Published in Geek Culture

Rodrigo Figueroa Follow

Nov 6, 2021 · 4 min read · Listen

Save

Testing Express.js with Jest

Hi again, I was reading about QA for Node.js, and they talked about Jest who is a really good option for Node.js and other libraries like React.js, actually I was working on a small Node.js project, and we will test him with Jest!

Important! Since Express-handlebars update its library, you need to change the import

```
import { engine } from 'express-handlebars';
```

And change the handleExp for this

```
app.engine( 'handlebars', engine({
defaultLayout: 'main'
}))
```

You can watch the video too

Complete post

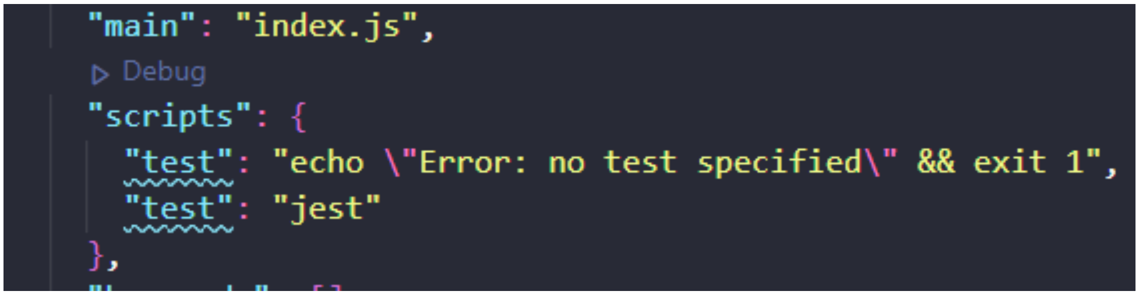
Installing Jest

```
npm i --save-dev jest
```

Add command to the package JSON

To run Jest testings, add this script to your package JSON

```
"test": "jest"
```



```
"main": "index.js",
▷ Debug
"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1",
  "test": "jest"
},
```

Example script to the package JSON

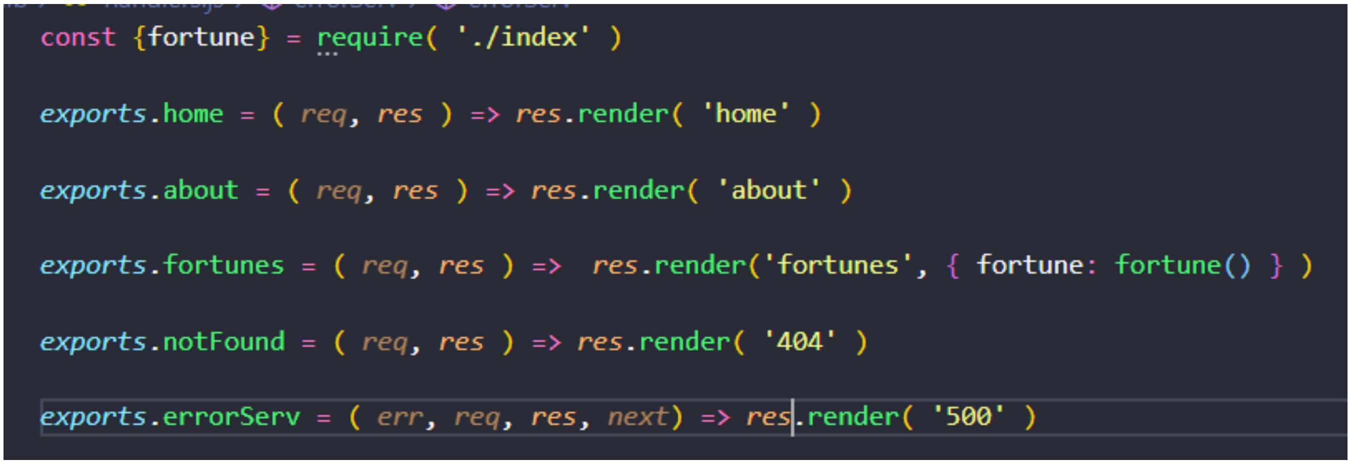
Structure

It is really recommended creating a directory called __test__ and inside add your test files for each part you want to test the name of the files are nameyouwant.test.js in this case you can save time searching for your test files.

Example test directory and test file

Separate functions

We need to separate every function that we use to render the page, first we need to create a file to encapsulate the functions, and we need the exports



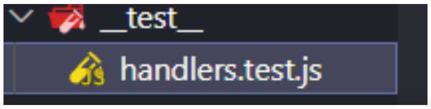
```
const {fortune} = require( './index' )

exports.home = ( req, res ) => res.render( 'home' )

exports.about = ( req, res ) => res.render( 'about' )

exports.fortunes = ( req, res ) =>  res.render('fortunes', { fortune: fortune() } )

exports.notFound = ( req, res ) => res.render( '404' )

exports.errorServ = ( err, req, res, next) => res.render( '500' )
```

Example encapsulate function and using exports

```
const {fortune} = require( './index' )
exports.home = ( req, res ) => res.render( 'home' )
exports.about = ( req, res ) => res.render( 'about' )
exports.fortunes = ( req, res ) =>  res.render('fortunes', {
fortune: fortune() } )exports.notFound = ( req, res ) => res.render(
'404' )
exports.errorServ = ( err, req, res, next) => res.render( '500' )
```

Create our test file and start testing

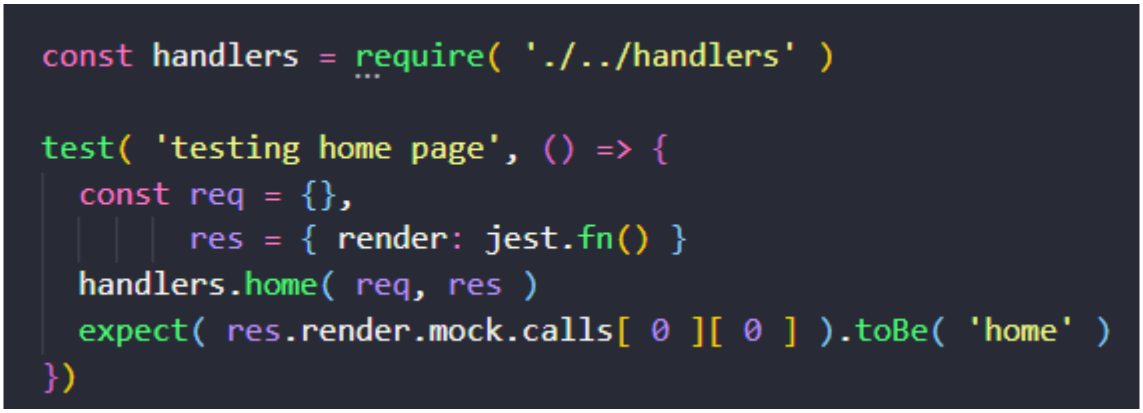
Second we need to create our handlers.test.js file



__test__

handlers.test.js

To test our function we need to start with the test function who needs two parameters the first one is the name of the test and the second one is a function where you can expect what you need for example we need to test render and it has to be the home.



```
const handlers = require( './../handlers' )

test( 'testing home page', () => {
  const req = {},
        res = { render: jest.fn() }
  handlers.home( req, res )
  expect( res.render.mock.calls[ 0 ][ 0 ] ).toBe( 'home' )
})
```

Example testing home page

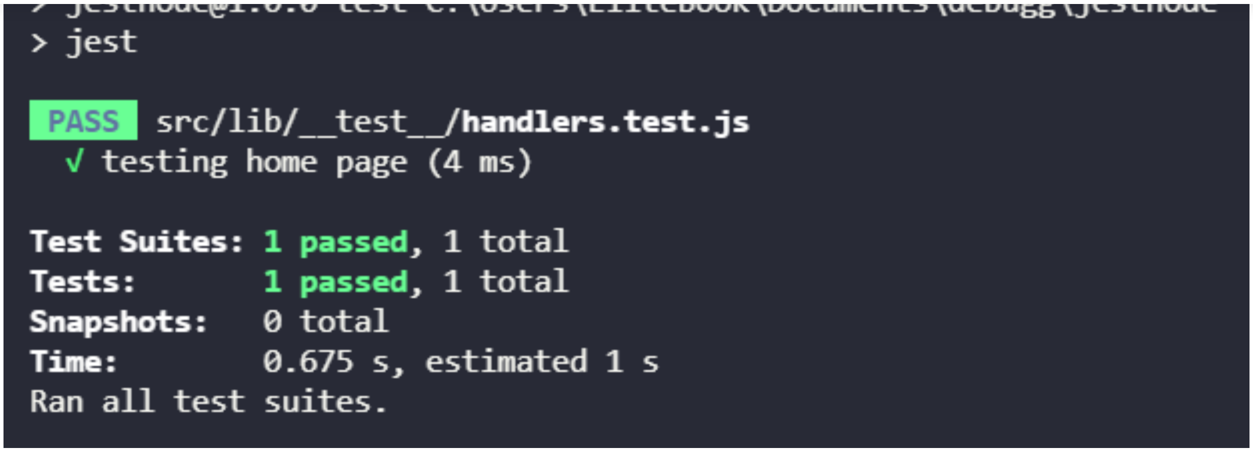
```
const handlers = require( './../handlers' )
test( 'testing home page', () => {
const req = {},
res = { render: jest.fn() }
handlers.home( req, res )
expect( res.render.mock.calls[ 0 ][ 0 ] ).toBe( 'home' )
})
```

If you see we have the req and the res variables but the res variable had render and a jest function fn to mock and after that we expect the render be home.

Run Jest

to run command use this command

```
npm run jest
```

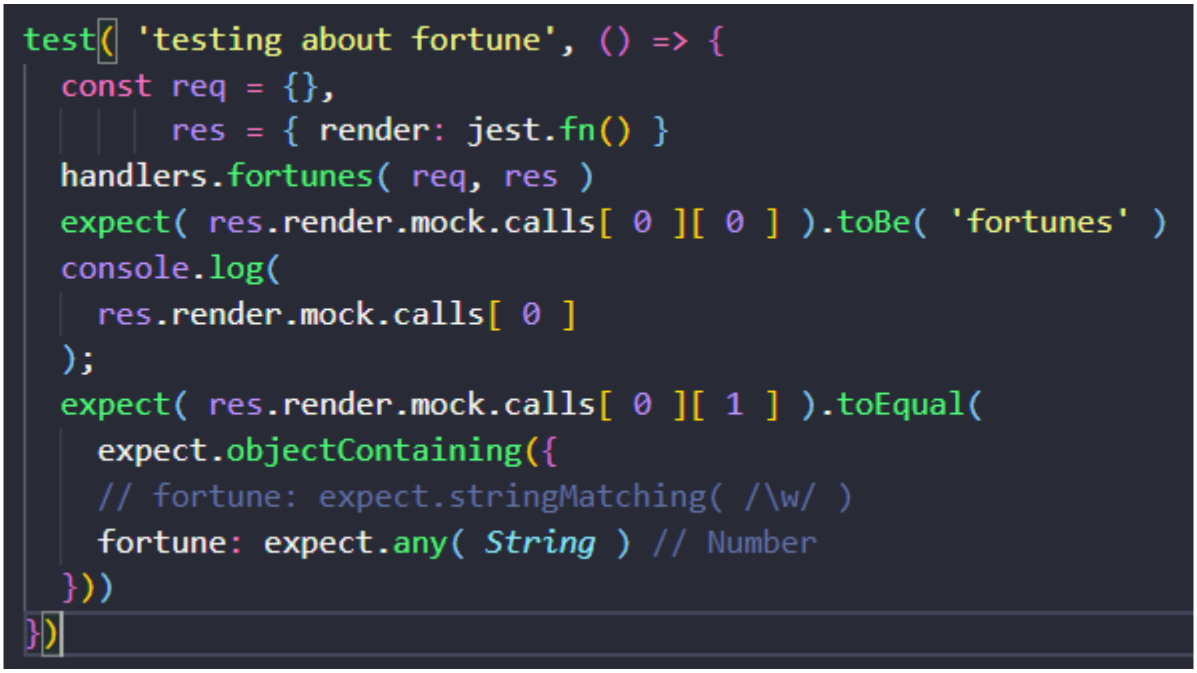


```
> jestnode@1.0.0 test C:\Users\Elitebook\Documents\debugg\jestnode
> jest

 PASS  src/lib/__test__/handlers.test.js
  √ testing home page (4 ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        0.675 s, estimated 1 s
Ran all test suites.
```

Example run test complete

If everything is correct we can see that the “testing home page” passed

And we can do it with the others because they are the same we can use the same example from above, but if we need to test something else we can use the fortunes page when we need to test the object



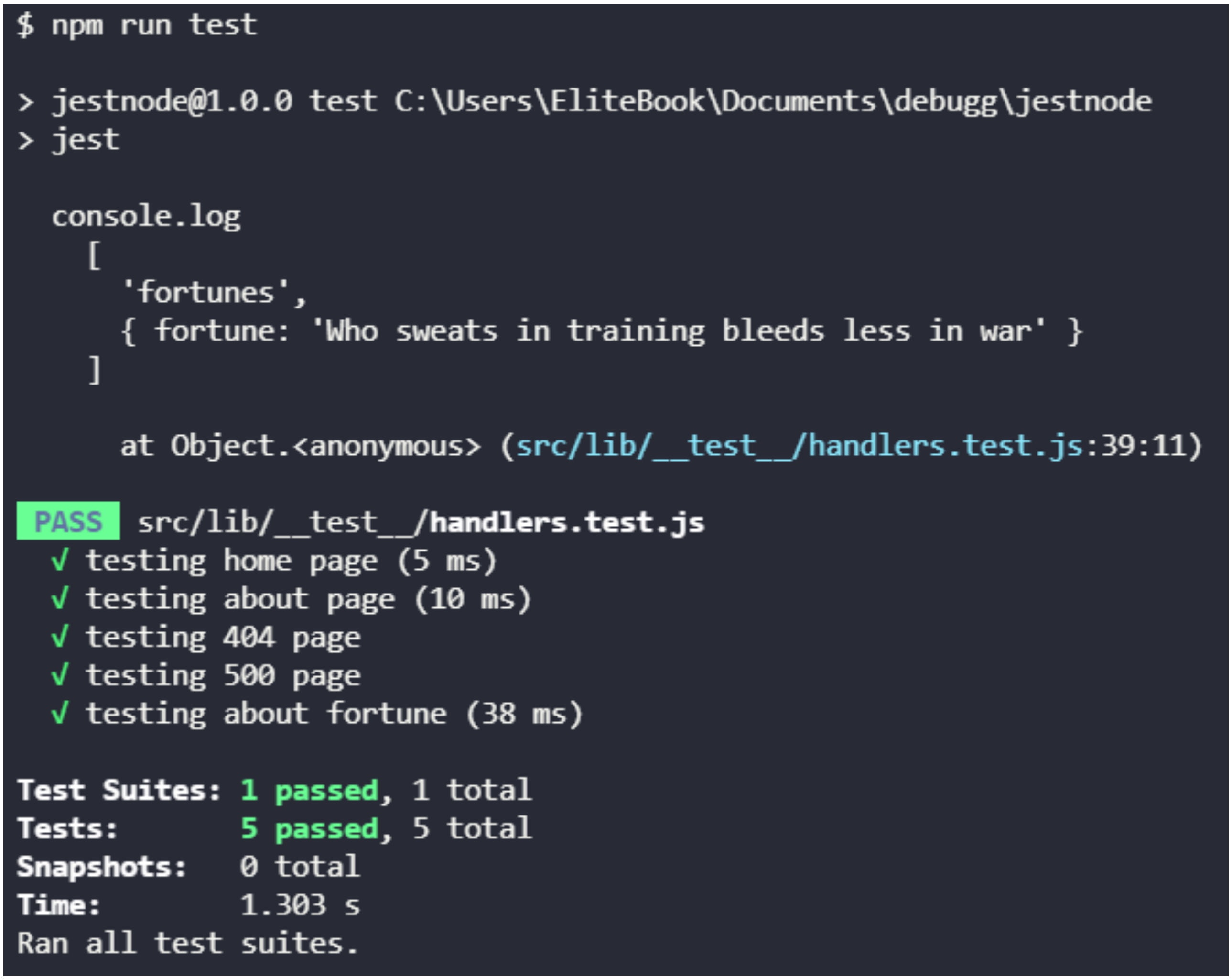
```
test( 'testing about fortune', () => {
  const req = {},
        res = { render: jest.fn() }
  handlers.fortunes( req, res )
  expect( res.render.mock.calls[ 0 ][ 0 ] ).toBe( 'fortunes' )
  console.log(
    res.render.mock.calls[ 0 ]
  );
  expect( res.render.mock.calls[ 0 ][ 1 ] ).toEqual(
    expect.objectContaining({
    // fortune: expect.stringMatching( /\w/ )
    fortune: expect.any( String ) // Number
  }))
})
```

Example testing fortune page and object

For this it is almost the same unless we need to check the object for the dynamic string

Finish

Run the command again and we can see that everything is correct!



```
$ npm run test

> jestnode@1.0.0 test C:\Users\EliteBook\Documents\debugg\jestnode
> jest

  console.log
    [
      'fortunes',
      { fortune: 'Who sweats in training bleeds less in war' }
    ]

      at Object.<anonymous> (src/lib/__test__/handlers.test.js:39:11)

 PASS  src/lib/__test__/handlers.test.js
  √ testing home page (5 ms)
  √ testing about page (10 ms)
  √ testing 404 page
  √ testing 500 page
  √ testing about fortune (38 ms)

Test Suites: 1 passed, 1 total
Tests:       5 passed, 5 total
Snapshots:   0 total
Time:        1.303 s
Ran all test suites.
```

Example all test in Jest

Conclusion

To summarize, Jest is a fantastic tool to test our projects like Express.js, the documentation is really great, and I really found what I needed for objects string, numbers etc. To keep making your projects high levels of standard, you could use Jest.

Jest is a JavaScript testing framework designed to ensure correctness of any JavaScript codebase. It allows you to...

jestjs.io

Expect · Jest

When you're writing tests, you often need to check that values meet certain conditions. expect gives you access to a...

jestjs.io

Web Development with Node and Express: Leveraging the JavaScript Stack

Web Development with Node and Express: Leveraging the JavaScript Stack [Brown, Ethan] on Amazon.com. *FREE* shipping on...

www.amazon.com