Criando Componentes Compartilhados em React

Por Escola Dnc

Introdução

Este ebook aborda a criação de componentes compartilhados em React, com foco no desenvolvimento de uma aplicação web. Vamos explorar a criação de um componente Hero, discutir boas práticas de estruturação de projetos e abordar aspectos importantes como responsividade e estilização.

Criando o Componente Hero

Estrutura Básica

O componente Hero é um elemento fundamental em muitas páginas web, geralmente ocupando uma posição de destaque na parte superior da página. Para criar este componente:

1. Crie uma nova pasta chamada `Hero` dentro da pasta `components`
2. Adicione dois arquivos: `hero.jsx` e `hero.css`
3. Importe os estilos necessários e crie a estrutura básica do componente:

```
import React from 'react';import './hero.css';const Hero = () => {
return (    <div className="hero dflex alignCenter">      {/*
Conteúdo do Hero */}    </div>  );};export default Hero;
```

Conteúdo do Hero

O conteúdo do Hero geralmente inclui:

- Um título principal (h1)
- Um parágrafo de texto explicativo
- Um botão de call-to-action

Vamos adicionar esses elementos:

```
<div className="hero dflex alignCenter">  <div className="hero-
text">    <h1>Título Principal em Inglês</h1>    <p>Texto
explicativo sobre a aplicação</p>    <Button
buttonStyle="secondary" arrow>      Get Started    </Button>
</div></div>
```

Observação: Estamos utilizando um componente `Button` que foi previamente criado como um componente compartilhado.

Adicionando Navegação

Para tornar o botão funcional, vamos envolver o `Button` com um componente `Link` do React Router:

```
import { Link } from 'react-router-dom';// ...<Link to="/about">
<Button buttonStyle="secondary" arrow>    Get Started  </Button>
</Link>
```

Estilização do Componente Hero

A estilização é crucial para que o Hero tenha o impacto visual desejado. Vamos adicionar os estilos no arquivo `hero.css`:

```
.hero {  background-image: url('/hero.jpg');  background-position: center;  background-size: cover;  border-radius: 70px;  height: 708px;  padding: 50px;}.hero h1 {  font-size: 65px;  line-height: 125%;}.hero-text {  width: 40%;}/* Media queries para responsividade */@media (max-width: 768px) {  .hero-text {  width: 100%;  }}@media (max-width: 568px) {  .hero h1 {    font-size: 50px;  }}
```

Pontos importantes:

- Utilizamos a imagem do Hero diretamente da pasta `public`
- Definimos um tamanho fixo para o Hero
- Adicionamos media queries para melhorar a responsividade

Boas Práticas e Considerações

Estrutura de Pastas

É importante manter uma estrutura de pastas organizada:

```
src/  components/    Hero/      hero.jsx      hero.css    Button/
button.jsx      button.css    // Outros componentes compartilhados
pages/    // Componentes de página  styles/    main.css  // Outros
arquivos e pastas
```

Uso da Pasta Public

A pasta `public` é útil para:

- Armazenar imagens e outros assets que precisam ser acessados diretamente
- Facilitar o acesso a esses arquivos tanto no CSS quanto no JavaScript

Flexibilidade dos Componentes

Ao criar componentes compartilhados, é importante mantê-los flexíveis:

- Evite adicionar o componente diretamente dentro de um container
- Use props para permitir customização
- Considere diferentes cenários de uso

Responsividade

Adicione media queries proativamente para garantir uma boa experiência em dispositivos móveis:

- Ajuste tamanhos de fonte
- Modifique layouts conforme necessário
- Teste em diferentes tamanhos de tela

Integrando o Componente Hero

Para utilizar o componente Hero em uma página:

1. Importe o componente:

```
import Hero from '../components/Hero/hero';
```

2. Adicione o componente à estrutura da página:

```
<div className="container">  <Hero />  {/* Outros componentes da página */}</div>
```

Dica: Ao adicionar o Hero dentro de um container, ele se ajustará ao layout definido no Figma.

Conclusão

A criação de componentes compartilhados como o Hero é fundamental para o desenvolvimento eficiente de aplicações React. Ao seguir boas práticas de organização, estilização e responsividade, você cria uma base sólida para seu projeto.

Próximos passos:

- Criar mais componentes compartilhados (ex: lista de projetos)
- Refinar a responsividade
- Implementar o menu mobile

Lembre-se de revisar constantemente a estrutura do seu projeto e manter a consistência na criação e uso dos componentes compartilhados.