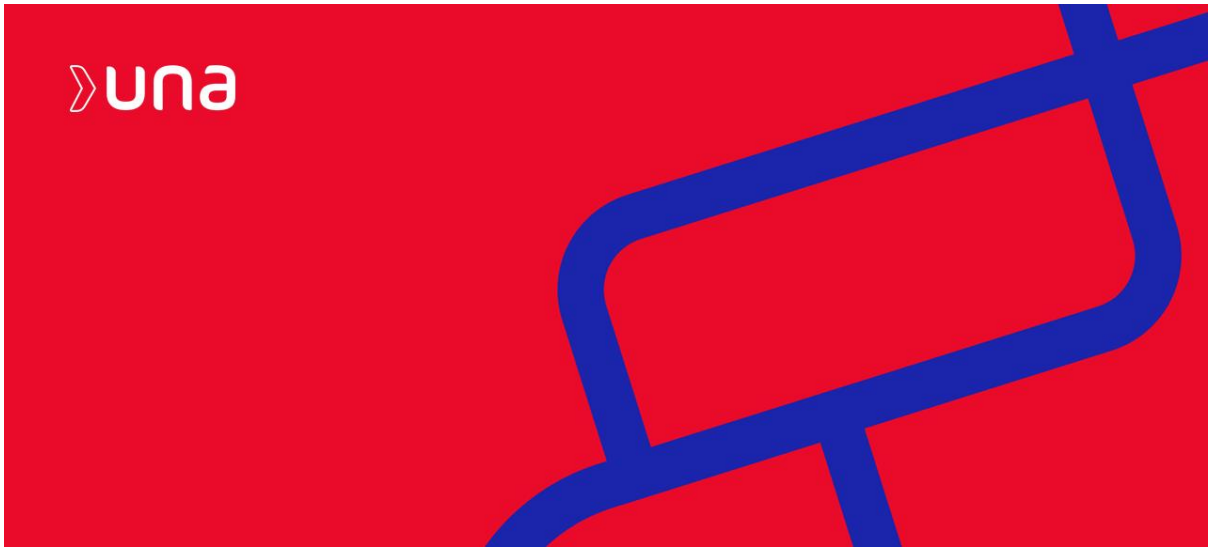


Flutter

Wesley Dias Maciel

2021/02

Centro Universitário UNA
Flutter
Wesley Dias Maciel
2021/02

Prática 10

Entrada & Saída

Documentação: https://api.flutter.dev/flutter/material/TextField-class.html, https://api.flutter.dev/flutter/widgets/TextEditingController-class.html, https://api.flutter.dev/flutter/material/ElevatedButton-class.html

Objetivo: apresentar widgets de entrada e saída.

1) Crie um novo projeto Flutter, usando:
    a. Visual Studio Code, ou;
    b. https://dartpad.dev/, ou;
    c. https://flutlab.io/ide, ou;
    d. https://flutterstudio.app/, ou;
    e. https://codemagic.io/.

Classe TextField

A classe TextFiled gera um campo de texto do material design. Um campo de texto permite que o usuário insira texto, seja com teclado de hardware ou com teclado na tela.

O campo de texto chama o método de callback onChanged sempre que o usuário altera o texto no campo. Por outro lado, se o usuário indicar que terminou de digitar o texto no campo (por exemplo, pressionando um botão no teclado virtual), o campo de texto chama o método de callback onSubmitted.

Para controlar o texto que é exibido no campo de texto, use o atributo controller. O atributo controller pode receber um objeto da classe TextEditingController; por exemplo, para definir o valor inicial do campo de texto ou ler o texto informado pelo usuário. É uma boa prática descartar um TextEditingController quando ele não for mais necessário. Isso garantirá o descarte de quaisquer recursos usados pelo objeto.

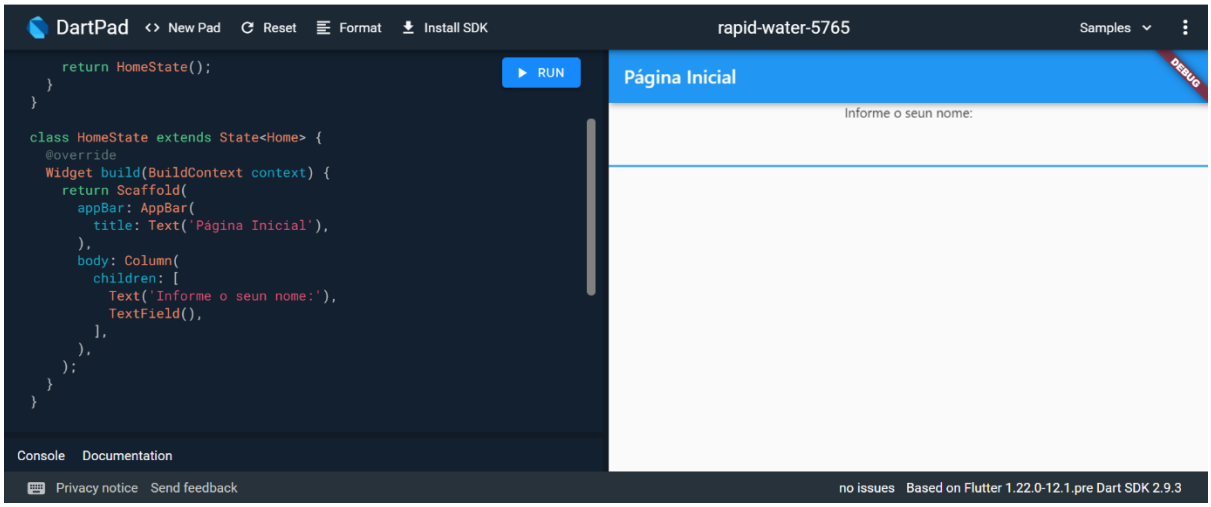
O campo de texto possui um atributo decoration que serve para configurar a aparência do campo. Por padrão, o atributo decoration desenha uma barra divisória abaixo do campo de

texto. Entretanto, você pode usar a propriedade decoration para alterar essa aparência padrão; por exemplo, adicionando um rótulo ou um ícone. Se você definir o atributo decoration como null, toda a configuração visual (decoração, layout) do campo de texto será removida. Se o atributo decoration não for igual a null (que é o padrão), o campo de texto exigirá que um de seus ancestrais seja um widget Material.

Para incluir um campo de texto em um formulário (Form) com outros widgets de tipo FormField, use um TextFormField.

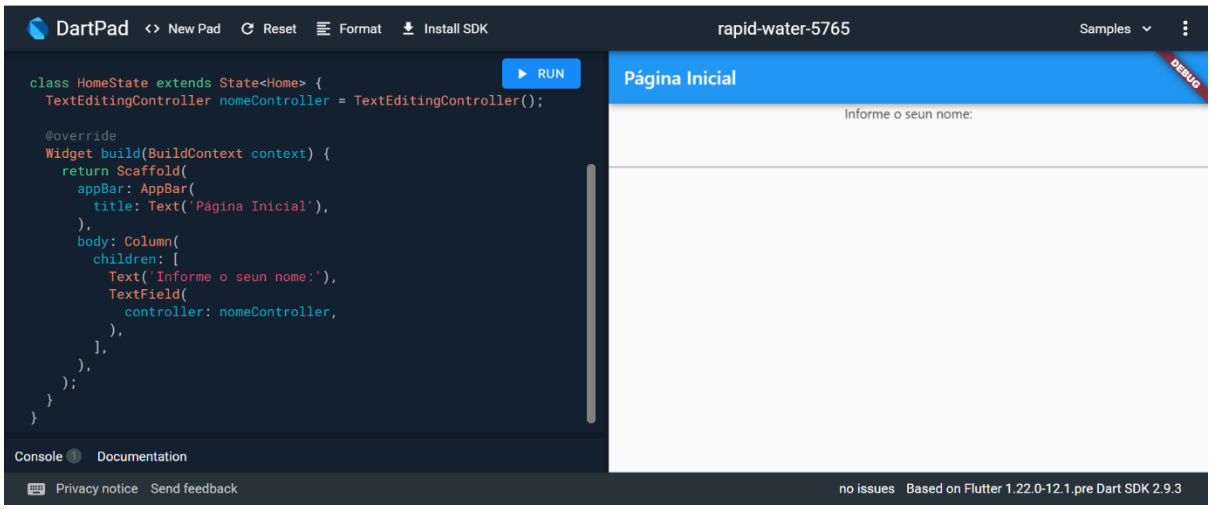
2) O exemplo abaixo exemplifica a utilização da classe TextField.

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

void main() => runApp(MaterialApp(home: Home()));

class Home extends StatefulWidget {
  HomeState createState() {
    return HomeState();
  }
}

class HomeState extends State<Home> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Página Inicial'),
      ),
      body: Column(
        children: [
          Text('Informe o seun nome:'),
          TextField(),
        ],
      ),
    );
  }
}
```



3) O próximo exemplo associa um controlador TextEditingController ao atributo controller do campo de texto criado.

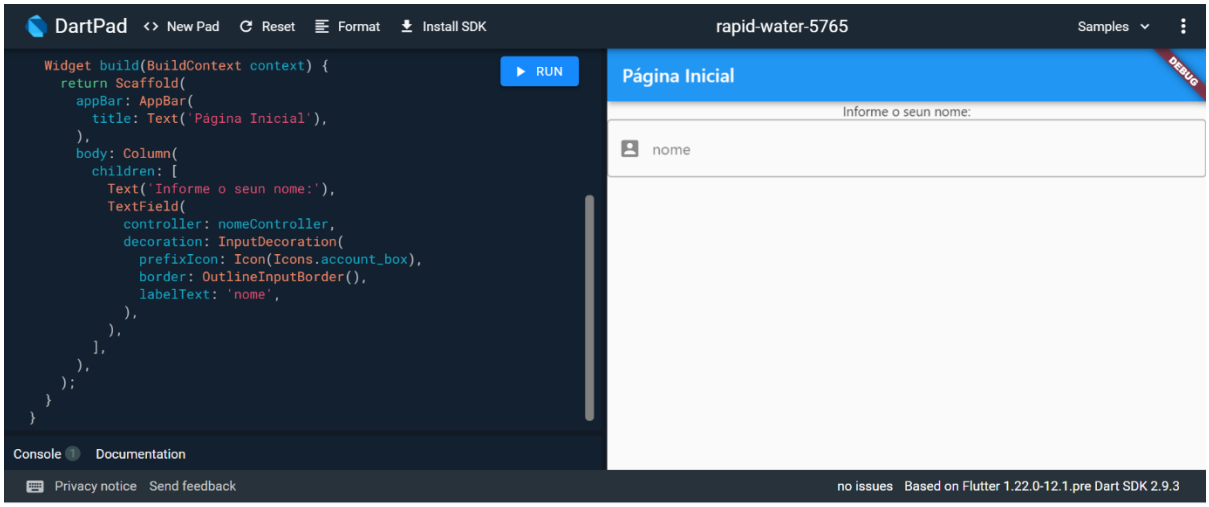
```dart
import 'package:flutter/material.dart';

void main() => runApp(MaterialApp(home: Home()));

class Home extends StatefulWidget {
  HomeState createState() {
    return HomeState();
  }
}

class HomeState extends State<Home> {
  TextEditingController nomeController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Página Inicial'),
      ),
      body: Column(
        children: [
          Text('Informe o seun nome:'),
          TextField(
            controller: nomeController,
          ),
        ],
      ),
    );
  }
}
```

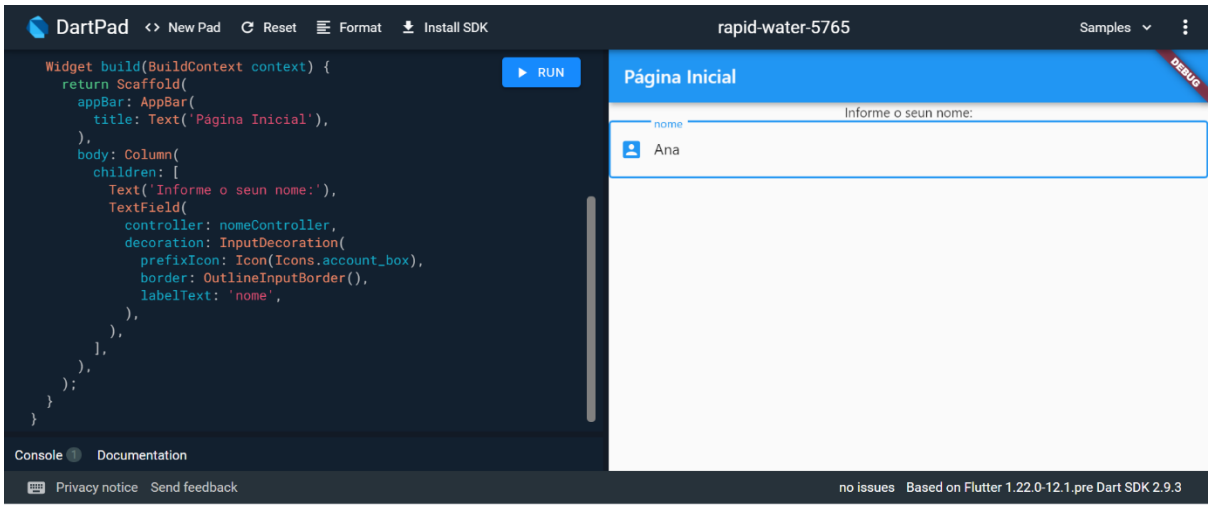


4) O exemplo seguinte associa uma configuração visual ao atributo decoration do campo de texto criado. Um ícone é apresentado dentro do campo de texto à esquerda.

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

void main() => runApp(MaterialApp(home: Home()));

class Home extends StatefulWidget {
  HomeState createState() {
    return HomeState();
  }
}

class HomeState extends State<Home> {
  TextEditingController nomeController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Página Inicial'),
      ),
      body: Column(
        children: [
          Text('Informe o seun nome:'),
          TextField(
            controller: nomeController,
            decoration: InputDecoration(
              prefixIcon: Icon(Icons.account_box),
              border: OutlineInputBorder(),
              labelText: 'nome',
```

```
            ),
          ),
        ],
      ),
    );
  }
}
```

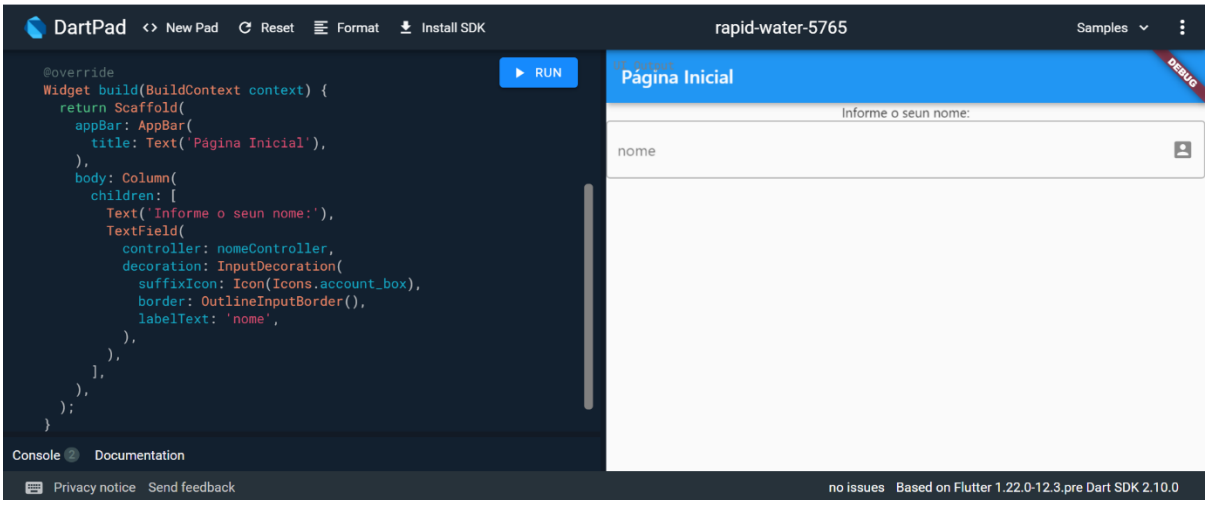


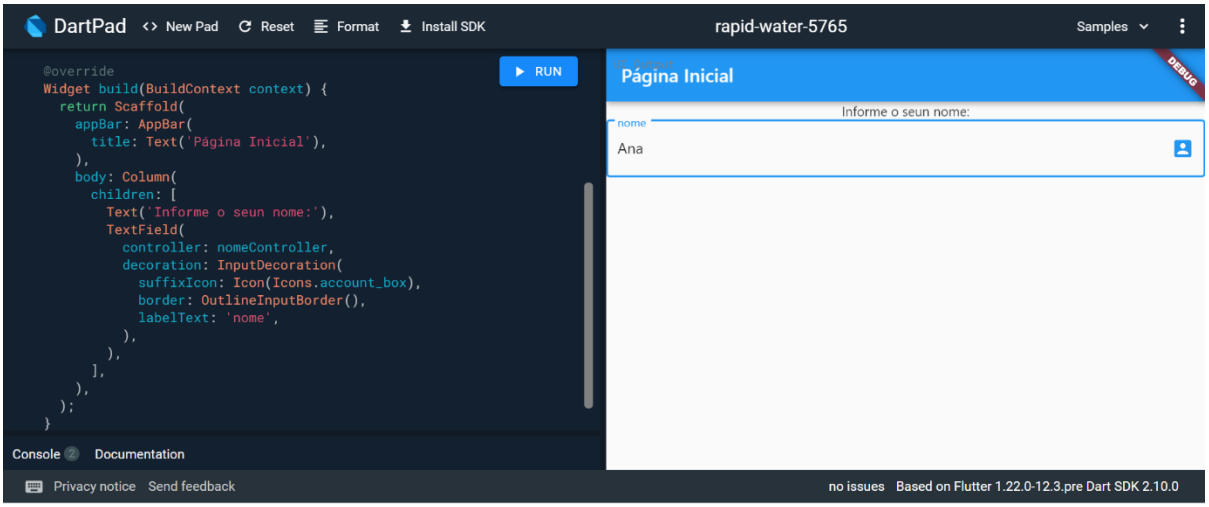


5) No exemplo abaixo o ícone é apresentado dentro do campo de texto à direita.

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

void main() => runApp(MaterialApp(home: Home()));

class Home extends StatefulWidget {
  HomeState createState() {
```

```
    return HomeState();
  }
}

class HomeState extends State<Home> {
  TextEditingController nomeController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Página Inicial'),
      ),
      body: Column(
        children: [
          Text('Informe o seun nome:'),
          TextField(
            controller: nomeController,
            decoration: InputDecoration(
              suffixIcon: Icon(Icons.account_box),
              border: OutlineInputBorder(),
              labelText: 'nome',
            ),
          ),
        ],
      ),
    );
  }
}
```

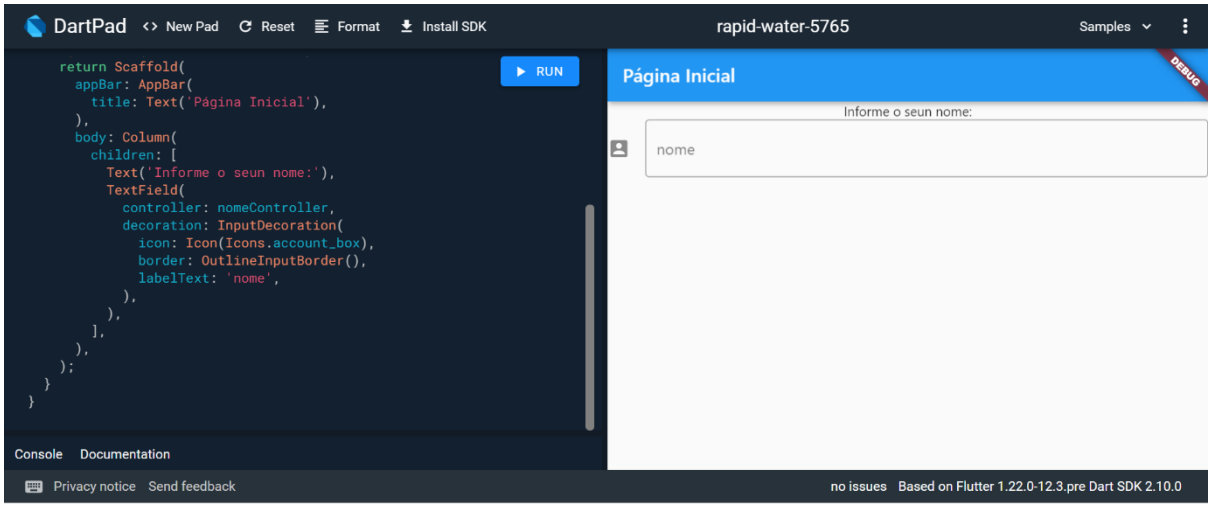


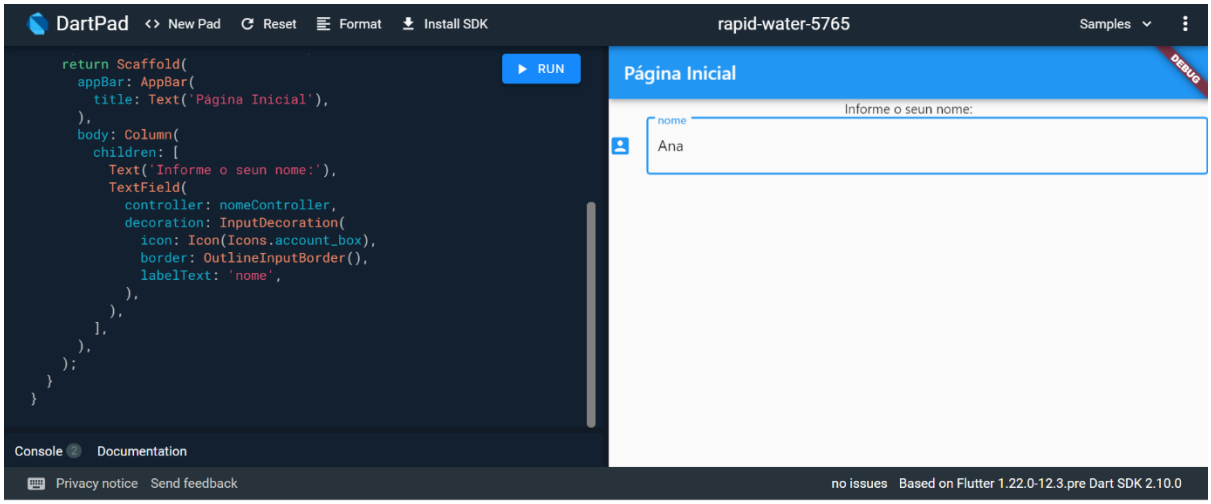


6) No exemplo seguinte o ícone é apresentado fora do campo de texto.

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

void main() => runApp(MaterialApp(home: Home()));

class Home extends StatefulWidget {
  HomeState createState() {
    return HomeState();
  }
}

class HomeState extends State<Home> {
  TextEditingController nomeController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Página Inicial'),
      ),
      body: Column(
        children: [
          Text('Informe o seun nome:'),
          TextField(
            controller: nomeController,
            decoration: InputDecoration(
              icon: Icon(Icons.account_box),
              border: OutlineInputBorder(),
              labelText: 'nome',
            ),
          ),
        ],
```

```
      ),
    );
  }
}
```



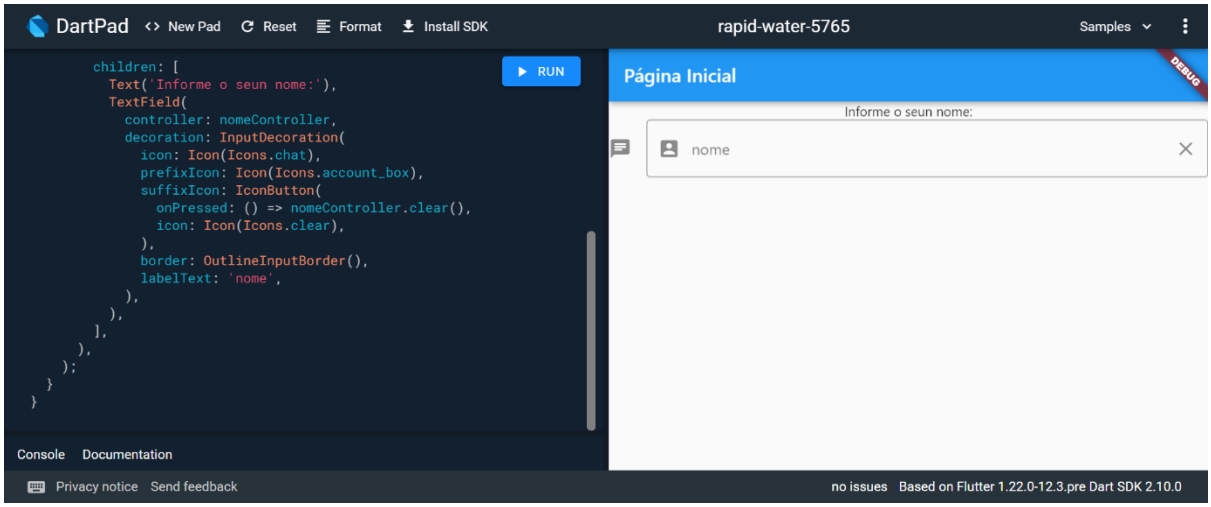


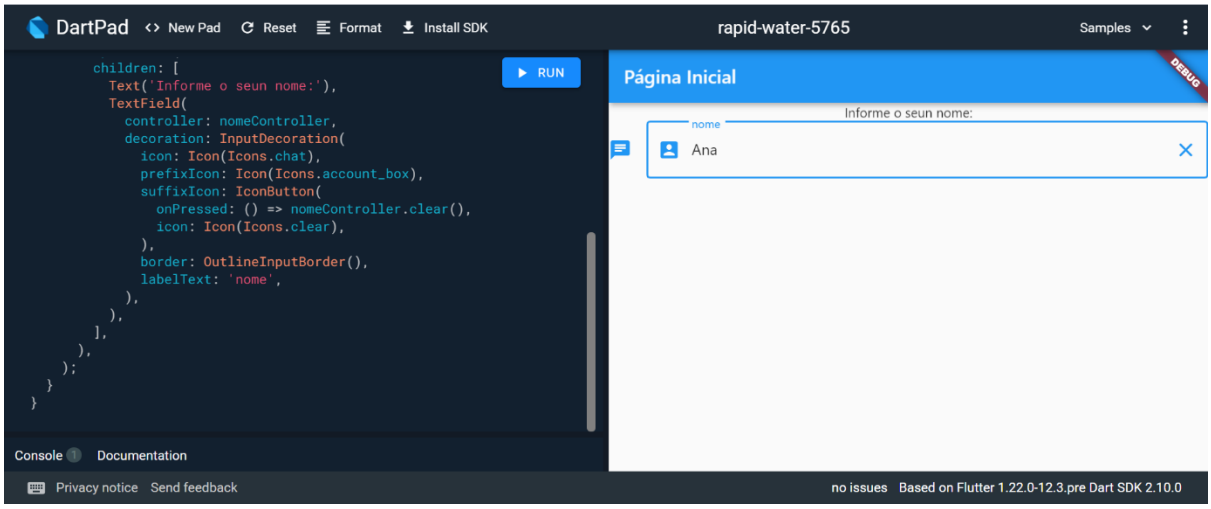
7) O exemplo a seguir apresenta os três ícones simultaneamente. Além disso, o ícone da direita recebe uma função que apaga o texto informado no campo de texto quando esse ícone é clicado.

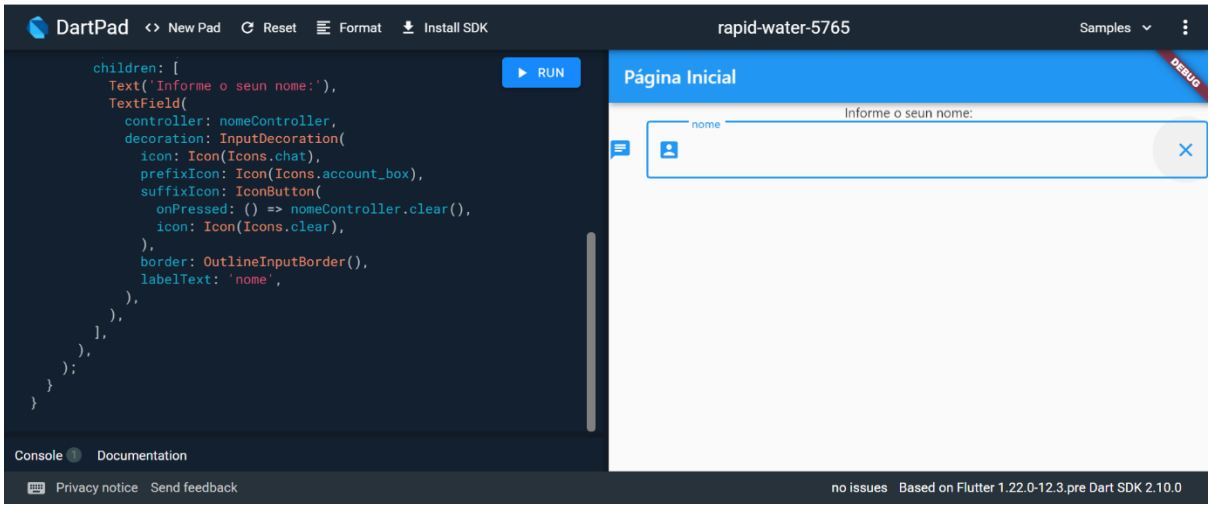
```
import 'package:flutter/material.dart';

void main() => runApp(MaterialApp(home: Home()));

class Home extends StatefulWidget {
  HomeState createState() {
    return HomeState();
```

```
  }
}

class HomeState extends State<Home> {
  TextEditingController nomeController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Página Inicial'),
      ),
      body: Column(
        children: [
          Text('Informe o seun nome:'),
          TextField(
            controller: nomeController,
            decoration: InputDecoration(
              icon: Icon(Icons.chat),
              prefixIcon: Icon(Icons.account_box),
              suffixIcon: IconButton(
                onPressed: () => nomeController.clear(),
                icon: Icon(Icons.clear),
              ),
              border: OutlineInputBorder(),
              labelText: 'nome',
            ),
          ),
        ],
      ),
    );
  }
}
```

Centro Universitário UNA
Flutter
Wesley Dias Maciel
2021/02







Classe ElevatedButton

A classe ElevatedButton gera um "botão elevado"", ou "botão em relevo", do Material Design. Você deve usar botões elevados para adicionar dimensão a layouts que são geralmente planos, como listas longas ou em áreas com espaços amplos. Evite usar botões elevados em widgets que também são elevados, como caixas de diálogo ou cartões.

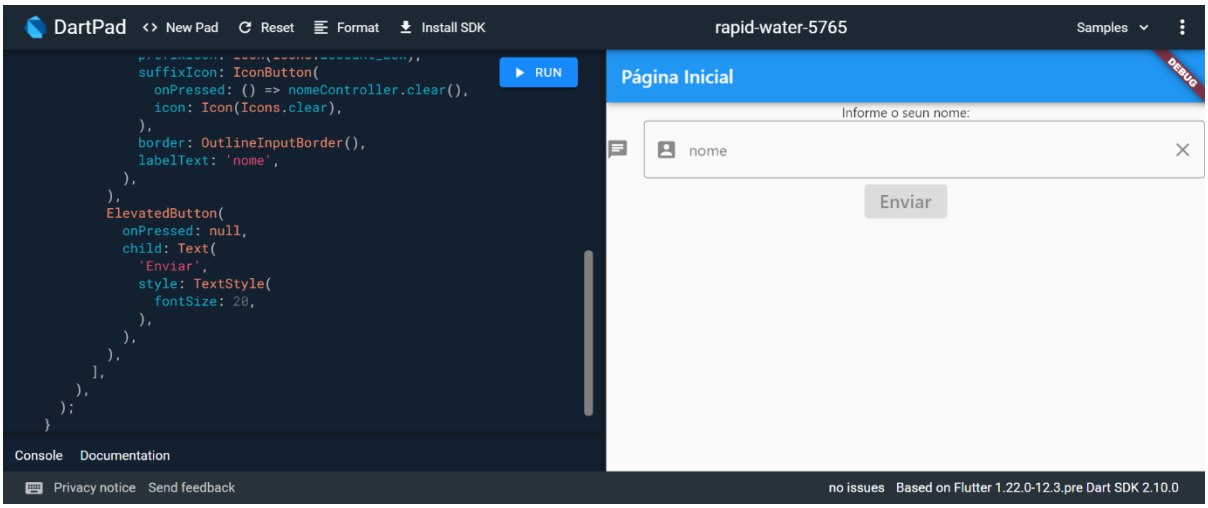
Um botão elevado é baseado em um widget cujo atributo Material.elevation aumenta quando o botão é pressionado. Um botão elevado possui os widgets filhos Text e Icon.

Botões elevados também possuem os atributos onPressed e onLongPress. Esses atributos devem receber as funções que serão executadas quando esses eventos ocorrerem. Por outro lado, se esses atributos receberem o valor null, o botão será desabilitado.

8) O exemplo abaixo apresenta um botão desabilitado. O atributo onPressed recebe o valor nul.

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

void main() => runApp(MaterialApp(home: Home()));

class Home extends StatefulWidget {
  HomeState createState() {
    return HomeState();
  }
}

class HomeState extends State<Home> {
  TextEditingController nomeController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Página Inicial'),
      ),
      body: Column(
        children: [
          Text('Informe o seun nome:'),
          TextField(
            controller: nomeController,
            decoration: InputDecoration(
              icon: Icon(Icons.chat),
              prefixIcon: Icon(Icons.account_box),
```

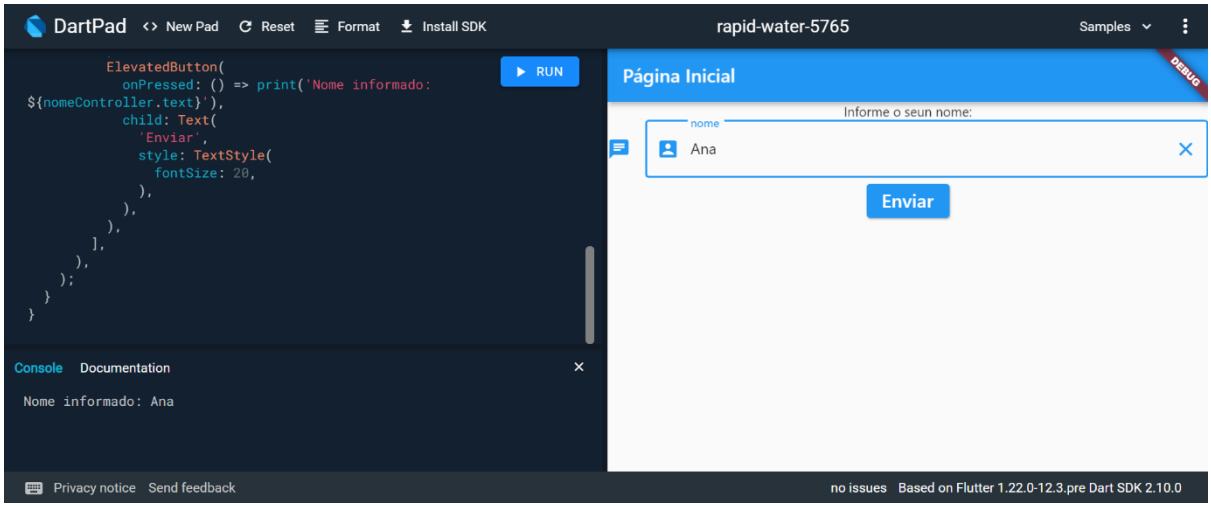
```
            suffixIcon: IconButton(
              onPressed: () => nomeController.clear(),
              icon: Icon(Icons.clear),
            ),
            border: OutlineInputBorder(),
            labelText: 'nome',
          ),
        ),
        ElevatedButton(
          onPressed: null,
          child: Text(
            'Enviar',
            style: TextStyle(
              fontSize: 20,
            ),
          ),
        ),
      ],
    ),
  );
 }
}
```



9) O exemplo abaixo apresenta um botão habilitado. O atributo onPressed recebe uma função que imprime no console o nome informado.

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

void main() => runApp(MaterialApp(home: Home()));
```

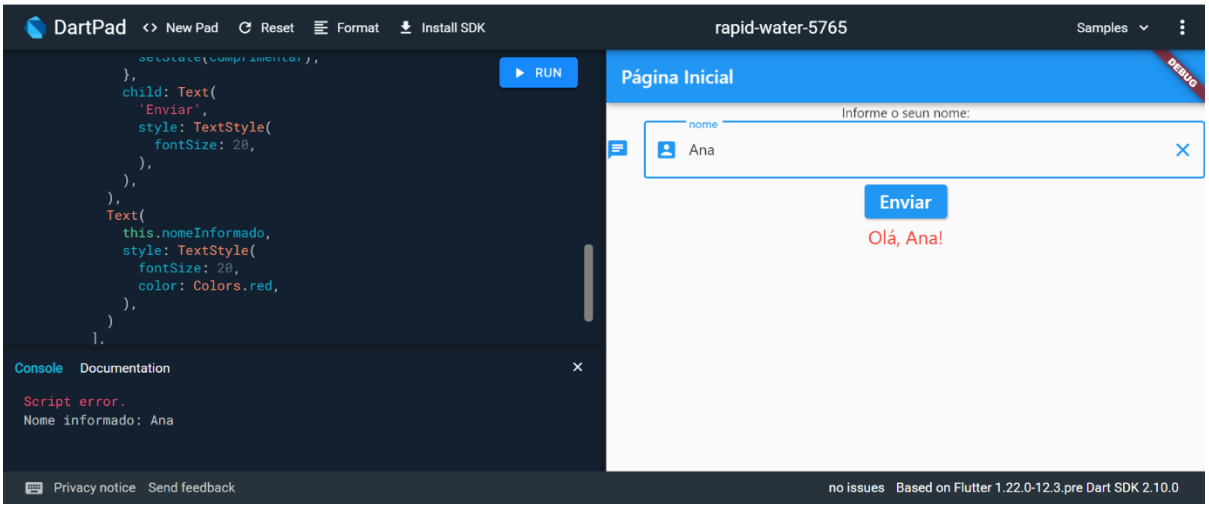
```dart
class Home extends StatefulWidget {
  HomeState createState() {
    return HomeState();
  }
}

class HomeState extends State<Home> {
  TextEditingController nomeController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Página Inicial'),
      ),
      body: Column(
        children: [
          Text('Informe o seun nome:'),
          TextField(
            controller: nomeController,
            decoration: InputDecoration(
              icon: Icon(Icons.chat),
              prefixIcon: Icon(Icons.account_box),
              suffixIcon: IconButton(
                onPressed: () => nomeController.clear(),
                icon: Icon(Icons.clear),
              ),
              border: OutlineInputBorder(),
              labelText: 'nome',
            ),
          ),
          ElevatedButton(
            onPressed: () => print('Nome informado: ${nomeController.text}'),
            child: Text(
              'Enviar',
              style: TextStyle(
                fontSize: 20,
              ),
            ),
          ),
        ],
      ),
    );
  }
}
```



10) O exemplo abaixo apresenta na tela o nome informado. Como há uma alteração de estado do widget, precisamos usar o método **setState ()** quando o botão é pressionado. No exemplo, o método setState () recebe o método cumprimentar () como parâmetro. O método cumprimentar () gera um string para apresentação na tela.

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

void main() => runApp(MaterialApp(home: Home()));

class Home extends StatefulWidget {
  HomeState createState() {
    return HomeState();
  }
}

class HomeState extends State<Home> {
  TextEditingController nomeController = TextEditingController();
  String nomeInformado = "";

  cumprimentar() {
    this.nomeInformado = 'Olá, ${this.nomeController.text}!';
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Página Inicial'),
      ),
      body: Column(
        children: [
```

```
          Text('Informe o seun nome:'),
          TextField(
            controller: nomeController,
            decoration: InputDecoration(
              icon: Icon(Icons.chat),
              prefixIcon: Icon(Icons.account_box),
              suffixIcon: IconButton(
                onPressed: () => nomeController.clear(),
                icon: Icon(Icons.clear),
              ),
              border: OutlineInputBorder(),
              labelText: 'nome',
            ),
          ),
          ElevatedButton(
            onPressed: () {
              print('Nome informado: ${nomeController.text}');
              setState(cumprimentar);
            },
            child: Text(
              'Enviar',
              style: TextStyle(
                fontSize: 20,
              ),
            ),
          ),
          Text(
            this.nomeInformado,
            style: TextStyle(
              fontSize: 20,
              color: Colors.red,
            ),
          )
        ],
      ),
    );
  }
}
```

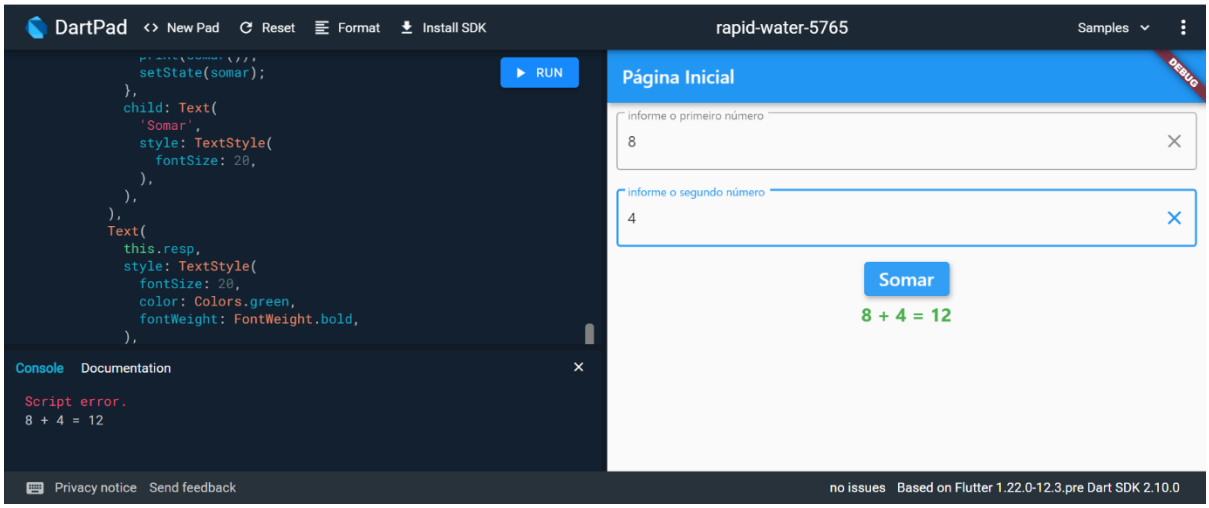


11) No próximo exemplo, o usuário informa dois números inteiros. O aplicativo apresenta a soma dos números na tela. Os números são lidos como strings. Portanto, é preciso convertê-los para inteiro antes de realizar a soma.

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

void main() => runApp(MaterialApp(home: Home()));

class Home extends StatefulWidget {
  HomeState createState() {
    return HomeState();
  }
}

class HomeState extends State<Home> {
  TextEditingController num01Controller = TextEditingController();
  TextEditingController num02Controller = TextEditingController();
  String resp = "";

  somar() {
    int num01 = int.parse(this.num01Controller.text);
    int num02 = int.parse(this.num02Controller.text);
    int soma = num01 + num02;
    this.resp = '$num01 + $num02 = $soma';
    return this.resp;
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
```

```
      title: Text('Página Inicial'),
    ),
    body: Column(
      children: [
        Container(
          margin: EdgeInsets.all(10),
          child: TextField(
            controller: num01Controller,
            decoration: InputDecoration(
              suffixIcon: IconButton(
                onPressed: () => num01Controller.clear(),
                icon: Icon(Icons.clear),
              ),
              border: OutlineInputBorder(),
              labelText: 'informe o primeiro número',
            ),
          ),
        ),
        Container(
          margin: EdgeInsets.all(10),
          child: TextField(
            controller: num02Controller,
            decoration: InputDecoration(
              suffixIcon: IconButton(
                onPressed: () => num02Controller.clear(),
                icon: Icon(Icons.clear),
              ),
              border: OutlineInputBorder(),
              labelText: 'informe o segundo número',
            ),
          ),
        ),
        ElevatedButton(
          onPressed: () {
            print(somar());
            setState(somar);
          },
          child: Text(
            'Somar',
            style: TextStyle(
              fontSize: 20,
            ),
          ),
        ),
        Text(
          this.resp,
          style: TextStyle(
```

```
              fontSize: 20,
              color: Colors.green,
              fontWeight: FontWeight.bold,
            ),
          )
        ],
      ),
    );
  }
}
```

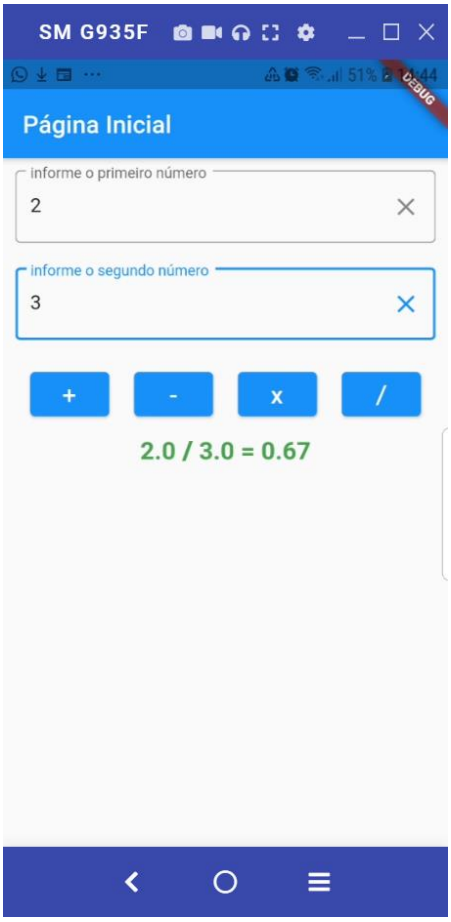


Exercício

1) Usando o framework Flutter, escreva o algoritmo de uma calculadora que permite realizar as 4 operações básicas: somar, subtrair, multiplicar e dividir. O usuário deve informar números reais.

   Dica:

```
  dividir() {
    double num01 = double.parse(this.num01Controller.text); //te
    double num02 = double.parse(this.num02Controller.text);
    double result = num01 / num02;
    this.resp = '$num01 / $num02 = ' + result.toStringAsFixed(2);
    return this.resp;
  }
```



2) Usando o framework Flutter, escreva o algoritmo que leia o peso e altura de uma pessoa e que calcule o índice de massa corporal (IMC) dessa pessoa. O algoritmo também deve apresentar na tela a classificação apresentada na tabela abaixo.

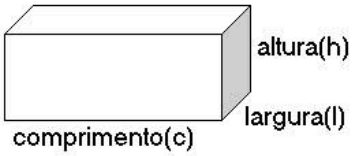
Dica:
IMC = peso / (altura × altura)

$$\text{IMC} = \frac{\text{Peso}}{\text{Altura} \times \text{Altura}}$$

IMC	Classificação
< 16	Magreza grave
16 a < 17	Magreza moderada
17 a < 18,5	Magreza leve
18,5 a < 25	Saudável
25 a < 30	Sobrepeso
30 a < 35	Obesidade Grau I
35 a < 40	Obesidade Grau II (severa)
≥ 40	Obesidade Grau III (mórbida)

3) Usando o framework Flutter, escreva o algoritmo que leia o comprimento, a largura e a altura de um paralelepípedo e que retorne o volume desse sólido.

   Dica:
   volume = comprimento × largura × altura



$V = c \times l \times h$