Android Kotlin Developer

Profº Raphael Melo (profraphael.silva@fiap.com.br)

Checkpoint 2 – 2º Semestre 2023 – 3SI

Activity, Corrotinas, ViewModel, LiveData e Constraint Layout Individual

Obs.: É importante que seu programa rode. Caso tenha algum erro, inutilize a linha com o erro utilizando “//”, dessa forma consigo avaliar os outros itens que estão funcionando.

Upload: Ao finalizar o seu aplicativo compacte o projeto inteiro clicando com o botão direito do mouse sobre a pasta e escolha comprimir (O arquivo comprimido deve conter o mesmo nome seguindo o modelo no item 1 da prova). Depois faça upload na área de entrega de trabalhos.

Importante: Caso o seu projeto seja cópia do material disponibilizado ou de outro aluno, **ambas as provas serão zeradas**!

1 – Crie um projeto do tipo **Empty Views Activity** com o seguinte modelo de nome: **SEUNOME_SEUSOBRENOME_RMXXXXX**.

Por exemplo: **RAPHAEL_MELO_RM12345**.

Package: **br.com.fiap.corrida;**

Linguagem: **Kotlin**;

Minimum SDK **API 26 (Android 8.0 Oreo).**

2 – Desenvolva uma activity que vai simular o progresso de uma corrida de cavalos.

Regras:

- Exibir um TextView centralizado no topo com a seguinte string: “Corrida de cavalos”. **(0,5 ponto)**
- Exibir dois botões, um para iniciar a corrida, e outro para parar a corrida. **(0,5 ponto)**
- Exibir as informações de dois cavalos, sendo elas: O Nome e a cor do cavalo, e uma barra de progresso que exiba o progresso do cavalo na corrida e tenha a mesma cor do cavalo. **(1 ponto)**
- No início da corrida, as barras de progresso de cada cavalo devem estar zeradas. **(0,5 ponto)**
- Ao clicar no botão iniciar, as duas barras de progresso devem iniciar a corrida simultaneamente. **(0,5 ponto)**
- Ao clicar no botão parar, as barras de progresso devem parar de incrementar. **(1 pontos)**
- Utilize corrotinas para gerar valores aleatórios de 0 a 20 que devem ser gerados fora do thread principal (Main Thread). Esses valores devem ser incrementados em cada barra de progresso e devem ser gerados de acordo com um delay de 1500 ms (1,5 segundo). Cada barra de progresso deve ser incrementada com um valor aleatório diferente. Sua lógica deve ser feita dentro de uma ViewModel.
  (5 pontos)
- No final da corrida, quando a primeira barra de progresso atingir 100%, exiba um TextView informando qual cavalo venceu a corrida. **(1 ponto)**

Pontos importantes:

Para ativar o **viewBinding** no seu projeto, não esqueça de habilitar o uso dele no arquivo **app/build.gradle** e sincronizar o projeto.

```
buildFeatures {
    viewBinding true
}
```

Adicione as dependências da Corrotina e ViewModel em seu arquivo **app/build.gradle**

def lifecycle_version = "2.5.1"

def coroutine_version = "1.6.1"

implementation("androidx.lifecycle:lifecycle-runtime-ktx:$lifecycle_version")

implementation("org.jetbrains.kotlinx:kotlinx-coroutines-core:$coroutine_version")

implementation("org.jetbrains.kotlinx:kotlinx-coroutines-android:$coroutine_version")

implementation 'androidx.lifecycle:lifecycle-viewmodel-ktx:2.6.1'

implementation 'androidx.activity:activity-ktx:1.7.2'