Testing

Eigenheiten mit Kotlin

Kotlin mit Mockito

- In Kotlin: Klassen und Funktionen sind standardmässig final
- Mockito muss jedoch Klassen und Funktionen überschreiben
- Mockito erlaubt folgende Konfiguration um das zu umgehen:
  - Im 'resources'-Ordner eine Datei erstellen: "mockito-extensions/org.mockito.plugins.MockMaker"
  - Inhalt des files: "mock-maker-inline"

Exercise 1 – use Mockito

Schritt 1

Füge die Mockito library in der Modul-Gradle-Datei hinzu. testImplementation**"org.mockito:mockito-core:1.5.0"**

Schritt 2

Die nhaarman-mockito-kotlin ist ein Wrapper um Mockito für Kotlin welcher einen ausdrucksvolleren Ansatz erlaubt.
Ausserdem gibt Mockito Null-Werte zurück für Aufrufe von any. Dieses Problem wird mit dieser library behoben.
Füge die nhaarman-library hinzu: testImplementation"com.nhaarman:mockito-kotlin-kt1.1:2.12.0"

Schritt 3

Final Klassen überschreiben:
Im 'resources'-Ordner kann eine Datei erstellen: "mockito-extensions/org.mockito.plugins.MockMaker" mit dem Inhalt "mock-maker-inline"

Testing Exercise Lösung

TAG: 04_testing_TAG

Tag: 04_testing_TAG
Branch: step_04_testing

Integration-Testing Room Code Beispiel

Tag: 04_testing_room_TAG
Branch: step_04_testing_room

Alternatives Test-Framework: Spek

- Spek hat eine vorgegebene Struktur – Tests sind Spezifikationen
- Die Testbedingung, was der Test macht und was das erwartete Ergebnis ist, muss explizit angegeben werden.
- Wenn die Test durchlaufen wird eine klare Beschreibung angezeigt

```
object CalculatorSpec: Spek({
    given("a calculator") {
        val calculator = SampleCalculator()
        on("addition") {
            val sum = calculator.sum(2, 4)
            it("should return the result of adding the first number to the
second number") {
                assertEquals(6, sum)
            }
        }
    }
})
```

Nachteil Spek

- Spek enthält alle Tests als Funktionen im Konstruktor
- Gewisse Annotationen können nicht hinzugefügt werden
- LiveData kann nicht getestet werden, weil alles auf einem einzigen Thread laufen muss:

```
class MainViewModelTest {
    @Rule
    @JvmField
    val rule = InstantTaskExecutorRule()


    @Test
    fun someTest(){
    }
}
```


Die Annotation kann in Spek nicht hinzugefügt werden

Quellen

- https://medium.com/google-developers/7-steps-to-room-27a5fe5f99b2
- https://medium.com/google-developers/testing-room-migrations-be93cdb0d975
- https://pbochenski.pl/blog/07-12-2017-testing_livedata.html
- http://static.javadoc.io/org.mockito/mockito-core/2.12.0/org/mockito/Mockito.html#Mocking_Final

FAQ

- Wie überschreibt Mockito mit der ‚mock-maker-inline‘-Konfiguration die final Klassen und Funktionen von Kotlin?
  - Mockito benutzt java-agent -> bytecode
  - https://github.com/mockito/mockito/wiki/What%27s-new-in-Mockito-2

*”Mocking of final classes and methods is an **incubating**, opt-in feature. It uses a combination of Java agent instrumentation and subclassing in order to enable mockability of these types.”*