



# Software Testing for Managers

An Introduction to Strategies,  
Technologies, and Best Practices

---

Ross Radford

Apress®

# **Software Testing for Managers**

**An Introduction to Strategies,  
Technologies, and Best  
Practices**

**Ross Radford**

**Apress®**

# ***Software Testing for Managers: An Introduction to Strategies, Technologies, and Best Practices***

Ross Radford  
Elgin, TX, USA

ISBN-13 (pbk): 979-8-8688-0571-4  
<https://doi.org/10.1007/979-8-8688-0572-1>

ISBN-13 (electronic): 979-8-8688-0572-1

Copyright © 2024 by Ross Radford

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Managing Director, Apress Media LLC: Welmoed Spahr  
Acquisitions Editor: Susan McDermott  
Development Editor: Laura Berendson  
Project Manager: Jessica Vakili

Distributed to the book trade worldwide by Springer Science+Business Media New York, 1 NY Plaza, New York, NY 10004. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail [orders-ny@springer-sbm.com](mailto:orders-ny@springer-sbm.com), or visit [www.springeronline.com](http://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](mailto:booktranslations@springernature.com); for reprint, paperback, or audio rights, please e-mail [bookpermissions@springernature.com](mailto: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 the Github repository: <https://github.com/Apress/Software-Testing-For-Managers>. For more detailed information, please visit <https://www.apress.com/gp/services/source-code>.

If disposing of this product, please recycle the paper

# Table of Contents

- About the Author .....vii**
- About the Technical Reviewer .....ix**
- Introduction .....xi**
  
- Chapter 1: Test Awareness ..... 1**
  - 1.1 The Software Engineer .....3
  - 1.2 Summary.....6
  
- Chapter 2: Test-Driven Development vs. Test During Development .....7**
  - 2.1 Testing Dogma vs. Practicality .....7
  - 2.2 Specification ..... 11
  - 2.3 Planning ..... 13
  - 2.4 Development ..... 15
  - 2.5 Staging: Internal..... 16
  - 2.6 Staging: External..... 18
  - 2.7 Production..... 19
  - 2.8 Retrospective ..... 21
  - 2.9 Summary..... 22
  
- Chapter 3: Quality and Assurance .....25**
  - 3.1 What Does Quality Mean to You? .....25
  - 3.2 Standards: You Should Have Some .....27
  - 3.3 Standards: Your Users .....29

TABLE OF CONTENTS

- 3.4 Standards: The Government.....31
  - 3.4.1 HIPAA.....31
  - 3.4.2 GDPR/CCPA.....31
  - 3.4.3 PCI DSS.....32
  - 3.4.4 Fair Credit Reporting Act .....32
  - 3.4.5 Accessibility.....32
- 3.5 Assuring Yourself .....33
- 3.6 Business Domain Experts .....34
- 3.7 New Bugs and Regressions .....35
- 3.8 Location, Location, Location.....37
- 3.9 User Acceptance Testing .....39
- 3.10 Dashboards and Reports.....39
- 3.11 Summary.....41
- Chapter 4: Specification .....43**
  - 4.1 Roadmaps, Stories, and Behavior .....43
  - 4.2 *Specification by Example*: Book Reference and Summary .....45
  - 4.3 Unknown Unknowns: Unspecified System Behaviors Are Not Testable .....47
  - 4.4 Refactoring.....51
  - 4.5 Upgrades.....52
  - 4.6 Given, When, Then.....55
  - 4.7 Summary.....58
- Chapter 5: Automation Reuse .....61**
  - 5.1 Be Less Assertive.....62
    - 5.1.1 Hard vs. Soft Assertions .....63
    - 5.1.2 Unit Tests .....63
  - 5.2 Shift Left: Reuse Tests Every Step of the Lifecycle .....64
  - 5.3 Reuse Tests Everywhere .....65

5.4 Fixtures, Personas, and Test Data .....	68
5.4.1 Personas.....	69
5.4.2 Fixtures.....	69
5.4.3 Test Data as a Service .....	69
5.5 End-to-End Testing and Workflows .....	70
5.6 Specification Reuse .....	72
5.7 Image Comparison Testing.....	73
5.8 Summary.....	75
<b>Chapter 6: Coverage .....</b>	<b>79</b>
6.1 The Myth of Code Coverage .....	80
6.2 Test Coverage.....	82
6.3 Only Broken Tests Find Bugs.....	82
6.4 Tests That Never Break .....	83
6.5 Skip, Ignore, and Pay Attention .....	84
6.6 A Better Metric: ERA Model .....	85
6.6.1 Score .....	87
6.6.2 Nodes vs. Versions .....	88
6.6.3 Risk.....	88
6.6.4 Risk Factors.....	88
6.6.5 How To .....	89
6.6.6 Don't Forget.....	89
6.7 Summary.....	89
<b>Chapter 7: Security .....</b>	<b>91</b>
7.1 Low Hanging Fruit: Static Code Analysis.....	91
7.2 Check the Manifest: Software Composition Analysis.....	93
7.3 You Need Help: Auditing Services .....	94
7.4 Once Again, the Government: Compliance and Data Breach.....	95

TABLE OF CONTENTS

7.5 Data Storage .....96

7.6 Data Transport.....98

7.7 Access: Principle of Least Privilege .....98

7.8 Secure Coding Practices .....99

7.9 Security Is Social: Team Education ..... 101

    7.9.1 Make a Plan..... 102

7.10 Summary..... 103

**Chapter 8: Conclusion.....105**

**Chapter 9: Pitfalls .....107**

**Glossary.....111**

**Index.....117**

# About the Author

**Ross Radford** is the founder of [testfromthetop.com](http://testfromthetop.com) and is on a mission to bring test awareness to leadership. He has spent a career developing and testing enterprise-level software for huge projects serving millions of users. He knows leaders are in a unique position to unblock and prioritize quality assurance at every level. Radford developed knowledge sharing presentations for peers, online courses and conference talks, with the common feedback from software engineers: “This is great. Now convince my boss!” He set out to do just that.



# About the Technical Reviewer

**Kelly I. Hitchcock** graduated from Missouri State University with what her parents called a useless BA in creative writing. She leveraged that useless degree into a 10-year technical writing career and then another 10 years (and counting!) in quality assurance. She derives an unhealthy amount of satisfaction from a well-written test plan and working, human-readable test automation. In the spare time she desperately lacks, Kelly writes fiction and picks up the LEGO bricks her kids can't see.

# Introduction

My name is Ross Radford, and I love software testing. Weird but true!

I've spent my whole career, first as a software engineer and later as a manager, testing software. Let me back up a bit; I wasn't always focused exclusively on testing. I spent a decade building enterprise-level software for millions of users at one of the largest financial technology companies in the world, Experian. That's the credit bureau. Not the one that had the nasty data breach—you're thinking of the other one with a similar name. At least as of the writing of this book.

Writing software is all about bugs. Other people's bugs, of course, since your code always works. Just kidding. If you write code, you write bugs. We can boil it down even more:

Code = bugs.

Wait, that's an assignment operation. A bug in logic.

Code == bugs;

Always, all code. Yes, even that code, yes really. As a coder, you can only see so many bugs made by yourself and others before you start to wonder about the imperfect nature of human creative experience. Existential crisis perhaps, or Buddhist-style acceptance that life is suffering. Or maybe you start looking into testing. That's what I did.

I'm going to bare my soul a little and admit I'm not a great coder. Above-average at best. I make up for it with my extreme versatility, intrepid attitude, and ability to quickly learn and adapt.

Okay no, that's bullshit. I'll try again. I make up for my average coding skills with methodical adherence to process. Yep, that's closer to true.

## INTRODUCTION

Project management made my life easier as an engineer. Less vagary around expectations means less guesswork and misunderstanding. Properly using tools like version control, not to mention using it the right way when collaborating with a large team, is the difference between hair-on-fire catastrophe and an uneventful professional routine.

Testing gives an engineer invaluable confidence. A reliable check against common bugs *before* anyone else sees your work. For a shy junior coder, this is a revelation.

Another reason I'm an okay-coder (last one, promise) is my natural curiosity. If there's a "correct" or "best" or just "better" way to do something, I'll try it out. So, once I felt like I was hitting a ceiling for skills developing enterprise web applications, I jumped deep into automated testing.

Now, if you're picturing me in a software job just running whole-hog into a niche technology without blessing from management, let me assure you it wasn't that easy. Buy-in from test-aware leadership is absolutely crucial. I'm fortunate my boss (and their boss, and their boss's boss) received a mandate for better quality and less bugs in production. Boom, there was my business case.

Testing is great, but it's not just a technical effort, it's an organization-level concern. The entire company, yes. Chapter 1 - Test Awareness will make it obvious why. I've read every book I could find on software management and testing. There's a glut of books about unit testing or the latest techniques to test software on a code level, but not many on how management facilitates testing, when that is absolutely key to setting your team up for success.

It's a real problem! I've given presentations on software testing at conferences. I would get excited about a new concept or technology and carefully prepare a guide for my peers to get started with this new software testing panacea I think will help. I speak my mind and the audience applauds. Or not, depending on how late in the day and the conference catering situation. Maybe someone got something out of my talk, hopefully. I've heard the feedback many times from engineers though: "This is great; good idea. Now convince my boss!"

I've interviewed friends in the spirit of "What do you wish your boss knew about testing?" I tried to get the most candid answer possible, all to make the point that leadership has to *lead* testing efforts.

This book is about testing software from a high-level perspective. We will avoid venturing into the wilderness of technical detail, because it piles up fast and can be a major distraction.

I'll explain just what I mean by that: in Chapter 1, we contrast the perspectives of individual team members against the leader owning and overseeing the whole process. Closer to the code, an individual engineer simply can't have the full view of what is needed to build testing practice into every level of the software development lifecycle. They might only see a green check mark on their unit tests and call it a day. Granted, many can see the bigger picture, but still they are not empowered to set policy.

Policy and process is *our* job.<sup>1</sup>

My goal is to present high-level information that is concise, comprehensible, and actionable immediately.

I'll try not to bury you in technical jargon. There will be some esoteric terminology occasionally. There's a glossary in the back when you need it.

We will refer to the software development lifecycle frequently. I assume you are aware of this concept if not intimately familiar. Not necessarily in the sense of an Agile workflow, but the larger idea of a pipeline from idea to product.

I'm also going to skip making the case for testing. If you're reading this, you probably have a mandated testing requirement or pressure to improve quality, or maybe you simply want to produce better software for less time and effort. Whatever your reason, I'm glad we met.

Thank you for taking an interest in testing. For me, it's personal.

---

<sup>1</sup> If you're an engineer or other individual contributor: First of all, thank you for buying this book, and I hope you like it. Please give a copy to your boss!