FIAP

RESPONSIVE WEB DEVELOPMENT

REACT - USEEFFECT

Prof. Alexandre Carlos profalexandre.jesus@fiap.com.br

Prof. Luís Carlos lsilva@fiap.com.br

Prof. Wellington Cidade profwellington.tenorio@fiap.com.br

USEEFFECT

CICLO DE VIDA DOS COMPONENTES

Em uma aplicação precisamos organizar e integrar as ações de forma que as atividades fluam e cada tarefa seja executada no momento e forma correta.

- Nas aplicações do React utilizamos um hook que controla e escuta o ciclo de vida destes componentes, é o useEffect.

USEEFFECT

UTILIZANDO O USEEFFECT NA APLICAÇÃO

Vamos fazer alguns exemplos para entender que fazes do ciclo de vida dos componentes podemos manipular.

- Após criar uma nova aplicação React utilizando o Vite, limpe o componente App.tsx desta forma:

```tsx
import { useState } from 'react'

function App() {

  const [valor, setValor] = useState(0)

  return (
    <div>
      <h1>COMPONENTE APP</h1>
    </div>
  )
}
```

Vamos criar um state para fazermos algumas interações.

USEEFFECT

UTILIZANDO O USEEFFECT NA APLICAÇÃO

Dentro de **src**, crie a pasta **components** e nela um componente chamado **ExemploEffect.tsx**. Vamos utilizar nosso state valor dentro dele.

```tsx
function ExemplosEffect(props) {


  return(
    <div>
        <h2>Exemplos Effect</h2>

        <p>Valor do State: {props.valor}</p>
        <button onClick={()=> props.setValor(props.valor + 1)}>Aumentar</button>

    </div>
  )
}

export default ExemplosEffect;
```

Temos aqui uma contagem simples, acrescentando 1 ao valor sempre que for clicado no botão..

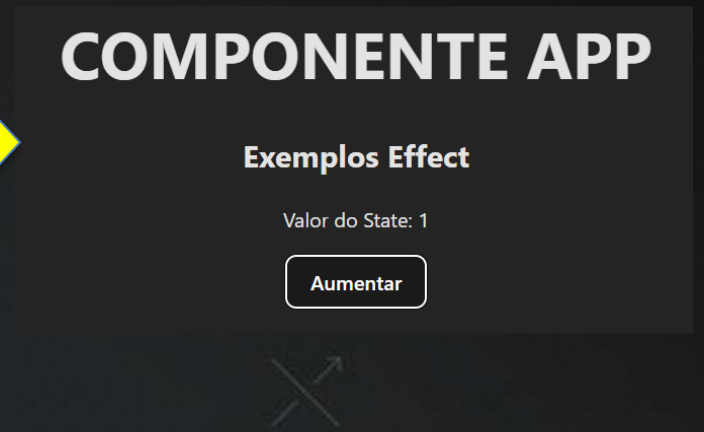
USEEFFECT

UTILIZANDO O USEEFFECT NA APLICAÇÃO

Não se esqueça de chamar o **ExemploEffect** dentro do componente **App** e passar o **valor** e **setValor** por props. Assim o componente filho poderá utilizar o state armazenado no componente pai.

```
1   import { useState } from 'react'
2   import './App.css'
3   import ExemplosEffect from './components/ExemplosEffect'
4
5   function App() {
6
7     const [valor, setValor] = useState(0)
8
9     return (
10      <div>
11        <h1>COMPONENTE APP</h1>
12        <ExemplosEffect valor={valor} setValor={setValor}/>
13      </div>
14    )
15  }
16  export default App
```

Agora já conseguimos utilizar o state valor a partir do componente ExemploEffect, certo?



USEEFFECT

CHAMANDO A AÇÃO EM QUALQUER EVENTO DO COMPONENTE

A partir de agora podemos testar vários eventos no ciclo de vida de nosso componentes.
Eventos como **criação**, **alteração**, **alteração de um valor específico** e até a **exclusão** dele na página.
Para controlar estes eventos utilizaremos o Hook **useEffect** no componente **"ExemploEffect"**, vamos começar utilizando ele para disparar uma mensagem sempre que ele tiver um evento, seja qual for.

```
import { useEffect } from "react";

function ExemplosEffect(props) {

  //Exemplo de chamada em todos os eventos.

  useEffect(
    ()=>{
      console.log("Em todas as chamadas eu sou chamado!!!");
    }
  )

  return(
```

O método do useEffect espera receber uma função anônima, que apresentada desta forma, é chamada sempre que temos alguma mudança de estado em nosso componente.

Abra o inspetor de código do navegador para vermos a mensagem sendo exibida no console.

USEEFFECT

OBSERVAÇÃO IMPORTANTE

A partir da versão 18 do react, sempre que precisamos trabalhar com o useEffect devemos retirar a Tag do React.StrictMode que vem nos templates, dentro do arquivo main.tsx.

```tsx
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.tsx'

ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
)
```

```tsx
import React from 'react'
import ReactDOM from 'react-dom/client'
import App from './App.tsx'

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

Só retirar a tag do React.StrictMode

USEEFFECT

CHAMANDO A AÇÃO SOMENTE NA CRIAÇÃO DO COMPONENTE

Aqui temos o useEffect somente executando a função quando o componente é criado. Repare que, o que mudou no código foi que, no final da função, colocamos uma virgula e um par de colchetes vazios. Nestes colchetes ele espera um array de valores que ele deve observar, como está vazio, ele só executa quando o componente é criado.

```
//Exemplo de chamada na criação

useEffect(
  ()=>{
    console.log("Eu só quando é iniciado!!!");
  },[]
)
```

Como o array está vazio, ele só executa a primeira vez, quando criado, depois não tem mais ninguém para acompanhar.

USEEFFECT

CHAMANDO A AÇÃO QUANDO UM VALOR É ALTERADO

Como já começamos falar no exemplo anterior, também podemos controlar a chamada de quando um valor específico é alterado. Para fazer isso, devemos indicá-lo dentro do array no final da função.

```
//Exemplo de chamada de um valor específico

useEffect(
  ()=>{
    console.log(`Quando o valor está sendo mudado: ${props.valor}`);
  },[props.valor]
)
```

Aqui ele será chamado sempre que valor sofrer alteração.

USEEFFECT

CHAMANDO A AÇÃO QUANDO O COMPONENTE É EXCLUÍDO

Por fim vamos ver como fazer para, se o componente for destruído, ou desmontado, retirado da tela, podemos também pegar este evento. Vamos ter que passar a ação como uma função sendo retornada da função anônima que temos nele.

```
// Quando o elemento é destruído

    useEffect(
      ()=>{
        return ()=> console.log(`Ops, me apagaram!!!`)
      },[]
    )
```

Retornamos a função que queremos executar.

USEEFFECT

CHAMANDO A AÇÃO QUANDO O COMPONENTE É EXCLUÍDO

Mas para vermos funcionando, vamos ao componente App. Aqui criaremos um state chamado filho para controlarmos sua criação e exclusão.
A lógica que vamos usar é bem simples, se filho for igual a true mandamos inserir o componente, se for false vamos substituí-lo por uma string vazia. Para isso vamos usar o ternário.

Criação do state filho.

Criação da lógica como falamos acima.

```
function App() {

  const [valor, setValor] = useState(0)
  const [filho, setFilho]= useState(true)

  return (
    <div>
      <h1>COMPONENTE APP</h1>
      <button onClick={()=>setFilho(!filho)}>
      {filho ? "Apagar" : "Criar" }
      </button>
      { filho ? <ExemplosEffect valor={valor} setValor={setValor}/> : "" }
    </div>
  )
}
export default App
```

EXERCÍCIO

Crie um novo projeto chamado exercicio5.

Crie uma pasta chamada components e dentro um arquivo chamado Aviao.tsx, ele deve ter um h2 o identificando e um state chamado altura, que deve começar em “0” e apresentar a altura em um parágrafo.

Você deve criar um botão que quando clicado aumente a altura em 100.

Após a primeira parte apresente no console, usando o useEffect, mensagens avisando que:

“o Avião está ligado!!!” sempre que sofrem uma alteração.

“o avião decolou” (quando o componente for criado),

“O avião está em XXX pés”(quando ele subir através do botão) e

“O avião foi derrubado”(quando o componente for removido).

OBRIGADO

FIAP

Copyright © 2024 | Professores Titulares

Todos os direitos reservados. Reprodução ou divulgação total ou parcial deste documento, é expressamente proibido sem consentimento formal, por escrito, do professor/autor.


FIAP