CSS Modules

O que é

CSS Modules é uma abordagem de escopo local para escrever estilos em CSS. Em vez de aplicar os estilos globalmente, como é o caso com o CSS tradicional, os CSS Modules geram classes únicas para cada componente, evitando conflitos de nomes e garantindo que os estilos sejam aplicados apenas aos componentes onde são definidos. Isso é particularmente útil em aplicações com muitos componentes, onde o risco de colisão de nomes de classes é maior
Documentação: https://github.com/css-modules/css-modules

Vantagens

Escopo Local de Estilos
Evita conflitos de nomes, garantindo que os estilos são aplicados apenas aos componentes específicos.

Manutenção Facilitada
Como os estilos são locais, fica mais fácil gerenciar e manter o código CSS em projetos grandes.

Composição de Classes
Permite combinar e reutilizar estilos entre diferentes componentes, promovendo a modularidade.

Segurança de Estilo
Reduz a chance de estilos não intencionais afetarem outros componentes, garantindo uma aplicação mais previsível.

Usando CSS Modules

1. Vamos abrir o um projeto já criado com o Vite + React + TypeScript como já vimos em aulas anteriores. Caso não se lembre de como criar, é só executar npm create vite@latest e seguir os passos que vimos nas aulas anteriores.

2. Agora vamos renomear o arquivo App.css para App.module.css

3. Vamos trocar a importação no nosso App.tsx

```tsx
import { useState } from 'react'
import reactLogo from './assets/react.svg'
import viteLogo from '/vite.svg'
import './App.css'

function App() {
```

```tsx
import { useState } from 'react'
import reactLogo from './assets/react.svg'
import viteLogo from '/vite.svg'
import appStyles from './App.module.css'

function App() {
  const [count, setCount] = useState(0)
```

Usando CSS Modules

4. Agora vamos transformar o nosso HTML, ao invés de usarmos as classes como String, vamos usar elas como objetos do nosso modulo de CSS importado

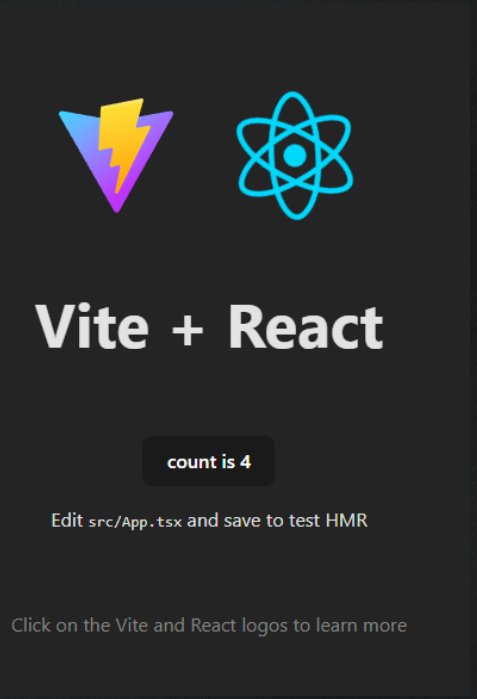
```jsx
return (
  <>
    <div>
      <a href="https://vitejs.dev" target="_blank">
        <img src={viteLogo} className="logo" alt="Vite logo" />
      </a>
      <a href="https://react.dev" target="_blank">
        <img src={reactLogo} className="logo react" alt="React logo" />
      </a>
    </div>
    <h1>Vite + React</h1>
    <div className="card">
      <button onClick={() => setCount((count) => count + 1)}>
        count is {count}
      </button>
      <p>
        Edit <code>src/App.tsx</code> and save to test HMR
      </p>
    </div>
    <p className="read-the-docs">
      Click on the Vite and React logos to learn more
    </p>
  </>
)
```

```jsx
return (
  <>
    <div>
      <a href="https://vitejs.dev" target="_blank">
        <img src={viteLogo} className={appStyles.logo} alt="Vite logo" />
      </a>
      <a href="https://react.dev" target="_blank">
        <img src={reactLogo} className={`${appStyles.logo} ${appStyles.react}`} alt="React logo" />
      </a>
    </div>
    <h1>Vite + React</h1>
    <div className={appStyles.card}>
      <button onClick={() => setCount((count) => count + 1)}>
        count is {count}
      </button>
      <p>
        Edit <code>src/App.tsx</code> and save to test HMR
      </p>
    </div>
    <p className={appStyles['read-the-docs']}>
      Click on the Vite and React logos to learn more
    </p>
  </>
)
```

Usando CSS Modules

5. Por fim vamos mover a classe #root do arquivo App para o arquivo index.css. Vamos fazer isso porque na nossa tela não temos como usar o #root já que ele se refere a uma div global, que está no HTML dentro do index.html.

Então teremos quase o mesmo resultado



Usando CSS Modules

FIAP

É quase o mesmo resultado porque o nosso HTML FINAL é ligeiramente diferente:

As classes que colocamos ganharam vários underline e sufixos gerados automaticamente. Dessa forma as classes são únicas e não vão se misturar caso com outras de mesmo nome, como poderia acontecer anteriormente. Isso é muito útil para evitar concorrência de nome de classes

Dúvidas, críticas ou sugestões?


FIAP