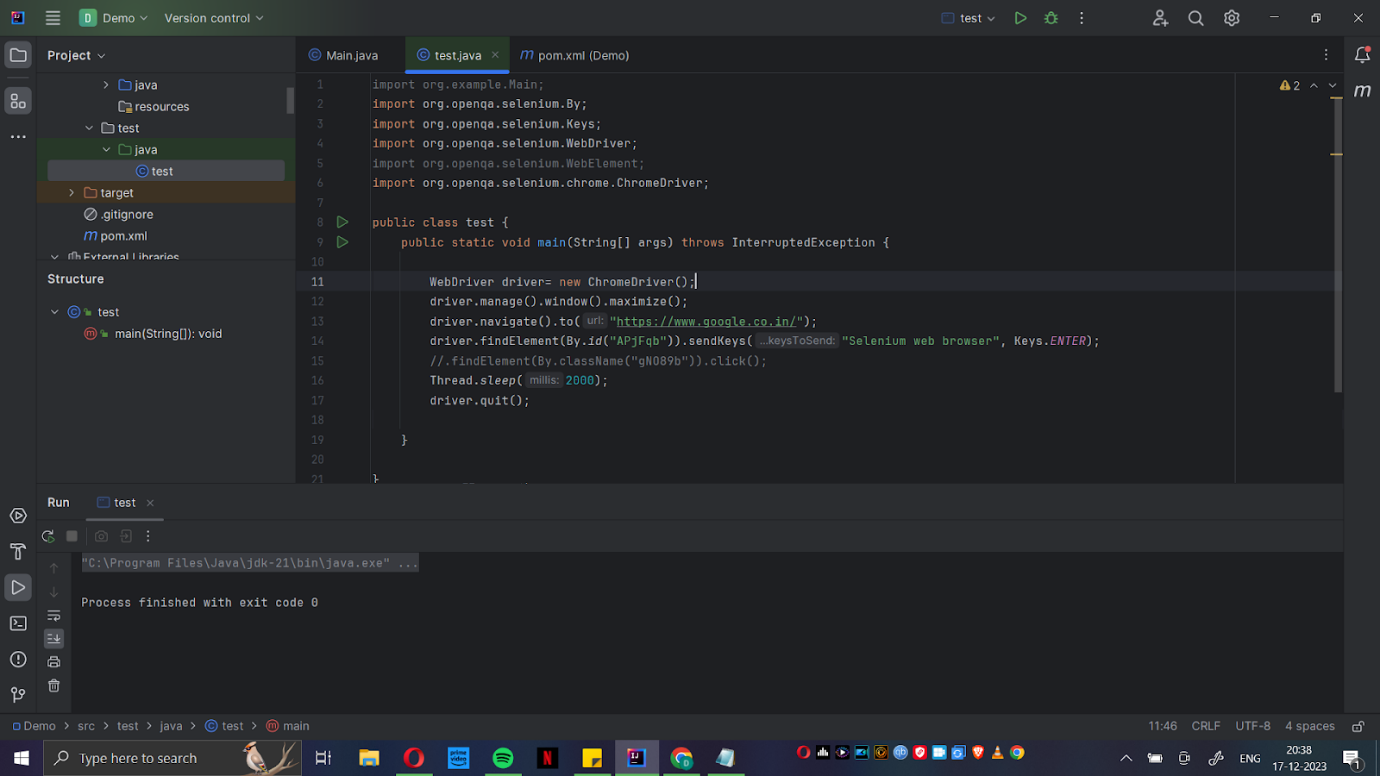
1. **Differences between Selenium IDE, Selenium WebDriver, Selenium Grid**

|  |  |  |
| --- | --- | --- |
| **Selenium IDE** | **Selenium WebDriver** | **Selenium Grid** |
| Selenium IDE (Integrated Development Environment) is a browser extension. | Selenium WebDriver is a programming interface and not a tool by itself. | Selenium Grid is a tool for parallel test execution. |
| It is primarily used for record and playback of browser interactions. | Provides a programming interface to create and execute test cases.It allows more flexibility and control over the browser | Allows the execution of test scripts on multiple machines in parallel,facilitating the testing of web applications across different environments and browsers simultaneously |
| It has limited programming language support and is more suitable for users with less programming experience. | Supports multiple programming languages like Java,Python,C#,Ruby,etc. | NA |
| Originally, it was a Firefox-only extension, but newer versions support other browsers through WebDriver integration. | Supports multiple browsers and the same test script can be run on different browsers without modification | Supports running tests in parallel on different browsers and platforms |
| Suitable for quick test case creation, prototyping, and learning Selenium. | Ideal for creating robust,maintainable and scalable test automation frameworks | Supports running tests in parallel on different browsers and platforms |

1. **Write a Selenium script in Java to open Google and search for Selenium browser driver**

**(Note: Code is uploaded in the github url: )**

Output

1. **What is Selenium? How it is useful in Automation Testing?**

Selenium is a powerful open-source framework primarily used for automating web applications for testing purposes, but it can also be extended to perform various tasks on web browsers. It provides a way for developers and testers to write scripts in multiple programming languages such as Java, Python, C#, Ruby, and more, to automate interactions with web browsers.

Some key aspects of Selenium and its usefulness in automation testing:

**Cross-Browser Compatibility Testing:**

Selenium supports multiple web browsers such as Chrome, Firefox, Safari, Edge, and others. This allows testers to write scripts once and execute them across different browsers, ensuring that web applications work consistently across various platforms.

**Programming Language Support:**

Selenium supports various programming languages, enabling testers and developers to choose the language they are most comfortable with. This flexibility makes it easier to integrate Selenium into existing development and testing workflows.

**Record and Playback:**

Selenium allows users to record their interactions with a web application and then play back those interactions as a script. While record-and-playback is a helpful feature for creating basic scripts, it is often recommended to write more robust and maintainable scripts using programming languages.

**Parallel Test Execution:**

Selenium allows for parallel test execution, which means multiple test scripts can be run simultaneously on different machines or browsers. This can significantly reduce the time required to execute a suite of tests.

**Integration with Testing Frameworks:**

Selenium can be easily integrated with popular testing frameworks such as JUnit, TestNG, NUnit, and others. This integration helps in organizing test cases, managing test data, and generating test reports.

**Support for Headless Browser Testing:**

Selenium supports headless browser testing, which means tests can be executed without a graphical user interface. This is useful for running tests in environments where a graphical interface is not available or practical.

**Dynamic Page Elements Handling:**

Selenium provides mechanisms to interact with dynamic web page elements, allowing testers to handle AJAX-based applications and perform actions like waiting for specific elements to appear or disappear.

**Extensibility and Community Support:**

Selenium has a large and active community, which means there is a wealth of resources, forums, and plugins available. This makes it easy to find solutions to common challenges and extends the functionality of Selenium as needed.

1. **List of Browser Driver used in Selenium.**

Selenium interacts with web browsers through components called browser drivers. These drivers act as intermediaries between Selenium scripts and the web browsers, allowing Selenium to send commands to the browsers and retrieve results. Each browser has its own specific driver. Here are some popular browser drivers used with Selenium:

1. Google chrome - ChromeDriver

2. Firefox - GeckoDriver

3. Microsoft Edge - Microsoft WebDriver (EdgeDriver)

4. Safari - SafariDriver

5. Opera -OperaDriver

**5. What are the steps to create a simple WebDriver script in java**

1) Set up development environment:

Install Java Development Kit (JDK) on Local system.

Install an Integrated Development Environment (IDE) such as Eclipse or IntelliJIDEA.

2) Open your IDE and create a new Java Maven project.

3) Configure WebDriver dependencies:

Add the WebDriver dependency to your project.

4) Write your WebDriver script:

//Cerating object to Webdriver interface and ChromeDriver Class

WebDriver driver = new ChromeDriver();

// launch the browser

driver.get("https://www.google.co.in/");

//Page will be idle for 2 seconds

Thread.sleep(2000);

// Close the browser

driver.quit();

5) Run your Java program to execute the WebDriver script.