Introduction to JavaScript

CST8285

Review

- So far, we have focused on HTML and CSS
- We have learned:
  - The syntax and structure of HTML and CSS
  - How to display text and images with HTML
  - How to create tables to display data with HTML
  - How to create forms (that do nothing)
  - How to change the look of HTML elements with CSS
  - How to change the layout of HTML elements with CSS

Static vs Dynamic Web Pages

- The content of static web pages does not change, unless the source code is modified.
- Static web pages do not allow for user input/interaction.
- Dynamic web pages have content that changes, without having to edit the source code.
- Most web pages have some dynamic content.
- All the pages we have done so far have been static pages.

Adding functionality to a web page

- Some examples of ways to add functionality to a web page:
  - HTML5 & CSS (beyond the scope of this course)
  - Flash
  - JavaScript
  - PHP

- In this course, we will focus on the last two.

Client Side vs Server Side

- In web programming, you can have client-side programming, or server-side programming
- Client side code runs in the browser.
- Server side code runs on the server
- The client and the server don't know what the other is doing.

Client Side vs Server Side

Client Side	Server Side
Runs in the browser	Runs on the server
Languages: • JavaScript • Angular.js • jquery	Languages: • PHP • PHP frameworks such as (Laravel, CakePHP, and more) • Ruby on Rails • Python • C#
Has direct access to the browser and the page elements.	Does not have access to the browser.
Unreliable (scripts can be disabled, input validation can be manipulated)	Reliable (user has no direct access, theoretically secure)

What is JavaScript?

- JavaScript is a versatile, interpreted client-side* programming language.
- Supported by all major browsers
- Used to add functionality to web pages
  - JavaScript can respond to events
  - JavaScript can access and modify HTML elements
- JavaScript is **NOT JAVA!**

*Although most commonly used for client-side programming, JavaScript can also be used as a server side programming language (ex: Node.js)

Using JavaScript in your web page

- You can put JavaScript in you web page in two ways:
  - Inside a <script> element

```
<script>
  document.write("Hello World");
</script>
```

  - Link to a JavaScript file (.js) using the `<script> src` attribute:

```
<script src="scripts/jquery.js"></script>
```

When is JavaScript used?

- Like all programming languages, it is used to solve a problem.
- Examples of uses for JavaScript:
  - Form Validation
  - Hide/Show elements
  - Dynamically change text/images
  - Mathematical problems
  - Responding to events (clicks, key presses, etc)

The Basics

- What we will cover:
  - Declaring functions
  - Declaring variables
  - Data Types
  - Operators
  - Comparisons
  - Conditions
  - Prompts and Alerts
  - DOM Events

Function Declaration

- A basic function declaration looks like this:

```
function myFunction () {
    //do stuff here
}
```

- A function declaration with parameters looks like this:

```
function myFunction (param1, param2){
    //do stuff here
}
```

- There are two other ways to define a function, but we will focus on the function declaration for now.

Declaring Variables

- Variables are declared with the term `var`
  - Ex 1: `var myName = "Matt";`
  - Ex 2: `var myName;`
- Variables declared **outside** of a function have **global** scope.
- Variables declared **inside** of a function have **local** scope.
- The equals sign (=) is the **assignment operator**
- Variables are dynamically typed – they do not require a data type specification

Data types - String

- Strings can be enclosed in single or double quotes
- Examples:

```
var a = “hello world!”;
var b = ‘hello world!’;
var c = “55”;
```

- You can use quotes in a string so long as they don’t match the container quotes
- Ex:

```
var statement = “He said ‘Hello World!’ to me.”;
```

Data types - String

- Strings can be added together using the + operator
- Examples:

```
var firstName = “Matt”;
var lastName = “LaCasse”;

var fullName = firstName + “ “ + lastName;
```

Data Types - Number

- JavaScript only has one number type
- Can be written with or without a decimal
- Examples:

```
var a = 50;
var a = 50.5;
```

- Very large and very small numbers can be written in scientific notation
- Examples:

```
var a = 19e7; //190000000
var a = 12e-7 //.0000012
```

Data Types - Boolean

- Can be `true` or `false`
- Ex:

```
var isThisANumber = false;
var isThisABoolean = true;
```

Arithmetic Operators

Given that `a = 10`

Operator	Description	Example	Outcome
+	Addition	`b = a + 2`	`b = 12`
-	Subtraction	`b = a - 3`	`b = 7`
*	Multiplication	`b = a * 5`	`b = 50`
/	Division	`b = a / 2`	`b = 5`
%	Modulus	`b = a % 3`	`b = 1`
++	Increment	`b = a++` `b = ++a`	`b=10, a=11` `b=11, a=11`
--	Decrement	`b = a--` `b = --a`	`b=10, a=9` `b=9, a=9`

Note: adding a number contained in a string with a regular number will return a string

Assignment Operators

Given `a = 12`, and `b = 3`

Operator	Sample	Equivalent	Outcome
=	`a = b`	`N/A`	`a = 3`
+=	`a += b`	`a= a + b`	`a = 15`
-=	`a -= b`	`a= a - b`	`a = 9`
*=	`a *= b`	`a= a * b`	`a = 36`
/=	`a /= b`	`a= a / b`	`a = 4`
%=	`a %= b`	`a = a % b`	`a = 0`

Comparisons

Given a = 10

Operator	Description	Statement	Outcome
==	equals	a == 2 a == 10	false true
===	exactly equal (type and value)	a === “10” a === 10	false true
!=	not equal	a != 10 a != 20	false true
!==	not equal (value or type different)	a !== “10” a !== 10	true false
>	greater than	a > 15	false
<	less than	a < 15	true
>=	greater than or equal	a >= 9	false
<=	less than or equal	a <= 9	true

Logical Operators

Given a = 7 and b = 4

Operator	Description	Sample
&&	and	(a > 2 && b < 10) would be true
||	or	(a != 7 || b == 4) would be true
!	not	if !(a > b) would be false

Conditional statements

- JavaScript uses the if, if…..else, if…..else if…..else, and switch statements for conditional actions
- if statement syntax:

```
if(expression){
      //do something
}
```

Conditional statements

- if…..else example:

```
if(expression){
        //do something if the expression is true
} else {
        //do something if the expression is
        false
}
```

Conditional statements

- if…..else if……else example:

```
if (expression1){
      //do something if expression 1 is
      //true
} else if (expression 2){
      //do something if expression 2 is
      //true
} else {
      //do something if neither
      //expression is true
}
```

Conditional Statements

- Switch example:

```
switch(variable)
{
case 1:
      //do something
      break;
case 2:
      //do something
      break;
default:
      //do this if none of the above
            //cases are true
}
```

Conditional Statements - Switch

- The value after `case` is called the case label
- The code found after the case label will run if the variable matches the case label.
- If there is no match between the variable and the case label, the code after `default` will run.
- The `break;` statements tells the switch statement to stop.

Prompt

- A JavaScript prompt displays a message box to the user with a message, and a textbox for them to reply.
- Called by the `prompt` function
- You can assign the result of a prompt to a variable
  - Ex:

```
var name = prompt("What is your name?");
```

- A prompt can have a default value, specified by a second parameter.
  - Ex:

```
var name = prompt("What is your name?",
"noname");
```

Alert

- An alert is similar to a prompt, but it does not accept user input.
  - Ex:

```
alert("Transaction was successful.");
```

DOM Events

- HTML elements have attributes that can run JavaScript when a certain event happens
- Some common events are:
  - onclick
  - onmouseover
  - onload
  - onchange
  - onmouseup
  - onmousedown

DOM Events

- To run a JavaScript function, add the name of the function as the event attribute's value.
  - Ex:

```
<input type="button" name="btnConvertTemp"
id="btnConvertTemp" onclick="convertTemp();"
value="Convert Temperature">
```

  - When the above button is clicked, the `convertTemp()` function will be run.