ПРИМЕНЕНИЕ АРХИТЕКТУРЫ VIPER ДЛЯ РАЗРАБОТКИ МОБИЛЬНОГО ПРИЛОЖЕНИЯ CITYQUEST НА БАЗЕ ОС ANDROID

Мазун Александр Александрович

студент, ФГАОУ ВО «Сибирский федеральный университет» Институт космических и информационных технологий, РФ, г. Красноярск

Хоркуш Анатолий Владимирович

магистрант, ФГАОУ ВО «Сибирский федеральный университет» Институт космических и информационных технологий, РФ, г. Красноярск

Осипов Павел Андреевич

магистрант, ФГАОУ ВО «Сибирский федеральный университет» Институт космических и информационных технологий, РФ, г. Красноярск

Иванова Яна Сергеевна

магистрант, ФГАОУ ВО «Сибирский федеральный университет» Институт космических и информационных технологий, РФ, г. Красноярск

Скворцов Семен Геннадьевич

магистрант, ФГАОУ ВО «Сибирский федеральный университет» Институт космических и информационных технологий, РФ, г. Красноярск

Мосин Дмитрий Александрович

магистрант, ФГАОУ ВО «Сибирский федеральный университет» Институт космических и информационных технологий, РФ, г. Красноярск

Применение архитектурных решений в разработке программного обеспечения облегчает процесс добавления и изменения функционала проекта, ознакомление с кодом путем разбиение оного на части. В статье описывается реализация Android-приложения на популярной архитектуре VIPER (View, Interactor, Presenter, Entity, Router).

В настоящее время каждый человек владеет мобильным гаджетом, который содержит множество приложений, увеличивающие функционал устройства. Будь-то файловый менеджер или мессенджер (Viber, WhatsUp). Без них уже невозможно представить себе современное устройство связи. Существует множество разработчиков приложений, а их продукцию легко могут установить пользователи через различные площадки (PlayMarket, AppStore).

В процессе создания проекта пишется огромное количество строк кода, если все описывается в одном паттерне с кодом становится сложно работать, для внесения изменений приходится постоянно искать где начинается та или иная функция, происходит постоянное повторение кода. Для упрощения работы создаются различные архитектуры приложений.

VIPER архитектура, которая занимает одно из ведущих мест в создание приложений. Изначально эта архитектура создавалась для построения приложений на операционной

системе IOS, но с течением времени разработчики стали использовать её для создания приложений на Android-устройства.

VIPER - это подход к архитектуре мобильных приложений, основанный на идеях Роберта Мартина, изложенных им в статье The Clean Architecture.

Основные достоинства и недостатки VIPER.

Достоинства:

· Повышение тестируемости Presentation-слоя приложений.

· Полная независимость модулей друг от друга – это позволяет независимо их разрабатывать и переиспользовать.

· Передача проекта другим разработчикам, либо внедрение нового, дается намного проще, так как общие подходы к архитектуре заранее определены.

Недостатки:

· Резкое увеличение количества классов в проекте, сложности при создании нового модуля.

· Отсутствие в открытом доступе набора конкретных рекомендаций, best practices и примеров сложных приложений [3].

Концепция архитектуры VIPER отображена на рисунке 1. Приложение делится на 3 слоя и 5 модулей, существует строгое разграничение между слоями программы.

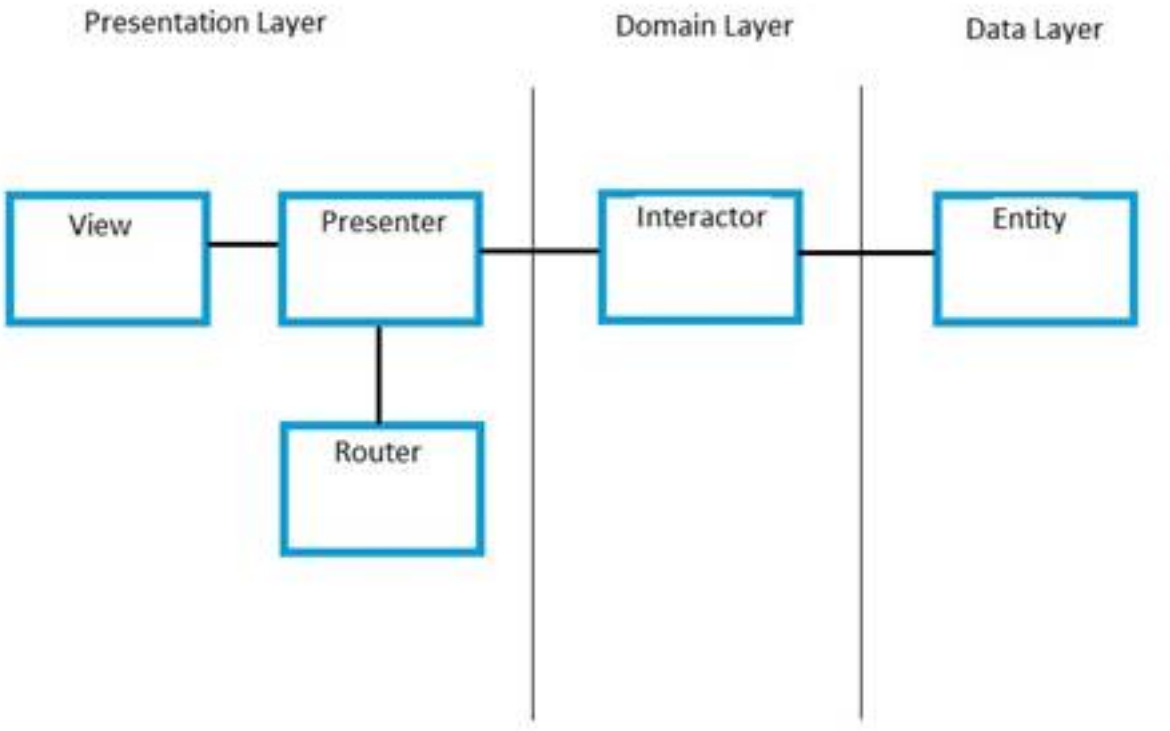

Рисунок 1. Архитектура VIPER

Для более детального представления архитектуры VIPER, рассмотрим её на примере приложения CityQuest. Для этого достаточно модуля авторизации в приложении, сама суть и функционал приложения не интересен. Интерфейс авторизации пользователя,

соответствующий View компоненту, представлен на рисунке 2.

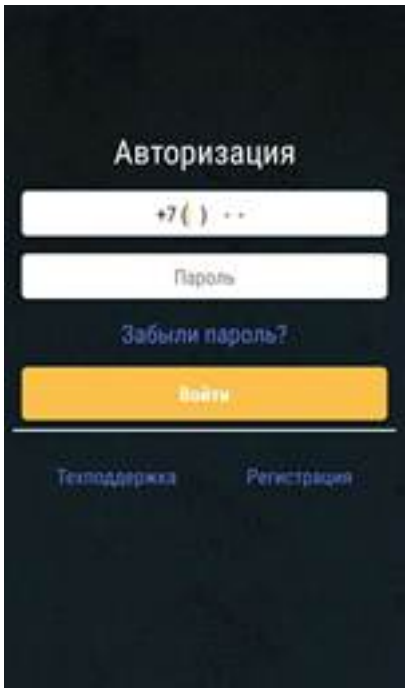

Рисунок 2. Интерфейс Авторизации

Для удобства восприятия связей между различными уровнями приложения, используем диаграмму классов изображенную на рисунке 3.

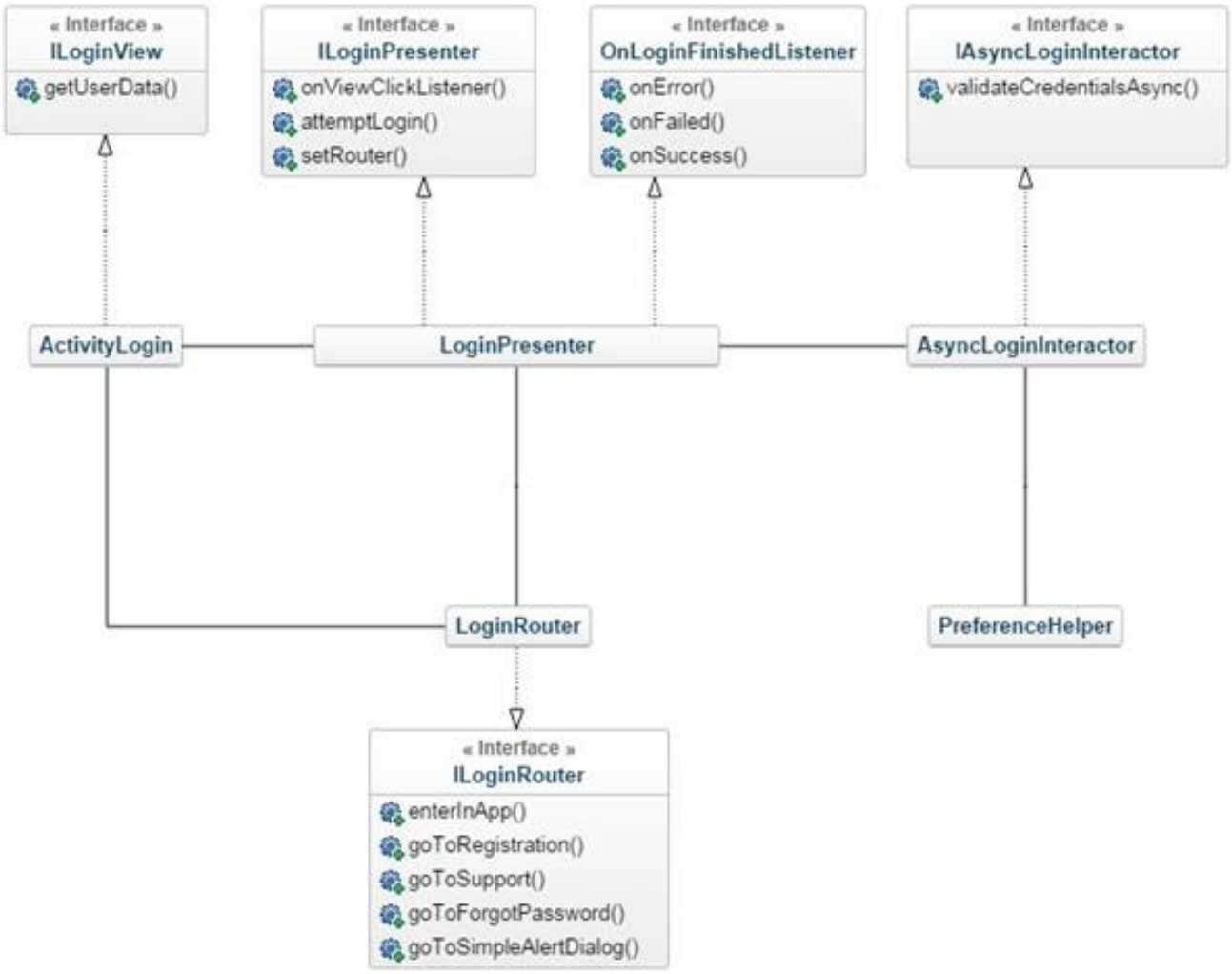

Рисунок 3. Диаграмма классов

В диаграмме отображены различные классы, выполняющие следующие функции:

ActivityLogin(View) – содержит интерфейс ввода данных.

LoginPresenter(Presenter) – определяет куда отдать управление и передать данные.

LoginRouter(Router) – отвечает за переключение между различными экранами.

AsyncLoginInteractor(Interactor) – содержит в себе бизнес-логику приложения.

PerferenceHelper(Entity) – репозиторий приложения.

View

Представляет собой интерфейс, с которым взаимодействует пользователь на экране своего мобильного устройства. Содержит то, что передал ему Presenter и ретранслирует в Presenter данные введение в нем. На рисунке 4 показан интерфейс с функцией передачи данных от View к Presenter.

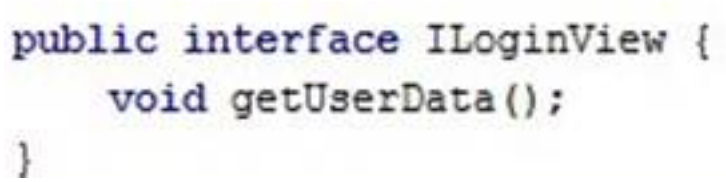

Рисунок 4. Представление View

Presenter

Содержит логику управления

Является неким «проводником», между представлением и бизнес-логикой, (который решает куда передать управление). Он определяет запрос, поступающий со View, и решает, куда отправить его дальше в Router для изменения окна Activity или же передает функцию Interactor. Чаще всего представлен в виде оператора switch-case. На рисунке 5 представлены основные функции Presenter такие как: определение клика по View, передача данных в Interactor, инициализация Router.

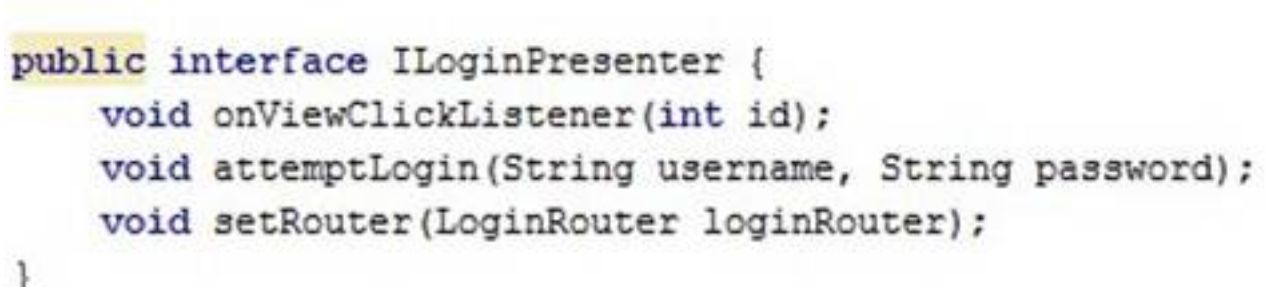

Рисунок 5. Представление Presenter

Router

Осуществляет переключения между экранами приложения. На рисунке 6 представлен интерфейс Router.

Рисунок 6. Представление Router

Interactor

Включает в себя бизнес-логику для управления объектами данных(Entity). Функции, выполняющиеся в Interactor, не зависят от пользовательского интерфейса. На рисунке 7 представлен интерфейс с функцией запроса на сервер.

```
public interface IAsyncLoginInteractor {
    void validateCredentialsAsync(Context context, OnLoginFinishedListener listener, String username, String password);
}
```

Рисунок 7. Представление Interactor

Entity

Это представление объектов данных, которыми может управлять только Interacor. Он никогда не передает данные уровню представления. На диаграмме классов Entity представлена как PreferenceHelper.

Безусловно построение приложения на архитектуре VIPER требует больше времени и большего количества задействованных классов, но приложение, построенное таким образом, представляется более простым и понятным с точки зрения чтения кода. Вся логика запросов на сервер и действий самого приложения отделена от пользовательского интерфейса и не содержится в одном файле (классе), это позволяет легче разобраться в коде и внедрить новый функционал добавляя элементы и новые классы, не переписывая весь код одного экрана.

Список литературы:

1. Книга VIPER – [Электронный ресурс] – Режим доступа: https://habrahabr.ru/company/rambler-co/blog/311248/.

2. Android VIPER на реактивной тяге – [Электронный ресурс] – Режим доступа: https://habrahabr.ru/company/rambler-co/blog/277003/.

3. Robert Martin, The Clean Architecture – [Электронный ресурс] – Режим доступа: https://8thlight.com/blog/uncle-bob/2012/08/13/the-clean-architecture.html.