Zephyrus Engineering Coding Challenge Explanations

Task 1: Reverse String

The first thing I do is a quick check that the input indeed is a string. Next, the simplest, cleanest solution is using the array reverse method (as I've shown in my first approach). You could also use a standard "for" loop to iterate backwards along the string, adding each character to a new string. For fun, I've included a second approach where I do the array reverse part myself using a "forEach" loop. The loop iterates forward through the characters to build a new array from the end to the beginning of the array (i.e. the first character gets placed in the last index of the new array, and so forth).

Task 2: FizzBuzz

For this, I just use modulus for the various conditions. "FizzBuzz" is just multiples of 15, so I know to check that condition first, otherwise "Fizz" and "Buzz" will also print for multiples of 15. Then I check the next two specific conditions, then the most common general case of just printing the numbers themselves encompasses everything else, so can be left for last.

Task 3: Even or Odd

Here we don't want incorrect results due to inputs like decimals or fractions, so I only allow integers. Then, since there are only two possibilities to return, I use a ternary operator.

Task 4: Simple Interactive Feature

I decided to make the page a little more interactive by allowing the user to input what they want to change the heading to. My HTML page consists of an h1, input field, and button, grouped using divs. The button starts disabled; and both the input and button have an event handler.

The CSS is simple flexbox to center the elements on the page. I also change the color and cursor for the button when enabled and hovered over.

Finally, I have two JavaScript functions. The first is triggered whenever the user types into the input field. Its purpose is to check whether the input has a value or not and enable or disable the button accordingly. I call trim() on the value to account for the edge case where the user just enters space(s), so the heading can't be changed to that. The second function is triggered when the enabled button is clicked. It changes the heading to the value that the user entered into the input field, if valid. After that, it also clears the input field and disables the button so that a new value must be entered to change the heading again. This accounts for the very rare edge case where the user enters a value, then simultaneously holds the "delete" key down clearing out the entire input field, and without letting go of the "delete" key (such that the button does not yet have a chance to become disabled again), clicks the button, thereby changing the heading to an empty string. It's very unlikely the user would ever perform this highly specific sequence of actions, but I prevent the potential "bug" just in case.

One last note, although it might seem like overkill for this task, as I could nevertheless do it even cleaner and easier, I would be happy to do it in Angular if you like ☺.