**Passo 1: Criar um novo projeto com Vite**

Primeiro, crie um novo projeto React com TypeScript usando Vite.

bash

Copiar código

npm create vite@latest react-styled-emotion-example --template react-ts

cd react-styled-emotion-example

npm install

**Passo 2: Instalar as dependências necessárias**

Instale as bibliotecas styled-components e @emotion/react junto com seus tipos para TypeScript.

bash

Copiar código

npm install styled-components @emotion/react @emotion/styled

npm install --save-dev @types/styled-components

**Passo 3: Estrutura básica do projeto**

No seu projeto, organize as pastas e arquivos da seguinte forma:

css

Copiar código

src/

│

├── components/

│ ├── HeaderStyled.tsx

│ ├── HeaderEmotion.tsx

│ └── Button.tsx

│

└── App.tsx

**Passo 4: Implementando o componente Header com Styled-components**

**HeaderStyled.tsx**

typescript

Copiar código

import React from 'react';

import styled from 'styled-components';

// Definindo os componentes estilizados

const HeaderContainer = styled.header`

background-color: #282c34;

padding: 20px;

text-align: center;

color: white;

`;

const Title = styled.h1`

font-size: 2.5rem;

`;

const Description = styled.p`

font-size: 1.2rem;

color: #61dafb;

`;

function HeaderStyled() {

return (

<HeaderContainer>

<Title>Welcome to React with Styled-components</Title>

<Description>This is an example using Styled-components.</Description>

</HeaderContainer>

);

}

export default HeaderStyled;

**Passo 5: Implementando o componente Header com Emotion**

**HeaderEmotion.tsx**

typescript

Copiar código

import React from 'react';

/\*\* @jsxImportSource @emotion/react \*/

import { css } from '@emotion/react';

const headerStyle = css`

background-color: #282c34;

padding: 20px;

text-align: center;

color: white;

`;

const titleStyle = css`

font-size: 2.5rem;

`;

const descriptionStyle = css`

font-size: 1.2rem;

color: #61dafb;

`;

function HeaderEmotion() {

return (

<header css={headerStyle}>

<h1 css={titleStyle}>Welcome to React with Emotion</h1>

<p css={descriptionStyle}>This is an example using Emotion.</p>

</header>

);

}

export default HeaderEmotion;

**Passo 6: Implementando o componente Button com Styled-components e Emotion**

**Button.tsx**

typescript

Copiar código

import React from 'react';

import styled from 'styled-components';

/\*\* @jsxImportSource @emotion/react \*/

import { css } from '@emotion/react';

interface ButtonProps {

label: string;

onClick: () => void;

}

// Estilização com Styled-components

const StyledButton = styled.button`

background-color: #61dafb;

color: white;

border: none;

padding: 10px 20px;

cursor: pointer;

font-size: 1rem;

border-radius: 5px;

&:hover {

background-color: #21a1f1;

}

`;

// Estilização com Emotion

const emotionButtonStyle = css`

background-color: #e91e63;

color: white;

border: none;

padding: 10px 20px;

cursor: pointer;

font-size: 1rem;

border-radius: 5px;

&:hover {

background-color: #d81b60;

}

`;

function Button({ label, onClick }: ButtonProps) {

return (

<div>

<StyledButton onClick={onClick}>{label} (Styled-components)</StyledButton>

<button css={emotionButtonStyle} onClick={onClick} style={{ marginLeft: '10px' }}>

{label} (Emotion)

</button>

</div>

);

}

export default Button;

**Passo 7: Atualizar o componente App.tsx**

**App.tsx**

typescript

Copiar código

import React from 'react';

import HeaderStyled from './components/HeaderStyled';

import HeaderEmotion from './components/HeaderEmotion';

import Button from './components/Button';

function App() {

const handleClick = () => {

alert('Button clicked!');

};

return (

<div>

<HeaderStyled />

<HeaderEmotion />

<div style={{ textAlign: 'center', marginTop: '20px' }}>

<Button label="Click Me" onClick={handleClick} />

</div>

</div>

);

}

export default App;

**Passo 8: Executar o projeto**

Para executar o projeto, use o comando:

bash

Copiar código

npm run dev

Acesse http://localhost:3000 no seu navegador para ver a aplicação funcionando.

**Explicação**

* **Styled-components:** É uma biblioteca que permite escrever estilos CSS em arquivos JavaScript/TypeScript. Ela utiliza tagged template literals para aplicar estilos aos componentes.
* **Emotion:** É outra biblioteca de estilização que permite aplicar estilos CSS diretamente nos componentes React. O Emotion também permite o uso de css para definir estilos com poderosas funcionalidades, como composição e interpolação.

Essas duas abordagens permitem criar componentes estilizados com escopo de estilos, mantendo os estilos organizados e reutilizáveis.