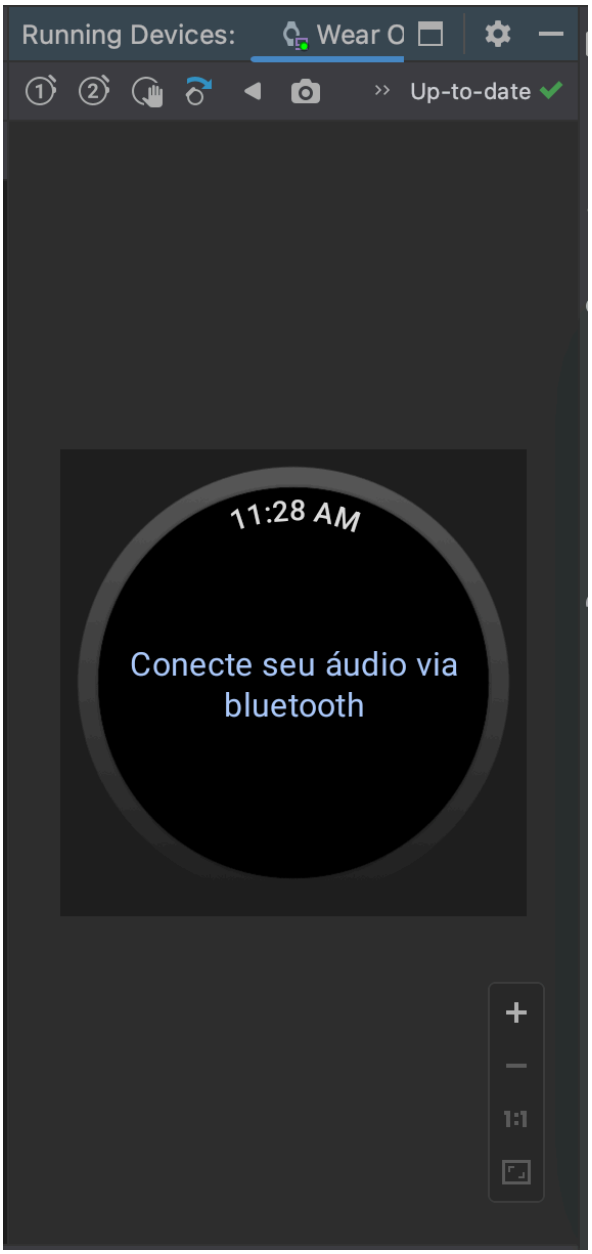
MISSÃO PRÁTICA - NÍVEL 3 - MUNDO 4

Aluno: Alexandre Henrique Fernandes Nolla

RELATÓRIO DO PROJETO

- **Aplicativo funcionando**



- **CustomAudioHelper** → Classe que fornece funcionalidades para verificar a disponibilidade de saída de áudio em um dispositivo Android Wear OS.

```
package com.example.mp3_wearos.presentation

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

class CustomAudioHelper(private val context: Context) {
```

```
    private val audioManager: AudioManager =
        context.getSystemService(Context.AUDIO_SERVICE) as AudioManager

    fun checkAudioOutput(type: Int): Boolean {
                                                                                if
(!context.packageManager.hasSystemFeature(PackageManager.FEATURE_AUDIO_OUTPUT)) {
            return false
        }
        return audioManager.getDevices(AudioManager.GET_DEVICES_OUTPUTS).any { it.type
== type }
    }
}
```

- **Classe MainActivity** → O presente código apresenta a estrutura de uma MainActivity destinada a um aplicativo voltado para o sistema operacional Wear OS. Assim que é inicializada, a função onCreate() é acionada, desempenhando um papel crucial na configuração da atividade.

  Neste ponto, um AudioDeviceCallback é registrado, desempenhando a função de monitorar constantemente a conexão e desconexão de dispositivos de áudio associados ao dispositivo. Se, durante a execução, um dispositivo Bluetooth compatível com o padrão A2DP for adicionado, uma notificação é prontamente exibida, informando ao usuário que o dispositivo está conectado.

  Por outro lado, se um dispositivo for removido, uma mensagem indicando que o dispositivo foi desconectado é imediatamente apresentada ao usuário.

```
package com.example.mp3_wearos.presentation

import android.content.Context
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 androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.core.splashscreen.SplashScreen.Companion.installSplashScreen
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.Box
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.res.stringResource
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.tooling.preview.Devices
import androidx.compose.ui.tooling.preview.Preview
import androidx.wear.compose.material.MaterialTheme
import androidx.wear.compose.material.Text
```

```
import androidx.wear.compose.material.TimeText
import com.example.mp3_wearos.R
import com.example.mp3_wearos.presentation.CustomAudioHelper
import com.example.mp3_wearos.presentation.theme.Mp3_wearosTheme
import android.widget.Toast

class MainActivity : ComponentActivity() {

   private lateinit var audioManager: AudioManager
   private lateinit var customAudioHelper: CustomAudioHelper
   private lateinit var context: Context

   override fun onCreate(savedInstanceState: Bundle?) {
       installSplashScreen()

       super.onCreate(savedInstanceState)

       setTheme(android.R.style.Theme_DeviceDefault)

       audioManager = getSystemService(Context.AUDIO_SERVICE) as AudioManager
       customAudioHelper = CustomAudioHelper(this)
       context = this

       val audioDeviceCallback = object : AudioDeviceCallback() {
           override fun onAudioDevicesAdded(addedDevices: Array<out AudioDeviceInfo>?)
{
               super.onAudioDevicesAdded(addedDevices)
                                                                                   if
(customAudioHelper.checkAudioOutput(AudioDeviceInfo.TYPE_BLUETOOTH_A2DP)) {
                       showCustomToast(context,"Conectado")
               }
           }

                       override  fun  onAudioDevicesRemoved(removedDevices:  Array<out
AudioDeviceInfo>?) {
               super.onAudioDevicesRemoved(removedDevices)
               if (removedDevices != null) {
                   for (device in removedDevices) {
                       if (device.type == AudioDeviceInfo.TYPE_BLUETOOTH_A2DP) {
                           promptCustomBluetoothConnection()
                           showCustomToast(context,"Desconectado")
                       }
                   }
               }
           }
       }

       audioManager.registerAudioDeviceCallback(audioDeviceCallback, null)

       setContent {
           CustomWearApp("Android")
       }
   }
```

```
    private fun showCustomToast(context: Context, message: String) {
        Toast.makeText(context, message, Toast.LENGTH_SHORT).show()
    }

    private fun promptCustomBluetoothConnection() {
        val intent = Intent(Settings.ACTION_BLUETOOTH_SETTINGS)
                                    intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK     or
Intent.FLAG_ACTIVITY_CLEAR_TASK)
        startActivity(intent)
    }
}

@Composable
fun CustomWearApp(greetingName: String) {
    Mp3_wearosTheme {
        Box(
            modifier = Modifier
                .fillMaxSize()
                .background(MaterialTheme.colors.background),
            contentAlignment = Alignment.Center
        ) {
            TimeText()
            CustomGreeting(greetingName = greetingName)
        }
    }
}

@Composable
fun CustomGreeting(greetingName: String) {
    Text(
        modifier = Modifier.fillMaxWidth(),
        textAlign = TextAlign.Center,
        color = MaterialTheme.colors.primary,
        text = stringResource(id = R.string.hello_world)
    )
}

@Preview(device = Devices.WEAR_OS_SMALL_ROUND, showSystemUi = true)
@Composable
fun CustomPreview() {
    CustomWearApp("Preview Android")
}
```