Jay Annadurai

MAD SP23

T. Faas

14 Feb 2023

Exercise 7: User Input

[Link to GitHub](https://jannadurai.com/Exercises/Exercise%207/index.html)

**Introduction**

     The HTML DOM Exercises are the first set of exercises to discard the p5.js library and rely entirely on vanilla Javascript. The goal of the exercise set is to establish the basics of interacting with the HTML Document Model hierarchy and its corresponding JS Object Model. This particular set modifies style properties of HTML elements utilizing event listeners.

**Algorithms & Planning**

     All files contain a similar structure of potential JS Classes followed by any general JS which is then ultimately followed by any JS functions. Within the specific exercise set, all exercises have the same program flow of defining a HTML object as a JS object and using signature methods within a corresponding HTML Event listener.

**Reflection**

     Ex4.1 was the basis of all the other exercises. The core functionality developed in Ex4.1 could be extended to encompass the functionality required in Ex4.2 and Ex4.3. As such, it was developed somewhat modularly. Initially, the Exercise used an algorithm as seen as in the pre-Application algorithm above: Create a JS Object Literal to represent attributes and properties of a HTML Object, Create the HTML Document Entity with the attributes of the JS Object, Create a Function to accomplish the desired functionality and enact it on the JS Object, and lastly update the HTML Document Entity.

The first issue came in the form of extraneous amounts of repetition. Each attribute and property was developed individually and the solution was to take advantage of the 'forIn' loop and iterate through Objects as if they were associative arrays. However, I needed to call specific JS Methods to alter the HTML element's properties. The only method I could gather from Googling was using the Window object and using array notation to call the methods. However, the Window object only stores global variables and usage of global variables created issues with scope. With Professor Faas's help, I realized array notation could be applied directly to objects and that rapidly improved my code.

The final iterative stage came in the form of shifting from an 'object literal with paired functions' approach to a 'instantiated object with instance methods' approach. This would maximize the modularity and minimize unique code. This conversion was a bit troublesome due to typecasting and my realization that classes do not benefit from the same program flow exception that functions do. Nevertheless, the class's completion dramatically minimized code outside the class. Any methods with moderate utility, such as the scale method of Ex4.1 and the colorChange method of Ex4.3 were directly implemented into the class. Extending the code from Ex4.1 to Ex4.2 resulted in a few errors mainly from artifacts of the initial functional approach but they were quickly resolved. Ex 4.2's signature 'mcDiv' function was modified to accept the HTMLasJS object rather than an HTML element.

Overall, I'm quite happy with my solutions to exercise set 4. While the final solutions do not look spectacular, and that is definitely an area for improvement, the backend is highly modular and can be iterated on for successive sets of exercises. The process in this week's code made me more intimate with the nature of JS and its objects. Functionality I thought that didn't exist was elucidated to me in the form of Objects and their array notation in JS. I'm very excited to see how this information can benefit my future exercises.