YALOVA UNIVERSITY

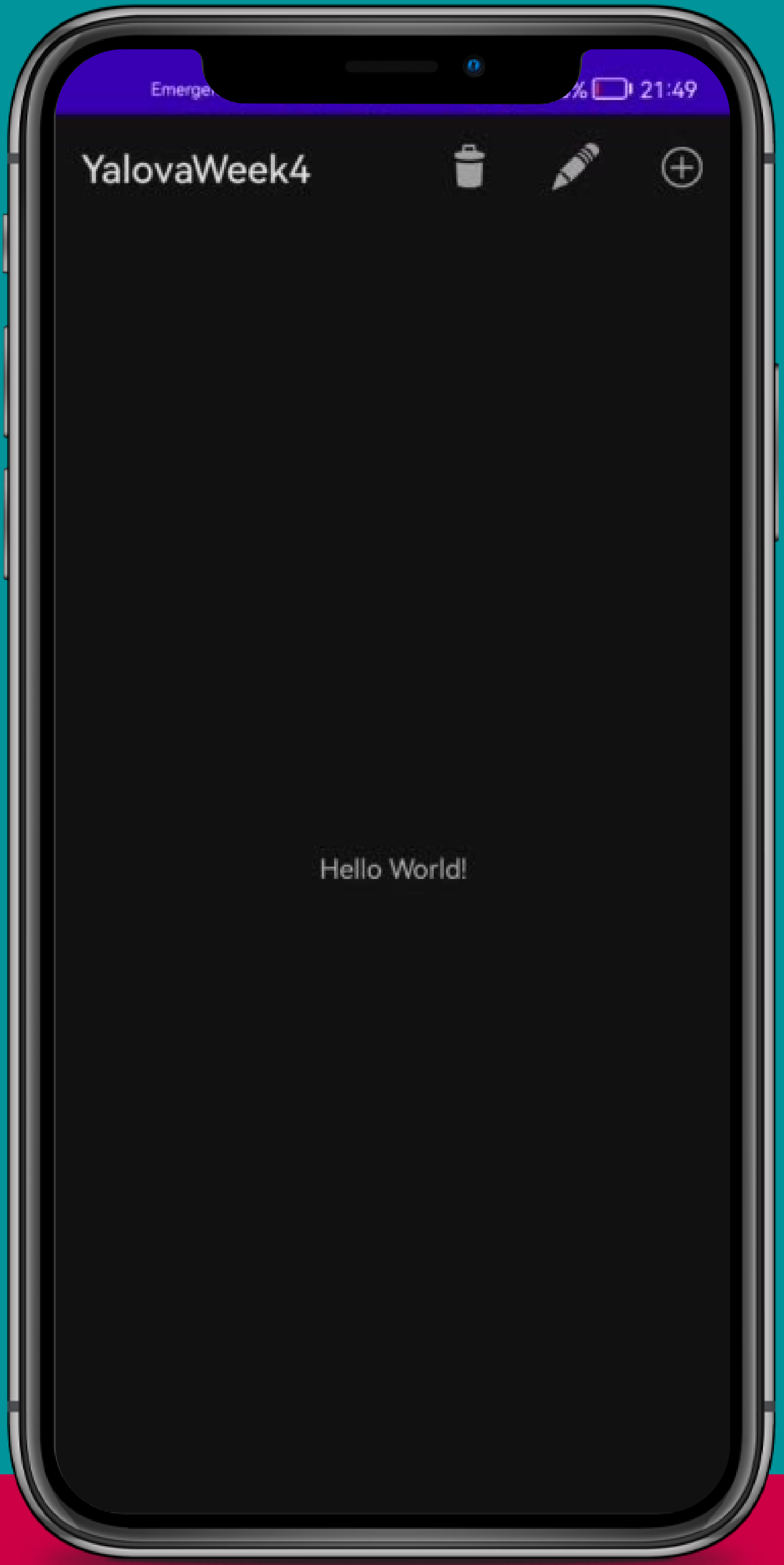
HUAWEI

Android Programming with Huawei Mobile Services

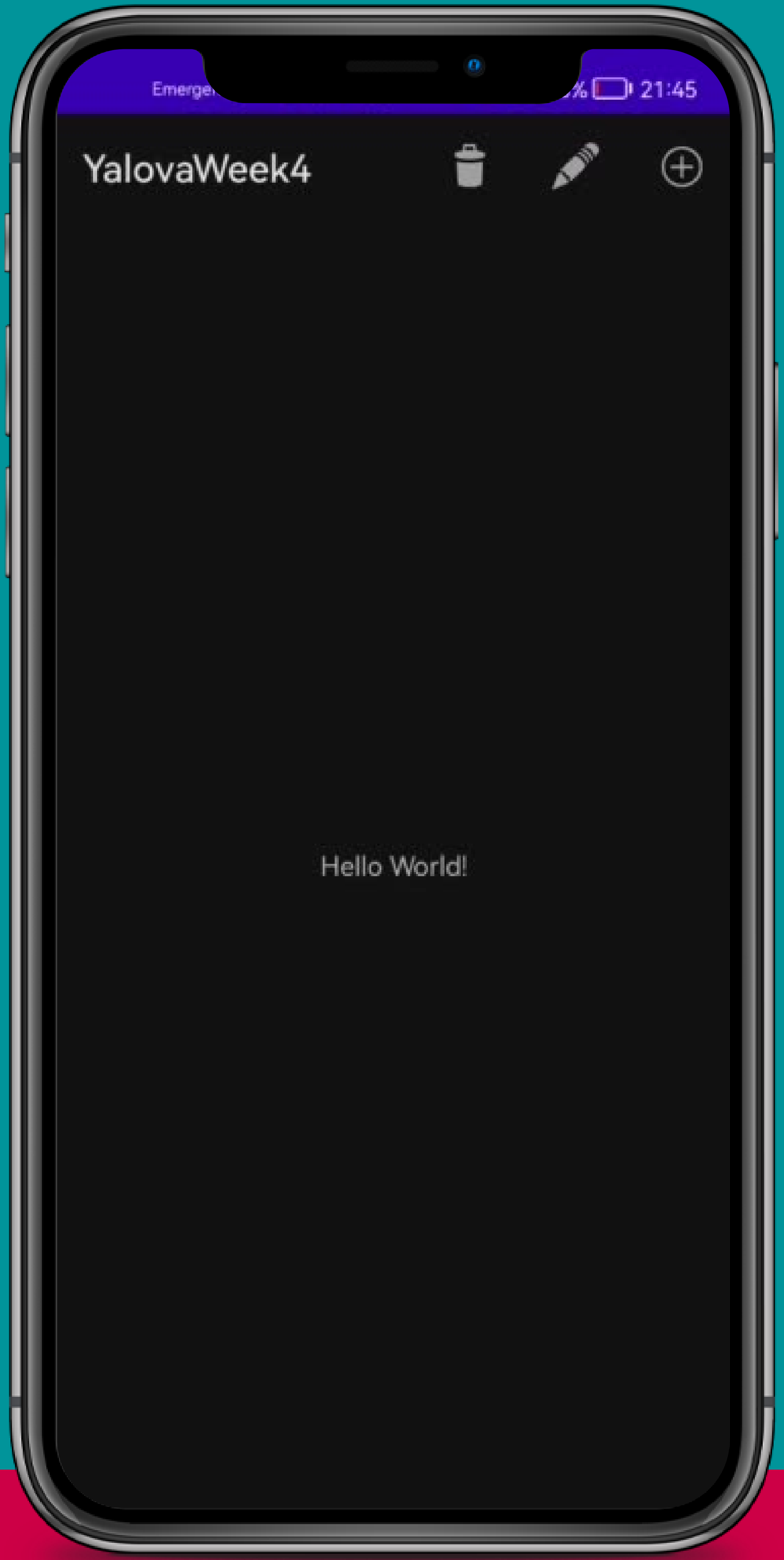
Berk Ozyurt
Mehmet Yozgatli
Cengiz Toru

W9:
ANDROID

- **Retrofit**
- **Coroutines**
- **MVVM Design Pattern**
    - **Observer Pattern**
    - **LiveData**



Hello World!



Hello World!

Retrofit - Network Requests

https://square.github.io/retrofit/

https://randomuser.me/documentation#howto

https://www.w3schools.com/tags/ref_httpmethods.asp

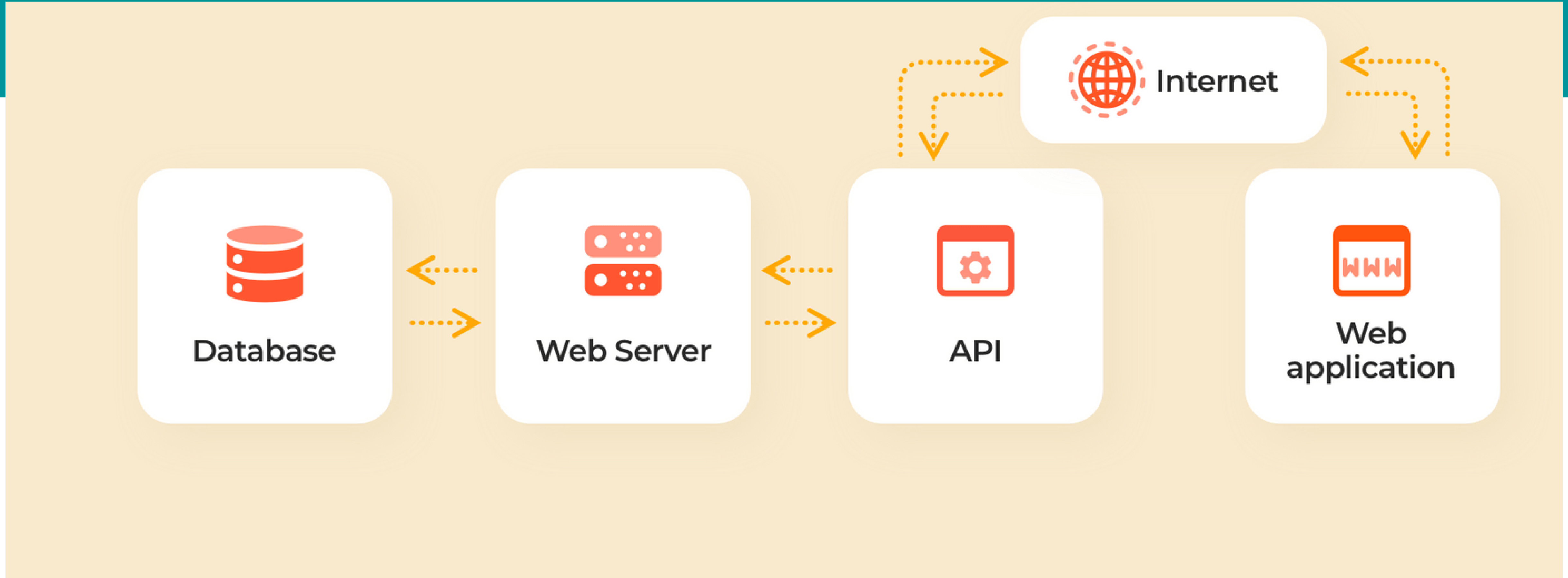
https://www.postman.com/downloads/?utm_source=postman-home

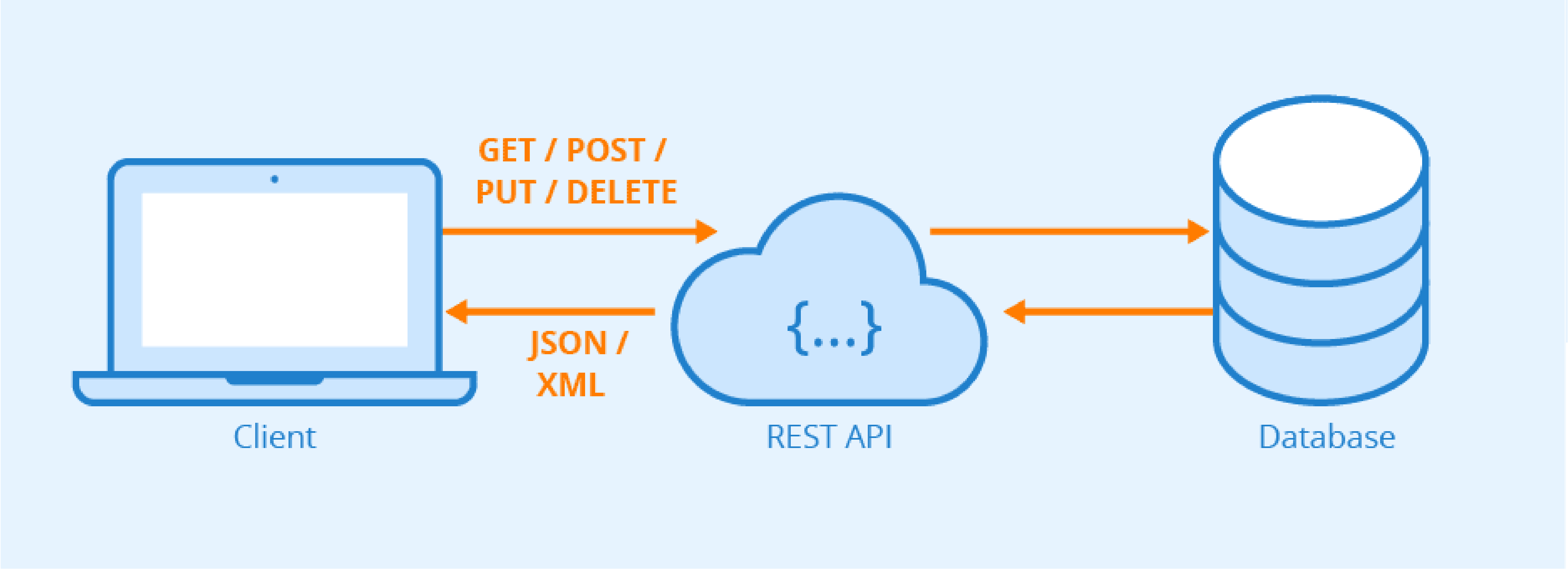
https://medium.com/gdgtekirdag/android-ile-retrofit-kullan%C4%B1m%C4%B1-spacexdata-71a5117666d1

https://medium.com/@snowronark/androidde-retrofit-2-ile-rest-api-kullan%C4%B1m%C4%B1-get-429aa6c4f2af

Some Keywords

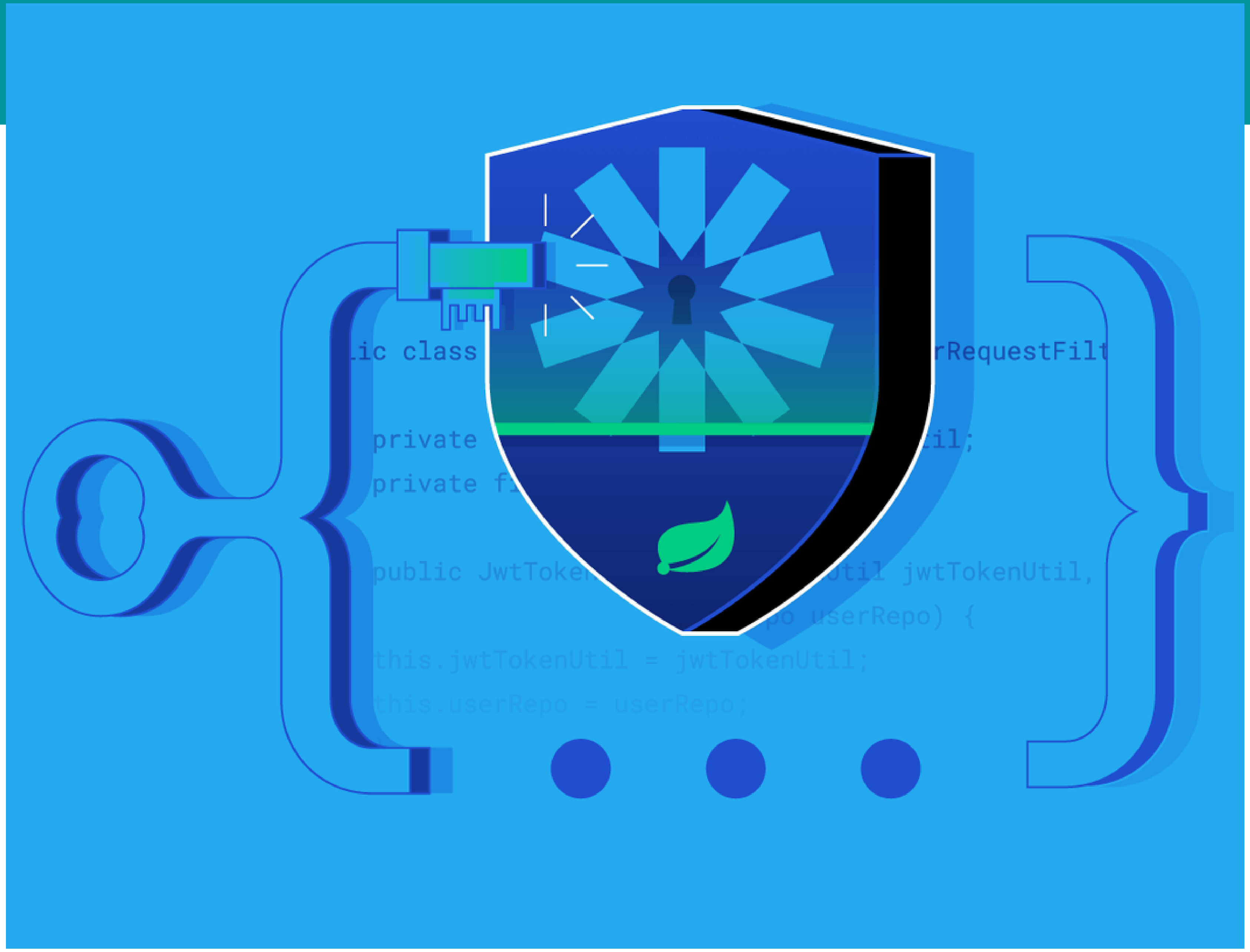
- ✓ Backend
- ✓ Client
- ✓ API
- ✓ Request
  - ✓ GET
  - ✓ POST
  - ✓ PUT
  - ✓ DELETE





Retrofit

- ✓ A type-safe HTTP client for Android
- ✓ Steps
  - ✓ Creating data model equal of JSON response
  - ✓ Create an interface which contains API enpoints
  - ✓ Build Retrofit instance
  - ✓ Send request and obtain response



Coroutines

https://developer.android.com/kotlin/coroutines

https://kotlinlang.org/docs/coroutines-guide.html

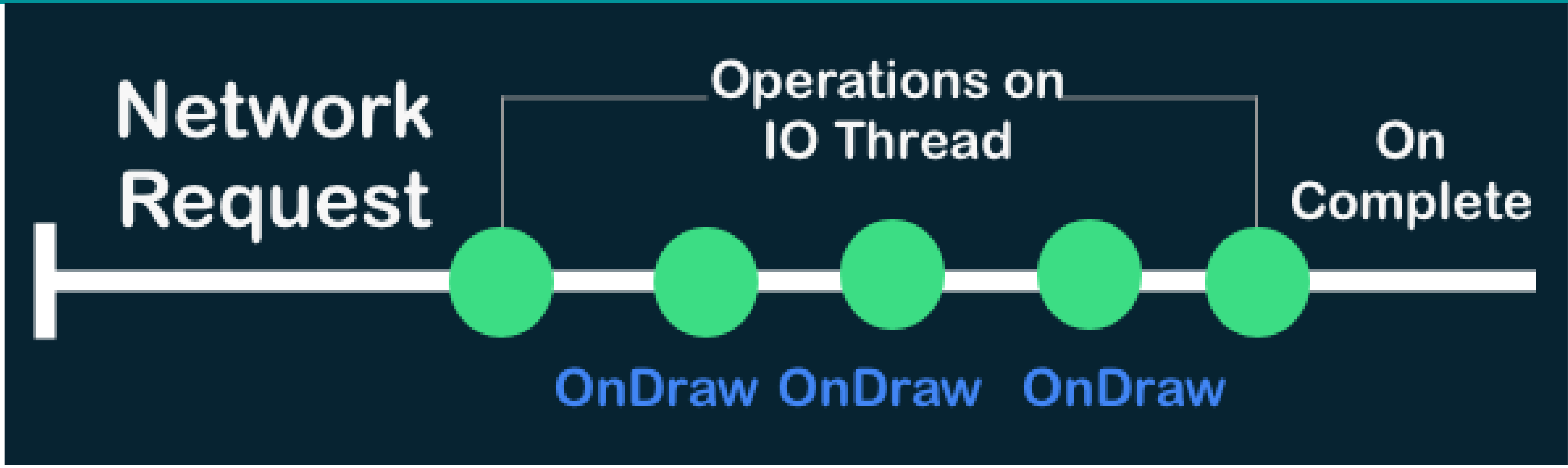
https://halilozel1903.medium.com/kotlin-coroutines-nedir-886a8f50010c

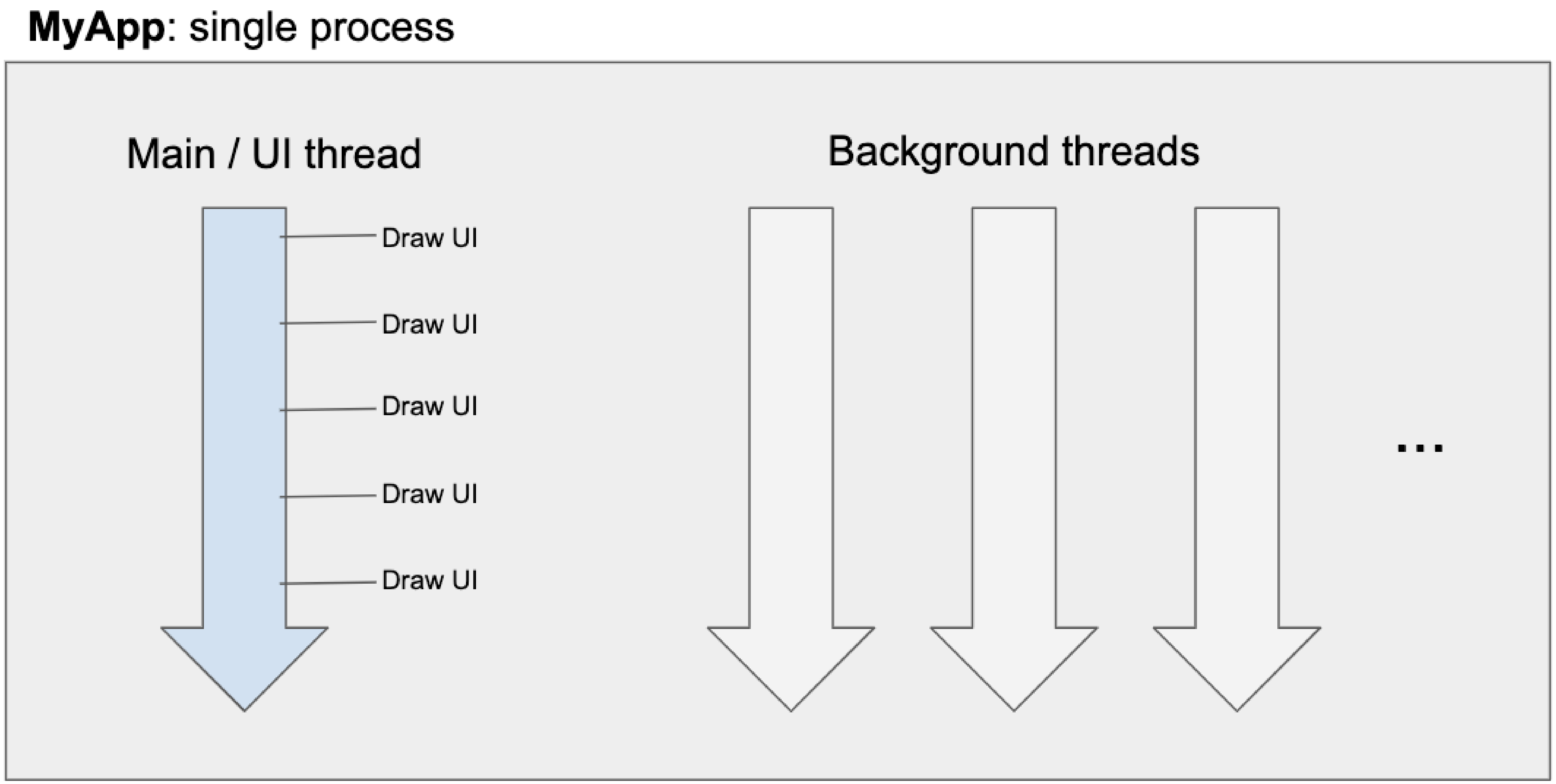
https://medium.com/gradeup/introduction-to-kotlin-coroutines-for-beginners-8b1d2a41c008

https://medium.com/trendyol-tech/kotlin-coroutine-asenkron-ve-paralel-programlama-c5a0b9e945c8

Some Keywords related to OS

- ✓ Process
- ✓ Threads
- ✓ Asynchronous Programming
- ✓ Sending network requests in the background



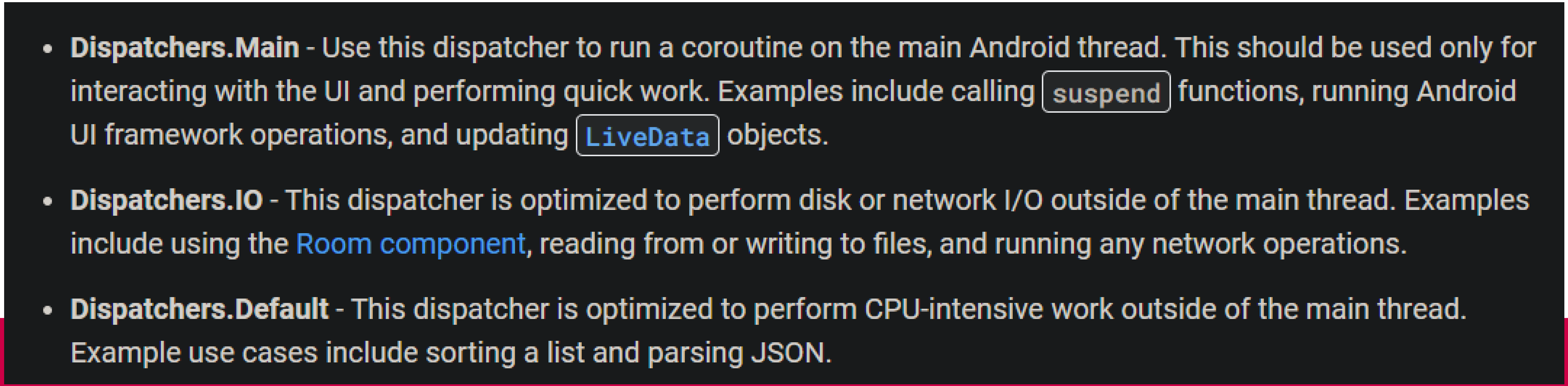


Coroutines

✓ A coroutine is a concurrency design pattern that you can use on Android to simplify code that executes asynchronously. Coroutines were added to Kotlin in version 1.3

✓ On Android, coroutines help to manage long-running tasks that might otherwise block the main thread and cause your app to become unresponsive.

✓ Light-Weight ✓ Light-Fewer memory leaks ✓ Built-in cancellation support



- **Dispatchers.Main** - Use this dispatcher to run a coroutine on the main Android thread. This should be used only for interacting with the UI and performing quick work. Examples include calling `suspend` functions, running Android UI framework operations, and updating `LiveData` objects.
- **Dispatchers.IO** - This dispatcher is optimized to perform disk or network I/O outside of the main thread. Examples include using the Room component, reading from or writing to files, and running any network operations.
- **Dispatchers.Default** - This dispatcher is optimized to perform CPU-intensive work outside of the main thread. Example use cases include sorting a list and parsing JSON.

Retrofit + Coroutines Example

✓ https://randomuser.me/api/?results=100

✓ https://randomuser.me/documentation#howto

MVVM
(Model-View-ViewModel)

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

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

https://code.tutsplus.com/tutorials/android-design-patterns-the-observer-pattern--cms-28963

https://proandroiddev.com/understanding-mvvm-pattern-for-android-in-2021-98b155b37b54

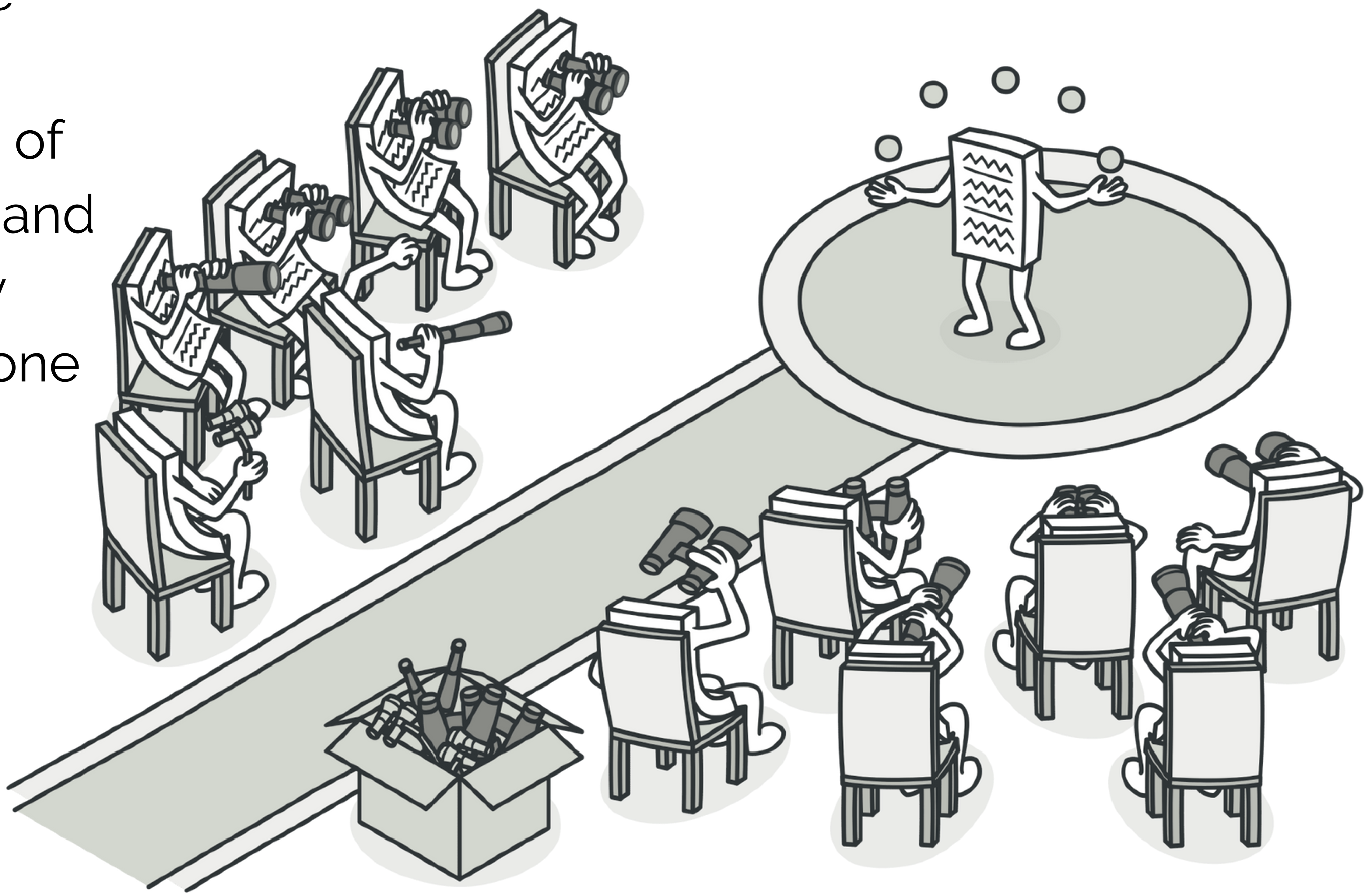
https://www.raywenderlich.com/636803-mvvm-and-databinding-android-design-patterns

https://betterprogramming.pub/everything-to-should-understand-about-livedata-507dd83adea7

https://blog.mindorks.com/mvvm-architecture-android-tutorial-for-beginners-step-by-step-guide

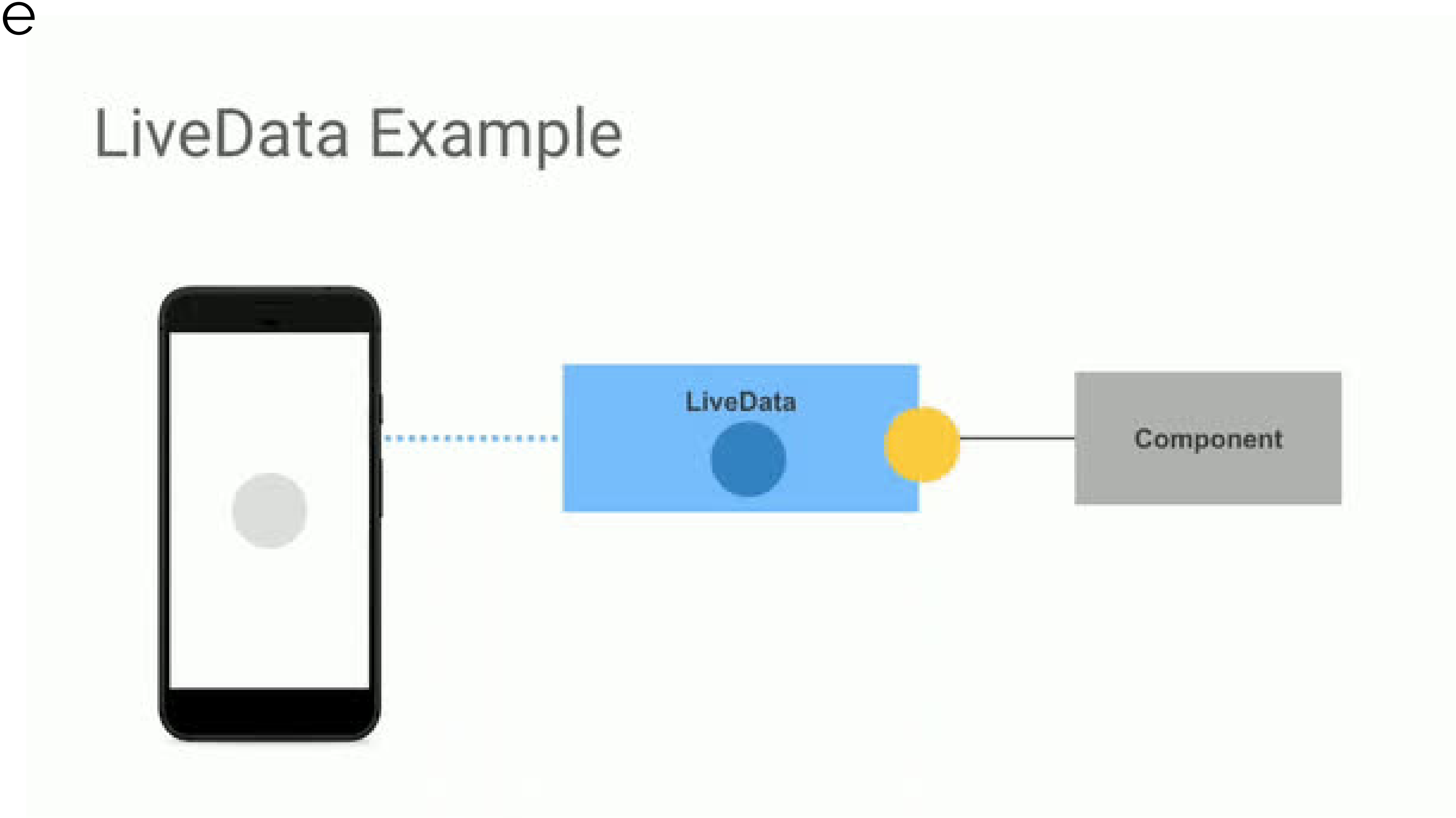
Observer Design Pattern

- ✓ The observer pattern is a software design pattern in which an object, called the subject, maintains a list of its dependents, called observers, and notifies them automatically of any state changes, usually by calling one of their methods.



LiveData

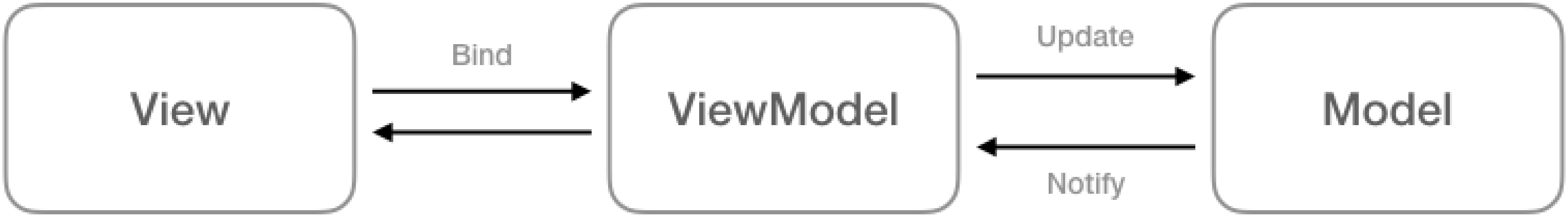
- LiveData is an observable data holder class. Unlike a regular observable, LiveData is lifecycle-aware, meaning it respects the lifecycle of other app components, such as activities, fragments, or services. This awareness ensures LiveData only updates app component observers that are in an active lifecycle state.
- Ensures your UI matches your data state
- No memory leaks
- No crashes due to stopped activities
- No more manual lifecycle handling
- Always up to date data
- Proper configuration changes



MVVM (Model- View- ViewModel)

- MVVM architecture is a Model-View-ViewModel architecture that removes the tight coupling between each component. Most importantly, in this architecture, the children don't have the direct reference to the parent, they only have the reference by observables.

- Clean, Understable Code
- Flexible, Extensible Architecture
- Testing the Code
- Officialy recommended and supported architecture
- Easily find bugs



View

Bind

ViewModel

Update

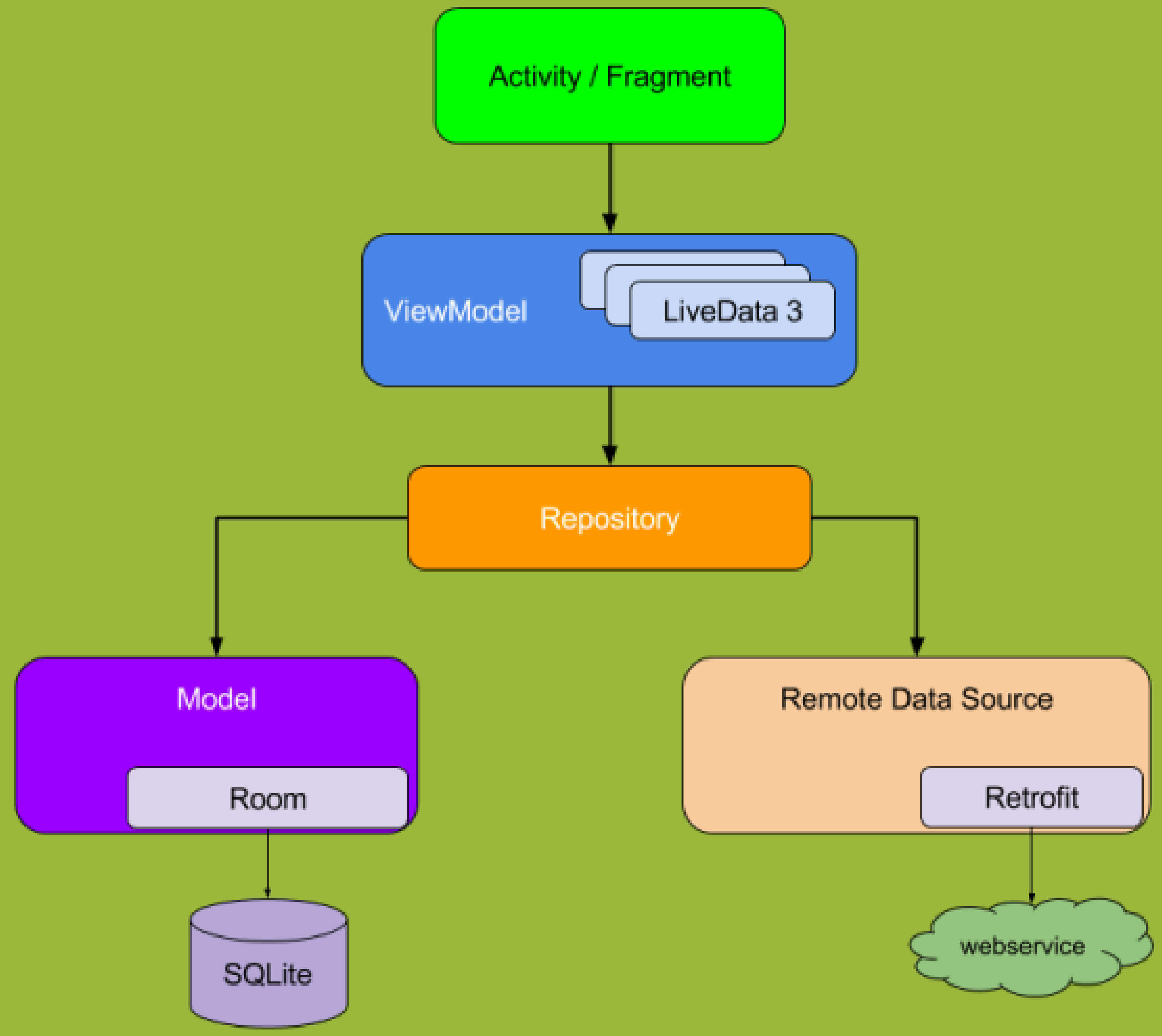
Notify

Model

MVVM (Model- View- ViewModel)

Layers

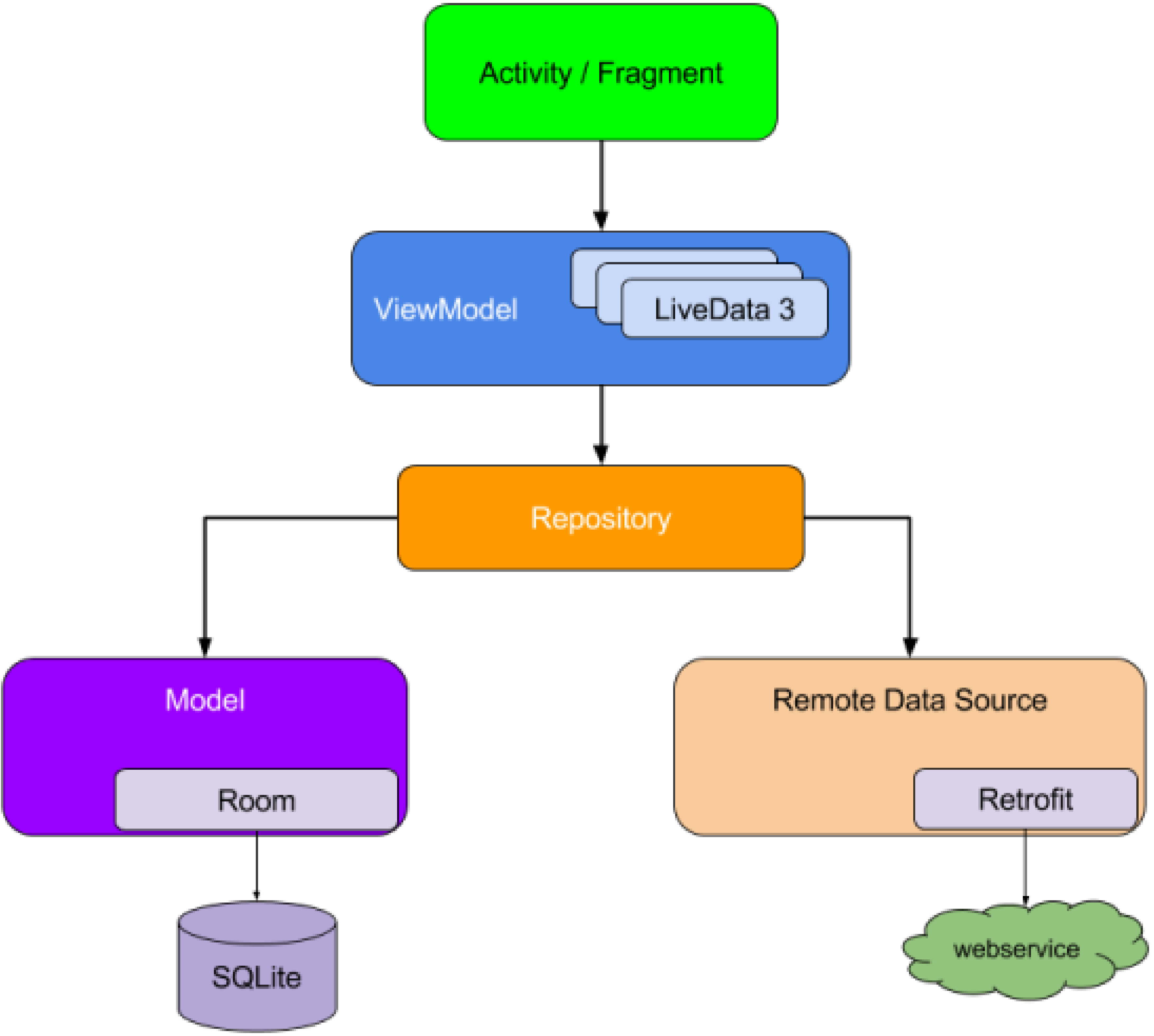
- ✓ Model
- ✓ View
- ✓ ViewModel



MVVM (Model- View- ViewModel)

View Layer

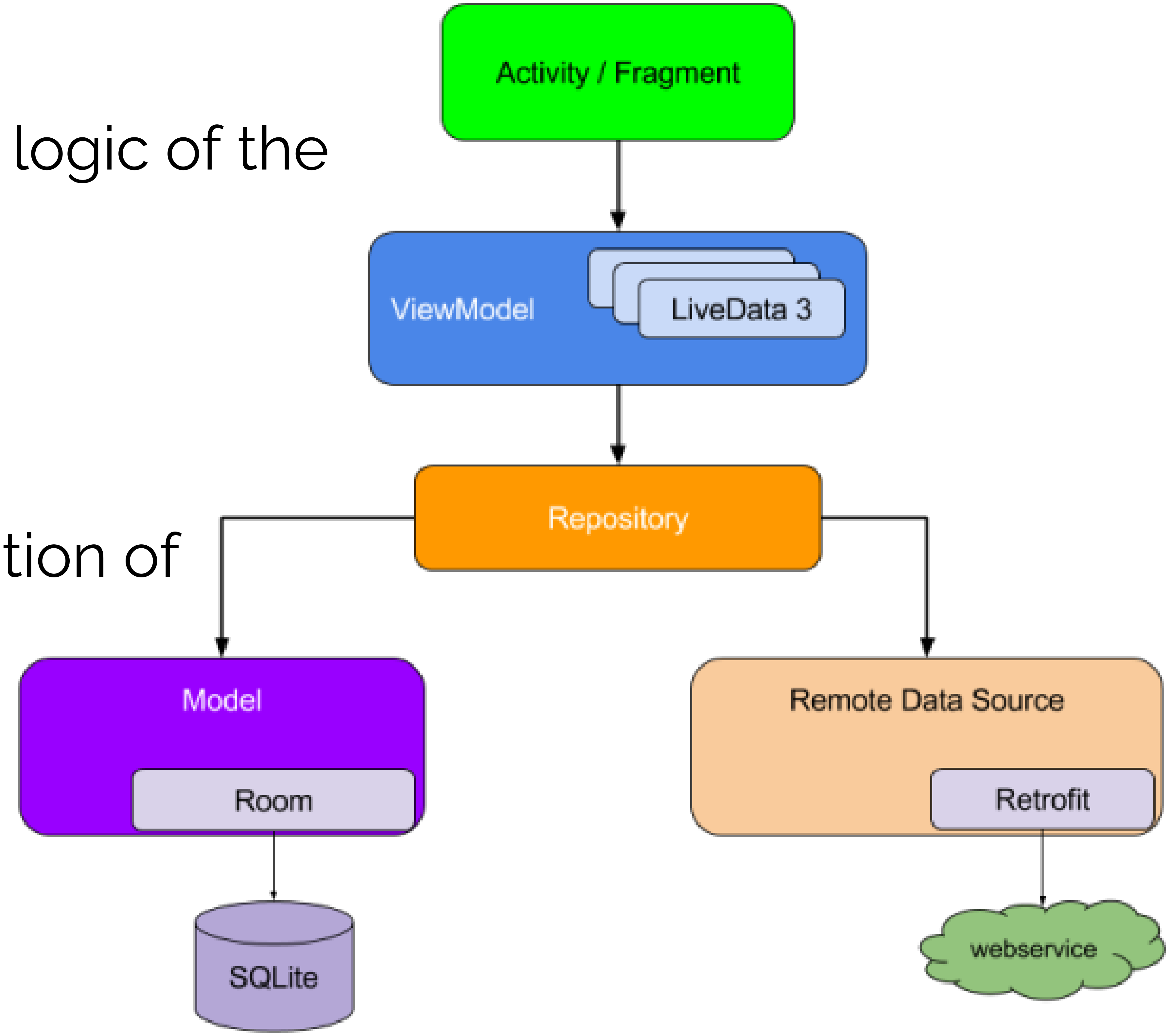
- ✓ Its role is to display whatever it receives from the ViewModel and forward input to it. It’s as simple as that!
- ✓ Activiy, Fragment, Composables
- ✓ No business logic
- ✓ Everything UI related



MVVM (Model- View- ViewModel)

Model Layer

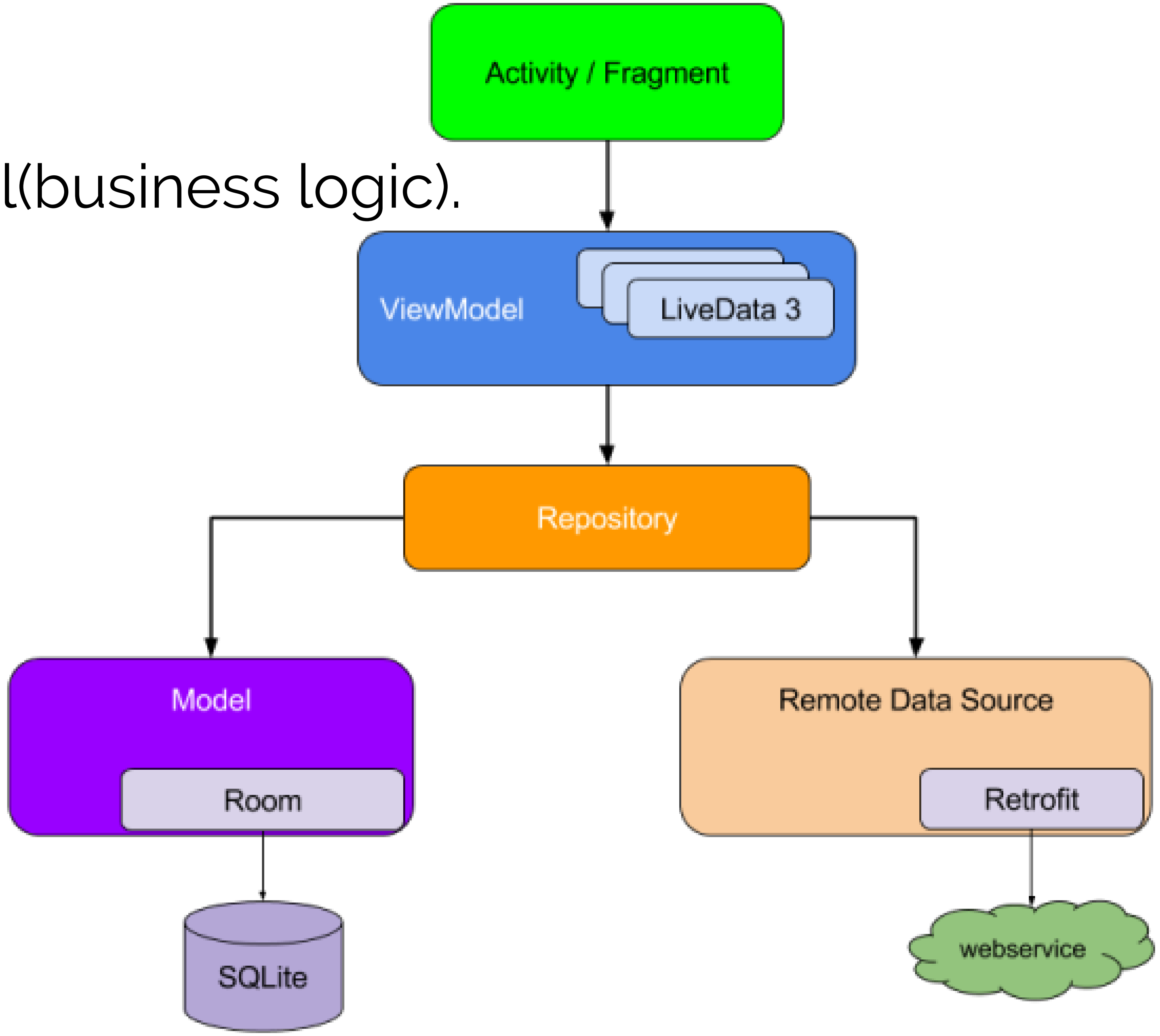
- ✓ It represents the data and the business logic of the Android Application.
- ✓ The Model is often created using a **Repository** pattern.
- ✓ This layer is responsible for the abstraction of the data sources.
- ✓ It store the state of the system and let the ViewModel query it.



MVVM (Model- View- ViewModel)

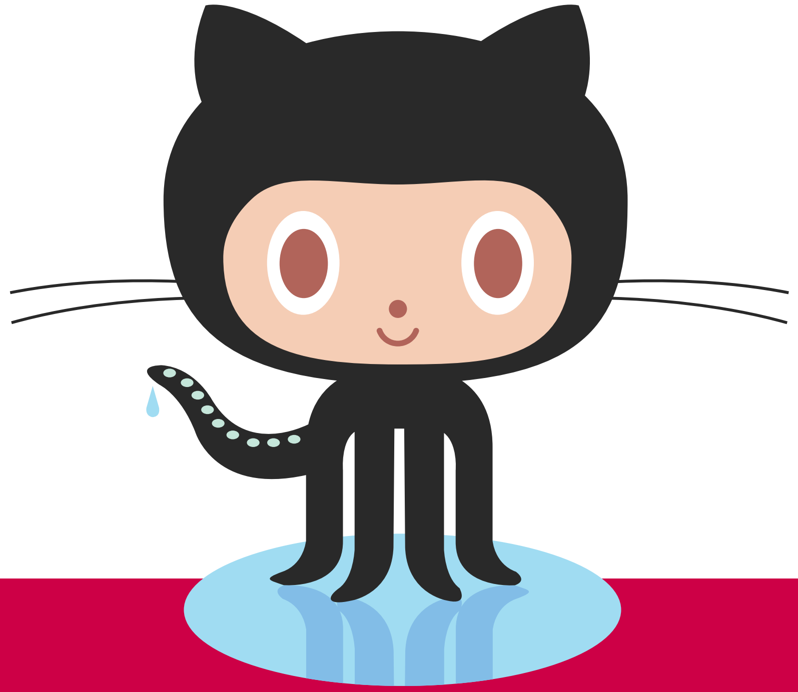
ViewModel Layer

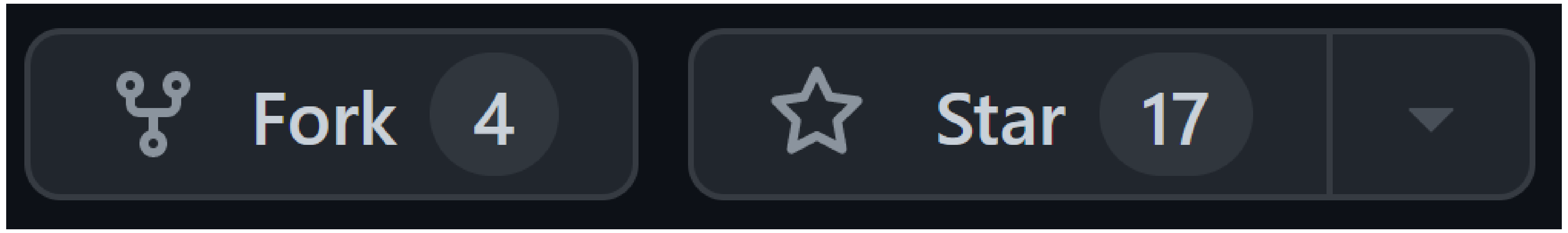
- ✓ It is a bridge between the View and Model(business logic).
- ✓ It interacts with the Model and exposes the observable that can be observed by the View.
- ✓ No View Reference
- ✓ Survive configuration changes



Next Week

- ✓ https://github.com/Explore-In-HMS/Explore-Landmarks
- ✓ https://github.com/Explore-In-HMS/Explore-Landmarks/blob/master/app/src/main/java/com/hms/explorelandmarks/ui/image_gallery/LandmarkImageGalleryFragment.kt
- ✓ https://developer.huawei.com/consumer/en/hms





LET'S TALK

CONTACT INFORMATION

Berk Ozyurt
berk.ozyurt1@huawei.com

Mehmet Yozgatli
mehmet.yozgatli1@huawei.com

Cengiz Toru
cengiz.toru@huawei.com

Telegram Channel
will be created

Official Website
https://developer.huawei.com