CO456
Web

most materials adapted from ***Moseley (2007)***, Chapter 5 –
supplemented with extracts from Bates (2006) and w3schools.com

Lecture 5

JavaScript ***variables*** & programming ***structures***

Module schedule

Wk.	Lecture/subject area(s)	Practical	Reading (Moseley, 2007)
1	Introduction How the Web works	Internet/Web definitions and HTML report	Ch 1 (The way the Web works)
2	HTML 1 (Introductory - inc. lists and hyperlinks)	HTML	Ch 2 pp 24-36 (HTML)
3	HTML 2 (inc. tables, images and forms)	HTML	Ch 2 pp 36-48 (HTML) Ch 3 (XHTML and frames)
4	CSS 1 (Introduction and core CSS principles)	CSS – introductory styles, embedded styles.	Ch 4 pp 76-96.
5	CSS 2 (Positioning elements).	CSS– using IDs, classes and layout control.	Ch 4 pp 97-103.
6	CSS 3 (Advanced layout & navigation)	CSS – using CSS to produce button-like navigation from HTML list elements. (CW2a to be demonstrated).	Specialised articles.
7	JavaScript 1 (Fundamentals, variables)	JS – foundation constructs.	Ch 5 pp 108-116
8	Guided Learning Week	Consolidate Internet & W3 knowledge and HTML & CSS skills.	Review Ch 1 to Ch 4.
9	JavaScript 2 (Functions, branches, loops).	JS – calling functions.	Ch 5 pp 117-124.
10	JavaScript 3 (Objects and the DOM).	JS – manipulating the DOM.	Ch 6 126-139.
11	JavaScript 4 (Forms and validation). And DHTML	JS– validating user completed forms.	Ch 6 139-145, Ch 7.
12	HTML***5***, CSS***3***, - media, forms, gradients, SVG ('Edge') and other enhancements	Web frameworks taster session 1	See practical sheets for information sources
	Vacation		
13	Advanced HTML5, CSS3 & JS frameworks (e.g. jQuery, jQuery Mobile, Box2DWeb)	Web frameworks taster session 2	See practical sheets for information sources
14	Assignment workshop 1	Assignment workshop 1	N/A
15	Assignment workshop 2	Assignment workshop 2	N/A

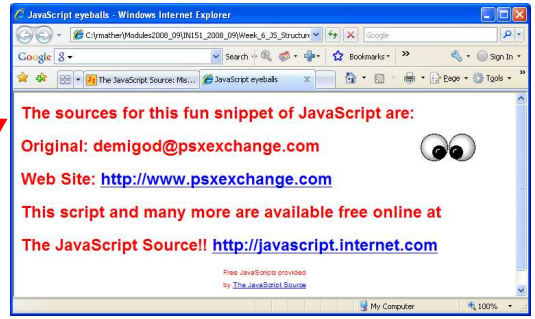
JavaScript – what and why?

- Scripting provides a means for dynamically changing content with *time*, according to *events* and by *user interaction*.
- Client side scripts are embedded in Web pages and interpreted by browsers.
- Two main client side scripting languages JavaScript and VBScript.
- JavaScript was developed by Netscape Communication.
- Jscript is Microsoft's version.
- JavaScript was used *retrospectively* as the ECMA standard (European Computer Manufacturers Association - *not the with the Euro. Carton Makers Assoc.!*).
- Advantages of client side scripting are
  - Reduced burden on server (the client's Web browser processes scripts)
  - Relatively simple for developers to use
- Disadvantages client side scripting are
  - Visible and modifiable code
  - Reliance on users to have scripting/active-content enabled
  - Limited specification for complex processes – e.g. business/financial transactions
  - Local files and databases are not accessible

JavaScript – uses

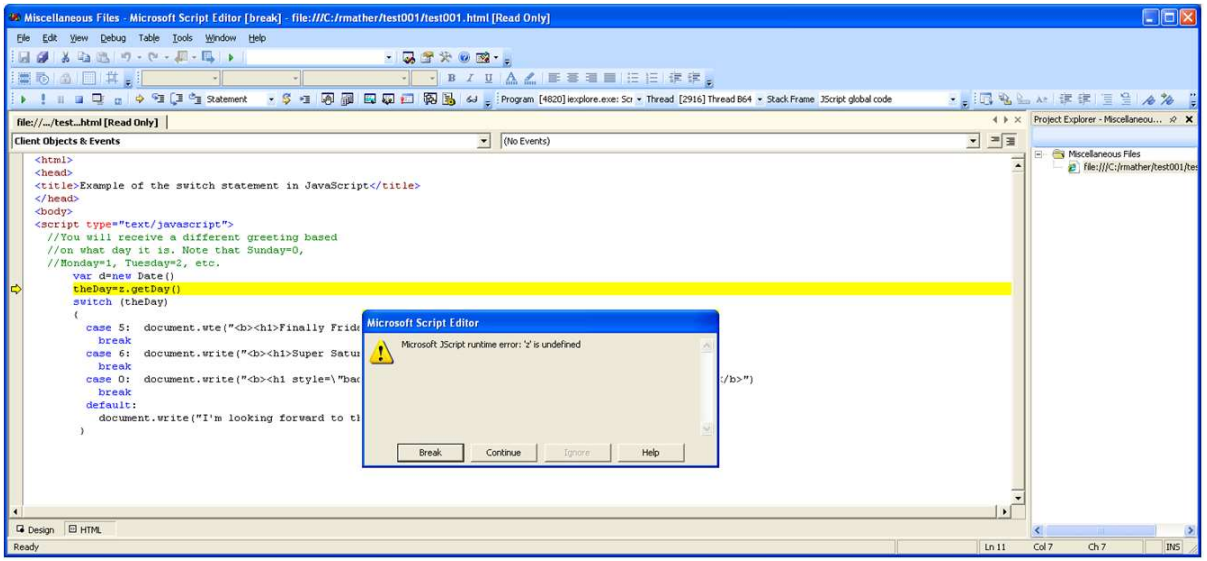
Adapted from http://www.dcs.uwaterloo.ca/~anderson/JavaScript/SimpleUsesOf.html **and** http://www.webmasterworld.com/javascript/3089286.htm

- Dynamic forms that include built-in validation and error checking
- Calculators and calculation areas on pages
- User interaction for warnings and getting confirmation (e.g. ***alert***, ***prompt*** and ***confirm*** popup boxes)
- Dynamically changing background and text colours, or "buttons"
- To look at the URL history and take action based on it.
- Open and control windows.
- Provide different documents or parts thereof based on user request - e.g. framed vs not-framed
- Match document versions to browser by sensing user browser applications
- Image effects – buttons, slideshows
- client/browser statistics reporting
- click tracking
- AJAX (Asynchronous JavaScript And XML) - "load-on-demand" – allows web applications to retrieve data from the server asynchronously (in the background) without interfering with the display and behaviour of the existing page (Wikipedia, accessed 2013)
- Manipulating the DOM
- "Sniffing" for plugins and ***browser types***
- Other DHTML effects
- Games and fun applications
- ***Note***: this works in IE but not many other browsers



Development Environments

- TextPad; MS Notepad; DreamWeaver; NotePad++; some great tools/plugins with FireFox and Chrome etc.
- Microsoft Script Editor
  - Available from within MS applications under
  - Tools>Macro>Microsoft Script Editor
  - Shortcut - Alt Shift F11
  - Automatic debugging through MS Explorer: Internet Options > Advanced > uncheck “Disable script debugging (Internet Explorer)”



Where to put JavaScript?

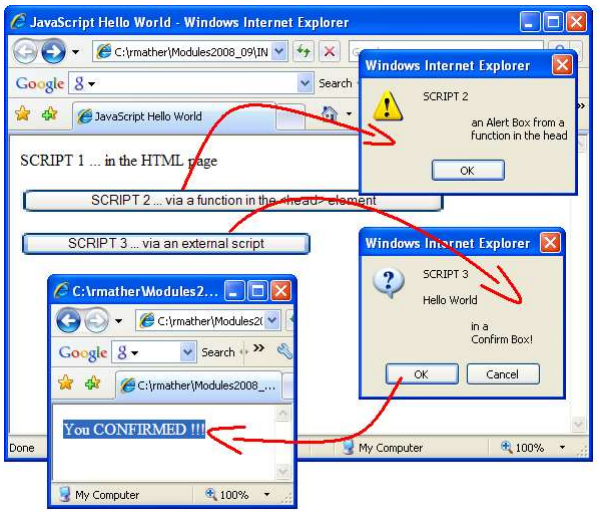
Embedded in the <body>

- <script type="text/javascript"> some script here </script>
- Sometimes older browsers will display code. Easily prevented by treating code as html comments at the beginning and end of scripts
  - <script language="JavaScript"> <!– ***… script here …*** //--> </script>
- Scripts placed in the <body> are executed when the page loads

Embedded in the <head>

- Useful for 'dynamic' available functions
  - e.g. user/event driven after page has loaded

Click the image for examples



External scripts

- May be placed in <head> or <body>
  - e.g. <head><script src="myScript.js"></script></head>

JavaScript - Variables

- Like other programming languages, JavaScript uses variables to store information
- A variable is a bit of memory with a name ***assigned*** to it
- Variables may be of several ***primitive types***
  - examples: [1] number; [2] string (of characters in quotes); [3] boolean (true/false); [4] null (empty); [5] function
- … or may be of composite ***object*** types
- JavaScript variables:
  - are case sensitive
  - cannot contain punctuation, spaces or start with digits
  - cannot be JavaScript reserved words
  - unlike C and Java, may be ***untyped***

JavaScript – Variable Scope

- Variables created ***with var*** (e.g. var = a; or var = a,b,c;) ***may have global scope*** depending on whether it is declared in the main program or in a function
- Variables created without var ***will*** have global scope regardless of where they are declared (inside or outside functions)
- Global and function (local) variables may have the same name but the latter will override the former inside the function where they are declared
- Examine the scope_v1.html to determine the scopes of variables declared

```
<html><head><title>JavaScript Scope Demonstration</title></head>
<body>  <script type="text/javascript">
        var myname="Fred";
        tester();
        document.write(myname+"<br/>");
        tester();
        document.write(othername+"<br/>");
        document.write(hername+"<br/>");

        function tester()
        {
            var myname="Bert";
            var hername="Jane";
            othername="Sid";
            document.write(myname+"<br/>");
        }
        </script></body></html>
```

Browser output … ???

Click the image for live example and code with commented solution

Assigning values to variables

- Assign values with the "=" operator (e.g. x=123; x="Bert")
- Assignments with simple arithmetic operators

Assignment	Alternative	Notes
x=x+1	x++	Increment
x=x-1	x--	Decrement
x=x+y	x+=y	
x=x-y	x-=y	
x=x*y	x*=y	
x=x/y	x/=y	
x=x%y	x%=y	Here % is modulus or division remainder

JavaScript – about “Strings”

- sequences of characters
- enclosed in single or double quotes
- unlike C/C++/Java - no ‘char’ type for a single character
- like C/C++/Java – can use escape: e.g. \n (new line) and \t (tab)
- to insert HTML output into strings, you may need other escape sequences (e.g. escaping quotes \” and \’)
- may be joined (‘concatenated’) with other strings and variables
  - myString = “my full name is ” + firstName + “ and ” + “secondName”;
- a string is also an ***object type***, therefore has internal properties (data) and methods (operations) that may be invoked
  - e.g. using a string property - lengthOfString = myString.length;
  - e.g. using a string method - thirdCharacter = myString.charAt(3);
  - e.g. using a string method – myString.toLowerCase()
  - e.g. using a string method – myString.toUpperCase()

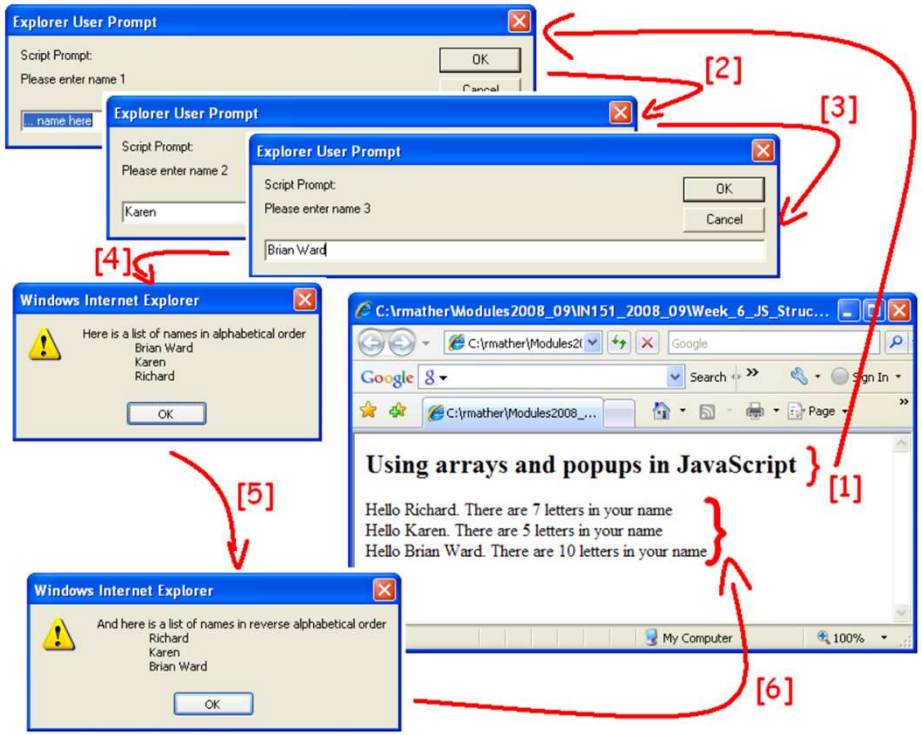
JavaScript – Arrays

- an array is a ***collection*** of data. It can store a list of values against a single variable name.
- each ***element*** has an ***index*** for accessing/addressing
- the first item/element index is zero – e.g. myArray[0] = 123;
- created with a "new Array()" ***constructor*** (similar to full OO languages)
  - **var myArray = new Array();**
  - **myArray[0] = 1; myArray[1] = 2; myArray[2] = "three"; myArray[3] = true;**
- may ***initialise*** values at the same time as ***declaring*** and array
  - **var myArray = new Array(1, 2, "three", true);**
- or may declare the number of elements
  - **var myArray = new Array(15);**
- like strings, arrays are also of ***object type*** with internal properties (data) and methods (operations) that may be invoked
  - e.g. using an array property - lengthArray = myArray.length;
  - e.g. using an array method to sort into alhphanumeric order - myArray.sort();
  - using an array method to sort into reverse alhphanumeric order - myArray.reverse();

JavaScript exercise (see Practical Sheet)

Write a short JavaScript program that:

1. uses “prompt” boxes to store three names in an array;
2. outputs to the HTML page (i.e. using document.write()) on three separate rows messages in the form “Hello nameX, there are the Y letters in your name”;
3. outputs in an “alert” box the message “Here is the list of names in alphabetical order” followed by each name sorted alphanumerically on separate rows in the alert box;
4. outputs an “alert” box the message “Here is the list of names in reverse alphabetical order” followed by the alphanumerically reversed list in the alert box.



JavaScript summary

- Client-side scripting language widely used for dynamic web pages and real time user responses
- Can places scripts in <head>, <body> or externally
- A number of ready made pop up functions (alert, prompt, confirm)
- Have covered variables, strings and arrays

- NEXT WEEK – functions, branching statements and iteration (loops)