УДК 004

Создание аудио плеера на андроид смартфон

Семченко Регина Викторовна
Приамурский государственный университет имени Шолом-Алейхема
студент

Еровлев Павел Андреевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье описан метод написания аудио плеера для андроид смартфона в среде разработке Android studio на языке программирования java script. Практическим результатом является рабочее мобильно приложение с разрешениями на просмотр файлом и функциями надлежащим аудиоплееру.
Ключевые слова: Android, Android Studio, приложение, аудио плеер.

Create audio player an android smartphone

Semchenko Regina Viktorovna
Sholom-Aleichem Priamursky State University
student

Erovlev Pavel Andreevich
Sholom-Aleichem Priamursky State University
Student

Abstract
This article describes the method of writing an audio player for an Android Smartphone in the development environment of Android studio in the programming language java script. The practical result is a working mobile application with permissions to view the file and functions appropriate to the audio player
Keywords: Android, Android Studio, application, audio player

На сегодняшний день большое количество людей любят слушать музыку, когда идут на работу или с нее, гуляют или едут на дальнее расстояние, прослушивание происходит либо в социальной сети, через приложение музыки, либо через приложение аудиоплеер в телефоне.

Цель данной статьи создать рабочее мобильное приложение с поддержкой прослушивания аудиофайлов и отвечающее всем критериям аудиоплеера, таким как переход к следующей или предыдущей песне, паузе,

отображения на заблокированном экране. Прилоение будет создано на языке программирования JavaScript в среде разработки Android Studio.

Исследованиями в области разработки мобильных приложений занимались многие российские и зарубежные исследователи. А.С. Винокуров, Р.И. Баженов [1] рассмотрели разработку приложений для мобильных устройств. С.К. Заманова, Г.Е. Сейдахметова, Г.Г. Масимова, А.Е. Манатова[2]. Они изучили современные подходы к разработке мобильных приложений. Также они рассмотрели разработку приложения в среде Rad Studio XE7. Е.А. Зотова, М.И. Притчина [3] рассмотрели развитие программных платформ iOS и Android. И.С. Полотнянсчиков, Л.А.Залогова описали выбор инструментов, провели анализ предметной области и продемонстрировали реализацию собственного мобильного приложения. Е.Н.Амиргалиев и др. разработал свою собственную модель отправки информационных сообщений для мобильных операционных систем.[4]

Для начала создадим новый проект в Android Studio и добавим следующие разрешения в файл AndroidManifest.xml (рис.1).

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

Рисунок 1 – объявление разрешений

Приложению необходимы эти разрешения для доступа к мультимедийным файлам через Интернет. Поскольку целью этой статьи является создание приложения аудиоплеера, необходимо добавить MEDIA_CONTENT_CONTROL для управления воспроизведением мультимедии. Использование READ_PHONE_STATE разрешения для доступа к состоянию телефона, чтобы можно было остановить звук ,например во время разговора.

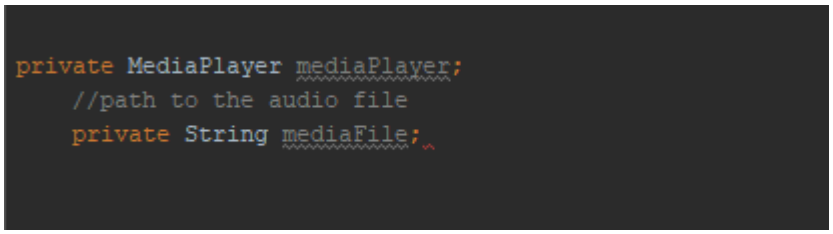
Основой приложения AudioPlayer является сервис медиаплеера. Следующий класс является примером этого сервиса. Класс имеет несколько MediaPlayer реализаций для обработки событий, которые могут произойти во время воспроизведения аудио. Реализация AudioManager.OnAudioFocusChangeListener необходима для обработки запросов AudioFocus от других приложений, которые хотят воспроизводить медиа-файлы (рис.2).

```
    @Override
    public IBinder onBind(Intent intent) {
        return iBinder;
    }

    @Override
    public void onBufferingUpdate(MediaPlayer mp, int percent) {

    }

    @Override
    public void onCompletion(MediaPlayer mp) {

    }

    //Handle errors
    @Override
    public boolean onError(MediaPlayer mp, int what, int extra) {

        return false;
    }

    @Override
    public boolean onInfo(MediaPlayer mp, int what, int extra) {

        return false;
    }

    @Override
    public void onPrepared(MediaPlayer mp) {

    }

    @Override
    public void onSeekComplete(MediaPlayer mp) {

    }

    @Override
    public void onAudioFocusChange(int focusChange) {

    }

    public class LocalBinder extends Binder {
        public MediaPlayerService getService() {
            return MediaPlayerService.this;
        }
    }
}
```

Рисунок 2 – обработка запросов

Приведенный выше код является шаблоном всех методов, которые будут обрабатывать MediaPlayer события. Единственный завершенный код - это привязка Service. Необходимо связать этот сервис, потому что он взаимодействует с активным файлом для получения аудиофайлов

Объявим Service в файле AndroidManifest.xml. Мультимедийная структура Android поддерживает множество распространенных типов мультимедиа. Одним из ключевых компонентов этой платформы является класс MediaPlayer, который с минимальными настройками можно использовать для воспроизведения аудио и видео, но понадобится больше, чем этот пример Service для воспроизведения мультимедиа. Далее описывается необходимые методы, которые необходимо настроить в MediaPlayerService классе. Необходимо создать следующие глобальные экземпляры MediaPlayer и String в Service классе (рис.3).



```
private MediaPlayer mediaPlayer;
    //path to the audio file
    private String mediaFile;
```

Рисунок 3 – создание экземпляров

Теперь инициализируем mediaPlayer (рис.4).

```
private void initMediaPlayer() {
    mediaPlayer = new MediaPlayer();
    mediaPlayer.setOnCompletionListener(this);
    mediaPlayer.setOnErrorListener(this);
    mediaPlayer.setOnPreparedListener(this);
    mediaPlayer.setOnBufferingUpdateListener(this);
    mediaPlayer.setOnSeekCompleteListener(this);
    mediaPlayer.setOnInfoListener(this);
    mediaPlayer.reset();
    mediaPlayer.setAudioStreamType(AudioManager.STREAM_MUSIC);
    try {
        mediaPlayer.setDataSource(mediaFile);
    } catch (IOException e) {
        e.printStackTrace();
        stopSelf();
    }
    mediaPlayer.prepareAsync();
}
```

Рисунок 4 – инициализация mediaPlayer

При работе с мультимедиа необходимо реализовать некоторые функции для обработки основных действий при воспроизведении мультимедиа. Этими основными функциями являются воспроизведение, остановка, пауза и возобновление. Сначала добавим еще одну глобальную переменную для сохранения позиции паузы / возобновления. Добавим if операторы, чтобы убедиться в отсутствии проблем при воспроизведении мультимедиа(рис.5).

```
private void playMedia() {
    if (!mediaPlayer.isPlaying()) {
        mediaPlayer.start();
    }
}

private void stopMedia() {
    if (mediaPlayer == null) return;
    if (mediaPlayer.isPlaying()) {
        mediaPlayer.stop();
    }
}

private void pauseMedia() {
    if (mediaPlayer.isPlaying()) {
        mediaPlayer.pause();
        resumePosition = mediaPlayer.getCurrentPosition();
    }
}

private void resumeMedia() {
    if (!mediaPlayer.isPlaying()) {
        mediaPlayer.seekTo(resumePosition);
        mediaPlayer.start();
    }
}
```

Рисунок 5 – создание функций

Теперь, когда создали функции инициализации, пришло время реализовать @Override методы, созданные в исходном Service шаблоне. Эти методы важны MediaPlayer, потому что все ключевые действия, которые будет выполнять пользователь, будут вызываться из этих методов. Заменяем оригинальные методы в Service шаблоне следующим (рис6).

```
@Override
public boolean onError(MediaPlayer mp, int what, int extra) {
    switch (what) {
        case MediaPlayer.MEDIA_ERROR_NOT_VALID_FOR_PROGRESSIVE_PLAYBACK:
            Log.d( tag: "MediaPlayer Error", msg: "MEDIA ERROR NOT VALID FOR PROGRESSIVE PLAYBACK " + extra);
            break;
        case MediaPlayer.MEDIA_ERROR_SERVER_DIED:
            Log.d( tag: "MediaPlayer Error", msg: "MEDIA ERROR SERVER DIED " + extra);
            break;
        case MediaPlayer.MEDIA_ERROR_UNKNOWN:
            Log.d( tag: "MediaPlayer Error", msg: "MEDIA ERROR UNKNOWN " + extra);
            break;
    }
    return false;
}

@Override
public void onPrepared(MediaPlayer mp) {
    playMedia();
}
```

Рисунок 6 – реализация методов

Осталось только определить onStartCommand() метод услуг. Этот метод будет обрабатывать инициализацию MediaSession и MediaPlayer, загружая кэшированные аудио воспроизведения и создание MediaStyle уведомления. В классе обслуживания меняем старый onStartCommand() метод следующим (рис.7).

```
@Override
public int onStartCommand(Intent intent, int flags, int startId) {
    try {
        StorageUtil storage = new StorageUtil(getApplicationContext());
        BitSet audioList = storage.loadAudio();
        boolean audioIndex = storage.loadAudioIndex();

        if (audioIndex != -1 && audioIndex < audioList.size()) {
            boolean activeAudio = audioList.get(audioIndex);
        } else {
            stopSelf();
        }
    } catch (NullPointerException e) {
        stopSelf();
    }

    //Request audio focus
    if (requestAudioFocus() == false) {
        stopSelf();
    }

    Object mediaSessionManager;
    if (mediaSessionManager == null) {
        try {
            initMediaSession();
            initMediaPlayer();
        } catch (RemoteException e) {
            e.printStackTrace();
            stopSelf();
        }
        buildNotification(PlaybackStatus.PLAYING);
    }

    handleIncomingActions(intent);
    return super.onStartCommand(intent, flags, startId);
}
```

Рисунок 7 – замена метода

В initMediaPlayer()функции меняем setDataSource() вызов следующей строкой:

mediaPlayer.setDataSource(activeAudio.getData());

Теперь запустим приложение и воспроизводите аудио (рис.8).

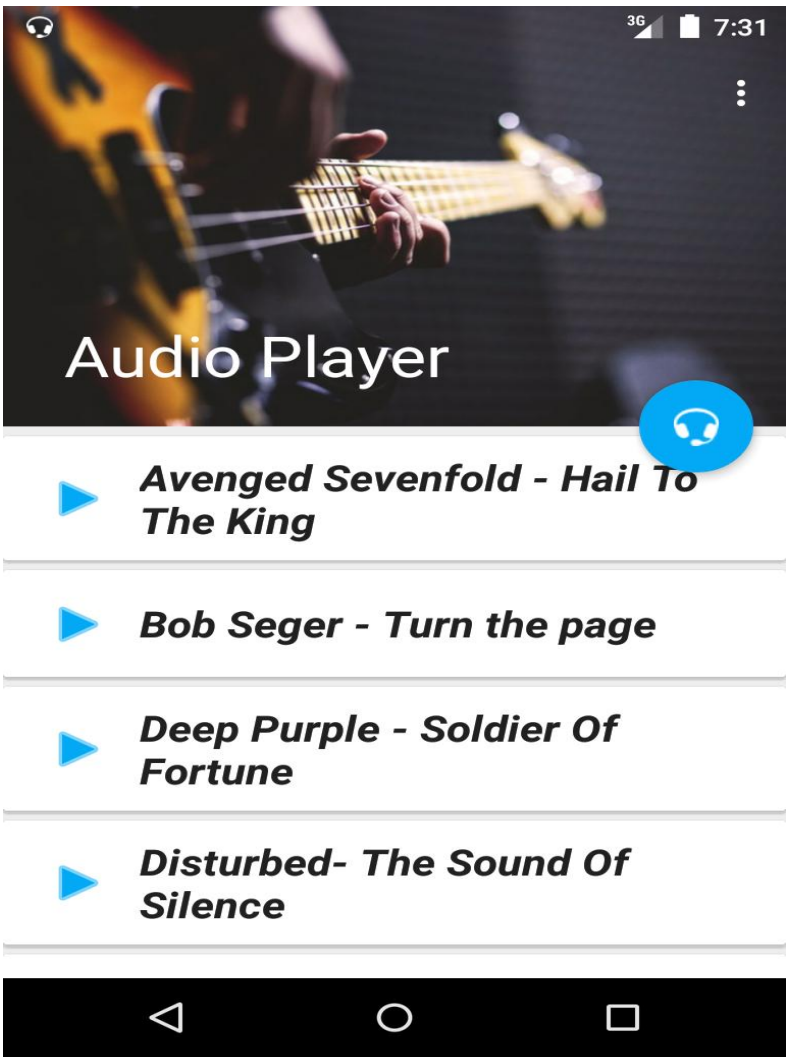


Рисунок 8 – Рабочее приложение

Теперь запуская приложение, будет открываться вот такой аудиоплеер созданный на android смартфоны.

Подводя итоги можно сказать, что был реализован рабочий медиаплеер для android смартфона на языке программирования JavaScript в среде AndroidStudio.

Библиографический список

1. Винокуров А.С., Баженов Р.И. Разработка мобильного приложения информационного сайта для абитуриентов и первокурсников университета // Современные научные исследования и инновации. 2015. № 7-2 (51). С. 54-62.
2. Заманова С.К., Сейдахметова Г.Е., Масимова Г.Г., Манатова А.Е. Разработка мобильного приложения в среде Rad Studio XE7 // Труды Международного симпозиума «Надежность и качество». 2015. №1. С. 237-240.
3. Amirgaliyev E.N., Kalizhanova A.U., Kozbakova A.KH. Development of applications to mobile devices in Android platform // Труды Международного

симпозиума «Надежность и качество». 2015. №1. С. 240-242.
4. Полотнянщиков И. С. Технология создания трехмерных приложений реального времени для ОС ANDROID //Вестник Пермского университета. Серия: Математика. Механика. Информатика. 2012. №. 3. С. 94-97.