Preface

As an application's code base increases, it becomes harder for developers to maintain existing features and introduce new ones. In this clean architecture book, you'll learn how to identify when and how this problem emerges and how to structure your code to overcome it.

The book starts by explaining clean architecture principles and Android architecture components and then explores the tools, frameworks, and libraries involved. You'll learn how to structure your application in the Data and Domain layers, the technologies that go in each layer, and the role that each layer plays in keeping your application clean. You'll understand how to arrange the code into these two layers and the components involved in assembling them. Finally, we'll cover the Presentation layer and the patterns that can be applied to have a decoupled and testable code base.

By the end of this book, you'll be able to build an application following clean architecture principles and have the knowledge you need to maintain and test the application easily.

Who this book is for

This book is for Android developers who want to learn about managing the complexity of their applications and is also highly recommended for intermediate or advanced Android developers looking for a go-to guide for clean architecture and the integration of various Android technologies. New developers familiar with the fundamentals of Android app development will find this book useful, too.

What this book covers

Chapter 1, *Getting Started with Clean Architecture*, starts by presenting the evolution of Android apps with regards to how business logic was structured, and the problems caused by these approaches. It will then transition to how certain patterns were applied to tackle these issues, revealing other sets of issues. Finally, the concept of clean architecture will be introduce, as well as how its principles can be used to solve some of the problems presented previously.

Chapter 2, *Deep Diving into Data Sources*, covers what Android tools and frameworks are available to use with regard to the implementation of the data layer and details and expands on the ones that will be used later in the book, such as Kotlin flows and coroutines, Retrofit, Room, and DataStore.

Chapter 3, *Understanding Data Presentation on Android*, covers what Android tools and frameworks are available to use with regard to the implementation of the presentation layer and will detail and expand on the ones that will be used later in the book, such as Android ViewModel and Jetpack Compose.

Chapter 4, *Managing Dependencies in Android Applications*, provides a quick overview of dependency injection and how it works. It briefly explores some of the dependency injection tools available for Android development, ending with the Hilt dependency injection framework, about which it goes into a more detailed explanation because it will be used in many of the exercises in the book.

Chapter 5, *Building the Domain of an Android Application*, describes how to build a domain layer and what components are part of this layer. You will learn about entities and use cases or interactors and what roles they play when it comes to designing the architecture of your application.

Chapter 6, *Assembling a Repository*, covers the Data layer and the responsibilities this layer has when it comes to managing an application's data, and how it can use the Repository pattern to achieve this.

Chapter 7, *Building Data Sources*, continues the exploration into the Data layer and some examples of data sources that can be defined in Android. You will learn about using remote data sources to load data from various servers as well as local data sources such as Room and DataStore.

Chapter 8, *Implementing an MVVM Architecture*, presents the MVVM architecture pattern and how it can be used in an application's presentation layer. You will learn how to use the Android ViewModel and LiveData to build an app with MVVM and integrate use cases into the ViewModel.

Chapter 9, *Implementing an MVI Architecture*, presents the MVI architecture pattern and how it can be used in an application's presentation layer. You will learn how to use Kotlin flows and Android ViewModel to implement the MVI pattern.

Chapter 10, *Putting It All Together*, covers the benefits of clean architecture by analyzing an example of an application that implements the concepts and then adding instrumentation tests with Espresso and Jetpack Compose. The introduction of UI tests serves as a good example of how we can inject and change certain behaviors in the application for testing purposes without needing to modify the application's code.

To get the most out of this book

You'll need the Android Studio IDE installed on your computer (version Arctic Fox 2020.3.1 Patch 3 or above) and Java 8 to be installed. Using later versions of Java such as Java 11 might cause errors when building some of the exercises. Knowing how to trigger builds on an emulator or device and Gradle Synchronizations from Android Studio is recommended before attempting the exercises presented in the book.

Software/hardware covered in the book	Operating system requirements
Android SDK 21-32	Windows, macOS, or Linux
Java 8	
Kotlin 1.5.31	

You can expand on the final exercise of the book by optimizing the way the data is loaded, introducing in-memory caches, or integrating new network calls to fetch additional data for the users. You can also improve the instrumentation testing by adding interaction with the list of data and opening new screens and asserting that the correct data is displayed.

If you are using the digital version of this book, we advise you to type the code yourself or access the code from the book's GitHub repository (a link is available in the next section). Doing so will help you avoid any potential errors related to the copying and pasting of code.

Download the example code files

The code bundle for the book is also hosted on GitHub at https://github.com/PacktPublishing/Clean-Android-Architecture ↗️. If there's an update to the code, it will be updated on the existing GitHub repository.

We also have other code bundles from our rich catalog of books and videos available at https://github.com/PacktPublishing/ ↗️. Check them out!

Code in Action

The Code in Action videos for this book can be viewed at https://bit.ly/3LqAa30 ↗️

Download the color images

We also provide a PDF file that has color images of the screenshots and diagrams used in this book. You can download it here: https://static.packt-cdn.com/downloads/9781803234588_ColorImages.pdf ↗️

Conventions used

There are a number of text conventions used throughout this book.

`Code in text`: Indicates code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles. Here is an example: "Inside the **`resources`** folder, create a subfolder called **`mockito-extensions`**. Inside this folder, create a file named **`org.mockito.plugins.MockMaker`**, and inside this file, add the text **`mock-maker-inline`**."

A block of code is set as follows:

```
data class User(
    val id: String,
    val firstName: String,
    val lastName: String,
    val email: String
) {
    fun getFullName() = "$firstName $lastName"
}
```

When we wish to draw your attention to a particular part of a code block, the relevant lines or items are set in bold:

```
…
@Composable
fun Screen(viewModel: MainViewModel =
viewModel(factory = MainViewModelFactory())) {
    viewModel.uiStateLiveData.observeAsState().value?
.let {
```

```
            UserList(uiState = it)
        }
    }
    ...
```

Bold: Indicates a new term, an important word, or words that you see on-screen. For instance, words in menus or dialog boxes appear in **bold**. Here is an example: "Create a new project in Android Studio using an **Empty Compose Activity**."

TIPS OR IMPORTANT NOTES

Appear like this.

Get in touch

Feedback from our readers is always welcome.

General feedback: If you have questions about any aspect of this book, email us at customercare@packtpub.com and mention the book title in the subject of your message.

Errata: Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you have found a mistake in this book, we would be grateful if you would report this to us. Please visit www.packtpub.com/support/errata ↗️ and fill in the form.

Piracy: If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at copyright@packt.com with a link to the material.

If you are interested in becoming an author: If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, please visit authors.packtpub.com ↗️.

Share Your Thoughts

Once you've read *Clean Android Architecture*, we'd love to hear your thoughts! Please click here to go straight to the Amazon review page for this book and share your feedback.

Your review is important to us and the tech community and will help us make sure we're delivering excellent quality content.