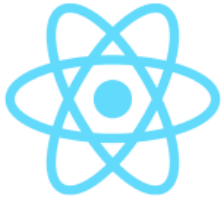
A Beginner's guide to

Unit Testing in React Apps

in antra verma

By the end of this tutorial, you will be able to write your first unit test in react.



As prerequisite, you must have the knowledge about fundamentals of react js.

What is Unit Testing ?

The unit test is the test category with the finest granularity based on the **test pyramid**.

Usually, **it is focused on the functionality of a class, function, or a UI component** and **isolated from the external system** like databases and third-party API.

In simple terms, Unit Testing means **ensuring that each unit** of your application **is working fine**.

Tools for Testing

In order to write unit tests, there are many tools avaliable, but will be using the following combination:

1

JEST

Jest is a test runner, which gives you the ability to run tests with Jest from the command line

2

REACT TESTING LIBRARY

React testing library renders React components like a browser and we can select the elements just as we can using the DOM APIs.

General Structure of a Test

Test Block

render the component

select the elements

interact with the elements

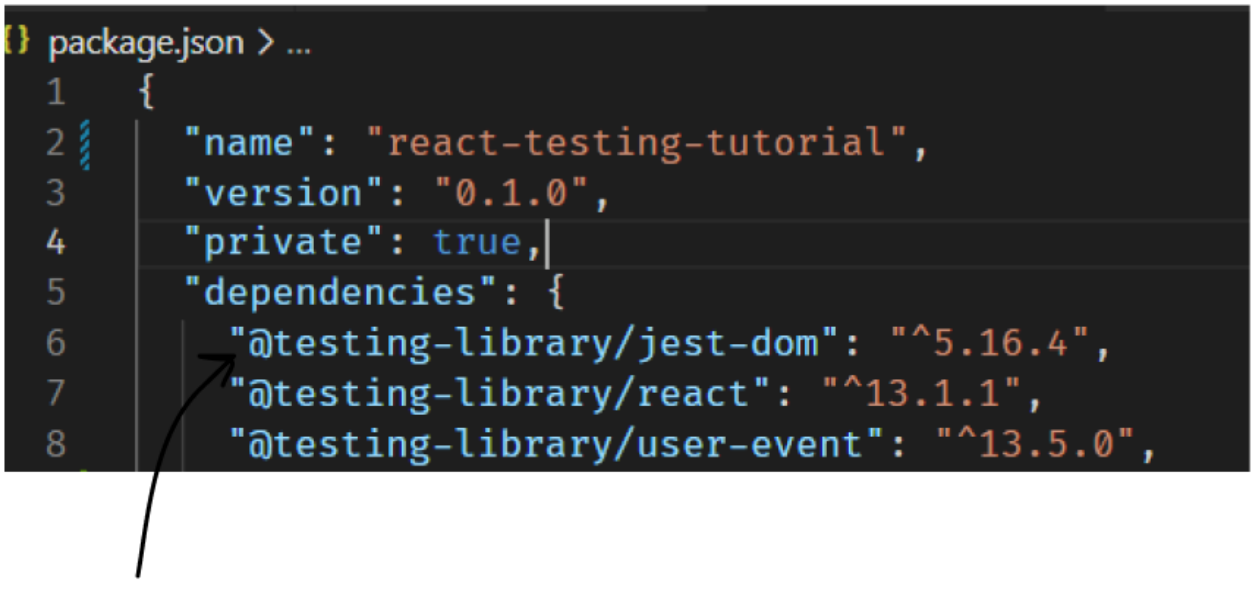
assert the expected results

Create new react app

- create a new react app

```
npx create-react-app react-testing-tutorial
```

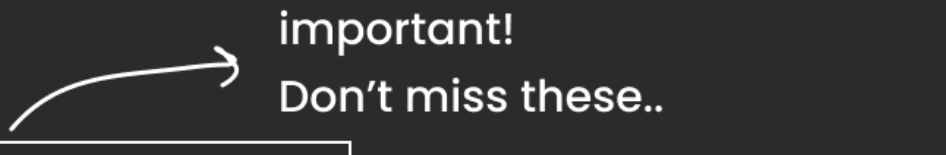
- when you use create-react-app, you get out-of-the box support for jest and react-testing-library, meaning you do not have to add them to your project manually.



Create a component

components/Counter/Counter.js

```jsx
import React, { useState } from "react";

const Counter = () => {
  const [counter, setCounter] = useState(0);

  const incrementCounter = () => {
    setCounter((prevCounter) => prevCounter + 1);
  };

  const decrementCounter = () => {
    setCounter((prevCounter) => prevCounter - 1);
  };

  return (
    <>
      <button data-testid="increment" onClick={incrementCounter}>
        +
      </button>
      <p data-testid="counter">{counter}</p>
      <button data-testid="decrement" onClick={decrementCounter}>
        -
      </button>
    </>
  );
};

export default Counter;
```



Write a Unit Test

components/Counter/Counter.test.js

```
import { render, fireEvent, screen } from "@testing-library/react";
import Counter from "./Counter";

// test block
test("increments counter", () => {

// render the component on virtual dom
render(<Counter />);

// select the elements you want to interact with
const counter = screen.getByTestId("counter");
const incrementBtn = screen.getByTestId("increment");

// interact with those elements
fireEvent.click(incrementBtn);

// assert the expected result
expect(counter).toHaveTextContent("1");
});
```

note: it is important to use **.test.js** as an extension which helps jest to locate the test files in your application

Run the Test

- run the following command on the terminal

```
npm test
```

- You should get the result like this:

```
PASS src/components/Counter/Counter.test.js (16.974 s)
  √ increments counter (175 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        38.224 s
Ran all test suites related to changed files.
```

Was it helpful?