**Monday April 29th, 2019**

8:25 — Today I plan on finished Sass and getting started with React. Let’s get to it!

8:26 — Right out of the gate we’re learning about Sass’s while directive (which is basically the equivalent of a JavaScript for loop. For example, we can set our variable x to 1, increment it by 1, and as long as x remains less than 13 we can create an additional column (much like BootStrap’s grid system…).

8:36 — FreeCodeCamp just introduced me to partials, which are separate files in that hold CSS code and are imported into Sass files as a convenient way of grouping similar modules of code.

8:40 — Sass has a feature called extend that reminds me a bit of inheritance in JavaScript. Basically, by using @extend, we can take all off the Sass rules we’ve applied to one element and apply them to another. This has the potential to eliminate unnecessary copying/pasting and greatly reduce the length of our code.

8:49 — I’ve now finished learning Sass… for now. I’ll implement it into my future front-end library projects when I get to them of course. For the time being, however, I’m going to focus on learning React and Redux. I’ve never built a project with React or Redux.

I know virtually nothing about either except for the fact that React was made popular by Facebook, and that it is an open-source JavaScript library that once had a lot of drama surrounding it regarding legal issues (I listened to a podcast once about Facebook putting some things into the React user license that many companies looked down upon).

I think the issue was later fixed, but anyway those are random details that aren’t important for now. Let’s get to learning React!

8:53 — Apparently React combines HTML with JavaScript to created its own markup language JSX… Ok…

8:55 — Because JSX is not valid JavaScript, JSX code must be compiled into JavaScript. So that’s why I saw Babel on codepen…

8:59 — Nested JSX can only return a single element. For example, we cannot transpile three <p> tags written by themselves. We’d have to nest all of the <p> tags within a div, thus only needing to return a single element. Why is this? I don’t know… yet.

9:04 — To create comments inside JSX, we use the following syntax *{/\* Insert your comment here\*/}*.

9:05 — This was a solid pomodoro session to start the day. Now I’m going to go shower, brush my teeth, and take a quick walk outside to get my circadian rhythm in order.

13:31 — I’ve been exploring a new part of Saigon, and getting some exercise. Now it’s time to get another pomodoro session in!

13:40 — 10 minutes in, I’m feeling a bit confused about how to use ReactDOM.render().

13:43 — I now have a rough understand of the ReactDOM.render(). Basically, DOM stands for Document Object Model, something I knew already. However, it seems that writing directly to the DOM is slow and thus using JavaScript to update the DOM and only directly writing to the DOM when absolutely necessary offers better performance? Ehhh that’s my best guess for now at least.

13:46 — It seems there are some differences between JSX and HTML. One obvious difference is that HTML uses the keyword *class* to define classes while JSX uses the keyword className. Another difference is that any JSX element can be written with a self-closing tag.

HTML for example requires us to write <div> </div> while JSX could simply be written <div />. However, in this example, a div written in the second format could not contain anything? What’s the point in being able to write it in this way then? The answer has something to do with being useful in rendering React components. More information coming later…

13:54 — I keep hearing that everything in React is a component, but I have absolutely no idea what that means. Time to do some investigating to find out…

14:00 — The video of the guy I’m watching says that a component is just a part of the website. By using components we can somehow create custom HTML tags or something like that haha. I’ll do more investigating in the next pomodoro session.

22:16 — I’m back for a late night coding session. Let’s get to it!

22:27 — To be honest, I still don’t fully understand how to use states in React. What I do understand thus far, however, is that we use properties for parts of our component that will not change and state for parts of our component that may change.

22:39 — I’ve watched a couple videos about state now as a supplementary to the FreeCodeCamp exercises. To be honest, I’m feeling pretty lost beyond lost right now. I do know, however, by this point in my programming journey that this feeling is normal. I’m confident it will fade if I continue putting in focused hours learning to code. As MJD from FunFunFunction always says, “It’s ok if you’re feeling confused. It means you’re learning!”

22:51 — FreeCodeCamp’s difficulty is feeling so steep for me that I’m wondering if doing FCC is even teaching me much at this point. I’m going to take a break from FCC to look at some more basic tutorials to solidify my understanding of the fundamentals before advancing any further in FCC.

22:52 — I think I’ll spend some time following this course: <https://scrimba.com/g/glearnreact>

It seems to be well produced.

22:58 — What are some of the benefits of learning React?

#1 It’s very fast.

#2 Many intelligent developers and the open-source community maintain React ensuring that it will continue to be relevant for the foreseeable future.

#3 React is one of the most in-demand front-end libraries that makes it easy to get hired.

23:08 — So now I’m fairly confident in my ability to use ReactDOM.render() in basic use cases. I’m finding Scrimba to be an excellent platform for learning code. I’m going to learn more tomorrow. For now, however, I’d like to get to bed earlier and try to get my sleep schedule back on track.

Total time spent coding today: 2 hours 17 minutes

Total time spent coding thus far in April 2019: 42 hours 44 minutes