



WIDGETS EM FLUTTER



O QUE SÃO WIDGETS E COMO FUNCIONAM EM FLUTTER

- Widgets são os blocos de construção básicos de uma interface de usuário em Flutter.
- Tudo em Flutter é um widget, incluindo elementos visuais, layouts, estilos e até mesmo o aplicativo em si.
- Os widgets são declarativos e reativos, o que significa que a interface de usuário é atualizada automaticamente sempre que ocorrem mudanças nos widgets.



WIDGETS BÁSICOS

- **Container:** Um widget versátil usado para criar layouts, aplicar margens, preenchimentos e decorações aos elementos.
- **Row:** Um widget que organiza seus filhos horizontalmente em uma linha.
- **Column:** Um widget que organiza seus filhos verticalmente em uma coluna.
- **Text:** Um widget usado para exibir texto com estilo.
- **Image:** Um widget usado para exibir imagens em um aplicativo Flutter.
- E muitos outros widgets disponíveis no Flutter para diferentes finalidades.



COMPOSIÇÃO DE WIDGETS PARA CRIAR LAYOUTS

1

Em Flutter, os layouts são criados pela composição de vários widgets.

2

Os widgets são aninhados uns dentro dos outros para criar a hierarquia visual do aplicativo.

3

A composição flexível de widgets permite a criação de layouts complexos e personalizados.

EXEMPLO DE COMPOSIÇÃO DE WIDGETS

- O código cria um **Container** contendo um **Row** com dois filhos: um **Icon** representando uma estrela e um **Text** exibindo a mensagem "Avaliação: 4.5". O **Row** organiza esses dois filhos em uma linha horizontal dentro do **Container**

```
1  Container(  
2    child: Row(  
3      children: [  
4        Icon(Icons.star),  
5        Text('Avaliação: 4.5'),  
6      ],  
7    ),  
8  );  
9  |
```

**colocar exemplos com
códigos?**

MÉTODO CONSTRUTOR

- Um método construtor, como o próprio nome já diz, é responsável pela criação do objeto daquela classe, iniciando com valores seus atributos ou realizando outras funções que possam vir a ser necessárias.
- Em Flutter, classes também têm construtores que permitem personalizar a criação de objetos

TIPOS DE CONSTRUTORES EM FLUTTER

- Construtor padrão: é fornecido automaticamente pelo Flutter se nenhum construtor personalizado for definido
- Construtor nomeado: permite criar construtores adicionais com nomes diferentes para atender a necessidades específicas

EXEMPLO DE MÉTODO CONSTRUTOR PADRÃO

```
1  import 'package:flutter/material.dart';
2
3  class ExemploClasse {
4    String nome;
5    int idade;
6
7    // Construtor padrão
8    ExemploClasse({
9      this.nome = '',
10     this.idade = 0,
11   });
12 }
13
14 void main() {
15   // Criando uma instância da classe ExemploClasse
16   ExemploClasse exemplo = ExemploClasse();
17
18   // Atribuindo valores aos atributos da instância
19   exemplo.nome = 'João';
20   exemplo.idade = 25;
21
22   // Imprimindo os valores
23   print('Nome: ${exemplo.nome}');
24   print('Idade: ${exemplo.idade}');
25 }
```

EXEMPLO DE MÉTODO CONSTRUTOR PADRÃO

- Neste exemplo, a classe **ExemploClasse** possui dois atributos: **nome** e **idade**.
- O construtor padrão é definido entre chaves {} e utiliza parâmetros nomeados para permitir a atribuição dos valores dos atributos no momento da criação de uma instância da classe.
- O construtor padrão define valores padrão vazios para os atributos caso nenhum valor seja fornecido.
- No método **main**, criamos uma instância da classe **ExemploClasse** chamada **exemplo** e atribuímos valores aos atributos **nome** e **idade**. Em seguida, imprimimos os valores na saída.

EXEMPLO DE METÓDO CONSTRUTOR NOMEADO

```
1  import 'package:flutter/material.dart';
2
3  class ExemploClasse {
4    String nome;
5    int idade;
6
7    // Construtor nomeado
8    ExemploClasse.nomeado({String nome = '', int idade = 0}) {
9      this.nome = nome;
10     this.idade = idade;
11   }
12 }
13
14 void main() {
15   // Criando uma instância da classe ExemploClasse usando o construtor nomeado
16   ExemploClasse exemplo = ExemploClasse.nomeado(nome: 'Maria', idade: 30);
17
18   // Imprimindo os valores
19   print('Nome: ${exemplo.nome}');
20   print('Idade: ${exemplo.idade}');
21 }
```

EXEMPLO DE MÉTODO CONSTRUTOR PADRÃO

- Neste exemplo, a classe **ExemploClasse** possui dois atributos: **nome** e **idade**. O construtor nomeado **ExemploClasse.nomeado** é definido utilizando a sintaxe {} para parâmetros nomeados. Ele permite a atribuição dos valores dos atributos no momento da criação de uma instância da classe.
- No método **main**, criamos uma instância da classe **ExemploClasse** chamada **exemplo** utilizando o construtor nomeado **ExemploClasse.nomeado**. Fornecemos os valores dos parâmetros **nome** e **idade** ao criar a instância.
- Em seguida, imprimimos os valores dos atributos **nome** e **idade** na saída.

GESTÃO DE EVENTOS EM WIDGETS

- Em Flutter, os widgets podem responder a eventos, como toques, gestos e interações do usuário.
- Cada widget tem métodos específicos para lidar com eventos.
- Alguns exemplos de eventos incluem onTap, onLongPress, onDoubleTap, entre outros.

EXEMPLO DE GESTAO DE EVENTOS

1. Importamos as bibliotecas necessárias.
2. Definimos uma classe **MyApp** que é o ponto de entrada do aplicativo.
3. **MyApp** retorna um **MaterialApp** que configura o aplicativo com um título e a página inicial.
4. A página inicial é definida como um **Scaffold** com um **AppBar** e um **body** centralizado.
5. No corpo (**body**), temos um único **ElevatedButton** com um texto "Clique aqui".
6. Quando o botão é clicado (**onPressed**), uma função anônima é chamada para exibir a mensagem "Botão clicado!" no console.

```
1 import 'package:flutter/material.dart';
2
3 void main() {
4   runApp(MyApp());
5 }
6
7 class MyApp extends StatelessWidget {
8   @override
9   Widget build(BuildContext context) {
10     return MaterialApp(
11       title: 'Eventos em Flutter',
12       home: Scaffold(
13         appBar: AppBar(
14           title: Text('Gestão de Eventos'),
15         ),
16         body: Center(
17           child: ElevatedButton(
18             onPressed: () {
19               print('Botão clicado!');
20             },
21             child: Text('Clique aqui'),
22           ),
23         ),
24       ),
25     );
26   }
27 }
```

CONCLUSÃO

OS WIDGETS SÃO OS BLOCOS DE CONSTRUÇÃO FUNDAMENTAIS EM FLUTTER PARA CRIAR INTERFACES DE USUÁRIO.

IMPORTÂNCIA DO USO DE CONSTRUTORES EM FLUTTER PARA CRIAR E INICIALIZAR OBJETOS DE FORMA EFICIENTE

A GESTÃO DE EVENTOS EM WIDGETS PERMITE A INTERAÇÃO DO USUÁRIO EM SEU APLICATIVO FLUTTER.

REFERÊNCIAS

- <https://devsimo.es.medium.com/widgets-do-flutter-básico-8d5b4fcd62dd>
- <https://docs.flutter.dev/reference/widgets>