# ОГЛАВЛЕНИЕ

[ОГЛАВЛЕНИЕ 1](#_Toc359785212)

[ВВЕДЕНИЕ 2](#_Toc359785213)

[ГЛАВА 1. АНАЛИЗ ТРЕБОВАНИЙ К ПРОГРАММЕ 4](#_Toc359785214)

[Анализ рынка мобильных приложений 4](#_Toc359785215)

[Требования к программе 7](#_Toc359785216)

[Средства разработки 15](#_Toc359785217)

[ГЛАВА 2. РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ 17](#_Toc359785218)

[Проектирование базы данных 17](#_Toc359785219)

[Проектирование программного обеспечения 21](#_Toc359785220)

[Интерфейс 28](#_Toc359785221)

[ГЛАВА 3. ТЕСТИРОВАНИЕ ПРОГРАММЫ 37](#_Toc359785222)

[Тестирование 37](#_Toc359785223)

[Функциональное тестирование 38](#_Toc359785224)

[Тестирование производительности 41](#_Toc359785225)

[Тестирование пользовательского интерфейса 44](#_Toc359785226)

[Перспективы 44](#_Toc359785227)

# ВВЕДЕНИЕ

Начиная с 2000 года, технологии мобильных устройств активно укрепляются в различных сферах человеческой деятельности. Не так давно, мобильный телефон для людей ассоциировался только со средством для использования услуг связи, предоставляемых мобильными операторами. В нынешнее же время дела обстоят иначе. Технический прогресс не стоит на месте и поэтому производители мобильных устройств стремятся вложить в свой продукт поддержку актуальных на данное время инноваций. Как следствие, современные мобильные телефоны имеют более богатый функционал по сравнению со своими предшественниками. Данный функционал позволяет упростить жизнь человеку. Существует много примеров подтверждающих этот факт. Например, не менее 50% различных покупок в США осуществляется с участием мобильного телефона, а, например, в крупных городах России внедряется система оплаты проезда в общественном транспорте с помощью мобильных телефонов.

Тенденцией последних лет является замена традиционных способов взаимодействия потребителя с окружающим миром на способы, которые могу быть предоставлены мобильными инновациями. В СМИ неоднократно звучали новости о появлении систем мобильных платежей и прочих удобных людям сервисах. Возможно, что в скором времени телефон станет заменой кошелька, видеокамеры, утренней газеты, пропуска на работу. При этом, во многих случаях, при переносе в мобильное устройство, привычная всем услуга становится гораздо более удобной в использовании, нежели устаревший ее аналог. Например, в случае с электронной газетой существует возможность в короткие сроки узнать дополнительную информацию о каком-либо событие или человеке, просто кликнув на ссылку в статье. Также возможность оплаты проезда в общественном транспорте с помощью мобильного телефона позволит экономить время и избегать возникающих в некоторых случаях неудобств.

Таким образом, из вышесказанного можно сделать предположение о том, что в скором будущем огромное количество используемых в повседневной жизни услуг будут реализованы в мобильных устройствах. На основе данного предположения можно заключить, что весьма перспективными становятся любые разработки в области мобильных технологий. Особенно актуальными являются те разработки, которые включают в себя функционал, реализующий каждодневные действия людей в окружающем их мире.

Помимо развития мобильных технологий и инноваций не менее популярной сферой в области информационных технологий является разработка мобильных приложений. Мобильные приложения могут быть применены как в бизнесе, так и в повседневной жизни людей. Приложения, ориентированные на развитие бизнеса могут сделать его более привлекательным и современным, также они позволяет привлечь новых клиентов и увеличить число продаж. Данные приложения часто являются аналогами или дополнениями корпоративного программного обеспечения. Другой вид мобильных приложений – это приложения, ориентированные на рядового потребителя. Такие программные продукты зачастую относятся к разряду развлекательных и призваны для того, что бы скрасить времяпрепровождение при использование мобильного устройства. Также существуют мобильные приложения на базе оповещения. Данные продукты располагают функционалом, который позволяет пользователю создавать некий набор оповещений или, по простому, «напоминалок». В данной выпускной квалификационной работе будет рассмотрен процесс создания приложения на базе мобильной операционной системы Android.

# ГЛАВА 1. АНАЛИЗ ТРЕБОВАНИЙ К ПРОГРАММЕ

## Анализ рынка мобильных приложений

Рынок мобильных приложений во многом зависит от рынка мобильных устройств, в частности от установленных на них операционных систем. Это объясняется тем, что большее количество приложений являются нативными, то есть, ориентированы под определенную операционную систему. Далее на диаграмме представлены компании лидеры по выпуску мобильных устройств.

Рисунок - Диаграмма "Рынок мобильных устройств"

По данным компании IDC лидирующими в рыночном соотношении мобильными операционными системами являются Android и iOS. Операционная система Windows Phone лишь в квартале 2013 года смогла выйти на третье место. Причиной этому послужило тесное сотрудничество финской компании Nokia с компанией Microsoft, производителем операционной системы Windows Phone. За время этого сотрудничества Nokia выпустила более 20 миллионов устройств с установленной на них операционной системой производства Microsoft.

Рисунок - Диаграмма "Мобильные операционные системы"

При анализе рынка мобильных приложений можно заметить, что этот рынок имеет лидирующие позиции в отношении темпов роста, как отрасль интеллектуальной продукции. Разработка мобильных приложений стала одной из ведущих областей в развитии информационных технологий. В год запуска App Stores, в 2008 году, рынок мобильных приложений только начинал формироваться, а уже к нынешнему времени он вступил в фазу активного роста. Следствием таких темпов роста является большое внимание к рынку со стороны инвесторов. В денежном выражении, за период 2009-2012 гг., объем рынка мобильных приложений в мире увеличился в 25 раз. Согласно данным J'son & Partners Consulting, за 2012 год рынок мобильных приложений в мире составил 7,83 миллиардов долларов. И, по прогнозам, к 2016 году составит 65,79 миллиардов долларов. Относительно рынка мобильных приложений в России известно следующее: общий объем рынка на 2012 год составил около 160 миллионов долларов, что в 3.5 раза больше, чем в 2011 году. По оценкам J'son & Partners Consulting, российский рынок мобильных приложений к 2016 году может вырасти еще в 8 раз, что составит около 1,3 миллиарда долларов.

Большой вклад в развитие рынка мобильных приложений вносят площадки для распространения и продажи самих приложений. Как правило, в роли этих площадок выступают интернет – магазины мобильных приложений. Во всех случаях данные интернет – магазины ориентированы под одну определенную мобильную операционную систему. Наиболее популярными интернет магазинами являются следующие:

* Apple App Store
* Android Market
* Microsoft Marketplace
* BlackBerry App World

В следующей таблице представлена статистика по вышеуказанным интернет – магазинам мобильных приложений.

|  |  |  |  |
| --- | --- | --- | --- |
| Название интернет магазина | Поддерживаемая платформа | Количество доступных приложений | Общее количество скачанных приложений, млн |
| Apple App Store | iOS | 650000 | 30000 |
| Android Market | Android | 500000 | 15000 |
| Microsoft Marketplace | Windows Phone | 125000 | Данные не разглашаются |
| BlackBerry App World | Blackberry OS | 70000 | 3000 |

Таблица - Статистика интернет - магазинов мобильных приложений

По мнению экспертов из компании J’son & Partners Consulting рынок мобильных приложений можно разделить на следующий сегменты:

* контентные приложения – это приложения, позволяющие взаимодействовать с контентом. Например, программы для прослушивания музыки, чтения книг, просмотра фильмов и видео роликов.
* Бизнес приложения – зачастую это приложения, дополняющие корпоративное программное обеспечение. Такие программы позволяют повысить скорость и удобство работников фирмы;
* Мобильные игры – наиболее востребованный сегмент рынка мобильных приложений. Такие приложения используются пользователями того, чтобы скрасить свой досуг или же просто скоротать время.
* Мобильные социальные сети – на мой взгляд, данный сегмент рынка можно поделить на три типа:
  + мобильные версии сайтов интернет сообществ;
  + социальные сети, изначально ориентированные на мобильные устройства;
  + приложения, адаптирующие некий набор функции социальной сети к возможностям мобильного устройства.

## Требования к программе

Для предъявления списка требований к разрабатываемой программе было оформлено следующее техническое задание:

1. Цель проекта: разработать программу для хранения данных о ребенке с возможностью добавления заметок и уведомлений на определенную дату.
2. Для кого разрабатывается: приложение разрабатывается для родителей, желающих сохранить памятные моменты своих детей в электронном виде
3. Основные функции:

* регистрация пользователя;
* авторизация пользователя;
* добавление и редактирование профиля ребенка;
* добавления и изменения изображения в профиле ребенка, идентифицирующего ребенка;
* добавление и редактирование заметок на определенную дату;
* возможность добавления и изменения изображения при редактировании заметок;
* в качестве источников добавляемых изображений необходимо использовать камеру мобильного устройства либо галерею;
* добавление и редактирование уведомлений.

1. Системные требования:

* минимальная версия Android: 2.3.3 (GINGERBREAD\_MR1, API level: 10);
* целевая версия Android: 4.2 (JELLY\_BEAN\_MR1, API level: 17);
* наличие в устройстве камеры.

1. Требования к отображению данных:

* форматы дат должны соответствовать следующим:
  + на формах просмотра и редактирования заметок, редактирования и добавления профиля ребенка: "dd.MM.yyyy" (пример: 01.01.2001);
  + на формах просмотра и редактирования уведомлений: "HH:mm:ss dd-MM-yyyy" (пример: 14:00:00 01.01.2001);
* формат дробных чисел должен ограничиваться тремя цифрами после запятой;
* ограничение строковых полей происходит методом урезания строки без добавления других символов.

1. Требования к формам и вводимым значениям:

* форма аутентификации:
  + значение «имя учетной записи» не должно быть меньше четырех символов;
  + значение «пароль» не должно быть меньше шести символов;
  + в случае введения некорректных значений необходимо вывести сообщение с помощью метода Toast.makeText(…) или с помощью метода текстового поля .setError(…).
* Форма регистрации:
  + значение «имя учетной записи» не должно быть меньше четырех символов;
  + значение «пароль» не должно быть меньше шести символов;
  + значение «подтверждение пароля» должно в точности соответствовать значению «пароль»;
  + в случае соответствия значений «пароль» и «подтверждение пароля» задний фон поля значения «подтверждение пароля» должен быть зеленым, в противном случае красным;
  + поля «имя», «фамилия», «отчество» не должны быть пустыми при выполнении регистрации;
  + в случае введения некорректных значений необходимо вывести сообщение с помощью метода Toast.makeText(…) или с помощью метода текстового поля .setError(…).
* Форма редактирования профиля ребенка:
  + поля «имя», «фамилия», «отчество» не должны быть пустыми при окончании редактирования профиля;
  + поля «вес» и «рост» - дробное или целое число;
  + значение «дата рождения» должно быть оформлено с помощью элемента управления DatePicker;
  + в случае введения некорректных значений необходимо вывести сообщение с помощью метода Toast.makeText(…) или с помощью метода текстового поля .setError(…).
* Форма выбора текущего профиля ребенка:
  + доступные профили должны быть оформлены в виде списка;
  + текущий выбранный профиль должен быть подсвечен зеленым цветом;
  + переход на другие формы осуществляется с помощью элемента управления ActionBar.
* Форма редактирования заметки:
  + поля «заголовок» и «описание» не должны быть пустыми;
  + значение «дата заметки» должно быть оформлено с помощью элемента управления DatePicker;
  + в случае введения некорректных значений необходимо вывести сообщение с помощью метода Toast.makeText(…) или с помощью метода текстового поля .setError(…).
* Форма просмотра заметок на дату:
  + заметки должны отображаться списком в соответствии с текущей выбранной датой;
  + переход на другие формы осуществляется с помощью элемента управления ActionBar.
* Форма редактирования уведомления:
  + поля «заголовок» и «описание» не должны быть пустыми;
  + значение «время уведомление» должно быть оформлено с помощью элемента управления TimePicker;
  + значение «дата уведомления» должно быть оформлено с помощью элемента управления DatePicker;
  + в случае введения некорректных значений необходимо вывести сообщение с помощью метода Toast.makeText(…) или с помощью метода текстового поля .setError(…).
* Форма просмотра уведомлений:
  + уведомления должны отображаться списком в соответствии с текущей выбранной датой;
  + переход на другие формы осуществляется с помощью элемента управления ActionBar.

1. Макеты форм:
   * форма аутентификации:

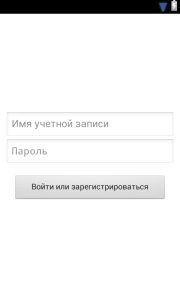


Рисунок – Макет формы аутентификации

* форма регистрации:

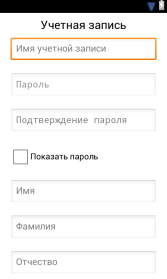


Рисунок – Макет формы регистрации

* форма редактирования профиля ребенка:

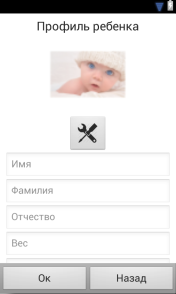


Рисунок – Макет формы редактирования профиля ребенка

* форма выбора текущего профиля ребенка:

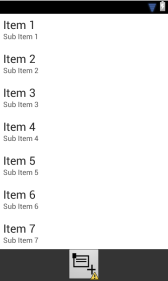


Рисунок – Макет формы выбора текущего профиля ребенка

* форма редактирования заметок:

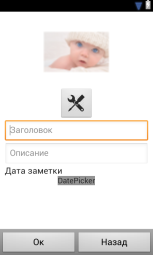


Рисунок – Макет формы редактирования заметок

* форма просмотра заметок на дату:

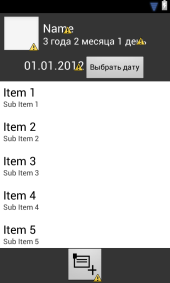


Рисунок – Макет формы просмотра заметок

* форма просмотра уведомлений:

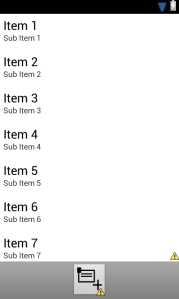


Рисунок – Макет формы просмотра уведомлений

* форма редактирования уведомлений:

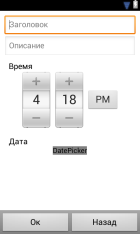


Рисунок - Макет формы редактирования уведомлений

1. Варианты переходов:

* Форма аутентификации:
  + форма регистрации;
  + форма выбора профиля ребенка.
* Форма регистрации:
  + форма редактирования профиля ребенка.
* Форма редактирования профиля ребенка:
  + форма выбора профиля ребенка.
* Форма выбора профиля ребенка:
  + форма просмотра заметок;
  + форма редактирования профиля ребенка;
  + форма просмотра уведомлений;
  + форма аутентификации.
* Форма просмотра заметок:
  + форма редактирования заметки;
  + форма выбора профиля ребенка;
  + форма просмотра уведомлений;
  + форма аутентификации.
* Форма редактирования заметки:
  + форма просмотра заметок.
* Форма редактирования уведомления:
  + форма просмотра уведомлений.
* Форма просмотра уведомлений:
  + форма редактирования уведомления;
  + форма просмотра заметок;
  + форма выбора профиля ребенка;
  + форма аутентификации.

1. Поведение приложения при исключительных ситуациях: при возникновении исключительных ситуаций, таких как инициализация исключения приложением, программа должна выполнить откат неудачного действия, если оно было выполнено частично, и выполнить процедуру выхода из учетной записи.

## Средства разработки

В качестве основного средства разработки была выбрана свободная интегрированная среда разработки модульных кроссплатформенных приложений Eclipse (версия 3.8.0). Данная среда разработки, изначально разрабатываемая фирмой IBM а в дальнейшем переданная независимому от IBM сообществу, широко применяется в области разработки всевозможных программ и приложений. Eclipse имеет полностью открытый исходный код, а архитектура этой среды разработки позволяет дополнять ее различными модулями. В основе IDE Eclipse лежит язык программирования Java, поэтому она является платформо – независимым продуктом.

Для разработки приложения под мобильную операционную систему Android был установлен плагин Android Development Tools (ADT), разработанный специально для IDE Eclipse и содержащий в своей основе инструментарий для эмуляции мобильного устройства. Разработка логической части приложения осуществлялась на объектно-ориентированном языке Java. Вёрстка форм приложения выполнена с использованием языка расширяемой разметки XML.

Для хранения данных в мобильном приложении была использована легковесная встраиваемая система управления базами данных Sqlite. Даная СУБД по умолчанию встроена в операционную систему Android. Она поддерживает стандартные возможности реляционных баз данных и, кроме того, требует очень небольшого количество памяти для работы – около 250 килобайт.

В качестве второстепенных средств разработки были выбраны следующие продукты:

* Git – распределенная система контроля версий.
* Tortoise Git – графический интерфейс для контроля версий в репозитории Git.
* Microsoft Visio – векторный графический редактор диаграмм и блок-схем.
* DeZign for Databases – графический редактор ER-моделей.
* ObjectAid Class Diagram – плагин для eclipse, позволяющие строить диаграммы классов.
* DDMS (Dalvik Debug Monitor Server) – компонент среды разработки Eclipse, используемый для диагностики разрабатываемого мобильного приложения.

# ГЛАВА 2. РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## Проектирование базы данных

При проектировании базы данных разрабатываемого мобильного приложения были определены следующие объекты:

* учетная запись;
* ребенок;
* заметка;
* уведомление.

Определение данных объектов исходит из требований к программе. Также на основании требований были определены атрибуты вышеуказанных объектов:

* учетная запись:
  + имя родителя;
  + фамилия родителя;
  + отчество родителя;
  + дата рождения родителя;
  + имя учетной записи;
  + пароль;
* ребенок:
  + имя;
  + фамилия;
  + отчество;
  + вес;
  + рост;
  + дата рождения;
* заметка:
  + заголовок;
  + описание;
  + дата заметки;
* уведомление:
  + заголовок;
  + описание;
  + дата уведомления.

На основании этих данных была разработана логическая модель базы данных:



Рисунок - Логическая модель базы данных

**Таблица «Account»**

Таблица содержит сведения о родителе, а также логин и пароль учетной записи.

| ***Поле*** | ***Комментарии*** |
| --- | --- |
| AccountId | Идентификационный номер учетной записи, первичный ключ |
| ParentName | Имя родителя |
| ParentSurname | Фамилия родителя |
| ParentMiddlename | Отчество родителя |
| Login | Имя учетной записи |
| Password | Пароль от учетной записи |

Таблица - Описание таблицы "Account"

**Таблица «Children»**

Таблица содержит сведения о ребенке.

| ***Поле*** | ***Комментарии*** |
| --- | --- |
| ChildrenId | Идентификационный номер ребенка, первичный ключ |
| Weight | Весь ребенка |
| Growth | Рост ребенка |
| ChildrenName | Имя ребенка |
| ChildrenSurname | Фамилия ребенка |
| ChildrenMiddlename | Отчество ребенка |
| AccountId | Идентификационный номер учетной записи, внешний ключ |
| Awatar | Фотография ребенка |

Таблица - Описание таблицы "Children"

**Таблица «Note»**

Таблица содержит сведения о добавляемых заметках.

| ***Поле*** | ***Комментарии*** |
| --- | --- |
| NoteId | Идентификационный номер заметки, первичный ключ |
| Description | Описание заметки |
| Postdate | Дата заметки |
| ChildrenId | Идентификационный номер ребенка, внешний ключ |
| Photo | Изображение, хранящееся в заметке |
| Title | Заголовок заметки |

Таблица - Описание таблицы "Note"

**Таблица «Notice»**

Таблица содержит сведения о добавляемых уведомлениях.

| ***Поле*** | ***Комментарии*** |
| --- | --- |
| NoticeId | Идентификационный номер уведомления, первичный ключ |
| NotifiDateTime | Дата и время уведомления |
| Title | Заголовок уведомления |
| Description | Описание уведомления |
| ChildrenId | Идентификационный номер ребенка, внешний ключ |

Для представления типов данных в таблицах реализована физическая модель.

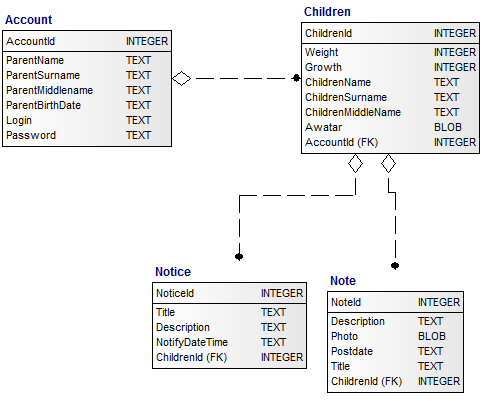


Рисунок - Физическая модель базы данных

На основе логической и физической моделей были написаны скрипты для реализации базы данных в мобильном приложении.

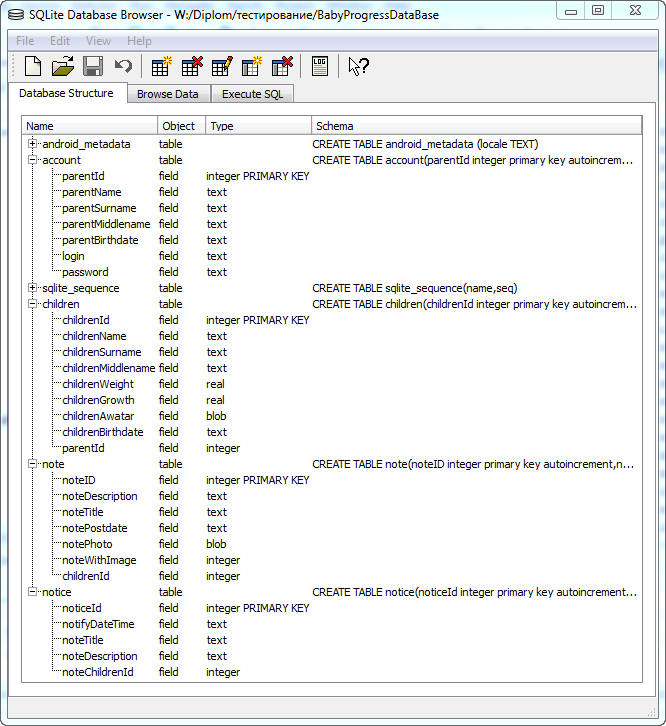


Рисунок - Реализация базы данных в программе

Из рисунка (рис. 11) видно, что все таблицы базы данных созданы в соответствии с логической и физической моделями. Помимо проектируемых таблиц операционной системой Android были созданы две служебные таблицы:

* android\_metadata – таблица, содержащая информацию о локализации;
* sqlite\_sequence – таблица, содержащая информацию о количестве строк всех таблиц базы данных.

## Проектирование программного обеспечения

Для проектирования программного обеспечения был использован графический язык UML. UML – язык графического описания объектного моделирования в области разработки программного обеспечения. В процессе проектирования были созданы следующие диаграммы:

* диаграмма прецедентов;
* диаграмма классов, взаимодействующих с базой данных;
* диаграмма классов без взаимодействия с базой данных;
* диаграмма классов, определяющих шаблон проектирования «состояние».

Диаграмма прецедентов – это диаграмма, отражающая отношения различных актеров с имеющимися прецедентами. Под актерами понимается некая роль, взаимодействующая с рассматриваемым объектом, а под прецедентом – некая последовательность действий. Эта диаграмма позволяет описать систему на концептуальном уровне. В данном случае были выявлены следующие актеры и соответствующие им прецеденты:

|  |  |
| --- | --- |
| **Прецедент** | **Актер** |
| регистрация | пользователь |
| авторизация | пользователь |
| редактирование профиля ребенка | пользователь |
| редактирование заметки | пользователь |
| редактирование уведомления | пользователь |
| инициализация уведомления | мобильное устройство |

Таблица - Актеры и соответствующие им прецеденты

Таким образом, исходя из данных таблицы, можно построить следующую диаграмму прецедентов:



Рисунок - Диаграмма прецедентов

Для реализации объектно-ориентированных методик разработки программного обеспечения были спроектированы классы программы. В первую очередь, для проектирования классов, за основу были взяты объекты базы данных. Эти классы необходимы для использования данных, хранящихся в базе данных. Использование подразумевает собой добавление данных в базу, редактирование и удаление. Также были спроектированы классы, используемые в кастомизации стандартных элементов управления, формировании логики программы и выполнении процедур по конвертации данных. В качестве процедур по конвертации данных были использованы следующие процедуры:

* преобразование дат к необходимым форматам;
* преобразование используемых пользователем изображений в массивы байт;
* преобразование массивов байт в изображение.

Перечень спроектированных классов:

* Account – физическая реализация учетной записи.
* Children – физическая реализация профиля ребенка.
* Note – физическая реализация заметки.
* Notice – физическая реализация уведомления.
* DataManager – класс, предоставляющий методы взаимодействия со статическими переменными проекта, методы для конвертации данных.
* DataAdapter – класс, предоставляющий методы взаимодействия с базой и статические переменные форматов дат.
* ActivityEnum – перечисляемый тип данных, множеством значений которого являются идентификаторы форм.
* MyAsyncTask – класс, необходимый для запуска асинхронного запроса с выводом сообщения на экран.
* ListViewNoteAdapter – адаптер кастомизированных пунктов элемента управления ListView, в котором содержится список заметок.
* ListViewChildrenAdapter – адаптер кастомизированных пунктов элемента управления ListView, в котором содержится список профилей детей.
* ListViewNoticeAdapter – адаптер кастомизированных пунктов элемента управления ListView, в котором содержится список уведомлений.
* BabyProgressDataBaseHelper – класс, содержащий скрипты создания таблиц базы данных и статические константы, необходимые для взаимодействия с базой данных.
* State – абстрактный класс, от которого происходит наследование классов – состояний.
* ChangeChildrenProfileState – класс, реализующий состояние пребывания на форме выбора профиля ребенка.
* ChildrenProfileState – класс, реализующий состояние пребывания на форме профиля ребенка.
* EditNoteState – класс, реализующий состояние пребывания на форме редактирования заметки.
* LoginState – класс, реализующий состояние пребывания на форме аутентификации.
* NotesState – класс, реализующий состояние пребывания на форме просмотра заметок на дату.
* RegistrationState – класс, реализующий состояние пребывания на форме регистрации.
* ViewChildrenProfileState – класс, реализующий состояние пребывания на форме просмотра профиля ребенка.
* ViewNoteState – класс, реализующий состояние пребывания на форме просмотра заметки.
* EditNoticeState – класс, реализующий состояние пребывания на форме редактирования уведомления.
* NotificationsState – класс, реализующий состояние пребывания на форме просмотра заявок.
* NotificationUtils – класс, позволяющий создавать уведомления в статус баре.
* AlarmReceiver – класс, инициирующий создание уведомления в статус баре на момент наступления даты и времени уведомления.
* NotificationAsyncTask – класс, использующийся для создания уведомления в дополнительном потоке.

При проектировании классов были использованы шаблоны проектирования. Шаблон проектирования (паттерн) – повторимая архитектурная конструкция, представляющая собой решение проблемы проектирования в рамках некоторого часто возникающего контекста. Использованные в работе шаблоны проектирования представлены ниже.

Одиночка – шаблон проектирования, гарантирующий, что какой – либо класс имеет только единственный экземпляр, и предоставляющий глобальную точку доступа к этому экземпляру.

Состояние – шаблон проектирования, который представляет собой реализацию конечного автомата. Задача паттерна состоит в изменении поведения объекта при изменении его внутреннего состояния.

Стратегия – поведенческий шаблон проектирования, предназначенный для определения семейства алгоритмов, инкапсуляции каждого из них и обеспечения их взаимозаменяемости. Данный паттерн позволяет менять выбранный алгоритм независимо от объектов – клиентов, которые его используют.

MVC – схема использования нескольких шаблонов проектирования, с помощью которых модель данных приложения, пользовательский интерфейс и взаимодействие с пользователем разделены на три отдельных компонента так, что модификация одного из компонентов оказывает минимальное воздействие на остальные.

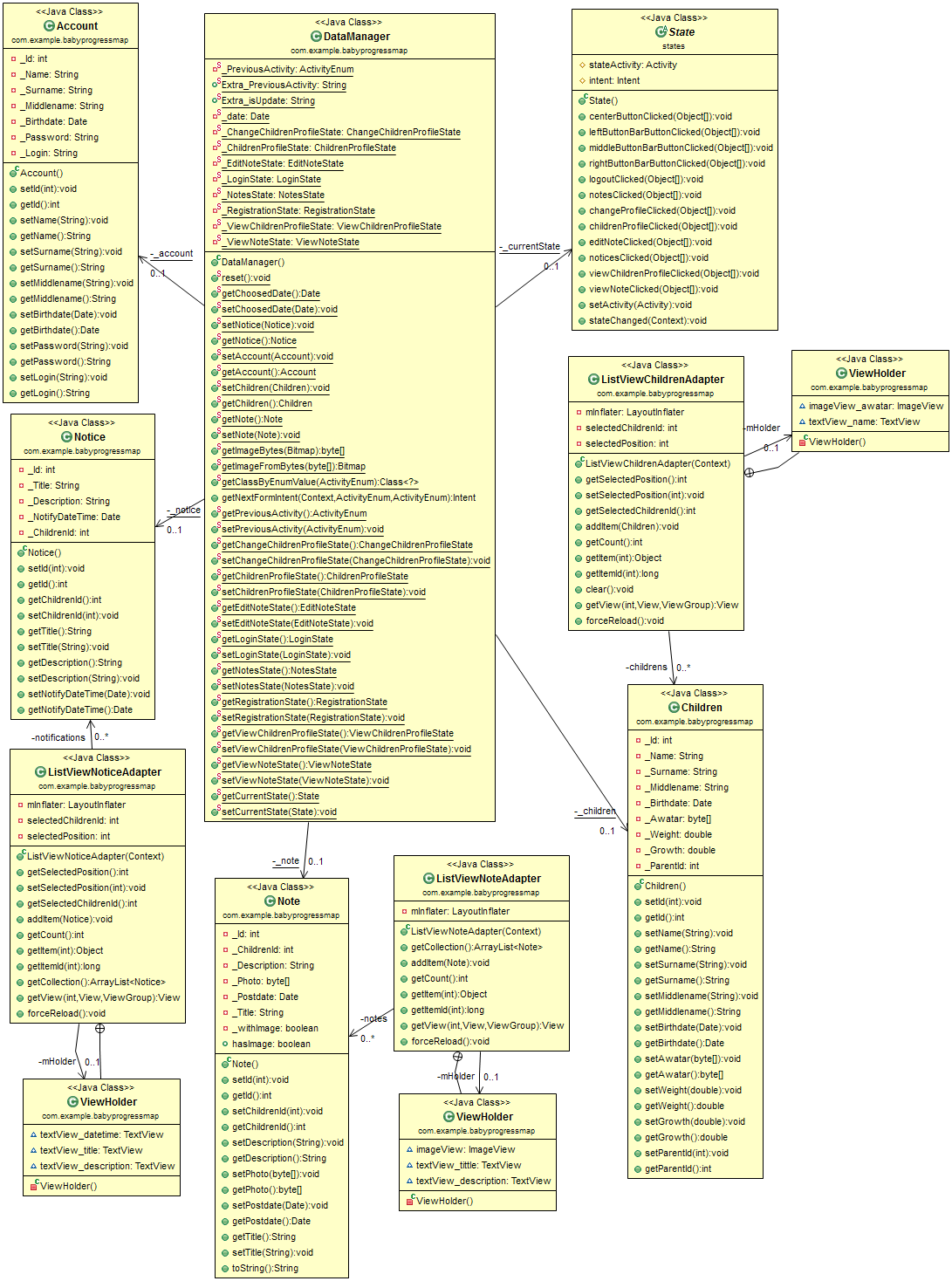


Рисунок - Диаграмма классов без взаимодействия с базой данных

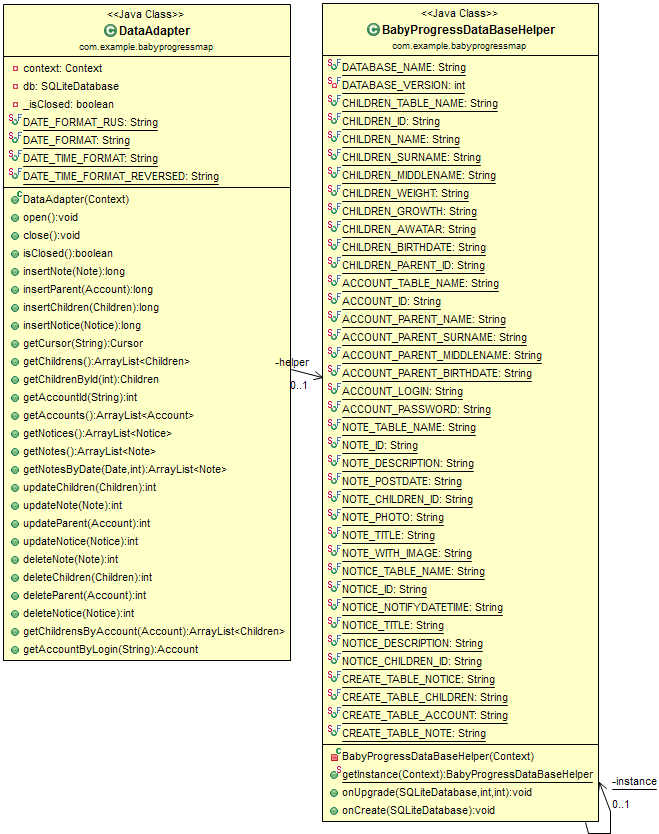


Рисунок - Диаграмма классов, взаимодействующих с базой данных

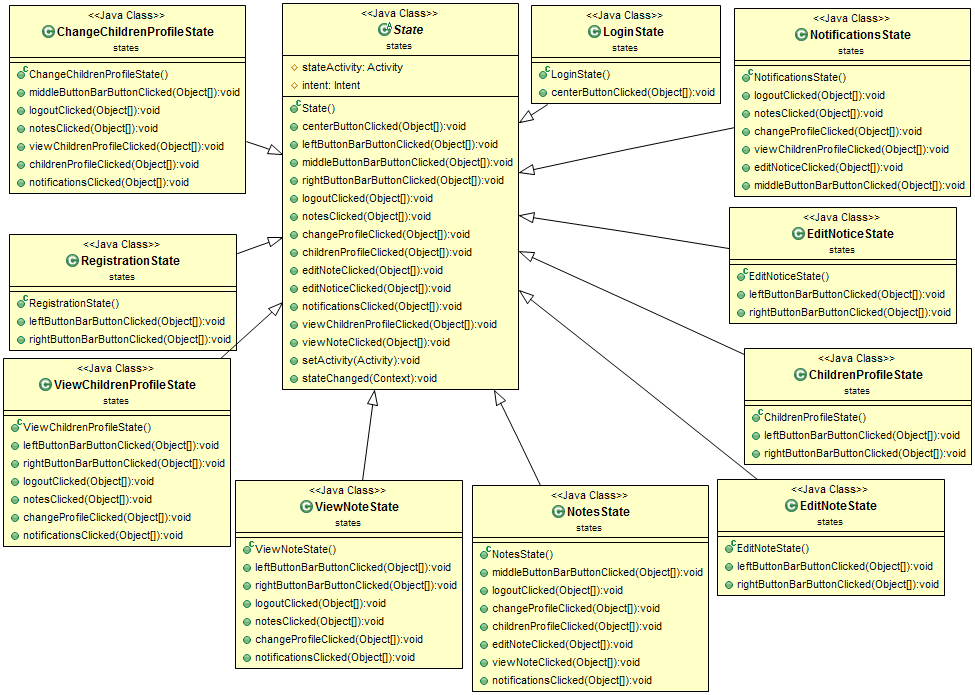


Рисунок - Диаграмма классов, определяющих шаблон проектирования «состояние».

## Интерфейс

Графический интерфейс пользователя в Android основан на иерархии View и ViewGroup. View объекты – это, как правило, графические виджеты, такие как кнопки, текстовые поля и невидимые контейнеры, которые определяют расположение или варианты использования дочерних элементов. Каждая из форм пользовательского интерфейса Android состоит из макета – файл определяющий набор элементов управления на форме – и класса, который выполняет загрузку элементов управления и содержит логику формы.

В процессе разработки графического интерфейса мобильного приложения были разработаны следующие формы:

Форма аутентификации – форма необходима для входа пользователем в учетную запись.

Возможные переходы:

* форма регистрации – в случае, если введенного пользователем имени учетной записи не существует;
* форма выбора профиля ребенка – в случае, если пользователь ввел корректные данные.

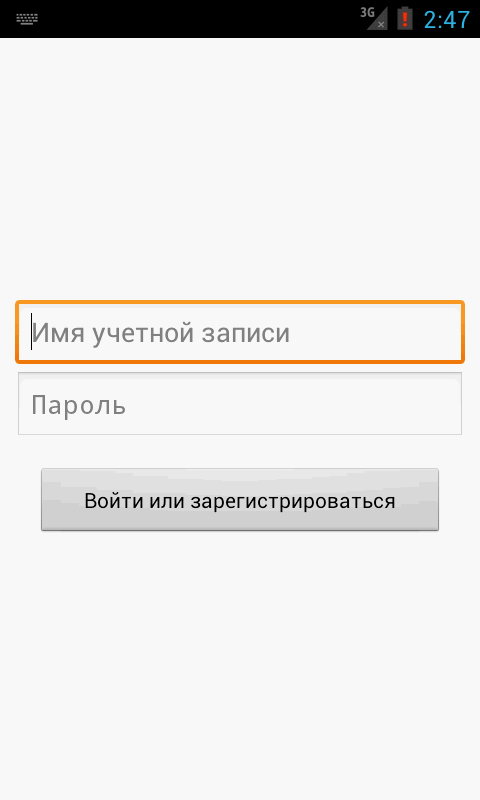


Рисунок - Форма аутентификации

Форма регистрации – форма необходима для регистрации пользователем новой учетной записи.

Возможные переходы:

* форма редактирования профиля ребенка – переход осуществляется в случае, если пользователем были введены корректные данные и новая учетная запись успешно создана;
* форма аутентификации – только после нажатия пользователем кнопки «назад».

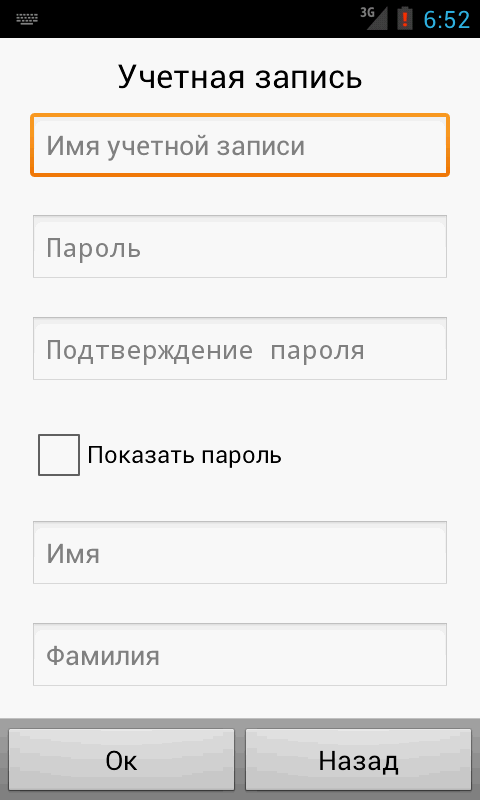


Рисунок - Форма регистрации

Форма редактирования профиля ребенка – форма не обходима для внесения изменений в данные о ребенка.

Возможные переходы:

* форма выбора профиля ребенка – переход происходит в случае ввода пользователем корректных данных и подтверждения изменения профиля ребенка нажатием кнопки «ок»;
* форма выбора профиля ребенка – при условии, что эта форма была предыдущей и нажата кнопка «назад»;
* форма просмотра профиля ребенка – при условии, что эта форма была предыдущей и нажата кнопка «назад».

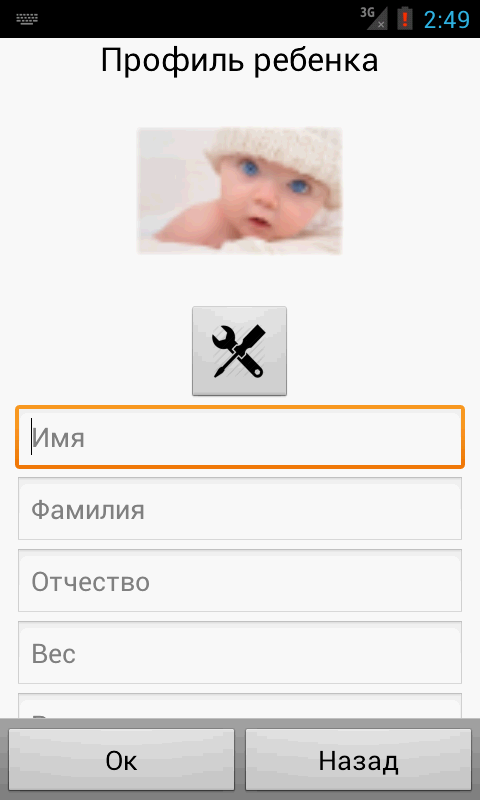


Рисунок – Форма редактирования профиля ребенка

Форма просмотра профиля ребенка – используется для просмотра данных, содержащихся в текущем профиле ребенка.

Возможные переходы:

* форма редактирования профиля ребенка – при нажатии на кнопку «изменить»;
* форма выбора профиля ребенка:
  + при нажатии на кнопку «Выбор профиля» элемента управления ActionBar;
  + при нажатии на кнопку «назад», при условии, что форма выбора профиля ребенка была предыдущей;
* форма просмотра заметок на дату:
  + при нажатии на кнопку «Заметки» элемента управления ActionBar;
  + при нажатии на кнопку «назад», при условии, что форма выбора заметок была предыдущей.



Рисунок – Форма просмотра профиля ребенка

Форма профиля ребенка – используется для выбора текущего профиля а также для создания нового профиля ребенка.

Возможные переходы:

* форма редактирования профиля ребенка – при нажатии на кнопку добавления нового профиля;
* форма просмотра заметок на дату – при нажатии на кнопку «заметки» элемента управления ActionBar;
* форма аутентификации – при нажатии на кнопку «выход» элемента управления ActionBar.

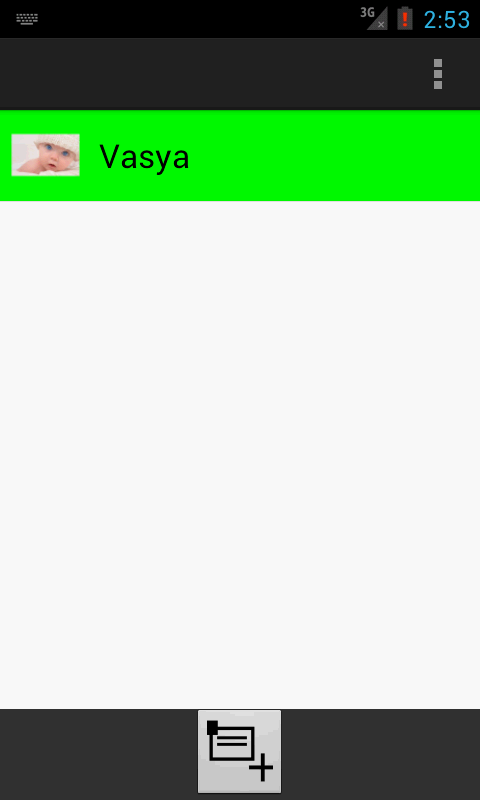


Рисунок – Форма выбора профиля ребенка

Форма просмотра заметок на дату – форма служит для просмотра добавленных на определенную дату заметок и для добавления новой заметки.

Возможные переходы:

* форма просмотра профиля ребенка – при нажатии на кнопку «профиль ребенка» элемента управления ActionBar;
* форма выбора профиля ребенка – при нажатии на кнопку «выбор профиля» элемента управления ActionBar;
* форма аутентификации - при нажатии на кнопку «выход» элемента управления ActionBar;
* форма редактирования заметки – при нажатии на кнопку добавления заметки.

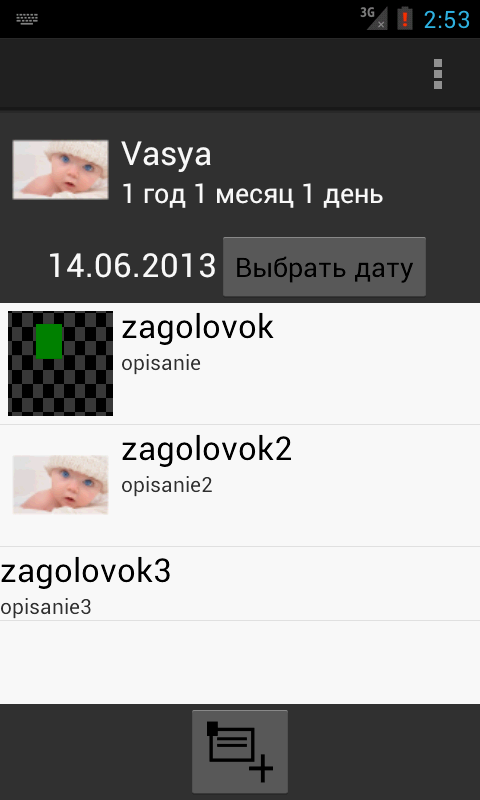


Рисунок – Форма просмотра заметок на дату

Форма просмотра заметки – форма служит для отображения заметки на весь экран.

Возможные переходы:

* форма просмотра заметок на дату – при нажатии на кнопку «назад»;
* форма редактирования заметки – при нажатии на кнопку «изменить»;
* форма просмотра заметок на дату – при нажатии на кнопку «заметки» элемента управления ActionBar;
* форма выбора профиля ребенка – при нажатии на кнопку «выбор профиля» элемента управления ActionBar;
* форма аутентификации – при нажатии на кнопку «выход» элемента управления ActionBar.

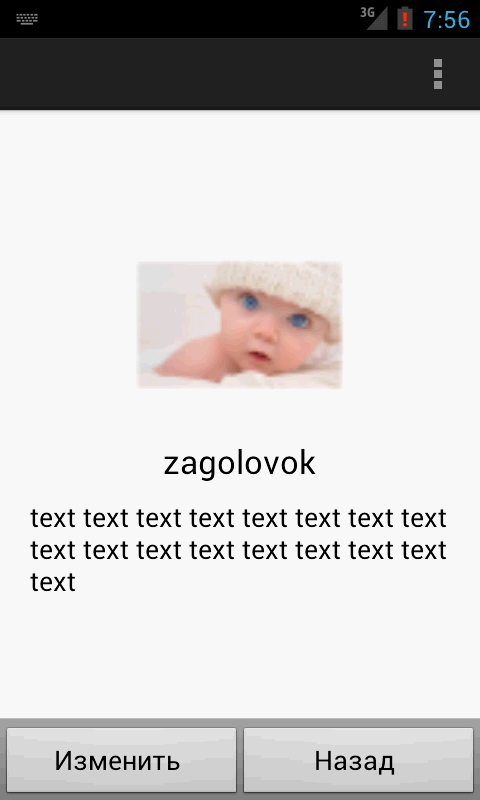


Рисунок - Форма просмотра заметки

Форма редактирования заметки – служит для внесения изменений в выбранную заметку или добавления новой.

Возможные переходы:

* форма просмотра заметок на дату – при нажатии на кнопку «назад» или «ок» и при условии, что эта форма была предыдущей;
* форма просмотра заметки – при нажатии на кнопку «назад» или «ок» и при условии, что эта форма была предыдущей.

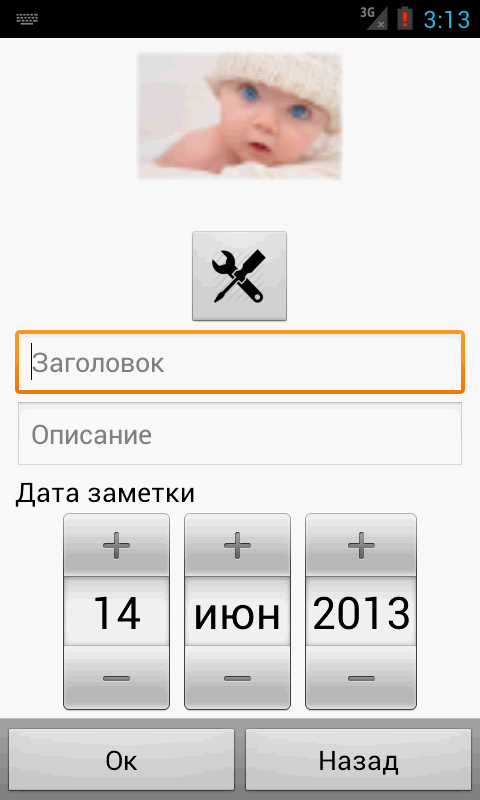


Рисунок - Форма редактирования заметок

Форма редактирования уведомлений – служит для добавления нового уведомления или внесения изменений в уже имеющиеся.

Возможные переходы:

* форма просмотра уведомлений – при нажатии на кнопку «ок» или кнопку «назад».

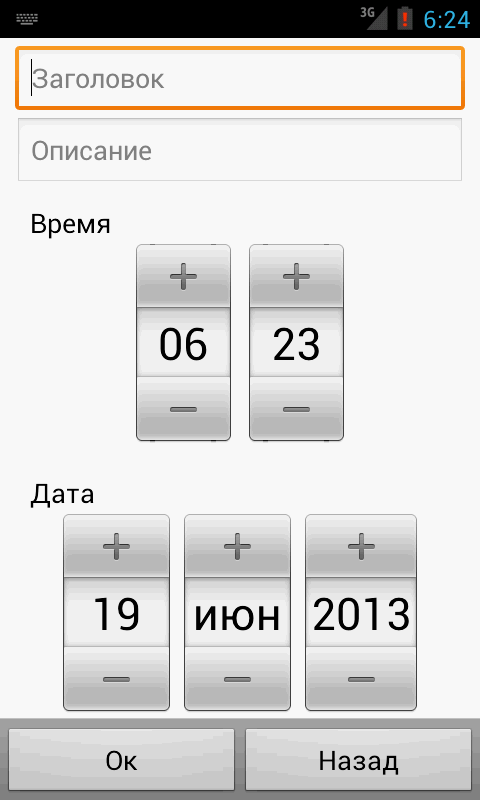


Рисунок - Форма редактирования уведомлений

Форма просмотра уведомлений – форма служит для просмотра и добавления уведомлений на определенные дату и время.

Возможные переходы:

* форма просмотра профиля ребенка – при нажатии на кнопку «Профиль ребенка» элемента управления ActionBar;
* форма выбора профиля ребенка – при нажатии на кнопку «Выбор профиля» элемента управления ActionBar;
* форма просмотра заметок на дату – при нажатии на кнопку «Заметки» элемента управления ActionBar;
* форма аутентификации – при нажатии на кнопку «Профиль ребенка» элемента управления ActionBar.

# ГЛАВА 3. ТЕСТИРОВАНИЕ ПРОГРАММЫ

## Тестирование

Тестирование программы – это процесс, целью которого является выявление дефектов и установление корректности работы анализируемой программы. Также в ходе тестирования необходимо выявить потенциальные источники появления ошибок в программном коде. Самый простой способ сделать это: перебрать все возможные варианты вводимых данных и проверить реакцию программы на эти данные. Этот метод не подходит для информационных систем большого масштаба, так как требует составление большого количества контрольных примеров и огромных трудозатрат. Но, в случае с разработанным в данной работе мобильным приложением, этот метод тестирования программного обеспечения полностью себя оправдывает. Ниже перечислены методики тестирования, которые актуальны при проверке работоспособности разработанного мобильного приложения:

1. Функциональное тестирование – это тестирование программного обеспечения в целях проверки реализуемости функций, заложенных в программу.
2. Тестирование производительности – это тестирование программного обеспечения, нацеленное на проверку быстроты работы при выполнении заложенных в программу функций.
3. Тестирование пользовательского интерфейса – в контексте разработки мобильного приложения этот вид тестирования применяется для проверки корректного отображения пользовательского интерфейса на мобильных устройствах с экранами разных размеров.

## Функциональное тестирование

В основе разрабатываемого мобильного приложения лежат функции по взаимодействию с базой данных, функции по работе с изображениями и функции, обеспечивающие корректные переходы по формам приложения. Тестирование функции по взаимодействию с базой данных будет изображено на примере тестирования формы регистрации и формы редактирования профиля ребенка, при этом, для выявления корректности работы процедур по взаимодействию с базой данных будет использован SQLite Database Browser. Ход тестирования будет представлен в виде нескольких скриншотов на момент начала и окончания тестирования.

Тестирование формы регистрации

На примере формы регистрации проведен тест добавления данных в базу данных мобильного приложения. Для этого инициирована процедура регистрация с использованием тестовых данных.

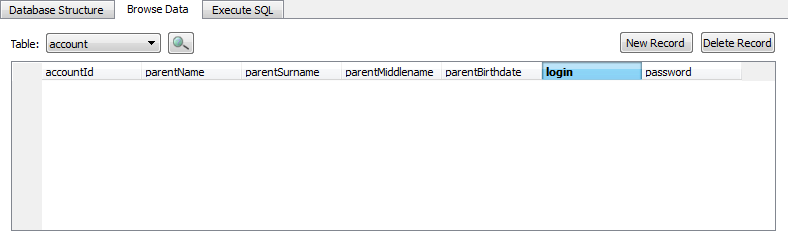


Рисунок – Данные таблицы «account»

Из рисунка (рис. 25) видно, что на момент начала тестирования в таблице «account» отсутствовали какие-либо данные об учетных записях. Далее была использована форма регистрации для создания новой учетной записи.

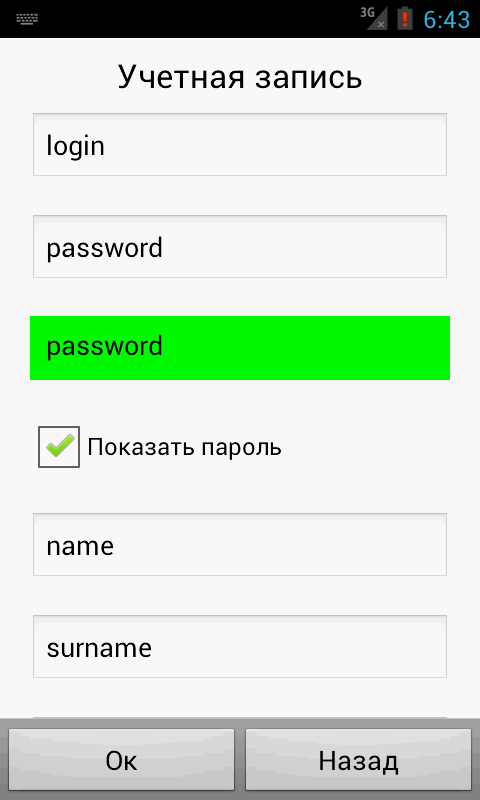


Рисунок - Регистрация новой учетной записи

На рисунке (рис. 27) видно, что после создания новой учетной записи с использованием формы регистрации данные были успешно добавлены в таблицу базы данных.

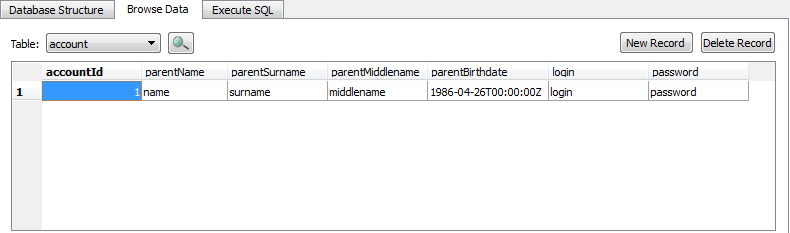


Рисунок - Данные таблицы "account"

Тестирование формы редактирования профиля ребенка

На примере формы редактирования профиля ребенка проведено тестирование процедур по изменению данных, содержащихся в базе данных, а также тестирование функции по работе с изображениями. На момент начала тестирования в таблице «children» уже имелась строка, содержащая информацию о профиле ребенка. Эти данные можно увидеть на следующем рисунке:

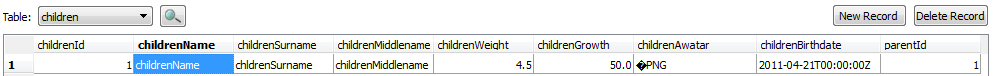


Рисунок - Данные таблицы "children"

Для проверки функций по изменению данных была использована форма редактирования профиля ребенка со следующими тестовыми данными (рис. 29):

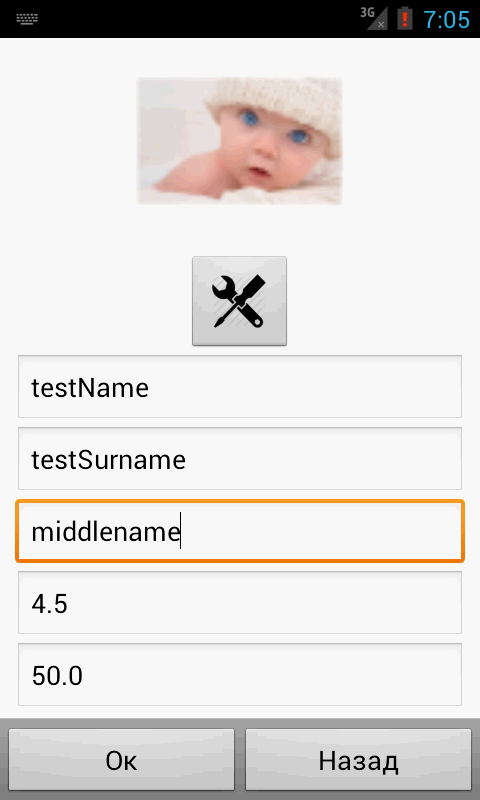
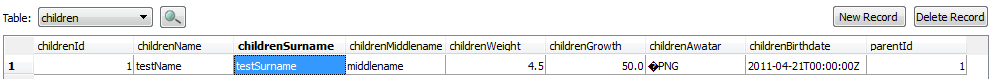


Рисунок - Редактирование существующего профиля ребенка

После применения изменений на рисунке (рис. 30) можно увидеть, что данные таблицы «children» успешно изменены.



После проведения тестирования можно заключить о корректности работы функций по изменению данных таблицы. Также можно заключить о корректности выполнения процедур по работе с изображениями исходя из следующих фактов:

* изображение успешно помещено в таблицу базы данных в виде массива байт;
* массив байт успешно извлечен из базы данных и отображен на форме в виде полноценного изображения.

## Тестирование производительности

Для проведения тестирования производительности разработанного мобильного приложения использовалась программа DDMS. Это программа от компании Google, использующаяся для диагностики ресурсов, потребляемых мобильным приложением. В качестве основных функций программы можно выделить следующие:

* сервис перенаправления портов;
* взаимодействием с файловой системой эмулятора мобильного устройства;
* создание скриншотов экрана эмулятора;
* информирование о запущенных процессах и потоках эмулятора;
* логирование всех выполняющихся процессов;
* эмуляция входящих звонков и СМС;
* информирование о потребляемых запущенными процессами ресурсах.

Мониторинг потоков

Данный вид мониторинга предоставляет информацию о выполняющихся потоках конкретного процесса, в данном случае разработанного мобильного приложения.

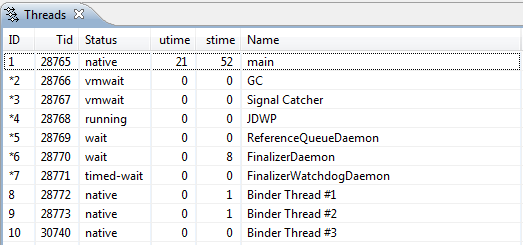


Рисунок - Окно мониторинга потоков

Окно мониторинга потоков процесса представляет собой таблицу, каждая запись которой отображает основную характеристику потока. Основными системными потоками каждого процесса являются следующие:

* main – основной поток, протекающий в процессе;
* GC – поток, отвечающий за «сборку мусора»;
* Signal Catcher – поток, отвечающий за работу по приему сигнала;
* JDWP – поток, отвечающий за работу протокола взаимодействия среды разработки и виртуальной машины Java.

| ***Поле*** | ***Описание*** |
| --- | --- |
| ID | Уникальный идентификатор потока, выданный виртуальной машиной Java |
| Tid | Уникальный идентификатор потока, выданный ядром Linux |
| Status | Текущий статус потока |
| utime | Общее время, затраченное на выполнение пользовательского кода (единица измерения jiffy[[1]](#footnote-1)) |
| stime | Общее время, затраченное на выполнение системного кода (единица измерения jiffy) |
| Name | Наименование потока |

Таблица - Описание полей окна мониторинга потоков

Статус потока позволяет определить разновидность работы потока в текущий момент времени. В каждый момент времени статус потока может представлять собой значение из следующего перечня:

* running – выполняется код приложения;
* sleeping – поток в режиме сна вследствие вызова метода Thread.sleep();
* monitor – ожидание блокировки монитора;
* wait – поток в режиме ожидания вследствие вызова метода Object.wait();
* native – поток выполняет собственный код;
* vmwait – поток в режиме ожидания ресурсов виртуальной машины;
* zombie – поток находится в режиме завершения;
* init – поток находится в режиме инициализации;
* starting – поток находится в режиме запуска;

Из рисунка (рис. 32) видно, что на момент проведения тестирования значительно преобладает выполнение системного кода. Суммарное время выполнения пользовательского кода составляет всего около 210 миллисекунд. Отсюда можно сделать вывод, что пользовательский код программы не требователен к ресурсам и обеспечивает высокую производительность.

Мониторинг распределения оперативной памяти

Данный вид мониторинга позволяет выявить значение используемой приложением оперативной памяти в общем и по каждому виду распределения в частности.

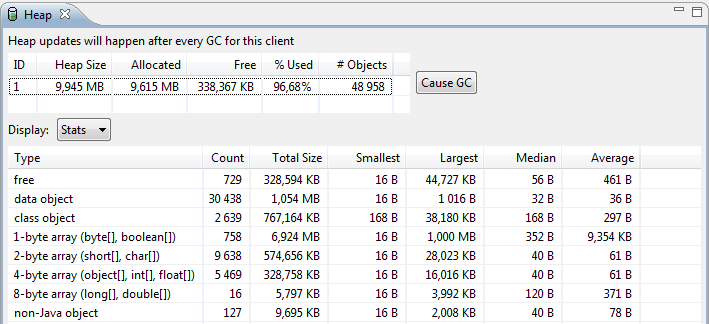


Рисунок - Окно мониторинга распределения оперативной памяти

Общий объем используемой оперативной памяти приложением отображается в верхней части окна, в поле «Heap Size». На данном рисунке (рис. 34) видно, что общий объем оперативной памяти, потребляемый приложением, составляет около 10 мегабайт. Данное значение потребляемой оперативной памяти объясняется тем, значительная часть памяти требуется на обработку изображений, которые хранятся в виде массивов байт.

## Тестирование пользовательского интерфейса

Данный вид тестирования чрезвычайно важен при разработке мобильных приложений. Это объясняется тем, что мобильные устройства оснащаются экранами с разным соотношением сторон и плотностью пикселей. Поэтому графический интерфейс приложения должен быть гибким, чтобы корректно отображаться на экранах, обладающих разными размерами. Тестирование пользовательского интерфейса проведено на примере формы редактирование профиля ребенка, так как эта форма имеет наибольше число элементов управления, по сравнению с другими формами разработанного приложения. Для проведения тестирования использовался функционал IDE Eclipse, позволяющий отобразить верстку формы на экране выбранного размера. В ходе тестирования проверке подвергалась корректность отображения элементов управления формы на экранах следующего размера:

* 10.1" (1280х800 mdpi)
* 4.0" (480х800 hdpi)
* 3.3" (240x480 ldpi)
* 2.7" (240x320 ldpi)

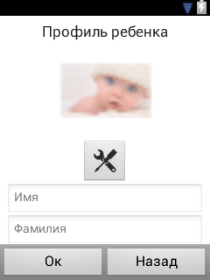


Рисунок - Тестируемая форма на экране 2.7"

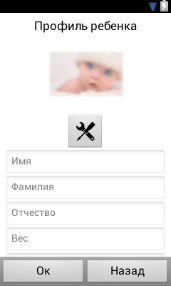


Рисунок - Тестируемая форма на экране 3.3"

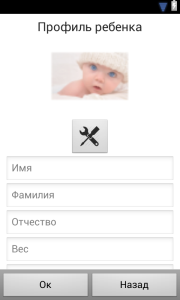


Рисунок - Тестируемая форма на экране 4.0"



Рисунок - Тестируемая форма на экране 10.1"

На основе изображений формы на различных экранах можно заметить, что на экранах с малыми и средними размерами не отображены сразу все элементы управления. Данный недостаток компенсируется тем, что среди элементов верстки формы использован элемент ScrollView. Этот элемент позволяет совершать скроллинг[[2]](#footnote-2) формы вверх и вниз. Вывод: данная форма может корректно отображаться на мобильных устройствах с различными размерами экрана. Данный вывод применим и к остальным формам разработанного приложения, так как при их верстке также учитывался принцип масштабируемости интерфейса.

## Перспективы

В перспективе развития разработанного мобильного приложения планируется реализовать функцию синхронизации данных с сервером. Под синхронизацией данных понимается создание резервной копии базы данных приложения на удаленном сервере. Помимо создания резервной копии данных, эта функция должна предоставить возможность копирования данных с удаленного сервера в локальную базу данных приложения. Также к перспективе развития программы можно отнести создание более привлекательного интерфейса и создания дополнительных опций просмотра данных.

1. Единица измерения времени. Длительность определяется системой и обычно составляет 10 мс. [↑](#footnote-ref-1)
2. Форма представления информации, при которой содержимое (текст, изображение) двигается в вертикальном или горизонтальном направлении. Таким образом, скроллинг не изменяет содержимое, но передвигает «камеру». [↑](#footnote-ref-2)