streamaxia

OpenPlayer 3.0 for Android

Setup Guide

Contents

Introduction

Streamaxia OpenPlayer is a media playback library that can play streams over RTMP / DASH / HLS on mobile devices.

This is a short programming guide about how to integrate Streamaxia OpenPlayer library in your Android project.

Components

OpenPlayer Main Interface

The *StreamaxiaPlayer* interface is responsible for the initialization and configuration of the SDK. It also provides information about the available features.

Important: Before using the player, the SDK must be properly initialized and configured.

OpenPlayer States

The *StreamaxiaPlayerState* interface represents all the states of the OpenPlayer. It allows you to get notified about the latest player state. Your Activity must implement this interface.

Advice: To provide the best user experience, you must treat each state of the player differently.

The states of the player are as follow:

- ***ENDED***
- ***BUFFERING***
- ***IDLE***
- ***PREPARING***
- ***READY***
- ***UNKNOWN***

Set up an OpenPlayer Project

1. Open an existing project or create a new project
2. Add the *.aar* file provided as a module to the main project
3. Add the license.dat file in the assets folder of the main project

Initialize the OpenPlayer Main Interface

To initialize the SDK, you need to make sure that the *license.dat* file is added to the */assets* folder. Create a new StreamaxiaPlayer object as follows:

```
private StreamaxiaPlayer mStreamaxiaPlayer = new StreamaxiaPlayer();
```

In the activity XML, create an *AspectRatioFrameLayout* that contains a *SurfaceView* inside. Those views will be the place where the stream will be rendered, and you will need to pass them in the *init()* method of the *StreamaxiaPlaye*.

```
<com.google.android.exoplayer.AspectRatioFrameLayout
    android:id="@+id/video_frame"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_centerInParent="true"
    android:layout_gravity="center"
    android:background="@android:color/black">

    <SurfaceView
        android:id="@+id/surface_view"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_gravity="center"/>

</com.google.android.exoplayer.AspectRatioFrameLayout>
```

Initialize the object:

```
mStreamaxiaPlayer.initStreamaxiaPlayer(surfaceView,aspectRatioFrameLayout,
stateText, this, this, uri);
```

The parameters are as follows:

1. *SurfaceView*
2. *AspectFrameLayout*
3. *TextView*
4. *IStreamaxiaPlayerState*
5. *Context*
6. *URL*

Set up the StreamaxiaPlayerState

To provide the best user experience, you must treat each state of the player differently and notify the user of the currents player state.

```
@Override
public void stateENDED() {
    progressBar.setVisibility(View.GONE);
    playBtn.setImageResource(R.drawable.play);
}

@Override
public void stateBUFFERING() {
    progressBar.setVisibility(View.VISIBLE);
}
```

```
@Override
public void stateIDLE() {
    progressBar.setVisibility(View.VISIBLE);
}

@Override
public void statePREPARING() {
    progressBar.setVisibility(View.VISIBLE);
}

@Override
public void stateREADY() {
    progressBar.setVisibility(View.GONE);
}

@Override
public void stateUNKNOWN() {
    progressBar.setVisibility(View.VISIBLE);
}
```

Start/Stop/Pause/Resume Playback

These functions should be called after the *StreamaxiaPlayer* is initiated and all the states are handled.

```
mStreamaxiaPlayer.play(uri, STREAM_TYPE);
```

You will need to pass the *URI* of the stream to be played, the stream type (one: TYPE_RTMP, TYPE_DASH, TYPE_HLS - see the *StreamaxiaPlayer* class for more details).
The player support to set the *bufferSegment* size as a third parameter or none.

```
mStreamaxiaPlayer.play(uri, STREAM_TYPE, BUFFER_SIZE);
```

Stop the playback and reset the player:

```
mStreamaxiaPlayer.stop();
```

Pause the current playback without restarting the player. If the stream is available, the player tries to continue from the previous position.

```
mStreamaxiaPlayer.pause();
```

Resources

OpenSDK for Broadcasting

For iOS and Android Mobile App Developers

Easy to integrate, low-latency live video streaming library.
Open broadcast – not limited to any specific CDN, RTMP Media Server or proprietary protocols.

Drag, Drop & Go Live!

OpenSDK 3.0 for iOS

OpenSDK 3.0 for Android

BroadcastMe App

Live Video Streaming Broadcast Apps

Private Label for iOS and Android Available

BroadcastMe Developer Edition is designed by Streamaxia to be used by Mobile App Developers and Digital Media experts as is, and it is available for private label for your brand.

Read More

Help and Support

Our team looks forward to helping you with business inquiries and technical support questions.
We will review all messages within two business days and respond back to you promptly about your request for information.

support@streamaxia.com

Mo-Fri 9AM-5PM EET