ReactJS - Testing

Testing is one of the processes to make sure that the functionality created in any application is working in accordance with the business logic and coding specification. React recommends *React testing library* to test React components and *jest* test runner to run the test. The *react-testing-library* allows the components to be checked in isolation.

It can be installed in the application using below command −

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

Create React app

Create React app configures *React testing library* and *jest* test runner by default. So, testing a React application created using *Create React App* is just a command away.

```
cd /go/to/react/application
npm test
```

The *npm* test command is similar to *npm build* command. Both re-compiles as and when the developer changes the code. Once the command is executed in the command prompt, it emits below questions.

```
No tests found related to files changed since last commit.
Press `a` to run all tests, or run Jest with `--watchAll`.

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.
```

Pressing a will try to run all the test script and finally summaries the result as shown below −

```
Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        4.312 s, estimated 12 s
Ran all test suites.

Watch Usage: Press w to show more.
```

Testing in a custom application

Let us write a custom React application using *Rollup bundler* and test it using *React testing library* and *jest* test runner in this chapter.

First, create a new react application, *react-test-app* using *Rollup* bundler by following instruction in Creating a *React application* chapter.

Next, install the testing library.

```
cd /go/to/react-test-app
npm install --save @testing-library/react @testing-library/jest-dom
```

Next, open the application in your favorite editor.

Next, create a file, *HelloWorld.test.js* under *src/components* folder to write test for *HelloWorld* component and start editing.

Next, import react library.

```
import React from 'react';
```

Next, import the testing library.

```
import { render, screen } from '@testing-library/react'; import '@testing-library/jest-dom';
```

Next, import our *HelloWorld* component.

```
import HelloWorld from './HelloWorld';
```

Next, write a test to check the existence of *Hello World text* in the document.

```
test('test scenario 1', () => {
   render(<HelloWorld />);
   const element = screen.getByText(/Hello World/i);
   expect(element).toBeInTheDocument();
});
```

The complete source code of the test code is given below −

```
import React from 'react';
import { render, screen } from '@testing-library/react';
import '@testing-library/jest-dom';
import HelloWorld from './HelloWorld';

test('test scenario 1', () => {
   render(<HelloWorld />);
   const element = screen.getByText(/Hello World/i);
   expect(element).toBeInTheDocument();
});
```

Next, install jest test runner, if it is not installed already in the system.

```
npm install jest -g
```

Next, run jest command in the root folder of the application.

```
jest
```

Next, run jest command in the root folder of the application.

```
PASS  src/components/HelloWorld.test.js
   √ test scenario 1 (29 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        5.148 s
Ran all test suites.
```

Revision #1
Created 14 December 2022 10:48:36 by Admin
Updated 14 December 2022 10:49:13 by Admin