THEMEDATA

O objeto *ThemeData* é responsável por manter em si as configurações de aparência da aplicação, como cor e tamanho das fontes, margens e padding padrões dos *Widgets*, entre outras estilizações. Pense nesse objeto como uma configuração global de estilos para toda a nossa aplicação.

Seu uso nos trás diversos benefícios, como por exemplo:

- Unificação dos estilos da aplicação;
- Agrupamento dos estilos em um único local, facilitando a manutenção e futuras mudanças que afetam a aplicação como um todo;
- Padronização de componentes comuns entre páginas;

Esse objeto representa, basicamente, todas as customizações possíveis para nossos Widgets. Sendo assim, podemos utilizá-lo para criar um estilo padrão para todos os nossos Widgets.

> É importante ressaltar que devido ao grande número de *Widgets* e customizações possíveis para estes, trataremos das principais customizações possíveis para os principais e mais utilizados *Widgets*.

primaryColor

A propriedade *primaryColor*, como o próprio nome já sugere, define a cor primária da aplicação. Ou seja, a principal cor da aplicação, sendo aplicada a *toolbars, tab bars,* etc.

```dart
void main() {
  runApp(
    MaterialApp(
      title: "Exemplo Flutter",
      theme: ThemeData(
        primaryColor: Colors.indigo,
      ),
      home: Home(),
    ),
  );
}
```

accentColor

A propriedade *accentColor*, como o próprio nome já sugere, define a cor de acentuação da aplicação. Ou seja, a cor de contraste/destaque da aplicação, sendo aplicada a botões, texto, efeito de borda de deslocamento excessivo, etc.

```dart
void main() {
  runApp(
    MaterialApp(
      title: "Exemplo Flutter",
      theme: ThemeData(
        primaryColor: Colors.indigo,
        accentColor: Colors.amber,
      ),
      home: Home(),
    ),
  );
}
```

primarySwatch

A propriedade *primarySwatch* nos permite criar um esquema de cores padrão para nossa aplicação Flutter. Desta forma podemos utilizar um esquema de cores baseado em um cor base e suas sombras, onde variamos a cor base de um tom mais claro até um mais escuro.

Essa propriedade recebe um objeto *MaterialColor*, o qual utilizamos para definir a cor base e suas sombras, valores esses que variam de 50 até 900, formando assim 10 variações da cor base.

Podemos utilizar as cores pré-definidas no *Material Design* ou podemos criar nós mesmos as dez variações utilizando uma ferramenta para tal. Uma dessas ferramentas se encontra no site TL Dev Tech (https://www.tldevtech.com/flutter-color-generator/), onde podemos informar nossa cor base e ele irá gerar nossa paleta de cores *material*.

Com essa propriedade podemos criar um esquema de cores padrão para todos os nossos componentes materiais, criando uma harmonia de cores entre os mais diversos Widgets.

É importante ressaltar que caso a cor de acentuação for declarada, essa propriedade não irá afetar os Widgets estilizados por essa propriedade.

```dart
void main() {
  runApp(
    MaterialApp(
      title: "Exemplo Flutter",
      theme: ThemeData(
        primarySwatch: MaterialColor(0xFF673AB7, {
          50: Color(0xFFF8F6FC),
          100: Color(0xFFF0ECF8),
          200: Color(0xFFD9CEED),
          300: Color(0xFFC1AFE2),
          400: Color(0xFF9576CD),
          500: Color(0xFF673AB7),
          600: Color(0xFF5C34A3),
          700: Color(0xFF3E236E),
          800: Color(0xFF2F1853),
          900: Color(0xFF1E1136),
        }),
        accentColor: Colors.amber,
      ),
      home: Home(),
    ),
  );
}
```

scaffoldBackgroundColor

A propriedade *scaffoldBackgroundColor*, como o próprio nome já sugere, nos permite modificar a cor de fundo do *Scaffold* de nossa aplicação.

```dart
void main() {
  runApp(
    MaterialApp(
      title: "Exemplo Flutter",
      theme: ThemeData(
        primarySwatch: Colors.indigo,
        accentColor: Colors.amber,
        scaffoldBackgroundColor: Colors.grey[200],
      ),
      home: Home(),
    ),
  );
}
```

6.1 appBarTheme

A propriedade *appBarTheme*, como o próprio nome já sugere, define o tema padrão para as *AppBar* da aplicação. Sendo assim, teremos um tema padrão para a *AppBar* em todas as telas, centralizando assim a customização da aplicação.

Dentre as propriedade que podemos customizar estão:

- Cor de fundo;
- Elevação;
- Brilho;
- Posição do título;
- Entre outras.

```dart
void main() {
  runApp(
    MaterialApp(
      title: "Exemplo Flutter",
      theme: ThemeData(
        appBarTheme: AppBarTheme(
          centerTitle: true,
          elevation: 8,
          backgroundColor: Colors.indigo[900],
        ),
      ),
      home: Home(),
    ),
  );
}
```

textTheme

A propriedade *textTheme*, como o próprio nome já sugere, define o tema padrão para os textos da aplicação. Sendo assim, teremos um tema padrão para os textos em todas as telas, centralizando assim a customização da aplicação.

O *Material Design* classifica seus estilos de fontes em 3 grupos: *headLine*, *subtitle* e *bodyText*. Esses grupos são utilizados para definir a importância de cada elemento de texto em uma tela, assim teremos os títulos, subtítulos e textos de corpo. Sendo que todos eles recebem um objeto *TextStyle*.

Dentre as propriedade que podemos customizar estão:

- Cor do texto;
- Tamanho da fonte;
- Peso da fonte;
- Decoração do texto;
- Estilo da fonte;
- Família da fonte;
- Entre outras.

Theme

```dart
void main() {
  runApp(
    MaterialApp(
      title: "Exemplo Flutter",
      theme: ThemeData(
        textTheme: ThemeData.light().textTheme.copyWith(
              headline1: TextStyle(
                fontSize: 60.0,
                color: Colors.black,
              ),
              subtitle1: TextStyle(
                fontSize: 40.0,
              ),
              bodyText1: TextStyle(
                fontSize: 14.0,
              ),
              bodyText2: TextStyle(
                fontSize: 12.0,
              ),
            ),
      ),
      home: Home(),
    ),
  );
}
```

Uso do estilo

```dart
Text(
  "Titulo 1",
  style: Theme.of(context).textTheme.headline1,
),
Text(
  "Subtitulo 1",
  style: Theme.of(context).textTheme.subtitle1,
),
Text(
  "Texto de corpo 1",
  style: Theme.of(context).textTheme.bodyText1,
),
Text(
  "Texto de corpo 2",
  style: Theme.of(context).textTheme.bodyText2,
),
```

inputDecorationTheme

A propriedade *inputDecorationTheme,* define o tema padrão para decorações dos campos de texto da aplicação. Sendo assim, teremos um tema padrão para as decorações de campos de texto em todas as telas, centralizando assim a customização da aplicação.

Dentre as propriedade que podemos customizar estão:

- *border;*
- *enabled;*
- *fillColor;*
- *filled;*
- *hintStyle;*
- *hintText;*
- *labelStyle;*
- *labelText;*

- Entre outras.

```
void main() {
  runApp(
    MaterialApp(
      title: "Exemplo Flutter",
      theme: ThemeData(
        inputDecorationTheme: InputDecorationTheme(
          border: OutlineInputBorder(
            borderRadius: BorderRadius.only(
              topLeft: Radius.circular(10),
              topRight: Radius.circular(10),
            ),
          ),
        ),
      ),
      home: Home(),
    ),
  );
}
```

floatingActionButtonTheme

A propriedade *floatingActionButtonTheme* define o tema padrão para os botões *floatingActionButton* " da aplicação. Sendo assim, teremos um tema padrão para os *floatingActionButtons* em todas as telas, centralizando assim a customização da aplicação.

Dentre as propriedade que podemos customizar estão:

- *backgroundColor;*
- *elevation;*
- *shape;*
- Entre outras.

```dart
void main() {
  runApp(
    MaterialApp(
      title: "Exemplo Flutter",
      theme: ThemeData(
        floatingActionButtonTheme: FloatingActionButtonThemeData(
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(10.0),
          ),
        ),
      ),
      home: Home(),
    ),
  );
}
```