FIAP

GRADUAÇÃO

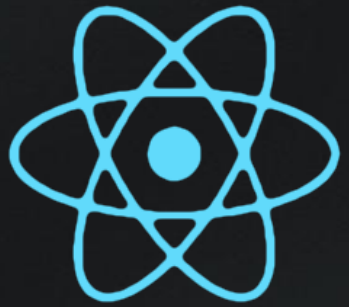
45697056

TDS
Front-End Design

Prof. Alexandre Carlos profalexandre.jesus@fiap.com.br

Prof. Luís Carlos lsilva@fiap.com.br

45697056



React

CSS em Componentes

A forma de estilização de componentes é muito parecida com a que utilizamos nos projetos sem o React. Podemos ter arquivos de estilização CSS dedicados a um ou vários componentes. Dentro da pasta src, crie um arquivo chamado App.css e insira o código abaixo:

main.jsx | App.jsx | App.css

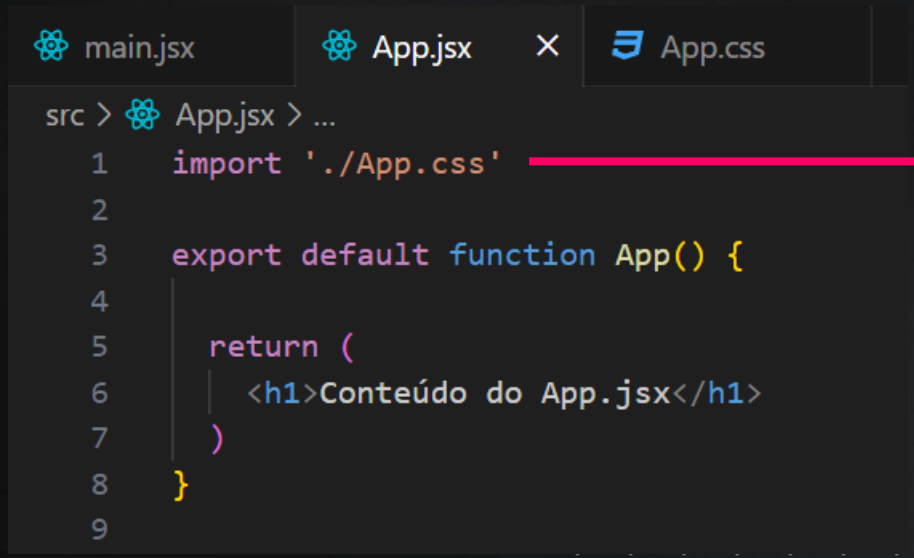
src > App.css > ...

```css
h1{
  color: darkblue;
  font-family: Arial, Helvetica, sans-serif;
  text-align: center;
}
```

OBS. Por boa prática, colocamos os nomes dos arquivos CSS iguais aos do componente.

CSS em Componentes

Criado o arquivo CSS, agora vamos importar ele dentro do arquivo App.jsx. Faça conforme abaixo:

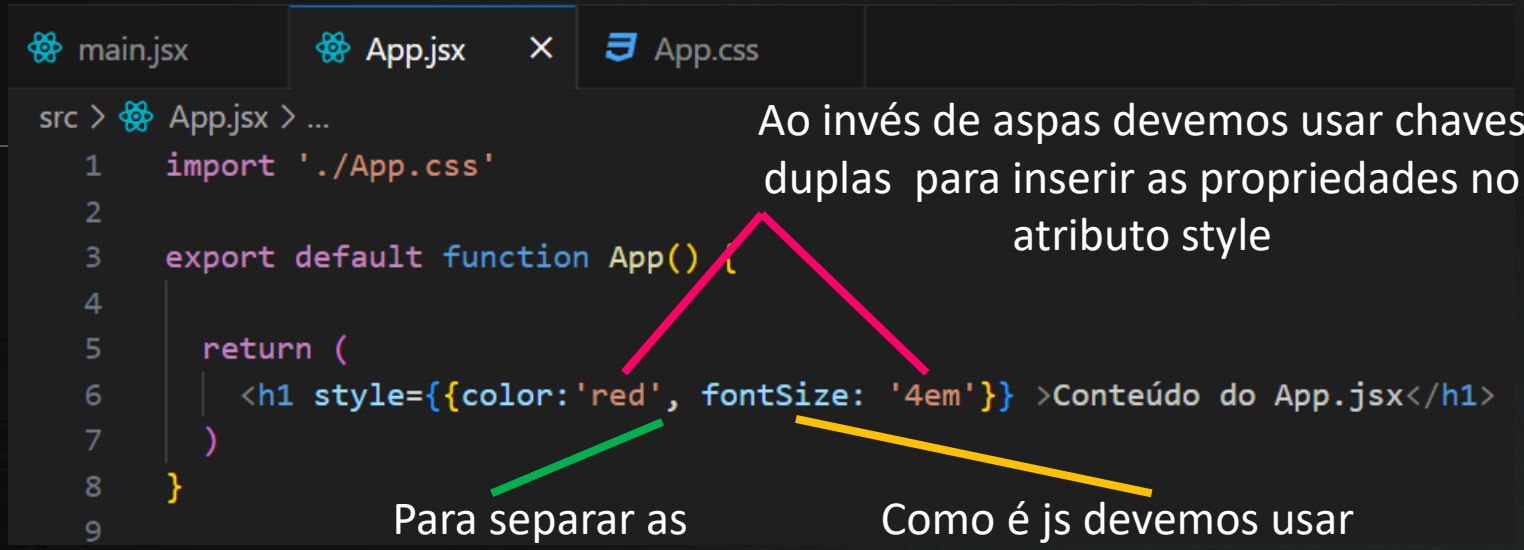


No caso do CSS o import é mais simples, não precisamos atribuir nome a ele.

CSS em Componentes

45697056

Para inserirmos valores de forma inline no elemento devemos nos atentar a pequenas diferenças:



CSS em Componentes

Quando temos muitas propriedades para passar de forma inline, podemos criar um objeto, usando as propriedades como atributos:

```
export default function App() {

  let paragr={
    textAlign: 'justify',
    color: 'darkgreen',
    textIndent: '50px',
    fontSize: '2em'
  }

  return (
    <>
      <h1 style={{color:'red', fontSize: '4em'}} >Conteúdo do App.jsx</h1>
      <p style={paragr}>Formas de inserir CSS em um elemento.</p>
    </>
  )
}
```

Criando um objeto chamado paragr e inserindo as propriedades.

Desta vez para inserir usamos chaves simples.

CSS em Componentes

Outro detalhe importante é que, para inserir uma classe no elemento devemos usar className ou invés de class:

```jsx
  return (
    <>
      <h1 style={{color:'red', fontSize: '4em'}} >Conteúdo do App.jsx</h1>
      <p style={paragr}>Formas de inserir CSS em um elemento.</p>
      <p className='exemplo'>Aqui um exemplo de className.</p>
    </>
  )
}
```

Usando o atributo className com o valor exemplo

src > App.css > ...

```css
h1{
  color: darkblue;
  font-family: Arial, Helvetica, sans-serif;
  text-align: center;
}

.exemplo{
  color: orange;
  font-size: 2em;
}

```

Criando formatação da classe exemplo.

CSS em Componentes

45697056

O componente também pode receber a estilização quando for inserido no componente pai através de seu arquivo CSS. Crie uma pasta no src chamada components e nela crie um arquivo chamado ComponenteTeste.jsx e insira o código abaixo:

main.jsx | App.jsx | App.css | ComponenteTeste.jsx

src > components > ComponenteTeste.jsx > ...

```jsx

export default function ComponenteTeste(){
    return(<p className="exemplo">Conteúdo do Componente Teste</p>)
}

```

Repare que neste exemplo, como não temos código js na função podemos usar desta forma.

CSS em Componentes

Agora é só chamar o componente no App.jsx:

src > App.jsx > ...

```jsx
import './App.css'
import ComponenteTeste from './components/ComponenteTeste'

export default function App() {

  let paragr={
    textAlign: 'justify',
    color: 'darkgreen',
    textIndent: '50px',
    fontSize: '2em'
  }

  return (
    <>
      <h1 style={{color:'red', fontSize: '4em'}} >Conteúdo do App.jsx</h1>
      <p style={paragr}>Formas de inserir CSS em um elemento.</p>
      <p className='exemplo'>Aqui um exemplo de className.</p>
      <ComponenteTeste/>
    </>
  )
}
```

Exercício

45697056

1. Crie uma nova aplicação chamada exercício2;
2. Limpe o conteúdo da pasta src e comece a estrutura da pasta do zero, criando o main.jsx e o App.jsx como seu componente principal;
3. Crie uma pasta chamada components em src;
4. Crie um componente chamado Header.jsx, dentro uma tag header com um tílulo h1 e um parágrafo;
5. Crie um componente chamado Corpo.jsx, dentro coloque um subtítulo, uma imagem e 4 parágrafos;
6. Crie um componente chamado Footer.jsx, dentro uma tag footer e um parágrafo;
7. Crie arquivos CSS para os componentes Header, Corpo e Footer e estilize a página.

CSS MODULES

45697056

- **CSS Modules** é uma abordagem de estilização que visa resolver problemas comuns do CSS tradicional, como o escopo global e a colisão de estilos. Algumas vantagens:

1. **Escopo Local**: Diferente do CSS convencional, onde os estilos são globais e podem afetar todos os elementos de uma página, os CSS Modules encapsulam os estilos em escopos locais
2. **Nomes de Classes Únicos**: As classes definidas dentro de um CSS Module são automaticamente gerenciadas e transformadas em nomes de classe exclusivos.
3. **Reutilização de Estilos:** Com CSS Modules, você pode reutilizar estilos em vários componentes sem se preocupar com conflitos de nome1.
4. **Sintaxe Padrão do CSS**: Uma das vantagens dos CSS Modules é que eles ainda usam a sintaxe padrão do CSS.

CSS MODULES

45697056

O CSS Modules é muito simples e fácil de se usar, basta seguir algumas regras:

- O arquivo CSS deve ter a nomenclatura: Nomecomponente.Module.css;
- Não pode haver traço no nome da classe ex.: .border-yellow{color: yellow;};
- O import do arquivo CSS é diferente: import style from ‘./Comp.module.css’;

CSS MODULES

45697056

React

Exemplo:

1 – Vamos alterar e simplificar o App.css

src > App.css > ...

```css
div{
    border: 2px solid blue;
    padding: 20px;
}
```

2 - Importando o arquivo App.css modificado

src > App.jsx > ...

```jsx
import './App.css'
import Componente1 from './components/Componente1'

function App() {
  return (
    <div>
      <h1>Olá Pessoal!</h1>
      <Componente1/>
    </div>
  )
}
export default App
```

3 – Na mesma pasta do componente crie o arquivo: Componente1.module.css

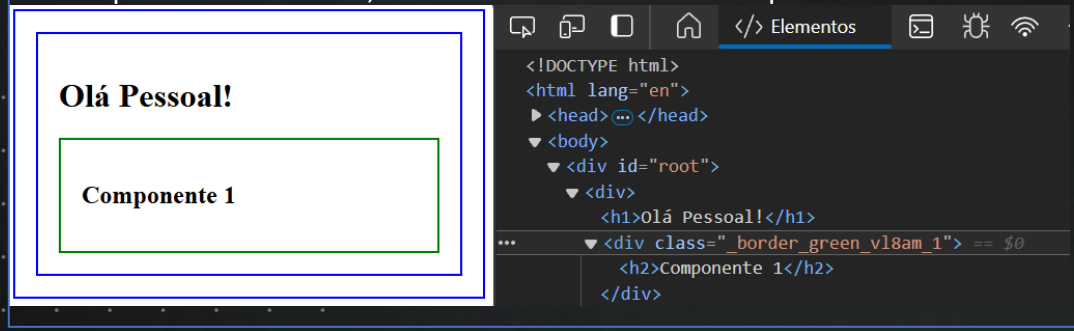
src > components > Componente1.module.css > ...

```css
.border_green{
    border: 2px solid green;
}
```

4 – Importe o CSS e atribua a classe ao elemento

```jsx
import style from './Componente1.module.css'

function Componente1() {
  return (
    <div className={style.border_green}>
        <h2>Componente 1</h2>
    </div>
);
}

export default Componente1;
```

5 – Inspecione no Browser, você tem um nome de classe personalizado.



Utilizando Styled Components

45697056

O Styled Component é uma biblioteca de estilização para agilizar a resolver alguns probleminhas de estilização nos componentes do React. Para instalar, vamos parar a aplicação e digitar “npm install --save styled-components”

```
Experimente a nova plataforma cruzada PowerShell https://aka.ms/pscore6

PS D:\_React\aula5-react> npm install --save styled-components
npm WARN @babel/plugin-bugfix-v8-spread-parameters-in-optional-chaining@7.14.5 requires a peer
of @babel/core@^7.13.0 but none is installed. You must install peer dependencies yourself.
npm WARN tsutils@3.21.0 requires a peer of typescript@>=2.8.0 || >= 3.2.0-dev || >= 3.3.0-dev |
| >= 3.4.0-dev || >= 3.5.0-dev || >= 3.6.0-dev || >= 3.6.0-beta || >= 3.7.0-dev || >= 3.7.0-bet
a but none is installed. You must install peer dependencies yourself.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@2.3.2 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.3.2: wanted {
```

Utilizando Styled Components

45697056

Mas antes de começarmos a usar, vamos entender um possível problema. Crie um componente chamado Componente1.jsx na pasta componentes:

```
import './Componente1.css'

export default function Componente1(){

    return(
        <div>
            <p>Teste</p>
        </div>
    )
}
```

Utilizando Styled Components

45697056

Como já deve ter reparado no arquivo que acabamos de criar, estamos importando um arquivo chamado Componente1.css, certo? Então vamos criá-lo:

```css
div{
    border: 3px Solid blue;
    padding: 20px;
}

p{
    color: blue;
}
```

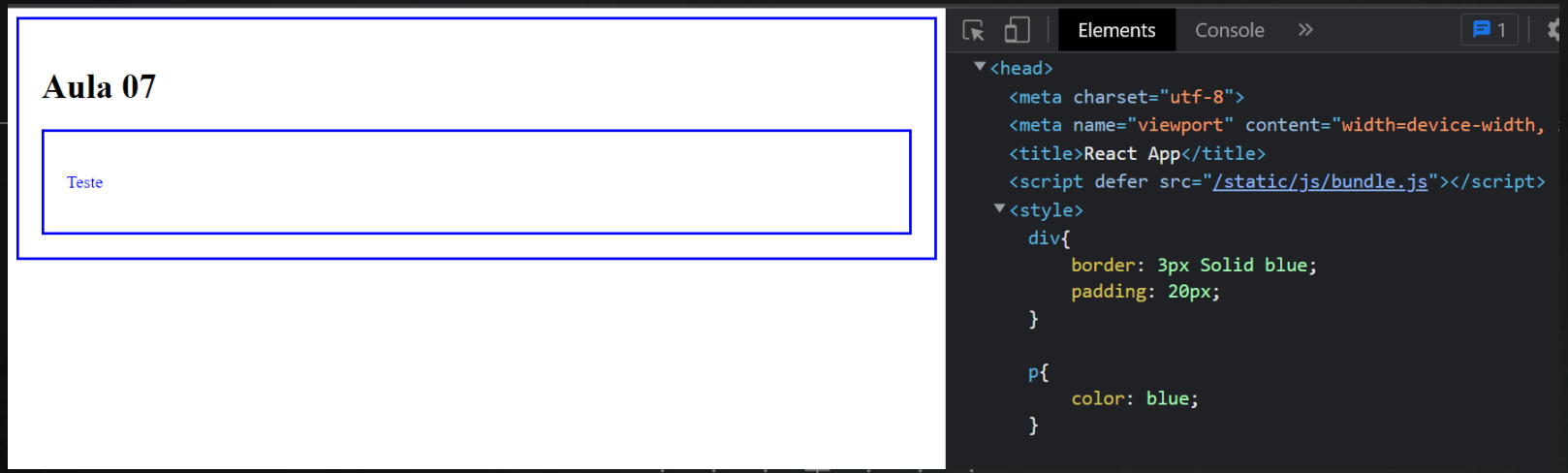
Ah! Não se esqueça em chamar o Componente1.js no App.js,

```jsx
import Componente1 from "./components/Componente1";

export default function App(){

    return(
        <>
            <h1>Aula 07</h1>
            <Componente1/>
        </>
    )
}
```

Utilizando Styled Components

45697056

O resultado não será como esperamos, perceba que a nossa div ‘root’ também recebeu a estilização, isso porque o react renderiza o css em uma tag style.



Vamos ver como podemos resolver isso usando o styled-component

Utilizando Styled Components

45697056

A primeira coisa que devemos fazer é criar nosso arquivo para criarmos as estilizações, vamos cria-lo na pasta src mesmo e vamos chamar de styled.js.

src > styled.js

```
1  import styled from 'styled-components'
2
3
4
5
```

Devemos fazer o import do styled-components

Utilizando Styled Components

45697056

No arquivo Componente1.js vamos transformar nossa div em um novo componente chamado DivComp1 e importa-lo de nosso arquivo styled.js. Calma ainda vamos criar ele.

Importamos ele e usamos no lugar da div

```
import './Componente1.css'
import { DivComp1 } from './../styled'


export default function Componente1(){

    return(
        <DivComp1>
            <p>Teste</p>
        </DivComp1>
    )
}
```

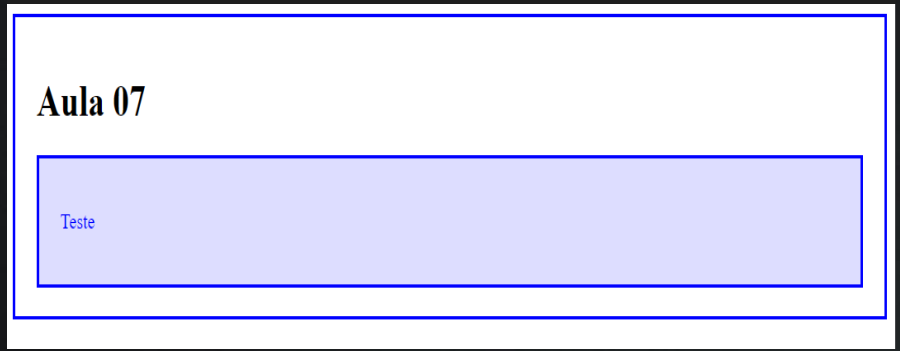
Utilizando Styled Components

Vamos fazer um teste para deixar o fundo da div na cor #ddf, volte para o arquivo styled.js e crie o código abaixo:

```
import styled from 'styled-components'

export const DivComp1 = styled.div `
    background-color: #ddf;
`
```

Repare que criamos e exportamos nosso novo elemento.

Cuidado, a estilização deve estar entre crases.



Utilizando Styled Components

Agora vamos arrumar nosso problema atual passando toda a nossa estilização para o nosso novo elemento:

```
import styled from 'styled-components'

export const DivComp1 = styled.div `
    background-color: #ddf;
    border: 3px Solid blue;
    padding: 20px;

    p{
        color: blue;
    }
`
```

Não esqueça de comentar a importação do css no Componente1.js

```
//import './Componente1.css'
import { DivComp1 } from './..
```

Utilizando Styled Components

Nós podemos também criar nossos componentes de estilização na própria página. Vamos criar o Componente2.js e criar nosso componente dentro dele:

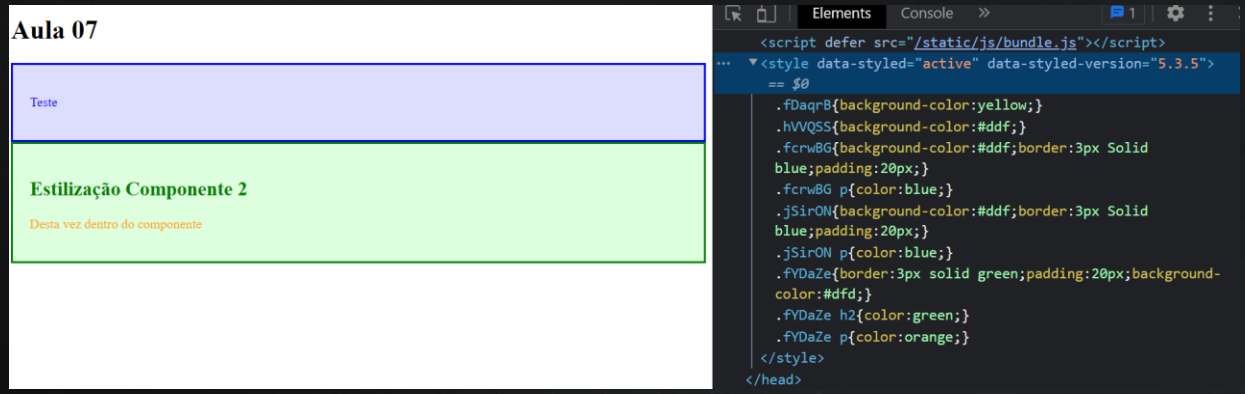
Não esqueça de chamar o componente2.js no App.js

```
import styled from 'styled-components'

const DivComp2 = styled.div `
    border: 3px  solid green;
    padding: 20px;
    background-color: #dfd;

    h2{ color: green;}
    p{ color: orange;}
`

export default function Componente2(){

    return(
        <DivComp2>
            <h2>Estilização Componente 2</h2>
            <p>Desta vez dentro do componente</p>
        </DivComp2>
    )
}
```

Utilizando Styled Components

45697056

Observe o resultado final, ele cria um controle interno em classes para garantir que não teremos problemas como aqueles do início.



Utilizando Styled Components

Para finalizarmos vamos ver como criar valores globais de estilização. Crie um arquivo chamado estiloGlobal.js na pasta src. Depois chame ele no index.js, junto com App.

```
45697056
```

```
import { createGlobalStyle } from "styled-components";

const globalStyle = createGlobalStyle `
    *{
        margin: 0;
        padding: 0;
        box-sizing: border-box;

    }

`

export default globalStyle
```

```
import React from 'react';
import ReactDOM from 'react-dom/client';

import App from './App';


const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

Exercício

45697056

Crie um novo projeto chamado exercicio7, limpe todos os arquivos das pastas public e src, após isso recrie o arquivo index.html na pasta public e os arquivos index.js e App.jsx na pasta src.

Pensando em um tema de sorveteria, crie um componente Cabecalho.jsx, dentro dele um h1 e um parágrafo.

Logo após crie um componente chamado Corpo.jsx, dentro dele um h2 e uma lista de sabores, outro h2 e uma lista de acompanhamentos.

Por fim crie um componente chamado Rodape.jsx, dentro um parágrafo com o endereço da sorveteria.

Após todos os componentes criados, use um styled-componentes externo para o cabeçalho e rodapé e um interno para o corpo.

No Corpo crie uma estilização para 2 componentes criando suas estilizações individuais.

Faça também algumas estilizações globais.

DUVIDAS

Copyright © 2015 - 2021 Prof. Luís Carlos S. Silva
Prof. Alexandre Carlos de Jesus

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proíbido sem o consentimento formal, por escrito, do Professor (autor).