

React

Vamos aprender a usar o framework do react

Primeiros usamos o comando: **npm i create-react-app** .

Para criar uma pasta toda estruturada para nosso projeto

```
public > index.html > ...
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8" />
5     <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
6     <title>React App</title>
7   </head>
8   <body>
9     <noscript>You need to enable JavaScript to run this app.</noscript>
10    <div id="root"></div>
11  </body>
12 </html>
```

```
src > index.js > ...
1 import React from 'react';
2 import ReactDOM from 'react-dom/client';
3 import './index.css';
4 import App from './App';
5
6 const root = ReactDOM.createRoot(document.getElementById('root'));
7 root.render(
8   <React.StrictMode>
9     <App />
10   </React.StrictMode>
11 );
12 // <React.StrictMode> Ele serve apenas para identificar possíveis problemas durante o desenvolvimento.
```

Mas como o js é integrado no html sem a tag `<script>`?

há uma interação importante entre o React e o arquivo index.html, mesmo sem a tag `<script>` explícita. Isso acontece porque em projetos React modernos (criados com Create React App, Vite ou similares), o sistema de build cuida dessa integração automaticamente. Vou explicar como isso funciona:

O que acontece por baixo dos panos

Ponto de entrada: O `<div id="root"></div>` no HTML é o container onde sua aplicação React será renderizada.

ReactDOM.createRoot: No seu arquivo JavaScript (index.js),

A linha **ReactDOM.createRoot(document.getElementById('root'))** encontra esse elemento div e prepara-o para receber a aplicação React.

Processo de build: Quando você executa `npm start` ou `npm run build`, as ferramentas de build (como webpack no caso do Create React App) fazem o seguinte:

Empacotam todo seu código React em arquivos JavaScript otimizados
Injetam automaticamente as tags `<script>` necessárias no HTML final
Geram os arquivos finais na pasta `build/` (ou similar)

O que é Build em um Projeto React?

Build é o processo de transformar seu código React (JSX, JavaScript moderno, CSS, assets) em arquivos otimizados que o navegador pode entender e executar. Isso inclui:

Transpilar código (ex: JSX → JavaScript puro, ES6+ → ES5 para compatibilidade)

Agrupar (bundle) módulos em arquivos menores e eficientes

Minificar código (remover espaços, encurtar nomes de variáveis)

Otimizar imagens e assets

Gerar HTML, CSS e JS prontos para produção

O que é Webpack?

Webpack é um module bundler (empacotador de módulos) para JavaScript. Ele pega vários arquivos (JavaScript, CSS, imagens, etc.) e os combina em pacotes otimizados para o navegador.

Por que o Webpack é usado no React?

Quando você escreve código React, geralmente:

Usa JSX (que não é entendido pelo navegador).

Divide o código em múltiplos arquivos e dependências.

Usa CSS, imagens, fonts e outros recursos.

O Webpack ajuda a:

- ✓ Transpilar JSX e ES6+ para JavaScript compatível com navegadores.
- ✓ Agrupar (bundle) todos os arquivos em um ou mais pacotes otimizados.
- ✓ Gerenciar dependências (como bibliotecas do `node_modules`).
- ✓ Otimizar código para produção (minificação, tree shaking).
- ✓ Carregar assets (CSS, imagens, fonts) como módulos.

```
src > .js App.js > ...
1 //Componentes / react
2 import First from './components/FirstComponents.js';
3 //Style / CSS
4 import './App.css';
5 import foto from './assets/logo192.png';
6
7
8 function App() {
9   return (
10     <div>
11       <h1>Hello, word!</h1>
12       <First/>
13       <img src={foto} alt="foto"></img>
14     </div>
15   );
16 }
17
18 export default App;
```

Essa é o arquivo onde irá funcionar nossa aplicação, onde escrevemos um JSX, no qual o index.js irá renderizar