## **TESTING BASICS**

### **Introduction**

Test Driven Development is a *big* deal in the modern development landscape. This is a concept that we introduced way back in our Fundamentals section with our JavaScript Exercises. The main idea is simply that you start working on your code by writing automated tests *before* writing the code that is being tested. There are tons of benefits to working like this, all of which will be discussed in the resources below.

There are many test-running systems available in JavaScript: [Mocha](https://mochajs.org/), [Jasmine](https://jasmine.github.io/), [Tape](https://github.com/substack/tape) and [Jest](https://facebook.github.io/jest/) to name a few. Fortunately the syntax for each one is very similar. They all have their own set of special features, but the basic syntax is almost identical, so in the end it doesn’t matter which one you use. In fact, simply picking which library to use for this curriculum has been quite tricky!

This lesson is going to center around Jest. The biggest reasons for this decision are that one of the best resources we’ve found for explaining JavaScript testing uses it and they have fantastic documentation. In the end, writing tests is less about the syntax and more about the TDD philosophy. The most important issues are knowing why we write tests and what we test rather than how.

### **Assignment**

1. [Read this short article](http://godswillokwara.com/index.php/2016/09/09/the-importance-of-test-driven-development/) that outlines the basic process and the benefits of TDD.
2. Watch at least the first 3 videos of [this video series](https://www.youtube.com/watch?v=Eu35xM76kKY) about testing in JavaScript. The first video focuses heavily on the WHY, while the next two go into more depth about the process. Later videos in the series are *definitely* worthwhile, but the first 3 are enough to get you up and running.
3. Read and follow the [Getting Started](https://facebook.github.io/jest/docs/en/getting-started.html#content) tutorial on the main Jest website.
4. Read and follow the [Using Matchers](https://facebook.github.io/jest/docs/en/using-matchers.html#content) document on the main Jest website. This one demonstrates some of the other useful functions you can use in your tests.
5. Watch [this amazing video](https://www.youtube.com/watch?v=URSWYvyc42M) that covers *what* to test in your codebase. The video is specifically about testing the Ruby language, but that doesn’t matter *at all*. The concepts here ring true in any language, and luckily Ruby is a clear enough language that you will be able to follow along just fine. This video might be worth re-visiting after you’ve done some testing of your own.