ToiletPaper #120

Android Architecture Components

Autor: Stefan Baumgartner / Software Architect / Business Division Banking & Insurance

✕ Problem

Wie beim Hausbau ist auch in der Softwareentwicklung eine gute Architektur die Basis für den Erfolg eines Projekts. Da spätere Änderungen meistens aufwendig und teuer sind, sollte man möglichst früh eine gute Grundlage schaffen. Wie das Ganze in einer Android App aussehen kann, möchte ich hier kurz vorstellen.

✓ Lösung

Google hat mit der Einführung von Android Jetpack sehr viele Komponenten für die Entwicklung von Apps zusammengefasst. In diesem Paket findet man auch die sogenannten "Architecture Components". Diese nehmen einem nicht nur sehr viel Arbeit bei der Entwicklung ab, sondern werden auch im Rahmen des Guide to app architecture empfohlen. Dabei wird der Einsatz der "Architecture Components", anhand einer auf dem Model-View-ViewModel (MVVM) Pattern basierenden Architektur, demonstriert. Im Folgenden ein kleiner Überblick über die einzelnen Komponenten und deren Nutzen.

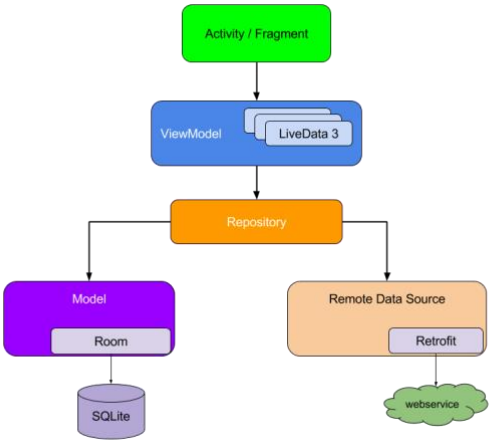

Abb. 1: Android Architecture Components (URL: https://developer.android.com/jetpack/docs/guide)

- Activity/Fragment: Activities und Fragments sind nicht Teil der "Architecture Components", sondern sind Grundkomponenten in der Entwicklung von Android Apps.
- ViewModel: Verwaltet die UI-relevanten Daten. Im Gegensatz zur Activity, lebt das ViewModel unabhängig vom Activity Lifecycle.
- LiveData: Daten-Container, der observable ist. Dadurch werden die Activity/Fragments automatisch über Datenänderungen informiert, ohne dass sie vom ViewModel explizit aufgerufen werden müssen. Dabei wird aber der Lebenszyklus der Komponente respektiert und somit Memory Leaks vermieden.
- Repository: Ist zwar keine eigene Android Komponente, wird aber als "best practice" empfohlen, da wir damit das ViewModel von der eigentlichen Datenquelle trennen können. Das Ganze ist vor allem dann hilfreich, wenn wir unsere Daten, wie in Abb 1. dargestellt, aus unterschiedlichen Quellen beziehen und synchronisieren müssen.
- Room: Ist eine Bibliothek, die den Zugriff auf die SQLite Datenbank kapselt und vereinfacht. Room arbeitet viel mit Annotations (z.B. @Entity, @Dao) und reduziert damit unnötigen Boilerplate Code.

➔ Beispiel

- https://bitbucket.org/Honkisponk/androidarchitectureexample/src/master/
- https://github.com/googlesamples/android-sunflower
- https://github.com/googlesamples/android-architecture-components/

✚ Weiterführende Aspekte

- https://developer.android.com/topic/libraries/architecture