Приднестровский Государственный Университет

им. Т.Г. Шевченко

**физико-математический факультет**

**кафедра информатики и вычислительной техники**

Работа допущена к защите

заведующий кафедрой ИВТ

доцент Коровай А.В.\_\_\_\_\_\_\_\_\_

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2015 г.

**Тема: «Автоматизированная система для мобильных устройств «Обеспечение родительского контроля учащихся» на базе операционной системы Android».**

**КВАЛИФИКАЦИОННАЯ РАБОТА**

**ВЫПОЛНИЛ:**

Студент стационарного отделения

V-го курса 504 группы

Специальности «Информатика»

**Гагауз Сергей Васильевич**

**НАУЧНЫЙ РУКОВОДИТЕЛЬ:**

старший преподаватель кафедры ИВТ

**Кураксин Сергей Валерьевич**

г. Тирасполь

2015г.

**АННОТАЦИЯ**

Данная дипломная работа на тему «Школьная автоматизированная система для мобильных устройств по информированию родителей на базе операционной системы Android» выполнена студентом 5-го курса Приднестровского государственного университета им. Т.Г. Шевченко физико-математического факультета специальности «Информатика и Информационно-Коммуникационные Технологии в образовании».

Квалификационная работа выполнена на листах формата А4, содержит (69) страниц, (19) рисунков, (2) схемы, (2) таблицы, (20) литературных источников, (1) приложение. Работа состоит из введения, (2) глав, заключения, списка изучаемой литературы и приложения.

В первой главе проводится анализ, рассматриваются основные инструменты, используемые при разработке, их теоретические аспекты и области применений.

Во второй главе сформулирована постановка задачи квалификационной работы, рассмотрен процесс создания приложения от прототипирования и скетчей до готового программного продукта с демонстрацией рабочих областей.

**ОГЛАВЛЕНИЕ**

[**ВВЕДЕНИЕ** 4](#_Toc422307003)

[**1.** **АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ** 6](#_Toc422307004)

[1.1. Операционная система Android. Общие сведения 6](#_Toc422307005)

[1.2. Java*.* История языка. Основные возможности 10](#_Toc422307006)

[1.3. SQLite. Компактные базы данных. Область применения 13](#_Toc422307007)

[1.4. PHP. Краткое введение в язык 15](#_Toc422307008)

[1.5. Интегрированная среда разработки Android Studio 17](#_Toc422307009)

[**2.** **ПРОГРАММИНАЯ РЕАЛИЗАЦИЯ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ** 18](#_Toc422307010)

[2.1. Клиентская часть 18](#_Toc422307011)

[2.1.1. Основные функции приложения 18](#_Toc422307012)

[2.1.2. Функциональное проектирование приложения 20](#_Toc422307013)

[2.1.3. Графическое проектирование приложения 23](#_Toc422307014)

[2.2. Серверная часть 28](#_Toc422307015)

[2.2.1. Проектирование базы данных 28](#_Toc422307016)

[**ЗАКЛЮЧЕНИЕ** 33](#_Toc422307017)

[**СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ** 34](#_Toc422307018)

[**ПРИЛОЖЕНИЕ (ЛИСТИНГ ПРОГРАММЫ)** 36](#_Toc422307019)

# **ВВЕДЕНИЕ**

Потребность людей быть в курсе событий выводит технологии на основную линию по разработке новых цифровых устройств. Неудобство использования компьютеров и ноутбуков обуславливает появление различных мини-компьютеров, смартфонов и коммуникаторов, в основе которых лежит все та же операционная система. Основную долю рынка мобильных ОС на сегодня заняли платформы от Apple и Google (iOS и Android соответственно). Данные платформы могут функционировать полноценно лишь при условии, что для них были разработаны мобильные приложения.

Android – OC для коммуникаторов, планшетов, медиа проигрывателей, цифровых фоторамок, наручных часов, TV приставок, очков (Google Glass), нетбуков и смартбуков, основанная на ядре Linux. Целевым языком для разработки является Java, однако существуют библиотеки, предоставляющие возможность вести разработку нативных приложений на языке С++ (NDK).

**Задание дипломной работы** – разработать программное обеспечение для платформы Android: информационное приложения для родителей и учителей, с возможностью просмотра или добавлении новостей как конкретного класса, так и учебного заведения, статистики и иной информации.

**Актуальность** данной темы объяснятся следующими причинами:

Во-первых, использование мобильных приложений становится неотъемлемой частью нашей повседневной жизни.

Во-вторых, использование устарелых методов обратной связи «учитель-родитель» становится все более не эффективным. Внедрение школьных сайтов и электронных дневников не исправило эту ситуацию.

**Целью дипломной работы** является разработка такого приложения для мобильных устройств и планшетов на основе OS Android. Требуется спроектировать и запрограммировать программное обеспечение с поддержкой большинства современных устройств под управлением OS Android v2.3 «GINGERBREAD» и выше. Одной из характеристик разрабатываемого приложения является применение процессов и сервисов, не зависящих от самой программы и выполняющих обработку данных в фоновом режиме.

**Объектом** исследования является операционная система Android и технологии разработки мобильных приложений под нее.

**Предметом** исследования является мобильное приложение под операционную систему Android.

**Методы исследования:**

* анализ научной литературы;
* изучение и обобщение опыта разработки мобильных приложений.

В ходе выполнения работы были рассмотрены и решены следующие задачи:

- Были получены знания о принципах разработки для мобильных платформ (xml разметка, жизненный цикл, особенности отладки приложений, написанных для мобильных устройств, разработка сервисов и фоновых задач). Целевой платформой является платформа Android v5.1 «Lollipop». Основным языком для разработки – язык Java 1.7 расширенное издание;

- Для реализации поставленной задачи использована среда Android Studio.

# **АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

## **Операционная система Android. Общие сведения**

***Android*** («Андро́ид»), *как и говорилось ранее* — операционная система для смартфонов, планшетных компьютеров, электронных книг, цифровых проигрывателей, наручных часов, игровых приставок, нетбуков, смартбуков, очков Google и других устройств. В будущем планируется поддержка автомобилей и телевизоров. Основана на ядре Linux и собственной реализации виртуальной машины Java от Google. Изначально разрабатывалась компанией Android Inc., которую затем купила Google. Впоследствии Google инициировала создание альянса Open Handset Alliance (OHA), который сейчас занимается поддержкой и дальнейшим развитием платформы. Android позволяет создавать Java-приложения, управляющие устройством через разработанные Google библиотеки. Android Native Development Kit позволяет портировать (но не отлаживать) библиотеки и компоненты приложений, написанные на Си и других языках.

В 86 % смартфонов, проданных во втором квартале 2014 года, была установлена операционная система Android.

Приложения под операционную систему Android являются программами в нестандартном байт-коде для виртуальной машины Dalvik, для них был разработан формат установочных пакетов \*.APK. Для работы над приложениями доступно множество библиотек: Bionic (библиотека стандартных функций, несовместимая с glibc); мультимедийные библиотеки на базе PacketVideo OpenCORE (поддерживают такие форматы, как MPEG-4, H.264, MP3, AAC, AMR, JPEG и PNG); SGL (движок двухмерной графики); OpenGL ES1.0 ES 2.0 (движок трёхмерной графики); Surface Manager (обеспечивает для приложений доступ к 2D/3D); WebKit (готовый движок для веб-браузера; обрабатывает HTML, JavaScript); FreeType (движок обработки шрифтов); SQLite (легковесная СУБД, доступная для всех приложений); SSL (протокол, обеспечивающий безопасную передачу данных по сети). По сравнению с обычными приложениями Linux приложения Android подчиняются дополнительным правилам: Content Providers — обмен данными между приложениями; Resource Manager — доступ к таким ресурсам, как файлы XML, PNG, JPEG; Notification Manager — доступ к строке состояния; Activity Manager — управление активными приложениями.

Google предлагает для свободного скачивания инструментарий для разработки (Software Development Kit), который предназначен для x86-машин под операционными системами Linux, Mac OS X (10.4.8 или выше), Windows XP, Windows Vista и Windows 7. Для разработки требуется JDK 5 или более новый.

Разработку приложений для Android можно вести на языке Java (не ниже Java 1.5). Существует плагин для Eclipse — Android Development Tools (ADT), предназначенный для Eclipse версий 3.3—3.7. Также существует плагин для IntelliJ IDEA, облегчающий разработку Android-приложений, и для среды разработки NetBeans IDE, который, начиная с версии NetBeans 7.0, перестал быть экспериментальным, хоть пока и не является официальным. Кроме того, существует Motodev Studio for Android — комплексная среда разработки на базе Eclipse, позволяющая работать непосредственно с Google SDK.

В 2009 году в дополнение к ADT был опубликован Android Native Development Kit (NDK) — пакет инструментариев и библиотек, позволяющий реализовать часть приложения на языке С/С++. NDK рекомендуется использовать для разработки участков кода, критичных к скорости.

В 2013 году Google представила новую среду разработки Android Studio, основанная на IntelliJ IDEA от JetBrains.

В 2013-м году состоялся релиз Embarcadero RAD Studio — XE5. Возможность разработки нативных приложений для платформы Android. Процесс создания Android приложения не требует дополнительных устройств, кроме, собственно, Android устройства (в принципе, можно обойтись и эмулятором).

***Интересные факты***

* Кодовое имя каждой версии операционной системы Android, начиная с версии 1.5, представляет собой название какого-либо десерта. Первые буквы наименований в порядке версий соответствуют буквам латинского алфавита: 1.5 Cupcake («кекс»), 1.6 Donut («пончик»), 2.0/2.1 Eclair («эклер» или «глазурь»), 2.2 Froyo (сокращение от «замороженный йогурт»), 2.3 Gingerbread («имбирный пряник»), 3.0 Honeycomb («медовые соты»), 4.0 Ice Cream Sandwich («брикет мороженого»), 4.1/4.2/4.3 Jelly Bean («желейная конфета»), 4.4 KitKat (в честь одноимённого бренда шоколадных батончиков), 5.0/5.1 Lollipop ("леденец на палочке") и грядущая версия 6.0 M (превью версия, доступная только для разработчиков).
* Специально для платформы Android было создано семейство шрифтов Droid и Roboto.
* На официальных сайтах Android до сих пор не указаны минимальные требования к аппаратным средствам для запуска операционной системы (присутствуют только требования к аппаратным средствам для пакета разработки под Android).
* По данным на февраль 2011 года устройства с ОС Android уверенно лидировали в списке самых популярных смартфонов Великобритании, оставив позади iPhone 4 c iOS. По прогнозам экспертов, это первый шаг ОС Android к глобальному лидерству на рынке мобильных операционных систем.
* Компания Microsoft, продвигающая альтернативную ОС Windows Phone, получает лицензионные отчисления от производителей техники и ПО для Android. При этом доход компании Microsoft, получаемый таким образом, более чем в 2 раза выше, чем доход компании-разработчика, то есть Google.
* В версиях Android 2.3 и выше есть пасхальное яйцо. Чтобы его запустить, нужно зайти в «Настройки», потом зайти в «Информация об Android», найти функцию «Версия Android», и быстро нажимать на неё несколько раз (чаще всего 3 раза). На экране в версии 2.3 появится рисунок; 4.0 — один Android, который увеличивается, а затем появляется много летающих; 4.1, 4.2 и 4.3 — летающие конфеты; в 4.4 — вращающаяся буква «K», затем надпись «Android» в стиле батончика KitKat, затем — появляются плитки в стиле Windows с логотипами предыдущих версий Android; в 5.0 сначала появляется леденец с надписью «Lollipop», затем откроется мини-игра в стиле «Flappy Bird».
* В версиях Android 4.2 и выше по умолчанию отключены инструменты разработчика, для включения нужно семь раз нажать на номер сборки.

## **Java*.* История языка. Основные возможности**

***Java*** — объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems (в последующем приобретённой компанией Oracle). Приложения Java обычно транслируются в специальный байт-код, поэтому они могут работать на любой виртуальной Java-машине вне зависимости от компьютерной архитектуры. Дата официального выпуска — 23 мая1995 года.

Изначально язык назывался Oak («Дуб») разрабатывался Джеймсом Гослингом для программирования бытовых электронных устройств. Впоследствии он был переименован в Java и стал использоваться для написания клиентских приложений и серверного программного обеспечения. Назван в честь марки кофе Java, которая, в свою очередь, получила наименование одноимённого острова (Джава), поэтому на официальной эмблеме языка изображена чашка с парящим кофе. Существует и другая версия происхождения названия языка, связанная с аллюзией на кофе-машину, как пример бытового устройства, для программирования которого изначально язык создавался.

Программы на Java транслируются в байт-код, выполняемый виртуальной машиной Java (JVM) — программой, обрабатывающей байтовый код и передающей инструкции оборудованию как интерпретатор.

Достоинством подобного способа выполнения программ является полная независимость байт-кода от операционной системы и оборудования, что позволяет выполнять Java-приложения на любом устройстве, для которого существует соответствующая виртуальная машина. Другой важной особенностью технологии Java является гибкая система безопасности, в рамках которой исполнение программы полностью контролируется виртуальной машиной. Любые операции, которые превышают установленные полномочия программы (например, попытка несанкционированного доступа к данным или соединения с другим компьютером) вызывают немедленное прерывание.

Часто к недостаткам концепции виртуальной машины относят снижение производительности. Ряд усовершенствований несколько увеличил скорость выполнения программ на Java:

- применение технологии трансляции байт-кода в машинный код непосредственно во время работы программы (JIT-технология) с возможностью сохранения версий класса в машинном коде,

- широкое использование платформенно-ориентированного кода (native-код) в стандартных библиотеках,

- аппаратные средства, обеспечивающие ускоренную обработку байт-кода (например, технология Jazelle, поддерживаемая некоторыми процессорами фирмы ARM).

По данным сайта shootout.alioth.debian.org, для семи разных задач время выполнения на Java составляет в среднем в полтора-два раза больше, чем для C/C++, в некоторых случаях Java быстрее, а в отдельных случаях в 7 раз медленнее. С другой стороны, для большинства из них потребление памяти Java-машиной было в 10-30 раз больше, чем программой на C/C++. Также примечательно исследование, проведённое компанией Google, согласно которому отмечается существенно более низкая производительность и бо́льшее потребление памяти в тестовых примерах на Java в сравнении с аналогичными программами на C++.

Идеи, заложенные в концепцию и различные реализации среды виртуальной машины Java, вдохновили множество энтузиастов на расширение перечня языков, которые могли бы быть использованы для создания программ, исполняемых на виртуальной машине. Эти идеи нашли также выражение в спецификации общеязыковой инфраструктуры CLI, заложенной в основу платформы .NET компанией Microsoft.

***Основные возможности:***

* автоматическое управление памятью;
* расширенные возможности обработки исключительных ситуаций;
* богатый набор средств фильтрации ввода-вывода;
* набор стандартных коллекций: массив, список, стек и т.п.;
* наличие простых средств создания сетевых приложений (в том числе с использованием протокола RMI);
* наличие классов, позволяющих выполнять HTTP-запросы и обрабатывать ответы;
* встроенные в язык средства создания многопоточных приложений;
* унифицированный доступ к базам данных:
* на уровне отдельных SQL-запросов — на основе JDBC, SQLJ;
* на уровне концепции объектов, обладающих способностью к хранению в базе данных — на основе Java Data Objects (англ.) и Java Persistence API;
* поддержка обобщений (начиная с версии 1.5);
* параллельное выполнение программ.

## **SQLite. Компактные базы данных. Область применения**

**SQLite** — компактная встраиваемая реляционная база данных. Исходный код библиотеки передан в общественное достояние. В 2005 году проект получил награду Google-O’Reilly Open Source Awards.

Слово «встраиваемый» означает, что SQLite не использует парадигму клиент-сервер, то есть движок SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а предоставляет библиотеку, с которой программа компонуется и движок становится составной частью программы. Таким образом, в качестве протокола обмена используются вызовы функций (API) библиотеки SQLite. Такой подход уменьшает накладные расходы, время отклика и упрощает программу. SQLite хранит всю базу данных (включая определения, таблицы, индексы и данные) в единственном стандартном файле на том компьютере, на котором исполняется программа. Простота реализации достигается за счёт того, что перед началом исполнения транзакции записи весь файл, хранящий базу данных, блокируется; ACID-функции достигаются в том числе за счёт создания файла журнала.

Несколько процессов или потоков могут одновременно без каких-либо проблем читать данные из одной базы. Запись в базу можно осуществить только в том случае, если никаких других запросов в данный момент не обслуживается; в противном случае попытка записи оканчивается неудачей, и в программу возвращается код ошибки. Другим вариантом развития событий является автоматическое повторение попыток записи в течение заданного интервала времени.

В комплекте поставки идёт также функциональная клиентская часть в виде исполняемого файла sqlite3, с помощью которого демонстрируется реализация функций основной библиотеки. Клиентская часть работает из командной строки, позволяет обращаться к файлу БД на основе типовых функций ОС.

Благодаря архитектуре движка возможно использовать SQLite как на встраиваемых системах, так и на выделенных машинах с гигабайтными массивами данных.

Сама библиотека SQLite написана на C; существует большое количество привязок к другим языкам программирования, в том числе Delphi, C++, Java, C#, VB.NET, Python, Perl, PHP, PureBasic, Tcl (средства для работы с Tcl включены в комплект поставки SQLite), Ruby, Haskell, Scheme, Smalltalk, Lua и Parser, а также ко многим другим. Полный список существующих средств размещён на странице проекта.

Простота и удобство встраивания SQLite привели к тому, что библиотека используется в браузерах, музыкальных плеерах и многих других программах.

В частности, SQLite используют:

* Adobe Integrated Runtime — среда для запуска приложений (частично);
* Gears;
* Autoit;
* Фреймворк Qt;
* Платформа XUL на движке Gecko 1.9+, XULRunner 1.9+ и, потенциально, все приложения, основанные на этой платформе, в том числе:
  + Mozilla Firefox (начиная с версии 3.0);
  + Mozilla Thunderbird (начиная с версии 3.0);
  + Songbird;
  + SQLite Manager.
* Skype;
* Некоторые модели GPS-навигаторов Garmin;
* AndroidAPI.

## **PHP. Краткое введение в язык**

**PHP** (англ. PHP: Hypertext Preprocessor — «PHP: препроцессор гипертекста»; первоначально Personal Home Page Tool — «Инструменты для создания персональных веб-страниц»; произносится пи-эйч-пи) — скриптовый язык программирования общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков программирования, применяющихся для создания динамических веб-сайтов.

Язык и его интерпретатор разрабатываются группой энтузиастов в рамках проекта с открытым кодом. Проект распространяется под собственной лицензией, несовместимой с GNU GPL.

В области веб программирования, в частности серверная часть, PHP — один из популярных сценарных языков (наряду с JSP, Perl и языками, используемыми в ASP.NET) благодаря своей простоте, скорости выполнения, богатой функциональности, кроссплатформенности и распространению исходных кодов на основе лицензии PHP.

Популярность в области построения веб-сайтов определяется наличием большого набора встроенных средств для разработки веб-приложений. Основные из них:

* автоматическое извлечение POST и GET-параметров, а также переменных окружения веб-сервера в предопределённые массивы;
* взаимодействие с большим количеством различных систем управления базами данных (MySQL, MySQLi, SQLite, PostgreSQL, Oracle (OCI8), Oracle, MicrosoftSQL Server, Sybase, ODBC, mSQL, IBMDB2, Cloudscape и Apachе Derby, Informix, Ovrimos SQL, LotusNotes, DB++, DBM, dBase, DBX, FrontBase, FilePro, Ingres II, SESAM, Firebird/InterBase, Paradox File Access, MaxDB, Интерфейс PDO);
* автоматизированная отправка HTTP-заголовков;
* работа с HTTP-авторизацией;
* работа с cookies и сессиями;
* работа с локальными и удалёнными файлами, сокетами;
* обработка файлов, загружаемых на сервер;
* работа с XForms.

В настоящее время PHP используется сотнями тысяч разработчиков. Согласно рейтингу корпорации TIOBE, базирующемся на данных поисковых систем, в июне 2013 года PHP находился на 5 месте среди языков программирования. К крупнейшим сайтам, использующим PHP, относятся Facebook, Wikipedia и др.

Входит в LAMP — распространённый набор программного обеспечения для создания и хостинга веб-сайтов (Linux, Apache, MySQL, PHP).

## **Интегрированная среда разработки Android Studio**

**Android Studio** — это новая среда для разработки ПО под Android, основанная на IntelliJ IDEA. Она предоставляет новые возможности и качество, которых нет в Eclipse ADT. Android Studio – это официальная Android IDE. На вершине возможностей, которые вы ожидаете от IntelliJ, Android Studio предлагает:

* Гибкую Gradle-based систему построения.
* Вариативное построение и множественную генерацию \*.APK файлов.
* Расширенную поддержку шаблонов для Сервисов Google и различных типов устройств.
* Богатый редактор слоев с поддержкой редактирования тем.
* Lint инструменты для лучшей производительности, простоты использования, совместимости версий, и других проблем.
* Возможности ProGuard и подписывание приложений.
* Встроенная поддержка для Google Cloud Platform, что делает легким интеграцию Google Cloud Сообщений.

Следующая таблица демонстрирует некоторые ключевые отличия между Android Studio и Eclipse with ADT:

|  |  |  |
| --- | --- | --- |
| Возможность | Android Studio | ADT |
| Система построения | Gradle | Ant |
| Maven-based зависимости для сборки | Да | Нет |
| Вариативное построение и множественная генерация \*.APK файлов | Да | Нет |
| Продвинутый редактор и рефактор кода | Да | Нет |
| Графический редактор слоев | Да | Да |
| Подписывание \*.APK и управление хранилищем ключей | Да | Да |
| Поддержка NDK | Да | Да |

*Табл.1 – Сравнение двух сред разработки Android Studio и*

*Eclipse with ADT*

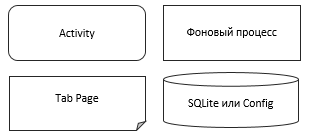
# **ПРОГРАММИНАЯ РЕАЛИЗАЦИЯ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ**

## **Клиентская часть**

### **Основные функции приложения**

К основным функциям приложения относятся:

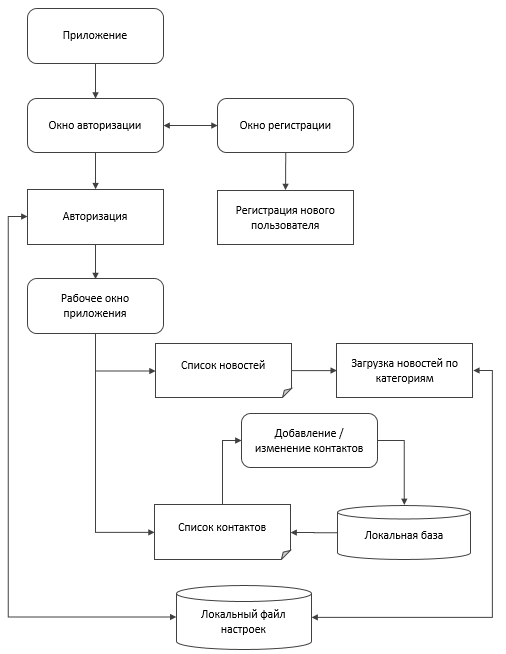
* Регистрация новых пользователей;
* Авторизация пользователей;
* Разграничение пользователей по уровню доступа;
* Вывод новостей согласно уровню доступа (общие новости учебного заведения, новости конкретного класса и т.д.);
* Имеет место понятие «Красной новости» – важная новость, которую трудно упустить из виду из-за характерного красного оформления;
* В зависимости от уровня доступа доступны следующие операции:
  + Добавление новостей;
  + Изменение новостей;
  + Удаление новостей;
  + Просмотр информации об авторе;
* Push-оповещения для общих новостей;
* Изменяемая контактная база с возможностью быстрого доступа к данным (звонки, SMS, отправка e-mail, поиск места проживания на карте по адресу прописки). Будет полезна для группирования контактов родителей класса без дополнительных переключений между приложениями;
* Локальный файл настроек для запоминания последней активности пользователя (запоминание авторизованных пользователей, отслеживание данных авторизованного пользователя).



*Рис 1. – Легенда функциональной схемы*

Общая функциональная схема показана на схеме 1. Схема является упрощенной и отображает основные функции приложения.

Стрелки, используемые при соединении элементов схемы, отображают одностороннюю или двустороннюю связь во взаимодействии компонентов, классов, методов.



*Схема 1 – Общая функциональная схема разработанного приложения*

### **Функциональное проектирование приложения**

***Окно авторизации***

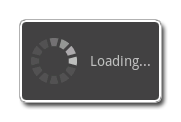
Приложение основывается на авторизации пользователей и согласно тому, являются ли они простыми гостями/родителями/учителями, будут выдаваться соответствующие правила пользования приложением.

На главном окне входа будут располагаться следующие графические компоненты:

* Поле ввода логина (E-mail адреса, под которым пользователь регистрировал аккаунт);
* Поле ввода пароля;
* Кнопка авторизации для существующих пользователей;
* Кнопка регистрации новых пользователей;
* Быстрая ссылка перехода на сайт;
* Быстрая ссылка для восстановления забытого пароля.

Прежде всего, приложение требует подключение к интернету. При запуске программы, происходит определение возможности доступа к сети. Если подключение отсутствует, то выводится соответствующее текстовое оповещение.

При отправке запроса для авторизации выводится окно прогресса с текстом «Авторизация…» (рис.2).



*Рис.2 – Общий вид компонента pregressDialog*

При попытке авторизации предусмотрены следующие текстовые сообщения:

* Сообщение о том, что не все поля заполнены (поле логина или поле пароля);
* Сообщение о том, что логин или пароль являются неверными;
* Сообщение о том, что e-mail адрес не подтвержден (не завершена процедура регистрации);
* Сообщение о том, что невозможно подключиться к удаленному серверу.
* Общие сообщения непредусмотренных затруднений при подключении.

При успешной авторизации пользователь перенаправляется в основное окно приложения, а на сервере фиксируется текущее время входа.

При нажатии на кнопку регистрации пользователь переходит в окно для регистрации новых пользователей.

***Окно регистрации***

Окно регистрации содержит следующие поля:

* Поле e-mail адреса (обязательное поле);
* Поле подтверждения e-mail адреса (обязательное, предусмотрена на случай опечатки, чтобы пользователь действительно был зарегистрирован под верным e-mail адресом);
* Поле пароля (обязательное поле, дополнительным условием является то, что пароль должен состоять минимум из 8 символов);
* Поле подтверждения пароля (обязательное поле);
* Контактное поле (номер телефона);
* Поле ФИО;
* Кнопка завершения регистрации.

В целях безопасности, предусмотрено шифрование паролей (применение криптографических библиотек) с использованием метода «добавление соли» в пароль – текстовый суффикс или префикс, который добавляется автоматически в самом приложении без ведома пользователя и затем шифруется, что предотвращает взлом аккаунтов, путем сравнения шифров паролей из базы с шифрами распространенных паролей и иными методами взлома.

При попытке завершения регистрации выводится то или иное текстовое уведомление:

* Уведомление о том, что e-mail’ы не совпадают;
* Уведомление о том, что пароли не совпадают;
* Уведомление о том, что пароль недостаточно длинный;
* Уведомление о том, что данный e-mail уже зарегистрирован;
* Уведомление о том, что аккаунт успешно создан, но необходимо завершить процесс регистрации и подтвердить почту, перейдя по ссылке в полученном письме.

***Основное окно приложения***

Главное окно приложения имеет две вкладки:

* Вкладка новостей;
* Вкладка контактов (записная книжка);

Во вкладке новостей загружаются последние обновления, интересные моменты из жизни школы, организационные уведомления.

Как и говорилось ранее, специально для глав родительских комитетов / учителей / администрации реализована возможность написания / изменения / удаления новостей прямо из мобильного приложения.

Вкладка с контактной книжкой разработана специально для родителей и классного руководства. Куда удобнее манипулировать данными из приложения, где будут в идеале только номера родителей одного класса, чем искать их по записной книжке, предустановленной с системой.

Специально добавлены дополнительные функции, отображающиеся при длительном нажатии на какой-либо контакт, что упростит отправление SMS, электронных сообщений, выполнение звонков и т.д.

### **Графическое проектирование приложения**

Первым шагом является подбор цветовой палитры и стиль, в котором выполняются все компоненты приложения. Решение пало на пастельные цвета и минималистичный (плоский) дизайн (акцентное приближение к целевой платформе 5.0 и выше, с характерным для неё «Материальным дизайном»).

Цветовая палитра была подобрана чтобы не напрягать и не раздражать глаза и создаёт общее приятное впечатление.



*Рис.3 – Примерная цветовая палитра приложения*



*Рис.4 – Примитивы кнопок при выбранной цветовой палитре*

Как можно увидеть на рис.4 – минимализм характеризуется простотой геометрических форм, лаконичностью цветов и малыми объёмами.

При разработке приложения были применены навыки работы в графических редакторах. На рис.5 показано изображение, вариации которого используются в качестве фона в самом приложении. На рис.6 показана иконка самого приложения:

|  |  |
| --- | --- |
|  |  |
| *Рис.5 – Основной фон, используемый в приложении* | *Рис.6 – Иконка приложения* |

Главная часть работы в этом направлении заключалась в прототипировании пользовательского интерфейса – создание чернового варианта. Именно из-за неверно выстроенного интерфейса могут возникать затруднения у пользователей.

Для создания гармоничного интерфейса были изучены и рассмотрены приложения для социальных сетей и популярных сервисов (ВКонтакте, Twitter, Instagram, Facebook и т.д.).

Выделены следующие основные пункты:

* Унификация элементов (дизайн, цветовая схема);
* Понятность цветов (пример: красный – предупреждение; желтый – акцент и т.д.);
* Отзывчивость на трансформации.

Любой пользователь, установивший приложение, сперва увидит окно авторизации и было принято решение вести разработку именно с него:

|  |  |
| --- | --- |
|  |  |
| *Рис.7 – Прототип главного окна* | *Рис.8 – Снимок экрана главного окна* |

Главная идея данного окна:

* выравнивание всех элементов по вертикали, что не вызывает графического хаоса и общего диссонанса (данный пункт является общим для всего приложения);
* отсутствие ActionBar’а, что придаёт увеличение обзора на экране приветствия;
* отзывчивость для любых трансформаций (смена ориентации экрана, появление клавиатуры, растяжение текстовых полей по высоте при большом количестве текста).

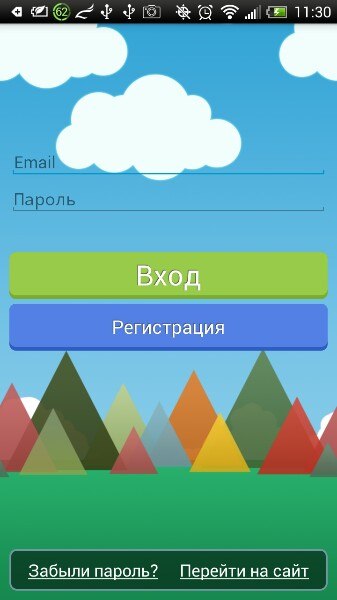
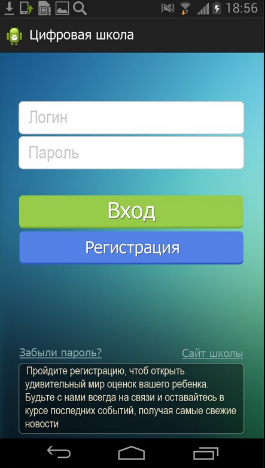
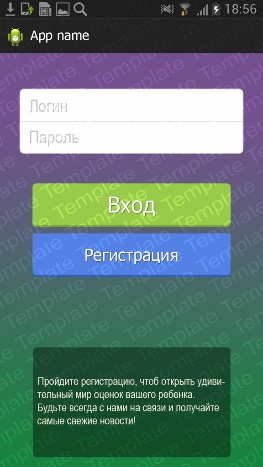
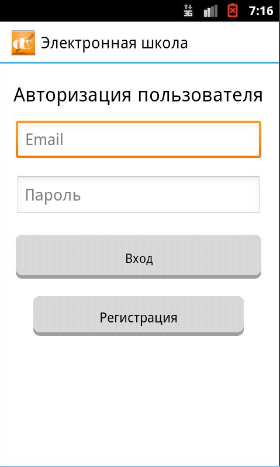
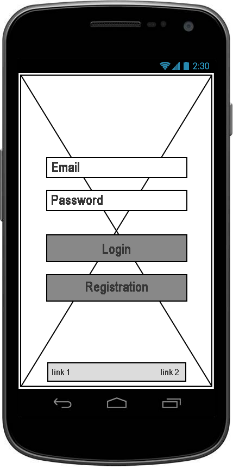
|  |  |
| --- | --- |
|  |  |
| *Рис.9 – Прототип главного окна при появлении клавиатуры* | *Рис.10 – Снимок экрана главного окна при появлении клавиатуры* |

При сравнении экрана с клавиатурой и без можно заметить «сплющивание» свободного места между элементами.

Аналогичным образом (начиная со скетчей и концептов) разрабатывались и согласовывались на основе пользовательских опросов все дальнейшие разработки пользовательского интерфейса.

На протяжении всего периода разработки графическая часть не раз претерпевала изменения кардинальные или косметические.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | > |  | > |  |
| *Рис.11 – Эволюция иконок для приложения*  *(хронологический порядок с лева на право)* | | | | |



*Рис.12 – Метаморфозы графического пользовательского интерфейса на примере окна авторизации*

|  |  |  |  |
| --- | --- | --- | --- |
| https://pp.vk.me/c622926/v622926303/31095/tvQNOht2Yp4.jpg | C:\Users\Сергей\AppData\Local\Microsoft\Windows\INetCache\Content.Word\2015-06-10_09-53-32.png |  |  |
| *Рис.13 – Окно авторизации* | *Рис.14 – Окно регистрации* | *Рис.15 – Главное окно, новости* | *Рис.16 –Контактная книжка* |
|  |  |  |  |
|  | *Рис.17 – Окно добавления нового контакта* | *Рис.18 – Окно добавления новой новости* |  |

## **Серверная часть**

### **Проектирование базы данных**

Любое клиент-серверное приложение должно иметь базу данных для хранения, обработки и передачи информации.

Для квалификационной работы была разработана БД, содержащая 2 (две) основные и 4 (четыре) вспомогательные таблицы.

Основные таблицы:

* users – таблица зарегистрированных пользователей:
  + id – идентификатор, уникальный ключ;
  + email – электронная почта;
  + password – пароль;
  + fio – Ф.И.О.;
  + phone – контактный телефон;
  + access – id текущего уровня доступа из одноименной таблицы;
  + class\_id – id класса, к которому имеет отношение (пользователь), из таблицы class;
  + last\_login – дата и время последнего входа в приложение;
* news – таблица, содержащая все новости учебного заведения:
  + id – идентификатор, уникальный ключ;
  + news\_head – заголовок новости;
  + news\_text – текст новости;
  + publishing\_time – дата добавления / изменения новости;
  + category – idкатегории для текущей новости из таблицы categories;
  + who\_add – id пользователя из таблицы users, который добавил / изменил новость;
  + important – булевый идентификатор того является ли новость важной (1 – да, 0 – нет).

Вспомогательные таблицы:

* categories – таблица, содержащая перечень категорий:
  + id – идентификатор, уникальный ключ;
  + name – имя категории;
* class – таблица, содержащая список всех классов в школе:
  + id – идентификатор, уникальный ключ;
  + name – имя класса;
* access – таблица, содержащая перечень всех уровней доступа для приложения:
  + id – идентификатор, уникальный ключ;
  + name – название уровня доступа;
* users\_temp – таблица, содержащая список пользователей, ожидающих подтверждения регистрации:
  + id – идентификатор, уникальный ключ;
  + guid – id пользователя из таблицы users;
  + confirmation\_code – уникальный символьный ключ (случайно сгенерированная MD5 строка) для подтверждения регистрации.

На данный момент существуют следующие уровни доступа:

|  |  |  |
| --- | --- | --- |
| **Уровень доступа** | **Краткое описание** | **Действия** |
| 0 | Почта не проверена | не может авторизоваться пока не завершит регистрацию перейдя по ссылке в присланном на почту письме |
| 1 | подтвержденный пользователь | может авторизоваться в приложении и просматривать общие новости |
| 2 | родитель | может просматривать не только общие новости, но и новости класса к которому имеет отношение |
| 3 | глава родительского комитета | может создавать новости, а также изменять и удалять только те новости, которые создал; может просматривать краткую информацию об авторах новостей |
| 4 | учитель | может просматривать все новости учебного заведения |
| 5 | классный руководитель | может удалять и изменять чужие новости |
| 6 | администрация | - |

*Табл.2 – Уровни доступа в квалификационном приложении*

Для обработки данных на сервере приложения использовались PHP модули:

* register.php – модуль регистрации новых пользователей и подтверждения регистрации для существующих;
* news.php – модуль получения / изменения / удаления / добавления новостей;
* categories.php – модуль получения всех существующих категорий для добавления / изменения новости в мобильном приложении;
* login.php – модуль авторизации пользователей;
* users.php – модуль получения данных об авторе новости.

Данная структура позволяет без труда сопровождать и поддерживать эксплуатацию запросов на сервер не только первоначальным разработчикам, но и сторонними разработчиками.

Клиентское приложение

Сервер

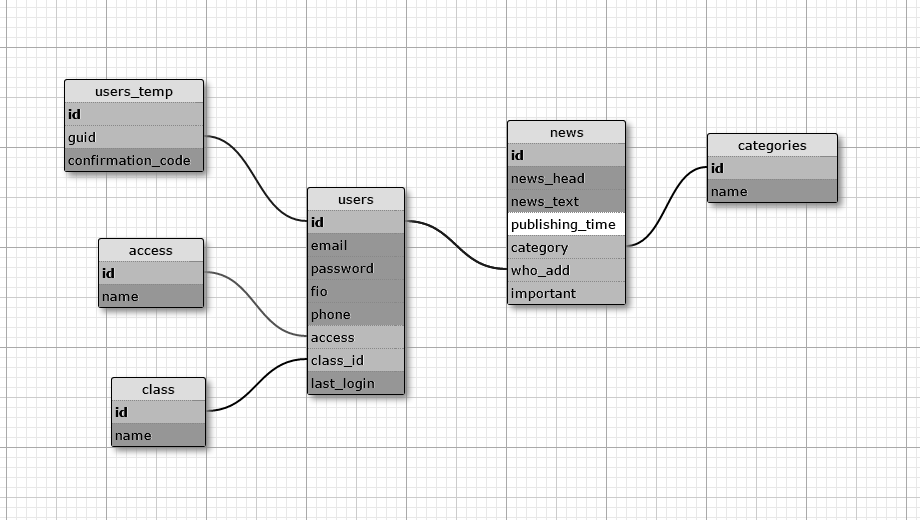
База данных

Интернет

*Схема.2 – Общая схема работы клиент-серверных систем*

На схеме 2 изображена общая пошаговая схема работы клиент-серверных приложений:

1. Отправка через интернет запроса на сервер;
2. Выбор данных из базы данных согласно полученному запросу;
3. Получение и обработка данных по принятому запросу;
4. Отправка ответа через интернет.



*Рис.19 – Структура базы данных приложения*

# **ЗАКЛЮЧЕНИЕ**

Квалификационная работа направлена на создание двусторонней связи между администрацией школы, учителями, классными руководителями и родителями. Полученные в процессе обучения знания были применены в реализации обозначенной проблемы.

В процессе реализации поставленных задач требовалось:

* создание приложения с дружественным пользовательским интерфейсом;
* поддержка устаревшей версии системы Android;
* организация стабильной работы всех функций программы, связанных непосредственно с функциями управления данным и с сервисными функциями приложения.

Выполнение работы основывалось на следующих программных инструментах: Java, Android Studio, PHP, SQL, SQLite. Возможности современного программирования огромны. С помощью программирования можно решать задачи различного характера и сложности.

Написанное приложение может быть с лёгкостью использовано в любом учебном заведении.

Считаю, что цель, поставленная руководителем, была достигнута.

# **СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ**

1. Блог на «хабре» о разработке под Android[Электронный ресурс]

// URL: <http://habrahabr.ru/blogs/android_development/>

2. Официальная справка для Android разработчиков [Электронный ресурс]

// URL <http://developer.android.com/index.html>

3. Программирование для Android. Самоучитель /

Колисниченко Д. – СПб.: Санкт-Петербург, 2011. – 736 с.

4. Android 2. Программирование приложений для планшетных компьютеров и смартфонов / Рето Майер. – СПб.: Санкт-Петербург, 2011. – 672 с.

5. Статьи о программировании для Android[Электронный ресурс]

// URL: <http://flashbot.ru/android-dev>

6. Официальная справка по среде программирования [Электронный ресурс]

// URL: [http://www.jetbrains.com](http://www.jetbrains.com/)

7. Форум о программировании для Android[Электронный ресурс]

// URL: <http://www.cyberforum.ru/android-dev/>

8. Форум о программировании для мобильных устройств [Электронный ресурс]

// URL: <http://www.4pda.ru>

9. Программирование под Android / Блэйк Мик. – СПб.: Санкт-Петербург, 2012. – 496 с.

10. Смартфоны Android без напряга. Руководство пользователя / Андрей Жвалевский. – СПб.: Санкт-Петербург, 2012. – 224 с.

11. Программирование для Android. Самоучитель / Денис Колиснеченко. – СПб.: Санкт-Петербург, 2011. – 272 с.

12. Программирование под Android. Для профессионалов / Б. Харди, Б. Филлипс. – СПб.: Питер, 2014. – 592 с.

13. WWWSQLDESIGNER – Визуальный конструктор таблиц [Электронный ресурс]

// URL: <http://ondras.zarovi.cz/sql/demo/>

14. SQLTutorial – Справочник по языку SQL[Электронный ресурс]

// URL: <http://www.w3schools.com/sql/>

15. Профессионал PHP – Видеокурс по языку PHP

// URL: <http://prog-school.ru/products/phpro/>

16. StackOverflow – Сайт вопросов и ответов для профессиональных программистов и энтузиастов [Электронный ресурс]

// URL: <http://stackoverflow.com>

17. GitHub – Сервис контроля версий кода [Электронный ресурс]

// URL: <https://github.com>

18. Библиотека программиста – страничка с набором литературы и видеокурсов по множеству языков программирования [Электронный ресурс]

// URL: <http://theasder.github.io>

19. ITMozg – страничка в соц.сети с набором литературы и видеокурсов по множеству языков программирования [Электронный ресурс]

// URL: <https://vk.com/itmozg>

20. FreeWebHostingarea – Бесплатный Веб хостинг [Электронный ресурс]

// URL: <http://ueuo.com>

21. Исходные коды приложения [Электронный ресурс]

// URL: <https://github.com/Gagauz2010/DigitalSchool>

# **ПРИЛОЖЕНИЕ (ЛИСТИНГ ПРОГРАММЫ)**

**Окно авторизации*(LogIn.java)***

package ru.spsu.fmf.digitalschool;

import android.app.Activity;

import android.app.AlertDialog;

import android.content.Context;

import android.content.DialogInterface;

import android.content.Intent;

import android.content.SharedPreferences;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.graphics.Shader;

import android.graphics.drawable.BitmapDrawable;

import android.graphics.drawable.Drawable;

import android.net.ConnectivityManager;

import android.net.NetworkInfo;

import android.preference.PreferenceManager;

import android.support.v7.app.ActionBarActivity;

import android.os.Bundle;

import android.text.Html;

import android.text.method.LinkMovementMethod;

import android.util.DisplayMetrics;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.Toast;

import org.json.JSONException;

import org.json.JSONObject;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

import ru.spsu.fmf.digitalschool.php.$login;

@SuppressWarnings("deprecation")

public class LogIn extends ActionBarActivity {

private EditText emailTxt, passwordTxt;

public static Activity loginActivity;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_log\_in);

getSupportActionBar().hide();

View view = this.findViewById(R.id.logMainView).getRootView();

Drawable d = getRepeatingBG(this, R.drawable.main\_bg);

view.setBackgroundDrawable(d);

loginActivity = this;

TextView site = (TextView) findViewById(R.id.textViewSite);

TextView forgetPassword = (TextView) findViewById(R.id.textViewPassword);

String htmlSite = getString(R.string.site);

String htmlForget = getString(R.string.forget\_password);

site.setText(Html.fromHtml(htmlSite));

site.setMovementMethod(LinkMovementMethod.getInstance());

forgetPassword.setText(Html.fromHtml(htmlForget));

forgetPassword.setMovementMethod(LinkMovementMethod.getInstance());

final Button loginBtn = (Button) findViewById(R.id.logBtnLogin);

final Button registerBtn = (Button) findViewById(R.id.logBtnRegister);

emailTxt = (EditText) findViewById(R.id.logEditEmail);

passwordTxt = (EditText) findViewById(R.id.logEditPassword);

SharedPreferences sp = PreferenceManager.getDefaultSharedPreferences(getApplicationContext());

Boolean remember = sp.getBoolean("remember\_user", false);

String address = sp.getString("address", "");

String password = sp.getString("password", "");

if (remember) {

if (!isNetworkOnline()) {

Toast.makeText(getApplicationContext(), getText(R.string.no\_internet\_message), Toast.LENGTH\_LONG).show();

} else {

emailTxt.setText(address);

passwordTxt.setText(password);

auth(String.valueOf(emailTxt.getText()), passEncrypt(String.valueOf(passwordTxt.getText())));

}

}

loginBtn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if (String.valueOf(emailTxt.getText()).trim().length() == 0 || String.valueOf(passwordTxt.getText()).trim().length() == 0) {

Toast.makeText(getApplicationContext(), R.string.fill\_all\_fields, Toast.LENGTH\_SHORT).show();

return;

}

String email = String.valueOf(emailTxt.getText());

String password = passEncrypt(String.valueOf(passwordTxt.getText()));

if (!isNetworkOnline())

Toast.makeText(getApplicationContext(), getText(R.string.no\_internet\_message), Toast.LENGTH\_SHORT).show();

else {

auth(email, password);

}

}

});

registerBtn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(getApplicationContext(), Registration.class);

startActivity(intent);

}

});

}

private void auth(String email, String password) {

try {

tryToAuth(new $login(this).execute(email, password).get());

}

catch (Exception ex) {

Log.e("LogIn", ex.getMessage());

}

}

private void tryToAuth(String result) {

if (!result.equals("User not found")) {

try {

JSONObject object = new JSONObject(result);

String cl, access, id;

access = object.getString("access").trim();

id = object.getString("id").trim();

cl = object.getString("class").trim();

if (access.contains("0")) {

forgotUser();

DialogInterface.OnClickListener okDialogClickListener = new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

switch (which) {

case DialogInterface.BUTTON\_POSITIVE: // Yes button clicked

break;

case DialogInterface.BUTTON\_NEGATIVE: // No button clicked // do nothing

break;

case DialogInterface.BUTTON\_NEUTRAL: // Ok button clicked // do nothing

break;

}

}

};

AlertDialog.Builder builder = new AlertDialog.Builder(this);

builder.setTitle(getString(R.string.auth))

.setMessage(getString(R.string.contact\_need\_to\_confirm))

.setNeutralButton(R.string.btn\_ok, okDialogClickListener).show();

} else {

rememberUser(id, access, cl);

}

}

catch (JSONException e){

Log.e("login", e.getMessage());

}

}

else if (result.contains("User not found")){

forgotUser();

Toast.makeText(getApplicationContext(), getText(R.string.post\_auth\_wrong\_data), Toast.LENGTH\_SHORT).show();

}

else if (result.contains("Exception: Unable to resolve host")) {

Toast.makeText(getApplicationContext(), getText(R.string.no\_internet\_message), Toast.LENGTH\_LONG).show();

}

else{

forgotUser();

Toast.makeText(getApplicationContext(), result, Toast.LENGTH\_SHORT).show();

}

}

private void rememberUser(String id, String access, String cl) {

SharedPreferences.Editor prefs = PreferenceManager.getDefaultSharedPreferences(getApplicationContext()).edit();

prefs.putBoolean("remember\_user", true);

prefs.putString("address", String.valueOf(emailTxt.getText()));

prefs.putString("password", String.valueOf(passwordTxt.getText()));

prefs.putString("level", access);

prefs.putString("id", id);

prefs.putString("class", cl);

prefs.apply();

Intent intent = new Intent(LogIn.this, Main.class);

startActivity(intent);

}

private void forgotUser() {

SharedPreferences.Editor prefs = PreferenceManager.getDefaultSharedPreferences(getApplicationContext()).edit();

prefs.putBoolean("remember\_user", false);

prefs.putString("address", null);

prefs.putString("password", null);

prefs.putString("level", null);

prefs.putString("id", null);

prefs.putString("class", null);

prefs.apply();

}

private Drawable getRepeatingBG(Activity activity, int center)

{

DisplayMetrics dm = new DisplayMetrics();

activity.getWindowManager().getDefaultDisplay().getMetrics(dm);

BitmapFactory.Options options = new BitmapFactory.Options();

options.inScaled = true;

Bitmap center\_bmp = BitmapFactory.decodeResource(activity.getResources(), center, options);

center\_bmp.setDensity(Bitmap.DENSITY\_NONE);

center\_bmp=Bitmap.createScaledBitmap(center\_bmp, center\_bmp.getWidth()/(center\_bmp.getHeight()/dm.heightPixels), dm.heightPixels, true);

BitmapDrawable center\_drawable = new BitmapDrawable(activity.getResources(),center\_bmp);

center\_drawable.setTileModeX(Shader.TileMode.REPEAT);

return center\_drawable;

}

private String passEncrypt(String password) {

try {

MessageDigest sha256 = MessageDigest.getInstance("SHA-256");

String salt = "50me\_r@nd0m\_s41t";

String passWithSalt;

passWithSalt = password + salt;

byte[] passBytes = passWithSalt.getBytes();

byte[] passHash = sha256.digest(passBytes);

StringBuilder sb = new StringBuilder();

for (byte aPassHash : passHash) {

sb.append(Integer.toString((aPassHash & 0xff) + 0x100, 16).substring(1));

}

return sb.toString();

} catch (NoSuchAlgorithmException e) { e.printStackTrace(); }

return null;

}

public boolean isNetworkOnline() {

boolean status=false;

try{

ConnectivityManager cm = (ConnectivityManager) getSystemService(Context.CONNECTIVITY\_SERVICE);

NetworkInfo netInfo = cm.getNetworkInfo(0);

if (netInfo != null && netInfo.getState() == NetworkInfo.State.CONNECTED) {

status= true;

}else {

netInfo = cm.getNetworkInfo(1);

if(netInfo!=null && netInfo.getState() == NetworkInfo.State.CONNECTED)

status= true;

}

}catch(Exception e){

e.printStackTrace();

return false;

}

return status;

}

}

**Окнорегистрации*(Registration.java)***

package ru.spsu.fmf.digitalschool;

import android.app.Activity;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.graphics.Shader;

import android.graphics.drawable.BitmapDrawable;

import android.graphics.drawable.Drawable;

import android.os.Bundle;

import android.support.v7.app.ActionBarActivity;

import android.util.DisplayMetrics;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

import ru.spsu.fmf.digitalschool.php.$registration;

@SuppressWarnings("deprecation")

public class Registration extends ActionBarActivity {

private EditText email, email1, pas, pas1, fio, phone;

public static Activity regActivity;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_registration);

View view = this.findViewById(R.id.regMainView).getRootView();

Drawable d = getRepeatingBG(this, R.drawable.main\_bg\_gauss);

view.setBackgroundDrawable(d);

regActivity = this;

email = (EditText) findViewById(R.id.regEmail);

email1 = (EditText) findViewById(R.id.regConfEmail);

pas = (EditText) findViewById(R.id.regPas);

pas1 = (EditText) findViewById(R.id.regConfPas);

fio = (EditText) findViewById(R.id.regFIO);

phone = (EditText) findViewById(R.id.regPhone);

Button reg = (Button) findViewById(R.id.regRegBtn);

reg.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if (String.valueOf(email.getText()).trim().length() == 0 ||

String.valueOf(email1.getText()).trim().length() == 0 ||

String.valueOf(pas.getText()).trim().length() == 0 ||

String.valueOf(pas1.getText()).trim().length() == 0 ||

String.valueOf(fio.getText()).trim().length() == 0 ||

String.valueOf(phone.getText()).trim().length() == 0) {

Toast.makeText(getApplicationContext(), R.string.fill\_all\_fields, Toast.LENGTH\_SHORT).show();

return;

}

if (!String.valueOf(email.getText()).equals(String.valueOf(email1.getText()))) {

Toast.makeText(getApplicationContext(), R.string.different\_emails, Toast.LENGTH\_SHORT).show();

return;

}

if (!String.valueOf(pas.getText()).equals(String.valueOf(pas1.getText()))) {

Toast.makeText(getApplicationContext(), R.string.different\_passwords, Toast.LENGTH\_SHORT).show();

return;

}

if (String.valueOf(pas.getText()).trim().length() < 8) {

Toast.makeText(getApplicationContext(), R.string.password\_not\_strong, Toast.LENGTH\_SHORT).show();

return;

}

String emailText, passwordText, fioText, phoneText;

emailText = String.valueOf(email.getText());

passwordText = String.valueOf(pas.getText());

fioText = String.valueOf(fio.getText());

phoneText = String.valueOf(phone.getText());

new $registration(getApplicationContext()).execute(emailText, passEncrypt(passwordText), fioText, phoneText);

}

});

}

private String passEncrypt(String password) {

try {

MessageDigest sha256 = MessageDigest.getInstance("SHA-256");

String salt = "50me\_r@nd0m\_s41t";

String passWithSalt = password + salt;

byte[] passBytes = passWithSalt.getBytes();

byte[] passHash = sha256.digest(passBytes);

StringBuilder sb = new StringBuilder();

for (byte aPassHash : passHash) {

sb.append(Integer.toString((aPassHash & 0xff) + 0x100, 16).substring(1));

}

return sb.toString();

} catch (NoSuchAlgorithmException e) { e.printStackTrace(); }

return null;

}

private Drawable getRepeatingBG(Activity activity, int center)

{

DisplayMetrics dm = new DisplayMetrics();

activity.getWindowManager().getDefaultDisplay().getMetrics(dm);

BitmapFactory.Options options = new BitmapFactory.Options();

options.inScaled=true;

Bitmap center\_bmp = BitmapFactory.decodeResource(activity.getResources(), center, options);

center\_bmp.setDensity(Bitmap.DENSITY\_NONE);

center\_bmp=Bitmap.createScaledBitmap(center\_bmp, center\_bmp.getWidth()/(center\_bmp.getHeight()/dm.heightPixels), dm.heightPixels, true);

BitmapDrawable center\_drawable = new BitmapDrawable(activity.getResources(),center\_bmp);

center\_drawable.setTileModeX(Shader.TileMode.REPEAT);

return center\_drawable;

}

}

**Главноеокно*(Main.java)***

package ru.spsu.fmf.digitalschool;

import android.app.Activity;

import android.app.AlertDialog;

import android.content.ActivityNotFoundException;

import android.content.Context;

import android.content.DialogInterface;

import android.content.Intent;

import android.content.SharedPreferences;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.graphics.Shader;

import android.graphics.drawable.BitmapDrawable;

import android.graphics.drawable.Drawable;

import android.net.Uri;

import android.os.AsyncTask;

import android.preference.PreferenceManager;

import android.support.v4.view.MenuItemCompat;

import android.support.v7.app.ActionBarActivity;

import android.os.Bundle;

import android.util.DisplayMetrics;

import android.util.Log;

import android.view.ContextMenu;

import android.view.LayoutInflater;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.view.ViewGroup;

import android.view.animation.Animation;

import android.view.animation.AnimationUtils;

import android.widget.AdapterView;

import android.widget.ArrayAdapter;

import android.widget.CheckBox;

import android.widget.EditText;

import android.widget.ImageView;

import android.widget.ListView;

import android.widget.ProgressBar;

import android.widget.Spinner;

import android.widget.TabHost;

import android.widget.TextView;

import org.json.JSONArray;

import org.json.JSONException;

import org.json.JSONObject;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.OutputStreamWriter;

import java.net.URL;

import java.net.URLConnection;

import java.net.URLEncoder;

import java.util.ArrayList;

import java.util.List;

import java.util.concurrent.ExecutionException;

import ru.spsu.fmf.digitalschool.adapters.newsAdapter;

import ru.spsu.fmf.digitalschool.php.$categories;

import ru.spsu.fmf.digitalschool.php.$news;

import ru.spsu.fmf.digitalschool.php.$who;

import ru.spsu.fmf.digitalschool.sqLite.ContactDbHandler;

import ru.spsu.fmf.digitalschool.structures.Contact;

import ru.spsu.fmf.digitalschool.structures.jsonCategories;

import ru.spsu.fmf.digitalschool.structures.jsonNews;

public class Main extends ActionBarActivity {

int longClickedItemIndex;

private static final int EDIT = 0, CALL = 1, SMS = 2, EMAIL = 3, MAP = 4, DELETE = 5;

private static final int CONFIRM\_LOGOFF = 0, CONFIRM\_DELETE\_CONTACT = 1, CONFIRM\_DELETE\_NEWS = 2;

private static final String TAG = "MAIN ACTIVITY";

List<Contact> Contacts = new ArrayList<>();

ContactDbHandler dbHandler;

ListView contactListView;

ArrayAdapter<Contact> contactAdapter;

private ArrayList<jsonNews> feedList = null;

private ArrayList<jsonCategories> categoriesList = null;

private ProgressBar progressbar = null;

private ListView feedListView = null;

private TextView errorMessage = null;

private static final int WHO = 6, DELETE\_NEWS = 7, EDIT\_NEWS = 8, NEWS\_INFO\_DEVELOPER = 9;

TabHost tabHostMain;

SharedPreferences sp;

String userAccess, userId, userClass;

Menu myMenu;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

if (LogIn.loginActivity != null)

LogIn.loginActivity.finish();

Drawable dAll = getRepeatingBG(this, R.drawable.main\_bg\_gauss);

View view1 = this.findViewById(R.id.tabNews).getRootView();

view1.setBackgroundDrawable(dAll);

View view2 = this.findViewById(R.id.tabContacts).getRootView();

view2.setBackgroundDrawable(dAll);

tabHostMain = (TabHost) findViewById(R.id.tabHostMain);

tabHostMain.setup();

tabHostMain.setOnTabChangedListener(new TabHost.OnTabChangeListener() {

@Override

public void onTabChanged(String tag) {

switch (tag) {

case "news":

getSupportActionBar().setTitle(getString(R.string.tab1));

break;

case "contacts":

getSupportActionBar().setTitle(getString(R.string.tab2));

break;

default:

break;

}

supportInvalidateOptionsMenu();

}

});

this.setNewTab(tabHostMain, "news", R.string.empty, R.drawable.ic\_action\_chat, R.id.tabNews);

this.setNewTab(tabHostMain, "contacts", R.string.empty, R.drawable.ic\_action\_cc\_bcc, R.id.tabContacts);

sp = PreferenceManager.getDefaultSharedPreferences(getApplicationContext());

userAccess = sp.getString("level", "0");

userId = sp.getString("id", "-1");

userClass = sp.getString("class", "0");

contactListView = (ListView) findViewById(R.id.contactListView);

dbHandler = new ContactDbHandler(getApplicationContext());

if (dbHandler.getContactsCount() != 0)

Contacts.addAll(dbHandler.getAllContacts());

populateList();

registerForContextMenu(contactListView);

contactListView.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {

@Override

public boolean onItemLongClick(AdapterView<?> parent, View view, int position, long id) {

longClickedItemIndex = position;

return false;

}

});

progressbar = (ProgressBar) findViewById(R.id.progressBar);

errorMessage = (TextView) findViewById(R.id.news\_error\_message);

new DownloadFilesTask().execute();

}

private void populateList(){

contactAdapter = new ContactListAdapter();

contactListView.setAdapter(contactAdapter);

}

private class ContactListAdapter extends ArrayAdapter<Contact>{

public ContactListAdapter(){

super(Main.this, R.layout.contact\_listview\_item, Contacts);

}

@Override

public View getView(int position, View view, ViewGroup parent){

if (view == null)

view = getLayoutInflater().inflate(R.layout.contact\_listview\_item, parent, false);

Contact currentContact = Contacts.get(position);

TextView name = (TextView) view.findViewById(R.id.contactName);

name.setText(currentContact.getName());

TextView phone = (TextView) view.findViewById(R.id.phoneNumber);

phone.setText(currentContact.getPhone());

TextView email = (TextView) view.findViewById(R.id.emailAddress);

email.setText(currentContact.getEmail());

TextView address = (TextView) view.findViewById(R.id.cAddress);

address.setText(currentContact.getAddress());

ImageView image = (ImageView) view.findViewById(R.id.ivContactImage);

image.setImageURI(currentContact.getImageURI());

return view;

}

}

public void onCreateContextMenu(ContextMenu menu, View view, ContextMenu.ContextMenuInfo menuInfo){

super.onCreateContextMenu(menu, view, menuInfo);

menu.setHeaderTitle(getText(R.string.menu\_header\_opt));

switch (view.getId()) {

case R.id.contactListView:

//menu.setHeaderIcon(R.drawable.edit\_icon);

menu.add(Menu.NONE, EDIT, Menu.NONE, R.string.menu\_edit);

menu.add(Menu.NONE, CALL, Menu.NONE, R.string.menu\_call);

menu.add(Menu.NONE, SMS, Menu.NONE, R.string.menu\_sms);

menu.add(Menu.NONE, EMAIL, Menu.NONE, R.string.menu\_email);

menu.add(Menu.NONE, MAP, Menu.NONE, R.string.menu\_map);

menu.add(Menu.NONE, DELETE, Menu.NONE, R.string.menu\_delete);

break;

case R.id.newsListView:

if (Integer.valueOf(userAccess) >= 3){

menu.add(Menu.NONE, WHO, Menu.NONE, getText(R.string.menu\_who));

if (Integer.valueOf(userAccess) >= 5 || feedList.get(longClickedItemIndex).getWhoAdd() == Integer.valueOf(userId)) {

menu.add(Menu.NONE, DELETE\_NEWS, Menu.NONE, getText(R.string.menu\_delete));

menu.add(Menu.NONE, EDIT\_NEWS, Menu.NONE, getText(R.string.menu\_edit));

}

}

break;

}

}

public boolean onContextItemSelected(MenuItem item){

switch (item.getItemId()){

case EDIT:

Intent intent = new Intent(this, ContactEditor.class);

intent.putExtra("id", Contacts.get(longClickedItemIndex).getID());

startActivity(intent);

break;

case CALL:

try {

Intent callIntent = new Intent(Intent.ACTION\_CALL);

callIntent.setData(Uri.parse("tel:" + Contacts.get(longClickedItemIndex).getPhone()));

startActivity(Intent.createChooser(callIntent, getString(R.string.menu\_call)));

} catch (ActivityNotFoundException activityException) {

Log.e("Вызов", "Звонокнеудался", activityException);

}

break;

case SMS:

try {

Intent smsIntent = new Intent(Intent.ACTION\_SENDTO);

smsIntent.setType("vnd.android-dir/mms-sms");

smsIntent.setData(Uri.parse("sms:" + Contacts.get(longClickedItemIndex).getPhone()));

startActivity(Intent.createChooser(smsIntent, getString(R.string.menu\_sms)));

}catch (ActivityNotFoundException activityException) {

Log.e("Отправка SMS", "Отправканеудалась", activityException);

}

break;

case EMAIL:

try{

Intent emailIntent = new Intent(android.content.Intent.ACTION\_SEND); /\*, Uri.fromParts("mailto",Contacts.get(longClickedItemIndex).getEmail(), null)\*/

emailIntent.putExtra(android.content.Intent.EXTRA\_EMAIL, Contacts.get(longClickedItemIndex).getEmail());

emailIntent.setType("plain/text");

startActivity(Intent.createChooser(emailIntent, getString(R.string.menu\_email)));

}catch (ActivityNotFoundException activityException) {

Log.e("Отправкапочты", "Отправканеудалась", activityException);

}

break;

case MAP:

Intent mapIntent = new Intent(android.content.Intent.ACTION\_VIEW, Uri.parse("geo:0,0?q=" + Contacts.get(longClickedItemIndex).getAddress()));

startActivity(Intent.createChooser(mapIntent, getString(R.string.menu\_map)));

break;

case DELETE:

alertMessageTwoButtons(getString(R.string.dialog\_del\_header), getString(R.string.dialog\_del\_contact\_message), CONFIRM\_DELETE\_CONTACT);

break;

case WHO:

try {

showAuthorInfo(new $who(this).execute(String.valueOf(feedList.get(longClickedItemIndex).getWhoAdd())).get());

} catch (InterruptedException | JSONException | ExecutionException e) {

Log.e("Main", e.getMessage());

}

break;

case DELETE\_NEWS:

alertMessageTwoButtons(getString(R.string.dialog\_del\_header), getString(R.string.dialog\_del\_news\_message), CONFIRM\_DELETE\_NEWS);

break;

case EDIT\_NEWS:

String[] categoriesStingList;

AlertDialog.Builder builder = new AlertDialog.Builder(this);

LayoutInflater inflater = this.getLayoutInflater();

View dialogView = inflater.inflate(R.layout.dialog\_add\_news, null);

final Spinner spinner = (Spinner) dialogView.findViewById(R.id.addNewsSpinner);

final EditText newsHeader = (EditText) dialogView.findViewById(R.id.addNewsHeader);

final EditText newsBody = (EditText) dialogView.findViewById(R.id.addNewsBody);

final CheckBox isImportant = (CheckBox) dialogView.findViewById(R.id.addNewsImportant);

newsHeader.setText(feedList.get(longClickedItemIndex).getNewsHead());

newsBody.setText(feedList.get(longClickedItemIndex).getNewsText());

isImportant.setChecked(feedList.get(longClickedItemIndex).getImportant() == 1);

int category\_id = feedList.get(longClickedItemIndex).getNewsCategory();

int position = 0;

try {

String result = new $categories(this).execute().get();

JSONObject json = new JSONObject(result);

JSONArray categories = json.getJSONArray("categories");

categoriesList = new ArrayList<>();

categoriesStingList = new String[categories.length()];

for (int i = 0; i < categories.length(); i++) {

JSONObject category = categories.getJSONObject(i);

jsonCategories categoryItem = new jsonCategories();

categoryItem.setId(category.getInt("id"));

categoryItem.setName(category.getString("name"));

categoriesList.add(categoryItem);

categoriesStingList[i] = category.getString("name");

if (categoryItem.getId() == category\_id)

position = i;

}

ArrayAdapter<String> adapter = new ArrayAdapter<>(this, android.R.layout.simple\_spinner\_item, categoriesStingList);

adapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);

spinner.setAdapter(adapter);

spinner.setSelection(position);

} catch (InterruptedException | ExecutionException | JSONException e) {

e.printStackTrace();

}

builder.setView(dialogView)

.setTitle(getString(R.string.dialog\_edit\_news\_header))

.setPositiveButton(R.string.action\_edit\_news, new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int id) {

new $news(Main.this).execute("update", String.valueOf(newsHeader.getText()),

String.valueOf(newsBody.getText()),

String.valueOf(categoriesList.get(spinner.getSelectedItemPosition()).getId()),

userId,

isImportant.isChecked() ? "1":"0",

String.valueOf(feedList.get(longClickedItemIndex).getNewsId())

);

updateListOnRefresh(myMenu.findItem(R.id.news\_action\_refresh));

}

})

.setNegativeButton(R.string.action\_cancel, new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog, int id) {

//nothing...

}

});

builder.show();

break;

}

return super.onContextItemSelected(item);

}

private void showAuthorInfo(String result) throws JSONException {

JSONObject object = new JSONObject(result);

AlertDialog.Builder builder = new AlertDialog.Builder(this);

LayoutInflater inflater = this.getLayoutInflater();

View dialogView = inflater.inflate(R.layout.dialog\_author\_info, null);

TextView fio = (TextView) dialogView.findViewById(R.id.dialog\_author\_fio);

fio.setText(object.getString("fio").trim());

TextView phone = (TextView) dialogView.findViewById(R.id.dialog\_author\_phone);

phone.setText(object.getString("phone").trim());

TextView cl = (TextView) dialogView.findViewById(R.id.dialog\_author\_class);

cl.setText(object.getString("class").trim());

TextView access = (TextView) dialogView.findViewById(R.id.dialog\_author\_access);

access.setText(object.getString("access").trim());

TextView date = (TextView) dialogView.findViewById(R.id.dialog\_author\_date);

date.setText(object.getString("last\_login").trim());

builder.setView(dialogView)

.setTitle(getString(R.string.menu\_who))

.setNeutralButton(R.string.btn\_ok, new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int id) {

//nothing

}

});

builder.show();

}

public void updateList() {

feedListView = (ListView) findViewById(R.id.newsListView);

feedListView.setVisibility(View.VISIBLE);

progressbar.setVisibility(View.GONE);

errorMessage.setVisibility(View.GONE);

feedListView.setAdapter(new newsAdapter(this, feedList));

registerForContextMenu(feedListView);

feedListView.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {

@Override

public boolean onItemLongClick(AdapterView<?> parent, View view, int position, long id) {

longClickedItemIndex = position;

return false;

}

});

}

private void updateListOnError(String result) {

errorMessage.setText(result);

errorMessage.setVisibility(View.VISIBLE);

progressbar.setVisibility(View.GONE);

feedListView.setVisibility(View.GONE);

}

private void updateListOnRefresh(MenuItem item) {

errorMessage.setText(null);

feedListView.setVisibility(View.GONE);

errorMessage.setVisibility(View.GONE);

progressbar.setVisibility(View.VISIBLE);

feedList = null;

LayoutInflater inflater = (LayoutInflater)getSystemService(Context.LAYOUT\_INFLATER\_SERVICE);

ImageView iv = (ImageView)inflater.inflate(R.layout.refresh, null);

Animation rotation = AnimationUtils.loadAnimation(this, R.anim.refresh\_rotate);

rotation.setRepeatCount(Animation.INFINITE);

iv.startAnimation(rotation);

MenuItemCompat.setActionView(item, iv);

new DownloadFilesTask().execute();

}

private class DownloadFilesTask extends AsyncTask<String, Integer, String> {

@Override

protected void onProgressUpdate(Integer... values) {

}

@Override

protected String doInBackground(String... params) {

try {

JSONObject json = getJSONFromUrl();

parseJson(json);

return null;

} catch (Exception ex){

return ex.getMessage();

}

}

@Override

protected void onPostExecute(String result) {

if (result == null)

if (feedList != null)

updateList();

else

updateListOnError("Не удалось получить список новостей\nПопробуйте обновить позже");

else

updateListOnError(result);

resetUpdating();

}

}

public JSONObject getJSONFromUrl() {

JSONObject jObj = null;

String json = null;

try {

String data = URLEncoder.encode("access", "UTF-8")

+ "=" + URLEncoder.encode(userAccess, "UTF-8");

data += "&" + URLEncoder.encode("class", "UTF-8")

+ "=" + URLEncoder.encode(userClass, "UTF-8");

data += "&" + URLEncoder.encode("user\_id", "UTF-8")

+ "=" + URLEncoder.encode(userId, "UTF-8");

String newsUrl = "http://dsc.freevar.com/news.php";

URL url = new URL(newsUrl);

URLConnection conn = url.openConnection();

conn.setDoOutput(true);

OutputStreamWriter wr = new OutputStreamWriter

(conn.getOutputStream());

wr.write(data);

wr.flush();

BufferedReader reader = new BufferedReader(new InputStreamReader(conn.getInputStream(), "UTF-8"), 8);

StringBuilder sb = new StringBuilder();

String line;

while ((line = reader.readLine()) != null) {

sb.append(line).append("\n");

}

json = sb.toString();

} catch (IOException e) {

Log.e("Main", e.getMessage());

}

try {

jObj = new JSONObject(json);

} catch (JSONException e) {

Log.e("JSON Parser", "Error parsing data " + e.toString());

}

return jObj;

}

public void parseJson(JSONObject json) {

try {

JSONArray posts = json.getJSONArray("news");

feedList = new ArrayList<>();

for (int i = posts.length()-1; i >=0 ; i--) {

JSONObject post = posts.getJSONObject(i);

jsonNews item = new jsonNews();

item.setNewsId(post.getInt("id"));

item.setNewsHead(post.getString("news\_head"));

item.setNewsText(post.getString("news\_text"));

item.setPublTime(post.getString("publishing\_time"));

item.setNewsCategory(post.getInt("category"));

item.setWhoAdd(post.getInt("who\_add"));

item.setImportant(post.getInt("important"));

feedList.add(item);

}

} catch (JSONException e) {

e.printStackTrace();

}

}

private Drawable getRepeatingBG(Activity activity, int center)

{

DisplayMetrics dm = new DisplayMetrics();

activity.getWindowManager().getDefaultDisplay().getMetrics(dm);

BitmapFactory.Options options = new BitmapFactory.Options();

options.inScaled=true;

Bitmap center\_bmp = BitmapFactory.decodeResource(activity.getResources(), center, options);

center\_bmp.setDensity(Bitmap.DENSITY\_NONE);

center\_bmp=Bitmap.createScaledBitmap(center\_bmp, center\_bmp.getWidth()/(center\_bmp.getHeight()/dm.heightPixels), dm.heightPixels, true);

BitmapDrawable center\_drawable = new BitmapDrawable(activity.getResources(),center\_bmp);

center\_drawable.setTileModeX(Shader.TileMode.REPEAT);

return center\_drawable;

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

getMenuInflater().inflate(R.menu.menu\_main, menu);

myMenu = menu;

return true;

}

@Override

public boolean onPrepareOptionsMenu(Menu menu) {

switch (tabHostMain.getCurrentTabTag()){

case "news":

//news menus

menu.findItem(R.id.news\_action\_refresh).setVisible(true);

menu.findItem(R.id.news\_action\_write).setVisible(Integer.valueOf(userAccess) > 2);

//contact menus

menu.findItem(R.id.contacts\_action\_add\_person).setVisible(false);

break;

case "contacts":

//news menus

menu.findItem(R.id.news\_action\_write).setVisible(false);

menu.findItem(R.id.news\_action\_refresh).setVisible(false);

//contact menus

menu.findItem(R.id.contacts\_action\_add\_person).setVisible(true);

break;

case "subjects":

//news menus

menu.findItem(R.id.news\_action\_write).setVisible(false);

menu.findItem(R.id.news\_action\_refresh).setVisible(false);

//contact menus

menu.findItem(R.id.contacts\_action\_add\_person).setVisible(false);

break;

default:

break;

}

return super.onPrepareOptionsMenu(menu);

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

int id = item.getItemId();

if (id == R.id.contacts\_action\_add\_person){

Intent intent = new Intent(this, ContactEditor.class);

startActivity(intent);

}

if (id == R.id.action\_logout){

alertMessageTwoButtons(getString(R.string.dialog\_logout\_header), getString(R.string.dialog\_logout\_message), CONFIRM\_LOGOFF);

return true;

}

if (id == R.id.news\_action\_refresh){

updateListOnRefresh(item);

return true;

}

if (id == R.id.news\_action\_write){

String[] categoriesStingList;

AlertDialog.Builder builder = new AlertDialog.Builder(this);

LayoutInflater inflater = this.getLayoutInflater();

View dialogView = inflater.inflate(R.layout.dialog\_add\_news, null);

final Spinner spinner = (Spinner) dialogView.findViewById(R.id.addNewsSpinner);

final EditText newsHeader = (EditText) dialogView.findViewById(R.id.addNewsHeader);

final EditText newsBody = (EditText) dialogView.findViewById(R.id.addNewsBody);

final CheckBox isImportant = (CheckBox) dialogView.findViewById(R.id.addNewsImportant);

try {

String result = new $categories(this).execute().get();

JSONObject json = new JSONObject(result);

JSONArray categories = json.getJSONArray("categories");

categoriesList = new ArrayList<>();

categoriesStingList = new String[categories.length()];

for (int i = 0; i < categories.length(); i++) {

JSONObject category = categories.getJSONObject(i);

jsonCategories categoryItem = new jsonCategories();

categoryItem.setId(category.getInt("id"));

categoryItem.setName(category.getString("name"));

categoriesList.add(categoryItem);

categoriesStingList[i] = category.getString("name");

}

ArrayAdapter<String> adapter = new ArrayAdapter<>(this, android.R.layout.simple\_spinner\_item, categoriesStingList);

adapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);

spinner.setAdapter(adapter);

spinner.setSelection(0);

} catch (InterruptedException | ExecutionException | JSONException e) {

e.printStackTrace();

}

builder.setView(dialogView)

.setTitle(getString(R.string.dialog\_add\_news\_header))

.setPositiveButton(R.string.action\_add\_news, new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int id) {

new $news(Main.this).execute("add", String.valueOf(newsHeader.getText()),

String.valueOf(newsBody.getText()),

String.valueOf(categoriesList.get(spinner.getSelectedItemPosition()).getId()),

userId,

isImportant.isChecked() ? "1":"0"

);

updateListOnRefresh(myMenu.findItem(R.id.news\_action\_refresh));

}

})

.setNegativeButton(R.string.action\_cancel, new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog, int id) {

//nothing...

}

});

builder.show();

return true;

}

return super.onOptionsItemSelected(item);

}

public void resetUpdating()

{

MenuItem m = myMenu.findItem(R.id.news\_action\_refresh);

if(MenuItemCompat.getActionView(m) != null)

{

MenuItemCompat.getActionView(m).clearAnimation();

MenuItemCompat.setActionView(m, null);

}

}

@Override

protected void onStart() {

super.onStart();

Log.i(TAG, "onStart()");

}

@Override

protected void onResume() {

super.onResume();

dbHandler = new ContactDbHandler(getApplicationContext());

Contacts = new ArrayList<>();

if (dbHandler.getContactsCount() != 0)

Contacts.addAll(dbHandler.getAllContacts());

populateList();

Log.i(TAG, "onResume()");

}

private void forgotUser() {

SharedPreferences.Editor prefs = PreferenceManager.getDefaultSharedPreferences(getApplicationContext()).edit();

prefs.putBoolean("remember\_user", false);

prefs.putString("address", null);

prefs.putString("password", null);

prefs.apply();

}

private void setNewTab(TabHost tabHost, String tag, int title, int icon, int contentID ){

TabHost.TabSpec tabSpec = tabHost.newTabSpec(tag);

tabSpec.setIndicator(getTabIndicator(tabHost.getContext(), title, icon));

tabSpec.setContent(contentID);

tabHost.addTab(tabSpec);

}

private View getTabIndicator(Context context, int title, int icon) {

View view = LayoutInflater.from(context).inflate(R.layout.tab\_layout, null);

ImageView iv = (ImageView) view.findViewById(R.id.imageView);

iv.setImageResource(icon);

TextView tv = (TextView) view.findViewById(R.id.textView);

tv.setText(title);

return view;

}

private void alertMessageTwoButtons(String title, String text, int id) {

DialogInterface.OnClickListener logoffDialogClickListener = new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

switch (which) {

case DialogInterface.BUTTON\_POSITIVE: // Yes button clicked

Intent login = new Intent(getApplicationContext(), LogIn.class);

startActivity(login);

forgotUser();

Main.this.finish();

break;

case DialogInterface.BUTTON\_NEGATIVE: // No button clicked // do nothing

break;

}

}

};

DialogInterface.OnClickListener deleteContactDialogClickListener = new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

switch (which) {

case DialogInterface.BUTTON\_POSITIVE: // Yes button clicked

dbHandler.deleteContact(Contacts.get(longClickedItemIndex));

Contacts.remove(longClickedItemIndex);

contactAdapter.notifyDataSetChanged();

break;

case DialogInterface.BUTTON\_NEGATIVE: // No button clicked // do nothing

break;

}

}

};

DialogInterface.OnClickListener deleteNewsDialogClickListener = new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

switch (which) {

case DialogInterface.BUTTON\_POSITIVE: // Yes button clicked

new $news(Main.this).execute("delete", String.valueOf(feedList.get(longClickedItemIndex).getNewsId()));

updateListOnRefresh(myMenu.findItem(R.id.news\_action\_refresh));

break;

case DialogInterface.BUTTON\_NEGATIVE: // No button clicked // do nothing

break;

}

}

};

AlertDialog.Builder builder = new AlertDialog.Builder(this);

switch (id) {

case CONFIRM\_LOGOFF:

builder.setTitle(title)

.setMessage(text)

.setPositiveButton(R.string.btn\_yes, logoffDialogClickListener)

.setNegativeButton(R.string.btn\_no, logoffDialogClickListener).show();

break;

case CONFIRM\_DELETE\_CONTACT:

builder.setTitle(title)

.setMessage(text)

.setPositiveButton(R.string.btn\_yes, deleteContactDialogClickListener)

.setNegativeButton(R.string.btn\_no, deleteContactDialogClickListener).show();

break;

case CONFIRM\_DELETE\_NEWS:

builder.setTitle(title)

.setMessage(text)

.setPositiveButton(R.string.btn\_yes, deleteNewsDialogClickListener)

.setNegativeButton(R.string.btn\_no, deleteNewsDialogClickListener).show();

break;

}

}

private void alertMessageOneButtons(String title, String text) {

DialogInterface.OnClickListener okDialogClickListener = new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

switch (which) {

case DialogInterface.BUTTON\_NEUTRAL: // No button clicked // do nothing

break;

}

}

};

AlertDialog.Builder builder = new AlertDialog.Builder(this);

builder.setTitle(title)

.setMessage(text)

.setPositiveButton(R.string.btn\_ok, okDialogClickListener).show();

}

}

Остальной код программы можно увидеть на приложенном диске или в репозитории по ссылке <https://github.com/Gagauz2010/DigitalSchool>