Desenvolver Interface Gráfica Para Dispositivos Móveis

Aula 13

Professor: Henrique Augusto Maltauro

Fecomércio PR
Sesc Senac IFPD
Senac

Flutter/Dart

- **Padding**

O Padding é uma classe/widget do Flutter que implementa um distanciamento ao redor de um componente.

Flutter/Dart

- **Padding**

As principais propriedades do Padding são:

➔ **child**

- Que recebe um Widget? para ser apresentado com o espaçamento.

➔ **padding**

- Que recebe um EdgeInsetsGeometry para definir o tamanho do espaçamento.

Flutter/Dart

- **Padding**

```
Padding(
    child: Widget?,
    padding: EdgeInsetsGeometry
)  // Padding
```

Flutter/Dart

- **EdgeInsetsGeometry**

O EdgeInsetsGeometry é uma classe do Flutter que define um valor de espaçamento nas bordas de um componente.

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

Flutter/Dart

- **EdgeInsets**

A melhor forma de usar o EdgeInsetsGeometry é com o EdgeInsets.

O EdgeInsets é uma classe do Flutter que possui uma série de métodos estáticos que definem os valores dos espaçamentos de forma padronizada.

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

Flutter/Dart

- **EdgeInsets**

Os principais métodos estáticos do EdgeInsets são:

➔ **all**

- Que recebe um valor double para defini-lo como espaçamento para todos os lados do componente.

Flutter/Dart

- **EdgeInsets**

```
EdgeInsets.all(double);
```

Flutter/Dart

- **EdgeInsets**

➔ **symmetric**

- Que recebe dois valores double para defini-los como espaçamento para os lados do componente, de forma simétrica. Os parâmetros são nomeados e são o **vertical** e o **horizontal**.

Flutter/Dart

- **EdgeInsets**

```
EdgeInsets.symmetric(vertical: double = 0.0,
                     horizontal: double = 0.0);
```

Flutter/Dart

- **EdgeInsets**

➔ **only**

  - Que recebe quatro valores double para defini-los como espaçamento para cada um dos lados do componente. Os parâmetros são nomeados e são o **left**, o **top**, o **right** e o **bottom**.

Flutter/Dart

- **EdgeInsets**

```
EdgeInsets.only(left: double = 0.0,
                top: double = 0.0,
                right: double = 0.0,
                bottom: double = 0.0);
```

Flutter/Dart

- **ButtonStyle**

O ButtonStyle é uma classe do Flutter que implementa uma estilização para um botão.

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

Flutter/Dart

- **ButtonStyle**

A melhor forma de se utilizar o ButtonStyle é sobrescrevendo a aparência do botão em questão com o método estático styleFrom.

Por exemplo: se a ideia é definir um estilo para o ElevatedButton, se utiliza o ElevatedButton.styleFrom.

Flutter/Dart

- **ButtonStyle**

```
ElevatedButton.styleFrom();
```

Flutter/Dart

- **ButtonStyle**

As principais propriedades do styleFrom são:

➔ **foregroundColor**

- Que recebe um Color? para definir a cor dos textos e ícones do botão.

➔ **backgroundColor**

- Que recebe um Color? para definir a cor do fundo do botão.

Flutter/Dart

- **ButtonStyle**

➔ **disabledForegroundColor**

- Que recebe um Color? para definir a cor dos textos e ícones do botão, quando ele estiver desativado.

➔ **disabledBackgroundColor**

- Que recebe um Color? para definir a cor do fundo do botão, quando ele estiver desativado.

Flutter/Dart

- **ButtonStyle**

```
ElevatedButton.styleFrom(
    foregroundColor: Color?,
    backgroundColor: Color?,
    disabledForegroundColor: Color?,
    disabledBackgroundColor: Color?
)
```

Flutter/Dart

- **ButtonStyle**

➔ **shadowColor**

- Que recebe um Color? para definir a cor da sombra do botão.

➔ **elevation**

- Que recebe um double? para definir o quanto o botão fica elevado.

Flutter/Dart

- **ButtonStyle**

➔ **textStyle**

  - Que recebe um TextStyle? para definir o estilo do texto do botão.

➔ **padding**

  - Que recebe um EdgeInsetsGeometry? para definir o espaçamento interno do botão.

Flutter/Dart

- **ButtonStyle**

```
ElevatedButton.styleFrom(
    shadowColor: Color?,
    elevation: double?,
    textStyle: TextStyle?,
    padding: EdgeInsetsGeometry?
)
```

Flutter/Dart

- **ButtonStyle**

➔ **side**

  - Que recebe um BorderSide? para definir a borda do botão.

➔ **shape**

  - Que recebe um OutlinedBorder? para definir o formato do botão.

Flutter/Dart

- **ButtonStyle**

```
ElevatedButton.styleFrom(
    side: BorderSide?,
    shape: OutlinedBorder?
)
```

Flutter/Dart

- **ElevatedButton**

Para definir o estilo do ElevatedButton utilizamos:

➔ **style**

- Que recebe um ButtonStyle? para definir o estilo do botão.

Flutter/Dart

- **IconButton**

O IconButton é uma classe/widget do Flutter que implementa um ícone botão.

Flutter/Dart

- **IconButton**

As principais propriedades do IconButton são:

➔ **icon**

- Que recebe um Widget para definir o ícone do botão.

➔ **iconSize**

- Que recebe um double? para definir o tamanho do ícone.

Flutter/Dart

- **IconButton**

➔ **padding**

- Que recebe um EdgeInsetsGeometry? para definir o espaçamento interno do botão.

➔ **onPressed**

- Que recebe uma Function para definir o que acontece quando o botão é clicado. Se essa função for nula, o botão é tido como desabilitado.

Flutter/Dart

- **IconButton**

```
IconButton(
    icon: Widget?,
    iconSize: double?,
    padding: EdgeInsetsGeometry?,
    onPressed: Function?,
)  // IconButton
```

Flutter/Dart

- **Icon**

Algumas outras propriedades do Icon são:

➔ **size**

- Que recebe um double? para definir o tamanho do ícone.

➔ **color**

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

Flutter/Dart

- **SingleChildScrollView**

O SingleChildScrollView é uma classe/widget do Flutter, que implementa um scroll para um componente.

Ou seja, ele permite que a tela do aplicativo seja rolável.

Flutter/Dart

- **SingleChildScrollView**

As principais propriedades do SingleChildScrollView são:

➔ **child**

- Que recebe um Widget? para definir o componente que será rolável.

➔ **padding**

- Que recebe um EdgeInsetsGeometry? para definir o espaçamento interno da área rolável.

Flutter/Dart

- **SingleChildScrollView**

➔ **reverse**

  - Que recebe um bool para definir se a rolagem irá acontecer no sentido inverso.

➔ **scrollDirection**

  - Que recebe um Axis para definir o sentido da rolagem.

Flutter/Dart

- **SingleChildScrollView**

```
SingleChildScrollView(
    child: Widget?,
    padding: EdgeInsetsGeometry?,
    reverse: bool,
    scrollDirection: Axis,
)  // SingleChildScrollView
```