Selenium WebDriver JavaScript

3 Day Classroom Course

This course covers all the essential principles of using the Selenium WebDriver JS automation framework in Visual Studio Code.

Designed for beginners as well as testers with previous automation experience, it takes the new-comer to Selenium through all the basic techniques of writing effective automated web tests.

Throughout the course practical examples are demonstrated, and the delegates get to practice what they have learned.

Key Technologies used

- Microsoft Visual Studio Code, Mocha, Chai, NodeJS & npm, WebDriverJS 3

Course Pre-requisites

- A basic understanding of HTML and how Web Pages are developed would be helpful though not essential
- Some programming experience would be advantageous - The course is based on WebDriver JavaScript, but does not teach you JavaScript.

Key Points

Introduction to JavaScript
An overview of the History & Terms, Basic Syntax, Browser Dev Tools & the Console

JavaScript in the IDE
IDEs, VS Code, Node & Npm, Setting up the workspace, executing JS, VS Code features.

First Test Case - WebDriverJS
What is Selenium? Selenium Projects, installing WebDriverJS and Browser drivers, creating our first script

Object Identification
Element locator strategies, Chrome Developer Tools and other extensions, Regular Expressions, Index, Xpath functions

Simple WebDriver Tests
Creating a simple test, execution & failures, debugging tools

Synchronisation
JavaScript Asynchronous nature, Callbacks, Promises, Async/Await, Adding Delay, Implicit versus Explicit Waits, common examples

Asserts
Node Asserts, Chai Asserts, Chai Styles, installing and using Chai

Mocha
What is Mocha? Why use Mocha, Installation, using Mocha, Mocha Options, Hooks, Mocha Timeouts, Execution options

Reporting
Mocha *--report* option, installing and using Mochawesome, writing to the results

Helper Libraries
Reusable code, writing, exporting & importing methods from modules, Common examples

Introducing Page Objects (POM)
Introduction to why we do it and how it reduces maintenance. Refactoring our tests to use a page object model. Developing a library of pages (POMs)

Handling Web Elements
Handling Web Tables, Drop-Down lists, executing JavaScript in the Browser, Keyboard input & more

Source Control
Integrating VS Code with Git & GitHub, adding our project to Source Control

Continuous Integration
Command-line execution, integration with Jenkins

3
Days

The introductory course that takes you from beginner through to writing effective automated tests in Selenium WebDriverJS

Recommended for anyone new to Selenium