**Making the most of client-side validation**

Have you ever completed a form online

and then after you've clicked on the submit button,

one of the fields turned red.

There are many reasons for

error messages when you complete online forms,

the format could be wrong,

but sometimes an error results when a field is

empty or if a value is too long or too short.

As a developer, you can use HTML and CSS to

guide users to enter

the correct data on forms and in this way,

save web server resources.

In this video, you will learn more about

the capabilities of client-side validation in

HTML and how to use CSS to

effectively alert users when data is incorrect.

By now you've learned about

the different HTML input types

and how they validate the format of data,

but there are additional attributes that you can

apply to elements for further validation.

First, let's explore how you can ensure

that the user provides a value to a form field.

This example of a log-in form contains two fields,

one for the username and one for the password.

It also contains a submit button to submit the form.

If a user hasn't added a username

or password and clicks the "Submit" button,

the browser will allow

the form to submit to the web server.

But this is a waste of server resources because although

the server-side validation will pick up

the missing information and

prevent the processing of the request,

a HTTP request was still submitted.

Let's improve this form by using

client-side validation to ensure

the user enters values to both fields.

Now with the required attribute in the code,

if one of the fields is empty and

the user clicks the submit button,

the request will not be submitted to the web server.

The web browser will focus on

the first empty input element and

inform the user that a field is empty.

Now the form creates a much better user experience

and prevents unnecessary usage of web server resources,

but user input that is too short or too long

would also result in

unnecessary submissions to the web server.

So another way to validate user input is

to specify the required length for data.

Let's say you want the usernames to be at least three

characters long and a maximum of 12 characters,

or maybe you want to ensure more secure passwords

by requiring a minimum password length

of five characters.

Fortunately, there are two more attributes you can

use to ensure the correct length of data.

You can add the min length and max length

attributes to the fields to specify the required length.

In this case, since

the minimum value for the username is set to three,

a two-character entry would be invalid.

If you click the submit button now,

the web browser will inform

you of the length requirements.

A built-in function of the browser will

generate the message displayed on screen.

Again, a much better user experience

and management of web server resources.

Now let's focus on the way the browser

communicates errors to the users,

like a form field that turns

red when the data is invalid.

This can be readily implemented using CSS.

To highlight a field in red when the data is invalid,

you use the input element selector.

You then apply the pseudo-class selector called invalid.

You will learn more about pseudo-class selectors

later in this course.

For now, the key thing to know is that they are

used to select elements based on their state,

such as whether the input data is valid or invalid.

Then you add a CSS property

to set the border of the element to read.

But since all the form fields are

empty when the form loads initially,

their state is still practically invalid.

This will cause all form fields to display read from

the start like demonstrated in

this Real-world example of a login screen.

Ideally, you only want fields to appear

red as the user enters invalid data.

To do this, you need to apply

another pseudo-class called focus,

which checks if the user is currently

entering data into this field.

Now the form will appear normal when

the page first loads,

and when the user starts inputting data,

the field will turn red if the data is invalid.

This video gave you

a brief introduction to how you can use

client-side validation and CSS

pseudo-class selectors to improve

the user experience of forms.

As you learn more about JavaScript in other courses,

you will be able to use even more advanced techniques

to improve form validation.

What additional attributes can you add to an element on an HTML form to improve client-side validation? Check all that apply.



maxlength

Correct

That's right. You can specify the maximum number of characters allowed for a form field by adding for example:

maxlength="12" inside the input type element.



required

Correct

That's right. If it is essential that a user submit a response to a specific field on your form, you should add:

Required inside the input type element.



invalid data



minlength

Correct

That's right. You can specify the minimum number of characters required for a form field by adding for example:

minlength="3" inside the input type element.