Факультет Информационных технологий

|  |  |  |  |
| --- | --- | --- | --- |
| **Направление** |  | **Кафедра** |  |
|  | *(код)* |  | *(аббревиатура)* |

#### ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ работа

|  |  |
| --- | --- |
| на тему | Разработка приложения ведения заметок |
| для ОС android | |
|  | |

|  |  |  |  |
| --- | --- | --- | --- |
| Обучающийся |  |  | ***\_\_\_\_\_\_\_\_\_\_\_\_\_*** |
|  | *(Ф.И.О. полностью)* |  | *подпись* |
| Руководитель |  |  | ***\_\_\_\_\_\_\_\_\_\_\_\_\_*** |
|  | *(Ф.И.О. полностью)* |  | *подпись* |
| Консультант |  |  |  |
| Заведующий кафедрой | *(Ф.И.О. полностью)* |  | *подпись* |
|  | *(Ф.И.О. полностью)* | *подпись* |

МОСКВА 2016 г.

Содержание

Изм.

Лист

№ докум.

Подпись

Дата

Лист

3

Содержание

Выполнил

Провер.

Н. Контр.

Утверд.

Пояснительная записка

Лит.

Листов

82

Введение…………………………………………………………………………...3

1 Аналитическая часть……………………………………………………………4

1.1 Анализ программ, решающих класс задач в рамках темы диплома………4

1.2 Анализ аппаратных средств, на которых функционирует разрабатываемое ПО…………………………………………………………………………………17

1.3 Анализ общего программного обеспечения……………………………….18

1.4Анализ интегрированных сред, применяемых для разработки ПО…..........21

1.5 Математическое обеспечение решаемых задач………………………........24

1.6 Постановка задачи……………………………………………………………26

2 Разработка программного обеспечения………………………………………27

2.1 Архитектура разрабатываемого ПО………………………………………..27

2.2 Реализация алгоритмов программы ………………………………………..39

2.3 Пользовательский интерфейс программы…………………………………45

2.4 Тестирование разрабатываемого ПО………………………………………49

3 Эффективность разрабатываемого ПО ……………………………………...66

Заключение………………………………………………………………………70

Список используемых источников……………………………………………..71

Приложение А. Исходный код....………………………………………………72

ВВЕДЕНИЕ

Изм.

Лист

№ докум.

Подпись

Дата

Лист

Введение

На сегодняшний день мы всё чаще и чаще пользуемся заметками. Мы ими пользуемся всегда. Иногда без заметок сложно запоминать про все встречи, про важные дела и про всю информацию, которая нам необходима.

Для мобильных устройств с операционной системой Android можно найти разные приложения ведения заметок в магазине приложений Google Play. Но у каждого приложения имеются разные недостатки.

Целью данной дипломной работы является разработка приложения ведения заметок для телефонов с операционной системой Android. Для выполнения цели необходимо выполнить следующие задачи:

* Анализ аналогов;
* Анализ аппаратных средств;
* Анализ операционной системы;
* Анализ интегрированной среды разработки;
* Разработку приложения ведения заметок;
* Экономическое обоснование проекта.

1 Аналитическая часть

Изм.

Лист

№ докум.

Подпись

Дата

Лист

*Аналитическая часть*

1.1Анализ программ, решающих класс задач в рамках темы диплома

Произведён анализ аналогов. К аналогам относятся следующие приложения:

* Notes;
* Notepad;
* Заметки L.

Приложение Notes можно найти в Google Play [7]. При первом запуске приложения запускается активность с пустым списком заметок. Пустой список заметок представлен на рисунке 1.1.

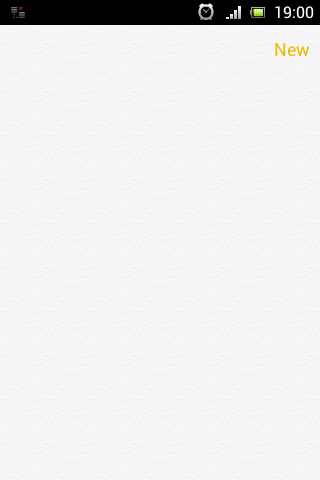


Рисунок 1.1 – Активность с пустым списком заметок

Для создания заметки используется кнопка New. После нажатия на эту кнопку происходит переход в активность редактора заметки (см. рисунок 1.2).



Рисунок 1.2 –Активность редактора заметки

После создания заметки появляется созданная заметка (см. рисунок 1.3).

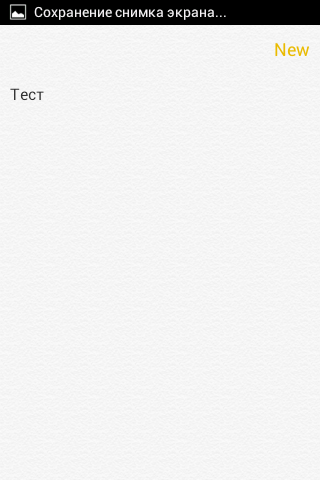


Рисунок 1.3 – Созданная заметка

Недостатки приложения Notes:

* В данном приложении не используется меню;
* Отсутствие возможности добавления изображений;
* Отсутствие поиска заметок;
* В окне создания заметки нет кнопки удаления заметки;
* Для удаления заметки необходимо стереть строку, в которой был написан текст выбранной заметки;
* Приложение может сохранять заметку, когда был произведён выход из активности создания заметки;
* Приложение в списке заметок не выводит дату создания/изменения заметки.

Приложение Notepad можно найти в Google Play [8]. При первом запуске приложения запускается активность с пустым списком заметок. Данная активность представлена на рисунке 1.4.



Рисунок 1.4 – Активность с пустым списком заметок

Для создания заметки используется кнопка со знаком плюс. После нажатия на эту кнопку происходит переход в активность редактора заметки. Активность редактора заметки представлена на рисунке 1.5.

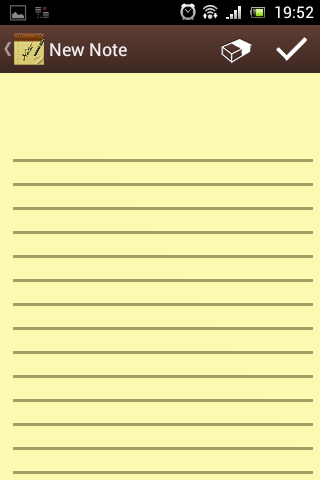


Рисунок 1.5 – Активность редактора заметки

После нажатия на кнопку сохранения заметка будет отображаться в списке заметок. Вид списка заметок представлен на рисунке 1.6.

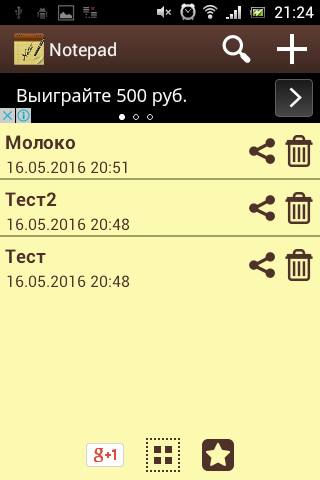


Рисунок 1.6 – Список заметок

После нажатия на кнопку поиска можно найти необходимую заметку. Результат поиска представлен на рисунке 1.7.

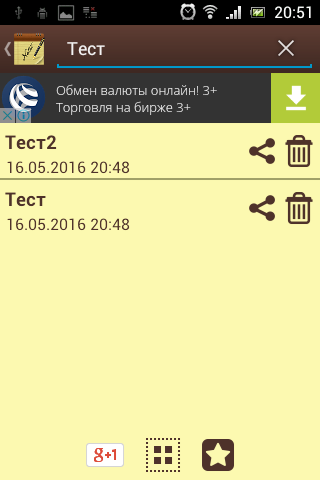


Рисунок1.7 – Результат поиска

В данном приложении имеется возможность удаления заметки в их списке (см. рисунок 1.8).

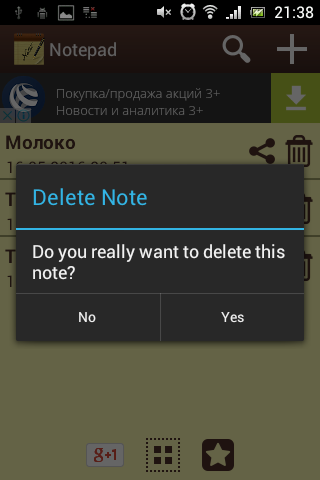


Рисунок 1.8 – Удаление заметки

Список после удаления заметки представлен на рисунке 1.9.

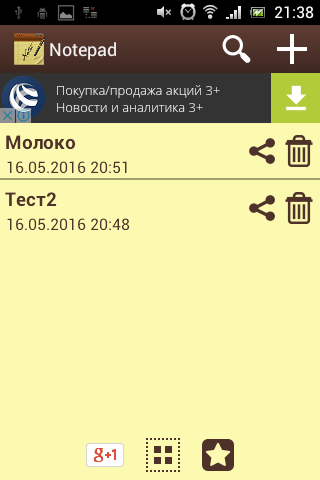


Рисунок 1.9 – Список после удаления заметки

В активности создания/изменения заметки можно очистить данные в текстовом поле. Если нажать на кнопку с изображением ластика, появится диалог с предложением о очищении данных в текстовом поле (см. рисунок 1.10).

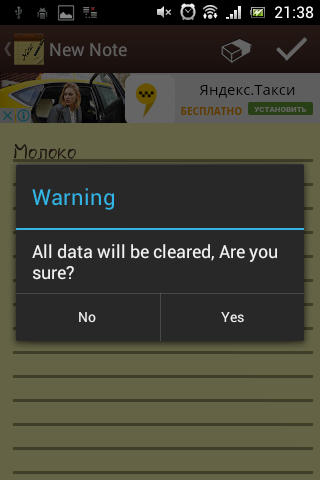


Рисунок 1.10 – Предложение о очищении данных в текстовом поле

Если в активности создания/редактирования заметки при пустом текстовом поле нажать на кнопку сохранения заметки, то появится всплывающее сообщение о том, что надо заполнить текстовое поле для сохранения заметки (см. рисунок 1.11).

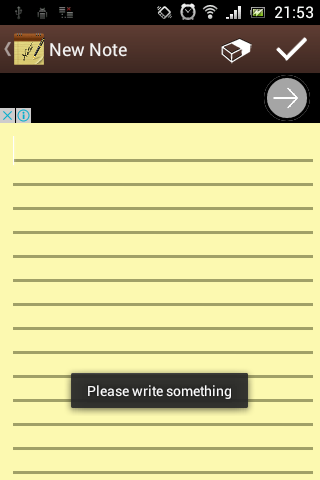


Рисунок 1.11 - Всплывающее сообщение

В данном приложении присутствует меню – вызывается с помощью кнопки меню в телефоне и с помощью кнопки в приложении, которая находится внизу посередине. Меню представлено на рисунке 1.12.

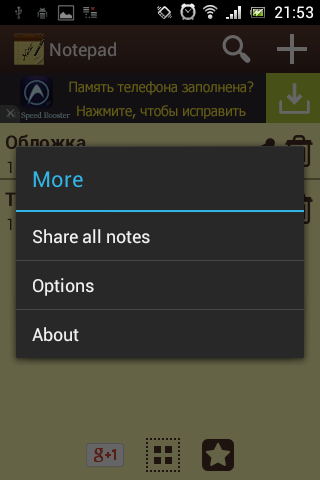


Рисунок 1.12 – Меню в приложении Notepad

Если нажать на элемент меню «Share all notes», в окне появятся приложения, куда можно отправить заметку. Список этих приложений представлен на рисунке 1.13.

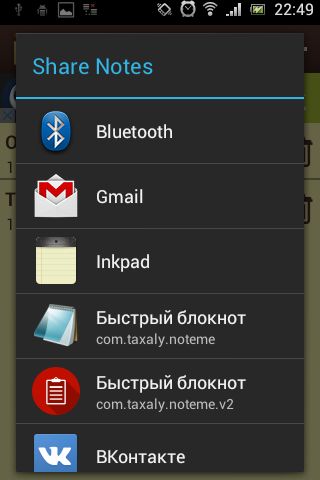


Рисунок 1.13 – Список приложений, в которых можно поделиться заметкой

Если нажать на элемент меню «Options», то будет произведён переход в настройки приложения (см. рисунок 1.14).

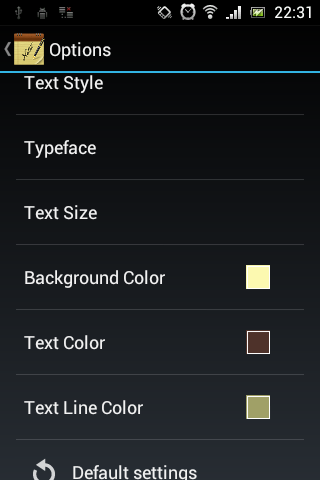


Рисунок 1.14 – Настройки приложения

Если нажать на элемент меню «About», то покажется информация о разработчике и приложении (см. рисунок 1.15).

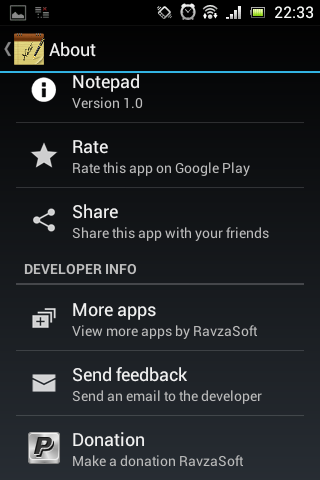


Рисунок 1.15 – Информация о приложении и разработчике

Недостатки приложения Notepad:

* Отсутствие возможности загрузки изображения в заметку;
* Частое появление рекламы в приложении.

Приложение Заметки можно найти в Google [9]. При первом запуске приложения появляется образец заметки. Образец заметки представлен на рисунке 1.16.

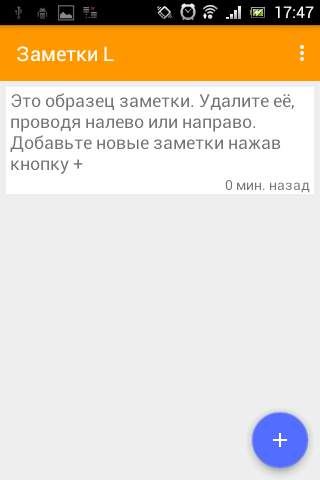


Рисунок 1.16 – Образец заметки

Если удалить образец заметки, то образец заметки переместится в корзину. Кроме данного образца, будет ещё образец удалённой заметки. Корзина представлена на рисунке 1.17.

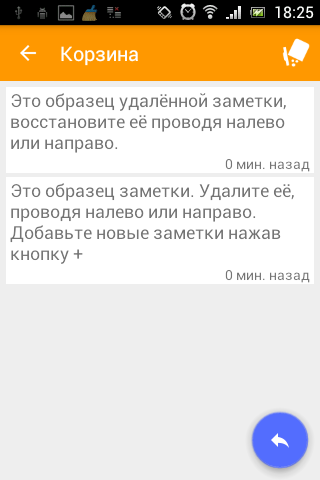


Рисунок 1.17 – Корзина

Если нажать на кнопку очищения корзины, появится диалог. Диалог при очищении корзины представлен на рисунке 1.18.

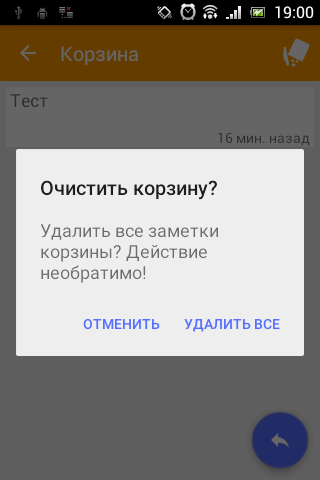


Рисунок 1.18 – Диалог при очищении корзины

После очищения корзины появится текст о том, что корзина пустая. Пустая корзина представлена на рисунке 1.19.

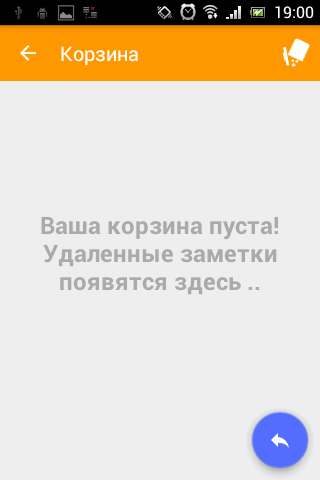


Рисунок 1.19 – Пустая корзина

Если приложение содержит пустой список, то в приложении выводится соответствующий текст. Пустой список заметок представлен на рисунке 1.20.

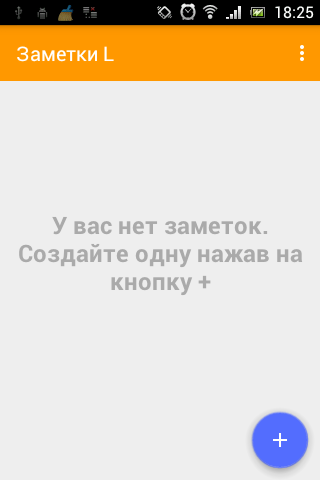


Рисунок 1.20 – Пустой список заметок

После нажатия на кнопку создания заметки происходит переход в активность редактора заметки (см. рисунок 1.21).

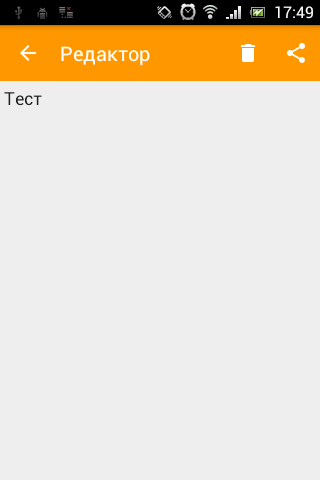


Рисунок 1.21 – Активность редактора заметки

Созданная заметка представлена на рисунке 1.22.

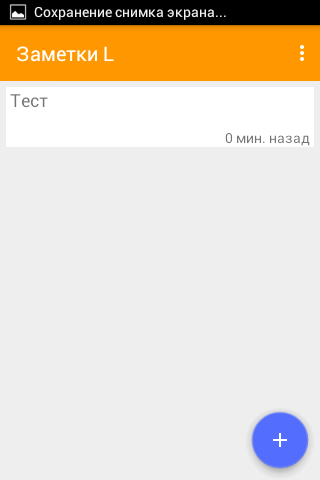


Рисунок 1.22 – Созданная заметка

Если в активности редактора нажать на кнопку с изображением корзины, то заметка удалится. Если в активности редактора нажать на кнопку «Поделиться», появится список приложений, в которых можно поделиться заметкой (см. рисунок 1.23).

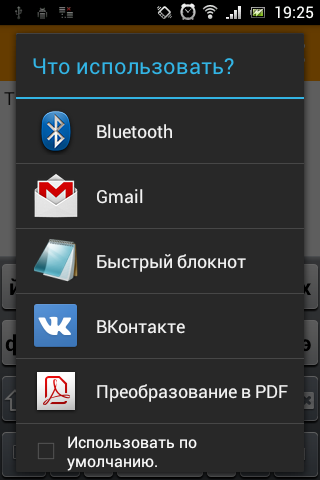


Рисунок 1.23 – Список приложений, в которых можно поделиться заметкой

Недостатки приложения «Заметки L»:

* В приложении отсутствует возможность работы с изображениями;
* Заметки сохраняются в случае выхода из активности редактора.

1.2 Анализ аппаратных средств, на которых функционирует разрабатываемое ПО

Разрабатываемое ПО предназначено для телефонов, планшетов с ОС Android.

Для запуска ПО необходимы следующие характеристики устройств:

* Процессор Qualcomm MSM 7227AA, 800 МГц, 1 ядро (и лучше);
* Видеопроцессор Adreno 200,1 ядро (и лучше);
* Диагональ экрана 3,2 дюйма, разрешение экрана 320 x 480 (и выше);
* Камера 3,2 МП, 2048 x 1536 (и выше);
* Оперативная память 512 МБ (и выше);
* Должно быть подключение к компьютеру через USB;
* Операционная система Android версии от 4.0 до 4.2;

1.3 Анализ общего программного обеспечения

Почти все современные планшеты, смартфоны, нетбуки и мобильные телефоны работают на операционной системе Android.

Android – это открытая операционная система, которая позволяет другим разработчикам внедрять новые программы. На основе операционной системы Android были разработаны электронные книги, очки Google и другие полезные изобретения.

У современных смартфонов с Android удобный интерфейс, их клавиатура похожа на компьютерную. Заметным преимуществом таких смартфонов является также тот факт, что они стабильно работают без сбоя в программе и никогда не зависают.

Android открыт для онлайн-приложений, поэтому можно без труда установить любую игру. Эта операционная система способна выдержать большое количество форматов. Android обладает широким функциональным обеспечением.

Данная операционная система способна поддерживать основные сервисы Google. Благодаря этому обладатели смартфонов и нетбуков на Android могут получать доступ к электронной почте, YouTube. Все, кто зарегистрировался на Gmail, получают возможность использовать приложения от Google.

Каждый человек может настроить пользовательский интерфейс по своему собственному желанию. Например, особенности Android предполагают возможность получать прогноз погоды в любое время и в любом месте.

Операционная система Android предусматривает наличие нескольких рабочих столов, благодаря этому можно быстро переходить с одного приложения на другое. Одновременно человек может слушать музыку и общаться в Интернете.

Современные смартфоны на Android предоставляют своим обладателям шанс проникнуть в «root» – в самое сердце системы. Недаром говорят, что Android дарит полную свободу действий.

Операционная система Android имеет навигатор, с помощью которого можно быстро найти нужный магазин или ресторан. Система дисплея может изменяться с горизонтальной на вертикальную, что позволяет просматривать картинки в удобном для пользователя положении.

В период развития Android пережила много этапов. Эта операционная система появилась на рынке, когда для неё не было ни единой программы навигации, ни одного редактора офисных документов и других приложений. Однако на сегодняшний день в магазине софта Google Play представлено свыше 90 тысяч различных полезных приложений. Огромную часть утилит можно загрузить бесплатно. [6]

Произведён анализ версий операционной системы. К данным версиям относятся следующие версии:

* Android 4.0 «Ice Cream Sandwich»;
* Android 4.1 «Jelly Bean»;
* Android 4.2 «Jelly Bean».

Особенности Android 4.0:

* Более широкие возможности управления запущенными приложениями;
* Android Beam, которая позволяет обмениваться различными данными между устройствами с операционной системой Android;
* Быстрая настройка Wi-Fi соединения между двумя устройствами для обмена файлами, передачи потокового аудио и видео;
* Режим быстрого ответа, смысл которого заключается в том, что при поступлении входящего звонка, SMS или другого сообщения на экране появляется специальная область, с помощью которой отвечать на звонки стало гораздо удобнее и быстрее.
* Широкий выбор разнообразных эффектов для изображений и видео, полученных с камеры;
* Автоматическая фокусировка камеры и возможность панорамной съёмки;
* Более тесная интеграция с социальной сетью Google+ и возможность на основе сети организовывать видеоконференции.

Особенности Android 4.1:

* Более быстрая и плавная работа интерфейса благодаря технологии «Project Butter»;
* Разворачиваемые уведомления и их улучшенная интеграция с сервисами;
* Включение голосового поиска простым движением пальца снизу-вверх;
* Возможность получения голосового ответа на некоторые вопросы;
* Функция Google Now;
* Преобразование голоса в текст без подключения к сети;
* Браузер Chrome установлен по умолчанию;
* Возможность изменения размера виджетов;
* Функция Android Beam поддерживает передачу больших файлов;
* Новый способ просмотра снимков в приложении камеры;
* Обновления в Google Play будут касаться не всего приложения, а только изменяемых компонентов;
* Наличие встроенного виджета Sound Search для распознавания музыки;
* Более высокое разрешение фотографий контактов;
* Более лёгкое управление операционной системой;
* Расширенная языковая поддержка;
* Многочисленные оптимизации интерфейса.

Особенности Android 4.2:

* Возможность набора текста методом Swype, не отрывая пальца от дисплея;
* Наличие многопользовательского режима для планшетов;
* Функция «Photo Sphere» для создания расширенных панорамных снимков;
* Наличие новых «карточек» с ответами на вопросы из базы знаний Google;
* Улучшенная работа голосового поиска;
* Новая версия сервиса Google Now, обеспечивающего получение пользователем необходимой информации;
* Поддержка стандарта Miracast для потоковой передачи видео на другой дисплей;
* Добавлено приложение Daydream, выводящее на дисплей полезную информацию во время простаивания;
* Улучшена работа функции Android Beam;
* Добавлена возможность увеличить элементы интерфейса при помощи тройного касания пальцем;
* Более широкая поддержка приложениями вызовов из области уведомлений;

1.4 Анализ интегрированных средств разработки, применяемых для разработки ПО

Eclipse — свободная интегрированная среда разработки модульных кроссплатформенных приложений. Развивается и поддерживается Eclipse Foundation.

Основные инструментальные средства Eclipse Java включают в себя: редактор исходного кода (создание и редактирование исходного текста программ), средства отладки и интеграции с Ant. Кроме этого в Eclipse доступны множество бесплатных и коммерческих дополнений (плагинов), таких, как инструментальные средства создания схем UML, разработка баз данных и другие.

Eclipse — это только платформа, которая предоставляет возможность разрабатывать дополнения, называемые плагинами, которые естественным образом встраиваются в платформу. В Eclipse доступны дополнения для следующих языков: C/C++, Html, Cobol, Perl, Php, Ruby и др. Вы можете также разработать собственное дополнение для расширения возможностей Eclipse [1].

Eclipse разработана для широкого круга операционных систем, таких как Linux, Microsoft Windows и Mac OS. Для ее запуска требуется JVM (Java Virtual Machine) — виртуальная Java-машина, а также JDK (Java Development Kit) — набор для Java-разработки. Загрузить данные пакеты можно с официального сайта разработчика Java — http://java-sun.com. В стандартной сборке ALT Linux «Мастер» данные пакеты уже предустановлены.

В таблице 1.1 представлены минимальные и рекомендуемые системные требования для работы Eclipse.

Таблица 1.1- Системные требования

|  |  |  |
| --- | --- | --- |
| Требование | Минимальное значение | Рекомендуемое значение |
| Версия Java | 1.4.0 | 1.6.0 и выше |
| Оперативная память | 128 Мб | 1 Гб и более |
| Свободное пространство на жёстком диске | 300 Мб | 1 Гб и более |
| Процессор | 533 МГц | 1,5 ГГц и более |

Интерфейс интегрированной среды разработки Eclipse представлен на рисунке 1.24.

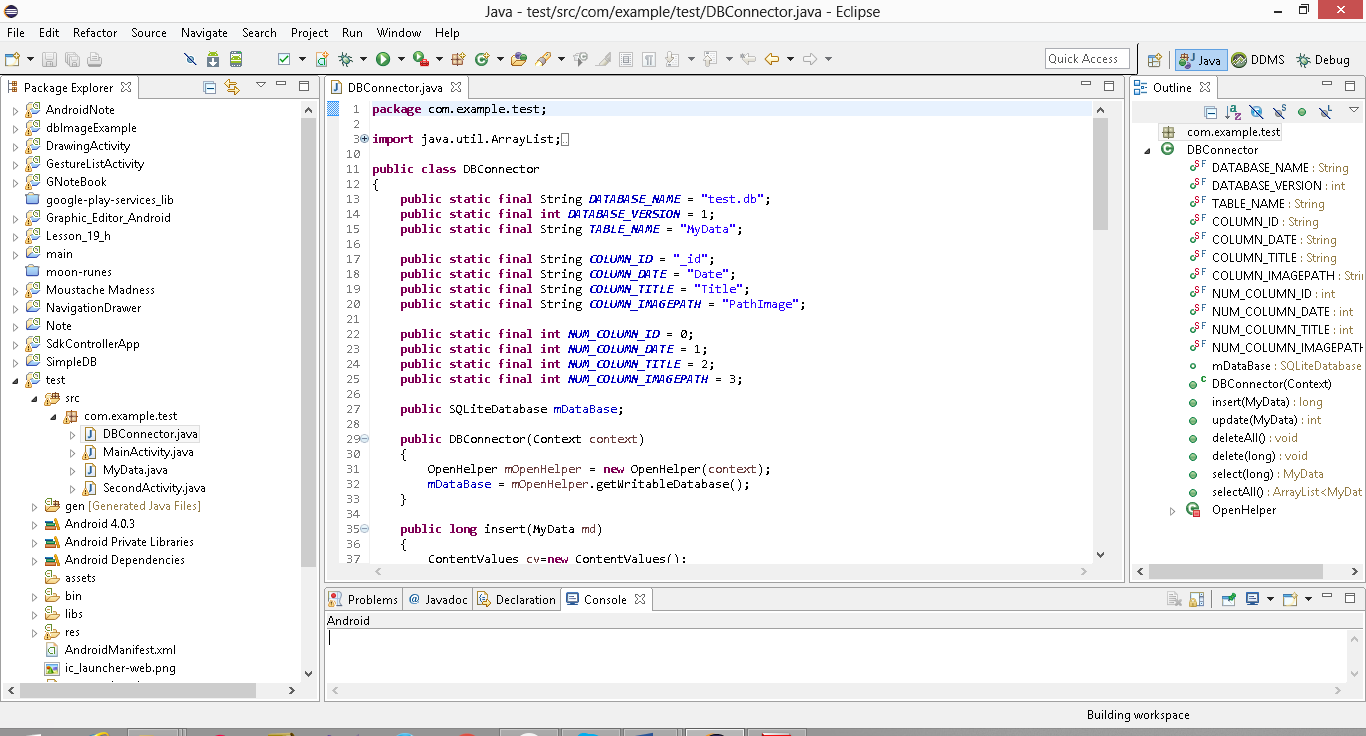


Рисунок 1.24 – Интерфейс интегрированной среды разработки Eclipse

Как показано на рисунке 1.24, интерфейс данной интегрированной среды разработки включает в себя:

1. Строку меню – главное меню платформы Eclipse с набором функций для работы с проектами (подробно рассмотрена в следующем разделе).
2. Панель инструментов – там хранятся инструменты, необходимые для разработки приложения;
3. Окно проекта и иерархии - предназначено для отображения структуры рабочего пространства проекта и представляет собой иерархическую структуру каталогов и файлов, входящих в проект;
4. Окно представлений – имеются вкладки:
5. Problems — предназначена для отображения ошибок при компиляции, а также во время написания программы;
6. Javadoc — отображение комментариев к выбранным объектам;
7. Declaration — отображение части кода, в котором происходит декларация выбранного объекта;
8. Console — системная консоль, в которую происходит вывод данных программы.
9. Перспективы (компоновки) – позволяют переключаться из одной компоновки в другую.;
10. Редактор кода – предназначен для ввода и редактирования исходного текста программ Java [1].

1.5 Математическое обеспечение решаемых задач

К математическому обеспечению решения задач относятся:

* Нахождения половины реальной ширины изображения – вычисляется по формуле 1.1:

, (1.1)

где halfwidth – половина реальной ширины изображения,

width – реальная ширина изображения.

* Нахождения половины реальной высоты изображения – вычисляется по формуле 1.2:

, (1.2)

где halfheight – половина реальной высоты изображения,

height – реальная высота изображения.

* Нахождение коэффициента уменьшения размера изображения вычисляется по алгоритму, который представлен на рисунке 1.25.



Рисунок 1.25 – Алгоритм вычисления коэффициента уменьшения изображения

Как показано на рисунке 1.25, алгоритм состоит из следующих этапов:

* Начало алгоритма;
* Определение высоты изображения (int height=options. outHeight), ширины изображения (int width=options. outWidth) и коэффициента уменьшения размера изображения (inSampleSize=1);
* Условие цикла if заключается в следующем: если ширина изображения больше требуемого изображения или высота изображения больше требуемой высоты;
* При выполнении условия if вычисляется половина высоты и ширины изображения;
* Условие цикла while заключается в следующем: пока половина высоты изображения, делённая на коэффициент уменьшения размера файла, больше требуемой высоты изображения и половина ширины изображения, делённая на коэффициент уменьшения размера файла, больше требуемой ширины изображения;
* При выполнении условия коэффициент уменьшения размера умножается на 2 и цикл повторяется заново;
* В противном случае происходит выход из цикла и определение коэффициента, который был вычислен последний раз в цикле while;
* Конец алгоритма.

1.6. Постановка задачи

Задача дипломной работы – разработать приложение ведения заметок для телефонов с ОС Android со следующей функциональностью:

* Ввод текста с помощью клавиатуры телефона;
* Ввод текста с помощью жестов;
* Добавление фотографий в заметку через галерею и камеру;
* Удаление изображений в заметке;
* Хранение заметок в базе данных;
* Изменение заметок;
* Отображение информации о приложении.

Для разработки приложения представлены следующие требования:

* Интегрированная среда разработки: Eclipse с инструментом Android SDK;
* Язык программирования: Java;
* Операционная система, для которой должно быть разработано ПО: Android версии от 4.0 до версии 4.2;

2 Разработка программного обеспечения

2.1 Архитектура разрабатываемого ПО

Изм.

Лист

№ докум.

Подпись

Дата

Лист

Разработка программного обеспечения

Архитектура приложения представлена на рисунке 2.1.

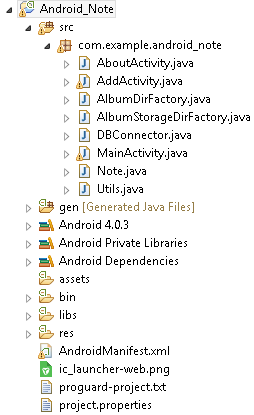


Рисунок 2.1 – Архитектура приложения

Как представлено на рисунке 2.1, приложение содержит следующие элементы:

1. Папку src – в данной папке хранятся исходные коды классов;
2. Папку gen – в данной папке хранятся java файлы, создаваемые автоматически во время разработки приложения. Файлы, находящиеся в данной папке, не должны изменяться вручную;
3. Библиотеку Android 4.0.3 – библиотека Android, на основе которой создаётся приложение;
4. Android Private Libraries – дополнительная библиотека android-support, обеспечивающая обратную совместимость с предыдущими версиями Android API;
5. assets — папка, позволяющая организовать совершенно произвольную структуру вложенных папок и файлов. Содержимое папки assets не считается ресурсами;
6. bin — каталог сборки приложения. При компиляции приложения Google Android получается сборка, размещаемая в этом каталоге;
7. Папку libs – там хранятся библиотеки;
8. res — содержит структуру папок ресурсов приложения. Содержит следующие папки:
9. res/drawable-hdpi/, res/drawable-ldpi/, res/drawable-mdpi/ - для изображений. Изображения расчитаны на соответствующие разрешение экрана мобильного устройства;
10. res/layout/ - для XML-файлов компоновки (компоновка графических элементов управления). Данная папка содержит следующие XML-файлы:

* activity-about.xml – компоновка графических элементов управления в активности с информацией о приложении;
* activity-add.xml – компоновка графических элементов управления в активности редактора заметки;
* activity-main.xml – компоновка графических элементов управления в главной активности;
* item.xml – компоновка графических элементов управления элемента в списке List View, который описан в главной активности;
* dialog\_view.xml – компоновка графических элементов управления диалога, который вызывается в активности редактора заметок;

1. res/menu/ - для XML-файлов меню. Данная папка содержит следующие XML-файлы:

* context\_menu.xml – контекстное меню, которое используется в главной активности;
* main.xml – меню, использующееся в главной активности;

1. res/values/ - для строковых, размерных ресурсов и визуальных стилей. Содержит следующие файлы:

* dimens.xml – XML-файл размерных ресурсов;
* strings.xml – XML-файл строковых ресурсов;
* styles.xml – XML-файл ресурсов стилей;

1. res/raw/ - для файлов необработанных ресурсов. В данной папке хранится файл с жестами gestures;
2. AndroidManifest.xml — XML-файл, который указывает точку входа в приложение. Служит для задания общей информации о приложении, определения атрибутов приложения, запроса прав доступа к определенным критическим операциям и другой информации;
3. Protect.properties — содержит параметры проекта, например, версию API. Файл создается автоматически и не может быть удален.

Как показано на рисунке 2.1, папка src содержит следующие файлы:

* AboutActivity.java – файл класса активности, содержащей информацию о приложении;
* AddActivity.java – файл класса активности изменения редактора заметки;
* AlbumDirFactory.java – файл класса, хранящего путь хранения альбома;
* AlbumStorageDirFactory.java – файл класса, хранящего путь хранения альбома, являющегося родителем класса AlbumDirFactory;
* DBConnector.java – файл класса, использующегося для работы с базой данных;
* MainActivity.java – файл класса главной активности;
* Note.java – файла класса, поля которого заполняют базу данных;
* Utils.java – файл класса, который используется для работы с изображениями.

Исходный код приложения представлен в приложении А.

В классе AboutActivity используются следующие методы:

* public void onCreate (Bundle savedInstanceState) – метод создания активности. В данном методе вызывается метод выбора вида активности через компоновку активности, содержащей информацию о приложении, setContentView (R.layout.activity\_about).

В классе AddActivity используются следующие методы:

* public void onCreate (Bundle savedInstanceState) – метод создания активности. В данном методе вызывается метод выбора вида активности через компоновку активности редактора setContentView (R.layout.activity\_add), метод получения данных из предыдущей активности getIntent().hasExtra("Note"). В данном методе объявляется переменная editText1 через метод findViewById (R.id.DescText), image через метод findViewById (R.id.imageView1), butAddImage через метод findViewById (R.id.AddImage), butSave через метод findViewById (R.id.Save), butCancel через метод findViewById (R.id.Cancel), butDeleteImage через метод findViewById (R.id.DeleteImage), gLib через метод GestureLibraries.fromRawResource (this, R.raw.gestures), gestures через метод findViewById (R.id.gestureOverlayView1). Переменная gestures вызывает метод, который является слушателем обработки жеста, addOnGesturePerformedListener (this) и метод удержания жеста setGestureStrokeAngleThreshold (150.0f);
* public void DeleteImageClick (View v) – метод обработки нажатия на кнопку удаления изображения. В данном методе переменная ImPath объявляется пустой (null), переменная image использует метод setImageDrawable (Drawable.createFromPath ((ImPath)));
* public void AddImageClick (View v) – метод добавления изображения в элемент Image View переменная dialog объявляется через метод new Dialog (AddActivity.this). Переменная dialog вызывает метод выбора заголовка setTitle ("Добавить фотографию через галерею или камеру?"), метод выбора вида диалога через компоновку диалога setContentView (R.layout.dialog\_view), метод показа диалога show(), метод закрытия диалога dismiss(). Переменная button1 объявляется через метод dialog.findViewById (R.id.button1), вызывает метод, который является слушателем setOnClickListener (new OnClickListener ()).Переменная mAlbumStorageDirFactory объявляется через метод new AlbumDirFactory (). Переменная Button2 объявляется через метод dialog.findViewById (R.id.button2), вызывает метод, который является слушателем setOnClickListener (new onClickListener ());
* public void CancelClick (View v) – метод обработки нажатия на кнопку отмены. В данном методе вызываются метод метод отказа возвращения данных setResult (RESULT\_CANCELED, intent), метод выхода из активности finish ();
* public void SaveClick (View v) – метод обработки нажатия на кнопку сохранения. В данном методе вызывается метод проверки пустого текстового поля editText1.getText ().toString ().equals (""), показа всплывающего сообщения Toast.makeText (getApplicationContext (), "Пустое текстовое поле", Toast.LENGTH\_LONG).show (), метода отправки данных setResult (RESULT\_OK, intent), метод выхода из активности finish (). Переменная date объявляется через метод System.currentTimeMillis (). Переменная SimpleDateFormat sdf объявляется через new SimpleDateFormat ("dd/MM/yyyy kk:mm"), переменная text объявляется через метод editText1.getText ().toString (). Переменная note объявляется через new Note (NoteID, dateString, text,ImPath). Переменная intent объявляется через метод getIntent (), вызывает метод отправки данных intent.putExtra ("Note", note);
* private String getRealPathFromURI (Uri uri) – метод получения пути, в котором находилось изображение, выведенное в элемент Image View;
* private void dispatchTakePictureIntent (int actionCode) – запуск намерения для запуска камеры. В данном методе вызывает метод startActivityForResult (takePictureIntent, actionCode). В данном методе переменная Intent takePictureIntent объявляется через метод new Intent(MediaStore.ACTION\_IMAGE\_CAPTURE), вызывает метод putExtra(MediaStore.EXTRA\_OUTPUT, Uri.fromFile(f)). Переменная f объявляется через метод setUpPhotoFile(). Переменная ImPath объявляется через метод f.getAbsolutePath();
* public void onGesturePerformed (GestureOverlayView overlay, Gesture gesture) – метод обработки жеста в виджете GestureOverlayView, является слушателем;
* protected void onActivityResult (int requestCode, int resultCode, Intent imageReturnedIntent) - метод возвращения результата;
* private File setUpPhotoFile () throws IOException – метод создания пути созданной фотографии;
* private File createImageFile () throws IOException – метод создания фотографии. В данном методе переменная timestamp объявляется через метод new SimpleDateFormat ("yyyyMMdd\_HHmmss").format (new Date ()). Переменная imageFileName объявляется строкой "IMG\_" + timeStamp + "\_". Переменная albumF объявляется через метод getAlbumDir(). Переменная imageF объявляется через метод File.createTempFile(imageFileName, JPEG\_FILE\_SUFFIX, albumF);
* private File getAlbumDir () – метод получения пути альбома, в которой должна сохраниться фотография. В данном методе переменная storageDir объявляется через метод mAlbumStorageDirFactory.getAlbumStorageDir (getAlbumName ());
* private String getAlbumName () – метод получения имени альбома. В данном методе объявлена переменная AlbumName объявлена строкой "Android\_Note";
* protected void onRestoreInstanceState (Bundle savedInstanceState) – метод восстановления состояния активности. В данном методе переменная ImPath объявляется через метод savedInstanceState.getString ("ImPath");
* protected void onSaveInstanceState (Bundle outState) – метод сохранения состояния активности. В данном методе вызывается метод outState.putString ("ImPath", ImPath);
* private void galleryAddPic () – метод добавления фотографии в галерею телефона. В данном методе вызывается метод this.sendBroadcast (mediaScanIntent). Переменная mediaScanIntent объявляется через метод new Intent ("android.intent.action.MEDIA\_SCANNER\_SCAN\_FILE"), вызывает метод mediaScanIntent.setData(contentUri). Переменная f объявляется через метод new File (ImPath). Переменная contentUri объявляется через метод Uri.fromFile (f);
* private void setPic () – метод вывода изображения в элемент Image View. В данном методе вызывается метод image.setImageBitmap (bitmap). Переменная ImageViewH объявляется через метод image.getWidth(). Переменная ImageViewH объявляется через метод image.getHeight(). Переменная bitmap объявляется через метод Utils.decodeSampledBitmapFromResource(ImPath, ImageViewW, ImageViewH);

В классе AddActivity используются следующие поля:

* GestureOverlayView gestures – объект класса GestureOverlayView, используется для рисования и обработки жестов;
* GestureLibrary gLib – объект класса GestureLibrary, хранятся все сохранённые жесты;
* Bitmap bitmap – объект класса Bitmap, используется для вывода изображения;
* String text – текст заметки;
* String dateString – дата создания/изменения заметки;
* String ImPath – путь изображения;
* Button butAddImage – объект класса Button, добавляет изображение;
* Button butSave – объект класса Button, сохраняет заметку;
* Button butCancel – объект класса Button, отменяет введение данных в заметку, выход из активности;
* Button butDeleteImage – объект класса Button, удаляет изображение;
* EditText editText1- объект класса EditText, текстовое поле;
* ImageView image – объект класса ImageView, используется для вывода изображения;
* AlbumStorageDirFactory mAlbumStorageDirFactory – объект класса, который используется для выбора пути альбома, в котором сохраняются фотографии, сделанные в камере;
* static final int ACTION\_TAKE\_PHOTO = 2 – код запроса на переход в камеру;
* static final int GALLERY\_REQUEST = 1 – код запроса на переход в галерею для выбора изображения;
* String JPEG\_FILE\_SUFFIX = ".jpg" – формат файла;

В данном классе используется интерфейс OnGesturePerformedListener. Данный интерфейс отвечает за метод OnGesturePerformed ().

Класс AlbumDirFactory наследуется от класса AlbumStorageDirFactory. В классе AlbumDirFactory используется метод получения пути альбома public File getAlbumStorageDir (String albumName) и переменная String path – путь альбома.

Класс AlbumStorageDirFactory является родителем класса AlbumDirFactory. В классе AlbumStorageDirFactory используется метод получения пути альбома public abstract File getAlbumStorageDir (String albumName).

В классе DBConnector используются следующие методы:

* public DBConnector (Context context) – конструктор класса DBConnector;
* public long insert (Note note) – метод добавления данных в таблицу Note;
* public int update (Note note) – метод изменения данных в таблице Note;
* public void deleteAll () – метод удаления всех заметок;
* public void delete (long id) – метод удаления заметки по идентификатору;
* public Note select (long id) – метод вывода заметки по идентификатору;
* public ArrayList <Note> selectAll () – метод вывода всех заметок;

В классе DBConnector используются следующие переменные:

* String DATABASE\_NAME – имя базы данных;
* DATABASE\_VERSION – версия базы данных;
* String TABLE\_NAME – имя таблицы;
* String COLUMN\_ID – название столбца идентификатора;
* String COLUMN\_DATE – название столбца даты создания/изменения заметки;
* String COLUMN\_TEXT – название столбца текста заметки;
* String COLUMN\_IMAGEPATH – название столбца пути изображения;
* int NUM\_COLUMN\_ID – номер столбца идентификатора;
* int NUM\_COLUMN\_DATE – номер столбца даты;
* int NUM\_COLUMN\_TEXT – номер столбца текста;
* int NUM\_COLUMN\_IMAGEPATH – номер столбца пути изображения;
* SQLiteDatabase dataBase – объект класса SQLiteDatabase, используется для работы с базой данных;
* OpenHelper openHelper – объект класса OpenHepler, используется для работы с базой данных;

В классе DBConnector ещё используется класс OpenHelper – класс работы с базой данных. В данном классе используются следующие методы:

* OpenHelper (Context context) – конструктор класса OpenHelper;
* public void onCreate (SQLiteDatabase db) – метод создания базы данных;
* public void onUpgrade (SQLiteDatabase db, int oldVersion, int newVersion)– метод изменения базы данных.

В классе MainActivity используются следующие методы:

* public void onCreate (Bundle savedInstanceState) – метод создания активности. В данном методе вызывается метод выбора вида активности через компоновку главной активности setContentView (R.layout.activity\_main), метод регистрации контекстного меню для объекта класса ListView registerForContextMenu (ListView). Переменная context объявляется как this. Переменная DBConnector объявляется через метод new DBConnector(this). Переменная ListView объявляется через метод findViewById (R.id.list). Переменная listAdapter объявляется через метод new myListAdapter (context, DBConnector.selectAll ()), вызывает метод setAdapter (listAdapter);
* public void onCreateContextMenu (ContextMenu menu, View v, ContextMenuInfo menuInfo) – создание контекстного меню;
* public boolean onContextItemSelected (MenuItem item) – обработка выбора элемента контекстного меню;
* public void onPause () – метод остановки активности. В данный момент очищается объект bitmap;
* public boolean onCreateOptionsMenu (Menu menu) – создание меню;
* public boolean onOptionsItemSelected (MenuItem item) – выбор элемента меню;
* protected void onActivityResult (int requestCode, int resultCode, Intent data) – метод возвращения данных из активности редактора заметок;
* private void updateList () – метод изменения списка заметок;

В классе MainActivity используются следующие поля:

* DBConnector DBConnector – объект класса DBConnector;
* Context context – объект класса Context;
* ListView ListView – объект класса ListView, там хранится список заметок;
* Bitmap bitmap – объект класса Bitmap, используется для вывода изображения и изменения его размера;
* myListAdapter listAdapter – объект класса myListAdapter;
* int ADD\_ACTIVITY = 1 – код запроса на переход в активность редактора заметок;
* int UPDATE\_ACTIVITY = 0 – код запроса на переход в активность редактора заметок;

В классе MainActivity используется ещё класс myListAdapter. В данном классе используются следующие методы:

* public ArrayList <Note> getArrayNote () – метод получения ArrayList <Note>;
* public void setArrayNote (ArrayList <Note> arrayNote) – метод выбора коллекции ArrayList <Note>;
* public int getCount () – метод получения размера коллекции ArrayList <Note>;
* public Object getItem (int position) – метод получения элемента в ArrayList <Note>;
* public long getItemId (int position) – метод получения идентификатора элемента;
* public View getView (int position, View convertView, ViewGroup parent) – метод получения вида активности;

В классе myListAdapter используются следующие переменные:

* LayoutInflater layoutInflater – объект класса LayoutInflater, используется для получения вида активности;
* ArrayList <Note> arrayNote – объект класса ArrayList <Note>, используется для вывода списка заметок;
* int mSize – размер изображения;
* Bitmap bitmap – объект класса Bitmap, используется для вывода изображения и изменения его размера;

В классе Note используются следующие методы:

* public long getID () – метод получения id заметки;
* public String getDate () – метод получения даты создания заметки;
* public String getText () – метод получения текста заметки;
* public String getImagePath () – метод получения пути изображения, которое было в заметке.

В классе Note используются следующие переменные:

* private long id – идентификатор заметки;
* private String date – дата создания/изменения заметки;
* private String text – текст заметки;
* private String imagepath – путь изображения, которое сохранилось в заметке;

В классе Utils используются следующие методы:

* public static Bitmap decodeSampledBitmapFromResource (String path, int reqWidth, int reqHeight) – считывание изображения по его пути;
* public static int calculateInSampleSize (BitmapFactory.Options options, int reqWidth, int reqHeight) – вычисление необходимого размера изображения;

Взаимодействие классов представлено в диаграмме классов (см. рисунок 2.2).

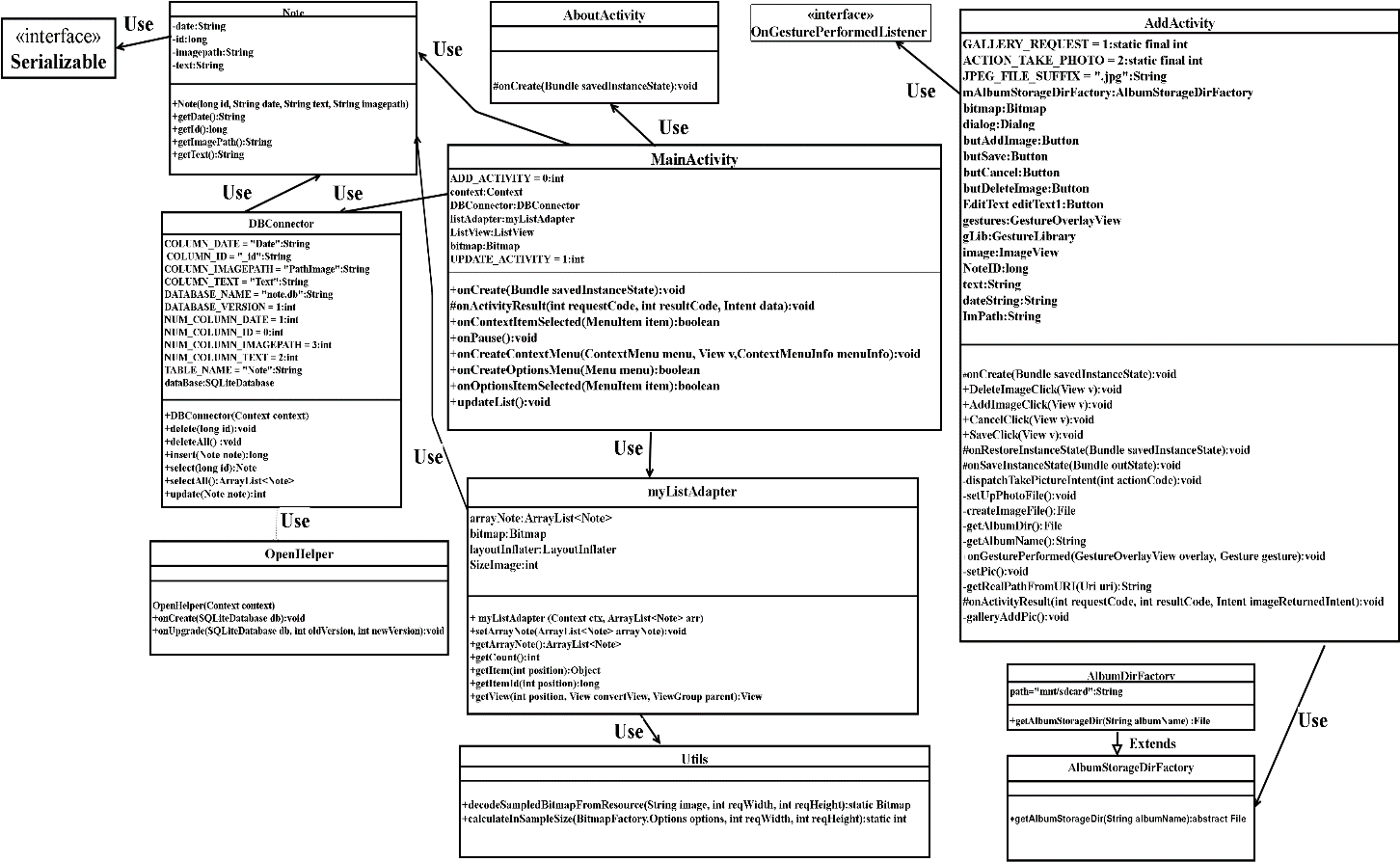


Рисунок 2.2 –Диаграмма классов

Для того, чтобы отображались заметки в приложении, используется база данных. В базе данных используется 1 таблица. База данных представлена на рисунке 2.3.



Рисунок 2.3 – База данных

Как показано на рисунке 2.3 база данных состоит из таблицы Note. Таблица состоит из следующих столбцов:

* \_id – идентификатор заметки, имеет первичный ключ, тип integer;
* Date – дата создания/изменения заметки, тип text;
* Text – текст заметки, тип text;
* ImagePath – путь изображения, тип text;

2.2 Реализация алгоритмов программы

В программе используются следующие алгоритмы:

1. Алгоритм нахождения коэффициента уменьшения размера изображения вычисляется по алгоритму, который представлен на рисунке 2.4.



Рисунок 2.4 – Алгоритм вычисления коэффициента уменьшения изображения

Как показано на рисунке 2.4, алгоритм представлен следующим образом:

* Начало алгоритма;
* Определение высоты изображения (int height=options. outHeight), ширины изображения (int width=options. outWidth) и коэффициента уменьшения размера изображения (inSampleSize=1);
* Условие цикла if заключается в следующем: если ширина изображения больше требуемого изображения или высота изображения больше требуемой высоты;
* При выполнении условия вычисляется половина высоты и ширины изображения;
* Условие цикла while заключается в следующем: пока половина высоты изображения, делённая на коэффициент уменьшения размера файла, больше требуемой высоты изображения и половина ширины изображения, делённая на коэффициент уменьшения размера файла, больше требуемой ширины изображения;
* При выполнении условия коэффициент уменьшения размера умножается на 2 и цикл повторяется заново;
* В противном случае происходит выход из цикла и определение коэффициента, который был вычислен последний раз в цикле while;
* Конец алгоритма.

1. Алгоритм определения жеста (см. рисунок 2.5):

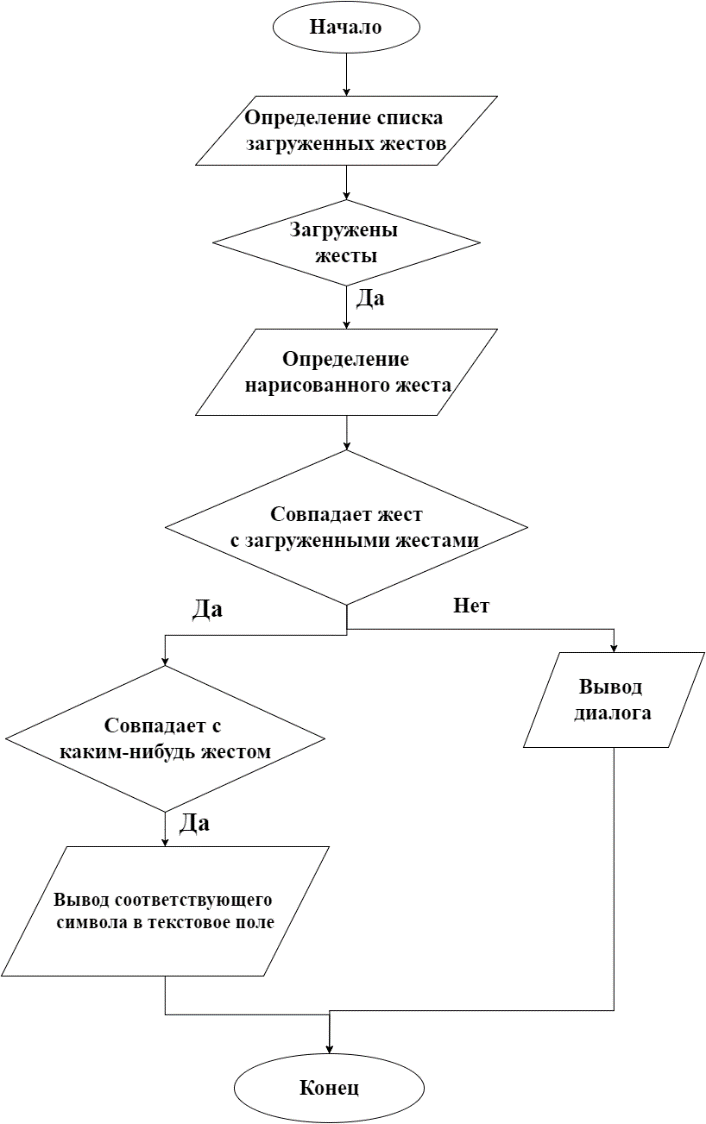
****

Рисунок 2.5 – Алгоритм определения жеста

Как показано рисунке 2.5, алгоритм представлен следующим образом:

* Начало алгоритма;
* Определение списка жестов списка жестов (ArrayList<Prediction> predictions = gLib.recognize(gesture));
* Условие цикла if заключается в следующем: если загружены жесты (if predictions.size()>0);
* При выполнении условия происходит определение нарисованного жеста (Prediction prediction = predictions.get (0));
* Условие цикла if заключается в следующем: если совпадает жест с загруженными жестами» (if prediction.score ()>3.0);
* При выполнении условия условие цикла if заключается в следующем: если нарисованный жест совпадает с каким-то жестом (if prediction.name.equals (“ ”));
* В противном случае выводится диалог;
* В случае выполнения условия цикла if выводится в текстовое поле соответствующий символ;
* Конец алгоритма.

1. Алгоритм работы всей программы (см. рисунок 2.6):



Рисунок 2.6 – Алгоритм работы всей программы.

Как показано на рисунке 2.6, представлены следующие шаги работы:

* Запуск приложения – при запуске происходит запуск главной активности;
* После запуска главной активности отображается список заметок;
* Вызов меню кнопкой вызова меню;
* Выбор элемента меню «Добавить»;
* Происходит переход в активность редактора;
* Нажатие на кнопку «Добавить изображение»;
* При нажатии на кнопку «Добавить изображение» вызывается диалог с 2 вариантами добавления изображения: галерея и камера;
* При выборе варианта «Галерея» происходит переход в галерею;
* Выбор изображения;
* После выбора изображения в галерее выводится изображение в элемент Image View;
* При выборе варианта «Камера» происходит переход в камеру;
* После снимка происходит обработка изображения;
* После обработки изображения добавляется в элемент Image View;
* Ввод текста в текстовое поле с клавиатуры;
* Рисование жестов в элементе Gesture Overlay View;
* Если жест подходит, в текстовое поле выводится соответствующий жест;
* Если жест не подходит, выводится диалог со следующим сообщением: «Неизвестный жест»;
* Нажатие на кнопку «Отмена»;
* При нажатии на кнопку «Отмена» происходит выход из активности редактора, и происходит переход в активность без сохранения заметки;
* Нажатие на кнопку «Удалить изображение»;
* При нажатии на кнопку «Удалить изображение» происходит удаление изображения;
* Нажатие на кнопку «Сохранить»;
* При нажатии на кнопку «Сохранить» происходит сохранение заметки;
* После сохранения заметки происходит переход в главную активность, и изменяется список заметок;
* Если текстовое поле в активности редактора пустое, то появляется всплывающее сообщение;
* После удерживания заметки вызывается контекстное меню. Контекстное меню состоит из 2 элементов: «Удалить» и «Редактировать»;
* Выбор элемента контекстного меню «Удалить»;
* При нажатии на элемент контекстного меню «Удалить», происходит удаление выбранной заметки;
* После удаления заметки обновляется список заметок;
* Выбор элемента контекстного меню «Редактировать»;
* При нажатии на элемент контекстного меню «Редактировать» данные выбранной заметки отправляются в активность редактора;
* Происходит переход в активность редактора
* После получения выбранной заметки выводятся текст выбранной заметки в текстовое поле и её изображение в элемент Image View;
* Выбор элемента меню «О приложении»;
* После выбора элемента меню «О приложении» происходит переход в активность «О приложении», отображение информации о приложении;
* Выход из активности «О приложении» при нажатии на телефоне кнопки Back;
* При нажатии на кнопку Back происходит переход в главную активность;
* Выбор элемента меню «Удалить всё»;
* После выбора элемента меню «Удалить всё» происходит удаление заметок;
* После удаления всех заметок отображается список заметок;
* Выбор элемента меню «Выход»;
* После выбора элемента меню «Выход» происходит завершается приложение;
* После нажатия на кнопку Back происходит завершение приложения;

2.3 Пользовательский интерфейс программы

При первой загрузке приложения появляется главная активность с пустым списком заметок. Данная активность представлена на рисунке 2.7.



Рисунок 2.7 – Главная активность с пустым списком заметок

В главной активности присутствует меню (см. рисунок 2.8).



Рисунок 2.8 – Список меню в главной активности

Активность редактора заметок представлена на рисунке 2.9.



Рисунок 2.9 – Активность редактора заметок

Как показано на рисунке 2.9, данная активность состоит из текста (Text View), кнопок (Button) и текстового поля (Edit Text). Рядом с кнопками «Добавить изображение» и «Удалить изображение» находится элемент-контейнер для хранения изображения Image View. Под текстовым полем находится элемент Gesture Overlay View, который используется для использования жестов.

В приложении имеется 2 варианта добавления изображения: через камеру и через галерею. При нажатии на кнопку «Добавить изображение» появляется диалог с вариантами добавления изображения (см. рисунок 2.10).



Рисунок 2.10 – Диалог с вариантами добавления изображения

Активность с информацией о приложении представлена на рисунке 2.11.

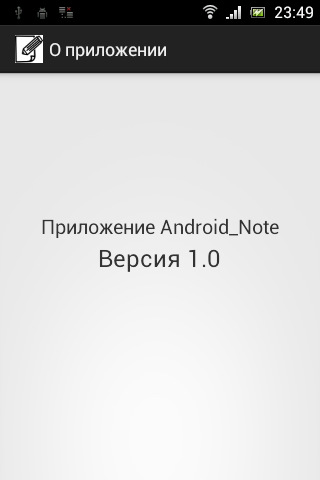


Рисунок 2.11 – Активность с информацией о приложении

Главная активность со списком заметок представлена на рисунке 2.12.

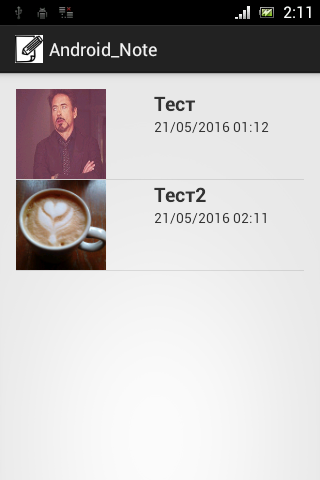


Рисунок 2.12 – Главная активность со списком заметок

Действия пользователя в главной активности представлены в диаграмме вариантов использования (см. рисунок 2.13).

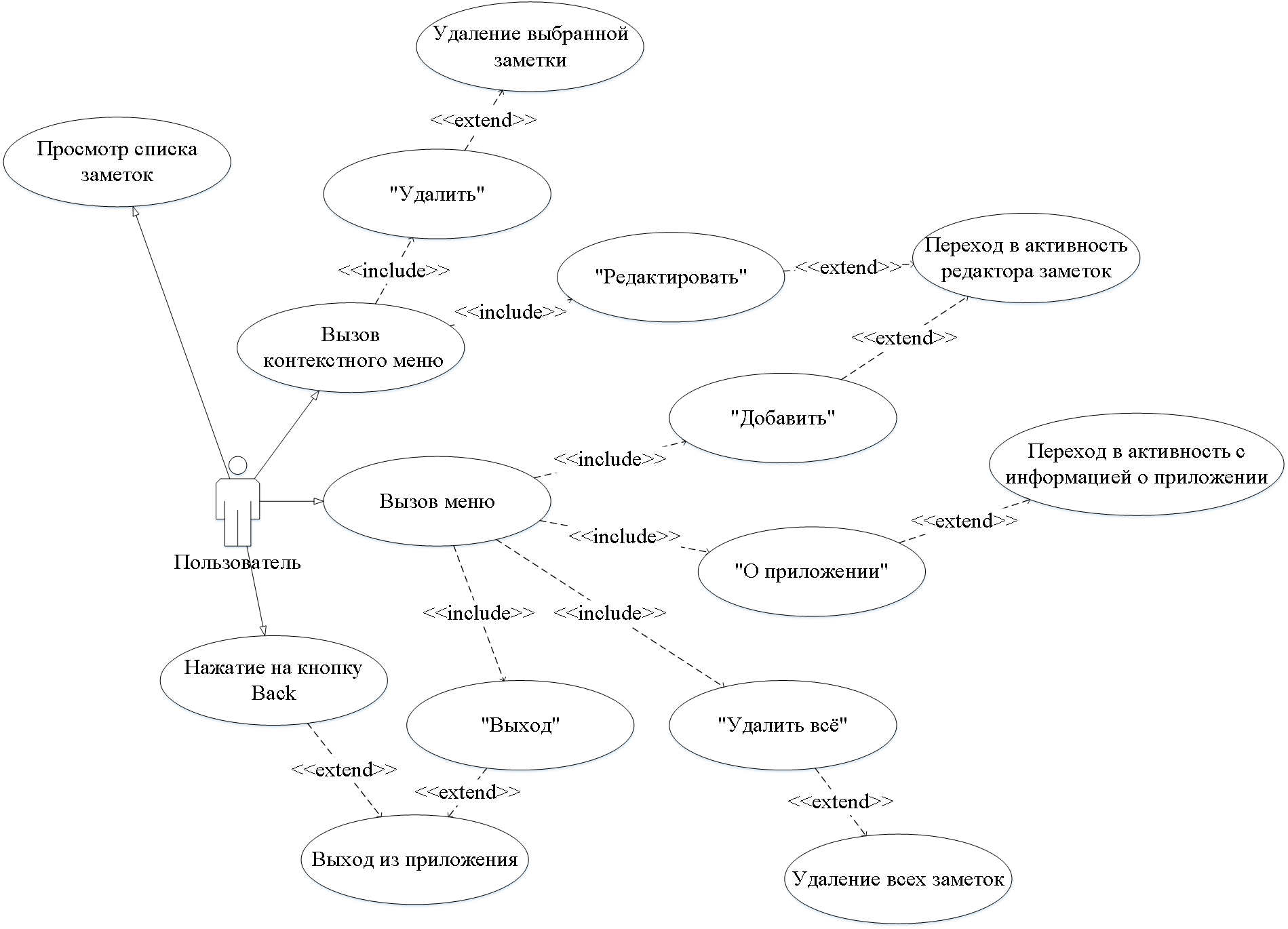


Рисунок 2.13 – Диаграмма вариантов использования в главной активности.

Как показано на рисунке 2.13, пользователь может выполнять следующие действия:

* Смотреть список заметок;
* Выходить из приложения, нажимая на телефоне кнопку Back, выбирая элемент меню «Выход»;
* Удалять все заметки, выбирая элемент меню «Удалить всё»;
* Удалять выбранную заметку, выбирая элемент контекстного меню «Удалить»;
* Переходить в активность редактора при выборе элемента меню «Добавить» и выборе элемента контекстного меню «Редактировать»;
* Переходить в активность «О приложении» при выборе элемента меню «О приложении»;

Действия пользователя в активности редактора представлены в диаграмме вариантов использования (см. рисунок 2.14).



Рисунок 2.14 – Диаграмма варианта использования в активности редактора

Как показано на рисунке 2.14, пользователь может выполнять следующие действия:

* Выходить из активности редактора заметок без сохранения заметок при нажатии на телефоне кнопки Back и нажатии на кнопку «Отмена»;
* Рисовать жест, при котором выводится соответствующий символ, если жест совпадает с загруженными жестами, диалог, если жест не совпадает с загруженными жестами;
* Добавлять изображение в элемент Image View при нажатии на кнопку «Добавить изображение» с помощью камеры и галереи;
* Удалять изображение при нажатии на кнопку «Удалить изображение»;
* Вводить текст в текстовое поле;
* Сохранять заметку при нажатии на кнопку «Сохранить»

2.4 Тестирование разрабатываемого ПО

При первой загрузке приложения появляется главная активность с пустым списком заметок. Данная активность представлена на рисунке 2.15.



Рисунок 2.15 – Приложение при первом запуске

Для создания заметки необходимо включить меню. При нажатии на телефоне на кнопку вызова меню запускается меню. Меню представлено на рисунке 2.16.



Рисунок 2.16 – Меню

Для добавления заметки необходимо выбрать элемент меню нажать. Происходит переход в активность редактора (см. рисунок 2.17).



Рисунок 2.17 – Активность редактора

Для добавления изображения необходимо нажать на кнопку «Добавить изображение». Диалог с 2 вариантами представлен на рисунке 2.18.



Рисунок 2.18– Диалог выбора варианта

Если выбрать вариант «Галерея», то произойдёт переход в галерею. Изображение, выбранное в галерее, добавилось элемент в Image View (см. рисунок 2.19).

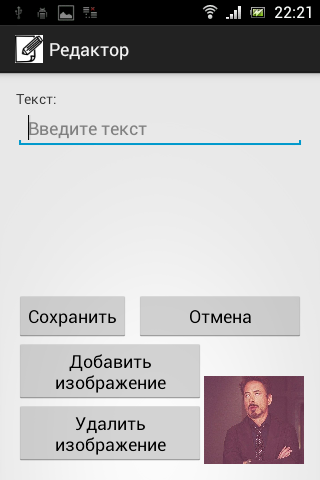


Рисунок 2.19 – Активность редактора с добавленным изображением

Необходимо ввести слово, такое как: «ТЕСТ», используя жесты. Необходимо сначала нарисовать в Gesture Overlay View букву «Т» (см. рисунок 2.20).

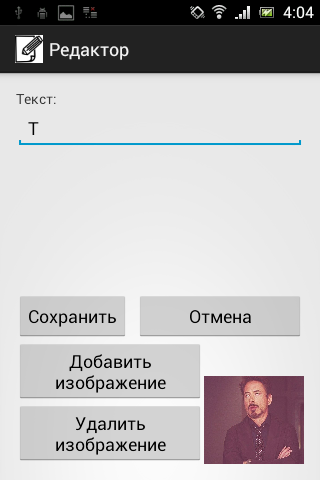


Рисунок 2.20 – Результат рисования буквы «Т»

Необходимо теперь нарисовать букву «Е». Результат представлен на рисунке 2.21.

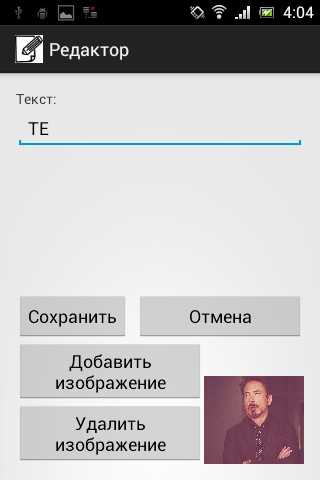


Рисунок 2.21 – Результат рисования буквы «Е»

Необходимо теперь нарисовать букву «С». Результат рисования данной буквы представлен на рисунке 2.22.

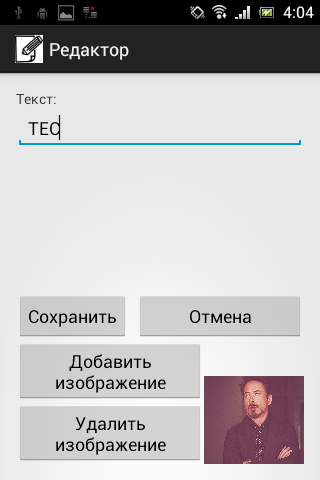


Рисунок 2.22 – Результат рисования буквы «С»

Теперь необходимо нарисовать последнюю букву «Т». Результат рисования данной буквы представлен на рисунке 2.23.

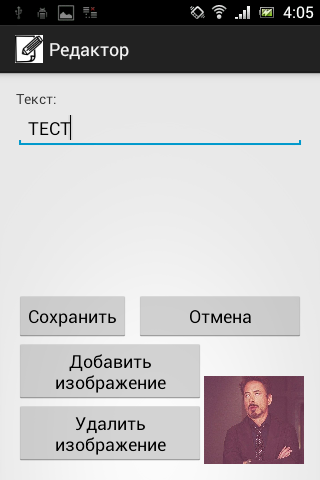


Рисунок 2.23 – Результат рисования буквы «Т»

Теперь необходимо сохранить заметку. Для сохранения необходимо нажать на кнопку «Сохранить». После нажатия на данную кнопку произойдёт переход в главную активность с созданной заметкой. Главная активность с созданной заметкой представлена на рисунке 2.24.

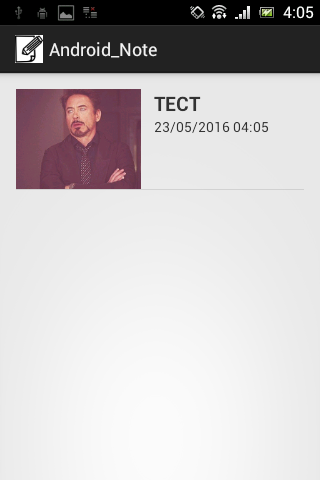


Рисунок 2.24 – Главная активность с созданной заметкой

Теперь необходимо создать новую заметку. Как добавляется изображение, было описано в процессе добавления изображения в первой заметке. Теперь в новой заметке необходимо ввести слово «Кофе». Сначала необходимо нарисовать букву «К». Результат рисования данной буквы представлен на рисунке 2.25.

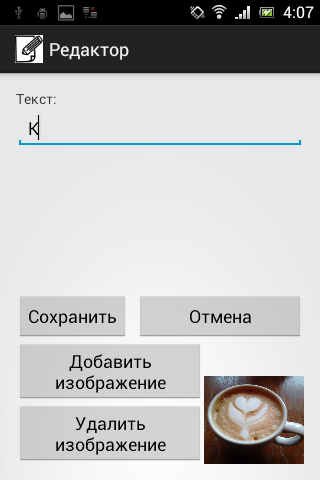


Рисунок 2.25 – Результат рисования буквы «К»

Теперь необходимо нарисовать букву «о». Результат рисования представлен на рисунке 2.26.

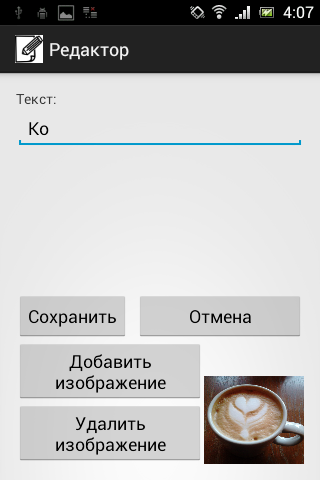


Рисунок 2.26 – Результат рисования буквы «о»

Теперь необходимо нарисовать букву «ф». Результат рисования данной буквы представлен на рисунке 2.27.

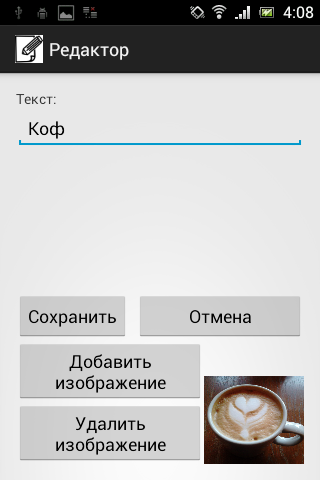


Рисунок 2.27 – Результат рисования буквы «ф»

Теперь необходимо нарисовать букву «е». Результат рисования данной буквы представлен на рисунке 2.28.

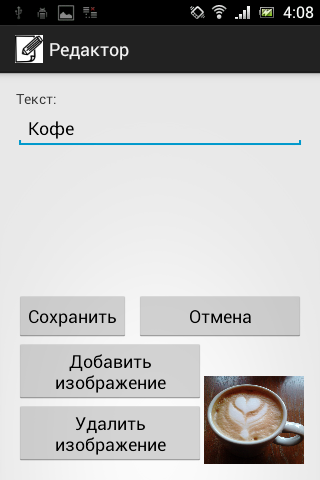


Рисунок 2.28 – Результат рисования буквы «е»

Необходимый текст набран. Процесс сохранения заметки был описан в процессе сохранения первой заметки. Список заметок представлен на рисунке 2.29.

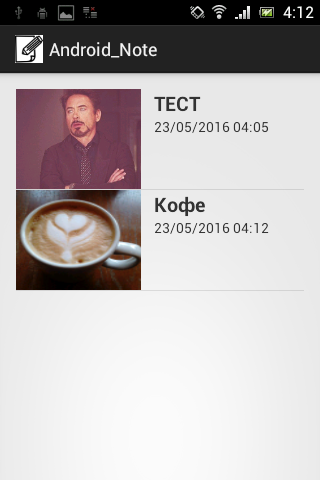


Рисунок 2.29 – Список заметок после создания новой заметки

Необходимо изменить содержимое заметки. Необходимо нажать на необходимую заметку и удерживать нажатие. После этого произойдёт вызов контекстного меню (см. рисунок 2.30).

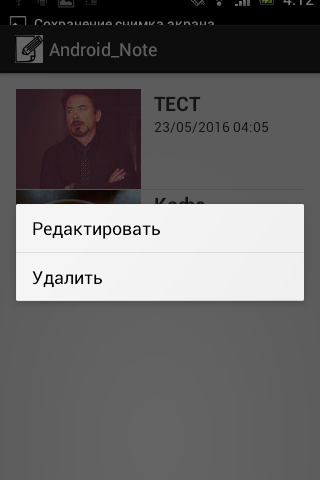


Рисунок 2.30 – Контекстное меню

После вызова данного меню необходимо нажать на элемент меню «Редактировать». После этого произойдёт отправка текста и изображения в активность редактора заметок (см. рисунок 2.31).

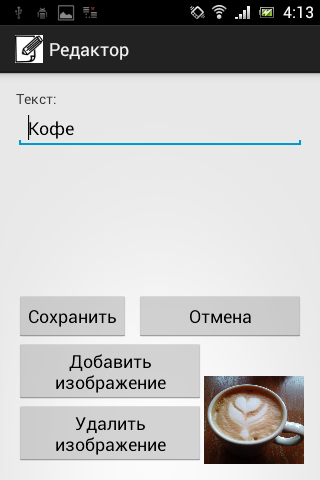


Рисунок 2.31 – Вывод текста и изображения

Необходимо поменять слово «Кофе» на слово «Мороз» и изменить фотографию. Сначала необходимо нарисовать букву «М». Результат рисования данной буквы представлен на рисунке 2.32.

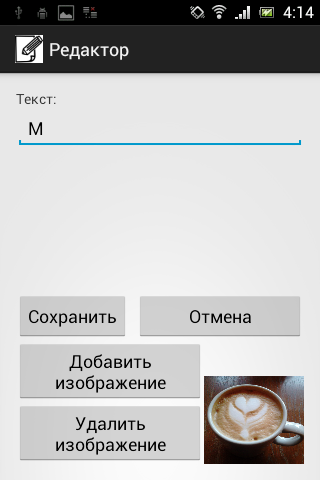


Рисунок 2.32 – Результат рисования буквы «М»

Теперь необходимо нарисовать букву «о». Результат рисования данной буквы представлен на рисунке 2.33.

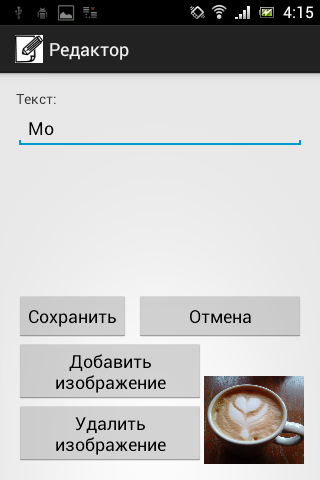


Рисунок 2.33 – Результат рисования буквы «о»

Теперь необходимо нарисовать букву «р». Результат рисования данной буквы представлен на рисунке 2.34.

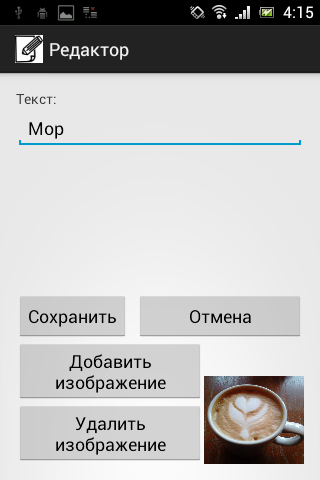


Рисунок 2.34 – Результат рисования буквы «р»

Теперь необходимо нарисовать букву «о». Результат рисования данной буквы представлен на рисунке 2.35.

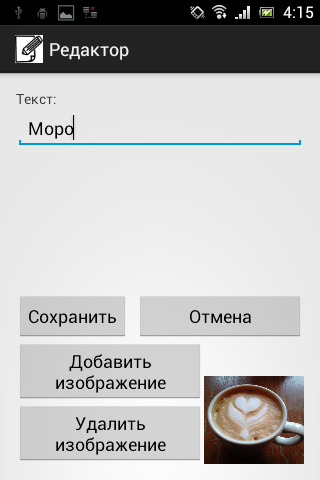


Рисунок 2.35 – Результат рисования буквы «о»

Теперь необходимо нарисовать букву «з». Результат рисования представлен на рисунке 2.36.

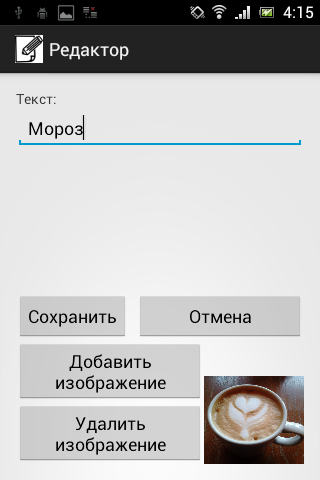


Рисунок 2.36 – Результат рисования буквы «з»

Теперь необходимо удалить изображение. Для удаления изображения необходимо нажать на кнопку удалить изображение. Результат нажатия на данную кнопку представлен на рисунке 2.37.

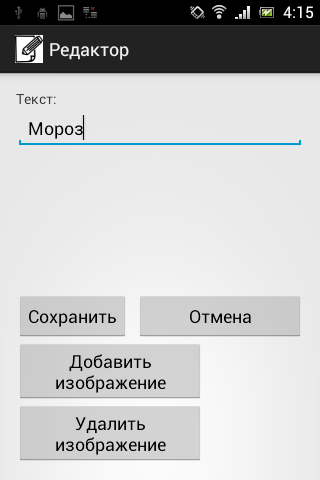


Рисунок 2.37 – Результат нажатия на кнопку «Удалить изображение»

Теперь необходимо добавить другое изображение. Теперь после этого необходимо сохранить заметку. Сохранённая заметка представлена на рисунке 2.38.

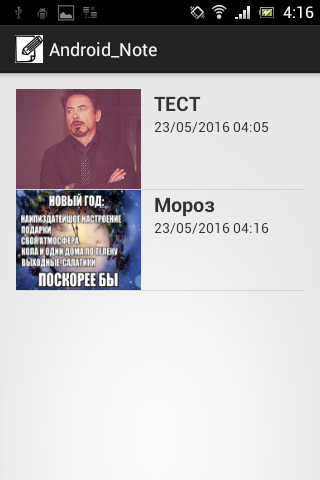


Рисунок 2.38 – Главная активность с изменённой заметкой

Если выбрать элемент меню «О приложении», то произойдёт переход в активность с информацией о приложении (см. рисунок 2.39).

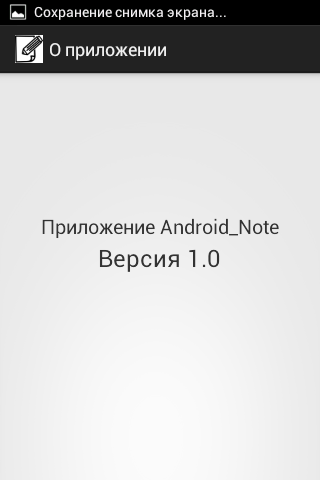


Рисунок 2.39 – Активность с содержанием о приложении

При нажатии на телефоне на кнопку Back можно перейти из данной активности обратно в главную активность.

Если выбрать элемент меню «Удалить всё», удалятся все заметки (см. рисунок 2.40).



Рисунок 2.40 – Список после удаления всех заметок

Необходимо убедиться в том, что будет, если нажать на кнопку «Сохранить» при пустом текстовом поле в активности редактора. Необходимо зайти в активность редактора (было описано, когда было создание первой заметки). Теперь необходимо нажать на кнопку «Сохранить», когда текстовое поле пустое. Результат представлен на рисунке 2.41.

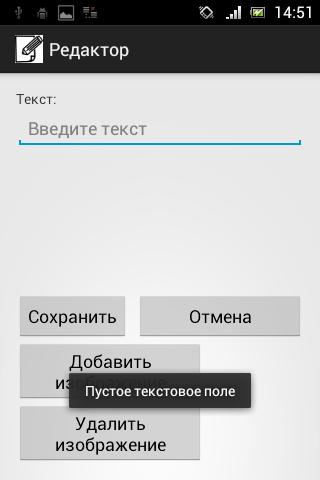


Рисунок 2.41 – Всплывающее сообщение при пустом текстовом поле

Необходимо добавить изображение через камеру. Сначала необходимо нажать на кнопку «Добавить изображение». Потом выбрать в диалоге вариант «Камера». Происходит переход в камеру (см. рисунок 2.42).

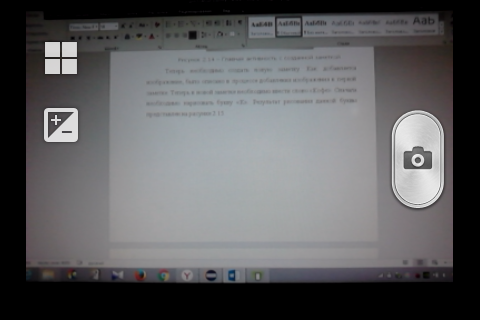


Рисунок 2.42 – Переход в камеру

После снимка фотография добавляется в элемент Image View. Результат представлен на рисунке 2.43.

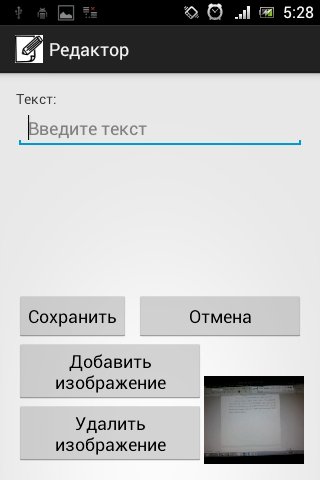


Рисунок 2.43 – Вывод созданной фотографии в элемент Image View

Необходимо ввести текст, чтобы можно сохранить заметку. После ввода текста необходимо сохранить заметку. Сохранённая заметка представлена на рисунке 2.44.

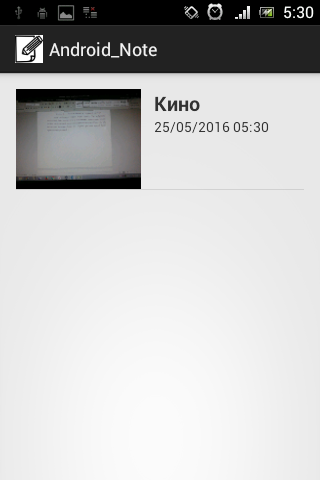


Рисунок 2.44 – Сохранённая заметка с созданной фотографией

Если в активности редактора нажать на кнопку «Отмена» или нажать на телефоне на кнопку Back, то заметка не сохранится.

3 Экономическое обоснование проекта

Изм.

Лист

№ докум.

Подпись

Дата

Лист

Экономическое обоснование проекта

При разработке программного обеспечения были учтены следующие затраты:

* Материальные затраты;
* Затраты на оборудование;
* Другие затраты;

К материальным затратам относятся диск CD-R, комплект бумаги для принтера формата А4, цветной и чёрный картридж для принтера. Расчёт данных затрат представлен в таблице 3.1.

Таблица 3.1 – Материальные затраты

|  |  |  |  |
| --- | --- | --- | --- |
| Элемент материальных затрат | Цена за шт., руб. | Количество, шт. | Общая стоимость, руб. |
| Диск CD-R | 30 | 1 | 30 |
| Комплект бумаги для принтера формата A4 | 300 | 1 | 300 |
| Чёрный картридж для принтера | 1000 | 1 | 1000 |
| Цветной картридж для принтера | 800 | 1 | 800 |
| Итого | | | 2130 |

Для разработки было использовано следующее оборудование:

* Ноутбук;
* Принтер.

Расчёт затрат на оборудование представлен в таблице 3.2.

Таблица 3.2 – Затраты на оборудование

|  |  |  |  |
| --- | --- | --- | --- |
| Оборудование | Цена за шт., руб. | Количество, шт. | Общая стоимость, руб. |
| Ноутбук | 20000 | 1 | 20000 |
| Принтер | 7000 | 1 | 7000 |
| Итого | | | 27000 |

К другим затратам относятся следующие затраты:

* Затраты на потребление электроэнергии;
* Затраты на Интернет;

Расчёт затрат на потребление электроэнергии представлено в таблице 3.3.

Таблица 3.3 – Затраты на потребление электроэнергии

|  |  |  |  |
| --- | --- | --- | --- |
| Тарифная зона | Расход, кВт/ч | Тариф | Сумма, руб. |
| День | 140 | 5.03 | 704,2 |

Расчёт затрат на Интернет представлен в таблице 3.4.

Таблица 3.4 – Затраты на Интернет

|  |  |  |  |
| --- | --- | --- | --- |
| Тип интернета | Оплата в месяц, руб. | Лимит трафика за месяц в Мб | Сумма, руб. |
| Домашний | 650 | - | 650 |

Расчёт других затрат представлена в таблице 3.5.

Таблица 3.5 - Расчёт других затрат

|  |  |
| --- | --- |
| Тип затраты | Сумма, руб. |
| Затраты на потребление электроэнергии | 704,2 |
| Затраты на Интернет | 650 |
| Общий итог | 1304,2 |

Расчёт суммы всех затрат представлен по формуле 3.1:

S=M+T+O, (3.1)

где S – сумма всех затрат,

M – материальные затраты,

O – другие затраты.

Расчёт суммы всех затрат представлен в таблице 3.6.

Таблица 3.6 - Расчёт суммы всех затрат

|  |  |
| --- | --- |
| Тип затрат | Сумма, руб. |
| Материальные затраты | 2130 |
| Затраты на оборудование | 27000 |
| Другие затраты | 1304,2 |
| Общий итог | 30354,2 |

Затрата на разработку приложения 30354,2 рубля.

ЗАКЛЮЧЕНИЕ

Изм.

Лист

№ докум.

Подпись

Дата

Лист

Заключение

Для реализации программного обеспечения были решены следующие задачи:

* анализ аналогов;
* анализ аппаратных средств;
* анализ операционной системы;
* анализ интегрированной среды разработки;
* разработка приложения ведения заметок;
* экономическое обоснование проекта.

Для разработки программного обеспечения использовалась интегрированная среда разработки Eclipse. Программное обеспечение работает на аппаратных средствах с операционной системой Android с версией от 4.0 до 4.2.

Для создания программного обеспечения использовались научно-техническая литература и материалы с сайтов, посвящённых программированию для мобильных телефонов с операционной системой Android.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Изм.

Лист

№ докум.

Подпись

Дата

Лист

Список использованных источников

1. Казарин С.А., Клишин А.П. Среда разработки Java-приложений Eclipse: (ПО для объектно-ориентированного программирования и разработки приложений на языке Java): Учебное пособие. Москва 2008. — 77 с.
2. http://startandroid.ru/ru/uroki/vse-uroki-spiskom/376-urok-161-risovanie-bitmap-memory-kesh-picasso.html
3. https://developer.android.com/training/camera/photobasics.html?hl=ru
4. http://developer.alexanderklimov.ru/android/theory/dialog.php
5. http://ru-code-android.livejournal.com/3923.html
6. http://sergey-oganesyan.ru/interesnaya-rubrika/os-android-osobennosti-preimushestva.html
7. https://play.google.com/store/apps/details?id=com.mervsy.notes&hl=ru
8. https://play.google.com/store/apps/details?id=com.ravzasoft.notepad&hl=ru
9. https://play.google.com/store/apps/details?id=com.superpad.android&hl=ru

ПРИЛОЖЕНИЕ А. Исходный код

Изм.

Лист

№ докум.

Подпись

Дата

Лист

Приложение А. Исходный код

package com.example.android\_note;

import java.util.ArrayList;

import android.app.Activity;

import android.content.Context;

import android.content.Intent;

import android.database.sqlite.SQLiteDatabase;

import android.graphics.Bitmap;

import android.view.ContextMenu;

import android.view.ContextMenu.ContextMenuInfo;

import android.view.LayoutInflater;

import android.view.Menu;

import android.view.MenuInflater;

import android.view.MenuItem;

import android.view.View;

import android.view.ViewGroup;

import android.os.Bundle;

import android.widget.AdapterView.AdapterContextMenuInfo;

import android.widget.BaseAdapter;

import android.widget.ImageView;

import android.widget.ListView;

import android.widget.TextView;

public class MainActivity extends Activity

{

int ADD\_ACTIVITY = 0;

Context context;

DBConnector DBConnector;

myListAdapter listAdapter;

ListView ListView;

Bitmap bitmap;

int UPDATE\_ACTIVITY = 1;

@Override

public void onCreate(Bundle savedInstanceState)

{

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

context = this;

DBConnector = new DBConnector(this);

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

listAdapter = new myListAdapter(context, DBConnector.selectAll());

ListView.setAdapter(listAdapter);

registerForContextMenu(ListView);

}

@Override

protected void onActivityResult(int requestCode, int resultCode, Intent data)

{

if (resultCode == Activity.RESULT\_OK)

{

Note note = (Note) data.getExtras().getSerializable("Note");

if (requestCode == UPDATE\_ACTIVITY)

DBConnector.update(note);

else

DBConnector.insert(note);

updateList();

}

}

@Override

public boolean onContextItemSelected(MenuItem item)

{

AdapterContextMenuInfo info = (AdapterContextMenuInfo) item.getMenuInfo();

switch(item.getItemId())

{

case R.id.edit:

Intent i = new Intent(context, AddActivity.class);

Note note = DBConnector.select(info.id);

i.putExtra("Note", note);

startActivityForResult (i, UPDATE\_ACTIVITY);

updateList();

return true;

case R.id.delete:

DBConnector.delete (info.id);

updateList();

return true;

default:

return super.onContextItemSelected(item);

}

}

@Override

public void onPause()

{

super.onPause();

if (bitmap != null)

{

bitmap.recycle();

bitmap = null;

}

}

@Override

public void onCreateContextMenu(ContextMenu menu, View v,ContextMenuInfo menuInfo)

{

super.onCreateContextMenu(menu, v, menuInfo);

MenuInflater inflater = getMenuInflater();

inflater.inflate(R.menu.context\_menu, menu);

}

@Override

public boolean onCreateOptionsMenu(Menu menu)

{

MenuInflater inflater = getMenuInflater();

inflater.inflate(R.menu.main, menu);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item)

{

switch (item.getItemId())

{

case R.id.add:

Intent add = new Intent(context, AddActivity.class);

startActivityForResult (add, ADD\_ACTIVITY);

updateList();

return true;

case R.id.deleteAll:

DBConnector.deleteAll();

updateList();

return true;

case R.id.exit:

finish();

return true;

case R.id.about:

Intent about=new Intent(context, AboutActivity.class);

startActivity(about);

return true;

default:

return super.onOptionsItemSelected(item);

}

}

private void updateList ()

{

listAdapter.setArrayNote(DBConnector.selectAll());

listAdapter.notifyDataSetChanged();

}

class myListAdapter extends BaseAdapter

{

ArrayList<Note> arrayNote;

Bitmap bitmap;

LayoutInflater layoutInflater;

int SizeImage;

public myListAdapter (Context ctx, ArrayList<Note> arr)

{

layoutInflater = LayoutInflater.from(ctx);

SizeImage = ctx.getResources().getDimensionPixelSize(R.dimen.dimen);

setArrayNote(arr);

}

public void setArrayNote(ArrayList<Note> arrayNote)

{

this.arrayNote = arrayNote;

}

public ArrayList<Note> getArrayNote()

{

return arrayNote;

}

public int getCount ()

{

return arrayNote.size();

}

public Object getItem (int position)

{

return position;

}

public long getItemId (int position)

{

Note note = arrayNote.get(position);

if (note != null)

{

return note.getID();

}

return 0;

}

public View getView(int position, View convertView, ViewGroup parent)

{

if (convertView == null)

convertView = layoutInflater.inflate(R.layout.item, null);

ImageView ivImage = (ImageView)convertView.findViewById(R.id.Image);

TextView tvText = (TextView)convertView.findViewById(R.id.Text);

TextView tvDate = (TextView)convertView.findViewById(R.id.Date);

Note note = arrayNote.get(position);

tvDate.setText(note.getDate());

tvText.setText(note.getText());

String image = note.getImagePath();

bitmap = Utils.decodeSampledBitmapFromResource(image, SizeImage, SizeImage);

ivImage.setImageBitmap(bitmap);

return convertView;

}

}

}

Дипломный проект выполнен мной совершенно мной самостоятельно, использованные в работе материалы и концепции из опубликованной научной литературы и других источников имеют другие ссылки.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| *(Ф.И.О. )* |  | *подпись* |

|  |  |
| --- | --- |
| дата |  |
|  |  |