**Apostila Flutter (Parte 1): Guia Visual de Componentes (Widgets)**

**Introdução**

Bem-vindo ao guia visual de componentes do Flutter! Esta apostila foi criada para ajudar desenvolvedores, especialmente os iniciantes, a identificar e utilizar os widgets mais comuns na construção de interfaces de usuário (UI).

**O que são Widgets?**

Em Flutter, **quase tudo é um widget**. Pense neles como blocos de montar (LEGOs). Um botão é um widget, um texto é um widget, o espaçamento entre eles é um widget e até a tela inteira é um widget. Você combina esses blocos para construir a árvore de componentes que forma a sua interface.

**Material Design vs. Cupertino**

O Flutter oferece dois conjuntos principais de widgets que seguem diretrizes de design específicas:

* **Material Design (Google):** Um sistema de design arrojado e visual para plataformas Android, web e desktop. É o mais comum e o foco desta apostila.
* **Cupertino (Apple):** Um conjunto de widgets que seguem as diretrizes de design do iOS.

**1. Widgets Essenciais de Estrutura**

Estes widgets não têm uma aparência própria, mas são fundamentais para organizar e posicionar outros widgets na tela.

**Scaffold**

* **Visual:** É o esqueleto principal de uma página Material Design. Fornece uma estrutura padrão com App Bar, corpo (body), gaveta de navegação (drawer), etc.
* **Descrição:** Facilita a criação da estrutura visual básica de uma tela.
* **Principais Propriedades:**
  + appBar: A barra no topo da tela (geralmente um widget AppBar).
  + body: O conteúdo principal da tela.
  + floatingActionButton: O botão de ação flutuante.
  + drawer: O menu lateral que desliza a partir da borda.
  + bottomNavigationBar: A barra de navegação na parte inferior.

Dart

Scaffold(

  appBar: AppBar(

    title: Text("Minha Primeira Tela"),

  ),

  body: Center(

    child: Text("Olá, Mundo!"),

  ),

  floatingActionButton: FloatingActionButton(

    onPressed: () {},

    child: Icon(Icons.add),

  ),

)

**Container**

* **Visual:** Um "caixa" retangular que pode ser personalizada com cores, bordas, sombras, etc.
* **Descrição:** Um widget de conveniência que combina funcionalidades de pintura, posicionamento e dimensionamento.
* **Principais Propriedades:**
  + color: A cor de fundo do container.
  + width, height: As dimensões do container.
  + padding: Espaçamento interno.
  + margin: Espaçamento externo.
  + decoration: Para estilizações complexas como gradientes e bordas (BoxDecoration).
  + child: O widget que ficará dentro do container.

Dart

Container(

  width: 200,

  height: 100,

  padding: EdgeInsets.all(16.0),

  margin: EdgeInsets.all(20.0),

  decoration: BoxDecoration(

    color: Colors.blue,

    borderRadius: BorderRadius.circular(10),

    boxShadow: [

      BoxShadow(

        color: Colors.black.withOpacity(0.3),

        spreadRadius: 2,

        blurRadius: 5,

        offset: Offset(0, 3),

      ),

    ],

  ),

  child: Center(child: Text("Eu sou um Container", style: TextStyle(color: Colors.white))),

)

**Column e Row**

* **Visual:** Invisíveis, mas organizam seus filhos na vertical (Column) ou horizontal (Row).
* **Descrição:** Widgets de layout flexíveis e essenciais.
* **Principais Propriedades:**
  + children: A lista de widgets a serem organizados.
  + mainAxisAlignment: Como os filhos devem ser alinhados no eixo principal (vertical para Column, horizontal para Row). Ex: start, center, end, spaceBetween.
  + crossAxisAlignment: Como os filhos devem ser alinhados no eixo cruzado. Ex: start, center, end, stretch.

Dart

// Exemplo de Column

Column(

  mainAxisAlignment: MainAxisAlignment.center, // Alinha no centro do eixo Y

  children: <Widget>[

    Icon(Icons.star, size: 50),

    Text("Estrela Principal"),

  ],

)

// Exemplo de Row

Row(

  mainAxisAlignment: MainAxisAlignment.spaceAround, // Espaço igual entre eles

  children: <Widget>[

    Icon(Icons.favorite),

    Icon(Icons.thumb\_up),

    Icon(Icons.share),

  ],

)

**2. Widgets de Exibição de Conteúdo**

Estes widgets são usados para mostrar informações ao usuário.

**Text**

* **Visual:** Exibe uma string de texto.
* **Descrição:** O widget mais básico e essencial para mostrar texto.
* **Principais Propriedades:**
  + 'Seu Texto Aqui': O conteúdo textual.
  + style: Para customizar a aparência (TextStyle). Define cor, tamanho, peso da fonte, etc.
  + textAlign: Alinhamento do texto (center, left, right).

Dart

Text(

  'Flutter é incrível!',

  textAlign: TextAlign.center,

  style: TextStyle(

    color: Colors.deepPurple,

    fontSize: 24,

    fontWeight: FontWeight.bold,

    fontStyle: FontStyle.italic,

  ),

)

**Icon**

* **Visual:** Exibe um ícone gráfico.
* **Descrição:** Usa a fonte de ícones pré-definida do Material Design.
* **Principais Propriedades:**
  + Icons.nome\_do\_icone: O ícone a ser exibido.
  + size: O tamanho do ícone.
  + color: A cor do ícone.

Dart

Icon(

  Icons.home,

  size: 48.0,

  color: Colors.green,

)

**Image**

* **Visual:** Exibe uma imagem.
* **Descrição:** Pode carregar imagens de diversas fontes.
* **Principais Construtores:**
  + Image.asset('caminho/para/imagem.png'): Carrega uma imagem dos assets do seu projeto.
  + Image.network('URL\_DA\_IMAGEM'): Carrega uma imagem da internet.
  + Image.file(File('caminho/no/dispositivo.jpg')): Carrega uma imagem do armazenamento do dispositivo.

Dart

// Imagem da Internet

Image.network(

  'https://flutter.dev/images/flutter-logo-sharing.png',

  width: 150,

)

// Imagem local (requer configuração no pubspec.yaml)

// Image.asset('assets/images/meu-logo.png')

**Card**

* **Visual:** Um painel retangular com uma leve elevação (sombra) e cantos arredondados.
* **Descrição:** Usado para agrupar informações relacionadas de forma visualmente distinta.
* **Principais Propriedades:**
  + child: O conteúdo dentro do card.
  + elevation: A intensidade da sombra.
  + shape: A forma do card (ex: RoundedRectangleBorder).
  + margin: Espaçamento externo.

Dart

Card(

  elevation: 5,

  margin: EdgeInsets.all(10),

  shape: RoundedRectangleBorder(

    borderRadius: BorderRadius.circular(15.0),

  ),

  child: Padding(

    padding: const EdgeInsets.all(16.0),

    child: Column(

      mainAxisSize: MainAxisSize.min, // Para a coluna ocupar o mínimo de espaço

      children: <Widget>[

        ListTile(

          leading: Icon(Icons.album, size: 50),

          title: Text('Título do Card'),

          subtitle: Text('Este é um subtítulo de exemplo.'),

        ),

      ],

    ),

  ),

)

**3. Widgets de Entrada e Ação (Botões e Campos)**

Esses widgets permitem que o usuário interaja com o aplicativo.

**ElevatedButton**

* **Visual:** Um botão com fundo preenchido e uma leve sombra, que se eleva ao ser pressionado. É o botão padrão para ações principais.
* **Descrição:** Botão de destaque.
* **Principais Propriedades:**
  + onPressed: A função a ser executada quando o botão é pressionado. Se for null, o botão fica desabilitado.
  + child: O conteúdo do botão (geralmente um Text ou Icon).
  + style: Para customização avançada (ButtonStyle).

Dart

ElevatedButton(

  onPressed: () {

    print("Botão elevado pressionado!");

  },

  child: Text('Clique Aqui'),

  style: ElevatedButton.styleFrom(

    backgroundColor: Colors.red, // Cor de fundo

    foregroundColor: Colors.white, // Cor do texto/ícone

  ),

)

**TextButton**

* **Visual:** Um botão sem fundo visível, apenas o texto. Usado para ações secundárias ou menos importantes.
* **Descrição:** Botão discreto, geralmente usado em diálogos ou barras de ferramentas.
* **Principais Propriedades:**
  + onPressed: Função a ser executada.
  + child: O conteúdo do botão.

Dart

TextButton(

  onPressed: () {

    print("Botão de texto pressionado!");

  },

  child: Text('CANCELAR'),

)

**IconButton**

* **Visual:** Um botão que contém apenas um ícone, com um efeito de "tinta" que se espalha ao ser pressionado.
* **Descrição:** Ideal para ações em AppBar ou barras de ferramentas.
* **Principais Propriedades:**
  + onPressed: Função a ser executada.
  + icon: O widget Icon a ser exibido.
  + tooltip: Texto que aparece ao manter o botão pressionado (bom para acessibilidade).

Dart

IconButton(

  icon: Icon(Icons.settings),

  onPressed: () {

    print("Configurações");

  },

  tooltip: 'Abrir configurações',

)

**TextField**

* **Visual:** Um campo de entrada de texto.
* **Descrição:** Permite que o usuário digite texto usando o teclado.
* **Principais Propriedades:**
  + controller: Para controlar o texto do campo (TextEditingController).
  + decoration: A parte mais importante para a estilização (InputDecoration). Define rótulos, dicas, ícones, bordas, etc.
  + keyboardType: O tipo de teclado a ser exibido (numérico, email, etc.).
  + obscureText: Para esconder o texto (campos de senha).

Dart

TextField(

  decoration: InputDecoration(

    border: OutlineInputBorder(), // Borda ao redor do campo

    labelText: 'Nome de usuário', // Rótulo que sobe ao focar

    hintText: 'Digite seu nome de usuário', // Dica dentro do campo

    prefixIcon: Icon(Icons.person), // Ícone no início

  ),

)

**Checkbox e Switch**

* **Visual:** Checkbox é uma caixa de marcação, e Switch é um interruptor de ligar/desligar.
* **Descrição:** Usados para opções binárias (verdadeiro/falso).
* **Principais Propriedades:**
  + value: O estado atual (um bool).
  + onChanged: Função chamada quando o valor muda. Você **precisa** usar setState aqui para atualizar a UI.

Dart

// Exemplo dentro de um StatefulWidget

bool \_isChecked = false;

// ... no método build ...

CheckboxListTile( // Uma versão que já vem com texto

  title: Text("Aceito os termos de uso"),

  value: \_isChecked,

  onChanged: (bool? newValue) {

    setState(() {

      \_isChecked = newValue!;

    });

  },

  controlAffinity: ListTileControlAffinity.leading, // Checkbox na frente do texto

)

// Switch

Switch(

  value: \_isChecked,

  onChanged: (bool newValue) {

    setState(() {

      \_isChecked = newValue;

    });

  },

)

**Exemplo Prático: Tela de Login Simples**

Vamos juntar vários desses widgets para criar uma tela de login básica.

Dart

import 'package:flutter/material.dart';

class SimpleLoginScreen extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text("Login"),

        backgroundColor: Colors.deepPurple,

      ),

      body: Padding(

        padding: const EdgeInsets.all(20.0), // Espaçamento em toda a tela

        child: Column(

          mainAxisAlignment: MainAxisAlignment.center, // Centralizar verticalmente

          crossAxisAlignment: CrossAxisAlignment.stretch, // Esticar horizontalmente

          children: <Widget>[

            // Ícone de cadeado no topo

            Icon(Icons.lock, size: 80, color: Colors.deepPurple),

            SizedBox(height: 30), // Espaçamento vertical

            // Campo de Email

            TextField(

              keyboardType: TextInputType.emailAddress,

              decoration: InputDecoration(

                labelText: 'Email',

                border: OutlineInputBorder(),

                prefixIcon: Icon(Icons.email),

              ),

            ),

            SizedBox(height: 20), // Espaçamento vertical

            // Campo de Senha

            TextField(

              obscureText: true, // Esconde o texto da senha

              decoration: InputDecoration(

                labelText: 'Senha',

                border: OutlineInputBorder(),

                prefixIcon: Icon(Icons.vpn\_key),

              ),

            ),

            SizedBox(height: 30), // Espaçamento vertical

            // Botão de Entrar

            ElevatedButton(

              onPressed: () {

                // Lógica de login aqui

              },

              child: Padding(

                padding: const EdgeInsets.symmetric(vertical: 15.0),

                child: Text('ENTRAR', style: TextStyle(fontSize: 16)),

              ),

              style: ElevatedButton.styleFrom(

                backgroundColor: Colors.deepPurple,

                shape: RoundedRectangleBorder(

                  borderRadius: BorderRadius.circular(12),

                ),

              ),

            ),

            SizedBox(height: 10),

            // Botão de "Esqueci a senha"

            TextButton(

              onPressed: () {},

              child: Text('Esqueci minha senha'),

            ),

          ],

        ),

      ),

    );

  }

}

**Próximos Passos**

Este guia cobriu os widgets visuais mais fundamentais. O ecossistema Flutter é vasto. Explore outros widgets como:

* **ListView e GridView:** Para listas e grades roláveis.
* **Stack:** Para sobrepor widgets.
* **AlertDialog:** Para exibir caixas de diálogo.
* **BottomNavigationBar:** Para navegação principal em apps.

A melhor maneira de aprender é praticar. Tente recriar telas de aplicativos que você já usa e consulte a [documentação oficial do Flutter](https://flutter.dev/docs) e o [Catálogo de Widgets](https://docs.flutter.dev/ui/widgets) sempre que tiver dúvidas. Boa codificação!