**EXPERIMENT NO.: 10.**

Install and Explore Selenium for automated testing

**AIM:** To install the necessary software and explore Selenium WebDriver for automated web testing. This includes setting up the Java Development Kit (JDK), Selenium WebDriver, browser drivers, and an Integrated Development Environment (IDE), and then writing and executing a basic test script.

**DESCRIPTION:**

- Selenium is a widely-used open-source framework for automating web browsers. It provides a suite of tools for automating web applications for testing purposes but is also used for web scraping and automating repetitive tasks.
- Selenium supports a variety of programming languages through the use of language-specific bindings or drivers. The languages supported by Selenium include java, C#, python, Ruby, Javascript, perl.
- Selenium test scripts can be written in any of the supported programming languages and executed in modern web browsers.
- Selenium supports a wide range of web browsers, including:
  - **Google Chrome**
  - **Mozilla Firefox**
  - **Internet Explorer**
  - **Microsoft Edge**
  - **Safari**

A simple example demonstrating how to use Selenium WebDriver with Java to open Google's homepage.

**Step1:** Install JDK and set up environmental variables.

**Step2**: Install selenium web driver

Selenium WebDriver is a set of APIs that allow you to control web browsers.

Go to https://www.selenium.dev/downloads/ and download java driver under selenium clients and webdriver language bindings. Extract the zip file and place in C:\Drivers\

**Step3:** Install browser driver.

Each supported browser has a corresponding driver that Selenium WebDriver uses to communicate with the browser. For example:

- ChromeDriver: For Google Chrome
- GeckoDriver: For Mozilla Firefox
- IEDriverServer: For Internet Explorer
- SafariDriver: For Safari.

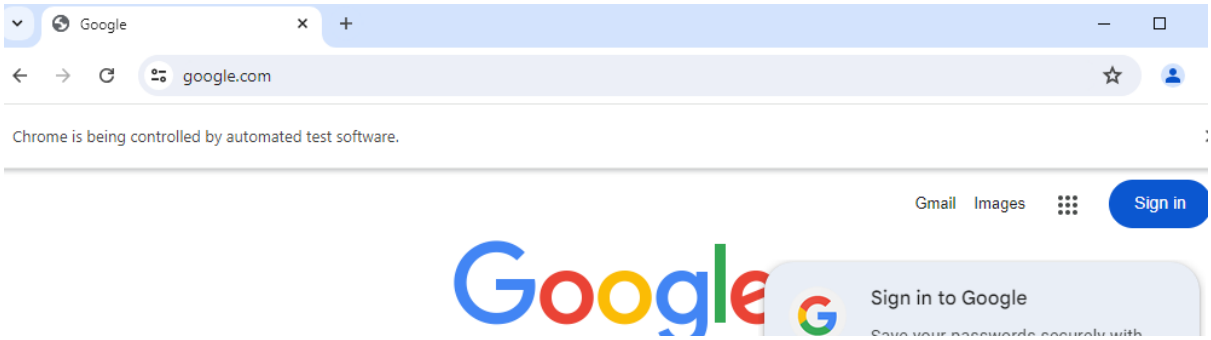
For chrome driver, check the google chrome version and go to https://googlechromelabs.github.io/chrome-for-testing/ . Download the stable release, Extract the zip file and place in C:\Drivers\selenium-java-<version>\

**Step4:** Create a simple test application test.java

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

public class Test {
    public static void main(String[] args) {
        // Set the path to the ChromeDriver executable
        System.setProperty("webdriver.chrome.driver", "C:\\Drivers\\selenium-java-4.23.0\\chromedriver-win64\\chromedriver.exe");

        // Initialize the ChromeDriver
        WebDriver driver = new ChromeDriver();

        // Open a webpage
        driver.get("https://www.google.com");

        // Print the title of the page
        System.out.println("Page title is: " + driver.getTitle());

        // Close the browser
        driver.quit();
    }
}
```

**Step5:** Compile and run the program with the following commands

```
C:\Users\kuppa\testing>javac -cp "C:\Drivers\selenium-java-4.23.0\*" Test.java

C:\Users\kuppa\testing>java -cp "C:\Drivers\selenium-java-4.23.0\*;." Test
Page title is: Google
```




**EXPERIMENT NO.: 11.**
Write a simple program in JavaScript and perform testing using Selenium

**AIM:** To write a selenium test case for a simple JavaScript program that increments a counter on a webpage. The javaScript program adds a counter functionality to a button. The goal is to test this functionality using Selenium.

## Description:

**Step1: Write JavaScript Code**: Develop a web application with JavaScript that performs certain tasks (e.g., incrementing a counter).

**Step2: Write Selenium Test in Java**: Use Selenium to write a test in Java that interacts with the web page, triggering JavaScript actions and verifying outcomes.

**Step3:** Download the following jar files and place them in c:\Drivers

- **JUnit:** Defines and runs test cases in Java, checking that the Selenium WebDriver interactions produce the expected outcomes. Install Junit jar and place it in c:\Drivers
- **Hamcrest:** Provides expressive assertion methods for verifying conditions in your tests. Install Hamcrest jar and place it in c:\Drivers

counter.html

```html
<!DOCTYPE html>
<html>
<head>
    <title>Simple JavaScript Program</title>
</head>
<body>
    <p id="output">0</p>
    <button id="increment-button">Increment</button>
    <script>
        const output = document.getElementById("output");
        const incrementButton = document.getElementById("increment-button");
        let count = 0;
        incrementButton.addEventListener("click", function() {
            count += 1;
            output.innerHTML = count;
        });
    </script>
</body>
</html>
```

## Main.java

```java
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.junit.After;
import org.junit.Before;
import org.junit.Test;

public class Main {
    private WebDriver driver;

    @Before
    public void setUp() {
        System.setProperty("webdriver.chrome.driver", "C:\\Drivers\\selenium-java-4.23.0\\chromedriver-win64\\chromedriver.exe");
        driver = new ChromeDriver();
    }

    @Test
    public void testIncrementButton() throws InterruptedException {
        driver.get("file:///C:/Users/sample/counter.html");
        driver.findElement(By.id("increment-button")).click();
        String result = driver.findElement(By.id("output")).getText();
        assert result.equals("1") : "Expected output to be 1, but got " + result;
        Thread.sleep(5000);
    }

    @After
    public void tearDown() {
        driver.quit();
    }
}
```

**Step4: compile and run the program.**

```
C:\Users\sample>javac -cp "C:\Drivers\selenium-java-4.23.0\*" Main.java

C:\Users\sample>java -cp "C:\Drivers\selenium-java-4.23.0\*;." org.junit.runner.JUnitCore Main
JUnit version 4.13.2
.
Time: 10.976

OK (1 test)
```


Simple JavaScript Program
File
C:/Users/sample/counter.html
Chrome is being controlled by automated test software.
1
Increment