**Sunday May 5th, 2019**

16:44 — I’m getting quite a late start to the day aren’t I? ;) Fortunately, I imagine things will only be like this for another few weeks. Once summer rolls around, I plan to cut everything else out in order to progress my programming skills as quickly as possible.

16:51 — Oh lord almight lol. I thought after spending several days on the React course outside FreeCodeCamp that things would be easier upon my return. Things are easy enough for me to complete the exercises, but I’m nowhere close to being able to explain what’s going on in the code.

16:52 — What is this doing?

class ParentComponent extends React.Component {

constructor(props) {

super(props);

}

16:53 — Honestly don’t know. The word props and ParentComponent are literally the only two things I understand in the above code. To Google we go…

16:56 — I’ve stumbled upon the blog of a chap named Cheng. According to him the above code is somehow related to ES6 class syntax.

16:57 — Apparently, whenever we have a constructor we need to call super( ). Again to reiterate, we don’t always need to call super with all of our React components, but it is necessary to do so whenever we have a constructor in our code.

17:00 — If you’d like to read more on super( ), this is Cheng’s blog entry on this topic: <http://cheng.logdown.com/posts/2016/03/26/683329>

17:04 — I’m reading an article on the Reactjs documentation website. According to the website once we have a function, we can convert it into a class. We do so by going through the following process (I’m copying these five steps straight from the documentation):

#1 Create an ES6 class, with the same name, that extends React.Component.

#2 Add a single empty method to it called render( ).

#3 Move the body of the function into the render( ) method.

#4 Replace props with this.props in the render( ) body.

#5 Delete the remaining empty function declaration.

(Source: <https://reactjs.org/docs/state-and-lifecycle.html>)

Upon completion of these five steps our function will no longer be defined as a function but rather as a class. This allows us to use features such as lifecycle methods and local state.

17:09 — What are lifecycle methods and local state? When do the questions end? Hahahaa. In all seriousness, I feel I have no reference for any of these concepts. I’m not going to give up on React, but I am also introspective enough that I may get discouraged or frustrated if I continue on this path asking questions that are seemingly well above my current competency level. For that reason… time to go back to the *Learn React for Free* course.

17:16 — Now I’m working on using the .map( ) function and props to import the items on our todo list from another file (todosData.js) rather than hard coding everything straight into App.js.

17:20 — Unfortunately, my code is currently producing the following error:

*Error: SyntaxError: unknown: Namespace tags are not supported. ReactJSX is not XML. (/App.js:1)*

17:23 — I thought the problem was related to using a variable name of “text” that was already reserved by the XML language. Changing my variable name to “texty” didn’t help though. I’ll have to do more investigating later. I’m meeting a friend in literally 5 minutes across the city for a park workout. Haha but being late to become a better developer and have extra time coding is worth it! ;)

19:52 — I’m back. My friend I was supposed to meet for an exercise session at the park felt “unmotivated” and thus we didn’t work out lol. Say whaaaat? We ended up having a “bro chat” after having not met for a month or so. Now let’s get back to the most important thing in my life — becoming a better developer.

19:56 — I just searched and found [teamtreehouse.com](http://teamtreehouse.com) has a Learn React course. I intend to continue learning with the course I’ve been doing on Scrimba, but I may also pay for a new membership on teamtreehouse if I get **really** stuck on my current course, or to solidify the fundamentals after I finish it.

20:02 — I keep getting errors when trying to include JavaScript in my JSX. I’m going to go review material in this area to see if I can figure out where I’m going wrong.

20:12 — Props in React are very similar to parameters in JavaScript.

20:17 — I can console.log(props) fine, but actually rendering it to the page in my App.js folder is a whole ‘nother beast lol.

20:19 — One simple mistake I was making was writing:

<p>{TodoItem.text}</p>

A silly mistake in retrospect. Now I’ve changed it to be:

<p>{props.text}</p>

Things are now working as one would expect. The only problem is that I cannot successfully display false without getting an error.

20:26 — By writing my code like this:

<TodoItem text = "hello" completed = “false"/>

I no longer have any areas. But… I am not sure if it is correct to use quotation marks with boolean values in React. I would assume not, but everything else continually resulted in errors.

\*sigh\*

20:32 — I’m also having some problem with my logic in another component. This in-line styling is completely useless it seems:

function TodoItem(props) {

if (props.completed == "false") {

<p style = {{color: "blue"}}> Hellgfgdgo </p>

}

Normally I don’t worry about details when implementing a new features. I don’t care if something is visually appealing, at least initially. I just like to get something on the page and refactor code to improve things from there. Today, however, I can’t even get any color changes to occur.

20:34 — I’ll try to get in another coding session later tonight, but no guarantee as I have to do some content marketing client work.

**Total time spent coding today:**

**Total time spent coding thus far in May 2019:**