

Разработка медиа-плеера на Android с помощью AudioManager

Андриенко Иван Сергеевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Аннотация

В данной работе рассмотрен процесс разработки медиа-плеера для платформы Android. Представлена пошаговая реализация приложения с использованием класса MediaPlayer, обеспечивающего воспроизведение аудиофайлов, и базовых элементов пользовательского интерфейса. Также описаны основные аспекты управления воспроизведением аудио, взаимодействием с пользователем и выводом уведомлений. Работа направлена на демонстрацию практического подхода к разработке мультимедийных приложений и может служить базой для дальнейших усовершенствований.

Ключевые слова: Android, медиа-плеер, MediaPlayer, воспроизведение аудио, Java, пользовательский интерфейс, AudioManager.

Developing an Android media player using AudioManager

Andrienko Ivan Sergeevich

Sholom-Aleichem Priamursky State University

Student

Abstract

This paper discusses the process of developing a media player for the Android platform. A step-by-step implementation of the application using the MediaPlayer class, which provides audio file playback, and basic user interface elements is presented. It also describes the main aspects of controlling audio playback and user interaction. The work is aimed at demonstrating a practical approach to the development of multimedia applications and can serve as a basis for further improvements.

Keywords: Android, Media Player, Media Player, audio playback, Java, user interface, AudioManager.

1 Введение

1.1 Актуальность

Мультимедийные приложения занимают значительное место в экосистеме Android, предоставляя пользователям удобный доступ к аудиоконтенту. Создание собственных медиа-плееров является важным навыком для разработчиков, поскольку такие приложения часто востребованы как в развлекательной, так и в образовательной сфере. Работа с MediaPlayer позволяет изучить ключевые аспекты управления мультимедиа, включая

обработку воспроизведения, паузы и остановки аудио. Данная тема особенно актуальна для начинающих разработчиков, так как она позволяет изучить базовые принципы работы с мультимедиа на платформе Android и освоить управление взаимодействием между пользовательским интерфейсом и внутренней логикой приложения.

1.2 Обзор исследований

Разработка мультимедийных приложений для платформы Android активно изучается в научной литературе. И. Д. Торжевский в своем обзоре программных библиотек для работы с аудио под ОС Android рассматривает возможности использования таких инструментов, как MediaPlayer, ExoPlayer и других, уделяя внимание их производительности и совместимости [1]. Курышжан А. акцентирует внимание на образовательном аспекте мультимедиа, исследуя применение аудиоматериалов в мобильном обучении, что подчеркивает важность удобного интерфейса для работы с аудио [2]. Д. И. Пандей и Х. Гоэл описывают разработку плеера Intellisense, акцентируя внимание на интеллектуальных функциях, таких как автоматическое создание плейлистов, что расширяет возможности современных аудиоприложений [3]. Практическое применение технологий описано в исследовании И. П. Деви и соавторов, где рассматривается создание приложения для музыки Минанг на базе Android с упором на локальные культурные особенности [4]. П. А. Кумават и его соавторы в своей работе демонстрируют использование Firebase для хранения и обработки аудиофайлов в приложениях, что позволяет реализовать облачные функции и синхронизацию данных [5].

1.3 Цель исследования

Цель исследования – создание медиа-плеера для платформы Android, который предоставляет базовые функции воспроизведения аудио с использованием класса MediaPlayer. Работа направлена на изучение процесса интеграции мультимедийных компонентов в приложения и разработку удобного интерфейса для управления воспроизведением.

2 Материалы и методы

Для реализации приложения использовались Android Studio версии 2022.3.1, язык программирования Java и класс MediaPlayer из Android SDK. Приложение тестировалось на устройствах с Android 8.0 и выше, а аудиофайлы хранились в директории res/raw проекта.

3 Результаты и обсуждения

Перед началом разработки необходимо настроить минимальную версию SDK для проекта. Это обеспечивает совместимость приложения с современными инструментами и библиотеками. В файле build.gradle модуля app в разделе android необходимо установить параметры (рис. 1).

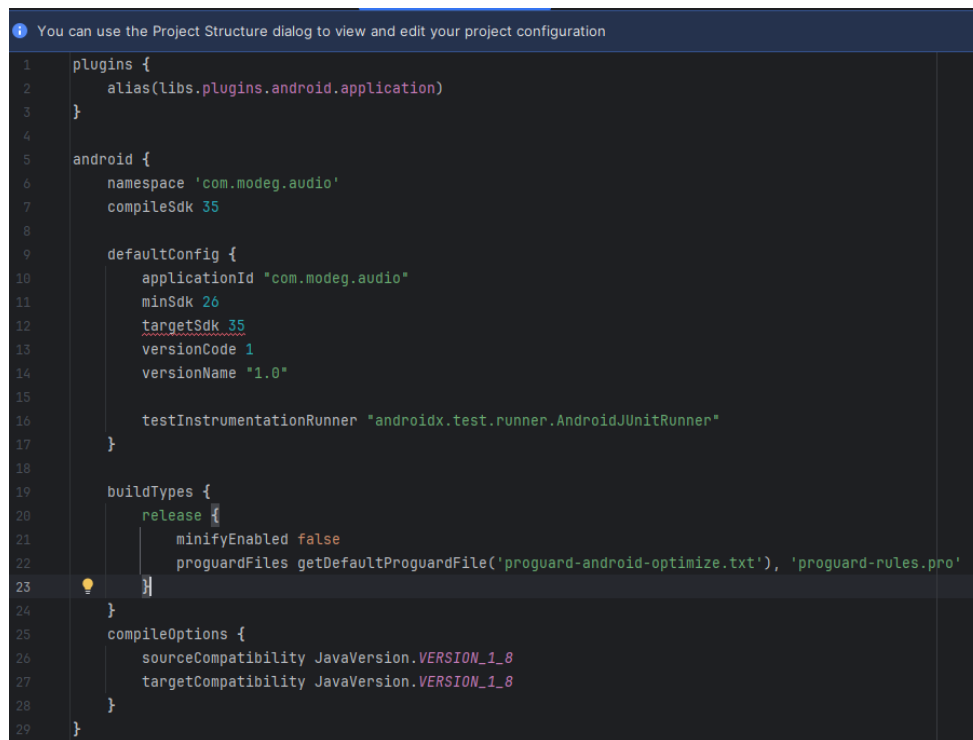


Рис. 1. Настройка файла конфигурации

После обновления настроек рекомендуется выполнить синхронизацию проекта с Gradle, чтобы убедиться в корректности конфигурации. Этот шаг гарантирует, что проект сможет использовать все возможности платформы Android на текущем уровне API.

Далее в директории `res` создается папка `raw`, куда помещается аудиофайл, предназначенный для воспроизведения. Это позволяет получить доступ к ресурсу из приложения без необходимости обращения к внешнему хранилищу.

Интерфейс приложения разрабатывается с использованием компонента `LinearLayout`, который обеспечивает упорядоченное расположение элементов на экране. Для управления воспроизведением создаются три кнопки: `Play`, `Pause` и `Stop`. Эти элементы размещаются по центру экрана и имеют отступы для удобства взаимодействия пользователя с интерфейсом (рис. 2).

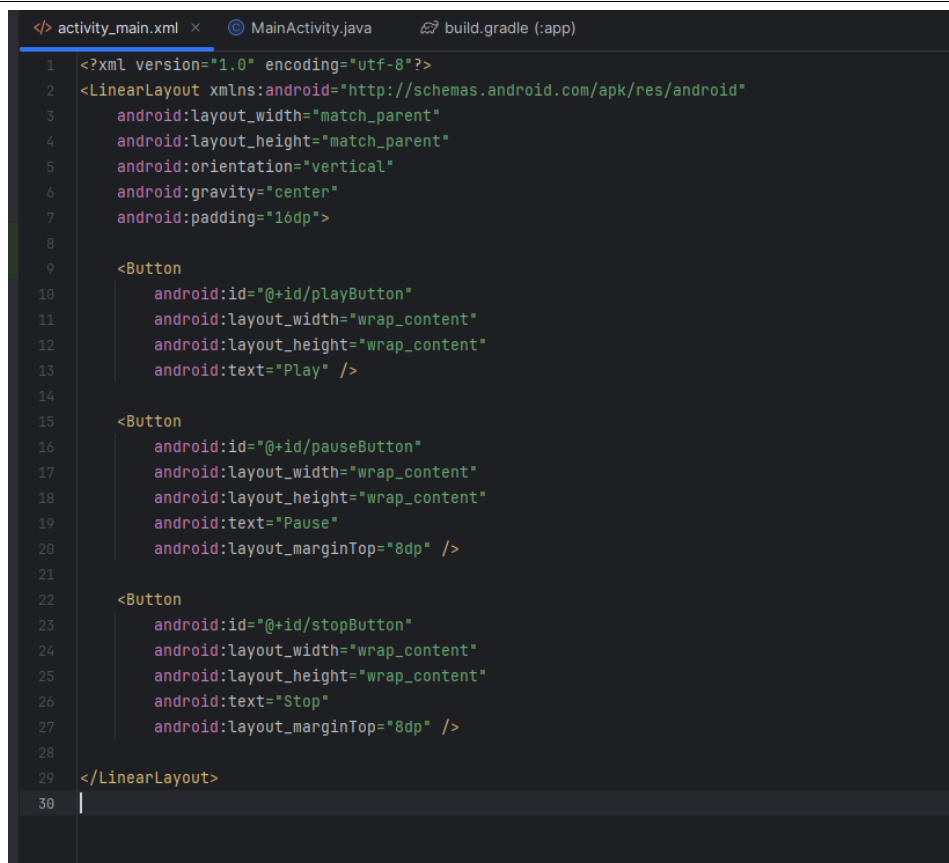


Рис. 2. Разработка макета

Основная логика работы приложения реализуется в классе MainActivity. Для воспроизведения аудио используется класс MediaPlayer, который инициализируется с помощью метода create. В приложении обрабатываются события нажатия на кнопки. Нажатие на кнопку Play запускает воспроизведение аудиофайла, Pause приостанавливает его, а Stop останавливает воспроизведение и освобождает ресурсы. Чтобы повторно воспроизвести файл после остановки, объект MediaPlayer создается заново (рис. 3). Для завершения работы приложения и предотвращения утечек памяти в методе onDestroy реализуется освобождение ресурсов, связанных с MediaPlayer. Это важный этап разработки, поскольку неправильное управление ресурсами может привести к сбоям в работе приложения.

```
9 public class MainActivity extends AppCompatActivity {
10
11     private MediaPlayer mediaPlayer; 11 usages
12
13     @Override
14     protected void onCreate(Bundle savedInstanceState) {
15         super.onCreate(savedInstanceState);
16         setContentView(R.layout.activity_main);
17
18         Button playButton = findViewById(R.id.playButton);
19         Button pauseButton = findViewById(R.id.pauseButton);
20         Button stopButton = findViewById(R.id.stopButton);
21
22         mediaPlayer = MediaPlayer.create(context, this, R.raw.audio_file);
23
24         playButton.setOnClickListener(v -> {
25             if (!mediaPlayer.isPlaying()) {
26                 mediaPlayer.start();
27             }
28         });
29
30         pauseButton.setOnClickListener(v -> {
31             if (mediaPlayer.isPlaying()) {
32                 mediaPlayer.pause();
33             }
34         });
35
36         stopButton.setOnClickListener(v -> {
37             if (mediaPlayer.isPlaying()) {
38                 mediaPlayer.stop();
39                 mediaPlayer = MediaPlayer.create(context, this, R.raw.audio_file);
40             }
41         });
42     }
43
44     @Override
45     protected void onDestroy() {
46         super.onDestroy();
47         if (mediaPlayer != null) {
48             mediaPlayer.release();
49             mediaPlayer = null;
50         }
51     }
52 }
```

Рис. 3. Код MainActivity.java

Для улучшения пользовательского опыта было добавлено отображение уведомлений в панели состояния. Уведомление включает название текущего трека и кнопки для управления воспроизведением, такие как "Play" и "Pause". Реализация уведомлений позволяет пользователю управлять воспроизведением даже при свернутом приложении.

Для создания уведомления используется класс `NotificationCompat.Builder`, который позволяет настроить текст уведомления, название трека, отображающееся пользователю, и добавить кнопки для управления воспроизведением, такие как "Play" и "Pause". Для поддержки устройств с Android 8.0 и выше создается канал уведомлений с использованием класса `NotificationChannel`. Этот канал задает параметры, такие как уровень важности уведомлений и их поведение. Кнопки управления воспроизведением связаны с `PendingIntent`, который обрабатывает действия пользователя, например запуск или пауза трека (рис. 4, 5).

```

60     private void createNotificationChannel() { 1 usage
61         if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
62             NotificationChannel channel = new NotificationChannel(
63                 CHANNEL_ID,
64                 "Уведомления плеера",
65                 NotificationManager.IMPORTANCE_LOW
66             );
67             notificationManager.createNotificationChannel(channel);
68         }
69     }
70
71     private void showNotification(String contentText, boolean isPlaying) { 2 usages
72         Intent intent = new Intent( packageContext: this, MainActivity.class);
73         PendingIntent pendingIntent = PendingIntent.getActivity(
74             context: this, requestCode: 0, intent, PendingIntent.FLAG_IMMUTABLE
75         );
76
77         Intent playIntent = new Intent( packageContext: this, MainActivity.class);
78         playIntent.setAction("ACTION_PLAY");
79         PendingIntent playPendingIntent = PendingIntent.getBroadcast(
80             context: this, requestCode: 1, playIntent, PendingIntent.FLAG_IMMUTABLE
81         );
82
83         Intent pauseIntent = new Intent( packageContext: this, MainActivity.class);
84         pauseIntent.setAction("ACTION_PAUSE");
85         PendingIntent pausePendingIntent = PendingIntent.getBroadcast(
86             context: this, requestCode: 2, pauseIntent, PendingIntent.FLAG_IMMUTABLE
87         );
88     }

```

Рис. 4. Реализация уведомлений

```

83     Intent pauseIntent = new Intent( packageContext: this, MainActivity.class);
84     pauseIntent.setAction("ACTION_PAUSE");
85     PendingIntent pausePendingIntent = PendingIntent.getBroadcast(
86         context: this, requestCode: 2, pauseIntent, PendingIntent.FLAG_IMMUTABLE
87     );
88
89     NotificationCompat.Builder builder = new NotificationCompat.Builder( context: this, CHANNEL_ID)
90         .setSmallIcon(R.drawable.ic_launcher_foreground)
91         .setContentTitle("Плеер")
92         .setContentText(contentText)
93         .setPriority(NotificationCompat.PRIORITY_LOW)
94         .setContentIntent(pendingIntent)
95         .addAction(isPlaying ? R.drawable.ic_launcher_foreground : R.drawable.ic_launcher_foreground,
96             isPlaying ? "Пауза" : "Пуск",
97             isPlaying ? pausePendingIntent : playPendingIntent)
98         .setOngoing(isPlaying);
99
100     notificationManager.notify( id: 1, builder.build());
101 }

```

Рис. 5. Реализация уведомлений

Приложение было протестировано на устройстве с Android 12. В процессе тестирования проверялась работа воспроизведения, паузы и остановки аудиофайла, а также корректное отображение уведомлений. Все функции показали стабильную и корректную работу (рис. 6, 7).

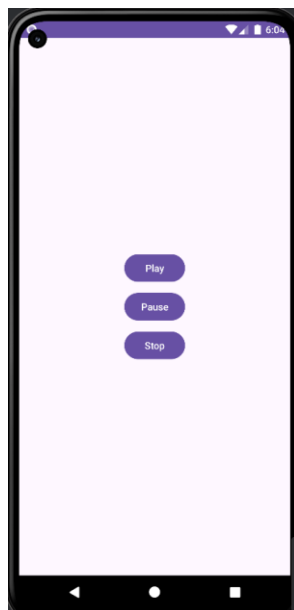


Рис. 6. Макет приложения

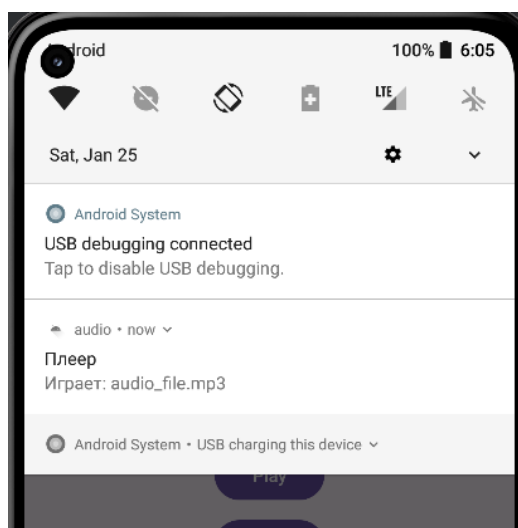


Рис. 7. Центр уведомлений

4 Выводы

В результате разработки был создан медиа-плеер для платформы Android с базовыми функциями воспроизведения аудио, включая управление через пользовательский интерфейс и панель уведомлений. Приложение продемонстрировало корректную работу и стабильность, что подтверждает возможность использования данного подхода для создания мультимедийных приложений. Реализация уведомлений улучшила интерфейс, позволив управлять треком без необходимости открывать приложение.

Библиографический список

1. Торжевский И. Д. Обзор программных библиотек для работы с аудио под ОС Android // В сборнике: Решетневские чтения. Материалы XXV Международной научно-практической конференции, посвященной памяти

- генерального конструктора ракетно-космических систем академика М. Ф. Решетнева. В 2-х частях. Под общей редакцией Ю. Ю. Логинова. Красноярск, 2021. С. 412-413.
2. Курышжан А. Мобильное обучение при преподавании казахского языка (аудиоработа) // Alatoo Academic Studies. 2016. № 4. С. 274-280.
 3. Pandey A., Goel H. Intellisense Media Player // International Journal of Computer Applications. 2021. Т. 183. № 3. С. 47-50.
 4. Dewi I. P., Kamil F. M., Ambiyar A., Refdinal R., Indarta Y. Design and Build Minang Music Applications Based on Android // SinkrOn. 2022. Т. 7. № 2. С. 367-375.
 5. Kumawat P., Jain S., Jain A., Jain P., Bohra A. A Music Player Android App Using Firebase // International Journal of Engineering, Business and Management. 2021. Т. 5. № 3. С. 91-93.