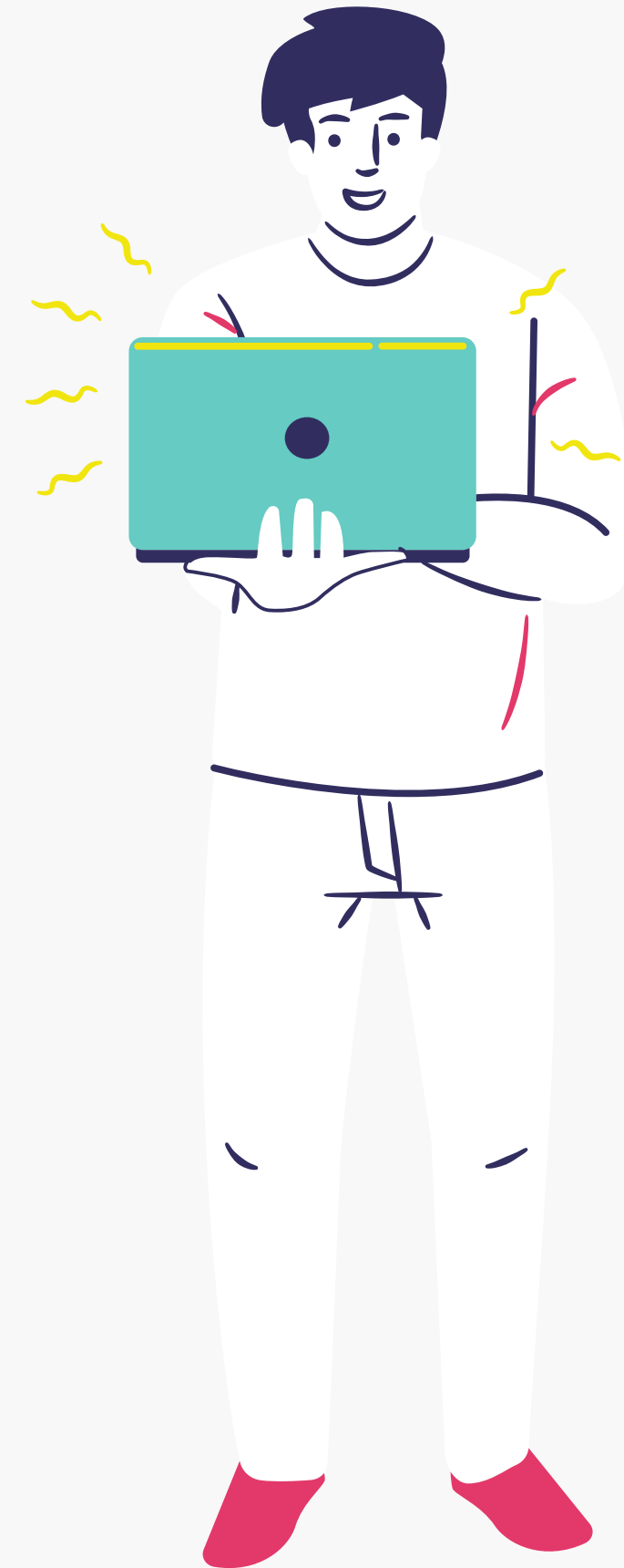


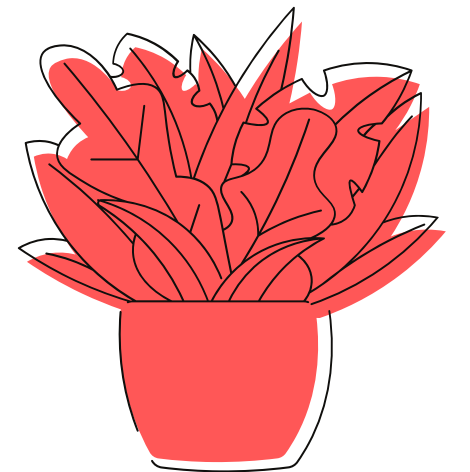
Flutter

Curso Flutter de Verão



- Navegação entre telas
- push, pop e retorno
- Componentes de entrada

Objetivo da aula de hoje



Navegação entre telas

Navigator

refere-se a um componente que gerencia a navegação entre diferentes telas ou rotas em um aplicativo. O conceito é semelhante ao de uma pilha de páginas ou páginas empilhadas, onde você pode adicionar novas páginas (rotas) à pilha e remover ou voltar para páginas anteriores.

Push

É uma operação na pilha de navegação que adiciona uma nova rota à pilha. Quando você realiza o push para uma nova rota, o Flutter empilha a rota mais recente no topo, e a interface do usuário transita para a nova tela. Essa nova rota torna-se a rota ativa, e a rota anterior ainda existe na pilha, mas fica oculta.



```
Navigator.push(  
  context,  
  MaterialPageRoute(  
    builder: (context) => NewScreen(),  
  ),  
);
```

Pop

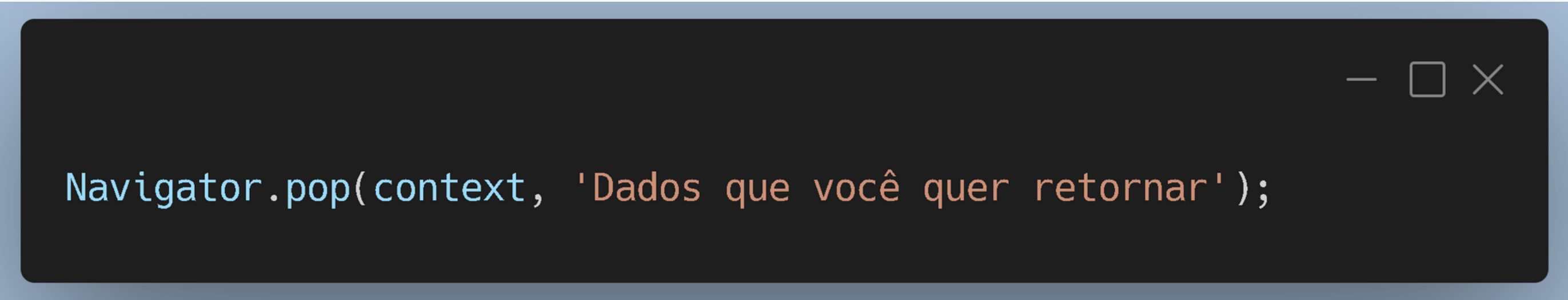
É uma operação na pilha de navegação que remove a rota atual da pilha e retorna à rota anterior, tornando-a a rota ativa novamente. É como pressionar o botão "Voltar" em um aplicativo, onde você retorna à tela anterior na pilha.



```
Navigator.pop(context);
```


Retorno de dados

Você também pode usar o método pop para retornar dados da tela de destino para a tela anterior. Isso é útil quando você precisa passar informações de volta para a tela anterior após a conclusão de alguma ação.



```
Navigator.pop(context, 'Dados que você quer retornar');
```

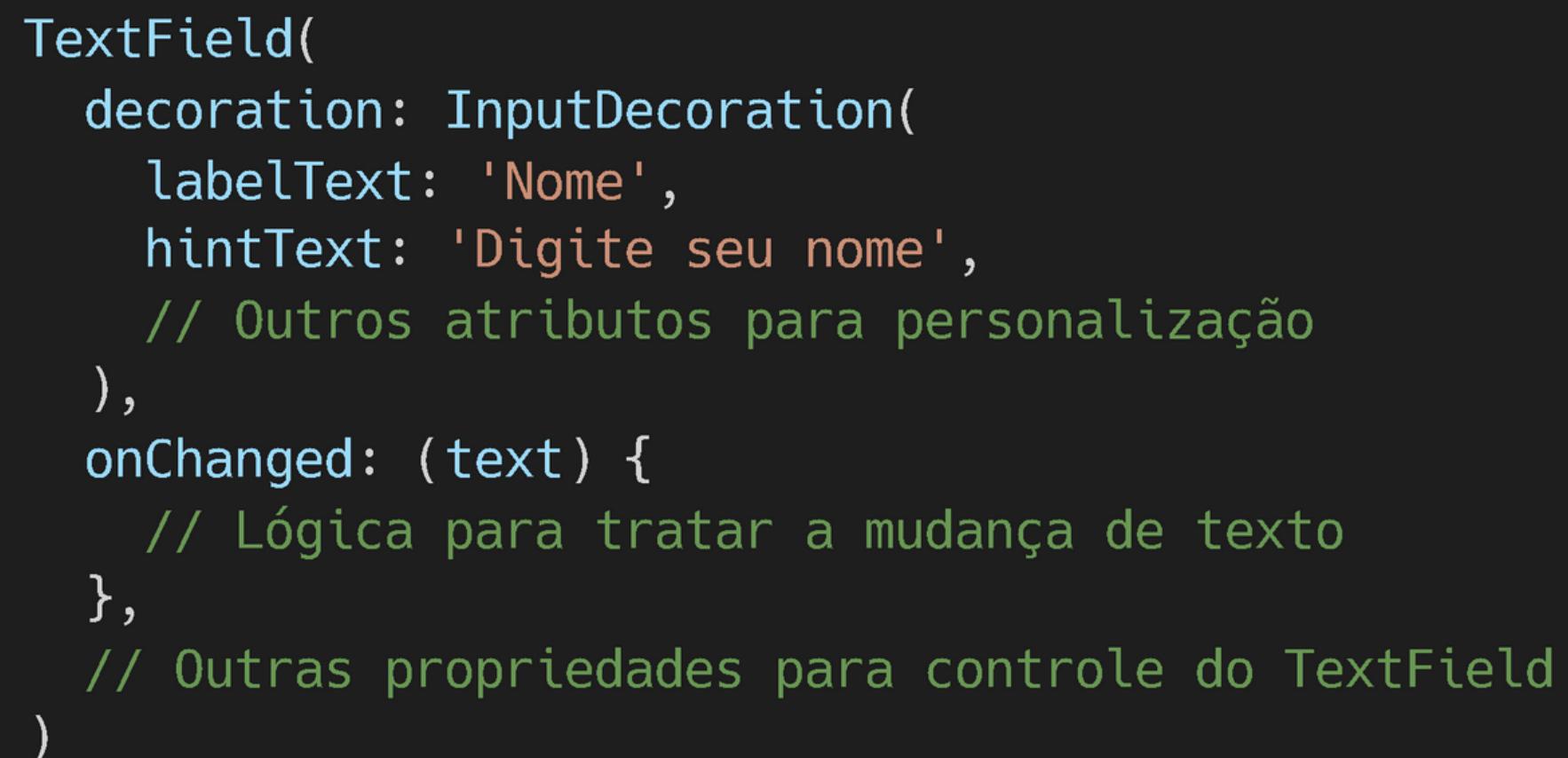
showDialog

`showDialog` é uma função global em Flutter que exibe um modal ou caixa de diálogo na parte superior da pilha de navegação atual. Essa caixa de diálogo é geralmente usada para comunicar informações importantes ou solicitar uma ação ao usuário antes de continuar.

AlertDialog

AlertDialog é um widget que define o conteúdo e a aparência de um diálogo. Geralmente, é construído dentro do método builder do showDialog. O AlertDialog é altamente personalizável e pode conter título, conteúdo e botões de ação.

Componentes de entrada



```
TextField(  
  decoration: InputDecoration(  
    labelText: 'Nome',  
    hintText: 'Digite seu nome',  
    // Outros atributos para personalização  
  ),  
  onChanged: (text) {  
    // Lógica para tratar a mudança de texto  
  },  
  // Outras propriedades para controle do TextField  
)
```

```
int selectedRadio = 0; // Variável para armazenar a opção  
selecionada
```

```
RadioListTile(  
  title: Text('Opção 1'),  
  value: 1,  
  groupValue: selectedRadio,  
  onChanged: (value) {  
    // Lógica para atualizar a seleção  
    setState(() {  
      selectedRadio = value;  
    });  
  },  
);
```

```
RadioListTile(  
  title: Text('Opção 2'),  
  value: 2,  
  groupValue: selectedRadio,  
  onChanged: (value) {  
    // Lógica para atualizar a seleção  
    setState(() {  
      selectedRadio = value;  
    });  
  },  
);
```

```
bool checkboxValue1 = false;  
bool checkboxValue2 = false;
```

```
Checkbox(  
  value: checkboxValue1,  
  onChanged: (value) {  
    // Lógica para atualizar o estado do checkboxValue1  
    setState(() {  
      checkboxValue1 = value;  
    });  
  },  
);
```

```
Checkbox(  
  value: checkboxValue2,  
  onChanged: (value) {  
    // Lógica para atualizar o estado do checkboxValue2  
    setState(() {  
      checkboxValue2 = value;  
    });  
  },  
);
```




```
bool switchValue = false;
```

```
Switch(  
  value: switchValue,  
  onChanged: (value) {  
    // Lógica para atualizar o estado do switchValue  
    setState(() {  
      switchValue = value;  
    });  
  },  
);
```

```
String dropdownValue = 'Opção 1'; // Valor inicial selecionado
```

```
DropDownButton<String>(  
  value: dropdownValue,  
  onChanged: (newValue) {  
    // Lógica para atualizar o valor selecionado  
    setState(() {  
      dropdownValue = newValue;  
    });  
  },  
  items: <String>[  
    'Opção 1',  
    'Opção 2',  
    'Opção 3',  
    // Outras opções...  
  ].map<DropDownMenuItem<String>>((String value) {  
    return DropDownMenuItem<String>(  
      value: value,  
      child: Text(value),  
    );  
  }).toList(),  
);
```

Exercício

Utilizando o código a aula anterior: faça um campo de entrada para o usuário buscar os pratos pelo nome.



Exercício



Transforme o seu container com a apresentação dos pratos em botões através do InkWell.

Direciona o usuário para uma nova tela que apresentará mais informações sobre o prato selecionado.

Ao retorna para a tela principal, utilize o `showDialog` para confirmar a ação do usuário.