Lesson Monday

Intermediate JavaScript (/intermediate-javascript) / Test-Driven Development and Environments with JavaScript (/intermediate-javascript/test-driven-development-and-environments-with-javascript) / Expanding our Testing Tools: Adding Setup and Teardown

Text

In the last lesson, we wrote six tests for a triangle tracker application. In each test, we needed to instantiate a `Triangle` object to test either the constructor or the `Triangle.prototype.checkType()` method. These `Triangle` objects were all different, but what would happen if we needed to instantiate the same `Triangle` object for many tests?

This is a very common use case because we might want to instantiate the same object in different tests so you can test it in different ways. Fortunately, testing frameworks like Jest account for this by having additional methods for **setup** and **teardown** between tests. Setup is commonly used for instantiating variables and objects. Teardown is used for reinitializing objects or any variables that don't automatically fall out of scope between tests.

We can use Jest's `beforeEach()` function for setup and `afterEach()` function for teardown.

Using `beforeEach()` and `afterEach()`

Let's imagine that we have a `Triangle` object that we want to reuse for multiple tests.

If this is the case, we'd use `beforeEach()` to DRY up the code. Here's an example:

__tests__/triangle.test.js

```
...
describe('show how beforeEach() works', () => {
  let reusableTriangle;

  beforeEach(() => {
    reusableTriangle = new Triangle(5, 5, 5);
  });

  test('should use the resuableTriangle and update the valu
e of one side', () => {
    reusableTriangle.side1 = 6;
    console.log(reusableTriangle);
  });

  test('should show how this reusableTriangle is reset to s
ides of 5', () => {
    console.log(reusableTriangle);
  });
});
```

Here we declare `reusableTriangle` in the top-level scope of the `describe()` function, and because of this, we can call on `reusableTriangle` from within any test.

The `beforeEach()` function assigns an instantiated `Triangle` object to the `reusableTriangle` variable. The `beforeEach()` function is run before each test is run, so a new `Triangle` object will be created and set as the value of `reusableTriangle` before each test!

A nice benefit of using the `beforeEach()` function to define our example `Triangle` object is that we now don't need to do this in each test, which makes our code more DRY.

If we run our tests, we'll see that the value of the first `console.log()` is printed to the terminal: `Triangle { side1: 6, side2: 5, side3: 5 }`. It may come as a bit of a surprise that `console.log()` prints to the terminal, but that is built-in functionality that Node provides. The benefits of `console.log()` aren't limited to the browser!

Then in the second `consolg.log()` message, we'll see that the triangle's sides have all been reset to 5: `Triangle { side1: 5, side2: 5, side3: 5 }`

Take note that the example above will not be helpful for the Shape Tracker project, since we need to instantiate different triangles for each test to determine the triangle's type.

There's also an `afterEach()` function as well. The code inside of the `afterEach()` function is run after each test has been run.

```
afterEach(() => {
  // this code is run after each test
});
```

This function is particularly useful if you need to perform some kind of teardown after each test. For instance, if data from one test somehow ends up in another test, it could result in a failed test or other unexpected behavior.

Check the documentation on setup and teardown (https://jestjs.io/docs/en/setup-teardown) for more information.

Previous (/intermediate-javascript/test-driven-development-and-environments-with-javascript/testing-best-practices)
Next (/intermediate-javascript/test-driven-development-and-environments-with-javascript/improving-test-reports-adding-test-coverage-information)

Lesson 31 of 49
Last updated more than 3 months ago.

disable dark mode

(http://www.epicodus.com)

© 2023 Epicodus (http://www.epicodus.com/), Inc.