CSIS 3380
Advanced Web Programming with JavaScript & AJAX

Loop and DOM Manipulation

Week 4

Review

- Data Types
- Object
- Array

Loop

- A for loop repeats until a specified condition evaluates to false

```
for (initializer; exit-condition; increment-expression) {
  // code to run
}
```

- Example:

```
var students = [‘Bill’, ‘Jeff’, ‘Jack’, ‘Peter’];
for (var i = 0; i < students.length; i++) {
   console.log(students[i]);
}
// the for loop will prints out all the names of the students
```

```
var students = ['Bill', 'Jeff', 'Jack', 'Peter'];
var message = "The students' name are "
for (var i = 0; i < students.length; i++) {
   message += students[i] + ', ';
}
console.log(message);
```

Loop

- A while statement executes its statements as long as a specified condition evaluates to true. A while statement looks as follows:

```
while (condition)
    statement
```

- Example:

```
var counter = 100;
while (counter > 0) {
  console.log('counter in while loop = ' +counter);
  counter -= 10;
}
console.log('counter outside while loop = ' +counter);
```

break statement

- Use the **break** statement to terminate a loop or a switch
- Example:

```
for (var i = 0; i < a.length; i++) {
  if (a[i] === theValue) {
    break;
  }
}
```

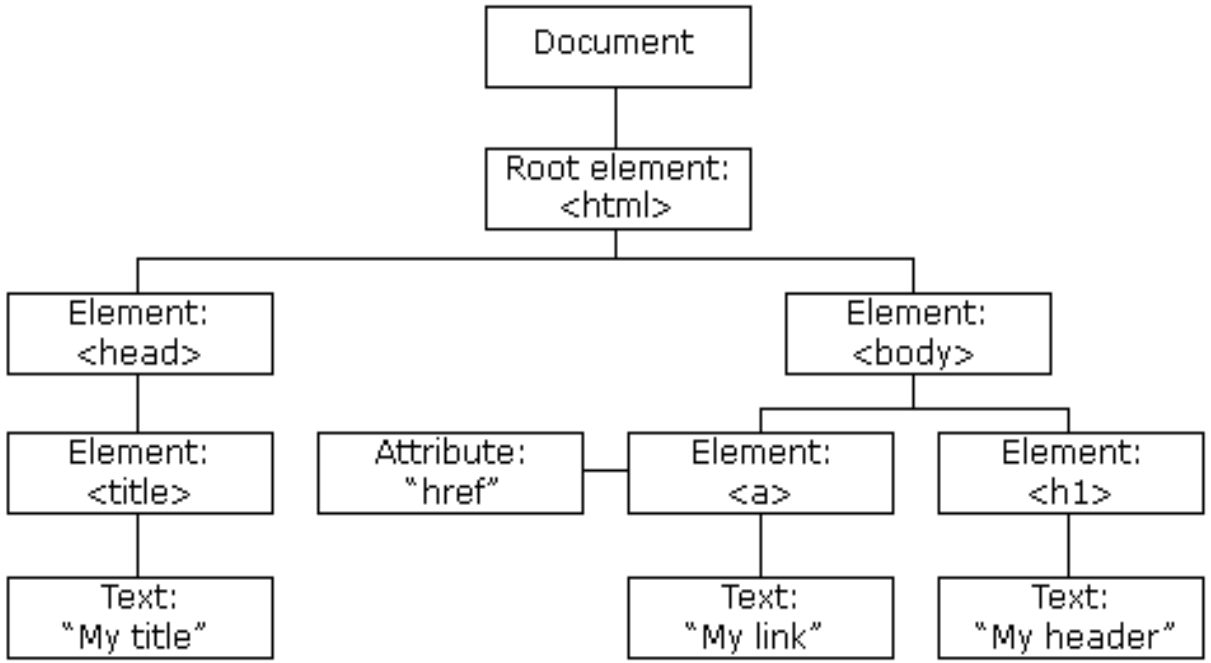
continue statement

- The continue statement terminates the current iteration of the innermost enclosing while or for statement and continues execution of the loop with the next iteration
- Example:

```
var i = 0;
while (i < 5) {
  i++;
  if (i === 3) {
    continue;
  }
  console.log('i = ', i);
}
```

What is the DOM?

- A Web page is a document. This document can be either displayed in the browser window or as the HTML source.
- The Document Object Model (DOM) represents that same document so it can be manipulated.



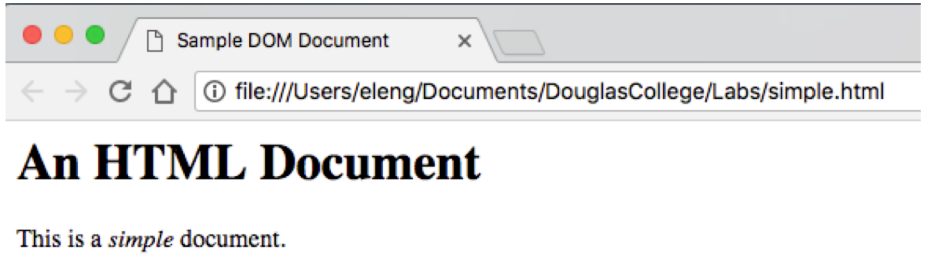
http://www.w3schools.com/js/js_htmldom.asp

DOM

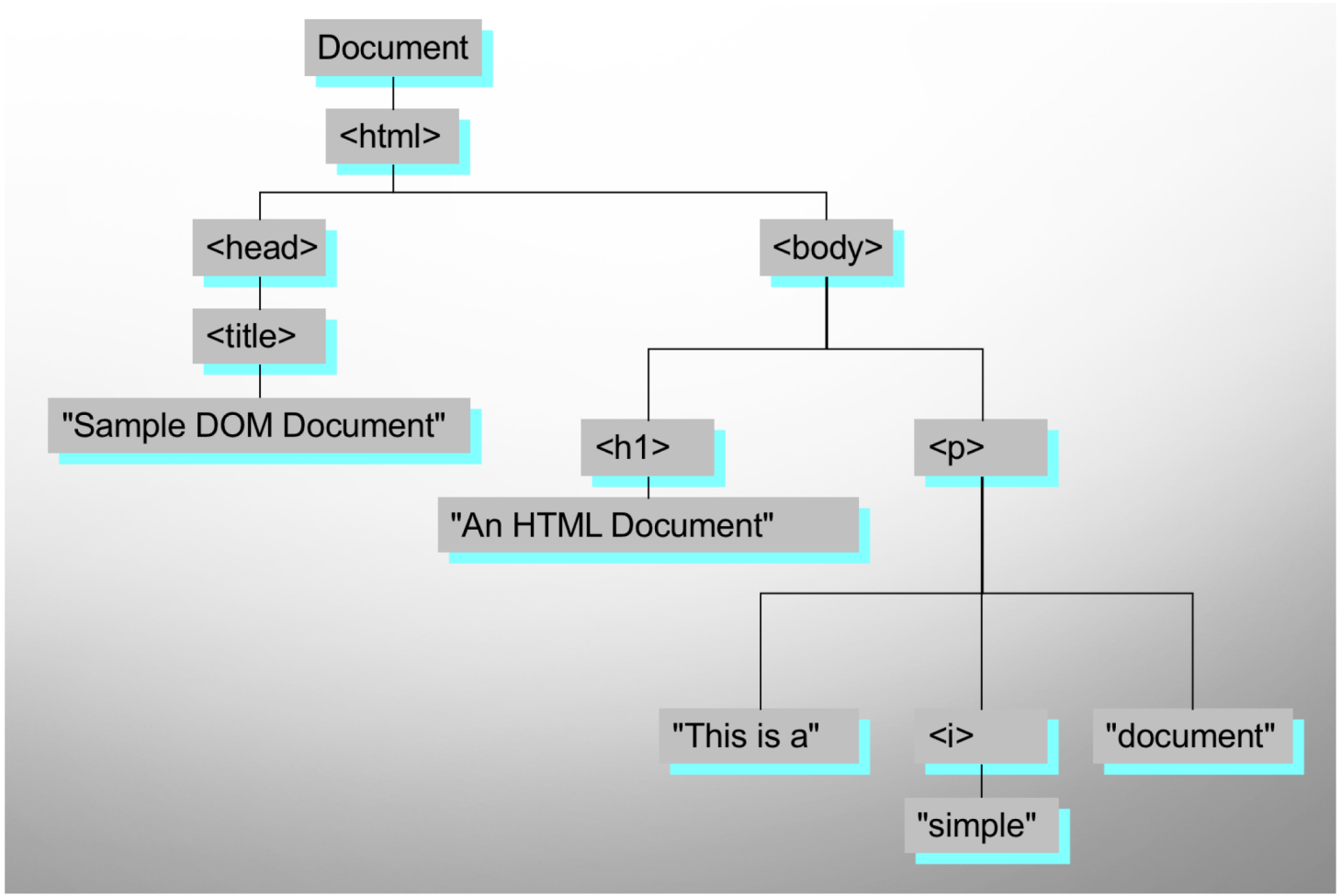
This is what the browser reads

```
<html>
  <head>
    <title>Sample DOM Document</title>
  </head>
  <body>
    <h1>An HTML Document</h1>
    <p>This is a <i>simple</i> document.
  </body>
</html>
```

This is what the browser displays on screen.



DOM



This is a drawing of the model that the browser is working with for the page.

Why is this useful?

- Because we can access the model too!
- The model is made available to javascript running in the browser
- Javascript uses the DOM to access the document and its elements
- We can use javascript to:
  - change the HTML elements or attributes on a page
  - change all the CSS styles in the page
  - delete the HTML elements or attributes on a page
  - add new HTML elements or attributes to a page
  - react to all existing HTML events in the page
  - create new HTML events in the page

DOM

- The DOM 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: **The HTML DOM is a standard for how to get, change, add, or delete HTML elements**

Referencing objects

- Objects can be referenced
  - by their id or name (this is the easiest way, but you need to make sure a name is unique in the hierarchy)
  - by their numerical position in the hierarchy, by walking the array that contains them
  - by their relation to parent, child, or sibling (parentNode, previousSibling, nextSibling, firstChild, lastChild or the childNodes array

Example

```
<div id="mydiv">
This is some simple html to display
</div>
```

- the div is an element with an id of *mydiv*
- It contains a text element, which can be referenced by childNodes[0] (childNode being an array of all childen of a node)
- So the text in the div is not a value of the div, but rather the value of the first (and only) childNode of the div

Manipulating Objects

- It's easiest to reference objects by id
- To do this easily, use getElementById(), which returns the element with the given id
- For example, if you want to find a div with the id of "my_cool_div", use getElementById("my_cool_div")
- Keep in mind that it's the element itself that's returned, not any particular property

innerHTML

- innerHTML is a property of any document element that contains all of the html source and text within that element
- This is not a standard property, but widely supported--it's the old school way to manipulate web pages
- Much easier than building actual dom subtrees, so it's a good place to start
- Very important--innerHTML treats everything as a string, not as DOM objects (that's one reason it's not part of the DOM standard)

Using these together…

- You can reference any named element with getElementById()
- You can read from or write to that element with innerHTML
- For example:

```
getElementById("mydiv").innerHTML ="new text string";
```

A simple DOM example

```
<div id="mydiv">
  <p>This some <i>simple</i> html to display</p>
</div>
<form id="myform">
  <input type="button" value="Alert innerHTML of mydiv"
     onclick="alert(getElementById('mydiv').innerHTML)" />
</form>
```

Other useful lookup methods

- document.getElementById(id)
- document.getElementsByTagName(name)
- document.getElementsByClassName(name)
- document.querySelector(selectors)
  - This *selectors* string must be a valid CSS selector string

Adding and deleting element

- parentNode.removeChild(element)
- parentNode.appendChild(element)
- parentNode.replaceChild(element)

Element attribute

- element.setAttribute():
  - Sets the value of an attribute on the specified element
- element.getAttribute()
  - returns the value of a specified attribute on the element
- element.removeAttribute()
  - removes an attribute from the specified element

Lab

- Write a function **findLargest(arr)** that takes an array of 5 numbers and returns the largest of the 5 numbers
  - Sample inputs: [1, 6, 3, 0, 8]
  - Returns: 8

- Write a function printAsterisks() that will prints out in the console a 5 by 5 asterisks pattern like this

```
*****
*****
*****
*****
*****
```

Lab

- Go through the APIs for Document, Element and Node
  - https://developer.mozilla.org/en-US/docs/Web/API/Document
  - https://developer.mozilla.org/en-US/docs/Web/API/Element
  - https://developer.mozilla.org/en-US/docs/Web/API/Node

Lab

- Download **week4-lab.html** in blackboard. There are 3 images on week4.html
  - Use **document.getElementById** to dynamically retrieve the **height** and **width** of the 3 images
  - Use **innerHTML** to insert these heights and widths into an unordered HTML list at the specified
  - The output should look like this