IOS – Instituto de Oportunidade Social

React 01 - Introdução ao React

Objetivos

- React
- JSX
- Renderização de elementos React
- Componentes
- Props

IOS – Instituto de Oportunidade Social

React

React

> Hooks

React é uma biblioteca usada para a construção de interfaces. Ou seja, ele é utilizado para o desenvolvimento de aplicações do lado do cliente. O React é frequentemente chamado de um “framework” para Frontend, pois possui capacidade semelhantes e é comparado com os frameworks Angule e Vue.

React

Single Page Application (SPA, aplicação de página única).

X

Múltiplas páginas (MPA)

React

Utilizar SPA no desenvolvimento de aplicações web é vantajoso, pois o gerenciamento da interface é realizado do lado do cliente e as consultas no servidor só são necessárias para buscas novos dados e não páginas inteiras. Por isso, as aplicações desenvolvidas com SPA fazem as transições entre páginas mais rápidas e o usuário tem a sensação de estar navegando por um aplicativo nativo

IOS – Instituto de Oportunidade Social

JSX

JSX

O React utiliza uma sintaxe chamada JSX (JavaScript Syntax Extension), que é uma extensão da sintaxe do JavaScript e a documentação oficial (https://reactjs.org/docs) recomenda a utilização dessa linguagem para desenvolver a sua aplicação. A vantagem de desenvolver a sua aplicação React utilizando o JSX ao invés de JavaScript puro é por ela parecer uma linguagem de marcação que possui todo o poder de desenvolvimento provido pela JavaScript.

JSX

A sintaxe JSX é formada de algo que chamamos de elementos React:

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

```
const nome = 'Irmão do Jorel';

const mensagem = <h1>Olá, {nome}</h1>;
```

Vejamos outro exemplo:

```
const user = {
    firstName: 'Irmão',
    lastName: 'do Jorel',
};

function formataNome(user) {
    return user.firstName + ' ' + user.lastName;
}

const mensagem = <h1>Hello, {formataNome(user)}!</h1>;
```

JSX

Você pode usar atributos nos elementos React da mesma forma que no HTML, por exemplo:

```
const elemento01 = <div tabIndex="0"></div>;
const elemento02 = <img src={user.avatarUrl}></img>;
```

JSX

Importante: Como o JSX está mais próximo do JavaScript do que do HTML, o React DOM usa a convenção camelCase de nomenclatura de propriedade em vez de nomes de atributos HTML. Sendo assim, o atributo class do HTML torna-se className em JSX e tabindex torna-se tabIndex.

Apenas um elemento pai:

Sintaxe inválida

```
// Código (a)
const mensagem = (
    <h1>Olá!</h1>
    <h2>Como é bom ver vocês.</h2>
  );
```

Sintaxe válida

```
// Códido (b)
const mensagem = (
    <div>
        <h1>Olá!</h1>
        <h2>Como é bom ver vocês.</h2>
    </div>
);
```

JSX

A sintaxe JSX também permite criar uma marcação vazia como pai de várias marcações filhas. O código abaixo mostra outra sintaxe válida para a criação do elemento React.

```
const mensagem = (
    <>
        <h1>Olá!</h1>
        <h2>Como é bom ver vocês.</h2>
    </>
);
```

Sintaxe válida com marcação vazia.

IOS – Instituto de Oportunidade Social

Renderização de elementos React

Renderização de elementos React

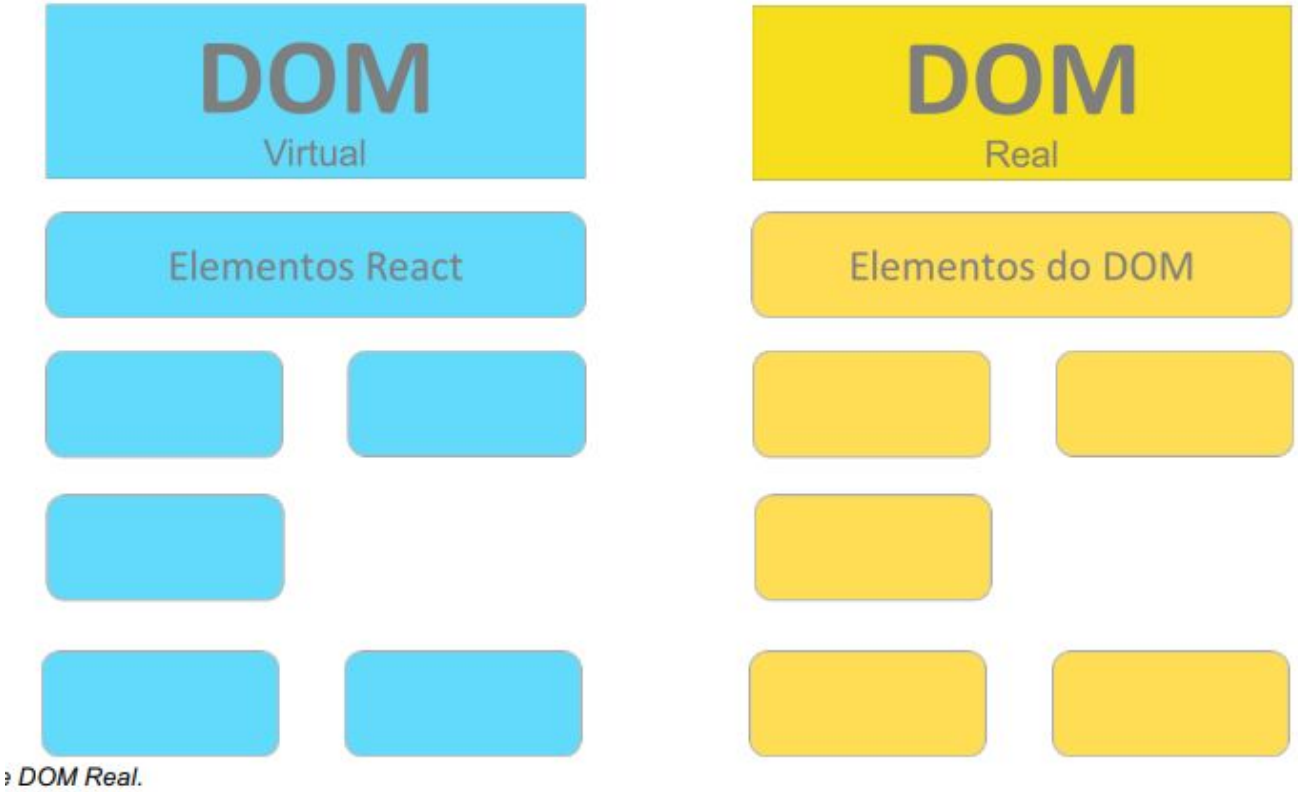
Um elemento React descreve o que você visualiza na interface que está sendo desenvolvida. Diferente de elementos DOM do navegador, elementos React são objetos simples e utilizam menos recursos para serem criados e gerenciados.

Renderização de elementos React

O React possui um DOM Virtual, que é formando pelos elementos React e é responsável por atualizar o DOM do navegador (DOM Real) para exibir os elementos React renderizados como marcações HTML na página web.

Renderização de elementos React

DOM real X Dom virtual



e DOM Real.

Renderização de elementos React

Todo elemento React é renderizado na página web. Renderizar algo, nesse contexto, é transformar uma sintaxe do JSX em algo que vai ser exibido na página web. Por exemplo, é comum que no arquivo index.html de uma aplicação React exista a marcação da seguinte forma:

```
<div id="root"></div>
```

Renderização de elementos React

Essa marcação é chamada de nó raiz do DOM, pois toda a aplicação React será renderizada e gerenciada dentre dessa marcação. Para renderizar um elemento React em um nó raiz como o mostrado anteriormente, deve-se utilizar o método render(). Por exemplo:

```
const mensagem = <h1>Hello, world</h1>;
ReactDOM.render(mensagem, document.getElementById('root'));
```

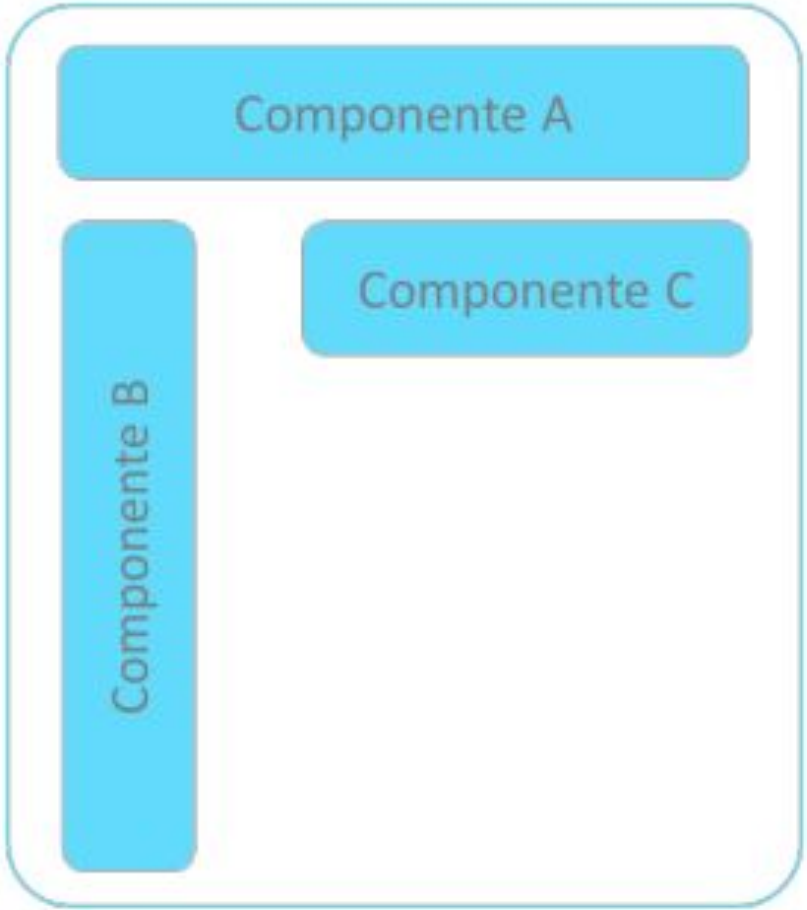
IOS – Instituto de Oportunidade Social

Componentes

Componentes

O coração de toda aplicação desenvolvida em React são os componentes. Os componentes são essencialmente uma parte da interface do usuário e podem ser formados por um ou mais elementos React. Quando estamos construindo aplicação com React estamos construindo vários componentes independentes, isolados e reutilizáveis. E esses componentes juntos são usados para compor uma interface do usuário complexa.

Componentes

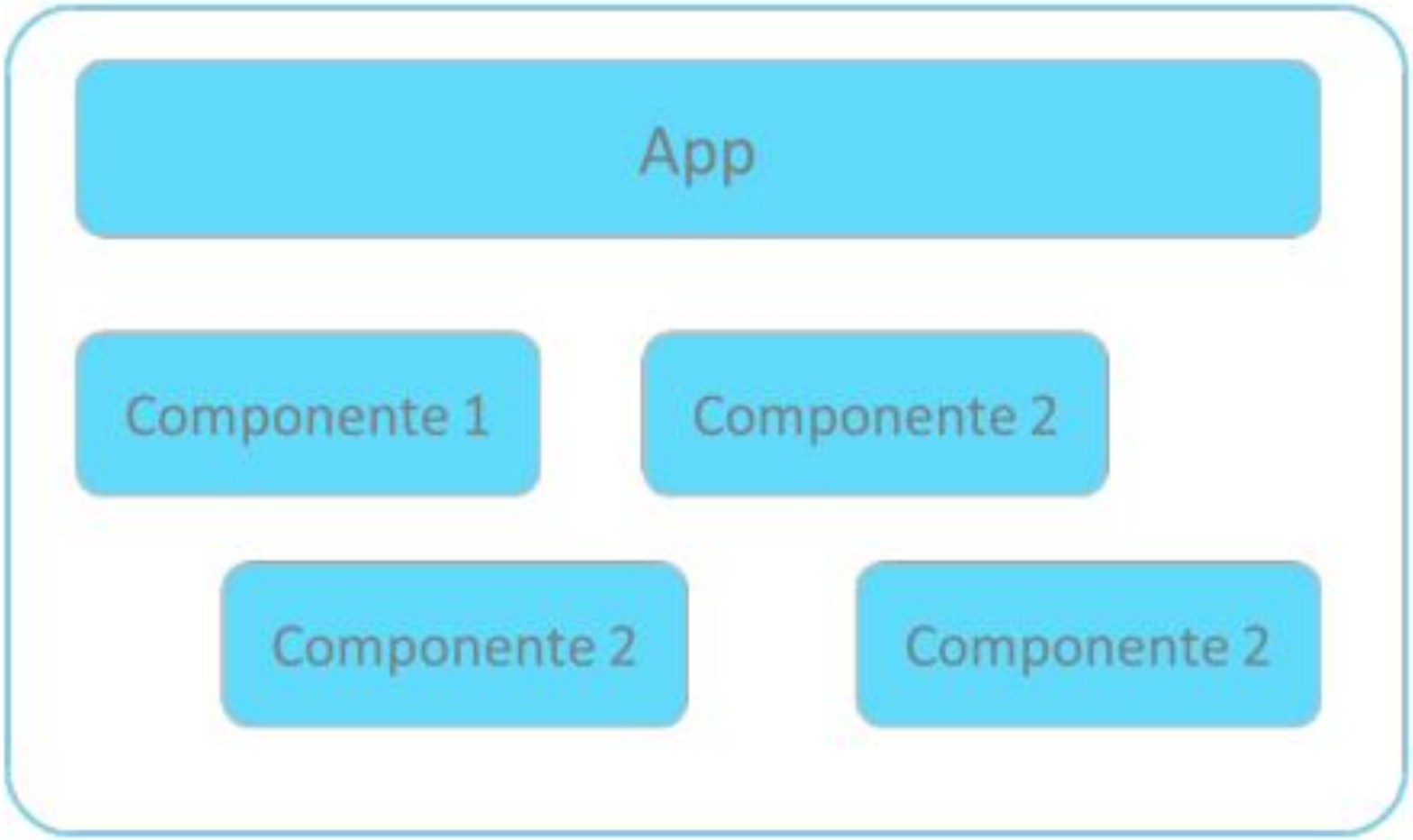


Interface do usuário com diversos componentes

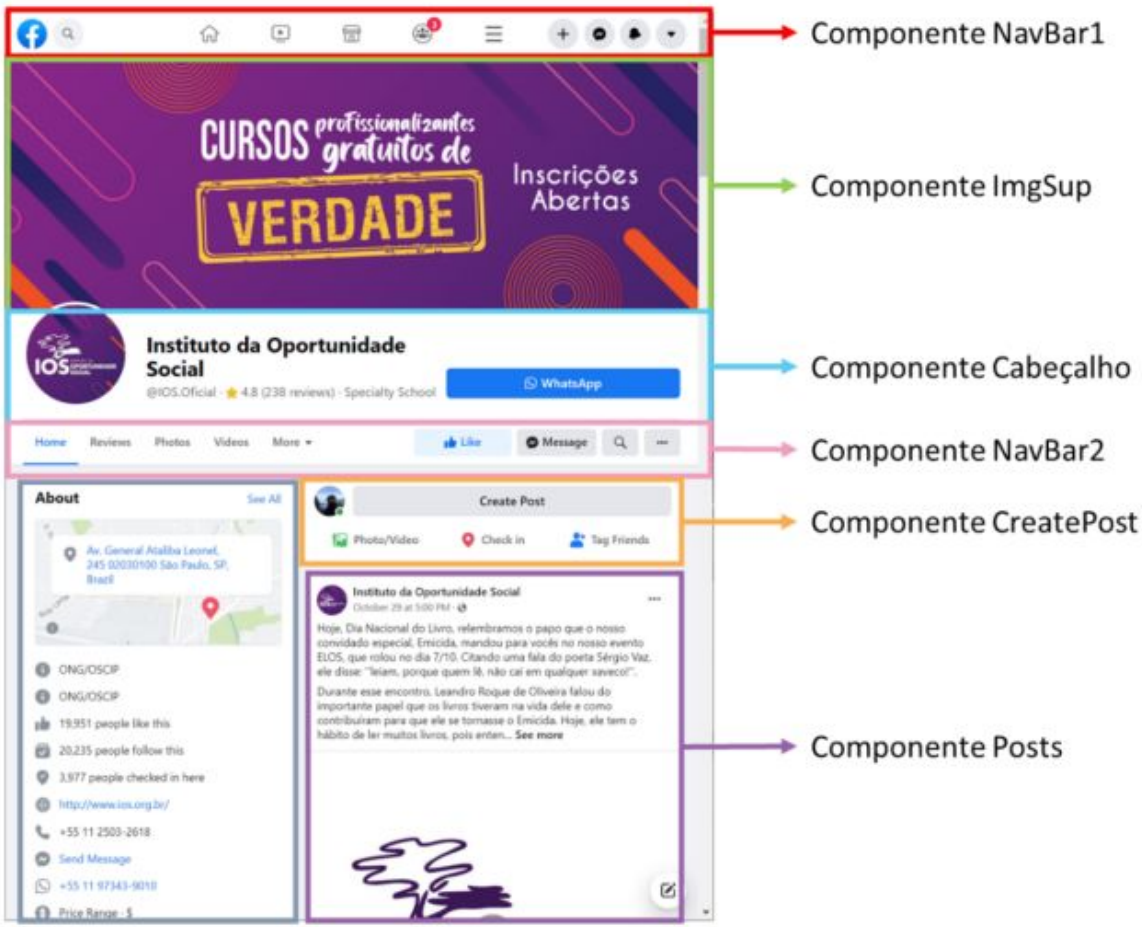
Componentes

Todas as aplicações React têm pelo menos um componente, que é referenciado como componente raiz e na maioria das aplicações que você encontrar o nome desse componente é App. Esse componente representa a aplicação interna e contém outros componentes filhos. Então, toda aplicação React é essencialmente uma árvore de componentes, onde existe um único componente raiz (App), que está no topo da hierarquia dos componentes.

Componentes



Componentes



Componente About

Componentes

Em termos de implementação, o componente pode ser implementado como uma classe ou como uma função do JavaScript. Componentes funcionais foram introdução no React a partir da versão 16.8 no final de 2019 com a inserção do conceito de hooks, até então os componentes eram criados apenas utilizando classes. Atualmente, é bem provável que você irá encontrar em uma empresa uma implementação híbrida, que utiliza a abordagem de componentes com classe e componentes funcionais.

Componentes

```
// Código (a)
function Mensagem() {
    return <h1>Olá, pessoal!</h1>;
}
```

```
// Código (b)
class Mensagem extends React.Component {
    render() {
        return <h1>Olá, pessoal!</h1>;
    }
}
```

Componente Mensagem implementado com a abordagem (a) de func

IOS – Instituto de Oportunidade Social

Props

Props

Componentes podem ter props (Abreviação de propriedades), que é a maneira de passar uma informação para um componente. Props também são usadas para passar dados de um componente pai para um componente filho na hierarquia de componentes.

Props

```
function Mensagem(props) {
    return <h1>Olá, {props.name}</h1>;
}

const elemento = <Mensagem name="Irmão do Jorel" />;

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

Componente Mensagem com uma propriedade name.

Props

Componentes podem ter props (Abreviação de propriedades), que é a maneira de passar uma informação para um componente. Props também são usadas para passar dados de um componente pai para um componente filho na hierarquia de componentes.

IOS – Instituto de Oportunidade Social

Exercício

Exercício

Com suas palavras explique o que é:

React;

Diferença de biblioteca e Framework;

Props;

hooks;

Componentes;

Coloque todas as suas reposta em um arquivo TXT e abaixo de cada resposta insira os links que você utilizou para elaborar e entender o conteúdo suba no GitHub na pasta da aula de hoje e disponibilize o link na plataforma do IOS