Section 8- JavaScript Skill Assessment

Firstly, I'd like to say I am very disappointed in this skill assessment. Or rather, its pairing with the content that proceeds it. The JavaScript section is pretty rushed, but honestly not a bad *very* quick introduction to the language. However, at *no point* did the Javascript section really discuss its integration *with* HTML. We were not instructed how to use <script> tags, how to access elements of the document, or really any of the required skills needed to complete this assignment.

Technically you can just use the built in HTML verification for forms: If you set the input for the email field and make the field required, HTML will do some rudimentary validation, but thats not what is being asked for, nor is that relevant to learning JS.

Instead, I had to spend several additional hours searching out content on my own to learn the skills needed to complete this assignment. I'm not really upset by that, this is a *FAST* and fairly shallow class: I am often doing additional research to gain a deeper understanding of what we are learning in the course. This is, however, the first time I've seen where the course basically fails entirely:

The Intro JS section was… fine. Honestly, this is an excellent skill test as well. But the two do not match. There needs to be another whole section on integrating JS with HTML for interactive processing of pages. Really it should be one of the most important sections in this *entire course,* and I am pretty disappointed in how shallow this section is.

This would have been a wonderful way to introduce accessing elements in the DOM, or an intro to Regular Expression matching.

So, I've gone ahead and done the extra work to learn for myself the things needed to complete this assignment. Again, its not the assignment that is unfair, but the lack of material to prepare us students to *do* the assignment that is pitifully lacking.

—----------

I began with the client creation page. Following handy tutorials from multiple sites, mostly from Mozilla, I was able to disable the built in verification on forms (novalidate) and instead add my own JS validation. First this required some restyling of the inputs and including an active span element for each field so we could display error messages. In a JS script section, within the body of the HTML page itself, I initialized a number of variables to catch the contents of the form, looking at the following fields: Full Name, Email, Password1, and Password2. We then add various event listeners

The Input listener is constantly updated as users enter input. This listener is only active on the email field, and highlights the email field in red while the current text is an invalid email string (per css styling, added to the style sheet of this page specifically for these new error handline elements). The matching for this is done via regex, using a standard (and not very robust) email

string match. It more or less just requires some characters, an @ and then some more chars. Email valid matching is notoriously difficult, so I am leaving this very basic test for now.

The second main listener is active when a user clicks the submit button. It then verifies that the username is not of length 0 (no other validation tests are done for a user name), that the email field is non-empty and still a valid email form, according to our simple regex, and that the password fields are both at least 6 characters long, and the two passwords entered actually match.

In each case, if successful no message is displayed and the form is allowed to be submitted. This of course does nothing for us, as nothing is happening with the form yet. However, if any of these validations fail, a red pop-up error message appears next to the offending field advising the user on correcting their input.

For the Client Meeting page, I had to remove the type=”date” input, as that forces a calendar dropdown such that the input is of course valid. Instead, I made a text input for the date. Again, we add a listener to the submit button and validate the value entered into this field against a regular expression. In this case I played around with the RE to reacquaint myself with them: I often find date inputs are snottily specific. I crafted a looser date input that doesnt require leading zeros (or happily accepts them) for days or months, will take any of the following separators “ / , - ” between DDMMYY, and will accept a two or four digit year, but not three any other set. It accepts only valid combinations of digits for days and months (and would prevent you from inputting say the 45th day of the 22nd month), but does no such validation for the year. If I had more time I’d like it to reference the *actual* time, parse the input, and make sure the requested date is further than 24 hours from the current time. Anyhow, this version works well for now.