

Lesson 08 Demo 02

Testing the React Component Using Jest

Objective: To demonstrate the implementation of react component testing using Jest

Tools required: Visual Studio Code, Node.js, and npm

Prerequisites: Good understanding of JavaScript, HTML, React, DOM (document object model) and some knowledge of Jest

Steps to be followed:

1. Set up a new React project and install Jest
2. Create a simple React component
3. Write a Jest test
4. Configure the Jest
5. Run the Jest test

Step 1: Set up a new React project and install Jest

- 1.1 Open the terminal and create a react project by using the following command:
npm create-react-app react-jests-example

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  SQL CONSOLE

• labuser@ubuntu2204:~$ npx create-react-app react-jests-example

Creating a new React app in /home/labuser/react-jests-example.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts with cra-template...

added 1463 packages in 44s

242 packages are looking for funding
  run `npm fund` for details

Initialized a git repository.

Installing template dependencies using npm...

added 69 packages, and changed 1 package in 6s

246 packages are looking for funding
  run `npm fund` for details
Removing template package using npm...
```

Success! Created react-jests-example at /home/labuser/react-jests-example
Inside that directory, you can run several commands:

npm start
Starts the development server.

npm run build
Bundles the app into static files for production.

npm test
Starts the test runner.

npm run eject
Removes this tool and copies build dependencies, configuration files
and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

```
cd react-jests-example
npm start
```

Happy hacking!

1.2 Navigate to the folder **react-jests-example** and then install Jest using the following command:

```
cd react-jests-example
```

```
npm install --save-dev jest @testing-library/react @testing-library/jest
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL SQL CONSOLE
labuser@ubuntu2204:~$ cd react-jests-example
labuser@ubuntu2204:~/react-jests-example$ npm install --save-dev jest @testing-library/react @testing-library/jest
```

Step 2: Create a simple React component

2.1 Create a new component file named **Counter.js** in the **src** folder and add the below code in it:

```
import React, { useState } from 'react';
function Counter() {
  const [count, setCount] = useState(0);

  const increment = () => {
    setCount(count + 1);
  };

  const decrement = () => {
    setCount(count - 1);
  };

  return (
    <div>
      <h1>Counter</h1>
      <p>Count: {count}</p>
      <button onClick={increment}>Increment</button>
      <button onClick={decrement}>Decrement</button>
    </div>
  );
}
export default Counter;
```

JS Counter.js U X

src > JS Counter.js > ...

```
1  import React, { useState } from 'react';
2
3  function Counter() {
4    const [count, setCount] = useState(0);
5
6    const increment = () => {
7      setCount(count + 1);
8    };
9
10   const decrement = () => {
11     setCount(count - 1);
12   };
13
14   return (
15     <div>
16       <h1>Counter</h1>
17       <p>Count: {count}</p>
18       <button onClick={increment}>Increment</button>
19       <button onClick={decrement}>Decrement</button>
20     </div>
21   );
22 }
23 export default Counter;
```

Step 3: Write a Jest test

3.1 Now, write Jest tests for the **Counter** component by creating a test file named **Counter.test.js** in the **src** folder:

```
// src/Counter.test.js
import React from 'react';
import { render, fireEvent } from '@testing-library/react';
import Counter from './Counter';

test('renders Counter component with initial count of 0', () => {
  const { getByText } = render(<Counter />);
  const countElement = getByText('Count: 0');
  expect(countElement).toBeInTheDocument();
});

test('increments and decrements count correctly', () => {
  const { getByText } = render(<Counter />);
  const incrementButton = getByText('Increment');
  const decrementButton = getByText('Decrement');
  const countElement = getByText('Count: 0');

  fireEvent.click(incrementButton);
  expect(countElement).toHaveTextContent('Count: 1');

  fireEvent.click(decrementButton);
  expect(countElement).toHaveTextContent('Count: 0');
});
```

```
JS Counter.js U JS Counter.test.js U X
src > JS Counter.test.js > ...
1 // src/Counter.test.js
2 import React from 'react';
3 import { render, fireEvent } from '@testing-library/react';
4 import Counter from './Counter';
5
6 test('renders Counter component with initial count of 0', () => {
7   const { getByText } = render(<Counter />);
8   const countElement = getByText('Count: 0');
9   expect(countElement).toBeInTheDocument();
10 });
11
12 test('increments and decrements count correctly', () => {
13   const { getByText } = render(<Counter />);
14   const incrementButton = getByText('Increment');
15   const decrementButton = getByText('Decrement');
16   const countElement = getByText('Count: 0');
17
18   fireEvent.click(incrementButton);
19   expect(countElement).toHaveTextContent('Count: 1');
20
21   fireEvent.click(decrementButton);
22   expect(countElement).toHaveTextContent('Count: 0');
23 });
```

Step 4: Configure the Jest

- 4.1 Configure Jest by creating a **jest.config.js** file in the root directory of your project and specifying the **testEnvironment** as **jsdom**:

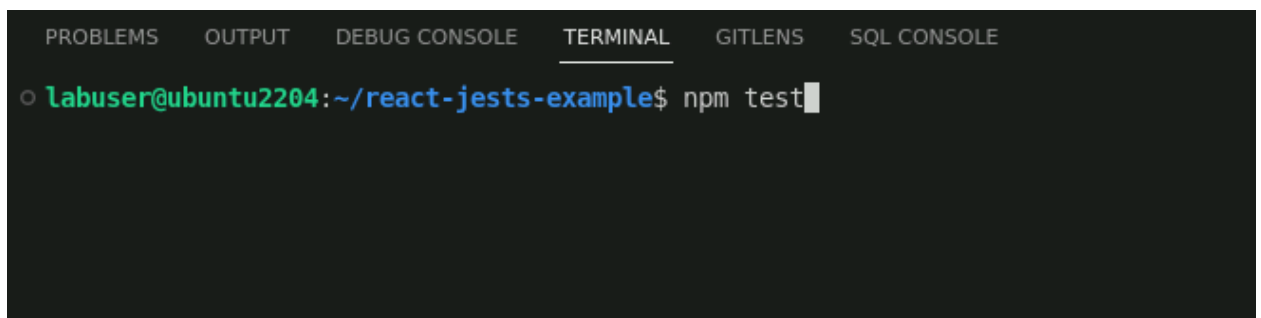
```
// jest.config.js
module.exports = {
  testEnvironment: 'jsdom',
};
```

A screenshot of a code editor with a dark theme. At the top, there are two tabs: 'JS Counter.js' and 'JS jest.config.js'. The 'jest.config.js' tab is active. The code in the editor is as follows:

```
JS jest.config.js > ...
1  // jest.config.js
2  module.exports = {
3    testEnvironment: 'jsdom',
4  };
5
```

Step 5: Run the Jest test

- 5.1 Run the Jest tests using the following command:
npm test

A screenshot of a terminal window. At the top, there are several tabs: 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL' (which is selected), 'GITLENS', and 'SQL CONSOLE'. The terminal shows a prompt for a user named 'labuser' on a machine named 'ubuntu2204' at the directory '~/react-jests-example'. The command 'npm test' has been entered and is followed by a cursor.

```
labuser@ubuntu2204:~/react-jests-example$ npm test
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  GITLENS  SQL CONSOLE

PASS src/Counter.test.js
  ✓ renders Counter component with initial count of 0 (34 ms)
  ✓ increments and decrements count correctly (72 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   0 total
Time:        1.244 s
Ran all test suites related to changed files.

Watch Usage
  > Press a to run all tests.
  > Press f to run only failed tests.
  > Press q to quit watch mode.
  > Press p to filter by a filename regex pattern.
  > Press t to filter by a test name regex pattern.
  > Press Enter to trigger a test run.
```

Jest will discover and execute your tests. If everything is set up correctly, you will see an output indicating that the test passed.

By following these steps, you have successfully set up a basic React project and conducted testing using Jest. This demonstrates how to test a React project within the Jest environment.