  boolean buttonPresence = driver.findElement(By.id(“some id”)).isDisplayed();
* isEnabled()  
  boolean searchIconEnabled = driver.findElement(By.id(“some id”)).isEnabled();
* isSelected()  
  boolean buttonSelected = driver.findElement(By.id(“some id”)).isSelected();

**Question: What do you mean by Same Origin Policy? How to handle it?**

**Answer**: An Origin is a sequential combination of host, scheme, and port of the URL. The issue of the [same-origin policy](https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy) restricts accessing the DOM of a document from an origin that is different from the one that a user is trying to access the document.

The Selenium Core isn’t allowed to access the elements from an origin that is different from where it was launched. Selenium Remote Control was introduced in order to handle the problem of Same Origin Policy.

**Question: Do you know how to get a text of a web element using Selenium?**

**Answer**: In order to retrieve the inner text of a specified web element, Selenium offers the get command. It returns a string value and doesn’t require any parameters. Get command is one of the most widely used commands for verifying errors, labels, messages, etc. displayed on webpages. The general syntax for the get command is:

String Text = driver.findElement(By.id(“Text”)).getText();

**Question: Please enumerate the various types of Drivers and Waits available in WebDriver.**

**Answer**: WebDriver provides support for the following drivers:

* AndroidDriver
* ChromeDriver
* FirefoxDriver
* HtmlUnitDriver
* InternetExplorerDriver
* IPhoneDriver
* OperaDriver
* SafariDriver

There are two types of waits available in WebDriver, explained as follows:

* **Implicit Wait –** Used for providing a default waiting time between each successive test step or command across the entire test script. Hence, the next test step or command will only execute when the set default waiting time, say 30 seconds, have passed since the execution completion of the previous test step or command. Can be applied to a particular instance or several instances.
* **Explicit Wait –** Used for halting the execution until the occurrence of a particular condition or till the elapsing of the maximum time. Applied for a particular instance only.

**Question: What do you understand by Object Repository? How do you create one in Selenium?**

**Answer**: The term Object Repository refers to the collection of web elements that belong to AUT (Application Under Test) and their locator values. A corresponding locator value can be populated from the Object Repository whenever an element is required within the script.

Instead of hardcoding locators within the scripts, they are stored in a centralized location using Object Repository. Typically, the objects are stored in an excel sheet in Selenium which acts as the Object Repository.

**Question: Please explain how to click on a hyperlink using its text.**

**Answer**: The following command finds a specified element using the linkText() method and then clicks on that element to redirect the user to the corresponding webpage:

driver.findElement(By.linkText(“Google”)).click();

Another command that can be used for the same purpose is:

driver.findElement(By.partialLinkText(“Goo”)).click();

In this command, we use the partialLinkText() method. The aforementioned command finds the element based on the substring, Goo in this case, of the link provided.

**Question: What is the most important difference between driver.close() and driver.quit() commands in Selenium?**

**Answer**: The close() method closes the currently accessed window by the WebDriver. Neither does the command requires any parameter nor does it returns any value.

Unlike the close() method, the quit() method is used for closing down all the windows opened by the program. Like close() command, the quit() method doesn’t require any parameter nor does have any return value type.

**Question: How will you find more than one web element in the list using Selenium?**

**Answer**: Selenium offers WebElement List for finding more than a single web element in the list. Its use is demonstrated by the following code snippet:

List elementList =

driver.findElements(By.xpath(“//div[@id=‘example’]//ul//li”));

Int listSize = elementList.size();

for (int i=0; i<listSize; i++)

{

serviceProviderLinks.get(i).click();

driver.navigate().back();

}

**Question: Can you explain the differences between Selenium and QTP?**

**Answer**:

* **Availability –** Selenium is an open-source and free-to-use testing tool. QTP, on the other hand, is a licensed and commercial testing tool.
* Browser Compatibility – While QTP provides support for only Chrome, Firefox, and Internet Explorer, Selenium can be used with the aforementioned plus Opera, Safari, and several others.
* **Object Repository –** QTP automatically creates and maintains an Object Repository. However, this is not the case with Selenium as one needs to create an Object Repository while working with the automation testing tool.
* **Programming Language Support –** The only programming language supported by QTP is VB but Selenium provides support for a multitude of programming languages, including C#, Java, Perl, Python, and Ruby.
* **Testing Support –** Whereas Selenium offers testing of only web applications, QTP provides testing support for both web-based and Windows-based applications.
* **Vendor Support –** No vendor support is available with Selenium while the same is available for QTP.

**Question: How will you handle web-based pop-ups in Selenium?**

**Answer**: WebDriver allows handling web-based pop-ups via the Alert interface. The general syntax is:

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

alert.accept();

A total of 4 methods are available for handling the web-based pop-ups, namely:

* String getText() – Returns text displayed on the alert box
* void accept() – Clicks on the ‘Ok’ button as soon as the pop-up appears
* void dismiss() – Clicks on the ‘Cancel’ button as soon as the pop-up appears
* void sendKeys(String stringToSend) – Inputs a specified string pattern in the alert box

**Question: Can you explain the various types of navigation commands supported by Selenium?**

**Answer**: Selenium supports a total of 4 navigation commands, listed as follows:

* navigate().back() – Takes the user back to the previous webpage as per the web browser history. Requires no parameters
* navigate().forward() – Navigates the user to the next webpage in the web browser history. Requires no parameters
* navigate().refresh() – Reload all the web elements by refreshing the current webpage. Requires no parameters
* navigate().to() – Lets the user launch a new web browser window and navigate to the specified URL given as a parameter

**Question: When should we use findElement() and findElements()?**

**Answer**: findElement() – Used for finding the first element in the current webpage matching to the specified locator value. Irrespective of the number of positive matches, only the first element will be fetched. Its general syntax is:

WebElement element = driver.findElements(By.xpath(“//div[@id=’some id’]//ul//li”));

findElements() – Used for finding all elements matching the specified locator value in the current webpage. All matching elements will be fetched and stored in the list of WebElements. The general syntax for the method is:

List elementList = driver.findElements(By.xpath(“//div[@id=’some id’]//ul//li”));

**Question: What is JUnit? Explain the various JUnit annotations.**

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

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

**Question: Please explain the various types of Test Automation Frameworks.**

**Answer**:

* **Behavior-Driven Development Framework –** Allows automating functional validations in an easy-to-read and understandable format for different professionals, including analysts, developers, and testers.
* **Data-Driven Testing Framework –** Helps in segregating the test script logic and the test data. Allows storing test data in some external database in the form of key-value pairs. These keys are used for accessing as well as populating the data within the test scripts.
* **Keyword-Driven Testing Framework –** It is an extension to the data-driven testing framework in a way that in addition to separating test data from the test scripts, a keyword-driven testing framework stores a part of the test script code in an external data file.
* **Library Architecture Testing Framework –** Works on the principle of determining the right steps and then grouping them together into functions under a library. These functions are called in the test scripts whenever required.
* **Module-Based Testing Framework –** Divides each application under testing into a number of logical and isolated modules. A distinct test script is created for each module.
* **Hybrid Testing Framework –** Offers features belonging to different types of testing frameworks. The idea is to reap in all the benefits of various approaches with a single testing tool.

**Question: What is Selenium? Define its composition?**

**Answer:** Selenium is a suite of various tools that are used explicitly for automated web testing purposes. Its compositions have Selenium IDE (Integrated Development Environment), WebDriver and RC, and Grid.

**Question: How are Selenium 2.0 and Selenium 3.0 different from Selenium?**

**Answer:** Selenium 2.0 has consolidated Selenium RC and WebDriver to make a single tool, while Selenium 3.0 is the latest version, which has Beta 1 and Beta 2 updates.

**Question: Identify the various test types that are supported by Selenium?**

**Answer:** The various test types that are supported by Selenium include the following:

1. Functional.
2. Regression.
3. CruiseCont.
4. Hudson.
5. Jenkins.
6. Quick Build.

**Question: What is the role of Assertion in Selenium?**

**Answer:** The role of Assertion in Selenium is to act as a verification point. It helps in verifying the state of the application that conforms to expectations.

**Question: What are the various types of Assertion in Selenium?**

**Answer:** There are three types of Assertion in Selenium, which include the following:

1. Assert.
2. Verify.
3. WaitFor.

**Question: What are the technical challenges with Selenium?**

**Answer:** There are several technical challenges with Selenium which includes:

1. It only supports web-based applications.
2. Bitmap comparison is not supported.
3. Third-party tools are sought for reporting purposes.
4. Vendor support is minimal as compared to other commercial tools such as HP UFT.
5. It is challenging to maintain objects in Selenium.

**Question: Differentiate between Type Keys and Type Commands in Selenium?**

**Answer:** Types Keys collects the different value attributes using the JavaScript while the Type Commands imitates like an actual user typing.

**Question: Differentiate between Assert and Verify commands?**

**Answer:** Assert commands helps in checking if the element is on the page or not. The test will fail in case of missing the required element and will get terminated. Verify commands helps in checking the element is on the page or not but will not terminate but will continue ahead on executing all the commands.

**Question: What are the distinct features of Selenium?**

**Answer:** The distinct features of Selenium include the following:

1. It supports C#, Python, Perl, JAVA, and PHP.
2. It can run on various operating systems, including Mac OS, Linux, and Windows.
3. It can easily locate elements using Xpath, CSS, and DOM.
4. Its developer community is supported by Google.

**Question: What makes Selenium better than QTP?**

**Answer:** The following features of Selenium makes it better than QTP:

|  |  |
| --- | --- |
| **Selenium** | **QTP** |
| It is an open-source | It is a commercial tool |
| It is used for testing various web-based applications. | It is used for web-based applications and testing client-server applications. |
| It supports Safari, Opera, and Firefox on Linux, Mac, and Windows. | It supports only the Internet Explorer on Windows. |
| It supports different programming languages such as Python, Perl, and Ruby. | It supports only VB Script. |

**Question: Define the parameters in Selenium?**

**Answer:** There are four parameters in Selenium which includes:

1. Host.
2. URL.
3. Port Number.
4. Browser.

**Question: What is the role of set Speed() and Sleep() methods in Selenium?**

**Answer:** The role of set Speed() and Sleep() in Selenium is to delay the speed of execution.

**Question: Define heightened privileges browsers?**

**Answer:** Heightened privileges browsers acts as proxy injections that allow different websites to do things that are normally not permitted. These browsers allow Selenium core to pen the AUT directly and thereby read and write its content without passing the whole AUT through the Selenium RC server.

**Question: What is a Borland Silk Test Tool?**

**Answer:** A Borland Silk Test Tool is used for the client-server application using a test scripting language.

**Question: What is a Selenium Test Tool?**

**Answer:** A Selenium Test Tool is used for web applications and has the flexibility to many languages, including JAVA, Perl, Ruby, and various others.

**Question: How is Webdriver beneficial over Selenium Server?**

**Answer:** Webdriver does not require the Selenium Server because it uses a completely different technology. It provides Selenium RC functionality, which provides backward compatibility to Selenium 1.0. Also, it makes a direct call to the various browsers for making automation. At the same time, in the case of Selenium RC, it requires the Selenium Server to input the Javascript into the browser.

**Question: What is the other name of Selenium WebDriver?**

**Answer:** The other popular name of Selenium WebDriver is Selenium 2.0

**Question: What are the distinct features 0f Selenium WebDriver from Selenium 1.0?**

**Answer:** The distinct features of Selenium WebDriver from Selenium 1.0 includes:

1. It helps in handling multiple frames, browsers, windows, alerts, and pop-ups.
2. It helps in-page navigation.
3. It offers a drag and drop facility on the page.
4. It applies the Ajax-based User Interface (UI) elements.
5. It offers multiple browser testing facilities which helps in improving functionality for browsers which were earlier not supported by Selenium 1.0

**Question: Can we handle colors in Web Driver?**

**Answer:** Yes, we can handle colors in Web Driver using the getCssValue(arg0) function. It will help in getting the color by sending the 'color' string as an argument.

**Question: Can we store a value, which is a text box?**

**Answer:** Yes, we can store a value, which is a text box using Web Driver. We can apply driver.findElement(By.id(“your Textbox”)).sendKeys(“your keyword”);

**Question: Can we Switch between frames?**

**Answer:** Yes, we can Switch between frames using the WebDrivers, which can take any arguments.

**Question: What are the three arguments that can be taken into consideration for Switching?**

**Answer:** The three arguments that can be taken into consideration for Switching includes:

1. **A number:** This will select the number by its zero-based index.
2. **A name or ID:** This will select a frame by its name or ID.
3. **Previously found WebElement:** This will help in using the previously located WebElement to select a frame.

Resource:-- https://hackr.io/blog/selenium-interview-questions