Campus: POLO SAGUAÇU - JOINVILLE - SC

Curso: DESENVOLVIMENTO FULL STACK

Disciplina: RPG0025 - Lidando com sensores em dispositivos móveis

Turma: 9001

Aluno: Jederson Borges de Oliveira

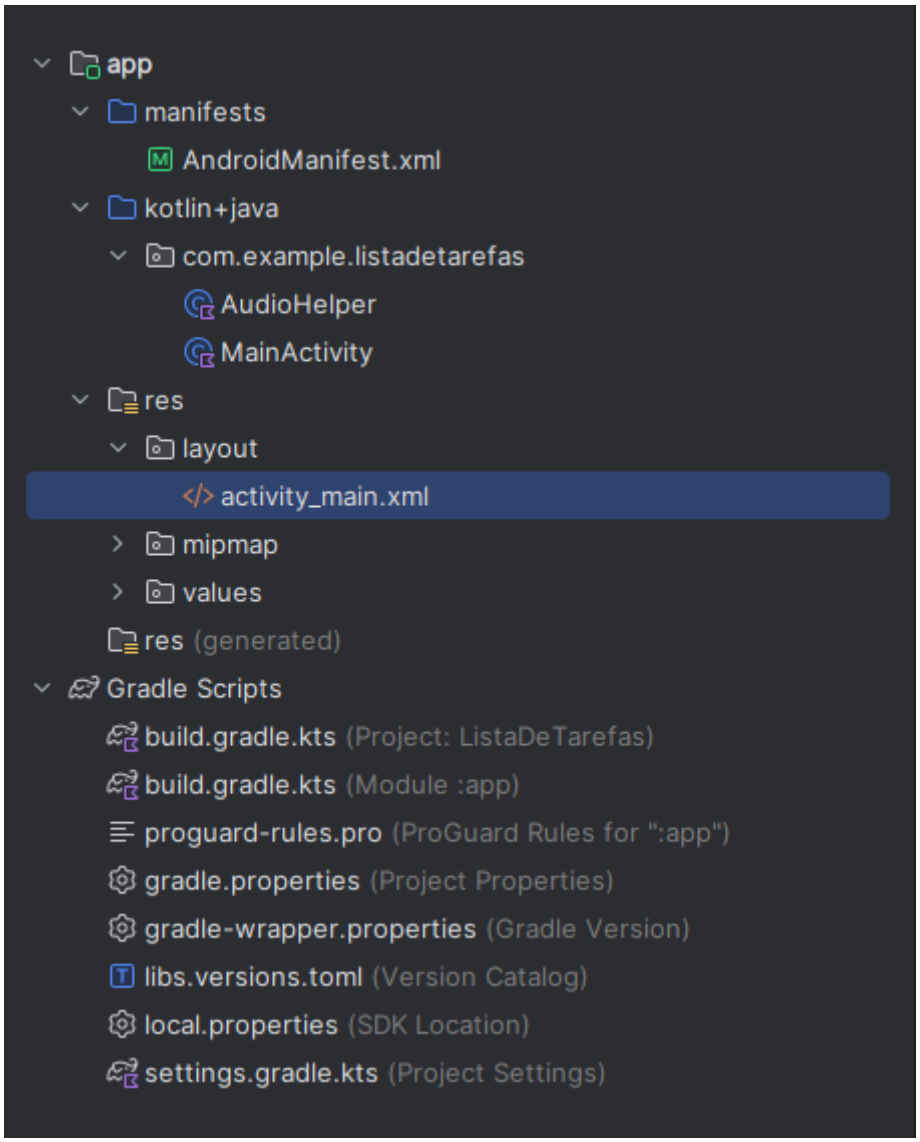
Link: https://github.com/JedersonBorges/estacio-RPG0025.git

Lidando com sensores em dispositivos móveis

Objetivos da prática

Instalação do Android Studio e do emulador
Criar um app para Wear OS
Executar um app no emulador
Fazer capturas de telas no Android Studio
Fazer capturas de telas com o app complementar

Estrutura:


app
manifests
AndroidManifest.xml
kotlin+java
com.example.listadetarefas
AudioHelper
MainActivity
res
layout
activity_main.xml
mipmap
values
res (generated)
Gradle Scripts
build.gradle.kts (Project: ListaDeTarefas)
build.gradle.kts (Module :app)
proguard-rules.pro (ProGuard Rules for ":app")
gradle.properties (Project Properties)
gradle-wrapper.properties (Gradle Version)
libs.versions.toml (Version Catalog)
local.properties (SDK Location)
settings.gradle.kts (Project Settings)

MainActivity:

```
package com.example.listadetarefas

import android.content.Context
import android.content.Intent
import android.media.AudioDeviceInfo
import android.media.AudioManager
import android.os.Bundle
import android.provider.Settings
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat
import android.widget.Toast

class MainActivity : AppCompatActivity() {

    private lateinit var audioHelper: AudioHelper

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_main)
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v, insets ->
            val systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom)
            insets
        }

        audioHelper = AudioHelper( context: this)


        if (!audioHelper.audioOutputAvailable(AudioDeviceInfo.TYPE_BLUETOOTH_A2DP)) {
            redirectToBluetoothSettings()
        } else {
            Toast.makeText( context: this,  text: "Fone Bluetooth conectado", Toast.LENGTH_SHORT).show()
        }
    }

    private fun redirectToBluetoothSettings() {
        val intent = Intent(Settings.ACTION_BLUETOOTH_SETTINGS).apply {
            addFlags( flags: Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK)
            putExtra( name: "EXTRA_CONNECTION_ONLY",  value: true)
            putExtra( name: "EXTRA_CLOSE_ON_CONNECT",  value: true)
            putExtra( name: "android.bluetooth.devicepicker.extra.FILTER_TYPE",  value: 1)
        }
        startActivity(intent)
    }
}
```

AudioHelper:

```
package com.example.listadetarefas

import android.content.Context
import android.media.AudioDeviceCallback
import android.media.AudioDeviceInfo
import android.media.AudioManager
import android.content.pm.PackageManager
import android.util.Log

class AudioHelper(private val 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_OUTPUT)) {
            return false
        }
        return audioManager.getDevices(AudioManager.GET_DEVICES_OUTPUTS).any { it.type == type }
    }


    fun registerAudioCallback() {
        audioManager.registerAudioDeviceCallback(object : AudioDeviceCallback() {
            override fun onAudioDevicesAdded(addedDevices: Array<out AudioDeviceInfo>?) {
                super.onAudioDevicesAdded(addedDevices)
                if (audioOutputAvailable(AudioDeviceInfo.TYPE_BLUETOOTH_A2DP)) {
                    Log.d( tag: "AudioHelper",  msg: "Fone de ouvido Bluetooth conectado!")
                }
            }

            override fun onAudioDevicesRemoved(removedDevices: Array<out AudioDeviceInfo>?) {
                super.onAudioDevicesRemoved(removedDevices)
                if (!audioOutputAvailable(AudioDeviceInfo.TYPE_BLUETOOTH_A2DP)) {
                    Log.d( tag: "AudioHelper",  msg: "Fone de ouvido Bluetooth desconectado!")
                }
            }
        },  handler: null)
    }
}
```

Build.grade.kts (module :app)

```
plugins {
    alias(libs.plugins.android.application)
    alias(libs.plugins.kotlin.android)
}

android {
    namespace = "com.example.listadetarefas"
    compileSdk = 34

    defaultConfig {
        applicationId = "com.example.listadetarefas"
        minSdk = 30
        targetSdk = 34
        versionCode = 1
        versionName = "1.0"
    }

    buildTypes {
        release {
            isMinifyEnabled = false
            proguardFiles(
                getDefaultProguardFile( name: "proguard-android-optimize.txt"),
                "proguard-rules.pro"
            )
        }
    }
    compileOptions {
        sourceCompatibility = JavaVersion.VERSION_11
        targetCompatibility = JavaVersion.VERSION_11
    }
    kotlinOptions {
        jvmTarget = "11"
    }
}

dependencies {
    implementation(libs.play.services.wearable)
    implementation(libs.appcompat)
    implementation(libs.material)
    implementation(libs.activity)
    implementation(libs.constraintlayout)
    implementation("androidx.recyclerview:recyclerview:1.2.1")
}
```