

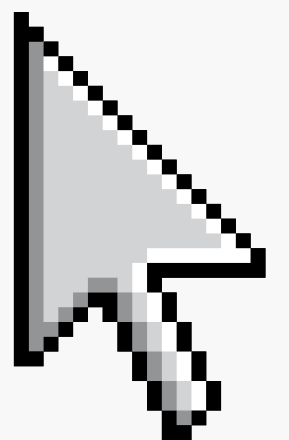
# Elementos básicos de site

Curso Flutter de Verão



- Criar um menu
- Definir os assets
- Soluções prontas

# Objetivo da aula de hoje

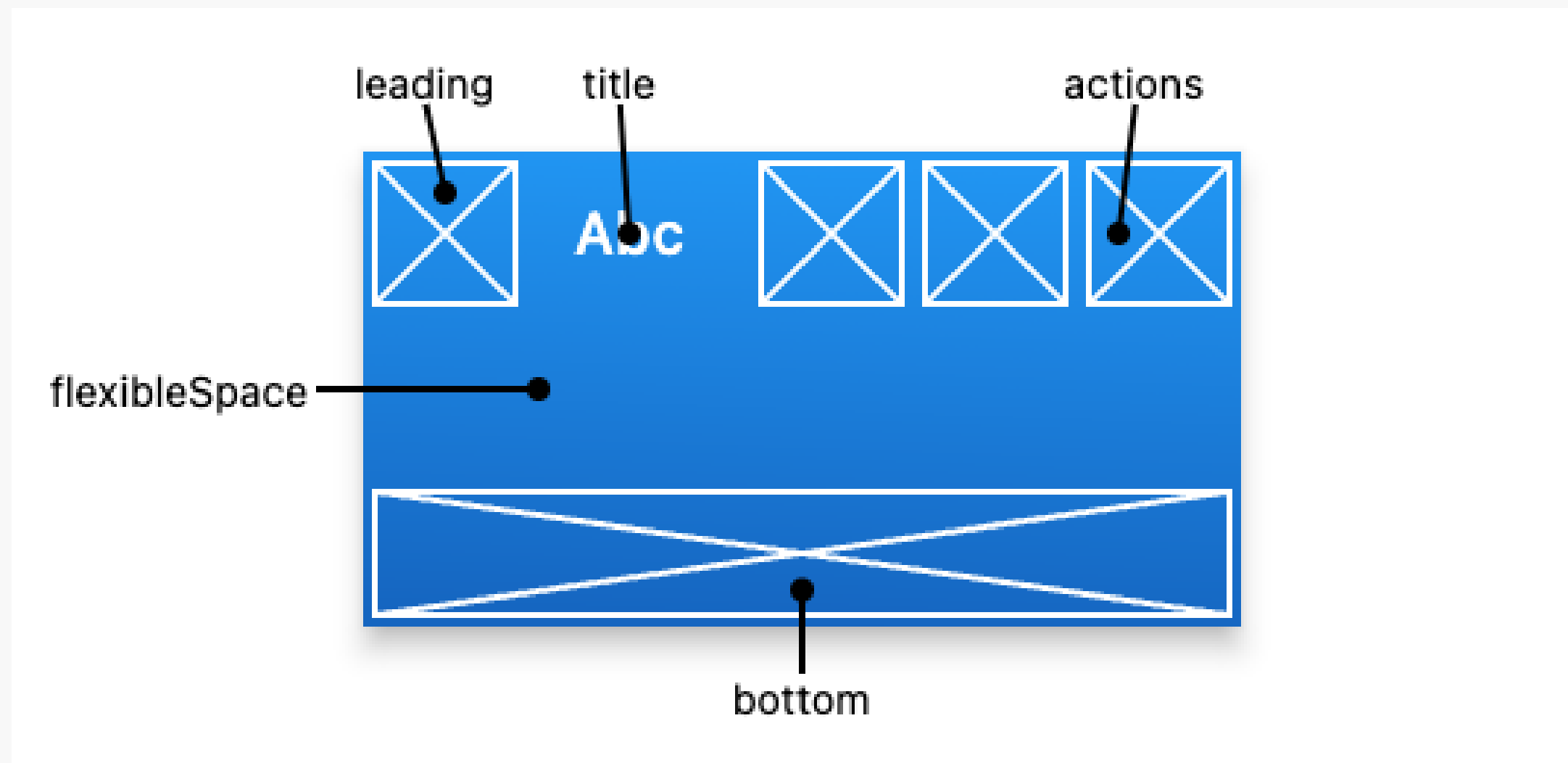


# Recursos de menu

Para criar menus em aplicativos Flutter, você pode aproveitar os widgets padrão do Flutter, como AppBar, AppBarSliver, Drawer, MenuBar ou até Columns e Rows para criar diferentes tipos de menus, como menus de navegação, menus laterais, menus suspensos e muito mais. Você pode personalizar esses widgets para atender às suas necessidades de design.

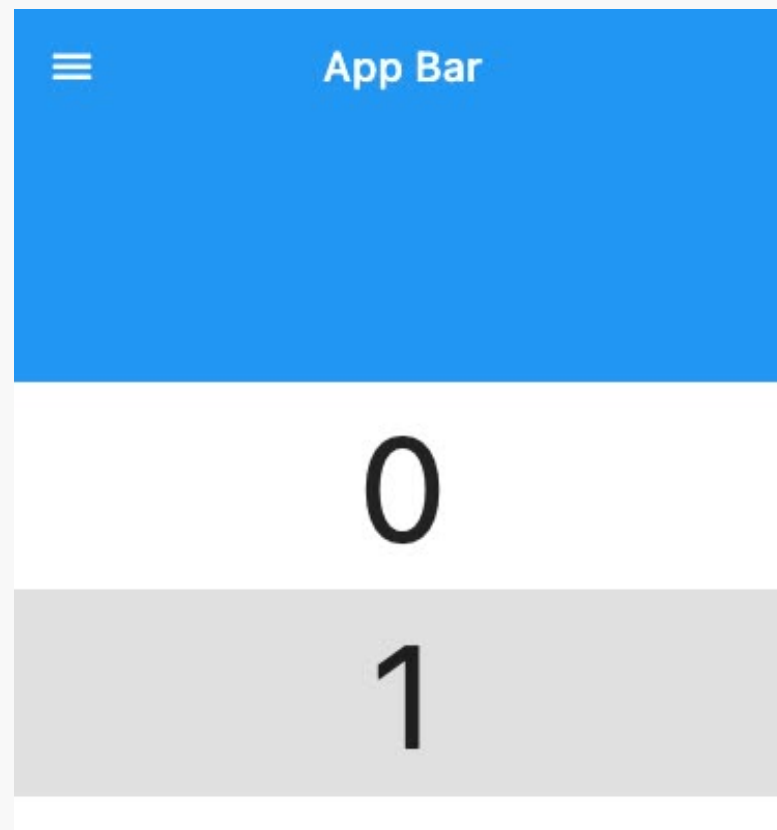
# AppBar

Uma barra de aplicativos no Material Design engloba uma barra de ferramentas e possivelmente outros elementos como TabBar e FlexibleSpaceBar. Essa barra exibe ações comuns por meio de IconButton e, às vezes, usa um PopupMenuButton para ações menos frequentes. Geralmente, é integrada na propriedade "Scaffold.appBar", mantendo-se fixa no topo.



# SliverAppBar

A classe SliverAppBar no Material Design é uma barra de aplicativos que se integra a um CustomScrollView. Ela contém uma barra de ferramentas, TabBar e FlexibleSpaceBar, exibindo ações comuns por meio de IconButton e ações menos frequentes com PopupMenuButton. Essa barra é usada como o primeiro elemento de um CustomScrollView, permitindo que sua altura varie com a rolagem ou flutue sobre o conteúdo. Para uma barra de aplicativos fixa na parte superior, a classe AppBar é usada. Ela exibe widgets de barra de ferramentas, "leading", título e ações acima da parte inferior, com opção de um FlexibleSpace widget por trás dela.



```
CustomScrollView(
  physics: const BouncingScrollPhysics(),
  slivers: <Widget>[
    const SliverAppBar(
      snap: true,
      floating: true,
      expandedHeight: 150.0,
      flexibleSpace: FlexibleSpaceBar(
        title: Text('SliverAppBar'),
        background: FlutterLogo(),
      ),
    ),
    SliverList(
      delegate: SliverChildBuilderDelegate((BuildContext context, int
index) {
        return Container(
          color: index.isOdd ? Colors.white : Colors.black12,
          height: 100.0,
          child: Center(
            child: Text('$index', textScaleFactor: 5),
          ),
        );
      },
      childCount: 20,
    ),
  ],
),
```

# MenuBar

Uma barra de menus do Material Design gerencia menus em cascata, ficando acima do conteúdo principal do aplicativo. Ela permite a seleção de itens de menu para invocar ações. Os menus podem ser abertos por clique ou toque e navegados por teclas de seta ou tab, fechando com Escape ou cliques externos. Itens de menu com submenus podem ser abertos passando o mouse sobre eles. Atalhos podem ser atribuídos a itens de menu através de SingleActivator ou CharacterActivator. O uso de ShortcutRegistry é recomendado para atalhos globais.



```
MenuBar(  
  controller: menuController,  
  children: [  
    MenuItemButton(  
      onPressed: () => GoRouter.of(context).go('/'),  
      child: const Text("Tela Inicial"),  
    ),  
    MenuItemButton(  
      onPressed: () => GoRouter.of(context).go('/projetos'),  
      child: const Text("Projetos"),  
    ),  
    MenuItemButton(  
      onPressed: () => GoRouter.of(context).go('/conhecimentos'),  
      child: const Text("Habilidades e Conhecimentos"),  
    ),  
    MenuItemButton(  
      onPressed: () => GoRouter.of(context).go('/informacoes'),  
      child: const Text("Informações Acadêmicas"),  
    ),  
    MenuItemButton(  
      onPressed: () => GoRouter.of(context).go('/sobre'),  
      child: const Text("Sobre"),  
    ),  
  ],  
)
```



# Drawer

Em aplicativos que usam Material Design, há duas opções principais de navegação: guias e gavetas. Quando não há espaço suficiente para suportar abas, as gavetas são uma alternativa prática.

No Flutter, use o **Drawer** em combinação com um Scaffold para criar um layout com uma gaveta do Material Design.

Siga as seguintes etapas:

1. Crie um Scaffold.
2. Adicione uma Drawer.
3. Preencha a Drawer com itens.
4. Feche a Drawer programaticamente.

```
Drawer(  
  child: ListView(  
    padding: EdgeInsets.zero,  
    children: [  
      const DrawerHeader(  
        decoration: BoxDecoration(  
          color: Colors.blue,  
        ),  
        child: Text('Drawer Header'),  
      ),  
      ListTile(  
        title: const Text('Item 1'),  
        onTap: () {},  
      ),  
      ListTile(  
        title: const Text('Item 2'),  
        onTap: () {},  
      ),  
    ],  
  ),  
);
```

# Assets

Aplicativos Flutter podem incluir tanto código quanto assets (às vezes chamados de ativos ou recursos). Um asset é um arquivo que é empacotado e implantado com o seu aplicativo, sendo acessível em tempo de execução. Tipos comuns de ativos incluem dados estáticos (como arquivos JSON), arquivos de configuração, ícones e imagens (JPEG, WebP, GIF, WebP/GIF animados, PNG, BMP e WBMP).

O Flutter utiliza o arquivo `pubspec.yaml`, localizado na raiz do seu projeto, para identificar os recursos necessários pelo aplicativo.

# Carregar assets

Seu aplicativo pode acessar seus ativos por meio de um objeto `AssetBundle`.

Os dois principais métodos de um pacote de ativos permitem que você carregue um ativo de string/texto (`loadString()`) ou um ativo de imagem/binário (`load()`) do pacote, fornecendo uma chave lógica. A chave lógica mapeia para o caminho do ativo especificado no arquivo `pubspec.yaml` durante a compilação.

# Textos


É recomendado obter o `AssetBundle` para o `BuildContext` atual usando `DefaultAssetBundle`, em vez do pacote de ativos padrão construído com o aplicativo; essa abordagem permite que um widget pai substitua um `AssetBundle` diferente em tempo de execução, o que pode ser útil para localização ou cenários de teste. Normalmente, você usará `DefaultAssetBundle.of()` para carregar indiretamente um ativo, como um arquivo JSON, do pacote de ativos em tempo de execução do aplicativo.

```
Future<String> loadAsset() async {  
  return await rootBundle.loadString('assets/config.json');  
}
```

# Imagens

Para carregar uma imagem, utilize a classe AssetImage no método build() de um widget.

Por exemplo, seu aplicativo pode carregar a imagem de fundo a partir das declarações de ativos no exemplo anterior.

A code editor window with a dark background and a light blue border. It contains a single line of Dart code: `return const Image(image: AssetImage('assets/background.png'));`. The text is color-coded: `return` is purple, `const` is pink, `Image` is light blue, `(image:` is cyan, `AssetImage` is light blue, `('assets/background.png')` is orange, and `);` is cyan. In the top right corner of the editor, there are three small, light gray icons: a minus sign, a square, and an 'X'.

# Bibliotecas externas

O uso de bibliotecas externas no Flutter e Dart é facilitado pelo gerenciador de pacotes, com o site `pub.dev` como repositório oficial. Ao adicionar dependências no arquivo `'pubspec.yaml'` e executar `'flutter packages get'`, as funcionalidades são integradas. No entanto, é vital escolher bibliotecas bem mantidas e revisar a documentação para garantir compatibilidade e qualidade. Isso acelera o desenvolvimento e expande as capacidades dos projetos.



# Visite o site pub.dev

Experimente as seguintes libs:

<https://pub.dev/packages/http> [https://pub.dev/packages/firebase\\_core](https://pub.dev/packages/firebase_core)

[https://pub.dev/packages/firebase\\_storage](https://pub.dev/packages/firebase_storage)

[https://pub.dev/packages/carousel\\_slider](https://pub.dev/packages/carousel_slider) [https://pub.dev/packages/flutter\\_svg](https://pub.dev/packages/flutter_svg)

[https://pub.dev/packages/cached\\_network\\_image](https://pub.dev/packages/cached_network_image)

[https://pub.dev/packages/image\\_picker\\_for\\_web](https://pub.dev/packages/image_picker_for_web)

[https://pub.dev/packages/font\\_awesome\\_flutter](https://pub.dev/packages/font_awesome_flutter)

[https://pub.dev/packages/google\\_fonts](https://pub.dev/packages/google_fonts) [https://pub.dev/packages/auto\\_size\\_text](https://pub.dev/packages/auto_size_text)

[https://pub.dev/packages/animated\\_text\\_kit](https://pub.dev/packages/animated_text_kit)

[https://pub.dev/packages/just\\_audio](https://pub.dev/packages/just_audio) <https://pub.dev/packages/pdf>

[https://pub.dev/packages/infinite\\_scroll\\_pagination](https://pub.dev/packages/infinite_scroll_pagination)

<https://pub.dev/packages/rive> [https://pub.dev/packages/fl\\_chart](https://pub.dev/packages/fl_chart)

**Como anda nosso portfólio?**

