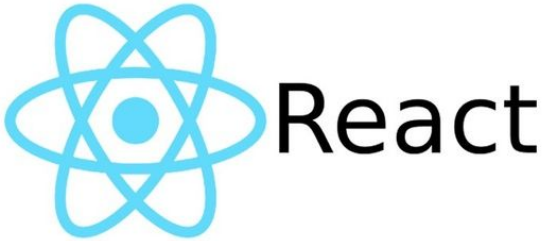
centro universitário FEI

CCP150
DES. DE APLICATIVOS MÓVEIS

Prof. Isaac
isaacjesus@fei.edu.br

DESENVOLVIMENTO DE APLICAÇÕES MOBILE

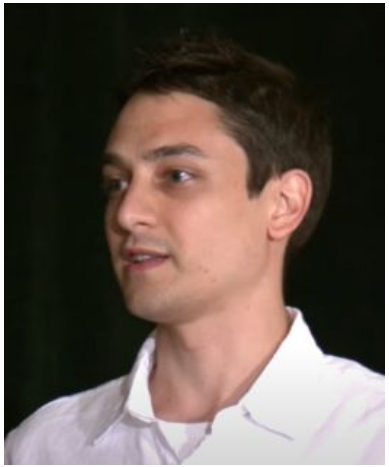


- **Biblioteca JavaScript para criar interfaces com o usuário**
- React faz com que a criação de UIs interativas seja uma tarefa fácil
- React irá atualizar e renderizar de forma eficiente apenas os componentes necessários na medida em que os dados mudam

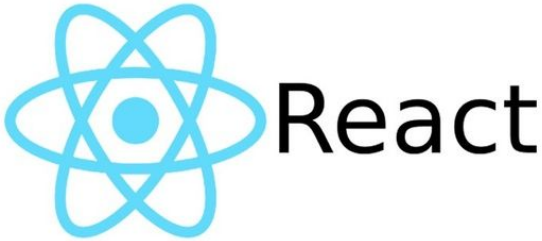
https://react.dev/

React

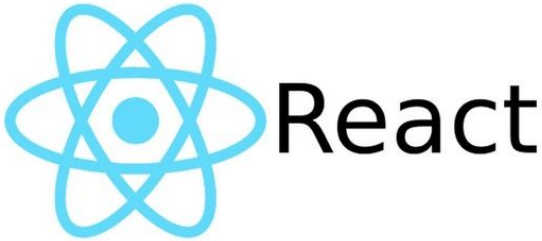
- Desenvolvido por **Jordan Walke (Facebook)**, lançado em 2013
- É mantido pelo Facebook e desenvolvedores independentes (milhões de desenvolvedores)
- Tem sido usado por grandes companhias ao redor do mundo: *Netflix, Airbnb, American Express, Facebook, WhatsApp, eBay e Instagram*



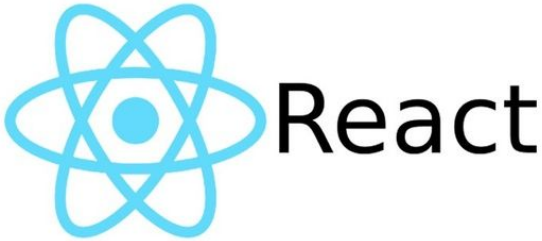
Fonte: https://www.hostinger.com.br/tutoriais/o-que-e-react-javascript



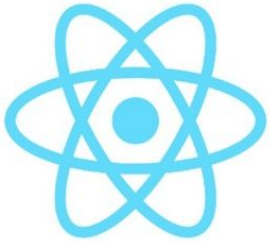
- O React é fácil de usar!
- Ele torna a experiência do usuário com a interface mais eficiente
- Ele pode ser categorizado como o **“V” no padrão MVC** (Model-View-Controller)
- React é usado para construir **aplicativos de página única (*Single-Page Application - SPA*)** - um aplicativo que tem apenas uma página HTML



- O React possibilita que **HTML seja escrito no JavaScript**
- Para isso, o React utiliza a ideia de **elementos**:
  - Elementos são os menores blocos de construção de aplicativos React
  - Um elemento descreve o que você quer ver na tela



- O React pode ser renderizado no servidor, usando **Node.js**
- Pode também ser usado para criar **aplicações mobile**, através do **React Native**



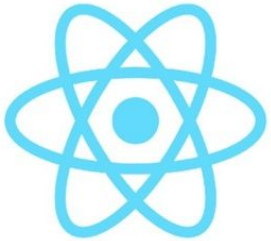
React

- Vamos iniciar uma **Aplicação React (Web)**:
  - Para fazer isso localmente, precisamos ter o Node.js instalado!
  - No terminal, utilizamos o seguinte comando:

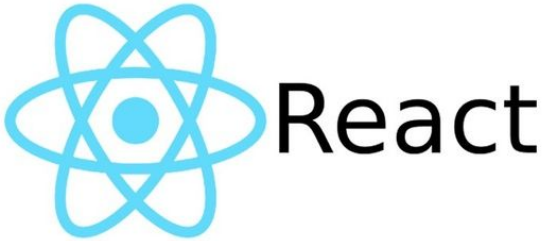
    ```
    npx create-react-app my-website
    ```

  - Então, `cd my-website`

    ```
    npm start
    ```

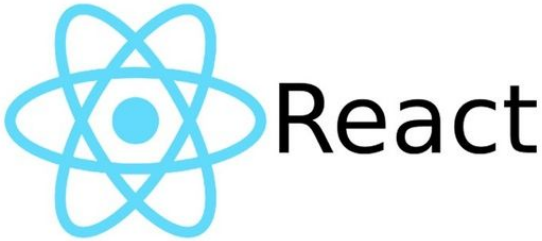


- ***NPX*** = *package runner* do ***NPM (Node Package Manager)***
- ***NPX = executa os pacotes***
    - *Busca a biblioteca, instala em uma pasta temporária, executa o comando e remove a biblioteca da máquina (não fica no* ***node_modules*** *global)*

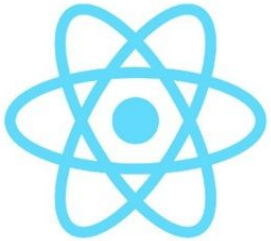


- Árvore de diretórios do projeto criado:

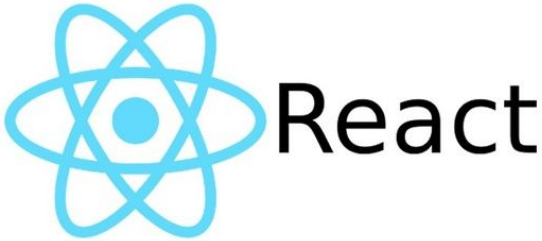
```
└── my-website
    ├── node_modules
    ├── package.json
    ├── public
    │   ├── favicon.ico
    │   ├── index.html
    │   ├── logo192.png
    │   ├── logo512.png
    │   ├── manifest.json
    │   └── robots.txt
    ├── README.md
    ├── src
    │   ├── App.css
    │   ├── App.js
    │   ├── App.test.js
    │   ├── index.css
    │   ├── index.js
    │   ├── logo.svg
    │   ├── serviceWorker.js
    │   └── setupTests.js
    └── yarn.lock
```



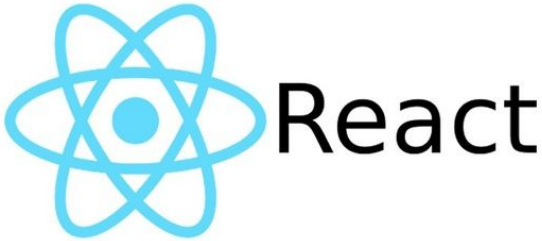
- Para continuar nossa aplicação, podemos apagar tudo da pasta ***src*** e da pasta ***public***



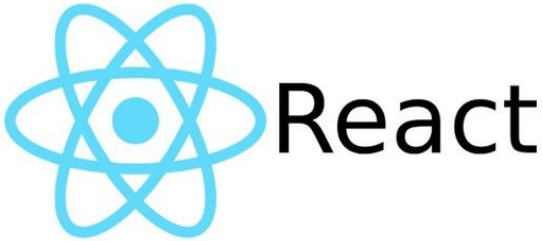
- O React é baseado em **componentes**!
  - Componentes permitem que você divida a UI em partes **independentes**, **reutilizáveis** e que você pense em cada parte isoladamente
  - **Ideia chave:** Encapsulamento de HTML em React



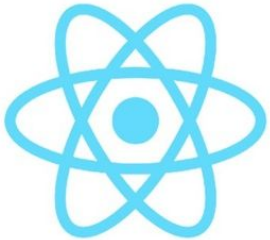
- O React é baseado em **componentes**!
  - Eles têm a mesma finalidade que as **funções JavaScript**
    - Aceitam **entradas arbitrárias**: chamadas ***props***
    - **Retornam elementos React** que descrevem o que deve aparecer na tela



- Existem **dois tipos** de componentes:
  - **Componentes de Função** ***(Function Components)***
  - **Componentes de Classe** ***(Class Components)***



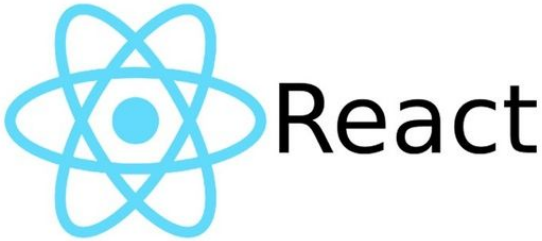
- Vamos começar definindo um **componente como uma função JavaScript**!
  - Vamos criar um arquivo **HTML** na pasta ***public***
  - Vamos criar um arquivo **JS** na pasta ***src***



React

- **HTML:**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>React</title>
</head>
<body>
 <div id="root"></div>
</body>
</html>
```

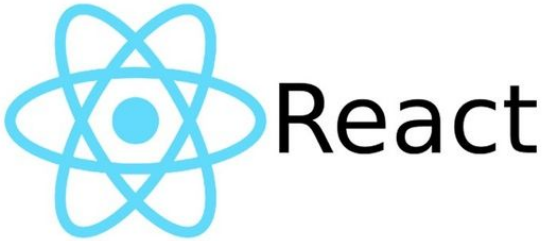


- **JS:** Começamos importando ***React*** e ***ReactDOM***

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

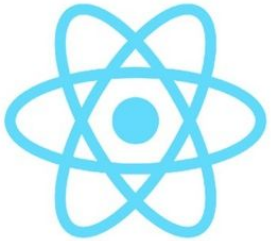
DOM - Document Object Model

- *interface que representa como os documentos HTML e XML são lidos pelo browser*



- **JS: Componente de Função**: retorna um **elemento React**

```
function Welcome() {
    return React.createElement(
        "div",
        null,
        "Olá Mundo!"
    );
}
```

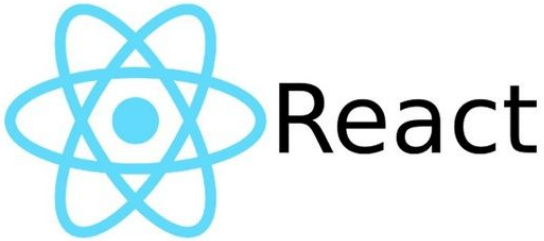


React

- **JS:** Renderiza o elemento criado no HTML -> id ‘root’

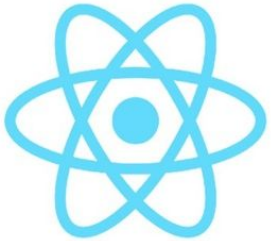
```
ReactDOM.render(
    React.createElement(Welcome),
    document.getElementById( elementId: 'root')
);
```

https://codepen.io/danilo-perico/pen/PozYbee?editors=1010



- Podemos desenvolver a mesma aplicação com **Componentes de Classes**

```
class Welcome extends React.Component {
    render() {
        return React.createElement(
            "h1",
            null,
            "Olá Mundo!"
        );
    }
}
```



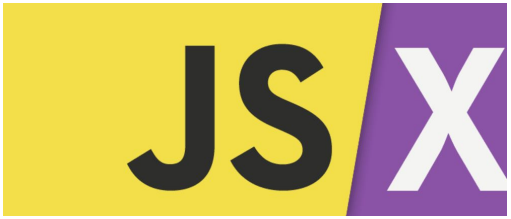
React

- JS: Renderiza o elemento criado no HTML -> *id ‘root’*

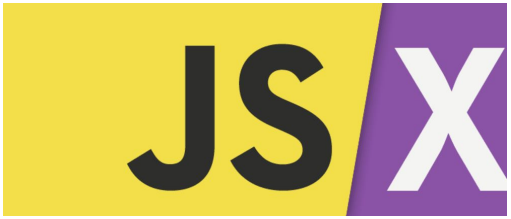
```
ReactDOM.render(
    React.createElement(Welcome),
    document.getElementById( elementId: 'root')
);
```

https://codepen.io/danilo-perico/pen/KKMPNbj?editors=1010

JSX



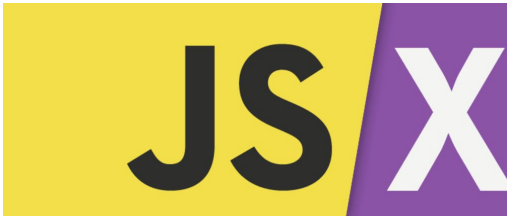
- **JSX - JavaScript + XML**
- Permite escrever código HTML dentro do JavaScript
- **Torna bem mais fácil o processo de escrever e adicionar HTML no React!**
- JSX transforma as **tags HTML** em **elementos React**



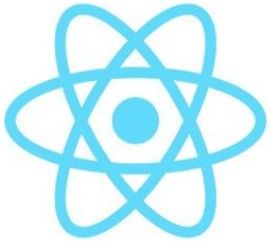
- Com o JSX, podemos utilizar algo desse tipo:

```
const element = <h1>Hello, world!</h1>;
```

- Essa sintaxe não é uma String, nem HTML
- **Isso é JSX!**



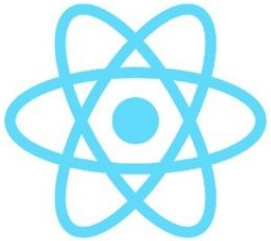
- O JSX é simples!
- É declarativo! Descreve o que deve ser renderizado, mas não como vai ser renderizado
- **O JSX é uma das melhores maneiras de descrever estruturas complexas da interface de usuário**



+

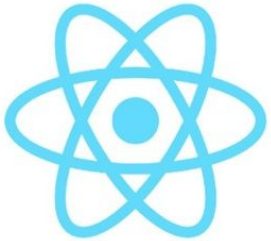
- ***ReactDOM.render()*** renderiza a marcação JSX e coloca o conteúdo em um nó DOM
- Exemplo:

https://codepen.io/danilo-perico/pen/QWELGYP?editors=1010



React +

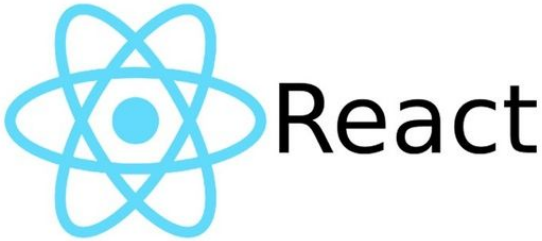
- Podemos usar o JSX nos nossos **componentes**!



React

- **JS: Componente de Função**

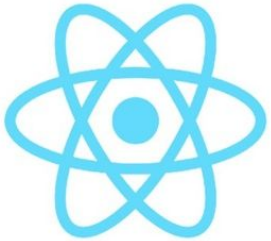
```
function Welcome(props) {
    return <h1>Olá Mundo!</h1>;
}

const element = <Welcome/>;
ReactDOM.render(element,
    document.getElementById( elementId: 'root'));
```



- **JS: Componente de Função**: retorna um **elemento React**

https://codepen.io/danilo-perico/pen/RwRbVjZ?editors=1010

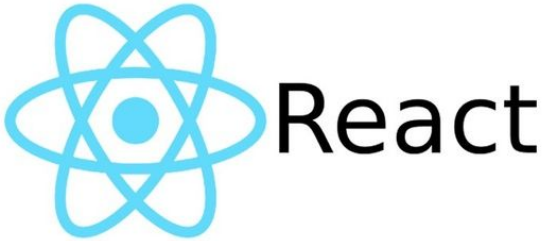
https://codepen.io/danilo-perico/pen/jOrNmKJ?editors=1010



React

- Mesma aplicação com **Componente de Classe**

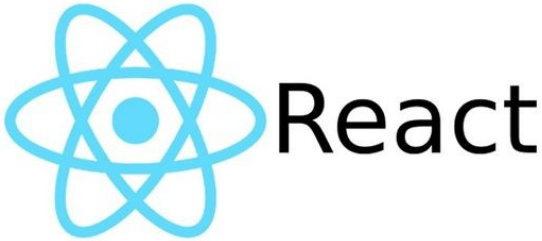
```
class Welcome extends React.Component {
    render() {
        return <h1>Olá Mundo</h1>;
    }
}

ReactDOM.render(<Welcome/>,
    document.getElementById( elementId: 'root'));
```



- **Componente de Classe**

https://codepen.io/danilo-perico/pen/LYZPyQb?editors=1010

https://codepen.io/danilo-perico/pen/RwRbVqx?editors=1010

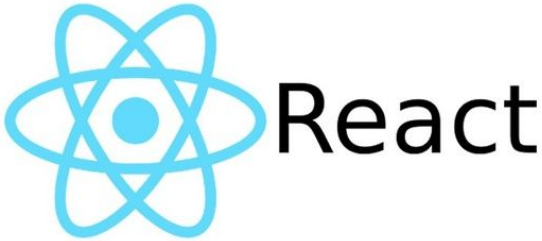


- **Usando 2 ou mais Componentes**

```
class App extends React.Component {
  render() {
    return (
      <div>
        <Classe01 />
        <Classe02 />
      </div>
    );
  }
}

ReactDOM.render(<App/>, document.getElementById('root'));
```

```
class Classe01 extends React.Component {
    render() {
        return <h1>Primeiro Componente</h1>;
    }
}

class Classe02 extends React.Component {
    render() {
        return <h1>Segundo Componente</h1>;
    }
}
```

https://codepen.io/Isaac-Jesus-Silva/pen/jOXQrZL?editors=1010



Usos de **Estilos:**

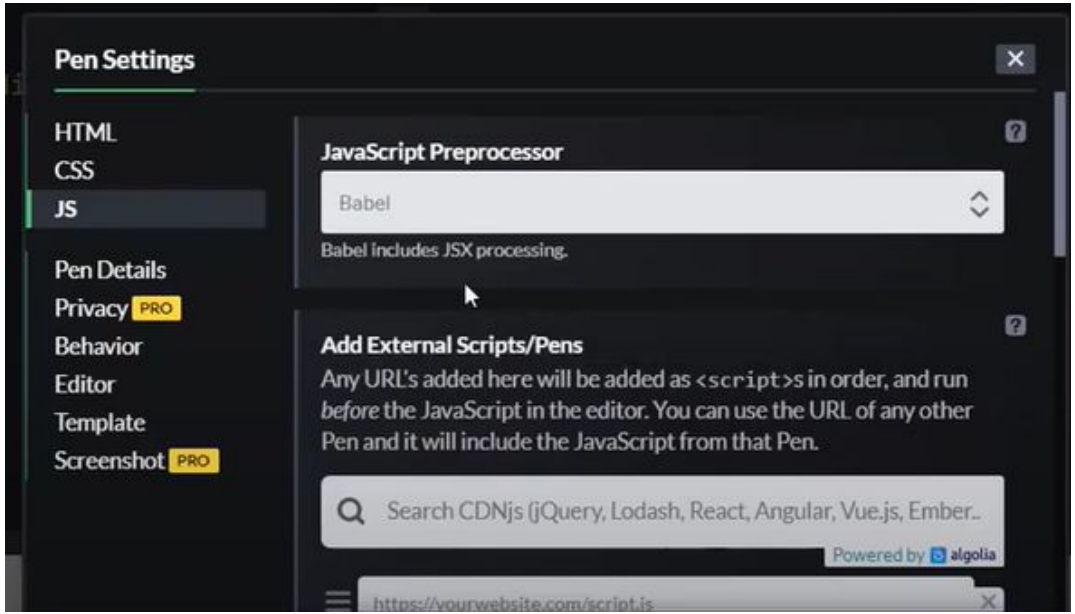
- Objeto com informações do estilo.
- Inline.

No CSS dentro do Javascript use ***camelCase*** ao invés de hífen.

```
class MeuComponente extends React.Component{
//Definição do componente vai aqui
  render(){
    const mystyle = {
      color: "white",
      backgroundColor: "DodgerBlue",
      padding: "10px",
      fontFamily: "Arial"
    };
    return (
      <section style={mystyle}>
      <h1>Componente</h1>
      <p> O HTML vai aqui!!!</p>
      <p style={{color:"red"}}>Podemos usar CSS!!!</p>
      </section>
    )
  }
}
```

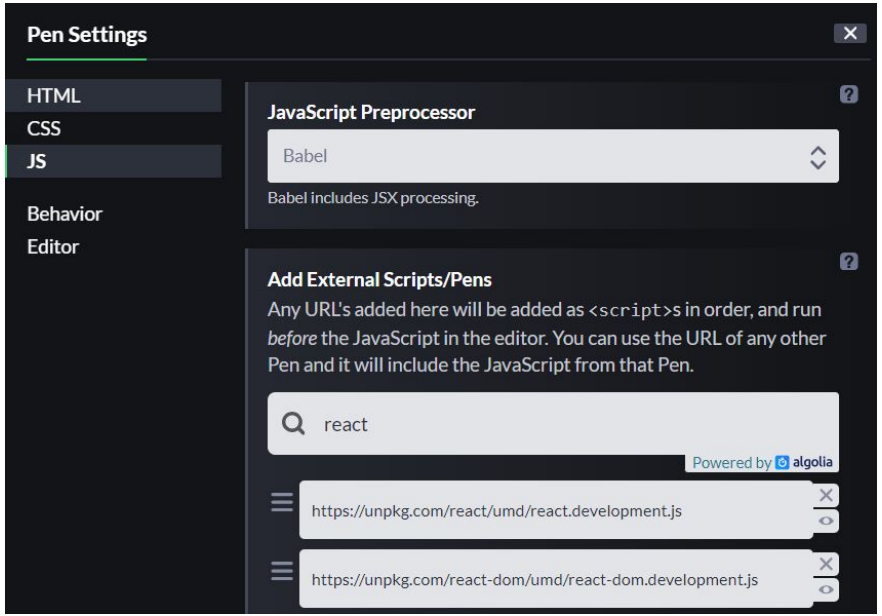
React no codepen

Para usar o React no codepen use Babel.

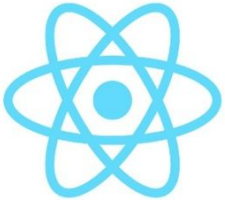


React no codepen

Adicione React e ReacDOM no *Add External Scripts/Pens*



PRÁTICA



+

1. Reproduza a página ao lado com ***React + JSX***

- ***Faça um código com função.***
- ***Faça um código com classe.***

Computação Móvel

React

Biblioteca JavaScript para criar UI

Aula Um

Vamos utilizar:

- HTML
- CSS
- JavaScript