Unioeste - Universidade Estadual do Oeste do Paraná
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
Colegiado de Ciência da Computação
Curso de Bacharelado em Ciência da Computação

Documentação da Atividade Prática Avaliativa AP02

Isadora Coelho, Marlon Pereira, Ronaldo Drecksler
Docente: Leonardo Medeiros

CASCAVEL - PR
5 de Janeiro de 2023

SUMÁRIO

1. Introdução

Para esse trabalho, foram desenvolvidos três versões diferentes de um mesmo aplicativo de cálculo de fatorial, uma delas usando o Android Studio com Framework Flutter e Dart, a segunda usando Android Studio com Java e a última por meio da linha de comando

2. Android Studio com Framework Flutter

Essa aplicação, assim como as demais, busca realizar o cálculo do número fatorial de um número. As três etapas requeridas são a entrada de dados, o processamento de dados, a saída de dados, respectivamente, atendidas pelo input do número, o cálculo do fatorial e o output do número.

2.1 Sobre o Flutter

Flutter é um framework de desenvolvimento de aplicativos móveis, lançado em 2017 pela Google. Ele permite que os desenvolvedores criem aplicativos nativos para iOS e Android usando uma única base de código. Flutter utiliza a linguagem de programação Dart e sua própria máquina virtual, permitindo que os aplicativos tenham uma performance nativa e uma interface de usuário altamente personalizável. O framework é conhecido por sua facilidade de uso, rapidez de desenvolvimento e por oferecer uma grande variedade de widgets prontos para uso. Ele é amplamente utilizado para desenvolver aplicativos móveis de alta qualidade e tem uma comunidade crescente de desenvolvedores em todo o mundo.

Foi usado o Android Studio para fins de emular um dispositivo Android, foi usado um plugin para que o Android Studio pudesse dar suporte ao Flutter e a linguagem Dart, na qual o Flutter é construído.

2.2 Arquitetura

Foi utilizada a arquitetura MVC – Model, View, Controller. No entanto como a camada model é desnecessária para essa aplicação, dada a simplicidade(vale lembrar que o objetivo da atividade é a exposição a diferentes maneiras de se desenvolver uma aplicação android, sendo assim foi recomendado pelo próprio docente que fosse uma aplicação simples).

A camada de view, responsável por tudo que o usuário visualiza, é construída em cima de um StatefulWidget, que é um widget que permite a modificação de estado, sendo assim podendo ser possível exibir o resultado na tela. Ela conta com uma ListView, dois TextField, um para input, sendo esse configurado apenas para receber número inteiros e positivos, e outro configurado como readOnly: true. e um

ElevatedButton para disparar o cálculo. Há alguns outros Widgets Container, apenas usados para estilização.

Sendo o resultado exposto na Figura 1 e Figura 2.

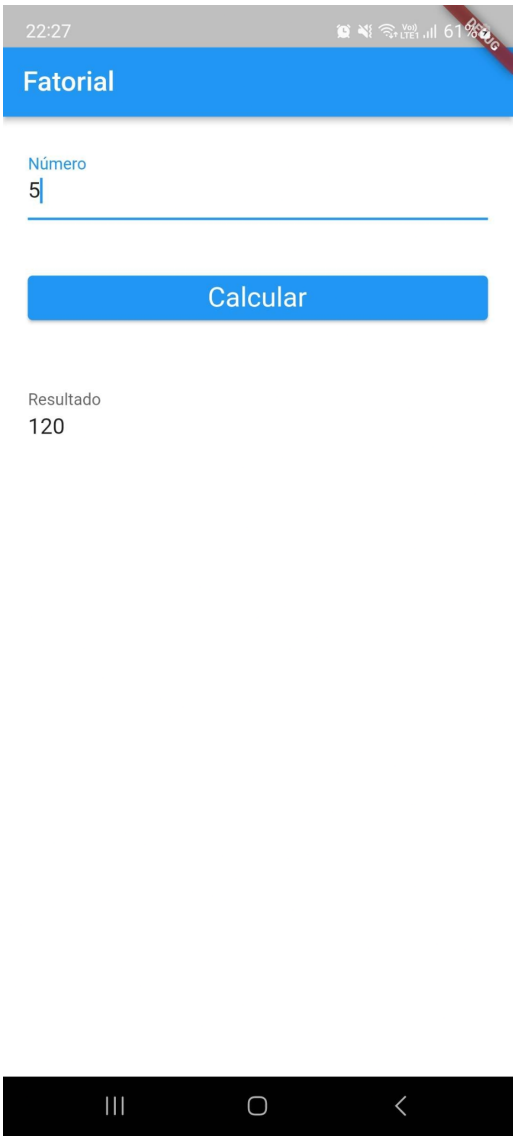

Figura 1. Tela do aplicativo.

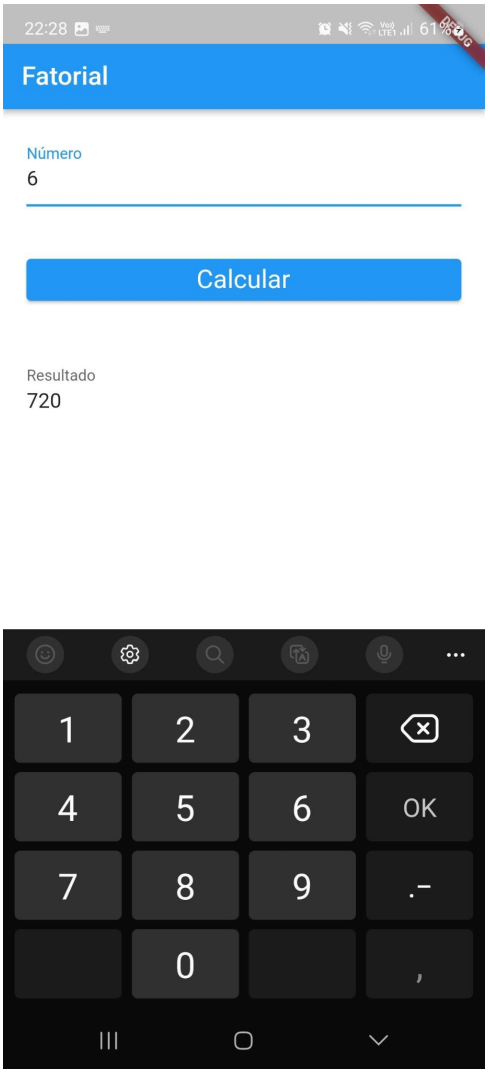

Figura 2. Tela do aplicativo com teclado ativado.

Na camada Controller, temos o cálculo do fatorial, ela recebe dois TextEditingController, um contém a string de input do usuário e a outra contém a string de output. Fatorial é uma função que tem um crescimento extremamente acelerado e rapidamente iria estourar o limite de representação do int. Por esse motivo o método calcFactorial, da class controller pega o texto de input e o converte para um int, sendo assim, a entrada ainda é limitada pelo tamanho máximo de um int, no entanto o resultado dos cálculos são armazenados em uma variável de tipo BigInt, podendo realizar o cálculos de fatoriais gigantescos como 9999.

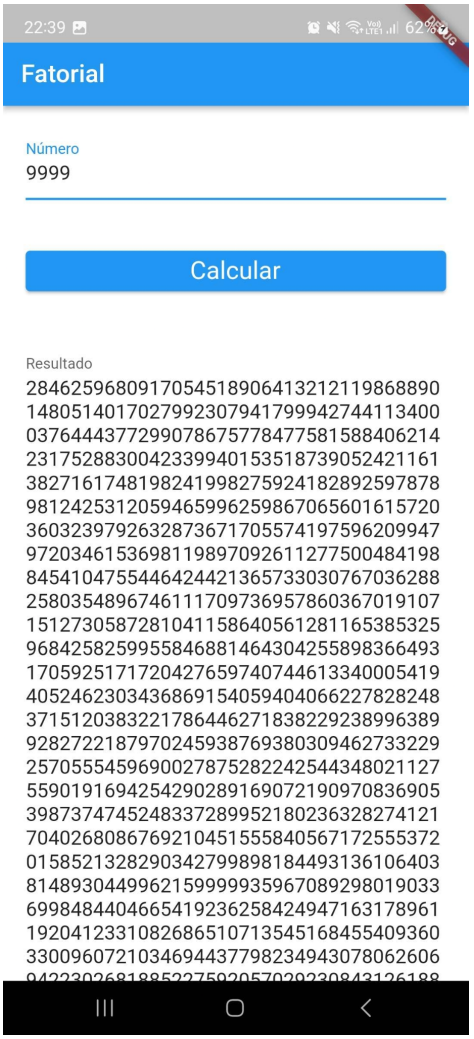

Figura 3. BigInt sendo posto em prática.

Como a entrada não permite número negativos ou com floats, e a saída é um BigInt, não há nenhuma necessidade de tratamento de erros.

2. Android Studio com Java

O aplicativo de cálculo de fatorial feito no Android Studio com Java foi desenvolvido mediante as configurações padrão apresentadas pelo Android Studio logo após sua instalação e o projeto foi inicializado a partir do template de atividade vazia.

A configuração da interface é dada por um arquivo XML gerado automaticamente e os elementos necessários para atingir o comportamento desejado do app foram adicionados por meio do editor de design integrado ao Android Studio. Com ele foi adicionada uma caixa de texto contendo a instrução para a realização do cálculo, um campo de inserção de texto, utilizado como entrada de dados ao programa, que permite a entrada apenas de dígitos numéricos, um

botão usado para chamar a função de cálculo e outra caixa de texto dentro de uma visualização de scroll, a fim de permitir a visualização de longos resultados.

Toda a lógica e cálculos necessários foram implementados diretamente no arquivo de MainActivity, contendo na função de criação um Click Listener, posto sobre o botão de cálculo, que recebe o valor encontrado no campo de inserção, o converte para Int e o passa para uma função que calcula o fatorial, essa realiza os cálculos em uma variável BigInt e devolve o resultado para a caixa de texto do resultado.

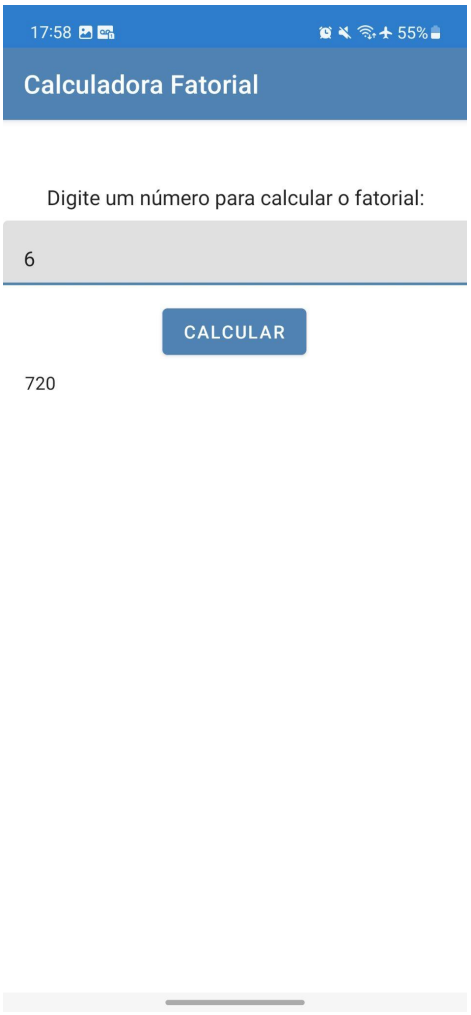

Figura 4. Tela do aplicativo com resultado de cálculo.

3. Linha de Comando

Para a realização do passo a passo, o principal material de referência foi o fornecido pelo docente “Building an Android App from the Command Line”. O desenvolvimento por linha de comando se inicia com o download das ferramentas necessárias, onde foram utilizadas as seguintes versões: Java Development Kit 20, Build-Tools 30.0.3, Android Platform 31 e Platform-Tools.

Em seguida é realizada a criação manual dos diretórios requeridos, Apptds como o diretório raiz e dentro dele, build, java e res, juntamente ao androidmanifest.

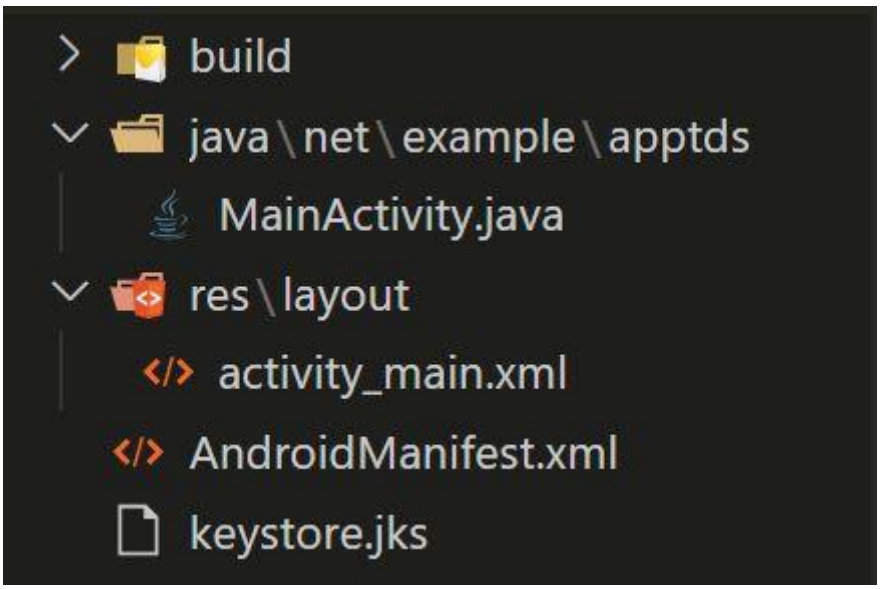

Figura 5. Organização dos arquivos no projeto.

Com os diretórios criados, o próximo passo foi converter o Layout de ConstraintLayout para LinearLayout, informar ao AndroidManifest as versões utilizadas e adequando o código da MainActivity após a remoção de alguns imports incompatíveis.

Com o projeto pronto, é iniciada a execução dos comandos de compilação. Abaixo, temos a configuração de variáveis futuramente necessárias, seguindo a hierarquia de pastas.

- Criação de paths para simplificação na execução dos próximos comandos:
  - $ SDK="${HOME}/android-sdk-linux" ;
  - $ BUILD_TOOLS="${SDK}/build-tools/25.0.0" ;
  - $ PLATFORM="${SDK}/platforms/android-16" ;
- Criação dos diretórios de build:
  - $ mkdir -p build/gen build/obj build/apk;
- Criação da classe R, usada para referenciar os diferentes recursos utilizados no projeto:
  - "${BUILD_TOOLS}/aapt" package -f -m -J build/gen/ -S res -M AndroidManifest.xml -I "${PLATFORM}/android.jar"
- Compilação do projeto em Java:
  - javac -source 1.8 -target 1.8 -bootclasspath "${JAVA_HOME}/jre/lib/rt.jar" -classpath "${PLATFORM}/android.jar" -d build/obj build/gen/net/example/apptds/R.java
- Tradução do código em Java para Dalvik, gerando um .dex usado para criar a APK:
  - "${BUILD_TOOLS}/dx" --dex --output=build/apk/classes.dex build/obj/
- Criação da APK:
  - "${BUILD_TOOLS}/aapt" package -f -M AndroidManifest.xml -S res/ -I "${PLATFORM}/android.jar" -F build/Apptds.unsigned.apk build/apk/ (gerar arquivo apk);

- Execução do Zipalign para facilitar o endereçamento de memória:
    - ${BUILD_TOOLS}/zipalign" -f -p 4 build/Apptds.unsigned.apk build/Apptds.aligned.apk:
- Criação de uma chave para assinar a APK:
    - keytool -genkeypair -keystore keystore.jks -alias androidkey \ -validity 10000 -keyalg RSA -keysize 2048 \ -storepass android -keypass android
- Assinando a APK com a chave criada anteriormente:
    - "${BUILD_TOOLS}/apksigner" sign --ks keystore.jks \ --ks-key-alias androidkey --ks-pass pass:android \ --key-pass pass:android --out build/Apptds.apk \ build/Apptds.aligned.apk
- Por último, instala-se o aplicativo a um dispositivo conectado por meio de:
    - $ "${SDK}/platform-tools/adb" install -r build/Apptds.apk
    - $ "${SDK}/platform-tools/adb" shell am start -n net.example.Apptds/.MainActivity

Com todo os comandos executados e funcionando corretamente, temos como resultado final as figuras abaixo.

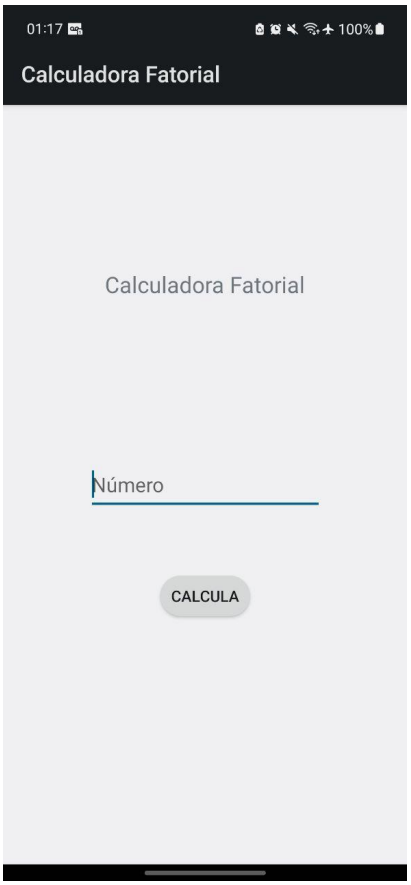

Figura 6. Tela inicial do aplicativo.

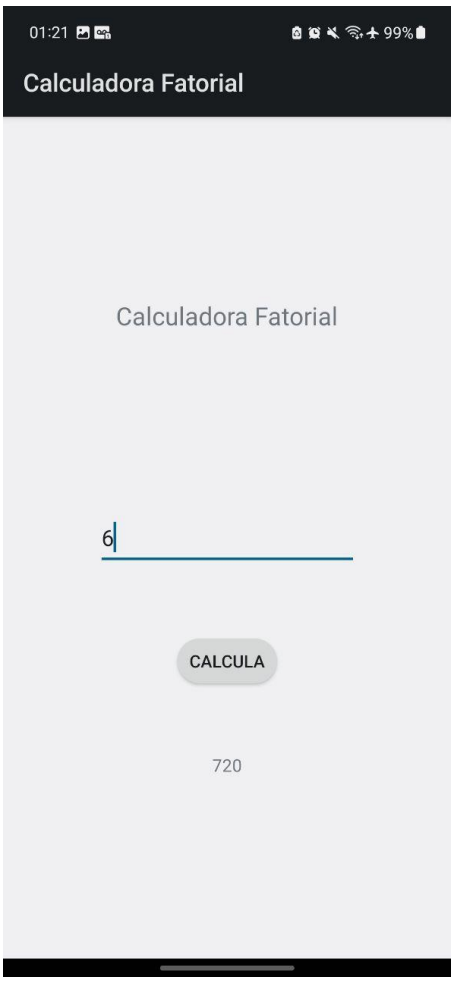

Figura 7. Tela do aplicativo com resultado de cálculo.