1. Atividades Práticas - Estilos Inline com React

Nesta atividade prática, vamos criar um exemplo simples passo a passo para ilustrar como podemos usar os Estilos Inline em uma aplicação React.

1.1 Criando o projeto e iniciando o servidor local

Crie um projeto chamado **compras**, digitando o seguinte comando no terminal:

```
npm create vite@latest my-css-inline -- --template react
```

Este processo de configuração inicial do projeto leva alguns segundos. Ao terminar, o Vite repassa instruções para que você termine de instalar as dependências do seu projeto. Desta forma, digite o seguinte comando para **entrar no diretório** recém criado do nosso projeto:

```
cd my-css-inline
```

Em seguida, **instale as dependências** necessárias do projeto executando no terminal o seguinte comando:

```
npm install
```

Até aqui, você criou um projeto React usando o Vite e adicionou todas as dependências ao projeto. Você pode, agora, **abrir a aplicação no editor Visual Studio Code**. Para isso, dentro do diretório do projeto digite o seguinte comando no terminal:

```
code .
```

Após a execução deste comando, o Visual Studio Code deverá abrir com a pasta raiz do seu projeto sendo acessada. Em seguida, você inicializará um servidor local e executará o projeto em seu navegador. Digite a seguinte linha de comando no terminal:

```
npm run dev
```

Ao executar esse script, você iniciará um servidor local de desenvolvimento, executará o código do projeto, iniciará um observador que detecta alterações no código e abrirá o projeto em um navegador web. Ele irá rodar a aplicação em modo desenvolvimento em http://localhost:5173/.

1.2 Passo 1: Aplicar Estilos Inline no componente React

No arquivo **App.jsx** começaremos aplicando estilos diretamente nos elementos utilizando a propriedade **style**. Vamos definir uma interface simples com um botão que, ao clicar, muda de cor dinamicamente, conforme quadro abaixo:

```
import { useState } from "react";

function App() {
  // Definindo um estado para mudar o estilo dinamicamente
  const [ativo, setAtivo] = useState(false);

  // Definindo estilos inline como objetos
  const estiloContainer = {
    backgroundColor: ativo ? "lightgreen" : "lightblue", // Muda com base no
estado
    padding: "20px",
    borderRadius: "10px",
    textAlign: "center",
  };

  const estiloTitulo = {
    color: ativo ? "darkgreen" : "darkblue",
    fontSize: "24px",
    marginBottom: "10px",
  };

  const estiloBotao = {
    backgroundColor: ativo ? "green" : "gray",
    color: "white",
    padding: "10px 20px",
    border: "none",
    borderRadius: "5px",
    cursor: "pointer",
    transition: "background-color 0.3s ease",
  };

  // Função para alternar o estado
  const alternarAtivo = () => {
    setAtivo(!ativo); // Alterna entre verdadeiro e falso
  };
```

```
  return (
    <div style={estiloContainer}>
      <h1 style={estiloTitulo}>Clique no Botão</h1>
      <button style={estiloBotao} onClick={alternarAtivo}>
        {ativo ? "Ativo" : "Inativo"}
      </button>
    </div>
  );
}

export default App;
```

Observe neste exemplo:

- **Objeto de Estilos Inline:** Para cada elemento (**div**, **h1**, **button**), criamos um objeto de estilos em JavaScript, como **estiloContainer**, **estiloTitulo** e **estiloBotao**.
- **Condicionais nos Estilos:** Usamos um estado booleano (**ativo**) para alterar dinamicamente as cores de fundo e texto do botão e do container. Se o estado for **true** (**ativo**), o botão será verde, caso contrário, será cinza.
- **Função de Alternância:** A função **alternarAtivo** é acionada ao clicar no botão e alterna o estado entre **true** e **false**, o que muda os estilos aplicados.

1.4 Passo 3: Testar a aplicação

Agora, você pode rodar o projeto para ver os estilos aplicados.

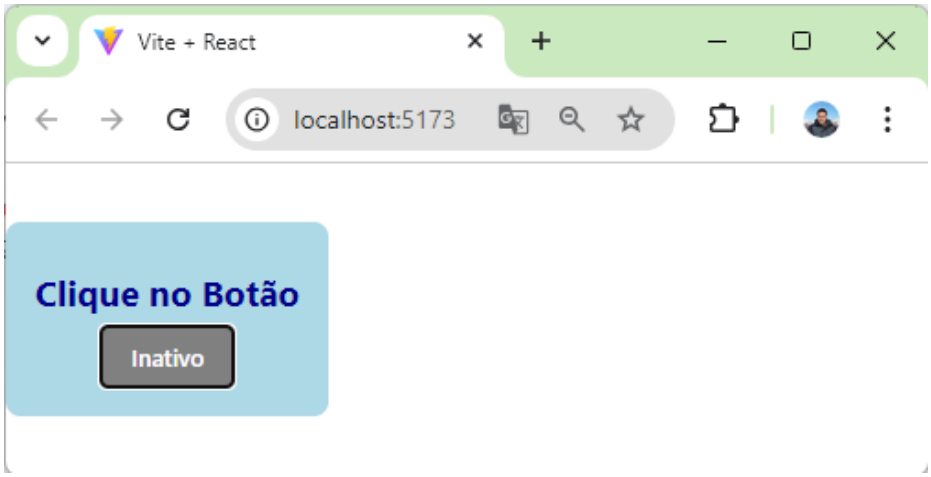

Figura 1: Exemplo de uso de Estilos Inline com React

Observe que a saída esperada no navegador é a seguinte:

- O fundo da div será azul-claro e o título será azul-escuro.
- O botão estará com o fundo cinza e o texto "Inativo".

Porém, ao clicar no botão, o fundo da **div** mudará para verde-claro, o título mudará para verde-escuro, e o botão mudará para verde com o texto "Ativo".

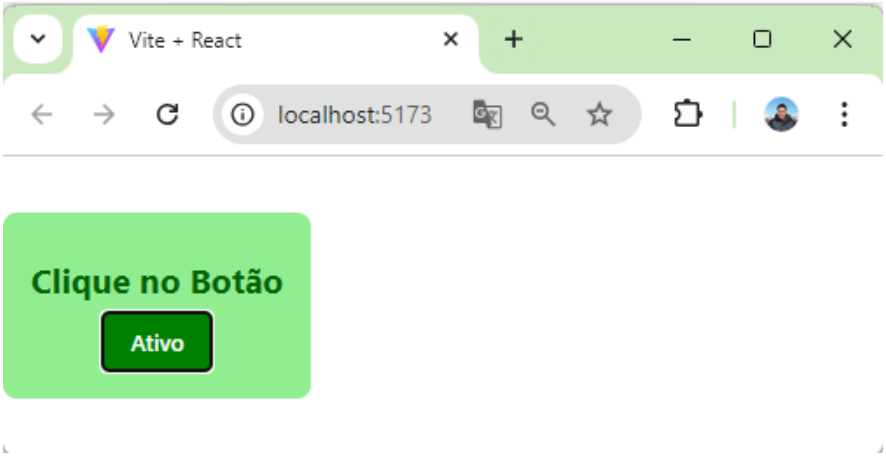

Figura 2: Exemplo de uso de Estilos Inline com React após clicar no botão

1.5 Passo 4: Estilos Inline Condicionais

Você pode expandir o exemplo para adicionar mais condições ou estilos baseados em interações. Por exemplo, podemos adicionar efeitos **hover** (interações visuais que ocorrem quando o usuário passa o cursor do mouse sobre um elemento em uma página web) ao botão de maneira programática. Cabe salientar que o React não suporta diretamente o **:hover** nos estilos inline, mas podemos simular isso utilizando um estado para identificar quando o mouse está sobre o botão:

```
// App.jsx
import { useState } from "react";

function App() {
  const [ativo, setAtivo] = useState(false);
  const [hover, setHover] = useState(false); // Estado para hover
```

```
  const estiloContainer = {
    backgroundColor: ativo ? "lightgreen" : "lightblue",
    padding: "20px",
    borderRadius: "10px",
    textAlign: "center",
  };
  const estiloTitulo = {
    color: ativo ? "darkgreen" : "darkblue",
    fontSize: "24px",
    marginBottom: "10px",
  };
  const estiloBotao = {
    backgroundColor: hover ? (ativo ? 'darkgreen' : 'darkgray') : (ativo ? 
'green' : 'gray'), // Efeito hover
    color: "white",
    padding: "10px 20px",
    border: "none",
    borderRadius: "5px",
    cursor: "pointer",
    transition: "background-color 0.3s ease",
  };
  const alternarAtivo = () => {
    setAtivo(!ativo);
  };
  return (
    <div style={estiloContainer}>
      <h1 style={estiloTitulo}>Clique no Botão</h1>
      <button
        style={estiloBotao}
        onClick={alternarAtivo}
        onMouseEnter={() => setHover(true)} // Detecta mouse sobre o botão
        onMouseLeave={() => setHover(false)} // Detecta quando o mouse sai do 
botão
      >
        {ativo ? "Ativo" : "Inativo"}
      </button>
    </div>
  );
}
export default App;
```

Observe neste código:

- **hover State:** Criamos um estado **hover** que monitora se o mouse está sobre o botão (**true** ou **false**).
- **onMouseEnter e onMouseLeave:** Estes eventos do mouse atualizam o estado **hover**. Quando o mouse entra no botão, **hover** é definido como **true**, e quando sai, é definido como **false**.
- **Condicionais no Estilo:** O **backgroundColor** do botão agora depende tanto do estado ativo quanto de **hover**. Se o mouse estiver sobre o botão (**hover === true**), o fundo muda para uma cor mais escura.

Teste novamente a aplicação e passe o mouse sobre o botão. Veja que sua cor mudou.

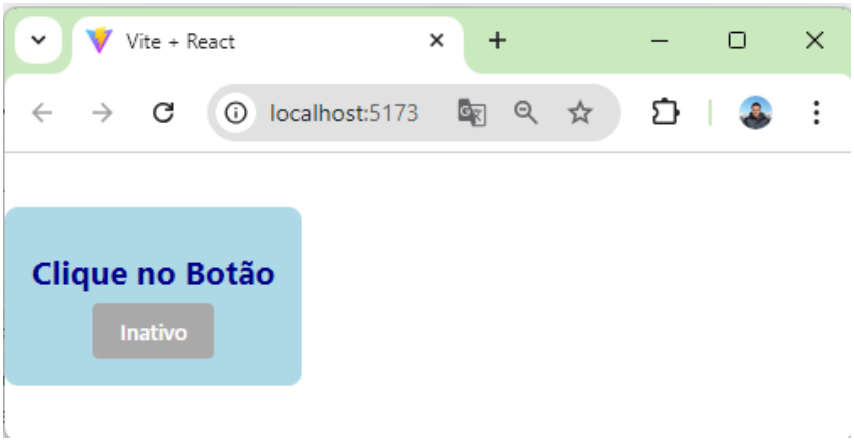

Figura 3: Exemplo de uso de Estilos Inline Condicionais