Arquitetura JavaScript (React I)

CAPÍTULO 3 – CONSTRUÇÃO DE COMPONENTES MAIS ROBUSTOS

PROF. RAPHAEL GOMIDE

React I

CAPÍTULO 3 – CONSTRUÇÃO DE COMPONENTES MAIS ROBUSTOS

PROF. RAPHAEL GOMIDE

Nesta aula

- Técnicas para a criação de componentes.
  - Renderização de arrays.
  - O problema do **prop drilling**.
  - Técnica ***Composition***.

- Criação do projeto **react-countries**.

Renderização de arrays

- Projeto de referência.
- Array:

```
const CARS = [
  {
    id: "c1",
    name: "Fiesta"
  },
  {
    id: "c2",
    name: "Versa"
  },
  {
    id: "c3",
    name: "Nivus"
  }
];
```

- **Má** prática – renderizar manualme

```
<ul>
  <li>{CARS[0].name}</li>
  <li>{CARS[1].name}</li>
  <li>{CARS[2].name}</li>
</ul>
```

Renderização de arrays

- Projeto de referência.
- Array:

```
const CARS = [
  {
    id: "c1",
    name: "Fiesta"
  },
  {
    id: "c2",
    name: "Versa"
  },
  {
    id: "c3",
    name: "Nivus"
  }
];
```

- **Boa** prática – renderizar com **array.map**:

```
<ul>
  {CARS.map((car) => {
    return <li key={car.id}>{car.name}</li>;
  })}
</ul>
```

Renderização de arrays

- Atenção à prop **key**.

```
<ul>
  {CARS.map((car) => {
    return <li key={car.id}>{car.name}</li>;
  })}
</ul>
```

- Ela “ajuda” o React a renderizar os dados mais corretamente.
- Recomenda-se a utilização de **identificadores únicos**.
- **Não** é recomendado que seja utilizado o **índice do array**.
- Caso a **key** não seja fornecida, o React emite o seguinte alerta no console do navegador:

```
Warning: Each child in a list should have a unique "key" prop.
```

O problema do *prop drilling*

- Em apps com muitos componentes, pode ocorrer este problema.
- Consiste na passagem de dados via props em *componentes intermediários.*
- Esses componentes intermediários não utilizam a prop, servem apenas para transportá-la para componentes filhos.
- Uma das formas de se resolver esse problema é adotar uma técnica conhecida como **Composition**:
  - Os componentes *containers* recebem *filhos* através da prop *children*.
  - A declaração dos filhos ocorre no mesmo arquivo da declaração do componente pai, evitando, assim, o problema do *prop drilling.*

O problema do *prop drilling*

- Projeto de referência.
- App.js:

```jsx
export default function App() {
  const fatherName = "Phil Dunphy";
  const motherName = "Claire Dunphy";
  const sonName = "Luke Dunphy";
  const daughterName = "Haley Dunphy";

  return (
    <div>
      {/* Prop drilling com "son" */}
      <Father name={fatherName} son={sonName} />

      {/* Composition, evitando o prop drilling */}
      <Mother name={motherName}>
        <Daughter>{daughterName}</Daughter>
      </Mother>
    </div>
  );
}
```

```jsx
function Father({ name, son }) {
  return (
    <ul>
      <li>Pai: {name}</li>
      <li>
        <Son son={son} />
      </li>
    </ul>
  );
}

function Son({ son }) {
  return <p>Filho: {son}</p>;
}
```

O problema do *prop drilling*

- Projeto de referência.
- App.js:

```jsx
export default function App() {
  const fatherName = "Phil Dunphy";
  const motherName = "Claire Dunphy";
  const sonName = "Luke Dunphy";
  const daughterName = "Haley Dunphy";

  return (
    <div>
      {/* Prop drilling com "son" */}
      <Father name={fatherName} son={sonName} />

      {/* Composition, evitando o prop drilling */}
      <Mother name={motherName}>
        <Daughter>{daughterName}</Daughter>
      </Mother>
    </div>
  );
}
```

```jsx
function Daughter({ children }) {
  return <p>Filha: {children}</p>;
}

function Mother({ name, children }) {
  return (
    <ul>
      <li>Mãe: {name}</li>
      <li>{children}</li>
    </ul>
  );
}
```

Prática

- Acompanhe o professor:
  - Criação do projeto **react-countries**.

Próxima aula

- Projeto **react-flash-cards**.