## Wrapping Up

Automated testing is a key technical practice for sustainable agile development. You may use tools that you're used to in Java to write automated tests for Kotlin. Alternatively, you may use tools that are targeted specifically for Kotlin, to enjoy fluency and to be able to use idiomatic Kotlin style. In this chapter, we used KotlinTest, Mockk, and Jacoco to write tests, to mock dependencies, and create the code coverage report. In addition, we used Gradle and Maven to run the builds.

KotlinTest provides elegant syntax and fluent asserts, along with many different options to write tests. Mockk is a powerful mocking tool that can be used to stub and mock classes, singletons, companion objects, and top-level functions. Mockk also has facilities to test code that makes use of coroutines.

In this chapter we've explored, step by step, how to unit test a small program that makes asynchronous calls, using coroutines, to an external web service. We've written unit tests, mocked the dependencies, written integration tests, and finished by writing a driver program to exercise the code created using automated tests.

---

In the next chapter, we'll take a quick look at using Spring Boot with Kotlin.