Professor Rodrigo Bossini
https://www.rodrigobossini.com.br

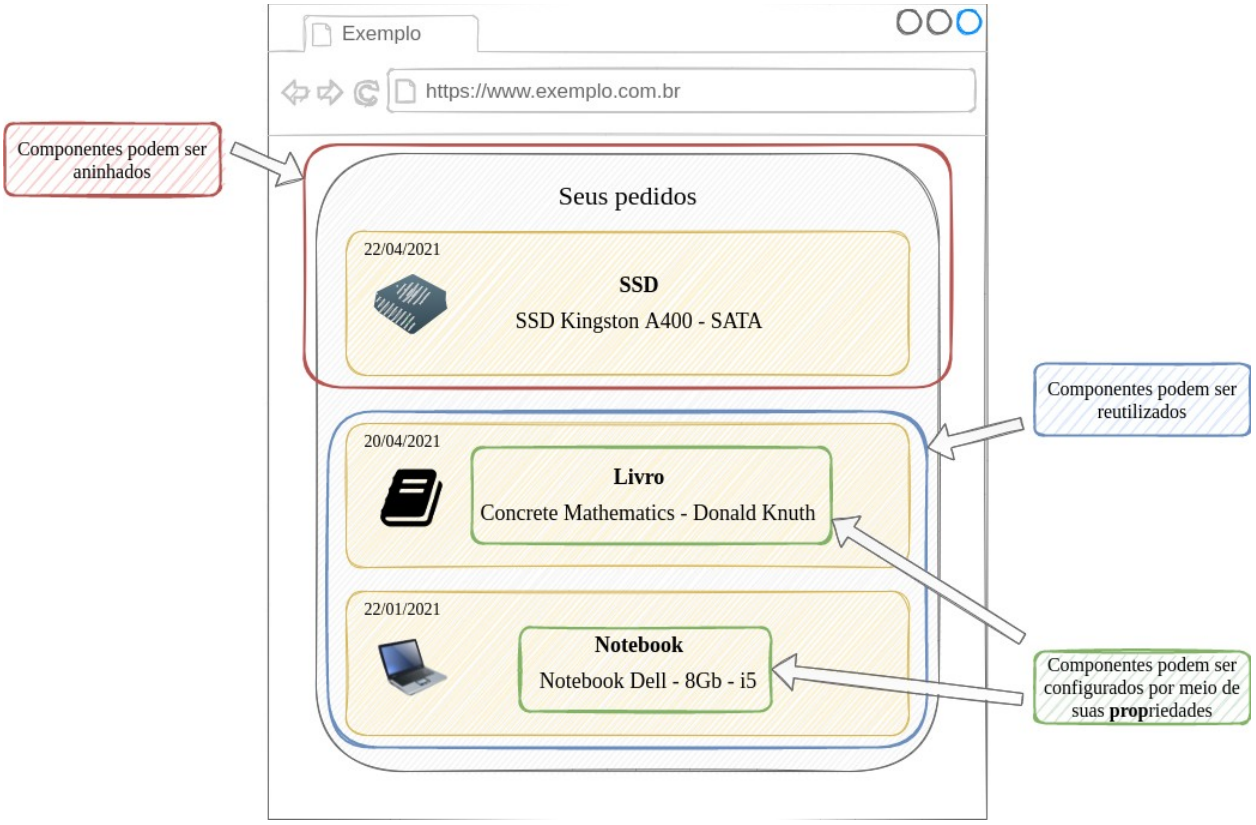
ReactJS
Componentes - Props

1 Introdução

Neste material, estudaremos mais sobre componentes React e sobre o mecanismo conhecido como **props** - que vem de **properties**. Trata-se de um mecanismo que permite a "entrega" de dados a componentes quando estes são utilizados por meio de sua tag. É análogo à passagem de argumentos entre funções, comum em qualquer linguagem de programação. A Figura 1.1 destaca algumas das principais características de componentes React. Assim como um componente React pode utilizar elementos HTML comuns, ele também pode utilizar outros componentes React previamente definidos. Ou seja, componentes React podem ser **aninhados**. Uma vez que um componente React tenha sido definido, ele pode ser utilizado inúmeras vezes. Ou seja, componentes React são **reutilizáveis**. Um componente React pode ser **configurado** por meio de suas **prop**riedades. A cada vez que é utilizado, um componente React pode se apresentar de forma diferente devido a valores diferentes que lhe foram entregues.

Figura 1.1

Produção: agosto de 2021



2 Desenvolvimento

Nesta seção, vamos ilustrar as características de componentes React destacadas desenvolvendo a aplicação que a Figura 1.1 exibe.

2.1 (Criando uma aplicação ReactJS) Use o comando

npx create-react-app react-componentes-props

para criar a aplicação. Caso seja a primeira execução do create-react-app usando o npx e a versão do npm seja 7+, a mensagem da Figura 2.1.1 deverá ser exibida. Basta confirmar para prosseguir.

Figura 2.1.1

```
Need to install the following packages:
  create-react-app
Ok to proceed? (y)
```

Nota. A ferramenta npx faz o download do pacote executado – neste caso, o create-react-app – e o mantém instalado em sua memória "cache" – um simples diretório em seu sistema de arquivos. O diretório utilizado depende de onde o NodeJS está instalado. No Linux com NodeJS instalado usando o nvm, ele pode ser encontrado em um diretório cujo path é parecido com **/home/rodrigo/.npm/_npx/**. Os arquivos ali existentes não fazem parte de sua aplicação React. São arquivos referentes ao pacote instalado apenas.

Depois de criar o projeto, use

```
cd react-componentes-props
```

para navegar até o diretório recém-criado. Abra uma instância do VS Code vinculada ao diretório atual com

```
code .
```

No VS Code, clique **Terminal >> New Terminal** para abrir um novo terminal.

Use

npm start

para colocar a aplicação em execução. A aplicação pode ser visualizada em **localhost:3000.** Apague todos os arquivos existentes na pasta **src**. A seguir, crie um arquivo chamado **index.js** na mesma pasta. Utilize o conteúdo do Bloco de Código 2.2.1 para criar um componente React simples.

Bloco de Código 2.2.1

```
import React from 'react'
import ReactDOM from 'react-dom'

const App = () => {
   return <div>Um componente</div>
}

ReactDOM.render(
   <App />,
   document.querySelector('#root')
)
```

2.3 (Usando o Bootstrap) Neste exemplo, utilizaremos o Bootstrap para tratar dos aspectos visuais da aplicação. Ele pode ser integrado à aplicação de diferentes formas:

- Podemos importá-lo via CDN no arquivo **public/index.html**, como mostra o Bloco de Código 2.3.1.

Bloco de Código 2.3.1

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta
      name="description"
      content="Web site created using create-react-app"
    />
    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
      <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet"
integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous">

    <title>React App</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
  </body>
</html>
```

- Também é possível instalar o Bootstrap com

npm install bootstrap

Feita a sua instalação, é preciso importar o conteúdo desejado no arquivo **src/index.js**, como ilustra o Bloco de Código 2.3.2.

Bloco de Código 2.3.2

```
import React from 'react'
import ReactDOM from 'react-dom'
import 'bootstrap/dist/css/bootstrap.min.css'

const App = () => {
   return <div>Um componente</div>
}

ReactDOM.render(
   <App />,
   document.querySelector('#root')
)
```

- Finalmente, também é possível fazer uso do Bootstrap por meio de pacotes como o **react-bootstrap**. Trata-se de uma implementação dos componentes do Bootstrap usando componentes React. Assim, cada componente do Bootstrap pode ser utilizado como um componente React. Veja mais sobre ele no Link 2.3.1.

Link 2.3.1
https://react-bootstrap.github.io/

Utilizaremos a segunda opção, ou seja, a instalação do bootstrap com npm. Por isso, ajuste seu código como destaca o Bloco de Código 2.3.2. Não é necessário, portanto, manter a alteração feita no arquivo public/index.html.

2.4 (Ícones do Font Awesome) Font Awesome é o nome de uma biblioteca que viabiliza o uso de ícones de alta qualidade. A sua página oficial pode ser encontrada por meio do Link 2.4.1.

Link 2.4.1
https://fontawesome.com/

Podemos habilitar o seu uso no projeto de diferentes formas. A importação via CDN e a instalação usando o **npm** são dois exemplos. Utilizaremos o segundo. Para tal, use

npm install --save @fortawesome/fontawesome-free

A seguir, ajuste o arquivo **index.js** como ilustra o Bloco de Código 2.4.1.

Bloco de Código 2.4.1

```
import React from 'react'
import ReactDOM from 'react-dom'
import 'bootstrap/dist/css/bootstrap.min.css'
import '@fortawesome/fontawesome-free/css/all.css'

const App = () => {
   return <div>Um componente</div>
}

ReactDOM.render(
   <App />,
   document.querySelector('#root')
)
```

2.5 (Criando a lista de três pedidos) Nesta seção, criamos a interface gráfica ilustrada na Figura 1.1.

- Começamos com um container responsivo do Bootstrap, como no Bloco de Código 2.5.1.

Bloco de Código 2.5.1

```
const App = () => {
   return (
      // container principal
      <div className="container border rounded mt-2">

      </div>
    );
}
```

- A seguir, adicionamos o título, como mostra o Bloco de Código 2.5.2.

Bloco de Código 2.5.2

```
const App = () => {
   return (
      // container principal
      <div className="container border rounded mt-2">

         {/* linha para o título */}
         <div className="row border-bottom m-2">
            <h1 className="display-5 text-center">Seus pedidos</h1>
         </div>

      </div>
    );
}
```

- A definição do primeiro pedido será feita em uma nova linha do bootstrap. Veja o Bloco e Código 2.5.3.

Bloco de Código 2.5.3

```
const App = () => {
   return (
      // container principal
      <div className="container border rounded mt-2">

         {/* linha para o título */}
         <div className="row border-bottom m-2">
            <h1 className="display-5 text-center">Seus pedidos</h1>
         </div>

         {/* linha para o primeiro pedido pedido*/}
         <div className="row">

         </div>
      </div>
    );
}
```

- Podemos utilizar algumas classes do modelo grid do Bootstrap para lidar com a responsividade, como no Bloco de Código 2.5.4.

Bloco de Código 2.5.4

```
const App = () => {
   return (
      // container principal
      <div className="container border rounded mt-2">

         {/* linha para o título */}
         <div className="row border-bottom m-2">
            <h1 className="display-5 text-center">Seus pedidos</h1>
         </div>

         {/* linha para o primeiro pedido pedido*/}
         <div className="row">
               {/* controle de colunas para responsividade*/}
               <div className="col-sm-8 col-md-6 m-2">

               </div>
         </div>
      </div>
    );
}
```

- Cada pedido será definido como um cartão do Bootstrap, com cabeçalho e corpo. Veja o Bloco de Código 2.5.5.

Bloco de Código 2.5.5

```
const App = () => {
   return (
      // container principal
      <div className="container border rounded mt-2">

         {/* linha para o título */}
         <div className="row border-bottom m-2">
            <h1 className="display-5 text-center">Seus pedidos</h1>
         </div>

         {/* linha para o primeiro pedido pedido*/}
         <div className="row">
               {/* controle de colunas para responsividade*/}
               <div className="col-sm-8 col-md-6 m-2">
               {/* cartão */}
               <div className="card">
                  {/* cabeçalho do cartão */}
                  <div className="card-header text-muted">22/04/2021</div>
                  {/* corpo do cartão */}
                  <div className="card-body d-flex">
                     <div className="d-flex align-items-center">
                        <i className="fas fa-hdd fa-2x"></i>
                     </div>
                     {/* flex-grow 1: tomar espaço remanescente */}
                     <div className="flex-grow-1 ms-2 border">
                        <h4 className="text-center">SSD</h4>
                        <p className="text-center">SSD Kingston A400 - SATA</p>
                     </div>
                  </div>
               </div>
            </div>
         </div>
      );
}
```

- O segundo pedido será definido de maneira análoga. Criamos uma linha do Bootstrap logo abaixo da linha do primeiro pedido. Veja o Bloco de Código 2.5.6.

Bloco de Código 2.5.6

```
const App = () => {
   return (
      // container principal
      <div className="container border rounded mt-2">

         {/* linha para o título */}
         <div className="row border-bottom m-2">
            <h1 className="display-5 text-center">Seus pedidos</h1>
         </div>

         {/* linha para o primeiro pedido pedido*/}
         <div className="row ">
               ...
          </div>

          {/* linha para o segundo pedido pedido*/}
         <div className="row">
            {/* controle de colunas para responsividade*/}
            <div className="col-sm-8 col-md-6 m-2">
               {/* cartão */}
               <div className="card">
                  {/* cabeçalho do cartão */}
                  <div className="card-header text-muted">20/04/2021</div>
                  {/* corpo do cartão */}
                  <div className="card-body d-flex">
                     <div className="d-flex align-items-center">
                        <i className="fas fa-book fa-2x"></i>
                     </div>
                     {/* flex-grow 1: tomar espaço remanescente */}
                     <div className="flex-grow-1 ms-2 border">
                        <h4 className="text-center">Livro</h4>
                        <p className="text-center">Concrete Mathematics - Donald Knuth</p>
                     </div>
                  </div>
               </div>
            </div>
         </div>
      );
}
```

- O terceiro pedido é definido de maneira análoga aos demais, como no Bloco de Código 2.5.7.

Bloco de Código 2.5.7

```
const App = () => {
   return (
      // container principal
      <div className="container border rounded mt-2">

         {/* linha para o título */}
         <div className="row  border-bottom m-2">
            <h1 className="display-5 text-center">Seus pedidos</h1>
         </div>

         {/* linha para o primeiro pedido pedido*/}
         <div className="row ">
               ...
          </div>

          {/* linha para o segundo pedido pedido*/}
         <div className="row">
            ...
         </div>

         {/* linha para o terceiro pedido pedido*/}
         <div className="row">
            {/* controle de colunas para responsividade*/}
            <div className="col-sm-8 col-md-6 m-2">
               {/* cartão */}
               <div className="card">
                  {/* cabeçalho do cartão */}
                  <div className="card-header text-muted">21/01/2021</div>
                  {/* corpo do cartão */}
                  <div className="card-body d-flex">
                     <div className="d-flex align-items-center">
                        <i className="fas fa-laptop fa-2x"></i>
                     </div>
                     {/* flex-grow 1: tomar espaço remanescente */}
                     <div className="flex-grow-1 ms-2 border">
                        <h4 className="text-center">Notebook</h4>
                        <p className="text-center">Notebook Dell - 8Gb - i5</p>
```

```
                    </div>
                </div>
            </div>
        </div>
    </div>

    </div>
  );
}
```

2.6 (Refatorando a aplicação) O código HTML que define cada pedido é essencialmente igual aos demais, a menos do conteúdo. Trata-se de um bloco de código que desejamos reutilizar em diferentes partes da aplicação. Esse tipo de código deve ser definido como um componente React à parte. Vejamos como fazê-lo.

- Comece criando um arquivo chamado **Pedido.js** na pasta **src**. Seu conteúdo inicial aparece no Bloco de Código 2.6.1.

Bloco de Código 2.6.1

```
import React from 'react'
import ReactDOM from 'react-dom'

const Pedido = () => {

}
```

- A seguir, **recorte** a definição do primeiro pedido existente no arquivo **index.js** e **cole** no arquivo **Pedido.js**, como mostra o Bloco de Código 2.6.2.

Bloco de Código 2.6.2

```
import React from 'react'

const Pedido = () => {
  return (
    <div className="card">
    {/* cabeçalho do cartão */}
    <div className="card-header text-muted">22/04/2021</div>
    {/* corpo do cartão */}
    <div className="card-body d-flex">
      <div className="d-flex align-items-center">
        <i className="fas fa-hdd fa-2x"></i>
      </div>
      {/* flex-grow 1: tomar espaço remanescente */}
      <div className="flex-grow-1 ms-2 border">
        <h4 className="text-center">SSD</h4>
        <p className="text-center">SSD Kingston A400 - SATA</p>
      </div>
    </div>
  </div>
  )
}

export default Pedido;
```

- O arquivo **index.js**, no momento, não possui mais a definição que foi recortada. Veja o Bloco de Código 2.6.3.

Bloco de Código 2.6.3

```
...
{/* linha para o primeiro pedido pedido*/}
<div className="row">
   {/* controle de colunas para responsividade*/}
   <div className="col-sm-8 col-md-6 m-2">
     {/* essa região fica sem conteúdo algum, por enquanto*/}
   </div>
</div>
...
```

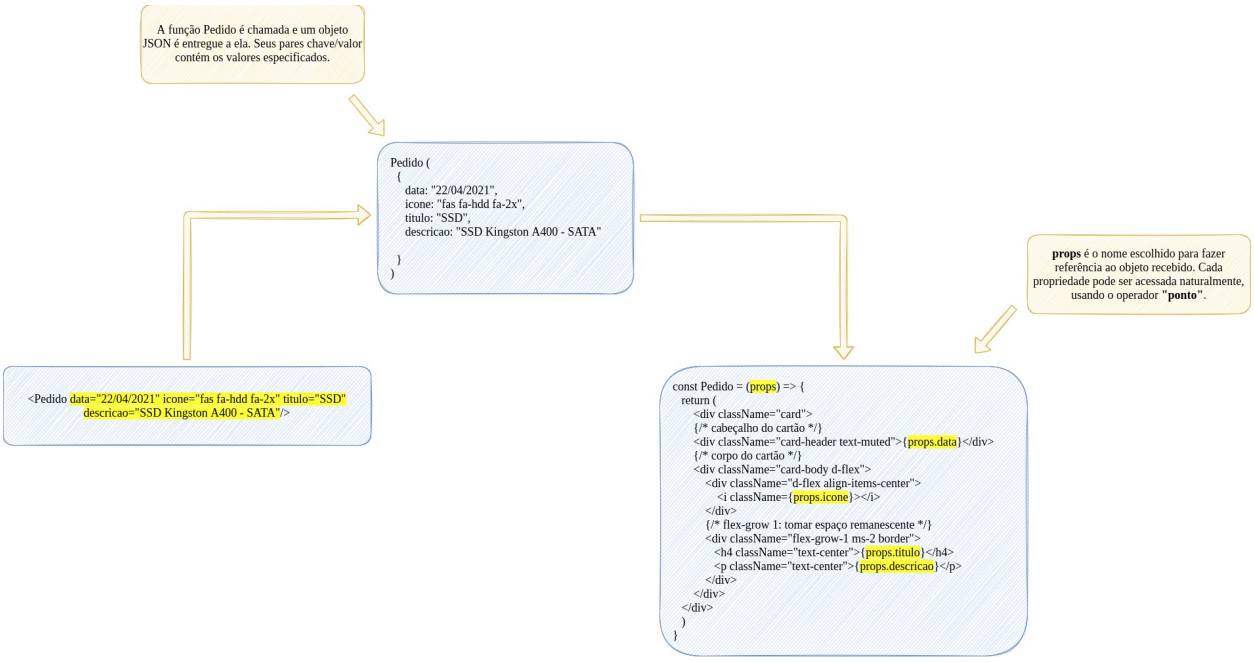
- O nome de um componente React pode ser utilizado de maneira semelhante ao uso de uma tag HTML. Quando fazemos isso, a função que o define é colocada em execução, produzindo a expressão JSX que especificamos. Assim, o componente React '**Pedido**' pode ser utilizado no arquivo **index.js** como destaca o Bloco de Código 2.6.4.

Bloco de Código 2.6.4

```
import React from 'react'
import ReactDOM from 'react-dom'
import 'bootstrap/dist/css/bootstrap.min.css'
import '@fortawesome/fontawesome-free/css/all.css'
import Pedido from './Pedido'
        ...
        {/* linha para o primeiro pedido pedido*/}
        <div className="row">
          {/* controle de colunas para responsividade*/}
          <div className="col-sm-8 col-md-6 m-2">
            <Pedido />
          </div>
        </div>
        ...
```

2.7 (O mecanismo props) Embora tenhamos refatorado a aplicação criando um componente próprio para a representação de pedidos, ele ainda não é realmente reutilizável. Afinal, seu conteúdo, como nome, descrição e ícone referentes ao pedido, está fixo. Precisamos de um mecanismo que viabilize a especificação do conteúdo a ser utilizado pelo componente no momento em que o utilizamos. Estamos falando do mecanismo conhecido como **props**. Trata-se de um mecanismo extremamente simples: quando utilizamos um componente por meio de sua tag, especificamos pares chave/valor que ele poderá utilizar internamente. Essas são as suas **prop**riedades. Daí o nome. Funciona exatamente da mesma forma como a passagem de argumentos para uma função, quando a colocamos em execução. Os valores especificados são "empacotados" em um objeto JSON que pode ser acessado pelo componente. A Figura 2.7.1 ilustra a ideia.

Figura 2.7.1



- Assim, podemos reutilizar o componente para definir os três pedidos da lista. A sua definição, feita no arquivo **Pedido.js**, aparece no Bloco de Código 2.7.1.

Nota. O nome **props** é uma convenção. É possível utilizar qualquer outro identificador válido.

Bloco de Código 2.7.1

```
import React from 'react'

const Pedido = (props) => {
  return (
    <div className="card">
    {/* cabeçalho do cartão */}
    <div className="card-header text-muted">{props.data}</div>
    {/* corpo do cartão */}
    <div className="card-body d-flex">
      <div className="d-flex align-items-center">
        <i className={props.icone}></i>
      </div>
      {/* flex-grow 1: tomar espaço remanescente */}
      <div className="flex-grow-1 ms-2 border">
        <h4 className="text-center">{props.titulo}</h4>
        <p className="text-center">{props.descricao}</p>
      </div>
    </div>
  </div>
  )
}
export default Pedido;
```

- O componente Pedido será utilizado três vezes no arquivo **index.js**. Uma vez para cada pedido da lista. Cada um tem conteúdo distinto, que será especificado via props. Veja o Bloco de Código 2.7.2.

Bloco de Código 2.7.2

```
...
{/* linha para o primeiro pedido pedido*/}
<div className="row">
    {/* controle de colunas para responsividade*/}
    <div className="col-sm-8 col-md-6 m-2">
        <Pedido data="22/04/2021" icone="fas fa-hdd fa-2x" titulo="SSD"
                descricao="SSD Kingston A400 - SATA"/>
    </div>
</div>

{/* linha para o segundo pedido pedido*/}
<div className="row">
    {/* controle de colunas para responsividade*/}
    <div className="col-sm-8 col-md-6 m-2">
        <Pedido data="20/04/2021" icone="fas fa-book fa-2x" titulo="Livro"
                descricao="Concrete Mathematics - Donald Knuth" />
    </div>
</div>

{/* linha para o terceiro pedido pedido*/}
<div className="row">
    {/* controle de colunas para responsividade*/}
    <div className="col-sm-8 col-md-6 m-2">
        <Pedido data="21/01/2021" icone="fas fa-laptop fa-2x" titulo="Notebook"
                descricao="Notebook Dell - 8Gb - i5" />
    </div>
</div>
...
```

2.8 (Nova refatoração: componente Cartao) "Cartões" são elementos muito utilizados por aplicações para exibir conteúdo. Podemos promover ainda mais o nível de reusabilidade de nossos componentes criando um componente que seja capaz de representar cartões de maneira independente, não estando atrelado à exibição de pedidos. Para fazê-lo, crie um arquivo chamado **Cartao.js** na pasta **src**. Seu conteúdo aparece no Bloco de Código 2.8.1. Repare como deixamos

Bloco de Código 2.8.1

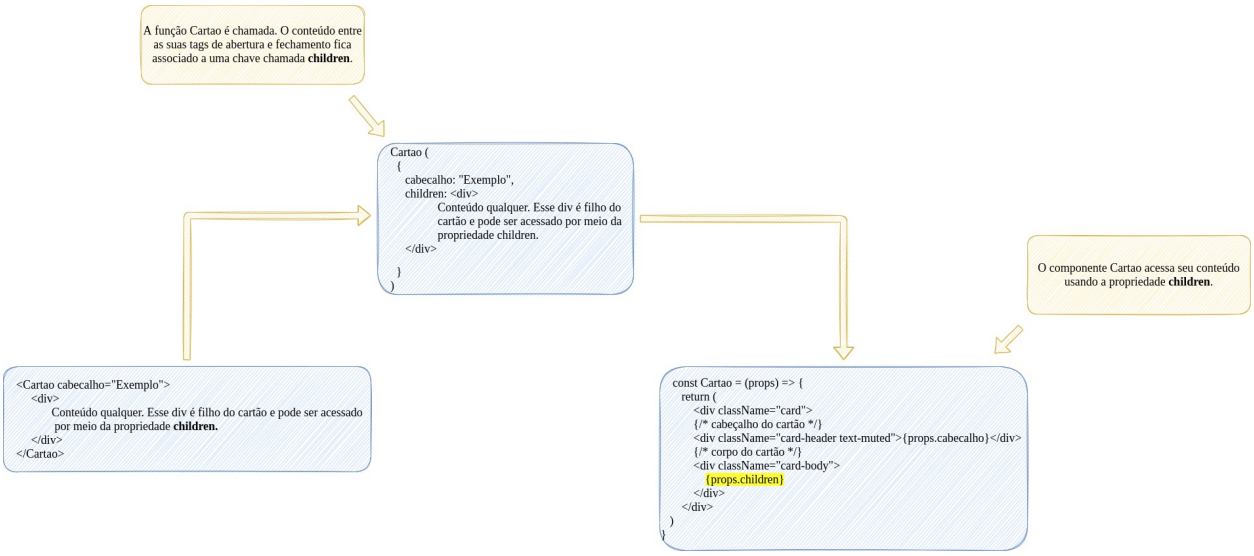
```
import React from 'react'

const Cartao = (props) => {
  return (
     <div className="card">
     {/* cabeçalho do cartão */}
     <div className="card-header text-muted">{props.cabecalho}</div>
     {/* corpo do cartão */}
     <div className="card-body">
        {props.children}
     </div>
  </div>
  )
}

export default Cartao;
```

Repare nos detalhes destacados. Deixamos em aberto o conteúdo do cabeçalho do cartão, o qual será entregue via props. Além disso, utilizamos a propriedade implícita chamada **children**. Ela representa os elementos "filhos" de um Cartao, especificados no momento de seu uso. A Figura 2.8.1 ilustra esse detalhe.

Figura 2.8.1



O Bloco de Código 2.8.2 destaca o uso do componente Cartao no arquivo **index.js**.

Bloco de Código 2.8.2

```
import React from 'react'
import ReactDOM from 'react-dom'
import 'bootstrap/dist/css/bootstrap.min.css'
import '@fortawesome/fontawesome-free/css/all.css'
import Pedido from './Pedido'
import Cartao from './Cartao'
...
{/* linha para o primeiro pedido pedido*/}
        <div className="row">
            {/* controle de colunas para responsividade*/}
            <div className="col-sm-8 col-md-6 m-2">
                <Cartao cabecalho="22/04/2021">
                    <Pedido icone="fas fa-hdd fa-2x" titulo="SSD" descricao="SSD Kingston A400
- SATA"/>
                </Cartao>
            </div>
        </div>

        {/* linha para o segundo pedido pedido*/}
        <div className="row">
            {/* controle de colunas para responsividade*/}
            <div className="col-sm-8 col-md-6 m-2">
                <Cartao  cabecalho="20/04/2021">
                    <Pedido icone="fas fa-book fa-2x" titulo="Livro" descricao="Concrete
Mathematics - Donald Knuth" />
                </Cartao>
            </div>
        </div>

        {/* linha para o terceiro pedido pedido*/}
        <div className="row">
            {/* controle de colunas para responsividade*/}
            <div className="col-sm-8 col-md-6 m-2">
                <Cartao cabecalho="21/01/2021">
                    <Pedido icone="fas fa-laptop fa-2x" titulo="Notebook" descricao="Notebook
Dell - 8Gb - i5" />
                </Cartao>
            </div>
        </div>
```

A essa altura, a definição do componente Pedido ainda inclui os detalhes referentes à definição de um cartão Bootstrap. Para completar esse passo da refatoração, precisamos removê-las, como no Bloco de Código 2.8.3.

Bloco de Código 2.8.3

```
const Pedido = ({icone, titulo, descricao}) => {
  return (
     <div className="d-flex">
       <div className="d-flex align-items-center">
       <i className={`${icone}`}></i>
       </div>
       <div className="flex-grow-1 ms-2 border">
         <h4 className="text-center">{titulo}</h4>
         <p className="text-center">{descricao}</p>
       </div>
     </div>
  )
}
export default Pedido
```

2.9 (Componente para Feedback) Temos uma nova funcionalidade a ser implementada: Cada pedido deve exibir botões para que o usuário possa confirmar se a entrega já ocorreu. Podemos, evidentemente, construir um componente React para isso. Ele será um componente que exibe dois botões. Um servirá para que o usuário confirme a entrega e, o outro, para dizer que a entrega ainda não aconteceu. Pensando em reusabilidade, deixaremos os textos que cada botão exibe bem como as funções que são chamadas quando clicados em aberto. Crie um arquivo chamado **Feedback.js** na pasta **src**. Seu conteúdo aparece no Bloco de Código 2.9.1.

Bloco de Código 2.9.1

```
import React from 'react';

const Feedback = props => {
   return (
      <div className="d-flex justify-content-evenly m-2">
         <button
            type="button"
            onClick={props.funcaoOK}
            className="btn btn-primary">
            {props.textoOK}
         </button>
         <button
            type="button"
            onClick={props.funcaoNOK}
            className="btn btn-danger">
            {props.textoNOK}
         </button>
      </div>
   )
}

export default Feedback;
```

No arquivo **index.js**, cada componente Pedido terá um "irmão" - filho do mesmo Cartao - do tipo Feedback. Veja o Bloco de Código 2.9.2.

Bloco de Código 2.9.2

```
…
import Feedback from './Feedback'
…
const App = () => {
   const textoOK = "Já chegou"
   const textoNOK = "Ainda não chegou"
   const funcaoOK = () => alert ("Agradecemos a confirmação!")
   const funcaoNOK = () => alert ("Verificaremos o ocorrido!")
const componenteFeedback = <Feedback textoOK={textoOK} funcaoOK={funcaoOK}
textoNOK={textoNOK} funcaoNOK={funcaoNOK}/>;
   return (
…
{/* linha para o primeiro pedido pedido*/}
         <div className="row">
            {/* controle de colunas para responsividade*/}
            <div className="col-sm-8 col-md-6 m-2">
               <Cartao cabecalho="22/04/2021">
                  <Pedido icone="fas fa-hdd fa-2x" titulo="SSD" descricao="SSD Kingston A400
- SATA"/>
                  {componenteFeedback}
               </Cartao>
            </div>
         </div>

         {/* linha para o segundo pedido pedido*/}
         <div className="row">
            {/* controle de colunas para responsividade*/}
            <div className="col-sm-8 col-md-6 m-2">
               <Cartao  cabecalho="20/04/2021">
                  <Pedido icone="fas fa-book fa-2x" titulo="Livro" descricao="Concrete
Mathematics - Donald Knuth" />
                  {componenteFeedback}
               </Cartao>
            </div>
         </div>

         {/* linha para o terceiro pedido pedido*/}
         <div className="row">
            {/* controle de colunas para responsividade*/}
            <div className="col-sm-8 col-md-6 m-2">
```

```
                <Cartao cabecalho="21/01/2021">
                    <Pedido icone="fas fa-laptop fa-2x" titulo="Notebook" descricao="Notebook
Dell - 8Gb - i5" />
                    {componenteFeedback}
                </Cartao>
            </div>
        </div>
```

Referências

React - A JavaScript library for building user interfaces. 2021. Disponível em <https://reactjs.org/>. Acesso em agosto de 2021.