Complete Guide to Test Automation

Techniques, Practices, and Patterns for Building and Maintaining Effective Software Projects

Arnon Axelrod

Apress®

Complete Guide to Test Automation: Techniques, Practices, and Patterns for Building and Maintaining Effective Software Projects

Arnon Axelrod
Matan, Israel

ISBN-13 (pbk): 978-1-4842-3831-8 ISBN-13 (electronic): 978-1-4842-3832-5
https://doi.org/10.1007/978-1-4842-3832-5

Library of Congress Control Number: 2018955901

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Shiva Ramachandran
Development Editor: Laura Berendson
Coordinating Editor: Rita Fernando

Cover designed by eStudioCalamar

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. 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 rights@apress.com, or visit http://www.apress.com/rights-permissions.

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/9781484238318. For more detailed information, please visit http://www.apress.com/source-code.

Printed on acid-free paper

In memory of my late grandparents Nathan and Lea Axelrod, pioneers of the Israeli cinema. Your memory is my inspiration.

Table of Contents

About the Author

Arnon Axelrod is a test automation expert, working as a senior consultant, architect, trainer, and lead of the test automation team at Sela Group. Arnon started programming his ZX-Spectrum when he was 10 and hasn't lost his passion for programming ever since.

After Arnon graduated with his B.Sc. in Math and Computer Sciences from Ben-Gurion University of the Negev in 1999, Arnon started to work for Microsoft as a Software Engineer in Test (SDET), where he was first exposed to the domain of Test Automation. Since then he has worked in several high-tech companies, mostly as a software engineer, until he rediscovered test automation from a new perspective. After working by Agile methodologies for several years, in 2010, while working at Retalix Ltd (later to be acquired by NCR Corporation), Arnon realized that effective test automation, and more specifically the Acceptance Test Driven Development (ATDD) technique, is crucial for delivering high-quality software rapidly and sustainably over time. While at NCR, Arnon established a test automation infrastructure that was used by over 100 developers and was running over 4,000 acceptance tests in less than 20 minutes.

In 2015, Arnon joined Sela Group, where he works now, with a mission to spread his knowledge to as many companies and individuals as possible, in order to help them develop quality software more effectively through proper use of test automation.

In his spare time, Arnon likes sailing, playing the piano, and singing in a chorus. Arnon lives in Matan, Israel, together with his lovely wife, Osnat, and their three boys: Ori, Elad, and Aviv.

You can follow Arnon Axelrod on LinkedIn, read his blog at `http://blogs.microsoft.co.il/arnona/`, or contact him directly at `arnonaxelrod@hotmail.com`.

About the Technical Reviewer

Bas Dijkstra is a testing and automation consultant and trainer. He specializes in creating and implementing automation strategies that support testing, starting with helping to answer the "why?" of automation all the way to writing effective automation solutions.

Bas delivers training on various subjects related to automation. He also regularly publishes blog posts and articles on various topics related to test automation, both on his website (`https://www.ontestautomation.com/`), as well as on other websites and in industry magazines.

Acknowledgments

First and foremost, to my wife Osnat – this book would not have been possible without the great support I got from you, and I know it wasn't easy! As much as I tried to not let this work affect our personal lives, I did leave you lonely for many long evenings and left you more of the house chores than I normally do. I don't think that it will make up for that, but I want to tell you one thing: I love you!

Next, I owe a big thank you to my direct manager and head of DevOps and Automation division at Sela, Shmulik Segal, who also supported me in this work and allowed me some precious time for working on this book, despite the fact that it had no economic justification. Shmulik, besides supporting me on this book, I appreciate you as a manager and as a person. You empower me to reach peaks in my career that I never even thought I could. And you do all of that very pleasantly.

I also want to thank Sasha Goldshtein, ex-CTO of Sela (author of *Pro .Net Performance* by Apress, 2012; and coauthor of *Introducing Windows 7 for Developers* by Microsoft Press, 2011), who tried to dissuade me from writing this book, but apparently failed. You were right at least about one thing: it took me much longer than I planned. But nonetheless you helped and advised me a lot, including recommending me to submit a book proposal to Apress.

Also, at Sela, I want to thank Zohar Lavy, who coordinates my schedule and helps me with many administrative tasks – it's a real pleasure working with you! To all the administrative staff at Sela for all the important and hard work you do behind the scenes; and to my skip managers and owners of Sela, David Basa, CEO; and Caro Segal, president of the Sela College; as well as Ishai Ram, VP Global, for leading Sela and making it such a great place to work at. And finally, for all of my talented coworkers – I learn a lot from each and every one of you.

To Carl Franklin and Richard Campbell, hosts of the ".Net Rocks" podcast, for expanding my horizons, making me laugh, and making my commute much more enjoyable. Carl, thanks also for creating the "Music to code by" collection that helped me stay focused while working on this book.

I must also thank all of the people that actually made this book take shape: first of all to Bas Dijkstra, my excellent and highly professional technical reviewer for reading every sentence thoroughly and providing his invaluable insights, feedback, and suggestions for making this book better. Without you, this book would probably be a piece of crap…

And lastly for all of my editorial staff at Apress: Rita Fernando Kim, my coordinating editor for managing the progress of this work, and for providing valuable tips and advice for anything I asked or should have asked. To Laura C. Berendson, development editor, for helping me shape and present my ideas in the best way possible; Shivangi (Shiva) Ramachandran, editor, for managing this project; and for Susan McDermott, senior editor, for accepting my book proposal and believing in me in the first place. Thank you all!

Introduction

There are many great books about test automation, and particularly about best practices of Test Automation. However, there's no one size fits all. As I once heard someone saying: "'Best Practices' is always contextual: even something as common as breathing may be catastrophic if the context is free diving…"

Most of the books that I have read so far about test automation are aimed mainly for developers, focusing mainly at unit tests or at developer-written end-to-end tests. Some other books that I either read or know about deal with a specific test automation technology, methodology, or are simply just too outdated. While I tend to agree that the idea of developers writing the tests may be very effective in many situations, in reality it doesn't fit all organizations at all stages. Moreover, test automation is a tool that serves and affects nearly all stakeholders of a software development organization, including testers, product managers, software architects, DevOps people, and the managers of the projects, and not only developers. As every software organization and project is different, trying to adopt techniques, practices, and tools that don't fit the team's needs or skills can cause the failure of the automation project and in some cases even the failure of the entire software project.

The goal of this book is to give a broad view on the subject of test automation in order to allow the reader to make smart decisions upon his particular case, giving his constraints and the benefits he wants to gain from having the test automation, but also to provide detailed and hands-on guidance for building it effectively, at least for the majority of cases.

Who Should Read This Book?

As test automation affects nearly all stakeholders of software development organizations, and as this book attempts to cover nearly all aspects of test automation, this book is for everyone who's involved in the process of software development and is interested in knowing how to get more value out of test automation. This includes: QA managers, dev managers, developers, testers, architects, product managers (AKA business analysts, system analysts, or various other titles), DevOps people, and more. Ah, and of course test automation developers whose main job is to develop automated tests…

While most of the book is not too technical and is aimed at the broader audience, Chapters 11–14 are very technical and aimed for people who write code and are proficient with object-oriented programming (OOP). In particular – professional test automation developers. The code in this section is written in C#, but the ideas and concepts are transferrable to any object-oriented language. As C# and Java are very similar, there shouldn't be any problem for Java programmers to understand the code, but I'm also sure that programmers of other languages can understand the code or at least the main ideas behind it pretty easily.

In particular, I hope that many Dev and QA managers will read this book, as they typically have the biggest effect on shaping the methodology and working processes in their organization, which test automation should integrate with, and can help to improve. Having said that, this book also contains useful tips and techniques for non-managers for improving the methodology and working processes of their organization even without any formal authority.

How This Book Is Organized?

When I first sat down to start writing this book, I tried to think about the high-level structure of the book, but I found this task very baffling because it seems that almost any topic is related to many other topics. At that time, I couldn't find a clear and logical way to divide the content to high-level sections, so I ended up writing a "laundry list" of topics I wanted to cover and just started writing by letting the knowledge spill from my head down to the paper (or keyboard to be more precise…). Naturally I started from the most basic and general stuff and slowly built upon that with more and more chapters that are more advanced or specific. Because the topics are so interrelated, I often wrote a forward reference to a topic I was yet to write, and of course references from more advanced chapters to earlier ones. Eventually, like in a good Agile project (talking about cross-references… see Chapter 1 for more about Agile), the high-level structure of the book gradually started to emerge. At some point I realized that the book took a pretty logical structure consisting of two parts: The first part answers more of the general "Why" and the "What" questions, and the second one answers the more specific and technical "How" questions.

Generally, I encourage most readers to read the entire book from cover to cover. However, as this book aims at a broad audience, with different concerns, different starting points, interests, needs, etc., you might prefer to focus on specific chapters

and skim, or even skip, others, optionally jumping back and forth to other chapters referred to from the chapter you're reading if you feel you need to fill in the gaps. Finally, keep this book within reach for later reference as the use of test automation in your organization matures and faces new challenges.

Here's an overview on each part and chapter in this book:

Part I: The "Why" and the "What"

This part covers the subject of test automation from many different aspects, but more in a "high-level" manner. This part is essential for those who don't have much experience with test automation and want to understand how it fits the big picture of software development, and where to start. This part will also help you understand what you can expect, as well as what you *shouldn't* expect from test automation. It is especially relevant for Dev or QA managers, as it discusses aspects like business structure, working processes, architecture, and more. It will guide you through many decisions that you'll have to make (which many people don't even consider!) and tell you what effect each decision might have. Even if you're not a manager and don't think that you have any influence over these things, I encourage you to read it in order to understand the constraints and advantages in your current situation, and to be able to communicate it better with your managers.

If you already have experience with test automation, this part can, and probably will, expand your horizons about the subject and show you alternatives and consequences of decisions you previously made less consciously.

Part II: The "How"

After you've gained the high-level understanding about the domain of test automation, it's time to roll up our sleeves and start writing some tests and the required infrastructure. After we write some tests, we'll discuss how to take it forward and to use the test automation most effectively in the development life cycle.

Conceptually, this part could be divided into two subparts (though this division is not mentioned explicitly anywhere except for here): Chapters 9–14 are written as a hands-on tutorial, in which we design and build a test automation system with few tests (using Selenium) for an existing open source project, and Chapters 15–19 provide guidance for using test automation in the most effective way, and how to get the most out of it.

Most of the chapters in the first subpart of Part II are very technical, while in the second subpart they are not. Therefore, the first subpart is more suited and relevant for developers, particularly test automation developers, with OOP skills, while the second subpart is relevant for everyone. For skilled programmers, I encourage you to follow along the tutorial step by step and do each step yourself, in order to *experience* it better. For non-programmers, I encourage you to skim over these more technical chapters in order to get the main idea behind them, even if not for knowing exactly how to implement it in your own project.

Here's a complete description of the chapters:

Part I:

- **Chapter 1: The Value of Test Automation** – this chapter discusses why test automation is needed and what its short-term and long-term benefits are.
- **Chapter 2: From Manual to Automated Testing** – this chapter discusses the differences between manual and automated testing and starts to set realistic expectations for test automation, as it's pretty different from just faster manual tests.
- **Chapter 3: People and Tools** – this chapter discusses who should write the tests and the automation infrastructure, and what the consequences of the alternatives are. In addition, it discusses how to choose the right tool according to these alternatives.
- **Chapter 4: Reaching Full Coverage** – this chapter sets realistic expectations for the long-term road map of the automation project, and shows how to start gaining precious value out of it long before the automation replaces most of the manual regression tests.
- **Chapter 5: Business Processes** – this chapter discusses how test automation is related to the business processes for developing software, and provides overviews for topics that will be discussed in greater depth toward the end of the book.
- **Chapter 6: Test Automation and Architecture** – this chapter discusses how test automation is related to the architecture of the tested system, and why it's important to adopt them to one another.

- **Chapter 7: Isolation and Test Environments** – this chapter discusses how to plan the automation and its execution environments to ensure that the tests are reliable and are not affected by any undesired effects.
- **Chapter 8: The Big Picture** – this chapter discusses the interdependencies between all of the subjects discussed in the previous chapters, mainly architecture, business structure, business processes, and of course test automation. It also discusses how all of these relate to business culture.

Part II:

- **Chapter 9: Preparing for the Tutorial** – this chapter describes the process that I'm going through in the tutorial, which is also applicable to most test automation projects. It also guides you how to set up your machine for following along with the tutorial.
- **Chapter 10: Designing the First Test Case** – this chapter teaches a specific technique for designing the test cases in a way that best suites automated tests.
- **Chapter 11: Start Coding the First Test** – this chapter shows you how to start writing the code for the first test. We start by writing a mere skeleton of the test in a way that will lead us to design and create a modular and reusable infrastructure. By the end of this chapter, our test compiles but does not work yet.
- **Chapter 12: Completing the First Test** – in this chapter we complete the work that we've started in the previous chapter. By the end of this chapter, we have a working test and a well-designed infrastructure to support it.
- **Chapter 13: Investigating Failures** – in this chapter we'll practice how to investigate and deal with a real test failure that occurred while we've got a new build of the tested system, and how to create a report that will help us investigate additional failures in the future.

- **Chapter 14: Adding More Tests** – in this chapter we'll add one more test, but also discuss how to go about adding more and more tests, while expanding and improving the infrastructure to support them, including support for cross-browser testing, support for multiple environments, and more.
- **Chapter 15: Continuous Integration** – this chapter (which starts the second subpart of Part II) discusses how to integrate the tests into a Continuous Integration (CI) build. More than the technical aspects, this chapter covers how to make it succeed as an organizational tool and provides advice for non-managers for how to gradually change the culture and processes of the organization for the best, by leveraging the CI.
- **Chapter 16: Acceptance Test Driven Development** – this chapter explains the benefits and how to implement the Acceptance Test Driven Development (ATDD) methodology, which expands on CI to encompass the entire development life cycle and help the team to become really effective with Agile.
- **Chapter 17: Unit Tests and TDD** – this chapter discusses the techniques that are traditionally attributed only to the application developers: unit tests and Test Driven Development (TDD) but are, in fact, an inseparable part of test automation.
- **Chapter 18: Other Types of Automated Tests** – this chapter discusses additional types of test automation, including performance and load testing, testing in production, Visual Testing, Installation tests, Artificial Intelligence, and more.
- **Chapter 19: Where to Go from Here** – this chapter provides some tips for how to continue to learn and improve in the domain of test automation.

In addition to these chapters, there are four appendices:

- **Appendix A: Real-world examples** – this appendix is supplementary to Chapter 6 ("Test Automation and Architecture") and provides four real-world examples of application architectures and their corresponding automation solutions.

- **Appendix B: Cleanup mechanism** – this appendix describes how to build a cleanup mechanism, which is described in Chapter 7 ("Isolation and Test Environments").
- **Appendix C: The Test Automation Essentials project** – this appendix describes the Test Automation Essentials open source project that I created, which contains many useful code utilities (in C#) for test automation projects.
- **Appendix D: Tips and practices for programmer's productivity** – this appendix supplements Chapters 9–14 with tips for increasing your productivity as a programmer. While these tips are relevant for any developer, I find it especially useful for test automation developers.

Happy reading!