

應用軟體實習

任課教師：黃士嘉 老師

班級：**電子三甲**

學生：**郭若蘭**

**106360146**

**Instructions**

Kotlin語言

1. 實作Lab8 (Chapter 10 Service)

2. 實作Lab9 (Chapter 11 Broadcast receiver)

**Process**

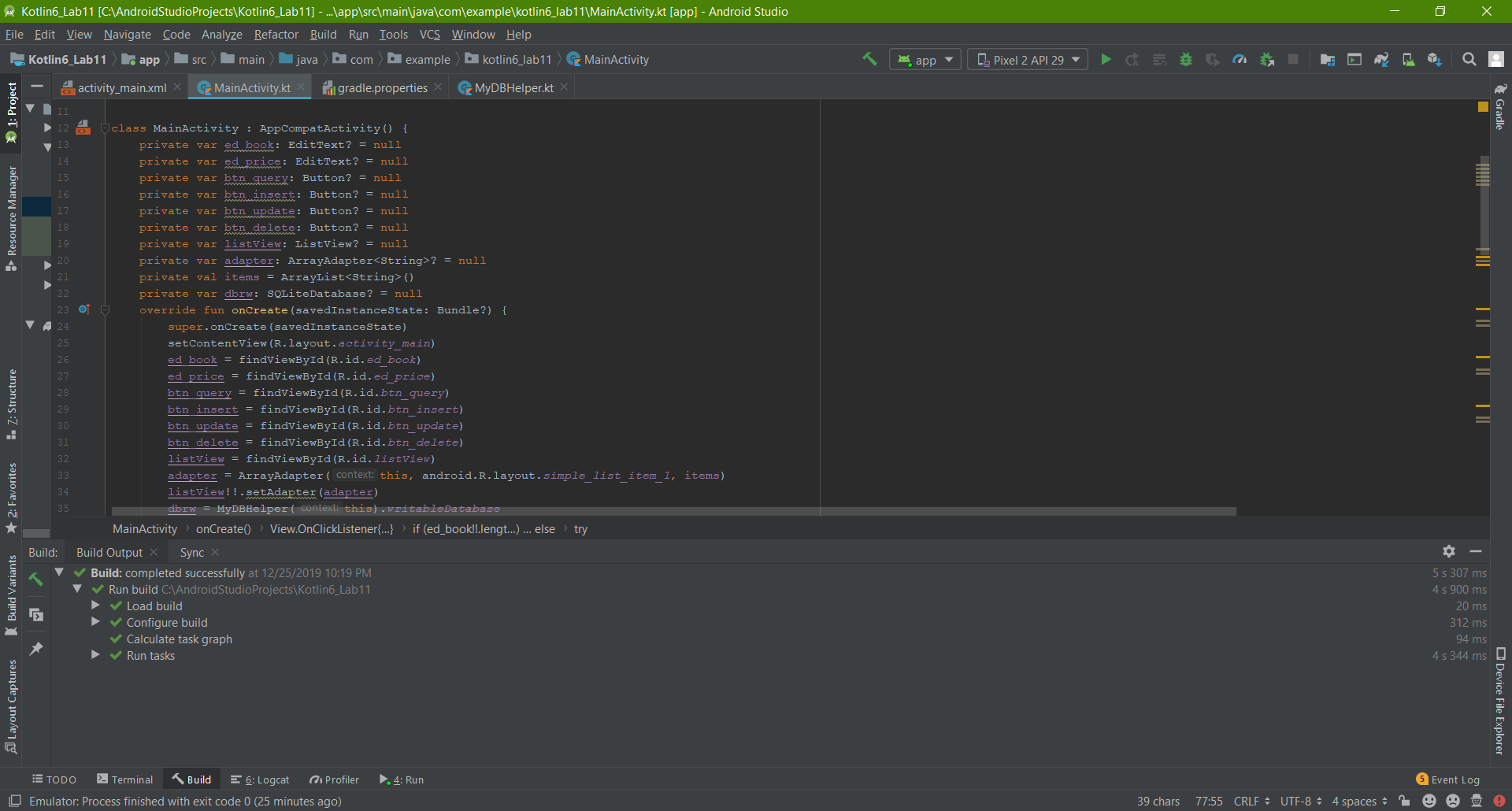
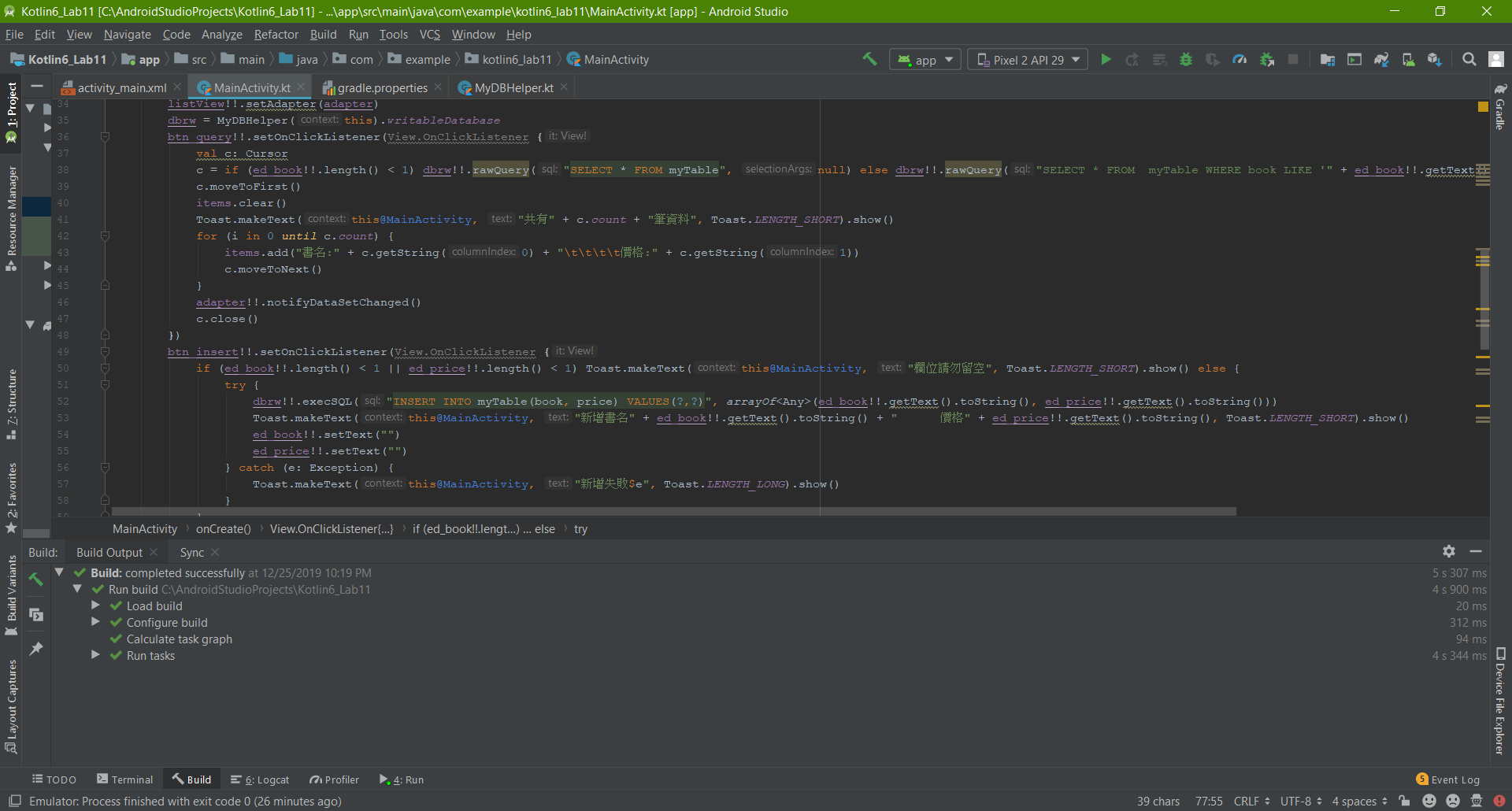
This time I found a very interesting feature of Kotlin that Java does not have:

Nullable types and Non-Null Types

One of the most common pitfalls in many programming languages, including Java, is that accessing a member of a null reference will result in a null reference exception. In Java this would be the equivalent of a NullPointerException or NPE for short.

Kotlin's type system is aimed to eliminate NullPointerException's from our code. The only possible causes of NPE's may be:

* An explicit call to throw NullPointerException();
* Usage of the !! operator that is described below;
* Some data inconsistency with regard to initialization, such as when:
  + An uninitialized this available in a constructor is passed and used somewhere ("leaking this");
  + A superclass constructor calls an open member whose implementation in the derived class uses uninitialized state;



Usage of the !! operator for making a *safe call*.

**Insight**

Although we have been comparing Java and Kotlin for a long time, there are always features that make these programming languages different from each other, but, at the same time, make us discover how Kotlin was designed to be a better version of Java, this version being specially for application development.

As you can see, the Nullable types and Non-Null Types are aimed to eliminate NullPointerException's from our code, which represent a big problem in Java coding.

# **References**

Kotlin. (n.d.). *Null Safety*. Retrieved from Nullable Types and Non-Null Types: https://kotlinlang.org/docs/reference/null-safety.html#the--operator

**GitHub:** https://github.com/106360146/KotlinHW6