Testing JavaScript

Assignment 4

Let's test `NoteUtilities.js` and `Storage.js`

I added two outlines (base structures) for testing `NoteUtilities.js` and `Storage.js`.

- [ ] Get the empty test suites by running `git pull` in the folder containing the code

You will now find them in `src/utils/__test__/NoteUtilities.test.js` and `src/utils/__test__/Storage.test.js`.

- [ ] You'll see new `describe` blocks in the code. Can you guess what they are for?
- [ ] Fill in the missing testcases in `NoteUtilities.test.js`
- [ ] Fill in the missing testcases in `Storage.test.js`

Assignment 5

Testing Errors

- [ ] Add a test file for `TestMe.js` which you should already have pulled from git in the previous task. Name it `TestMe.test.js`.
- [ ] Have a look the the `fail` function in `TestMe.js`. What does it do?
- [ ] Write a test that calls the `fail` function with `true` to make it fail. Run the tests. What happens?

- [ ] Can you change the test to succeed if the function fails?
  Hint: There is a function `expect(...).toThrow()` which can check if an error occurs. See also the jest docs for expect.
- [ ] Does you test check if the error has the message `'Failure!'`? How can you achieve this?
  Hint: there is something about that in the docs

Async Tests

To write asynchronous tests pass an async function to `test()` or `it()`. See below for an example:

```
test('my first async test', async () => {
    const result = await somethingThatNeedsTimeAndReturnsAPromise();
    expect(result).toBe(true);
});
```

- [ ] Test the `wait` funtion from `TestMe.js` to make sure it resolves to `'done'` when being called with a time of `1000`.
  - [ ] How long does this test run?
    Hint: You can run just this test using: `npm run test -- --runTestsByPath ./src/utils/__test__/TestMe.test.js`
  - [ ] What happens if you call the `wait` function with a time of `10000`?
- [ ] Have a look at the jest documentation about timer mocks. Can you find a way to use fake timers to not have to actually wait in the test?
- [ ] *Bonus*: Can you use `jest.advanceTimersByTime()` to make sure the `wait` function waits for exactly the time it is given?

Test Coverage

- [ ] Run the command `npm run test:coverage` and have a look at the results that are printed. What are they for?
- [ ] Skip a test and rerun the command. Can you notice differences to the previous result?

Hint: You can skip a test by changing the `test` or `it` to `test.skip` or `it.skip`.

- [ ] Can you reach 100% coverage of the `TestMe.js` file?