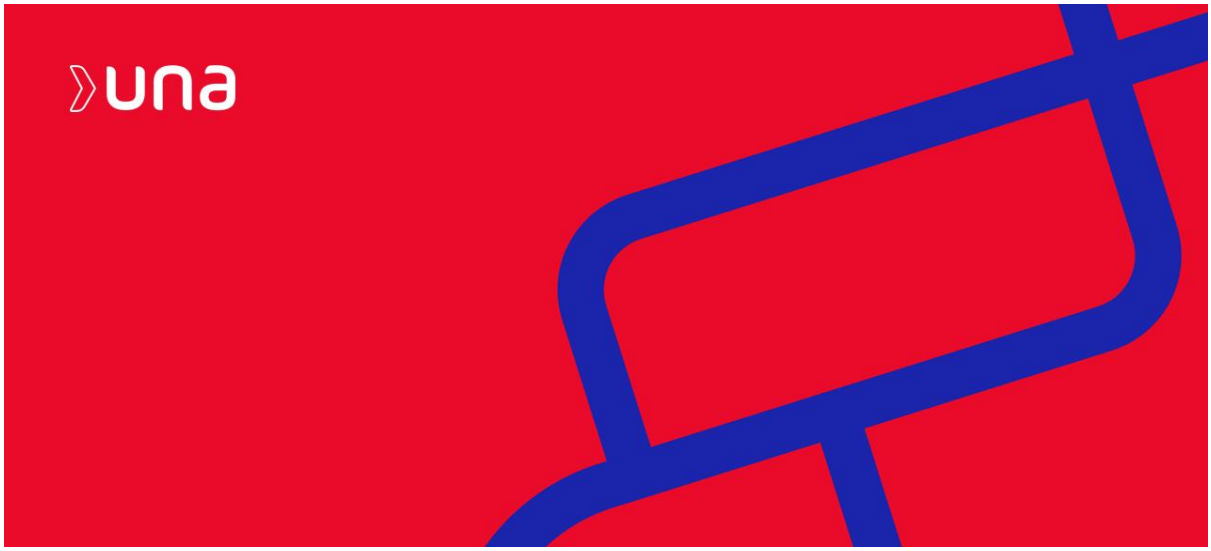


Flutter

Wesley Dias Maciel

2021/02

Centro Universitário UNA
Flutter
Wesley Dias Maciel
2021/02

Prática 13

Navegação com Refatoração de Telas

Documentação: https://flutter.dev/docs/cookbook/navigation/named-routes, https://api.flutter.dev/flutter/widgets/Navigator-class.html

Objetivo: apresentar o processo de navegação através de rotas nomeadas com refatoração de código das telas.

Na prática anterior, você aprendeu como navegar para uma nova tela através de rotas nomeadas, usando o método Navigator.pushNamed (). Nesta prática, aplicaremos navegação com rotas nomeadas, demonstrando a refatoração de código das telas.

1) Crie um novo projeto Flutter, usando:
    a. Visual Studio Code, ou;
    b. https://dartpad.dev/, ou;
    c. https://flutlab.io/, ou;
    d. https://flutterstudio.app/, ou;
    e. https://codemagic.io/.

Criar as Rotas

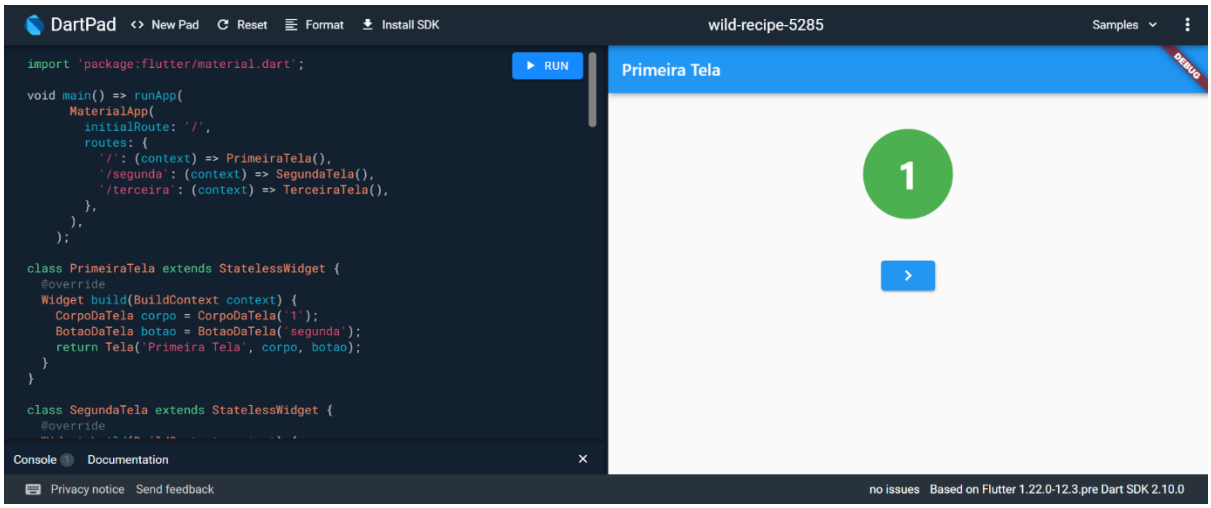
2) O exemplo abaixo cria as rotas. Cada rota é um widget.

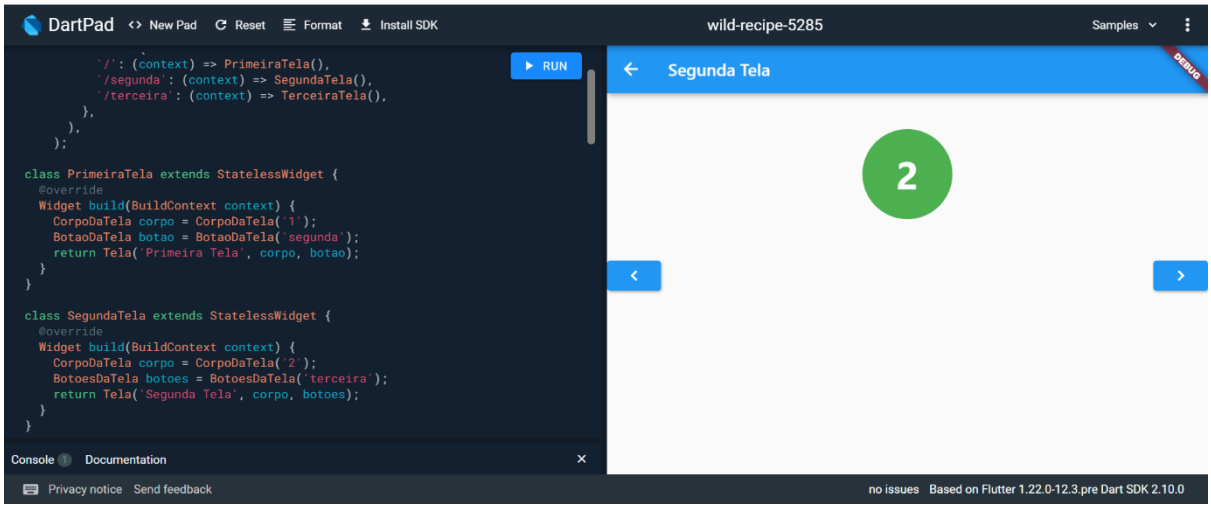
```
import 'package:flutter/material.dart';

void main() => runApp(
      MaterialApp(
        initialRoute: '/',
        routes: {
          '/': (context) => PrimeiraTela(),
          '/segunda': (context) => SegundaTela(),
          '/terceira': (context) => TerceiraTela(),
        },
      ),
    );

class PrimeiraTela extends StatelessWidget {
```

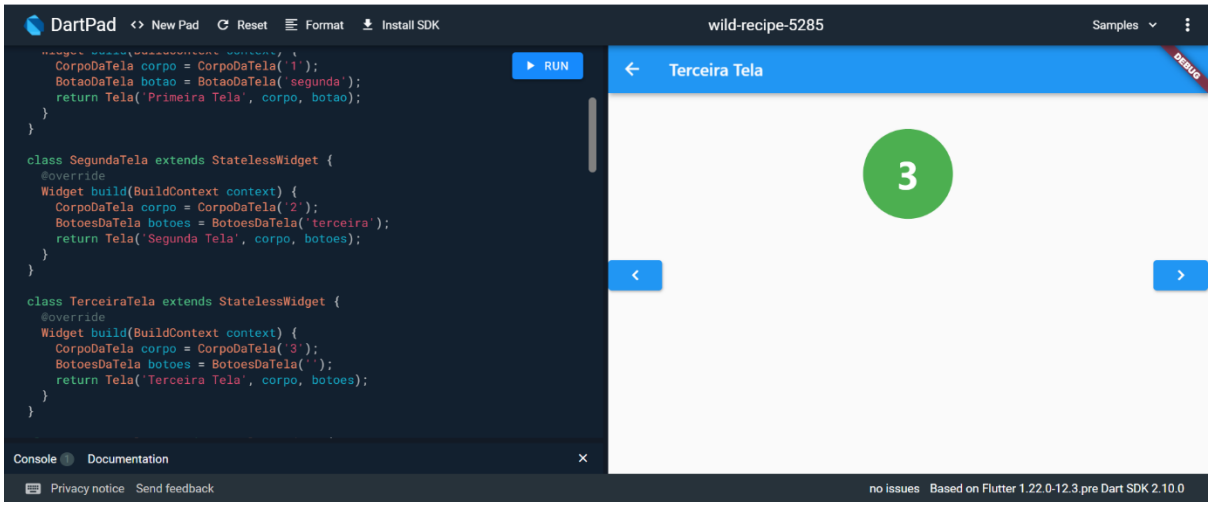
```
  @override
  Widget build(BuildContext context) {
    Corpo corpo = Corpo('1');
    Botao botao = Botao('segunda');
    return Tela('Primeira Tela', corpo, botao);
  }
}

class SegundaTela extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    Corpo corpo = Corpo('2');
    Botoes botoes = Botoes('terceira');
    return Tela('Segunda Tela', corpo, botoes);
  }
}

class TerceiraTela extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    Corpo corpo = Corpo('3');
    Botoes botoes = Botoes('');
    return Tela('Terceira Tela', corpo, botoes);
  }
}

class Corpo extends StatelessWidget {
  final String texto;

  Corpo(this.texto);

  @override
  Widget build(BuildContext context) {
    return Container(
      child: Text(
        '${this.texto}',
        style: TextStyle(
          fontSize: 45,
          fontWeight: FontWeight.bold,
          color: Colors.white,
        ),
      ),
      decoration: BoxDecoration(
        shape: BoxShape.circle,
        color: Colors.green,
      ),
      padding: EdgeInsets.all(40),
```

```
      margin: EdgeInsets.all(25),
    );
  }
}

class Botao extends StatelessWidget {
  final String proxima;

  Botao(this.proxima);

  @override
  Widget build(BuildContext context) {
    return ElevatedButton(
      child: Icon(Icons.navigate_next),
      onPressed: () {
        Navigator.pushNamed(context, '/${this.proxima}');
      },
    );
  }
}

class Botoes extends StatelessWidget {
  final String proxima;

  Botoes(this.proxima);

  @override
  Widget build(BuildContext context) {
    return Row(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: [
        ElevatedButton(
          child: Icon(Icons.navigate_before),
          onPressed: () {
            Navigator.pop(context);
          },
        ),
        ElevatedButton(
          child: Icon(Icons.navigate_next),
          onPressed: () {
            Navigator.pushNamed(context, '/${this.proxima}');
          },
        ),
      ],
    );
  }
}
```

```dart
class Tela extends StatelessWidget {
  final String titulo;
  final Widget corpo, navegacao;

  Tela(this.titulo, this.corpo, this.navegacao);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('${this.titulo}'),
      ),
      body: Center(
        child: Column(
          children: [
            corpo,
            navegacao,
          ],
        ),
      ),
    );
  }
}
```







Telas

3) O algoritmo apresenta as classes de três telas. A primeira tela é formada pelos widgets Corpo e Botao. A segunda e a terceira telas são formadas pelos widgets Corpo e Botoes.

```
class PrimeiraTela extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    Corpo corpo = Corpo('1');
    Botao botao = Botao('segunda');
    return Tela('Primeira Tela', corpo, botao);
  }
}

class SegundaTela extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
```

```
    Corpo corpo = Corpo('2');
    Botoes botoes = Botoes('terceira');
    return Tela('Segunda Tela', corpo, botoes);
  }
}

class TerceiraTela extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    Corpo corpo = Corpo('3');
    Botoes botoes = Botoes('');
    return Tela('Terceira Tela', corpo, botoes);
  }
}
```

Corpo

4) O widget Corpo gera a informação apresentada para o usuário.

```
class Corpo extends StatelessWidget {
  final String texto;

  Corpo(this.texto);

  @override
  Widget build(BuildContext context) {
    return Container(
      child: Text(
        '${this.texto}',
        style: TextStyle(
          fontSize: 45,
          fontWeight: FontWeight.bold,
          color: Colors.white,
        ),
      ),
      decoration: BoxDecoration(
        shape: BoxShape.circle,
        color: Colors.green,
      ),
      padding: EdgeInsets.all(40),
      margin: EdgeInsets.all(25),
    );
  }
}
```

A classe do widget Corpo possui um construtor. O construtor é um método especial que é executado sempre que um objeto da classe é instanciado. Por isso, métodos construtores podem ser usados para iniciar atributos da classe. Um construtor sempre tem o mesmo nome da classe.

```
Corpo(this.texto);
```

Botao

5) O widget Botão gera o botão da primeira tela.

```
class Botao extends StatelessWidget {
  final String proxima;

  Botao(this.proxima);

  @override
  Widget build(BuildContext context) {
    return ElevatedButton(
      child: Icon(Icons.navigate_next),
      onPressed: () {
        Navigator.pushNamed(context, '/${this.proxima}');
      },
    );
  }
}
```

O widget Botão também possui um construtor:

```
  Botao(this.proxima);
```

Botoes

6) O widget Botoes gera os botões da segunda e terceira telas.

```
class Botoes extends StatelessWidget {
  final String proxima;
```

```
  Botoes(this.proxima);

  @override
  Widget build(BuildContext context) {
    return Row(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: [
        ElevatedButton(
          child: Icon(Icons.navigate_before),
          onPressed: () {
            Navigator.pop(context);
          },
        ),
        ElevatedButton(
          child: Icon(Icons.navigate_next),
          onPressed: () {
            Navigator.pushNamed(context, '/${this.proxima}');
          },
        ),
      ],
    );
  }
}
```

Construtor do widget Botoes:

```
  Botoes(this.proxima);
```

Tela

7) O widget Tela recebe o título, corpo e botões de navegação em seu construtor e gera a tela.

```
class Tela extends StatelessWidget {
  final String titulo;
  final Widget corpo, navegacao;

  Tela(this.titulo, this.corpo, this.navegacao);

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('${this.titulo}'),
```

```
      ),
      body: Center(
        child: Column(
          children: [
            corpo,
            navegacao,
          ],
        ),
      ),
    );
  }
}
```

Construtor do widget Tela:

```
  Tela(this.titulo, this.corpo, this.navegacao);
```

Exercício

1) Altere o algoritmo apresentado nesta prática para que ele implemente a transição de telas apresentada na figura abaixo. Caso possível, apresente uma nova forma de refatorar o código.

Centro Universitário UNA
Flutter
Wesley Dias Maciel
2021/02