Testing

React, Jest, Enzyme

Types of Tests

- **Unit**: testing done to individual components/ functions/classes. Mock input and ensure output is as expected

- **Integration**: testing done to multiple components/modules to ensure they work together as expected

- **Functional (end to end)**: testing of the complete functionality of an application

Test Driven Development

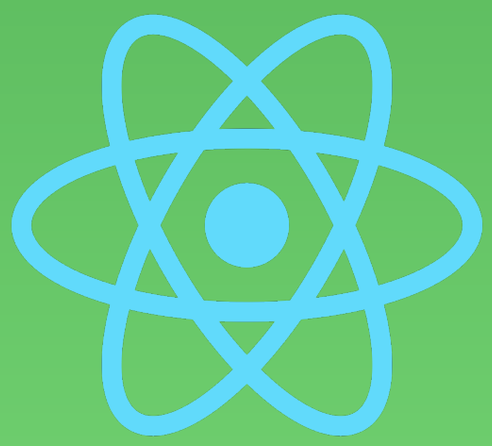
TDD is a programming workflow that involves writing tests before writing actual code

- Write a test
- Run tests
- Make the test past
- Run tests
- Refactor

TDD Benefits

- Produces verifiably working code
- Forces the developer to be concise when coding a solution
- Forces developer to consider architecture and API design before implementation
- Provides feature documentation for other developers

Our Testing Stack



Jest

- Jest is a no-configuration unit testing tool created by Facebook. Built on top of Jasmine, it was created to make testing quick and reliable.

- **API**: http://facebook.github.io/jest/docs/en/expect.html

Enzyme

- Enzyme is a Javascript testing utility for React written by AirBnB. It was created to make it easy to assert, manipulate, and traverse React components and their output

- **API**: **https://github.com/blainekasten/enzyme-matchers**

Testing Basics

- (**13**) Describe the test
- (**14**) Hooks into component
- (**16**) Mock data or simulate some event
- (**18**) Assert

```
12
13   it('renders welcome', () => {
14     const wrapper = shallow(<App />);
15
16     const welcome = <h2>Welcome to the TDD Todo!</h2>;
17
18     expect(wrapper).toContainReact(welcome);
19   });
20
```

Types of Enzyme Tests

- **Shallow render**: Renders a component without rendering its children
  - https://github.com/airbnb/enzyme/blob/master/docs/api/shallow.md
- **Full DOM render**: Renders a component in a DOM environment when you need to test lifecycle methods or interact with the DOM API
  - https://github.com/airbnb/enzyme/blob/master/docs/api/mount.md
- **Static render**: Renders a component to static HTML

What to Test

- Do components/children render?
- Does state update as expected?
- Do methods run when the UI is interacted with? (intended and not intended user actions)
- Do components receive the correct props?
- Do methods generate the expected output?

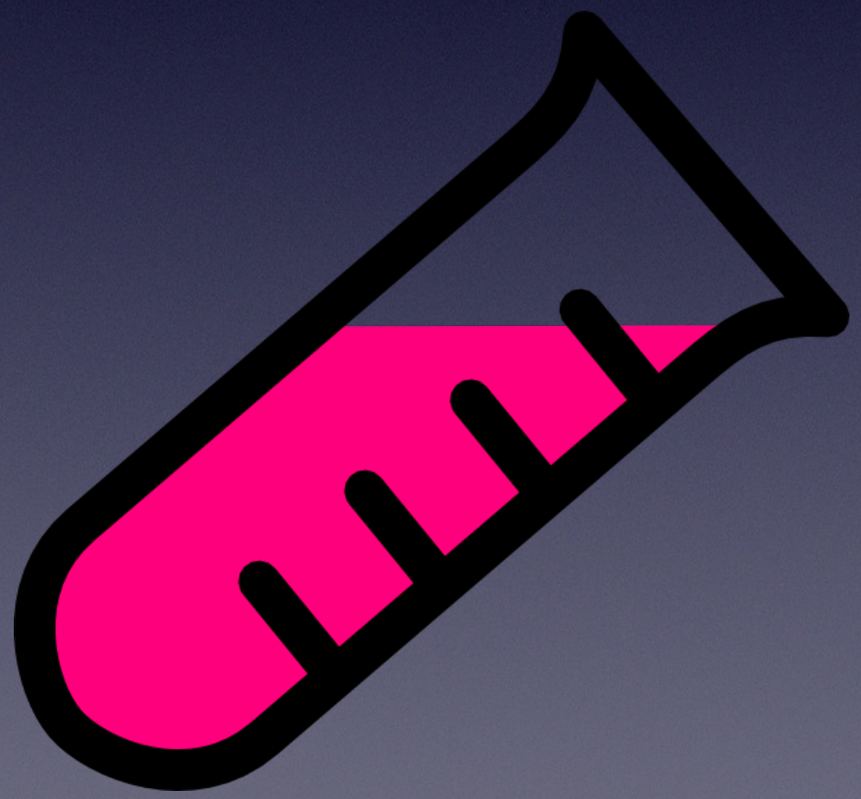
Considerations While Writing Tests

- What component aspect are you testing?
- What should the feature do?
- What is the actual output?
- What is the expected output?

Tests Should...

- Be isolated
- Be specific
- Be as simple as possible
- Describe what the component does, not how it does it

Testing Overview



Testing is like lifting with your legs…

We all know we **SHOULD** do it

The General Idea

- Write code that tests code
- Determine what **expected** results should be
- Run code
- Check **actual** result against **expected** result

Why Test Your Code?

- Saves time (eventually)
- Prevents regressions
- Better code
- Executable documentation

Many Types of Tests

http://www.softwaretestinghelp.com/types-of-software-testing/

Many Testing Frameworks

http://www.designyourway.net/drb/the-full-list-of-testing-frameworks-for-javascript-47-items/

We will focus on Facebook's Jest and Airbnb's Enzyme

Jest overview

https://facebook.github.io/jest/#use

Simple Jest example