Validation and JavaScript

WE'LL COVER THE FOLLOWING

- The JavaScript side of the story
- Turning validation off
- New HTML5 input types

The JavaScript side of the story

Although validation attributes help you in many situations, in complex systems you cannot avoid writing your own JavaScript validation code or using a validation framework. Browsers apply form validation only when you click a submit button. Great web applications with enhanced user experience validate fields as soon as you leave them, or sometimes as you type. In these situations, you cannot only leverage validation attributes, you also need to create a custom validation code. When the validation logic does not fit into the frame provided by the validation attributes, you fall back to writing JavaScript. NOTE: Custom validation is a complex topic and often bound to server-side frameworks.

Turning validation off

There might be situations where you need to turn off form validation. For example, if you have a long form with many inputs, you need to provide a way users can save the input already typed in, without validating the data in order to finish filling out the form later. You can add a second submit button that saves the form. Append the `formnovalidate` attribute to the submit button to avoid form validation before submission, like in the following example:

index.html

```
<form action="processform.html" method="post" >
  <!-- Input fields -->
  <input type="submit" value="Process" />
  <input type="submit" value="Save"
    formaction="SaveForm.aspx"
    formnovalidate />
</form>
```

You already learned that the server side must always validate data received from the client side. When you want to test whether the validation at the server side works as you expect, you need to transfer invalid data to the server side.

Adding the `novalidate` attribute to the `<form>` tag turns the client-side validation off. It provides you a context to prove that the server-side validation logic is invoked properly, and this logic can be triggered from the browser.

For example, the following code snippet turns off the client-side form validation; so, if you specify invalid data, you can be sure the wrong information will reach the server side:

index.html

```
<form action="processform.html" method="post" novalidate >
  <!-- Input fields -->
  <input type="submit" value="Process" />
</form>
```

New HTML5 input types

The HTML5 markup defines new input types, such as `color`, `date`, `datetime`, `datetime-local`, `email`, `month`, `number`, `range`, `search`, `tel`, `time`, `url`, `week`.

Unfortunately, they all have different support in different browsers, which means that some of them are not supported in certain browsers or that different browsers render the same control with totally diverse appearance.

When creating web pages, check that the input type you intend to use is supported by the set of browsers utilized by you or your customers.

Now that we have satisfactorily covered everything we need to know regarding forms and controls, let's summarize what we gained from this chapter, in the *next lesson*.