fonte: https://testing-library.com/docs/react-testing-library/intro/

React Testing Library

Visão geral

Os utilitários fornecidos por esta biblioteca facilitam a consulta ao DOM da mesma forma que o usuário faria. Encontrar elementos de formulário por seu texto de rótulo (como um usuário faria), encontrar links e botões de seu texto (como um usuário faria). Ele também expõe uma maneira recomendada de localizar elementos data-testid como uma "saída de emergência" para elementos em que o conteúdo do texto e o rótulo não fazem sentido ou não são práticos.

Esta biblioteca incentiva seus aplicativos a serem mais acessíveis e permite que você aproxime seus testes de usar seus componentes da maneira que um usuário fará, o que permite que seus testes lhe dêem mais confiança de que seu aplicativo funcionará quando um usuário real o usar.

Esta biblioteca é uma substituição para Enzyme . Embora você possa seguir essas diretrizes usando a própria Enzyme, aplicá-la é mais difícil por causa de todos os utilitários extras que a Enzyme fornece (utilitários que facilitam os detalhes de implementação de teste). Leia mais sobre isso no FAQ .

Api

Debug

OBS: É recomendável usar em seu screen.debug no seu lugar.

Este método é um atalho para console.log(prettyDOM(baseElement)).

```
import React from "react";
import { render } from "@testing-library/react";

const HelloWorld = () => <h1>Hello World</h1>;
const { debug } = render(<HelloWorld />);
debug();
// <div>
//   <h1>Hello World</h1>
// </div>
// you can also pass an element: debug(getByTestId('messages'))
// and you can pass all the same arguments to debug as you can
```

```
// to prettyDOM:
// const maxLengthToPrint = 10000
// debug(getByTestId('messages'), maxLengthToPrint, {highlight: false})
```

Rerender

Provavelmente seria melhor se você testar o componente que está fazendo a atualização de props para garantir que os props estão sendo atualizados corretamente (consulte a seção Princípios de Orientação). Dito isso, se você preferir atualizar os adereços de um componente renderizado em seu teste, esta função pode ser usada para atualizar os adereços do componente renderizado.

```
import { render } from "@testing-library/react";
const { rerender } = render(<NumberDisplay number={1} />);

// re-render the same component with different props
rerender(<NumberDisplay number={2} />);
```

Unmount

Isso fará com que o componente renderizado seja desmontado. Isso é útil para testar o que acontece quando seu componente é removido da página (como testar se você não deixa manipuladores de eventos perambulando causando vazamentos de memória).

Este método é uma abstração bem pequena sobre ReactDOM.unmountComponentAtNode

```
import { render } from "@testing-library/react";
const { container, unmount } = render(<Login />);
unmount();
//your component has been unmounted and now: container.innerHTML === ''
```