

# MIT em Desenvolvimento Full Stack

Front-end Jamstack com Gatsby

# Agenda

## **Aula 3:** Desenvolvimento da Estrutura da Aplicação com Gatsby.

- Componentes React.
- Estilos.
- Dados Externos X Dados Internos.
- React Hooks.
- Dados Externos.

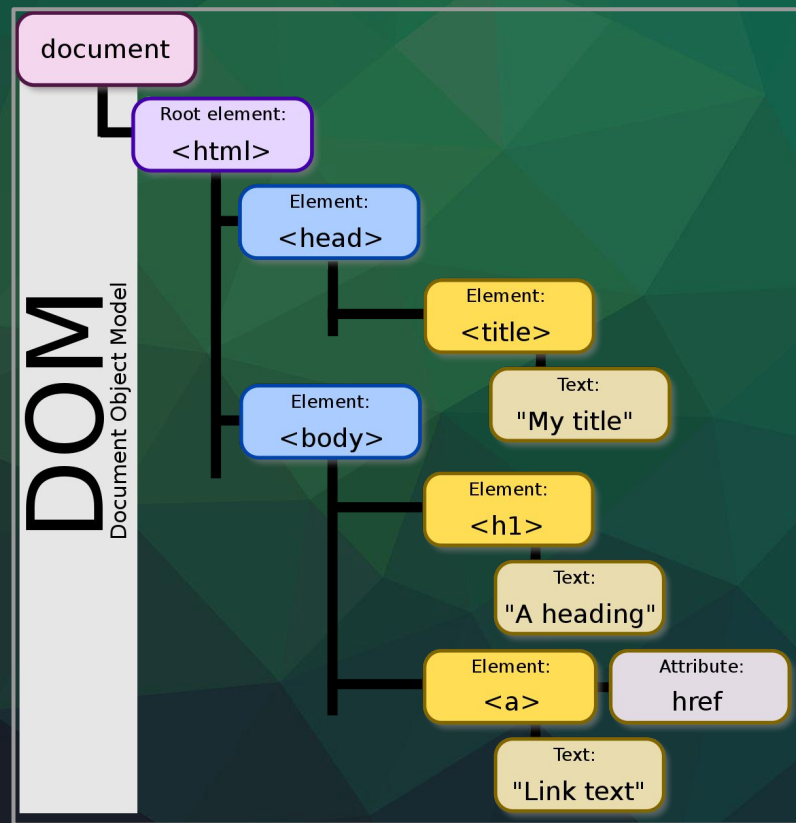




# Componentes React

O foco principal do **React** é a **componentização** que visa reutilizar partes de interface e manipulação eficiente do **DOM** - Document Object Model.

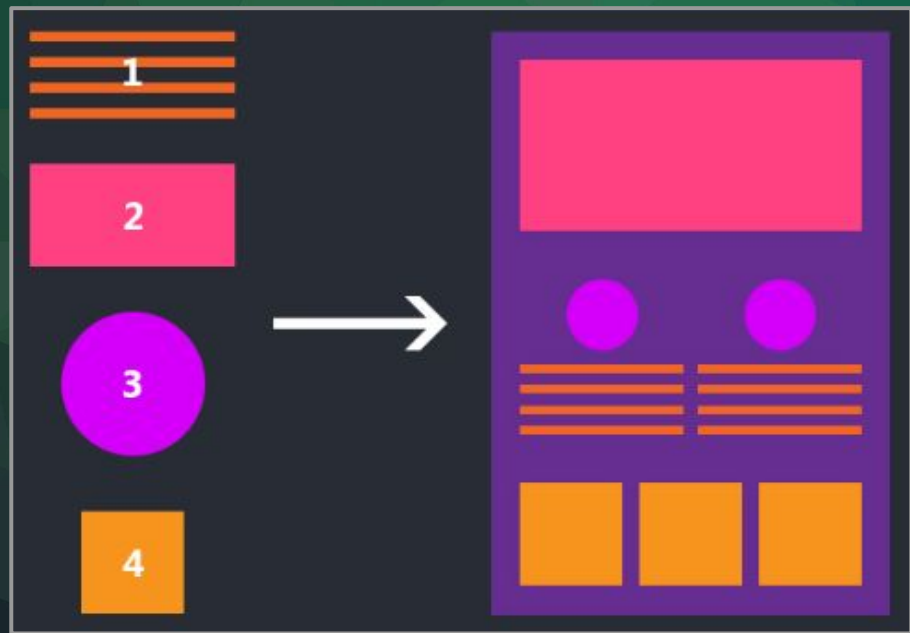
O **DOM** é uma forma de exibir a estrutura de um documento HTML como uma árvore lógica.



Os componentes do **React** são blocos de funções altamente independentes e têm comportamento próprio.

Tais componentes têm uma representação visual e lógica dinâmica.

Alguns componentes podem até conversar com o servidor por conta própria a fim de executar consultas e atualizações.









# Estilos



Existem várias abordagens diferentes de aplicação de **CSS** no **Gatsby**, permitindo ajustar a tipografia, as cores e o layout.

### **CSS Global**

Em muitos usos de CSS, uma única folha de estilo (arquivo CSS) é baixada e usada para estilizar um site.

### **CSS Modular**

Gatsby também fornece folhas de estilo modulares que permitem definir o escopo de declarações para componentes individuais.

### **CSS em JavaScript**

As declarações CSS ainda têm escopo local e são gerenciadas pelo JS, sendo possível definir estilos dinâmicos baseados em eventos assíncronos.



# layout.css U x

1

src > components > # layout.css > .topnav a

```
1  .layout {
2      width: 100%;
3      height: 95vh;
4      display: grid;
5      grid: "header" auto "main" 1fr "footer" auto / 1fr;
6      background-color: #aliceblue;
7  }
8
9  .header {
10     grid-area: header;
11     background-color: #aquamarine;
12     padding: 10px;
13 }
14
15 .main {
16     grid-area: main;
17     padding: 10px;
18 }
19
20 .footer {
21     grid-area: footer;
22     background-color: #aquamarine;
23     padding: 10px;
24 }
```

# layout.css U x

2

src > components > # layout.css > .topnav a

```
26  .topnav {
27      overflow: hidden;
28      background-color: #AECFDF;
29  }
30
31  .topnav a {
32      float: left;
33      display: block;
34      color: black;
35      text-align: center;
36      padding: 14px 16px;
37      text-decoration: none;
38  }
39
40  .topnav a:hover {
41      background-color: #9F9FAD;
42      color: black;
43  }
```

CSS Global

JS layout.js U X

src &gt; components &gt; JS layout.js &gt; ...

```
1  import React from "react"
2  import "../layout.css"
3  import { Link } from "gatsby"
4
5  export default function Layout({ children }) {
6    return (
7      <main className="layout">
8        <div className="header">
9          <h3>Estudos de Jamstack - Gatsby</h3>
10         <nav className="topnav">
11           <Link to="/">Início</Link>
12           <Link to="/pagina2">Página 2</Link>
13         </nav>
14       </div>
15       <div className="main">
16         {children}
17       </div>
18       <div className="footer">
19         <h3>Aqui vai um rodapé</h3>
20       </div>
21     </main >
22   )
23 }
```

## Estudos de Jamstack - Gatsby

[Início](#) [Página 2](#)

# Olá Mundo Gatsby

Esse é o primeiro parágrafo

Esse é o segundo parágrafo

Aqui vai um rodapé

4

# footer.module.css U X

src &gt; components &gt; # footer.module.css &gt; ...

```
1  .footer {
2    grid-area: footer;
3    background-color: aquamarine;
4    padding: 10px;
5    color: gray;
6    text-align: center;
7  }
```

## CSS Modular

5

JS footer.js U X

src &gt; components &gt; JS footer.js &gt; ...

```
1  import React from "react"
2  import * as footerStyles from "./footer.module.css"
3
4  export default function Footer(props) {
5    return (
6      <p className={footerStyles.footer}>
7        © {props.copyrightYear} Estudos de Gatsby. Todos os direitos reservados.
8      </p>
9    )
10 }
```

## CSS Global X CSS Modules

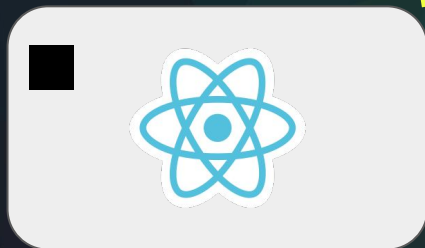
Os módulos CSS nos fornecem uma maneira de **garantir que um arquivo CSS seja exclusivo entre os componentes** e não seja acessado acidentalmente em outro lugar.

Isso pode ser especialmente útil ao trabalhar em **soluções maiores com muitas equipes diferentes**.

Também pode nos fornecer segurança de ao usar importações nomeadas para reduzir a chance de importar uma classe que não está definida.

# Dados Externos X Dados Internos



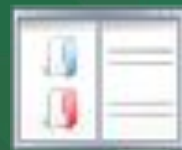


http request

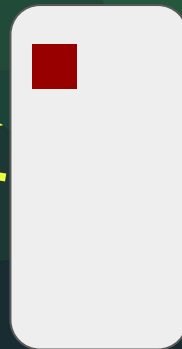
http response

http request

http response

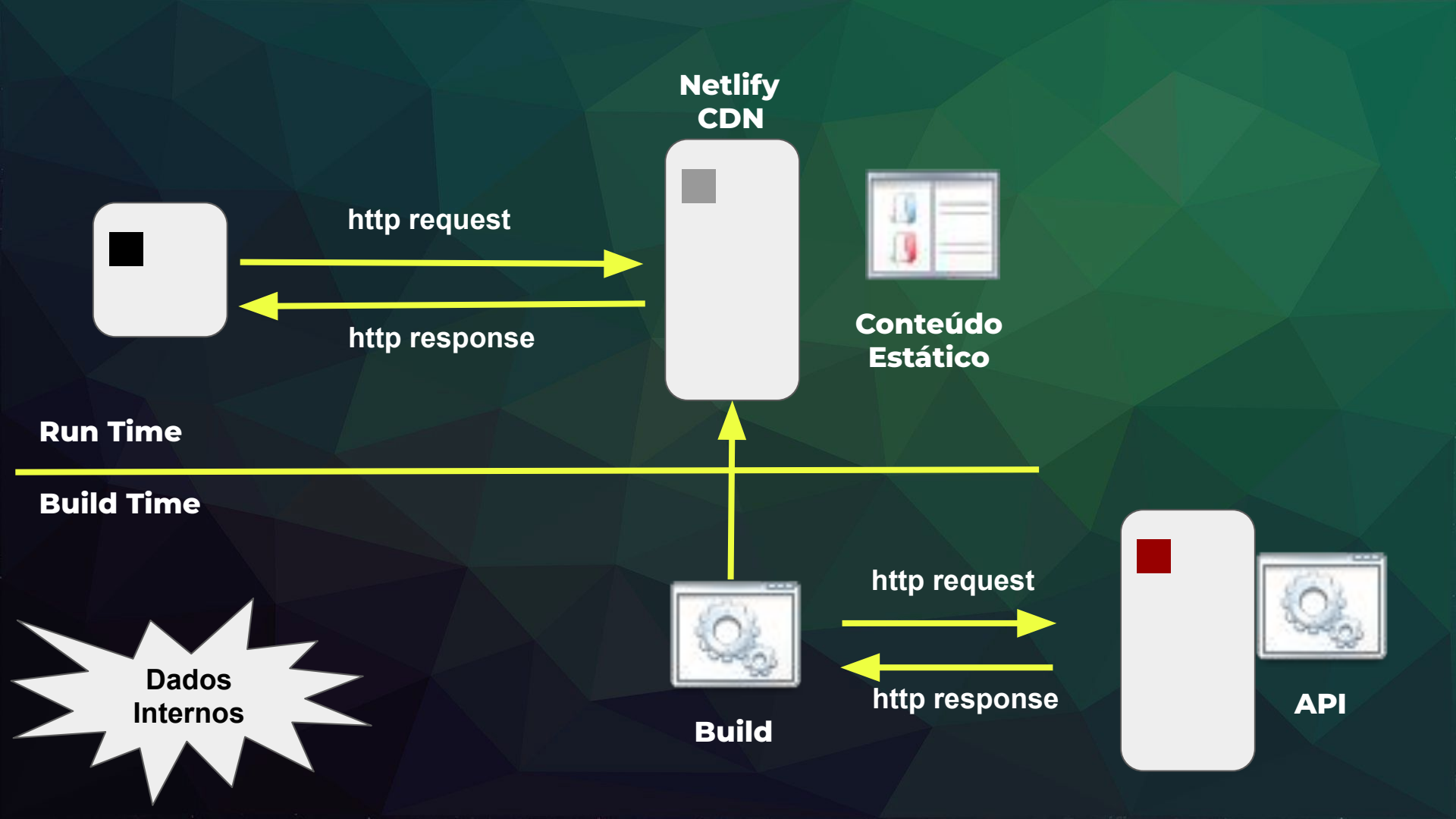


**Conteúdo Estático**



**API**

**Dados Externos**



The background is a low-poly, abstract geometric pattern composed of numerous triangles. The color palette is a gradient of greens and blues, ranging from dark, muted tones on the left to lighter, more vibrant greens on the right. The triangles vary in size and orientation, creating a textured, crystalline effect.

# React Hooks

**Hooks** nada mais são do que **funções JavaScript** que permitem conectar-se ao estado do **React** e aos recursos do ciclo de vida dos componentes.

**Hooks** permitem adicionar estado aos componentes e compartilhar lógica entre os componentes.

Existem três vantagens principais em usar **Hooks**: reutilização, legibilidade e testabilidade.

Algumas regras básicas:

- **Hooks** só podem ser usados dentro de componentes de função.
- Não é possível chamar **Hooks** dentro de loops, condições ou funções aninhadas - eles devem sempre ser chamados no topo do componente de função.
- Não é possível chamar **Hooks** de funções JavaScript comuns.

## Hooks

- State → **useState**, useReducer
- Context → useContext
- Ref → useRef, useImperativeHandle
- Effect → **useEffect**, useLayoutEffect, useInsertionEffect
- Performance → useMemo, useCallback, useTransition, useDeferredValue
- Resource → use
- Outros → useDebugValue, useId, useSyncExternalStore

**Hooks** permitem usar diferentes recursos do React de seus componentes. Você pode usar os Ganchos integrados ou combiná-los para construir o seu próprio.

## State

O estado permite que um componente “lembre” informações como a entrada do usuário.

Por exemplo, um componente de formulário pode usar o estado para armazenar o valor de entrada, enquanto um componente de galeria de imagens pode usar o estado para armazenar o índice da imagem seleciona

## Contexto

O contexto permite que um componente receba informações de pais distantes sem passá-las como propriedades.

Por exemplo, o componente de nível superior do seu aplicativo pode transmitir o tema da IU atual para todos os componentes abaixo, não importa a profundidade.



## Refs

Refs permitem que um componente retenha algumas informações que não são usadas para renderização, como um nó DOM, por exemplo.

Ao contrário do estado, atualizar uma referência não renderiza novamente seu componente.

**Refs** são uma “fuga” do paradigma React.

## Effect

Os efeitos permitem que um componente se conecte e sincronize com sistemas externos.

Isso inclui lidar com rede, DOM do navegador, animações, widgets escritos usando uma biblioteca de UI diferente e outros códigos não React.

## Performance

Uma maneira comum de otimizar o desempenho da nova renderização é pular trabalhos desnecessários.

Por exemplo, você pode dizer ao React para reutilizar um cálculo armazenado em cache ou ignorar uma nova renderização se os dados não tiverem sido alterados desde a renderização anterior.

## Resource

Os recursos podem ser acessados por um componente sem tê-los como parte de seu estado.

Por exemplo, um componente pode ler uma mensagem de um *promisse* ou ler informações de estilo de um contexto.

## Renderização

Processo “puro”: não envolve efeitos colaterais.

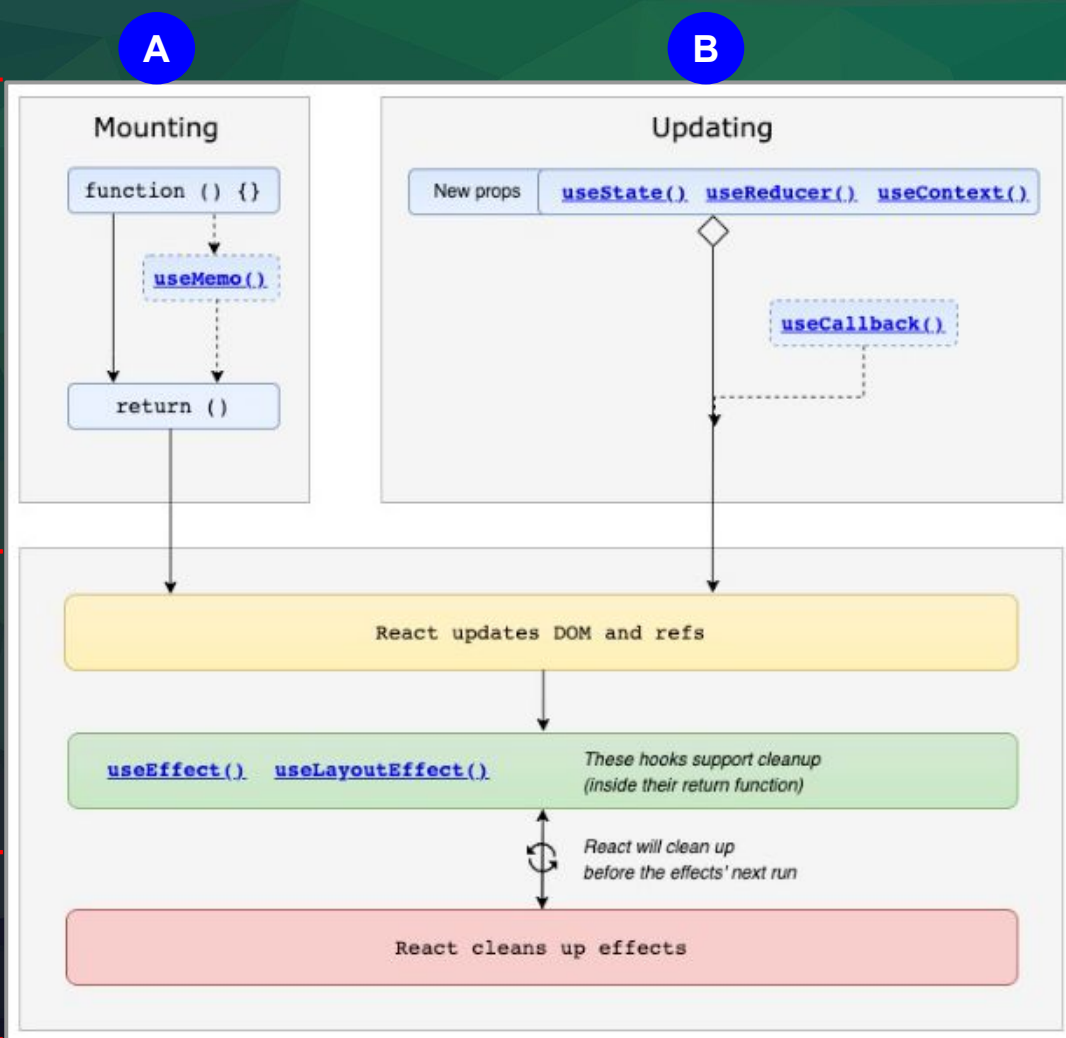
Pode ser pausada, abortada ou reiniciada pelo **React**.

## Commit

Atualiza o DOM, executa efeitos colaterais e programa atualizações.

## Cleanup

Executa antes que o componente seja removido.



# Dados Externos

IBGE - API de serviço de dados

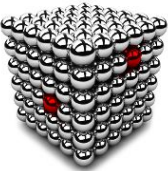
servicodados.ibge.gov.br/api/docs


Administrativo Conteúdo Ferramentas Livros Localhost Todos os favoritos


**gouv** ACESSO À INFORMAÇÃO PARTICIPE LEGISLAÇÃO ÓRGÃOS DO GOVERNO

**IBGE**

API de serviço de dados

 **Agregados**  
Análise multidimensional ao seu alcance.  
3.0

 **Banco de Dados Geodésicos**  
API para acesso às informações das estações geodésicas  
1.0

 **Calendário**  
Conheça o cronograma de ações e publicações do IBGE  
3.0

servicodados.ibge.gov.br/api/v1/localidades/estados...

Administrativo Conteúdo Ferramentas Livros Localhost Todos os favoritos

Estilos de formatação

```
[{"id":12,"sigla":"AC","nome":"Acre","regiao":{"id":1,"sigla":"N","nome":"Norte"}}, {"id":27,"sigla":"AL","nome":"Alagoas","regiao":{"id":2,"sigla":"NE","nome":"Nordeste"}}, {"id":16,"sigla":"AP","nome":"Amapá","regiao":{"id":1,"sigla":"N","nome":"Norte"}}, {"id":13,"sigla":"AM","nome":"Amazonas","regiao":{"id":1,"sigla":"N","nome":"Norte"}}, {"id":29,"sigla":"BA","nome":"Bahia","regiao":{"id":2,"sigla":"NE","nome":"Nordeste"}}, {"id":23,"sigla":"CE","nome":"Ceará","regiao":{"id":2,"sigla":"NE","nome":"Nordeste"}}, {"id":53,"sigla":"DF","nome":"Distrito Federal","regiao":{"id":5,"sigla":"CO","nome":"Centro-Oeste"}}, {"id":32,"sigla":"ES","nome":"Espírito Santo","regiao":{"id":3,"sigla":"SE","nome":"Sudeste"}}, {"id":52,"sigla":"GO","nome":"Goiás","regiao":{"id":5,"sigla":"CO","nome":"Centro-Oeste"}}, {"id":21,"sigla":"MA","nome":"Maranhão","regiao":{"id":2,"sigla":"NE","nome":"Nordeste"}}, {"id":51,"sigla":"MT","nome":"Mato Grosso","regiao":{"id":5,"sigla":"CO","nome":"Centro-Oeste"}}, {"id":50,"sigla":"MS","nome":"Mato Grosso do Sul","regiao":{"id":5,"sigla":"CO","nome":"Centro-Oeste"}}, {"id":31,"sigla":"MG","nome":"Minas Gerais","regiao":{"id":3,"sigla":"SE","nome":"Sudeste"}}, {"id":15,"sigla":"PA","nome":"Pará","regiao":{"id":1,"sigla":"N","nome":"Norte"}}, {"id":25,"sigla":"PB","nome":"Paraíba","regiao":{"id":2,"sigla":"NE","nome":"Nordeste"}}, {"id":41,"sigla":"PR","nome":"Paraná","regiao":{"id":4,"sigla":"S","nome":"Sul"}}, {"id":26,"sigla":"PE","nome":"Pernambuco","regiao":{"id":2,"sigla":"NE","nome":"Nordeste"}}, {"id":22,"sigla":"PI","nome":"Piauí","regiao":{"id":2,"sigla":"NE","nome":"Nordeste"}}, {"id":33,"sigla":"RJ","nome":"Rio de Janeiro","regiao":{"id":3,"sigla":"SE","nome":"Sudeste"}}, {"id":24,"sigla":"RN","nome":"Rio Grande do Norte","regiao":{"id":3,"sigla":"SE","nome":"Sudeste"}}, {"id":43,"sigla":"RS","nome":"Rio Grande do Sul","regiao":{"id":4,"sigla":"S","nome":"Sul"}}, {"id":11,"sigla":"RO","nome":"Rondônia","regiao":{"id":1,"sigla":"N","nome":"Norte"}}, {"id":14,"sigla":"RR","nome":"Roraima","regiao":{"id":1,"sigla":"N","nome":"Norte"}}, {"id":42,"sigla":"SC","nome":"Santa Catarina","regiao":{"id":4,"sigla":"S","nome":"Sul"}}, {"id":35,"sigla":"SP","nome":"São Paulo","regiao":{"id":3,"sigla":"SE","nome":"Sudeste"}}, {"id":28,"sigla":"SE","nome":"Sergipe","regiao":{"id":2,"sigla":"NE","nome":"Nordeste"}}, {"id":17,"sigla":"TO","nome":"Tocantins","regiao":{"id":1,"sigla":"N","nome":"Norte"}}, {"id":1,"sigla":"N","nome":"Norte"}]
```



JS Ufs.js U X

src > components > JS Ufs.js > ...

```
1  import React from "react";
2  import { useState, useEffect } from "react";
3
4  export default function Ufs({ setOpcaoUf }) {
5
6      const [options, setOptions] = useState([]);
7
8  >  useEffect(() => { ...
18  }, []);
19
20 >  const handleChange = (event) => { ...
22  }
23
24  return (
25      <div>
26          <select name="uf" onChange={handleChange}>
27              {options.map((option) => {
28                  return (
29                      <option key={option.key} value={option.key}>
30                          {option.value}
31                      </option>
32                  );
33              })}
34          </select>
35      </div>
36  );
37  }
```

1



2

JS Ufs.js U X

src &gt; components &gt; JS Ufs.js &gt; Ufs

```
8      useEffect(() => {
9          const opt = [{ key: "", value: "Selecione..." }];
10         fetch("https://servicodados.ibge.gov.br/api/v1/localidades/estados?orderBy=nome")
11             .then(results => results.json())
12             .then(data => {
13                 data.forEach(uf => {
14                     opt.push({ key: uf.id, value: uf.nome });
15                 });
16                 setOptions(opt);
17             });
18     }, []);
19
20     const handleChange = (event) => {
21         setOpcaoUf({ key: event.target.value, value: event.target[event.target.selectedIndex].text });
22     }
```

JS Ufs.js U

JS pagina2.js M X

src &gt; pages &gt; JS pagina2.js &gt; ...

```
1  import * as React from "react"
2  import Layout from "../components/layout"
3  import Ufs from "../components/Ufs"
4  import { useState, useEffect } from "react";
5
6  const Pagina2 = () => {
7
8      const [opcaoUf, setOpcaoUf] = useState({key:"", value:""});
9
10     useEffect(() => {
11         if (opcaoUf.value.length > 0) {
12             alert(opcaoUf.value);
13         }
14     }, [opcaoUf]);
15
16     return (
17         <Layout>
18             <h1>Essa é uma página 2</h1>
19             <div>
20                 <p>Unidades da Federação:</p>
21                 <Ufs setOpcaoUf={setOpcaoUf} />
22             </div>
23         </Layout>
24     )
25 }
26
27 export default Pagina2
28
29 export const Head = () => <title>Página 2</title>
```

3

JS Municipios.js X

src > components > JS Municipios.js > ...

```
1  import React from "react";
2  import { useState, useEffect } from "react";
3
4  export default function Municipios({ uf, setOpcaoMunicipio }) {
5
6      const [options, setOptions] = useState([]);
7
8      >  useEffect(() => { ...
18      }, [uf]);
19
20      >  const handleChange = (event) => { ...
22      }
23
24      return (
25          <div>
26              <select name="uf" onChange={handleChange}>
27                  {options.map((option) => {
28                      return (
29                          <option key={option.key} value={option.key}>
30                              {option.value}
31                          </option>
32                      );
33                  })}
34              </select>
35          </div>
36      );
37  }
```

4

5

JS Municipios.js X

src &gt; components &gt; JS Municipios.js &gt; ...

```
8   useEffect(() => {
9     const opt = [{ key: "", value: "Selecione..." }];
10    fetch(`https://servicodados.ibge.gov.br/api/v1/localidades/estados/${uf}/municipios`)
11      .then(results => results.json())
12      .then(data => {
13        data.forEach(mun => {
14          opt.push({ key: mun.id, value: mun.nome });
15        });
16        setOptions(opt);
17      });
18    }, [uf]);
19
20    const handleChange = (event) => {
21      setOpcaoMunicipio({ key: event.target.value, value: event.target[event.target.selectedIndex].text });
22    }
```

```
2 import Layout from "../components/layout"
3 import Ufs from "../components/Ufs"
4 import { useState, useEffect } from "react";
5 import Municipios from "../components/Municipios";
6
7 const Pagina2 = () => {
8
9   const [opcaoUf, setOpcaoUf] = useState({ key: "", value: "" });
10  const [opcaoMunicipio, setOpcaoMunicipio] = useState({ key: "", value: "", uf: "" });
11
12  useEffect(() => {
13    if (opcaoUf.value.length > 0) {
14      setOpcaoMunicipio({ ...opcaoMunicipio, uf: opcaoUf.key })
15    }
16  }, [opcaoUf]);
17
18  return (
19    <Layout>
20      <h1>Essa é uma página 2</h1>
21      <div>
22        <p>Unidades da Federação:</p>
23        <Ufs setOpcaoUf={setOpcaoUf} />
24        <p>Municípios do Estado: <b>{opcaoUf.value}</b></p>
25        <Municipios uf={opcaoUf.key} setOpcaoMunicipio={setOpcaoMunicipio} />
26      </div>
27    </Layout>
28  )
29 }
30
31 export default Pagina2
32
33 export const Head = () => <title>Página 2</title>
```