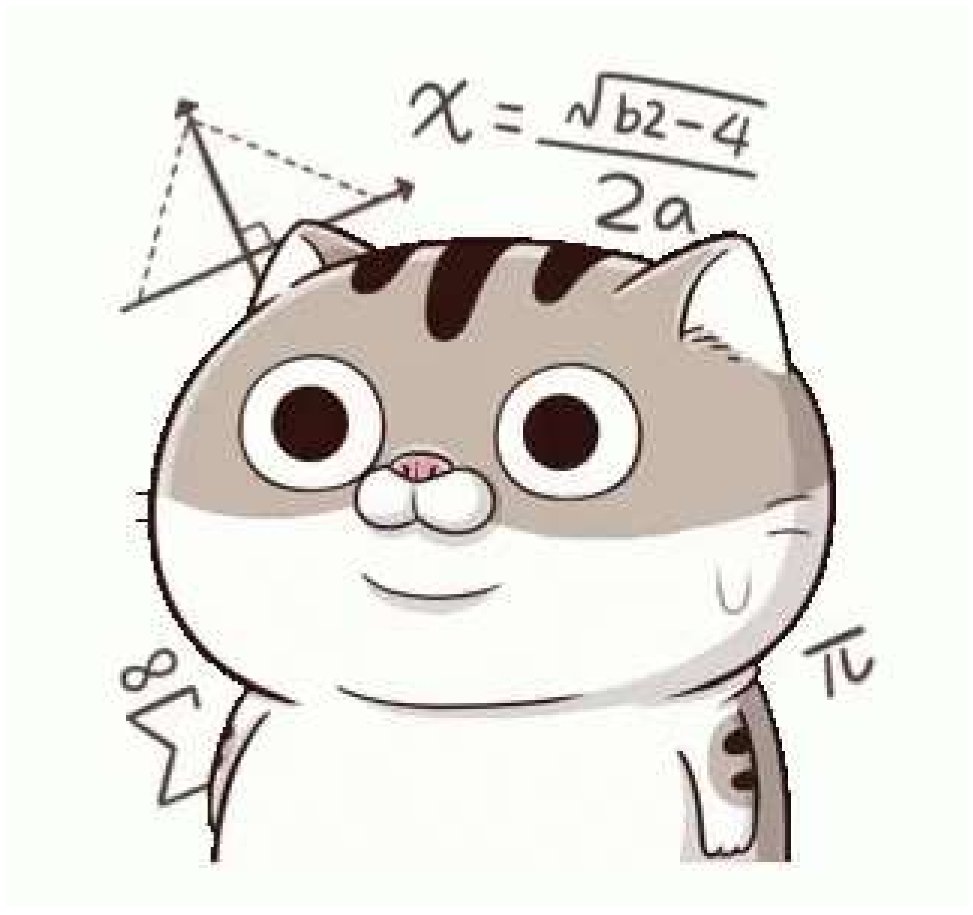
Connecting to a Bluetooth A2DP Device from Android

Kushangi Patel
Oct 9, 2020 · 3 min read

For a while now, I've been wondering about how to connect Bluetooth Headset and stream audio from my android device with a swift easy to use program. Finally after some exhausting hours of searching I've found a solution!

So, as a developer it didn't come as shock to me that I would have to learn and understand how this could be done in Android Apps. But the question that haunted and tormented me was how? It wasn't a terrain I had crossed before but it sure was an interesting path.



But before we dive into the easy steps, there are some basic concepts worth understanding. Earlier, we had to go to all the trouble of creating the socket and establishing the connection between the Bluetooth devices.

Starting in Android 3.0, the Bluetooth API includes support for working with Bluetooth profiles. A ***Bluetooth profile*** is a wireless interface specification for establishing Bluetooth-based communication between devices.

So, all we have to do is just add the class BluetoothA2dp that would be used to connect to an A2DP device.

What is Bluetooth A2DP?

Bluetooth A2DP(`Advanced Audio Distribution Profile`) is the Bluetooth Stereo profile which defines how high quality stereo audio can be streamed from one device to another over a Bluetooth connection — for example, music streamed from a mobile phone to wireless headphones.

In older Android versions the classes which are required to connect to a Bluetooth A2DP device was hidden in the API level and in the later versions from Android 4.0

onwards, it is made visible partially.

But there is a solution which will work on all the Android versions, you can easily interact with any A2DP device by simply making some **IPC** (*Inter Process Communication*) calls.

Basically, this is achieved by the use of **AIDL** (*Android Interface Definition Language*) which basically allows to define the programming interface that both the client and service agree upon in order to communicate with each other using IPC.

How to Connect a Bluetooth Device to Any Android Device?

To enable the connection to a Bluetooth device to any android device programmatically, you just need to follow these simple steps.

Step 1: Create AIDL files

Our first step is to get hold on the instance of Bluetooth A2DP class. For that, what you need to do is create 2 AIDL files first in your *"src"* folder of Android Project under package `android.bluetooth` as follows :

```
interface IBluetooth {
    /**
     * System private API for Bluetooth service
     */
    String getRemoteAlias(in String address);
      boolean setRemoteAlias(in String address, in String name);
}
```

IBluetooth.aidl hosted with ♡ by GitHub view raw

```
interface IBluetoothA2dp {
    /**
     * System private API for Bluetooth A2DP service
     */
     boolean connectSink(in BluetoothDevice device); // Pre API 11 only
       boolean disconnectSink(in BluetoothDevice device); // Pre API 11 only
       boolean connect(in BluetoothDevice device); // API 11 and up only
       boolean disconnect(in BluetoothDevice device); // API 11 and up only
       boolean suspendSink(in BluetoothDevice device); // all
       boolean resumeSink(in BluetoothDevice device); // all
       BluetoothDevice[] getConnectedSinks();  // change to Set<> once AIDL supports, pre AP
       BluetoothDevice[] getNonDisconnectedSinks();  // change to Set<> once AIDL supports
```

```
        BluetoothDevice[] getNonDisconnectedSinks();  // change to Set<> once AIDL supports,
        int getSinkState(in BluetoothDevice device); // pre API 11 only
        int getConnectionState(in BluetoothDevice device); // API 11 and up
        boolean setSinkPriority(in BluetoothDevice device, int priority); // Pre API 11 only
        boolean setPriority(in BluetoothDevice device, int priority); // API 11 and up only
        int getPriority(in BluetoothDevice device); // API 11 and up only
        int getSinkPriority(in BluetoothDevice device); // Pre API 11 only
        boolean isA2dpPlaying(in BluetoothDevice device); // API 11 and up only
}
```

Step 2: Add permissions

Then in the Android Manifest, add permissions for using Bluetooth and the following services :

```
android.permission.BLUETOOTH
android.permission.BLUETOOTH_ADMIN
android.permission.BROADCAST_STICKY
android.permission.BIND_ACCESSIBILITY_SERVICE
```

Step 3: Enable Bluetooth and Get Paired Devices

First, you need to ensure whether Bluetooth is enabled. To request that Bluetooth be enabled, call `startActivityForResult()` with the `ACTION_REQUEST_ENABLE` action Intent. This will issue a request to enable Bluetooth through the system settings.

```
val enableBtIntent = Intent(BluetoothAdapter.
ACTION_REQUEST_ENABLE)
startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT)
```

You can get the already paired devices from BluetoothAdapter class as follows:

```
devices = BluetoothAdapter.getDefaultAdapter().bondedDevices
```

From the devices you can get your headset device which you want to pair

Step 4: Establish Bluetooth Connection

Use `getProfileProxy()` to establish a connection to the profile proxy object associated with the profile. Set up a `BluetoothProfile.ServiceListener()`, This listener notifies `BluetoothProfile` IPC clients when they have been connected to or disconnected from the service.

```
fun connectUsingBluetoothA2dp(
        deviceToConnect: BluetoothDevice?
    ) {
        try {
            val c2 = Class.forName("android.os.ServiceManager")
            val m2: Method = c2.getDeclaredMethod("getService", String::class.java)
            b = m2.invoke(c2.newInstance(), "bluetooth_a2dp") as IBinder?
            if (b == null) {
                // For Android 4.2 Above Devices
                device = deviceToConnect
                //establish a connection to the profile proxy object associated with the pro
                BluetoothAdapter.getDefaultAdapter().getProfileProxy(
                    this,
                    // listener notifies BluetoothProfile clients when they have been connec
                    object : ServiceListener {
                        override fun onServiceDisconnected(profile: Int) {
                            setIsA2dpReady(false)
                            disConnectUsingBluetoothA2dp(device)
                        }

                        override fun onServiceConnected(
                            profile: Int,
                            proxy: BluetoothProfile
                        ) {
                            a2dp = proxy as BluetoothA2dp
                            try {
                                //establishing bluetooth connection with A2DP devices
                                a2dp.javaClass
                                    .getMethod("connect", BluetoothDevice::class.java)
                                    .invoke(a2dp, deviceToConnect)
                            } catch (e: Exception) {
                                e.printStackTrace()
                            }
                            setIsA2dpReady(true)
                        }
                    }, BluetoothProfile.A2DP
                )
                )
            } else {
                val c3 =
                    Class.forName("android.bluetooth.IBluetoothA2dp")
                val s2 = c3.declaredClasses
                val c = s2[0]
                val m: Method = c.getDeclaredMethod("asInterface", IBinder::class.java)
                m.isAccessible = true
                ia2dp = m.invoke(null, b) as IBluetoothA2dp
                ia2dp.connect(deviceToConnect)
            }
        } catch (e: Exception) {
            e.printStackTrace()
        }
    }
```

Next step is to stream audio which you want to play on connected Bluetooth Headset. Initialize *MediaPlayer* and play the streaming url as below:

```
private fun playMusic() {
        //streaming music on the connected A2DP device
        mPlayer = MediaPlayer()
        try {
            mPlayer?.setAudioAttributes(
                AudioAttributes.Builder()
                    .setContentType(AudioAttributes.CONTENT_TYPE_MUSIC).build()
            )
            mPlayer?.setDataSource(
                this,
                url
            )
            mPlayer?.prepare()
            mPlayer?.start()
        } catch (e: IOException) {
            e.printStackTrace()
        }
    }
```

Step 6: Disconnect the device

Finally done with playing the audio and want to disconnect the device close proxy connection, use the following method to disconnect and release the MediaPlayer:

```
fun disConnectUsingBluetoothA2dp(
        deviceToConnect: BluetoothDevice?
    ) {
        try {
            // For Android 4.2 Above Devices
            if (b == null) {
                try {
                    //disconnecting bluetooth device
                    a2dp.javaClass.getMethod(
                        "disconnect",
                        BluetoothDevice::class.java
                    ).invoke(a2dp, deviceToConnect)
                    BluetoothAdapter.getDefaultAdapter()
                        .closeProfileProxy(BluetoothProfile.A2DP, a2dp)
                } catch (e: Exception) {
                    e.printStackTrace()
                }
            } else {
            // For Android 4.2 Below Devices
                ia2dp.disconnect(deviceToConnect)
            }
        } catch (e: Exception) {
            e.printStackTrace()
        }
    }

    private fun releaseMediaPlayer() {
        mPlayer?.release()
    }
```

So that's it! Hope you enjoyed this blog !!!

You can see the entire sample of Bluetooth A2DP device connectivity here, go & check it out.

For any queries feel free to leave a comment below. Your ideas are always welcome. 😊

Thanks to Dhruv Mevada.

Android App Development Bluetooth Connectivity Mobile App