
# Programming Kotlin® Applications

## Building Mobile and Server-Side Applications with Kotlin

Brett McLaughlin

# PROGRAMMING KOTLIN® APPLICATIONS

# Programming Kotlin® Applications

# Programming Kotlin® Applications

## BUILDING MOBILE AND SERVER-SIDE APPLICATIONS WITH KOTLIN

**Brett McLaughlin**


A Wiley Brand

Published simultaneously in Canada

ISBN: 978-1-119-69618-6
ISBN: 978-1-119-69616-2 (ebk)
ISBN: 978-1-119-69621-6 (ebk)

Manufactured in the United States of America



**Library of Congress Control Number:** 2020947753

*for Leigh, as always, my person*

# ABOUT THE AUTHOR

**BRETT MCLAUGHLIN** has been working and writing in the technology space for over 20 years. Today, Brett's focus is squarely on cloud and enterprise computing. He has quickly become a trusted name in helping companies execute a migration to the cloud—and in particular Amazon Web Services—by translating confusing cloud concepts into a clear executive-level vision. He spends his days working with key decision makers who need to understand the cloud as well as leading and building teams of developers and operators who must interact with the ever-changing cloud computing space. He has most recently led large-scale cloud migrations for NASA's Earth Science program and the RockCreek Group's financial platform. Brett is currently the Chief Technology Officer at Volusion, an ecommerce platform provider.

# ABOUT THE TECHNICAL EDITOR

**JASON LEE** is a software developer happily living in the middle of the heartland. He has over 23 years of experience in a variety of languages, writing software running on mobile devices all the way up to big iron. For the past 15+ years, he has worked in the Java/Jakarta EE space, working on application servers, frameworks, and user-facing applications. These days, he spends his time working as a backend engineer, primarily using Kotlin, building systems with frameworks like Quarkus and Spring Boot. He is the author of *Java 9 Programming Blueprints*, a former Java User Group president, an occasional conference speaker, and a blogger. In his spare time, he enjoys spending time with his wife and two sons, reading, playing the bass guitar, and running. He can be found on Twitter at `twitter.com/jasondlee`, and on his blog at `jasondl.ee`.

# ACKNOWLEDGMENTS

**I USED TO WATCH MOVIES AND STARE** in amazement at the hundreds of names that scrolled by at the end. How could so many people be involved in a single movie?

Then I wrote a book. Now I understand.

Carole Jelen is my agent at Waterside, and she replied to an email and picked up the phone at a time when I really needed someone to help me find my way back into publishing. I'm incredibly grateful.

On the Wiley side, Brad Jones was more patient than he ever should have been. Thanks, Brad! Barath Kumar Rajasekaran handled a million tiny details, and Pete Gaughan and Devon Lewis kept the train on the tracks. Christine O'Connor handled production, and Jason Lee caught the technical mistakes in the text that you wouldn't want to stumble over. Seriously, Jason in particular made this a much better book with his keen eye.

As usual, it's an author's family that pays the highest price. Long days, more than a few weekends and evenings, and a constant support keep us going. My wife, Leigh, is the best, and my kids, Dean, Robbie, and Addie, always make finishing one of these a joy.

Let's do brunch, everyone! Mimosas and breakfast tacos are on me.

—Brett McLaughlin

# CONTENTS

# INTRODUCTION

For decades, the Java programming language has been the dominant force in compiled languages. While there have been plenty of alternatives, it's Java that has remained core to so many applications, from desktop to server-side to mobile. This has become especially true for Android mobile development.

Finally, though, there is a real contender to at least live comfortably beside Java: Kotlin, a modern programming language shepherded by JetBrains (`www.jetbrains.com`). It is not Java, but is completely interoperable with it. Kotlin feels a lot like Java, and will be easy to learn for developers already familiar with the Java language, but offers several nice improvements.

Further, Kotlin is a full-blown programming language. It's not just for mobile applications, or a visual language that focuses on one specific application. Kotlin supports:

- Inheritance, interfaces, implementations, and class hierarchies
- Control and flow structures, both simple and complex
- Lambdas and scope functions
- Rich support for generics while still preserving strong typing
- Idiomatic approaches to development, giving Kotlin a "feel" all its own

You'll also learn that while Kotlin is a new language, it doesn't feel particularly new. That's largely because it builds upon Java, and doesn't try to reinvent wheels. Rather, Kotlin reflects lessons that thousands of programmers coding in Java (and other languages) employ on a daily basis. Kotlin takes many of those lessons and makes them part of the language, enforcing strong typing and a strict compiler that may take some getting used to, but often produces cleaner and safer code.

There's also an emphasis in Kotlin, and therefore in this book, on understanding inheritance. Whether you're using packages from third parties, working with the standard Kotlin libraries, or building your own programs, you need a solid understanding of how classes interrelate, how subclassing works, and how to use abstract classes along with interfaces to define behavior and ensure that behavior is implemented. By the time you're through with this book, you'll be extremely comfortable with classes, objects, and building inheritance trees.

The Kotlin website (`kotlinlang.org`) describes Kotlin as "a modern programming language that makes developers happier." With Kotlin and this book, you'll be happier *and* more productive in your Kotlin programming.

## WHAT DOES THIS BOOK COVER?

This book takes a holistic approach to teaching you the Kotlin programming language, from a beginner to a confident, complete Kotlin developer. By the time you're finished, you'll be able to write Kotlin applications in a variety of contexts, from desktop to server-side to mobile.

### WILL THIS BOOK TEACH ME TO PROGRAM MOBILE APPLICATIONS IN KOTLIN?

Yes, but you'll need more than *just* this book to build rich mobile applications in Kotlin. Kotlin is a rich language, and while there are books on all the packages needed to build mobile languages, this is fundamentally a book on learning Kotlin from the ground up. You'll get a handle on how Kotlin deals with generics, inheritance, and lambdas, all critical to mobile programming.

You can then take these concepts and extend them into mobile programming. You can easily add the specifics of Android-related packages to your Kotlin base knowledge, and use those mobile packages far more effectively than if you didn't have the fundamentals down.

If you are anxious to begin your mobile programming journey sooner, consider picking up a book focused on Kotlin mobile programming, and hop back and forth. Read and work through Chapter 1 of this book, and then do the same for the book focused on mobile programming. You'll have to context switch a bit more, but you'll be learning fundamentals alongside specific mobile techniques.

This book covers the following topics:

**Chapter 1: Objects All the Way Down** This chapter takes you from getting Kotlin installed to writing your first Kotlin program. You'll learn about functions from the start, and how to interact with the command line through a not-quite "Hello, World!" application. You'll also immediately begin to see the role of objects and classes in Kotlin, and refine your understanding of what a class is, what an object is, and what an object instance is.

**Chapter 2: It's Hard to Break Kotlin** This chapter delves into one of the distinguishing features of Kotlin: its rigid stance on type safety. You'll learn about Kotlin's types and begin to grasp choosing the right type for the right task. You'll also get familiar with `val` and `var` and how Kotlin allows for change.

**Chapter 3: Kotlin Is Extremely Classy** Like any object-oriented language, much of your work with Kotlin will be writing classes. This chapter digs into classes in Kotlin and looks at the basic building blocks of all Kotlin objects. You'll also override some functions and get deep into some of the most fundamental of Kotlin functions: `equals()` and `hashCode()`.

**Chapter 4: Inheritance Matters** This chapter begins a multichapter journey into Kotlin inheritance. You'll learn about Kotlin's constructors and the relatively unique concept of secondary constructors. You'll also learn more about the `Any` class, understand that inheritance is truly essential for all Kotlin programming, and learn why writing good superclasses is one of the most important skills you can develop in all your programming learning.

**Chapter 5: Lists and Sets and Maps, Oh My!** This chapter moves away (briefly) from classes and inheritance to add Kotlin collections to your arsenal. You'll use these collection classes over and over in your programming, so understanding how a `Set` is different from a `Map`, and how both are different from a `List`, is essential. You'll also dig further into Kotlin mutability and immutability—when data can and cannot change—as well as a variety of ways to iterate over collections of all types.

**Chapter 6: The Future (in Kotlin) Is Generic** Generics are a difficult and nuanced topic in most programming languages. They require a deep understanding of how languages are built. This chapter gets into those depths, and provides you more flexibility in building classes that can be used in a variety of contexts than possible without generics. You'll also learn about covariance, contravariance, and invariance. These might not be the hot topics at the water cooler, but they'll be key to building programs that use generics correctly, and also level up your understanding of inheritance and subclassing.

**Chapter 7: Flying through Control Structures** Control structures are the bread and butter of most programming languages. This chapter breaks down your options, covering `if` and `else`, `when`, `for`, `while`, and `do`. Along the way, you'll focus on controlling the flow of an application or set of applications all while getting a handle on the semantics and mechanics of these structures.

**Chapter 8: Data Classes** This chapter introduces data classes, another very cool Kotlin concept. While not specific to only Kotlin, you'll find that data classes offer you a quick and flexible option for representing data more efficiently than older languages. You'll also really push data classes, going beyond a simple data object and getting into constructors, overriding properties, and both subclassing with and extending from data classes.

**Chapter 9: Enums and Sealed, More Specialty Classes** This chapter introduces enums, a far superior approach to `String` constants. You'll learn why using `String`s for constant values is a really bad idea, and how enums give you greater flexibility and type safety, as well as making your code easier to write. From enums, you'll move into sealed classes, a particularly cool feature of Kotlin that lets you turbo-charge the concept of enums even further. You'll also dig into companion objects and factories, all of which contribute to a robust type-safe approach to programming where previously only `String` types were used.

**Chapter 10: Functions and Functions and Functions** It may seem odd to have a chapter this late in the book that purports to focus on functions. However, as with most fundamentals in any discipline, you'll have to revisit the basics over and over again, shoring up weaknesses and adding nuance. This chapter does just that with functions. You'll dig more deeply into just how arguments really work, and how many options Kotlin provides to you in working with data going into and out of your functions.

**Chapter 11: Speaking Idiomatic Kotlin** Kotlin, like all programming languages, has certain patterns of usage that seasoned programmers revert to time and time again. This chapter discusses these and some of the idioms of Kotlin. You'll get a jump start on writing Kotlin that looks like Kotlin is "supposed to" all while understanding how you have a tremendous amount of flexibility in choosing how to make your Kotlin programs feel like "you."

**Chapter 12: Inheritance, One More Time, with Feeling** Yes, it really is another chapter on inheritance! This chapter takes what you've already learned about abstract classes and superclasses and adds interfaces and implementations into the mix. You'll also learn about the delegation pattern, a common Kotlin pattern that helps you take inheritance even further with greater flexibility than inheritance alone provides.

**Chapter 13: Kotlin: The Next Step** No book can teach you everything you need to know, and this book is certainly no exception. There are some well-established places to look for next steps in your Kotlin programming journey, though, and this chapter gives you a number of jumping-off points to continue learning about specific areas of Kotlin.

## Reader Support for This BookCompanion Download Files

As you work through the examples in this book, the project files you need are available for download from `www.wiley.com/go/programmingkotlinapplications`.

## How to Contact the Publisher

If you believe you've found a mistake in this book, please bring it to our attention. At John Wiley & Sons, we understand how important it is to provide our customers with accurate content, but even with our best efforts an error may occur.

In order to submit your possible errata, please email it to our Customer Service Team at `wileysupport@wiley.com` with the subject line "Possible Book Errata Submission."

## How to Contact the Author

We appreciate your input and questions about this book! Email me at `brett@brettdmclaughlin.com`, or DM me on Twitter at `@bdmclaughlin`.