front-end

prof. Lucas Ferreira

engenharia de software

engenharia de computação

Roteamento entre telas com React.js

em aplicações 100% front-end

Breve introdução

Uma reflexão sobre os tipos de um "projeto" e o que o React.js nos propõe.

O React.js *(junto de Vite.js, CRA e etc)* não pressupõe que todo e qualquer projeto terá mais de uma tela, e muito menos navegações completissimas.

Você pode estar fazendo uma simples lista de compras com no máximo 1 tela e poucas "janelinhas".

Logo não teremos nada "de fábrica" para cumprir a trivial tarefa de "trocar de telas".

Breve introdução

Óbvio, acredito que 90% dos projetos "por aí a fora" precisarão de um sistema de navegação e troca de telas, aí se o React.js não nos ajuda por padrão, precisaremos contar com "outros amiguinhos".

Existem outros projetos como o Remix.run, Next.js e o Gatsby que já "vem de fábrica" com opções de múltiplas telas/páginas e rotas de navegações facilitadas.

E no caso de projetos básicos, baseados 100% em front-end como o que estamos fazendo em Vite.js até aqui, podemos por exemplo usar o **React Router**.

React Router

```
npm install react-router-dom
```

React Router

O **React Router** é sem sombra de dúvidas umas das bibliotecas mais populares a ser usada em conjunto com React.js para web em projetos baseados 100% em front-end.

Têm como objetivo básico resolver da forma mais simples possível a tarefa de rotear links e exibir determinados componentes de acordo com a URL/endereço atual de navegação dentro de um web app.

Sua instalação em um projeto baseado em CRA é super fácil, basta rodarmos o seguinte comando:

```
npm install react-router-dom
```

React Router

Vale adiantar e observar que o React Router como projeto é algo muito "grande" e com diversas possibilidades de uso, inclusive ele é usado por baixo dos panos em outro projeto muito legal chamado **Remix.run**.

Recomendo que vocês, caso queiram se aprofundar mesmo acessem o site da documentação: https://reactrouter.com/en/main/start/concepts

React Router

Após instalarmos, começando do básico, deveremos preparar o "terreno" de nosso app para exibir ou esconder "telas" de acordo com nossa rota.

Primeiro precisaremos do **RouterProvider** para delimitarmos a área "navegável" de nosso app.

Além deles recomenda-se usarmos uma função chamada ***createBrowserRouter*** para criar a primeira estrutura de telas a ser incorporada em nível de navegação entre telas.

```
import {
  createBrowserRouter,
  RouterProvider
} from "react-router-dom";
```

React Router

Após a importação, basta organizar nosso *App.js* (ou outro arquivo principal).

A função **createBrowserRouter** irá listar todas as possíveis telas e seus caminhos de navegação *(endereço/path)*.

E o componente **RouterProvider** deverá ser inicializado em tela recebendo como parâmetro o ***router*** criado na função anterior.

```
const router = createBrowserRouter([
  {
    path: '/',
    element: <Home />,
  },
]);

export default function App() {
  return <RouterProvider router={router} />;
}
```

React Router

Caso seja necessário mudar uma URL (e tela) de forma programática podemos usar o hooks ***useNavigate*** que vem no pacote do React Router.

```
import { useNavigate } from "react-router-dom";

function HomeButton() {
  let navigate = useNavigate();

  function handleClick() {
    navigate("/home");
  }

  return (
    <button type="button" onClick={handleClick}>
      Go home
    </button>
  );
}
```

React Router

Recomenda-se que na hora de montarmos nosso "menu" os links navegáveis seja feitos usando o *component* **Link**.

```
import { Link } from "react-router-dom";

return (
  <nav>
    <ul>
      <li>
        <Link to="/">Home</Link>
      </li>
      <li>
        <Link to="/teste">Teste</Link>
      </li>
    </ul>
  </nav>
);
```

Rotas Dinâmicas

Para complementarmos o pacote básico, existe a possibilidade muito grande de termos que passar parâmetros junto das URLs navegáveis do nosso app, por exemplo uma lista de produtos que ao clicados irá exibir mais detalhes do mesmo.

Listagem => localhost:3000/produtos

Exibe o Produto #14 => localhost/produtos/view/14

E como podemos interceptar parâmetros passados na URL de nossa rota?

Através do hook useParams().

Rotas Dinâmicas

Primeiro devemos definir nossa Route como uma Route que recebe parâmetros:

```
const router = createBrowserRouter([
  {
    path: 'produtos/view/:id',
    element: <ProdutoView />,
  },
]);
```

Depois podemos recuperar no local desejado, os parâmetros caso preenchidos na URL:

```
import { useParams } from "react-router-dom";

function ProdutoView() {
  let { id } = useParams();
  return <div>Now showing product {id}</div>;
}
```

obrigado 🚀