COS10024 Web Development

Lecture 9 – Document Object Model

Dr. Gamunu Dassanayake

TP3/2022

SWINBURNE UNIVERSITY OF TECHNOLOGY

Contents

- **JavaScript Objects**
- **Document Object Model (DOM)**
  - Predefined Objects
  - Document
  - Elements
  - Specific Elements
  - Class and Style
- **Array Object**
- **Date Object**
- **Global Functions**
- **Validating Form Values**

JavaScript objects

JavaScript Objects

- **JavaScript is an object-based language and does some things procedurally.**
- **It can support polymorphism, inheritance and encapsulation.**
- **It can access objects such as**
  - browser **objects** such as **window**, **navigator**
  - webpage **objects** such as **document**, **images**, **dates**, **forms** and the hierarchy of form control elements such as **inputs**, **checkboxes**, **select**, and **buttons**, etc., within forms.

JavaScript Principle 1:

All the elements on a webpage are objects!

JavaScript Principle 2:

Get access to the right elements/objects, use the right properties and the right functions.

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

JavaScript Objects [2]

An object has

Principle 1!

- **properties which describe the object**
  - A form <input> object has properties: id, *value*, etc.
  - Usually *nouns* as they *describe things*.

- **functions which describe actions that the object can do.**

Principle 2!

  - A form element can submit: myForm.submit().
  - A image change its href attribute: myImage.setAttribute("href", "image2.png")
  - Usually *verbs* as they *describe actions*.

Intrinsic Object Types

- Array
- Boolean
- Date
- Math
- Number
- String

Examples

- allows you to create an object

```
// creates a date object with the
current date

var today = new Date();
alert(today);
```

DEMO!

Must be capitalised.

- provides related functions

```
// returns PI
var x = Math.PI;
```

There are also predefined **global** functions

Document Object Model

DOM History

Document Object Model (DOM)

- a platform and language neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style of a document [W3C]

  http://www.w3.org/DOM/

- a way to represent and navigate an HTML or XML document as a tree

Document Object Model (DOM) [2]

- **The W3C has developed DOM "levels" to represent the different features that may be supported**

DOM Level 0:	The earlier *vendor specific* intermediate DOMs
DOM Level 1:	HTML & XML document tree structures, including HTML specific elements and node add / move / delete.
DOM Level 2:	XML namespaces, styles, views, and events
DOM Level 3:	Divided into specific modular sections

http://www.w3.org/DOM/DOMTR

How well are the Core and HTML DOMs implemented in browsers?

http://quirksmode.org/dom/core/

http://quirksmode.org/dom/w3c_html.html

SWINBURNE UNIVERSITY OF TECHNOLOGY

Document Object Model (DOM) [3]

- Current standard is DOM Level 3, released in 2004.
- DOM is not part of core JavaScript, but JavaScript uses the DOM to interact with the Web browser. This technique is referred to as DOM manipulation.
- DOM does use JavaScript's Intrinsic Objects, such as Array, Boolean, Date, Math, Number, RegExp, String, …

DOM Support

- There were many problems related to browser specific DOM implementation! Code-writers had to create "browser detection" code and "browser-specific" routines to get around the different DOM.

- W3C DOM Level 1 (rec. Oct 1998) and DOM Level 2 (rec. Nov 2000) are now largely supported by recent browsers.

- See what DOM your browser supports
  http://www.w3.org/2003/02/06-dom-support.html

- See the DOM compatibility tests
  http://www.quirksmode.org/compatibility.html

Document Object Model

Predefined Objects

Predefined Objects

- *window*
- *document*
- *navigator*
- *screen*
- *history*
- *location*

Examples

```
window.close();(Chrome vs Firefox)
window.alert();
```

`document.getElementById("myID");`

```
navigator.platform;
navigator.language;

screen.height;
screen.width;

history.back();
history.forward();

location.href;
```

Document Object Model

Window

Window Object – window

- Methods *(this is not a complete list of its methods)*

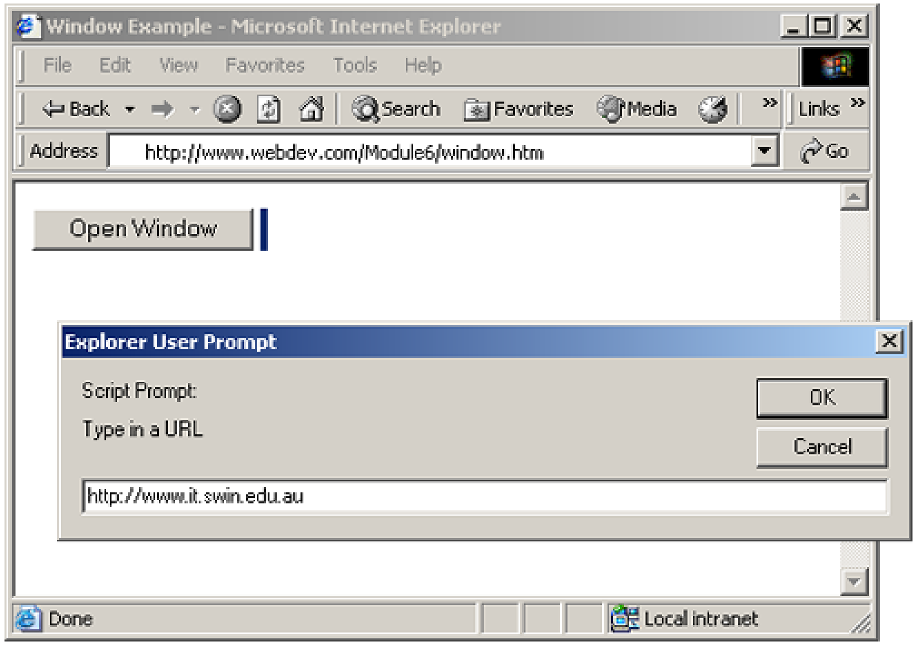
alert(*text*)	- pops up an alert box
confirm(*text*)	- pops up a box with 'OK' or 'Cancel'
prompt(*text, def*)	- retrieves a line of text from the user
open(*url, [ops]*)	- opens up a new window
close()	- closes a window
focus()	- gives focus to a window
blur()	- removes focus from a window

- Window HTML Event Handling

onload	- occurs when the page has completed the loading process.

Window Object – Example

```
function newWindow() {
  theUrl = window.prompt("Type in a URL",
                                                    window.location);
  window.open(theUrl);
}
```


Window Example - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media Links
Address http://www.webdev.com/Module6/window.htm Go
Open Window
Explorer User Prompt
OK
Cancel
Done
Local intranet

Script Prompt:

Type in a URL

http://www.it.swin.edu.au

Document Object Model

document

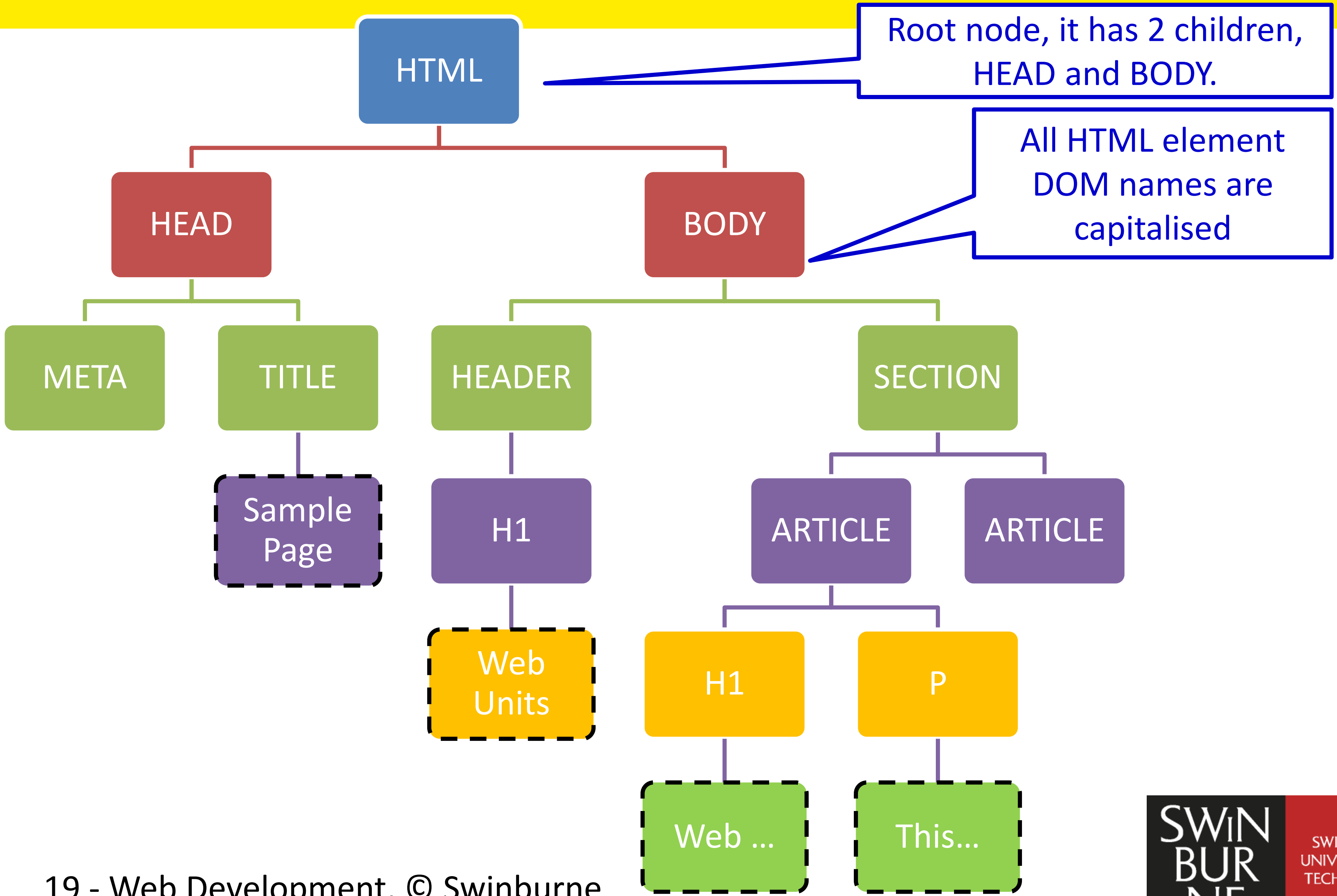
DOM Object - Example

- A HTML document is represented as a tree of nodes.
- The first node is referred to as the root node.
- Each node can have children.
- Node with no children is referred to as leaf node.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <title>Sample Page</title>
</head>
<body>
  <header>
    <h1>Web Units</h1>
  </header>
  <section>
     <article>
       <h1>Web Development</h1>
       <p>This unit covers…</p>
     </article>
     <article>
     </article>
  </section>
</body>
</html>
```

SWIN BUR NE SWINBURNE UNIVERSITY OF TECHNOLOGY

DOM Object – Tree Structure



Root node, it has 2 children, HEAD and BODY.

All HTML element DOM names are capitalised

Document Objects

Where are the objects?

- **The entire HTML page is made up of objects**
- **Using the tree representation, each node is an object.**
- **In our example, we have 16 objects.**

Document Object – Property/Function

- **A frequently used function of the document object is**

  document.getElementById(<id>)

  It returns the reference to a specific HTML element using the ID attribute specified in the HTML document.

  Sample use:

```
var x = document.getElementById("intro");
x.innerHTML = "This is introduction.";
x.style.color = "red";
x.style.backgroundColor = "blue";
```

Principle 1! **Principle 2!**

Document Object – Property/Function

- Some useful properties and functions of the document object:

document.
  - documentElement
  - **getElementById()**
  - getElementsByTagName()
  - createElement()
  - createTextNode()
  - createAttribute()

Document Object Model

Elements

Accessing Elements

- Three most frequently-used way to access HTML elements using JavaScript.

An individual element.

```
var element1 = document.documentElement;
```

An individual element Id.

```
var element2 =
  document.getElementById("btnExecute");
```

An array of multiple elements.

```
var elements =
  document.getElementsByTagName("a");
```

Principle 1!

Principle 2!

Accessing Elements (Examples)

Question:
How many `<body>` elements will be obtained?

- **Get the body element (get all tags named "body")**

```
var bodyElements =
document.getElementsByTagName("body");
```

- **Get all images from the `<body>` element**

```
var imgElements =
bodyElement[0].getElementsByTagName("img
");
```

`bodyElement` is an array of only one `<body>` element. Thus, `bodyElement[0]` returns that only `<body>` element.

Accessing Elements (Examples)

- Get the element with `id="intro"`:

```
var introElement = document.getElementById("intro");
```

- Get all <p> elements that are descendants of the element with `id="main"`

```
var mainElement = document.getElementById("main")

var mainParagraphElements =
mainElement.getElementsByTagName("p");
```

Using Properties and Functions

```
element.
        id
        className
        tagName
        getAttribute()
        setAttribute()
        removeAttribute()
```

```
<input
type="button"
id="btnExecute"
value="Execute"
class="myClass" />
```

- **For example:**
  - `element.id` → `"btnExecute"`
  - `element.tagName` → `"INPUT"`
  - `element.getAttribute("type")` → `"button"`

```
var element = document.getElementById("btnExecute");
```

SWINBURNE UNIVERSITY OF TECHNOLOGY

Using Properties and Functions

- **How do you check the type of an element?**
  - Property tagName
  - Example:

Principle 2!

```
var tagName =
document.getElementById("btnExecute ").tagName;
if(tagName == "INPUT") {
    alert("This is an input element.");
}
```

DEMO!

Using Properties and Functions

- **Other properties**
  - parentNode
  - firstChild
  - lastChild
  - previousSibling
  - nextSibling

```
<article>
  <header id="h">
     Header!
  </header>
  <section>
     Section!
  </section>
</article>
```

- **Examples**

var element=document.getElementById("h");

- `element.parentNode` → The `article` element.
- `element.firstChild` → The `Header!` text node.
- `element.nextSibling` → The `section` element.

SWINBURNE UNIVERSITY OF TECHNOLOGY

Document Object Model

Specific Elements

Specific Elements

- **The following HTML elements will have specific properties**
  - Links <a …>…</a>
  - Forms <form …>…</form>
  - Select / Option elements <select …>… </select>
  - Input (text, radio, checkbox, password, hidden, submit …) <input … />
  - Textarea <textarea … >… </textarea>
  - Images <img … />

Specific Elements – <a>

Anchor Element `<a href="…">…</a>`

- **anchorElement.**
  href

- **Examples** `<a href="ads.html" id="s">`

  `var myLink=document.getElementById("s");`
  - `myLink.href`

The absolute URL of `ads.html`.

Specific Elements – <form>

Form Element `<form …>…</form>`

- **`myForm.`**

  elements[]
  action
  method
  submit()
  reset()
  length

An array of all the elements in the form.

- **For example**
  - myForm.length
  - myForm.reset()
  - myForm.submit()

Specific Elements – <select>

Select Element **<select …>…</select>**

- **selectElement.**

type	`multiple`
selectedIndex	`name`
value	`options[]`
disabled	`add()`
size	`remove()`

```
<select>
  <option value="iPhone4">iPhone 4</option>
  <option value="iPhone5">iPhone 5</option>
  <option value="iPhone6">iPhone 6</option>
</select>
```

- **For example**
  - mySelect.value
  - mySelect.options[0]

Specific Elements – <option>

Option Element <option …>…</option>

- optionElement.
```
    text
    disabled
    selected
    value, …
```
- For example
  - myOption.text

Specific Elements – <input>

Input Element <input … />

- inputElement.

  form
  checked
  disabled
  name

  `readOnly`
  `value`
  `select()`
  `click(), …`

- For example
  - myInput.checked

DEMO!

Specific Elements – <textarea>

Text Area Element

`<textarea … >…</textarea>`

- **textAreaElement**
```
    form
    disabled
    name
    readOnly
    value
    select(), …
```

- **For example**
  – `myTextArea.value`

DEMO!

Specific Elements – <img>

Image Element **<img … />**

- `imgElement.`

```
    src
    alt, …
```

DEMO!

- **For example**
  - myImg.src
  - myImg.alt

Document Object Model

Class and Style

Document Object (Class and Style)

- Element attribute names are directly matched to DOM property names. For example,

```
<a href="page1.htm" class="button">
```

```
linkElement.href
```

- The exception of using the attribute name is the `class` attribute, which is mapped to

```
objElement.className
```

Not `"class"`, as `"class"` is a *reserved word* in JavaScript

Principle 2!

SWIN BUR NE SWINBURNE UNIVERSITY OF TECHNOLOGY

Document Object (Class and Style)

- Class is often used to associate style with elements. If we change the class of an element in JavaScript, the browser changes the associated presentation of that element.

```
if (objElement.className == "blue") {
      objElement.className = "red";
}
```

```
<h1 class="blue">   ->   <h1 class="red">
```

Document Object (Class and Style)

- objElement.style.
  color
  background
  backgroundAttachment
  backgroundColor
  backgroundImage
  backgroundPosition
  backgroundPositionX
  backgroundPositionY
  backgroundRepeat

- For example,

```
if(objElement.style.color == "blue")
  {
  objElement.style.color = "red"
  }
CSS:   color:blue;   ->  color:blue;
```

border
borderCollapse
borderColor
borderSpacing
borderStyle
borderTop
borderRightColor
borderLeftStyle
borderBottomWidth

Principle 2!

SWINBURNE UNIVERSITY OF TECHNOLOGY

Document Object Model

Array Object

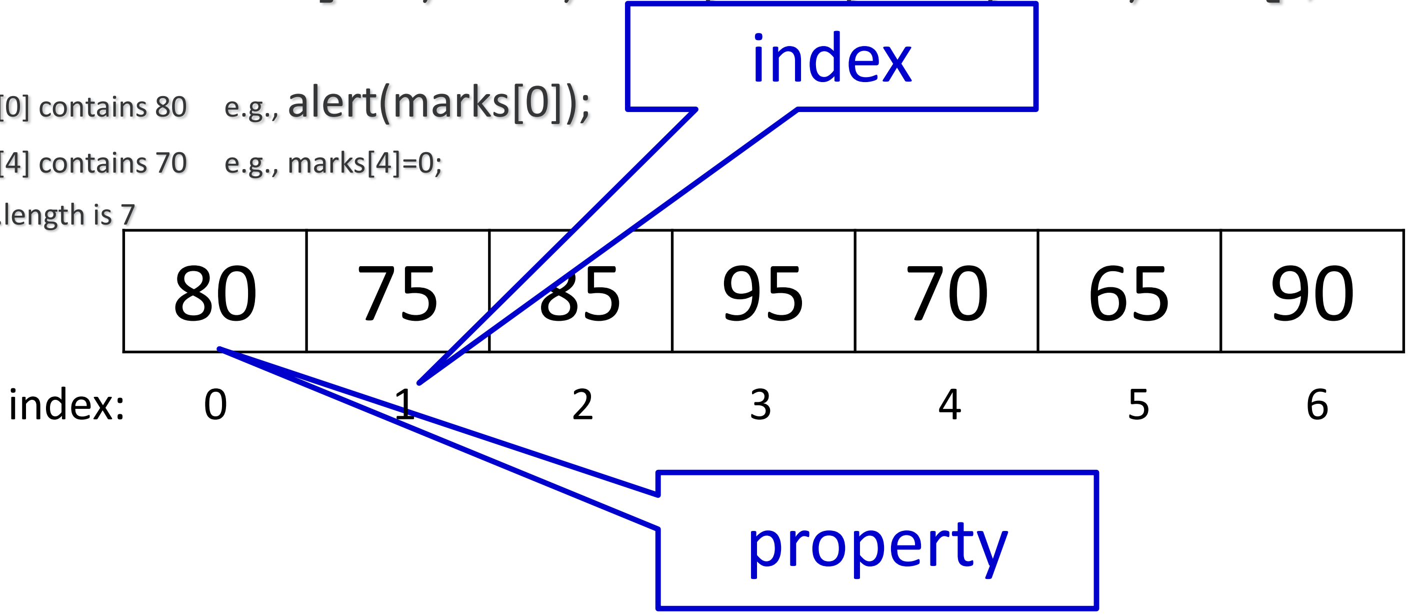
Array Object

- An indexed collection of variables
- A particular variable in an array is referenced by the array name and the index of the variable.
- For example:

```
var marks=[80, 75, 85, 95, 70, 65, 90];
```

`marks`

- marks[0] contains 80 e.g., alert(marks[0]);
- marks[4] contains 70 e.g., marks[4]=0;
- marks.length is 7



Array Object (continued)

- In JavaScript an Array is an object.
- The new keyword is used in JavaScript to create y object.

```
var marks;
marks = new Array(10);
                // 10 variables [0]-[9]
// or
var marks = new Array(15);
                // 15 variables, [0]-[14]
```

parenthesis

use plural form to indicate array

Array Object (continued)

- Values can be assigned to variables in the array after the array has been created:

```
var subjects = new Array(2);
subjects[0] = "WD";
subjects[1] = "WP";
```

parenthesis

Square brackets

- Variables in an array may be initialized when the array is created:

```
var subjects = new Array("WD","WP");
var numbers = new Array(1,1,2,3);
```

SWIN BUR NE SWINBURNE UNIVERSITY OF TECHNOLOGY

Array Object (continued)

- **The length of the array can be accessed using the length property.**
  - e.g. numbers.length;
- **Values can be set programmatically:**

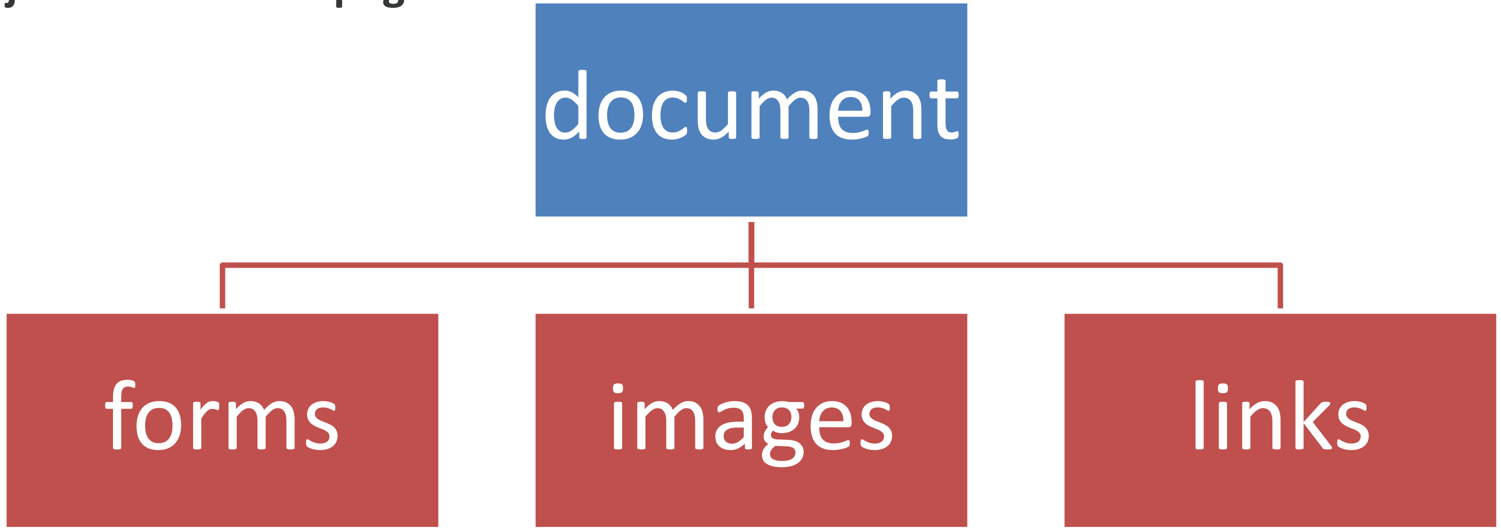
```
// create an array
var numbers = new Array(10);
// fill array with numbers
for (i=0; i < numbers.length; i++) {
    numbers[i] = i*2;
}
// display the last element
alert(numbers[numbers.length - 1]);
```

Why not subtract 1?

Why subtract 1?

Array Object (continued)

- The document object and its arrays of specific objects.
- These are arrays of specific objects, e.g. forms is an array of all the <form> objects on the webpage.



Array Object (continued)

- These arrays are created and initialised automatically.
- Use indexes to accessing the objects in those arrays:

```
var myForm = document.forms[1];
var myImage = document.images[2];
var myLink = document.links[0];
```

An alternative to using document.getElementById() to access individual elements.

Array Object (continued)

- display scores array as a horizontal chart

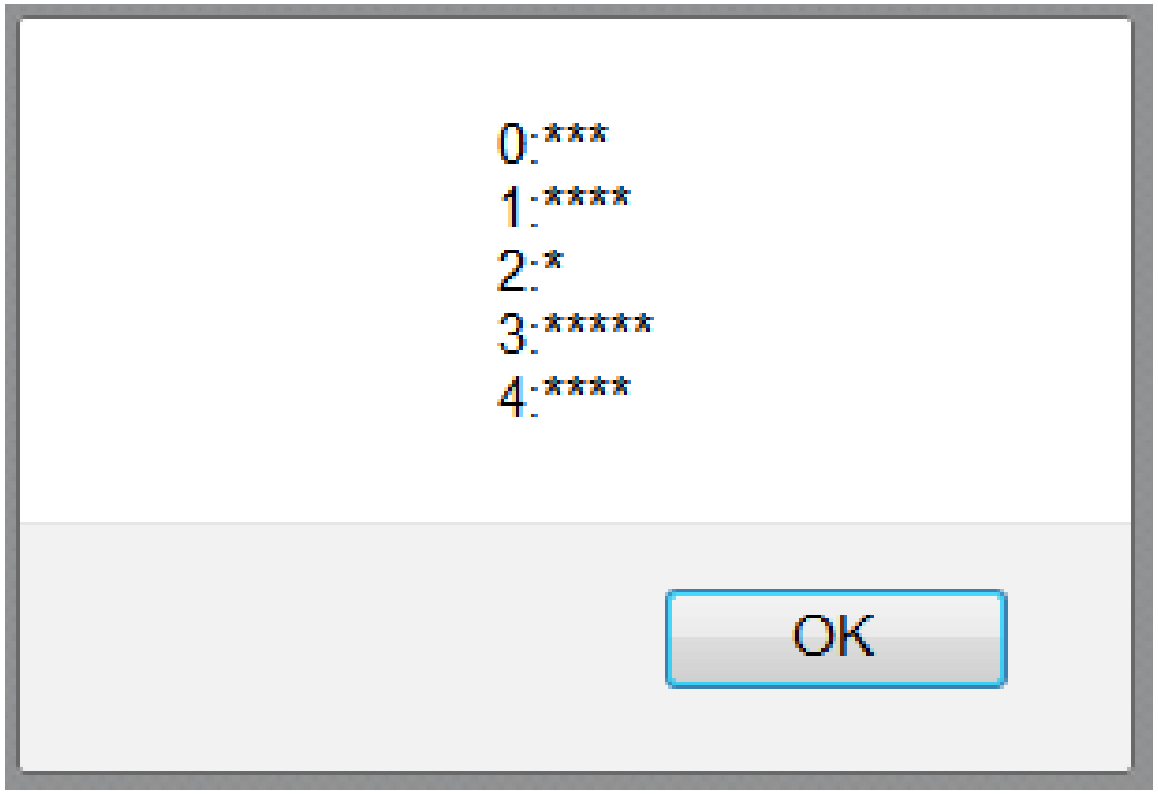
```
var scores = new Array(3,4,1,5,4);
var index;           // array index
var num;             // number
var ans = "";              // string for output
// how to use for loop to traverse an array
for (index=0; index<scores.length; ++index) {
   num = scores[index];
   ans = ans + index.toString() + ": ";

   for (var i=0; i<num; i++) {
      ans = ans + "*";
   }
   ans = ans + "\n";
}
alert(ans);
```

Function to convert a number to a string

"\n" for line break

Array Object (continued)

The alert box will display:



```
0:***
1:****
2:*
3:*****
4:****
```

Array Object – Properties/Functions

Function	Description
`length`	returns length of the array
`join(delimiter)`	makes a string delimited with the items
`pop()`	removes the last item and return it
`push(item)`	Add item to the end of the array
`reverse()`	reverses the order of items
`shift()`	removes the first item and returns it
`slice(start,[end])`	returns a sub-array
`sort(fn)`	fn needs (a<b)==-1, (a==b)=0, (a>b)==1
`unshift(item)`	add item to start of array

https://developer.mozilla.org/en/JavaScript/Guide/Predefined_Core_Objects

SWIN BUR NE SWINBURNE UNIVERSITY OF TECHNOLOGY

Document Object Model

Date Object

Date Object

- Represents a date

- Numeric value is expressed as millisecond

```
var d = new Date("May 8, 2013 17:30:00");
```

Full or 3-letter month

```
var d = new Date();
```

New instance of client’s current date and time

- Functions can be used to obtain values within the date object

```
var n = d.getDate();
```

SWIN BUR NE SWINBURNE UNIVERSITY OF TECHNOLOGY

Date Object - Some Functions

Function	Description
`getDate()`	Returns the day of the month (from 1-31)
`getDay()`	Returns the day of the week (from 0-6)
`getFullYear()`	Returns the year (four digits)
`getHours()`	Returns the hour (from 0-23)
`getMilliseconds()`	Returns the milliseconds (from 0-999)
`getMinutes()`	Returns the minutes (from 0-59)
`getMonth()`	Returns the month (from 0-11)
`getSeconds()`	Returns the seconds (from 0-59)

DEMO!

SWIN BUR NE
SWINBURNE UNIVERSITY OF TECHNOLOGY

Document Object Model

Global Functions

Global Functions

Be careful of case

Function	Description
`eval()`	Evaluates a string and executes it as if it was script code
`isFinite()`	Determines whether a value is a finite, legal number
`isNaN()`	Determines whether a value is an illegal number
`Number()`	Converts an value to a number
`parseFloat()`	Parses a string and returns a floating point number
`parseInt()`	Parses a string and returns an integer
`String()`	Converts an object's value to a string

Global Functions (Examples)

Function	Example	Result
`eval()`	`eval("2 + 3")`	5

`isFinite()`	`isFinite(5)`
	`isFinite("Web")`
		true
		false

`isNaN()`	`isNaN(5)`
	`isNaN("Web")`
		false
		true

`Number()`	`Number("22")`
	`Number("2 2")`
		22
		NaN - invalid number

`parseFloat()`	`parseFloat("2")`
	`parseFloat("2.34")`
	`parseFloat("2 34")`
	`parseFloat("2 units)`
	`parseFloat("unit 2)`
		2
		2.34
		2
		2
		NaN

SWINBURNE UNIVERSITY OF TECHNOLOGY

Global Functions (Examples)

Function	Example	Result

`parseInt()`	`parseInt("2")`
	`parseInt("2.34")`
	`parseInt("2 34")`
	`parseInt("2 units)`
	`parseInt("unit 2)`
		2
		2
		2
		2
		NaN

`String()`	`String(0)`
	`String(`**`true`**`)`
	`String("2")`
		"0"
		"true"
		"2"

JavaScript

Validating form values

(Critical Part in assignment 2)

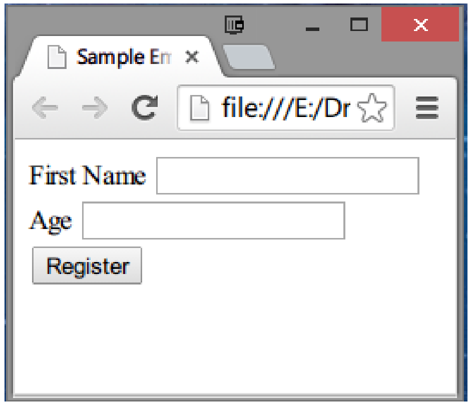
Forms and JavaScript

- **JavaScript provides much greater control over the use of forms by:**
  - Checking form values entered, before the form is submitted:
    - that **required** form values have been supplied
    - that values **conform to a type**
      (e.g., must be an integer, or a string, etc)
    - that values are **logical** or **constrained**
      (e.g., end date after start date, value in a range, etc)
  - Alerting users if invalid form values have been entered
  - Pre-processing form data before submission

Checking Form Data

Given the following HTML form, *take note of the IDs*

```
<form id="regform" method="post" action="process.php">
    <div class="textinput">
        <label for="firstname">First Name</label>
        <input type="text" name="firstname"  id="firstname" >
    </div>
    <div class="textinput">
        <label for="age">Age</label>
        <input type="text" name="age" id="age" >
    </div>
    <div class="buttoninput">
        <input type="submit" value="Register" >
    </div>
</form>
```



Checking Form Data (continued)

- **Using the JavaScript template**

Part 1

```
function validate() {
  /* validation code here */
  return true/false;
}
```

Write the data validation code, and return **`true`** if all valid, otherwise **`false`**

Part 2

```
function init() {
 var formElement =
   document.getElementById("regform");
   formElement.onsubmit = validate;
}
```

Link function `validate()` to the `onsubmit` event of the form

Part 3

```
window.onload = init;
```

Make sure function `init()` is executed when the page window is loaded.

Checking Form Data in Steps

- JavaScript validation Parts 2 and 3

Part 2

```
function init() {
var regForm =
document.getElementById("regform");
regForm.onsubmit = validate;
}
```

Part 3

```
window.onload = init;
```

Checking Form Data [2]

- JavaScript validation Part 1A

```
function validate() {
 var errMsg = "";
 var result;
 var firstName =

document.getElementById("firstname").value;
 var age =
     document.getElementById("age").value;
```

value property of an HTML element

Checking Form Data [3]

- **JavaScript validation Part 1B**

```
if (firstName == "") {
    errMsg = errMsg + "First Name cannot be
                                    empty.\n";
}
if (age == "") {
    errMsg += "Age cannot be empty.\n";
}
if (isNaN(age)) {
    errMsg += "Age is
                              number.\n";
}
```

Concatenate error message

Add a new line for when displayed in the alert window

Use global function `isNaN()` to check if age contains a valid number.

SWINBURNE UNIVERSITY OF TECHNOLOGY

Checking Form Data [3]

- **JavaScript validation Part 1C**

```
if (errMsg != "") {
    alert(errMsg);
    result = false;
} else {
        result = true;
}
  return result;
}
```

Checks if any error messages

Error detected

Returns true is no errors detected, otherwise false

SWINBURNE UNIVERSITY OF TECHNOLOGY

Checking Form Data (Regular Expressions)

- If you use regular expressions, define the regular expression pattern to be used, then use the match() function for checking, e.g.,

```
var ageRE = /^\d\d$/;
if (!age.match(ageRE)) {
      errMsg+="Invalid age.\n"
}
```

This regular expression allows only two digits for age, numbers only.

/^[a-zA-Z]+$/ allows letters and spaces only.

Next Lecture

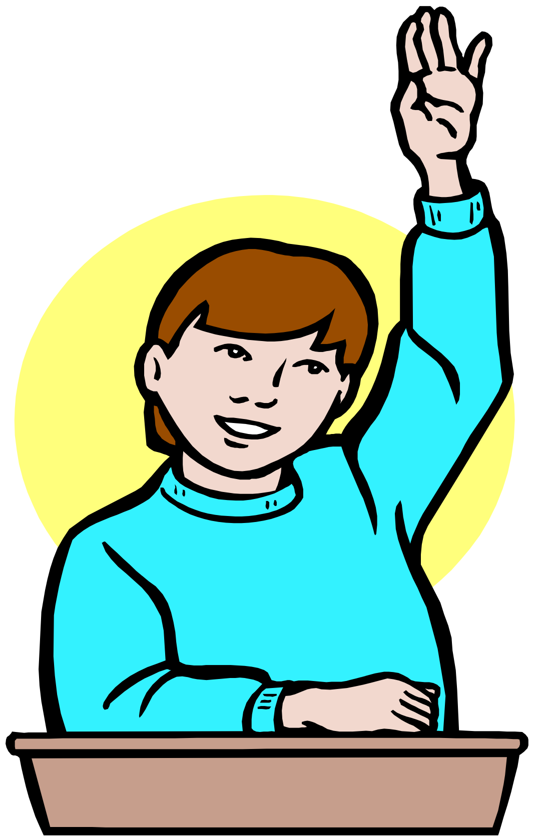
jQuery and Server Side Scripting

Reminder

- Week – 9 Lab Submission
- Assignment 2 (Due Week 12)

Question?

- A good question deserve a good grade…



Thanks Lot!!!