medium.com

Modularization: Android room entities in a Java/Kotlin library module

Gowtham

2 minutes

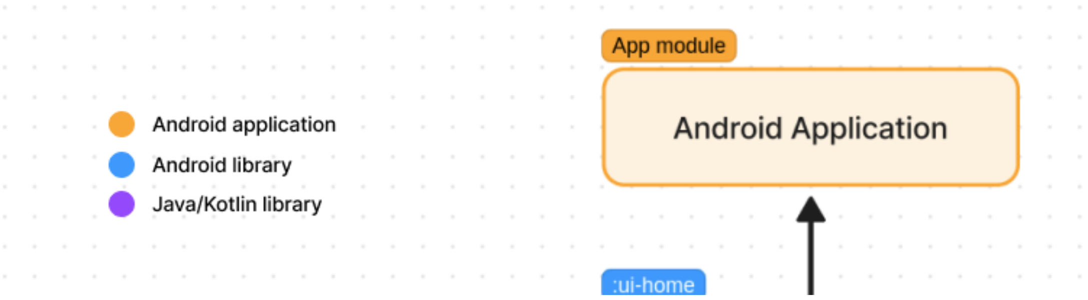
The room library makes ***persisting data locally*** a lot easier. If you are working on a multi-module android app that uses room, then you might have a bad time placing room entities in a **Java/Kotlin module**. We will see the issue in detail

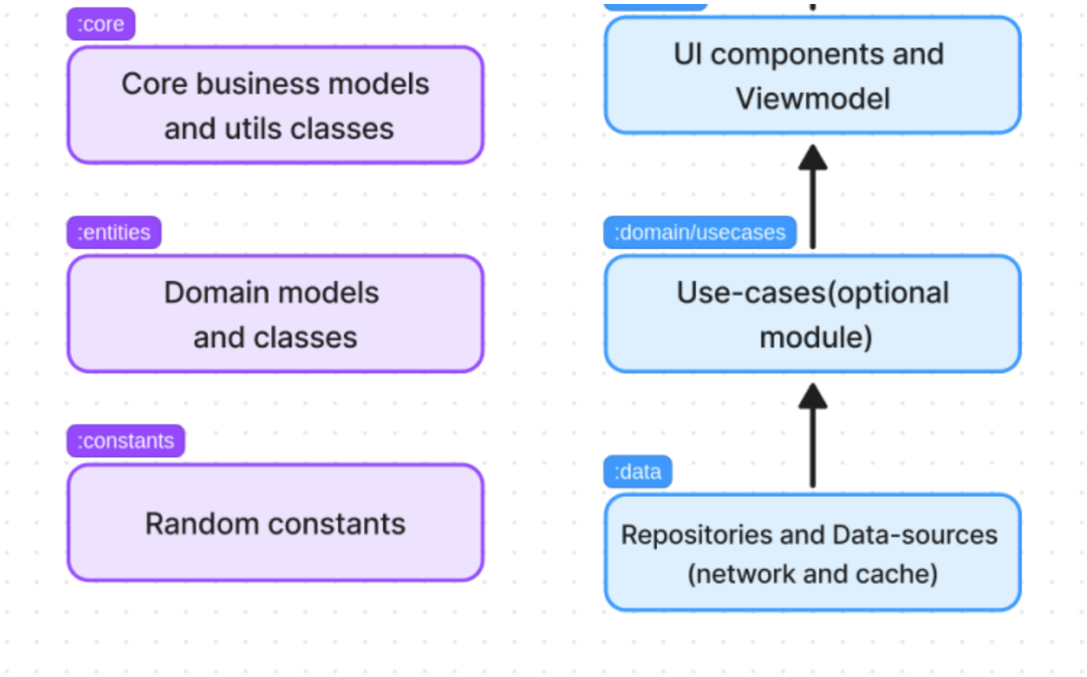
The Problem in Room Modularizing:

There are multiple ways to modularize the project.

1. By Feature
2. By Layer
3. Hybrid by MitchTabian

Let’s say your multi-module structure looks like this: Hybrid




:core
Core business models and utils classes
:entities
Domain models and classes
:constants
Random constants
UI components and Viewmodel
:domain/usecases
Use-cases(optional module)
:data
Repositories and Data-sources (network and cache)

my-awesome-app-module-structure.png

Here we use **Java/Kotlin module** wherever possible as it takes less build time.

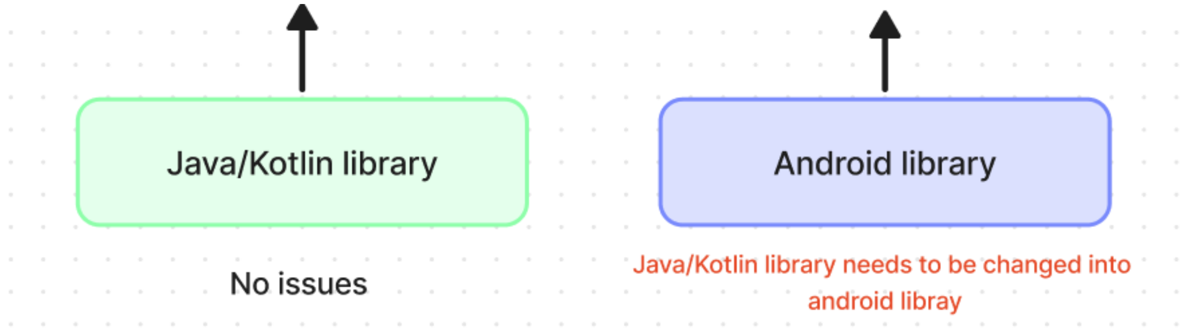
:data — is where the room database is created and it is an android module. because the room uses classes from android SDK.

:entities — holds data/model classes and room entities.

We tend to use it as Java/Kotlin module. but, the @Entity annotation is coming from **room-runtime** library which is basically an android SDK.

So, we are forced to make our**:entities** module into android module in order to use the @Entity annotation.

Android library
Java/Kotlin library



Room-Common library:

Instead of using **room-runtime,** room common library could be used in
:entities module. It’s a java library that just contains annotation files of room library.

Conclusion:

I shared a way to keep room entity in a Java/Kotlin module as it is a domain model.