

Styled-components

Labenu_



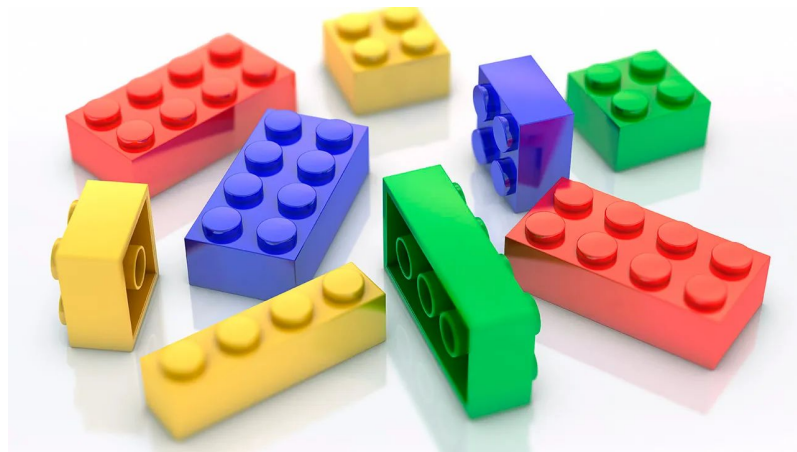
Hora de recordar

Labenu_



Components 🧐

- São **pequenos blocos** de códigos que podem ser **reutilizados**.
- Todo componente React deve iniciar com a letra **Maiúscula** e utilizar o **CamelCase** quando necessário.



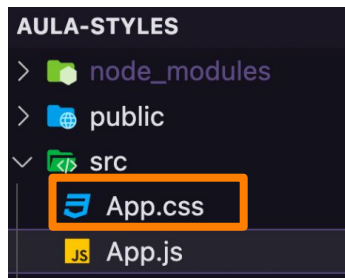
Components 🕶️

- Todo **componente React** precisa **importar** a função React.
- Um **componente React** não pode conter irmãos, nesse caso devemos envolvê-lo em um outro elemento (só um elemento pai).
- **Props** é a abreviação para propriedades e são representadas por um objeto.



Como era a estilização antes 🕶️

- Para estilizar nossos componentes, era criado um arquivo CSS. Após esse processo, o arquivo era importado na parte superior e a estilização acontecia normalmente no arquivo CSS.



```
JS App.js > ...
import logo from './logo.svg';
import './App.css';

function App() {
  return (
    <div className="App">
```



```
App.css ×
src > App.css > .App
1 .App {
2   text-align: center;
3 }
4
```



O que vamos ver hoje?

- Styled-components
 - O que é?
 - Passo a passo de utilização
 - O que muda ?
 - Dicas



styled-components

Labenu_



Motivação 🤩

- Já temos HTML e JS juntos, só o CSS permanece separado
- **Trabalhoso manter nomes de classes**
 - O React **unifica** todo o **código de CSS** em um arquivo só, então, temos que garantir que **não** iremos **repetir** os nomes
 - Isso é chato (ou até mesmo impossível)



Styled Components

- **Styled Components** é uma lib que garante que o CSS de cada componente seja totalmente restrito àquele componente
- Existem outras libs, mas escolhemos essa por ser **famosa no mercado**



Mas qual a vantagem disso? 🤔

- **Automatização** do **CSS**
- Eliminação de **bugs** devido a **classes** com o **mesmo nome**
- **Facilidade** para dar **manutenção**
- Não precisa aprender uma linguagem nova, utilizaremos o **JS** e o **CSS**



Mas qual a vantagem disso? 🤔

- A estilização **não altera outro componente** da aplicação (CSS Escopado)
- **Redução nas linhas de códigos**
- Promove uma **hierarquia visual mais semântica** e permite uma **melhor leitura do código**.



Usando styled-components

Passo a passo

Labenu_



Passo a Passo para a utilização

1. Dentro da pasta da sua aplicação, instale a biblioteca e rode : `npm install styled-components`

2. Crie um arquivo com extensão **.JS** onde será feita toda a estilização: `style.js`

3. No arquivo `style.js` , "ative" o styled-components para poder utilizá-lo : `import styled from 'styled-components'`

Passo a Passo para a utilização

4. Escolha a tag HTML que você irá estilizar

5. Dentro do arquivo **style.js** , crie uma variável para a estilização da tag HTML desejada, seguindo o padrão

```
export const Titulo = styled.h1`  
  color : red;  
`
```

Palavra reservada para você poder usar a variável em outro arquivo

Nome da variável que você escolheu

Palavra reservada da biblioteca

Tag HTML a ser estilizada

Labenu_

Passo a Passo para a utilização

6. No arquivo que você deseja estilizar a tag, importe do arquivo de estilização a variável que será utilizada :

```
import {Variável} from './style'
```

7. Chame essa estilização na tag HTML escolhida

<Titulo>Essa é uma tag H1, estilizada no
Styled Component **</Titulo>**

Variável que você estilizou usando styled
components

Exemplo sem styled-components


```
1  import './App.css';
2
3  function App() {
4    return (
5      <section>
6        <h1 className='titulo-vermelho'>Meu Título</h1>
7      </section>
8    );
9  }
10
11  export default App;
```

Meu Título

Vamos ver na prática! 



Exemplo sem styled-components 🖌️

```
1  .titulo-vermelho{  
2  |    color:  red;  
3  |  }
```

Meu Titulo

Vamos ver na prática! 🧐



Exemplo com styled-components

```
1 import styled from 'styled-components'
2
3 const RedTitle = styled.h1`
4   color: red;
5 `
6
7 function App() {
8   return (
9     <div>
10       <RedTitle>Meu Título</RedTitle>
11     </div>
12   )
13 }
```

Meu Título

Vamos ver na prática! 



Exemplo com styled-components

```
1 import styled from 'styled-components'
2
3 const RedTitle = styled.h1`
4   color: red;
5 `
6
7 const TitleContainer = styled.div`
8   display: flex;
9   align-items: center;
10  height: 60px;
11  background-color: blue;
12  padding-left: 10px;
13 `
14
15 function App() {
16   return (
17     <TitleContainer>
18       <RedTitle>Meu Título</RedTitle>
19     </TitleContainer>
20   )
21 }
```

Meu Título

Vamos ver na prática! 

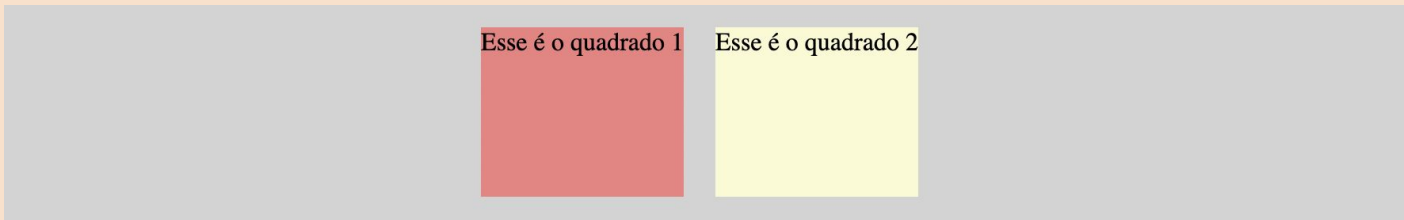




Exercício 1

- Crie uma div, que contenha dois spans. Transforme cada um desses spans em quadrados.

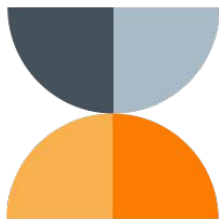
Estilize usando styled components, para que fique igual a imagem.



Pausa para relaxar 🤪

10 min

- **As bibliotecas** são importantes, pois com elas nós não precisamos reinventar a roda.
- Com o styled components, podemos unir o **CSS** com o **JS**.
- Devemos usar sintaxe **const Variável = styled.tagHTML ``** para estilizar no styled



Exemplo 2 sem styled-components 🖌️

```
1  import './App.css';
2
3  function App() {
4    return (
5      <section>
6        <div className="Imagem" />
7      </section>
8    );
9  }
10
11  export default App;
```



Vamos ver na prática! 🧐



Exemplo 2 sem styled-components

```
1  .Imagem{  
2    background-image: url(homer.jpg);  
3    width: 200px;  
4    height: 200px;  
5  }
```



Vamos ver na prática! 



Exemplo 2 com styled-components 🖌️

```
1  import styled from 'styled-components';
2  import imagem from './homer.jpg';
3
4  const Imagem = styled.div`
5    background-image: url(${(props)=>props.backgroundImage});
6    width: 200px;
7    height: 200px;
8  `;
9
10 function App() {
11   return (
12     <section>
13       <Imagem backgroundImage={imagem}/>
14     </section>
15   );
16 }
17
18 export default App;
```



Vamos ver na prática! 🧐





Exercício 2

- Crie um footer para a sua aplicação que tenha as seguintes regras da imagem ao lado:
- Transfira essas regras de estilização para o styled-components

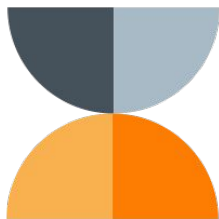
```
1 footer {  
2   background: #333b3e;  
3   color: white;  
4   position: fixed;  
5   bottom: 0;  
6   width: 100%;  
7   display: flex;  
8   padding: 0 10px;  
9 }
```



Pausa para relaxar 🤪

10 min

- É uma boa prática:
 - Sempre que tiver estilizações nais quais serão utilizadas **várias vezes**: criem **componentes**.
 - Separar os componentes em pastas diferentes quando necessário.





Exercício 3


- Crie um header para a sua aplicação que tenha as seguintes regras da imagem ao lado:
- Transfira essas regras de estilização para o styled-components

```
header {  
  background-color: orange;  
  display: flex;  
  align-items: center;  
  justify-content: space-between;  
  padding: 0 50px;  
  height: 10%;  
}
```





Dicas

- Instalem essa extensão no de vocês VSCode. Ela ajuda muito a usar styled-components!



vscode-styled-components

jpoissonnier.vscode-styled-components

Julien Poissonnier |  352.536 |  | [Repository](#) | [License](#) | 0.0.29

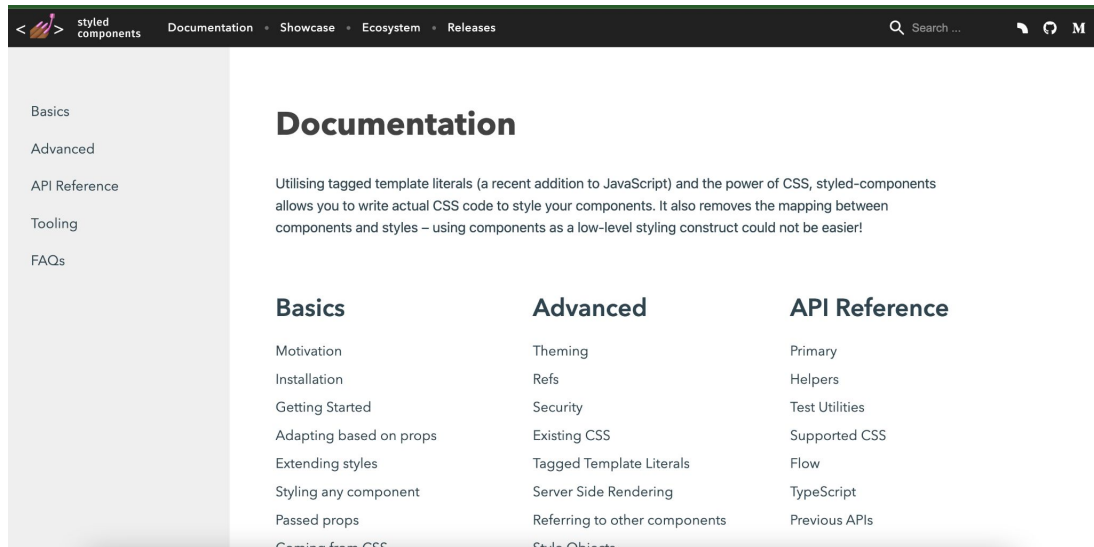
Syntax highlighting for styled-components

[Disable ▼](#) [Uninstall](#) *This extension is enabled globally.*



Dicas

- Como uma pessoa desenvolvedora, você deve ler as documentações das tecnologias para entender o que pode ou não ser feito. Com o styled não é diferente!



The screenshot shows the styled-components documentation website. The header includes the 'styled components' logo, navigation links for 'Documentation', 'Showcase', 'Ecosystem', and 'Releases', a search bar, and icons for a dark theme, a refresh button, and a user profile. The left sidebar lists navigation items: 'Basics', 'Advanced', 'API Reference', 'Tooling', and 'FAQs'. The main content area is titled 'Documentation' and contains a paragraph about using tagged template literals. Below this, there are three columns of links categorized under 'Basics', 'Advanced', and 'API Reference'.

Basics	Advanced	API Reference
Motivation	Theming	Primary
Installation	Refs	Helpers
Getting Started	Security	Test Utilities
Adapting based on props	Existing CSS	Supported CSS
Extending styles	Tagged Template Literals	Flow
Styling any component	Server Side Rendering	TypeScript
Passed props	Referring to other components	Previous APIs
Coming from CSS	Style Objects	

<https://styled-components.com/docs>



Dicas 💖

- Nós não passaremos por cada detalhezinho de como fazer coisa X ou Y usando styled components
- A sintaxe é bem parecida com o CSS comum e, se você não souber algo, basta pesquisar no google **"Como fazer TAL COISA com styled components"**
- Ainda assim, deixamos um compilado de dicas pronto pra ajudar vocês [nesse documento](#)



Resumo

Labenu_



Resumo

- **Styled Components** é uma das libs que permite que todas as partes do site estejam em **um lugar só**.
- Com o **styled components**, não precisamos nos preocupar com o nome das classes, evitando o que chamamos de **colisões de nomes**.
- Com ele, temos a facilidade para encontrar **erros**.



Resumo



```
export const Titulo = styled.h1`  
  color : red;  
`
```

`<Titulo>`Essa é uma tag H1, estilizada no
Styled Component `</Titulo>`

Na estilização

Palavra reservada para você poder usar a
variável em outro arquivo

Nome da variável que você escolheu

Palavra reservada da biblioteca

Tag HTML a ser estilizada

No componente

Variável que você estilizou usando styled
components



Resumo

- Precisamos tomar cuidado com a **sintaxe** e as **crases**!
- **Separem** e **organizem** seus componentes
- Caso não saiba como fazer algo utilizando styled-components, não se desespere! **PESQUISE** <3



Dúvidas? 🧐

Labenu_





Obrigado(a)!