Desenvolver Interface Gráfica Para Dispositivos Móveis

Aula 16

Professor: Henrique Augusto Maltauro

Fecomércio PR
Sesc Senac IFPD
Senac

Flutter/Dart

- **Expanded**

O Expanded é uma classe/widget do Flutter que implementa uma expansão de componentes, permitindo que eles ocupem os espaços disponíveis.

Flutter/Dart

- **Expanded**

A principal propriedade do Expanded é:

➔ **child**

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

Flutter/Dart

- **Expanded**

```
Expanded(
    child: Widget?
)   // Expanded
```

Flutter/Dart

- **GestureDetector**

O GestureDetector é uma classe/widget do Flutter que implementa uma espaço em tela, que detecta movimentos no touch screen.

Ele possui uma enormidade de funções, que permite que seja detectado os movimentos mais complexos que possam ser feitos no touch screen.

Isso permite, transformar qualquer componente num botão.

Flutter/Dart

- **GestureDetector**

Algumas das propriedades do GestureDetector são:

➔ **child**

- Que recebe um Widget? para definir o componente que irá detectar os movimentos.

➔ **onTap**

- Que recebe uma Função?, para definir o que acontece quando o componente é clicado.

Flutter/Dart

- **GestureDetector**

➔ **onDoubleTap**

- Que recebe uma Função?, para definir o que acontece quando o componente é clicado duas vezes bem rápido.

➔ **onLongPress**

- Que recebe uma Função?, para definir o que acontece quando o componente é segurado pressionado.

Flutter/Dart

- **GestureDetector**

```
GestureDetector(
    onTap: Função?,
    onDoubleTap: Função?,
    onLongPress: Função?,
    child: Widget?
)  // GestureDetector
```

Flutter/Dart

- **Decoration**

O Decoration é uma classe abstrata do Flutter utilizada para representar um grupo de classes que realiza decorações a outros componentes.

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

Flutter/Dart

- **BoxDecoration**

Uma das formas de se utilizar o Decoration é com o BoxDecoration, que é uma classe do Flutter que implementa de forma concreta Decoration no formato de uma caixa.

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

Flutter/Dart

- **BoxDecoration**

As principais propriedades do BoxDecoration são:

➔ **color**

- Que recebe um Color? para definir a cor de fundo da decoração.

Flutter/Dart

- **BoxDecoration**

➔ **border**

- Que recebe um BoxBorder? para definir uma borda que será desenhada sobre o fundo da decoração.

➔ **borderRadius**

- Que recebe um BorderRadiusGeometry? para definir um formato dos cantos da decoração.

Flutter/Dart

- **BoxDecoration**

```
BoxDecoration(
    color: Color?,
    border: BoxBorder?,
    borderRadius: BorderRadiusGeometry?
)  // BoxDecoration
```

Flutter/Dart

- **BoxDecoration**

➔ **padding**

- Que recebe um EdgeInsetsGeometry? para definir uma borda interna do componente.

➔ **shape**

- Que recebe um BoxShape para definir o formato do preenchimento da decoração.

Flutter/Dart

- **BoxDecoration**

```
BoxDecoration(
    padding: EdgeInsetsGeometry?,
    shape: BoxShape
)  // BoxDecoration
```

Flutter/Dart

- **BorderRadiusGeometry**

O BorderRadiusGeometry é uma classe do Flutter que implementa um formato para os cantos de um componente.

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

Flutter/Dart

- **BorderRadius**

Uma das formas de se trabalhar com o BorderRadiusGeometry, é utilizando o BorderRadius, que é uma classe do Flutter que implementa o BorderRadiusGeometry de forma concreta.

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

Flutter/Dart

- **BorderRadius**

O BorderRadius possui uma série métodos construtores, que são:

➔ **all**

- Que recebe um único parâmetro do tipo Radius, para definir o mesmo valor para todos os cantos.

Flutter/Dart

- **BorderRadius**

```
BorderRadius.all(Radius);
```

Flutter/Dart

- **BorderRadius**

➔ **circular**

- Que recebe um único parâmetro do tipo double, para definir o mesmo valor para todos os cantos de forma arredondada.

Flutter/Dart

- **BorderRadius**

```
BorderRadius.circular(double);
```

Flutter/Dart

- **BorderRadius**

➔ **horizontal**

- Que recebe dois parâmetros do tipo Radius, para definir os cantos de forma simétrica horizontalmente. Esses parâmetros são o left e o right.

Flutter/Dart

- **BorderRadius**

```
BorderRadius.horizontal(left: Radius,
                        right: Radius);
```

Flutter/Dart

- **BorderRadius**

➔ **vertical**

- Que recebe dois parâmetros do tipo Radius, para definir os cantos de forma simétrica verticalmente. Esses parâmetros são o top e o bottom.

Flutter/Dart

- **BorderRadius**

```
BorderRadius.vertical(top: Radius,
                      bottom: Radius);
```

Flutter/Dart

- **BorderRadius**

➔ **only**

  - Que recebe quatro parâmetros do tipo Radius, para definir cada um dos cantos. Esses parâmetros são o topLeft, topRight, bottomLeft e bottomRight.

Flutter/Dart

- **BorderRadius**

```
BorderRadius.only(topLeft: Radius,
                  topRight: Radius,
                  bottomLeft: Radius,
                  bottomRight: Radius);
```

Flutter/Dart

- **Radius**

O Radius é uma classe do Flutter que implementa uma forma circular ou elíptica.

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

Flutter/Dart

- **Radius**

O Radius possui dois métodos construtores, que são:

➔ **circular**

- Que recebe um único parâmetro do double, para definir um formato circular.

Flutter/Dart

- **Radius**

```
Radius.circular(double);
```

Flutter/Dart

- **Radius**

➔ **elliptical**

  - Que recebe um único parâmetro do double, para definir um formato elíptico.

Flutter/Dart

- **Radius**

```
Radius.elliptical(double);
```

Flutter/Dart

- **BoxShape**

O BoxShape é um enum do Flutter, utilizado para definir o formato que um BoxDecoration ou um Border será renderizado.

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

Flutter/Dart

- **BoxShape**

Os valores do enum BoxShape são:

➔ **rectangle**

- Que vai definir um formato retangular.

➔ **circle**

- Que vai definir um formato circular.

Flutter/Dart

- **BoxShape**

```
BoxShape.rectangle;
BoxShape.circle;
```