Eventos no React, Formulários e Variáveis de ambiente

Link do Notion: https://cherry-client-b8f.notion.site/Eventos-no-React-Formul-rios-e-Vari-veis-de-ambiente-62dbd1b544fb4c7f869d2c654f0e3d96?pvs=73

Eventos

Eventos no React e jo JS são a forma de **responder a interações do usuário** ou a certas ações que acontecem na página como clique, digitação, envio de formulário e outros.
A forma de lidar com eventos em React é bem parecida com a forma em que inserimos eventos no Javascript Vanilla dentro do HTML, com a diferença que utilizamos o **camelCase** ao chamar o evento e a função é chamada **dentro das chaves e sem parenteses, senão a função é chamada no momento que o componente é carregado na tela.**

```
function App() {
  function handleClick() {
    alert("Clicou!");
  }

  return <button onClick={handleClick}>Clique aqui</button>;
}
```

Como passar parâmetros para a função

Para passar parâmetros para a função, você deve chamar a função dentro de uma arrow function.

```
<button onClick={() => handleClick(5)}>Clique</button>
```

Lidando com formulários

Para lidar com formulários no React, utilizamos estados para sincronizar o que está sendo digitado com o valor armazenado em tempo real, garantindo a criação de formulários dinâmicos e flexíveis.

Como capturar conteúdo de input?

Para capturar o conteúdo de um input, devemos sempre **vinculá-lo à um estado**, tornando a propriedade **value** **igual ao state** criado e passando o evento **onChange**, executando uma função que irá atualizar o estado toda vez que o input for alterado.

```
import { useState } from 'react';
import ReactDOM from 'react-dom/client';

export default function MyForm() {
  const [nome, setNome] = useState("");

  return (
    <label htmlFor="inputNome">Escreva seu nome: </label>
      <input
        type="text"
        value={nome}
        id="inputNome"
        onChange={(e) => setNome(e.target.value)}
      />
    <button type="submit" />
```

```
  )
}
```

Como capturar conteúdo enviado através de um formulário?

Para lidar com envios em formulários, utilizamos o evento **onSubmit** na tag de formulário, passando para ele a função que será executada.

```
import { useState } from 'react';
import ReactDOM from 'react-dom/client';

export default function MyForm() {
  const [nome, setNome] = useState("");

  const handleSubmit = (event) => {
    event.preventDefault();
    alert(`O nome que você digitou foi: ${nome}`)
  }

  return (
    <form onSubmit={handleSubmit}>
      <label htmlFor="inputNome">Escreva seu nome: </label>
        <input
          type="text"
          value={nome}
          id="inputNome"
          onChange={(e) => setNome(e.target.value)}
        />
      <button type="submit" />
    </form>
  )
}
```

Lidando com formulários grandes

Muitas vezes teremos que lidar com formulários grandes, que possuem diversos campos a serem preenchidos e com isso, muitos estados precisam ser criados. Para evitar a criação de um estado para cada input, é possível guardar tudo dentro de um **único estado**, através de um **objeto**.

```
const [formData, setFormData] = useState({
  nome: "",
  email: "",
  senha: "",
});

function handleChange(e) {
  setFormData({
    ...formData,
    [e.target.name]: e.target.value, // chave dinâmica
  });
}
```

Deixando o input dessa maneira:

```
<input name="nome" value={formData.nome} onChange={handleChange} />
<input name="email" value={formData.email} onChange={handleChange} />
```

Variáveis de ambiente

No React, variáveis de ambiente são **valores externos que o código lê durante o processo de compilação para configurar a aplicação de forma dinâmica, permitindo separar informações sensíveis, como chaves de API, e configurar URLs para diferentes ambientes (desenvolvimento, produção) sem alterar o código principal**.

Para isso, é necessário criar um arquivo chamado **.env** na pasta raiz do projeto. Em projetos utilizando VIte, a variável deve sempre iniciar com VITE_, dessa maneira:

```
VITE_API_URL=https://meu-backend.com/api
VITE_APP_NAME=MeuApp
```

Para utilizar a variável no código, você pode acessar dessa maneira:

```
const apiUrl = import.meta.env.VITE_API_URL
console.log("Minha API:", apiUrl)
```

Lembre-se de se certificar que esse arquivo esteja constando no .gitignore, assim seus dados não ficarão expostos no GitHub. É comum criar um arquivo **.env.example** com dados fictícios, mostrando como essas variáveis devem ser preenchidas para quem clonar o repositório.

Referências

https://www.geeksforgeeks.org/react-js-events/

https://blog.logrocket.com/react-onclick-event-handlers-guide/

https://pt.vite.dev/guide/env-and-mode