Course Outline

Selenium Bootcamp with Python

Course Summary

Description

Selenium is an open source testing tool that automated testing of Web based applications. This course starts with an exploration of the Selenium 3 architecture and the Selenium "ecosystem," which is the collection of tools and related components that work with Selenium so that students can understand exactly what Selenium does and how Selenium test scripts work. Building on that base knowledge, the various Selenium testing best practices are covered along with how Selenium should be integrated into a larger testing strategy.

The latest Selenium IDE is used to write and execute scripts so that students can focus on what Selenium is doing without having to write code. Through the hands on work, students learn how to build high quality scripts that are robust and consistent testing protocols and best practices, as well as having a chance to experiment with the different Selenium capabilities and commands.

This is an intensive workshop on writing Selenium scripts using Python. The course covers all of the core functionality of the Selenium API from finding elements using different selector types to manipulating widgets and working with the various other Web artifacts like pop-ups, and AJAX. The advanced features of Selenium are covered including complex mouse operations and the use of event listeners.

However, the course goes beyond the mechanics of writing code by exploring the internals of what is happening in the API so that students can learn how to both optimize their scripts and troubleshoot issues that may arise in their test environments.

Because Selenium is generally not used as a standalone tool but tends to be used in conjunction with other test frameworks and tools, students also learn how to integrate Selenium code into the popular unit test frameworks (unittest for example) and behavior driven test frameworks (behave).
The best practices of the Selenium community are reviewed, including the essential Page Object Pattern.

Objectives

After taking this course, students will be able to:

- Use all of the Selenium commands correctly and effectively.
- Write high quality and maintainable Selenium test scripts.
- Evaluate and optimize a Selenium test script.
- Design a Selenium test suite to meet testing goals and objectives.
- Troubleshoot common problems encountered when running a Selenium test.
- Convert testing requirements into Selenium test suites.
- Develop Selenium execution protocols for execution: standalone or within a larger testing environment – for example, Acceptance Test Driven Development
- Write a high quality Selenium script in Python
- Use all of the core functionality of the API in manipulating widgets
- Use waits, timeouts and other features to tune their scripts
- Work with pop-ups, alerts, JavaScript code and AJAX applications
- Integrate Selenium into different test frameworks and other testing tools
- Effectively use the Page Object Pattern
- Work with complex mouse operations
- Create and use event listeners
- Work with advanced features like HTML5 canvas and sessions

Course Outline

Selenium Bootcamp with Python

Course Summary (cont.)

Topics

- The Selenium Ecosystem
- The Selenium Web Driver
- Script Basics
- Testing with Selenium
- Finding and Working with Web Elements
- Trouble Shooting Selenium
- Planning a Selenium Project
- The Selenium Python Binding
- Web Driver Navigation Commands
- Locating Elements
- Working with Widgets
- Mouse Actions and Listeners
- The Page Object Pattern
- Selenium Integration
- Advanced Topics
- Best Practices

Audience

This course is designed for testers or others who need to integrate Selenium into their testing activities but don't need to know how to write Selenium Web Driver code. The course is targeted at Python developers who will also be or are writing Selenium Web Driver scripts in Python. It is also designed to provide developers, who will be writing Selenium Web Driver code, a conceptual understanding of the architecture, functionality and capabilities of Selenium.

Prerequisites

Students must be able to program at an intermediate level in Python; students that do not meet this requirement will not be able to follow the course.

Duration

Three Days

Course Outline

Selenium Bootcamp with Python

Course Outline

I. *The Selenium Ecosystem*
A. What Selenium 3 is and how it came to be
B. Selenium components: Web Driver, language bindings, Selenium Grid, etc.
C. SeleniumHQ: The official Selenium project
D. Approved third party extensions
E. The Selenium IDE

II. *The Selenium Web Driver*
A. The parts of the Web Driver
B. How Selenium interacts with different browsers
C. The W3C standard based on Web driver
D. How the Web Driver interacts with a specific web page

III. *Script Basics*
A. Structure of a Selenium script
B. Recording and playing back a script in the IDE
C. Why we use the IDE for script development
D. Reading from a Web page
E. Manipulating elements on a Web page
F. Writing pass/fail criteria for a script

IV. *Testing with Selenium*
A. Interface testing with Selenium
B. Acceptance and functional testing with Selenium
C. Integrating Selenium into other tools: Cucumber, etc
D. Regression testing with Selenium
E. Test planning with Selenium scripts
F. Test execution with Selenium scripts
G. Troubleshooting common test issues

V. *Finding and Working with Web Elements*
A. The concept of a "selector"
B. Finding elements by HTML attributes (id, name, link, etc)
C. Finding elements by XPATH selectors
D. Finding elements with CSS selectors
E. Using "accessors" to read information
F. Using "actions" to modify a Web Page
G. Using "assertions" to test a Web Page
H. Browser navigation commands
I. Timeouts and page loading issues

VI. *Trouble Shooting Selenium*
A. Typical Selenium issues
B. Issues that can only be addressed in code
C. Common underlying causes for script failures
D. Issues with pop-up, roll-overs, AJAX and others

VII. *Planning a Selenium Project*
A. Developing testing goals
B. Verifying the test cases
C. Developing the use cases for interaction
D. Recording the Selenium script suite
E. Validating and Verifying the suite

VIII. *The Selenium Python Binding*
A. The Web Driver API and architecture
B. Setting up the Selenium environment for Python
C. Basic scripting workflows
D. Web Driver command categories
E. Advantages of using scripting over the Selenium IDE
F. Selenium 3 differences from previous versions
G. Review of the Web Driver architecture

Course Outline

Selenium Bootcamp with Python

Course Outline (cont.)

IX. *Web Driver Navigation Commands*

A. Page navigation commands
B. Working with windows and frames
C. Working with pop-ups and alerts
D. Managing HTTPS
E. Working with cookies
F. Using implicit and explicit waits and timeouts

X. *Locating Elements*

A. Concept of selectors: single return value versus multiple return values
B. How the "by" clause functions
C. Selecting by HTML attributes
D. Using CSS selectors
E. Using XPATH selectors
F. Deprecated selectors
G. Comparison of selector efficiency

XI. *Working with Widgets*

A. Reading and writing text based widgets
B. Manipulating widget attributes
C. Working with visibility and other issues
D. Radio buttons, check boxes and drop down lists
E. Working with IFRAMES
F. Working with forms and form submissions
G. Executing JavaScript

XII. *Mouse Actions and Listeners*

A. Moving the mouse
B. Click & hold, drag & drop operations
C. The Builder technique
D. Implementing a Web Driver event listener
E. Event types: navigation and element change events
F. Listening for script execution and exceptions
G. Using multiple listeners

XIII. *The Page Object Pattern*

A. Encapsulating Web pages – insulation Selenium code from HTML changes
B. Creating and using page objects
C. Decoupling test logic from imperative code
D. Best practices for using the page object pattern
E. Page object factories

XIV. *Selenium Integration*

A. Integrating Selenium with testunit
B. Integrating Selenium with PyTest
C. Integrating Selenium with Behave

XV. *Advanced Topics*

A. Deep dive into working with AJAX
B. Data driven testing
C. Dealing with file system I/O
D. Taking screen shots
E. Recording videos
F. Reporting tools
G. HTML5 canvas and video elements
H. HTML5 sessions and local storage

XVI. *Best Practices*

A. Best practices for writing Python Web Driver code
B. Common pitfalls and mistakes
C. Migrating older Selenium Python code to Selenium 3
D. Best practices for the workflow of developing Python scripts