УДК 000.41

# О ПОДХОДАХ К РАЗРАБОТКЕ МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ ФОРМИРОВАНИЯ ЛИЧНЫХ СПИСКОВ И ОТСЛЕЖИВАНИЯ РЕЛИЗОВ

*Л.В. Рудикова, В.А. Ломакин*

*В статье рассматривается архитектура и общие подходы к реализации мобильного приложения на платформе Android, которое предназначено для формирования личных списков игр. Приводятся основная функциональность программного обеспечения, связанного с разработкой данного мобильного приложения. Описываются общие модели разрабатываемого мобильного приложения.*

## Введение

В настоящее время индустрия компьютерных игр представляет собой, с одной стороны, совокупность различных компаний, сообществ и отдельных пользователей, а с другой – технологий и процессов, которые совместно образуют полный цикл производства (разработка, продажа, продвижение, потребление) компьютерных игр. Кроме того, компьютерные игры – это также и сектор экономики, и инструмент культуры [1].

По оценкам NewZoo [2] в 2013 году игровая индустрия занимала 71% мирового рынка и ее объем составил 75,5 млрд. долларов в 2013 году, а в ближайшие три года мировой рынок игр превысит отметку в 100 млрд. долларов.

На рост игрового рынка положительно повлияли несколько тенденций. Во-первых, увеличилось количество устройств, которые предназначены для игр: компьютеры, смартфоны, консоли (игровые приставки), планшеты и т.п. Во-вторых, растет популярность бесплатных игр, и появляются бизнес-модели по монетизации – добавление видеоконтента и элементов электронного спорта в игры становится частью новой стратегии разработчиков игр. В-третьих, – возможность глобализации: разработчик может распространять свою игру в Интернете по всему миру.

Основной тенденцией в развитии мирового рынка игровой индустрии является разработка под Android и iOS в силу популярности мобильных и планшетных устройств. По данным NewZoo [2], мобильные игры в 2013 составляли 23% общего рынка игр году, а в 2016 году будут аккумулировать более 30% совокупной выручки игрового рынка.

Особенностью современного игрового рынка является возможность одновременного существования как крупных компаний, выпускающим высокобюджетные игры-блокбастеры (игры AAA-класса), так и отдельных разработчиков, создающих авторские игры без финансирования (инди-игры).

Отметим также еще и социальный момент, связанный с игровой индустрией. Так, при правильном подходе, идеи, заложенные в компьютерные игры, гораздо лучше усваиваются детьми, чем идеи из книг или мультфильмов.

Это означает, что многие факты исторического, научного и повседневного плана могут легко переноситься в мир игр и выполнять соответствующую образовательно-познавательную функции. Таким образом, культурный потенциал игр ещё только начинает раскрываться и развиваться, однако уже сейчас понятно, что потенциал является огромным.

Итак, исходя из вышеизложенного, можно сделать вывод о том, что в ближайшее время численность людей, увлекающихся компьютерными играми или соответствующими практико-ориентированными игровыми приложениями, будет возрастать. В силу этого разработка предлагаемого мобильного приложения «Gaming Assistant» под ОС Android является актуальной, так как она позволяет создавать и настраивать списки личных игр, которые могут быть интегрированы с социальными сетями, а также дает возможность в дальнейшем получать различные итоговые выборки и обработки по накопленным данным, представляется новым и перспективным аспектом в линейке похожих предложений.

**Особенности мобильного приложения «Gaming Assistant»**

Основной функционал приложения «Gaming Assistant» для формирования личного списка игр и отслеживания их релизов на базе ОС Android включает следующие возможности:

- поиск игр;
- формирование личного списка;
- поиск игр по личному списку;
- добавление игр в различные категории личного списка;
- хранение информации в локальной базе данных для offline использования;
- система автоматического обновления для подержания информации в актуальном состоянии;
- сохранение пользовательских данных на веб-сервере;
- возможность настройки приложения, а также хранение введенных настроек;
- регистрация и авторизация пользователей через социальную сеть VKontakte;
- поддержка функций социальной сети VKontakte;
- наличие вкладок для перехода по разделам приложения;
- удобный интерфейс пользователя для работы с приложением (меню, поиск, отображение личного списка игр по категориям);
- форма для отображения сохраненных пользователем игр в различные категории (Пройденные игры, Список желаемого и т.д.);
- форма для отображения поиска;
- форма для просмотра списка вышедших игр, а также еще тех, которые еще не вышли.

При использовании мобильного клиента «Gaming Assistant» пользователи могут производить следующие действия:

- просматривать информацию, относящуюся к конкретной игре, т.е. иметь возможность просмотреть название игры, дату выхода, описание и т.д.;
- добавлять выбранные игры в пользовательский список игр;
- получать список игр для выбранного личного списка игр.

Кроме того, одним из главных приоритетов приложения «Gaming Assistant» является организация интуитивно понятного и удобного интерфейса.

Отметим также, что на рынке игровой индустрии имеется ряд приложений (например, My Game Collection (Tracker), My Game Collection (Tracker), Game Keeper Library Tracker), которые предназначены для составления списков игр пользователей.

Несмотря на то, что рассмотренные выше приложения обладают достаточной функциональностью, у них имеется и ряд серьезных недостатков. Практически, все приложения указанного вида имеют отдельный платный функционал. Следует также отметить, что отсутствует сохранение пользовательских данных на веб-сервере, что вызывает неудобство при использовании приложений. Отсутствует также авторизация пользователя для получения доступа к своему списку игр. Кроме того, пользователи не могут делиться своими списками игр и просматривать списки игр друзей.

Таким образом, исходя из изложенного выше, явно просматривается актуальность предлагаемой разработки «Gaming Assistant» с расширенной функциональностью для формирования личного списка игр и авторизацией пользователей через социальную сеть.

Для реализации мобильного приложения были выбраны следующие программные средства: ОС Android; СУБД SQLite; язык программирования Java; среда разработки Android Studio;

**Этапы разработки приложения**

Работу над реализацией приложения «Gaming Assistant» можно разбить на следующие этапы:

- проектирование веб-сервиса для обновления информации, связанной с играми;
- проектирование базы данных;
- получение модели функций разрабатываемого приложения;
- проектирование интерфейса;
- реализация базы данных и логики взаимодействия с ней;
- реализация интерфейса активностей, используемых в приложении;
- создание классов, используемых для привязки полученных данных к представлениям;
- реализация связи базы данных с интерфейсом;
- реализация интеграции с социальными сетями.

При первом входе в систему, приложение создает базу данных, а также необходимые таблицы, которые заполняются данными. В первую очередь, пользователю необходимо провести авторизацию через социальную сеть ВКонтакте. В процессе работы приложения модифицируется содержимое базы данных, информация, полученная с веб-сервера, заносится в локальную базу

данных. Это необходимо для доступа пользователя к собственным данным при отсутствии соединения с сетью. Следует отметить, что при добавлении данных пользователем в базу при отсутствии доступа к сети, в дальнейшем при наличии доступа в сеть, произойдет синхронизация данных с веб-сервером.

**Общие подходы к реализации приложения**

Рассмотрим основные компоненты архитектуры мобильного приложения для организации личного списка игр и отслеживания их релизов (рисунок 1).

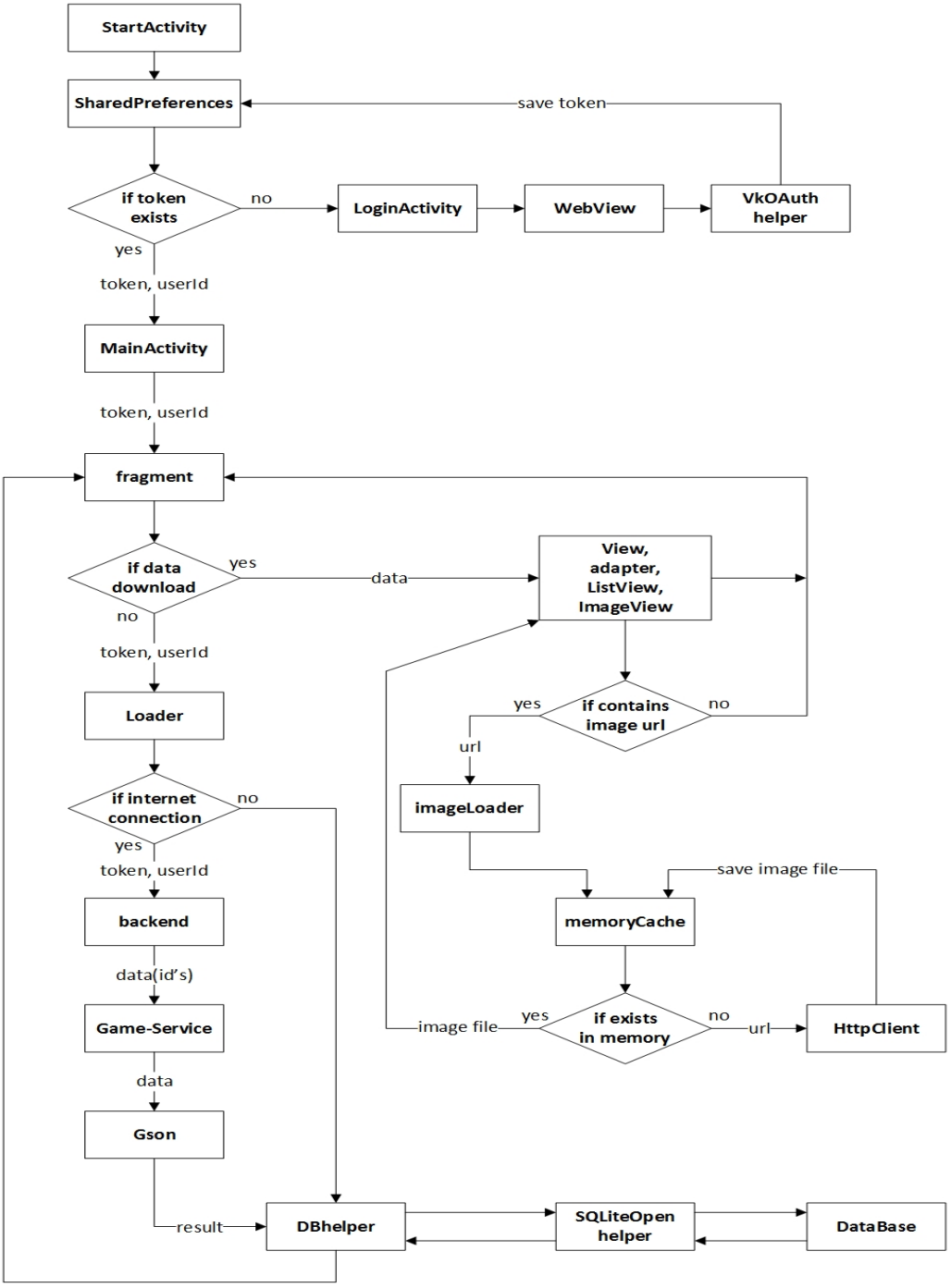


Рис. 1. Общая архитектура приложения «Gaming Assistant»

Приложение «Gaming Assistant» состоит из следующих частей:

- базы данных, в которой хранятся данные;
- веб-сервера, сохраняющего пользовательские данные;

- пользовательского интерфейса, предоставляющего пользователю доступ к функциям приложения;
- сервиса, который осуществляет поиск игр по заданным критериям;
- авторизации через социальную сеть ВКонтаке.

Для того чтобы пользователь мог использовать функционал приложения, ему необходимо совершить авторизацию через социальную сеть ВКонтаке. Авторизация реализована через технологию OAUTH 2.0, что подразумевает генерацию token-авторизации для работы с функционалом ВКонтакте. Прежде всего, приложение должно получить token пользователя, который хочет авторизироваться в приложении. Работа приложения начинается со StartActivity (наследуется от класса Activity – класс приложения, представляющий визуальный интерфейс), в ходе работы которого проверяется, был ли пользователь уже авторизирован или нет. Если пользователь уже авторизовался, то необходимо вызвать MainActivity, чтобы пользователь получил доступ к основному функционалу приложения. Однако, если при обращении к SharedPreferences (используется в приложении для создания именованных ассоциативных массивов типа «ключ – значение», которые могут быть использованы различными компонентами приложения) отсутствует token, то пользователю предлагается произвести авторизацию под своим логином социальной сети ВКонтакте.

Для авторизации StartActivity вызывает LoginActivity. Интерфейс авторизации реализован с помощью WebView (TODO). WebView обращается к VkOAuthHelper (класс, который инкапсулирует логику авторизации пользователя): в окне браузера отображаются поля для ввода логина и пароля. Если же у пользователя логин отсутствует, ему будет предложено завести новый логин. После ввода комбинации логина-пароля, пользователю будет предложено разрешить приложению использовать свою личную информацию. Если же авторизация прошла успешно, то полученный token из LoginActivity передается в StartActivity. Однако если при авторизации произошла ошибка (т.е. после запроса авторизовать пользователя, веб-сервер вернул ошибку), то сообщение ошибки передается в StartActivity. После успешной авторизации LoginActivity вновь вызывает StartActivity, а само оно уничтожается.

После того как был получен token из LoginActivity, StartActivity обращается к SharedPreferences для записи полученого token в хранилище типа «ключ-значение». Это необходимо для того, чтобы пользователь каждый раз не вводил свою комбинацию логин/пароль. Приложение использует сохраненный token для работы с функционалом ВКонтакте. Далее StartActivity вызывает MainActivity для предоставления основного функционала приложения пользователю. StartActivity помещает полученный token в механизм Bundle и при вызове MainActivity также передает его. После успешных операций StartActivity уничтожается.

MainActivity – это основной экран отображения приложения, в котором все экраны приложения реализованы через фрагменты (fragments). Для переходов между фрагментами используется navigationDrawer. После

перехода в MainActivity, используется fragmentManager, который отображает выбранный фрагмент. MainActivity также передает token в выбранный фрагмент для работы с социальной сетью. После того как фрагмент был вызван, он обращается к Loader для загрузки данных. Далее Loader проверяет наличие Интернета и, если он отсутствует, то все данные читаются из локальной базы данные, а, если соединение присутствует, то Loader обращается к веб-серверу для получения данных.

Loader обращается к веб-серверу Firebase, где в качестве идентификации пользователя используется User-id, полученный из социальной сети ВКонтакте. После загрузки данных Loader обращается к сервису с играми для загрузки информации об играх. Полученные данные хранятся в формате JSON, поэтому для удобной работы с ними используется библиотека GSON которая структурирует данные для удобного пользования. После загрузки данных Loader обращается к DBhelper для того, чтобы записать полученные с веб-сервера данные в локальную базу данных. Это необходимо для использования приложения без соединения с Интернетом.

DBhelper взаимодействует с базой данных Sqlite, используя SQLiteOpenHelper, который упрощает работу с базой данных. После сохранения данных в базу данных, полученная информация передается непосредственно в класс фрагмента, который описывает интерфейс. Во фрагменте данные присваиваются View (отображениям), а если на экране необходимо отобразить список элементов, то используется ListView для работы/, с которым используется adapter, в который фрагмент также передает необходимые данные.

Для загрузок картинок используется класс ImageLoader, инкапсулирующий логику загрузки и кэширования картинок. Фрагмент передает в ImageLoader адрес картинки в сети и view, в который необходимо загрузить файл картинки. ImageLoader использует LRUcache для хранения картинок в оперативной памяти устройства. После того как был передан Url и View, загрузчик картинок, прежде всего, проверяет наличие файла картинки в кэше: если картинка отсутствует, то загрузчик, используя HttpClient, загружает ее. После загрузки картинка помещается в кэш. Для того, чтобы картинка не загружалась дважды и при следующем обращении к картинке, она уже не будет загружаться из сети, а будет извлекаться из кэша, что обеспечивает экономию трафика. После загрузки картинка присваивается указному ImageView.

Отметим, что пользовательский интерфейс реализован с помощью шаблона проектирования MVVM, включающего три составные части. Модель (Model), аналогично классическому шаблону MVC, представляет собой фундаментальные данные, необходимые для работы приложения. Представление (View) – это графический интерфейс, т.е. все необходимые элементы управления, доступные пользователю. Представление является подписчиком на событие изменения значений свойств или команд, предоставляемых Моделью представления. В случае если в Модели

представления изменилось какое-либо свойство, то она оповещает всех подписчиков об этом, и Представление, в свою очередь, запрашивает обновленное значение свойства из Модели представления. В случае если пользователь воздействует на какой-либо элемент интерфейса, Представление вызывает соответствующую команду, предоставленную Моделью представления. Модель представления (View Model) является, с одной стороны, абстракцией Представления, а, с другой, предоставляет обертку данных из Модели, которые подлежат связыванию. Таким образом, она содержит Модель, преобразованную к Представлению, а также содержит в себе команды, которыми может пользоваться Представление, чтобы влиять на Модель.

Интерфейс реализуется с помощью классов, унаследованных от `Activity`, которые предоставляют доступ к элементам интерфейса и связывают их с данными. На рисунке 2 представлены окно приветствия и окно авторизации. На них использованы элементы интерфейса: `TextView` – для отображения надписей, `Button` – как элемент управления, `WebView` – для отображения окна браузера.

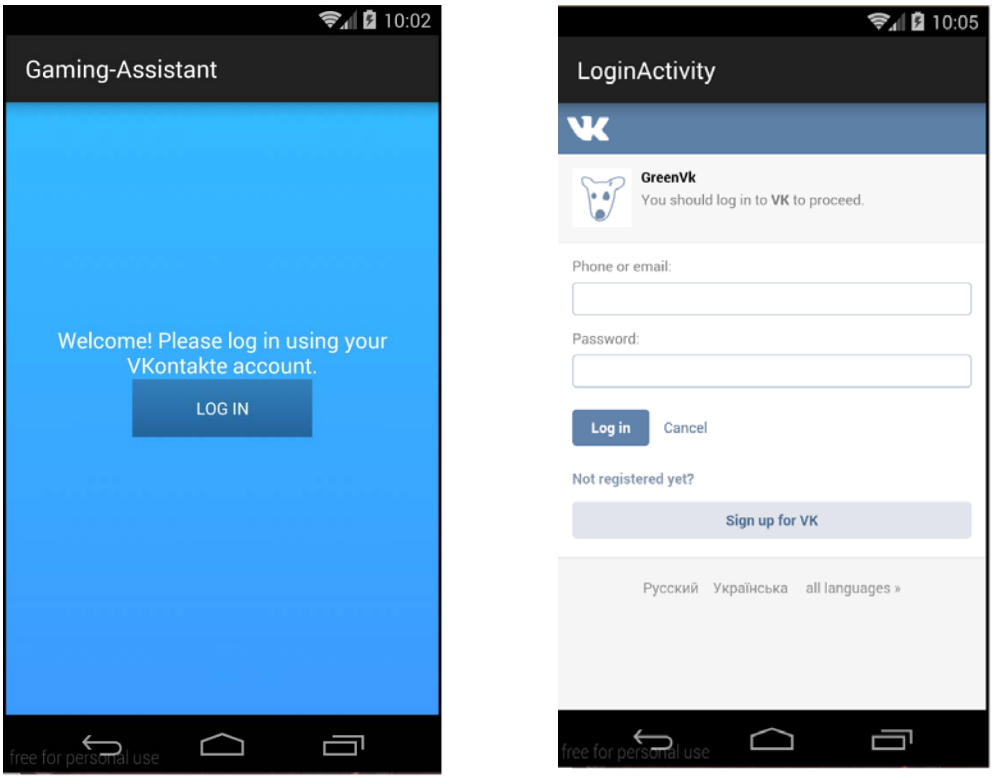


Рис.2. Окно приветствия и окно авторизации

При проектировании приложения и создании пользовательского интерфейса, было принято решение использовать в качестве одного из элементов управления – `ListView`. Этот компонент представляет собой список с возможностью прокручивания элементов. Данный элемент использовался для реализации списка друзей пользователя и списка игр.

**Выводы**

При разработке функционала приложения «Gaming Assistant» реализован механизм для регистрации и входа пользователя, интерфейс пользователя для

работы с приложением, функции для выполнения операций с приложением. Кроме того, пользователям доступны также следующие возможности: таблицы со списками игр, платформ, издателей, жанров; поиск с использованием фильтра; сохранение выбранных игр в память телефона и на веб-сервер; интерфейс пользователя для работы с приложением (меню, поиск с фильтром, отображение личного списка игр, отображение списка игр (по названию, издателю, жанру, платформе); форма для отображение сохраненных пользователем игр в различные категории; форма для отображения поиска с фильтрами; форма для просмотра списка вышедших игр а также еще не вышедших; веб-сервер обновлений списка игр и сохранения пользовательских данных и т.д.

Таким образом, разработанное приложение «Gaming Assistant» позволяет просматривать имеющиеся игры с создания личного списка. Данное приложение может использоваться для быстрого доступа без сети Интернет.

В дальнейшем приложение может быть адаптировано и использоваться для составления личных списков и отслеживания релизов достаточно широкого круга приложений различных предметных областей и с использованием авторизации через различные социальные сети.

**Список литературы**


1. Компьютерные игры как искусство [Электронный ресурс]. – Доступ http://gamesisart.ru/game_dev_structure.html – Дата доступа: 20.03.2016.
2. Игрушки без внимания [Электронный ресурс] / Российская Бизнес-газета. – Режим доступа: http://www.rg.ru/2014/12/02/igry.html. – Дата доступа: 20.03.2016.



*Рудикова Лада Владимировна, заведующий кафедрой современных технологий программирования Учреждения образования «Гродненский государственный университет имени Янки Купалы», кандидат физико-математических наук, доцент, rudikowa@gmail.com*

*Ломакин Всеволод Александрович, магистрант кафедры современных технологий программирования Учреждения образования «Гродненский государственный университет имени Янки Купалы», bloodseva@ya.ru*