## 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 of the following sets of tools:

- **Selenium Integrated Development Environment (IDE)** – 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** – 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** – 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.
- The user is expected to possess prior programming language knowledge.

## Q #7) What is the difference between Selenium IDE, Selenium RC, and WebDriver?

| Feature | Selenium IDE | Selenium RC | WebDriver |
|---|---|---|---|
| Feature | Selenium IDE | Selenium RC | WebDriver |
| Browser Compatibility | Selenium IDE comes as a Firefox plugin, thus it supports only Firefox | Selenium RC supports a varied range of versions of Mozilla Firefox, Google Chrome, Internet Explorer and Opera | WebDriver supports a varied range of versions of Mozilla Firefox, Google Chrome, Internet Explorer and Opera. Also supports HtmlUnitDriver which is a GUI less or headless browser. |
| Record and Playback | Selenium IDE supports record and playback feature | Selenium RC doesn't supports record and playback feature | WebDriver doesn't support record and playback feature |
| Server Requirement | Selenium IDE doesn't require any server to be started before executing the test scripts | Selenium RC requires server to be started before executing the test scripts | WebDriver doesn't require any server to be started before executing the test scripts |
| Architecture | Selenium IDE is a Javascript based framework | Selenium RC is a JavaScript based Framework | WebDriver uses the browser's native compatibility to automation |
| Object Oriented | Selenium IDE is not an object oriented tool | Selenium RC is semi object oriented tool | WebDriver is a purely object oriented tool |
| Dynamic Finders (for locating web elements on a webpage) | Selenium IDE doesn't support dynamic finders | Selenium RC doesn't support dynamic finders | WebDriver supports dynamic finders |
| Handling Alerts, Navigations, Dropdowns | Selenium IDE doesn't explicitly provides aids to handle alerts, navigations, dropdowns | Selenium RC doesn't explicitly provides aids to handle alerts, navigations, dropdowns | WebDriver offers a wide range of utilities and classes that helps in handling alerts, navigations, and dropdowns efficiently and effectively. |
| WAP (iPhone/Android) Testing | Selenium IDE doesn't support testing of iPhone/Andriod applications | Selenium RC doesn't support testing of iPhone/Andriod applications | WebDriver is designed in a way to efficiently support testing of iPhone/Android applications. The tool comes with a large range of drivers for WAP based testing. For example, AndroidDriver, iPhoneDriver |
| Listener Support | Selenium IDE doesn't support listeners | Selenium RC doesn't support listeners | WebDriver supports the implementation of Listeners |
| Speed | Selenium IDE is fast as it is plugged in with the web-browser that launches the | Selenium RC is slower than WebDriver as it doesn't communicates directly with the | WebDriver communicates directly with the web browsers. Thus making it much faster. |

| Feature | Selenium IDE | Selenium RC | WebDriver |
|---|---|---|---|
| Feature | Selenium IDE | Selenium RC | WebDriver |
| | test. Thus, the IDE and browser communicates directly | browser; rather it sends selenese commands over to Selenium Core which in turn communicates with the browser. | |

### 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 make 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:

- 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 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 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 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:

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, 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:
**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?

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 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 with a very efficient way to handle these pop-ups using Alert interface. There are the four methods that we would be using along with the Alert interface.

- void dismiss() – The accept() 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 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 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 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 an 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 Automation framework?

**The advantage of Test Automation framework**

- 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 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 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.

### 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 | Users can easily get the vendor's support in case of |

| Feature | Selenium | Quick Test Professional (QTP) |
|---|---|---|
| | vendor’s support in troubleshooting issues | 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 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.