**Chapter 10 Notes**

**MDN: Validating forms** - <https://developer.mozilla.org/en-US/docs/Learn/HTML/Forms/Form_validation>

**MDN: Using Fetch** - <https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch>

**Form Data Validation**

Form validation types:

* Client-side
  + Javascript
  + Built-in
* Server-side
  + Not as friendly as client-side because it doesn't provide errors until the entire form has been submitted.

**Built-in**

HTML5 simplest validation feature: require attribute.

Ex.

1 
2 
3 
4 
5 
< form > 
(label for:" choose" >Wou1d you prefer a banana or cherry?</label 
(input " choose" like" required> 
(button /button > 
form> 

1 
2 
3 
4 
5 
6 
7 
input: invalid { 
border: 2px dashed red; 
input: valid { 
border: 2px solid black; 

This just simply requires that there should be something written in the input field, otherwise the border turns dashed red.

**Pattern attribute**

• a — Matches one character that is a (not b , not aa and so on.) 
• abc — Matches a , followed by b, followed by c 
a I b Matches one character that is a or b _ 
• abc I xyz — Matches exactly abc or exactly xyz (but not abcxyz or a or y, and so on). 
• There are many more possibilities that we don't need to cover here. 
Let's implement an example. Update your HTML to add a pattern attribute like this: 
1 
2 
3 
4 
5 
< form > 
(label for:" choose" >Wou1d you prefer a banana or a cherry?</label> 
(input id=" choose" like" required pattern: "banana I 
(button /button > 
form> 

\*\*Note\*\*

Email type, or url type <input> elements don't require validation

<textarea> element doesn't support pattern attribute

**Minlength and maxlength attribute**

* Sets a specified value length for the input

1 
2 
3 
<input type= "text" id="choose" name= " i_like" required minlength= "6" 
4 
5 
6 
7 
max:' 10 > 
8 
9 
le 
11 
12 
13 
< form > 
<1abe1 for=" you prefer a banana or a 
</div> 
<1abe1 for=" number" >How many would you 
<input type= " number" " number" name:" amount" value: "1" 
</div> 
< button > Submit</button> 
</div> 
form> 
mln= 

**setCustomValidity()** in javascript

1 
2 
3 
4 
5 
< form > 
(label for: would like you to provide me an e-mail</label> 
(input "mail" name: "mail" > 
(button /button > 
form> 
JavaScript; you call the setcustomvalidity() method: 
1 
2 
3 
4 
5 
6 
7 
8 
9 
document . getE1ementById ( "mail " ) ; 
var email 
email . addEventListener( "input", function 
event 
if (email. validity. typeMismatch) { 
email . setcustomvalidity( "I expect an e-mail, darling! " 
} else { 
email . setCu stomVa1idity ( " " ) ; 
I would like you to provide me an e-mail asdfgl 

* The error will show in the form of a pop up notice underneath the text input

**From validation using Javascript**

Constraint Validation API

Constraint validation API properties 
Property 
validationMessage 
validity 
will Validate 
Description 
A localized message describing the validation constraints that the control doesn't satisfy (if any) or the empty string if the 
control is not a candidate for constraint validation (willvalidate is false) or the element's value satisfies its 
constraints. 
A validitystate object describing the validity state of the element. See that article for details of possible validity 
states 
Returns true if the element will be validated when the form is submitted; false othenvise_ 
Constraint validation API methods 
Method 
checkvalidity() 
HTMLFormE1ement . reportVa1idity( ) 
set CustomVa1idity (message) 
Description 
Returns true if the element's value has no validity problems; false otherwise. If the element is 
invalid, this method also causes an invalid event at the element. 
Returns true if the element or its child controls satisfy validation constraints. When false is 
returned, cancelable invalid events are fired for each invalid element and validation problems are 
reported to the user. 
Adds a custom error message to the element; if you set a custom error message, the element is 
considered to be invalid, and the specified error is displayed This lets you use JavaScript code to 
establish a validation failure other than those offered by the standard constraint validation APL The 
message is shown to the user when reporting the problenm 
If the argument is the empty string, the custom error is cleared. 

|  |  |
| --- | --- |
| **< form novalidate >** | This will turn off the browser's automatic validation and will let the script take over validation, but you can still do it yourself |
|  |  |

**Using Fetch**

* The **Fetch API** provides a global **fetch()** method

* Fetch specifications wont reject on HTTP error status
* Fetch won't send or receive any cookies by default

Basic fetch code:

1 
2 
3 
4 
5 
6 
7 
fetch ( ' http : //example.com/movies.ison ' 
. then (function ( response 
o; 
return response.json 
. then { 
console . log (ISON . ) 

* Fetch can accept a second parameter, **init**, object that allows you to control a number of different settings

Example of fetch with second parameter:

1 
2 
3 
4 
5 
6 
7 
8 
9 
10 
11 
12 
13 
14 
15 
16 
18 
19 
20 
21 
23 
// Example POST method implementation : 
postData( ' http://example.com/answer ' , 
{answer: 42}) 
. then(data console. . stringify(data 
//' JSON-string from 
. catch(error console. error error 
response .json() 
only-if-cached 
call 
function postData (url 
data 
// Default options are marked with * 
return fetch(url, 
method: 'POST', // *GET, POST, 
mode : 
// no-cons, cors, 
cors 
cache : 
'no-cache' 
//' *default, 
c redentials : 
' same-omgln 
headers : 
PUT, DELETE, etc. 
same-or Igun 
no-cache, reload, force-cache, 
include, 
* same-origin, omit 
' Content-Type ' 
' application/j son ' 
// 'Content-Type' : 'application/x-www-form-urlencoded ' 
redirect: 'follow', // manual, *follow, error 
' no-referrer' 
// no-referrer, 
*client 
referrer : 
body: JSON 
. then (response 
. stringify(data), // body data type must match "Content-Type" header 
response.json()); // parses JSON response into native JavaScript objects 

**Credentials:** - causes browsers to send a request with credentials included, even for a cross-origin call

Credentials: 'include'

'same-origin'

'omit'

* **Fetch()** can upload JSON data, file upload, and multiple file upload

* It can process a text file line by line

* It can error check

1 
2 
3 
4 
5 
6 
7 
8 
9 
10 
11 
fetch( 'flowers . jpg' ) . then (function (response 
response. ok) { 
return response .blob(); 
throw new Error( ' Network response was not ok. ' 
} ) . then(function(my810b) { 
URL. createObjectlJRL (myB10b) ; 
var objectlJRL 
objectURL; 
mylmage. src 
} ) . catch(function (error) { 
console. log( 'There has been a problem with your fetch operation: 
error. message 

**Headers**

* Allows you to create your own headers object via the **Headers()** constructor

1 
2 
3 
4 
5 
var content 
var myHeaders 
'Hello World' ; 
new Headers() 
myHeaders . append( ' Content -Type' , 'text/ plain ' 
myHeaders . append( ' Content-Length , 
content . length . toString() ) ; 
myHeaders . append( 'X-Custom-Header' , 'ProcessThisImmediate1y' ) ; 
The same can be achieved by passing an array of arrays or an object literal to the constructor: 
1 
2 
3 
4 
5 
new Headers({ 
myHeaders 
' Content -Type' 'text/plain ' 
'Content-Length ' • 
'X-Custom-Header' : 
content . length . toString() , 
' ProcessThisImmediate1y' , 
The contents can be queried and retrieved: 
1 
2 
3 
4 
5 
6 
7 
8 
9 
10 
console. log(myHeaders . has( 'Content-Type ' //' true 
console . log (myHeaders . has( ' Set -Cookie 
' ) ) ; // false 
myHeaders . set( 'Content -Type' , 'text/html ) ; 
myHeaders . append( 'X-Custom-Header ' , 
' AnotherVa1ue ' 
console. log(myHeaders . get( 'Content-Length ' // 11 
console . log (myHeaders . get( 
' ) ) ; //' [ 'ProcessThisImmediate1y', 'AnotherVa1ue ' ] 
' X-Custom -Header 
myHeaders . delete( 'X-Custom 
-Header ' 
console . log (myHeaders . get( 
' X-Custom -Header ' 

* Headers have a guard property. Here are some of the values:

Machine generated alternative text:
• none: default. 
• request: guard for a headers object obtained from a request (Request .headers 
• request-no-cors: guard for a headers object obtained from a request created with 
Request . mode no-cors. 
• response: guard for a Headers obtained from a response (Response. headers). 
• immutable: Mostly used for ServiceWorkers; renders a headers object read-only 
Note: You may not append or seta request guarded Headers' Content-Length header. 
Similarly, inserting Set -Cookie into a response header is not allowed: ServiceWorkers are 
not allowed to set cookies via synthesized responses. 