REACT.

MATERIAL DE APOIO

descomplica

Disciplina: React

Nome da aula: Introdução

Professor: Ayrton Teshima

Tutor: Rodrigo Vasconcelos

Introdução

O desenvolvimento web evoluiu muito nos últimos anos. Podemos destacar nessa evolução os seguintes itens:

- Evolução do navegador
- Evolução da linguagem JavaScript
- Evolução dos dispositivos (hardware de computadores e devices no geral)
- Evolução da internet

A partir de toda essa evolução, novas experiências dentro do navegador foram permitidas e impulsionadas. Com isso, a complexidade do desenvolvimento também cresceu, é aí que surgem frameworks e bibliotecas JavaScript como o React.

Objetivos da aula

- Identificar o motivo do React existir
- Os benefícios do React
- Diferenciais do React
- Inserir o React em seu projeto

Resumo

Antes tínhamos um JavaScript simples rodando no navegador, muitas vezes para fazer uma animação simples ou buscar um dado no servidor. Hoje conseguimos criar aplicações para navegar offline, animações complexas e fluidas. Podemos criar um editor de imagem no navegador, construtor de páginas, editor de vídeos e muito mais. Como dito, o desenvolvimento com JavaScript ficou muito complexo e cheio de desafios. É aí que surgem as bibliotecas e frameworks. O React veio para atenuar esses desafios de criar interfaces UI interativas, com fácil desenvolvimento e performática.

descomplica

O React agiliza muito o trabalho complexo que faríamos com JavaScript puro, além de já solucionar diversos problemas que teríamos se fôssemos criar do zero. O React muda a mentalidade, facilitando muito nossas vidas e ao mesmo tempo trazendo desafios diferentes

Características do React:

O React possui três características muito significativas:

- Declarativo
- Baseado em componentes
- Sistema reativo

Vamos entender um pouco sobre cada um desses.

Declarativo:

React tem uma característica que facilita muito nossa vida de desenvolvedor, que é o desenvolvimento declarativo. Declarativo é você descrever **o que** quer e não **como** quer.

Existe uma diferença substancial entre o **como** e **o quê**.

O como é o passo a passo, lembra muito quando você está aprendendo lógica de programação e você precisa pensar no passo a passo de tudo.

Ex: atravessar a rua.

Passo a passo para atravessar a rua

Como

1. Pare na faixa;
2. Espere o sinal fechar;
3. Olhe para o lado, veja se vem carro;
4. Olhe para o outro lado, veja se vem carro;
5. Se não estiver vindo carro de ambos os lados, atravesse

O quê

1. Atravesse

Quando você envia o comando “atravesse”, todas as outras coisas já estão subentendidas (parar na faixa, esperar sinal, etc.). Entende a diferença? Podemos transpor isso para o código. É uma mentalidade (paradigma) diferente de pensar em programação.

Vamos a outro exemplo: sabe quando você já faz tudo no automático, como dirigir? É sobre isso. Quando você está aprendendo a dirigir, você tem que pensar em todos os passos. Dirigir significa:

1. Ligue o carro
2. De a seta
3. Olhe no retrovisor e para frente
4. Aperte embreagem
5. troque de marcha
6. Saia com o carro

Depois q vc sabe dirigir e está tudo no automático, basicamente é:

- Dirija

O **como** é o passo a passo e o **o que** é o declarativo.

Trazendo um exemplo simples e real de código declarativo, é o **HTML.**

HTML é uma linguagem declarativa, pois você diz o que você quer e não o passo a passo para chegar.

Você diz que quer um cabeçalho h1, uma tabela, um parágrafo, um botão, mas não tem que construir o passo a passo para criar um h1, um botão, etc.

Diferente de como criamos um h1 no JavaScript, que é o passo a passo, veja:

```
const h1 = document.createElement('h1');
h1.classList.add('titulo');
h1.textContent = 'Olá mundo';

document.body.appendChild(h1);
```

Fonte: autoral

O exemplo acima é o passo a passo para chegar em um título de cabeçalho h1 com classe *titulo* e texto ‘Olá mundo’

Diferente do HTML que é mais direto ao que queremos:

```
<body>
  <h1 class="titulo">Olá mundo</h1>
</body>
```

Fonte: autoral

Como aplicar na prática o que aprendeu

A própria documentação mostra algumas formas de começar o trabalho com React, seja através das toolchains ou através de link de CDN.

Vamos aprender mais sobre toolchain nas próximas aulas, a ideia aqui é começar bem passo a passo e iniciar pelos links da CDN, acesse o link https://pt-br.reactjs.org/docs/cdn-links.html **(acesso em 14/10/2022)** e copie os scripts de desenvolvimento do React.

Veja o exemplo de um HTML com React sendo adicionado através dos scripts CDN:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Descomplica | React</title>
</head>
<body>
  <script crossorigin
src="https://unpkg.com/react@17/umd/react.development.js"></script>
  <script crossorigin src="https://unpkg.com/react-dom@17/umd/react-
dom.development.js"></script>
</body>
</html>
```

Fonte: autoral

Dessa forma, já temos uma configuração mínima rodando o React para podermos desbravar a biblioteca.

Você pode adicionar uma outra tag script logo abaixo das importações do React para escrever os códigos da próxima aula.

Veja que os scripts React que adicionamos via CDN, são dois: React e ReactDOM. O React em si é onde está toda a funcionalidade da biblioteca, como os componentes, gerenciamento de eventos, virtual dom e muito mais. O ReactDOM basicamente pega seu componente React e renderiza no navegador.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Descomplica | React</title>
</head>
<body>
  <div id="app"></div>
  <script crossorigin
src="https://unpkg.com/react@17/umd/react.development.js"></script>
  <script crossorigin src="https://unpkg.com/react-dom@17/umd/react-
dom.development.js"></script>
  <script>
    // Código React aqui
  </script>
</body>
</html>
```

Fonte: autoral

Também adicionei uma *div* com *id app* para renderizar nosso componente React ali dentro.

Conteúdo bônus

Tópicos avançados

Baseado em componentes

Algo avassalador do React foi trazer o conceito de componentes.

Os componentes são elementos HTML que constroem uma unidade que pode ser reutilizável e customizável em diversos lugares. Quase tudo no React são componentes. Só não afirmamos tudo, pois agora também temos o conceito de hooks, que não são componentes, mas são criados para serem utilizados dentro de componentes.

Você pode criar um componente de tabela e reutilizar esse componente em todo o seu projeto, como se o componente fosse uma tag HTML que aceita atributos.

Sistema reativo

Característica que popularizou o React é ter um sistema reativo e que vem de encontro com a forma declarativa de desenvolver com React.

Resumidamente, no sistema reativo do React, sempre que uma informação muda, você não precisa manualmente forçar a atualização na tela, pois o React faz isso automaticamente para você. Então, sempre que um dado altera, o React muda na tela/interface para você.

Você pode conhecer mais sobre o sistema reativo aprendendo sobre design patterns como **observer** e **pub/sub**.

Documentação

Site oficial do React: https://pt-br.reactjs.org/ **(acesso em 14/10/2022)**

Tutoriais - passo a passo da documentação:

https://pt-br.reactjs.org/tutorial/tutorial.html **(acesso em 14/10/2022)**

Referência Bibliográfica

SILVA, M. S. **React** - Aprenda Praticando: Desenvolva Aplicações web Reais com uso da Biblioteca React e de Seus Módulos Auxiliares. Novatec Editora, 2021.

STEFANOV, S. **Primeiros passos com React:** Construindo aplicações web. Novatec Editora, 2019.

descomplica

Exercícios

1. Dentre os principais fatores que permitiram a criação do React, podemos destacar:

a) A criação de novos paradigmas de programação

b) Evolução do desenvolvimento Web permitindo criação de interfaces ricas

c) Concorrência entre navegadores modernos

d) Devices com telas maiores

e) Internet das Coisas

2. As principais características do React são:

a) Nível de segurança elevado e baseado componentes

b) Fortemente tipado, baseado em componentes sistema reativo

c) Mobile first, tipagem e sistema reativo

d) Declarativo, baseado em componentes e sistema reativo

e) Declarativo, nível de segurança elevado e mobile first

3. O que é o desenvolvimento declarativo que o React possui?

a) Desenvolver pensando em classes e objetos

b) Desenvolver pensando em resultado “o que” e não no passo a passo “como”

c) Necessário declarar muitas variáveis

d) Desenvolvimento focado na experiência do usuário

e) Desenvolvimento “facilitado” pelo uso de decorators

4. O que são componentes visuais no React?

a) São peças físicas necessárias para codificar em React

b) Conjunto de recursos que juntos foram o React

c) São funções disponibilizadas pelo React para uso na biblioteca

d) São os algoritmos utilizados na renderização da página

e) São unidades de elementos de uma página/site/sistema

Gabarito

1. Letra B. O React veio para ajudar e otimizar a construção de interfaces ricas e interativas, melhorando a experiência do desenvolvedor e também do usuário final que vai acessar o site, já que uma das premissas da biblioteca é a performance.

2. **Letra D.** Vimos que o React trouxe algumas inovações em sua biblioteca: desenvolvimento declarativo, baseado em componentes e sistema reativo. Essa tríade fez o React se popularizar, hoje “copiado” por diversas outras bibliotecas e frameworks.

3. **Letra B.** Com React podemos descrever interfaces “HTML” no JavaScript de forma declarativa, que é uma forma muito mais natural e direta do que utilizar a DOM API (do JavaScript puro) onde temos que fazer passo a passo “como” queremos a interface.

4. **Letra E.** O React é todo baseado em elementos visuais da web, sejam cards, navbars, banners etc. Todos esses são componentes visuais.