INTRODUCTION TO SELENIUM

Selenium is a popular open-source tool used for automating web applications for testing purposes. It allows testers to write test scripts in multiple programming languages like Python, Java, C#, Ruby, JavaScript, etc., to simulate user interactions with web applications.

Selenium: What it is?

- Selenium is not a single tool, but a suite of tools designed to automate web browsers.
- It helps to perform functional testing, regression testing, and browser compatibility testing efficiently.
- Selenium interacts directly with web elements like buttons, links, text boxes, drop-downs, etc., to simulate real user actions.

Why Selenium?

- Free and Open Source → No licensing cost.
- Cross-platform and cross-browser → Test once, run anywhere.
- Supports multiple languages → Flexible choice for developers/testers.
- Strong community support → Quick help and resources available.
- Integration with CI/CD tools → Automate testing in the DevOps pipeline.

USAGE OF SELENIUM

Selenium is widely used across industries for **web application automation testing**.
Here are the main areas of usage

1. Functional and Regression Testing

- Verify if the website behaves as expected.
- Automate repetitive regression test cases after code changes.

2. Cross-Browser Testing

- Run the same test cases on Chrome, Firefox, Edge, Safari, etc.
- Ensure web app consistency across browsers.

3. Cross-Platform Testing

- Use Selenium Grid to test on different OS (Windows, Linux, macOS).

4. Integration with Testing Frameworks

- Works with frameworks like:
    - Pytest / Unittest (Python)
    - TestNG / JUnit (Java)
    - NUnit (C#)
- Helps organize tests, manage test data, and generate reports.

5. Continuous Integration (CI/CD)

- Integrated into DevOps pipelines with tools like Jenkins, GitHub Actions, or Azure DevOps.
- Enables automated test execution on every build or deployment.

Notes By - Ramya R

6. Data-Driven Testing

- Read data from Excel, CSV, or databases to run the same test with multiple inputs.

7. End-to-End Web Application Testing

- Automate workflows like:
    - Logging in
    - Searching for products
    - Adding items to cart
    - Checking out

SELENIUM COMPONENTS

1. Selenium IDE
    - Record and playback tool
    - Used for quick test creation
    - Browser extension (Chrome / Firefox)
    - No coding required
2. Selenium RC (Remote Control) *(*Deprecated*)*
    - Older Selenium tool
    - Allowed automation using different languages
    - Required a separate server
    - No longer used (replaced by WebDriver)
3. Selenium WebDriver
    - Most widely used Selenium component
    - Directly communicates with the browser
    - Supports multiple languages (Python, Java, etc.)
    - Faster and more reliable than RC
4. Selenium Grid
    - Used for parallel execution
    - Run tests on multiple browsers & machines
    - Supports distributed testing

INTRODUCTION TO WEBDRIVER

- **Definition:**
  Selenium WebDriver is a powerful automation tool used to control web browsers programmatically.
  It allows testers to automate browser actions like clicking buttons, entering text, navigating pages, and validating results.
- **Key Points:**
  - Introduced as part of Selenium 2.0.
  - It interacts directly with the browser (unlike Selenium RC, which used a proxy server).
  - Supports multiple browsers — Chrome, Firefox, Edge, Safari, etc.
  - Test scripts can be written in many languages — Python, Java, C#, Ruby, JavaScript, etc.

SELENIUM 4.0 INTRODUCTION

- **Release Overview:**
  Selenium 4, released in **October 2021**, brought a major architectural overhaul and many new features.
- **Key Features:**
  1. W3C WebDriver Standard Compliance – Improved browser compatibility and stability.
  2. Relative Locators – Find elements using positions like above, below, near.
  3. New Chrome DevTools Protocol (CDP) Integration – Enables network, performance, and console log capture.
  4. Improved Selenium Grid – Fully distributed, supports Docker and cloud setups.
  5. Better IDE & Documentation – Selenium IDE redesigned for modern browsers.

SELENIUM WEBDRIVER ARCHITECTURE

Selenium WebDriver follows a client–server architecture based on the JSON Wire Protocol (in Selenium 3) and W3C Protocol (in Selenium 4).

Flow of Communication:

1. Test Script (Client) — Written in your programming language (Python/Java, etc.)
2. WebDriver API — Sends commands (HTTP requests) to the browser driver.
3. Browser Driver — Acts as a server that translates those commands to the actual browser.
4. Browser — Executes the commands (click, input, navigate, etc.) and sends back responses.

Selenium WebDriver Architecture – Components

1. **Selenium Client Library**
    - Available for multiple languages: Python, Java, C#, Ruby, etc.
    - Contains APIs to write automation scripts.
2. **W3C Protocol**
    - Defines the communication format between client and browser driver.
3. **Browser Drivers**
    - Each browser has its own driver:
        - Chrome → chromedriver
        - Firefox → geckodriver
        - Edge → msedgedriver
        - Safari → safaridriver
4. **Browser**
    - The actual web browser where automation is performed.

Benefits of WebDriver over Selenium IDE and RC

1. Direct Communication: No intermediate server → faster execution.
2. Multiple Browser Support: Works with Chrome, Edge, Firefox, Safari, etc.
3. Cross-Language Support: Write scripts in Python, Java, C#, etc.
4. Dynamic Web Handling: Supports AJAX and JavaScript-heavy applications.
5. Integration Friendly: Works with TestNG, Pytest, Jenkins, etc.
6. Supports Headless Testing: Execute without UI for CI/CD pipelines.

Limitations of WebDriver

1. Desktop Apps Not Supported: Works only for web browsers.
2. No Built-in Reporting: Needs integration with frameworks like Allure or Extent.
3. Difficult to Handle Captchas or Barcodes.
4. No Record & Playback: Requires manual coding.
5. Browser Compatibility Issues: Different driver versions needed.
6. Limited to Browser-Level Automation: Cannot test mobile or APIs directly

---

BASIC SETUP FOR AUTOMATION SCRIPT

Installing Selenium Libraries in Python

Open Command Prompt (Windows) or Terminal (Mac/Linux) → pip install selenium

To install the specific version of selenium:

Go to command prompt : pip install selenium==version

To verify if the installation is proper or not

In command prompt/terminal → pip show selenium

Create a Project in PyCharm

1. Open PyCharm.
2. Click File → New Project.
3. Select Python as the project type.
4. Choose a location to save the project.
5. Click Create