Information Technologies

Anselm Spoerri PhD (MIT)
SC&I @ Rutgers University
aspoerri@rutgers.edu
anselm.spoerri@gmail.com

Lecture 8 - Overview

JavaScript

- **Objects**
- **Arrays**
- **Scope** of Variable
- **DOM**: add and style page elements dynamically
- **Google Maps API**

Lectures – Week 8 Content

http://comminfo.rutgers.edu/~aspoerri/Teaching/InfoTech/Lectures.html#week8

JavaScript Objects

In JavaScript, almost "everything" is an object.

Objects have **Properties** and **Methods**.

new Object () versus **{ ... }**

Object Literal

Comma-separated list of **name-value pairs** wrapped in **curly braces**

var literalObj = **{** prop1**:** 10**,** prop2**:** 'hello'**}**;

Object literals encapsulate data, enclosing it in tidy package.

Minimizes use of global variables which can cause problems when combining code.

Object Literal Syntax

Colon : separates property name from value.

Comma , separates each name-value pair from the next.

No comma after the last name-value pair.

JavaScript Objects

Objects

Properties (property) **keys** are used to access (property) **values**

var car = {**type**:"Fiat", **model**:"500", **color**:"white"};

objectName.propertyName *or* objectName**["**propertyName**"]**

car.type car["type"]

Methods are stored in properties as **function definitions**

```
var person = {firstName: "John", lastName : "Doe",
        fullName : function() {
            return this.firstName + " " + this.lastName;
        }
};
```

objectName.methodName(); person.fullName();

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

JavaScript – Objects and Arrays

Arrays are always objects … a special kind of object

Difference Between **Arrays** and **Objects**

- **Arrays** use **numbered indexes** items=[3,8] ➔ items[1]
- **Objects** use **named indexes** car={type:"Fiat"} ➔ car["type"]

Accessing Array

for (i = 0; i < **items.length**; i++) { … **items[i]** … }

items.**forEach**(myFunction);
function myFunction (**value**, index, array)

https://www.w3schools.com/js/tryit.asp?filename=tryjs_array_foreach

JavaScript – Scope of Variable

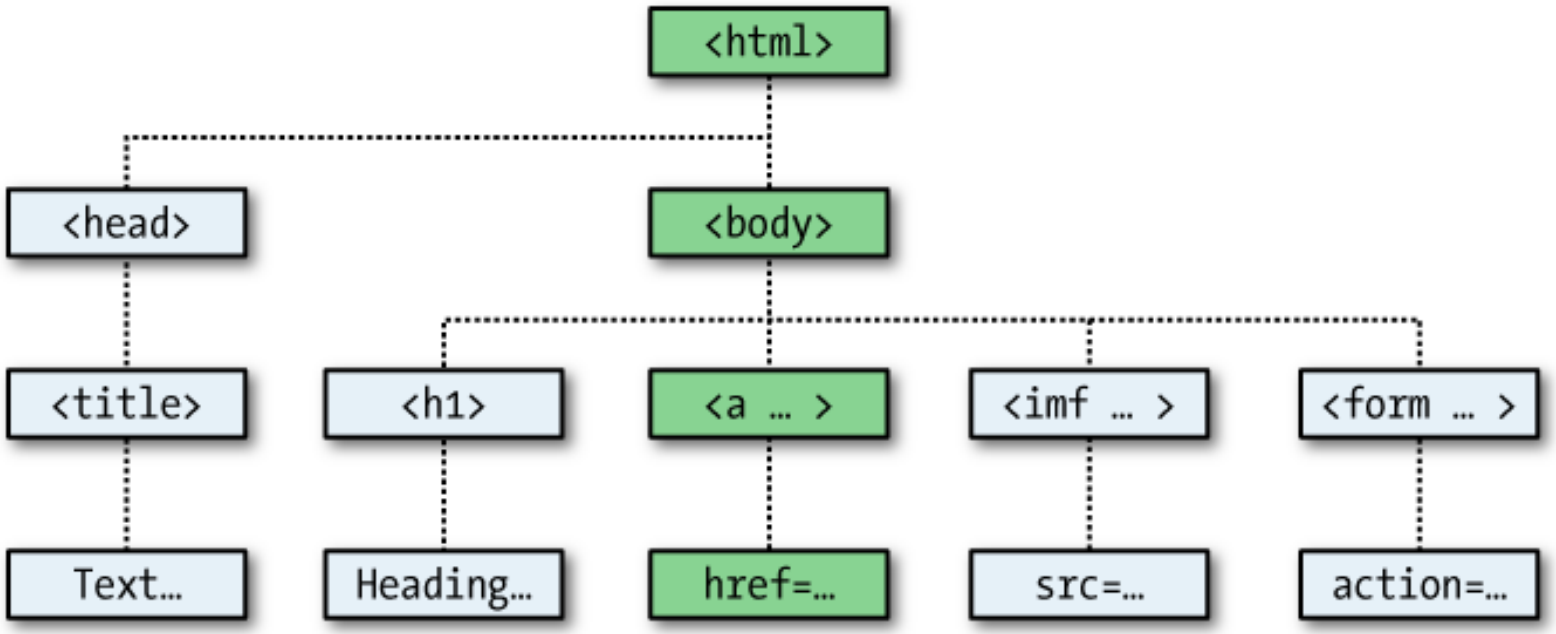
variables http://www.w3schools.com/js/js_variables.asp

functions http://www.w3schools.com/js/js_functions.asp

Scope - http://www.w3schools.com/js/js_scope.asp

- Variable declared **outside function**, becomes **GLOBAL**.
- Global variable has global scope: All scripts and functions on page can access it.
- Variables created **without the keyword var**, are always **global**, even if they are created inside a function.
- Global variables live as long as your application (window / web page) lives.
- Variables declared **within JavaScript function**, become **LOCAL** to the function.
- Local variables have local scope: They can only be accessed within the function.
- Local variables have short lives. They are created when the function is invoked, and deleted when the function is finished.

Document Object Model



JavaScript HTML DOM

DOM Document Object https://www.w3schools.com/jsref/dom_obj_document.asp

document.**createElement**("elementType")

```
var node = document.createElement("P");
```

https://www.w3schools.com/jsref/met_document_createelement.asp

document.**createTextNode**("text")

```
var textnode = document.createTextNode("Hello World.");
```

https://www.w3schools.com/jsref/met_document_createtextnode.asp

DOM Element Object https://www.w3schools.com/jsref/dom_obj_all.asp

node.**appendChild**(nodeToAdd)

```
node.appendChild(textnode);
document.getElementById("myList").appendChild(node);
```

https://www.w3schools.com/jsref/met_node_appendchild.asp

insertBefore(newItem, whichNode)

```
list.insertBefore(newItem, list.childNodes[0]);
```

https://www.w3schools.com/jsref/met_node_insertbefore.asp

JavaScript HTML DOM

DOM Element Object https://www.w3schools.com/jsref/dom_obj_all.asp

node.**setAttribute**

document. getElementById("myDIV").setAttribute("class", "myStyle");

node.**className**

document.getElementById("myDIV").className = "mystyle";

https://www.w3schools.com/jsref/prop_html_classname.asp

DOM Style Object https://www.w3schools.com/jsref/dom_obj_style.asp

Better way for setting of inline CSS since it **does not overwrite** other CSS properties in style attribute

JavaScript – Modifying the DOM – Key Steps

Create **content object** { elementType, elementFloat, elementContent }

Create **content array** [content1, content2, content3, content4]

Create function **addElement_to_DOM (contentObj)** { … }

```
var node = document.createElement(elementCAPS);
var textnode = document.createTextNode(contentObj.elementContent);
node.appendChild(textnode);
var insertionNode = document.getElementById("pageContent");
if ... else  // to assign floating CSS class node.className = "floatLeft";
        insertionNode.insertBefore(node, insertionNode.childNodes[0]);
        or
        insertionNode.appendChild(node);
```

contentArray. forEach(addElement_to_DOM); // matters where it is called

Google Maps API

Get Google Maps API Key

https://developers.google.com/maps/documentation/javascript/ select Web

Create Google Map

https://developers.google.com/maps/documentation/javascript/tutorial

google.maps.Map class

Constructor: **Map**(mapDiv:Element,opts?:MapOptions)

```
var map1 = new google.maps.Map (
    document.getElementById('mapDiv'),
    objectLiteral);
```

Google Maps Reference:

https://developers.google.com/maps/documentation/javascript/reference#Map

Change Center Point need to know Geocode: lat & lng

Create Geocode – http://www.gpsvisualizer.com/geocode
4 Huntington Street, New Brunswick, NJ

https://developers.google.com/maps/documentation/javascript/reference#MapOptions

https://developers.google.com/maps/documentation/javascript/reference#LatLng

new google.maps.LatLng(40.505, -74.453) or {lat: 40.505, lng: -74.453}

Google Maps API

Change Map Type and Zoom Level

Google Maps Reference:
https://developers.google.com/maps/documentation/javascript/reference

MapTypeId | MapOptions Object (specify keys)

Chrome Console: map.getZoom ();

Customize map: draggable: false; | scrollwheel: false;

Add Marker

https://developers.google.com/maps/documentation/javascript/examples/marker-simple

https://developers.google.com/maps/documentation/javascript/reference#Marker

google.maps.Marker class

Constructor: Marker(opts?:MarkerOptions)

```
var marker1 = new google.maps.Marker (objectLiteral);
```