In this lesson, we’ve explored form validation from many angles. Let’s review what we covered:

* Modern websites require a lot of information from their users and they collect a lot of this information through HTML forms.
* It’s essential to validate the data submitted through forms to keep websites secure and to make sure they function correctly.
* Regular expressions are sequences of characters that define patterns to look for in text. They are an important tool used in validating input.
* Modern HTML comes with useful built-in methods for form validation.
* Custom and complicated client-side validation can be accomplished with JavaScript.
* Asynchronous requests to the server can perform back-end validations before a form has been submitted.
* A final back-end validation of all data is required to ensure an application’s security and sanitize all data.

Nice work interacting with the extremely common and useful <form> element!

In this lesson we went over:

* The purpose of a <form> is to allow users to input information and send it.
* The <form>‘s action attribute determines where the form’s information goes.
* The <form>‘s method attribute determines how the information is sent and processed.
* To add fields for users to input information we use the <input> element and set the type attribute to a field of our choosing:
  + Setting type to "text" creates a single row field for text input.
  + Setting type to "password" creates a single row field that censors text input.
  + Setting type to "number" creates a single row field for number input.
  + Setting type to "range" creates a slider to select from a range of numbers.
  + Setting type to "checkbox" creates a single checkbox which can be paired with other checkboxes.
  + Setting type to "radio" creates a radio button that can be paired with other radio buttons.
  + Setting type to "list" will pair the <input> with a <datalist> element if the id of both are the same.
  + Setting type to "submit" creates a submit button.
* A <select> element is populated with <option> elements and renders a dropdown list selection.
* A <datalist> element is populated with <option> elements and works with an <input> to search through choices.
* A <textarea> element is a text input field that has a customizable area.
* When a <form> is submitted, the name of the fields that accept input and the value of those fields are sent as name=value pairs.

Using the <form> element in conjunction with the other elements listed above allows us to create sites that take into consideration the wants and needs of our users. Take the opportunity to take what you’ve learned, and apply it!

Awesome job adding client-side validation to <form>s!

Let’s quickly recap:

* Client-side validations happen in the browser before information is sent to a server.
* Adding the required attribute to an input related element will validate that the input field has information in it.
* Assigning a value to the min attribute of a number input element will validate an acceptable minimum value.
* Assigning a value to the max attribute of a number input element will validate an acceptable maximum value.
* Assigning a value to the minlength attribute of a text input element will validate an acceptable minimum number of characters.
* Assigning a value to the maxlength attribute of a text input element will validate an acceptable maximum number of characters.
* Assigning a regex to pattern matches the input to the provided regex.
* If validations on a <form> do not pass, the user gets a message explaining why and the <form> cannot be submitted.

These quick checks help ensure that input data is correct and safe for our servers. It also helps give users immediate feedback on what they need to fix instead of having to wait for a server to send back that information.