# ITMD-361
# WEEK 12
# MARCH 28, 2018

# TONIGHT'S AGENDA

- **JavaScript Introduction**
- **Frameworks**

ITMD-361

# JAVASCRIPT INTRODUCTION

# JAVASCRIPT INTRODUCTION

- **JavaScript is the behavioral layer of web pages.**
  - HTML = structural
  - CSS = presentational
- **JS can target all the elements, attributes, and text a page**
  - Targets the DOM (Document Object Model)
- **Can test for browsers features and capabilities**
- **Modify html and css properties**
  - Show, hide, change, or add event to elements
- **Makes AJAX interactions possible**
- **Historically, browsers differed in support**

# JAVASCRIPT INTRODUCTION

- **Not Related to Java Programming Language**
- **Originally named LiveScript and created by Brendan Eich at Netscape in 1995. Later renamed JavaScript for marketing reasons because of popularity of Java Language at the time.**
- **Standardized by ECMAScript**
  - Current latest stable version is ES8 (2017)
  - ES9 released February 2018: See Proposal
- **Lightweight Object-oriented scripting language**
  - Procedural, object-oriented (prototype-based), and functional style
- **Dynamic Language**
  - Doesn't need to be compiled to machine code
  - Loosely typed - Don't need to declare variable types
  - Read and interpreted on the fly

# JAVASCRIPT INTRODUCTION

**Mozilla JavaScript Guide**

https://developer.mozilla.org/en-US/docs/Web/JavaScript

**Node.js for Command Line JavaScript Intro**

http://javascript.cs.lmu.edu/notes/commandlinejs/

**JavaScript and Basic Programming Introduction Reading**

http://eloquentjavascript.net/

# JAVASCRIPT INTRODUCTION

**Embedded Scripts and External Scripts (How to Guide)**

1. **Embedded Scripts**
   - Use script tags <script> JS Here </script>
2. **External Scripts**
   - Use script tag with src attribute <script src="myscript.js"></script>
   - Script tag must be empty inside
3. **Can be placed anywhere on the page**
   - Most common: in <head> or right before </body>
4. **Execution attribution: async vs defer**
   1. Async executes as page parsing
   2. Defer executes script when page finishes parsing
   3. Neither (default), executes immediately then parses page

# JAVASCRIPT INTRODUCTION

- JavaScript is **case-sensitive:** “foo” not equal “Foo”
- JS statements **should** end with a semicolon;
  - “should” because parsers will forgive you.
- Contains reserved words you can not use. Search for a list of JavaScript reserved words for details.
  - https://developer.mozilla.org/en-US/docs/JavaScript/Reference/Reserved_Words
- JS comments can be single or multi line
  - Single Line – two slashes  // This is a comment
  - Multi Line – similar to css  /* This is a comment */

ITMD-361

# LANGUAGE FEATURES AND SYNTAX

# JAVASCRIPT LANGUAGE

Comprised mainly of:

- Variables
- Operators
- Statements
- Blocks
- Functions
- Comparison
- Conditional Statements
- Looping
- Objects
- Events

# JAVASCRIPT VARIABLES

**Variables hold values or objects**

- **Declare var with keyword;** `var foo;`
- **"Set" value with = ;** `foo = 5;`

**Variables Properties:**

- **Names are case sensitive and must begin with a letter or the underscore**
- **Can be a set of very basic data types**
- **No special characters in name (! . , / \ + * =)**
- **Has functional scope – not block scope**
- **Variable declared in a function without var keyword, it's global**

**Array – grouping/list of objects**

- **Arrays are defined with -** `new Array()` **or** `[ ]`
- **Zero indexed so first element is -** `arrayname[0]`

# VARIABLE DATA TYPES

1. **Undefined**
   - No value declared: var foo;
2. **“null”**
   - Officially declares no value: var foo = null;
3. **Numbers**
   - Assigns number to variable: var foo = 5;
4. **Boolean**
   - Assigns True or Fales: var foo = true;
5. **String**
   - Uses “ ” to declare line of text: var joe = “five”

     var foo = “5”

     alert (foo + foo) //gives 55

# JAVASCRIPT MATHEMATICAL OPERATORS

**Mathematical operators are used to perform math on numeric objects (see page 469)**

- **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

# JAVASCRIPT STATEMENTS

- **Statements are execution commands to the browser**
  - Execute in order of presentation
  - Browsers have built in statements/functions (defaults)
  - Basics are: alert(); confirm(); prompt();

- **Statements can be grouped together in blocks with the curly brackets { }**
  - Used for defining functions or using conditionals
  - JavaScript does not use block scope like most programming languages. It has function scope.

# JAVASCRIPT FUNCTIONS

**Functions: named blocks of code that can be called and executed by events or other code**

```
function funcname (var1, var2, …) {
code block (may make use of parameters)
}
```

**The return statement will stop executing the function and return the value**

```
function addnum(n1, n2) {
return n1 + n2;
}
```

**JavaScript has many built in functions**

# JAVASCRIPT COMPARISON OPERATORS

- **Comparisons are used to compare the value of two objects and return true or false**
  - == Is equal to
  - != Is not equal to
  - === Is identical to (equal to and 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
- **alert( 5 > 1 );  // Will alert “true”**

# JAVASCRIPT OBJECTS

- **Objects: All JavaScript items are objects**
  - Including functions
  - But excluding core data types
- **No true class system in JavaScript. Uses Prototypes instead.**
  - Examples: The browser is the `window` object the html page is the `document` object
- **Objects are composed of properties and methods**
  - Properties are basically variables
  - Methods are basically functions
- **Access and objects property –** `obj.propertyName`
  - **or** `obj["propertyName"]`
- **Execute an object method –** `obj.methodName()`

# JAVASCRIPT OBJECTS

**JavaScript Objects Creation and Use**

- **Created by a function with new keyword**
  - `var obj = new Object();`
- **Created with an object literal**
  - `var obj = {};`
  - `var obj = { key: value, key2: value2 };`
    - Key needs to be a string with no spaces
    - `var obj = { color: "red", quantity: 5, instock: true };`
- **Access or set properties with dot notation**
  - `obj.color = "blue";` sets color of obj to blue
  - `obj.quantity;` would be equal to 5
  - Can also set or execute methods this way
  - You can also access properties with the array like syntax of `obj["color"]`
    - Useful when you need the property value to come from another variable

# JAVASCRIPT OBJECTS

**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.

# JAVASCRIPT CONDITIONAL STATEMENTS

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

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

# 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.
- For Loop Syntax

```
for (initialize the variable; test the condition;
alter the value;){

    code to loop here

}
```

# JAVASCRIPT EVENT HANDLING

**Event Handling**

Three Methods

- As attribute on HTML element
- As a method attached to a DOM object
- Using the add event handler method of a object
  - object.addEventListener("click", myFunction);
  - object.attachEvent('onclick', modifyText);

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

```
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 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

# JAVASCRIPT DOM

- Accessing the DOM elements
- 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.

# 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

# 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.

# FRAMEWORKS

# BOOTSTRAP CSS FRAMEWORK

- CSS Framework with built in classes to start from
- Use grid systems to build responsive designs
- **Bootstrap:**
  - You can download other bootstrap themes that modify these styles or override them yourself
  - Created in house at Twitter
  - getbootstrap.com
- **Foundation:**
  - Open sourced
  - Created by ZURB product design
  - My silicon valley peeps say it's overtaken bootstrap
  - foundation.zurb.com

# FOUNDATION: LINK

- **Based on grid system**
  - 940 pixel wide, flexible grid layout
  - Fully responsive without additional coding!
- **Includes JS plug ins**
  - Uses Jquery
- **Utilizes custom <class>**
  - Use their <class> and get instant css code
- **Compared to Bootstrap:**
  - Foundation is more flexible
  - Smaller library