Assignment 6B: Finishing the Shopping Cart

Programming Usable Interfaces
Michelle Lee

Reflection

One of the biggest roadblocks I encountered while programming this site was the management of in-scope variables and localStorage. When testing my site, I realized that users often go between the shopping cart and the browse page, however this interaction always resulted in unexpected behavior in my code. This is in large part due to the fact that I was instantiating variables to a default value every time the page was reloaded. These values would then get mixed into my localStorage variables causing all kinds of miscalculations and issues. To avoid this, I believe it's good practice to define variables with respect to localStorage as shown in this code snippet:

```
var bunsInBasket;
if (localStorage.getItem("bunsInBasket") != null) {
   bunsInBasket = JSON.parse(localStorage.getItem("bunsInBasket"));
} else {
   bunsInBasket = [];
}
```

Alternatively, I could use JSON.parse(localStorage.getItem("key")) and localStorage.setItem("key", JSON.stringify(data)) whenever I need to access the variable to ensure that I am using or updating the right value each time.

Another roadblock I encountered while programming this site was the decision of when to get an element by its ID or to pass in the element itself. I ended up switching back and forth between the two methods, however in order to maintain abstraction and not have the same function again and again with different IDs, I think it's best practice to pass in as much information as is needed to a generalized function.

The final big roadblock I faced was the process of adding HTML elements that were nested into one another. I went through a long process of creating elements and then assigning them to their respective parent elements which looks quite unapproachable, not to mention unreadable (look to script-basket.js). However, I realize now that I lacked an understanding of how the attribute .innerHTML worked. Rather than laboriously creating each element, assigning its class, text, and parent, I could've just used the HTML code I originally had and set the .innerHTML attribute of a to that code. There would still have to be some working out of the remove function with each item in the cart, however, this understanding of putting in blocks of HTML would have saved me a lot of time and effort.

Ultimately, I found that I was not able to implement all the ideas I had planned in my Figma mockups. For instance, the shopping cart modal never was developed. If I had more time, I would definitely want to explore that further, however, I've realized that front-end programming takes a lot of time and much more planning that I had expected. It's easy to "program as you go" and create things as you need them, but I've realized that this can lead to disorganized and/or fragmented code, which I fell into with the project at times. Thus, as I approach this next project, I want to be mindful of the feasibility of building out certain interfaces/interactions as well as the need for planning classes and JS functions before writing a line of code.

Five Programming Concepts

Five Javascript programming concepts that I used in this project are localStorage, events, objects, native methods, and variable scope. The site uses localStorage to store a list of all items in the shopping cart so that no matter what page the user is on, the information of what's in their cart is accessible and mutable. Also, the site uses events like onmouseover and onclick so that the user can interact with the site. For example, on the browse page, all the buns are hoverable to show a button to add the bun to your cart. Additionally, the core of the functionality of the site is a list of bun objects that's stored in localStorage as mentioned before. These objects hold attributes about a bun order such as the price of the order and the quantity of buns. This project also uses a variety of native Javascript methods to manipulate data and elements on the HTML pages. For instance, the methods getElementById() and createElement() are used extensively throughout the project. Lastly, this site uses the variables with a variety of scopes to manage data. You can see this in the basket script where each iteration of the for loop assigns respective values to the critical variables. Through making a site that employs all these concepts, I've learned how to use them (as well as how not to use them).