

# Flutter

Aula 06

Prof. Dr. Rodrigo Plotze

[rodrigoplotze@gmail.com](mailto:rodrigoplotze@gmail.com)



- Material Design
- Scaffold
- Text
- Column e Row
- Stateful Widgets

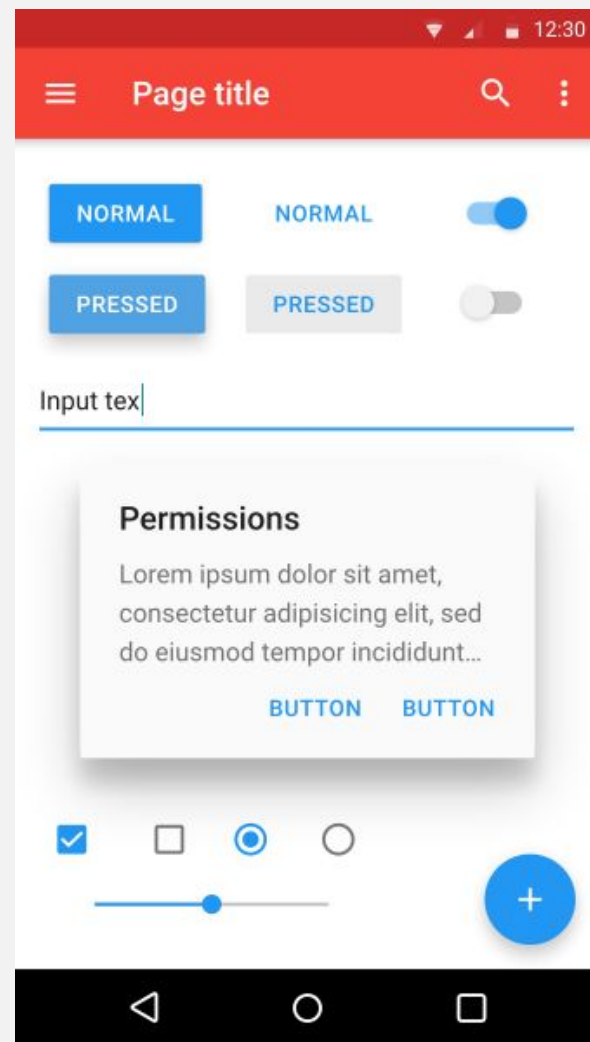
# MATERIAL DESIGN





- Linguagem de design desenvolvida pela Google em 2014.
- Referência para o design de UI para aplicativos.

<https://material.io/design/>





## ■ Material App

- Um widget que agrupa vários widgets normalmente necessários para aplicativos com material design.
- Um widget MaterialApp é definido apenas uma vez no aplicativo.
- Propriedades
  - title: título do aplicativo.
  - theme: tema empregado no App, que permite a configuração de diversos atributos.
  - home: primeira tela a ser mostrado no App.



```
MaterialApp( theme: ThemeData(  
  primaryColor: Colors.blue,  
  accentColor: Colors.green,  
  textTheme: TextTheme(body1:  
    TextStyle(  
      color: Colors.purple)  
    ),  
  ),
```

## ThemeData Demo

primaryColor

Button pressed 0 times

body1

+

accentColor

# SCAFFOLD



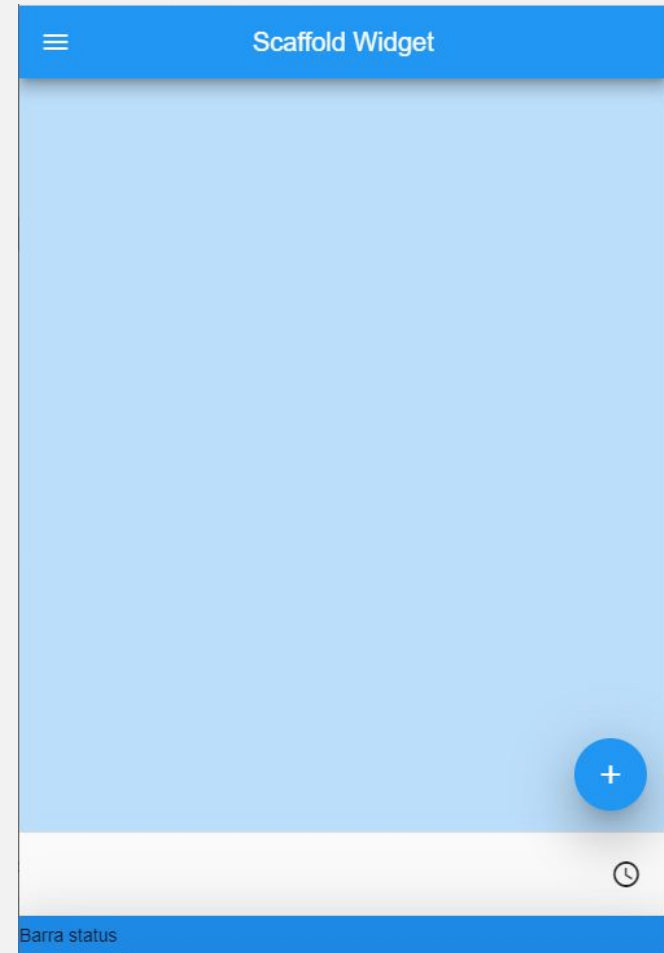


- Principal estrutura de layout do Material Design.
- Propriedades
  - appBar
  - body
  - drawer
  - floatingActionButton
  - bottomNavigationBar
  - persistentFooterButtons



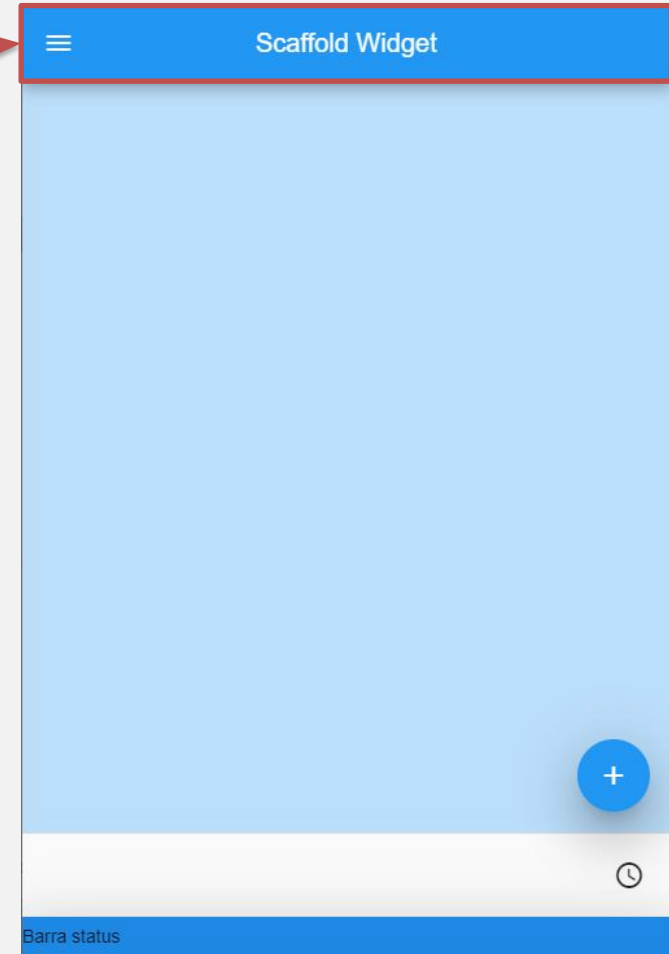
```
import 'package:flutter/material.dart';

class ScaffoldWidget extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Scaffold Widget"),
        centerTitle: true,
      ),
      body: Container(color: Colors.blue[100]),
      drawer: Container(color: Colors.red, ),
      floatingActionButton: FloatingActionButton(
        child: Icon(Icons.add),
        onPressed: () {print("Botão pressionado!");} ,
      ),
      bottomNavigationBar: BottomAppBar(
        child: Container(
          child: Text("Barra status"),
          height: 30.0,
          alignment: Alignment.centerLeft,
        ),
        color: Colors.blue[600],
      ),
      persistentFooterButtons: <Widget>[
        IconButton(icon: Icon(Icons.access_time),
          onPressed: () {print("Alarme!");}), ],
    );
  }
}
```



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

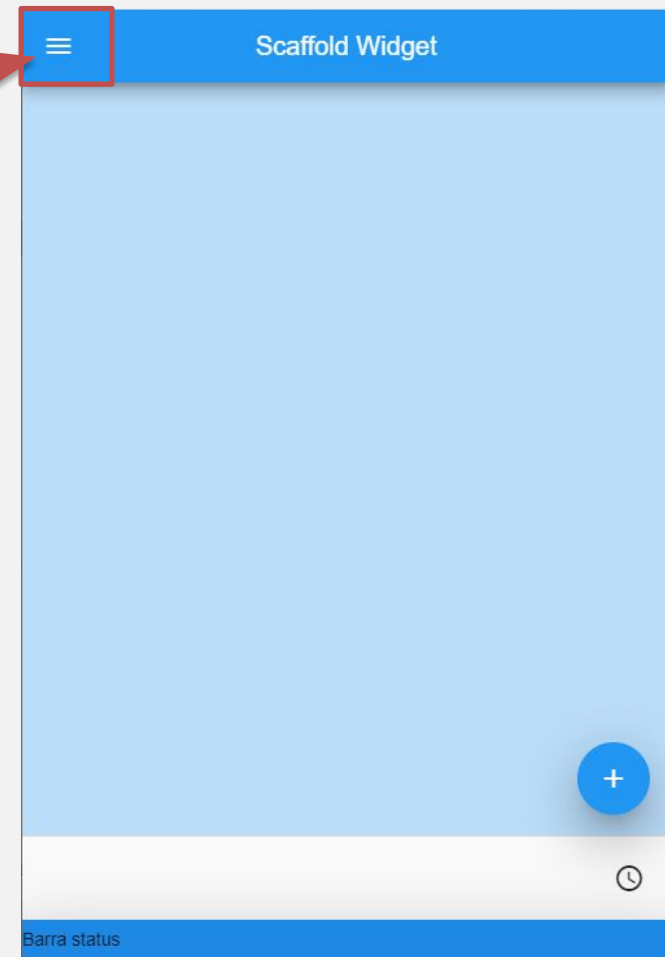
class ScaffoldWidget extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Scaffold Widget"),
        centerTitle: true,
      ),
      body: Container(color: Colors.blue[100]),
      drawer: Container(color: Colors.red, ),
      floatingActionButton: FloatingActionButton(
        child: Icon(Icons.add),
        onPressed: () {print("Botão pressionado!");} ,
      ),
      bottomNavigationBar: BottomAppBar(
        child: Container(
          child: Text("Barra status"),
          height: 30.0,
          alignment: Alignment.centerLeft,
        ),
        color: Colors.blue[600],
      ),
      persistentFooterButtons: <Widget>[
        IconButton(icon: Icon(Icons.access_time),
          onPressed: () {print("Alarme!");}), ],
    );
  }
}
```





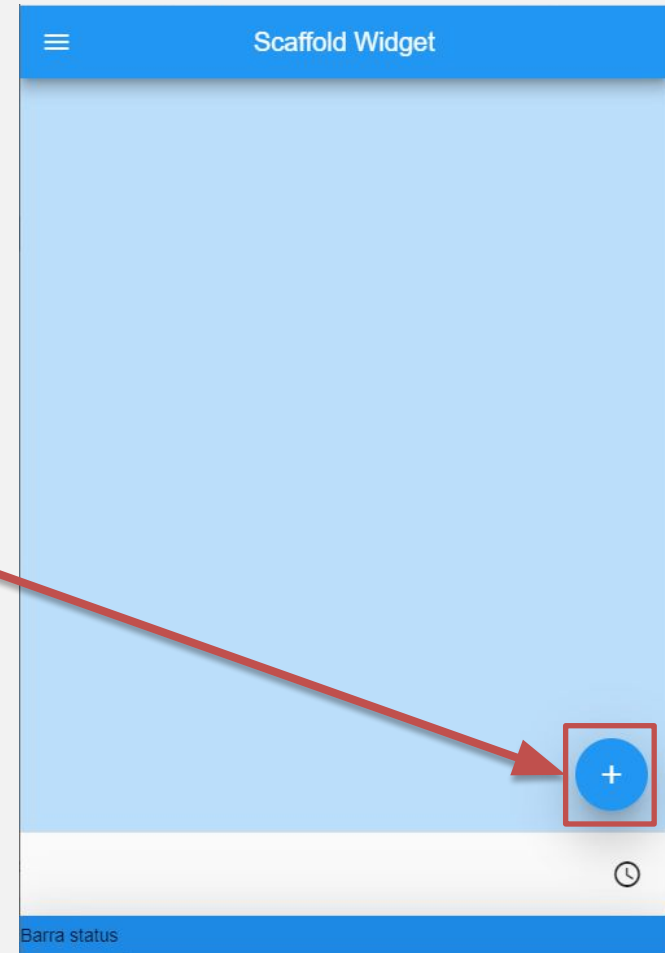
```
import 'package:flutter/material.dart';

class ScaffoldWidget extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Scaffold Widget"),
        centerTitle: true,
      ),
      body: Container(color: Colors.blue[100]),
      drawer: Container(color: Colors.red, ),
      floatingActionButton: FloatingActionButton(
        child: Icon(Icons.add),
        onPressed: () {print("Botão pressionado!");} ,
      ),
      bottomNavigationBar: BottomAppBar(
        child: Container(
          child: Text("Barra status"),
          height: 30.0,
          alignment: Alignment.centerLeft,
        ),
        color: Colors.blue[600],
      ),
      persistentFooterButtons: <Widget>[
        IconButton(icon: Icon(Icons.access_time),
          onPressed: () {print("Alarme!");}), ],
    );
  }
}
```



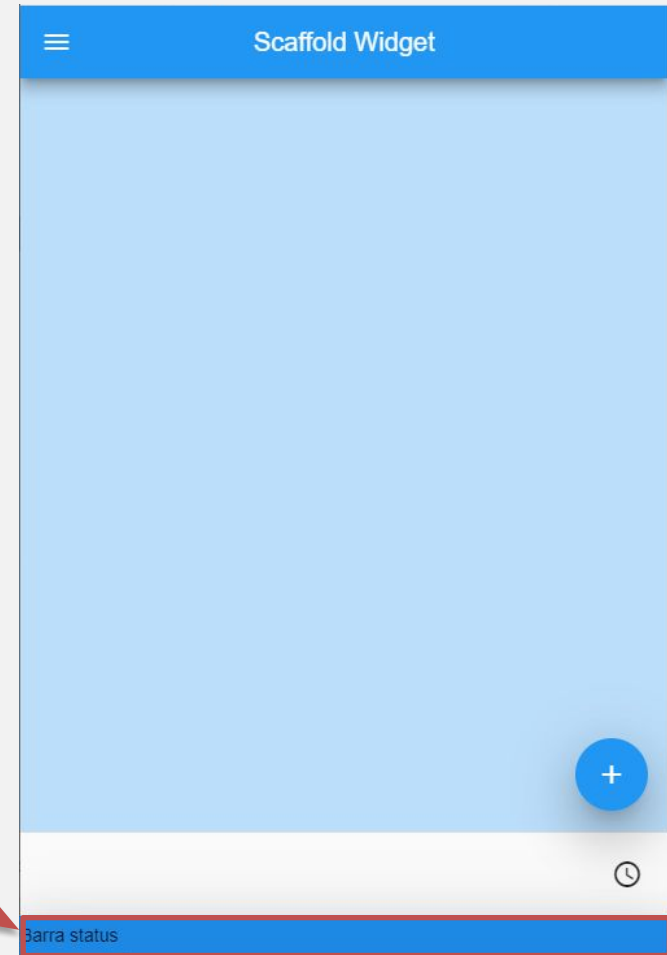
```
import 'package:flutter/material.dart';

class ScaffoldWidget extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Scaffold Widget"),
        centerTitle: true,
      ),
      body: Container(color: Colors.blue[100]),
      drawer: Container(color: Colors.red, ),
      floatingActionButton: FloatingActionButton(
        child: Icon(Icons.add),
        onPressed: () {print("Botao pressionado!");} ,
      ),
      bottomNavigationBar: BottomAppBar(
        child: Container(
          child: Text("Barra status"),
          height: 30.0,
          alignment: Alignment.centerLeft,
        ),
        color: Colors.blue[600],
      ),
      persistentFooterButtons: <Widget>[
        IconButton(icon: Icon(Icons.access_time),
          onPressed: () {print("Alarme!");}), ],
    );
  }
}
```



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

class ScaffoldWidget extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Scaffold Widget"),
        centerTitle: true,
      ),
      body: Container(color: Colors.blue[100]),
      drawer: Container(color: Colors.red, ),
      floatingActionButton: FloatingActionButton(
        child: Icon(Icons.add),
        onPressed: () {print("Botão pressionado!");} ,
      ),
      bottomNavigationBar: BottomAppBar(
        child: Container(
          child: Text("Barra status"),
          height: 30.0,
          alignment: Alignment.centerLeft,
        ),
        color: Colors.blue[600],
      ),
      persistentFooterButtons: <Widget>[
        IconButton(icon: Icon(Icons.access_time),
          onPressed: () {print("Alarme!");}), ],
    );
  }
}
```

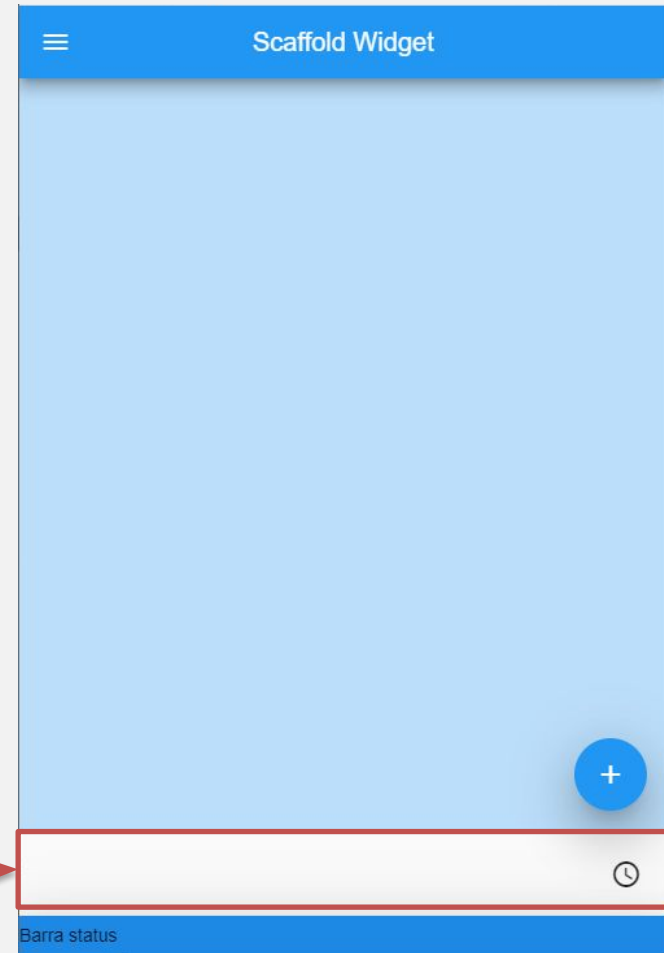


```

import 'package:flutter/material.dart';

class ScaffoldWidget extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Scaffold Widget"),
        centerTitle: true,
      ),
      body: Container(color: Colors.blue[100]),
      drawer: Container(color: Colors.red, ),
      floatingActionButton: FloatingActionButton(
        child: Icon(Icons.add),
        onPressed: () {print("Botão pressionado!");} ,
      ),
      bottomNavigationBar: BottomAppBar(
        child: Container(
          child: Text("Barra status"),
          height: 30.0,
          alignment: Alignment.centerLeft,
        ),
        color: Colors.blue[600],
      ),
      persistentFooterButtons: <Widget>[
        IconButton(icon: Icon(Icons.access_time),
          onPressed: () {print("Alarme");}), ],
    );
  }
}

```



**TEXT**





- Widget ***Text*** exibe uma sequência de texto com estilo único.
- A cadeia de caracteres pode quebrar várias linhas ou pode ser exibida na mesma linha, dependendo das restrições de layout.

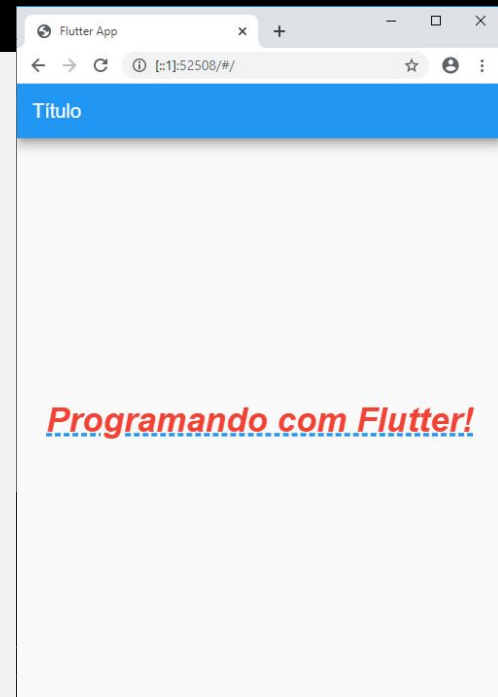
Hello beautiful world

```
Text.rich(  
  TextSpan(  
    text: 'Hello', // default text style  
    children: <TextSpan>[  
      TextSpan(text: ' beautiful ', style: TextStyle(fontStyle: FontStyle.italic)),  
      TextSpan(text: 'world', style: TextStyle(fontWeight: FontWeight.bold)),  
    ],  
  ),  
)
```

O construtor ***Text.rich*** pode exibir um parágrafo com TextSpans com estilos diferentes.



```
Text("Programando com Flutter!",  
    style: TextStyle(  
      color: Colors.red,  
      fontSize: 35.0,  
      fontStyle: FontStyle.italic,  
      fontWeight: FontWeight.bold,  
      decoration: TextDecoration.underline,  
      decorationColor: Colors.blue,  
      decorationStyle: TextDecorationStyle.dashed,  
    ),  
),
```



**COLUMN E ROW**





## ■ ***Column***

- Widget utilizado para exibir widgets filhos em um arranjo vertical.





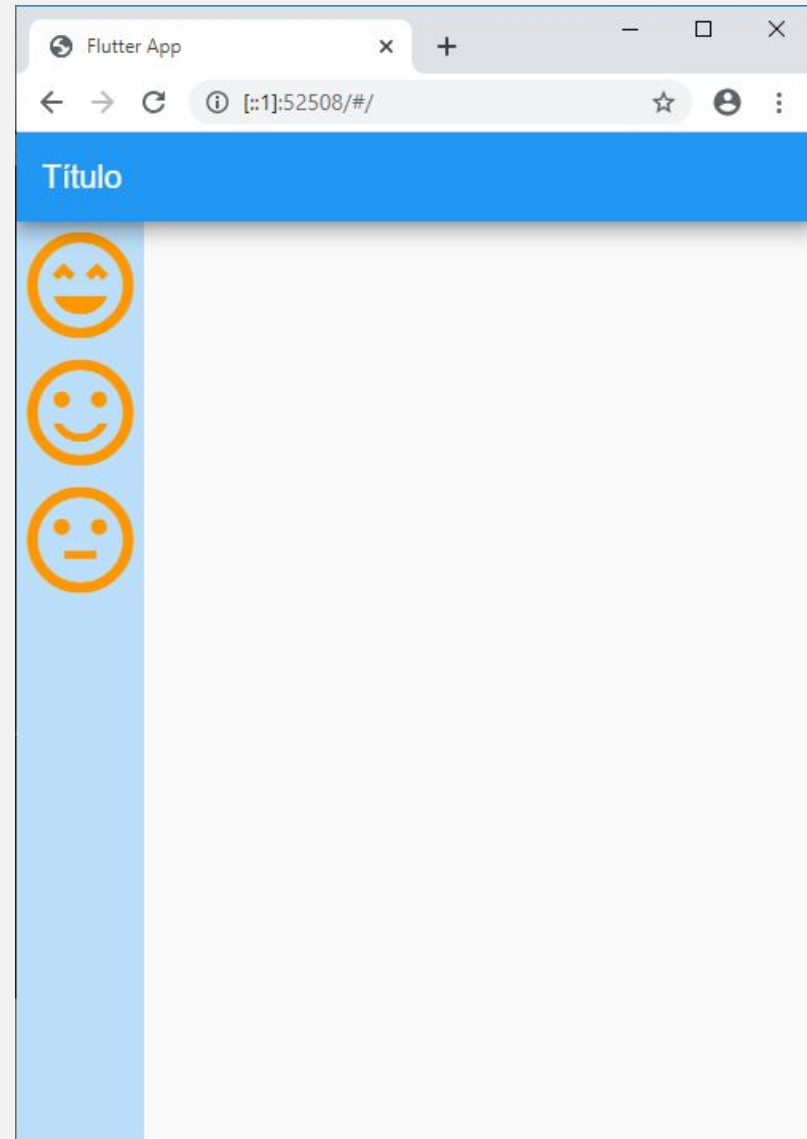
## ■ **Row**

- Widget utilizado para exibir widgets filhos em um arranjo vertical.



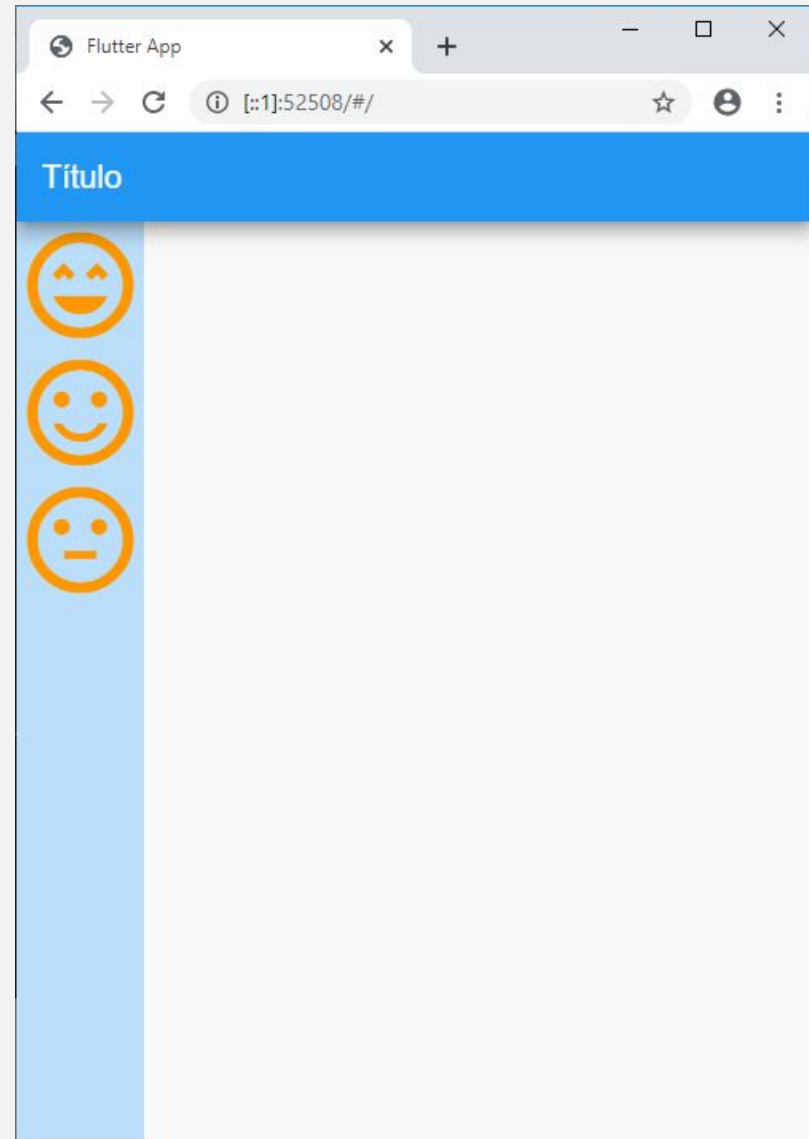


```
Column(  
  children: <Widget>[  
    Icon(  
      Icons.sentiment_very_satisfied,  
      size: 80.0,  
      color: Colors.orange  
    ),  
    Icon(  
      Icons.sentiment_satisfied,  
      size: 80.0,  
      color: Colors.orange  
    ),  
    Icon(  
      Icons.sentiment_neutral,  
      size: 80.0,  
      color: Colors.orange  
    ),  
  ],  
)
```



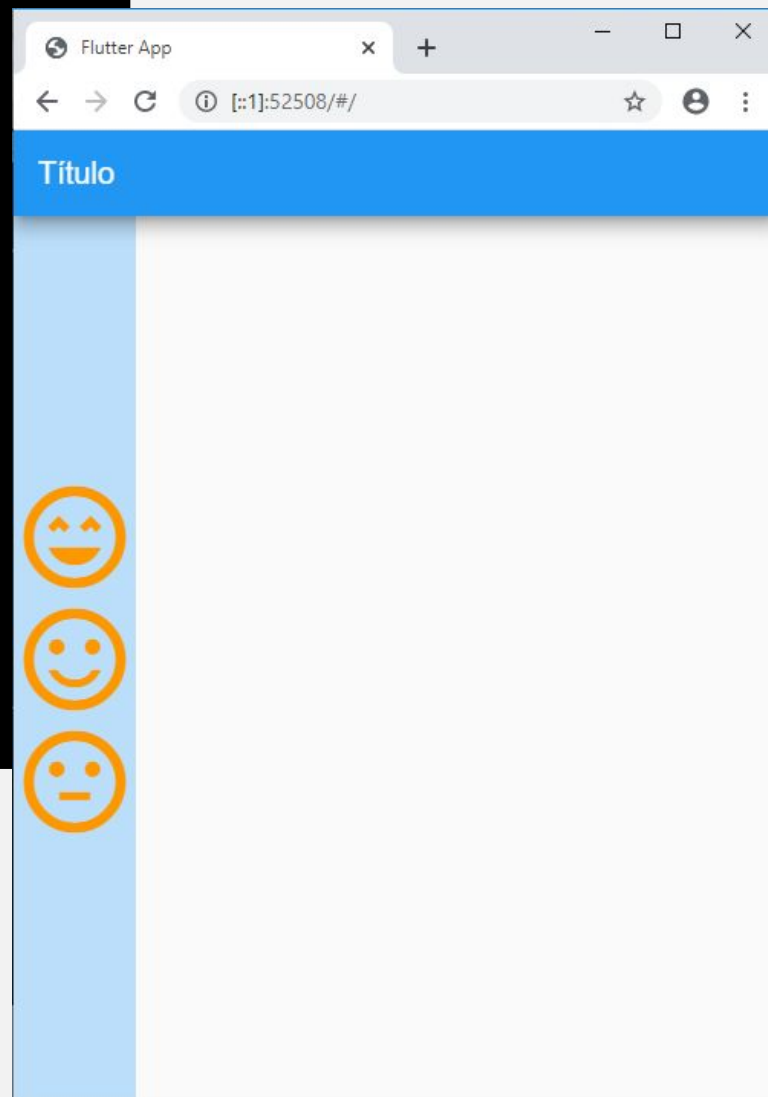


```
Column(  
  children: <Widget>[  
    Icon(  
      Icons.sentiment_very_satisfied,  
      size: 80.0,  
      color: Colors.orange  
    ),  
    Icon(  
      Icons.sentiment_satisfied,  
      size: 80.0,  
      color: Colors.orange  
    ),  
    Icon(  
      Icons.sentiment_neutral,  
      size: 80.0,  
      color: Colors.orange  
    ),  
  ],  
)
```



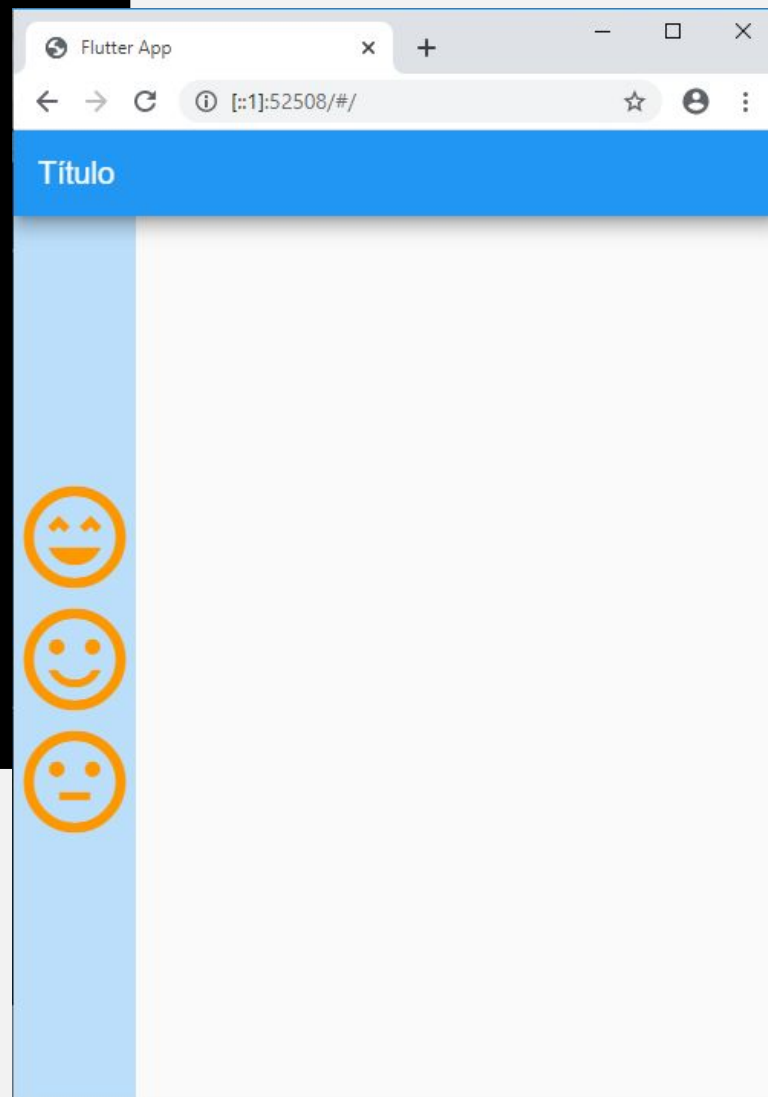


```
Column(  
  mainAxisAlignment: MainAxisAlignment.center,  
  children: <Widget>[  
    Icon(  
      Icons.sentiment_very_satisfied,  
      size: 80.0,  
      color: Colors.orange  
    ),  
    Icon(  
      Icons.sentiment_satisfied,  
      size: 80.0,  
      color: Colors.orange  
    ),  
    Icon(  
      Icons.sentiment_neutral,  
      size: 80.0,  
      color: Colors.orange  
    ),  
  ],  
)
```





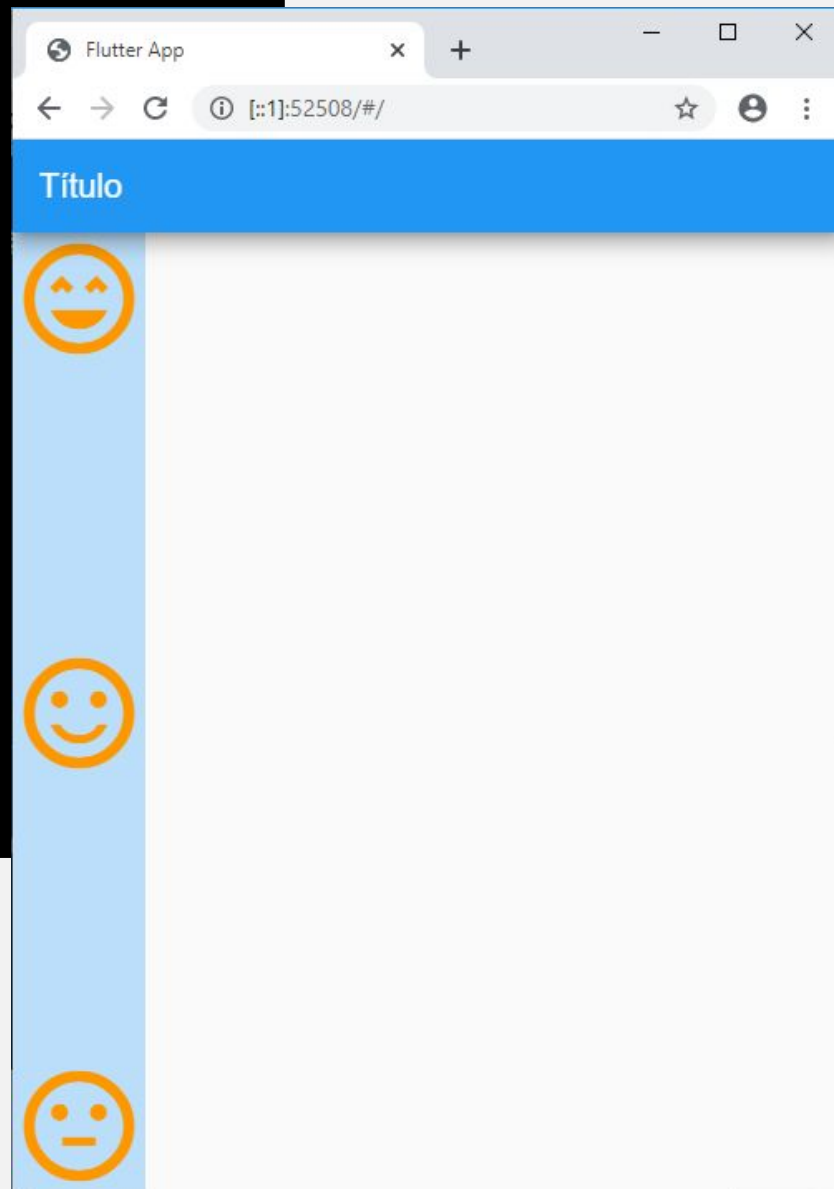
```
Column(  
  mainAxisAlignment: MainAxisAlignment.center,  
  children: <Widget>[  
    Icon(  
      Icons.sentiment_very_satisfied,  
      size: 80.0,  
      color: Colors.orange  
    ),  
    Icon(  
      Icons.sentiment_satisfied,  
      size: 80.0,  
      color: Colors.orange  
    ),  
    Icon(  
      Icons.sentiment_neutral,  
      size: 80.0,  
      color: Colors.orange  
    ),  
  ],  
)
```





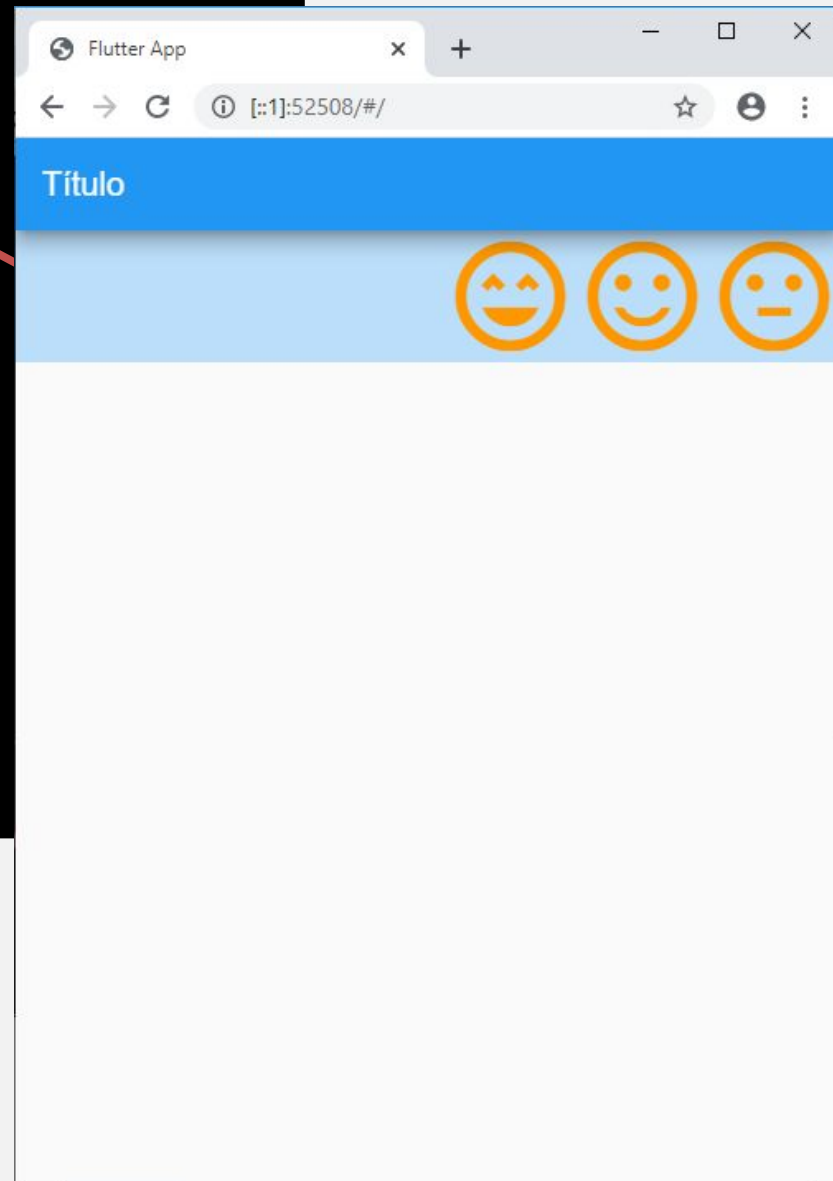


```
Column(  
  mainAxisAlignment: MainAxisAlignment.spaceBetween,  
  children: <Widget>[  
    Icon(  
      Icons.sentiment_very_satisfied,  
      size: 80.0,  
      color: Colors.orange  
    ),  
    Icon(  
      Icons.sentiment_satisfied,  
      size: 80.0,  
      color: Colors.orange  
    ),  
    Icon(  
      Icons.sentiment_neutral,  
      size: 80.0,  
      color: Colors.orange  
    ),  
  ],  
)
```





```
Row(  
  mainAxisAlignment: MainAxisAlignment.end,  
  children: <Widget>[  
    Icon(  
      Icons.sentiment_very_satisfied,  
      size: 80.0,  
      color: Colors.orange  
    ),  
    Icon(  
      Icons.sentiment_satisfied,  
      size: 80.0,  
      color: Colors.orange  
    ),  
    Icon(  
      Icons.sentiment_neutral,  
      size: 80.0,  
      color: Colors.orange  
    ),  
  ],  
)
```



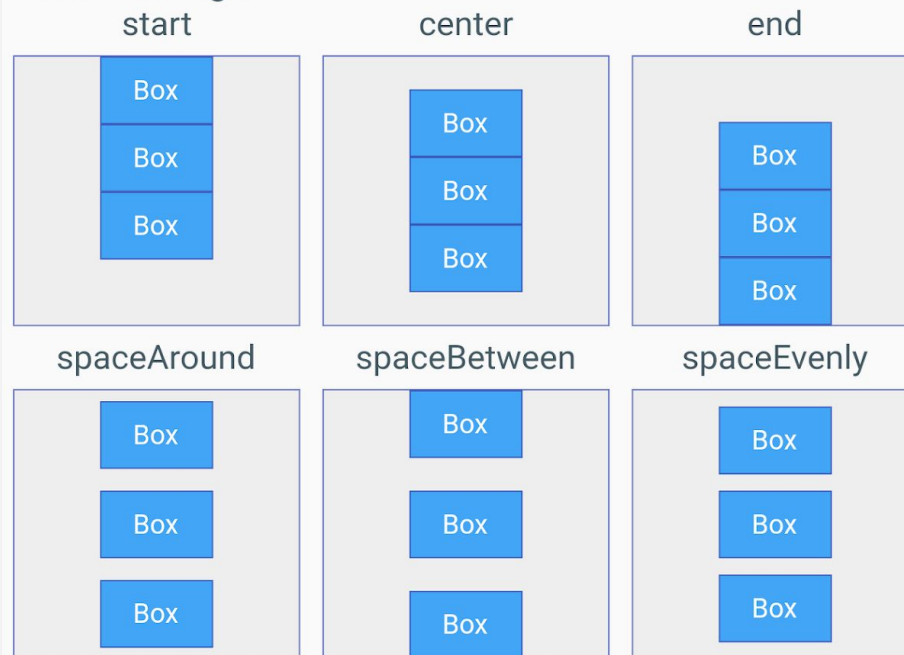


## Column Widget Alignment

### \* **CrossAxisAlignment**



### \* **MainAxisAlignment**



Alinhamento Row

← Row widget - MainAxisAlignment

Default (MainAxisAlignment.start)

Box

Box

Box

MainAxisAlignment.center

Box

Box

Box

MainAxisAlignment.end

Box

Box

Box

MainAxisAlignment.spaceEvenly

Box

Box

Box

MainAxisAlignment.spaceBetween

Box

Box

Box

MainAxisAlignment.spaceAround

Box

Box

Box

← Row widget - CrossAxisAlignment

default (CrossAxisAlignment.center)

Box

Box

Box

Default (CrossAxisAlignment.start)

Box

Box

Box

CrossAxisAlignment.end

Box

Box

Box

CrossAxisAlignment.stretch

Box

Box

Box

CrossAxisAlignment.baseline

>HelloHelloHello

CrossAxisAlignment.center & MainAxisAlignment.center

Box

Box

Box

# STATEFUL





- São widgets que possuem estado mutável.
- Utilizados sempre que é necessário realizar alterações nos widgets da UI.
- Um *Stateful Widget* é composto por:
  - uma classe ***StatefulWidget***
  - uma classe ***State***



- São widgets que possuem estado mutável.
- Utilizados sempre que é necessário realizar alterações nos widgets da UI.
- Um *Stateful Widget* é composto por:
  - uma classe ***StatefulWidget***
  - uma classe ***State***



Responsável pela inicialização  
da classe *State*.



- São widgets que possuem estado mutável.
- Utilizados sempre que é necessário realizar alterações nos widgets da UI.
- Um *Stateful Widget* é composto por:
  - uma classe ***StatefulWidget***
  - uma classe ***State***



A classe ***State*** contém as variáveis e informa a classe ***StatefulWidget*** quando e como se construir.





- São widgets que possuem estado mutável.
- Utilizados sempre que é necessário realizar alterações nos widgets da UI.
- Um *Stateful Widget* é composto por:
  - uma classe ***StatefulWidget***
  - uma classe ***State***



O método ***setState*** é usado para notificar que o estado interno de um objeto foi alterado.



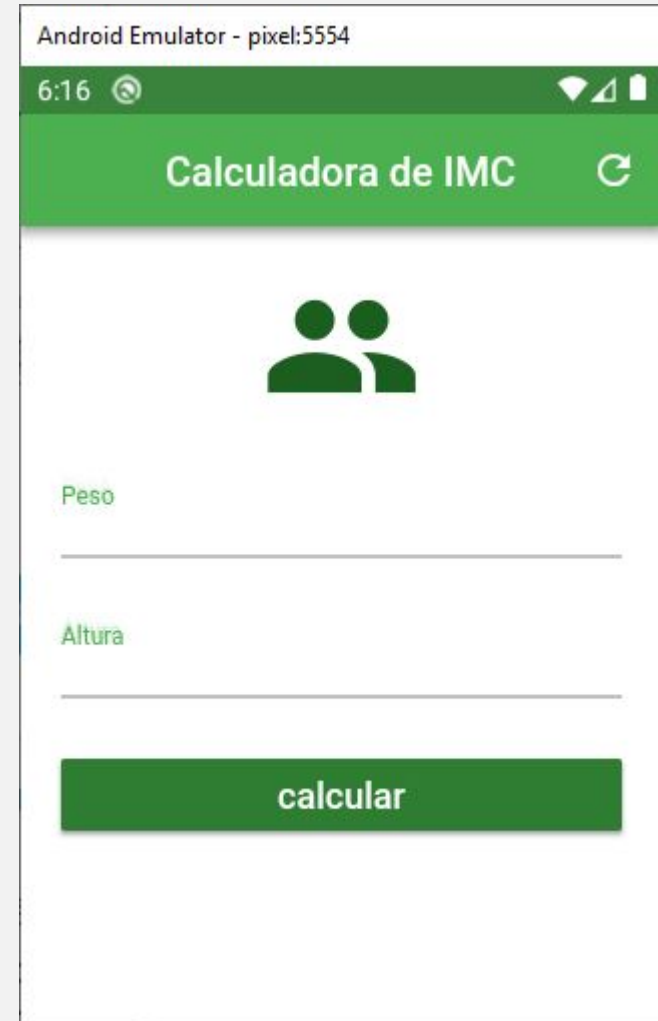
```
class Exemplo extends StatefulWidget {  
  @override  
  _ExemploState createState() => _ExemploState();  
}
```

```
class _ExemploState extends State<Exemplo> {  
  @override  
  Widget build(BuildContext context) {  
    return Container(  
  
    );  
  }  
}
```

# APP CALCULADORA IMC



- Objetivo
  - Desenvolver um App para calcular o Índice de Massa Corporal (IMC)






Android Emulator - pixel:5554

6:16

Calculadora de IMC



Peso

---

Altura

---

calcular

## Principais widgets:


- Scaffold
- IconButton
- Form
- Icon
- TextFormField
- RaisedButton



Android Emulator - pixel:5554

6:16

Calculadora de IMC



Peso

Altura

calcular

## Principais widgets:


- Scaffold
- IconButton
- Form
- Icon
- TextFormField
- RaisedButton



Android Emulator - pixel:5554

6:16

Calculadora de IMC



Peso

Altura

calcular

## Principais widgets:

- Scaffold
- IconButton
- Form
- Icon
- TextFormField
- RaisedButton



Android Emulator - pixel:5554

6:16

Calculadora de IMC



Peso

Altura

calcular

## Principais widgets:

- Scaffold
- IconButton
- Form
- Icon
- TextFormField
- RaisedButton






Android Emulator - pixel:5554

6:16

Calculadora de IMC



Peso

Altura

calcular

## Principais widgets:

- Scaffold
- IconButton
- Form
- Icon
- TextFormField
- RaisedButton



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

void main() => runApp(MyApp());

class MyApp extends StatefulWidget {
  @override
  _MyAppState createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: "Calculadora IMC",
      home: ImcWidget(),
    );
  }
}
```



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

class ImcWidget extends StatefulWidget {
  @override
  _ImcWidgetState createState() => _ImcWidgetState();
}

class _ImcWidgetState extends State<ImcWidget> {
  GlobalKey<FormState> _formkey = GlobalKey<FormState>();

  TextEditingController txtPeso = TextEditingController();
  TextEditingController txtAltura = TextEditingController();
  String _resultado = "";

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Calculadora de IMC"),
        backgroundColor: Colors.green,
        actions: <Widget>[
          IconButton(
            icon: Icon(Icons.refresh),
            onPressed: () { /*_reset();*/ }
          ),
        ],
      body: null,
      backgroundColor: Colors.white);
  }
}
```



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

class ImcWidget extends StatefulWidget {
  @override
  _ImcWidgetState createState() => _ImcWidgetState();
}

class _ImcWidgetState extends State<ImcWidget> {
  GlobalKey<FormState> _formkey = GlobalKey<FormState>();

  TextEditingController txtPeso = TextEditingController();
  TextEditingController txtAltura = TextEditingController();
  String _resultado = "";

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Calculadora de IMC"),
        backgroundColor: Colors.green,
        actions: <Widget>[
          IconButton(
            icon: Icon(Icons.refresh),
            onPressed: () { /* _reset(); */ }
          ),
        ],
      body: null,
      backgroundColor: Colors.white);
  }
}
```

### GlobalKeys

- Identificam exclusivamente os elementos.
- Fornecem acesso a outros objetos associados a esses elementos, como BuildContext.
- Para StatefulWidget, as chaves globais também fornecem acesso ao State.



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

class ImcWidget extends StatefulWidget {
  @override
  _ImcWidgetState createState() => _ImcWidgetState();
}

class _ImcWidgetState extends State<ImcWidget> {
  GlobalKey<FormState> _formkey = GlobalKey<FormState>();

  TextEditingController txtPeso = TextEditingController();
  TextEditingController txtAltura = TextEditingController();
  String _resultado = "";

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Calculadora de IMC"),
        backgroundColor: Colors.green,
        actions: <Widget>[
          IconButton(
            icon: Icon(Icons.refresh),
            onPressed: () { /* _reset(); */ }
          ),
        ],
      body: null,
      backgroundColor: Colors.white);
  }
}
```

### *TextEditingController*

- Um controlador para um campo de texto editável.
- Sempre que o usuário modifica um campo de texto com um TextEditingController associado, o campo de texto atualiza o valor e o controlador notifica seus ouvintes.
- Os ouvintes podem ler as propriedades de texto e seleção para saber o que o usuário digitou ou como a seleção foi atualizada.



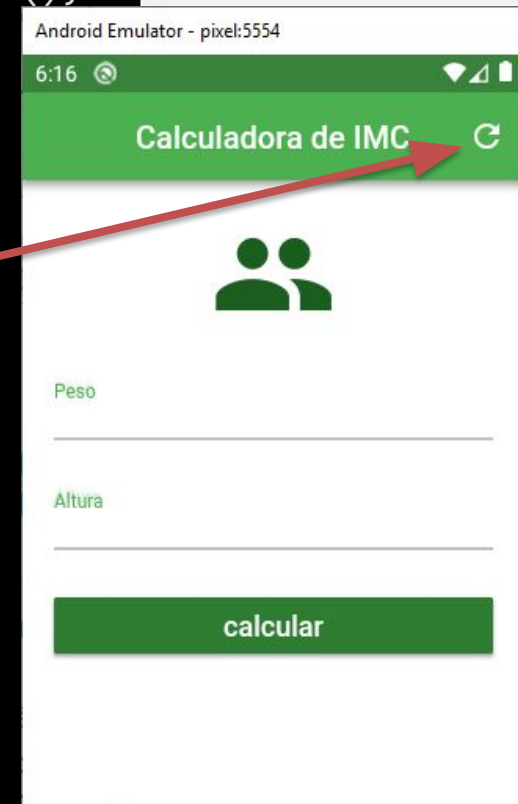
```
import 'package:flutter/material.dart';

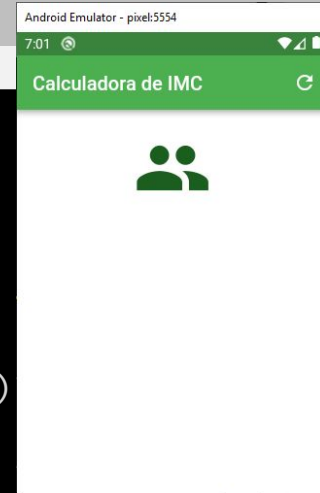
class ImcWidget extends StatefulWidget {
  @override
  _ImcWidgetState createState() => _ImcWidgetState();
}

class _ImcWidgetState extends State<ImcWidget> {
  GlobalKey<FormState> _formkey = GlobalKey<FormState>();

  TextEditingController txtPeso = TextEditingController();
  TextEditingController txtAltura = TextEditingController();
  String _resultado = "";

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Calculadora de IMC"),
        backgroundColor: Colors.green,
        actions: <Widget>[
          IconButton(
            icon: Icon(Icons.refresh),
            onPressed: () { /* _reset(); */ }
          ),
        ],
      ),
      body: null,
      backgroundColor: Colors.white);
  }
}
```





```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text("Calculadora de IMC"),
      backgroundColor: Colors.green,
      actions: <Widget>[
        IconButton(icon: Icon(Icons.refresh), onPressed: () { /*_reset();*/ })
      ],
    ),
    body: SingleChildScrollView(
      padding: const EdgeInsets.all(20.0),
      child: Form(
        key: _formkey,
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.stretch,
          children: <Widget>[
            Icon(Icons.people, size: 80, color: Colors.green[900]),

            //campos do formulário

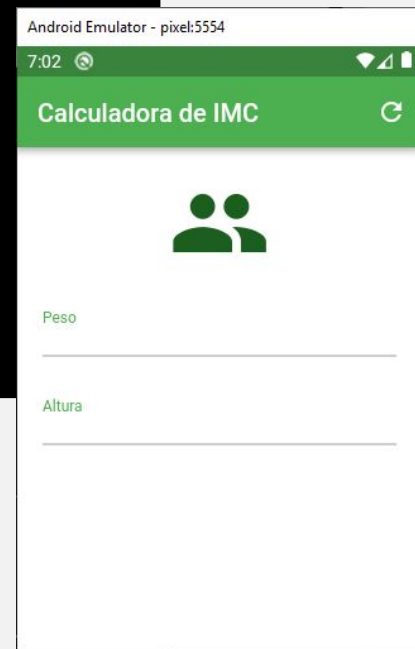
          ],
        ),
      ),
    ),
  ),
  backgroundColor: Colors.white);
}
```



```
campoTexto(rotulo, controle) {  
  return Container(  
    padding: const EdgeInsets.symmetric(vertical: 5),  
    child: TextFormField(  
      keyboardType: TextInputType.number,  
      style: TextStyle(color: Colors.green[900], fontSize: 20),  
      decoration: InputDecoration(  
        labelText: rotulo,  
        labelStyle: TextStyle(  
          color: Colors.green[500],  
          fontSize: 12,  
        ),  
      ),  
      controller: controle,  
      validator: (value) {  
        return (value.isEmpty) ? "Informe o valor" : null;  
      },  
    ),  
  );  
}
```

Chamar método no **body** do **Scaffold** (abaixo do Icon).

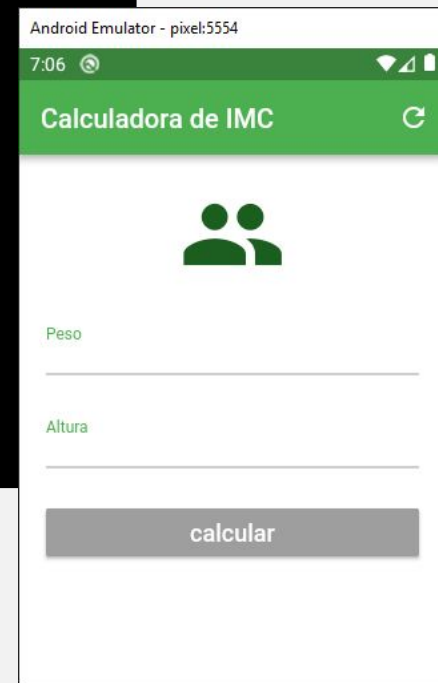
```
campoTexto("Peso", txtPeso),  
campoTexto("Altura", txtAltura),
```







```
botaoCalcular(BuildContext context) {  
  return Container(  
    padding: const EdgeInsets.only(top: 20),  
    child: RaisedButton(  
      child: Text(  
        "calcular",  
        style: TextStyle(  
          color: Colors.white,  
          fontSize: 18,  
        ),  
      ),  
      color: Colors.grey[500],  
      onPressed: () {  
        if (_formkey.currentState.validate()) {  
          // _calcular();  
        }  
      },  
    ));  
}
```



Chamar método no **body** do **Scaffold**

botaoCalcular(context),



```
void _reset() {  
  setState(() {  
    txtPeso.text = "";  
    txtAltura.text = "";  
    _resultado = "";  
  });  
}
```

### *setState*

- Notifique o framework que o estado interno deste objeto foi alterado.


```
void _calcular() {  
  setState(() {  
    double peso = double.parse(txtPeso.text);  
    double altura = double.parse(txtAltura.text);  
    double imc = peso / pow(altura, 2);  
    _resultado = "Resultado\ndIMC = ${imc.toStringAsPrecision(2)}";  
  });  
}
```

`import 'dart:math';`

```
exibirResultado() {  
  return Container(  
    padding: EdgeInsets.only(top: 20),  
    child: Text(_resultado,  
      style: TextStyle(color: Colors.green[900], fontSize: 20)),  
  );  
}
```



Adicionar a chamada do método ***exibirResultado***

```
child: Column(  
  crossAxisAlignment: CrossAxisAlignment.stretch,  
  children: <Widget>[  
  
    Icon(Icons.people, size: 80, color: Colors.green[900]),  
    campoTexto("Peso", txtPeso),  
    campoTexto("Altura", txtAltura),  
    botaoCalcular(context),  
     exibirResultado(),  
  
  ]),
```

Remover o comentário da chamada ***\_reset()***

Remover o comentário da chamada ***\_calcular()*** e adicionar a linha abaixo para não ocorrer problemas em relação ao foco.


```
FocusScope.of(context).requestFocus(new FocusNode());
```



Android Emulator - pixel:5554

7:17

Calculadora de IMC



Peso

90

Altura

1.90

calcular

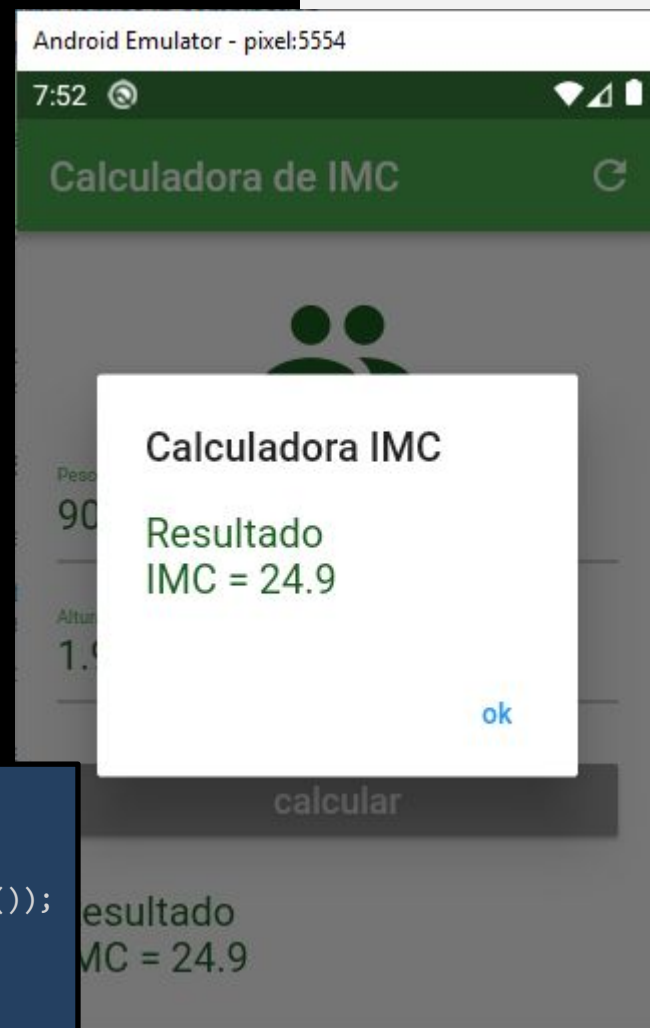
Resultado

IMC = 24.9



Para exibir o resultado na forma de uma *caixa de diálogo*

```
exibirResultadoDialog(BuildContext context) {  
  return showDialog(  
    context: context,  
    builder: (BuildContext context) {  
      return AlertDialog(  
        title: Text("Calculadora IMC"),  
        content: Text(_resultado),  
        actions: <Widget>[  
          FlatButton(  
            child: Text("ok"),  
            onPressed: () {  
              Navigator.pop(context);  
            },  
          ),  
        ],  
      );  
    },  
  );  
}  
  
onPressed: () {  
  if (_formkey.currentState.validate()) {  
    _calcular();  
    FocusScope.of(context).requestFocus(new FocusNode());  
    exibirResultadoDialog(context);  
  }  
},
```



# **ATIVIDADE PRÁTICA**

# Atividade Prática

1. Construir um App a partir do seguinte protótipo:

**Cadastro de Livros**

Título

Autor

Editora

gravar

limpar

limpar o conteúdo dos campos de texto

exibir em uma caixa de diálogo o conteúdo informado nos campos

Detailed description: The image shows a wireframe for a 'Book Registration' (Cadastro de Livros) application. It features a title bar, three input fields for 'Título', 'Autor', and 'Editora', and two buttons at the bottom: 'gravar' (save) and 'limpar' (clear). Annotations with arrows point to the buttons: 'gravar' is linked to 'exibir em uma caixa de diálogo o conteúdo informado nos campos' (display in a dialog box the content entered in the fields), and 'limpar' is linked to 'limpar o conteúdo dos campos de texto' (clear the content of the text fields).

# Atividade Prática

## 2. Construir um App a partir do seguinte protótipo:

**Registro de Notas Fiscais**

Valor R\$  
890.00  → Armazenar um número indeterminado de notas

---

Informações sobre as Notas

Qtde. de Notas	<input type="text"/>	→ somente leitura
Maior valor R\$	<input type="text"/>	→ somente leitura
Menor valor R\$	<input type="text"/>	→ somente leitura
Soma total R\$	<input type="text"/>	→ somente leitura



# Atividade Prática

3. Construir um App a partir do seguinte protótipo:



**FIM**