Yumi Sato

HW6: Reflection and Programming Concepts

Reflection

While completing this assignment, I faced numerous obstacles. The biggest challenge I faced was implementing my shopping cart page. First, I had trouble getting the products in the shopping cart to align vertically, with the text descriptions to the right of the product image. Initially, I had implemented by using an unordered list and having each product be a list element. However, I scrapped this implementation and chose to implement it with each product having its own div. This made my life much easier, as it also helped me implement my order summary box and removing products more easily. Prior to this, I did not have a solid understanding of what a div was and how it worked, but after overcoming this challenge, I now have a solid understanding of divs. In the future, I think I will first try to implement similar features with a div rather than an unordered list, as I think divs are more versatile and can adapt to most situations.

Another major obstacle was implementing the removal of products and dealing with its side-effects properly. For example, while I removed the HTML elements from the page, I had trouble removing the relevant data from sessionStorage. Specifically, I had trouble trying to access into the array in order to remove them, because I could not figure out how to find their indices. I ended up looping through the contents of the array and individually comparing them. When they matched, I manually removed them one by one. In the future, I would like to figure out a better way to do this, as my current implementation is not very efficient.

Programming Concepts

1. Using Session Storage to store data between different HTML pages.

   I used session storage to store items that the user added to their shopping cart so that I could show them in the shopping cart page. In addition, I used it to keep track of when the user would remove items from their cart as well. Here is one example of when I used session storage in my code.

```
//when "add to cart" clicked, store the roll info into
//session storage
function addOriginalRoll() {
    var info = originalRoll.toString();
    //check if rolls already in cart
    if (sessionStorage.rolls) {
        sessionStorage.rolls = Number(sessionStorage.rolls) + 1;
        sessionStorage.cart = sessionStorage.cart.concat(',',info);
    }
    //if first item to add
    else {
        sessionStorage.rolls = 1;
        sessionStorage.cart = info;
    }
}
```

2. Adding HTML elements to a page via “appendChild(),” “createElement(),” and “createTextNode().”

When updating the shopping cart page to show the user what they currently have in their shopping cart, I used these functions. Here is one place in code where I used these functions.

```
//creates the order summary box on the right of the cart page
function createOrderSummary (subtotal, summary) {
    //initialize variables
    var headingT = "Order Summary";
    var itemsT = "Items: $" + subtotal.toFixed(2);
    var shipHandleT = "Fees: $4.50"
    var subtotalT = "Subtotal: $" + (subtotal + 4.50).toFixed(2);

    //create the HTML elements
    var heading = document.createElement("H2");
    var items = document.createElement("H3");
    var shipHandle = document.createElement("H3");
    var total = document.createElement("H2");

    heading.appendChild(document.createTextNode(headingT));
    items.appendChild(document.createTextNode(itemsT));
    shipHandle.appendChild(document.createTextNode(shipHandleT));
    total.appendChild(document.createTextNode(subtotalT));

    //will be used to recalculate cost when removing products
    items.id = "itemCost";
    total.id = "subtotalCost";

    //add to the summary div
    summary.appendChild(heading);
    summary.appendChild(items);
    summary.appendChild(shipHandle);
    summary.appendChild(total);

}
```

3. Converting between numbers and strings via “.toString()”, “.toFixed()”, and “parseInt().”

When calculating total costs and item costs, I used these functions so that I could get the data from session storage, where the numbers are stored as strings, convert them to numbers, do the math, then convert back to strings to insert them as text in my HTML. Here is one example:

```
else {
    sessionStorage.rolls = total;
    sessionStorage.cart = rolls;

    //update order summary
    var itemT = document.getElementById("itemCost");
    var subT = document.getElementById("subtotalCost");
    var itemC = Number(itemT.textContent.slice(8));
    var subC = Number(subT.textContent.slice(11));
    var minus = Number(num) * 3.85;
    itemC -= minus;
    subC -= minus;
    itemT.textContent = "Items: $" + itemC.toFixed(2);
    subT.textContent = "Subtotal: $" + subC.toFixed(2);
}
```

4. Accessing HTML elements via “getElementById().”

   I used this concept whenever I was changing the HTML contents, mainly in the product details page and the shopping cart page. Here is an example:

```
//give feedback when picking quantity, bg-color changes
function pickQuantity(item) {
    //get the html elements
    var one = document.getElementById("one");
    var three = document.getElementById("three");
    var six = document.getElementById("six");
    var twelve = document.getElementById("twelve");
    //if quantity = 1
    if (item.id == "one"){
        one.style.backgroundColor = "#fce5cdff";
        three.style.backgroundColor = "white";
        six.style.backgroundColor = "white";
        twelve.style.backgroundColor = "white";
        originalRoll.num = 1;
    }
    //if quantity = 3
    else if (item.id == "three") {
        one.style.backgroundColor = "white";
        three.style.backgroundColor = "#fce5cdff";
        six.style.backgroundColor = "white";
        twelve.style.backgroundColor = "white";
        originalRoll.num = 3;
    }
    //if quantity = 6
    else if (item.id == "six") {
        one.style.backgroundColor = "white";
        three.style.backgroundColor = "white";
        six.style.backgroundColor = "#fce5cdff";
        twelve.style.backgroundColor = "white";
        originalRoll.num = 6;
    }
    //if quantity = 12
    else {
        one.style.backgroundColor = "white";
        three.style.backgroundColor = "white";
        six.style.backgroundColor = "white";
```

5. Changing the HTML elements' classes, IDs, and style.

   I used this concept when I was adding HTML elements via Javascript and updating the page to respond to the user's actions (ie when the user personalized their product). Here is an example:

```
//if glaze = none
if (item.id == "none") {
    none.style.backgroundColor = "#fce5cdff";
    sugar.style.backgroundColor = "white";
    vanilla.style.backgroundColor = "white";
    chocolate.style.backgroundColor = "white";
    originalRoll.glaze = "none";
}
//if glaze = sugar-milk
else if (item.id == "sugar") {
    none.style.backgroundColor = "white";
    sugar.style.backgroundColor = "#fce5cdff";
    vanilla.style.backgroundColor = "white";
    chocolate.style.backgroundColor = "white";
    originalRoll.glaze = "sugar";
}
```