

Aluno: Altair Silva Filho

Matrícula: 202403254409

Unidade: Palhoça – SC

Curso: Desenvolvimento Full Stack

Disciplina: Interação Com Sensores de Smartphones e Wearables

Período: 2025.3

Trabalho Prático | Lidando com sensores em dispositivos móveis

Interação Com Sensores de Smartphones e Wearables

Procedimento | Aplicativo para uma Agência de Viagens

Objetivo

Para uma melhoria na eficiência e na comunicação interna, a empresa “Doma” quer desenvolver um aplicativo Wear OS para assistência aos funcionários que têm necessidades especiais, uma forma de solidificar a interação entre os mesmos.

Assim, com os aplicativos wearables podem usar áudio para fornecer informações em tempo real, como leitura de mensagens de texto, notificações, lembretes e respostas a comandos de voz. Isso pode ser especialmente útil para pessoas com deficiência visual.

Além de serem úteis para treinamento e educação. Aplicativos podem usar áudio para fornecer instruções, dicas e feedbacks durante o aprendizado ou a prática de novas habilidades.

Outra funcionalidade que a empresa quer adotar, é um aplicativo wearable que pode usar o áudio para fornecer alertas de segurança, como notificações de emergência, alertas de tempestades, notícias importantes ou informações críticas.

Assim, o objetivo do trabalho prático é ajudar os alunos a desenvolverem este aplicativo que proporciona essa comunicação eficaz.

Códigos do Projeto

Arquivos em Java

MainActivity.java

```
package com.example.listadetarefas;

import android.widget.ListView;
import android.content.Intent;
import android.media.AudioDeviceCallback;
import android.media.AudioDeviceInfo;
import android.media.AudioManager;
import android.os.Bundle;
import android.provider.Settings;
import android.widget.Button;
import android.widget.ArrayAdapter;
import androidx.appcompat.app.AppCompatActivity;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

    private ListView listView;
    private AudioManager audioManager;
    private AudioHelper audioHelper;
    private ArrayList<String> audioList = new ArrayList<>();
    private ArrayAdapter<String> adapter;

    private final AudioDeviceCallback audioCallback = new AudioDeviceCallback() {
        @Override
        public void onAudioDevicesAdded(AudioDeviceInfo[] addedDevices) {
            runOnUiThread(() -> {
                loadAudioOutputs();
                addStatus("Dispositivo adicionado");
            });
        }
    }

    @Override
    public void onAudioDevicesRemoved(AudioDeviceInfo[] removedDevices) {
        runOnUiThread(() -> {
            loadAudioOutputs();
            addStatus("Dispositivo removido");
        });
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        audioManager = (AudioManager) getSystemService(AUDIO_SERVICE);
        audioHelper = new AudioHelper(this);

        listView = findViewById(R.id.listView);
        adapter = new ArrayAdapter<>(this, android.R.layout.simple_list_item_1, audioList);
        listView.setAdapter(adapter);
    }
}
```

```

// Botão para Bluetooth
Button btnConfigBt = findViewById(R.id.btnConfigBt);
btnConfigBt.setOnClickListener(v -> {
    Intent intent = new Intent(Settings.ACTION_BLUETOOTH_SETTINGS);
    intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
    startActivity(intent);
});

// Carrega a lista inicial
loadAudioOutputs();
}

@Override
protected void onResume() {
    super.onResume();
    audioManager.registerAudioDeviceCallback(audioCallback, null);
}

@Override
protected void onPause() {
    super.onPause();
    audioManager.unregisterAudioDeviceCallback(audioCallback);
}

/**
 * Lista todas as saídas de áudio no ListView
 */
private void loadAudioOutputs() {
    audioList.clear();

    AudioDeviceInfo[] outputs = audioManager.getDevices(AudioManager.GET_DEVICES_OUTPUTS);

    for (AudioDeviceInfo device : outputs) {
        audioList.add(describeDevice(device));
    }

    // Mostrar status geral
    audioList.add("Alto-falante disponível: " +
        audioHelper.isBuiltInSpeakerAvailable());

    audioList.add("Bluetooth A2DP conectado: " +
        audioHelper.isBluetoothA2dpAvailable());

    adapter.notifyDataSetChanged();
}

//Mensagem de status
private void addStatus(String message) {
    audioList.add(0, "STATUS: " + message);
    adapter.notifyDataSetChanged();
}

```

```

//Conexão dos dispositivos
private String describeDevice(AudioDeviceInfo device) {
    switch (device.getType()) {
        case AudioDeviceInfo.TYPE_BUILTIN_SPEAKER:
            return "Alto-falante interno";
        case AudioDeviceInfo.TYPE_WIRED_HEADPHONES:
            return "Fones com fio";
        case AudioDeviceInfo.TYPE_BLUETOOTH_A2DP:
            return "Bluetooth (A2DP)";
        case AudioDeviceInfo.TYPE_BLUETOOTH_SCO:
            return "Bluetooth (SCO)";
        default:
            return "Outro dispositivo: " + device.getType();
    }
}
}
}

```

AudioHelper.java

```

package com.example.listadetarefas;

import android.content.Context;
import android.content.pm.PackageManager;
import android.media.AudioDeviceInfo;
import android.media.AudioManager;

public class AudioHelper {

    private final AudioManager audioManager;
    private final Context context;

    public AudioHelper(Context context) {
        this.context = context;
        this.audioManager = (AudioManager) context.getSystemService(Context.AUDIO_SERVICE);
    }

    public boolean audioOutputAvailable(int type) {

        if
        (!context.getPackageManager().hasSystemFeature(PackageManager.FEATURE_AUDIO_OUTPUT)) {
            return false;
        }

        AudioDeviceInfo[] devices = audioManager.getDevices(AudioManager.GET_DEVICES_OUTPUTS);

        for (AudioDeviceInfo d : devices) {
            if (d.getType() == type) return true;
        }

        return false;
    }
}

```

```
public boolean isBuiltInSpeakerAvailable() {  
    return audioOutputAvailable(AudioDeviceInfo.TYPE_BUILTIN_SPEAKER);  
}  
  
public boolean isBluetoothA2dpAvailable() {  
    return audioOutputAvailable(AudioDeviceInfo.TYPE_BLUETOOTH_A2DP);  
}  
}
```

Arquivo xml (Layout)

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="vertical"  
    android:padding="16dp">  
  
    <Button  
        android:id="@+id	btnConfigBt"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:text="Config Bluetooth" />  
  
    <ListView  
        android:id="@+id/listView"  
        android:layout_width="match_parent"  
        android:layout_height="200dp" />  
  
</LinearLayout>
```

Resultados da execução dos códigos

A janelas com o resultado do procedimento e execuções do código é mostrada a seguir:

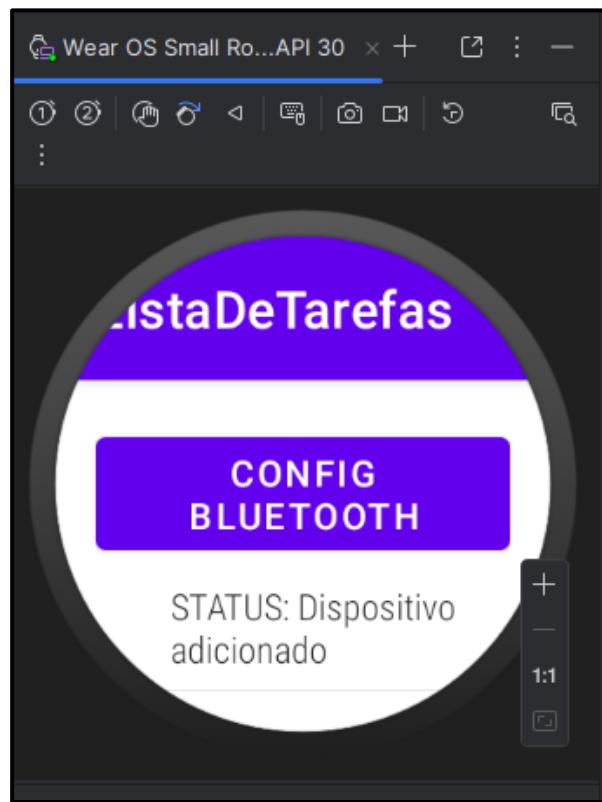


Figura 1 – Tela do emulador

Armazenado no Github: <https://github.com/altairsf/ListaDeTarefas.git>