УДК 004

Создание многооконного приложения для Android: жизненный цикл активностей и передача данных между ними

Эрдман Александр Алексеевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В статье рассмотрен процесс создание мобильного приложения для Android, а также работа с активностями: жизненный цикл и передача данных между активностями. Приложение реализовано на языке программирования Java в IDE Android Studio. Результатом исследование будет являться мобильное приложение с несколькими активностями, а также их подробное описание.
Ключевые слова: Java, Android, мобильное приложение

Название статьи на английском языке

Erdman Alexander Alekseevich
Sholom-Aleichem Priamursky State University
Student

Abstract
The article discusses the process of creating a mobile application for Android, as well as working with activities: the lifecycle and data transfer between activities. The application is implemented in the Java programming language in the Android Studio IDE. The result of the study will be a mobile application with several activities, as well as their detailed description.
Keywords: Java, Android, Mobile app

1 Введение

1.1 Актуальность

Мобильные приложения становятся популярнее с каждым днём, так как у большинства людей имеется мобильное устройство будь то смартфон или планшет. По статистике видно, что наличие смартфона преобладает наличие компьютера у обычных людей. Местами смартфоны даже вытесняют компьютеры – на телефоны давно уже выпускают не только игры, но и даже профессиональные программы, такие как редакторы видео и аудио, редакторы изображений, существуют даже IDE для смартфонов, в которых можно разрабатывать полноценное ПО. В связи с такой популярностью смартфонов, не малая доля разработчиков уходит в сферу мобильной разработки, так как это в первую очередь очень выгодная отрасль, так как охватывает большой круг потребителей. Зачастую приложения создаются не простыми, а сложными – то есть приложения имеют не одно окно

приложения, а несколько, которые при этом активно передают данные между собой. Так как подавляющее большинство приложений содержит более одного окна, то очень важно понимать, как это работает и как устроены такие приложения.

1.2 Обзор исследований

Р.В. Семченко и П.А. Еровлев описали разработку приложений и игр на Android-смартфоны в среде разработки «Android Studio» [1]. М.В. Попова и Т.Р. Лысак рассмотрели основные этапы разработки приложений в AndroidStudio [2]. С.Ю. Михайлов и С.О. Иванов описали процесс разработке мобильного приложения для системы Android [3]. Р.В. Наумов рассмотрел базовые принципы android программирования: активности, состояния, пути активности [4]. В.И. Макаров проанализировал один из методов получения доступа к информации о звонках и смс приходящих на устройство работающее под операционной системой Android [5]. И.Д. Тухватуллин рассмотрел причины популяризации Android-приложений [6]. О.В. Кобзев и Л.Ю. Забелин провели анализ операционной системы Android [7].

1.3 Цель исследования

Целью исследования является создание приложения с несколькими активностями для Android систем, а также описание активностей.

2 Материалы и методы

Для создания приложения использовался язык программирования Java. В качестве IDE выбрана Android Studio.

3 Результаты и обсуждения

В качестве приложения с несколькими активностями будет выбрано приложение, представляющее собой викторину, где пользователю нужно выбирать ответ на случайно генерирующийся математический пример в течение определённого времени. По истечению времени будет представлен результат пользователя.

Для начала разработки необходимо создать проект в IDE Android Studio. При создании проекта создаётся главная активность и макет: MainActivity.java и activity_main.xml.

Приложение состоит из экранов. Каждый экран в свою очередь состоит из активности и макета. Активность – это одна чётко определённая операция, которую может выполнять пользователь. Активности ассоциируется с одним экраном и программируются на языке программирования Java. Макет описывает внешний вид экрана. Макеты создаются в формате .xml и сообщают Android, где размещены те или иные объекты экрана.

После создания проекта с активностью и макетом, в макете размешаются визуальные элементы – TextView. TextView – это визуальный элемент, который используется для вывода текста. Создаются семь текстовых элементов. Два таких элемента отвечают за вывод: счётчика правильных

ответов и вопросов; вывод таймера. Один TextView используется для вывода вопроса. Оставшиеся четыре будут выводить варианты ответов. С помощью параметра color background настраивается цвет заднего фона элементов. Расположение элементов редактируется визуально – с помощью перетаскивание элемента мышью. Макет будущего приложения выглядит следующим образом (рис. 1).

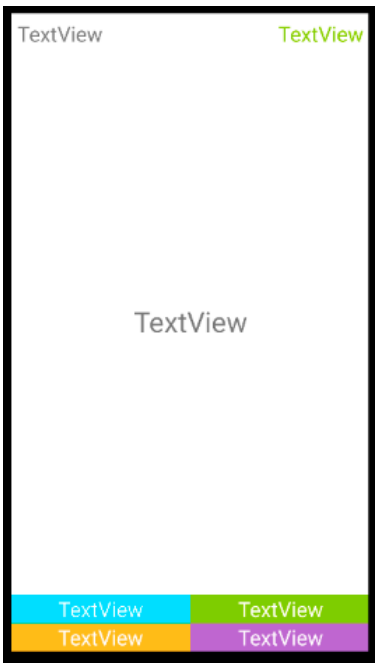

Рисунок 1. Внешний вид приложения

После визуального редактирования макета, в главной активности MainActivity.java создаются переменные private TextView для каждого элемента TextView. С помощью метода findViewById(R.id.) идёт инициализация переменных – переменным присваивается id конкретного элемента TextView (рис. 2).

```
private TextView textViewOpinion0;
private TextView textViewOpinion1;
private TextView textViewOpinion2;
private TextView textViewOpinion3;
textViewTimer = findViewById(R.id.textViewTimer);
textViewOpinion0 = findViewById(R.id.textViewOpinion0);
textViewOpinion1 = findViewById(R.id.textViewOpinion1);
textViewOpinion2 = findViewById(R.id.textViewOpinion2);
textViewOpinion3 = findViewById(R.id.textViewOpinion3);
textViewQuestion = findViewById(R.id.textViewQuestion);
textViewScore = findViewById(R.id.textViewScore);
```

Рисунок 2. Создание и инициализация переменных

Инициализация текстовых элементов закончена. Добавляются ещё переменные, которые будут использоваться в генерации вопросов – переменная вопроса, ответа и позиции правильного ответа, булевая переменная будет хранить знак выражения, переменные максимального значения и минимального, которые отвечают за диапазон генерации чисел:

```
private String question;
private int rightAnswer;
private int rightAnswerPosition;
private boolean isPositive;
private int min = 5;
private int max = 30;
```

Далее создаётся метод, который будет генерировать вопрос (рис. 3). В данном методе используются методы Match.random для создание случайных величин. Условие if () отвечает за вывод на экран вопроса. Вопрос может содержать как знак сложения, так и знак вычитания. Для того, чтобы выводился верный знак выражения, условие проверяет: если переменные a и b складываются, то в вопросе для пользователя будет выводиться два числа со знаком сложение. Если же переменные a и b вычитаются, то в вопросе будет указан знак вычитания. После условия следует метод setText(), с помощью которого в элемент TextView будет передаваться строка с вопросом. После следует непосредственная случайная генерация позиции правильного ответа с помощью метода Math.random.

```
private void generateQuestion() {
    int a = (int) (Math.random() * (max - min + 1) + min);
    int b = (int) (Math.random() * (max - min + 1) + min);
    int mark = (int) (Math.random() * 2);
    isPositive = mark == 1;
    if (isPositive) {
        rightAnswer = a + b;
        question = String.format("%s + %s", a, b);
    } else {
        rightAnswer = a - b;
        question = String.format("%s - %s", a, b);
    }
    textViewQuestion.setText(question);
    rightAnswerPosition = (int) (Math.random() * 4);
}
```

Рисунок 3. Метод генерации вопроса

Второй метод генерирует неправильные ответы и возвращает их в варианты выбора ответа (рис. 4).

```
private int generateWrongAnswer() {
    int result;
    do {
        result = (int) (Math.random() * max * 2 + 1) - (max - min);
    } while (result == rightAnswer);
    return result;
}
```

Рисунок 4. Метод шенерации неправильных ответов

Создаём метод нажатия на кнопки android:onClick="Answer" ответов и добавляем его ко всем кнопкам в коде файла activity_main.xml. Далее

реализуется метод playNext, который отвечает за начало генерации вопросов (вызывает метод генерации вопроса) и набора очков (рис. 5). В данном методе идёт проверка – если был выбран элемент с правильным ответом, то он передаётся в виде строки в строку с очками. В строке String score идёт прибавление очков, если был выбран правильный ответ, а также ведётся счётчик пройденных вопросов. Очки отображаются с помощью метода setText() в соответствующем textViewScore.

```
private void playNext() {
    generateQuestion();
    for (int i = 0; i < options.size(); i++) {
        if (i == rightAnswerPosition) {
            options.get(i).setText(Integer.toString(rightAnswer));
        } else {
            options.get(i).setText(Integer.toString(generateWrongAnswer()));
        }
    }
    String score = String.format("%s / %s", countOfRightAnswers, countOfQuestions);
    textViewScore.setText(score);
}
```

Рисунок 5. Метод PlayNext

Реализация метода нажатия на кнопку (рис. 6). Данный метод представлен получением и передачей вопроса в элемент TextView, проверкой полученного ответа от пользователя (нажатия на ответ), вывод всплывающей подсказки – верно или неверно, а также повторный запуск метода playNext.

```
public void onClickAnswer(View view) {
    if (!gameOver) {
        TextView textView = (TextView) view;
        String answer = textView.getText().toString();
        int chosenAnswer = Integer.parseInt(answer);
        if (chosenAnswer == rightAnswer) {
            countOfRightAnswers++;
            Toast.makeText( context: this,  text: "Верно", Toast.LENGTH_SHORT).show();
        } else {
            Toast.makeText( context: this,  text: "Неверно", Toast.LENGTH_SHORT).show();
        }
        countOfQuestions++;
        playNext();
    }
}
```

Рисунок 6. Метож нажатия на вариант ответа

У активностей есть жизненный цикл - это означает, что она может быть в одном из различных стадий, в зависимости от того, что происходит с приложением при действиях пользователя. Например, при переходе от одного экрана приложения к другому могут происходить несколько событий с активностями экранов – остановка и завершение, пауза или фоновая работа. В реализуемом приложение рассмотрен случай, когда одна активность завершает свои операции, передаёт данные в следующую активность и останавливается.

Создаётся новая активность – ScoreActivity.java, в которой будет показываться результат текущий и лучший, а также повторный запуск приложения. В данной активности добавляются два элемента – TextView и Button. В первый будет выводиться результат, а второй позволит запустить приложение заново. Код новой активности представляет из себя получение Intent (объект описания операции, которую необходимо выполнить через систему Android) от предыдущей активности, из которой он принимает результаты (рис. 7). В методе onCreate осуществляется инициализация элемента TextView. Далее идёт работа с Intent - с помощью метода get получается Intent главной активности. Далее идёт получение данных очков. Получаются текущие очки пользователя. Далее идёт проверка – если текущий ответ больше, максимального, то максимальный обновляется. Если же текущий результат меньше или равен максимальному, то изменения максимального не происходят. Через метод setText элемент TextView получает данные результата и выводит его. Также создаётся метод кнопки, который позволяет перейти снова к главной активности:

```
public void onClickStartNewGame(View view) {
    Intent intent = new Intent(this, MainActivity.class);
    startActivity(intent);
}
```

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_score);
    textViewResult = findViewById(R.id.textViewResult);
    Intent intent = getIntent();
    if (intent != null && intent.hasExtra( name: "result")) {
        int result = intent.getIntExtra( name: "result",  defaultValue: 0);
        SharedPreferences preferences = PreferenceManager.getDefaultSharedPreferences( context: this);
        int max = preferences.getInt( s: "max",  i: 0);
        String score = String.format("Ваш результат: %s\nМаксимальный результат: %s", result, max);
        textViewResult.setText(score);
    }
}
```

Рисунок 7. Метод вывода статистики игры

В первой активности создаётся метод перехода между активностями по истечению таймера (рис. 8). Приницип следуюший – если значение таймера равно нулю, то через Intent идёт переход в следующую активность, а именно в активность с результатом.

```
public void onFinish() {
    gameOver = true;
    SharedPreferences preferences = PreferenceManager.getDefaultSharedPreferences(getApplicationContext());
    int max = preferences.getInt( s: "max",  i: 0);
    if (countOfRightAnswers >= max) {
        preferences.edit().putInt( s: "max", countOfRightAnswers).apply();
    }
    Intent intent = new Intent( packageContext: MainActivity.this, ScoreActivity.class);
    intent.putExtra( name: "result", countOfRightAnswers);
    startActivity(intent);
}
```

Рисунок 8. Метод onFinish

Далее создаются методы для работы таймера (рис. 8 и 9). В методе onTick получаются данные значения таймера. Если у таймера остаётся значение меньше 10 секунд, то цвет элемента, в котором находится таймер, меняется на красный. В методе getTimer принимаются значения таймера в наносекундах, далее переводятся в секунды и выводятся в виде строки формата минуты-секунды.

```
CountDownTimer timer = new CountDownTimer( millisInFuture: 20000, countDownInterval: 1000) {
    @Override
    public void onTick(long millisUntilFinished) {
        textViewTimer.setText(getTime(millisUntilFinished));
        if (millisUntilFinished < 10000) {
            textViewTimer.setTextColor(getResources().getColor(android.R.color.holo_red_light));
        }
    }
```

Рисунок 9. Метод onTick

```
private String getTime(long millis) {
    int seconds = (int) (millis / 1000);
    int minutes = seconds / 60;
    seconds = seconds % 60;
    return String.format(Locale.getDefault(), format: "%02d:%02d", minutes, seconds);
}
```

Рисунок 10. Метод getTimer

После реализации всех методов, можно проверить работоспособность приложения (рис. 11).

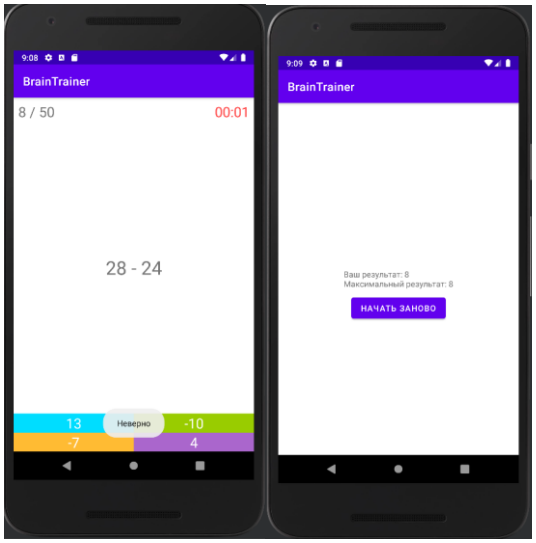

Рисунок 11. Результат работы приложения с несколькими окнами

Выводы

Таким образом было реализованно мобильное приложение с нескольими окнами с помощью языка программирования Java в IDE Android Studio. Также были описаны функцианальности активностей каждого окна приложения.

Библиографический список

1. Семченко Р.В., Еровлев П.А. Создание приложения для смартфона на основе сайта с помощью AndroidStudio // Постулат. 2019. № 6 (44). С. 10.
2. Попова М.В., Лысак Т.Р. Использование среды разработки Androidstudio для разработки android-приложений // В сборнике: Социально-экономические и информационные проблемы устойчивого развития региона. Международная научно-практическая конференция. 2015. С. 158-160.
3. Михайлов С.Ю., Иванов С.О. разработка мобильного приложения "Фитнес-дневник" под ОС ANDROID // В сборнике: Информатика и вычислительная техника. Сборник научных трудов. Чебоксары, 2018. С. 152-155.
4. Наумов Р.В. Android программирование. Начало программирования // Academy. 2016. № 1 (4). С. 46-48.
5. Макаров В.И. Программное обращение к службам android с использованием компонента telephony manager // Мировая наука. 2018. № 7 (16). С. 67-69.
6. Тухватуллин И.Д. Популяризация android – приложений // В сборнике: научные исследования и разработки. Сборник статей Международной научно-практической конференции. Ответственный редактор: Сукиасян Асатур Альбертович. 2015. С. 65-66.
7. Кобзев О.В., Забелин Л.Ю. Разработка приложения для android на java // В сборнике: Современные проблемы телекоммуникаций. Материалы Российской научно-технической конференции. 2018. С. 445-449.