

React

<Conceitos>

Linguagens Script @ LEI / LEI-PL / LEI-CE

Departamento de Engenharia Informática e de Sistemas

Cristiana Areias < cris@isec.pt >

2022/2023

Conceitos Principais

- › Componentes
 - › Componentes Funcionais
 - › Componentes de Class
- › JSX
- › Props e State
- › Ciclo de Vida

> React > Componentes

React

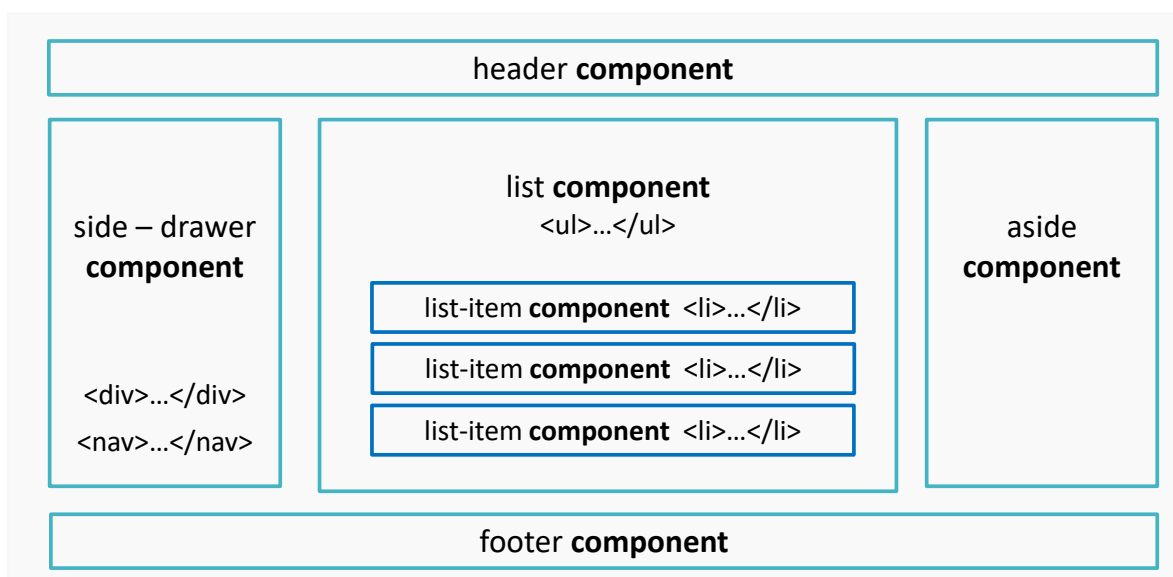
- O interface com o utilizador é composto por componentes (*React Components*)
 - Cada componente pode ser decomposto em outros componentes (arquitetura baseada numa estrutura hierárquica de *Components*)
- Os componentes permitem utilizar a mesma estrutura com outros dados (reutilização)



isec
Politécnico de Coimbra

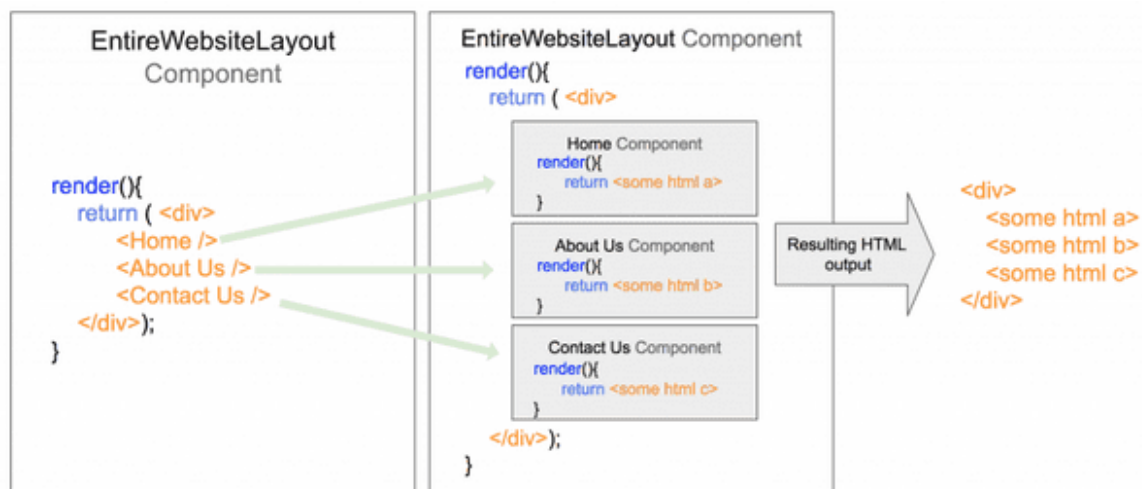
> React > Componentes

UI



isec
Politécnico de Coimbra

> React > Componentes



> React > Componentes

- Componentes permitem que um interface com o utilizador seja decomposto num conjunto de elementos reutilizáveis
 - cada um desses elementos pode ser considerado/aplicado de forma independente
- Componentes (à semelhança de uma função no JS):
 - aceitam inputs (**props**)
 - retornam o que deve ser renderizado no viewport do browser.
- Componentes podem ser definidos de duas formas distintas originando dois tipos de componentes:

> React > Componentes

▪ *Stateless Functional Component*

- Componente simples, sem estado, que se trata apenas de uma função que obtém dados via props e retorna algum JSX que será renderizado na página.

▪ *Class-based componentes*

- Componentes criados com `class`
- Cada componente de class tem o método `render`, o qual descreve como o componente será renderizado na página;

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

```
class Welcome extends React.Component {  
  render() {  
    return <h1>Hello, {this.props.name}</h1>;  
  }  
}
```

- *Statefull Component* pois controla como o estado altera e a implementação da lógica do componente. Além disso, tem acesso a todas as diferentes faces do método do ciclo de vida do React

▪ *Statefull Functional Component with Hooks*

> Componentes > Class Component

▪ **Class Component**

- Declarado com base em **class** e faz o **extends** do `React.Component` o que lhe permite ter acesso aos seus métodos
- O método `render()` é **obrigatório** e faz o return dos conteúdos definidos no componente através de código **JSX** para o ReactDOM fazer a ligação ao DOM (através dos métodos do DOM).
- O seu nome inicia-se sempre por **maiúscula**
- Não pode retornar **2 siblings elements** (tem que existir sempre um wrapper)

```
class Site extends React.Component {  
  render() {  
    ...  
    return (  
      <div style={cardStyle}>  
        <Header {...this.props}/>  
        <Sidebar {...this.props}/>  
        <Content {...this.props}/>  
        <Footer {...this.props}/>  
      </div>  
    )  
  }  
}
```

> Componentes > Functional Component

React

■ Functional Component









- Uma função que retorna **JSX**.
- Tanto pode ser definida com base na declaração com base em **function()** ou em alternativa numa **arrow function**
- O seu nome inicia-se sempre por **maiúscula**
- Não pode retornar **2 siblings elements** (tem que existir sempre um *wrapper*)

```
const Person = () => {  
  
  return <h1> Function Component </h1>  
  
}
```

isec
Politécnico de Coimbra

> React Components

React

	Functional Component	Class Component
<i>Permite estados dentro do componente*</i>		
<i>Usa métodos do ciclo de vida dentro do componente*</i>		
<i>Usa atributo ref no component*</i>		
<i>Usa hooks dentro do componente</i>		

* Situação alterada com a introdução dos Hooks

isec
Politécnico de Coimbra

> React Components

- Functional Components vs Class Components

Class Component	Functional Component (Recorrendo aos Hooks)
<i>componentDidMount</i> <i>componentDidUpdate</i> <i>componentWillUnmount</i>	<i>useEffect</i>
<i>state</i>	<i>useState</i>
<i>ref={contentRef => ref}</i>	<i>useRef</i>

A ver mais tarde...

> Class Component

```
import React from "react";
const root = ReactDOM.createRoot(document.getElementById("root"));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

Cria um componente designado por HelloWorld. É um componente criado com base numa class que faz o extend de um React.Component

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

export default function App() {
  return <HelloWorld />;
}
```

Todos os componentes React têm um método render() que geram os elementos HTML no browser através de JSX

Chamada do componente de Class <HelloWorld/>

> Functional Component

```
import React from "react";
const root = ReactDOM.createRoot(document.getElementById("root"));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

Cria um componente designado por HelloWorld. É um componente criado com base numa class que faz o extend de um React.Component

```
function HelloWorld() {
  return <p> Primeiro Componente.... </p>;
}

export default function App() {
  return <HelloWorld />;
}
```

Todos os componentes React têm um método render() que geram os elementos HTML no browser através de JSX

Componente App

Chamada do componente de Class <HelloWorld/>

> Functional and Class Component

```
function HelloWorld() {
  return <p> Primeiro Componente.... </p>;
}

const helloWorld = () => <p>Primeiro Componente....</p>;

class HelloWorld extends React.Component {
  render() {
    return <p> Segundo Componente.... </p>;
  }
}

export default function App() {
  return <HelloWorld />;
}
```



Componente App

> Functional and Class Component

```
import React from "react";

const helloWorld2 = () => <p>E isto?....</p>;

class HelloWorld extends React.Component {
  render() {
    return helloWorld2();
  }
}

export default function App() {
  return <HelloWorld />;
}
```

E isto?....

Componente App



> Functional and Class Component

```
import React from "react";

const helloWorld2 = () => { <p>E isto?....</p>; }

class HelloWorld extends React.Component {
  render() {
    return helloWorld2();
  }
}

export default function App() {
  return <HelloWorld />;
}
```

Componente App



> Functional and Class Component

```
import React from "react";

const helloWorld2 = () => { return <p>E isto?....</p>; }

class HelloWorld extends React.Component {
  render() {
    return helloWorld2();
  }
}

export default function App() {
  return <HelloWorld />;
}
```

E isto?....

Componente App

