Trabalho Prático | DGT2816 Interação com sensores de smartphones e wearebles

Material de **orientações** para desenvolvimento do **trabalho prático** da disciplina DGT2816 Interação com sensores de smartphones e wearebles

📍 **O trabalho prático deve ser feito individualmente.**

DGT2819 - Interação com sensores de smartphones e wearebles

Objetivos do trabalho prático

- 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 app complementar.

Entrega

- As microatividades irão dar suporte para o desenvolvimento do Trabalho Prático. Elas têm apoio/gabarito para resolução no próprio documento;
- A entrega esperada é o Trabalho Prático, descrito neste documento após as Microatividades;

👉 Atividades práticas

Lidando com sensores em dispositivos móveis

Microatividade 1: Implementar a visão geral e melhores práticas para acesso a sensores

- Material necessário para a prática

- **Android Studio:** Para o desenvolvimento de aplicativos Android.
- **Simulador Android ou iOS:** Para testar aplicativos no ambiente simulado.
- **Navegador Web:** Google Chrome, Firefox, MS Edge, Safari ou Opera.

- Procedimentos

1. **Instalação do Android Studio**

1.1 - Abra qualquer navegador da Web e acesse a página de download do Android Studio[1]. A página faz parte do site Android Developers, onde você pode fazer o download do Android Studio. Essa página detecta automaticamente seu sistema operacional.

1.2 - Clique em Fazer o download do Android Studio. A página Termos e Condições com o Contrato de licença do Android Studio vai ser aberta.

1.3 - Leia o Contrato de licença.

1.4 - Concorde com os Termos e Condições, marque a caixa de seleção Li e aceite os Termos e Condições acima na parte de baixo da página.

1.5 **-** Clique em Fazer o download do Android Studio para iniciar o download.

1.6 **-** Quando necessário, salve o arquivo em um local em que ele possa ser encontrado facilmente, como a pasta Downloads.

1.7 - Aguarde a conclusão do download. Talvez isso demore um pouco.

- **Windows:**

1. Abra a pasta onde você salvou o arquivo de instalação do Android Studio.
2. Clique duas vezes no arquivo.
3. Se a caixa de diálogo Controle da conta do usuário aparecer pedindo para permitir que a instalação faça mudanças no computador, clique em Sim para confirmar a instalação.

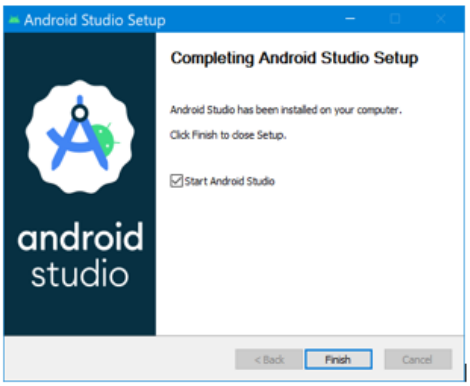
[1] https://developer.android.com/studio?hl=pt-br#get-android-studio

4. A caixa de diálogo Welcome to Android Studio Setup vai aparecer.

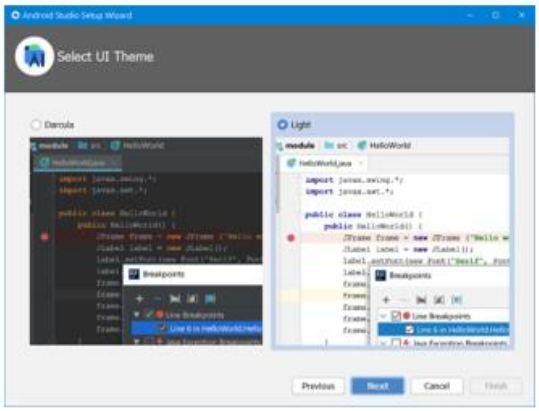
5. Clique em Next para iniciar a instalação.

6. Aceite as configurações de instalação padrão para todas as etapas.

7. Clique em Finish quando a instalação terminar para iniciar o Android Studio.



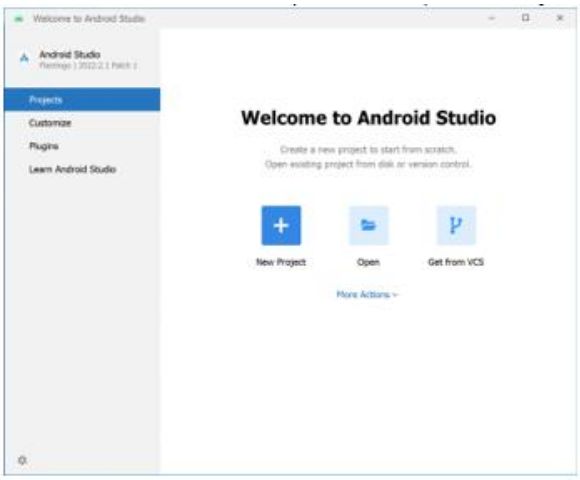
8. Escolha se prefere o tema claro ou escuro quando o Android Studio for iniciado pela primeira vez. As capturas de tela deste curso usam o tema claro, mas escolha o que você preferir.



9. Durante a instalação, o assistente de configuração vai fazer o download e a instalação de outros componentes e ferramentas necessários para o desenvolvimento de apps Android. Isso pode levar algum tempo, dependendo da sua velocidade de Internet. Durante esse período, a caixa de diálogo de Controle da conta do usuário pode ser exibida para o Windows Command Processor. Clique em Sim para aceitar.

10. Talvez você também receba um Alerta de segurança do Windows sobre o adb.exe. Clique em Permitir acesso para continuar a instalação, se necessário.

11. Quando o download e a instalação estiverem concluídos, clique em Finish. A janela **Welcome to Android Studio** vai aparecer e você poderá começar a criar apps.



Após a instalação, configure o emulador no Android Studio.

- **Linux:**

1. Extraia o arquivo baixado em /usr/local/ (para um único usuário) ou /opt/ (para usuários compartilhados).
2. No Terminal, navegue até a pasta descompactada, vá até a subpasta /bin e execute o arquivo studio.sh.

- **macOS:**

1. Abra a pasta onde você salvou o arquivo de instalação do Android Studio.
2. Clique duas vezes no arquivo. A caixa de diálogo abaixo vai aparecer:

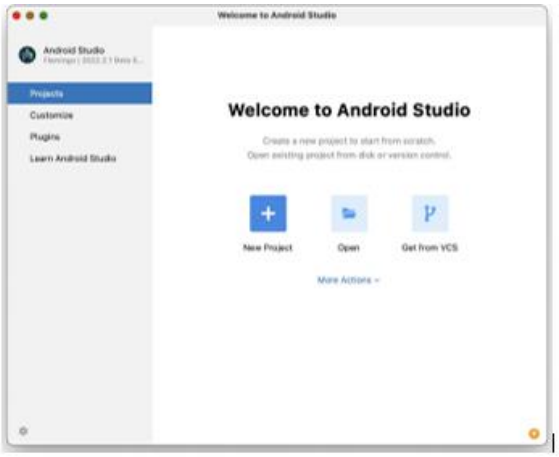
3. Arraste o ícone do Android Studio para a pasta Applications.

4. Na pasta Applications, clique duas vezes no ícone do Android Studio para abrir o Assistente de configuração do Android Studio.

5. Se você receber um aviso sobre a instalação ou execução de um arquivo transferido por download da Internet, aceite a instalação.

6. Siga as instruções do assistente de configuração do Android Studio e aceite as configurações padrão para todas as etapas. Durante a instalação, o assistente de configuração vai fazer o download e a instalação de outros componentes e ferramentas necessários para o desenvolvimento de apps Android. Isso pode levar algum tempo, dependendo da velocidade da sua Internet.

7. Quando a instalação for concluída, o Android Studio vai ser aberto automaticamente. A janela **Welcome to Android Studio** vai aparecer e você poderá começar a criar apps.



Após a instalação, configure o emulador no Android Studio.

Linux

1. Abra a pasta Downloads no terminal.

2. Extraia o arquivo usando o comando tar.

tar -xzvf android-studio-2022.2.1.20-linux.tar.gz

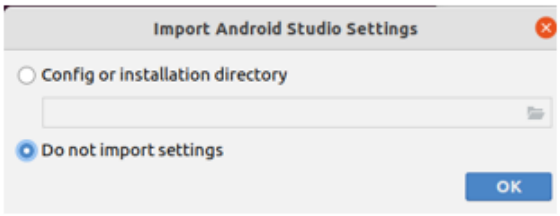
2. Navegue até o diretório android-studio/bin.

cd android-studio/bin

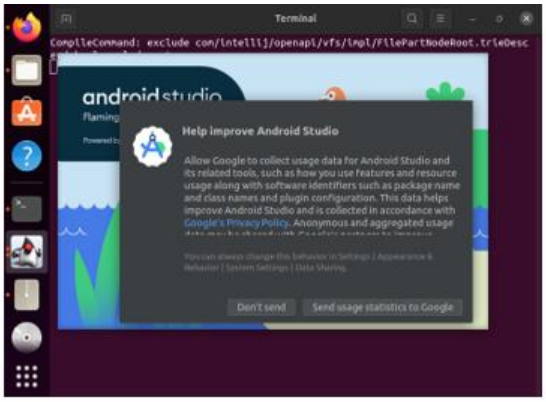
3. Execute o comando studio.sh

./studio.sh

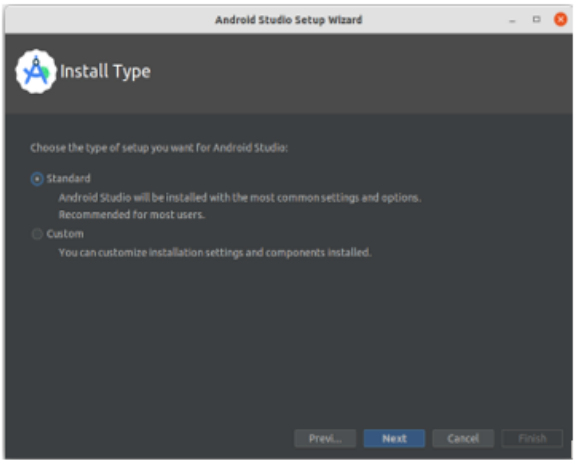
4. Mantenha a opção **Não importar configurações** (Do not import settings) selecionadas e clique em OK na solicitação.



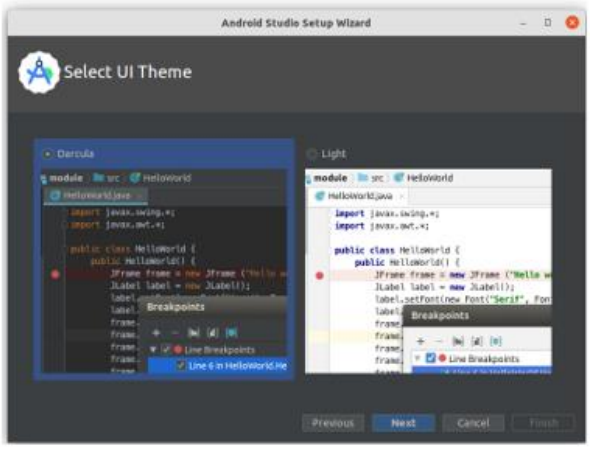
5. Escolha se quer ou não compartilhar dados de uso com o Google.



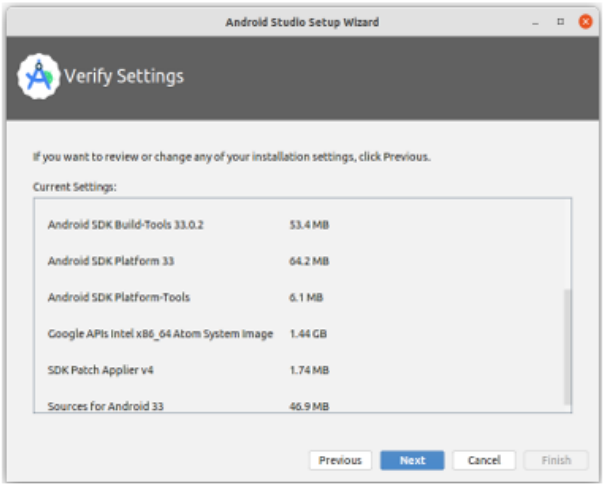
6. Mantenha o Standard como o tipo de instalação selecionado. Clique em Next para continuar.



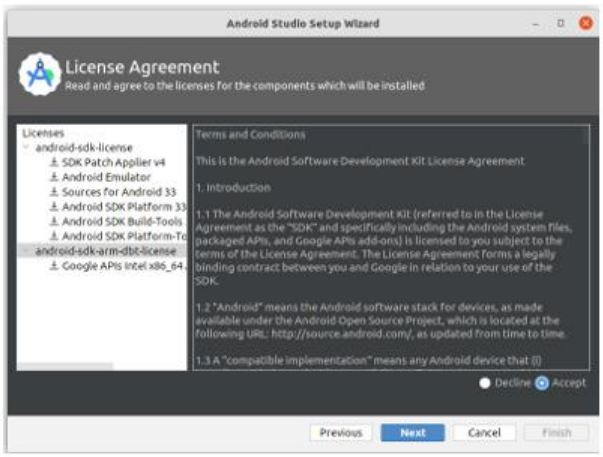
7. Escolha se prefere o tema claro ou escuro. As capturas de tela deste curso usam o tema claro, mas escolha o que você preferir. É possível mudar essa configuração quando quiser.



8. Aceite todas as configurações padrão e clique em Next.

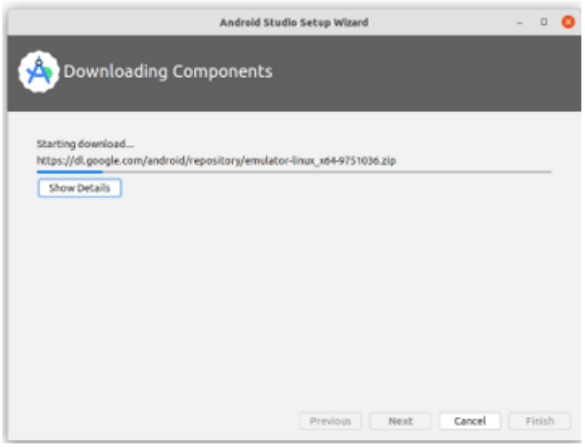


9. Leia e concorde com o Contrato de licença do SDK do Android e do Android NDK e clique em Next.

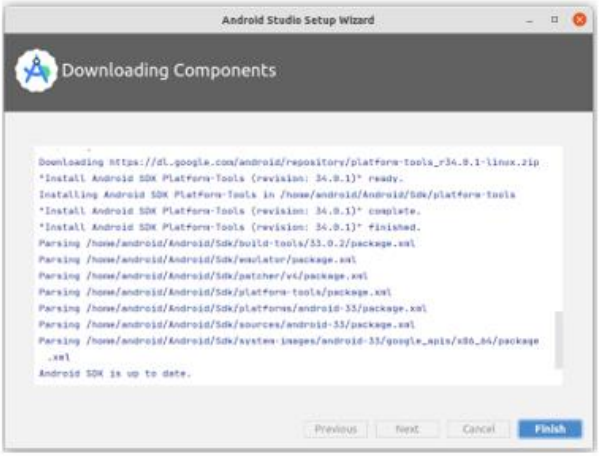


10. Você também pode conferir algumas outras informações sobre aceleração de hardware e o Android Emulator. Clique em Finish.

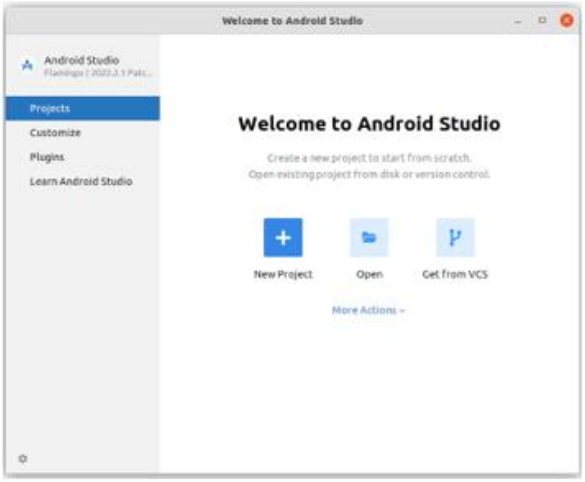
11. Durante a instalação, o assistente de configuração vai fazer o download e a instalação de outros componentes e ferramentas necessários para o desenvolvimento de apps Android.



Quando a instalação estiver concluída, clique em Finish.



12. A caixa de diálogo Welcome to Android Studio vai aparecer e você poderá começar a criar apps.



Após a instalação, configure o emulador no Android Studio.

2. **Criando um Emulador no Android Studio**

- Resultados esperados ✨

É esperado que com essa microatividade o aluno desenvolva passo a passo a instalação do Android Studio e configurar o emulador para poder aplicar algumas ferramentas do app.

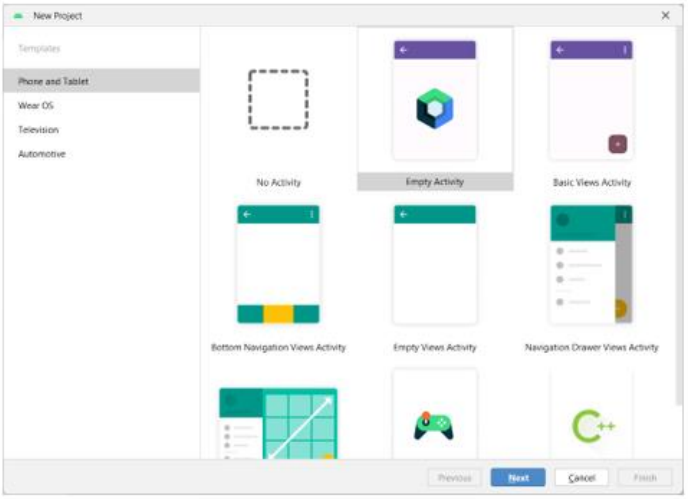
Microatividade 2: Criando um novo projeto no Android Studio

- Material necessário para a prática

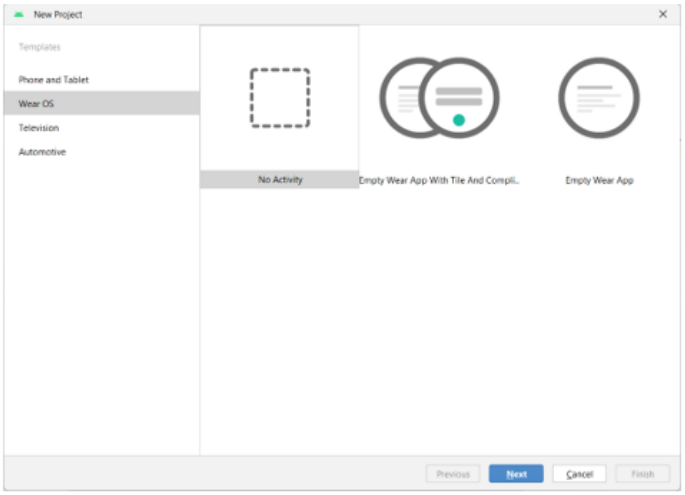
- **Android Studio:** Para o desenvolvimento de aplicativos Android.
- **Simulador Android ou iOS:** Para testar aplicativos no ambiente simulado.
- **Navegador Web:** Google Chrome, Firefox, MS Edge, Safari ou Opera.

- **Procedimentos** 💻

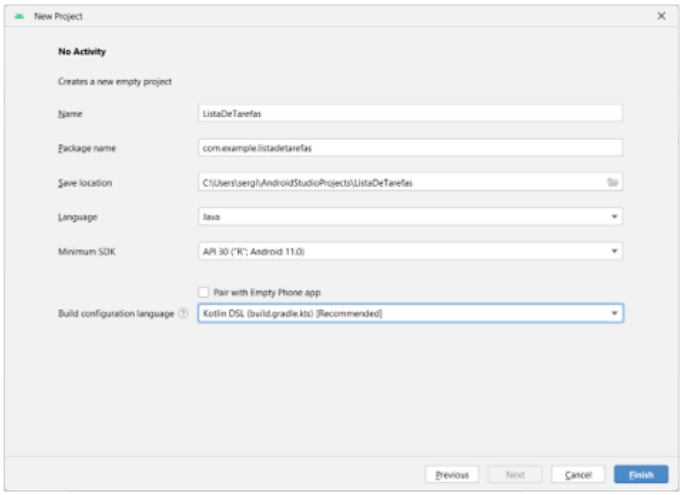
1. Abra o Android Studio e acesse File > New > New Project. A janela New Project vai aparecer.



2. No painel Templates, selecione Wear OS. Em seguida, no painel principal, selecione selecione o modelo "No Activity" e clique em "Next".



3. Em Name, nós vamos utilizar "ListaDeTarefas" para esse exemplo. No campo "Package name", o próprio Android Studio irá sugerir algo baseado no nome do projeto, como "com.example.listadetarefas". Em "Minimum SDK", utilizaremos a API 30: Android 11.0 (R), por ser a mais recente, depois basta clicar em “Finish” e o Android Studio criará o projeto para você.



- Resultados esperados ✨

Esta microatividade permitirá que o aluno execute os passos iniciais para criar seu primeiro aplicativo para Wear OS. Ao seguir esses procedimentos, o aluno terá configurado um projeto usando um modelo do Android Studio e estará pronto para iniciar o desenvolvimento do aplicativo.

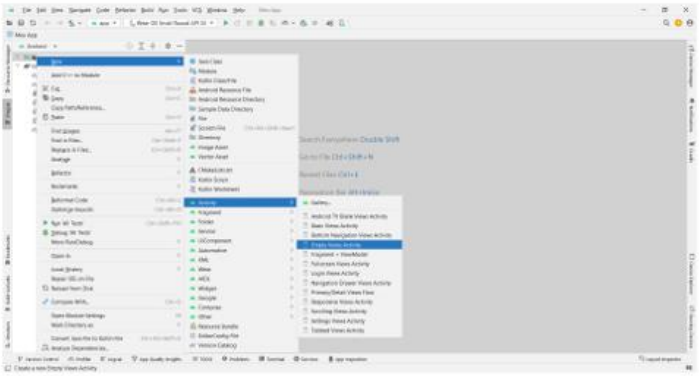
Microatividade 3: Arquivos de Lógica e Configurações

- Material necessário para a prática

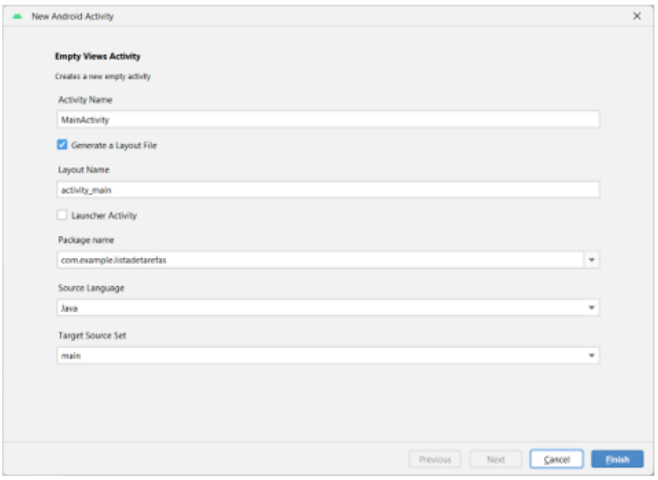
- **Editor de Texto ou IDE:** Recomenda-se o uso do VS Code**.**
- **Flutter SDK:** Necessário para a utilização da ferramenta Flutter.
- **Android Studio:** Para o desenvolvimento de aplicativos Android.
- **Simulador Android ou iOS:** Para testar aplicativos no ambiente simulado.
- **Navegador Web:** Google Chrome, Firefox, MS Edge, Safari ou Opera.

- Procedimentos 💻

1. Criação da MainActivity: Crie a `MainActivity.java` clicando com o botão direito em "app" e selecione New > Activity > Empty Views Activity..



2. Nome e Layout da Atividade: Na janela, mantenha o nome da atividade como `MainActivity` e o "Layout Name" como `activity_main`.



3. Interface de Usuário: Desenvolva a interface da primeira tela do aplicativo com uma `ListView` e um `Button`.

4. Permissões no AndroidManifest.xml: Localize o `AndroidManifest.xml` na pasta manifests e adicione as permissões:

<uses-permission android:name="android.permission.BODY_SENSORS"/>

<uses-permission android:name="android.permission.WAKE_LOCK"/>

Isso permite a interação com partes do sistema.

5. Intent-filter para MainActivity: No arquivo maAdicione o elemento `intent-filter` para especificar as intents que a atividade pode responder, respondendo a intents com a ação MAIN e a categoria LAUNCHER. Exemplo:

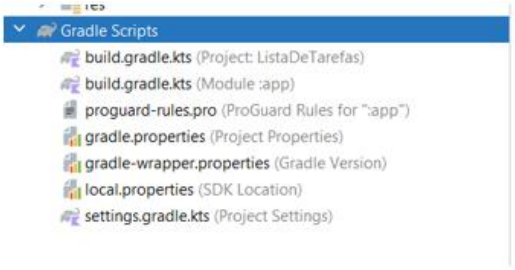
<intent-filter>

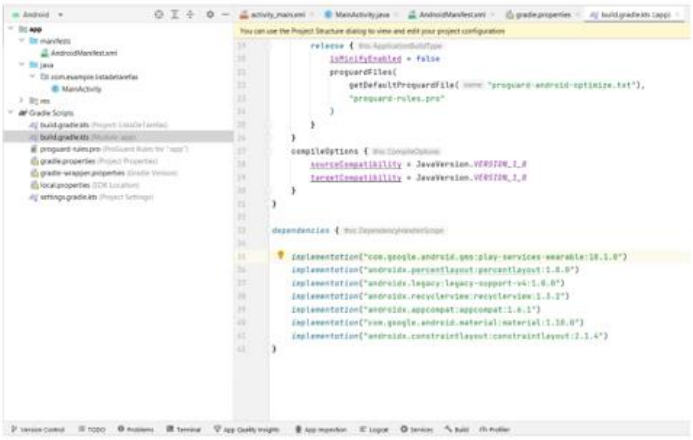
<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

6. Dependências no build.gradle: Na área Gradle Scripts, temos o build.gradle, e lá encontraremos as dependências do projeto.





Lembre-se de sincronizar o projeto após realizar essas alterações para garantir que as dependências sejam baixadas corretamente.

- Resultados esperados ✨

Nesta microatividade o aluno aprenderá os primeiros passos para criação do aplicativo. No Android Studio precisamos configurar alguns arquivos com informações do aplicativo e do dispositivo para o qual iremos desenvolver.

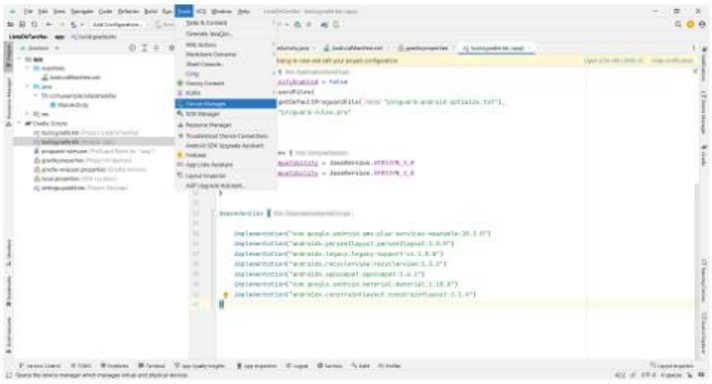
Microatividade 4: Criando um emulador

- Material necessário para a prática

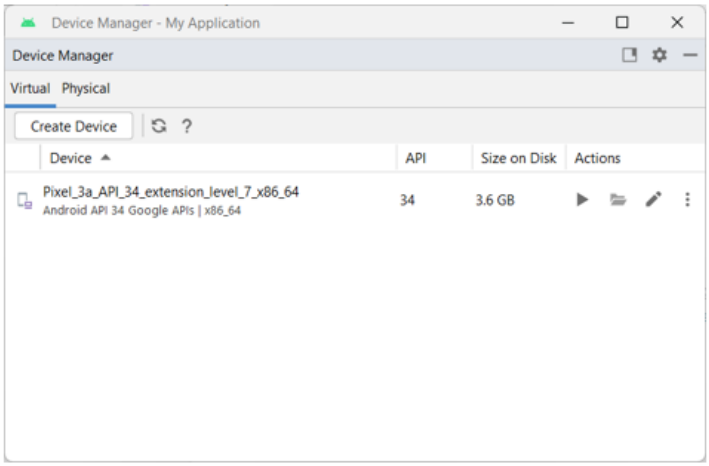
- Editor de texto ou IDE sendo opções sugeridas: VS Code;
- Flutter SDK, o arquivo que permite utilizar a ferramenta;
- Android Studio e/ou xCode;
- Simulador Android ou iOS.
- Navegador Web: Google Chrome, Firefox, MS Edge, Safari ou Opera.

- Procedimentos 💻

1. No Android Studio[2], acesse o Device Manager pelo caminho Tools > Device Manager. É um botão do lado direito da barra de ferramentas que mostra um Android abrindo a cabeça ao lado de um dispositivo com um display roxo.



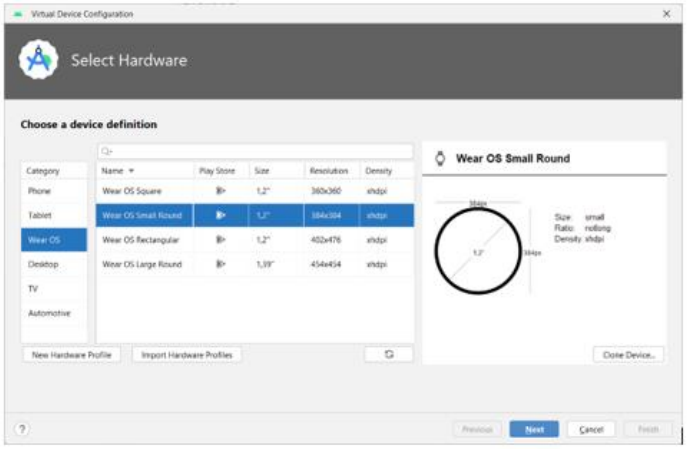
2. Depois que o Device Manager abrir, provavelmente você verá um emulador já criado e alguns detalhes sobre ele, principalmente o tipo de emulador, a API que está sendo usada e o tipo de CPU. Importante o Device Manager pode abrir como uma janela dentro do Android Studio ou como uma janela flutuante.



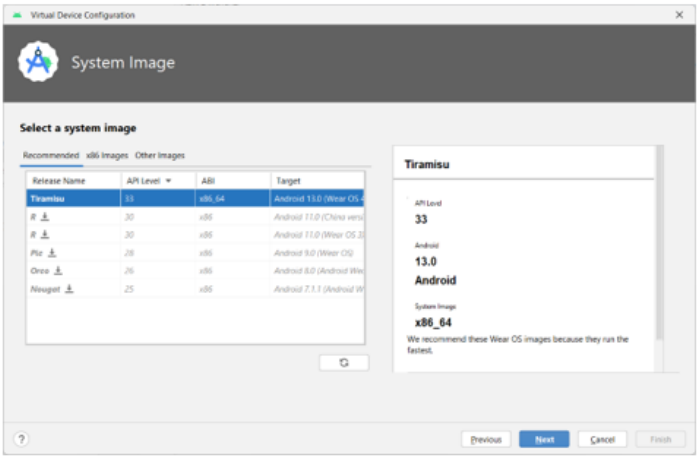
3. Para entender melhor entendimento desse processo, vamos criar um novo dispositivo virtual:

- Clique em Create Device, escolha a categoria Wear OS no lado esquerdo. Selecione o hardware que deseja emular (no nosso exemplo, Wear OS Small Round). Clique em Next

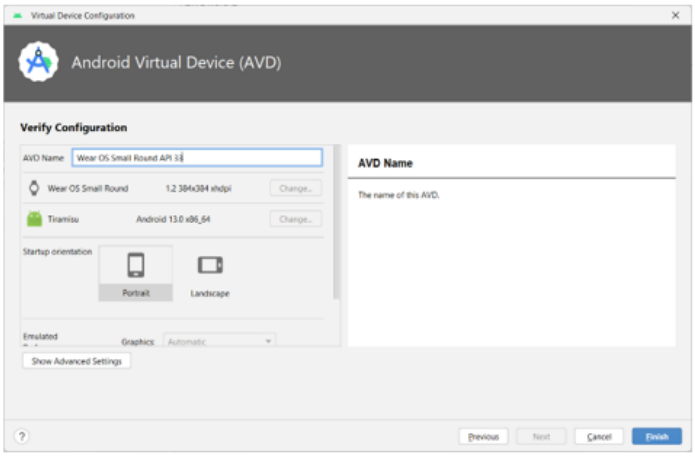
[2] https://www.androidpro.com.br/blog/android-studio/android-studio-configurando-ambiente/



4. Escolha o sistema operacional que você deseja emular (por exemplo, Wear OS API 30). Se a imagem do sistema não estiver disponível, clique no link “Download” ao lado do nome para baixá-lo. Após selecionar a imagem do sistema, clique no botão Next.



5. A última tela permite confirmar suas escolhas e oferece opções para configurar algumas outras propriedades, como nome do dispositivo, orientação de inicialização e tamanho da memória RAM. Por enquanto, use os padrões e clique em Finish.



- Resultados esperados ✨

Esta microatividade destaca como criar emuladores de dispositivos Wearable, permitindo testar o funcionamento de aplicativos. Isso é útil para o desenvolvimento e teste de aplicativos Wear OS antes de implantá-los em dispositivos reais.

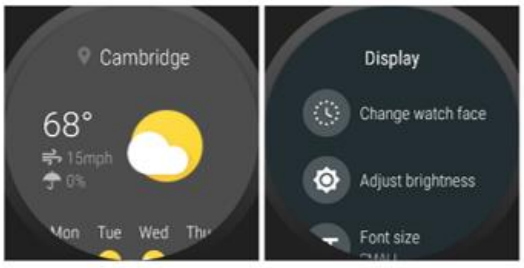
Microatividade 5: Fazer capturas de telas com app complementar

- Material necessário para a prática

- Editor de texto ou IDE sendo opções sugeridas: VS Code;
- Flutter SDK, o arquivo que permite utilizar a ferramenta;
- Android Studio e/ou xCode;
- Simulador Android ou iOS.
- Navegador Web: Google Chrome, Firefox, MS Edge, Safari ou Opera.

- Procedimentos 💻

1. Na IU, encontre a tela que você quer capturar.
2. No smartphone Android, ative as Opções do desenvolvedor, se ainda não estiverem ativadas. Para isso, acesse Configurações > Sobre o telefone e toque em Número da versão sete vezes.
3. Abra o app complementar do Wear no smartphone.
4. Toque no botão flutuante de três pontos no canto superior direito para abrir o menu.
5. Toque em “Fazer captura de tela do wearable”. Esta mensagem vai aparecer: Solicitação de captura de tela enviada. Depois, você receberá estas notificações: Pronto para enviar uma captura de tela do relógio e Toque para enviar.
6. Toque na notificação para conferir as opções de envio ou compartilhamento da captura de tela por Bluetooth, Gmail ou outros meios.



- Resultados esperados ✨

Com esta microatividade o aluno compreenderá outra forma de realizar a captura de tela da UI app para wearables.

Trabalho Prático | Lidando com sensores em dispositivos móveis

Nesta atividade a seguir compreenderemos que os apps do Wear OS podem funcionar como um dos principais frameworks para o desenvolvimento de aplicações mobile. Um aplicativo Wearable pode ter várias especialidades, desde entretenimento e comunicação.

Contextualização

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.

Roteiro de prática

- Material necessário para a prática

- Editor de texto ou IDE sendo opções sugeridas: VS Code;
- Flutter SDK, o arquivo que permite utilizar a ferramenta;
- Android Studio e/ou xCode;
- Simulador Android ou iOS.
- Navegador Web: Google Chrome, Firefox, MS Edge, Safari ou Opera.

- Procedimentos 💻

1. **Configuração do Ambiente:**

- Certifique-se de ter seu ambiente configurado.

- Prepare um ambiente de simulação para Wear OS ou conecte um dispositivo wearable real.

2. **Implementação de Saídas de áudio 🔊:**

- AudioDeviceInfo.TYPE_BUILTIN_SPEAKER[3], em dispositivos com um alto-falante integrado.
- AudioDeviceInfo.TYPE_BLUETOOTH_A2DP[4] quando um fone de ouvido Bluetooth estiver pareado e conectado.
- Utilize o método getDevices()[5] com o valor de FEATURE_AUDIO_OUTPUT para enumerar todas as saídas de áudio:

```
import android.content.Context

import android.media.AudioDeviceInfo

import android.media.AudioManager

import android.content.pm.PackageManager


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_OUTPUT))
{

         return false

      }


      return audioManager.getDevices(AudioManager.GET_DEVICES_OUTPUTS).any { it.type
== type }
```

[3] https://developer.android.com/reference/android/media/AudioDeviceInfo?hl=pt-br#TYPE_BUILTIN_SPEAKER
[4] https://developer.android.com/reference/android/media/AudioDeviceInfo?hl=pt-br#TYPE_BLUETOOTH_A2DP
[5] https://developer.android.com/reference/android/media/AudioManager?hl=pt-br#getDevices(int)

```
    }
}

// Exemplo de uso
fun main() {
    val audioHelper = AudioHelper(context) // Substitua 'context' pelo contexto atual do seu aplicativo

    val isSpeakerAvailable = audioHelper.audioOutputAvailable(AudioDeviceInfo.TYPE_BUILTIN_SPEAKER)
    // True if the device has a speaker

    val isBluetoothHeadsetConnected = audioHelper.audioOutputAvailable(AudioDeviceInfo.TYPE_BLUETOOTH_A2DP)
    // True if a Bluetooth headset is connected
}
```

3. **Detecção Dinâmica de Dispositivos de Áudio:**

   - Seu app pode registrar um callback para detectar quando isso acontece usando registerAudioDeviceCallback[6]:

```
// Supondo que 'audioManager' já tenha sido inicializado
audioManager.registerAudioDeviceCallback(object : AudioDeviceCallback() {
    override fun onAudioDevicesAdded(addedDevices: Array<out AudioDeviceInfo>?) {
        super.onAudioDevicesAdded(addedDevices)
        if (audioOutputAvailable(AudioDeviceInfo.TYPE_BLUETOOTH_A2DP)) {
            // Um fone de ouvido Bluetooth acabou de ser conectado
        }
    }
```

[6] https://developer.android.com/reference/android/media/AudioManager?hl=pt-br#registerAudioDeviceCallback(android.media.AudioDeviceCallback, android.os.Handler)

```
    override fun onAudioDevicesRemoved(removedDevices: Array<out AudioDeviceInfo>?) {
        super.onAudioDevicesRemoved(removedDevices)
        if (!audioOutputAvailable(AudioDeviceInfo.TYPE_BLUETOOTH_A2DP)) {
            // Um fone de ouvido Bluetooth não está mais conectado
        }
    }
}, null)

fun audioOutputAvailable(type: Int): Boolean {
    // Implementação da função audioOutputAvailable
    // Retorna verdadeiro se o tipo de dispositivo de áudio especificado estiver disponível
}
```

4. **Facilitando a Conexão Bluetooth:**

   - Se o app exigir que um fone de ouvido seja conectado para continuar, em vez de mostrar uma mensagem de erro, ofereça a opção de direcionar o usuário diretamente às configurações do Bluetooth para facilitar a conexão. Para isso, envie uma intent com ACTION_BLUETOOTH_SETTINGS[7]:

```
val intent = with (Intent(Settings.ACTION_BLUETOOTH_SETTINGS)) {
    addFlags(Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK)
    putExtra("EXTRA_CONNECTION_ONLY", true)
    putExtra("EXTRA_CLOSE_ON_CONNECT", true)
    putExtra("android.bluetooth.devicepicker.extra.FILTER_TYPE", 1)
}
startActivity(intent)
```

[7] https://developer.android.com/reference/android/provider/Settings?hl=pt-br#ACTION_BLUETOOTH_SETTINGS

5. **Reprodução de Áudio:**

   - Depois de detectar uma saída de áudio adequada, o processo para tocar áudio no Wear OS é o mesmo usado em dispositivos móveis ou outros dispositivos.

6. **Uso de Alto-falantes em Dispositivos Wear OS:**

   - Para dispositivos Wear OS que incluem alto-falantes, incorpore funcionalidades de áudio para enriquecer a experiência do usuário.
   - Exemplos de uso incluem alarmes de relógio com notificações sonoras, apps de fitness com instruções de voz para exercícios, e apps educativos com feedback auditivo.

- Resultados esperados ✨

Ao concluir esta missão, os alunos terão desenvolvido um aplicativo Wear OS que proporciona uma comunicação eficaz e assistência para funcionários com necessidades especiais. O aplicativo deverá ser capaz de ler mensagens e notificações em voz alta, responder a comandos de voz e fornecer alertas de segurança e instruções através de áudio. Este aplicativo não apenas melhora a eficiência e a comunicação interna na empresa "Doma", mas também demonstra a aplicação prática de tecnologias wearables para criar soluções acessíveis e inclusivas no local de trabalho.

📌*Referências*

Não foram utilizadas referências bibliográficas para a elaboração das atividades.

Entrega do trabalho prático

Chegou a hora, gamer!

✍️ Armazene o projeto em um repositório no GIT.

✍️ Anexar a documentação do projeto (PDF) no GIT.

✍️ Compartilhe o link do repositório do GIT com o seu tutor para correção da prática, por meio da **Sala de Aula Virtual**, na aba "**Trabalhos**" do respectivo nível de conhecimento.

✍️ **Ei, verifique o prazo de entrega deste trabaho pois não recebemos trabalhos fora do prazo!**