

Hooks e aplicações

DEVinHouse

Parcerias para desenvolver a sua carreira

SENAI

<LAB365>

Hooks são funções que permitem a você “ligar-se” aos recursos de state e ciclo de vida do React a partir de componentes funcionais. Hooks não funcionam dentro de classes — eles permitem que você use React sem classes.

A principal vantagem dos Hooks, é o de poder utilizar componentes funcionais para guardar estado e gerenciar o ciclo de vida do componente. E componentes funcionais nos aproximam mais da natureza do JavaScript, que está relacionada à programação funcional.

Isso elimina a necessidade de utilizar o operador `this`, que é usado para classes e é pouco intuitivo no JavaScript. Outra vantagem é que podemos construir nossos próprios hooks para implementar nas aplicações.

Hooks - Regras

Existem algumas regras a serem observadas quando usamos um hook no React:

- Não podem ser chamados dentro de condicionais;
- Não podem ser chamados dentro de funções aninhadas;
- Não podem ser chamados dentro de loops;
- Só podem ser chamados dentro de componentes React funcionais;

Exemplo Class Based

```
5  export default class Home extends React.Component {  
6    constructor(props) {  
7      super(props);  
8      this.state = { text: 'Senai' };  
9      this.changeText = this.changeText.bind(this);  
10   }  
11  
12   changeText(event) {  
13     this.setState({  
14       text: event.target.value  
15     });  
16   }  
17  
18   render() {  
19     return (  
20       <div style={{  
21         'display': 'flex', 'flexDirection': 'column', 'max-width': '240px', 'padding': '20px'  
22       }}>  
23         <span style={{'marginBottom': '10px'}}>{this.state.text}</span>  
24         <input value={this.state.text} onChange={this.changeText}></input>  
25       </div>  
26     )  
27   }  
}
```

Exemplo Componentes Funcionais

```
3  import React, { useState } from 'react';
4
5  export default function Home() {
6    const [text, setText] = useState('Senai')
7
8    return (
9      <div style={{
10        'display': 'flex', 'flexDirection': 'column', 'max-width': '240px', 'padding': '20px'
11      }}>
12        <span style={{ 'marginBottom': '10px' }}>{text}</span>
13        <input value={text} onChange={(e) => setText(e.target.value)}></input>
14      </div>
15    )
16  }
```



useState



DEVinHouse

Parcerias para desenvolver a sua carreira

SENAI

<LAB365>

O ***useState*** nos permite criar estados em um componente criado a partir de uma função, assim como o *“state”* presente em componentes criados a partir de classes. A inicialização do *“useState”* retorna uma variável e um método para atualizá-la.

Importante destacar que a cada atualização de uma variável com `useState`, seu valor será atualizado também na view.

```
const [text, setText] = useState('Senai')
```


Declarando useState

```
const [text, setText] = useState('Senai')
```


Variável

Método para atualizar
a variável

Valor inicial de "text"

Exemplo 1

```
1  import '../App.css';
2  import React, { useState } from 'react';
3
4  export default function Home() {
5
6      const [count, setCount] = useState(0)
7
8      return (
9          <div style={{
10              'display': 'flex', 'flexDirection': 'column', 'max-width': '240px', 'padding': '20px'
11          }}>
12              <span style={{'marginBottom': '10px'}}>Você clicou {count} vezes</span>
13              <button onClick={(e) => setCount(prevValue => prevValue + 1)}>Clique aqui</button>
14          </div>
15      )
16  }
```



Mas por qual motivo devemos utilizar o hook `useState` ao invés de atualizar nossas variáveis diretamente?

```

4  export default function Case1() {
5    let a = 0;
6    const [b, setB] = useState(0);
7
8    function setNewValueA() {
9      a += 1;
10   }
11   return (
12     <div className="App">
13       <div>
14         Local: {a}
15         <button onClick={setNewValueA}>Variável Local (A)</button>
16       </div>
17       <div>
18         Estado: {b}
19         <button onClick={() => setB(prevB => prevB + 1)}>
20           Variável de Estado (B)
21         </button>
22       </div>
23     </div>
24   );
25 }

```

Pelo recurso criar atualizações automáticas na view, seu uso é bem empregado em controles de estado de menus de seleção, modals/popups, controle de formulários e para quaisquer ações que necessitem de um bom controle de dados entre a parte lógica e o HTML.

Exemplo 2

```
1  import '../App.css';
2  import React, { useState } from 'react';
3
4  export default function Home() {
5
6      const [count, setCount] = useState(0)
7
8      return (
9          <div style={{
10              'display': 'flex', 'flexDirection': 'column', 'max-width': '240px', 'padding': '20px'
11          }}>
12              <span style={{'marginBottom': '10px'}}>Você clicou {count} vezes</span>
13              <button onClick={(e) => setCount(prevValue => prevValue + 1)}>Clique aqui</button>
14          </div>
15      )
16  }
```

Exercício 1 - NFT Store (25min)

Construir um componente com o nome CardNFT que irá receber um objeto por meio de props. Esse objeto irá conter nome, imagem, valor em eth e real, categoria, nome e nick do criador.



Ref: <https://github.com/yanestevesufjf/clamed-react/tree/master/src/pages/nftStore>

Exercício 1 - NFT Store (25min)

 **Pasti 2**
@sonyart.eth



3 dias restantes

(R\$2524,97)
0.295 ETH

 **Opened Stash Box**
@snoopdoggbodr



3 dias restantes

(R\$8.559,22)
1 ETH

 **2 Geez (Instrumental)**
@snoopdoggbodr



3 dias restantes

(R\$10699,02)
1.25 ETH

 **Bored Ape Yacht Club #7369**
@franklinisbored



3 dias restantes

(R\$650500,40)
76 ETH



useEffect

DEVinHouse

Parcerias para desenvolver a sua carreira

SENAI

<LAB365>

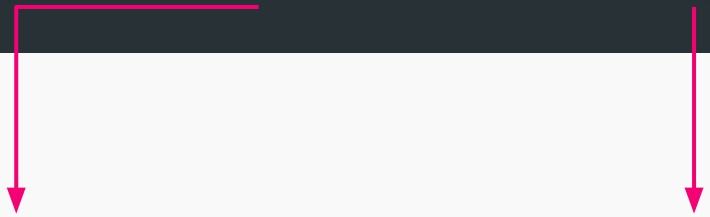
useEffect

O ***useEffect*** é um Hook que serve para lidar com os efeitos. Podemos usá-los como os lifeCycles `componentDidMount`, `componentDidUpdate` e `componentWillUnmount`.

```
useEffect(() => {  
  console.log('Lorem ipsum.')})
```

Declarando useEffect

```
useEffect(() => { /* Realiza Algo */ }, [])
```



Execução do useEffect

Condição para
execução

Declarando useEffect

Existem 3 formas de configurarmos o useEffect:

- Sem passar nenhum array de dependências;
- Passando um array de dependências vazio;
- Passando um array de dependências com valores;

useEffect - Sem array de dependências

Quando o useEffect é chamado **sem um array de dependências, ele é executado a cada renderização do componente.**

Portanto, toda vez que alteramos um estado do componente utilizando a função setter do useState, o useEffect é executado.

```
useEffect(() => {  
  console.log(`Componente foi atualizado`)  
})
```

useEffect - Array de dependências vazio

Quando o `useEffect` é chamado **com o array de dependências vazio, ele executa uma única vez na primeira renderização.**

Mesmo que o estado altere e o componente renderize novamente, a callback do `useEffect` não é chamada novamente.

```
useEffect(() => {  
  console.log(`Componente foi montado`)  
}, [])
```

useEffect - Array de dependências com valores

Quando o useEffect é chamado **com o array de dependências com valores, ele executa toda vez que este valor é alterado.**

Esta sintaxe é utilizada quando nosso callback do useEffect depende de algum valor externo.

```
useEffect(() => {  
  console.log(`Variável "text" foi modificada`)  
}, [text])
```

O `useEffect` capta alterações na aplicação, com a possibilidade de criar mapeamentos nas variáveis também, com isso acaba sendo um excelente recurso para buscar/atualizar dados após determinada ação de usuário, desenvolvimento de carousel com passagem automática de imagens e outras possibilidades.

Aplicação Prática

```
4 export default function Home() {
5   const [categoria, setCategoria] = useState('todos');
6   useEffect(() => {
7     if (categoria === 'todos') {
8       console.log('Buscando todos os dados.')
9       return;
10    }
11    console.log(`Buscar dados da categoria ${categoria}.`)
12  }, [categoria])
13
14  return (
15    <section style={{ 'maxWidth': '250px', 'padding': '20px' }}>
16      <div style={{
17        'display': 'flex', 'flexDirection': 'row', 'justifyContent': 'space-between'
18      }}>
19        <button style={{ 'fontSize': '14px' }} onClick={() => setCategoria('todos')}>Todos</button>
20        <button style={{ 'fontSize': '14px' }} onClick={() => setCategoria('celular')}>Celulares</button>
21        <button style={{ 'fontSize': '14px' }} onClick={() => setCategoria('roupa')}>Roupas</button>
22      </div>
23      <span style={{ 'display': 'block', 'marginTop': '20px' }}>Categoria selecionada: {categoria}</span>
24    </section>
25  )
26 }
```

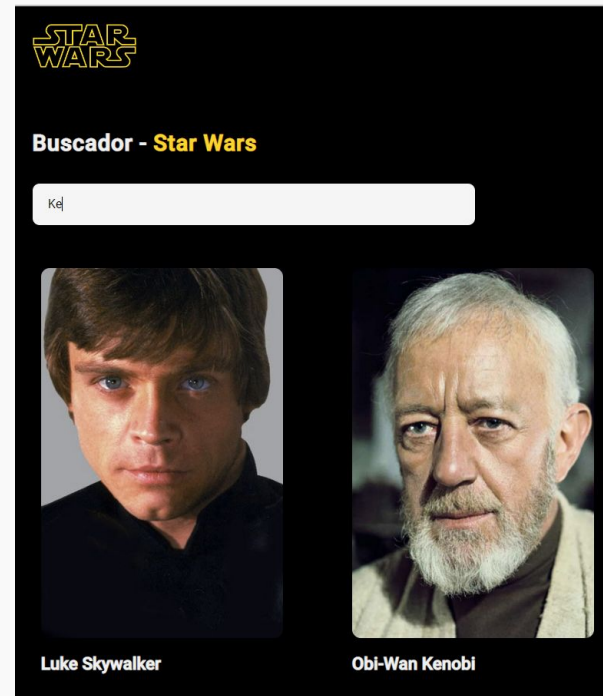
Exemplo “useEffect” para buscar dados categorizados.

O que acontecerá com o código a seguir?

```
16  useEffect(( ) => {  
17    |   setCount(prevValue => prevValue+1)  
18  | })  
19  
20  return (  
21    |   <section style={{ 'maxWidth': '250px', 'padding': '20px' }}>  
22    |     <span>{count} vezes</span>  
23    |   </section>  
24  | )
```

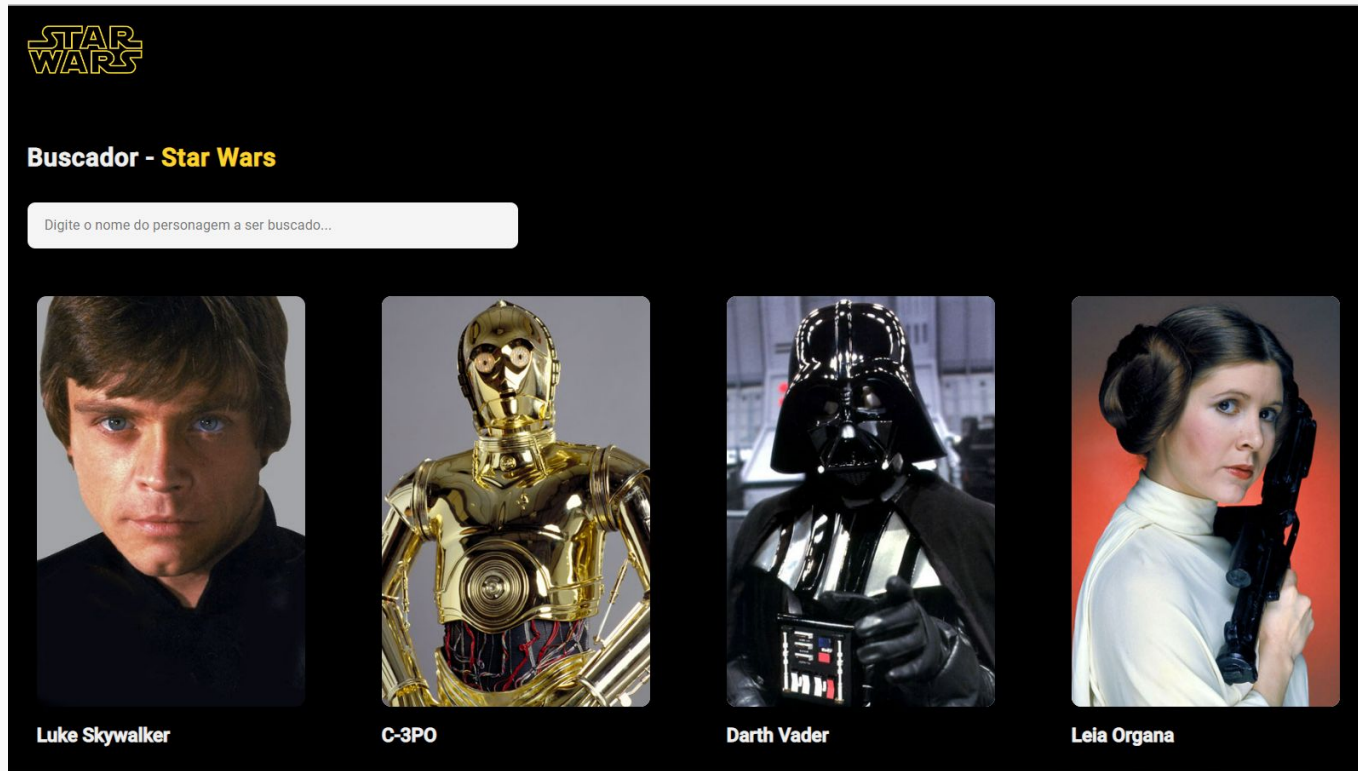
Exercício 2 - Star Wars (20min)

Com Hooks, construa um campo de texto e uma busca dinâmica por nome dos personagens do Star Wars.



Ref: <https://github.com/yanestevesufjf/clamed-react/tree/master/src/pages/starWars>

Exercício 2 - Star Wars (20min)



Ref: <https://github.com/yanestevesufjf/clamed-react/tree/master/src/pages/starWars>

Exercício 3 - NFT Store (20min)

Criar um menu na aplicação da NFT Store que realize o filtro das nfts a cada clique nos botões de navegação.

Os botões devem conter o nome das categorias, como: Ilustração, música, jogos e ver todos.


Ref: <https://github.com/yanestevesufjf/clamed-react/tree/master/src/pages/nftStore>




Exercício 3 - NFT Store (20min)


NFT Store


Todas Ilustração Música Games

**Pasti 2**
@sonyart.eth




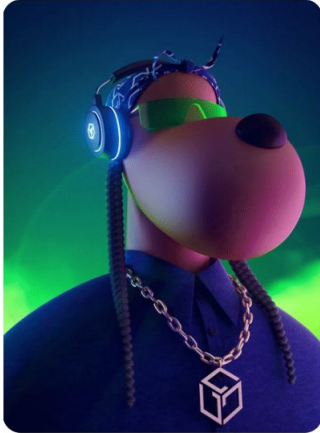
3 dias restantes (R\$2524,97)
0.295 ETH

**Opened Stash Box**
@snoopdoggbodr





3 dias restantes (R\$8.559,22)
1 ETH

**2 Geez (Instrumental)**
@snoopdoggbodr



3 dias restantes (R\$10699,02)
1.25 ETH

**Bored Ape Yacht Club #7369**
@franklinisbored



3 dias restantes (R\$650500,40)
76 ETH

- Documentação oficial sobre Hooks - <https://pt-br.reactjs.org/docs/hooks-intro.html>
- React useEffect - <https://pt-br.reactjs.org/docs/hooks-effect.html>
- Site com diversos exemplos de aplicações dos Hooks - <https://usehooks.com/>
- Repo com os códigos das aulas - <https://github.com/yanestevesufjf/clamed-react>

AVALIAÇÃO DOCENTE

O que você está achando das minhas aulas neste conteúdo?

[Clique aqui](#) ou escaneie o QRCode ao lado para avaliar minha aula.

Sinta-se à vontade para fornecer uma avaliação sempre que achar necessário.





DEVinHouse

Parcerias para desenvolver a sua carreira

OBRIGADO!



<LAB365>

AGENDA

- Informações do template
- Exemplo com texto (Light mode)
- Exemplo com texto (Dark mode)
- Exemplo com imagens (Light mode)
- Exemplo com imagens (Dark mode)

INTERVALO DE AULA

DEV!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

Início: 20:20

Retorno: 20:40



INFORMAÇÕES DO TEMPLATE

- Título da Apresentação:
 - Fonte: Ubuntu Bold
 - Formato: Maiúsculo
 - Tamanho: 34
 - Cor: Branco
- Título do Slide:
 - Fonte: Ubuntu Bold
 - Formato: Maiúsculo
 - Tamanho: 22
 - Cor: Branco
- Parágrafos:
 - Fonte: Open Sans Normal
 - Tamanho: 14 a 18
 - Cores: Branco (Dark Mode) ou Preto (Light Mode)
- Marcadores de tópicos:
 - Formatos: Símbolos ou Alfanuméricos
 - Cor: Laranja
- Padrão de Cores:
 - Cinza - #868584
 - Preto - #1C1C19
 - Branco - #FAFAFA
 - Laranja - #F08305
 - Rosa - #c71d81
 - Azul - #0e1d8e

EXEMPLO COM TEXTO (LIGHT MODE)

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

EXEMPLO COM TEXTO (DARK MODE)

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

EXEMPLO COM IMAGENS (LIGHT MODE)

Título

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.



Título

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

EXEMPLO COM IMAGENS (DARK MODE)

Título

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.



Título

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.