

Formulários

Usabilidade, Desenv. Web, Mobile e Jogos

Prof. Gustavo Custodio
gustavo.custodio@anhembí.br

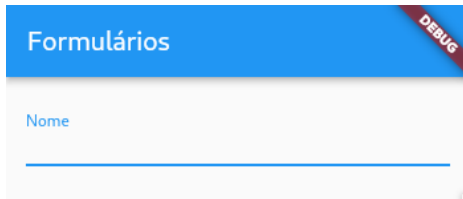


Formulários

Formulários

Campo de Texto

- Junto com botões, uma forma extremamente importante de interações de usuário são os campos de texto.
 - Em Flutter, são criados utilizando o widget `TextField`.



Campo de Texto

- Os campos de texto frequentemente são utilizados em conjunto com formulários.
 - Por isso, Flutter possui uma versão do `TextField` dedicada para formulários.
 - `TextFormField`.

Formulário

- Vamos começar criando uma aplicação inicial.
 - Primeiro criamos a tela principal, um `StatelessWidget`.
 - Depois criamos o formulário, um `StatefulWidget`.

Formulário

- No arquivo formulario.dart:

```
1 class Formulario extends StatefulWidget {  
2   @override  
3   State<Formulario> createState() =>  
4     _FormularioState();  
5 }  
6  
7 class _FormularioState extends State<Formulario> {  
8   @override  
9   Widget build(BuildContext context) {  
10    return Scaffold(  
11      appBar: AppBar(title: Text('Formulários')),  
12      body: Form( // Widget para formularios  
13        child: Padding(  
14          padding: EdgeInsets.all(15),  
15          child: Column(),  
16        ),  
17      ),  
18    );  
19  }  
20 }
```

Formulário

- Utilizamos um Widget chamado Form para colocar os widgets que farão parte do aplicativo

Formulário

- No arquivo main.dart:

```
1 import 'package:flutter/material.dart';
2 import 'formulario.dart';
3
4 void main() {
5   runApp(const MyApp());
6 }
7
8 class MyApp extends StatelessWidget {
9   const MyApp({Key? key}) : super(key: key);
10
11   @override
12   Widget build(BuildContext context) {
13     return MaterialApp(
14       title: 'Flutter Demo',
15       theme: ThemeData(
16         primarySwatch: Colors.blue,
17       ),
18       home: Formulario(),
19     );
20   }
21 }
```


TextFormField

- Adicione um campo de texto no formulário.

```
1   final controllerNome = TextEditingController();
2
3   Widget _criarTextFromField(BuildContext context) {
4       return TextFormField(
5           controller: controllerNome,
6           decoration: InputDecoration(labelText: "Nome"),
7       );
8   }
```

TextFormField

```
1   child: Column(  
2     children: [  
3       _criarTextFromField(context),  
4     ],  
5   ),
```

- O `TextEditingController` é utilizado para recuperar o valor do texto quando necessário.

DropDownButton

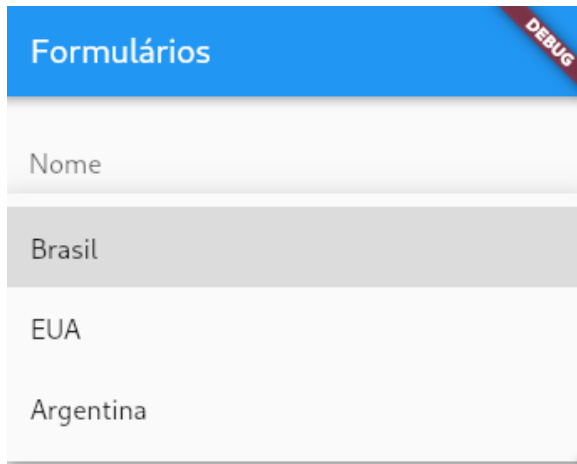
- Outro elemento que pertence a um formulário é o `DropDownButtonFormField`.
 - Quando clicado ele mostra um conjunto de opções.
- Vamos adicionar um desses menus no formulário.

DropDownButton

```
1 String? _nomePais;
2
3 Widget _criarDropDown(BuildContext context) {
4   return DropDownButtonFormField(
5     hint: Text('Escolha um país'),
6     items: [
7       DropdownMenuItem(child: Text("Brasil"), value: "Brasil"),
8       DropdownMenuItem(child: Text("EUA"), value: "EUA"),
9       DropdownMenuItem(child: Text("Argentina"), value: "Argentina"),
10    ],
11    value: _nomePais,
12    onChanged: (String? valorEscolhido) {
13      setState(() {
14        _nomePais = valorEscolhido;
15      });
16    },
17  );
18 }
```

- Criamos um botão com 3 opções: Brasil, EUA e Argentina.

DropDownButton



The image shows a screenshot of a Windows Forms application. At the top, there is a blue title bar with the text "Formulários" in white. In the top right corner of the title bar, there is a red ribbon-like button with the word "DEBUG" in white. Below the title bar, there is a light gray rectangular area containing a "DropDownButton" control. The control has a light gray background and a small downward-pointing arrow on its right side. The text "Nome" is displayed on the first line, "Brasil" on the second line, "EUA" on the third line, and "Argentina" on the fourth line. The control is currently displaying "Brasil".

DropDownButton

- A variável `_nomePais` é a variável associada ao valor selecionado.
- Explicação dos atributos:
 - `items`:
 - Formados por widgets `DropDownMenuItem`, que possuem atributos `value`, que indicam qual valor a variável `_nomePais` recebe.
 - `value`: variável que recebe o valor do campo.
 - `hint`: o que aparece na tela ante de interagirmos com o campo de texto.
 - `onChanged`: o que acontece quando mudamos o valor do `DropDown`.

DropDownButton

- Também é possível criar um DropDownButton por meio de uma lista.

```
1 List<String> listaPaíses = ["Brasil", "EUA", "Argentina"];
2
3 return DropDownButtonFormField(
4   hint: Text("Escolha um país"),
5   items: listaPaíses.map((element) {
6     return DropdownMenuItem(
7       value: element,
8       child: Text(element),
9     );
10  }).toList(),
```

Salvando Offline

- Vamos adicionar um botão para salvar as informações do formulário de forma offline:

```
1  Widget _criarBotaoDeSalvar(BuildContext context) {  
2      return ElevatedButton(  
3          onPressed: _salvarInformacoes,  
4          child: Text("Salvar Info"),  
5      );  
6  }
```


Salvando Offline

```
1 Future _salvarInformacoes() async {
2     SharedPreferences prefs = await SharedPreferences.getInstance();
3     await prefs.clear();
4     if (_nomePais != null) {
5         await prefs.setString("pais", _nomePais!);
6     }
7     await prefs.setString("nome", controllerNome.text);
8
9 }
```

Salvando Offline

```
1  child: Column(  
2    children: [  
3      _criarTextFromField(context),  
4      _criarDropdown(context),  
5      Spacer(),  
6      _criarBotaoDeSalvar(context),  
7    ],  
8  ),
```

Salvando Offline

- Caso tenha algum problema com esse código, entre na pasta do projeto e execute os comandos:
 - `flutter clean.`
 - `flutter pub get.`

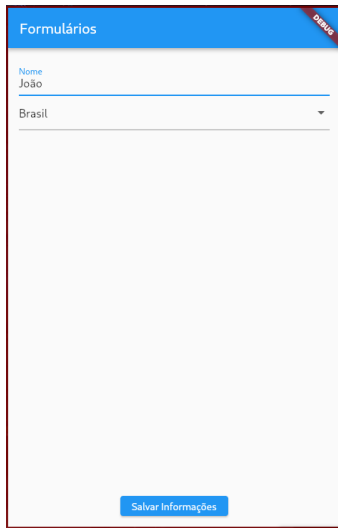
Carregando as informações

- Depois de salvar os elementos do formulário, vamos carregá-los automaticamente quando abrirmos o aplicativo.
- Para isso, utilizamos o `initState`.

Carregando as Informações

```
1  @override
2  void initState() {
3    _carregarInformacoes();
4    super.initState();
5  }
6
7  Future _carregarInformacoes() async {
8    SharedPreferences prefs = await SharedPreferences.getInstance();
9    controllerNome.text = prefs.getString("nome") ?? "";
10   setState(() {
11     _nomePais = prefs.getString("pais");
12   });
13 }
```

Carregando as Informações



The image shows a mobile application interface for a form titled "Formulários". The form has a blue header bar with the title "Formulários" and a red "DEBUG" label in the top right corner. Below the header, there is a text input field labeled "Nome" with the value "João". Below that is a dropdown menu labeled "Brasil" with a downward arrow. At the bottom of the form is a blue button labeled "Salvar Informações".

Formulários

DEBUG

Nome
João

Brasil

Salvar Informações



Formulários

Exercícios

Exercício 1

- Adicione um alerta no botão de “Salvar Informações” avisando que as informações do formulário foram salvas com sucesso.

Exercício 2

- Adicione os seguintes campos no formulário:
 - Idade;
 - O campo só pode aceitar valores numéricos.
 - CPF;
 - Data de Nascimento.

Exercício 3

- Adicione um atributo validator nos campos de texto para impedi-los de serem vazios.
- Exemplo:

```
1  validator: (value) {  
2    if (value == null || value.isEmpty) {  
3      return "Por favor, insira algum texto";  
4    }  
5    return null;  
6  }
```

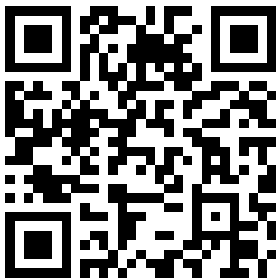
Referências

 Simone Alessandria and Brian Kayfitz.

Flutter Cookbook: Over 100 proven techniques and solutions for app development with Flutter 2.2 and Dart.

Packt Publishing Ltd, 2021.

Conteúdo



<https://gustavotcustodio.github.io/usabilidade.html>

Obrigado

gustavo.custodio@anhembi.br