**Thursday May 9th, 2019**

8:45 — Yo yo let’s get to it!

8:54 — I just spent the last several minutes reviewing event handlers in React/JavaScript. I find it interesting that when using an even handler that we don’t call the function with parentheses ( ). I suppose that the parentheses are not utilized because including parentheses would cause the function to be immediately invoked as the page is loaded.

9:00 — It seems like everything I’ve learned is starting to come together. I’ve just implemented an event handler to listen for an onClick event on my input element. Once the click occurs, we log “subject to change” to the console.

9:01 — The next obvious step would be to use the event handler to listen for a click on my input element. Once the click is detected I could then change the state of the element. This would allow for a visual representation of whether the toDoItem was completed or not yet completed!

9:03 — Now I’m going to learn about changing state.

9:09 — This video was really interesting. I’ll watch it again later and take more detailed notes in the next pomodoro session.

10:39 — I uploaded this to GitHub for a morning commit. Now let’s get back to the lessons.

10:45 — Bob Rizoll noted that when we define a function in our class, that when we are inside of the render, we need to reference the function using the “this” keyword. Example: this.myFunction.

10:46 — A strange observation I just made is that it seems like we don’t need to use the word function when defining functions inside of our classes. For example, the following seems to be perfectly valid code, contrary to my expectations:

handleClick() {

console.log("you clicked me up good")

}

10:50 — One of the reasons our classes tend to “extends” React.Component is that by doing so React.Component will give us access to the setState( ) method.

10:51 — According to the course instructor, anytime we want to change state we will do so using the setState( ) method.

10:54 — Get ready. This material is a bit dense. Let me paraphrase… Anytime you create a class method that you want to use state on, you’re going to need to bind said method to your class.

(Source: <https://scrimba.com/p/p7P5Hd/cLKWEcZ>)

(Extended listening on bind and this: <https://www.youtube.com/watch?v=GhbhD1HR5vk>)

10:57 — Let’s dive into the instructor’s code and see how exactly we should go about applying his above conviction we just finished discussing:

class App extends React.Component {

constructor() {

super()

this.state = {

count: 0

}

this.handleClick = this.handleClick.bind(this)

}

10:59 — The first thing we need to note is that we’re going to do our “binding” inside of our constructor function. Next, we’re setting this.nameOfOurMethod = this.nameOfOurMethod.bind(this).

11:01 — Let me be honest, I don’t have a deep understanding of this material. But, from my interpretation of the instructor’s lesson, what we’re basically doing with the above code is taking our method and binding it to whatever “this” is in the context of our current class.

11:04 — Hmmm… maybe I can talk about the above explanation on my next dinner date with the ladies hahahaa.

11:05 — My attention is starting to fade and I need to go to the bathroom. I’m going to check out for a little bit now and come back to the video with about 6 or so minutes remaining and finish taking notes in the next pomodoro session.

14:44 — Let’s get a quick 10 minute study session is.

14:46 — If we don’t care what the previous version of state was, we can simply pass an object to our setState( ) method.

14:48 — If we do care about what the previous state was, however, we need to approach things a bit differently. In this case, we need to pass a function into set.State( ). The function passed will then have a parameter of prevState (previous state).

14:55 — Unfortunately I don’t have more time to study now. When I come back to the code, I’ll work to implement a change in the Todo app that will actually enable the user to check and uncheck things by changing the state of isCompleted from true to false and rendering the proper style properties based on that state. While it seems like a simple change on the surface, it actually is somewhat complex upon actually diving into the code. Anyway, I’ll come back later. Peace!

16:38 — It started raining when I was walking so now I’m at the convenience store. But… all the chairs were taken so I made an improv standing desk with the ice cream freezer and my computer stand and now I’m learning here lol.

14:45 — I tried using the setState( ) method using the following syntax:

handleClick( ) {

this.setState({ count: 1 })

}

16:46 — After getting the following error:

TypeError: Cannot read property 'setState' of undefined (/App.js:38)

I knew that I had to somehow use state or setState to bind the current context of the this keyword to state. Unfortunately, I don’t remember the exact syntax and having only a surface level understanding of this concept, it isn’t something I can really recreate from scratch. Let’s go back and pick up the syntax again…

16:52 — Ahhh. I was close but I still a bit off on the syntax. I should’ve written:

this.nameOfMethod = this.nameOfMethod.bind.(this)

16:55 — While my ability to explain the above to a newbie would be pretty sketchy at this point, I have crammed it into at least my short-term memory and was able to construct the above from scratch.

17:03 — I’m just continuing to play around with the changing state video. I know this concept is super important so I was to really hammer it in before continuing with the course.

17:08 — While I haven’t yet built anything from scratch, I used the existing code as a reference to add a second button to my counter app that would double the current value of count. It may not be impressive to outsiders, but I slowly feel the possibilities of what I’m capable of doing with React expanding.

17:12 — It’s crazy that just a week ago I was feeling really shaky with the concept of props. I’ll be the first to admit that I’m far from being a master of *anything* in React. But… it does feel good to be able to look back at past entries and see how fast I’m learning. Anyway, I’m going to take a break now. See ya later!

**Total time spent coding today: N/A**

**Total time spent coding thus far in May 2019: N/A**

**Total lifetime hours of coding: N/A**