

Министерство образования и науки Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Санкт-Петербургский государственный технологический институт  
(технический университет)»

Кафедра систем автоматизированного проектирования и управления

Петров Д.Н.

**Разработка мобильных приложений  
для ОС Android**

**П Р А К Т И К У М**

Задания, рекомендации, примеры

Санкт-Петербург  
2023

## Содержание

Аннотация .....	3
Лабораторная работа 1 .....	5
Лабораторная работа 2 .....	9
Лабораторная работа 3 .....	10
Лабораторная работа 4 .....	15
Лабораторная работа 5 .....	18
Приложение А Варианты для выполнения лабораторных работ .....	22
Приложение Б Примеры мобильных приложений .....	26

## Аннотация

Практикум предназначен для закрепления обучающимися УГНС 09.00.00 навыков разработки приложений для мобильных устройств, работающих под управлением операционной системы Android.

Практикум ориентирован на начинающий уровень разработчика Android-приложений и состоит из пяти лабораторных работ. Лабораторные работы выполняются по вариантам в подгруппах (бригадах) до трех обучающихся преимущественно в компьютерном классе кафедры учебного заведения. Результатом выполнения каждой лабораторной работы является мобильное приложение и отчет, оформленный в соответствии с правилами оформления и структурой. В отчетах каждый этап работы сопровождается подробным текстовым описанием выполнения, листингами кода разметки макета интерфейсов и дисплейными фрагментами (снимками экрана).

Для выполнения лабораторных работ требуется ЭВМ с поддержкой аппаратной виртуализации (VTx), многоядерным ЦПУ, ОЗУ объемом не менее 8 Гб. Лабораторные работы выполняются в среде программирования Android Studio версии не ниже 2021.X.X (Dolphin), рекомендуется использовать версию 2022.X.X (Flamingo, Giraffe) или более новую. При создании нового проекта приложения ориентироваться на SDK версии не ниже 29. Для отладки мобильных приложений также рекомендуется использовать физическое устройство (смартфон), работающий под управлением ОС Android версии не ниже 10.

При разработке графического интерфейса приложения учитывать правила эргономики, в том числе адаптацию под формат и ориентацию экрана, под светлую и темную тему. Рекомендуется для каждого разработанного приложения определять правильное (понятное) системное наименование, например, com.g470\_1.Notes, где 470 – номер группы, 1 – номер подгруппы, Notes – часть названия, соответствующая заданию. Также, рекомендуется для каждого приложения установить логотип в виде векторного изображения. При разработке управляющей логики обязательно использовать проверки заполнения / формата / диапазона варьирования входных данных и вывод сообщения «Toast» об ошибке ввода. Все текстовые константы, цвета, шрифты размещать в файлах ресурсов, а не в коде. Из кода разметки или управляющего кода на строку (цвет, шрифт) делать ссылку.

Тестирование мобильных приложений выполнять на виртуальном устройстве, при наличии технической возможности в режиме отладки мобильное приложение тестируется на физическом устройстве с фиксацией результатов тестирования снимками дисплея (скриншотами). Требуется выполнение тест-кейсов на проверку заполнения входных данных, корректности их формата и вхождения в допустимый диапазон, а также тест-кейсов с различным сочетанием заведомо правильных входных данных.

В рамках первой лабораторной работы обучающиеся знакомятся со средой программирования Android Studio, основными терминами, инструментами разработчика, базовыми операциями языка программирования Java, архитектурой проекта мобильного приложения для ОС Android. Результатом выполнения первой лабораторной работы является простейшее мобильное приложение для вычисления параметров геометрического, физического или химического объекта или решения базовой задачи линейного программирования.

Вторая лабораторная работа направлена на практическое освоение механизмов взаимодействия пользователя с аппаратным обеспечением мобильного устройства (датчиками, модулем связи, подсистемой геопозиционирования и т.д.). В итоге, обучаемые предоставляют к защите мобильное приложение с функцией доступа к встроенному в мобильное устройство аппаратному модулю (датчику, фонарику, модулю GPS и т.д.) с целью управления или получения технических сведений о нем.

В третьей лабораторной работе обучаемым необходимо продемонстрировать практические навыки работы с локальной базой данных SQLite для хранения предметных сведений об объекте (в соответствии с вариантом) и управления ими посредством графического интерфейса. С хранимыми данными требуется выполнить ряд базовых операций: создание, отображение, редактирование, удаление. Кроме того, при выполнении третьей лабораторной работы более подробно изучается графический компонент RecyclerView, часто применяемый в больших проектах для отображения списка разнотипных данных.

1-3 лабораторные работы направлены на изучение механизмов разработки локального мобильного приложения одноуровневой или файл-серверной архитектуры. Четвертая лабораторная работа состоит в изучении компонента WebView, использовании клиент-серверного взаимодействия при переносе «рутинных» вычислительных операций с мобильного устройства на сервер приложений. По сути, работа заключается в создании адаптивной веб-страницы с доступом к простейшему сервису массового обслуживания, отображаемой в компоненте WebView мобильного приложения. Здесь же, обучаемые демонстрируют навыки программирования веб-сервиса на скриптовом языке Php, веб-верстки на языке HTML, навыки работы с каскадными стилями CSS3 и применения библиотеки JQuery для асинхронных посылок.

В пятой лабораторной работе с использованием ранее разработанного веб-сервиса при его дополнении интерфейсом приема HTTP-запроса (GET), содержащего входные данные, и функцией возврата результата в формате JSON или XML требуется разработать мобильное приложение – клиент с графическим интерфейсом, содержащим инструменты ввода и отправки на обработку на сервер приложений входных данных и отображения полученного ответа в эргономичном виде.

## Лабораторная работа 1

Тема: Интерфейс интегрированной среды разработки Android Studio.  
Создание мобильного приложения для простых математических вычислений.

Задачи:

- 1) Создание проекта мобильного приложения:
  - выбор шаблона;
  - указание наименования приложения и выбор версии SDK;
  - выбор языка программирования.
- 2) Ознакомление со структурой проекта, инструментами разработчика и элементами интерфейса Android Studio.
- 3) Настройка проекта
- 4) В соответствии с вариантом задания:
  - создание графического интерфейса;
  - создание управляющей логики;
  - тестирование мобильного приложения.

### 1) Создание проекта мобильного приложения

Выполнение действия в каждом интерфейсе / на форме сопровождать дисплейным фрагментом с номером, наименованием и описанием.

Откройте Android Studio, выберите New Project (рисунок 1).

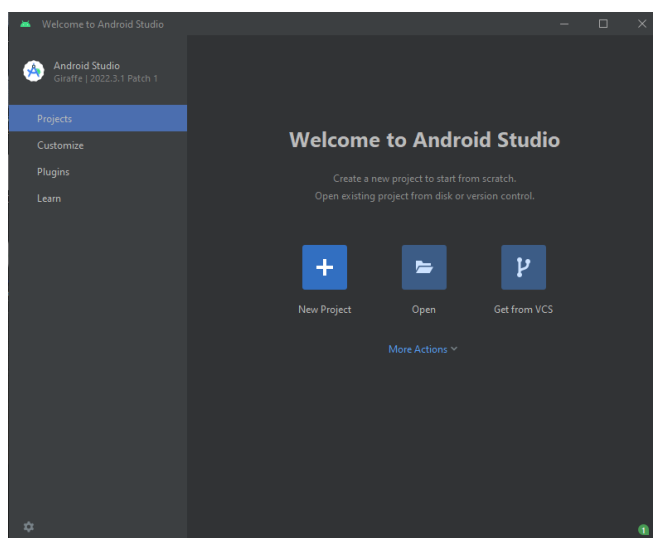


Рисунок 1 – Создание нового проекта

Создать новый проект также возможно из рабочего интерфейса по команде File->New->New Project (рисунок 2)

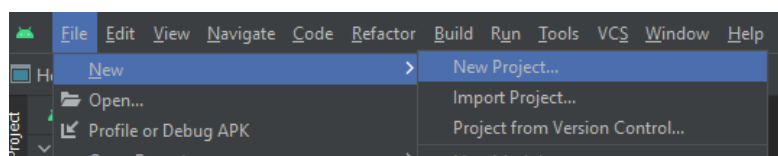


Рисунок 2 – Создание нового проекта

На появившейся форме требуется выбрать тип целевого мобильного устройства (Phone and Tablet) и шаблон приложения (Empty Activity до версии Android Studio Giraffe, а начиная с Giraffe, Empty Views Activity, т.к. требуется поддержка языка Java) (рисунок 3).

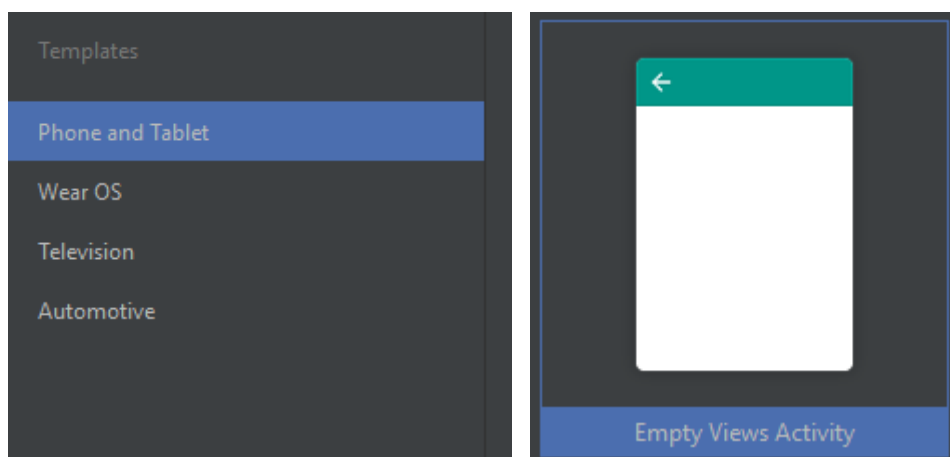


Рисунок 3 – Выбор типа целевого устройства и шаблона приложения

Далее, следует указать наименование приложения, пакета, выбрать директорию для сохранения файлов проекта и выбрать минимальную версию SDK, от которой зависит поддержка приложения операционными системами (рисунок 4).

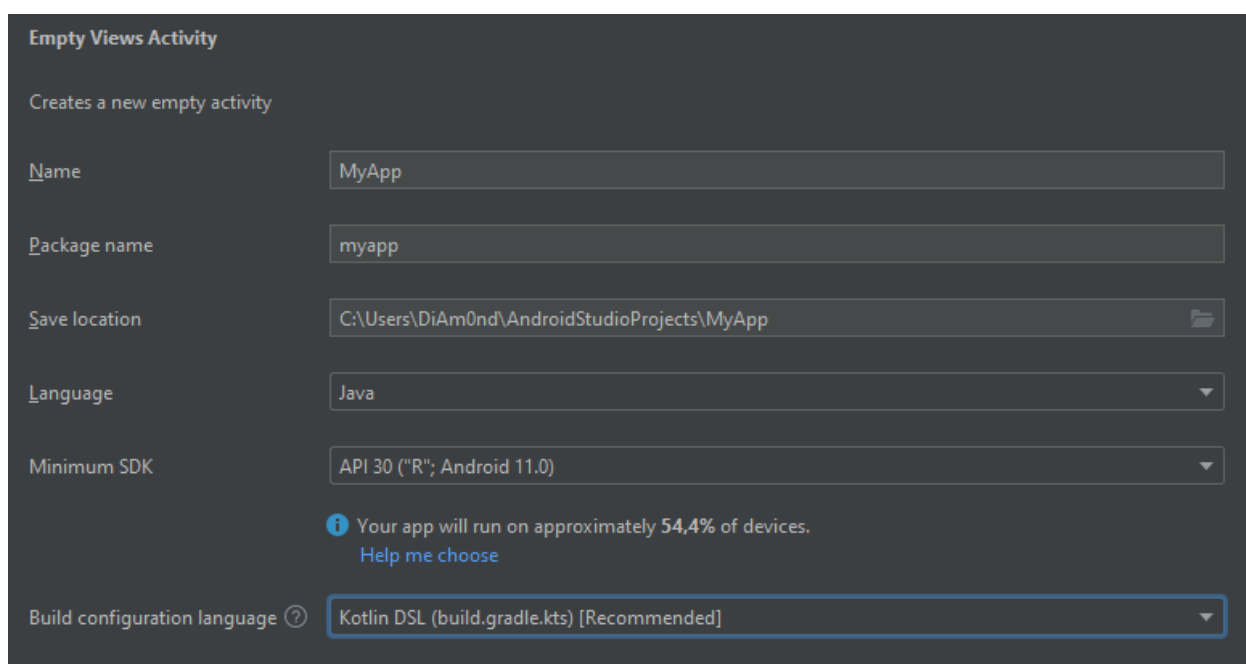


Рисунок 4 – Настройка нового приложения

В качестве наименования пакета рекомендуется указывать свое в соответствии с решаемой прикладной задачей, слово «example» из Package name рекомендуется убрать.

Если проект создается в компьютерном классе кафедры, на рабочем столе ЭВМ создать каталог с названием приложения, в качестве Save location следует указывать рабочий стол и созданный на нем каталог, совпадающий по имени с названием приложения (Name).

Если проект создается на домашнем ПК, в качестве Save location можно указать любой удобный каталог, доступный для записи / чтения / исполнения файлов. По умолчанию Android Studio создает проекты в личном каталоге пользователя: C:\Users\Имя\_пользователя\AndroidStudioProjects.

В поле со списком языков программирования нужно выбрать язык программирования **Java**. В качестве минимальной версии Android SDK, соответствующей версии ОС Android, поддерживающей ваше будущее приложение, выбирается API 30 или выше.

2) Ознакомление со структурой проекта, инструментами разработчика и элементами интерфейса Android Studio.

Перечислить основные каталоги проекта приложения с описанием их назначения, описать разделы структуры проекта File->Project Structure, вкладки Project, Modules и Dependencies. Показать режимы разработки Code, Split, Design. Описать назначение Device Manager и SDK Manager, инструменты для отладки: LogCat, Problems, Terminal. Описание вышеперечисленного в п. 2 сопровождать дисплейными фрагментами.

### 3) Настройка проекта

Установить в проект подходящее под версию SDK виртуальное устройство. Рекомендуется выбирать устройство среди Pixel и Nexus, ОС Android не менее 11.

В Dependencies выбрать версию appcompat, подходящую под выбранную минимальную версию SDK и ОС Android. Если минимальная версия SDK = 30, ОС Android версии 11, то рекомендуется выбрать версию appcompat 1.5.1.

После завершения настройки проекта, запустить его отладку на выбранном виртуальном устройстве. При наличии технической возможности запустить отладку проекта на физическом устройстве, предварительно включив на нем режим отладки. Если приложение запустилось, можно переходить к п. 4, если возникли проблемы с запуском, проверьте соответствие версий SDK, appcompat и выбранной ОС для виртуального устройства / версию ОС на физическом устройстве.

### 4) Выполнение задания по варианту

#### 4.1 Создание графического интерфейса

Показать разработанную структуру макета activity.main в режиме дизайна. Перечислить используемые графические компоненты с их назначенными идентификаторами в главной активности, обосновать их выбор.

Для цветов, картинок, шрифтов, строковых констант использовать соответствующие файлы ресурсов.

Привести листинг файлов ресурсов colors.xml, strings.xml, векторных изображений и т.д, xml-разметки главной активности. Листинги разметок и исходных кодов оформлять моноширинным шрифтом, например, Courier New, без абзацного отступа, размер шрифта на 1-2 размера меньше основного текста, выравнивание текста по левому краю.

#### 4.2 Создание управляющей логики

Оформить требуемые расчетные операции в виде отдельной функции, привести ее описание (аргументы, их тип данных и диапазон варьирования), возвращаемое значение. Если требуется разработка нескольких функций, обосновать их создание. При разработке управляющей логики обязательно использовать проверки заполнения / формата / диапазона варьирования входных данных и вывод сообщения «Toast» об ошибке ввода.

В программном коде главные вычислительные операции комментировать.

Привести листинг программного кода.

#### 4.3 Тестирование мобильного приложения

Запустить готовое мобильное приложение на виртуальном устройстве. При наличии технической возможности, запустить приложение на физическом устройстве. Выполнить несколько тестов на проверку исключительных ситуаций (ошибки ввода), тесты с заведомо корректными данными, привести дисплейные фрагменты интерфейса для каждого выполненного теста.

Выводы сформулировать в соответствии с выполненными задачами 1-4. В выводах подтвердить успешность тестирования мобильного приложения, указать использованные при выполнении лабораторной работы программные (системное, инструментальное ПО с версиями) и аппаратные (характеристики ЭВМ разработчика, марка и модель физического мобильного устройства, при использовании) средства. Возможно указание прочих комментариев / примечаний к выполненной работе.



## Лабораторная работа 2

Тема: Аппаратное обеспечение мобильного устройства. Создание приложения для управления периферией и сбора сведений от внутренних аппаратных компонентов.

Задачи:

- 1) Изучение и описание аппаратного модуля (в соответствии с вариантом)
- 2) Разработка мобильного приложения:
  - создание и настройка нового проекта приложения;
  - разметка макета графического интерфейса;
  - создание управляющей логики;
  - тестирование мобильного приложения.

### 1) Изучение и описание аппаратного модуля

На данном этапе выполнения лабораторной работы требуется выполнить обзор аппаратного модуля (по варианту), указать его назначение, описать принцип работы, формат обращения к модулю, структуру возвращаемых данных.

### 2) Разработка мобильного приложения

#### 2.1 Создать и настроить новый проект приложения.

2.2 Создать разметку графического интерфейса главной активности. Если вариант требует работы с технологическим датчиком, рекомендуется создать дополнительную активность для вывода полного списка подключенных к мобильному устройству датчиков с их параметрами. В ином случае дополнительную активность создавать не требуется. Рекомендуется добавить кнопку Button с контекстным логотипом (в соответствии с выполняемым действием), а полученные от устройства сведения отображать в текстовом поле TextView, что и предусматривается большинством вариантов.

2.3 Разработать программный интерфейс (управляющую логику) для решения задачи опроса технологического датчика / управления модулем / получения сведений о модуле. Рекомендуется создать отдельную функцию – слушатель (для датчика) для отображения текущих значений, а не тех, которые были получены при обращении.

2.4 Тестировать опрос технологического датчика / управление модулем устройства или получение сведений о модуле устройства или статистических сведений о работе устройства. Входные данные при этом не вводятся. Возможно представление только одного тест-кейса.

Выводы сформулировать в соответствии с выполненными задачами. В выводах подтвердить успешность тестирования мобильного приложения, указать использованные при выполнении лабораторной работы программные (системное, инструментальное ПО с версиями) и аппаратные (характеристики ЭВМ разработчика, марка и модель физического мобильного устройства, при использовании) средства. Указать возможные варианты применения аппаратного компонента мобильного устройства.

## Лабораторная работа 3

Тема: Управление данными локального хранилища мобильного устройства на основе интегрированной СУБД SQLite и графического компонента RecyclerView.

Задачи:

1) Изучение специфики языка запросов и типов данных SQLite, механизма подключения и использования интегрированной СУБД SQLite для управления данными в памяти мобильного устройства. Ознакомление с принципами настройки адаптера данных, связывания данных, особенностями доступа к данным с использованием объекта cursor.

2) Разработка мобильного приложения:

- создание и настройка нового проекта приложения;
- работа с ресурсами приложения: строковыми константами, цветами, шрифтами, стилями, пиктограммами, установка иконки приложения;
- разметка макета графического интерфейса для двух активностей и элемента списка;
- создание управляющей логики (класса для работы с SQLite, адаптера данных, кода главной активности, кода активности для создания/редактирования элемента списка);
- тестирование мобильного приложения.

1) SQLite – набор интегрированных библиотек для работы со структурированным локальным хранилищем данных. SQLite называют системой управления базами данных (СУБД) для локальных баз данных. SQLite входит в Android SDK и может быть использована для организации локального хранилища данных. Как СУБД SQLite поддерживает основные конструкции языка запросов SQL. Но «чистым» SQL разработчики в Android Studio пользуются редко. Для создания и выполнения SQL-запросов используется класс-помощник SQLiteOpenHelper или наследованный от него класс.

SQLite поддерживает языковые группы DDL, DQL, DML.

Пример DDL (создание таблицы): **CREATE TABLE** tblNotes (id **INTEGER PRIMARY KEY AUTOINCREMENT**, note\_title **TEXT**, note\_text **TEXT**, note\_prior **INTEGER**);

Пример DQL (оператор выборки данных): **SELECT** id, note\_title, note\_text, note\_prior **FROM** tblNotes **ORDER BY** note\_prior;

Пример DML (оператор удаления): **DELETE FROM** tblNotes **WHERE** id=1;

Как видно из примеров выше, SQL, используемый для SQLite имеет сильное сходство с SQL, используемым для серверных СУБД, таких, как MS SQL Server, MySQL, PostgreSQL.

Основные типы данных SQLite:

**INTEGER** и **NUMERIC** – для целых чисел;

**REAL** – для десятичных чисел;

**TEXT** – для текстовых данных;

**NONE** – не определенный тип данных.

Для сопряжения внутренних списков данных с данными из БД, а также этих списков с виджетами для визуализации данных используют адаптеры.

Например, виджет RecyclerView mainList, как и ListView, имеют собственный адаптер, который можно переназначать на свой (кастомный): mainList.setAdapter(customAdapter). customAdapter настраивается таким образом, чтобы виджеты элемента списка mainList были сопряжены прямо или косвенно с каким-либо внутренним списком ArrayList<string>.

Связывание (bind) осуществляется для каждого элемента внутреннего списка и элемента виджета через «держатель» представления (или макета) элемента виджета, например:

holder.note\_id\_txt.setText(String.valueOf(note\_id.get(position))), где position – позиция элемента внутреннего списка. Получается, что адаптер автоматически создает ровно столько элементов в виджете списка, сколько элементов во внутреннем списке.

Создание «держателя» представления выполняется в методе:

```
public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType)
```

Связывание данных внутренних списков с виджетами, а также назначение действий (обработчиков) для виджетов (например, нажатие на виджет) происходит в методе:

```
public void onBindViewHolder(@NonNull final MyViewHolder holder, final int position)
```

Для загрузки данных из БД SQLite в Android-проектах используется класс Cursor.

В созданный объект data класса Cursor можно вернуть массив строк с нумерованными колонками (полями), в котором колонки нумеруются начиная с нулевой:

Cursor data = DBH.loadData(), где loadData – метод, возвращающий структуру данных, полученную вследствие выполнения запроса SELECT: cursor = db.rawQuery(Query,null). Query – текст SQL-запроса на выборку данных.

data.getCount() – определяет количество записей в структуре;

data.moveToNext() – переход к следующей записи структуры;

data.getString(0) – получение значения первой колонки текущей строки;

data.getString(1) – получение значения второй колонки текущей строки и т.д.

## 2) Разработка мобильного приложения

### 2.1 Создание и настройка нового проекта приложения

По аналогии с предыдущими лабораторными работами.

### 2.2 Работа с ресурсами приложения

Рекомендуется начать с описания строковых констант, группируя их по каким-либо признакам, например, сначала, имя приложения (<string name="app\_name">Notes</string>), затем, прочие системные строковые константы (не для отображения пользователю), затем, короткие строковые константы (заголовки, подписи, подсказки), затем, длинные строковые константы (например, тексты сообщений пользователю).

Цвета настраиваются вместе с темами (темной и светлой). Выбираются такие сочетания цветов для фона и текста, чтобы данные легко читались.

Рекомендуется загрузка и настройка сторонних шрифтов для элементов списка.

Логотип приложения создается в редакторе на основе имеющегося выбранного из библиотеки. При желании разработчика, он может получить логотип приложения в онлайн-редакторе векторных изображений с загрузкой логотипа в проект.

Пиктограммы для кнопок и диалогов можно создать из имеющихся в библиотеке векторных изображений. Размер пиктограмм подбирается разработчиком в соответствии с эргономикой.

Понадобится, как минимум, три дополнительных пиктограммы – для добавления новой записи, удаления записи и удаления всех записей.

### 2.3 Создание разметки графических интерфейсов

Сначала создается разметка экрана главной активности. Рекомендуемый вид слоя `ConstraintLayout`. В слое на всю ширину и длину экрана размещается виджет списка `RecyclerView`. На отдельной панели инструментов или снизу размещаются кнопки для создания нового пункта списка и очистки всего списка с назначениям кнопкам соответствующих пиктограмм.

В соответствии с заданием, размечается макет строки пункта списка. Определяется перечень требуемых виджетов, их местоположение, привязки, размеры. Далее – создать макет экрана активности для добавления и редактирования записи. В макете учитывается эргономика интерфейса. Рекомендуется в данном макете создать дополнительную кнопку для удаления записи.

### 2.4 Создание управляющей логики

Последовательность создания программного кода определяется по использованию какой-либо функции или класса в другом модуле (функции, классе). В рамках выполнения лабораторной работы сначала рекомендуется разработка класса-помощника на основе класса `SQLiteOpenHelper` для взаимодействия приложения с БД. В классе-помощнике объявить константы для хранения имени БД, таблицы, полей (наименования разработчик определяет самостоятельно).

Далее, разработать собственный адаптер данных (`CustomAdapter` или `MyAdapter`) для `RecyclerView`, наследуя `RecyclerView.Adapter`. В `CustomAdapter` определяется структура данных. Для этого нужно с использованием класса `LayoutInflater` из содержимого указанного `layout`-файла (в нашем случае, макет строки списка) создать `View`-элемент (виджет).

В методе `onBindViewHolder` с каждым текстом виджета макета пункта списка в `holder` сопоставляется текст (значение) пункта соответствующего строкового списка. Сколько нужно виджетов для отображения текста в макете пункта списков, столько и строковых списков + свой строковый список для `id`, но `id` выводить пользователю не обязательно (если задание этого не предусматривает).

В `onBindViewHolder` также определяется событие, происходящее по нажатию на пункт списка. Здесь рекомендуется использовать процедурный подход, т.е. создать обработчика `holder.mainLayout.setOnClickListener...` При нажатии на пункт списка создается новое намерение для перехода к активности для изменения пункта списка `RecyclerView`: `Intent intent = new Intent(context, AddEditActivity.class),`

далее, передача намерению данных текущего списка, например, `intent.putExtra("id",String.valueOf(note_id.get(position)))`, и в завершении, запуск указанной активности с ожиданием от нее кода результата: `activity.startActivityForResult(intent, 1)`.

Для «держателя» вида нужно поставить в соответствие его объектов виджетам из макета строки списка, а также назначить действия тем, которые участвуют в непосредственном диалоге, например, удаление пункта списка при нажатии на виджет-кнопку с крестиком, для этого используется `class MyViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener`. Фрагмент метода в нем:

```
MyViewHolder(@NonNull View itemView) {  
    ...  
    note_n_txt = itemView.findViewById(R.id.note_n_txt);  
    ...  
    mainLayout = itemView.findViewById(R.id.mainLayout);  
    btnDelRV = itemView.findViewById(R.id.delBtnRV);  
    btnDelRV.setOnClickListener(this);  
}
```

Требуется решить, на какое событие в отношении каждого пункта списка будет открываться диалог-вопрос на удаление пункта. Существует несколько способов:

- создание кнопки с пиктограммой и «слушателя» ее нажатия (как во фрагменте выше);
- создание «слушателя» левого или правого свайпа;
- создание «слушателя» длительного нажатия на пункт списка.

Что будет при нажатии на кнопку прописывается в функции:

```
@Override  
public void onClick(View view) {  
    ...  
}
```

В `MainActivity.java` объявляются нужные объекты (адаптер, внутренние строковые списки), а также создаются обработчики событий, функция заполнения внутренних строковых списков и т.д.

В активности для создания/редактирования пункта списка также объявляются объекты, необходимые строковые переменные, создается поведение активности при открытии ее в режиме добавления новой записи и редактирования существующей. Если в данной активности предусмотрены управляющие виджеты (например, кнопки), к ним прописываются события `onClick(View view)`.

Обязательно требуется заполнить текстовые виджеты или изменить состояние флажка под редактируемую запись. Для этого используется функция извлечения переданных в намерение данных: `title = getIntent().getStringExtra("title")`. Также, при попытке сохранения записи в БД предусмотреть исключительную ситуацию ввода пользователем неправильных или неполных данных с выводом коротких сообщений (`Toast`).

## 2.5 Тестирование

Выполнить проверку работоспособности мобильного приложения при сохранении нескольких пунктов списка при правильном поведении пользователя и при возникновении исключительной ситуации. Проверить корректность

отображения текста на дисплее в стандартной и темной теме и при смене ориентации экрана с портретной на альбомную (горизонтальную). Проверить функцию редактирования и удаления записи, удаления всех записей.

## Лабораторная работа 4

Тема: Клиент-серверная архитектура мобильного приложения для систем массового обслуживания и сложных вычислений.

Задачи:

1) Изучение функций и принципов взаимодействия элементов клиент-серверной архитектуры (прикладные протоколы, виды посылок, форматы данных). Основы клиент-серверного взаимодействия мобильного приложения и веб-сервера. Сервисно-ориентированный подход. Основы программирования на скриптовом серверном языке Php.

2) Разработка мобильного приложения:

- создание и настройка нового проекта приложения;
- разметка макета графического интерфейса главной активности (в соответствии с вариантом);
- создание управляющей логики для отправки HTTP-запроса на веб-сервер, получения и отображения ответа;
- создание веб-сервиса в виде файла \*.php для приема и обработки поступающих запросов и выдачи результата решения вычислительной задачи (в соответствии с вариантом);
- тестирование мобильного приложения.

1) Предприятия в сфере массового обслуживания населения для нужд контроля, мониторинга, предоставления полезных инструментов и сервисов клиентам используют многоуровневые клиент-серверные приложения, в которых клиентский уровень (клиент) не выполняет сложные вычисления и обработки данных, а только посылает запрос на серверный уровень (сегмент). Сервер, в свою очередь, понимая по принятым данным из запроса, что клиент желает получить и перенаправляет клиента в нужный сервис. Вызванный сервис может использовать распределенный или кластерный серверный сегмент, чаще всего состоящий из большого количества связанных в одну сеть ЭВМ, расположенный нередко в разных странах и на разных континентах. Актуальным трендом в сфере IT и развития систем массового обслуживания (электронных сервисов) является применение микросервисной архитектуры на основе концепции REST API (Representational State Transfer Application Programming Interface). Используется протокол прикладного уровня передачи данных HTTP, исключительно защищенный, т.е. HTTPS, обеспечивающий целостность и конфиденциальность данных при их передаче между веб-сервисом и устройством пользователя. Как архитектурно и программно устроен серверный сегмент пользователь не знает, для него доступно только приложение-клиент и точка входа (API Gateway) для вызова доступных сервисов. В нашем случае точкой входа является веб-ссылка на файл \*.php, принимающий, обрабатывающий HTTP-запросы (GET/POST/PUT), формирующий и отправляющий ответ на эти запросы (в виде HTML-кода или в формате JSON или XML). Здесь следует упомянуть MVC (Model-View-Controller) как наиболее популярный способ организации кода, который предполагает выделение блоков, отвечающих за решение разных задач. Один блок отвечает за данные приложения, другой отвечает за внешний вид, а третий контролирует работу приложения. Выходит, что Controller принимает запрос, верифицирует его,

защищает систему от возможных атак, блокирует попытки клиента нарушить работу сервиса, перенаправляет правильный запрос к сервисной функции (это уже Model), которая результат возвращает в Controller для его последующего перенаправления во View с целью рендеринга ответа. Controller выдает ответ клиенту на его запрос. Работа контроллера в этом смысле очень важная и ответственная.

## 2) Разработка мобильного приложения

### 2.1 Создание и настройка нового проекта приложения

Этап выполняется по аналогии с предыдущими лабораторными работами.

При модификации файла манифеста приложения следует включить разрешение на доступ к сети Интернет:

```
<uses-permission android:name="android.permission.INTERNET" />
```

### 2.2 Разметка макета графического интерфейса

Учитывая, что задание может предусматривать использование того же макета (или разметки) графического интерфейса главной активности, что и в приложении к первой лабораторной работе, XML-разметку для данного приложения можно скопировать с проекта приложения по первой лабораторной работе.

Если задание соответствует использованию стороннего ресурса, макет интерфейса строится новый.

### 2.3 Создание управляющей логики

Перед написанием кода взаимодействия приложения-клиента с backend, как и в ранних лабораторных работах, требуется заполнить необходимые строковые константы. Одной из них обязательно должен быть URL, являющийся точкой входа.

Следующее требование – выполнение HTTP-запроса и ожидание ответа в отдельном потоке, без блокирования главного (управляющего) потока. Т.е. целесообразно выполнение функции взаимодействия с вычислительным сервером через поток:

```
Thread Process = new Thread(GetSolve);  
Process.start();
```

GetSolve – функция, объявленная как Runnable GetSolve. Runnable – позволяет передать код функции GetSolve в класс Thread для выполнения.

Для создания контекста соединения используется объект класса HttpURLConnection. Из него в объект класса InputStream извлекается входной поток (ответ): `InputStream stream = connection.getInputStream()`.

После получения ответа в память нужно его записать в обычную строковую переменную request, но делается данная операция через читатель буфера `BufferedReader reader` и построчно:

```
String line = "";  
String request = "";  
BufferedReader reader = new BufferedReader(new InputStreamReader(stream));  
while ((line = reader.readLine()) != null) {  
    request += line;  
}
```



Для однострочных ответов можно не использовать цикл while и прочитать только одну строку.

Пример текста GET-запроса:

URL = "https://mybackend/srv.php?a=" + coeffa + "&b=" + coeffb + "&c="+coeffc, где coeffa, coeffb, coeffc – входные данные в виде строк, взятые, например, с виджетов EditText.

Ответ будет получен в известном ранее (ожидаемом) формате. Ответ подлежит разбору (парсингу) и представлению пользователю в удобочитаемом (эргономичном) виде. Выполняются такие операции в отдельной функции:

```
try {
    JSONObject obj = new JSONObject(result);
    JSONObject res = obj.getJSONObject("res");
    resultInfo.setText(res.toString());
} catch (JSONException e) {
    e.printStackTrace();
}
```

В отчете по данному пункту представить комментируемый листинг файла \*.java.

## 2.4 Создание веб-сервиса

При создании веб-сервиса следует учесть, что приложение-клиент работает только по защищенному HTTPS-протоколу. При возникновении технической сложности конфигурации своего веб-сервера на основе Apache или Nginx, можно воспользоваться указанным преподавателем сервером кафедры.

В выделенный каталог веб-сервера размещается файл \*.php, в нем формируется код, часть которого отвечает за прием и верификацию запроса, часть кода выполняет главные вычислительные операции, и код, «упаковывающий» результат вычислений в нужный выходной формат с отправкой результата клиенту.

При выполнении п. 2.4 представить комментируемый листинг программного кода веб-сервиса.

## 2.5 Тестирование мобильного приложения

Следует разработать и выполнить тест-кейсы по проверке обработок исключений на заполнение полей, неправильный формат и диапазон входных данных, на эргономику интерфейса и визуализацию результата. Результаты тестирования сопроводить дисплейными фрагментами интерфейса мобильного приложения.

## Лабораторная работа 5

Тема: Гибридные технологии разработки нетиповых клиент-серверных мобильных систем.

Задачи:

1) Изучение принципов слияния технологий разработки нетиповых мобильных приложений, обоснование применения веб-технологий к разработке мобильных приложений. Описание назначения и функционала виджета WebView. Преимущества и недостатки использования WebView в мобильном приложении. Изучение основ JavaScript, технологии AJAX, обзор JS-библиотек, поддерживающих асинхронные HTTP-запросы. Изучение каскадных стилей CSS для оформления веб-страницы и динамики поведения ее DOM-объектов как реакций на события (смена расширения экрана, действия пользователя).

2) Разработка мобильного приложения:

- создание и настройка нового проекта приложения;
- разметка макета графического интерфейса главной активности;
- настройка виджета WebView, создание кода настройки и загрузки веб-страницы;
- создание и оформление веб-страницы;
- создание клиентского управляющего JS-кода;
- создание серверного скрипта (для вариантов лабораторной работы 4, использовавших сторонний веб-сервис (Приложение А, таблица 5));
- тестирование мобильного приложения.

1) Существует два варианта технической реализации приложений для мобильных устройств: мобильный веб-сайт и мобильное приложение. Мобильным веб-сайтом считается сайт, адаптирующийся для просмотра и функционирования на мобильном устройстве. Сайт может включать интерактивные компоненты с использованием JavaScript, HTML5, новых API браузеров. Мобильное приложение – это приложение, разработанное под конкретную мобильную платформу (iOS, Android, Windows Phone). Обычно мобильное приложение разрабатывается на языке высокого уровня (например, Java) и компилируется в нативный код ОС, дающий максимальную производительность.

Существует еще третий вариант – гибридное мобильное приложение, включающее виджет браузера (например, веб-компонент WebView) и другие требуемые виджеты. В этом случае часть мобильного приложения используется для навигации и интеграции с ОС, а веб-компонент – для показа веб-контента. Обычные пользователи не могут зачастую отличить такой вариант от нативного мобильного приложения.

В таблице 1 рассмотрим основные преимущества и недостатки использования нативных мобильных приложений (НМП), мобильных веб-сайтов (МВС) и гибридных мобильных приложений (ГМП).

Таблица 1 – Преимущества и недостатки вариантов технической реализации мобильных приложений

	НМП	МВС	ГМП
Преимущества	<ul style="list-style-type: none"> <li>1) Тесная связь с операционной системой и аппаратным обеспечением мобильного устройства.</li> <li>2) Высокое быстродействие.</li> </ul>	<ul style="list-style-type: none"> <li>1) Безграничные функциональные возможности при использовании современных технологий JS, HTML5, CSS3.</li> <li>2) Снижение затрат на проект с привлечением веб-разработчиков.</li> <li>3) Кроссплатформенность «из коробки».</li> </ul>	<ul style="list-style-type: none"> <li>1) Связь с операционной системой и аппаратным обеспечением мобильного устройства.</li> <li>2) Высокое быстродействие.</li> <li>3) Возможность использования технологий JS, HTML5, CSS3.</li> <li>4) При выходе новой версии МП, не затрагивающей нативный код, не требуется загрузка обновления на мобильные устройства.</li> <li>5) Относительно невысокие затраты на разработку проекта.</li> <li>6) Возможность «встраивания» контента на другие веб-ресурсы в виде отдельных компонентов.</li> </ul>
Недостатки	<ul style="list-style-type: none"> <li>1) Требуются высокооплачиваемые профессиональные разработчики ПО для мобильных устройств.</li> <li>2) При выходе новой версии МП необходима загрузка обновления на устройства клиентов.</li> <li>3) Требуется разработка (иногда «с нуля») и тестирование МП под каждую мобильную ОС.</li> <li>4) Возникновение «особых» условий лицензирования и распространения при использовании некоторых виджетов.</li> </ul>	<ul style="list-style-type: none"> <li>1) Нет связи с операционной системой и аппаратным обеспечением мобильного устройства.</li> <li>2) Невысокое быстродействие интерактивного сайта на устаревших мобильных веб-браузерах и устройствах.</li> <li>3) Трудности в адаптации МВС к работе на различных мобильных веб-браузерах.</li> </ul>	<ul style="list-style-type: none"> <li>1) Требуется портирование МП под каждую мобильную ОС.</li> <li>2) Снижение быстродействия при высокой нагрузке на виджет веб-браузера для сложного веб-контента.</li> </ul>

Как видно из таблицы 1, при использовании гибридных технологий разработки мобильных приложений, зачастую получаем высокий технико-экономический эффект. Однако, для принятия решения об использовании технологии разработки МП, со стороны предприятия-разработчика требуется анализ нескольких критериев, например, финансовое состояние компании и объем резерва средств под проект, опыт и квалификация программистов, требования к мобильному приложению (функциональные, аппаратные и программные), целевая аудитория и т.д.

## 2) Разработка мобильного приложения

### 2.1 Создание и настройка нового проекта приложения

Новый проект приложения создается и настраивается по аналогии с лабораторной работой 4.

### 2.2 Разметка макета графического интерфейса главной активности

В соответствии с вариантом и с использованием виджета `WebView` производится разметка макета графического интерфейса главной активности. Перед написанием разметки макета интерфейса главной активности, как и рекомендовано ранее, устанавливаются требуемые ресурсы (графические, шрифты, цвета, стили, строковые константы), в том числе, пиктограмма приложения. Адрес загружаемой веб-страницы должен быть строковой константой.

### 2.3 Настройка виджета `WebView`, создание кода загрузки веб-страницы

Виджет `WebView`, как основа графического интерфейса для последующей загрузки в него веб-страницы, размещается в слое, перекрывая его полностью или с отступами под панель инструментов (верхнюю или нижнюю). При желании разработчика в панель инструментов добавляются дополнительные виджеты (кнопки, меню и т.д.), т.е. приложение может сочетать работу через графические элементы и через DOM-объекты веб-страницы, загруженной в `WebView`.

Класс `WebView` позволяет создать объект с настройками в соответствии со спецификой загружаемого в него HTML-кода, например:

- `webSettings.setJavaScriptEnabled(true);` //разрешаем JavaScript
- //запрет на масштабирование страницы встроенными и дополнительными элементами управления

```
myWebView.getSettings().setBuiltInZoomControls(false);
webSettings.setDisplayZoomControls(false);
```

- `webSettings.setUseWideViewPort(true);` //загрузка видимой области страницы по умолчанию (как в десктопном браузере, «альбомный» режим)
- `webSettings.setLoadWithOverviewMode(true);` //загрузка веб-страницы без полос прокрутки, т.е. с полностью видимым содержимым.

Загрузка веб-страницы осуществляется при использовании класса `SimpleWebViewClient` – встроенного веб-браузера и виджета `WebView` как средства (контейнера) для просмотра загруженной веб-страницы.

Для начала требуется проверка, является ли указанный веб-ресурс целевой HTML-страницей (которая должна загрузиться в `WebView`):

```
private class SimpleWebViewClient extends WebViewClient {
    private Activity activity = null;
    @Override
    //проверяем возможность встраивания веб-страницы в контент приложения
    public boolean shouldOverrideUrlLoading(WebView webView, String url) {
        //если указанная строка - это ссылка
        if (url.contains(getString(R.string.API))) {
            //не нуждаемся в запуске стороннего браузера для указанной веб-страницы
            return false;
        }
        //или создание нового намерения для открытия обычного веб-браузера
        Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse(url));
        activity.startActivity(intent);
        return true;
    }
}
```

```
}  
}
```

Загрузка и визуализация веб-страницы выполняется созданием встроенного веб-браузера (webViewClient) и его указанием для визуализации веб-страницы в виджете myWebView:

```
SimpleWebViewClient webViewClient = new SimpleWebViewClient();  
myWebView.setWebViewClient(webViewClient);  
myWebView.loadUrl(getString(R.string.API));
```

## 2.4 Создание и оформление веб-страницы

Веб-страница верстается в удобном WYSIWYG-редакторе в виде документа HTML или создается файлом \*.php, генерирующим HTML-код в соответствии с вариантом.

Устанавливается кодировка веб-контента по умолчанию:

```
<meta charset="utf-8">
```

К веб-странице подключается файл с каскадными стилями, прикладная специальная JS-библиотека, например, JQuery, а также собственный JS-код:

```
<link rel="stylesheet" href="style.css" type="text/css">  
<script src="jquery-3.2.1.min.js" type="text/javascript"></script>  
<script src="myscript.js" type="text/javascript"></script>
```

Создается веб-форма с полями для ввода входных данных. Отдельно создается блок «DIV» для отображения результата запроса.

## 2.5 Создание управляющего JS-кода

Для отправки GET/POST запроса с входными данными на веб-сервер и приема ответа с результатом вычислений требуется создать JS-скрипт в виде отдельного файла, например, myscript.js с «прикреплением» его к веб-странице.

Здесь JS-скрипт используется только для выполнения асинхронного HTTP-запроса и может включать всего одну функцию, например, function send(){...}, вызываемую при нажатии на кнопку в веб-форме для отправки запроса:

```
<input id="btn" type="button" value="Решить" onclick="send();">
```

Пример полного JS-скрипта см. в приложении Б.

## 2.6 Создание серверного скрипта

Для вариантов лабораторной работы 4, использующих сторонний веб-сервис, требуется разработать собственный веб-сервис в виде файла \*.php для приема и обработки HTTP-запроса (GET или POST), так как это описано в лабораторной работе 4, п. 2.4.

## 2.6 Тестирование мобильного приложения

Выполнить тестирование мобильного приложения с проверкой обработок исключений, связанных с отсутствием или неполными входными данными, некорректностью их формата и диапазона варьирования. Результаты тестирования сопровождать скриншотами с мобильного устройства с описанием.

## Приложение А (обязательное)

### Варианты для выполнения лабораторных работ

Таблица А.1 – Варианты для выполнения лабораторной работы 1

№	Наименование	Входные данные	Результат
1	Вычисление параметров окружности	Радиус окружности	Площадь, диаметр и длина окружности
2	Вычисление значения аргумента функции $a_0 + a_1x + a_2x^2 + a_3x^3$ в точке методом золотого сечения	Коэффициенты $a_0, a_1, a_2, a_3$ , значение функции, допустимая погрешность	Значение аргумента функции, погрешность
3	Вычисление площади треугольника	Значения 2-х сторон треугольника и угла между ними	Значение площади треугольника
4	Вычисление параметров шара	Диаметр шара, материал изготовления (5 шт. на выбор)	Площадь поверхности, объем и масса шара
5	Вычисление параметров правильной пирамиды	Значения 2-х сторон основания и высота пирамиды	Площадь поверхности и объем пирамиды
6	Вычисление расстояния между двумя точками	Значения координат двух точек в 3-х мерном пространстве	Значение расстояния между двумя точками
7	Вычисление определенного интеграла от полинома 5-й степени: $a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5$ методом Ньютона-Лейбница	Коэффициенты полинома $a_0, a_1, \dots, a_5$	Значение определенного интеграла
8	Перевод числа из десятичной в двоичную, восьмеричную и шестнадцатеричную системы счисления	Число в десятичной системе счисления	Число в двоичной, восьмеричной и шестнадцатеричной системах счисления
9	Вычисление значения аргумента функции $a_0 + a_1x + a_2x^2$ в точке методом дихотомии	Коэффициенты полинома $a_0, a_1, a_2$ , значение функции, допустимая погрешность	Значение аргумента функции, погрешность
10	Вычисление времени нагрева жидкости, с от 24 °С до температуры кипения	Нагреваемая жидкость (5 шт. на выбор), объем нагреваемой жидкости, мощность нагревателя	Теплоемкость и температура кипения выбранной жидкости, время нагрева, с. до температуры кипения
11	Константа скорости химической реакции	Предэкспоненциальный множитель $k_0 \cdot 10^{15}$ , энергия активации $E \cdot 10^3$ Дж/моль, температура, К	Константа скорости $k$ химической реакции

№	Наименование	Входные данные	Результат
12	Расчет тормозного пути по известной скорости и ускорению тела	Скорость, м/с и ускорение $\text{м/с}^2$ тела	Тормозной путь $S$ , м.
13	Расчет сопротивления проводника по его удельному сопротивлению, длине и площади поперечного сечения	Удельное сопротивление, Ом·м, длина, м и площадь поперечного сечения, $\text{м}^2$ проводника	Сопротивление проводника, Ом

Таблица А.2.1 – Варианты для выполнения лабораторной работы 2

№	Наименование отслеживаемого параметра	Наименование устройства
1	Освещенность	Ambient Light Sensor
2	Наклон по осям	Gyroscope
3	Количество шагов	Pedometer
4	Расстояние	Proximity Sensor
5	Магнитное поле	Magnetic field
6	Поворот по осям	Game rotation vector
7	Движение	Motion detect
8	Гравитация	Gravity
9	Линейное ускорение	Linear acceleration
10	Абсолютное ускорение	Accelerometer
11	Температура	Temperature Sensor
12	Атмосферное давление	Atmosphere Pressure
13	Влажность	Humidity Sensor
14	Географическая позиция устройства по широте и долготе	GPS

*Примечание:* системное наименование датчика может отличаться в зависимости от марки / модели мобильного устройства. Точное системное наименование датчика можно определить при выводе списка всех подключенных датчиков.

Таблица А.2.2 – Дополнительные варианты для выполнения лабораторной работы 2

№	Наименование аппаратного модуля и задание
15	Фонарик. Требуется одной программной кнопкой обеспечить управление фонариком (включить / выключить) со сменой изображений на кнопке, соответственно, при включении и выключении фонарика.
16	Вывод сведений о батарее устройства (уровень заряда, количество часов с момента полной зарядки), а также времени (в часах) непрерывной работы устройства.
17	Вывод сведений о марке / модели устройства, пользовательском названии устройства, версии Android, номера сборки, IMEI-кодов слотов SIM и абонентских номеров.
18	Список сетей Wi-Fi с указанием для каждой названия сети (SSID), MAC-адреса адаптера (BSSID) и опорной мощности, дБм.

Таблица А.3 – Варианты для выполнения лабораторной работы 3

№	Наименование
1	Деловые заметки с порядковым номером заметки, многострочным текстом заметки, датой и временем создания и изменения заметки из текущей даты. Сортировку записей установить по идентификатору по возрастанию.
2	Ежедневник с порядковым номером записи, темой записи, многострочным текстом записи и связанной с записью датой (вводится вручную). Сортировку записей установить по дате по возрастанию.
3	Перечень приказов на подписание с указанием номера приказа, даты и статуса приказа (подписан / не подписан). Сортировка приказов устанавливается по дате по возрастанию.
4	Перечень оборудования учебных лабораторий с указанием названия помещения, типа и названия оборудования. Сортировку записей установить по названию помещения по возрастанию.
5	Список служебных записок и заявлений работников на подписание с указанием типа документа (служебная записка или заявление), ФИО работника – составителя документа, даты составления документа. Сортировка документов устанавливается по дате составления по возрастанию.
6	Реестр программного обеспечения на ЭВМ кафедры, включающий название ПО, наименование производителя, многострочное описание ПО, тип лицензии (свободная, проприетарная, коммерческая). Сортировка записей устанавливается по названию программного продукта по возрастанию.
7	Реестр агрессивных химических соединений, хранимых в лаборатории с указанием названия соединения, химической формулы, места хранения соединения, класса опасности соединения. Сортировка записей устанавливается по месту хранения по возрастанию.
8	Реестр установленных ЭВМ аудитории кафедры с указанием № класса, количества в нем ЭВМ с их характеристиками. Сортировку записей установить по номеру класса и количеству ЭВМ.
9	Расписание занятий группы с указанием дня недели, четности недели, № пары, названия предмета с видом занятия, № аудитории, ФИО преподавателя. Сортировку установить по дням недели и № пары.
10	Карточки преподавателей кафедры с указанием ФИО, должности, степени, звания и фото преподавателя. Фото доступно с официального сайта или Медиапортала по прямой ссылке при загрузке его в виджет ImageView. Сортировку установить по званию и ФИО преподавателей.
11	Реестр научных лабораторий учебного заведения с указанием номера помещения, названия лаборатории, ФИО ответственного и подводимой мощности, Вт. Сортировку записей обеспечить по номеру помещения.
12	Реестр реактивов учебной лаборатории с указанием наименования вещества, его CAS-номера, даты поставки, места хранения (№ ячейки) и срока хранения, мес. Записи сортировать по наименованию вещества.
13	Реестр лабораторной посуды с указанием названия сосуда, его типа, назначения и материала изготовления. Сортировку произвести по названию сосуда.

Варианты для выполнения лабораторной работы 4 указаны в таблице 1.



Варианты для выполнения лабораторной работы 5 для собственного веб-сервиса см. таблицу А.1. Для стороннего веб-сервиса варианты указаны в таблице А.5.

Таблица А.5 – Варианты для выполнения лабораторной работы 5 (сторонний веб-сервис)

№	Задание	Название
1	Просклонять введенное ФИО по падежам	Morpher <a href="https://morpher.ru">https://morpher.ru</a>
2	Перевести слово или фразу на английский, немецкий или французский язык (опционально)	Яндекс-переводчик <a href="https://yandex.ru/dev/translate">https://yandex.ru/dev/translate</a>
3	Получить сведения о погодных условиях в городе	OpenWeatherMap <a href="https://openweathermap.org">https://openweathermap.org</a>
4	Получить библиографические сведения о книге / публикации по ее DOI или ISBN	Google Scholar <a href="https://serpapi.com/google-scholar-api">https://serpapi.com/google-scholar-api</a>
5	Получить сведения о своей успеваемости с использованием web-API ЕИС «Электронный Университет»	Медиапортал СПбГТИ(ТУ) <a href="https://media.technolog.edu.ru">https://media.technolog.edu.ru</a>
6	Получить сведения о текущем курсе валют по отношению к рублю	Currate <a href="https://currate.ru">https://currate.ru</a>
7	Получить сведения об IP-адресе, текущем и любом введенном	IPInfo <a href="https://ipinfo.io">https://ipinfo.io</a>
8	Получить список университетов выбранной страны	Hypolabs <a href="http://universities.hipolabs.com">http://universities.hipolabs.com</a>
9	Получить сведения о введенном почтовом индексе указанной страны	Zippopotam <a href="https://api.zippopotam.us">https://api.zippopotam.us</a>
10	Получить возможные молекулярные структуры по массе или формуле соединения	Royal Society Of Chemistry <a href="https://developer.rsc.org">https://developer.rsc.org</a>

## Приложение Б (обязательное)

### Примеры мобильных приложений

**Пример 1.** Мобильное приложение «Привет, Мир» для решения квадратного уравнения.

Листинг файла манифеста приложения AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.HelloWorld"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>

            <meta-data
                android:name="android.app.lib_name"
                android:value="" />
            </activity>
        </application>
    </manifest>
```

Листинг файла строковых констант strings.xml:

```
<resources>
    <string name="app_name">Привет, Мир!</string>
    <string name="header">Расчет корней квадратного уравнения</string>
    <string name="coeff1_lbl">Коэффициент 1</string>
    <string name="coeff2_lbl">Коэффициент 2</string>
    <string name="coeff3_lbl">Коэффициент 3</string>
    <string name="result_hint">результат??</string>
    <string name="result_noroots">Корней нет!</string>
    <string name="error1_text">Введите все коэффициенты!</string>
    <string name="Calculate">Рассчитать</string>
</resources>
```

## Листинг файла цветов colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
    <color name="myColor">#2196F3</color>
</resources>
```

## Листинг файла светлой темы ...res\values-night\themes.xml:

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.HelloWorld"
parent="Theme.MaterialComponents.DayNight.DarkActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_500</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/black</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/myColor</item>
        <item name="colorOnSecondary">@color/black</item>
        <item name="android:textColor">#E91E63</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>
</resources>
```

## Листинг файла темной темы ...res\values-night\themes.xml:

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.HelloWorld"
parent="Theme.MaterialComponents.DayNight.DarkActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_200</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/myColor</item>
        <item name="colorOnSecondary">@color/black</item>
        <item name="android:textColor">#EFE9E9</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>
</resources>
```

В каталог «font» загружен шрифт «lobster» (рисунок 5) с его настройкой для отображения (файл lobster.xml):

AndroidStudioProjects > HelloWorld > app > src > main > res > font				
Имя	Дата изменения	Тип	Размер	
lobster.xml	30.04.2023 10:37	Документ XML	1 КБ	
lobster_regular.ttf	06.04.2021 11:34	Файл шрифта Ttu...	397 КБ	

Рисунок 5 – Содержимое каталога «font» приложения

```
<?xml version="1.0" encoding="utf-8"?>
<font-family xmlns:android="http://schemas.android.com/apk/res/android">

    <font
        android:font="@font/lobster_regular"
        android:fontStyle="normal"
        android:fontWeight="400" />

</font-family>
```

#### Листинг и описание макета главной активности activity\_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#4B00BCD4"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:fontFamily="@font/lobster"
        android:text="@string/header"
        android:textSize="9pt"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.022" />

    <TextView
        android:id="@+id/textView1"
        android:layout_width="148dp"
        android:layout_height="25dp"
        android:layout_marginStart="16dp"
        android:gravity="right"
        android:text="@string/coeffl_lbl"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toStartOf="@+id/editTextNumberDecimal"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView"
        app:layout_constraintVertical_bias="0.036" />
```

```

<TextView
    android:id="@+id/textView2"
    android:layout_width="148dp"
    android:layout_height="25dp"
    android:layout_marginStart="16dp"
    android:layout_marginTop="24dp"
    android:gravity="right"
    android:text="@string/coeff2_lbl"
    app:layout_constraintEnd_toStartOf="@+id/editTextNumberDecimal"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView1" />

```

```

<TextView
    android:id="@+id/textView3"
    android:layout_width="148dp"
    android:layout_height="25dp"
    android:layout_marginStart="16dp"
    android:gravity="right"
    android:text="@string/coeff3_lbl"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView2"
    app:layout_constraintVertical_bias="0.039" />

```

```

<EditText
    android:id="@+id/coeff1value"
    android:layout_width="237dp"
    android:layout_height="44dp"
    android:layout_gravity="center_vertical"
    android:layout_marginTop="4dp"
    android:autofillHints="коэффициент 1"
    android:cursorVisible="true"
    android:ems="10"
    android:focusable="true"
    android:focusableInTouchMode="true"
    android:inputType="numberSigned|numberDecimal"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toEndOf="@+id/textView2"
    app:layout_constraintTop_toBottomOf="@+id/textView" />

```

```

<EditText
    android:id="@+id/coeff2value"
    android:layout_width="237dp"
    android:layout_height="44dp"
    android:layout_gravity="center_vertical"
    android:layout_marginTop="4dp"
    android:cursorVisible="true"
    android:ems="10"
    android:focusable="true"
    android:focusableInTouchMode="true"
    android:inputType="numberSigned|numberDecimal"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toEndOf="@+id/textView2"
    app:layout_constraintTop_toBottomOf="@+id/coeff1value" />

```

```

<EditText
    android:id="@+id/coeff3value"
    android:layout_width="237dp"
    android:layout_height="44dp"

```

```

        android:layout_gravity="center_vertical"
        android:layout_marginTop="4dp"
        android:cursorVisible="true"
        android:ems="10"
        android:focusable="true"
        android:focusableInTouchMode="true"
        android:inputType="numberSigned|numberDecimal"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toEndOf="@+id/textView2"
        app:layout_constraintTop_toBottomOf="@+id/coeff2value" />

<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="164dp"
    android:onClick="Calculate"
    android:text="@string/Calculate"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.498"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView"
    app:layout_constraintVertical_bias="0.016" />

<EditText
    android:id="@+id/cResult"
    android:layout_width="194dp"
    android:layout_height="wrap_content"
    android:ems="10"
    android:focusable="false"
    android:hint="@string/result_hint"
    android:inputType="text"
    android:textAlignment="center"
    android:textSize="24sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.497"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/button"
    app:layout_constraintVertical_bias="0.107"
    tools:visibility="visible" />

</androidx.constraintlayout.widget.ConstraintLayout>

```

В главной активности activity\_main использованы виджеты для ввода (EditText) и вывода (TextView) текста.

Описание виджетов:

- textView – заголовок с курсивным шрифтом @font/lobster;
- textView1 – текст «Коэффициент 1»;
- textView2 – текст «Коэффициент 2»;
- textView3 – текст «Коэффициент 3»;
- coeff1value – ввод 1-го коэффициента уравнения;
- coeff2value – ввод 2-го коэффициента уравнения;
- coeff3value – ввод 3-го коэффициента уравнения;
- button – кнопка для вызова главной расчетной функции Calculate.
- cResult – виджет для вывода результата решения квадратного уравнения.

Внешний вид макета (рисунок 6):

Расчет корней квадратного уравнения

Коэффициент 1

Коэффициент 2

Коэффициент 3

РАССЧИТАТЬ

результат???

Рисунок 6 – Внешний вид макета экрана главной активности

#### Листинг кода главной активности MainActivity.java

```
package com.example.helloworld;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;
import android.widget.EditText;
import android.widget.TextView;
import java.text.DecimalFormat;

public class MainActivity extends AppCompatActivity {

    //объявление объектов для взаимодействия с ними через код
    EditText cResult;
    TextView c1, c2, c3;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        //скрываем панель инструментов по умолчанию
        //с заголовком приложения
        getSupportActionBar().hide();

        //поиск и присвоение объектам в коде соответствующих виджетов в
        макете activity_main.xml
        cResult = (EditText) findViewById(R.id.cResult);
        c1 = (EditText) findViewById(R.id.coeff1value);
        c2 = (EditText) findViewById(R.id.coeff2value);
        c3 = (EditText) findViewById(R.id.coeff3value);
    }
}
```

```

public int Calculate(View view){

    //строковым переменным присваиваются значения коэффициентов
    //квадратного уравнения, взятые из введенных в виджеты
    //редактирования текста
    String a = c1.getText().toString();
    String b = c2.getText().toString();
    String c = c3.getText().toString();

    cResult.setText("");

    //проверка на заполнение значений коэффициентов
    if(a.length()==0 || b.length()==0 || c.length()==0){
        //вывод сообщения
        Toast toast = Toast.makeText(getApplicationContext(),
            R.string.error1_text,
            Toast.LENGTH_SHORT);
        toast.show();
        return 0;
    }
    //создание формата десятичного числа для округления
    DecimalFormat df = new DecimalFormat("#.###");

    //основной алгоритм решения квадратного уравнения
    double d = Math.pow(Float.parseFloat(b),2)-
    4*Float.parseFloat(a)*Float.parseFloat(c);
    if(d<0) {
        cResult.setText(R.string.result_noroots);
        return 0;
    } else
        if(d==0){
            double root = (-Float.parseFloat(b)-
Math.sqrt(d))/(2*Float.parseFloat(a));
            cResult.setText(String.valueOf(df.format(root)));
        } else{
            double root1 = (-Float.parseFloat(b)-
Math.sqrt(d))/(2*Float.parseFloat(a));
            double root2 = (-
Float.parseFloat(b)+Math.sqrt(d))/(2*Float.parseFloat(a));
            cResult.setText(String.valueOf(df.format(root1))+",
"+String.valueOf(df.format(root2)));
        }

        return 0;
    }
}

```

Снимки экрана приложения (рисунки 7-9) в темной теме:



*Расчет корней квадратного уравнения*

Коэффициент 1 \_\_\_\_\_

Коэффициент 2 \_\_\_\_\_

Коэффициент 3 \_\_\_\_\_

РАССЧИТАТЬ

результат???

Рисунок 7 – Снимок исходного экрана приложения Привет, Мир

*Расчет корней квадратного уравнения*

Коэффициент 1 -1

Коэффициент 2 2

Коэффициент 3 3

РАССЧИТАТЬ

3, -1

Рисунок 8 – Квадратное уравнение имеет два корня

*Расчет корней квадратного уравнения*

Коэффициент 1 1

Коэффициент 2 2

Коэффициент 3 3

РАССЧИТАТЬ

Корней нет!

Рисунок 9 – Квадратное уравнение не имеет корней

**Пример 2.** Мобильное приложение «Мои датчики» для вывода списка всех датчиков мобильного устройства и показаний гироскопа.

Листинг файла манифеста приложения AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportRtl="true"
        android:theme="@style/Theme.Sensors"
        tools:targetApi="31">
        <activity
            android:name=".SensorsList"
            android:exported="false">
            <meta-data
                android:name="android.app.lib_name"
                android:value="" />
        </activity>
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>

            <meta-data
                android:name="android.app.lib_name"
                android:value="" />
        </activity>
    </application>

</manifest>
```

Листинг файла строковых констант strings.xml:

```
<resources>
    <string name="app_name">Мои датчики</string>
    <string name="Label1">показания датчиков</string>
    <string name="SLabel">Гироскоп: %1$s %2$s %3$s</string>
    <string name="SensorsList">Список датчиков</string>
</resources>
```

Листинг файла цветов colors.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
```

```

    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFF</color>
</resources>

```

### Листинг файла светлой темы ...res\values\themes.xml:

```

<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.Sensors"
parent="Theme.MaterialComponents.DayNight.DarkActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_500</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_700</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>

    <style name="Button.Orange" parent="ThemeOverlay.AppCompat">
        <item name="colorAccent">@android:color/holo_orange_light</item>
    </style>
</resources>

```

### Листинг файла темной темы ...res\values-night\themes.xml:

```

<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.Sensors"
parent="Theme.MaterialComponents.DayNight.DarkActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_200</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/black</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_200</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>

    <style name="Button.Orange" parent="ThemeOverlay.AppCompat">
        <item name="colorAccent">@android:color/holo_orange_dark</item>
    </style>
</resources>

```

### Листинг файла макета экрана главной активности activity\_main.xml:

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"

```

```

        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:context=".MainActivity">

        <TextView
            android:id="@+id/textView2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/Label1"
            android:textAlignment="center"
            android:textSize="18sp"
            android:textStyle="bold"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.498"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintVertical_bias="0.147" />

        <TextView
            android:id="@+id/sensor1value"
            android:layout_width="383dp"
            android:layout_height="29dp"
            android:text="@string/SLabel"
            android:textSize="18sp"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.428"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/textView2"
            app:layout_constraintVertical_bias="0.053" />

        <Button
            android:id="@+id/button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:onClick="Click"
            android:text="@string/SensorsList"
            style="@style/Button.Orange"
            android:backgroundTint="#FAAFAA"
            app:icon="@android:drawable/ic_menu_send"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.497"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintVertical_bias="0.041" />

    </androidx.constraintlayout.widget.ConstraintLayout>

```

**Описание виджетов экрана главной активности:**

- textView2 – текстовая область для вывода заголовка «показания датчиков»;
- sensor1value – текстовая область для вывода показаний гироскопа;
- button – кнопка для перехода к активности со списком всех датчиков мобильного устройства.

Дисплейный фрагмент макета экрана главной активности (режим Design) в светлой теме изображен на рисунке 10.

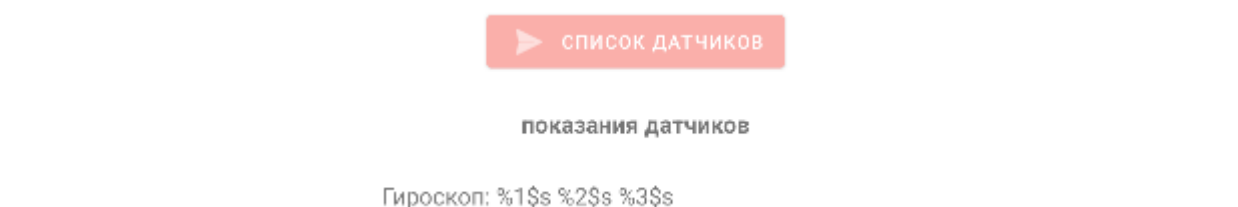


Рисунок 10 – Дисплейный фрагмент макета экрана главной активности в режиме Design

Листинг файла макета экрана активности activity\_sensors\_list.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".SensorsList">

    <ListView
        android:id="@+id/list_view"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginTop="8dp"
        android:layout_marginBottom="8dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Дисплейный фрагмент макета экрана активности activity\_sensors\_list (режим Design) в светлой теме и макета элемента списка в темной теме (рисунок 11):

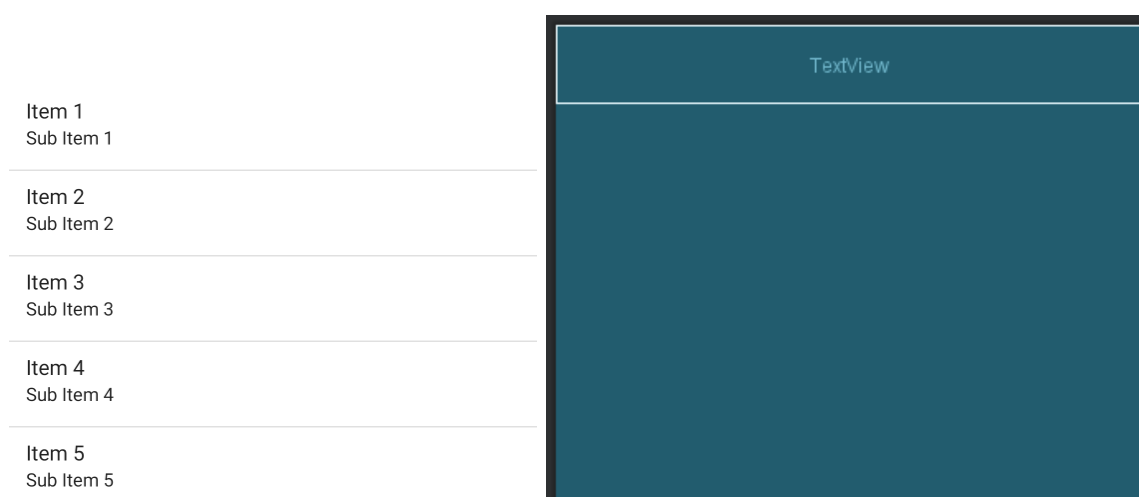


Рисунок 11 – Дисплейный фрагмент макета экрана главной активности (слева) и элемента списка (справа) в режиме дизайна

В макете activity\_sensors\_list.xml только один виджет list\_view, используемый для вывода списка датчиков устройства в элементы списка, оформленные макетом list\_item.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<TextView
    android:id="@+id/title"
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center_vertical"
    android:padding="16dp"
    android:textSize="16sp"/>
```

**title** – единственный виджет, используемый для вывода элемента списка, т.е. сведений о датчике. Виджет **list\_item** указывается для использования в качестве оформления элемента списка при создании адаптера списка.

Листинг кода главной активности MainActivity.java:

```
package com.sensors;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Context;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.widget.TextView;
import android.content.res.Resources;
import android.content.Intent;
import android.view.View;

public class MainActivity extends AppCompatActivity {

    private TextView sTxtValue;
    private SensorManager sensorManager;
    private Sensor mSensor;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        //с объектом sTxtValue сопрягаем виджет R.id.sensor1value
        sTxtValue = (TextView) findViewById(R.id.sensor1value);

        sensorManager = (SensorManager)
        getSystemService(Context.SENSOR_SERVICE);
        //указываем тип датчика для получения с него значений
        mSensor = sensorManager.getDefaultSensor(Sensor.TYPE_GYROSCOPE);

    }

    @Override
    protected void onStart() {
        //при запуске активности выполняется код ниже
        super.onStart();
    }

    @Override
    protected void onResume() {
```

```

        //регистрируем слушателя указанного датчика
        sensorManager.registerListener(listener,mSensor,
SensorManager.SENSOR_DELAY_NORMAL);
        //для класса-родителя
        super.onResume();
    }

    protected void onPause() {
        //для класса-родителя
        super.onPause();
        //приостановка слушателя датчика
        sensorManager.unregisterListener(listener);
    }

    private final SensorEventListener listener = new SensorEventListener() {

        public void onSensorChanged(SensorEvent event) {
            //к переменным присваиваем форматированный "ответ" от датчика
            String value1 = String.format("%.2f",event.values[0]);
            String value2 = String.format("%.2f",event.values[1]);
            String value3 = String.format("%.2f",event.values[2]);
            Resources res = getResources();
            //в переменную text помещаем текст с подстановкой трех значений
от датчика
            String text = String.format(getString(R.string.SLabel), value1,
value2, value3);
            //выводим в sTxtValue текст text
            sTxtValue.setText(text);
        }

        //вызываем этот метод при изменении точности датчика
        public void onAccuracyChanged(Sensor sensor, int accuracy) {

        }

    };

    public void Click(View view) {
        //создаем новое намерение для перехода на другую активность:
        Intent intent=new Intent(MainActivity.this,SensorsList.class);
        //запускаем активность:
        startActivity(intent);
    }

}

```

### Листинг кода активности SensorList.java:

```

package com.sensors;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Context;
import android.hardware.Sensor;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import java.util.ArrayList;
import java.util.List;

```

```

public class SensorsList extends AppCompatActivity {

    private List<String> list;
    private ArrayAdapter<String> adapter;
    private SensorManager sensorManager;
    private List<Sensor> deviceSensors;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sensors_list);

        //инициализируем sensorManager для работы с системными датчиками
        sensorManager = (SensorManager)
getSystemService(Context.SENSOR_SERVICE);
        //для списка датчиков указываем все (TYPE_ALL)
        deviceSensors = sensorManager.getSensorList(Sensor.TYPE_ALL);
        //инициализация строкового списка для заполнения сведений обо всех
датчиках на мобильном устройстве
        List<String> listSensorType = new ArrayList<>();
        for (int i = 0; i < deviceSensors.size(); i++) {
            listSensorType.add(Integer.toString(i)+"
"+deviceSensors.get(i).getName()+"\n"+
                "индекс типа: "+deviceSensors.get(i).getType()+"\n"+
                "производитель: "+deviceSensors.get(i).getVendor()+"\n"+
                "версия: "+deviceSensors.get(i).getVersion()+"\n"+
                "мощность: "+deviceSensors.get(i).getPower()+" мВт\n"+
                "задержка:
["+deviceSensors.get(i).getMinDelay()+", "+deviceSensors.get(i).getMaxDelay()+
"] мкс\n"+
                "макс. значение:
"+deviceSensors.get(i).getMaximumRange()+"\n"+
                "точность: "+deviceSensors.get(i).getResolution()+"\n"
            );
        }

        //создание адаптера списка и связывание через него представления
list_item
        //с заполненным ранее строковым списком listSensorType
        adapter = new ArrayAdapter<>(this, R.layout.list_item,
listSensorType);
        ListView listView = (ListView)findViewById(R.id.list_view);
        listView.setAdapter(adapter);
    }
}

```

Снимок экрана главной активности приложения (рисунок 12) в темной теме:

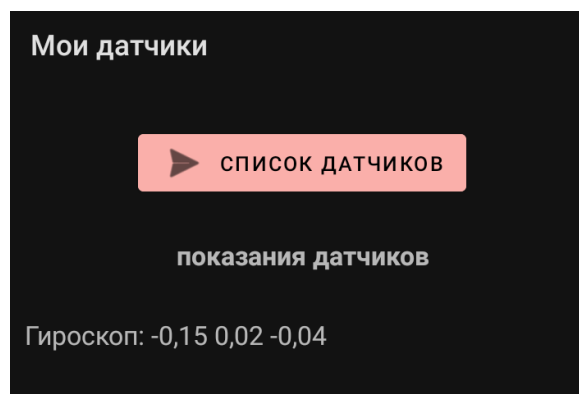


Рисунок 12 – Снимок экрана главной активности



При движении смартфона получаем его ускорение  $\text{м/с}^2$  по осям X, Y, Z.  
Дисплейный фрагмент экрана активности со списком датчиков мобильного устройства изображен на рисунке 13.

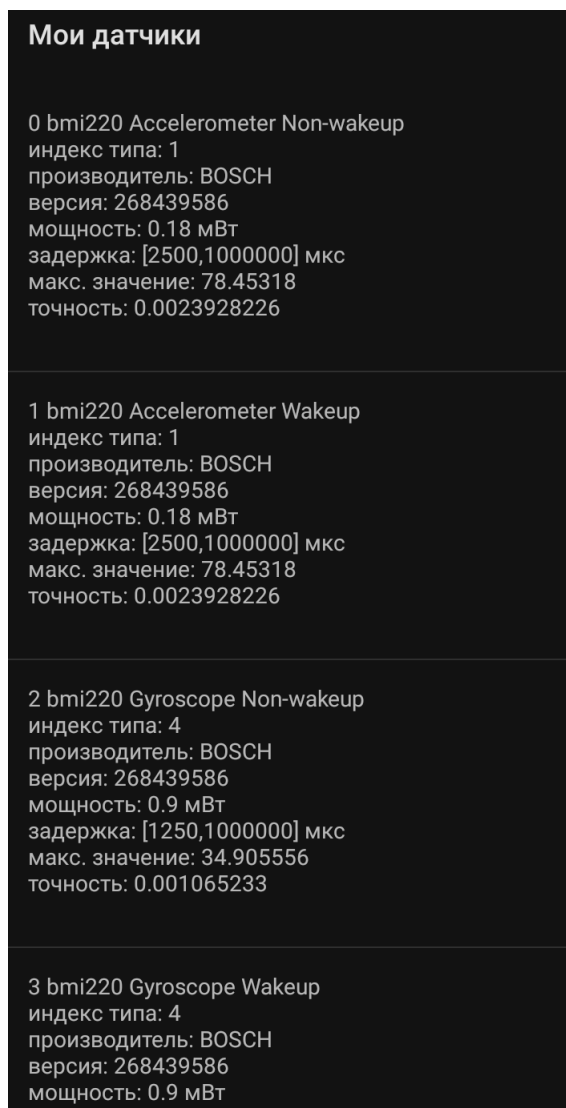


Рисунок 13 – Снимок экрана активности со списком всех датчиков на мобильном устройстве

**Пример 3.** Мобильное приложение «Мои заметки» для хранения текстовых заметок в локальной базе данных.

**Листинг файла манифеста приложения AndroidManifest.xml:**

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Notes"
        tools:targetApi="31">
        <activity
            android:name=".AddEditActivity"
            android:label="Заметка"
            android:exported="false">
            <meta-data
                android:name="android.app.lib_name"
                android:value="" />
        </activity>
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:label="Мои заметки">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category
                    android:name="android.intent.category.LAUNCHER" />
            </intent-filter>

            <meta-data
                android:name="android.app.lib_name"
                android:value="" />
        </activity>
    </application>

</manifest>
```

**Листинг файла строковых констант strings.xml:**

```
<resources>
    <string name="app_name">Мои заметки</string>
    <string name="c_id">id</string>
    <string name="c_title">title</string>
    <string name="c_text">text</string>
    <string name="c_prior">prior</string>
    <string name="note_title">Заголовок</string>
    <string name="note_text">Текст заметки</string>
    <string name="note_prior">Приоритет</string>
    <string name="priority">Приоритет</string>
    <string name="priority2">"приоритет: "</string>
    <string name="adding">Добавление заметки</string>
    <string name="editing">Редактирование заметки</string>
```

```

<string name="deleting">Удаление</string>
<string name="save">Сохранить</string>
<string name="delete_note">Удалить заметку?</string>
<string name="delete_all_notes">Удалить все заметки?</string>
<string name="yes">Да, удалить!</string>
<string name="no">Нет, не удалять!</string>
<string name="no_notes">заметок нет</string>
<string name="success_add">Новая заметка успешно добавлена!</string>
<string name="success_edit">Заметка успешно обновлена!</string>
<string name="success_delete">Заметка успешно удалена!</string>
<string name="success_delete_all">Все заметки успешно удалены!</string>
<string name="error_add">Ошибка при добавлении новой заметки!</string>
<string name="error_edit">Ошибка обновления!!!</string>
<string name="error_delete">Ошибка удаления!</string>
<string name="error">Ошибка!!!</string>
<string name="ok">ОК</string>
<string name="cancel">ОТМЕНА</string>
</resources>

```

### Листинг файла цветов colors.xml:

```

<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
</resources>

```

### Листинг файла светлой темы:

```

<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.Notes"
        parent="Theme.MaterialComponents.DayNight.DarkActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_500</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_700</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
        <item name="android:textColor">@color/purple_700</item>
    </style>
</resources>

```

### Листинг файла темной темы:

```

<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.Notes"
        parent="Theme.MaterialComponents.DayNight.DarkActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_200</item>

```

```

        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/black</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_200</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor"?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
        <item name="android:textColor">@color/teal_200</item>
    </style>
</resources>

```

Пиктограммы, используемые в приложении с их оформлением (Рисунок 14):



Рисунок 14 – Логотип приложения ic\_launcher.png

Растровый логотип приложения сопровождается его векторным двойником с кодом его оформления (ic\_launcher\_foreground.xml):

```

<vector xmlns:android="http://schemas.android.com/apk/res/android"
    android:width="108dp"
    android:height="108dp"
    android:viewportWidth="24"
    android:viewportHeight="24"
    android:tint="@android:color/holo_blue_dark">
    <group android:scaleX="0.5"
        android:scaleY="0.5"
        android:translateX="6.2"
        android:translateY="6.2">
        <path
            android:fillColor="@android:color/white"
            android:pathData="M20,8.69L20,4h-4.69L12,0.69 8.69,4L4,4v4.69L0.69,12
12,12 15.31,20L20,20v-4.69L23.31,12 20,8.69zM12,18c-
3.31,0 -6,-2.69 -6,-6s2.69,-6 6,-6 6,2.69 6,6 -2.69,6 -6,6zM12,8c-2.21,0 -
4,1.79 -4,4s1.79,4 4,4 4,-1.79 4,-4 -1.79,-4 -4,-4z"/>
        </group>
    </vector>

```

Задний фон логотипа приложения (файл ic\_launcher\_background.xml):




```

<?xml version="1.0" encoding="utf-8"?>
<vector
    android:height="108dp"
    android:width="108dp"
    android:viewportHeight="108"
    android:viewportWidth="108"
    xmlns:android="http://schemas.android.com/apk/res/android">
    <path android:fillColor="#ffff"
        android:pathData="M0,0h108v108h-108z"/>
</vector>

```

Логотипы для оформления кнопок помещаются в res/drawable (таблица 1)

Таблица 1 – Логотипы для оформления кнопок в приложении

Код разметки	Графический вид
<b>ic_add</b>	
<pre>&lt;vector android:height="48dp" android:tint="#7A62D0"     android:viewportHeight="24"     android:viewportWidth="24"     android:width="48dp"     xmlns:android="http://schemas.android.com/apk/res/android"&gt;     &lt;path android:fillColor="@android:color/white"         android:pathData="M19,13h-6v6h-2v-6H5v-2h6V5h2v6h6v2z"/&gt; &lt;/vector&gt;</pre>	
<b>ic_del</b>	
<pre>&lt;vector android:autoMirrored="true"     android:height="24dp"     android:tint="#7A62D0"     android:viewportHeight="24"     android:viewportWidth="24" android:width="24dp"     xmlns:android="http://schemas.android.com/apk/res/android"&gt;     &lt;path android:fillColor="@android:color/white"         android:pathData="M19,6.41L17.59,5 12,10.59 6.41,5 5,6.41 10.59,12 5,17.59 6.41,19 12,13.41 17.59,19 19,17.59 13.41,12z"/&gt; &lt;/vector&gt;</pre>	
<b>ic_delall</b>	
<pre>&lt;vector android:autoMirrored="true"     android:height="24dp"     android:tint="#7A62D0"     android:viewportHeight="24"     android:viewportWidth="24" android:width="24dp"     xmlns:android="http://schemas.android.com/apk/res/android"&gt;     &lt;path android:fillColor="@android:color/white"         android:pathData="M6,19c0,1.1 0.9,2 2,2h8c1.1,0 2,-0.9 2,-2L18,7L6,7v12zM8.46,11.88l1.41,-1.41L12,12.59l2.12,-2.12 1.41,1.41L13.41,14l2.12,2.12 -1.41,1.41L12,15.41l-2.12,2.12 -1.41,-1.41L10.59,14l-2.13,-2.12zM15.5,4l-1,-1h-5l-1,1L5,4v2h14L19,4z"/&gt; &lt;/vector&gt;</pre>	

Листинг файла разметки макета экрана главной активности (файл activity\_main.xml):

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/MainList"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:layout_editor_absoluteX="0dp"
        tools:layout_editor_absoluteY="0dp" />
```

```

<com.google.android.material.floatingactionbutton.FloatingActionButton
    android:id="@+id/AddBtn"
    android:layout_width="70dp"
    android:layout_height="70dp"
    android:layout_alignParentRight="true"
    android:layout_alignParentBottom="true"
    android:layout_margin="16dp"
    android:background="@drawable/ic_add"
    android:backgroundTint="@color/purple_200"
    app:tint="@color/teal_200"
    android:src="@drawable/ic_add"
    app:fabCustomSize="70dp"
    app:maxImageSize="30dp"
    app:layout_constraintBottom_toBottomOf="@+id/MainList"
    app:layout_constraintEnd_toEndOf="@+id/MainList"
    android:clickable="true"
    android:focusable="true" />

<com.google.android.material.floatingactionbutton.FloatingActionButton
    android:id="@+id/DelAllBtn"
    android:layout_width="70dp"
    android:layout_height="70dp"
    android:layout_alignParentRight="true"
    android:layout_alignParentBottom="true"
    android:layout_margin="16dp"
    android:layout_marginStart="16dp"
    android:background="@drawable/ic_delall"
    android:backgroundTint="@color/purple_200"
    android:clickable="true"
    android:focusable="true"
    android:src="@drawable/ic_delall"
    app:fabCustomSize="70dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:maxImageSize="30dp"
    app:tint="@color/teal_200" />

</androidx.constraintlayout.widget.ConstraintLayout>

```

Макет экрана главной активности в режиме дизайна изображен на рисунке 15.

Item 2  
Item 3  
Item 4  
Item 5  
Item 6  
Item 7  
Item 8  
Item 9



Рисунок 15 – Макет экрана главной активности в режиме дизайна

#### Листинг макета элемента списка (my\_row.xml):

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/mainLayout">

    <androidx.cardview.widget.CardView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="0dp"
        android:layout_marginBottom="1dp">

        <androidx.constraintlayout.widget.ConstraintLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:padding="5sp">

            <TextView
                android:id="@+id/note_id_txt"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="1"
                android:textColor="@color/teal_700"
                android:textSize="14sp"
                android:textStyle="bold"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent"
                app:layout_constraintVertical_bias="0" />

            <TextView
                android:id="@+id/note_n_txt"
```

```

        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="4dp"
        android:text="2"
        android:textColor="@color/teal_700"
        android:textSize="24sp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

<TextView
    android:id="@+id/note_title_txt"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="12dp"
    android:text="@string/note_title"
    android:textColor="@color/teal_200"
    android:textSize="16sp"
    app:layout_constraintStart_toEndOf="@+id/note_id_txt"
    app:layout_constraintTop_toTopOf="parent" />

<TextView
    android:id="@+id/note_text_txt"
    style="@style/Theme.Notes"
    android:layout_width="354dp"
    android:layout_height="32dp"
    android:layout_marginStart="4dp"
    android:layout_marginTop="12dp"
    android:maxLines="1"
    android:text="@string/note_text"
    android:textSize="24sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toEndOf="@+id/note_n_txt"
    app:layout_constraintTop_toBottomOf="@+id/note_title_txt"
    app:layout_constraintVertical_bias="0.0" />

<TextView
    android:id="@+id/note_prior_txt"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="3dp"
    android:layout_marginEnd="48dp"
    android:text="@string/note_prior"
    android:textColor="@color/teal_200"
    android:textStyle="bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<ImageButton
    android:id="@+id/delBtnRV"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="@null"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:srcCompat="@drawable/ic_del"
    tools:ignore="SpeakableTextPresentCheck,TouchTargetSizeCheck"
/>

</androidx.constraintlayout.widget.ConstraintLayout>
</androidx.cardview.widget.CardView>

```



</LinearLayout>

Внешний вид макета строки списка в режиме дизайна (рисунок 16).



Рисунок 16 – Макет строки списка в режиме дизайна

Листинг файла разметки активности для добавления и редактирования заметки (activity\_add\_edit.xml):

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="10sp"
    tools:context=".AddEditActivity">

    <Button
        android:id="@+id/delBtn"
        android:layout_width="335dp"
        android:layout_height="46dp"
        android:layout_marginTop="24dp"
        android:layout_marginEnd="32dp"
        android:drawableTintMode="add"
        android:foregroundTintMode="add"
        android:onClick="delBtnClick"
        android:text="Удалить"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/saveBtn" />

    <EditText
        android:id="@+id/etTitle"
        android:layout_width="327dp"
        android:layout_height="44dp"
        android:layout_marginTop="16dp"
        android:ems="10"
        android:hint="@string/note_title"
        android:inputType="textPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/etNote"
        android:layout_width="327dp"
        android:layout_height="228dp"
        android:layout_marginTop="56dp"
        android:ems="10"
        android:gravity="start|top"
        android:hint="@string/note_text"
        android:inputType="textMultiLine"
        app:layout_constraintEnd_toEndOf="parent"
```

```

        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="@+id/etTitle" />

<EditText
    android:id="@+id/etPrior"
    android:layout_width="335dp"
    android:layout_height="39dp"
    android:layout_marginTop="16dp"
    android:ems="10"
    android:hint="@string/priority"
    android:inputType="number"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/etNote" />

<Button
    android:id="@+id/saveBtn"
    android:layout_width="335dp"
    android:layout_height="51dp"
    android:layout_marginTop="12dp"
    android:layout_marginEnd="32dp"
    android:onClick="saveBtnClick"
    android:text="@string/save"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/etPrior" />
</androidx.constraintlayout.widget.ConstraintLayout>

```

Внешний вид макета экрана активности AddEditActivity в режиме дизайна (рисунок 17).

The image shows a design view of an Android activity. It features three text input fields stacked vertically. The first field is labeled 'Заголовок' (Title), the second 'Текст заметки' (Note text), and the third 'Приоритет' (Priority). Below these fields are two blue buttons with white text. The top button is labeled 'СОХРАНИТЬ' (Save) and the bottom button is labeled 'УДАЛИТЬ' (Delete).

Рисунок 17 – Макет экрана активности AddEditActivity в режиме дизайна

## Листинг кода MainActivity.java:

```
package com.example.notes;

import androidx.activity.result.ActivityResult;
import androidx.activity.result.ActivityResultCallback;
import androidx.activity.result.ActivityResultLauncher;
import androidx.activity.result.contract.ActivityResultContracts;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import android.content.DialogInterface;
import android.content.Intent;
import android.database.Cursor;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;
import com.google.android.material.floatingactionbutton.FloatingActionButton;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {
    RecyclerView mainList;
    FloatingActionButton AddBtn, DelAllBtn;
    DBHelper DBH;
    ArrayList<String> n_id, n_title, n_text, n_prior;
    CustomAdapter customAdapter;

    ActivityResultLauncher<Intent> addEditActivityLauncher = registerForActivityResult(
        new ActivityResultContracts.StartActivityForResult(),
        new ActivityResultCallback<ActivityResult>() {
            @Override
            public void onActivityResult(ActivityResult result) {
                //для отладки
                //Toast.makeText(MainActivity.this, result.getResultCode(), Toast.LENGTH_SHORT).show();
                mainListUpdate();
            }
        }
    );
};
```

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    mainList = findViewById(R.id.MainList);
    AddBtn = findViewById(R.id.AddBtn);
    DelAllBtn = findViewById(R.id.DelAllBtn);

    AddBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            //создаем новую намеренность и запускаем активность AddEditActivity для добавления
            //новой заметки
            Intent intent = new Intent(MainActivity.this, AddEditActivity.class);
            addEditActivityLauncher.launch(intent);
            //startActivityForResult(intent, 1);
        }
    });

    DelAllBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if(customAdapter.getItemCount() >0 ) {
                //если в адаптере есть хотя бы один пункт
                //формируем диалог - вопрос на удаление всех заметок
                AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
                builder.setTitle(R.string.deleting)
                    .setMessage(R.string.delete_all_notes)
                    .setIcon(R.drawable.ic_delall)
                    .setCancelable(true)
                    .setNegativeButton(R.string.no,
                        new DialogInterface.OnClickListener() {
                            public void onClick(DialogInterface dialog, int id) {
                                //просто закрываем диалог
                                dialog.cancel();
                            }
                        })
                    .setPositiveButton(R.string.yes,
                        new DialogInterface.OnClickListener() {
                            public void onClick(DialogInterface dialog, int id) {

```

```

        //создаем экземпляр класса DBHelper
        DBH = new DBHelper(MainActivity.this);
        //вызываем его метод, удаляющий все заметки из БД
        DBH.delAllNotes();
        //обновляем список
        mainListUpdate();
        //закрываем диалог
        dialog.cancel();
    }
    });
    AlertDialog alert = builder.create();
    //диалог сконфигурирован, отображаем его
    alert.show();
}
});

DBH = new DBHelper(MainActivity.this);
n_id = new ArrayList<>();
n_title = new ArrayList<>();
n_text = new ArrayList<>();
n_prior = new ArrayList<>();
//заполняем списки
fillArrays();
//в качестве адаптера для виджета списка указываем customAdapter
customAdapter = new CustomAdapter(MainActivity.this,this, n_id, n_title, n_text, n_prior);
mainList.setAdapter(customAdapter);
//для mainList устанавливаем линейный (Linear) слой в макете дисплея главной активности
mainList.setLayoutManager(new LinearLayoutManager(MainActivity.this));

}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode,resultCode, data);
    if (requestCode == 1) {
        //если код результата (возврата) от активности = 1, обновляем главный список
        mainListUpdate();
    }
}
}

```

```

void mainListUpdate() {
    //два действия ниже обновляют главный список заметок
    fillArrays();
    mainList.setAdapter(customAdapter);
}

void fillArrays() {
    Cursor cursor = DBH.loadData();
    n_id.clear();
    n_title.clear();
    n_text.clear();
    n_prior.clear();
    if(cursor.getCount() == 0){
        Toast.makeText(this, R.string.no_notes, Toast.LENGTH_SHORT).show();
    } else {
        while(cursor.moveToNext()){
            //по одной записи заполняем 4 списка
            //переходя по записям курсора
            n_id.add(cursor.getString(0));
            n_title.add(cursor.getString(1));
            n_text.add(cursor.getString(2));
            n_prior.add(cursor.getString(3));
        }
    }
}
}

```

## Листинг кода DBHelper.java:

```
package com.example.notes;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.widget.Toast;
import androidx.annotation.Nullable;

//класс-помощник для работы приложения с локальной БД
//создается на основе класса SQLiteOpenHelper
public class DBHelper extends SQLiteOpenHelper {
    private Context context;
    //объявляем константы с определением наименования БД, ее версии,
    //имя таблицы в БД и ее полей
    private static final String DB_NAME = "Notes.db";
    private static final int DB_VERSION = 1;
    private static final String TBL_NAME = "tblNotes";
    private static final String C_ID = "id";
    private static final String C_TITLE = "title";
    private static final String C_NOTE = "note";
    private static final String C_PRIOR = "prior";
    public DBHelper(@Nullable Context context){
        //указываем для DBHelper контекст класса, имя БД и ее версию,
        //далее, БД с указанным именем используется по умолчанию
        super(context,DB_NAME,null,DB_VERSION);
        this.context = context;
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        //создание новой таблицы в БД
        String Query = "CREATE TABLE " + TBL_NAME + "( " +
            C_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
            C_TITLE+" TEXT, " +
            C_NOTE+" TEXT, " +
            C_PRIOR+" INTEGER);";
        db.execSQL(Query);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int i, int il) {
        //удаление таблицы в БД, если она есть
        db.execSQL("DROP TABLE IF EXISTS " + TBL_NAME + ";");
        onCreate(db);
    }

    void addNote(String title, String note, int prior){
        //функция для создания новой заметки
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues cv = new ContentValues();
        cv.put(C_TITLE,title);
        cv.put(C_NOTE,note);
        cv.put(C_PRIOR,prior);
        long res = db.insert(TBL_NAME,null,cv);
        if(res == -1){
            Toast.makeText(context, R.string.error_add,
                Toast.LENGTH_SHORT).show();
        } else {
```

```

        Toast.makeText(context, R.string.success_add,
        Toast.LENGTH_SHORT).show();
    }
}

Cursor loadData(){
    //функция для загрузки всех заметок в курсор
    String Query = "SELECT " + C_ID+", "+C_TITLE+", "+C_NOTE+", "+C_PRIOR +
    " FROM " + TBL_NAME + " ORDER BY " + C_PRIOR + ";";
    SQLiteDatabase db = this.getReadableDatabase();
    Cursor cursor = null;
    if(db != null){
        cursor = db.rawQuery(Query,null);
    }
    //функция возвращает курсор, содержащий нумерованные колонки с
    данными,
    //номера колонок начинаются с нуля
    return cursor;
}

void editData(String id, String title, String text, int prior){
    //функция для редактирования заметки по ее id
    SQLiteDatabase db = this.getWritableDatabase();
    ContentValues cv = new ContentValues();
    cv.put(C_TITLE, title);
    cv.put(C_NOTE, text);
    cv.put(C_PRIOR, prior);
    long result = db.update(TBL_NAME, cv, C_ID+"=?", new String[]{id});
    if(result == -1){
        Toast.makeText(context, R.string.error_edit,
        Toast.LENGTH_SHORT).show();
    } else {
        Toast.makeText(context, R.string.success_edit,
        Toast.LENGTH_SHORT).show();
    }
}

void delOneNote(String id){
    //функция для удаления одной заметки по ее id
    SQLiteDatabase db = this.getWritableDatabase();
    long result = db.delete(TBL_NAME, C_ID+"=?", new String[]{id});
    if(result == -1){
        Toast.makeText(context, R.string.error_delete,
        Toast.LENGTH_SHORT).show();
    } else {
        Toast.makeText(context, R.string.success_delete,
        Toast.LENGTH_SHORT).show();
    }
}

void delAllNotes(){
    //функция для удаления всех заметок
    SQLiteDatabase db = this.getWritableDatabase();
    long result = db.delete(TBL_NAME,"",new String[]{});
    if(result == -1){
        Toast.makeText(context, R.string.error_delete,
        Toast.LENGTH_SHORT).show();
    } else {
        Toast.makeText(context, R.string.success_delete_all,
        Toast.LENGTH_SHORT).show();
    }
}
}

```



## Листинг кода CustomAdapter.java:

```
package com.example.notes;

import android.app.Activity;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageButton;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;
import androidx.recyclerview.widget.RecyclerView;
import java.util.ArrayList;

public class CustomAdapter extends RecyclerView.Adapter<CustomAdapter.MyViewHolder>{

    private Activity activity;
    private Context context;
    private ArrayList note_id, note_title, note_text, note_prior;

    CustomAdapter(Activity activity, Context context, ArrayList note_id, ArrayList note_title, ArrayList note_text,
ArrayList note_prior){
        //структура адаптера включает списки, каждый из которых является колонкой в таблице БД
        this.activity = activity;
        this.context = context;
        this.note_id = note_id;
        this.note_title = note_title;
        this.note_text = note_text;
        this.note_prior = note_prior;
    }

    @NonNull
```

```

@Override
public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType){
    //построитель представления на основе макета my_row закрепляет
    //макет my_row со всем его содержимым с представлением view для работы адаптера как класса с ним
    LayoutInflater inflater = LayoutInflater.from(context);
    View view = inflater.inflate(R.layout.my_row, parent, false);
    //функция возвращает экземпляр, управляющий указанным представлением
    return new MyViewHolder(view);
}

@Override
public void onBindViewHolder(@NonNull final MyViewHolder holder, final int position) {
    //метод присваивает тексту каждого объекта кода (адаптера) значение элемента списка по его позиции
    //идентификатор:
    holder.note_id_txt.setText(String.valueOf(note_id.get(position)));
    //номер по порядку берется из текущей позиции + 1:
    holder.note_n_txt.setText(String.valueOf(position+1));
    //заголовок заметки:
    holder.note_title_txt.setText(String.valueOf(note_title.get(position)));
    //текст заметки:
    holder.note_text_txt.setText(String.valueOf(note_text.get(position)));
    //приоритет заметки:
    holder.note_prior_txt.setText(activity.getString(R.string.priority2)+String.valueOf(note_prior.get(position)));

    holder.mainLayout.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            //обработка нажатия на элемент списка, при нажатии:
            //создается новое намерение для текущего контекста и класса AddEditActivity
            Intent intent = new Intent(context, AddEditActivity.class);
            //передаем новому намерению значения переменных
            intent.putExtra(activity.getString(R.string.c_id), String.valueOf(note_id.get(position)));
            intent.putExtra(activity.getString(R.string.c_title), String.valueOf(note_title.get(position)));
            intent.putExtra(activity.getString(R.string.c_text), String.valueOf(note_text.get(position)));
            intent.putExtra(activity.getString(R.string.c_prior), String.valueOf(note_prior.get(position)));
            //запускаем указанную активность AddEditActivity для изменения заметки с ожиданием
            //кода результата от нее
            activity.startActivityForResult(intent, 1);
        }
    });
}

```

```

@Override
public int getItemCount() {
    return note_id.size();
}

class MyViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener{
    TextView note_id_txt, note_n_txt, note_title_txt, note_text_txt, note_prior_txt;
    ImageButton btnDelRV;
    LinearLayout mainLayout;
    MyViewHolder(@NonNull View itemView) {
        //каждому объекту представления в адаптере присваивается
        //виджет указанного ранее макета (my_row.xml) по его идентификатору
        super(itemView);
        note_id_txt = itemView.findViewById(R.id.note_id_txt);
        note_n_txt = itemView.findViewById(R.id.note_n_txt);
        note_title_txt = itemView.findViewById(R.id.note_title_txt);
        note_text_txt = itemView.findViewById(R.id.note_text_txt);
        note_prior_txt = itemView.findViewById(R.id.note_prior_txt);
        mainLayout = itemView.findViewById(R.id.mainLayout);
        btnDelRV = itemView.findViewById(R.id.delBtnRV);
        //дополнительно, для кнопки с изображением (ImageButton) назначается "функция-слушатель" нажатия,
        //обработчик которой ниже
        btnDelRV.setOnClickListener(this);
    }

    @Override
    public void onClick(View view) {
        //обработчик нажатия на кнопку с изображением (btnDelRV)
        //формируем диалог-вопрос на удаление заметки
        AlertDialog.Builder builder = new AlertDialog.Builder(activity);
        builder.setTitle(R.string.deleting)
            .setMessage(R.string.delete_note)
            .setIcon(R.drawable.ic_delall)
            .setCancelable(true)
            .setNegativeButton(R.string.cancel,
                new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                        dialog.cancel();
                    }
                })
    }
}

```

```

        .setPositiveButton(R.string.ok,
            new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    DBHelper dbh = new DBHelper(activity);
                    //определяем текущую позицию записи из адаптера
                    int itemPos = getAdapterPosition();
                    //идентификатор заметки определяем по позиции элемента списка note_id
                    String nid = note_id.get(itemPos).toString();
                    //удаляем заметку по ее идентификатору
                    dbh.delOneNote(nid);
                    //пересоздаем главную активность, при этом
                    //полностью обновляется главный список с заметками
                    activity.recreate();
                    //закрываем диалог
                    dialog.cancel();
                }
            });
        AlertDialog alert = builder.create();
        //диалог создан, отобразим его
        alert.show();
    }
}

```

## Листинг кода AddEditActivity.java:

```
package com.example.notes;

import androidx.appcompat.app.ActionBar;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import android.content.DialogInterface;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class AddEditActivity extends AppCompatActivity {

    EditText etTitle, etNote, etPrior;
    Button saveBtn, delBtn;
    String id, title, text, prior;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add_edit);

        //присваивание объектам кода виджетов из макета activity_add_edit.xml
        etTitle = findViewById(R.id.etTitle);
        etNote = findViewById(R.id.etNote);
        etPrior = findViewById(R.id.etPrior);
        saveBtn = findViewById(R.id.saveBtn);
        delBtn = findViewById(R.id.delBtn);
        id = getIntent().getStringExtra(getString(R.string.c_id));

        ActionBar ab = getSupportActionBar();
        if(getIntent().hasExtra(getString(R.string.c_id))){
            //меняем текст заголовка экрана при запуске активности activity_add_edit для
            //редактирования заметки
            ab.setTitle(R.string.editing);
            delBtn.setVisibility(View.VISIBLE);
            getAndSetIntentData();
        }
    }
}
```

```

    } else {
        //меняем текст заголовка экрана при запуске активности activity_add_edit для
        //добавления заметки
        ab.setTitle(R.string.adding);
        delBtn.setVisibility(View.INVISIBLE);
    }

    saveBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            DBHelper dbh = new DBHelper(AddEditActivity.this);
            if(getIntent().hasExtra(getString(R.string.c_id))){
                //если в активность activity_add_edit передан id,
                //присваиваем полученный id переменной id
                id = getIntent().getStringExtra(getString(R.string.c_id));
                //вызываем метод editData для редактирования заметки по ее id
                dbh.editData(id,
                    etTitle.getText().toString().trim(),
                    etNote.getText().toString().trim(),
                    Integer.valueOf(etPrior.getText().toString())
                );
            } else {
                //иначе, вызываем метод addNote для добавления новой заметки
                dbh.addNote(
                    etTitle.getText().toString().trim(),
                    etNote.getText().toString().trim(),
                    Integer.valueOf(etPrior.getText().toString())
                );
            }
            //передаем код результата запущенной активности = 1
            setResult(1, null);
            //работа активности activity_add_edit завершается
            finish();
        }
    });

    delBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            //если активности передан id

```

```

        if (getIntent().hasExtra(getString(R.string.c_id))) {
            String nid = id;
            AlertDialog.Builder builder = new AlertDialog.Builder(AddEditActivity.this);
            builder.setTitle(R.string.deleting)
                .setMessage(R.string.delete_note)
                .setIcon(R.drawable.ic_delall)
                .setCancelable(true)
                .setNegativeButton(R.string.cancel,
                    new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog, int id) {
                            dialog.cancel();
                        }
                    })
                .setPositiveButton(R.string.ok,
                    new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog, int id) {
                            DBHelper dbh = new DBHelper(AddEditActivity.this);
                            dbh.delOneNote(nid);
                            dialog.cancel();
                            setResult(1, null);
                            finish();
                        }
                    });
            AlertDialog alert = builder.create();
            alert.show();
        }
    }
});
}

void getAndSetIntentData() {
    if (getIntent().hasExtra(getString(R.string.c_id))) {
        //выполняется для режима редактирования заметки
        id = getIntent().getStringExtra(getString(R.string.c_id));
        title = getIntent().getStringExtra(getString(R.string.c_title));
        text = getIntent().getStringExtra(getString(R.string.c_text));
        prior = getIntent().getStringExtra(getString(R.string.c_prior));

        etTitle.setText(title);
    }
}

```

```
        etNote.setText(text);
        etPrior.setText(prior);
    }
    //иначе, выводим сообщение об ошибке (id должен быть обязательно передан
    //для режима редактирования заметки)
    else Toast.makeText(this, R.string.error, Toast.LENGTH_SHORT).show();
}
}
```



Дисплейные фрагменты работы приложения «Мои заметки» в темном стиле показаны на рисунках 18, 19.

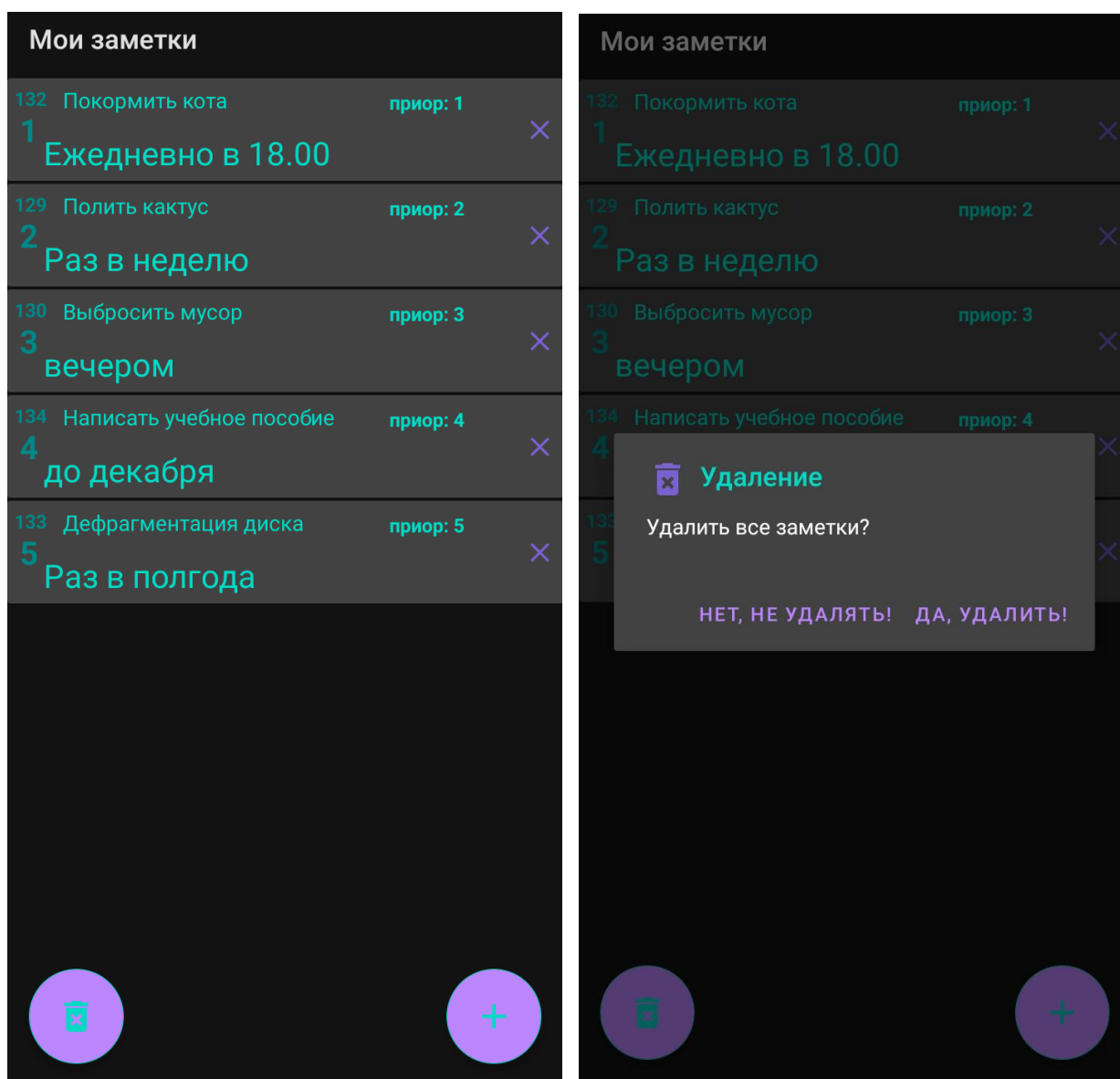


Рисунок 18 – Главный список заметок (слева) и диалог удаления всех заметок (справа)

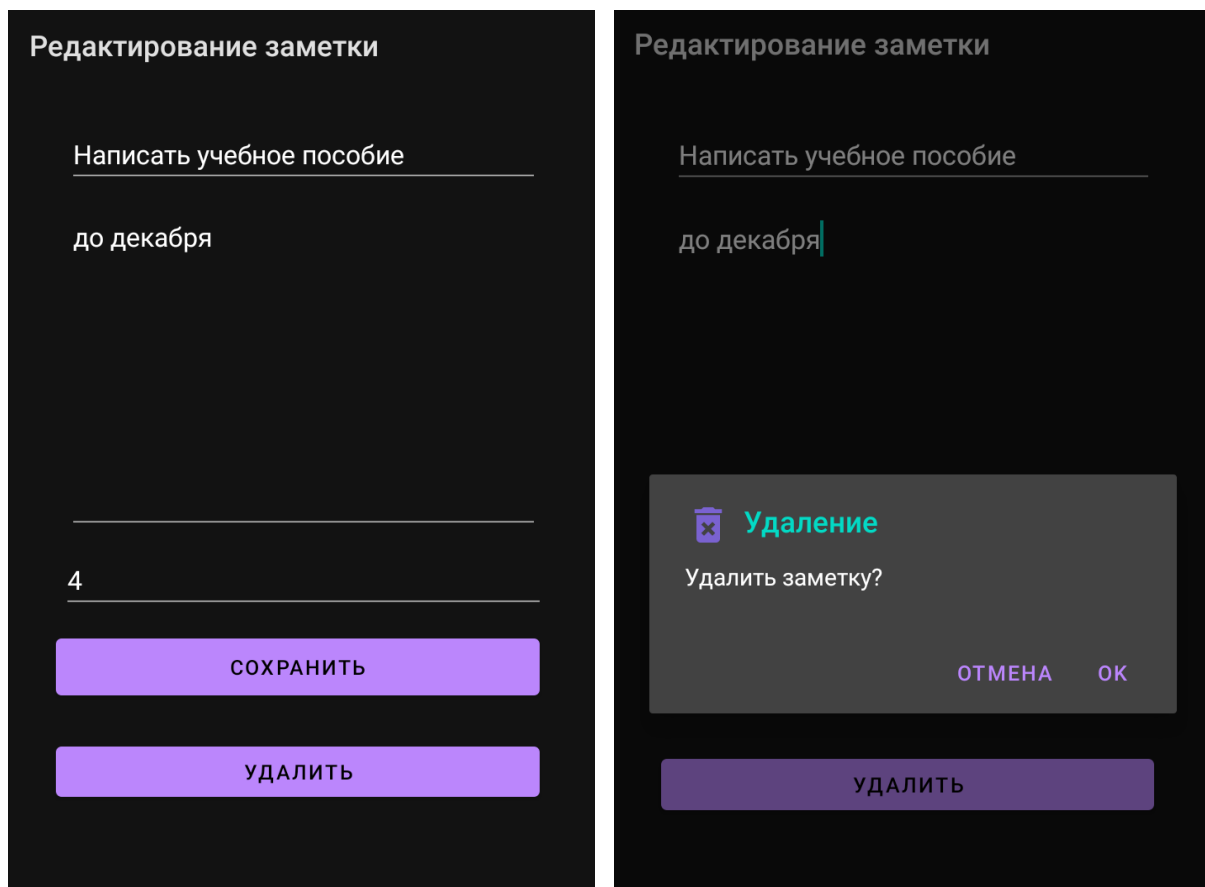


Рисунок 19 – Экран активности для добавления или редактирования заметки (слева) и диалог удаления заметки (справа)

**Пример 4.1** Мобильное приложение «Погодный Информер» для определения погодных условий в выбранном городе (с использованием стороннего внешнего веб-сервиса OpenWeatherMap).

Листинг файла манифеста приложения AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <uses-permission android:name="android.permission.INTERNET"/>

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.WeatherApp"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>

            <meta-data
                android:name="android.app.lib_name"
                android:value="" />
            </activity>
        </application>
    </manifest>
```

Листинг файла разметки макета экрана главной активности activity\_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#333"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/logo"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="15sp"
        android:text="@string/logo"
        android:textAlignment="center"
        android:textColor="@color/TextColor"
        android:textSize="30sp" />
```

```

<EditText
    android:id="@+id/cityName"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@id/logo"
    android:layout_marginTop="10sp"
    android:ems="10"
    android:hint="@string/cityNameText"
    android:text="Санкт-Петербург"
    android:inputType="textCapWords"
    android:minHeight="48dp"
    android:textAlignment="center"
    android:textColor="@color/teal_200"
    android:textColorHint="@color/purple_200"
    tools:ignore="TextContrastCheck" />

<Button
    android:id="@+id/btnGetWeather"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/cityName"
    android:layout_marginTop="15sp"
    android:backgroundTint="@color/purple_700"
    android:text="@string/getWeatherText"
    android:textColor="@color/TextColor" />

<TextView
    android:id="@+id/resultInfo"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/btnGetWeather"
    android:layout_marginStart="5sp"
    android:layout_marginTop="20sp"
    android:layout_marginEnd="5sp"
    android:layout_marginBottom="20sp"
    android:inputType="textNoSuggestions|textMultiLine"
    android:maxLines="1024"
    android:textColor="@color/teal_200"
    android:textSize="25sp" />

</RelativeLayout>

```

#### Описание виджетов:

- logo – текстовый логотип;
- cityName – текстовое поле для ввода названия города;
- btnGetWeather – кнопка для отправки запроса веб-сервису OpenWeatherMap;
- resultInfo – многострочная текстовая область для вывода сведений о полученных погодных условиях.

Дисплейный фрагмент макета графического интерфейса главной активности изображен на рисунке 20.

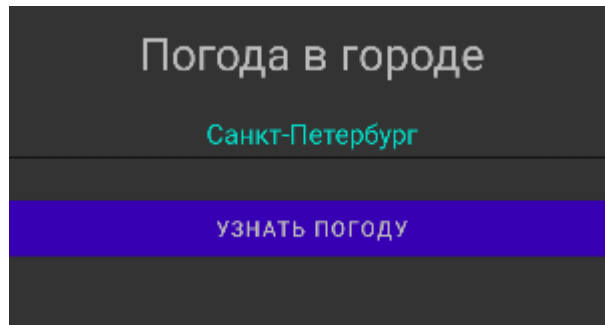


Рисунок 20 – Дисплейный фрагмент макета графического интерфейса главной активности

#### Листинг файла строковых констант strings.xml:

```
<resources>
    <string name="app_name">Погодный Информер</string>
    <string name="logo">Погода в городе</string>
    <string name="cityNameText">введите название города</string>
    <string name="getWeatherText">УЗНАТЬ ПОГОДУ</string>
    <string name="enterCity">Введите название города!</string>
    <string name="pleaseWait">Ожидайте...</string>
</resources>
```

#### Листинг файла цветов colors.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="ColorPrimary">#333</color>
    <color name="ColorPrimaryDark">#333</color>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
    <color name="TextColor">#BBB</color>
</resources>
```

#### Листинг файла темы по умолчанию themes.xml:

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.WeatherApp"
        parent="Theme.MaterialComponents.DayNight.NoActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/ColorPrimary</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_700</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor">@color/ColorPrimary</item>
        <!-- Customize your theme here. -->
    </style>
</resources>
```

## Листинг файла MainActivity.java:

```
package com.weatherapp;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.view.inputmethod.EditorInfo;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;

public class MainActivity extends AppCompatActivity {

    //Объявление объектов макета, соответствующих его виджетам
    private EditText cityName; //для наименования города
    private Button btnGetWeather; //кнопка для отправки запроса
    private TextView resultInfo; //здесь будет ответ
    private TextView wait; //место для текста «подождите...»

    String URL;
    //Объявление функции, исполняемой в отдельном потоке
    Runnable GetWeather;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        cityName = findViewById(R.id.cityName);
        btnGetWeather = findViewById(R.id.btnGetWeather);
        resultInfo = findViewById(R.id.resultInfo);

        btnGetWeather.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                if (cityName.getText().toString().trim().equals("")) {
                    Toast.makeText(MainActivity.this, R.string.enterCity,
                        Toast.LENGTH_LONG).show();
                } else {

                    String city = cityName.getText().toString();
                    String apiKey = "[Ваш ключ к API]";
                    URL = "https://api.openweathermap.org/data/2.5/weather?q=" + city
                        + "&appid=" + apiKey + "&units=metric&lang=ru";

                    Thread Process = new Thread(GetWeather); //создание потока
                    //Process.setPriority(Thread.MIN_PRIORITY); //приоритет процесса

                    resultInfo.setText(R.string.pleaseWait);
                }
            }
        });
    }
}
```

```

        Process.start();//запуск потока
    }
}
});

GetWeather = new Runnable() {
String request = "";
@Override
public void run() {
    HttpURLConnection connection = null;
    StringBuffer buffer = new StringBuffer();
    try {
        URL url = new URL(URL);
        connection = (HttpURLConnection) url.openConnection();
        connection.connect();
        InputStream stream = connection.getInputStream();
        String line = "";
        request = "";
        BufferedReader reader = new BufferedReader(new
InputStreamReader(stream));
        while ((line = reader.readLine()) != null) {
            request += line;
        }
        stream.close();
        reader.close();
    } catch (MalformedURLException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
        request = "";
    } finally {
        connection.disconnect();
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                PrintResult(request);
            }
        });
    }
}
};

//функция для обработки ответа в JSON и вывода результата
private void PrintResult(String result){

    try {
        JSONObject obj = new JSONObject(result);
        //для массива объектов под одним ключом «weather»
        JSONArray weather = obj.getJSONArray("weather");
        //для одиночных объектов
        JSONObject main = obj.getJSONObject("main");
        JSONObject wind = obj.getJSONObject("wind");
        String res = "";
        //обращение к нулевому элементу массива «weather» и к его подэлементу
        «description»
        String desc = weather.getJSONObject(0).getString("description");
        //собираем строку res конкатенацией
        res =
            desc+"\n"+
            "Температура воздуха: "+main.getDouble("temp")+" °C\n"+
            "(ощущается как "+main.getDouble("feels_like")+" °C)\n"+

```

```

        "Атмосферное давление: "+main.getInt("pressure")+" мм.рт.ст.\n"+
        "Влажность: "+main.getInt("humidity")+"%\n"+
        "Скорость ветра: "+wind.getDouble("speed")+" м/с\n";
//выводим данную строку в виде текста в виджет «resultInfo»
resultInfo.setText(res);

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

```

Снимок экрана главной активности изображен на рисунке 21.

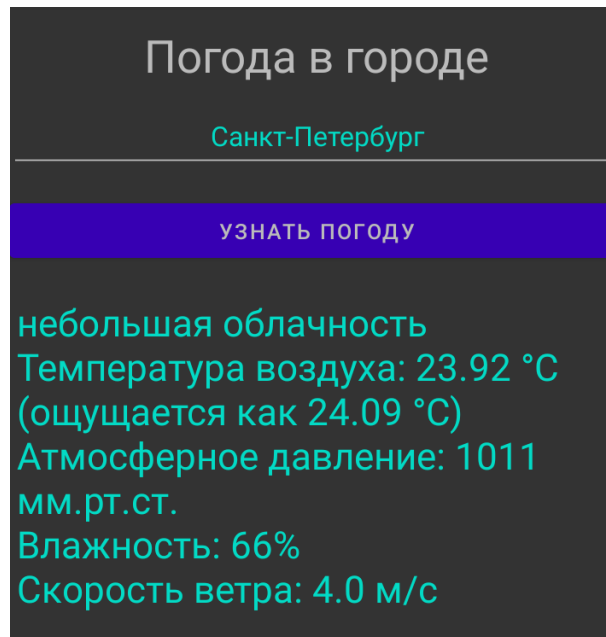


Рисунок 21 – Снимок экрана главной активности

При нажатии на кнопку «УЗНАТЬ ПОГОДУ» начинает выполняться отдельный параллельный поток, не блокирующий главный поток. В этом потоке происходит обращение к сервисной функции удаленного веб-сервера, прием ответа, его разбор (парсинг), сборка ответа, изменение свойства Text виджета для визуализации ответа на дисплее.

OpenWeatherMap предоставляет бесплатно 100 запросов в сутки. Возвращаемый JSON-пакет при этом содержит гораздо больше сведений о погодных условиях в указанном городе. Для детального изучения структуры возвращаемых OpenWeatherMap API данных следует обратиться на официальный сайт <https://openweathermap.org>.



**Пример 4.2** Мобильное приложение «QuadEq» для решения квадратного уравнения (с использованием собственного внешнего веб-сервиса).

Сначала приводится описание серверного кода, исполняющегося веб-сервером Apache или Nginx. Серверный код в данном случае сформирован в одном файле \*.srv и выполняет роль Модели, Представления и Контроллера одновременно, т.е. производит вычисления, формирует JSON-ответ, принимает и проверяет запрос от приложения-клиента и выдает ответ.

Листинг серверного кода srv.php представлен ниже.

```
<?php

//извлекаем из посылки GET в переменные значения параметров
$a=$_GET['a'];
$b=$_GET['b'];
$c=$_GET['c'];

//проверка заполнения, формата и диапазона варьирования входных данных
$result = "";
if(!is_numeric($a)) $result = "введите коэффициент a"; else
    if($a==0) $result = "введите a <> 0";
if(!is_numeric($b))
    if($result=="") $result = "введите коэффициент b"; else $result .=
PHP_EOL."введите коэффициент b";
if(!is_numeric($c))
    if($result=="")
        $result = "введите коэффициент c"; else $result .= PHP_EOL."введите
коэффициент c";

//если все хорошо, выполняем вычисления с записью результата в $result
if($result==""){

    $D = $b*$b-4*$a*$c;
    $X1 = null;
    $X2 = null;

    if($D<0) $result = "корней нет"; else {
        if($D==0){
            $X1 = -$b/(2*$a);
            $result = "X1=".round((float)$X1, 3);
        } else {
            $X1 = (-$b-sqrt($D))/(2*$a);
            $X2 = (-$b+sqrt($D))/2*$a;
            $result = "X1=".round((float)$X1, 3).", X2=".round((float)$X2,
3);
        }
    }
}
//сборка массива именованных элементов с их значениями
$data = [
    "mes" => $result
];

//указываем в заголовке возвращаемого документа его тип (json) и кодировку
header('Content-Type: application/json; charset=utf-8');
//вывод json-массива с измененной кодировкой в юникод
// (для минимизации объема данных и отладки)
echo json_encode($data,JSON_UNESCAPED_UNICODE);

?>
```

### Листинг файла манифеста приложения AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.QuadEq"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

### Листинг файла строковых констант strings.xml:

```
<resources>
    <string name="app_name">QuadEq</string>
    <string name="URL">https://myweb service.ru/srv.php</string>
    <string name="Coeff_a">коэффициент a</string>
    <string name="Coeff_b">коэффициент b</string>
    <string name="Coeff_c">коэффициент c</string>
    <string name="Solve">Решить</string>
    <string name="Result">результат</string>
</resources>
```

### Листинг файла цветов colors.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
</resources>
```

### Листинг файла темы themes.xml:

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Base.Theme.QuadEq"
        parent="Theme.Material3.DayNight.NoActionBar">
        <!-- Customize your light theme here. -->
        <!-- <item name="colorPrimary">@color/my_light_primary</item> -->
    </style>
```

```
    <style name="Theme.QuadEq" parent="Base.Theme.QuadEq" />
</resources>
```

Листинг файла разметки макета экрана главной активности  
activity\_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity">

    <EditText
        android:id="@+id/ca"
        android:layout_width="349dp"
        android:layout_height="56dp"
        android:layout_marginTop="28dp"
        android:ems="10"
        android:hint="@string/Coeff_a"
        android:inputType="numberSigned|numberDecimal"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/cb"
        android:layout_width="346dp"
        android:layout_height="54dp"
        android:layout_marginTop="8dp"
        android:ems="10"
        android:hint="@string/Coeff_b"
        android:inputType="numberSigned|numberDecimal"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.476"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/ca" />

    <EditText
        android:id="@+id/cc"
        android:layout_width="346dp"
        android:layout_height="52dp"
        android:layout_marginTop="8dp"
        android:ems="10"
        android:hint="@string/Coeff_c"
        android:inputType="numberSigned|numberDecimal"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.492"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/cb" />

    <Button
        android:id="@+id/button"
        android:layout_width="137dp"
        android:layout_height="60dp"
        android:layout_marginTop="28dp"
        android:text="@string/Solve"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/cc" />

    <TextView
        android:id="@+id/result"
        android:layout_width="313dp"
```

```

        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="@string/Result"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/button"
        android:textAppearance="?android:attr/textAppearanceLarge"
        app:layout_constraintVertical_bias="0.078" />
</androidx.constraintlayout.widget.ConstraintLayout>

```

Макет экрана главной активности представлен на рисунке 22.

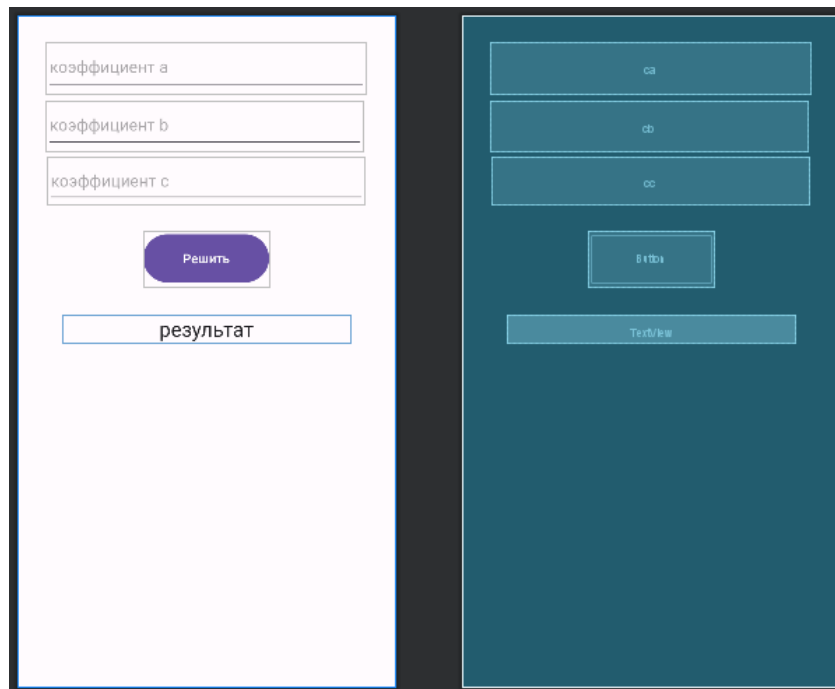


Рисунок 22 – Макет экрана главной активности

Описание виджетов:

- ca – текстовое поле для ввода коэффициента a;
- cb – текстовое поле для ввода коэффициента b;
- cc – текстовое поле для ввода коэффициента c;
- button – кнопка для выполнения HTTPS-запроса;
- result – текстовая надпись для вывода результата HTTPS-запроса.

Листинг управляющего кода MainActivity.java представлен ниже.

```

package com.quadeq;

import androidx.appcompat.app.AppCompatActivity;

import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.os.Bundle;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;

```

```

import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.net.URL;
import org.json.JSONException;
import org.json.JSONObject;
import javax.net.ssl.HttpURLConnection;

public class MainActivity extends AppCompatActivity {

    private EditText ca;
    private EditText cb;
    private EditText cc;
    private Button btnGetSolve;
    private TextView resultInfo;

    Runnable GetSolve; //будет выполняться в отдельном потоке
    String URL;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        ca = (EditText) findViewById(R.id.ca);
        cb = (EditText) findViewById(R.id.cb);
        cc = (EditText) findViewById(R.id.cc);
        btnGetSolve = (Button) findViewById(R.id.button);
        resultInfo = (TextView) findViewById(R.id.result);

        btnGetSolve.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                String coeffa = ca.getText().toString();
                String coeffb = cb.getText().toString();
                String coeffc = cc.getText().toString();
                //построение GET-запроса
                URL =getString(R.string.URL)+"?a=" + coeffa + "&b=" + coeffb
+ "&c="+coeffc;
                //создание нового потока для функции GetSolve()
                Thread Process = new Thread(GetSolve);
                //запуск потока на исполнение
                Process.start();

            }
        });

        GetSolve = new Runnable() {
            String request = "";
            @Override
            public void run() {
                //создание нового HTTPS-соединения
                HttpURLConnection connection = null;
                //создание строкового буфера для приема ответа от сервера
                StringBuffer buffer = new StringBuffer();
                try {
                    //создание новой ссылки с текстом GET-запроса
                    URL url = new URL(URL);
                    //открытие соединения
                    connection = (HttpURLConnection) url.openConnection();
                    connection.connect();
                    //запись входного потока в stream
                    InputStream stream = connection.getInputStream();
                    //поток читается по строкам, создаем строку line

```

```

        String line = "";
        //инициализируем строковый ответ
        request = "";
        BufferedReader reader = new BufferedReader(new
InputStreamReader(stream));
        //построчно через читателя входного потока reader пишем в
request ответ
        while ((line = reader.readLine()) != null) {
            request += line;
        }
        stream.close();
        reader.close();
    } catch (MalformedURLException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
        request = "";
    } finally {
        //не забываем закрыть соединение после HTTPS-запроса
        connection.disconnect();
        //метод, который выполняется в главном потоке (UI-потоке)
        runOnUiThread(new Runnable() {
            @Override
            //переопределение метода run для парсинга ответа
            //и вывода его в виджет в главном потоке
            public void run() {
                try {
                    JSONObject jsonObject = new JSONObject(request);
                    resultInfo.setText(jsonObject.getString("mes"));
                } catch (JSONException e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

};

}

}

```

При запуске мобильного приложения появляется экран главной активности (рисунок 23).

коэффициент а

коэффициент b

коэффициент с

Решить

результат

Рисунок 23 – Снимок экрана главной активности

При вводе неполных данных и нажатии на кнопку «Решить» выполняется HTTPS-запрос. Обработка исключительных ситуаций теперь производится на веб-сервере. При отсутствии значения коэффициента «с» ответ будет следующим (рисунок 24).

2

1|

коэффициент с

Решить

введите коэффициент с

Рисунок 24 – Неполные данные

В соответствии со спецификой решаемой вычислительной задачей, коэффициент «а» не должен быть  $= 0$ , что и проверяется (рисунок 25).



0

1.1

-5

Решить

введите a <> 0

Рисунок 25 – Коэффициент  $a=0$

Другие возможные нормальные ситуации представлены на рисунке 26.

2

1

3

Решить

корней нет

2

1.1

-5

Решить

$X1=-1.88, X2=5.32$

Рисунок 26 – Ответы от веб-сервера (нормальный режим работы)

**Пример 5** Гибридное мобильное приложение для расчета корней квадратного уравнения «QuadEq2».

На веб-сервере размещается html-страница для загрузки в интегрированный в мобильное приложение веб-браузер (виджет WebView). Код html-страницы (index.html) представлен ниже.

```
<html>
<head>
<!--установка кодировки страницы по умолчанию-->
<meta charset="utf-8">
<title>Пример</title>
<!--подключение файлов ресурсов-->
<link rel="stylesheet" href="style.css" type="text/css">
<script src="jquery-3.2.1.min.js" type="text/javascript"></script>
<script src="myscript.js" type="text/javascript"></script>
</head>

<body>

<form id="myform">
<!--веб-форма-->
<label for="ca">коэффициент a <input id="ca" type="number"
/></label><br/><br/>
<label for="cb">коэффициент b <input id="cb" type="number"
/></label><br/><br/>
<label for="cc">коэффициент c <input id="cc" type="number"
/></label><br/><br/>
<input id="btn" type="button" value="Решить" onclick="send();">
</form>
<!--блок для отображения результата вычислений-->
<div id="result">результат ???</div>

</body>
</html>
```

jquery-3.2.1.min.js – JS-библиотека для взаимодействия с DOM-элементами, манипулирования ими. Также библиотека jQuery предоставляет удобный API для работы с AJAX.

Для оформления html-страницы использовался файл каскадных стилей style.css, содержимое которого представлено ниже.

```
body{
  font-family: Tahoma; /*шрифт для всего документа*/
  font-size: 18pt; /*размер шрифта для всего документа*/
  margin-top: 50px; /*внешний отступ сверху*/
}

*, *::before, *::after {
  box-sizing: border-box; /*метод управления размерами элемента*/
}

input[type="number"] {
  font-family: inherit; /*наследование из родительского элемента*/
  font-size: inherit; /*размер шрифта*/
  line-height: inherit; /*высота строки с расположением элемента*/
  margin: 2px; /*внешний отступ*/
  display: block; /*отображение элемента блоком (без обтекания)*/
  width: 99%;
```

```

height: 120px;
padding: 0.375rem 0.75rem; /*rem - root em (по родителю)*/
font-family: inherit;
font-size: 4rem;
font-weight: 400; /*насыщенность шрифта*/
line-height: 1.5px;
color: #212529;
background-color: #fff; /*цвет фона - белый*/
background-clip: padding-box; /*метод вывода фона элемента под границами*/
border: 1px solid #bdbdbd; /*вид границы*/
border-radius: 0.25rem; /*радиус скругления углов элемента*/
transition: 0.15s ease-in-out, box-shadow 0.15s ease-in-out; /*вид
динамического эффекта при любых изменениях в элементе*/
}

input[type="number"]:focus {
  color: #212529;
  border-color: #1867D4;
  outline: 0; /*внешняя граница вокруг элемента*/
  box-shadow: 0 0 0 0.2rem #0E6DF5; /*тень от элемента*/
}

label{
  font-family: inherit;
  font-size: 4rem;
  font-weight: 400;
  color: #212529;
}

input[type="button"] {
  background-color: #83C0EF;
  border-style: none;
  border-width: 0px;
  height: 150px;
  width: 80%;
  padding: 20px;
  color: initial;
  display: block;
  margin: 0 auto;
  text-align: center;
  font-family: inherit;
  font-size: 4rem;
  font-weight: 400;
  transition: 0.15s ease-in-out, box-shadow 0.15s ease-in-out;
}

input[type="button"]:hover{
  background-color: #8080C0;
  color: #fff;
}

input::-webkit-outer-spin-button,
input::-webkit-inner-spin-button {
  /*убираем стрелки у числовых полей*/
  -webkit-appearance: none;
  margin: 0;
}

#result{
  width: 100%;
  margin: 60px 5px 5px 5px;
  font-size: 4rem;
  text-align: center; /*выравнивание содержимого блока по центру*/
}

```

Код JavaScript (в файле myscript.js) содержит только одну функцию send() для отправки GET-запроса с данными на обработку. Ответ придет в содержимое блока #result.

```
function send(){
    var ca = $("#ca").val();/* значение коэффициента a из DOM-элемента */
    var cb = $("#cb").val();/* значение коэффициента b из DOM-элемента */
    var cc = $("#cc").val();/* значение коэффициента c из DOM-элемента */
    $.ajax({
        url: 'srv.php', /* куда пойдет запрос */
        method: 'get', /* метод передачи (post или get) */
        dataType: 'json', /* тип данных в ответе (xml, json, script, html). */
        data: {a:ca,b:cb,c:cc}, /* параметры передаваемые в запросе. */
        success: function(data){ /* функция которая будет выполнена после
успешного запроса. */
            var result = data.mes.replace(/\n/g, "<br />"); /* извлечение из
ответа значения строки mes */
            $("#result").html(result); /* и его подстановка в блок result */
        },
        error: function() { /* блок error выполняется после времени таймаута при
неудачной попытке */
            $("#result").html("ошибка!!!");
        }
    });
}
```

В итоге, на веб-сервере должны быть расположены следующие файлы:

- index.html;
- style.css;
- jquery-3.2.1.min.js;
- myscript.js;
- srv.php.

Листинг файла srv.php, принимающего GET-запросы, проверяющего входные данные и отправляющего ответ, представлен в примере 4.2.

Клиентом при такой архитектуре приложения может быть обычный веб-браузер или часть страницы стороннего веб-ресурса. Портирование приложения-клиента на различные мобильные платформы не вызывает особых сложностей, т.к. управляющий код и макеты активностей при этом сильно упрощаются.

Для разнообразия в приложение включены три кнопки, формирующие панель инструментов. При нажатии на первую кнопку появляется пуш-уведомление.

Рассмотрим структуру проекта, ресурсы и управляющий java-код приложения-клиента.

Листинг файла манифеста AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.POST_NOTIFICATIONS" />

    <application
        android:allowBackup="true"
```

```

        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name2"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"

        android:theme="@style/Theme.QuadEq2"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>

            <meta-data
                android:name="android.app.lib_name"
                android:value="" />
            </activity>
        </application>

    </manifest>

```

### Листинг файла строковых констант strings.xml:

```

<resources>
    <string name="app_name">Решение квадратного уравнения</string>
    <string name="app_name2">QuadEq2</string>
    <string name="API">https://myweb service.ru/srv.php</string>
    <string name="channel_id">CHID</string>
    <string name="notification_title">Заголовок уведомления</string>
    <string name="notification_text">Текст уведомления</string>
    <string name="ChannelDesc">Описание канала</string>
    <string name="ChannelName">Мои уведомления</string>
    <string name="Button1">Кнопка 1</string>
    <string name="Button2">Кнопка 2</string>
    <string name="Button3">Кнопка 3</string>
</resources>

```

### Листинг файла themes.xml:

```

<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.QuadEq2"
        parent="Theme.MaterialComponents.DayNight.NoActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_500</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_700</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>
</resources>

```

## Листинг файла макета экрана главной активности activity\_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <androidx.appcompat.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="409dp"
        android:layout_height="50dp"
        android:background="?attr/colorPrimary"
        android:minHeight="?attr/actionBarSize"
        android:theme="?attr/actionBarTheme"
        app:layout_constraintTop_toTopOf="parent"
        tools:layout_editor_absoluteX="1dp"
        tools:layout_editor_absoluteY="0dp"/>

    <WebView
        android:id="@+id/MainWV"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:layout_constraintTop_toBottomOf="@id/toolbar"
        android:layout_marginTop="50dp"
        tools:layout_editor_absoluteX="1dp" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="50dp"
        android:layout_marginStart="12dp"
        android:layout_marginRight="12dp"
        android:textColor="@color/white"
        android:onClick="onMyButtonClick"
        android:text="@string/Button1"
        app:layout_constraintStart_toStartOf="@+id/toolbar"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0" />

    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="50dp"
        android:layout_marginStart="12dp"
        android:layout_marginRight="12dp"
        android:textColor="@color/white"
        android:text="@string/Button2"
        app:layout_constraintStart_toEndOf="@+id/button1"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0" />

    <Button
        android:id="@+id/button3"
        android:layout_width="wrap_content"
        android:layout_height="50dp"
        android:layout_marginStart="12dp"
        android:layout_marginRight="12dp"
        android:textColor="@color/white"
        android:text="@string/Button3"
```

```

        app:layout_constraintStart_toEndOf="@+id/button2"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0" />

</androidx.constraintlayout.widget.ConstraintLayout>

```

Внешний вид макета экрана главной активности в режиме дизайна изображен на рисунке 27.

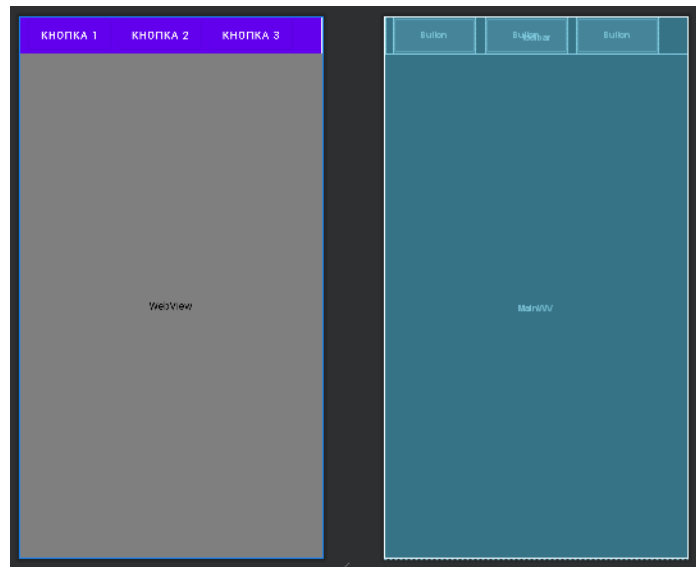


Рисунок 27 – Внешний вид макета экрана главной активности

Описание виджетов:

- toolbar – панель инструментов для кнопок;
- MainWV – встроенный веб-браузер для отображения HTML-страницы;
- button1 – кнопка для вывода пуш-уведомления;
- button2 – кнопка без назначения действия;
- button3 – кнопка без назначения действия.

Листинг управляющего кода MainActivity.java:

```

package com.example.quadeq2;

import androidx.appcompat.app.AppCompatActivity;

import android.app.Notification;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.content.Context;
import android.os.Build;
import android.os.Bundle;
import android.view.View;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.app.Activity;
import android.content.Intent;
import android.net.Uri;

import androidx.core.app.NotificationCompat;

```

```

public class MainActivity extends AppCompatActivity {

    private WebView myWebView;
    private class SimpleWebViewClient extends WebViewClient {
        private Activity activity = null;
        @Override
        //проверяем возможность встраивания веб-страницы в контент приложения
        public boolean shouldOverrideUrlLoading(WebView webView, String url)
        {
            if (url.contains(getString(R.string.API))) {
                return false;
            }
            Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse(url));
            activity.startActivity(intent);
            return true;
        }
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        WebView myWebView = findViewById(R.id.MainWV);

        WebSettings webSettings = myWebView.getSettings();
        //позволяем JavaScript
        webSettings.setJavaScriptEnabled(true);
        //webSettings.setCacheMode(2);
        //myWebView.getSettings().setLoadsImagesAutomatically(true);
        //запрет на масштабирование страницы встроенными и дополнительными
        элементами управления
        myWebView.getSettings().setBuiltInZoomControls(false);
        webSettings.setDisplayZoomControls(false);
        //myWebView.getSettings().setLoadWithOverviewMode(true);
        //загрузка видимой области страницы по умолчанию (как в десктопном
        браузере, "альбомный" режим)
        webSettings.setUseWideViewPort(true);
        //загрузка веб-страницы без полос прокрутки, т.е. с полностью видимым
        содержимым
        webSettings.setLoadWithOverviewMode(true);
        SimpleWebViewClient webViewClient = new SimpleWebViewClient();
        myWebView.setWebViewClient(webViewClient);
        myWebView.loadUrl(getString(R.string.API));
    }

    public void onMyButtonClick(View view)
    {
        //ID уведомления
        int NOTIFY_ID = 1;
        //ID канала уведомлений
        String CHANNEL_ID = getString(R.string.channel_id);
        //показываем уведомление при нажатии на кнопку 1
        //настройка менеджера уведомлений
        NotificationManager notificationManager = (NotificationManager)
        getSystemService(Context.NOTIFICATION_SERVICE);
        String NOTIFICATION_CHANNEL_ID = CHANNEL_ID;
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            //для новых версий SDK настройка канала уведомлений
            NotificationChannel notificationChannel = new
            NotificationChannel(NOTIFICATION_CHANNEL_ID, getString(R.string.ChannelName),
            NotificationManager.IMPORTANCE_MAX);

            notificationChannel.setDescription(getString(R.string.ChannelDesc));

```



```

        notificationChannel.enableLights(true);

notificationChannel.setLightColor(android.R.color.holo_green_light);
        notificationChannel.setVibrationPattern(new long[]{0, 1000, 500,
1000});
        notificationChannel.enableVibration(true);
    }

    //создание конструктора уведомлений
    NotificationCompat.Builder notificationBuilder = new
NotificationCompat.Builder(this, NOTIFICATION_CHANNEL_ID);

    //настройка уведомления
    notificationBuilder.setAutoCancel(true)
        //установка настроек по умолчанию, которые приняты для всех
уведомлений
        .setDefaults(Notification.DEFAULT_ALL)
        .setSmallIcon(R.drawable.forward)

    //.setTicker(getString(getString(R.string.notification_text))//до lolipop
        .setPriority(Notification.PRIORITY_DEFAULT)
        .setContentTitle(getString(R.string.notification_title))
        .setContentText(getString(R.string.notification_text))
        .setAutoCancel(true);//для автозакрытия при выборе
уведомления
        //.setContentInfo("Инфо");

    //показываем уведомление
    notificationManager.notify(NOTIFY_ID, notificationBuilder.build());
}
}

```

При запуске приложения появляется экран главной активности (рисунок 28), в веб-элементы загруженной в виджет WebView страницы пользователь вводит значения коэффициентов квадратного уравнения.

кнопка 1	кнопка 2	кнопка 3	кнопка 1	кнопка 2	кнопка 3
коэффициент а <input type="text" value="2.6"/>			коэффициент а <input type="text" value="2.6"/>		
коэффициент b <input type="text" value="-12.8"/>			коэффициент b <input type="text" value="3.1"/>		
коэффициент с <input type="text" value="4"/>			коэффициент с <input type="text" value="9"/>		
<input type="button" value="Решить"/>			<input type="button" value="Решить"/>		
X1=0.335, X2=31.013			корней нет		

Рисунок 28 – Работа приложения в обычном режиме

Также как и в предыдущем примере, проверку входных данных выполняет веб-сервер. При вводе неполных или некорректных данных пользователь получит сообщение об ошибке (рисунок 29).

кнопка 1

кнопка 2

кнопка 3

коэффициент a

0

коэффициент b

3.1

коэффициент c

Решить

введите  $a \neq 0$

введите коэффициент c

Рисунок 29 – Результат обработки исключений