ITMD-361
CLASS 11
NOVEMBER 07, 2017

TONIGHT'S AGENDA

- **JavaScript Loops**
- **JavaScript Event Handling**
- **JavaScript DOM Scripting**
- **Google Maps API Introduction**

REVIEW: BASICS

Embedded Scripts

- **Use script tags: <script> JS Here </script>**

External Scripts

- **Use script tag with src attribute:**

  <script src="myscript.js"></script>

Inside <script> is just code

```
var bar = 5;
var foo = "five";
var foo = [5, "five", "5"];
```

REVIEW: COMPARISON OPERATORS

==	Is equal to
!=	Is not equal to
===	Is identical to (equal to and of the same data type)
!==	Is not identical to
>	Is greater than
>=	Is greater than or equal to
<	Is less than
<=	Is less than or equal to

REVIEW: MATHEMATICAL OPERATORS

Mathematical operators are used to perform math on numeric objects

- **Addition + (plus operator is also used to concatenate strings)**
- **Subtraction -**
- **Multiplication ***
- **Division /**
- **Modulus (division remainder) %**
- **Increment ++**
- **Decrement --**
- **Add to self and reassign +=**
  - var car = 5; car += 2; car is now 7

REVIEW: CUSTOM FUNCTIONS

```
function name(type) {
code
}

function foo() {
alert("Our function just ran!");
}

foo(); //actually funs function
```

REVIEW: IF/ELSE

- Conditional statements
  - if statements
  - else statements
  - else if statements

```
if ( condition ) {
    run this block
} else if (condition) {
    run this block
} else {
}
```

REVIEW: NATIVE FUNCTIONS

There are hundreds of predefined functions built into JavaScript, including:

`alert()`, `confirm()`, and `prompt()`

These functions trigger browser-level dialog boxes.

`Date()`

Returns the current date and time.

`parseInt("123")`

This function will, among other things, take a string data type containing numbers and turn it into a number data type. The string is passed to the function as an argument.

`setTimeout(`*`functionName`*`, 5000)`

Will execute a function after a delay. The function is specified in the first argument, and the delay is specified in milliseconds in the second (in the example, 5,000 milliseconds equals 5 seconds).

JAVASCRIPT LOOPS

- Loops
  - **for** {//loops through a block a specific # of times}
  - **while** {//loops through a block while condition true}
  - **do** { }
    while {//loops through block once then repeats as long as a condition is true}
  - **for** { }
    in {//loops through objects in an array or properties of an object, be careful with this one can be error prone}

- Basic Loop Syntax

while (*condition*) {
code block to be executed }

LOOPS CONTINUED

Example 1

```
for (initialize the variable; test the condition; alter the value)
{      loop code here       }
```

Example 2

```
for ( var i = 0; i <= 2; i++ ) {
alert( i );}
```

Example 3

```
do {
    text += "The number is " + i;
    i++;
}
while (i < 10);
```

OBJECTS

- **Document Object:** (HTML)

- **Browser Object**

Property/method	Description
`event`	Represents the state of an event
`history`	Contains the URLs the user has visited within a browser window
`location`	Gives read/write access to the URI in the address bar
`status`	Sets or returns the text in the status bar of the window
`alert()`	Displays an alert box with a specified message and an OK button
`close()`	Closes the current window
`confirm()`	Displays a dialog box with a specified message and an OK and a Cancel button
`focus()`	Sets focus on the current window

EVENT OBJECTS

Event handler	Event description
onblur	An element loses focus
onchange	The content of a form field changes
onclick	The mouse clicks an object
onerror	An error occurs when the document or an image loads
onfocus	An element gets focus
onkeydown	A key on the keyboard is pressed
onkeypress	A key on the keyboard is pressed or held down
onkeyup	A key on the keyboard is released
onload	A page or an image is finished loading
onmousedown	A mouse button is pressed
onmousemove	The mouse is moved
onmouseout	The mouse is moved off an element
onmouseover	The mouse is moved over an element
onmouseup	A mouse button is released
onsubmit	The submit button is clicked in a form

APPLYING EVENT HANDLERS

How to use addEventListner in <script>

```
window.addEventListener("click", myFunction);
```

or as anonyms function:

```
window.addEventListener("click", function() {
function goes here
});
```

JAVASCRIPT EVENT HANDLING

Two other bad methods for Even Handling

- As attribute on HTML element:
  - <body onclick="myFunction();">
  - Only single binds
- As a method attached to a DOM object:
  - window.onclick = myFunction;
  - Only single binds; second bind will override first

EVENT HANDLING EXAMPLE

Check out the MDN Developer Link

Example:

```
function addEventHandler(elem,eventType,handler) {
  if (elem.addEventListener) {
       elem.addEventListener
  (eventType,handler,false);
  } else if (elem.attachEvent) {
       elem.attachEvent ('on' + eventType,handler);
  }}
```

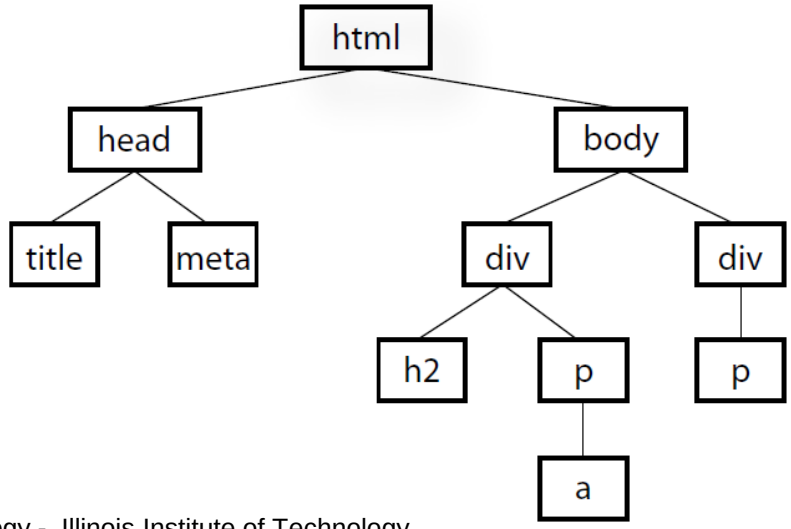
JAVASCRIPT DOM

- Document Object Model (DOM)
- Object representation of a HTML
- 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

The node tree

A simple way to think of the DOM is in terms of the document tree (Figure 20-1). You saw documents diagrammed in this way when you were learning about CSS selectors.

```
<html>
<head>
  <title>Document title</title>
  <meta charset="utf-8">
</head>
<body>
  <div>
    <h2>Subhead</h2>
    <p>Paragraph text with a <a href="foo.html">link</a> here.</p>
  </div>
  <div>
    <p>More text here.</p>
  </div>
</body>
</html>
```



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
- Set of methods to manipulate DOM objects.
- Can use `object.innerHTML` to set the HTML contents of an element.
- Can use `object.value` to get the contents of a form control

ACCESSING DOM NODES

- By element name:

  getElementsByTagName(“*p*”)

- By id attribute value:

  var img = document.getElementById(“*lead_photo*”)

- By class attribute value:

  getElementsByClassName(“*headlines*”)

- By selector:

  querySelectorAll(*“.headlines p*")

  - Similar to jquary

ACCESSING DOM NODES

Accessing an attribute value: **getAttribute()**

HTML =

```
<img src="stratocaster.jpg" alt="electric guitar" id="lead-image">
```

Javascript =

```
var bigImage = document.getElementById("lead-image");
alert( bigImage.getAttribute("src") ); // Alerts "stratocaster.jpg".
```

MANIPULATING DOM NODES

- **setAttribute()**

```
var bigpic = document.getElementById("lead-image");
bigpic.setAttribute(src, “pics/lead_photo2”));
```

- **innerHTML**

```
var introdiv = document.getElementsByClassName("intro");
introdiv.innerHTML = "<p>This is our intro text</p>";
```

- **style**

```
document.getElementById("intro").style.color = "#fff";
```

Or

```
document.getElementById("intro").style.backgroundColor =
"#f58220";
```

JAVASCRIPT DOM

JavaScript & DOM Reference

http://reference.sitepoint.com/javascript/domcore

http://www.javascriptkit.com/domref/elementproperties.shtml

https://developer.mozilla.org/en-US/docs/DOM/element

https://developer.mozilla.org/en/docs/JavaScript

ITMD-361

GOOGLE MAPS API

JAVASCRIPT OBJECT REVIEW

JavaScript Object Literal format

An object literal is a comma separated list of name value pairs wrapped in curly braces.

```
var myObject = {
    stringProp: 'some string',
    numProp: 2,
    booleanProp: false
};
```

Value can be any JavaScript Datatype including a function or other object.

GOOGLE MAPS JAVASCRIPT API

Google Maps JavaScript API: Developer link to maps

- Load the js API file from the googleapis server
- API Key parameter

Once the file is loaded:

1. Map options object: set using a JavaScript Object
2. Container element: is a DOM element
3. Pass the container element and options to the maps constructor.
   - `var map = new google.maps.Map(DOM element, options);`
4. Put all this in a function that runs when the document loads.