ToiletPaper #120

Android Architecture Components

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

✕ Problem

As well known from building a house, a good architecture is the basis for the success of a project – also in software development. Since later changes are usually complex and expensive, a good basis should be created as early as possible. I would like to introduce briefly, how this can look like in an android app.

✓ Solution

With the introduction of Android Jetpack, Google has combined a lot of components for app development. In this package you will also find the so-called "architecture components". They do not only save us a lot of work during the development process, but are also recommended in the guide to app architecture. How to use these architecture components is demonstrated in an architecture based on the Model-View-ViewModel (MVVM) pattern. The following is a brief overview of the individual components and their benefits:

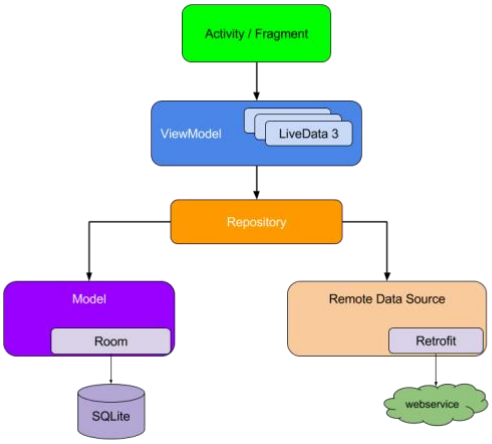

Fig. 1: Android Architecture Components (URL: https://developer.android.com/jetpack/docs/guide)

- Activity/Fragment: Activities and Fragments are not part of the "architecture components", but are basic components in the development of android apps.
- ViewModel: manages the UI-relevant data. Unlike the Activity, the ViewModel exists independently of the activity lifecycle.
- LiveData: is an observable data container which automatically notifies the Activity/Fragments of data changes without requiring an explicit call to the ViewModel. However, the lifecycle of the component is respected and thus memory leaks are avoided.
- Repository: recommended as "best practice", even if it's not an independent android component. It allows us to separate the ViewModel from the actual data source, which is especially helpful when we have to get and synchronize our data from different sources, as shown in figure 1.
- Room: is a library encapsulating and simplifying access to the SQLite database. Room works a lot with annotations (e.g. @Entity, @Dao) and thus reduces unnecessary boilerplate code.

→ Example

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

+ Further Aspects

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