Introdução ao React

FUTURE4

Sumário

O que vamos ver hoje? 👀

- Hoje temos 4 tópicos importantes:
  - *Node.js*
  - *Por que React?*
  - *Primeiro projeto em React*
  - *JSX*

Node.js

O que é? 🤔

- O *Node.js* é um facilitador muito grande na vida de um desenvolvedor de ***Javascript***

- Precisa de **um comando só** para rodar um código em *Javascript*, usando *Node.js*

- Desta forma, conseguimos rodar tudo que fica na tag **<script>** sem usar um **browser** nem a **internet**

O que é? 🤔

Vamos ver na prática! 🔬

FUTURE4

O que é? 🤔

- Além de rodar no terminal diretamente, podemos pedir para ele **rodar um arquivo**

- Para isso, rodamos o comando:

```
➜  ~ node <path do arquivo>.js
```

Vamos ver na prática! 🔬

FUTURE4

Atenção !!

- **Para entender bem o *Node.js*, vocês precisam ter mais aprofundamento técnico, então não explicaremos com detalhes nesta aula!**

- **Pela nossa programação, teremos, ao menos, uma aula inteira dedicada para isso**

Libs 

- Mas a principal vantagem do *Node.js* é a facilidade em **gerenciar as dependências/bibliotecas** do projeto

- **Biblioteca/Library (lib) :** é um conjunto de códigos disponibilizados, para que os desenvolvedores possam aproveitar funcionalidades já criadas

Libs 

- Basicamente é um conjunto de classes, funções e variáveis

- Existem várias **libs** para **JS**:
  - moment.js
  - clipboard.js
  - progressbar.js

```
➜ Future4 npm install <nome da dependencia/biblioteca>
```

Libs de Web 

- Elas existem para facilitar a nossa vida na hora de desenvolver sites

- Aqui estão as mais famosas:
  - ReactJS
  - Angular
  - Vue.js
  - jQuery

Por que React?

Por que React? 🧐

- **Ponto I:** Muito utilizada
- **Ponto II:** Simplicidade
- **Ponto III:** Equipe capacitada
- **Ponto IV:** Reatividade e Componentização
- **Ponto V:** Fácil de testar

Ponto I: Muito utilizada 🤩

- É, atualmente, MUITO utilizado no **mercado** para se criar sites

- Expandindo um pouco, é até para desenvolvimento **mobile** (React Native/PWA)

- **Comunidade grande** é sempre muito bom

Ponto II: Simplicidade 👶🏽

- O React permite que criemos as aplicações usando **só** JS

- Conseguimos fazer tudo em **um só** arquivo, em JS

- Além do fato de JS ser uma linguagem muito **poderosa**

Ponto III: Equipe capacitada 👩🏾‍🚀

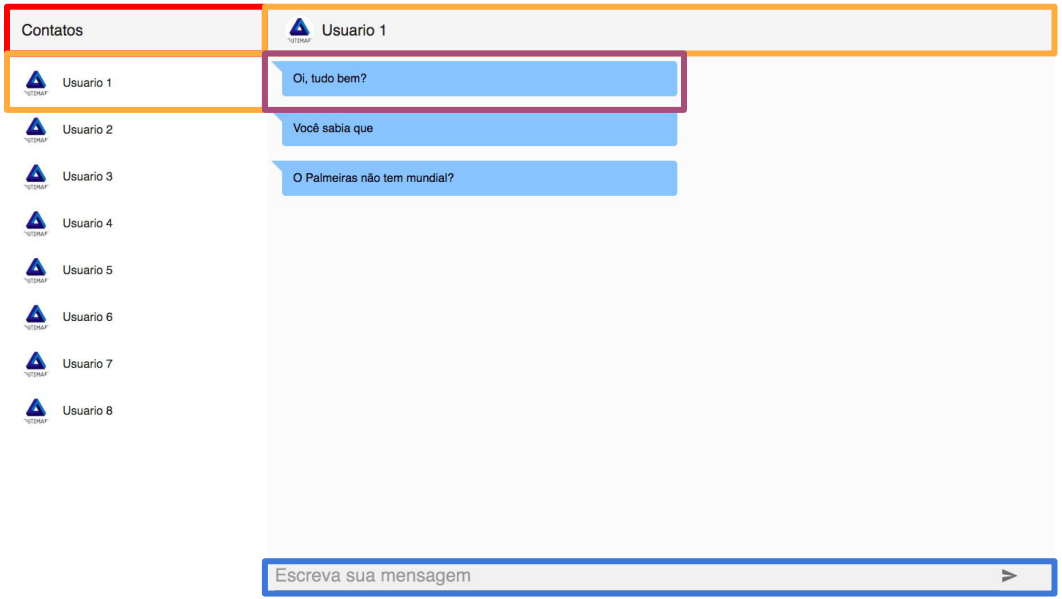
- Feito por uma equipe capacitada: **Facebook**
- Atualmente, o Facebook é um local em que muitos programadores desejam trabalhar
- Eles desenvolvem **outras** frentes, além do próprio produto

Ponto IV: Reatividade e Componentização🧪

- Componentização
  - Uma das principais vantagens do React é que ele permite que façamos **componentes**

  - Além disso, ele **nos incentiva** a pensar desta forma

Ponto IV: Reatividade e Componentização 🧪



FUTURE4

Ponto IV: Reatividade e Componentização🧪

- Reatividade
  - O conceito de reatividade, para sites, é bem complexo

  - Mas, em poucas linhas, isso quer dizer que é muito fácil atualizar todas as partes do site quando a informação muda

Ponto V: Fácil de testar 

- Vocês vão entender isso com mais calma depois

- **Testes** são essenciais em um projeto de programação

- Há maneiras de **automatizar** os testes
  - Fazer testes automatizados só com HTML e JS é **bem difícil**
  - O React **facilita** muito isso

Pausa para relaxar

5 min

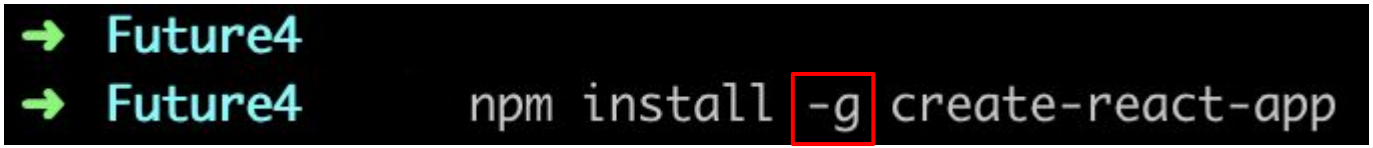
FUTURE4

Criando um projeto React

Instalando o React 😎

- Antes de criar um app usando React, precisamos **instalar** o React em nossas máquinas

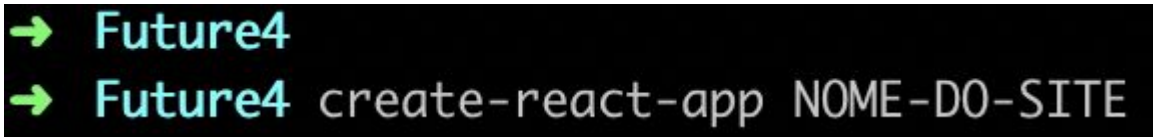
- Como o React é uma **dependência** de Javascript, o *Node.js* pode nos ajudar a instalar



Criando um Projeto de React 🏗️

- Uma maneira de **instalar** e **criar** um **app de react** é utilizando o comando **npx**

- Para criar um app em *React*
  - Precisamos **navegar** até a pasta que queremos (no terminal)
  - Digitar o seguinte comando



Criando um Projeto de React 

- Agora, vamos entender tudo que o comando fez
  - node_modules
  - public
    - index.html
  - src
    - index.css
    - index.js
    - **App.css**
    - **App.js**
  - package.json

Criando um Projeto de React 🏗️

- Resumo:
  - *App.js*: Onde vai ficar o **código principal** do React de vocês
  - *App.css*: Onde pode ficar o código principal do **CSS** de vocês
  - *index.js* e *index.css*: São arquivos criados automaticamente que **dificilmente** vamos mexer

Criando um Projeto de React 🏗️

- Resumo:
  - *npm run start*: É o comando que permite **rodar** o nosso app em React

  - *npm run build*: É o comando que permite **criar**/**buildar** o nosso app

Pausa para relaxar

5 min

FUTURE4

JSX

A sintaxe do React ✍🏾

- **JSX** é uma sintaxe que nos permite escrever códigos misturando JS, HTML, CSS
  - Assim a **lógica** e o **layout** ficam no mesmo lugar

- Conseguimos, facilmente, colocar lógica dentro dos nossos elementos

- Mas ele tem uma condição: os layouts só podem ter **UM** elemento pai

A sintaxe do React ✍🏾 - className

App.js

Um pai só

```
5   function App() {
6     return (
7       <div className="App-principal">
8
9         <div className="App-secundario">
10          Oi, sou uma div feita com CSS
11        </div>
12
13        <div className="App-secundario">
14          Oi, sou outra div feita com CSS
15        </div>
16
17        <div className="App-secundario">
18          Olaaaaaa (quis só ser diferentona)
19        </div>
20
21      </div>
22    )
23  }
```

App.css

```
1   .App-principal {
2     padding-top: 10px;
3     padding-left: 10px;
4   }
5
6   .App-secundario {
7     width: 200px;
8     border: 1px solid ■black;
9     margin-bottom: 10px;
10  }
```

FUTURE4

A sintaxe do React ✍🏾 - id

App.js

```
5   function App() {
6     return (
7       <div className="App-principal">
8
9         <div id="primeiraDiv" className="App-secundario">
10          Oi, sou uma div feita com CSS
11        </div>
12
13        <div className="App-secundario">
14          Oi, sou outra div feita com CSS
15        </div>
16
17        <div className="App-secundario">
18          Olaaaaaa (quis só ser diferentona)
19        </div>
20
21      </div>
22    )
23  }
```

App.css

```
1   .App-principal {
2     padding-top: 10px;
3     padding-left: 10px;
4   }
5
6   .App-secundario {
7     width: 200px;
8     border: 1px solid black;
9     margin-bottom: 10px;
10  }
11
12  #primeiraDiv {
13    background: violet;
14  }
```

FUTURE4

A sintaxe do React ✍🏾 - JS

```
1   import React from 'react';
2   import './App.css';
3
4   function App() {
5     const mensagem = "Olá Goli, bom dia!"
6     return (
7       <div className="App">
8       { mensagem }
9       </div>
10    );
11  }
```

FUTURE4

A sintaxe do React ✍🏾 - JS

- Dentro destes colchetes, você **NÃO PODE**
  - colocar if
  - declarar variável
  - colocar loops

- Basicamente, dá para colocar só "expressões javascript"
  - chamadas de funções
  - valores que podem ser diretamente aplicados a variáveis

A sintaxe do React ✍🏾 - Eventos

- Para eventos, nós temos que passar a **referência da função**

```
5   function clickNaDiv() {
6     console.log("CLICADO")
7   }
8   function App() {
9     return (
10      <div onClick={ clickNaDiv }>
11        DIV CLICAVEL
12      </div>
13    );
14  }
15
```

Vamos ver na prática! 🔬

FUTURE4

A sintaxe do React ✍🏾 - Eventos

- Alguns eventos:
  - **onClick**
  - **onBlur**
  - **onKeyPress**
  - **onChange**

Formulários em React - 🤵

- Existem várias maneiras de lidar com formulários em React

- Vamos ensinar agora uma que utiliza o evento onSubmit

Formulários em React - 🤵

```
15  function App() {
16    return (
17      <div className="App">
18        <form onSubmit={ onSubmit }>
19          <input name="nome" placeholder="nome"/>
20          <input name="idade" placeholder="idade"/>
21          <input name="nomeCachorro" placeholder="nome do cachorro"/>
22          <input name="rickyAndMorty" placeholder="ricky and morty?"/>
23          <button type="submit">ENVIAR</button>
24        </form>
25      </div>
26    );
27  }
```

Vamos ver na prática! 🔬

FUTURE4

Dúvidas?

FUTURE4

Obrigado!

FUTURE4