**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ   
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

|  |  |
| --- | --- |
| СОГЛАСОВАНО  Профессор департамента программной инженерии факультета компьютерных наук, доктор технических наук.  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.В. Александров  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2016 г. | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_2016 г. |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. Инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл.*** | RU.17701729.508890-01 12 01-1-ЛУ | | **Мобильный социальный сервис с геопозиционированием**  **на платформе Android**  **Текст программы**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.508890-01 12 01-1-ЛУ**  Исполнитель  студент группы БПИ 142  \_\_\_\_\_\_\_\_\_\_\_/Берюхов А.С./  «\_\_\_»\_\_\_\_\_\_\_\_\_\_2016 г.  **2016** | | |  |

УТВЕРЖДЕНО

RU.17701729.508890-01 12 01-1-ЛУ

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | ***Подп. и дата*** |  | | ***Инв. № дубл.*** |  | | ***Взам. Инв. №*** |  | | ***Подп. и дата*** |  | | ***Инв. № подл.*** | RU.17701729.508890-01 12 01-1 | | **Программная реализация алгоритма европейского стандарта А5 шифрования мобильной телефонной связи**  Текст программы  RU.17701729.508890-01 12 01-1  **Листов 12**  **2016** |  |

**СОДЕРЖАНИЕ**

[1. Application 3](#_Toc446541989)

[2. QuidPost 3](#_Toc446541990)

[3. QuidСomment 4](#_Toc446541991)

[4. DataTransform 4](#_Toc446541992)

[5. activities 4](#_Toc446541993)

[5.1. DispatchActivity 4](#_Toc446541994)

[5.2. WelcomeActivity 5](#_Toc446541995)

[5.3. SignUpActivity 5](#_Toc446541996)

[5.4. LogInActivity 5](#_Toc446541997)

[5.5. MainActivity 6](#_Toc446541998)

[5.6. GetLocationActivity 7](#_Toc446541999)

[5.7. UserPageActivity 7](#_Toc446542000)

[5.8. NewPostActivity 8](#_Toc446542001)

[5.9. PostInfoActivity 8](#_Toc446542002)

[3. Приложения 9](#_Toc446542003)

[3.1. Список используемой литературы 9](#_Toc446542004)

Программный продукт состоит из двух частей: клиентской и серверной. Серверная часть представлена сервисом Parse и настраивается из клиентской части. Клиентская часть содержит 4 основных класса и 9, описывающих отдельные состояния (activities) приложения.

Из-за объемности исходного кода, который представлен на электронном носителе, далее будут представлены только краткие описания классов.

# 1. Application

Класс, содержащий глобальные константы, метод onCreate, в котором инициализируется SDK Parse и регистрируются подклассы QuidPost и QuidСomment класса ParseObject для создания соответствующих таблиц в сервисе Parse.

Также внутри содержится класс ParseLogInterceptor, который необходим для функционирования SDK Parse в Android API 14-19 версий (bugfix от разработчиков Parse).

# 2. QuidPost

Класс, наследующий класс ParseObject, определяющий структуру для хранения записей (постов) на сервере Parse.

Методы: геттеры и сеттеры – инкапсулируют обращение к полям записей:

* **public** String getName()
* **public void** setName(String value)
* **public** ParseUser getAuthor()
* **public void** setAuthor(ParseUser value)
* **public** String getText()
* **public void** setText(String value)
* **public** ParseGeoPoint getLocation()
* **public void** setLocation(ParseGeoPoint value)
* **public** PostState getState()
* **public void** setState(PostState value)
* **public static** ParseQuery<QuidPost> getQuery()

Перечисление PostState определяет возможные виды постов.

# 3. QuidСomment

Класс, наследующий класс ParseObject, определяющий структуру для хранения комментариев к записям (постам) на сервере Parse.

Методы: геттеры и сеттеры – инкапсулируют обращение к полям записей:

* **public** String getText()
* **public void** setText(String value)
* **public** ParseUser getAuthor()
* **public void** setAuthor(ParseUser value)
* **public** String getPostID()
* **public void** setPostID(String value)
* **public static** ParseQuery<QuidComment> getQuery()

# 4. DataTransform

Содержит два метода:

* **private static** String GetTimePassed(Date date, Date now , Context context)
* **public static** String GetTimePassedTillNow(Date date , Context context)

Второй метод использует первый и отображает время, прошедшее со значения date до текущего момента (до значения now) в упрощенном виде, в соответствии с локализацией, используемой в context.

# 5. activities

Далее будут описаны классы, помещенные в соответствующий каталог, и представляющие возможное поведение для отдельных состояний (экранов) приложения. В соответствие каждому классу (кроме класса DispatchActivity) соответствует XML-файл разметки экрана, содержащийся в директории res/layout.

## 5.1. DispatchActivity

Основная активность, запускаемая при запуске приложения. Не имеет соответствующего XML-файла с разметкой. В зависимости от того, содержатся ли данные об активном пользователе, отображает MainActivity (если содержится), или WelcomeActivity с предложением зарегистрироваться или войти в зарегистрированную учетную запись.

Наследует класс android.app.Activity. Переопределяет метод protected void **onCreate**(Bundle savedInstanceState).

## 5.2. WelcomeActivity

Активность, запускаемая после DispatchActivity, если приложение не находит информацию об активном пользователе. Имеет графический интерфейс, описанный в файле activity\_welcome.xml. Отображает название и краткую информацию о приложении (это первый экран, который видит пользователь), а также две кнопки: «Зарегистрироваться» и «Войти» (здесь и далее названия кнопок и других элементов интерфейса могут отличаться в зависимости от локализации), по нажатию которых осуществляется переход к соответствующим активностям.

Наследует класс android.app.Activity. Переопределяет метод protected void **onCreate**(Bundle savedInstanceState).

## 5.3. SignUpActivity

Активность, запускаемая по нажатию кнопки «Зарегистрироваться» в WelcomeActivity. Имеет графический интерфейс, описанный в файле activity\_signup.xml. Отображает три текстовых поля для ввода имени пользователя, email и пароля, и кнопку «Зарегистрироваться», по нажатию которой происходит проверка корректности введенных данных и регистрация пользователя в системе.

Наследует класс android.app.Activity. Переопределяет метод protected void **onCreate**(Bundle savedInstanceState).

Содержит два собственных метода. Первый метод private void **signup**() осуществляет проверку данных и процесс регистрации. Второй – private String check(String name, String passw, String email) – проверяет введенные данные на корректность и возвращает строку “ОК” в случае успеха, либо текст ошибки.

## 5.4. LogInActivity

Активность, запускаемая по нажатию кнопки «Войти» в WelcomeActivity. Имеет графический интерфейс, описанный в файле activity\_login.xml. По умолчанию отображает два текстовых поля для ввода имени пользователя и пароля, кнопку «Войти», по нажатию которой происходит проверка корректности введенных данных и вход пользователя в приложение. Ниже также расположена кнопка «Забыли пароль», по нажатию которой отображается дополнительное текстовое поле для ввода email и кнопка «Восстановить пароль», по нажатию которой, в случае существующего аккаунта с данным email, пользователю высылается письмо с указаниями по восстановлению пароля.

Наследует класс android.app.Activity. Переопределяет метод protected void **onCreate**(Bundle savedInstanceState).

Содержит два собственных метода. Первый метод private void **login**() осуществляет проверку данных и процесс входа в приложение. Второй – private String check(String name, String passw) – проверяет введенные данные на корректность и возвращает строку “ОК” в случае успеха, либо текст ошибки.

## 5.5. MainActivity

Активность, запускаемая после DispatchActivity, если приложение находит информацию об активном пользователе. Имеет графический интерфейс, описанный в файле activity\_main.xml. Отображает меню с кнопками «Настройки аккаунта» и «Определить местоположение», по нажатию которых происходит переход в соответствующие активности; строкой, отображающей имя пользователя.

Ниже меню располагается список записей, найденных в заданном местоположении (или строку, сообщающую об отсутствии записей) и «плавающая» кнопка[2], для добавления новых записей, по нажатию которой вызывается NewPostActivity.

Внешний вид пунктов списка определяется файлом quid\_post\_item.xml.

Наследует класс android.app.Activity. Переопределяет методы: protected void **onCreate**(Bundle savedInstanceState), public void onBackPressed(),protected void onResume().

Содержит собственные методы: private void doListQuery() – делает запрос с полученными параметрами к серверу Parse и отображает результаты в виде списка; private void **startLocationActivity**() – запускает GetLocationActivity; private ParseGeoPoint **geoPointFromLocation**(Location loc) – преобразует координату – объект типа Location в объект типа ParseGeoPoint.

## 5.6. GetLocationActivity

Активность, запускаемая по нажатию кнопки «Определить местоположение» в MainActivity. Имеет графический интерфейс, описанный в файле activity\_get\_location.xml. Отображает меню с кнопками «Назад» и «Сохранить местоположение», по нажатию которых происходит переход в MainActivity без сохранения изменений либо с изменением местоположения; строкой, отображающей подсказку для пользователя.

Ниже расположены: кнопка «Определить местоположение через геолокацию», по нажатию которой программа пытается определить местоположение с помощью встроенных функций устройства; текстовое поле для ввода адреса и кнопка «Найти по адресу», по нажатию которой программа пытается определить местоположение по введенному адресу; элемент с выбором пункта из списка предложенных для определения радиуса поиска записей.

Наследует класс android.app.Activity, реализует интерфейсы GoogleApiClient.ConnectionCallbacks, GoogleApiClient.OnConnectionFailedListener. Переопределяет методы protected void **onCreate**(Bundle savedInstanceState), protected void **onStart**(), protected void **onStop**(), public void **onConnected**(Bundle bundle), public void **onConnectionSuspended**(int i), public void **onConnectionFailed**(@NonNull ConnectionResult connectionResult).

Содержит собственный метод public String **getJSON**(String urle) – возвращает строковое представление объекта JSON, получаемого в качестве GET-запроса по URL, передаваемому в виде строки urle.

## 5.7. UserPageActivity

Активность, запускаемая по нажатию кнопки «Настройки аккаунта» в MainActivity. Имеет графический интерфейс, описанный в файле activity\_user\_page.xml. Отображает меню с кнопками «Назад» и «Сохранить местоположение», по нажатию которых происходит переход в MainActivity без сохранения изменений либо с изменением данных пользователя; строкой, отображающей подсказку для пользователя; кнопкой «Выйти» для выхода из текущего аккаунта и возвращения в DispatchActivity.

Ниже расположены: строка-подсказка «Основная информация», текстовые поля с именем пользователя, email и именем пользователя в telegram (при наличии), изображение аватара.

Наследует класс android.app.Activity. Переопределяет метод protected void **onCreate**(Bundle savedInstanceState).

## 5.8. NewPostActivity

Активность, запускаемая по нажатию кнопки «Новая запись» в MainActivity. Имеет графический интерфейс, описанный в файле activity\_new\_post.xml. Отображает меню с кнопками «Назад» и «Добавить запись», по нажатию которых происходит переход в MainActivity без сохранения изменений либо с добавлением записи; строкой, отображающей подсказку для пользователя.

Ниже расположены: текстовые поля для ввода заголовка и описания записи; элемент с выбором пункта из списка предложенных для определения типа записи; кнопка «Добавить новую запись», которая дублирует действие кнопки «Добавить запись» из меню.

Наследует класс android.app.Activity. Переопределяет метод protected void **onCreate**(Bundle savedInstanceState).

Содержит метод private void **post**(), который осуществляет процесс добавления записи в систему.

## 5.9. PostInfoActivity

Активность, запускаемая по нажатию на выбранную запись в MainActivity. Имеет графический интерфейс, описанный в файле activity\_post\_info.xml. Отображает меню с кнопками: «Назад», по нажатию которой происходит переход в MainActivity; «Поделиться», по нажатию которой отображается окно с выбором подходящего приложения, в которое передается текстовое представление текущей записи; строкой, отображающей подсказку для пользователя.

Ниже расположены: аватар пользователя, опубликовавшего запись, его имя, название и описание записи, кнопка «Показать пост на карте» – отображает окно с выбором подходящего картографического приложения, в котором будет отображена точка, к которой прикреплена запись.

Еще ниже расположен список комментариев, оставленных к этой записи (формат которых определяется файлом quid\_comment\_item.xml), текстовое поле для ввода нового и кнопка «Добавить» для добавления нового комментария.

Наследует класс android.app.Activity. Переопределяет метод protected void **onCreate**(Bundle savedInstanceState).

Содержит методы: private String getLocHref() – для получения URL для отображения геометки записи в стороннем приложении; private void addcomment() – для добавления комментария в систему; private void doCommentsQuery() – для получения и отображения в виде списка комментариев с сервера; private void findPostByID(String thisPostId) – для загрузки записи по её ID.

# 3. Приложения

## 3.1. Список используемой литературы

1. Material design – Google design guidelines [Электронный ресурс] / URL: <https://www.google.com/design/spec/material-design> (Дата обращения: 20.01.2016, режим доступа: свободный, на англ. языке).

2. FloatingActionButton – Library on GitHub [Электронный ресурс] / URL: <https://github.com/makovkastar/FloatingActionButton> (Дата обращения: 12.02.2016, режим доступа: свободный (The MIT License (MIT)), на англ. языке).

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Изм. | Номера листов (страниц) | | | | Всего листов (страниц) в документе | № документа | Входящий № сопроводительного документа и дата | Подпись | Дата |
| измененных | замененных | новых | аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |