What's in this course?

We'll cover the following

- What will you learn in this course?

What will you learn in this course?

Kotlin is a multi-paradigm programming language. You may write plain scripts in Kotlin, write object-oriented code, functional style code, program asynchronously, and more. To provide reasonable coverage to this broad spectrum of topics, this book is divided into multiple parts.

Here's what's covered in each chapter:

- In Chapter 2, Hello Kotlin, we look at the reasons to use Kotlin, download the necessary tools, and get started with writing code.
- Programmers coming to Kotlin from Java have a few practices and syntax to unlearn before they can jump into learning what's new and different in Kotlin. We'll cover those in Chapter 3, Kotlin Essentials for the Java Eyes.
- Functions are first-class citizens in Kotlin, and the language has so much to offer, like default and named arguments, and `varargs`. Explore these function-related capabilities in Chapter 4, Working with Functions.
- When programming in the imperative style, we often use external iterators. You'll see in Chapter 5, External Iteration and Argument Matching, how Kotlin's iterators make that task bearable and how the argument matching syntax removes a lot of noise from conditional statements.
- We work with collections extensively when programming. Chapter 6, Using Collections, will show you how to use view interfaces to work with the JDK collections from Kotlin.
- Kotlin has a sound type system, and its compile-time type checking goes

beyond what we have come to expect from statically typed languages. In Chapter 7, Type Safety to Save the Day, we'll look at Kotlin's fundamental types, nullable and non-nullable references, smart casts, generics variance, and more.

- Though semantically equivalent, creating classes in Kotlin is quite different than in Java. In Chapter 8, Objects and Classes, you'll learn to create singletons, classes, companion objects, and the reasons to use data classes.
- Kotlin's treatment of inheritance is a lot different from the way it's used in Java. Classes are final by default, and the language places some rules to improve type safety and compile-time checks. We'll explore this topic in depth in Chapter 9, Class Hierarchies and Inheritance.
- As one of the languages that has direct support for delegation, Kotlin provides a few built-in delegates and also makes it easier to create custom delegates. We'll start with a discussion of when and why to use delegation and then dive into using delegates in Chapter 10, Extension Through Delegation.
- In Chapter 11, Functional Programming with Lambdas, you'll learn how to create lambda expressions and how to write higher-order functions. We'll also cover the facilities offered in Kotlin to eliminate function call overhead and improve performance.
- The internal iterators offer fluency, and sequences give us efficiency. We'll apply the functional style to iterate and process collections of objects in Chapter 12, Internal Iteration and Lazy Evaluation.
- Chapter 13, Fluency in Kotlin, will show many of the capabilities of Kotlin to create concise, fluent, elegant, and expressive code.
- Chapter 14, Creating Internal DSLs, builds on the topic of fluency to create internal DSLs, to define your own syntax for your specialized language, but with full compile-time type safety.
- Kotlin is one of the few languages on the JVM that provides tail call optimization. We'll see that in action in Chapter 15, Programming Recursion and Memoization, along with using memoization to reduce computational complexity.
- Coroutines are a stable feature in `Kotlin 1.3` and, along with continuations,

provide the fundamental infrastructure for programming asynchronously.

The basics of coroutines and continuations are covered in Chapter 16, Exploring Coroutines.

- In Chapter 17, Asynchronous Programming, you'll apply coroutines to create practical applications that can benefit from asynchronous program execution.
- Kotlin can run on different platforms, including the Java Virtual Machine. In Chapter 18, Intermixing Java and Kotlin, you'll learn how to intermix Kotlin with Java; how to use Kotlin in modern versions of Java—that is, with Java modules; how to use it with Maven and Gradle; and also how to smoothly work with both Java and Kotlin within the same application.
- Even though the Kotlin compiler will catch several errors, automated testing is an essential practice for sustainable agile development. You'll learn about creating unit tests and measuring code coverage in Chapter 19, Unit Testing with Kotlin.
- In Chapter 20, Programming Spring Applications with Kotlin, we'll explore the Spring libraries geared toward Kotlin programmers and the unique capabilities these offer.
- Finally, in Chapter 21, Writing Android Applications with Kotlin, we'll use Kotlin to create an Android application that talks to a back-end service.

Now let's jump into the course. The next chapter provides an introduction to Kotlin.