WESTCLIFF
UNIVERSITY
Educate. Inspire. Empower.

Week 4: JavaScript Essentials

Agenda

- Document Object Model (DOM)
- JavaScript DOM events and types of
- JavaScript DOM event handling
- Test codes using the browser console
- In-Class Demo
- Homework

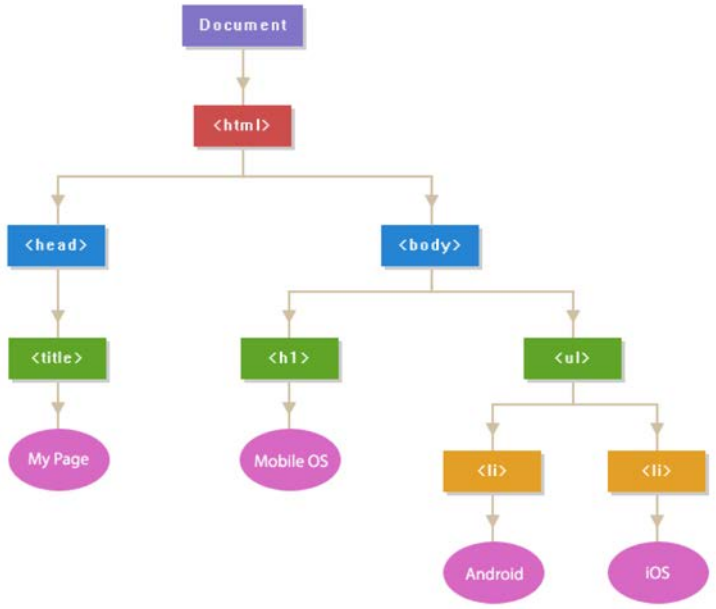
Document Object Model (DOM)

Document Object Model (DOM)

- When a web page is loaded, the browser creates a Document Object Model of the page. The HTML DOM model is constructed as a tree of objects:

Example

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title>My Page</title>
5  </head>
6  <body>
7      <h1>Mobile OS</h1>
8      <ul>
9          <li>Android</li>
10         <li>iOS</li>
11     </ul>
12 </body>
13 </html>
```



Document Object Model (DOM)

- With the object model, JavaScript can exercise its power to create a dynamic HTML. It can:
  - change all the HTML elements in the page
  - change all the HTML attributes in the page
  - change all the CSS styles in the page
  - remove existing HTML elements and attributes
  - add new HTML elements and attributes
  - react to all existing HTML events in the page
  - can create new HTML events in the page

Document Object Model (DOM)

- DOM is a W3C (World Wide Web Consortium) standard. It defines a standard for accessing documents' structure, contents, and styles.
- What is the HTML DOM?
  - HTML DOM is a standard object model and programming interface for HTML. It defines:
    - The HTML elements as *objects*
    - The *properties* of all HTML elements
    - The *methods* to access all HTML elements
    - The *events* for all HTML elements
  - In other words: HTML DOM is a standard for how to *get*, *change*, *add*, or *delete* HTML elements.

Document Object Model (DOM)

- As mentioned earlier, HTML DOM can be accessed with JavaScript. In the DOM, all HTML elements are defined as *objects*.
- The programming interface is the *properties* and *methods* of each object.
  - A *property* is a value that you can get or set (like changing the content of an HTML element).
  - A *method* is an action you can do (like add or deleting an HTML element).`

Examples of *Properties*:

- innerHTML
- attributes
- style
- childNodes

Examples of *Methods*:

- getElementById()
- getElementsByTagName()
- appendChild()
- removeChild()

Document Object Model (DOM)

- How to use Properties: (x = HTML Element)
  - x.innerHTML = ? - change the inner text value of x
  - x.attributes - change the attribute value of x
  - x.style - change the style of x
  - x.childNodes - change the child nodes of x

- Example:

```
<h2>HTML DOM Properties</h2>
<p>This demo illustrates how to insert content into an empty
element on the page using the innerHTML property.</p>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = "Hello World!";
</script>
```

View on Browser:

HTML DOM Properties

This demo illustrates how to insert content into an empty element on the page using the innerHTML property.

Hello World!

Document Object Model (DOM)

- How to use Methods:
  - x.getElementById(*id*) - get the element with a specified id
  - x.getElementsByTagName(*name*) - get all elements with a specified tag name
  - x.appendChild(*node*) - insert a child node to x
  - x.removeChild(*node*) - remove a child node from x
- Example:

```
<h2>HTML DOM Properties</h2>
<p>This demo illustrates how to select an element on the
page using the getElementById() method.</p>

<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = "Hello World!";
</script>
```

View on Browser:

HTML DOM Properties

This demo illustrates how to select an element on the page using the getElementById() method.

Hello World!

Events

- What are the events?
  - Events are things that will occur. Going to a birthday party is an event. Watching a YouTube video is an event. Events are therefore actions.
  - With events, there are also expected reactions. Going to a birthday party reaction => bringing a birthday gift. Watching a YouTube video reaction => laughing hard.
  - In JavaScript, those reactions are also refer to as event handling - in other words how one reacts to that action.
  - On an application like a website, common user interactions with the site are considered as events. In fact more appropriately JavaScript events. After all, JavaScript makes those interactions possible – recall that JavaScript is a behavioral language.

HTML Events

- Examples:
  - When a user *clicks* the mouse
  - When a web page has *loaded*
  - When an image has been *loaded*
  - When the *mouse moves over* an element
  - When an input field is *changed*
  - When an HTML form is *submitted*
  - When a user *strokes a key*

HTML DOM Events

- Types of events:
  - onmouseover
  - onmouseout
  - onmouseup
  - onmousedown
  - onclick
  - onload
  - onfocus
  - onchange
  - onsubmit

Reacting to Events

- When an event occurs, there's a reaction - JavaScript therefore can be executed when an event occurs, like when a user clicks on an HTML element.
- To execute code when a user clicks on an element, add JavaScript code to an HTML event attribute, like this: onclick=*JavaScript*
- A simple example:the user clicks on the existing text and is replaced with a new text.

```
<!DOCTYPE html>
<html>
<body>

<h1 onclick="this.innerHTML='Hello!'">Click This Text!</h1>

</body>
</html>
```

View on Browser (before click):

Click This Text!

View on Browser (after click):

Hello!

Reacting to Events & Handling Events

- The previous example illustrates how changes affect itself ie. the same element when clicked.
- But it can also affect other elements. Events are *handled* if changes are to affect another element. They are called *event handlers*.
- Event handlers are handled by JavaScript functions.
- The basic syntax:

```
<tagName onclick="functionName()">Some Text</tagName>

<script>
function functionName() {
  some statement
}
</script>
```

In this case, the event *onclick* calls the handler function *functionName*.

Reacting to Events & Handling Events

- Examples of Event Handling:

```
<h2 onclick="changeText()">Click This Text!</h2>
<p id="demo"></p>

<script>
function changeText() {
  document.getElementById("demo").innerHTML = "Hello!";
}
</script>
```

View on Browser:

Click This Text!

Hello!

```
<p>Click on the button to display the current date.</p>
<button onclick="displayDate()">Get Date &amp;amp; Time</button>

<p id="demo"></p>

<script>
function displayDate() {
  document.getElementById("demo").innerHTML = Date();
}
</script>
```

View on Browser:

Click on the button to display the current date.

Get Date & Time

Wed Nov 25 2020 15:55:23 GMT-0600 (CST)

Reacting to Events & Handling Events

- Examples of Event Handling:
  - You can assign an event to the DOM ie. without putting the event directly on the DOM element. Let's use the get date example:

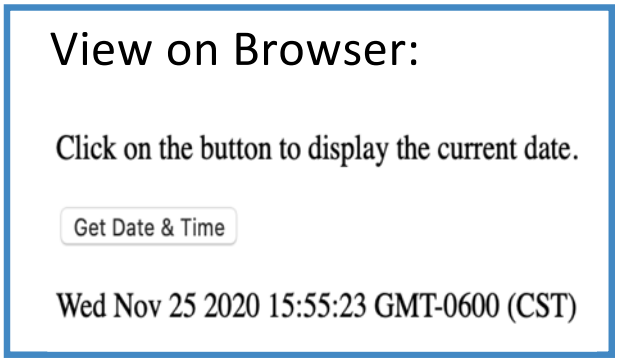
```
<p>Click on the button to display the current date.</p>
<button id="myBtn">Get Date &amp;amp; Time</button>

<p id="demo"></p>

<script>
document.getElementById("myBtn").onclick = displayDate;

function displayDate() {
  document.getElementById("demo").innerHTML = Date();
}
</script>
```

View on Browser:



Reacting to Events & Handling Events

- In this next example, we will look at non-user events – in this case, a browser triggered event. Events such as *onload* is mainly used for browser trigger – in other words, an event occurs when the webpage loads on the browser. The handling will be similar to those we saw in previous examples.

```
<body onload="insertText()">

<p>This example shows texts inserted in an empty
element when the page loads on the browser.</p>

<p id="demo"></p>

<script>
function insertText() {
  document.getElementById("demo").innerHTML = "Hello!";
}
</script>
```

View on Browser:

This example shows texts inserted in an empty element when the page loads on the browser.

Hello!

Reacting to Events & Handling Events

- Back to user events – in this case, a form element event. Events such as *onchange* is mainly used for form input trigger – for example, an event occurs when the user tab away after entering text in a text field. The handling will be similar to those we saw in previous examples.

```
<p>This example shows an event triggered when user tab
away after entering text in a text field.</p>

Enter your name: <input type="text" id="fname"
onchange="insertText()">

<p id="demo"></p>

<script>
function insertText() {
  document.getElementById("demo").innerHTML = "Hello!";
}
</script>
```

View on Browser:

This example shows an event triggered when user tab away after entering text in a text field.

Enter your name: Rich

Hello!

Reacting to Events & Handling Events

- These next properties are not about events and events handling but are commonly used in conjunction with event handling.
- We have seen in previous examples how information can be inserted or changed in a HTML element called by an event.
- What if we want to obtain or capture information from form elements, such as an input text field entered by users? These are common practices in a real-world application.
- The following are two form text properties we can use to do this:
  - value - capture/obtain information entered by user
  - length - find out a number of characters entered in the input element

Reacting to Events & Handling Events

- Example 1 of the value property by referencing the index o anf element via the parent id:

```
<p>This example illustrates how to capture text entered in a
text box and then display on the page.</p>

<form id="myForm" action="">
  First name: <input type="text" id="fname" name="fname"><br>
  Last name: <input type="text" id="lname" name="lname"><br>
</form>

<button onclick="myFunction()">Click to Display</button>

<p id="demo"></p>

<script>
function myFunction() {
  var x = document.getElementById("myForm").elements[0].value;
  document.getElementById("demo").innerHTML = "Hello " + x;
}
</script>
```

View on Browser:

This example illustrates how to capture text entered in a text box and then display on the page.

First name: John

Last name: Doe

Click to Display

Hello John

Reacting to Events & Handling Events

- Example 2 of the value property by referencing the id of the text box:

```
<p>This example illustrates how to capture text entered in a
text box and then display on the page.</p>

<form id="myForm" action="">
  First name: <input type="text" id="fname" name="fname"><br>
  Last name: <input type="text" id="lname" name="lname"><br>
</form>

<button onclick="myFunction()">Click to Display</button>

<p id="demo"></p>

<script>
function myFunction() {
  var x = document.getElementById("fname").value;
  document.getElementById("demo").innerHTML = "Hello " + x;
}
</script>
```

View on Browser:

This example illustrates how to capture text entered in a text box and then display on the page.

First name: Jane

Last name: Doe

Click to Display

Hello Jane

Reacting to Events & Handling Events

- Example of using the length property to display information entered in each text box:

```
<p>This example illustrates capturing texts entered in text
boxes and then display them collectively on the page.</p>

<form id="myForm" action="">
  First name: <input type="text" id="fname" name="fname"><br>
  Last name: <input type="text" id="lname" name="lname"><br>
  Age: <input type="text" id="age" name="age"><br>
  Email: <input type="email" id="email" name="email"><br>
</form>
<button onclick="myFunction()">Click to Display</button>
<p id="demo"></p>

<script>
function myFunction() {
  var ref = document.getElementById("myForm");
  var txt = "";
  for (var i=0; i<ref.length; i++) {
    txt = txt + ref.elements[i].value + "<br>";
  }
  document.getElementById("demo").innerHTML = txt;
}
</script>
```

View on Browser:

This example illustrates capturing texts entered in text boxes and then display them collectively on the page.

First name: Jane
Last name: Doe
Age: 33
Email: jd@email.com
Click to Display

Jane
Doe
33
jd@email.com

Reacting to Events & Handling Events

- You can also use HTML DOM method to change the style of contents in a html element. To do this, use the style object on the method. Here's the syntax:

  document.getElementById(*idname*).style.*property* = *new style*

- To use the DOM style object, you must specify which style property you want to style. Remember: font size, color or background color? The property names are a little different from its CSS counterpart. Here are some examples:

  backgroundColor, fontSize, listStyleType, marginLeft, textAlign

- For full list of them, visit:

  https://www.w3schools.com/jsref/dom_obj_style.asp

Reacting to Events & Handling Events

- Let's start by looking at a simple example without using events.

```
<h1 id="heading1">Hello World!</h1>

<p>Demo on changing the style of a HTML element.</p>

<script>
document.getElementById("heading1").style.color = "#ccef48";
document.getElementById("heading1").style.fontFamily = "Impact";
document.getElementById("heading1").style.fontSize = "2em";
</script>
```

View on Browser:

Hello World!

Demo on changing the style of a HTML element.

Reacting to Events & Handling Events

- This next example uses an event and a handler.

```
<h1 id="heading1">Hello World!</h1>

<p>Demo on changing the style of a HTML element.</p>

<button onClick="changeStyle()">Change</button>

<script>
document.getElementById("heading1").style.color = "#ccef48";
document.getElementById("heading1").style.fontFamily = "Impact";
document.getElementById("heading1").style.fontSize = "2em";

function changeStyle() {
    document.getElementById("heading1").style.letterSpacing = "0.2em";
}
</script>
```

View on Browser:

Questions?

Resources

https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Building_blocks/Events

https://www.w3schools.com/js/js_events.asp

https://www.w3.org/TR/WD-DOM/introduction.html

https://www.tutorialspoint.com/javascript/javascript_html_dom.htm

https://www.w3schools.com/jsref/prop_html_style.asp

http://w3schools.sinsixx.com/htmldom/dom_obj_style.asp.htm

https://www.w3schools.com/js/js_htmldom_eventlistener.asp