Cindy Liu

PUI Assignment 6B

11/7/2021

Reflection

The biggest challenge that I encountered was unable to debug because I wasn't used to checking my code with console.log. Especially when I received no error message, but the content wasn't showing as I expected. I learned and recognized the handy use of console.log after going to Humphrey's office hour. Later when I worked on my own, I used it to check 1) if the code runs to a certain line; 2) if the variable contains the correct content. This helped me to narrow down the problem to a certain line of code, rather than checking the bulk of code aimlessly.

Another challenge for me was the unfamiliarity of the language--javascript. I knew what to write, for instance a function that updates the order total prices, but I didn't know how to write it. The handout *Helpful Concepts for Assignment 6B* that Jesse sent out is useful as it suggests which concepts to use as well as their underlying logic. Since I was unfamiliar with these concepts, such as <template>, clone, and querySelector, I searched them further in W3Schools and Mozilla and gained an understanding of them.

Programming Concepts Learned

1. **Meaningful variable names.** When the code starts to be massive and complicated, I realize how crucial it is to name variables thoughtfully. By thoughtful names, I indicate the ones that 1) reflect the true meaning of the content. For instance, “productTitle” and “productImage” instead of “firstThing” or “secondThing”. 2) Be consistent. In my assignment 6, I named one content “itemName”, while the other “productImage”. However, I should be consistent in using the same prefix, either “item” or “product”, to avoid later confusion.
2. **Careful use of class and id.** Some of my bugs are caused by the misuse of getElementById and getElementsByClassName, which I use getElementById to find a class object or vice versa. Therefore, now I have the habit of double checking the class/id in my HTML, then write it in JS. In addition, some JS properties such as “.innerHTML” can only take an id, which I need to make sure that the corresponding element in my HTML should have an id rather than a class.
3. **Condition settings.** It is important that we elaborate on all conditions. For example, I wrote a function named addToCart that retrieves the previously added items to the cart array. While writing the code of retrieving items, we also need to consider what happens if there is no retrieving items; if that is the case, the cart should be an empty array. I always forget to include all conditions, which results in errors.
4. **Iterative revision between JS and HTML.** For me, I often go back and forth between JS and HTML, and I realize that it is not a one-edition process where

you write the HTML then JS and never go back to the HTML. As I’m writing the JS, I return to my HTML reassigning or changing the id or class names. With that revision, I may also need to make changes in my CSS.

5. **Define logic before you start.** To save my coding time in the long term, I spend some time defining the logic structure at the beginning—which feature/function I should write first, followed by which feature/function. This thinking process helps me to stay logical rather than doing fragments randomly. I also follow a top down process where I implement big features first (addToCart, updateProductInCart), then small features (updateOrderTotal, removeProduct).