```
package com.example.listadetarefas;
import android.media.AudioDeviceCallback;
import android.media.AudioDeviceInfo;
import android.os.Bundle;
import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;
import android.content.Context;
import android.media.AudioManager;
import android.content.pm.PackageManager;

public void class MainActivity extends AppCompatActivity {

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      EdgeToEdge.enable(this);
      setContentView(R.layout.activity_MainActivity);
      ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets)
-> {
         Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
         v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom);
         return insets;
      });
   }
}
public class AudioHelper(context: Context) {
private val audioManager :
AudioManager=context.getSystemService(Context.AUDIO_SERVICE) as AudioManager;
fun audioOutputAvailable(type: int): Boolean {
      if (!
context.packageManager.hasSystemFeature(PackageManager.FEATURE_AUDIO_OUTPU
T)) {
         return false
});
      return audioManager.getDevices(AudioManager.GET_DEVICES_OUTPUTS).any
{it.type == type}
      });
```

```
}
fun main() {
   private val audioHelper =
AudioHelper(context.audioOutputAvailable.GET_DEVICES_OUTPUTS);

   val IsSpeakerAvailable =
audioHelper.audioOutputAvailable(AudioDeviceInfo.TYPE_BUILTIN_SPEAKER); {
        return true
}):
   val
isBluetoothHeadsetConnected=audioHelper.audioOutputAvailable(AudioDeviceInfo.TY
PE_BLUETOOTH_A2DP); {
        return true
});
}
audioManager.registerAudioDeviceCallback(object:AudioDeviceCallback() {

   override onAudioDevicesAdded(addedDevices:Array<out AudioDeviceInfo>?) {
      super.onAudioDevicesAdded(addedDevices);
         if (! audioOutputAvailable)(AudioDeviceInfo.TYPE_BUILTIN_SPEAKER)); {
         return 'Um alto falante acabou de ser conectado';
               else (! audioOutputAvailable)(AudioDeviceInfo.TYPE_ BLUETOOTH_A2DP)) {
               return'Um fone de ouvido Bluetooth acabou de ser conectado';
      );
   }
 }
override onAudioDevicesRemoved(removedDevices:Array<out AudioDeviceInfo>?) {
      super.onAudioDevicesRemoved(removedDevices);
         if (! audioOutputAvailable)(AudioDeviceInfo.TYPE_BUILTIN_SPEAKER)); {
               return 'Um alto falante não está mais conectado';
                  else (! audioOutputAvailable)(AudioDeviceInfo.TYPE_BLUETOOTH_A2DP))
{
                  return 'Um fone de ouvido Bluetooth não esta mais conectado';
         });
      }
   }
  override audioOutputAvailable(type: int): boolean {
   if (! IsSpeakerAvaliable = (AudioDeviceInfo.TYPE_BUILTIN_SPEAKER)) {
      return true
   };
   (! IsBluetoothHeadSetConnected = (AudioDeviceInfo.TYPE_BLUETOOTH_A2DP)) {
```

```
        return true
    };
}
```

Nota. O arquivo manifest.xml deu problema, então não foi prossivel fazer os testes. Mas deixo aqui o código.

Desculpe pelo atraso, a plataforma tava bugada e não recebi aviso de quando voltaria. Obrigado!