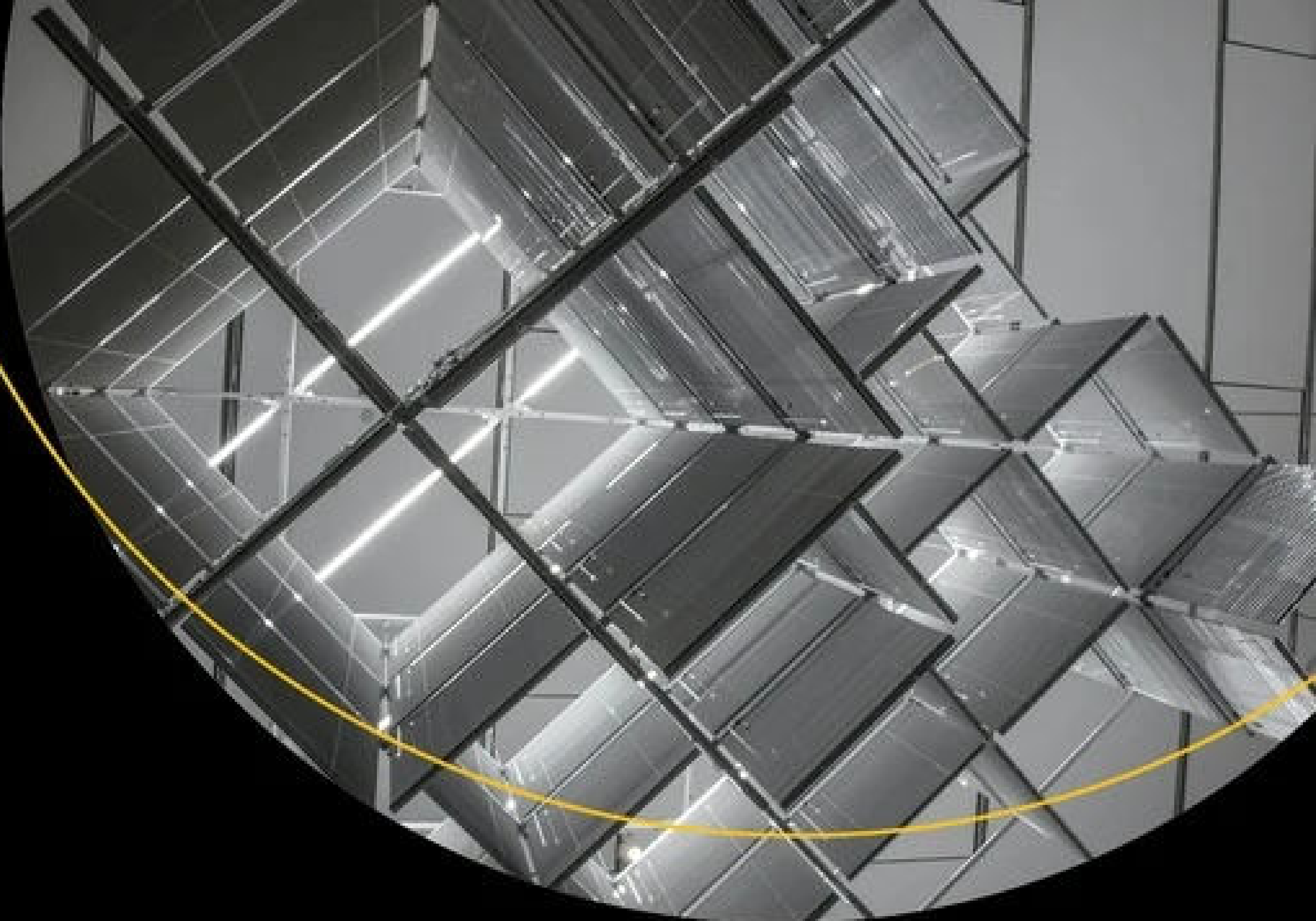


Pro Kotlin Web Apps from Scratch

Building Production-Ready Web Apps Without a Framework

—

August Lilleaas

Apress®

Pro Kotlin Web Apps from Scratch

Building Production-Ready Web Apps Without a Framework

August Lilleaas

Apress®

Pro Kotlin Web Apps from Scratch: Building Production-Ready Web Apps Without a Framework

August Lilleaas
Oslo, Norway

ISBN-13 (pbk): 978-1-4842-9056-9 ISBN-13 (electronic): 978-1-4842-9057-6
https://doi.org/10.1007/978-1-4842-9057-6

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Jonathan Gennick
Development Editor: Laura Berendson
Coordinating Editor: Jill Balzano

Cover Photo by George Tasios on Unsplash

Distributed to the book trade worldwide by Springer Science+Business Media LLC, 1 New York Plaza, Suite 4600, New York, NY 10004. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail booktranslations@springernature.com; for reprint, paperback, or audio rights, please e-mail 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 GitHub (https://github.com/Apress). For more detailed information, please visit http://www.apress.com/source-code.

Printed on acid-free paper

Dedicated to my mother and father, who made it all happen,

and

in loving memory of Ann-Cecilie Saltnes (11.03.1980–04.07.2022)

Table of Contents

About the Author

August Lilleaas has been building web apps, user interfaces, and real-time systems since 2004 and mobile apps since the app stores opened in the late 2000s. After picking up Clojure in 2012, he left the frameworks and ORMs behind and started to build web apps from scratch and has shipped to production using Clojure, Groovy, Node.js, Elixir, and Kotlin. He has worked as a consultant for a decade and is now an independent contractor and startup founder.

About the Technical Reviewer

Andres Sacco has been a professional developer since 2007, working with a variety of languages, including Java, Scala, PHP, Node.js, and Kotlin. Most of his background is in Java and the libraries or frameworks associated with it, like Spring, JSF, iBATIS, Hibernate, and Spring Data. He is focused on researching new technologies to improve the performance, stability, and quality of the applications he develops.

In 2017 he started to find new ways to optimize the transference of data between applications to reduce the cost of infrastructure. He suggested some actions, some of them applicable in all the microservices and others in just a few of them. As a result of these actions, the cost was reduced by 55%. Some of these actions are connected directly with the bad use of the databases.

Acknowledgments

Thanks to my fiancée, Tina, who believes in me and lets me know. Thanks to Jonathan Gennick for giving me the opportunity to become a technical writer. Thanks to everyone else at Apress for being a pleasure to work with. Thanks to Marius Mårnes Mathisen for catapulting my career way back when, despite my complete lack of programming education and professional experience. Thanks to Kolbjørn Jetne for further catapulting my career, despite my complete lack of consulting experience. Thanks to Magnar Sveen, Christian Johansen, and Alf Kristian Støyle for all the discussions, input, and inspiration through the years. A special thanks to Finn Johnsen, who helped form most of the patterns in this book and was a vital fellow traveler in my initial quest for escaping the framework and bridging the gap between the old and the new.

Finally, thanks to my son, Tobias, and my daughter, Lone, who are awesome. And, at my daughter's request/demand, "Pappa er en bok."

Introduction

Welcome to *Pro Kotlin Web Apps from Scratch*! In this book, you'll learn how to build professional and production-grade web apps, completely from scratch, without the use of big and unwieldy frameworks.

My personal web app journey started with frameworks, but when I learned more about what goes on under the hood and grew weary of fighting framework bugs and limitations, I started teaching myself how to build web apps from scratch instead. As it turns out, frameworks aren't a requirement!

You're in good hands when you're building from scratch, thanks to modern programming languages like Kotlin and amazing third-party open source libraries. Back in the day, you needed thousands of lines of boilerplate and XML configuration to wire up a framework-less web app. No wonder people preferred frameworks! Nowadays, though, all you need is a couple of handfuls of explicit code, completely free of bloat and magic, that only does what you tell it to.

In Part I, you'll set up a web app skeleton, completely from scratch. This code base forms the basis for Part II, where you'll learn a handful of patterns and practical solutions that build on the skeleton from Part I. Part III covers how to choose the right library and Kotlin tips and tricks that I didn't get to cover in the preceding parts. Finally, three appendixes explain how to replace some of the libraries chosen in Parts I and II, to demonstrate that you have free rein to choose different libraries than me and still write pro Kotlin web apps from scratch.

I've deliberately kept the chapter about Kotlin tips and tricks in Part III as short as I've been able to manage. Instead, you'll learn Kotlin tips and tricks in Parts I and II, and you'll learn how to build web apps from scratch alongside explanations of the various Kotlin language constructs you're using.

CHAPTER 1

Setting Up a Development Environment

In this chapter you'll set up a development environment for Kotlin. You'll install IntelliJ IDEA, a Java Development Kit (JDK), you'll write a small Kotlin program, and you'll compile and run it using the Gradle build system.

This chapter explains how to do everything using IntelliJ IDEA. For the other chapters in this book, though, using IntelliJ IDEA is not a requirement. For example, both the Eclipse IDE (integrated development environment) and Visual Studio Code have Kotlin plugins available, made by the Kotlin community. If you choose to use another editor, make sure that it supports *auto-completion* and *automatic imports*. Extension functions are prevalent in Kotlin, and you must import them before you can use them. You'll avoid lots of tedious manual work if your editor can search and match all available extension functions (auto-completion) and automatically add an import statement when you select one of the auto-completed options (automatic imports).

Get Started with IntelliJ IDEA Community Edition

You'll need an editor to write your code and a way to compile and run your code. IntelliJ IDEA gives you both.

IntelliJ IDEA is the canonical integrated development environment (IDE) for static languages on the Java platform. The free community edition of IntelliJ IDEA has everything you need, both for this book and for production-grade Kotlin development. Additionally, JetBrains makes both Kotlin and IntelliJ IDEA, making IntelliJ IDEA uniquely well suited for Kotlin work.

A. Lilleaas, *Pro Kotlin Web Apps from Scratch*, https://doi.org/10.1007/978-1-4842-9057-6_1

Download and Run IntelliJ IDEA

You can download and use IntelliJ IDEA Community for free from JetBrains. Go to the IntelliJ IDEA download page, shown in Figure 1-1, at the address `www.jetbrains.com/idea/download/` and download the community edition from there. JetBrains provides a .dmg file for macOS, an .exe file for Windows, and a tarball with a shell script in *bin/idea.sh* for Linux.

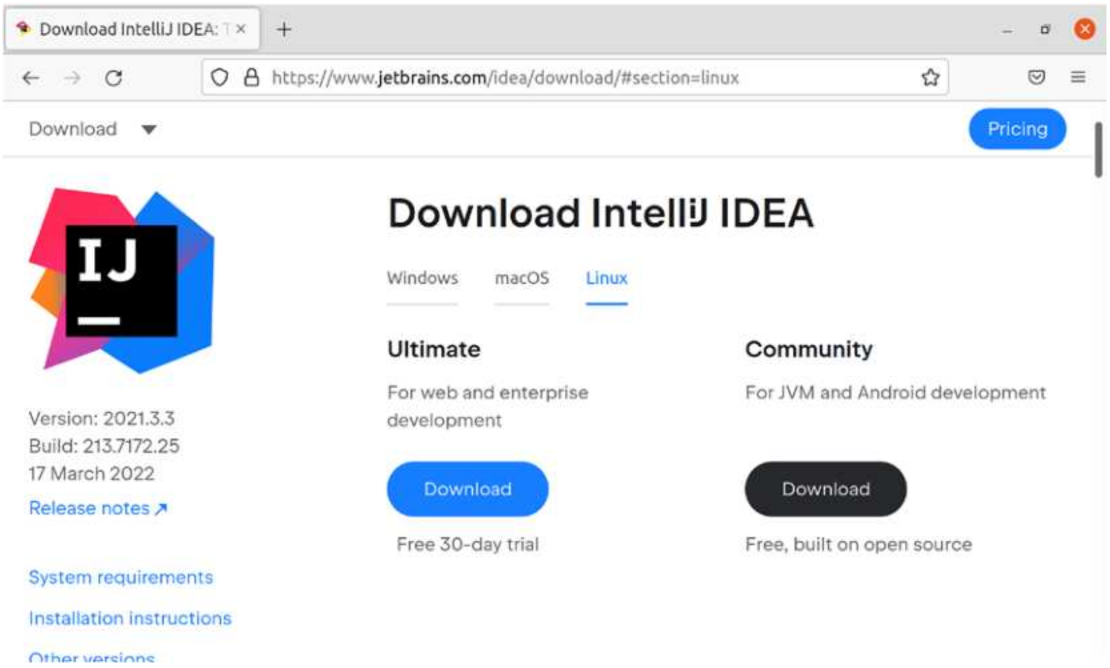


Figure 1-1. *Download IntelliJ IDEA Community Edition for your platform*

For more details on how to install and run IntelliJ IDEA, refer to the JetBrains website. For example, the download page has a link with installation instructions in the left sidebar, as seen in Figure 1-1.

Create a New Kotlin Project with IntelliJ IDEA

There are many ways to create a new Kotlin project. For this book, you'll be using IntelliJ IDEA to do it. It's nice to have a GUI to hold your hand through the process if you're not familiar with Kotlin and/or the JVM.

When you launch IntelliJ IDEA, you should see the splash screen shown in Figure 1-2.

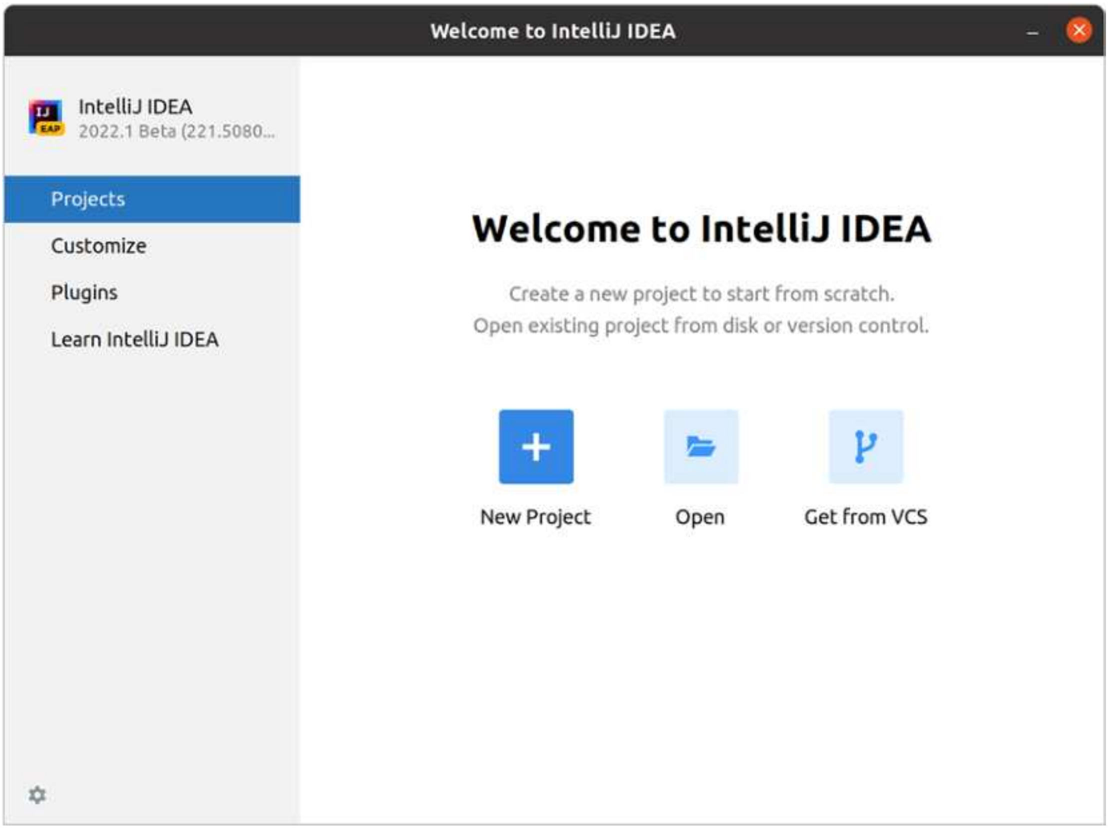
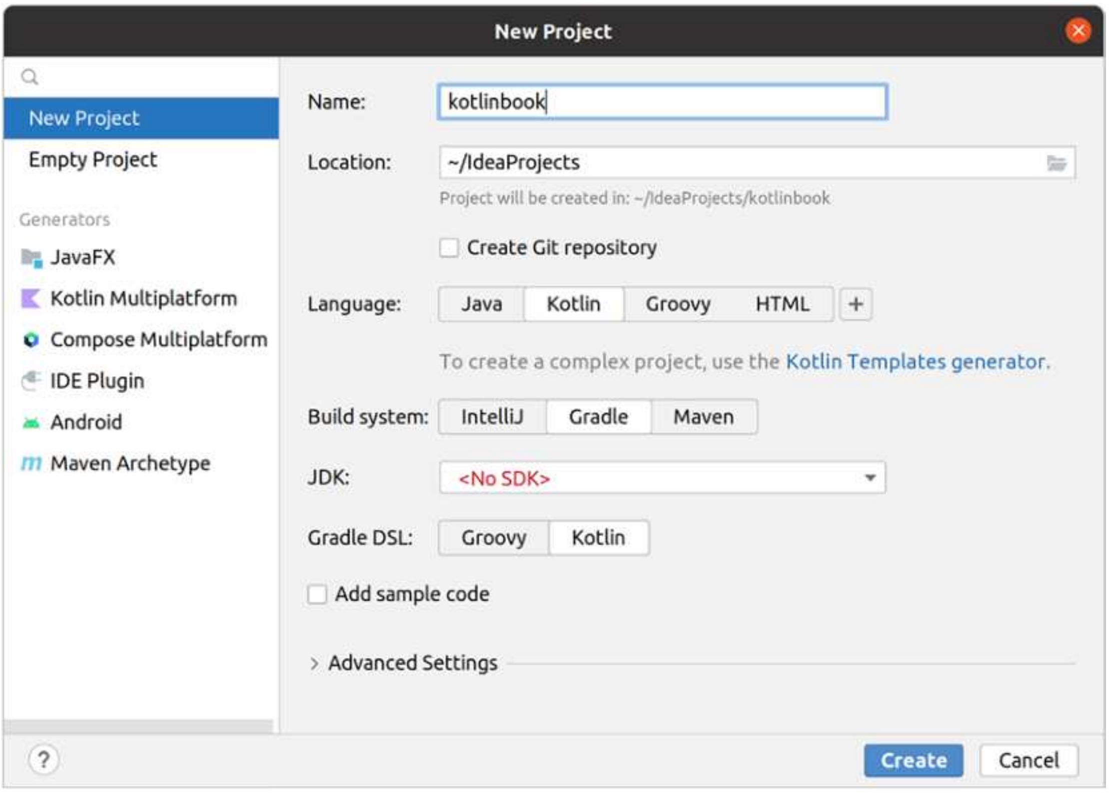


Figure 1-2. *Your first taste of IntelliJ IDEA*

Click “New Project” to create a new Kotlin project using the built-in wizard in IntelliJ IDEA, shown in Figure 1-3.



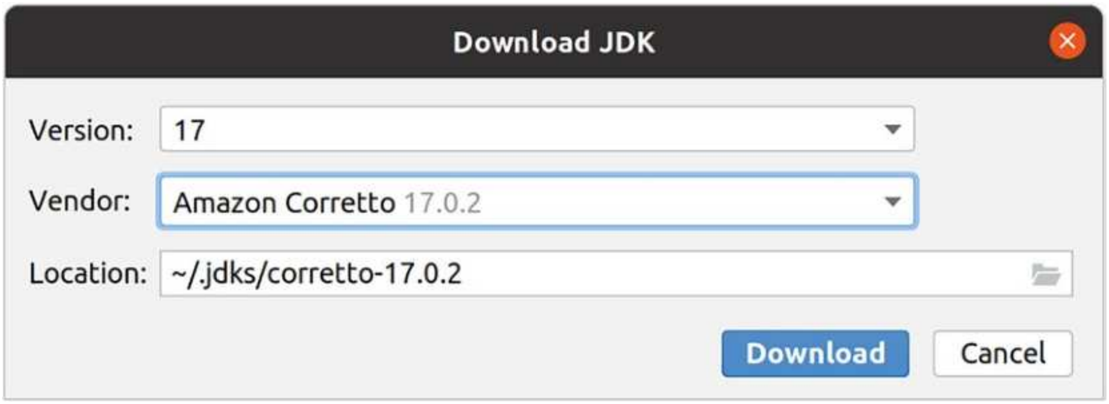
Name: kotlinbook
Location: ~/IdeaProjects
Project will be created in: ~/IdeaProjects/kotlinbook
Create Git repository
Language: Java | Kotlin | Groovy | HTML | +
To create a complex project, use the Kotlin Templates generator.
Build system: IntelliJ | Gradle | Maven
JDK: <No SDK>
Gradle DSL: Groovy | Kotlin
Add sample code

Figure 1-3. *The New Project screen in IntelliJ IDEA*

Use your good taste and judgment to name the project, and adjust the options as follows:

- Make sure you select "New Project" in the left sidebar.
- Set *Language* to *Kotlin*.
- Set *Build system* to *Gradle*.
- Set *Gradle DSL* to *Kotlin*.
- Uncheck *Add sample code*.

You need a JDK, and the easiest way to get one is to have IntelliJ IDEA download one for you. Click the red "<No SDK>" dropdown in the New Project dialog shown in Figure 1-3 and choose an appropriate JDK from the popup that appears, shown in Figure 1-4. JDK 17 or JDK 11 is a good choice, as those are stable Long-Term Support (LTS) releases.



Version: 17
Vendor: Amazon Corretto 17.0.2
Location: ~/.jdks/corretto-17.0.2

Figure 1-4. *Use IntelliJ IDEA to download a JDK*

There are multiple vendors available. They are all full-fledged JDKs that adhere to the JDK specifications. For the most part, it doesn't matter which one you choose. The differences lie in a handful of edge cases and different defaults. For most web apps, you'll be able to switch between vendors and not notice any differences. I personally prefer Amazon Corretto, mostly out of habit. But I've used other vendors in real-world web apps, such as Azul Zulu. I tend to choose whichever vendor is most convenient on the platform I use, such as Azul Zulu on Azure, Amazon Corretto on AWS, and so on.

Make sure you choose a JDK of version 11 or newer. Versions 11 and 17 are both Long-Time Support (LTS) releases, which means they'll receive security updates longer than other versions. And later in this book, you'll use libraries that require at least version 11 to work.

You can also choose an existing JDK if you have one installed. Alternatively, you can manage JDK installations yourself, by using something like sdkman (`https://sdkman.io/`) on macOS and Linux and Jabba (`https://github.com/shyiko/jabba`) on Windows.

Tip You can have bad luck and choose a JDK version that's incompatible with the Gradle version used by IntelliJ IDEA. For example, the 2021 version of IntelliJ uses Gradle 7.1, which only works with JDK versions lower than 16. If this happens, delete your project and start over, using a different JDK or Gradle version.

After IntelliJ IDEA finishes downloading the JDK, click the "Create" button at the bottom of the New Project dialog shown in Figure 1-3, and your brand-new project appears, as shown in Figure 1-5.

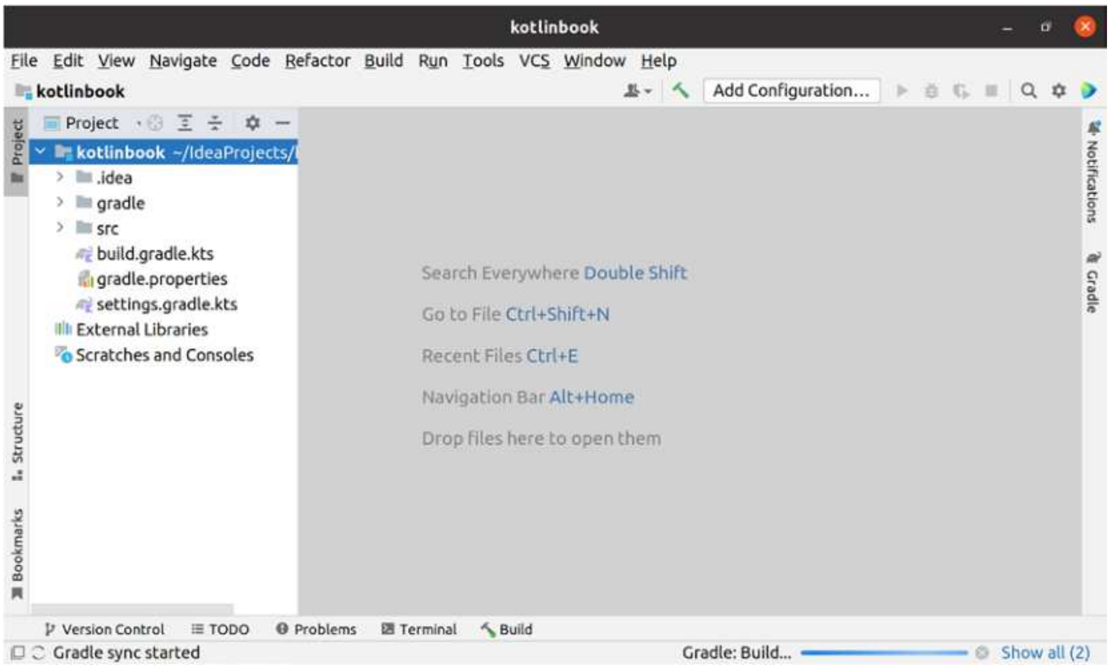


Figure 1-5. *IntelliJ IDEA after it has finished with project initialization*

With this setup, IntelliJ IDEA won't create sample code or pre-generated files, other than a Gradle build system skeleton. All you have now is a way to compile and run Kotlin code. And that's all you need in this book. There's no framework with tens of thousands of lines of code that needs to run first to get anything done.

Tip Gradle is a build system for the Java platform. This is where you add third-party code as dependencies, configure production builds, set up automated testing, and more. Maven is also an excellent choice for building production-grade Kotlin web apps. Most examples you'll see in books and online documentation use Gradle, though, and I've only used Gradle in real-world Kotlin apps, so that's why I use it in this book.
