ITMD 465/565
Rich Internet Applications

Lecture 3

Fall 2019 – Sept 4, 2019

Tonight's Agenda

- Continue JavaScript I
- JavaScript DOM
- JavaScript Event Handling
- JSON Data Format

JavaScript

JavaScript DOM

- Document Object Model (DOM)
- Object representation of a HTML or XML Document
- All elements are represented by objects
- DOM is an API that can be used in many languages
- JavaScript uses DOM scripting to modify the elements on a page
- DOM is a collection of nodes in a tree
- Also provides standard methods to traverse the DOM, access elements and modify elements
- https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction

JavaScript DOM

- Accessing the DOM elements
- Use methods of the document object
- Most common by id
  - `var a = document.getElementById("elementid");`
- Can also access by class, tag, selector
- Use the `object.getAttribute("src");` method to get a attribute's value from an object. Also a setAttribute to set or change one.
- Set of methods to manipulate DOM objects.
- https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model

DOM Selection

- **By Id** – always will return one DOM element since id needs to be unique on the page.
  `var element = document.getElementById('someId');`
  https://developer.mozilla.org/en-US/docs/Web/API/Document/getElementById

- **By Class Name** – returns an HTMLCollection
  `var elements = document.getElementsByClassName('classnames');`
  https://developer.mozilla.org/en-US/docs/Web/API/Document/getElementsByClassName

- **By Tag Name** – returns an HTMLCollection
  `var elements = document.getElementsByTagName('tagname');`
  https://developer.mozilla.org/en-US/docs/Web/API/Document/getElementsByTagName

- **By Selector** – returns one element or a NodeList of element objects, uses CSS style selectors
  `var element = document.querySelector('css selector string');`
  `var elementList = document.querySelectorAll('css selector string');`
  https://developer.mozilla.org/en-US/docs/Web/API/Document/querySelector
  https://developer.mozilla.org/en-US/docs/Web/API/Document/querySelectorAll

DOM Element Manipulation

- DOM elements have a property (innerHTML) that sets or gets the HTML syntax that describes all the element’s children.

```
var content = element.innerHTML;

element.innerHTML = ‘<p>New HTML</p>’;
```

- This property gives you a way quickly get all the HTML children of an element or replace all the contents of an element easily.
- https://developer.mozilla.org/en-US/docs/Web/API/Element/innerHTML

DOM Element Manipulation

- Getting values from form fields.
- Most basic is the input element. This could be of type text, button, checkbox, radio, and others that use the HTML <input> element form control.
  https://developer.mozilla.org/en-US/docs/Web/API/HTMLInputElement
  https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input
  https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input#Form_<input>_types
- Many properties to interact with the control. Very important one that applies to all types of input controls is value. Gets or Sets the value

```
element.value = ’5’;
var result = element.value;
```

- **All values are get or set as a string. You must do type conversions.**

DOM Element Manipulation

- Other popular form controls include
- Select lists
  https://developer.mozilla.org/en-US/docs/Web/API/HTMLSelectElement
  See properties `selectedIndex`, `selectedOptions`, `value`, and others
- Text Areas
  https://developer.mozilla.org/en-US/docs/Web/API/HTMLTextAreaElement
  Value property to get/set value same as input
- Helper methods you should looks into.
- **`Number()`** - Object
  https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Number
- Other built in global functions and objects
  https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects
- Be very careful with the parseInt() function's behavior as demoed in class and the NaN type.

DOM Element Manipulation

- Elements are the basic object everything descends from. Review this link for basic properties and methods that apply to all elements.
- https://developer.mozilla.org/en-US/docs/Web/API/Element
  https://developer.mozilla.org/en-US/docs/Web/API/HTMLElement
- To properly create, move, delete, or modify elements you need to use properties or methods on the elements.
- Changing Inline CSS Styles
  ```
  document.getElementById('anId').style.backgroundColor = '#000000';
  ```
  https://developer.mozilla.org/en-US/docs/Web/API/HTMLElement/style
  - CSS properties on style object usually use a camel case version of the name. Look it up if you can not figure it out.

DOM Element Manipulation

- Methods used to create and insert an element
- document.createElement()
  https://developer.mozilla.org/en-US/docs/Web/API/Document/createElement
- document.createTextNode()
  https://developer.mozilla.org/en-US/docs/Web/API/Document/createTextNode
- node.appendChild()
  https://developer.mozilla.org/en-US/docs/Web/API/Node/appendChild
- node.insertBefore()
  https://developer.mozilla.org/en-US/docs/Web/API/Node/insertBefore
- node.removeChild()
  https://developer.mozilla.org/en-US/docs/Web/API/Node/removeChild
- node.replaceChild()
  https://developer.mozilla.org/en-US/docs/Web/API/Node/replaceChild

DOM Element Manipulation

- For example to insert a new h3 in the body of an HTML page

```
var head3 = document.createElement('h3');

var headtext = document.createTextNode('This is my H3 headline');

head3.appendChild(headtext);

var b = document.getElementsByTagName('body');

b[0].appendChild(head3);
```

JavaScript Event Handling

- Three Methods – 3rd method is the most preferred way

1. As attribute on HTML element
2. As a method attached to a DOM object
3. Using the add event handler method of a object

JavaScript Event Handling

- As attribute on HTML element
- Not suggested, mixes JavaScript and HTML structure in the HTML markup.
- Uses the attribute that pertains to the particular event. Usually in the form of on + something. Click event is onclick.
- JavaScript code is embedded in the event attribute.

```
<div onclick=“alert(‘I was clicked’)”>click me</div>
```

JavaScript Event Handling

- As a method attached to a DOM object
- Not suggested, while it separates the event handler logic from the HTML markup you still have limitations.
- Can only apply one event handler to an element using this method.
- JavaScript function is assigned to the event name property on the element. Click is onclick.
- https://developer.mozilla.org/en-US/docs/Web/Guide/Events/Event_handlers

```
<div id="aButton">Click Me</div>

document.getElementById('aButton').onclick =  function(){
    //code in this block
};
```

JavaScript Event Handling

- Using the add event listener method of a object
- This is the preferred modern standards compliant method. This is what I want to see unless you have good reason to use the other two and can explain why.
- This allows you to bind multiple handlers on the same element to listen for the same event.
- Event types are without the word "on" before them. Click is click. https://developer.mozilla.org/en-US/docs/Web/Events
- **`element.addEventListener("click", myFunction, false);`**
- Make sure you pass the function object, not execute the function. Notice no parenthesis. Or declare an anonymous inline function.
- https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener

JavaScript

- JavaScript & DOM Reference
- https://developer.mozilla.org/en/docs/JavaScript
- https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction
- https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model
- http://reference.sitepoint.com/javascript/domcore
- https://developer.mozilla.org/en-US/docs/Web/API/element

Dom References

- https://developer.mozilla.org/en-US/docs/Web/API/Node
- https://developer.mozilla.org/en-US/docs/Web/API/Element
- https://developer.mozilla.org/en-US/docs/Web/API/Document
- https://developer.mozilla.org/en-US/docs/Web/API/Node/removeChild
- https://developer.mozilla.org/en-US/docs/Web/API/Node/appendChild
- https://developer.mozilla.org/en-US/docs/Web/API/Document/createElement
- https://developer.mozilla.org/en-US/docs/Web/API/Document/createTextNode
- https://developer.mozilla.org/en-US/docs/Web/API/Document/getElementById
- https://developer.mozilla.org/en-US/docs/Web/API/Window/alert
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Number
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/parseInt
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/NaN

JSON

JSON

- A format to transfer data
- Maps nicely to JavaScript Objects

```
var obj = {
boo: “hello”,
foo: “goodbye”
}
```

- Translates to this JSON

```
{
“boo”: “hello”,
“foo”: “goodbye”
}
```

JSON

- Key value pairs just like JavaScript Objects
- Keys must be strings and quoted (double quotes)
- You may get back objects or arrays typically, pay attention to what your API provides
- Convert JSON text to JavaScript objects using the JSON.parse() method

```
var objects = JSON.parse(JSON_TEXT);
```

- Convert JavaScript objects to JSON text with the JSON.stringify() method

```
var string = JSON.stringify(JS_Objects);
```

Assignments

Reading/Assignments

- If the DOM stuff is not clear, read links here: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction
- Read Chapter 4, 13, 14 in the Eloquent JavaScript Book
- Quiz will be assigned in the next couple days as discussed at the end of class.