SAMSUNG

IT Академия Kotlin

Kotlin

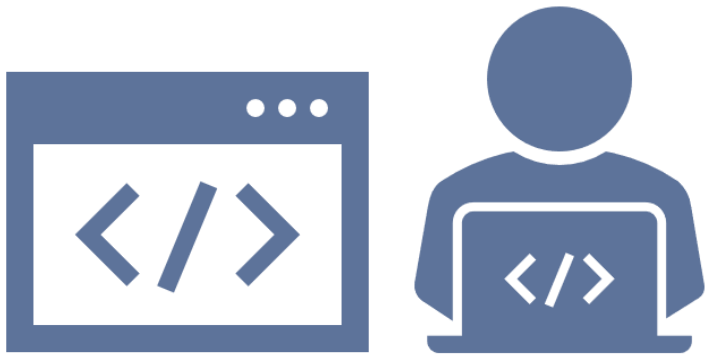
2.6. Архитектурные компоненты ViewModel и LiveData

Курс по программированию от
IT Академии Samsung

2.6.1 Введение в архитектурные компоненты

SAMSUNG IT Академия Kotlin

Kotlin

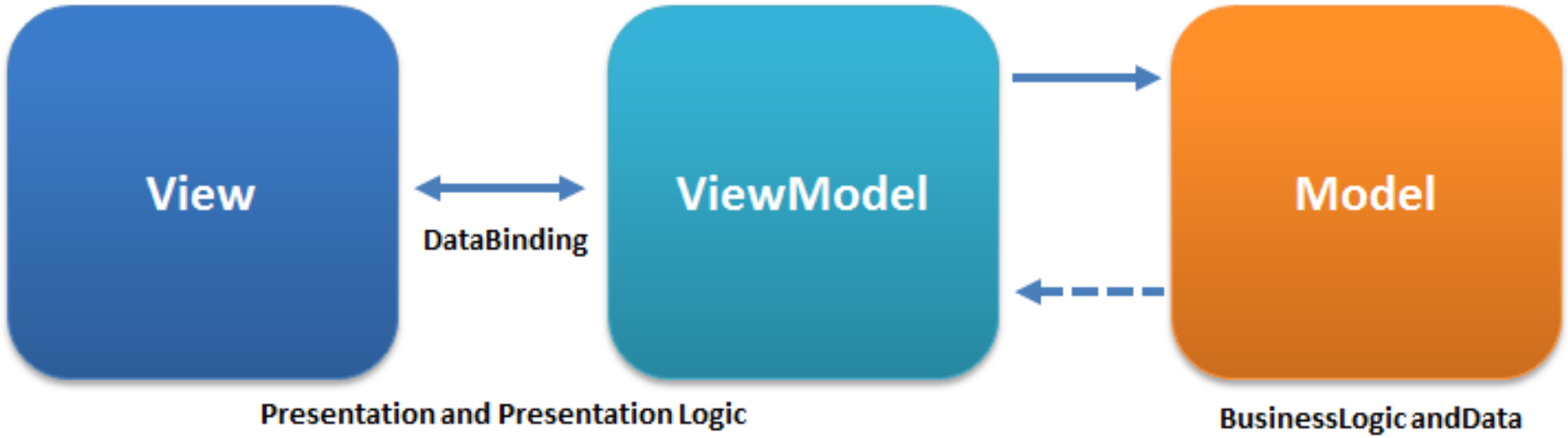


- Архитектурные компоненты в Android - набор библиотек, которые помогают разрабатывать легкие в поддержке, надежные и тестируемые приложения.
- Архитектурные компоненты включают множество новых классов, таких как: LifecycleObserver, LiveData, ViewModel, LifecycleOwner, а также библиотеку Room для работы с базой данных приложения.
- LiveData — это наблюдаемый объект для хранения данных, он уведомляет наблюдателей, когда данные изменяются. Этот компонент также является связанным с жизненным циклом LifecycleOwner (Activity или Fragment), что помогает избежать утечек памяти и других неприятностей. Компонент LiveData — предназначен для хранения объекта и разрешает подписаться на его изменения.
- ViewModel — предназначен для хранения и управления данными, связанными с пользовательским интерфейсом, с учетом жизненного цикла. Класс ViewModel позволяет данным сохраняться при изменении конфигурации, например при повороте экрана.

2.6.2. Шаблон проектирования MVVM

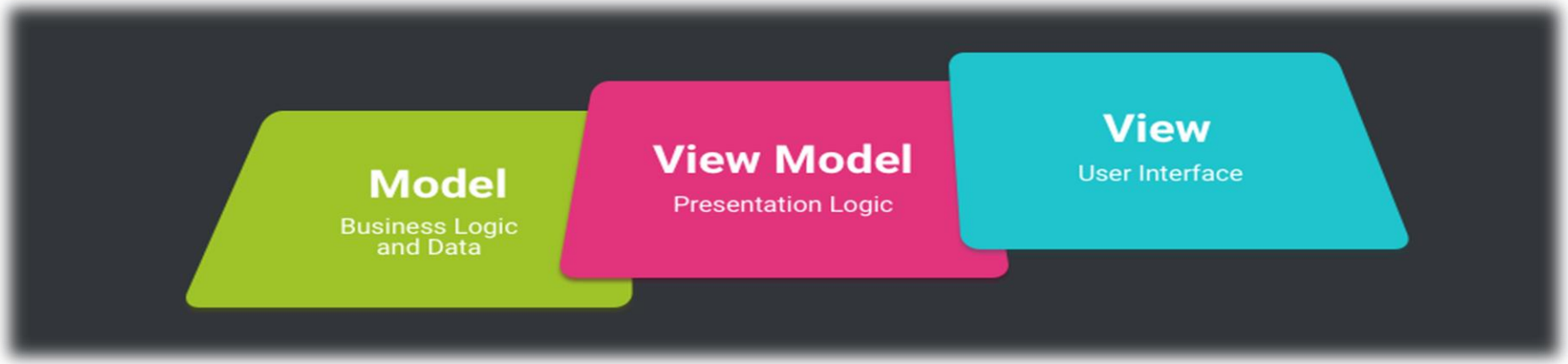
SAMSUNG IT Академия Kotlin

- Model-View-ViewModel (MVVM) — шаблон проектирования архитектуры приложения. Предложен в 2005 году Джоном Госсманом как модификация шаблона Presentation Model. Шаблон ориентирован на современные платформы разработки, в том числе Android.



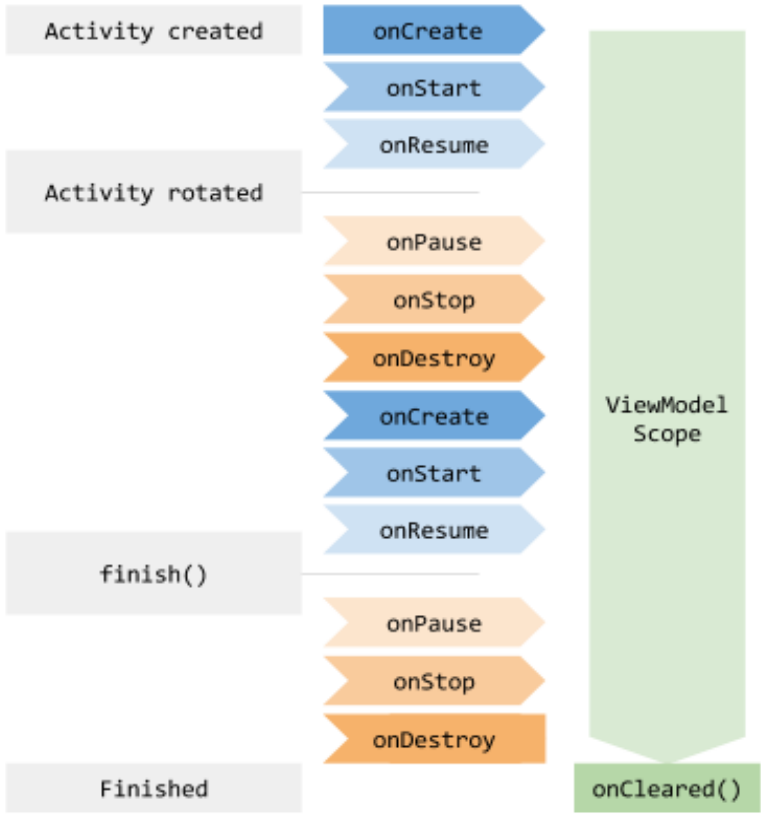
2.6.2. Шаблон проектирования MVVM

- **Model** - это логика, которая связанная с данными приложения. Другими словами - это POJO, классы для работы с API, базами данных и др.
- **View** - это разметка экрана (layout), в которой расположены необходимые представления для отображения данных.
- **ViewModel** - объект, в котором описана логика поведения **View** в зависимости от результата работы **Model**. В некоторой литературе этот объект называют моделью поведения **View**. Это может быть логика управления видимостью представлений, форматирование текста, отображение разнообразных состояний, таких как ошибки, загрузка и т.д. Также в ней описано поведение пользователей (свайпы, нажатия клавиш, касания и т.д.)



2.6.3. Класс ViewModel

ViewModel - класс, который был создан для возможности **Activity** и фрагментам сохранять необходимые им объекты при повороте экрана. Из картинки видно, что **ViewModel** жизнеспособен, пока **Activity** окончательно не закроется . При новом запуске **Activity** класс **ViewModel** все еще живет и задействован во вновь созданном **Activity**.



2.6.4. Класс LiveData

Класс LiveData - хранилище информации, работающий по принципу шаблона проектирования Observer (наблюдатель). Данное хранилище играет две ключевые роли:

- в него можно поместить какой-либо объект;
- на него можно подписаться и получать объекты, которые в него помещают.
- То есть во-первых пользователь может поместить объект в хранилище, во-вторых подписанные стороны могут получить этот объект .
- В таком виде хранения есть один большой плюс. Класс **LiveData** умеет определять активен подписчик или нет, и отправлять информацию будет только активным. Предполагается, что рассылка **LiveData** будет проводиться **Activity** и фрагментам. А их состояние активности будет определяться с помощью их жизненного цикла.

2.6.4. Класс LiveData

SAMSUNG IT Академия Kotlin

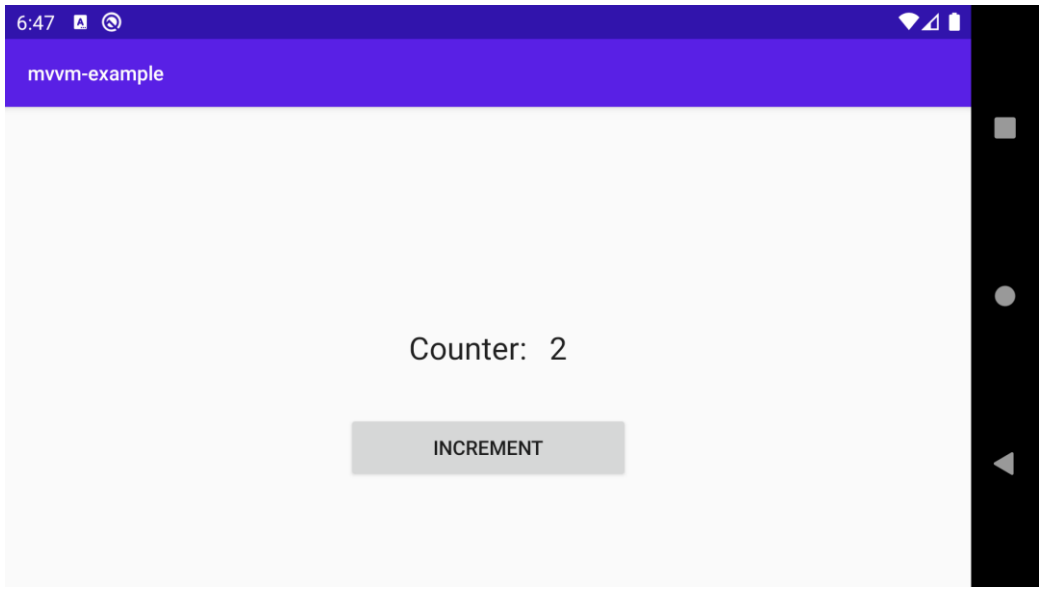
LiveData имеет ряд характеристик:

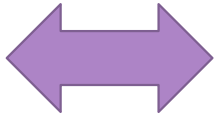
- предотвращает утечку памяти, когда наблюдатель привязан к жизненному циклу;
- предотвращает сбои из-за остановки активности;
- автоматически обрабатывает жизненный цикл;
- при использовании LiveData код становится значительно проще для тестирования.

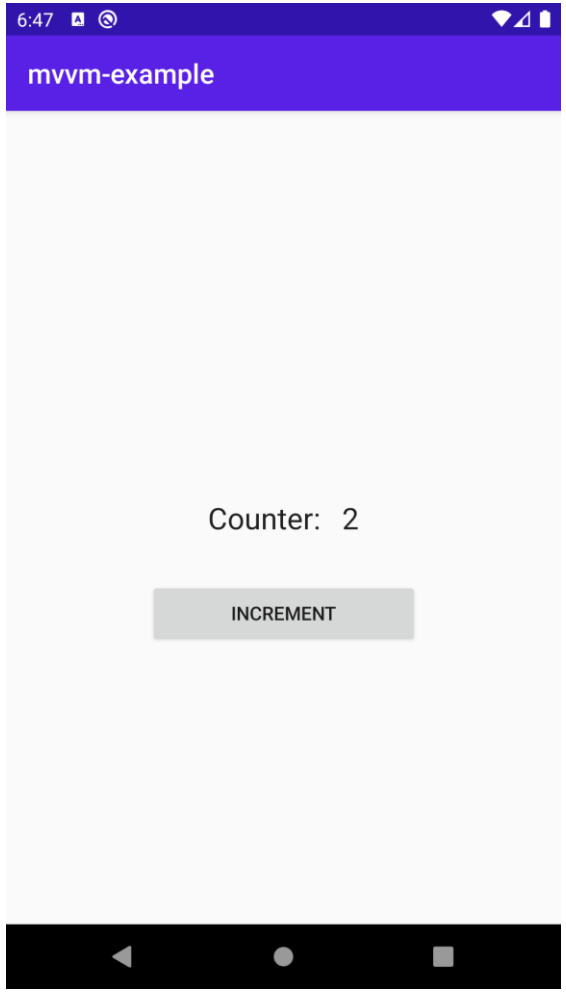
```
private val fullname = MutableLiveData<String>()

// Called on app launch
fun initNetworkRequest() {
        // expensive operation, e.g. network request
        fullname.value = "Andriiginting"
}

fun getFullname(): LiveData<String> {
        return fullname
}
// Called on Activity creation
getFullname().observe(this, Observer { user -> Log.d(TAG, user) })
```

Упражнение 2.6.

- Разработаем приложение приложение "Счетчик" с использованием архитектуры MVVM. Функционально приложение будет очень простым - по нажатию кнопки счетчик будет прибавляться.







SAMSUNG

IT Академия Kotlin

Kotlin

Спасибо за внимание!

Курс по программированию от
IT Академии Samsung