O ambiente React

Othon Oliveira

SENAC - PROA

Sumário

Introdução ao React.js

Aula 1: Introdução ao React.js

Duração: 4 horas

Othon Oliveira

SENAC - PROA

O que é React.js?

Definição

React.js é uma biblioteca JavaScript de código aberto que permite a construção de interfaces de usuário interativas e reativas.

- Desenvolvida e mantida pelo Facebook.
- Lançada em 2013.
- Ampla adoção na indústria de desenvolvimento web.

Vantagens do React

- Reatividade: Atualizações automáticas quando os dados mudam.
- Componentização: Interface dividida em componentes reutilizáveis.
- Virtual DOM: Melhora o desempenho das atualizações de interface.

Exemplo de Componente

```
function Button(props) {
    return <button>{props.label}<button>;
}
```

Passos para Configurar o Ambiente

1. Instale o Node.js e o npm (ou yarn) no seu sistema.
2. Abra o terminal ou prompt de comando.
3. Crie um novo diretório para o seu projeto:

```
mkdir meu-projeto-react
cd meu-projeto-react
```

4. Inicialize um novo projeto React usando o Create React App:

```
npx create-react-app .
```

5. Aguarde a instalação das dependências.
6. Inicie o servidor de desenvolvimento:

```
npm start
```

7. Seu ambiente de desenvolvimento React está pronto!

Criando um Componente

- Um componente é uma parte isolada da interface.
- Exemplo: um componente de "Botão".

Estrutura de um Componente

Estrutura Básica de um Componente

Um componente em React pode ser definido como uma função ou uma classe. Ambas as abordagens têm uma estrutura básica comum.

Estrutura de um Componente (Função)

Exemplo de Componente como Função

```
function MeuComponente(props) {
  return (
    <div>
      <h1>{props.titulo}</h1>
      <p>{props.texto}</p>
    </div>
  );
}
```

Estrutura de um Componente (Classe)

Exemplo de Componente como Classe

```
class MeuComponente extends React.Component {
  render() {
    return (
      <div>
        <h1>{this.props.titulo}</h1>
        <p>{this.props.texto}</p>
      </div>
    );
  }
}
```

Diferenças entre Função e Classe (1/3)

Componente como Função

- Declaração simples como uma função.
- Usa ‘props‘ como argumento.
- Não mantém estado interno (state).

Diferenças entre Função e Classe (2/3)

Componente como Classe

- Declaração como uma classe que estende ‘React.Component‘.
- Usa ‘this.props‘ para acessar as propriedades.
- Pode manter estado interno (state) com ‘this.state‘.

Diferenças entre Função e Classe (3/3)

Escolhendo entre Função e Classe

- Use componentes como funções quando possível para simplicidade.
- Use componentes como classes quando precisa de estado interno.
- React 16.8+ oferece Hooks para adicionar estado a componentes de função.

Estado Interno (Parte 1)

O que é Estado Interno?

O estado interno (state) em um componente React é uma forma de armazenar e gerenciar dados que podem mudar ao longo do tempo.
É usado para rastrear informações específicas ao componente que podem afetar a renderização.

Estado Interno (Função)

Uso de Estado Interno em Componentes como Função

Em componentes como função, você pode usar o Hook ‘useState‘ para adicionar estado interno.

Exemplo de Componente como Função com Estado Interno

```
import React, { useState } from 'react';
function Contador() {
  const [contador, setContador] = useState(0);
  return (
   <div>
    <p>Contagem: {contador}</p>
    <button onClick={() => setContador(contador + 1)}>
    Incrementar </button>
    </div>
  );
}
```

Estado Interno (Classe)

Uso de Estado Interno em Componentes como Classe

Em componentes como classe, você pode definir o estado interno usando ‘this.state‘ e atualizá-lo com ‘this.setState‘.

Exemplo de Componente como Classe com Estado Interno

```
class Contador extends React.Component {
  constructor(props) {
    super(props);
    this.state = { contador: 0 };
  }
  render() {
 // próximo slide
```

Estado Interno (Classe)

Uso de Estado Interno em Componentes como Classe

Em componentes como classe, você pode definir o estado interno usando ‘this.state‘ e atualizá-lo com ‘this.setState‘.

Exemplo de Componente como Classe com Estado Interno

```
// continua abaixo do render()
    return (
     <div>
      <p>Contagem: {this.state.contador}</p>
       <button onClick={() => this.setState({ contador: this.st
       </button>
     </div>
     );
  }
}
```

JSX (JavaScript XML)

- Sintaxe semelhante a HTML no JavaScript.

JSX (JavaScript XML)

- Sintaxe semelhante a HTML no JavaScript.
- Permite a criação de elementos de interface de forma declarativa.

Renderização de Componentes

- Usando o ReactDOM.render() para renderizar um componente em um elemento HTML existente na página.

Renderizando um Componente

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

const element = <Button label="Clique-me" />;

ReactDOM.render(element, document.getElementById('root'));
```

Atividade Prática

- Exercício prático: Crie um componente React simples e renderize-o em uma página web.

Código de um Componente "HelloWorld" React

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

function HelloWorld() {
  return <h1>Hello, React!</h1>;
}

ReactDOM.render(<HelloWorld />, document.getElementById('r
```