О РАЗРАБОТКЕ МОБИЛЬНОГО ПРИЛОЖЕНИЯ ДЛЯ ИНТЕГРАЦИИ С СОЦИАЛЬНЫМИ СЕТЯМИ

В. А. Ломакин, Л. В. Рудикова
Кафедра современных технологий программирования, Учреждение образования «Гродненский государственный университет имени Янки Купалы»
Гродно, Республика Беларусь
E-mail: rudikowa@gmail.com, lomakin_va_10@mf.grsu.by

Излагаются общие подходы к реализации мобильного приложения на платформе ОС Android, которое предназначено для формирования личных списков игр. Приводится основная функциональность программного обеспечения, связанного с разработкой указанного мобильного приложения.

Введение

В современном мире широкое распространение приобрели смартфоны благодаря удобству и простоте использования, а также наличию большого количества различных приложений. Кроме того, перед разработчиками для мобильных платформ с каждым днем открываются различные возможности, позволяющие создавать необходимые программы с гибкой функциональностью и интуитивно понятным интерфейсом. С другой стороны, индустрия компьютерных игр, для которой характерны экономический и культурный аспекты, занимает одно из первых мест по производству и прибыльности в мировом пространстве. Особенностью игровой индустрии является то, что наряду с крупными компаниями, специализирующимися на играх определенного плана, на рынке также присутствуют и отдельные самостоятельные разработчики. Отметим также и тот факт, что через компьютерные игры можно проводить те или иные идеи, а также преподносить книги, факты, научные и исторические дан-ные. Более того, при правильном подходе, идеи, заложенные в компьютерные игры, гораздо лучше усваиваются детьми, чем идеи из книг или мультфильмов.

Итак, в настоящее время на рынке выпускается большой спектр игр различных жанров от издателей, которые специализируются на разработке соответствующего программного обеспечения. В свою очередь, пользователи заинтересованы в составлении личного списка игр с последующим быстрым доступом к созданным спискам и возможностью гибкой настройки.

I. 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 представ-лениям; 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.

II. Общие подходы к архитектуре реализации приложения

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

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 ВКонтакте.

Для того чтобы пользователь мог использовать функционал приложения, ему необходимо совершить авторизацию через социальную сеть ВКонтакте. 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 поль-

зователь 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 опера-ций 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.

Отметим что, для загрузок картинок используется класс ImageLoader, инкапсулирующий логику загрузки и кэширования изображений, а для реализации пользовательского интерфейса использован шаблон проектирования MVVM.

Для разработки приложения выбран объектно-ориентированный язык Java. Android является открытой платформой и позволяет разработчикам свободно писать достаточно гибкие приложения [1]. Для хранения необходимой информации используется СУБД SQLite. Кроме того, при разработке используется Android SDK, который предоставляет необходимые инструменты для разработки мобильного приложения на платформе Android.

III. Выводы

Таким образом, разработанное приложение предназначен для широкого использования различными людьми, которые играют в компьютерные игры различной направленности и заинтересованы в составлении личного списка игр, который доступен через социальные сети. Приложение «Gaming Assistant» предоставляет возможность создания и настройки списка личных игр, которые могут быть интегрированы с социальными сетями.

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. Ломакин, В. А. О разработке мобильного приложения «Gaming Assistant» / В. А. Ломакин // Веб-программирование и Интернет-технологии WebConf 2015 : материалы 3-й Междунар. науч.-практ. конф., 12–14 мая 2015 г., Минск. – Минск : Изд. центр БГУ, 2015. – С. 174–175.