Relatório – Aplicativo Wear OS para Assistência com Áudio

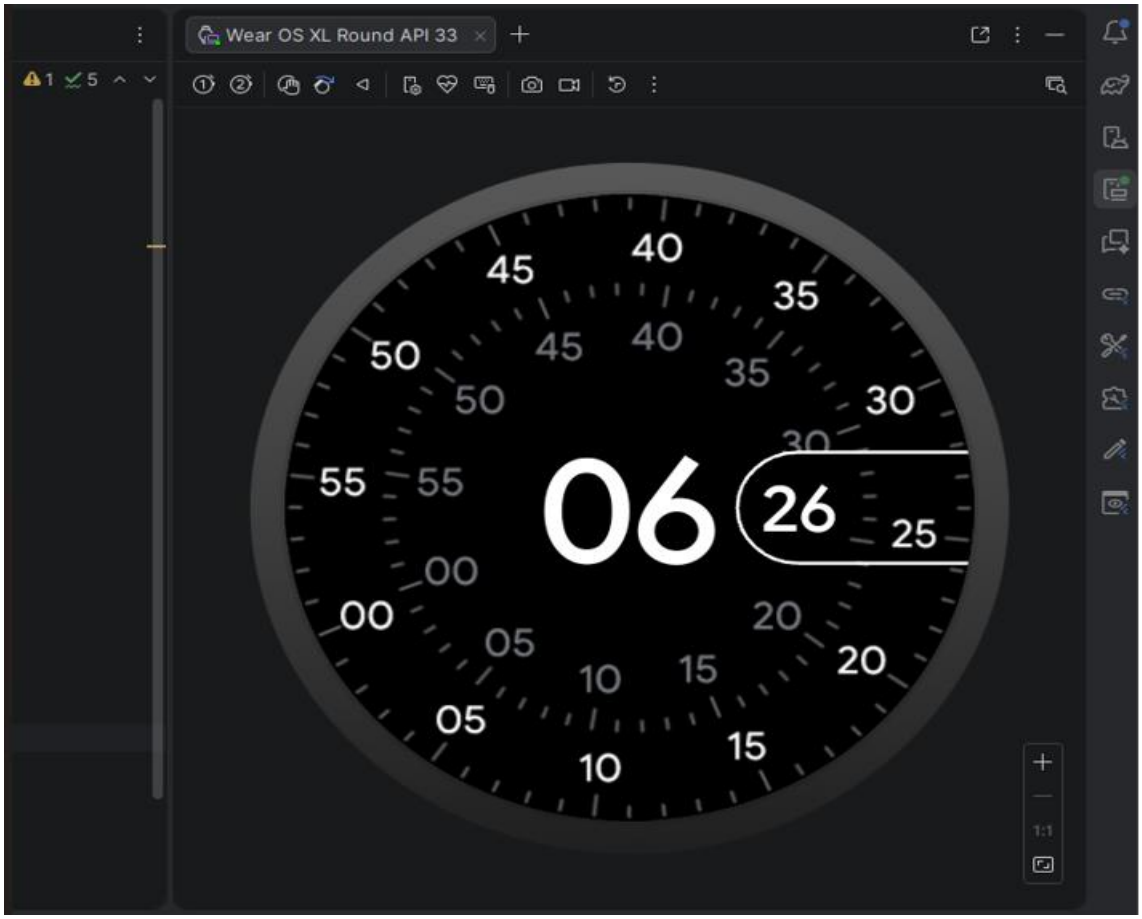
Gabriela Carolina Sobrinho - 2023 0617 1162

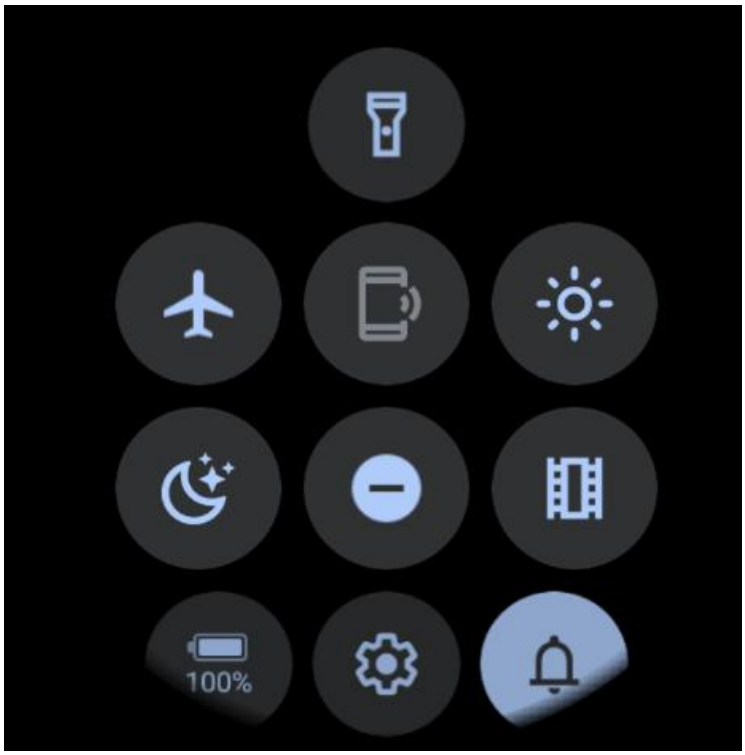
Polo jardim do mar – São Bernardo do campo – SP
Interação com sensores de smartphones e wearebles – 2025.4

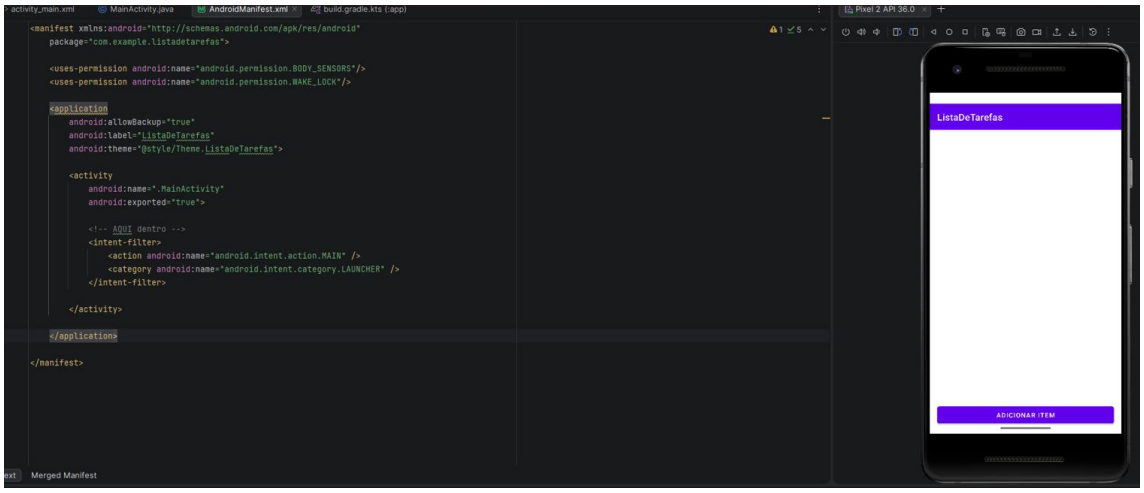
Objetivo da Prática

A empresa Doma queria melhorar a comunicação interna e aumentar a eficiência, especialmente aqueles com necessidades especiais. Nessa atividade foi proposta o desenvolvimento de um aplicativo para Wear OS que utiliza recursos de áudio para auxiliar.

Antes fiz todas as pequenas atividades e consegui obter um resultado bem legal e satisfatório.





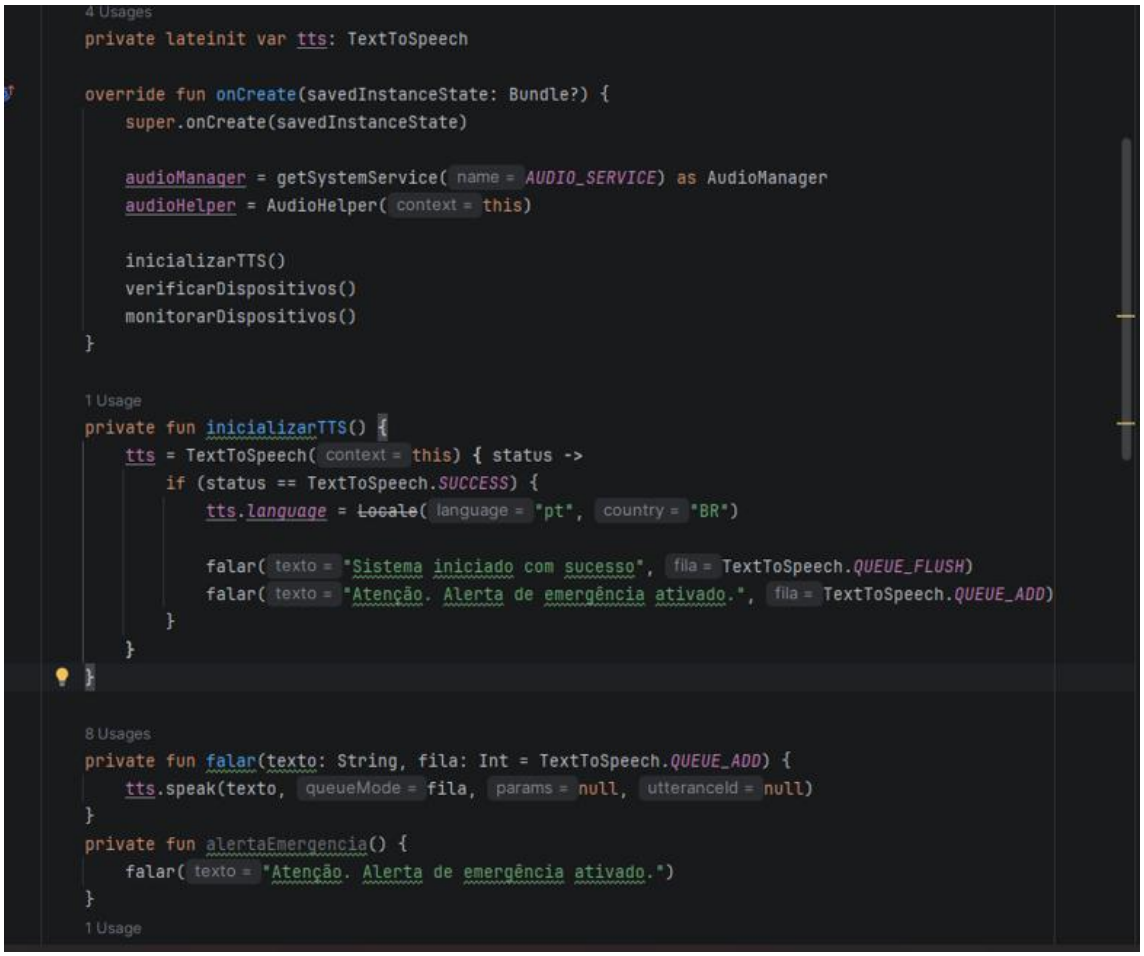


Voltando para a empresa Doma a ideia principal é usar como uma ferramenta de apoio, fornecendo informações em tempo real por meio de áudio.

Verificar se o dispositivo possui saída de áudio disponível. O aplicativo consegue identificar:

- Se o relógio possui alto-falante integrado
- Se há um fone de ouvido Bluetooth conectado

Para isso, foi utilizada a classe AudioManager junto com AudioDeviceInfo, permitindo verificar os dispositivos de saída disponíveis no sistema.



```
4 Usages
private lateinit var tts: TextToSpeech

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)

    audioManager = getSystemService( name = AUDIO_SERVICE) as AudioManager
    audioHelper = AudioHelper( context = this)

    inicializarTTS()
    verificarDispositivos()
    monitorarDispositivos()
}

1 Usage
private fun inicializarTTS() {
    tts = TextToSpeech( context = this) { status ->
        if (status == TextToSpeech.SUCCESS) {
            tts.language = Locale( language = "pt",  country = "BR")

            falar( texto = "Sistema iniciado com sucesso",  fila = TextToSpeech.QUEUE_FLUSH)
            falar( texto = "Atenção. Alerta de emergência ativado.",  fila = TextToSpeech.QUEUE_ADD)
        }
    }
}

8 Usages
private fun falar(texto: String, fila: Int = TextToSpeech.QUEUE_ADD) {
    tts.speak(texto,  queueMode = fila,  params = null,  utteranceId = null)
}
private fun alertaEmergencia() {
    falar( texto = "Atenção. Alerta de emergência ativado.")
}
1 Usage
```

Foi implementado um AudioDeviceCallback, que permite detectar quando:

- Um fone Bluetooth é conectado
- Um fone Bluetooth é desconectado

Isso garante que o aplicativo se adapte automaticamente à situação, oferecendo uma experiência mais inteligente e funcional.

```
private fun falar(texto: String) {
    tts.speak(texto, queueMode = TextToSpeech.QUEUE_FLUSH, params = null, utteranceId = null)
}
1 Usage
private fun alertaEmergencia() {
    falar( texto = "Atenção. Alerta de emergência ativado.")
}
1 Usage
private fun verificarDispositivos() {

    val speaker =
        audioHelper.audioOutputAvailable( type = AudioDeviceInfo.TYPE_BUILTIN_SPEAKER)

    val bluetooth =
        audioHelper.audioOutputAvailable( type = AudioDeviceInfo.TYPE_BLUETOOTH_A2DP)

    if (bluetooth) {
        falar( texto = "Fone Bluetooth conectado")
    } else if (speaker) {
        falar( texto = "Utilizando alto-falante do relógio")
    } else {
        falar( texto = "Nenhuma saída de áudio disponível")
    }
}
```

Facilitação da Conexão Bluetooth

Caso o aplicativo exija um fone Bluetooth para funcionar corretamente, ele não apenas informa o erro ao usuário.

Foi implementada uma funcionalidade que direciona o usuário automaticamente para as configurações de Bluetooth do dispositivo, facilitando a conexão.

Reprodução de Áudio

Depois de identificar uma saída de áudio válida, o aplicativo consegue reproduzir sons utilizando a classe MediaPlayer.

Os áudios podem ser usados para:

- Alertas de emergência
- Notificações importantes
- Instruções de treinamento
- Feedback educativo
- Avisos climáticos