Автор Юрий Петров vk.com/mobile_developing
https://t.me/mobile_developing_chat

MVVM

Model - View - ViewModel

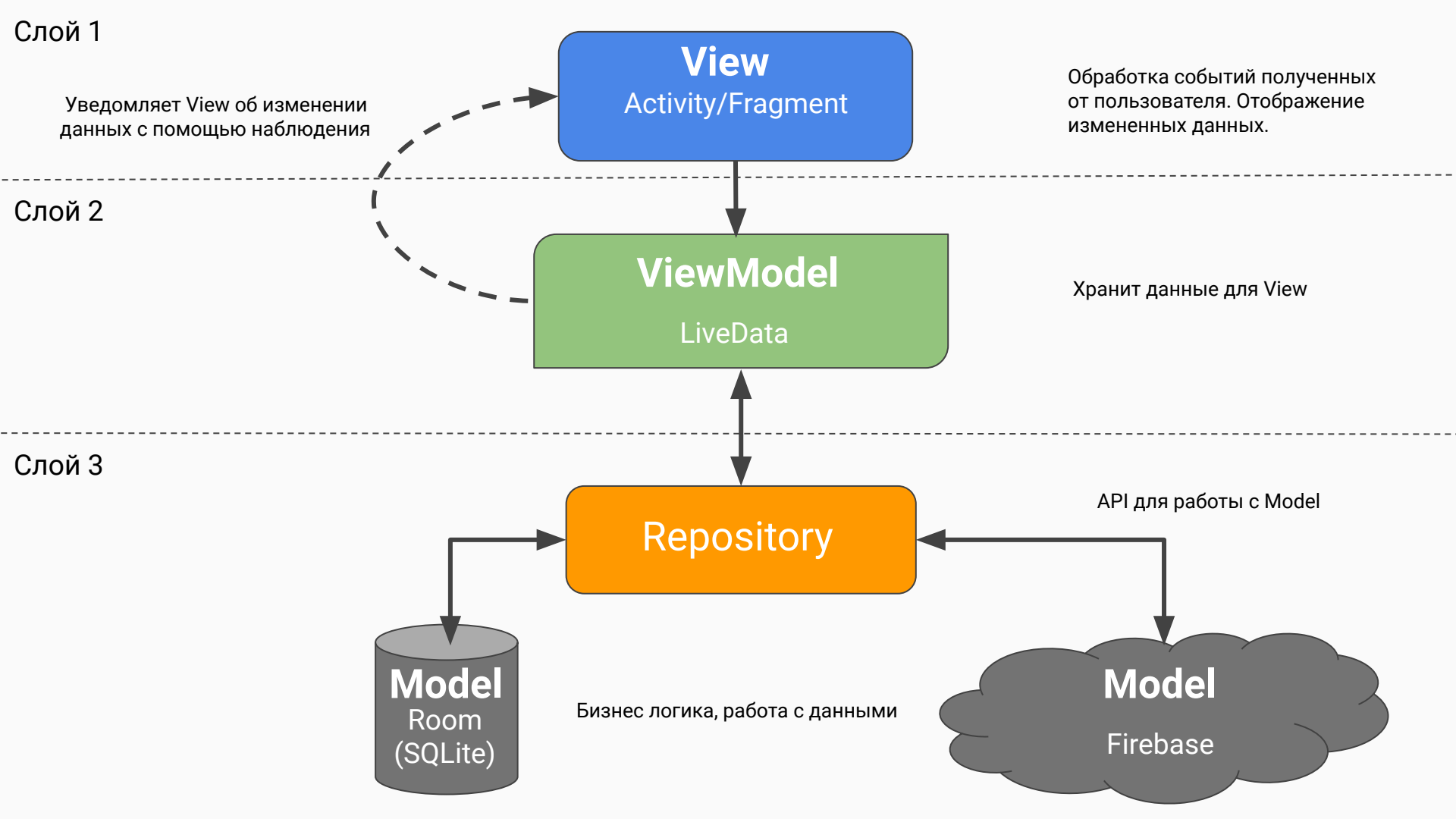
Шаблон проектирования архитектуры приложения.

Model - View - ViewModel

Model-View-ViewModel — Шаблон представлен в 2005 году Джоном Госсманом (John Gossman)

Цель - разделить приложение на 3 независимых слоя.

1. **View** - Activity/Fragment
2. **Model** (Repository) - работа с бизнес логикой (работа с базой данных Firebase, SQLite и другие)
3. **ViewModel** - является связующим звеном View с Model, а также хранит данные для View


Слой 1
View
Activity/Fragment
Уведомляет View об изменении данных с помощью наблюдения
Обработка событий полученных от пользователя. Отображение измененных данных.
Слой 2
ViewModel
LiveData
Хранит данные для View
Слой 3
Repository
API для работы с Model
Model
Room
(SQLite)
Бизнес логика, работа с данными
Model
Firebase

Архитектурные компоненты Android

Android Architecture Components

1. Lifecycle
2. LiveData
3. ViewModel (AndroidViewModel)
4. Navigation

Dependencies

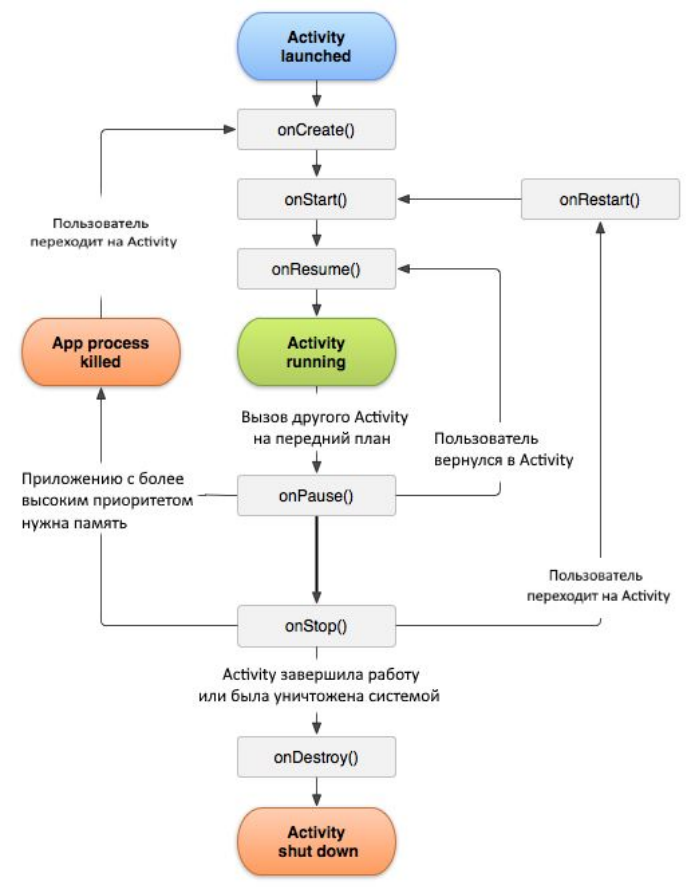
```
    def nav_version = "2.3.0-beta01"
    def lifecycle_version = "2.2.0"
    def coroutines_version = "1.3.4"

    // Lifecycle
  implementation "androidx.lifecycle:lifecycle-extensions:$lifecycle_version "
    // Kotlin
implementation "androidx.navigation:navigation-fragment-ktx:$nav_version"
implementation "androidx.navigation:navigation-ui-ktx:$nav_version"
    // Dynamic Feature Module Support
implementation "androidx.navigation:navigation-dynamic-features-fragment:$nav_version"
    // Coroutines
implementation 'org.jetbrains.kotlinx:kotlinx-coroutines-android:$coroutines_version'
```

Android Architecture Components

Lifecycle

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



Жизненный цикл Activity

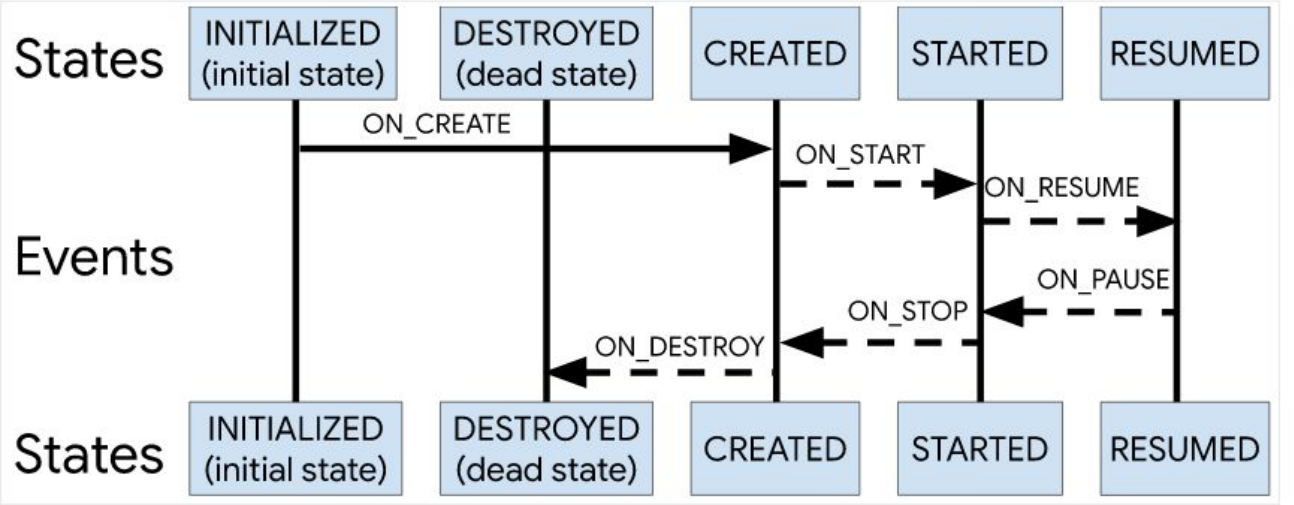
1. **onCreate():** вызывается при создании вашей Activity.
2. **onStart():** осуществляется подготовка к выводу activity на экран.
3. **onRestoreInstanceState():** восстановление данных
4. **onResume():** activity видно на экране и может взаимодействовать с пользователем.
5. **onPause():** фокусировка на другом Activity.
6. **onSaveInstanceState():** сохранение данных
7. **onStop():** activity больше не видна.
8. **onDestroy:** activity уничтожена

Жизненный цикл Fragment

1. **onAttach():** связывание с Activity
2. **onCreate():** инициализация фрагмента
3. **onCreateView():** установка макета
4. **onActivityCreated():** после установки макета
5. **onStart():** фрагмент становится видимым
6. **onResume():** фрагмент становится активным
7. **onPause():** фрагмент становится не активным
8. **onStop():** фрагмент становится видимым
9. **onDestroyView():** удаление макета из фрагмента
10. **onDestroy:** фрагмент уничтожен
11. **onDetach():** Отвязывание от Activity

Обработка жизненных циклов

- Компоненты Lifecycle выполняют действия в ответ на изменение состояния жизненного цикла Activity или Fragment.
- **Lifecycle** это класс, который содержит информацию о состоянии жизненного цикла Activity или Fragment, и позволяет другим объектам наблюдать за этим состоянием.



Обработка жизненных циклов

- Компоненты **Lifecycle** выполняют действия в ответ на изменение состояния жизненного цикла Activity или Fragment.
- **Lifecycle** это класс, который содержит информацию о состоянии жизненного цикла Activity или Fragment, и позволяет другим объектам наблюдать за этим состоянием. **Lifecycle** использует два основных перечисления для отслеживания статуса жизненного цикла для связанного с ним компонента:
  - **Event** - События жизненного цикла . Эти события влияют на события обратного вызова в Activity или Fragment.
  - **State** - Текущее состояние компонента, отслеживаемого **Lifecycle** объектом.
- **LifecycleOwner** является интерфейсом, который обозначает, что у класса есть **Lifecycle**. У него есть один метод, getLifecycle().
- **LifecycleObserver** интерфейс который дает возможность подписаться на **Lifecycle**.

Events (события жизненного цикла)

`Lifecycle.Event.ON_ANY` - Отслеживание всех возможных событий.

`Lifecycle.Event.ON_CREATE` - Отслеживание метода onCreate владельца жизненного цикла.

`Lifecycle.Event.ON_DESTROY` - Отслеживание метода onDestroy владельца жизненного цикла.

`Lifecycle.Event.ON_PAUSE` - Отслеживание метода onPause владельца жизненного цикла.

`Lifecycle.Event.ON_RESUME` - Отслеживание метода onResume владельца жизненного цикла.

`Lifecycle.Event.ON_START` - Отслеживание метода onStart владельца жизненного цикла.

`Lifecycle.Event.ON_STOP` - Отслеживание метода onStop владельца жизненного цикла.

States (текущее состояние владельца жизненного цикла)

`Lifecycle.State.CREATED` - состояние создано для владельца жизненного цикла.

`Lifecycle.State.DESTROYED` - состояние уничтожено для владельца жизненного цикла.

`Lifecycle.State.INITIALIZED` - состояние инициализировано для владельца жизненного цикла.

`Lifecycle.State.RESUMED` - состояние восстановлено для владельца жизненного цикла.

`Lifecycle.State.STARTED` - состояние запущено для владельца жизненного цикла.

Android Architecture Components

LiveData

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

LiveData является классом для хранения данных. В отличие от обычных данных, **LiveData** учитывает жизненный цикл. Это означает, что **LiveData** учитывает жизненный цикл компонентов приложения, таких как Activities, Fragments или Services. Эта осведомленность гарантирует, что **LiveData** обновляет только те компоненты приложения, которые находятся в состоянии активного жизненного цикла.

Методы

value - задать значение LiveData в главном потоке

postValue - задать значение LiveData из другого потока

Transformations.map - позволяет изменить тип LiveData

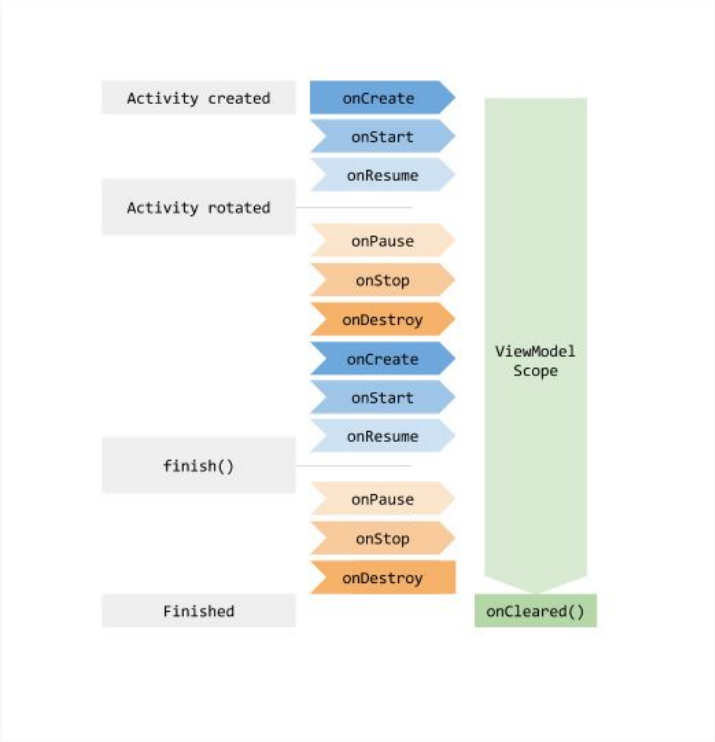
MediatorLiveData это подкласс LiveData, который может наблюдать за другими LiveData и реагировать на события от них.

Android Architecture Components

ViewModel

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

ViewModel - это класс, который отвечает за подготовку и управление данными для Activity или Fragment. Он также обрабатывает взаимодействие Activity / Fragment с остальной частью приложения (например, коммуникация с классами бизнес-логики).
Примечание : при смене ориентации экрана ViewModel не уничтожается.



Android Architecture Components

Navigation

https://developer.android.com/guide/navigation

Компонент **Navigation** - позволяет пользователям перемещаться между различными частями контента в вашем приложении. Компонент **Navigation** помогает реализовать навигацию, от простых нажатий кнопок до более сложных шаблонов. Также обеспечивает согласованное и предсказуемое взаимодействие с пользователем.

Компонент **Navigation** состоит из трех ключевых частей:

- **Navigation graph** : это XML ресурс, который содержит всю связанную с навигацией информацию в одном централизованном месте.
- **NavHost**: пустой контейнер, для объектов навигации
- **NavController**: объект, который управляет навигацией приложения в NavHost.