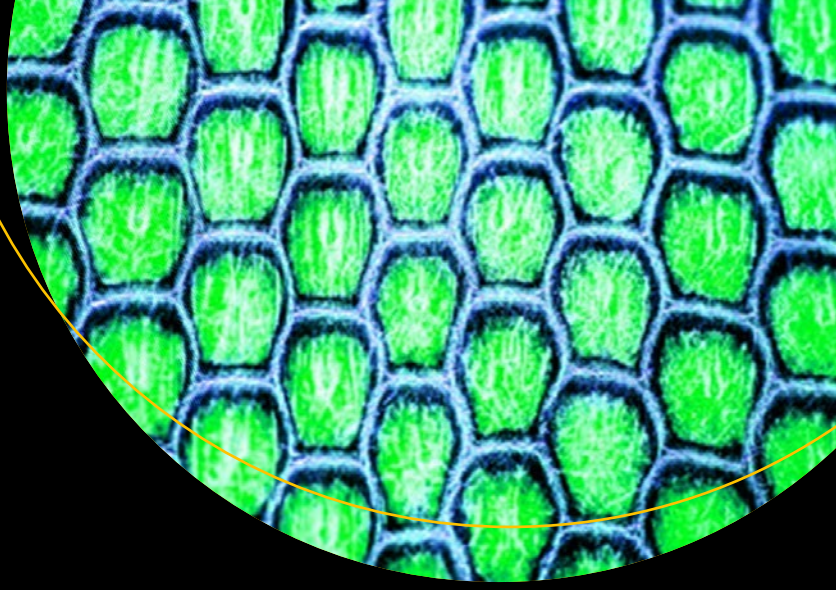

Python Testing with Selenium

Learn to Implement Different Testing Techniques Using the Selenium WebDriver

Sujay Raghavendra

Apress®

Python Testing with Selenium

Learn to Implement Different Testing Techniques Using the Selenium WebDriver

Sujay Raghavendra

Apress®

Python Testing with Selenium

Sujay Raghavendra
Dharwad, Karnataka, India

ISBN-13 (pbk): 978-1-4842-6248-1 ISBN-13 (electronic): 978-1-4842-6249-8
https://doi.org/10.1007/978-1-4842-6249-8

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Celestin Suresh John
Development Editor: James Markham
Coordinating Editor: Aditee Mirashi

Cover designed by eStudioCalamar

Cover image designed by Freepik (www.freepik.com)

Distributed to the book trade worldwide by Springer Science+Business Media New York, 1 New York Plaza, Suite 4600, New York, NY 10004-1562, USA. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail booktranslations@springernature.com; for reprint, paperback, or audio rights, please e-mail bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at http://www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/978-1-4842-6248-1. For more detailed information, please visit http://www.apress.com/source-code.

Printed on acid-free paper

To my Baba

Late Raghavendra A S

Table of Contents

About the Author

Sujay Raghavendra works primarily in data science, machine/deep learning, and artificial intelligence. He is currently the Executive Director of Raghavendra Training & Consultancy (RTC), a startup company based in Dharwad, Karnataka, India. He co-founded RTC with Sumedh Raghavendra, his brother, in 2014. He plans and evaluates new technology projects in research and product development. His projects include analyzing a pap smear filter for microscopic medical images, thermal heat sensing in hospitals, OCR for handwritten characters, satellite image analysis, and network automation for maps, forecasting models, text analytics, predictions, and more.

Raghavendra is also a consultant on building research centers for technical universities and colleges. His recent interests include automating test cases through machine learning. He has published numerous research articles in international journals and has been on reviewer committees for various journals and conferences.

About the Technical Reviewer

Supreeth Chandrashekhar is a seasoned software technology professional associated with Philips Healthcare, Bangalore. He has more than eight years of architect experience in building/running highly robust, massively scalable, and extremely secure systems, as well as setting up and managing mid-sized tech teams. He is extremely passionate about building great products that exceed customer expectations.

Acknowledgments

The person behind who I am today is my mother, Mrs. Indumati Raghavendra.

Introduction

This book focuses on how to implement testing techniques using Selenium WebDriver with the Python programming language. This quick reference provides simple, functional test cases with a syntax-based approach for Selenium WebDriver.

You'll begin by reviewing the basics of Selenium WebDriver and its architectural design history. Next, you move on to the configuration and installation of the Selenium library in various web browsers, including the basic commands needed to start test scripts. You'll review keyboard and mouse action commands for testing user interactions on a web page and see how hyperlinks are tested.

The book examines various web elements using the eight different locators provided by Selenium to help you choose the one best suited to your needs. All Python scripts are real ready-to-test examples that are explained thoroughly in problem statements. You'll use different Python design patterns to automate test scripts that can be incorporated with Selenium.

Python Testing with Selenium teaches the expertise to write your own test cases.

CHAPTER 1

Introduction to Selenium

Before Selenium, testing the functionality of a web application was done manually, which took many hours. The testing tended to rely on different scenarios. Each scenario was considered a test case to enact the behavior of the web app before its implementation. These test cases were deployed on various browsers to affirm any issues in the source code.

It requires a dedicated team of testers to check all test cases. Accuracy and time are major constraints in web development, which has led to automated test cases that can be used in different web applications without changing the source code. Selenium was developed to automate test cases.

This first chapter of the book offers a complete overview of Selenium and its core architectural design. The subtopics explain using Selenium and compares it to other testing tools in the domain. Later in the chapter, integrating Python with Selenium is explained. Let's start with a brief history and description of the Selenium tool and the reasons to use it.

What Is Selenium?

Selenium came into existence in 2004 at ThoughtWorks to test a web application named Time and Expenses by Jason Huggins. The tool was developed to test the front-end behavior of an application in various browsers. The tool was popular and open source. The increase in demand for automated testing led to the development of several versions of Selenium over the years, which are discussed next.

Selenium Tools and Versions

ThoughtWorks has released four major versions of Selenium to test web applications. Figure 1-1 shows each version and its release year.

The original version of this chapter was revised. A correction to this chapter is available at https://doi.org/10.1007/978-1-4842-6249-8_13

S. Raghavendra, *Python Testing with Selenium*, https://doi.org/10.1007/978-1-4842-6249-8_1

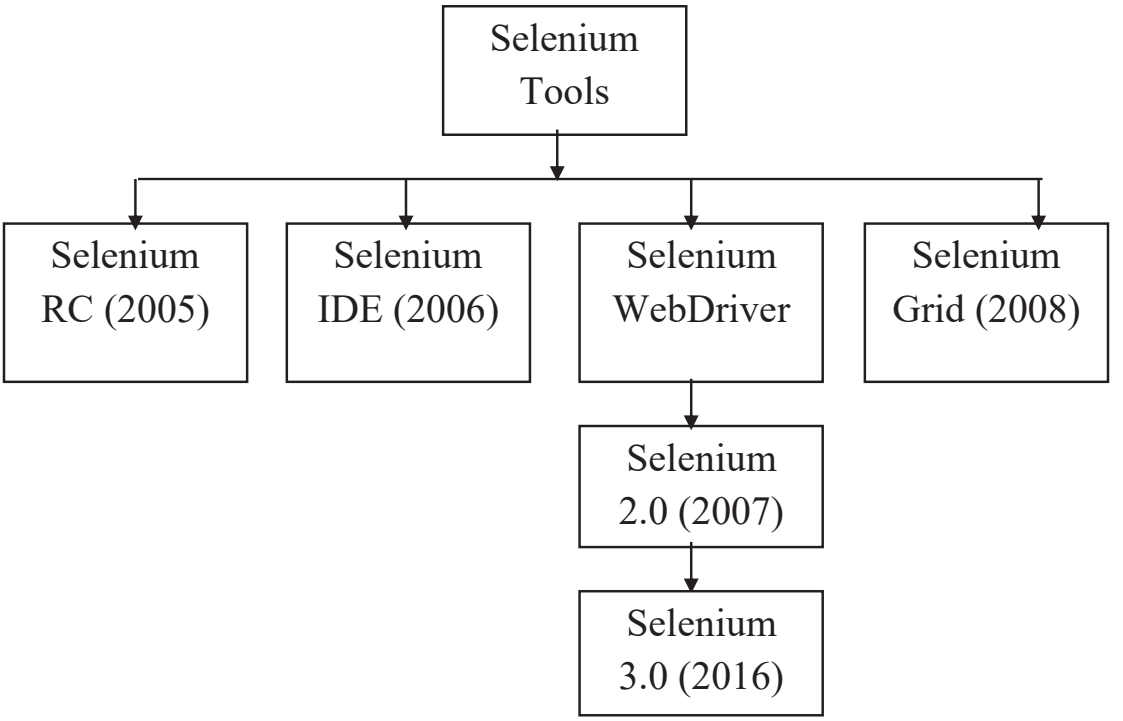

Figure 1-1. *Selenium suite*

Any one (or more) of these Selenium tools can be used by an organization; the choice depends on the test environment's requirements. The first tool developed by ThoughtWorks was Selenium Core in 2004. It allowed the tester to write their own code/script to automate front-end interactions like keyboard or mouse activities. These activities were similar to a user interacting with the application.

In web application security, there is a policy that grants a test script permission to access data in web pages from the same origin; this policy is called a *same-host* policy. A same-host policy only allows a test case to access pages within the same domain. For example, a test script can access multiple pages within `www.apress.com`, such as `www.apress.com/in/python` and `www.apress.com/in/about`, because of the same-host policy; however, this policy does not allow access to pages from different sites, such as `https://google.com` or `https://wikipedia.org`.

Due to the same-host policy, access to code elements is denied or blocked when using external scripts. To avoid this complication, Huggins and Paul Hammant developed a server component that enables you to test a web app with a test script that makes the browser believe that both are from the same source. This core Selenium was eventually known as Driven Selenium or Selenium B.