PET - Atividade 01

José Anderson da Silva Costa[1]

[1]Alameda José Quintino - Prado - S/N -
Instituto Federal de Ciência e Tecnológia - Campus Cedro (IFCE)
Caixa Postal 63400-000 – (88) 3564-1000 – Cedro – CE – Brazil

jose.anderson.silva09aluno.ifce.edu.br

1. Introdução

Nesta atividade será utilizado o código de demonstração fornecido a partir da criação de um projeto Flutter (Flutter Application) e realizar a leitura do código. A partir desse processo, definir:

- O conceito de widget;
- A diferença entre widget com estado (stateful) e widget sem estado (stateless);
- Indicar no código de demonstração, em que parte os conceitos anteriores podem ser encontrados.

2. Desenvolvimento

O Flutter é um framework desenvolvido pela Google para criação de aplicações móveis para android e iOS, utilizando uma base de código única proporcionando facilidade em desenvolver aplicações, estas aplicações também podem ser web e desktop, para a criação destas interfaces, o Flutter utiliza um conceito de Widgets,

O Flutter é um framework desenvolvido pela Google para criação de aplicativos para plataformas mobile(Android e iOS), web e desktop, utilizando a mesma base de código é possível desenvolver aplicações para estas plataformas, para criações das aplicações o Flutter utiliza o conceito de Widgets.

2.0.1. Widgets

Widgets são do que componentes que podem ser stateless(sem estado) ou stateful(com estado) e são usados para moldar nossa interface do usuário (UI), cada Widget pode possuir vários Widgets o que permite que cada um deles em sua maioria tenham um código legível.

Stateful Widget são “componentes” que possuem um estado, ou seja, durante a execução pode haver alterações naquele componente sem precisar redirecionar para outra tela.

Stateless Widget são componentes sem estado, ou seja, são componentes que não possuirão alteração após renderizado.

Nas imagens abaixo o código de demonstração que será utilizado como exemplo:

```dart
import 'package:flutter/material.dart';

Run | Debug | Profile
void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ), // ThemeData
      home: const MyHomePage(title: 'Flutter Demo Home Page'),
    ); // MaterialApp
  }
}

class MyHomePage extends StatefulWidget {
  const MyHomePage({super.key, required this.title});

  final String title;

  @override
  State<MyHomePage> createState() => _MyHomePageState();
}
```

Figure 1. demo

Na Figure 1 linha 7 temos um exemplo de Stateless Widget nesse primeiro bloco de código entre as linhas 7 e 20 temos propriedades que não sofrem alterações como o título da aplicação e cor.

Na linha 22 temos um exemplo de Stateful Widget como podemos observar ele realiza mudança de de estado através do ”createState()” onde há uma sobrescrita.

```dart
class _MyHomePageState extends State<MyHomePage> {
  int _counter = 0;

  void _incrementCounter() {
    setState(() {
      _counter++;
    });
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(widget.title),
      ), // AppBar
      body: Center(
       child: Column(
           mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            const Text(
              'You have pushed the button this many times:',
            ), // Text
            Text(
              '$_counter',
              style: Theme.of(context).textTheme.headline4,
            ), // Text
          ], // <Widget>[]
        ), // Column
      ), // Center
      floatingActionButton: FloatingActionButton(
        onPressed: _incrementCounter,
        tooltip: 'Increment',
        child: const Icon(Icons.add),
      ),  // FloatingActionButton
    ); // Scaffold
  }
}
```

Figure 2. demo

Na Figure 2 contiuando o código anterior temos o contator da aplicação demo sendo inicializado com o valor ”0” na linha 32, logo abaixo linha 34 temos a função que irá incrementar o contador, entre as linhas 40 e 67 temos a função que irá realizar as alterações na tela exibindo quantas vezes o botão de ”+” foi clicado e a cada interação realizar a alteração de estado na tela do usuário.