

DOM Cont.

User Interfaces

420-WC4-AB

Events



Event Handling

- When you want to create events in your JavaScript file instead of using the HTML attributes you can use JavaScript to "listen" for specific events
- Mouse events
 - mousedown, mouseup, mouseover, mouseout, click, dblclick, mousewheel
- Keyboard events
 - keydown, keypress, keyup
- Form events
 - focus, blur, change, submit
- Window events
 - scroll, resize, hashchange, load, unload

Events in HTML

- These event are added as attributes to HTML elements

```
<input type="button" onClick="doSomething()" value="Click Me" />
```

- Within the quotes we can add the JavaScript that can be executed
- There are many events that can occur depending on the action/event taking place with the HTML elements.

Events in HTML

- HTML

```
<button id="myBtn" onclick="sayHi('Bob')">Click Me!</button>
```

- JavaScript

```
function sayHi (name) {  
    alert('Hi ' + name );  
};
```


Common Events

<code>onchange="myFunc()"</code>	An HTML element has been changed
<code>onclick="myFunc()"</code>	The user clicks an HTML element
<code>onmouseover="myFunc()"</code>	The user moves the mouse over an HTML element
<code>onmouseout="myFunc()"</code>	The user moves the mouse away from an HTML element
<code>onkeydown="myFunc()"</code>	The user pushes a keyboard key
<code>onkeyup="myFunc()"</code>	The user released a keyboard key
<code>onload="myFunc()"</code>	The browser has finished loading the page
<code>onfocus="myFunc()"</code>	An element is focused on
<code>onblur="myFunc()"</code>	A form element loses focus
<code>onsubmit="myFunc()"</code>	Form has been submitted

Event Listening

- HTML

```
<button id="myBtn">Click Me!</button>
```

- JavaScript

```
let button = document.getElementById("myBtn");  
  
button.addEventListener("click", function (event) {  
    alert("Hi!");  
});
```


Event Listening

- JavaScript alternative – *reusable function*

```
let button = document.getElementById("myBtn");
```

```
function sayHi(event) {  
    alert("Hi!");  
};
```

```
button.addEventListener("click", sayHi);
```


Preventing Defaults

- Links and checkboxes have default behaviors controlled by the browser
- Our events can prevent these default behaviors.

```
//assuming myLink is an anchor link in our HTML  
let link = document.getElementById("myLink");  
  
link.addEventListener("click", function(event) {  
    event.preventDefault();  
});
```


Form Validation

Name (Print)

Signature _____

Date _____

Form Validation

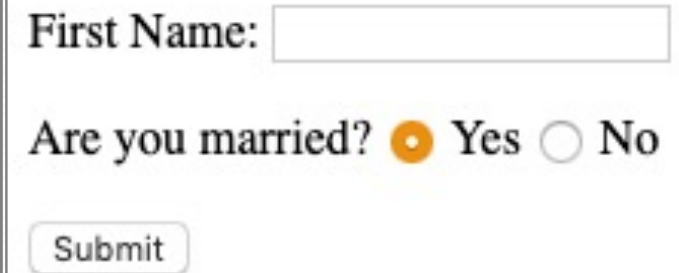
- Before you submit your form data to the backend, you should perform some front-end validation.
 - You should still perform server-side validation
- Validation Methods
 - Built-in HTML5 form validation
 - Custom JavaScript
 - Third party validation libraries

Built-in Form validation

- HTML5 validation, allows form validation with no javascript
- Uses HTML attributes
 - required, minlength, maxlength, min, max, type ...
- Uses CSS pseudo-classes
 - :valid, :invalid, :out-of-range
- If a user attempts to send the data, the browser is responsible for blocking them and displays an error message.

Forms

```
<form id="userForm">
  <p>
    First Name:
    <input type="text" id="name" name="name"/>
  </p><p>
    <span>Are you married?</span>
    <input type="radio" name="married" value="Yes" checked /> Yes
    <input type="radio" name="married" value="No" /> No
  </p>
  <input type="submit" id="submitBtn" value="Submit" />
</form>
```



First Name:

Are you married? ☒ Yes ☐ No

Form – Retrieve Data

- To retrieve data from form element we use the *value* method of JavaScript
- You can access simple data easily

```
let name = document.getElementById('name').value;
```

- You can retrieve the value of a form at any time.
 - Used on inputs and select elements
 - An event like blur is triggered when a form element loses focus

Form – Retrieve Data

- Radio's usually require a loop since there is more than one

```
let radios = document.getElementsByName('married');
let length = radios.length;
let radioValue;

for (let i = 0; i < length; i++) {
  if (radios[i].checked) {
    radioValue = radios[i].value;
    break; // since only ever 1 radio is checked - leave the loop
  }
}
```

Form – Retrieve Data

- Checkboxes can be verified using their attribute *checked*

```
// Get the value
```

```
var remember = document.getElementById('rememberMe').checked;
```

```
// Set a new value so it is checked
```

```
document.getElementById('rememberMe').checked = "checked";
```

```
// Set a new value so it is NOT checked
```

```
document.getElementById('rememberMe').checked = "";
```


Submit Buttons

- If you are going to retrieve form values with the submit button, be sure to prevent the default action!

```
let submitButton = document.getElementById('submitBtn');

submitButton.addEventListener("click", function(e) {
    e.preventDefault();

    let name = document.getElementById('firstName').value;
    console.log(name);
})
```



Questions ?