BUN BUN BAKE SHOP

HW 6B Reflection + Lessons

Matthew Fang

Reflection

Let vs Var Bugs!

Bugs that I encountered early on was declaring objects and constants with the result of using let or var. I used var frequently at first, and occasionally, I would get reference errors. This happened because var is scoped to the immediate function body or the function body.

```
let result = await fetch("products.json");
//let's wait until we get the result
let data = await result.json();
```

Also, sometimes my var objects were changinig unexpectedly. This occurred because var can be redeclared, my var variable would change unexpectedly. However, I learned that if I used let, I wouldn't have to worry about that because let can't be redeclared in the same scope. **Thus, I began to use let in my code to resolve these issues!**

Make Sure You're Working with Arrays!

Another bug I encountered was working with the right data type after I extracted from the DOM. One of the things I had to do was extract all the add cart buttons from the DOM and store it in a variable for further manipulation. However, when I was trying to manipulate my list of buttons, nothing was working!

```
//get bag buttons
getBagButtons(){
    //spread operator will turn nodelist into arraay
    const buttons = [...document.querySelectorAll(".bag-btn")];
    buttonsDOM = buttons;
```

After some further debugging, I found out when I was using querySelectorAll to get my buttons, I was extracting a **nodelist instead of an Array**. This was a problem, so I solved this by **using the spread operator** on the DOM element I extracted, so my nodelist could be an array. This bug was a huge lesson to know what datatypes I'm working with once I extract them from the DOM.

Programming Concepts Learned

1. LocalStorage

This project was the first time I used LocalStorage. I thought this was extremely useful because I now had the ability to store informaton to save product items to a cart, and have the information be stored across different browser sessions! For me, I had the ability saveProducts and getProducts, which helped me access my product information, and saveCart and getCart, to access my cart across different sessions!

2. Create your JS variables in the beginning!

Initially when I was coding, I would pull elements from the DOM as I was coding. However, as my code became longer, it was harder to keep track of what I needed to manipulate, and I would waste a lot of time searching for the variable I created. Thus, to save time, I created all the variables that I know I would need, and left them at the top of my JS file. This saved me a lot of time and helped with debugging - I think this is a best practice I will incorporate more frequently in the future.

3. Using console.log to debug!

This may seem self-explanatory, but I found myself frequently trying to fix the code on my own whenever I encountered on a bug. However, after spending pointless time staring at my code, I realized console logging my variables at different point in my functions would help me understand where the errors were occuring. One example was the button example from the past page, where I didn't understand my error until I console logged my variable, and I found out I was working with a nodeList instead of an array.

4. Having Good Names in the HTML and Javascript

At first, my classes were named arbitrarily. Whenever I extracted elements from the DOM in the JS, a lot of things just didn't make sense. I had names like bigDiv and productElements, so when I was coding, I got a bit confused. I took the time to refactor by giving better names to not only make the JS easier to work with, but to also make my HTML more understandable. For example, I turned my class names into "Products" --> "section-title" --> "Products-Center".

5. EventListeners

I thought working with eventListeners was extremely interesting because now I had much more flexibility while working with buttons. I know longer had to use onClick() and I actually had much more flexibility with EventListeners, since I can call event.target (for me I was able to call event.target.value very frequently, and it was very helpful!

Additional Note:

-Implemented a carousel of similar products on individual product page
-You may have to run my files on a local because of the font-awesome licensing problems