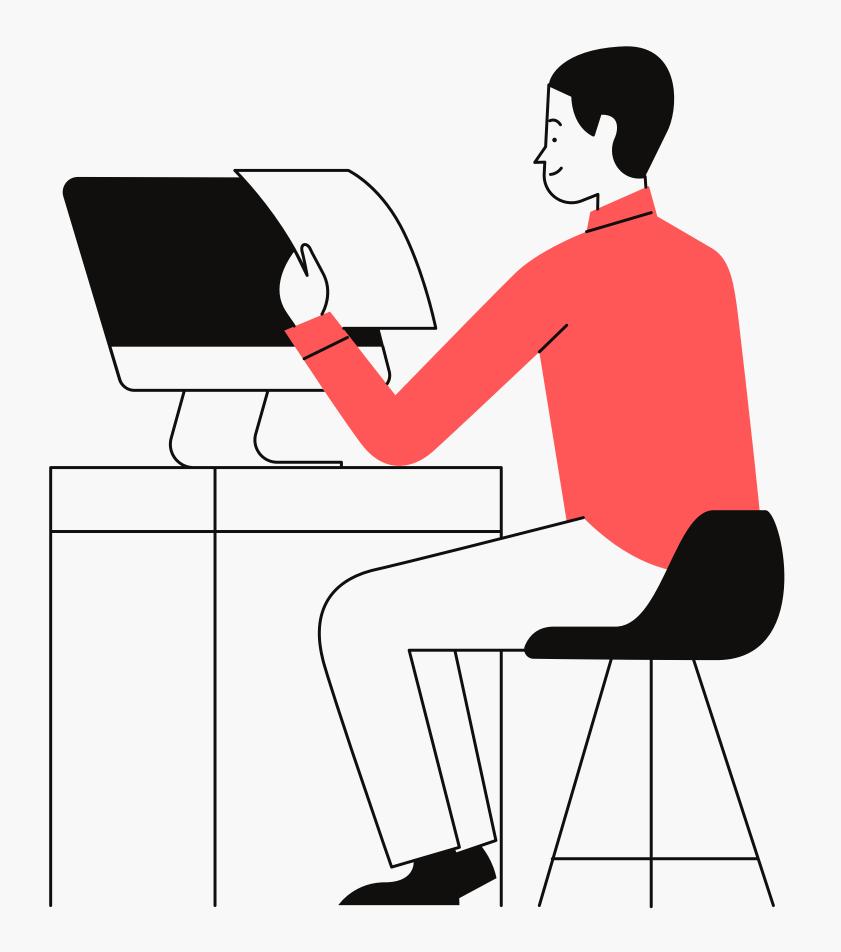
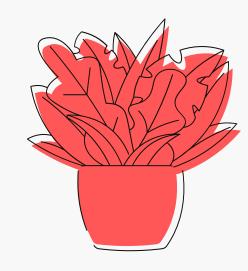
Estados

Curso Flutter de Verão



- Estados estáticos
- Estados dinâmicos

Objetivo da aula de hoje



Componente estático

Não pode ser modificado após ser construído

Podem ser usados para reduzir um componente com várias linhas de código

Podem ser usados para reutilizar outros componentes e estilos

```
class Componente extends StatelessWidget {
  const Componente({super.key});
 @override
  Widget build(BuildContext context) {
    return Text('Olá, mundo!');
```

BuildContext

Representa o contexto atual da renderização de um componente

Contém informações sobre a árvore de componentes e cada posição dos seus nós

Utilizado para acessar temas, estilos e outros recursos do MaterialApp

```
class Componente extends StatelessWidget {
 final int valor;
 const Componente({
    super.key,
    required this.valor,
  });
 @override
 Widget build(BuildContext context) {
    return Text('Olá, mundo!');
```

Componente dinâmico

Pode ser atualizado dinamicamente ao longo do tempo

Possuem um estado interno

Precisam ser atualizados em resposta a interações do usuário ou a mudanças nos dados atuais

Como converter?

```
class Tela extends StatelessWidget {
 const Tela({super.key});
 @override
 Widget build(BuildContext context) {
    return Text('0lá!');
```

Como converter?

```
class Tela extends StatefulWidget {
 const Tela({super.key});
 @override
 State<Tela> createState() => _EstadoTela();
class _EstadoTela extends State<Tela> {
 @override
 Widget build(BuildContext context) {
    return Text('0lá!');
```

Como converter? (c/ parâmetro)

```
class Tela extends StatelessWidget {
 final String nome;
 const Tela({super.key, required this.nome});
 @override
 Widget build(BuildContext context) {
    return Text(nome);
```

Como converter? (c/ parâmetro)

```
class Tela extends StatefulWidget {
  final String nome;
  const Tela({super.key, required this.nome});
 @override
  State<Tela> createState() => _EstadoTela();
class _EstadoTela extends State<Tela> {
  @override
  Widget build(BuildContext context) {
    return Text(widget.nome);
```

setState

Serve para chamar novamente a função build sem entrar em loop infinito

Pode apenas ser chamada dentro de um State

Deve conter dentro da chamada a atualização a ser realizada no estado

13

```
class _EstadoTela extends State<Tela> {
  int count = 0;
  @override
  Widget build(BuildContext context) {
    return Column(
      children: [
        Text('$count'),
        TextButton(
          child: Text('adicionar'),
          onPressed: () {
            setState(() {
              count += 1;
            });
          },
        ),
      ],
```

initState

Serve para executar código apenas **uma vez** durante a criação do componente

Não deve ser chamada, apenas sobrescrita

widget não pode ser acessado antes dessa função ser chamada

15 ----

```
class _EstadoTela extends State<Tela> {
 late int valor;
 @override
 void initState() {
   super.initState();
   valor = widget.valorInicial;
 @override
 Widget build(BuildContext context) {
   return Column(
     children: [
        Text('$valor'),
       TextButton(
         child: Text('adicionar'),
         onPressed: () {
            setState(() {
              valor += 1;
            });
         },
        ),
```

Estados

Curso Flutter de Verão

