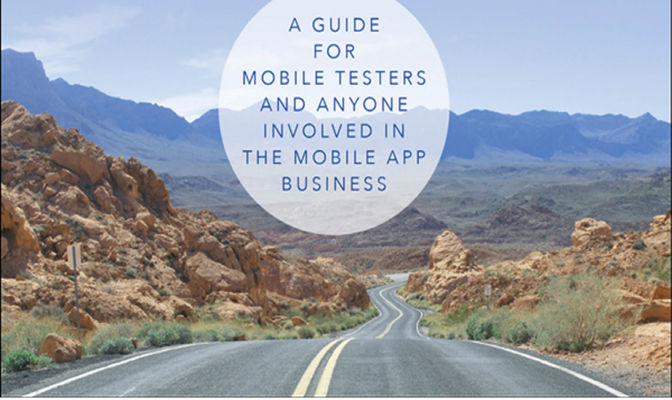
DANIEL KNOTT

HANDS-ON MOBILE APP TESTING

A GUIDE FOR MOBILE TESTERS AND ANYONE INVOLVED IN THE MOBILE APP BUSINESS



FREE SAMPLE CHAPTER

SHARE WITH OTHERS

Hands-On Mobile App Testing

This page intentionally left blank

Hands-On Mobile App Testing

A Guide for Mobile Testers and Anyone Involved in the Mobile App Business

Daniel Knott

Addison-Wesley

New York • Boston • Indianapolis • San Francisco
Toronto • Montreal • London • Munich • Paris • Madrid
Capetown • Sydney • Tokyo • Singapore • Mexico City

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The author and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact international@pearsoned.com.

Visit us on the Web: informit.com/aw

Library of Congress Cataloging-in-Publication Data
Knott, Daniel.
Hands-on mobile app testing : a guide for mobile testers and anyone involved in the mobile app business / Daniel Knott.
pages cm
Includes index.
ISBN 978-0-13-419171-3 (pbk. : alk. paper)—ISBN 0-13-419171-4
1. Mobile apps—Testing. I. Title.
QA76.76.T48K64 2015
006.3—dc23
2015009688

Copyright © 2015 Pearson Education, Inc.

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission to use material from this work, please submit a written request to Pearson Education, Inc., Permissions Department, 200 Old Tappan Road, Old Tappan, New Jersey 07675, or you may fax your request to (201) 236-3290.

ISBN-13: 978-0-13-419171-3
ISBN-10: 0-13-419171-4

Text printed in the United States on recycled paper at RR Donnelley in Crawfordsville, Indiana.
First printing, May 2015

For my wife, Sarah. Thank you
very much for your support and
encouragement while I wrote this book.

This page intentionally left blank

Contents

Preface xiii
Acknowledgments xvii
About the Author xix

Chapter 1: What's Special about Mobile Testing? 1
- User Expectations 2
- Mobility and Data Networks 3
- Mobile Devices 4
- Mobile Release Cycles 6
- Mobile Testing Is Software Testing 8
- Summary 9

Chapter 2: Introduction to Mobile Devices and Apps 11
- Overview of Mobile Networks 11
- Mobile Devices and the Year 2007 14
- The Big Two 16
- What Types of Apps Exist? 19
  - Native Apps 19
  - Hybrid Apps 20
  - Web Apps 22
- Business Models of Mobile Apps 23
  - Freemium 23
  - Paid 24
  - Transaction 24
  - Choosing a Business Model 24
- App Stores 25
- Summary 27

Chapter 3: Challenges in Mobile Testing 29
- The Customer 29
  - Customer Summary 32

Mobile Platforms and Fragmentation 32
Mobile Device Groups 33
Mobile Device Labs 34
Sensors and Interfaces 37
Ambient Light Sensor 37
Proximity Sensor 37
Acceleration Sensor 38
Gyroscope Sensor 38
Magnetic Sensor 38
Pressure, Temperature, and Humidity Sensors 39
Location Sensor 39
Touchless Sensor 40
Sensors Summary 40
Touchscreen 41
Microphones 42
Camera 43
System Apps 44
Internationalization (I18n) and Localization (L10n) 45
Mobile Browsers 47
Summary 48

Chapter 4: How to Test Mobile Apps 51
Emulator, Simulator, or Real Device? 51
Manual Testing versus Automated Testing 52
“Traditional” Testing 53
Mobile-Specific Testing 56
Mobile Functional Testing 56
Mobile Usability Testing 56
Accessibility Testing 60
Battery Usage Testing 63
Stress and Interrupt Testing 66
Performance Testing 68
Standby Testing 71
Installation Testing 71
Update Testing 72
Database Testing 75
Local Storage Testing 76
Security Testing 76

Platform Guideline Testing 81
Conformance Testing 82
Checking the Log Files 82
Be Creative! 83
Checklists, Mnemonics, and Mind Maps 84
Checklists 84
Mnemonics 87
Mind Maps 93
How to File Mobile Bugs 93
Bug ID 97
Description 97
Steps to Reproduce 98
Expected Result 98
Actual Result 98
Work-around 98
Reproducible 99
Operating System, Mobile Platform, and Mobile Device . . 99
Mobile-Device-Specific Information 99
Browser Version 99
Software Build Version 100
Network Condition and Environment 100
Language 100
Test Data 100
Severity 101
Bug Category 101
Screenshot or Video 101
Log Files 101
Tester Who Found the Bug 102
Three More Points 102
App Quality Alliance 103
Summary 103
Chapter 5: Mobile Test Automation and Tools 105
The Flipped Testing Pyramid 105
The Mobile Test Pyramid 108
Different Types of Test Automation Tools 109
Image Recognition 109
Coordinate-Based Recognition 110
OCR/Text Recognition 110

Native Object Recognition 111
Capture and Replay 111
Tool Type Recommendation 112
What Should Be Automated? 113
Emulator, Simulator, or Real Device? 115
Emulator/Simulator Pros 115
Emulator/Simulator Cons 115
Real Device Pros 116
Real Device Cons 116
Where to Automate? 117
How to Select the Right Mobile Test Automation Tool 118
Selection Criteria for a Test Automation Tool 118
Current State of Tools 120
Android Tools 120
iOS Tools 130
Mobile Test Automation Tools Summary 135
Continuous Integration System 136
Beta Distribution Tools 138
Summary 139

Chapter 6: Additional Mobile Testing Methods 141
Crowd Testing 141
Private Crowd Testing 145
Mobile Cloud Testing 147
Private Cloud 150
Cloud-Based Test Automation 152
Summary 152

Chapter 7: Mobile Test and Launch Strategies 155
Mobile Test Strategy 155
Define Requirements 156
Testing Scope 159
Define Test Levels and Test Techniques 161
Test Data 164
Select Target Devices and Test Environment 166
Manual and in-the-Wild Testing 167
Mobile Checklists and Tours 169
Test Automation 170
Product Risks 172

Mobile Launch Strategy 172
Pre-Launch—Check the Release Material 172
Post-Release—What Happens after an App Launch? 174
Community Support 174
Reviews 175
Crash Reports 176
Tracking and Statistics 176
Summary 177

Chapter 8: Important Skills for Mobile Testers 179
Skill Set of a Mobile Tester 179
Communication 179
Curiosity 181
Critical Thinking 181
Tenacity 183
Constant Learner 183
Creativity 184
Customer Focus 184
Programming and Technical Skills 185
How to Improve Your Mobile Testing Skills 185
Learn from Other Apps 186
Crashes on Specific Devices 187
Observe 192
Take Part in Competitions and Test Cycles 193
The Mobile Community and the Mobile World 194
Valuable Sources 195
Summary 198

Chapter 9: What's Next? And Final Thoughts 199
Internet of Things 199
Connected Home 200
Connected Car 201
Wearables 203
Smart Watches and Fitness Wristbands 203
Smart Glasses 204
Health Apps 205
Final Thoughts 207
Five Key Success Factors 207
Summary 209

Index 211

This page intentionally left blank

Preface

Mobile phones have been around since the middle of the 1970s. The devices have of course changed profoundly since then, but the biggest change came in 2007 when Apple presented its first iPhone. From that moment on, the mobile smartphone market has known only one direction—UP! Eight years later, touch devices such as smartphones and tablets have become ubiquitous. More than two million apps are available for download in the stores of the biggest vendors, and this number is still rising.[1] There are apps for every aspect of our lives, ranging from photos and music, to office applications and games, and on to fitness and health. But what about the quality of those apps? Are they reliable, trustworthy, easy to use, well developed, and tested?

This book is a practical guide to mobile testing for anyone who works in the mobile business, but it is especially aimed at mobile testers.

Why I Wrote This Book

It all started in 2010 when I had the opportunity to work on my first mobile project. The mobile team I worked in was responsible for developing a mobile Web app, a native Android app, and a native iOS app. This was the company's first mobile project and a completely new testing environment for the quality assurance department. Together with a colleague, I had the chance to build a mobile testing strategy from scratch. We evaluated several test automation tools to see which one fit best in our software development lifecycle. At that time, mobile testing tools were few and far between, and at a very early development stage. We then tried several testing approaches and tools. Of course we failed with some of them, but in the end the whole team, the company, and our customers were happy.

Another reason why I wrote this book was because of my blog, www.adventuresinqa.com. I started blogging in 2011 after giving a presentation at the Agile

1. iOS Store numbers, www.engadget.com/2014/06/02/apples-wwdc-2014-in-numbers-40-million-on-mavericks-and-more/; Android Play Store numbers, www.appbrain.com/stats/number-of-android-apps. Numbers are from June 2014.

Testing Days in Potsdam, Germany. This was my first talk at a major testing conference, and I was the only speaker on the agenda who spoke about mobile testing. After my presentation I was very busy for the rest of the conference as a lot of people approached me to ask about mobile testing, the approaches I use, what kind of tools I use, and so forth. The huge interest in and the lack of knowledge about mobile testing convinced me to start writing a blog. The goal was to share my knowledge of mobile testing and to exchange views and ideas with other mobile testers, while also improving my written English skills. So far I've written about 90 posts covering mobile apps and testing, and I never expected so many people from around the world to take an interest in my blog. The feedback I've gotten so far has been great, and it convinced me to take the next step.

That step is what you're reading: a book about mobile testing that captures my practical experience and knowledge for anyone involved in the mobile business. I hope you enjoy reading this book and can learn something new about the mobile testing business.

Who Should Read This Book?

This book is aimed at anyone who is interested in mobile apps and mobile testing, ranging from junior to expert mobile testers who are already involved in mobile development teams.

This book is also ideal for software test managers who need to manage mobile testing teams or to select a mobile test strategy. It's also great for software testers who are new to this topic and want to switch to mobile technologies.

Software developers who want to know more about mobile testing and testing their mobile apps have also come to the right place.

This book is also intended for product managers looking to gain further insights into the challenging job of mobile testing.

Topics Covered in This Book

This book contains the following chapters:

- **Chapter 1: What's Special about Mobile Testing?** The first chapter focuses on the special characteristics of mobile testing. It provides an introduction to mobile user expectations, mobile data networks, mobile devices, and why mobile testing is software testing.
- **Chapter 2: Introduction to Mobile Devices and Apps** Chapter 2 introduces mobile data networks and what is important to know about them.

The chapter also describes the mobile device evolution from dumb phones to the current smartphones. Furthermore, this chapter introduces the different types of apps and possible app business models.

- **Chapter 3: Challenges in Mobile Testing** Chapter 3 is all about mobile testing challenges and how to handle them. There are challenges such as the customer, device fragmentation, sensors and interfaces, system apps, and mobile browsers. Each section of the chapter provides solutions for handling those challenges in your daily business as a mobile tester.
- **Chapter 4: How to Test Mobile Apps** Chapter 4 is all about how to test mobile applications. This chapter explains the differences among emulators, simulators, and real devices. It also explains where to test a mobile app. Furthermore, this chapter provides several functional and nonfunctional approaches to testing a mobile app. In addition, this chapter presents mobile testing mind maps, mnemonics, and checklists to improve your mobile testing efforts.
- **Chapter 5: Mobile Test Automation and Tools** Chapter 5 covers the topic of mobile test automation, which is a very important one. The chapter introduces the different test automation tool types and approaches. It provides ideas for how to select the right mobile test automation tool for your test environment. Additionally, the chapter provides an overview of the current state of mobile test automation tools for Android and iOS.
- **Chapter 6: Additional Mobile Testing Methods** Chapter 6 provides an overview of additional mobile testing methods such as crowd and cloud testing. Both methods are explained, including the pros and cons and where it makes sense to use them in your mobile testing approach.
- **Chapter 7: Mobile Test and Launch Strategies** Chapter 7 deals with the topic of mobile test and launch strategies. It is very important for developers of mobile apps to have both in place in order to develop, test, and launch a mobile app with high quality. This chapter provides lots of ideas about and examples of how to establish mobile test and launch strategies.
- **Chapter 8: Important Skills for Mobile Testers** Chapter 8 describes the required skill set of a mobile tester. Furthermore, the chapter provides ideas and solutions on how to improve the skills of a mobile tester.
- **Chapter 9: What's Next? And Final Thoughts** Chapter 9 is the final chapter of this book and deals with possible topics that software testers may have to handle in the near future. The chapter contains topics such as the Internet of Things, connected homes, connected cars, and wearables. At the end, five key success factors are provided.

Each chapter focuses on the practical side of mobile testing. Sure, there will be some theoretical parts, but most of the content is based on real-life experience as a mobile tester.

How to Use This Book

This book is a practical guide to mobile testing. You can read it from front to back to get an overview of mobile testing, or you can jump straight to the chapters you're most interested in. There's one important piece of advice you should bear in mind while reading this book: make sure you have at least one mobile device next to you so you can try out the things you read.

If you want to get started with the test automation tools mentioned in this book, now would be a good time to get your computer.

Acknowledgments

Thanks to Tobias Geyer for being my dedicated reviewer. Without your contribution, encouragement, help, feedback, and critical questions this book would not have become what it is.

Thanks to Dominik Dary for your awesome feedback on mobile test automation and helping me shape the content of several chapters in this book.

Thanks to Rudolf Grötz for your great ideas about several graphics and images. Thank you very much for your help and contribution to the crowd and cloud testing chapter.

Thanks to Dagmar Mathes who gave me the opportunity to be part of the mobile testing business. Thanks for your trust, support, and encouragement.

Thanks to Sergej Mudruk and Christoph Wielgus for your support in reading my book and providing me with very useful feedback.

Thanks to Andrew Rennison for being my great copy editor, for your help and constant contribution to my text.

And last but not least, I want to say thank you to all software testers out there for sharing your knowledge of various software testing and mobile testing topics. Thanks for your contribution to the community and all of the time and effort you put in.

This page intentionally left blank

About the Author

Daniel Knott has been working in the field of software development and software testing since 2003. He started his career as a trainee at IBM where he was involved in enterprise software development and testing.

After his time at IBM, Daniel studied computer science at the University of Applied Sciences in Wiesbaden, Germany. Software testing became a passion during his time at university and is the reason he chose a career in the field. Daniel has worked at several companies in various industries where he was responsible for testing Web, desktop, and mobile applications. During a number of projects he developed fully automated testing frameworks for Android, iOS, and Web applications. Daniel is a well-known mobile expert, a speaker at various conferences in Europe, and a blog author (www.adventuresinqa.com).

Furthermore, Daniel is the founder and organizer of two local software testing user groups in central Germany. One is the Software Test User Group Rhein Main (www.stugrm.de) and the other is the Rhein Main Mobile Quality Crew (www.meetup.com/Rhein-Main-Mobile-Quality-Crew).

This page intentionally left blank

Chapter 1

What's Special about Mobile Testing?

Before I start describing the unique aspects of mobile testing, I'd like to share a true story with you.

What's special about mobile testing? Someone asked me this exact question several years ago while at a testing conference. I started talking about mobile technologies, apps, how to test them, and what's special about mobile testing. The guy simply smiled at me and said, "But it's software just on a smaller screen. There's nothing special about it." He was really arrogant and didn't see the challenges presented by mobile testing. No matter which arguments I used to convince him, he didn't believe in the importance of mobile technologies, apps, and testing.

I met the same guy again in 2014 while at a testing conference where he talked about mobile testing. He spoke about the importance of apps and how important it is to test them.

As you can see, it's very easy to underestimate new technologies. As a software tester it's especially helpful to be curious about learning something new and exploring new technologies to broaden your skills.

So let's come back to the initial question: What's special about mobile testing? I think I can assume you have at least one mobile device, namely, a smartphone. Or maybe you have a tablet, or even both. If you look at your device(s), what do you see? Just a small computer with little shiny icons on its screen? Or do you see a very personal computer with lots of sensors and input options that contains all of your private data? Please take a minute to think about that.

My smartphone and tablet are very personal computers that hold almost all of my data, be it e-mails, SMS, photos, music, videos, and the like. I can access my data no matter where I am and use my smartphone as a navigation and information system to find out more about my surroundings. For that reason I expect my apps to be reliable, fast, and easy to use.

In those three sentences I described my personal expectations of mobile devices and apps. But you may have entirely different expectations, as does the next person. And this brings me to the first special characteristic or unique aspect of mobile testing: user expectations.

User Expectations

In my opinion, the user of an app is the main focus and main challenge for mobile teams. The fact that every user has unique expectations makes it difficult to develop and deliver the "right" app to customers. As several reports and surveys have shown, mobile users have far higher expectations of mobile apps than of other software such as browser applications.[1] The majority of reports and surveys state that nearly 80% of users delete an app after using it for the first time! The top four reasons for deletion are always bad design, poor usability, slow loading time, and crashes immediately after installation. Nearly 60% of users will delete an app that requires registration, and more than half of users expect an app to launch in under two seconds. If the app takes more time, it gets deleted. Again, more than half of users experience crashes the very first time they start an app. An average user checks his or her mobile device every six minutes and has around 40 apps installed. Based on those numbers, you can deduce that mobile users have really high expectations when it comes to usability, performance, and reliability. Those three characteristics were mentioned most often by far when users were asked about their experience with mobile apps.

Currently there are more than two million apps available in the app stores of the biggest vendors. A lot of apps perform the same task, meaning that there's always at least one competitor app, which makes it very easy for consumers to download a different app as it's just a single tap away. Here are some points you should keep in mind when developing and testing a mobile app:

- Gather information about your possible target customer group.
- Ask your customers about their needs.
- Your app needs to solve a problem for the user.
- Usability is really important.

1. http://offers2.compuware.com/rs/compuware/images/Mobile_App_Survey_Report.pdf

- Your app needs to be reliable and robust.
- App performance is really important.
- Apps need to be beautiful.

There are, of course, a plethora of other things you should take into account, but if you pay attention to these points, your users are likely to be happy.

You've probably already heard of the KISS principle.[2] KISS is an acronym for Keep It Simple, Stupid and is always a useful reminder—especially for software projects—to not inflate the software with just another function or option. Keeping it small, easy, and simple is best in most cases and is likely to make your customers happy. Inspired by KISS, I came up with my own principle for mobile apps: KIFSU (see Figure 1.1). This abbreviation is a good mnemonic to help you cover customer needs and a constant reminder not to inflate apps with useless functions.

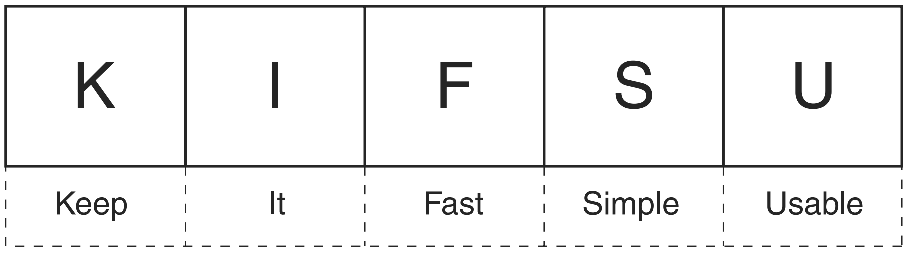


Figure 1.1 *KIFSU*

Mobility and Data Networks

Another challenge mobile apps have to deal with more than software running on computers is the fact that users are moving around while they use apps, which often requires an Internet connection to fetch data from the backend and serve the user with updates and information.

Mobile apps need to be tested in real life, in real environments where the potential user will use them. For example, if you're testing an app for snowboarders and skiers that accesses slope information, one that is able to record the speed of the current downhill run and makes it possible for users to share records directly with their friends, you need to test these functions on a slope. Otherwise you can't guarantee that every feature will work as expected.

2. http://people.apache.org/~fhanik/kiss.html

Of course, there are parts of an app that you can test in a lab situation, such as slope information availability or whether or not the app can be installed, but what about recording a person's speed, the weather conditions, or the Internet connection at the top of a mountain?

The weather conditions on a mountain, in particular, can be very difficult to handle as they can, of course, range from sunshine to a snowstorm. In such scenarios you will probably find lots of bugs regarding the usability and design of an app. Maybe you'll also find some functional bugs due to the temperature, which may have an impact on your hardware and, in turn, your app.

As I already mentioned, the speed and availability of Internet connections could vary in such regions. You will probably have a good network connection with high speed at the top of the mountain and a really poor one down in the valley. What happens if you have a bad or no Internet connection while using the app? Will it crash or will it still work? What happens if the mobile device changes network providers while the app is being used? (This is a common scenario when using apps close to an international border, such as when snowboarding in the Alps.)

All of these questions are very hard to answer when testing an app in a lab. You as a mobile tester need to be mobile and connected to data networks while testing apps.

As you can see, it's important to test your app in real-life environments and to carry out tests in data networks with different bandwidths as the bandwidth can have a huge impact on your app; for example, low bandwidth can cause unexpected error messages, and the switch between high and low bandwidth can cause performance issues or freezes.

Here's an exercise for you. Take any app you want and find three usage scenarios where the environment and/or network connection could cause problems.

Mobile Devices

Before you continue reading, pick up your mobile device and look at it. Take your device in your hand and look at every side of it without turning it on. What do you see?

You will most likely see a device with a touch-sensitive screen, a device with several hardware buttons with a charger, a headphone connection, and a camera. That's probably it—you're not likely to have more than five hardware buttons (except for smartphones with a physical keyboard).

In an era when the words *cell phone* have become synonymous with smartphone, it's important to remember that there used to be other types of cell phones, so-called dumb phones and feature phones that have lots more hardware buttons for making a call or typing a message. With a conventional dumb phone you are only able to make a call, type a message, or store a contact list; they're not usually connected to the Internet. The more advanced ones, the feature phones, have games, a calendar, or a very basic Web browser with the option to connect to the Internet. But all these phones are really basic in terms of functionality and expandability as users aren't able to install apps or easily update the software to a newer version, if it all. Both types of phones are still available, especially in emerging markets, but since 2013 more smartphones have been sold worldwide than dumb phones or feature phones,[3] and this trend is likely to continue as time goes on. In fact, in the next couple of years dumb phones and feature phones will be a thing of the past.

The phones we use nowadays are completely different from the "old" ones. Current smartphones are mini supercomputers with lots of functionality in terms of hardware and software. They're packed with various sensors such as brightness, proximity, acceleration, tilt, and much more. Besides that, all modern smartphones have both front- and rear-facing cameras, various communication interfaces such as Bluetooth, near field communication (NFC), and Global Positioning System (GPS), as well as Wi-Fi and cellular networks to connect to the Internet. Depending on the mobile platform and mobile manufacturer, you may find an array of other hardware features.

From a software point of view, smartphones offer lots of application programming interfaces (APIs) for manufacturers, developers, and users to extend smartphone capabilities with apps.

If you just focus on the major mobile platforms, iOS and Android, there are plenty of hardware and software combinations that mobile testers have to deal with. The fact that there are so many combinations is known as fragmentation. Mobile device fragmentation is a huge topic and yet another challenge when it comes to mobile testing.

You can't test your app with every possible hardware and software combination. And the fact that you should test your app in a real environment makes it even more impossible. Mobile testers need to find a strategy to downsize the effort of testing on different devices and to find a way to test on the right devices.

But how can that be accomplished? By testing on just one mobile platform? By testing on just the latest device? By testing with just the latest software version?

3. www.gartner.com/newsroom/id/2665715

Before you define a strategy, you should keep in mind that every app is unique, has unique requirements, has other problems to solve, and has a unique user base. With these points in mind, you can ask yourself the following questions to find the "right" mobile devices for testing:

- Who is my user base?
- How old is the average user?
- How many men or women are in my target user group?
- Which platform is used most among that user base?
- Which device is used most?
- Which software version is installed on most of the phones?
- What kind of sensors does my app use?
- How does the app communicate with the outside world?
- What is my app's main use case?

Of course, there are lots more questions to ask, but if you answer most of the ones I suggest, the list of possible devices you should consider testing is much shorter.

In later chapters I will describe other techniques for selecting the right devices for mobile testing.

Mobile Release Cycles

Now that you know how to find the right devices for testing your app, it doesn't mean that the process is over. To be honest, it's never going to end!

The main mobile manufacturers release a new flagship phone with more features every year. In and around those releases they bring out other phones for different user scenarios and user groups. This is especially true in the Android world where every new phone comes with a new version of the operating system packed with new features, designs, or APIs. There are multiple software releases within the course of a year, ranging from bug fixes to feature releases. You as a mobile tester need to be sure that your app will run on the latest hardware and software.

But how should you handle these situations? By buying every phone that appears on the market? By constantly updating to the latest operating system version?

Again, the most important factors are your target customer group and the app you're testing. When you know that your target group always uses the latest and fastest phones on the market, you need to buy those phones as soon as they appear. Regardless of whether or not your target group is up-to-date, you should always monitor the mobile market.

You need to know when the main vendors are due to release new flagship phones that a lot of people are likely to buy. You also need to know when the operating systems receive patches, new features, or new design patterns.

So the answer to the question of whether you need to buy every phone and constantly update the operating systems is yes and no. Of course you don't need to buy every phone that's on the market, but you should consider updating to the latest operating system version. When doing so, keep in mind that not every user will install the update. Many people don't know how to do that, or they don't care about new versions. You need at least some phones that are running older versions of the operating system to see how the app reacts in that environment. Older versions of the operating system are also needed to reproduce reported problems and bugs.

A good way to manage all this is to stick with the same operating system version on the phones that you have and buy new phones with the latest software version. This of course leads to another problem—it's really expensive! Not every manager wants to spend so much money on mobile devices when a phone is going to be used for only a couple of months. A solution for that is to rent devices. There are several providers and Open Device Labs where you can rent a device for a certain period of time (a list of providers can be found in Chapter 3, "Challenges in Mobile Testing"). Another way to rent devices is the mobile device cloud as there are a number of providers who give mobile testers exclusive access to the physical devices they have made available in the cloud. Just use your search engine and check them out.

In the mobile projects I've worked on, we always had the top ten to 15 devices used by our target user group in different variations for developing and testing. This was a good number of devices that covered nearly 90% of our target group. With those ten to 15 devices we were able to find most of the critical bugs; the remaining 10% of devices we didn't have were of no major consequence to the project or user expectations.

In order to handle the fast pace of mobile release cycles, you should keep the following things in mind:

- Monitor the mobile device and software market.
- Know when new phones will be rolled out.
- Find out about the new features of the operating systems.
- Keep an eye on your target customer group to see if new devices are showing up in your statistics.

- Think twice before updating a phone to the latest operating system version.
- Buy new phones with the latest operating system version.
- If buying is not an option, rent the devices.

Updating, buying, and maintaining all of your devices is a challenging task and should not be underestimated! At some point, depending on the number of test devices used within a project, this could be a full-time job.

Mobile Testing Is Software Testing

Let's come back to the story I told at the beginning of this chapter when the guy at the conference didn't believe in the importance of mobile testing. He had the attitude that mobile testing is not real software testing. In his opinion, mobile apps were only small programs with less functionality and no real challenges when it comes to software testing. But this is definitely not the case. If you look at the topics I described in this chapter, you should have an initial impression about the challenging job of a mobile tester. Mobile testing is totally different from testing software applications such as Web or desktop applications. With mobile apps, physical devices have far more influence over the software that is running on them when compared to other software such as Web applications. Because there are so many different smartphones available on the market, mobile testers need to focus a lot more on hardware during the testing process. In addition, users moving around and using different data networks force mobile testers to be on the move while testing.

Besides the hardware, user expectations play an important part in the daily business of a mobile tester and need to be taken seriously.

There are many more topics and issues mobile testers need to know about in order to help the whole team release a successful app. The rest of the chapters in this book will cover the following topics:

- More challenges for mobile testers and solutions to those challenges
- How to test mobile apps systematically
- How to select the right mobile test automation tool
- The different concepts of mobile test automation tools
- How to find the right mobile testing strategy
- Additional mobile testing methods
- Required skills for mobile testers

Keep the topics from this chapter in mind as a starting point. Keep your app simple and fast (remember KIFSU). Test while you're on the move, and test on different devices based on your target customer group.

Summary

The first chapter of this book mentioned some very important topics from the mobile testing world. As you have seen, mobile testing is completely different from testing on other technologies such as laptops or desktop computers. The biggest difference between mobile and other technologies is that the mobile user is on the move while he or she is using your product. Therefore, it is very important to know about the different data networks and the different types of mobile devices.

This chapter also provided a first overview of mobile users' high expectations. It is really important to keep KIFSU in mind when designing, developing, and testing a mobile app. It will help you to focus on the important elements and not waste time on unnecessary features that your users won't use.

And last but not least, this chapter should remind you to never underestimate a new technology. Be open-minded and curious to improve your daily work life.

This page intentionally left blank

This page intentionally left blank

Index

0G networks, 12
1G networks, 12
2G networks, 12–13
3G networks, 13
4G networks, 13
5G mobile standard, 13
99tests, 145

Acceleration sensor, testing, 38
Accessibility guidelines
  Android, 62
  iOS, 62
  WC3 initiative, 62
  Windows Phone, 63
Accessibility testing
  Android, 62
  auditory impairment, 61
  cognitive impairment, 61–62
  iOS, 62
  overview, 60
  physical impairment, 61
  visual impairment, 60–61
  WC3 initiative, 62
  Windows Phone, 63
Actual result, in bug reports, 98
Advanced User Interactions API, 125
Advertisements, selling, 24
Agile Record, 197
Agile Testing, 196
Agile Testing Days, 195
Amazon Appstore Guidelines, 26
Ambient light sensor, testing, 37
AMPS (Advanced Mobile Phone Service), 12
AMTS (Advanced Mobile Telephone Service), 12
Android
  design guide, 18
  guidelines, 81
Android apps
  test automation. *See* Test automation tools, Android.
  testing criteria, 103
Android Asset Packaging Tool, 66
Android Auto, 201
Android Emulator, 51
Android operating system
  accessibility guidelines, 62
  accessibility testing, 62
  App Quality Alliance site, 103
  pseudo-random test events, 66
Android smartphone
  battery status, displaying, 18
  dock section, 18
  generating pseudo-random test events, 66
  interrupt testing, 66–67
  introduction of, 14
  vs. iOS, 17–19
  JouleUnit energy profiling tool, 65
  Monkey tool, 66
  network strength, displaying, 18
  notifications, displaying, 18
  pinning apps, 18
  status bar, 18
  stress testing, 66–67
  time, displaying, 18
  user interface, 18
Android Web browser *vs.* other browsers, 48
App development
  development guidelines, 2–3

App development *(continued)*
KIFSU (Keep It Fast Simple Usable) principle, 3
KISS (Keep It Simple, Stupid) principle, 3
rejection, reasons for, 18
testing. *See* Mobile testing.
App Quality Alliance site, 103
App stores
Apple App Store, 25
BlackBerry World, 26
Google Play Store, 25
list of, 25
Microsoft Store, 26
number of available apps, 25
rejecting apps, reasons for, 18
review and publishing guidelines, 26
Samsung, 25
uploading apps to, 26
for Windows Phone, 26
Appaloosa, 138
AppBlade, 138
Appium
for Android, 127–128
for iOS, 134
native object recognition, 111
online resources, 111, 134
Applause, 138, 145
Apple App Store, 25
Apple App Store Guidelines, 26
Apple Watch, 204
"Applying the SFDPOT Heuristic to Mobile Testing," 88
Apps
developing. *See* App development.
health, 205–207
mHealth, 205–207
number available, 2, 25
pinning, 18
preinstalled, testing, 44
reasons for deleting, 2
system, testing, 44
testing. *See* Mobile testing.
uploading to app stores, 26
Windows Phone, downloading, 26
Apps, Android
test automation. *See* Test automation tools, Android.
testing criteria, 103
Apps, iOS
test automation. *See* Test automation tools, iOS.
testing criteria, 103
Apps, types of
hybrid, 20–21
native, 19–20
Web, 22–23
appsfire, 177
AppsFlyer, 177
AppThwack, 36, 149
Appurify, 149
Association for Software Testing, 196
Auditory impairment, accessibility testing, 61
Automated testing. *See* Test automation.

Bach, James, 88, 90
Bamboo, 137
Battery consumption tools, 64–66
Battery life, saving with proximity sensor, 37–38
Battery status, displaying, 18
Battery usage, testing
battery consumption tools, 64–66
Charles tool, 63
Fiddler tool, 64
fully charged battery, 63–64
Instruments energy usage function, 65
JouleUnit energy profiling tool, 65
low battery, 64
overheating, 64
usage statistics, 64–66
Beautiful Testing, 196
Belgium Testing Days, 195
Beta by Crashlytics, 138
Beta distribution tools, 138–139
BirdFlight, 138
Black Box Software Testing, 55

Black box testing, 54–55, 164
BlackBerry
  guidelines, 81
  relative market share, 14, 16
BlackBerry Web browser *vs*. other browsers, 48
BlackBerry World, 26
BlackBerry World Guidelines, 26
Blindness. *See* Visual impairment.
Blink, 47
Blogs, 196–197
Books about SQLite, 75
Books and publications. *See also* Online resources.
  Agile Record, 197
  Agile Testing, 196
  Beautiful Testing, 196
  Explore It!, 196
  How Google Tests Software, 196
  The Inmates Are Running the Asylum, 30
  Lessons Learned in Software Testing, 196
  Mobile Usability, 57
  Professional Tester, 197
  Specification by Example, 196
  Tap into Mobile Application Testing, 92, 196
  Tea-time with Testers, 197
  Testing Circus, 197
Bots, 137
Boundary value testing, 55
Browser version, in bug reports, 99
Bug Advocacy, 183
Bug category, in bug reports, 101
Bug ID, in bug reports, 97
Bug reports
  be polite, 102
  keep it simple, 102
  mind maps, examples, 94–96
  overview, 93, 96–97
  remain calm, 102
Bug reports, contents
  actual result, 98
  browser version, 99
  bug category, 101
  bug ID, 97
  description, 97
  expected result, 98
  language, 100
  log files, 101–102
  mobile device, 99
  mobile platform, 99
  mobile-device-specific information, 99
  network condition and environment, 100
  operating system, 99
  reproducible, 99
  screenshot or video, 101
  severity, 101
  software build version, 100
  steps to reproduce, 98
  test data, 100
  tester who found the bug, 102
  work-around, 98
Bugs, finding and protecting from, 53–55
Buildbot, 137
Business models
  choosing, 24–25
  freemium, 23–24
  in-app purchases, 24
  paid, 24
  selling advertisements, 24
  transaction, 24

Calabash
  for Android, 125–126
  for iOS, 131–132
  native object recognition, 111
Cameras
  mobile testing, 43–44
  resolution and image size, 43–44
  testing, 43–44
Capacitive touchscreens, 41
Capture and replay tools, 111–112
CarPlay, 201
CDMA (Code Division Multiple Access), 12

Cell phones
  generations of, 12–13
  heads, 12
  history of, 4–5
  transceivers, 12
Charles battery tool, 63
Checklists. *See also* Guidelines; Mind maps; Mnemonics.
  generic mobile testing, 85–87
  mobile test strategy, 169–170
  overview, 84
Checkstyle tool, 54
Chrome Web browser *vs.* other browsers, 48
CI systems, 137
Clicktale, 177
Cloud testing, 147–152
CloudMonkey, 36, 149
Cognitive impairment, accessibility testing, 61–62
Cohn, Mike, 105, 106
Color deficiencies. *See* Visual impairment.
Color-blindness. *See* Visual impairment.
Communication, tester skill, 179–181
Communities, 196
Community support, 174–175
Conferences, 195–196
Configuration data, mobile test strategy, 165
Connected cars, 201–203
Connected homes, 200–201
Constant learning, tester skill, 183–184
Cooper, Alan, 30
Coordinate-based recognition tools, 110
COP FLUNG GUN (Communication, Orientation, Platform, Function, Location, User scenarios, Notifications, Guidelines, Gesture, Updates, Network), 92
Crash reports, 176
Crashes on specific devices, tester skills improvement, 187–192
Creativity, tester skill, 184
Critical thinking, tester skill, 181–182
Crowd testing, 141–147
Crowd testing providers, 145
crowdsourcedtesting, 145
Crowdsourcing, 141
CruiseControl, 137
CSS3, 22
Cucumber for Android, 125–126
Curiosity, tester skill, 181
Customer focus, tester skill, 184–185
Customers. *See also* Users.
  getting information about, 29–32
  inviting to usability testing sessions, 31–32
  representing with personas, 30–32
  satisfying, checklist for, 32

Databases, testing, 75
Date formats, testing, 46
Deafness. *See* Auditory impairment.
Decision tables, testing, 55
Deleting apps, reasons for, 2
Description, in bug reports, 97
Developing apps. *See* App development.
Device movement, detecting, 38
Device orientation, detecting, 38
Dock section, Android and iOS devices, 18
Dolphin Web browser *vs.* other browsers, 48
Double tap gesture, 42
Downloading resources. *See* Online resources.
Drag gesture, 42
Dumb phones, 5
Dutch Testing Days, 195
Dynamic testing, 53–55

EDGE (Enhanced Data rates for Global Evolution), 13

eggPlant
  for Android, 129
  image recognition, 110
  OCR/text recognition, 111
Emulators
  Android Emulator, 51
  handling fragmentation, 35
  mobile device test clouds, 35
  pros and cons, 115–116
  vs. real devices, 51–52
  vs. simulators, 51–52
Energy Usage Instrument, 65–66
Espresso
  for Android, 128
  native object recognition, 111
ETSI (European Telecommunications Standards Institute) standards, 82
EuroSTAR, 195
Expected result, in bug reports, 98
Experitest
  for Android, 129
  for iOS, 134
  mobile cloud testing, 149
Explore It!, 196
ExtSolo, 122–123

FCC CUTS VIDS (Feature tour, Complexity tour, Claims tour, Configuration tour, User tour, Testability tour, Scenario tour, Variability tour, Interoperability tour, Data tour, Structure tour), 89–90, 170
Feature phones, 5
Fiddler battery tool, 64
FindBugs tool, 54
Firefox Mobile browser *vs.* other browsers, 48
Fitness wristbands, 203–204
5G mobile standard, 13
4G networks, 13
Fragmentation, definition, 5
Fragmentation, handling
  emulators, 35
  grouping test devices, 33–34
  mobile cloud testing, 35–36
  mobile device labs, 34–36
  prioritizing mobile device groups, 33–34
  privacy, 35
  renting test devices, 34–36
  security, 35
  simulators, 35
Frank, 134
Freemium business model, 23–24
Fully charged battery, testing, 63–64
Functional testing, 56, 162

Gathering customer information, 30
Gecko, 47
General Packet Radio Service (GPRS), 13
Generations of mobile devices, 11–12
Geographical regions, adapting to. *See* L10n (localization).
Gesture support for Android, 125
Gestures
  detecting, 40
  testing, 41–42
Gherkin for Android, 126
GHUnit, 134
Global App Testing, 145
Google, Our Mobile Planet, 34
Google Best Practices, 57
Google Glass, 204–205
Google Instrumentation Test Runner, 128
Google Play native App Beta Testing, 138
Google Play Store, 25
Google Play store Guidelines, 26
Google Test Automation Conference, 195
Google wearables, 204
GPRS (General Packet Radio Service), 13
GPS coordinates, detecting, 39–40
GPS location, setting for Android, 122
Grouping test devices, 33–34

GSM (Global System for Mobile Communications), 12
Guidelines. *See also* Checklists; Mind maps; Mnemonics.
  Amazon Appstore Guidelines, 26
  Android, 81
  app development, 2–3
  app stores, review and publishing, 26
  Apple App Store Guidelines, 26
  Apple iOS approval, 81
  BlackBerry, 81
  BlackBerry World Guidelines, 26
  for corporate look-and-feel, 59
  Google Play store Guidelines, 26
  platform-specific mobile testing, 81–82
  test automation, 114–115
  testing connected cars, 202
  Windows Phone, 81
Guidelines, accessibility testing
  for Android, 62
  for iOS, 62
  WC3 initiative, 62
  for Windows Phone, 63
Gyroscope sensor, testing, 38

Hardware usage testing, 192
Heads, cell phones, 12
Health apps, 205–207
Hearing impairment. *See* Auditory impairment.
Help. *See* Books and publications; Online resources.
HockeyApp, 138
HockeyKit, 139
HomeKit, 201
How Google Tests Software, 196
Howe, Jee, 141
HTC Dream smartphone, 14
HTML5, 22
Humidity sensor, testing, 39
Hybrid apps, 20–21

I SLICED UP FUN (Inputs, Store, Location, Interactions/interruptions, Communication, Ergonomics, Data, Usability, Platform, Function, User scenarios, Network), 90–92, 170
I18n (internationalization), 45–46
IEEE (Institute of Electrical and Electronics Engineers), 82
Ignite, 195
Image recognition tools, 109
iMobiTrax, 177
In-app purchases, 24
The Inmates Are Running the Asylum, 30
Inputs, Store, Location, Interactions/interruptions, Communication, Ergonomics, Data, Usability, Platform, Function, User scenarios, Network (I SLICED UP FUN), 90–92, 170
Installation testing, 71–72
Instruments, energy usage function, 65
Internationalization (I18n), 45–46
Internet connections, testing, 4
Internet Explorer Mobile Web browser *vs.* other browsers, 48
Internet of Things (IoT). *See* IoT (Internet of Things).
Interrupt testing, 66–68
In-the-wild testing, 167–169
iOS
  accessibility testing, 62
  approval guidelines, 81
  design guide, 18
  design resources, 81
  generating pseudo-random test events, 67
  simulators, 52
iOS apps
  test automation. *See* Test automation, iOS.
  testing criteria, 103

iOS smartphone
  vs. Android, 17–19
  battery status, displaying, 18
  dock section, 18
  generating pseudo-random test events, 67
  Instruments energy usage function, 65
  interrupt testing, 67–68
  logging energy usage, online resource, 66
  network strength, displaying, 18
  notifications, displaying, 18
  pinning apps, 18
  status bar, 18
  stress testing, 67–68
  time, displaying, 18
  UI AutoMonkey tool, 67–68
  user interface, 18
ios-driver, 132–133
IoT (Internet of Things)
  Android Auto, 201
  Apple CarPlay, 201
  Apple HomeKit, 201
  connected car, 201–203
  connected home, 200–201
  mobile testing, 199–203
  Nest Labs, 200
  overview, 199–200
ISTQB class, 55

Jamo Solutions
  Android, 129
  iOS, 134
Janky, 137
Java static analysis tools, 54
JavaScript, 22
Jenkins, 137
Johnson, Karen Nicole
  mind maps, 93–96
  mnemonics, 88
JouleUnit, energy profiling tool, 65
JSON Wire Protocol
  compatibility with Selendroid, 124
  and ios-driver, 132–133
JUnit tool, 54

Kaner, Cem, 183
Kelly, Mike, 89
Keyboard input, testing, 44
Keyboards, testing, 187–188
Keynote
  for Android, 129
  for iOS, 134
  mobile cloud testing, 149
Keynote Mobile Testing, 36
KIF (Keep It Functional), 133
KIFSU (Keep It Fast Simple Usable) principle, 3
KISS (Keep It Simple, Stupid) principle, 3, 102
Kiwi, 134
Kohl, Jonathan, 90, 92

L10n (localization), 45–46
Language
  in bug reports, 100
  switching, for Android, 122
Languages
  adapting to. *See* I18n (internationalization); L10n (localization).
  fallback, 46
Launch strategy. *See* Mobile launch strategy.
Lessons Learned in Software Testing, 196
Let's Test, 195
Lint tool, 54
Local storage testing, 76
Localization (L10n), 45–46
Location sensor, testing, 39–40
Log files
  in bug reports, 101–102
  checking, 82–83

Logging energy usage in iOS, 66
Login task, testing, 190
Long touch gesture, 41
Low battery, testing, 64
LTE (Long Term Evolution), 13

Magazines. *See* Books and publications.
Magnetic fields, detecting, 38–39
Magnetic sensor, testing, 38–39
Manual testing, mobile test strategy, 162, 167–169
Maximum device scope, 160
Metrics, mobile test strategy, 162
mHealth apps, testing, 205–207
Microphones, testing, 42–43
Microsoft Store, 26
Mind maps, 93, 94–96
Ministry of Testing, 93
Mnemonics. *See also* Checklists; Guidelines; Mind maps.
  COP FLUNG GUN (Communication, Orientation, Platform, Function, Location, User scenarios, Notifications, Guidelines, Gesture, Updates, Network), 92
  FCC CUTS VIDS (Feature tour, Complexity tour, Claims tour, Configuration tour, User tour, Testability tour, Scenario tour, Variability tour, Interoperability tour, Data tour, Structure tour), 89–90
  I SLICED UP FUN (Inputs, Store, Location, Interactions/interruptions, Communication, Ergonomics, Data, Usability, Platform, Function, User scenarios, Network), 90–92
  mobile test strategy, 170
  overview, 87
  SFDPOT (Structure, Function, Data, Platform, Operations, Time), 88
Mob4Hire, 145
Mobile, definition, 11
Mobile App Europe, 195
Mobile Cheat Sheet, 79
Mobile cloud testing
  handling fragmentation, 35–36
  mobile device test clouds, 35–36
Mobile cloud testing providers, 149–150
Mobile community, tester skills improvement, 194–195
Mobile device, in bug reports, 99
Mobile device labs, 34–36
Mobile device test clouds
  emulators, 35
  grouping test devices, 33–34
  mobile cloud testing, 35–36
  mobile device labs, 34–36
  prioritizing mobile device groups, 33–34
  privacy, 35
  renting test devices, 34–36
  security, 35
  simulators, 35
Mobile devices. *See also* Smartphones; *specific brands*.
  acquiring for testing, 6–8
  cell phones, 4–5
  dumb phones, 5
  feature phones, 5
  fragmentation, 5
  renting for testing, 7
  selecting for testing, 6
Mobile Frameworks Comparison Chart, 21
Mobile Labs, 36
Mobile launch strategy
  check release material, 172–174
  community support, 174–175
  crash reports, 176
  post-release, 174
  pre-launch, 172–174
  reviews, 175
  statistics, 176–177

timing of the release, 173
tracking, 176–177
Mobile networks, history of
AMPS (Advanced Mobile Phone Service), 12
AMTS (Advanced Mobile Telephone Service), 12
CDMA (Code Division Multiple Access), 12
EDGE (Enhanced Data rates for Global Evolution), 13
generations, 11–12
GPRS (General Packet Radio Service), 13
GSM (Global System for Mobile Communications), 12
LTE (Long Term Evolution), 13
Mobile Radio Telephone System, 12
MTD (Mobile Telephony System D), 12
MTS (Mobile Telephone Service), 12
NMT (Nordic Mobile Telephone), 12
OLT (Offentlig Landmobil Telefoni), 12
overview, 11–14
speed, 11–13
UMTS (Universal Mobile Telecommunications System), 13
WiMAX (Worldwide Interoperability for Microwave Access), 13
Mobile Personas, 30
Mobile platform, in bug reports, 99
Mobile QA Zone, 196
Mobile Radio Telephone System, 12
Mobile release cycles, 6–8
Mobile Security Testing, 79
Mobile Telephone Service (MTS), 12
Mobile Telephony System D (MTD), 12
Mobile test strategy
automated testing, 162
black box testing, 164
configuration data, 165
defining requirements, 156–158
FCC CUTS VIDS, 170
functional testing, 162
I SLICED UP FUN, 170
in-the-wild testing, 167–169
manual testing, 162, 167–169
mnemonics, 170
mobile checklists, 169–170
mobile tours, 169–170
nonfunctional testing, 162–163
overview, 155–156
possible metrics, 162
product risks, 172
regression testing, 162
stable data, 165
target devices, selecting, 166–167
target group information, gathering, 157–158
temporary data, 165
test automation, 170–171
test data, 164–166
test environment, selecting, 166–167
test levels, 161–163
test techniques, 163–164
usage scenarios, 168
white box testing, 163
Mobile test strategy, scope of testing
maximum device scope, 160
multidevice scope, 159–160
single-device scope, 159
use-case scope, 160–161
Mobile testing. *See also* Quality assurance measures; *specific tests*.
acquiring devices for, 6–8
vs. automated testing, 52–53
batteries. *See* Battery usage, testing.
be creative, 83
boundary value testing, 55

Mobile testing *(continued)*
cameras, 43–44
checking log files, 82–83
CI (continuous integration), 136–137
cloud testing, 147–152
connected car, 201–203
for connected cars, 202
connected home, 200–201
crowd testing, 141–147
databases, 75
decision tables, 55
documenting results. *See* Bug reports.
ETSI (European Telecommunications Standards Institute) standards, 82
functional testing, 56
generating pseudo-random test events for iOS smartphone, 67
generic checklist for, 85–87
gestures, 41–42
hardware usage, 192
IEEE (Institute of Electrical and Electronics Engineers), 82
installation testing, 71–72
interrupt testing, 66–68
IoT (Internet of Things), 199–203
keyboard input, 44
keyboards, 187–188
local storage testing, 76
login task, 190
mHealth apps, 205–207
microphones, 42–43
mind maps, 93, 94–96
on multiple platforms, 187–192
vs. other technologies, 1–2, 8
payment task, 190
for people with disabilities. *See* Accessibility testing.
performance. *See* Performance testing.
permissions, 190–191
platform-specific guidelines, 81–82
preinstalled apps, 44
pseudo-random test events for Android, 66
quality of Internet connections, 4
in real life environments, 3–4
remote locations, 4
renting devices for, 7
security. *See* Security testing.
selecting devices for, 6
smart glasses, 204–205
sound inputs, 42–43
standards conformance, 82
standby testing, 71
stress testing, 66–68
system apps, 44
Testing Map, 93
touchscreen, 41–42
uninstall testing, 72
user interface. *See* Usability testing.
voice commands, 42–43
W3C (World Wide Web Consortium), 82
wearables, 203–204
weather conditions, 4
widgets, 187–189
Mobile testing, mnemonics. *See also* Checklists; Guidelines; Mind maps.
COP FLUNG GUN (Communication, Orientation, Platform, Function, Location, User scenarios, Notifications, Guidelines, Gesture, Updates, Network), 92
FCC CUTS VIDS (Feature tour, Complexity tour, Claims tour, Configuration tour, User tour, Testability tour, Scenario tour, Variability tour, Interoperability tour, Data tour, Structure tour), 89–90
I SLICED UP FUN (Inputs, Store, Location, Interactions/interruptions, Communication,

Ergonomics, Data, Usability, Platform, Function, User scenarios, Network), 90–92
overview, 87
SFDPOT (Structure, Function, Data, Platform, Operations, Time), 88
Mobile testing, sensors and interfaces
acceleration sensor, 38
ambient light sensor, 37
changes in device movement, detecting, 38
device orientation, detecting, 38
gestures, detecting, 40
GPS coordinates, detecting, 39–40
gyroscope sensor, 38
humidity sensor, 39
location sensor, 39–40
magnetic fields, detecting, 38–39
magnetic sensor, 38–39
possible tests, 40–41
pressure sensor, 39
proximity sensor, 37–38
screen brightness, 37
temperature sensor, 39
touchless sensor, 40
Mobile testing, update testing
adb tool, 74–75
for Android apps, 74–75
for iOS apps, 73–74
overview, 72
typical scenarios, 73
Mobile Tools, 79
Mobile tours, 169–170
Mobile tracking tools, 177
Mobile Usability, 57
Mobile Web browsers
comparison of, 48
layout engines, 47
MobileAppTracking, 177
Mobile-device-specific information, in bug reports, 99
Mobility and data networks, 3–4
Monkey tool, 66
MonkeyTalk
for Android, 129
coordinate-based recognition, 110
for iOS, 134
OCR/text recognition, 111
Moolya, 92
MTD (Mobile Telephony System D), 12
MTS (Mobile Telephone Service), 12
Multidevice scope, 159–160
Multitouch gesture, 42

Native apps, description, 19–20
Native Object Recognition, 111
Neotys, 149
Nest Labs, 200
Network condition and environment, in bug reports, 100
Network strength, displaying, 18
99tests, 145
NMT (Nordic Mobile Telephone), 12
Nonfunctional testing, 162–163
Notifications, displaying, 18

OCMock, 134
OCR/text recognition, 110–111
OLT (Offentlig Landmobil Telefoni), 12
1G networks, 12
Online resources. *See also* Books and publications.
5G mobile standard, 13
99tests, 145
adjust, 177
Amazon Appstore Guidelines, 26
Android design guide, 18
Android Emulator, 51
Android guidelines, 81
App Quality Alliance site, 103
app rejection, reasons for, 18
app store review and publishing guidelines, 26
Appaloosa, 138
AppBlade, 138
Appium, 134

Online resources *(continued)*
  Applause, 138, 145
  Apple App Store, 25
  Apple App Store Guidelines, 26
  Apple iOS approval guidelines, 81
  Apple iOS design resources, 81
  Apple Watch, 204
  "Applying the SFDPOT Heuristic to Mobile Testing," 88
  appsfire, 177
  AppsFlyer, 177
  AppThwack, 36, 149
  Appurify, 149
  Bamboo, 137
  battery usage, 65–66
  Beta by Crashlytics, 138
  beta distribution tools, 138–139
  BirdFlight, 138
  BlackBerry guidelines, 81
  BlackBerry World, 26
  BlackBerry World Guidelines, 26
  Blink, 47
  Books about SQLite, 75
  Bots, 137
  Buildbot, 137
  CDMA (Code Division Multiple Access), 12
  CI systems, 137
  Clicktale, 177
  CloudMonkey, 36, 149
  crowd testing providers, 145
  crowdsourcedtesting, 145
  CruiseControl, 137
  CSS3, 22
  EDGE (Enhanced Data rates for Global Evolution), 13
  eggPlant, 129
  Energy Usage Instrument, 65–66
  Espresso, 128
  Experitest, 129, 134, 149
  Frank, 134
  gathering customer information, 30
  Gecko, 47
  GHUnit, 134
  Global App Testing, 145
  Google Best Practices, 57
  Google Instrumentation Test Runner, 128
  Google Play native App Beta Testing, 138
  Google Play Store, 25
  Google Play store Guidelines, 26
  Google wearables, 204
  GPRS (General Packet Radio Service), 13
  GSM (Global System for Mobile Communications), 12
  HockeyApp, 138
  HockeyKit, 139
  HTML5, 22
  iMobiTrax, 177
  iOS design guide, 18
  iOS simulators, 52
  ios-driver, 133
  Jamo Solutions, 129, 134
  Janky, 137
  JavaScript, 22
  Jenkins, 137
  Keynote, 129, 134, 149
  Keynote Mobile Testing, 36
  Kiwi, 134
  logging energy usage in iOS, 66
  Microsoft Store, 26
  Mob4Hire, 145
  mobile cloud testing providers, 149–150
  mobile device test cloud, 36
  Mobile Frameworks Comparison Chart, 21
  Mobile Labs, 36
  Mobile Personas, 30
  mobile tracking tools, 177
  mobile usability, 57
  mobile usability design principles, 57
  mobile user interface design principles, 57
  MobileAppTracking, 177
  MonkeyTalk, 129, 134
  Neotys, 149

OCMock, 134
Open Device Lab, 36
operating system market shares, 16
Our Mobile Planet, 34
passbrains, 145
Pebble, 204
Perfecto Mobile, 36, 129, 134, 150
PhoneGap framework, 20
platform-specific guidelines, 81–82
Presto, 47
Ranorex, 129, 134, 150
Roboelectric, 129
Robotium project pages, 122
Robotium wiki page, 122
Sauce Labs, 36, 150
Siesta, 129
Silk Mobile, 129, 134
SOASTA, 129, 134
Specta, 134
Subliminal, 134
TeamCity, 137
TenKod EZ TestApp, 129
TestBirds, 145
TestChameleon, 150
testCloud, 145
Testdroid, 150
TestFlight, 139
testing criteria for Android and iOS, 103
testing tools, 54
Testmunk, 36, 150
TestObject, 36, 129, 150
TestPlus, 145
TouchTest, 36
Travis CI, 137
Trident, 47
UI Automator, 129
UMTS (Universal Mobile Telecommunications System), 13
usability heuristics, 57
usability principles and techniques, 57
usability testing, 57
wearables, 204
Web browser layout engines, 47
WebKit, 47
widgets, 18
WiMAX (Worldwide Interoperability for Microwave Access), 13
Windows Phone guidelines, 81
Xamarin Test Cloud, 36, 150
XCTest, 134
Zucchini, 134
Online resources, for testers
Agile Testing Days, 195
Association for Software Testing, 196
Belgium Testing Days, 195
blogs, 196–197
books and publications, 196–197
communities, 196
conferences, 195–196
Dutch Testing Days, 195
EuroSTAR, 195
Google Test Automation Conference, 195
Ignite, 195
Let's Test, 195
Mobile App Europe, 195
Mobile QA Zone, 196
Øredev, 195
Software Testing Club, 196
STAREAST, 196
STARWEST, 196
TestBash, 196
TestExpo, 196
Testing Circus, 196
uTest community, 196
Open Device Lab, 36
Opera Mobile Web browser *vs.* other browsers, 48
Operating systems
the big two, 16–19. *See also* Android; iOS.
in bug reports, 99
relative market shares, 16
Øredev, 195
Our Mobile Planet, 34
Overheating, batteries, 64

OWASP (Open Web Application Security Project), 77
OWASP Mobile Security Page
  Mobile Cheat Sheet, 79
  Mobile Security Testing, 79
  Mobile Tools, 79
  Top 10 Mobile Risks, 80

Paid business model, 24
passbrains, 145
Payment task, testing, 190
Pebble, 204
Perfecto Mobile
  for Android, 129
  coordinate-based recognition, 110
  for iOS, 134
  mobile cloud testing providers, 150
  mobile device test cloud, 36
Performance testing
  critical areas, 69
  mobile apps, 70–71
  overview, 68–69
  reproducing issues, 189–190
Permissions, testing, 190–191
Personas, 30–32
PhoneGap framework, 20
Physical impairment, accessibility testing, 61
Pinch open/close gestures, 42
Pinning apps, 18
Platform-specific guidelines for testing, 81–82
PMD tool, 54
Post-release, mobile launch strategy, 174
Preinstalled apps, testing, 44
Pre-launch, mobile launch strategy, 172–174
Pressure sensor, testing, 39
Presto, 47
Prioritizing mobile device groups, 33–34
Privacy, 35
Process focus, 53–55
Product focus, 53–55
Product risks, mobile test strategy, 172
Professional Tester, 197
Programming language, choosing, for iOS tests, 132–133
Programming skills, tester skill, 185
Proximity sensor, testing, 37–38
Pyramid. *See* Test automation pyramid.

Quality assurance measures. *See also* Mobile testing.
  black box testing, 54–55
  dynamic testing, 53–55
  finding bugs, 53–55
  Java static analysis tools, 54
  process focus, 53–55
  process focus phase, 54
  product focus, 53–55
  protecting from bugs, 53–55
  static testing, 53–55
  white box testing, 54

Ranorex
  for Android, 129
  for iOS, 134
  mobile cloud testing providers, 150
Real devices, pros and cons of testing, 116–117
Reda, Renas, 122
Regression testing, mobile test strategy, 162
Rejecting apps, reasons for, 18
Release cycles. *See* Mobile release cycles.
Release material, checking, 172–174
Renting devices for testing, 7
Renting test devices, 34–36
Reproducible, in bug reports, 99
Requirements definition, mobile test strategy, 156–158
Research in Motion, 14
Resistive touchscreens, 41

Review and publishing guidelines, app stores, 26
Reviews, mobile launch strategy, 175
Roboelectric, 129
Robotium
  for Android, 121–122
  native object recognition, 111
  OCR/text recognition, 111
  recording test scripts, 122–123
Robotium Extensions, 122–123
Robotium project pages, 122
Robotium Recorder, 122–123
Robotium wiki page, 122
Rotate gesture, 42
Rotating touchscreens, 42
RoutineBot, 110

Safari Web browser vs. other browsers, 48
Samsung app store, 25
Sauce Labs, 36, 150
Scope of testing
  maximum device scope, 160
  multidevice scope, 159–160
  single-device scope, 159
  use-case scope, 160–161
Screen brightness, testing, 37
Screens. *See* Touchscreens.
Screenshot or video, in bug reports, 101
Security
  handling fragmentation, 35
  mobile device test clouds, 35
Security testing
  cache storage, 78
  client-side validation, 78
  common app security mistakes, 77–79
  encryption, 78
  fuzzing, 77
  overview, 76–77
  penetration testing, 77
  security requirements analysis, 79–80
  summary, 80–81
  testing tools, 77
  Top 10 Mobile Risks, 80
  unencrypted communications, 78
  unintended local storage, 78
  unnecessary app permissions, 78
SeeTest, 111
Selendroid
  for Android, 124–125
  native object recognition, 111
Selenium, 111
Selling advertisements, 24
Sensors, testing. *See* Mobile testing, sensors and interfaces.
Severity of error, in bug reports, 101
SFDPOT (Structure, Function, Data, Platform, Operations, Time), 88
Sherry, Rosie, 93–96
Siesta, 129
Sikuli, 110
Silk Mobile
  for Android, 129
  for iOS, 134
Simulators
  vs. emulators, 51–52
  handling fragmentation, 35
  iOS, 52
  mobile device test clouds, 35
  pros and cons, 115–116
  vs. real devices, 51–52
Single-device scope, 159
Smart glasses, 204–205
Smart watches, 203–204
Smartphones. *See also* Mobile devices; *specific brands*.
  common features, 5
  external components, 15–16
  hardware components, 15
  history of, 14–16
  market penetration, 5
  post-2007, 14–16
SOASTA
  for Android, 129
  for iOS, 134

Software build version, in bug reports, 100
Software Testing Club, 196
Sound inputs, testing, 42–43
Specification by Example, 196
Specta, 134
Speed of mobile networks, history of, 11–13
Spoon, 123–124
Square, 123, 133
Stable data, mobile test strategy, 165
Standards conformance, testing, 82
Standby testing, 71
STAREAST, 196
STARWEST, 196
Static analysis tools, 54
Static testing, 53–55
Statistics, mobile launch strategy, 176–177
Status bar, 18
Steps to reproduce, in bug reports, 98
Strategy
  for product launch. *See* Mobile launch strategy.
  for testing. *See* Mobile test strategy.
Stress testing, 66–68
Structure, Function, Data, Platform, Operations, Time (SFDPOT), 88
Subliminal, 134
Swipe gesture, 41
System apps, testing, 44

Tap gesture, 41
Tap into Mobile Application Testing, 92, 196
Target device selection, mobile test strategy, 166–167
Target group information gathering, mobile test strategy, 157–158
TeamCity, 137
Tea-time with Testers, 197
Technical skills, tester skill, 185
Temperature sensor, testing, 39
Temporary data, mobile test strategy, 165
Tenacity, tester skill, 183
TenKod EZ TestApp
  for Android, 129
  native object recognition, 111
Test automation
  cloud-based, 115–116, 152
  emulators, pros and cons, 115–116
  goals for, 113–114
  guidelines for, 114–115
  vs. manual testing, 52–53
  mobile test strategy, 162, 170–171
  vs. mobile testing, 52–53
  real devices, pros and cons, 116–117
  selecting tasks for, 113–115
  simulators, pros and cons, 115–116
Test automation pyramid
  default configuration, 105–106
  flipped configuration, 106–108
  mobile test configuration, 108–109
Test automation tools
  Appium, 111
  beta distribution, 138–139
  Calabash, 111
  capture and replay, 111–112
  coordinate-based recognition, 110
  eggPlant, 110, 111
  Espresso, 111
  image recognition, 109
  MonkeyTalk, 110, 111
  Native Object Recognition, 111
  OCR/text recognition, 110–111
  overview, 109
  Perfecto Mobile, 110
  recommendations, 112–113
  Robotium, 111
  RoutineBot, 110
  SeeTest, 111
  selection criteria, 118–120
  Selendroid, 111
  Selenium, 111
  Sikuli, 110

TenKod EZ TestApp, 111
TestObject, 110, 111
Test automation tools, Android
Advanced User Interactions API, 125
Appium, 127–128
automating UI acceptance tests, 125
Calabash for Android, 125–126
Cucumber, 125–126
eggPlant, 129
Espresso, 128
Experitest, 129
ExtSolo, 122–123
gesture support, 125
Gherkin, 126
Google Instrumentation Test Runner, 128
Jamo Solutions, 129
Keynote, 129
MonkeyTalk, 129
overview, 120–121
Perfecto Mobile, 129
Ranorex, 129
recommendations, 129–130
recording test scripts, 122–123
Roboelectric, 129
Robotium, 121–122
Robotium Extensions, 122–123
Robotium Recorder, 122–123
Selendroid, 124–125
setting GPS location, 122
Siesta, 129
Silk Mobile, 129
SOASTA, 129
Spoon, 123–124
switching current language, 122
TenKod EZ TestApp, 129
TestObject, 129
toggling Wi-Fi on and off, 122
UI Automator, 129
UI Automator Viewer, 121
Test automation tools, iOS
Appium, 134
Calabash for iOS, 131–132
choosing a programming language, 132–133
Experitest, 134
Frank, 134
GHUnit, 134
ios-driver, 132–133
Jamo Solutions, 134
Keynote, 134
KIF (Keep It Functional), 133
Kiwi, 134
MonkeyTalk, 134
OCMock, 134
Perfecto Mobile, 134
Ranorex, 134
Silk Mobile, 134
SOASTA, 134
Specta, 134
Subliminal, 134
UI Automation, 129–130
XCTest, 134
Zucchini, 134
Test data
in bug reports, 100
mobile test strategy, 164–166
Test environment selection, mobile test strategy, 166–167
Test levels, mobile test strategy, 161–163
Test scripts, recording, 122–123
Test strategy. *See* Mobile test strategy.
Test techniques, mobile test strategy, 163–164
TestBash, 196
TestBirds, 145
TestChameleon, 150
testCloud, 145
Testdroid, 150
Testers
blogs, 196–197
books and publications, 196–197
communities, 196
conferences, 195–196
named in bug reports, 102
success factors, 207–209

Testers, required skills
  communication, 179–181
  constant learning, 183–184
  creativity, 184
  critical thinking, 181–182
  curiosity, 181
  customer focus, 184–185
  programming skills, 185
  technical skills, 185
  tenacity, 183
Testers, skills improvement
  crashes on specific devices, 187–192
  engage with the mobile community, 194–195
  learn from other apps, 186
  observe other people, 192–193
  participate in testing competitions, 193–194
  Twitter, 194
TestExpo, 196
TestFlight, 139
Testing apps. *See* Mobile testing.
Testing Circus (community), 196
Testing Circus (magazine), 197
Testing competitions, tester skills improvement, 193–194
Testing criteria for Android and iOS, 103
Testing Map, 93
Testing tools
  adjust, 177
  appsfire, 177
  AppsFlyer, 177
  for automation. *See* Test automation tools.
  Checkstyle, 54
  Clicktale, 177
  FindBugs, 54
  iMobiTrax, 177
  JUnit, 54
  Lint, 54
  for mobile tracking, 177
  MobileAppTracking, 177
  PMD, 54
  static analysis, 54
  TestNG, 54
  white box testing, 54
  XCTest, 54
Testmunk, 36, 150
TestNG tool, 54
TestObject
  for Android, 129
  image recognition, 110
  mobile cloud testing providers, 150
  mobile device test cloud, 36
  OCR/text recognition, 111
TestPlus, 145
3G networks, 13
Time, displaying, 18
Timing of the release, 173
T-Mobile G1 smartphone, 14
Tools. *See* Online resources; Testing tools; *specific tools*.
Top 10 Mobile Risks, 80
Touch gestures, 41–42
Touchless sensor, testing, 40
Touchscreens
  capacitive, 41
  effects of weather conditions, 42
  mobile testing, 41–42
  resistive, 41
  rotating, 42
  testing, 41–42
  touch gestures, 41–42
  types of, 41
TouchTest, 36
Tracking, mobile launch strategy, 176–177
Tracking tools, 177
Transaction business model, 24
Transceivers, cell phones, 12
Travis CI, 137
Trident, 47
Twitter, tester skills improvement, 194
2G networks, 12–13

UI acceptance tests, automating, 125
UI Automation, 129–130

UI Automator, 129
UI Automator Viewer, 121
UMTS (Universal Mobile Telecommunications System), 13
Uninstall testing, 72
Update testing
  adb tool, 74–75
  for Android apps, 74–75
  for iOS apps, 73–74
  overview, 72
  typical scenarios, 73
Uploading apps to app stores, 26
Usability
  heuristics, 57
  The Inmates Are Running the Asylum, 30
  principles and techniques, 57
Usability testing
  allow mistakes, 58
  consistency, 59
  corporate identity guidelines for look-and-feel, 59
  details, checking, 58
  document your observations, 59–60
  inviting customers to, 31–32
  less is more, 57
  navigation, checking, 58
  overview, 56
  self-explanatory apps, 57
  usability and design patterns, 58
  verify text, 59
  workflows, checking, 58
Usability testing sessions, 31–32
Usage scenarios, mobile test strategy, 168
Usage statistics, battery life, 64–66
Use-case scope, 160–161
User interface
  Android *vs.* iOS, 18
  components, viewing properties of, 121
  design principles for mobile devices, 57
  testing. *See* Mobile testing, sensors and interfaces.
Users, expectations. *See also* Customers.
  overview, 2–3
  priority app characteristics, 2
  reasons for deleting apps, 2
uTest community, 196

Visual impairment, accessibility testing, 60–61
Voice commands, testing, 42–43

W3C (World Wide Web Consortium)
  accessibility guidelines initiative, 62
  conformance testing, 82
Wearables
  fitness wristbands, 203–204
  Google Glass, 204–205
  mobile testing, 203–204
  overview, 203
  smart glasses, 204–205
  smart watches, 203–204
Weather conditions
  effects on touchscreens, 42
  testing, 4
Web apps, description, 22–23
Web browser layout engines, 47
Web browsers. *See* Mobile Web browsers.
WebKit, 47
Weekend Testing, 193–194
White box testing
  mobile test strategy, 163
  tools for, 54
Widgets
  iOS *vs.* Android, 18
  testing, 187–189
Wi-Fi, toggling on and off, 122
WiMAX (Worldwide Interoperability for Microwave Access), 13
Windows Phone
  accessibility guidelines, 63
  accessibility testing, 63

Windows Phone *(continued)*
  app store, 26
  relative market share, 16
Windows Phone apps, downloading, 26
Work-around, in bug reports, 98
Worldwide Interoperability for Microwave Access (WiMAX), 13

Xamarin Test Cloud, 36, 150
XCTest, 134
XCTest tool, 54

0G networks, 12
Zucchini, 134