

Tarefa 01 - Construção e Reutilização de Componentes

Objetivo da Atividade

Esta atividade prática tem como objetivo transformar uma página estática, previamente criada em sala de aula, em uma aplicação React modularizada. O foco é estruturar corretamente os componentes e pages, garantindo a reutilização do código e facilitando sua manutenção.

Os alunos deverão organizar a aplicação utilizando componentes reutilizáveis, modelando cada um com um arquivo `index.jsx` para a lógica e um `module.css` para a estilização. Além disso, a estrutura de pastas deve seguir boas práticas, tornando o projeto mais organizado e escalável.

A formatação e estilização são baseadas no seguinte código CSS, que pode ser consultado e adaptado para a criação dos componentes e pages.

Estrutura do Projeto

```
src/
├── assets/
├── components/
│   ├── Aside/
│   │   ├── index.jsx
│   │   └── Aside.module.css
│   ├── Band/
│   │   ├── index.jsx
│   │   └── Band.module.css
│   ├── Footer/
│   │   ├── index.jsx
│   │   └── Footer.module.css
│   ├── Header/
│   │   ├── index.jsx
│   │   └── Header.module.css
│   ├── Main/
│   │   ├── index.jsx
│   │   └── Main.module.css
├── pages/
│   ├── Home/
│   │   ├── index.jsx
│   │   └── Home.module.css
├── App.jsx
├── index.jsx
└── main.jsx
```

Componentes

- **Header:** Cabeçalho da aplicação.
- **Footer:** Rodapé da aplicação.
- **Aside:** Barra lateral com informações adicionais.
- **Band:** Seção reutilizável para exibir conteúdos específicos.
- **Main:** Área principal da aplicação.

Cada componente deve ser estruturado em uma pasta própria dentro de `components/`, contendo um arquivo `index.jsx` para a lógica do componente e um `module.css` para sua estilização.

Estilização Base

A seguir, está o código CSS completo que pode ser utilizado como base para consulta e adaptação na criação dos componentes e pages:

```
:root {
  --cor-fundo: #333;
  --cor-texto: #fff;
  --cor-header: #222;
  --cor-main: #111;
  --cor-secao: #222;
  --cor-borda: #444;

  --fonte-padrao: Arial, sans-serif;
  --tamanho-h1: 2rem;
  --tamanho-h2: 1.5rem;
  --tamanho-p: 1rem;

  --padding-padrao: 1rem;
  --padding-header: 2rem;
  --margin-padrao: 1rem;

  --largura-max-main: 75rem;
  --largura-imagens: 6.25rem;
  --altura-imagens: 6.25rem;
  --border-radius: 0.3125rem;
  --border-size: 0.0625rem;
}

* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

body {
  font-family: var(--fonte-padrao);
  background-color: var(--cor-fundo);
  color: var(--cor-texto);
}

header {
  background-color: var(--cor-header);
  text-align: center;
```

```
text-align: center;
padding: var(--padding-header);
}

header img {
width: var(--largura-imagens);
height: var(--altura-imagens);
object-fit: cover;
}

h1 {
font-size: var(--tamanho-h1);
}

h2 {
font-size: var(--tamanho-h2);
margin: 0;
}

p {
font-size: var(--tamanho-p);
text-align: justify;
}

main {
max-width: var(--largura-max-main);
margin: var(--padding-header) auto;
background-color: var(--cor-main);
padding: var(--padding-header);
display: flex;
flex-wrap: wrap;
}

section {
flex: 1;
margin: var(--margin-padrao);
padding: var(--padding-padrao);
background-color: var(--cor-secao);
border: var(--border-size) solid var(--cor-borda);
border-radius: var(--border-radius);
}

.band {
display: flex;
align-items: center;
margin-bottom: var(--margin-padrao);
}

.band img {
width: var(--largura-imagens);
height: var(--altura-imagens);
object-fit: cover;
margin-right: var(--margin-padrao);
}

.band-info {
```

```
    flex: 1;
  }

  aside {
    margin: var(--margin-padrao);
    padding: var(--padding-padrao);
    background-color: var(--cor-secao);
    border: var(--border-size) solid var(--cor-borda);
    border-radius: var(--border-radius);
  }

  footer {
    position: fixed;
    bottom: 0;
    width: 100vw;
    text-align: center;
    padding: var(--padding-padrao);
    background-color: var(--cor-header);
  }
}
```



Dicas

- **Quebre o projeto em pequenos componentes reutilizáveis:** Sempre que um elemento se repete na interface, transforme-o em um componente separado.
- **Organize a estrutura de pastas:** Mantenha a separação entre componentes, pages e estilos para facilitar a manutenção do código.
- **Utilize CSS Modules para estilização:** Evite conflitos de estilos utilizando arquivos `.module.css` para cada componente.
- **Siga boas práticas de nomeação:** Nomeie arquivos e pastas de forma clara e consistente.



O Componente Band e o Uso de Props

No React, **props** (propriedades) são um mecanismo que permite passar dados de um **componente pai** para um **componente filho**. Elas são imutáveis dentro do componente filho, garantindo que os dados sejam controlados externamente e permitindo a reutilização de componentes.

O componente **Band** utiliza **props** para receber e exibir dinamicamente informações sobre diferentes bandas. Isso significa que, ao invés de criar um componente separado para cada banda, podemos reutilizar o mesmo componente **Band** e simplesmente passar diferentes valores como **props**.

◆ Benefícios do uso de **props** no **Band**

- **Reutilização:** O mesmo componente pode ser utilizado para exibir várias bandas sem duplicação de código.
- **Flexibilidade:** Permite alterar o conteúdo da página sem modificar a estrutura do componente.

- **Escalabilidade:** Facilita a adição de novas bandas ao sistema sem necessidade de grandes mudanças no código.

◆ Como **props** são aplicadas no **Band**?

1. Um **array de bandas** é definido no componente pai (por exemplo, **Main**).
2. O **método** `.map()` é utilizado para percorrer esse array e passar os dados de cada banda como **props** para **Band**.
3. O **componente** **Band** recebe essas **props** e exibe as informações dinamicamente.

Essa abordagem modular melhora a organização do código e torna a aplicação mais eficiente e de fácil manutenção.