Desenvolver Interface Gráfica Para Dispositivos Móveis

Aula 10

Professor: Henrique Augusto Maltauro

Fecomércio PR
Sesc Senac IFPD
Senac

github.com/hmaltaurodev/flutter-ui-tests

Flutter/Dart

- **Color.shade**

O shade é uma propriedade da classe Color, que permite aplicar um sombreamento sobre uma cor já existente.

Esse sombreamento pode fazer a cor ficar mais clara, ou mais escura.

Flutter/Dart

- **Color.shade**

As propriedades shade do Color são:

- ➔ **shade50**
- ➔ **shade100**
- ➔ **shade200**
- ➔ **shade300**

Flutter/Dart

- Color.shade

➔ shade400

➔ shade500

➔ shade600

➔ shade700

➔ shade800

➔ shade900

Flutter/Dart

- **Color.shade**

```
Colors.blueGrey.shade100;
Colors.yellow.shade50;
```

Flutter/Dart

- **Form**

O Form é uma classe/widget do Flutter, que implementa um formulário, permitindo que os campos de texto possam ser validados.

Esse é um exemplo de widget onde nós precisamos informar a Key dele, para ser possível usar todas as funcionalidades de um Form.

Flutter/Dart

- **Form**

Algumas propriedades do Form são:

➔ **key**

- Que recebe uma Key?, permitindo controlar o processo de validação e de reset.

➔ **child**

- Que recebe um Widget para ser apresentado dentro do Form.

Flutter/Dart

- **Form**

```
Form(
    key: Key?,
    child: Widget
)  // Form
```

Flutter/Dart

- **Form**

Para criar uma Key para o Form, é criado uma variável/constante do tipo Global Key, que recebe como parâmetro de tipo um FormState.

Flutter/Dart

- **Form**

```
final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
```

Flutter/Dart

- **TextFormField**

O TextFormField é uma classe/widget do Flutter que implementa o um campo de texto para ser utilizado dentro de um Form.

Ele é, convenientemente, uma junção de outros dois componentes: TextField e FormField.

Flutter/Dart

- **TextFormField**

Um widget Form como pai não é obrigatório para utilizar o TextFormField.

Contudo, o Form permite o processo de salvar, resetar e validar múltiplos TextFormFields de uma vez só.

Flutter/Dart

- **TextFormField**

Algumas propriedades do TextFormField são:

➔ **onChanged**

- Que recebe uma Função? para ser executada sempre que o valor do campo for alterado. Essa função tem um parâmetro do tipo String?, e que retorna void.

Flutter/Dart

- **TextFormField**

➔ **validator**

- Que recebe uma Função? para ser validar o valor do campo. Essa função tem um parâmetro do tipo String?, e que retorna uma String? com o texto de campo inválido.

Flutter/Dart

- **TextFormField**

➔ **autovalidateMode**

- Que recebe um AutovalidateMode? para definir se e quando a validação do campo deve acontecer.

➔ **controller**

- Que recebe um TextEditingController? que irá gerenciar todo os valores do campo de texto.

Flutter/Dart

- **TextFormField**

➔ **keyboardType**

  - Que recebe um TextInputType? para definir qual o tipo de teclado aquele campo irá utilizar.

➔ **maxLength**

  - Que recebe um int? para definir qual o tamanho máximo do texto digitado.

Flutter/Dart

- **TextFormField**

➔ **obscureText**

  - Que recebe um bool para definir se o texto daquele campo ficará escondido ou não. Normalmente utilizado para campos de senhas.

Flutter/Dart

- **TextFormField**

➔ **decoration**

  - Que recebe um InputDecoration? para definir se o campo de texto irá ter algum detalhamento, como um ícone, uma borda, uma cor diferente, e até mesmo uma label.

Flutter/Dart

- **TextFormField**

➔ **readOnly**

- Que recebe um bool para definir se aquele campo de texto é somente para leitura ou para alteração também.

➔ **enabled**

- Que recebe um bool? para definir se aquele campo de texto está habilitado ou não.

Flutter/Dart

- **TextFormField**

```
TextFormField(
    onChanged: Function?,
    validator: Function?,
    autovalidateMode: AutovalidateMode?,
    controller: TextEditingController?,
    keyboardType: TextInputType?
)  // TextFormField
```

Flutter/Dart

- **TextFormField**

```
TextFormField(
    maxLength: int?,
    obscureText: bool,
    decoration: InputDecoration?,
    readOnly: bool,
    enabled: bool?
)  // TextFormField
```

Flutter/Dart

- **AutovalidateMode**

O AutovalidateMode é um enum do Flutter, que configura o processo de validação de um FormField.

Ele não é classificado como um componente, ou seja, ele sempre vai precisar estar relacionado a algum outro componente.

Ele possui três valores diferentes.

Flutter/Dart

- **AutovalidateMode**

➔ **disabled**

- Que define que a validação não vai acontecer.

➔ **always**

- Que define que a validação sempre vai acontecer.

➔ **onUserInteraction**

- Que define que a validação só vai acontecer depois que o usuário interagir com o FormField.

Flutter/Dart

- **AutovalidateMode**

```
AutovalidateMode.disabled;
AutovalidateMode.always;
AutovalidateMode.onUserInteraction;
```

Flutter/Dart

- **TextEditingController**

O TextEditingController é uma classe do Flutter que implementa um controlador para um campo de texto editável.

Ele não é classificado como um componente, ou seja, ele sempre vai precisar estar relacionado a algum outro componente.

Flutter/Dart

- **TextEditingController**

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 então ler o texto e as propriedades da seleção para saber o que o usuário digitou ou como a seleção foi atualizada.

Flutter/Dart

- **TextEditingController**

Da mesma forma, se você modificar as propriedades do texto ou da seleção, o campo de texto será notificado e se atualizará adequadamente.

Um TextEditingController também pode ser usado para fornecer um valor inicial para um campo de texto.

Flutter/Dart

- **TextEditingController**

```
final TextEditingController _fieldController = TextEditingController();
```

Flutter/Dart

- **TextInputType**

O TextInputType é uma classe do Flutter utilizada para otimizar o teclado que será apresentado em tela, de acordo com a necessidade do campo.

Ele não é classificado como um componente, ou seja, ele sempre vai precisar estar relacionado a algum outro componente.

Flutter/Dart

- **TextInputType**

O TextInputType possui 11 constantes estáticas diferentes:

➔ **datetime**

- Que vai otimizar o teclado para uma informação de data e hora.

➔ **emailAddress**

- Que vai otimizar o teclado para um email.

Flutter/Dart

- **TextInputType**

➔ **multiline**

- Que vai otimizar o teclado para um texto de várias linhas.

➔ **name**

- Que vai otimizar o teclado para um nome.

➔ **none**

- Que vai fazer o teclado não ser exibido.

Flutter/Dart

- **TextInputType**

➔ **number**

- Que vai otimizar o teclado para uma informação numérica.

➔ **phone**

- Que vai otimizar o teclado para um telefone.

➔ **streetAddress**

- Que vai otimizar o teclado para um endereço.

Flutter/Dart

- **TextInputType**

➔ **text**

- Que vai otimizar o teclado para uma informação de texto.

➔ **url**

- Que vai otimizar o teclado para uma url.

➔ **visiblePassword**

- Que vai otimizar o teclado para uma senha visível.

Flutter/Dart

- **TextInputType**

```
TextInputType.datetime;
TextInputType.always;
TextInputType.multiline;
TextInputType.name;
TextInputType.none;
TextInputType.number;
```

Flutter/Dart

- **TextInputType**

```
TextInputType.phone;
TextInputType.streetAddress;
TextInputType.text;
TextInputType.url;
TextInputType.visiblePassword;
```

Flutter/Dart

- **InputDecoration**

O InputDecoration é uma classe do Flutter que implementa um decorador para o campo de texto.

Ele não é classificado como um componente, ou seja, ele sempre vai precisar estar relacionado a algum outro componente.

Flutter/Dart

- **InputDecoration**

Algumas propriedades do InputDecoration são:

- **icon**
  - Que recebe um Widget? para definir um ícone.
- **iconColor**
  - Que recebe um Color? para definir a cor do ícone.

Flutter/Dart

- **InputDecoration**

➔ **label**

- Que recebe um Widget? para definir o label.

➔ **labelText**

- Que recebe uma String? para definir o texto do label.

Nessa situação, somente um dos dois pode ser utilizado.

Flutter/Dart

- **InputDecoration**

```
InputDecoration(
    icon: Widget?,
    iconColor: Color?,
    label: Widget?,
    labelText: String?
)  // InputDecoration
```

Flutter/Dart

- **InputDecoration**

➔ **labelStyle**

- Que recebe um TextStyle? para definir a estilização do texto do label.

➔ **floatingLabelStyle**

- Que recebe um TextStyle? para definir a estilização do texto do label, quando ele for definido como um label flutuante.

Flutter/Dart

- **InputDecoration**

➔ **helperText**

  - Que recebe uma String? para definir um texto de ajuda.

➔ **helperStyle**

  - Que recebe um TextStyle? para definir a estilização do texto de ajuda.

Flutter/Dart

- **InputDecoration**

```
InputDecoration(
    labelStyle: TextStyle?,
    floatingLabelStyle: TextStyle?,
    helperText: String?,
    helperStyle: TextStyle?
)  // InputDecoration
```

Flutter/Dart

- **InputDecoration**

➔ **hintText**

- Que recebe uma String? para definir um texto de dica.

➔ **hintStyle**

- Que recebe um TextStyle? para definir a estilização do texto de dica.

Flutter/Dart

- **InputDecoration**

➔ **error**

- Que recebe um Widget? para definir uma mensagem de erro.

➔ **errorText**

- Que recebe uma String? para definir o texto de uma mensagem de erro.

Nessa situação, somente um dos dois pode ser utilizado.

Flutter/Dart

- **InputDecoration**

```
InputDecoration(
    hintText: String?,
    hintStyle: TextStyle?,
    error: Widget?,
    errorText: String?
)  // InputDecoration
```

Flutter/Dart

- **InputDecoration**

➔ **errorStyle**

- Que recebe um TextStyle? para definir a estilização do texto da mensagem de erro.

➔ **floatingLabelBehavior**

- Que recebe um objeto do tipo FloatingLabelBehavior? para definir se o label vai ser flutuante ou não.

Flutter/Dart

- **InputDecoration**

➔ **floatingLabelAlignment**

- Que recebe um objeto do tipo FloatingLabelAlignment? para definir a posição do label flutuante.

➔ **contentPadding**

- Que recebe um objeto do tipo EdgeInsetsGeometry? para definir as bordas internas do campo de texto.

Flutter/Dart

- **InputDecoration**

```
InputDecoration(
    errorStyle: TextStyle?,
    floatingLabelBehavior: FloatingLabelBehavior?,
    floatingLabelAlignment: FloatingLabelAlignment?,
    contentPadding: EdgeInsetsGeometry?
)  // InputDecoration
```

Flutter/Dart

- **InputDecoration**

➔ **prefix**

- Que recebe um Widget? para definir um prefixo.

➔ **prefixText**

- Que recebe uma String? para definir um texto de prefixo.

Nessa situação, somente um dos dois pode ser utilizado.

Flutter/Dart

- **InputDecoration**

➔ **prefixStyle**

- Que recebe um TextStyle? para definir a estilização do texto de prefixo.

➔ **prefixIcon**

- Que recebe um Widget? para definir um ícone de prefixo. É apresentado antes do prefix ou do prefixText.

Flutter/Dart

- **InputDecoration**

```
InputDecoration(
    prefix: Widget?,
    prefixText: String?,
    prefixStyle: TextStyle?,
    prefixIcon: Widget?
)  // InputDecoration
```

Flutter/Dart

- **InputDecoration**

➔ **suffix**

- Que recebe um Widget? para definir um sufixo.

➔ **suffixText**

- Que recebe uma String? para definir um texto de sufixo.

Nessa situação, somente um dos dois pode ser utilizado.

Flutter/Dart

- **InputDecoration**

➔ **suffixStyle**

- Que recebe um TextStyle? para definir a estilização do texto de sufixo.

➔ **suffixIcon**

- Que recebe um Widget? para definir um ícone de sufixo. É apresentado depois do sufix ou do sufixText.

Flutter/Dart

- **InputDecoration**

```
InputDecoration(
    suffix: Widget?,
    suffixText: String?,
    suffixStyle: TextStyle?,
    suffixIcon: Widget?
)  // InputDecoration
```

Flutter/Dart

- **InputDecoration**

➔ **prefixIconColor**

- Que recebe um Color? para definir a cor do ícone de prefixo.

➔ **sufixIconColor**

- Que recebe um Color? para definir a cor do ícone de sufixo.

Flutter/Dart

- **InputDecoration**

➔ **focusColor**

- Que recebe um Color? para definir a cor do campo de texto quando ele estiver em foco.

➔ **hoverColor**

- Que recebe um Color? para definir a cor do campo de texto quando o cursor estiver em cima.

Flutter/Dart

- **InputDecoration**

```
InputDecoration(
    prefixIconColor: Color?,
    sufixIconColor: Color?,
    focusColor: Color?,
    hoverColor: Color?
)  // InputDecoration
```

Flutter/Dart

- **InputDecoration**

➔ **errorBorder**

- Que recebe um objeto do tipo InputBorder? para definir a borda do campo de texto, quando existir algum erro.

➔ **focusedBorder**

- Que recebe um objeto do tipo InputBorder? para definir a borda do campo de texto, quando o mesmo estiver em foco.

Flutter/Dart

- **InputDecoration**

➔ **focusedErrorBorder**

- Que recebe um objeto do tipo InputBorder? para definir a borda do campo de texto, quando existir algum erro e o mesmo estiver em foco.

➔ **disabledBorder**

- Que recebe um objeto do tipo InputBorder? para definir a borda do campo de texto, quando o mesmo estiver desabilitado.

Flutter/Dart

- **InputDecoration**

```
InputDecoration(
    errorBorder: InputBorder?,
    focusedBorder: InputBorder?,
    focusedErrorBorder: InputBorder?,
    disabledBorder: InputBorder?
)  // InputDecoration
```

Flutter/Dart

- **InputDecoration**

➔ **enabledBorder**

- Que recebe um objeto do tipo InputBorder? para definir a borda do campo de texto, quando o mesmo estiver habilitado.

➔ **border**

- Que recebe um objeto do tipo InputBorder? para definir a borda padrão do campo de texto.

Flutter/Dart

- **InputDecoration**

➔ **filled**

- Que recebe um bool? para definir se o corpo do campo de texto vai ser preenchido com alguma cor.

➔ **fillColor**

- Que recebe um Color? para definir a cor que o corpo do campo de texto vai ser preenchido.

Flutter/Dart

- **InputDecoration**

```
InputDecoration(
    enabledBorder: InputBorder?,
    border: InputBorder?,
    filled: bool?,
    fillColor: Color?
)  // InputDecoration
```

Flutter/Dart

- **InputBorder**

O InputBorder é uma classe do Flutter que implementa uma configuração de borda para um InputDecoration.

Ele não é classificado como um componente, ou seja, ele sempre vai precisar estar relacionado a algum outro componente.

Flutter/Dart

- **InputBorder**

Para facilitar o nosso trabalho, o Flutter já possui duas classes que definem previamente essa estilização de bordas:

➔ **OutlineInputBorder**

- Que implementa uma borda ao redor do campo de texto

➔ **UnderlineInputBorder**

- Que implementa uma linha inferior ao campo de texto.