Selenium and Appium with Python

Build robust and scalable test automation frameworks using Selenium, Appium and Python

Yogashiva Mathivanan

www.bpbonline.com

First published: 2023

Published by BPB Online
WeWork
119 Marylebone Road
London NW1 5PU

UK | UAE | INDIA | SINGAPORE

ISBN 978-93-55518-354

www.bpbonline.com

Dedicated to

My beloved Parents:

Shri Mathivanan Gurusamy

Sita Kani Mathivanan

&

My brother Guru Prakash Mathivanan,
My wife Hymanivedita Yogashiva, and
My son Adyant Prakash Yogashiva

About the Author

Yogashiva Mathivanan is a highly accomplished Test Automation Architect with a Master's degree in Computer Engineering from New York, USA. He has over 10 years of experience in Software Test Automation and Framework design. Yogashiva has delivered top-quality test automation solutions to companies such as Infosys, IBM, HCL, MoneyGram, LPL Financial, T-Mobile, Experian, and Coinbase, and is currently working at CVS Health. Yogashiva has demonstrated exceptional proficiency in Java and Python programming languages and has an excellent track record of designing and implementing robust automation frameworks to automate Web, API, and Mobile Applications. Mathivanan's expertise also extends to data validation across diverse databases & data applications, performance, and security testing. Mathivanan's exceptional skills in problem-solving, attention to detail, and team management helped the companies deliver projects with the highest quality standards and have earned him a reputation as a highly reliable and effective leader in the software testing industry.

About the Reviewer

Animesh is a Test Automation Specialist with over 9.5 years of experience in India, Australia and Europe, across multiple technologies and domains. He loves talking about Python, Selenium, Appium, AWS, APIs, Docker, Kubernetes, GIT, SQL and many other technologies. He is currently working in Poland for a renowned corporate.

Acknowledgements

I feel grateful and want to thank my father, Mathivanan, my mother, Sita Kani, my brother, Guru Prakash, my wife, Hymanivedita, and my son, Adyant Prakash, for their support and inspiration throughout the writing of this book. Their encouragement and belief in my abilities helped me navigate through the challenges I encountered along the way. Their presence in my life has been a blessing.

I am also deeply grateful to Priyadarshini Institute of Technology, Nagpur, India where I completed my bachelor's degree, and New York Institute of Technology, NY, USA, where I earned my master's degree, for providing me with the knowledge and skills, that laid the foundation for this book. Moreover, I am thankful to the companies and co-workers I worked with in the software industry, who have given me invaluable experience and the confidence to pursue this project.

Finally, I would like to express my gratitude to the BPB Publications team for their support and consideration during the challenging moments I encountered while writing this book. Additionally, their guidance and expertise were instrumental in bringing out the best version of this book. Furthermore, I extend my heartfelt thanks to all the readers who have taken an interest in my work. I hope this book provides you with the tools and knowledge to help you advance in your career and achieve your goals.

Preface

Automation testing has become an essential component of software testing. The ability to automate tests not only improves efficiency but also reduces the risk of human error. This book provides a comprehensive guide to automation testing using Python programming language and popular automation testing tools Selenium and Appium, for Web and Mobile applications, respectively.

This book covers the fundamentals of automation testing and its role in testing. It then introduces Python programming for automation testing and explores Selenium and Appium for web and mobile app automation, including handling web elements, locators, and gestures on mobile. Advanced topics such as synchronization, exception handling, assertions, and hybrid application in mobile are also discussed.

The book also includes coverage of designing automation frameworks for web and mobile applications from scratch, Docker & Selenium grid, and a bonus chapter on Python interview questions to help readers in interview preparation.

This book is suitable for both beginners and experienced automation testers. It offers a practical approach and real-world examples to help you gain a deep understanding of automation testing and frameworks, enabling you to advance your automation testing career.

The book is structured into thirteen chapters that cover all aspects of automation testing, from the basics of testing and automation process, and framework design, to advanced concepts such as Dockerized Selenium Grid. The details are listed below.

Chapter 1: Testing Process and Role of Automation – provides key software testing and process concepts essential for interviews. The chapter sets the foundation for the software testing process and automation, and explains the significance of software testing in ensuring the quality & reliability of the application, and other benefits. The chapter covers the core concepts such as the different types of software testing and terms in software testing, the evolution of software testing with the Software Development Life Cycle (SDLC) models and modern Agile Methodology, defect/bug life cycle & different states of a defect, and the role and significance of automation in software testing.

Chapter 2: Python Programming - Setup and Core Concepts – provides fundamental concepts of the Python programming language, starting from installation/setup to core concepts such as variables, data types, expressions, control flow statements, and loops. The chapter also covers important data structures like lists, tuples, sets, and dictionaries, as well as functions in Python. This chapter sets the foundation for Python programming language and is essential for proceeding with software automation using Selenium and Appium with Python in the following chapter.

Chapter 3: Selenium for Web Automation – introduces the Selenium tool, covering its architecture, installation, and setup. It explains how to invoke browsers using Selenium Webdriver with and without webdriver manager, and configure the invoked browser's window, followed by the automation of the first scenario. The chapter details the key differences between Selenium 3 and 4 versions, along with the benefits of using Selenium for web automation with Python, emphasizing the popularity of Python + Selenium.

Chapter 4: Appium for Mobile Automation – introduces the Appium tool, its architecture, advantages, installation, and setup of appium and supporting tools like Appium Inspector, Android Studio, XCode, and so on. The chapter explains how to install and launch an Android application, configure an iOS simulator to launch an IOS application, and invoke and launch a mobile application and locate elements using Appium Inspector. The chapter details Appium's popularity and versatility in automating native, mobile web, and hybrid applications.

Chapter 5: Locators and Handling Web Elements – explains how to find and interact with both basic and advanced web application elements using Selenium Locators. The chapter provides a detailed explanation of these locators and how to construct and validate them, as well as information on useful browser extensions and finding multiple web elements. Additionally, readers will explore advanced web elements scenarios such as working with web tables, iFrames, window handles, and advanced operations like drag & drop and double click. The chapter emphasizes the correct usage of locators and understanding the HTML and DOM structure of the web page, to build stable automation scripts.

Chapter 6: Appium: Locators and Gestures – explains how to locate and interact with mobile application elements using Appium's locators and gestures. It covers various element locator strategies such as UIAutomator, AccessibilityID, ID, Class Name, Name, and Xpath, along with how to use the Android Inspector to locate elements and their properties. The chapter also covers essential mobile-specific driver methods, Android keycode usage, and mobile element properties/attributes. Furthermore, readers will learn how to perform different gestures such as tap, long press, swipe, and scroll gestures and how to automate them using Appium.

Chapter 7: Synchronization, Exception Handling and Assertions – provides an in-depth understanding of synchronization methods in Selenium, including unconditional and conditional synchronization, such as implicit, explicit, and fluent waits. It also covers common exceptions in Selenium and how to handle them with a try-catch block. Additionally, the chapter introduces assertions, which are utilized to validate the behavior of elements. Readers will learn how to ensure reliable and stable test execution in Selenium WebDriver.

Chapter 8: Hybrid Application Automation & Launching Multiple Apps – covers the automation of hybrid applications using Appium for both Android and iOS devices. It explains how hybrid apps consist of both native and web components, and how to identify and switch between the different contexts, such as native app and WebView, to perform actions on elements. Additionally, the chapter covers how to switch between multiple apps during execution. Overall, readers can expect to gain knowledge on how to automate hybrid apps.

Chapter 9: Selenium Automation Framework – Part 1 – introduces readers to the automation framework and the importance of choosing the right framework for a testing project. It covers various types of frameworks, their design, organizing scripts, and the use of Python packages to enhance their stability. The chapter discusses in detail the four popular Python testing frameworks, namely PyTest, Robot Framework, Unittest, and Behave, with particular emphasis on PyTest and explaining their unique features, strengths, and use cases.

Chapter 10: Selenium Automation Framework – Part 2 – focuses on implementing the Page Object Model (POM) for UI automation, covering important topics such as taking screenshots, PyAutoGUI module usage, working with configurations to manage test data, logging, parallel test execution using pytest-xdist, os & pathlib for file system management, and data-driven testing using Python packages such as NumPy, Pandas, CSV, and openpyxl. A well-structured folder hierarchy is also emphasized to help organize test files, resources, and configurations. Additionally, the chapter highlights the essential Python modules and libraries used in test automation, such as datetime, random, and faker.

Chapter 11: Mobile Automation Framework – discusses the design and implementation of a mobile automation framework using Appium. It begins with introducing the Allure reporting tool, which is used to generate detailed reports with additional features, and then it covers the various components of the Appium framework, including folder structure, driver initialization, base page, configuration, utilities, pages, and tests.

Chapter 12: Dockerized Selenium Grid – introduces Docker as a containerization platform and its advantages over traditional virtualization. The chapter covers essential Docker terminologies, and commands, and how to install and set up Docker to run Selenium tests and Selenium Grid in Docker containers. Readers will learn how to use Docker Compose to define and run multi-container Docker applications.

Chapter 13: Bonus Chapter - Python Interview Questions – contains a collection of basic and intermediate-level Python programming interview questions designed for beginners and junior engineers, as well as senior engineers. Readers can gain confidence in their Python programming skills and increase their chances of landing a job in a test automation role.

Code Bundle and Coloured Images

Please follow the link to download the
Code Bundle and the ***Coloured Images*** of the book:

https://rebrand.ly/g4bost0

The code bundle for the book is also hosted on GitHub at **https://github.com/bpbpublications/Selenium-and-Appium-with-Python**. In case there's an update to the code, it will be updated on the existing GitHub repository.

We have code bundles from our rich catalogue of books and videos available at **https://github.com/bpbpublications**. Check them out!

Errata

We take immense pride in our work at BPB Publications and follow best practices to ensure the accuracy of our content to provide with an indulging reading experience to our subscribers. Our readers are our mirrors, and we use their inputs to reflect and improve upon human errors, if any, that may have occurred during the publishing processes involved. To let us maintain the quality and help us reach out to any readers who might be having difficulties due to any unforeseen errors, please write to us at :

errata@bpbonline.com

Your support, suggestions and feedbacks are highly appreciated by the BPB Publications' Family.

Piracy

If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at **business@bpbonline.com** with a link to the material.

If you are interested in becoming an author

If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, please visit **www.bpbonline.com**. We have worked with thousands of developers and tech professionals, just like you, to help them share their insights with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions. We at BPB can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about BPB, please visit **www.bpbonline.com**.

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

https://discord.bpbonline.com

Table of Contents

Chapter 1

Testing Process and Role of Automation

Introduction

The definition of software testing is simple and has not changed since its origin. The actual developed software is in sync with the expected software, intended to be developed as defined in the business specification

The goal of software testing is to find potential errors in the developed software. Today, modern software testing is done using different automation tools, although a couple of years ago, the scripts were written manually and manual testers were doing the validations manually. Nonetheless, there remains a need for manual test engineers in high-sensitive software domains, for specific high-critical testing, which requires manual intervention.

Various questions might arise in our minds when we think of software testing. Where did it all start? How has software testing evolved? What is the standard testing process? What invoked the need for automation? In the field of software testing, knowing where and why it originated, will help understand the scenarios, different software testing types, their significance in the process, and how to implement them during real-time projects.

In this chapter, we will go over some of the majorly used concepts in software testing. This will provide you with great skills and a strong grip on the core concepts, while performing testing in real-world projects, as well as provide you with a great amount of confidence facing the software testing interviews.

Structure

In this chapter, we will discuss the following topics:

- Significance of software testing
- Core concepts of software testing
- Types of software testing
- Software Development Life Cycle (SDLC)
- Defect/Bug life cycle
- Automation in software testing

Objectives

By the end of this chapter, the reader will be able to understand the importance of software testing in the software industry, along with core concepts and terminologies used in software testing, different testing types, and their definitions. The reader will also have a complete understanding of the process of software testing and the defect cycle within the software development process and models, followed by testing automation significance.

Significance of software testing

Software Testing is a mandatory step in the software lifecycle. The completion of this step provides confidence and guarantees that the product is of a standard to be pushed to production for customers. It is significant because these software applications are bound to have errors, and identifying them in the early stage saves a great sum of money and time.

There are so many incidents that occurred in the past, that were caused due to software glitches. One such incident is the multiple Boeing 747 Max crashes in recent times. It was concluded that the key factor causing the crash was the **Maneuvering Characteristics Augmentation System** (**CAS**). It is assumed that maybe something went wrong during the testing of the updated system. The tragedy could have been avoided with complete comprehensive testing; the testers do impact the world. No major product launch happens without testing; any product, or even an app on

the phone you are using right now, is delivered to you after testing. Testing is very crucial to identify any bugs or errors in the system early in the stage, so that they can be fixed before being delivered to customers. Moreover, apart from quality, this ensures dependability, security, and performance, which can benefit in cost and time saving, and thus customer satisfaction. As evident from the example of Boeing 747 Max mentioned previously, the company lost its reputation and dependability, the cost for the company in compensation, and the settlement was around 3-4 billion dollars.

A few major reasons that make software testing crucial in the software development process are as follows:

- Software testing indicates the bugs and defects in the product that may have occurred during the development phase.
- Software testing provides a smooth user experience for the customer. Thus, the company can gain their trust and confidence.
- Software testing ensures that the application's performance is intact for updates or the addition of new features to existing applications.
- The application's continuous software testing over time creates a platform for the developer to improve the development process. This prevents repeating the same error that occurred before, thereby reducing the coding cycles.
- Software testing makes the process cost-efficient, by capturing early defects.

Software testing is a continuous process of delivering a clean product. Here are some major benefits of testing for companies and customers:

- **Better business optimization by reducing cost:** It is very critical to the project, to figure out at which level of the software development, the bug was identified. The later the stage that the bug was raised, the more is the cost for the company.
- **Security:** The testing phase case significantly finds the vulnerabilities in the software, which can prevent hackers from hacking the system. This ensures that the customer data or any significant information is safe.
- **Performance and efficiency of the application:** This is closely related to the reputation of the company. During the testing phase, the performance of the application can be identified, which ensures that in the long run, the customers are satisfied.
- **Reputation:** For any industry, the consumers are the most important part. As consumers, we too depend on companies that produce reliable products. The basis for this comes from how much time is spent on testing the product, to a point of its result in customer satisfaction.

- **User satisfaction:** Customers are the highest priority for any industry. Customer satisfaction is directly proportional to the customer's flawless experience with the product, which is tied to how much testing was performed to match the customer's expectations.
- **Support to the development process:** Regression testing of a product and tracking the testing over time, helps developers consider these potential error scenarios in the upcoming development cycle.

Core concepts of software testing

Some of the core concepts of software testing are as follows:

Software testing, quality assurance, and quality control: The terms software testing, quality assurance, and quality control are used closely with each other. Although they are thought to be the same, there are subtle differences among them all and these terms serve different purposes. Software testing is a process used to find possible bugs, defects, correctness, completeness, and quality of the developed software. Software testing is the core of the testing process, as shown in *Figure 1.1*, and it finalizes the software application by closing the gap between developed software and the business requirement, before the software is released to production for customer use. *Figure 1.1* illustrates these concepts of software testing:

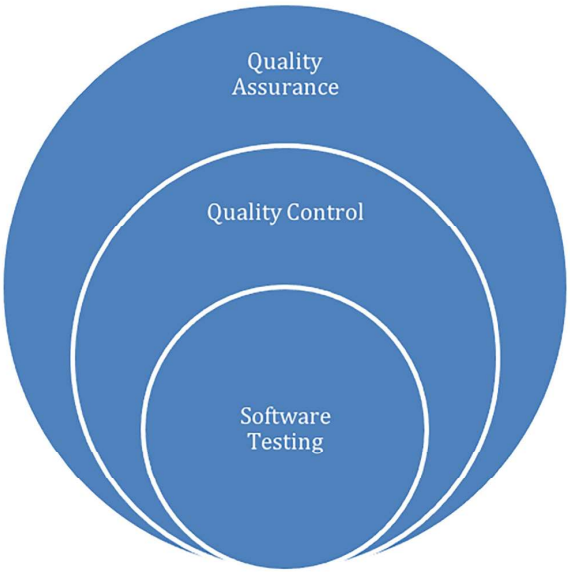

Figure 1.1: *Concepts of software testing*