CS2204 Fundamentals of Internet Applications Development

7. Javascript Part II

How Javascript works with objects and interacts with the Web page?

Finding & using objects in Web page:

- Self created objects (variables)
- DOM

Event handling.

7.1. Objects & Variables

7.1. Objects & Variables

In Javascript everything can be regarded as objects including the primitive data types and functions. In other Object Oriented languages, objects (object instances) are created from classes (template) through instantiation. However, the class concept is not obvious in Javascript. Object creation is done by the new operator or literal.

```
currentDT = new Date();
myvar = 123; //123 is a literal
```

Variables are then used to "store" or point to object.

Variables (in the form of objects) are therefore used to store temporary information when Javascript is being run in the Web page. These variables will be gone once the page is reloaded!.

There are 4 main kinds of objects commonly used:

- simple primitive objects - number, string and boolean
- built-in objects - Array, Date and Math, etc.
- self-defined objects - we define the structure of the object

- DOM – provided by the browser as the host environment

7.1.1. How does an object look like?

7.1.1. How does an object look like?

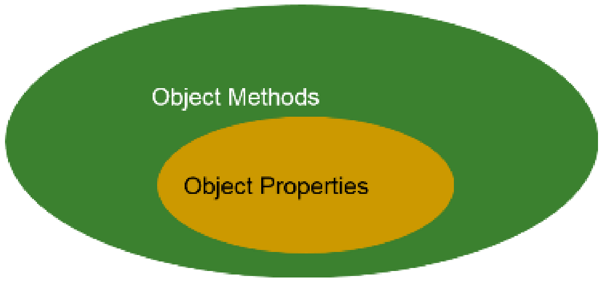
After getting hold of an object, we could work on its properties or with methods. Methods may not exist in some objects.

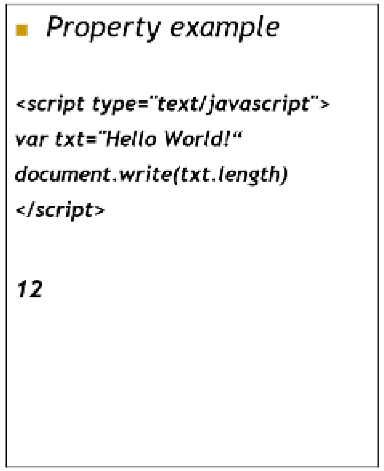
Properties

- values associated with an object, such as length, and width; styles and events are also properties
- can get/change their values by JS

Methods

- actions that can be preformed on objects, such as write() of the document object, i.e. document.write()
- use them in JS to do something

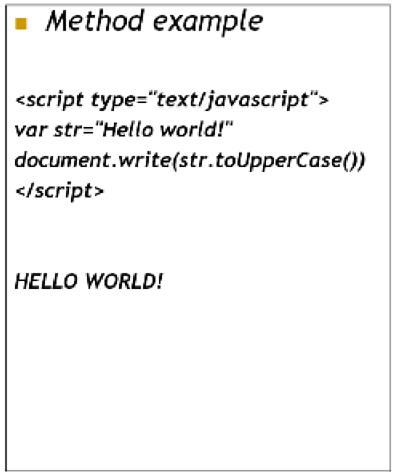




■ *Property example*

```
<script type="text/javascript">
var txt="Hello World!“
document.write(txt.length)
</script>
```

```
12
```



■ *Method example*

```
<script type="text/javascript">
var str="Hello world!"
document.write(str.toUpperCase())
</script>
```

```
HELLO WORLD!
```

7.1.2. How to define our own objects - JSON

7.1.2. How to define our own objects - JSON

Sometimes simple variable cannot store information in a structured or more complex way. Objects with our own structure needed to be defined. In this case, Javascript Object Notation (JSON) can be used.

Object literal

```
var myObj = {
“name”:”mtchan”,
“title”:”Dr”,
"age":25,
"class":[{"coursecode":"cs2204","title":"IAD"},
         {"coursecode":"CS3103","title":"Internet"}]
};
```

can define string, number, boolean, array, object & null.

JSON is mainly for data definition/exchange, not easy to define method and it is also tedious to type in coding.

7.1.3. Constructor function

Use a constructor function may save a lot of typing

```
function myObj(name, title) {
  this.name = name; //note the use of reserved word this
  this.title = title;
  this.getName = function() {return this.name;};//annonymous function
  return this;
}

t = myObj(“mtchan”, “mr”);
alert(t.getName());
```

note that only one copy of the object exists.

```
t = myObj(“mtchan”, “mr”);
s = myObj("kkchan","dr");
alert(t.getName());
```

Use the new operator

```
function myObj(name, title) {
  this.name = name;
  this.title = title;
  this.getName = function() {return this.name;}
  //no return
}

t = new myObj(“mtchan”, “mr”);
s = new myObj(“kkchan”, “dr”);
alert(t.getName());
alert(s.getName());
```

it is creating multiple object instances now.

7.2. Built-in Objects in Javascript

The following objects are built-in the JavaScript:

Array

Boolean

Date

Math

String

7.2.1. Array

7.2.1. Array

An array is a list of values associated with a variable name (a list of numbered boxes that can contain different objects, including another array).

Create array object MUST use the new operator

```
myarray = new Array(); //recommended
myarray = new Array (arrayLength);
myarray = new Array (element0, element1, … , elementN);
```

also literal

```
myarray = [ ];
```

Array length

- it specifies the initial length of the array
- you can access this value using the length property
- if the value specified is not a number, an array of length 1 is created, with the first element having the specified value

The elements

- it is a list of values for the array's elements
- when this form is specified, the array is initialized with the specified values as its elements, and the array's length property is set to the number of arguments

Should check out other properties and methods of array; a very useful object used all the time

7.2.2. Boolean

Boolean is a primitive data type but can also be viewed as object (object wrapper). Create Boolean by

```
myboolean = new Boolean (value) or
myboolean = true (or false)
```

Value

- specifies the initial value of the Boolean object
- the value is converted to a boolean value, if necessary
- if value is omitted or is 0, -0, null, false, NaN, undefined, or the empty string (“”), the object has an initial value of false
- all other values, including any object or the string “false”, create an object with an initial value of true

Print Boolean value

```
<script type="text/javascript">
var boo = new Boolean(false)
```

```
document.write(boo.valueOf())
</script>
```

- *All these give a false Boolean*
  - var myBoolean=new Boolean()
  - var myBoolean=new Boolean(0)
  - var myBoolean=new Boolean(null)
  - var myBoolean=new Boolean("")
  - var myBoolean=new Boolean(false)
  - var myBoolean=new Boolean(NaN)
- *All these give a true Boolean*
  - var myBoolean=new Boolean(true)
  - var myBoolean=new Boolean("true")
  - var myBoolean=new Boolean("false")
  - var myBoolean=new Boolean("Richard")

7.2.3. Date Creation

7.2.3. Date Creation

The Date object is used to work with dates and times. Create Date object by

```
mydate = new Date ()
new Date (milliseconds)
new Date (dateString)
new Date (yr_num, mo_num, day_num [, hr_num, min_num, sec_num, ms_num])
```

- no argument
  - the constructor creates a Date object for today's date and time according to local time
- milliseconds
  - an integer value, representing the number of milliseconds since 1 January 1970 00:00:00 UTC
- dateString
  - a string value, representing a date
  - the string should be in a format recognized by the parse method
- yr_num, mo_num, day_num

  - integer values, representing year, month, and day
  - month is representing by 0 to 11 with 0=January, and 11=December
- hr_num, min_num, sec_num, ms_num
  - integer values representing hours, minutes, seconds, and milliseconds

7.2.3.1. Date use

The most common error is not creating the date object and use the methods

- Remember to use Date constructor
- today = new Date();

Some useful methods of the date object

- today.getDate() - returns 1-31
- today.getDay() - returns 0-6
- today.getMonth() - returns 0-11
- today.getFullYear() - returns 0-23
- today.getHours() , etc.

Need to use with care, the date function in Javascript is quite messy, e.g. parse(); third-party date library is available.

7.2.4. Math Object | example use - http://courses.cs.cityu.edu.hk/cs2204/example/html/29-ArrayNDate.html

7.2.4. Math Object

Since Javascript views everything as objects, doing mathematics requires the Math object, consists of

- properties - mathematical constants
- methods - mathematical functions

The Math object is pre-created by JavaScript object, no need to “new”

- can automatically access it without using a constructor or calling a method
- mathematical constant examples: Math.PI, Math.LN2, and Math.LN10
- mathematical function examples: Math.max(number), and Math.round(number)

7.2.5. String

Similar to Boolean, string is a primitive type and can be viewed as object. To create:

```
var txt = new String(string);
```

or simply:

```
var txt = string;
```

Example :
26-DifferentLoops.html

- note how a string can be accessed like an array

t5-echo.html

- this is the a Web page responding to the form submission in tutorial 5
- note how the query string in form submission is accessed as an object property
- note the string method split which is powerful and the multiple arrays return from the method call

7.3. DOM | Example 1 - http://courses.cs.cityu.edu.hk/cs2204/example/html/26-DifferentLoops.html | Example 2 - http://courses.cs.cityu.edu.hk/cs2204/example/html/t5-echo.html | w3schools string - http://www.w3schools.com/jsref/jsref_obj_string.asp

7.3. DOM

The DOM (Document Object Model) is the main bridge between the Web page and Javascript. Similar to CSS rules, Javascript would find/select an object and then manipulate its properties or call its methods.

How to find an object? You already know how - use CSS selectors to find elements

```
document.querySelector( ‘CSS selector')
```

- only one element is returned
- document.querySelector(‘#myheader’) - return the element with id
- document.querySelector(‘.myclass’) - return first element with class in the order found in the document
- document.querySelector(‘#myid, #yourid’) - return either one found first

```
document.querySelectorAll( )
```

- return all elements found as a static node list

```
chaining
```

- apply the methods on an element instead of document
- document.querySelector(‘#myid’).querySelectorAll(‘.myclass’)
- only descendant of myid element will be returned

7.3.1. Older methods to find elements

7.3.1. Older methods to find elements

The query selectors had been around for a few years. Previously, there were different ways which are quite confusing and not easy to remember. Query selectors are recommended.

By Id

```
thatElement = document.getElementById(“idname”);
```

By Tag Name

```
mytags=document.getElementsByTagName(“h1”);
```

By combination

```
myelements=document.getElementById(“id”).getElementsByTagName(“p”)
```

By pre-defined arrays - only for certain elements (really old, out-dated)

- document.images[0] - the first <img> tag in the Web page
- document.links[2] - the third <a> tag

- document.forms[1] - the second <form>

7.3.2. How Javascript works with objects

7.3.2. How Javascript works with objects

Properties

- the characteristics of the object, such as font properties, color properties, and box properties can be read or changed with JS

Methods - use them to do something, such as :

- alert() - it should be window.alert(), the object is window, to alert something to the window
- document.write() - write something to the document object, i.e. the Web page
- document.querySelector(“video”).play()

Event Handlers

- they are functions are "attached" to objects and used to trap events happening to their owning object

- example events : onBlur, onChange, onFocus, onSubmit, and onReset

7.3.3. Interacting with the DOM

7.3.3. Interacting with the DOM

Programming Data in a Form

- form is a common interface to get user input
- DOM can be used to access data in each field (input)
- processing can be done
    - before the form is submitted
    - fill in a form with programming
    - after data received from another page

Examples - accessing form data

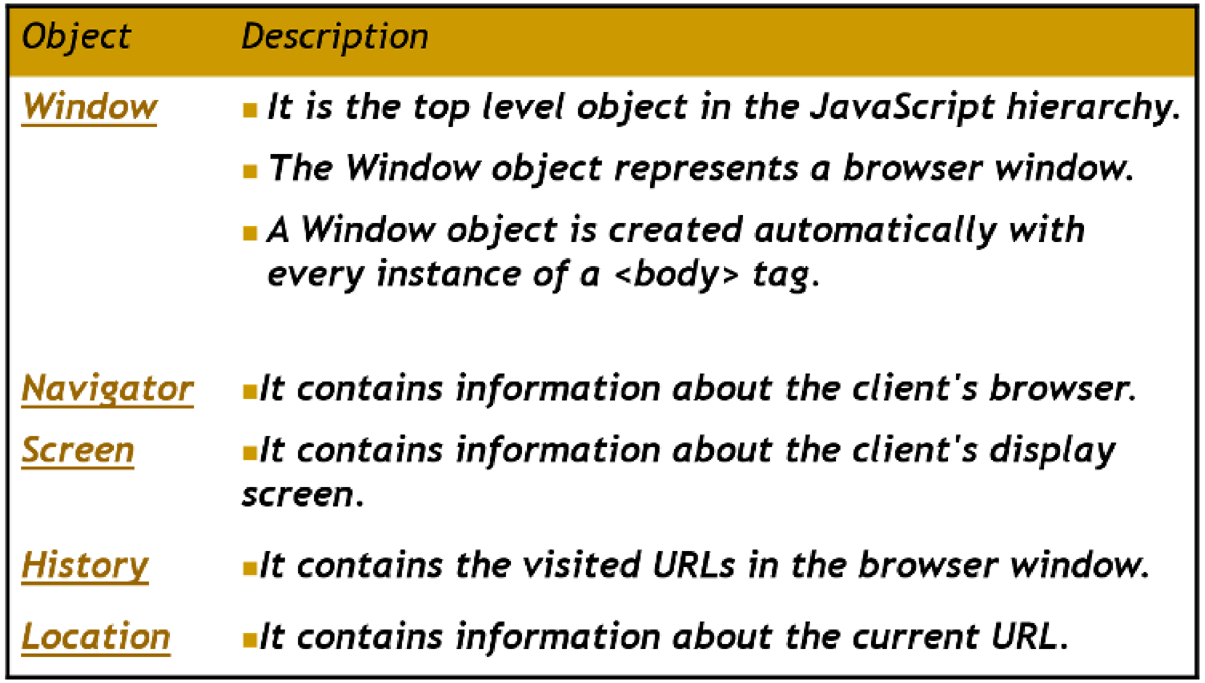
```
document.querySelector("#firstname").value=getVar("firstname");

agegroup=document.querySelectorAll("input[type='radio']");
agegroup[0].checked=true;

videolike=document.querySelectorAll("input[type='checkbox']");
videolike[0].checked=true;
```

7.3.4. HTML DOM Objects | Send form data - http://courses.cs.cityu.edu.hk/cs2204/example/html/31-DOMFormData.html | Receive and process form data - http://courses.cs.cityu.edu.hk/cs2204/example/html/31-DOMFormDataAccess.html

7.3.4. HTML DOM Objects

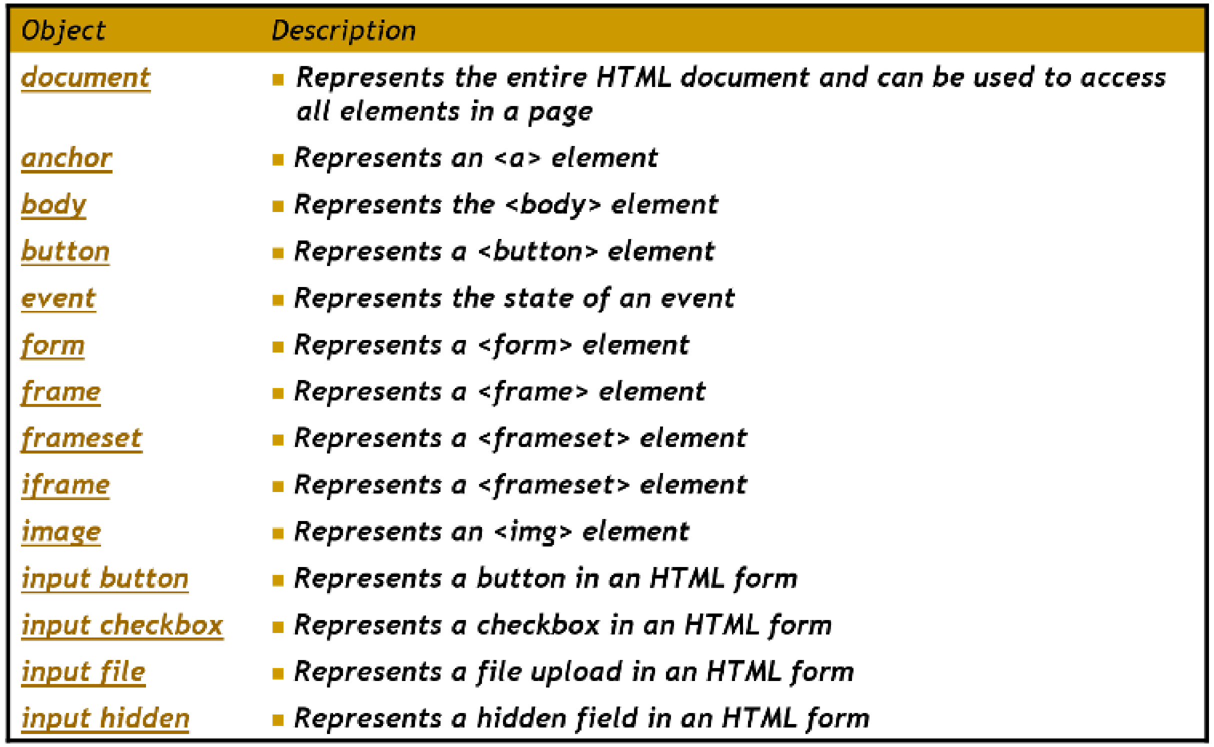


Object	Description
Window	▪ It is the top level object in the JavaScript hierarchy. ▪ The Window object represents a browser window. ▪ A Window object is created automatically with every instance of a <body> tag.
Navigator	▪It contains information about the client's browser.
Screen	▪It contains information about the client's display screen.
History	▪It contains the visited URLs in the browser window.
Location	▪It contains information about the current URL.

W3Schools (no dated). *JavaScript HTML DOM Objects*. http://www.w3schools.com/js/js_obj_htmldom.asp

7.3.5. HTML DOM Objects - more

7.3.5. HTML DOM Objects - more

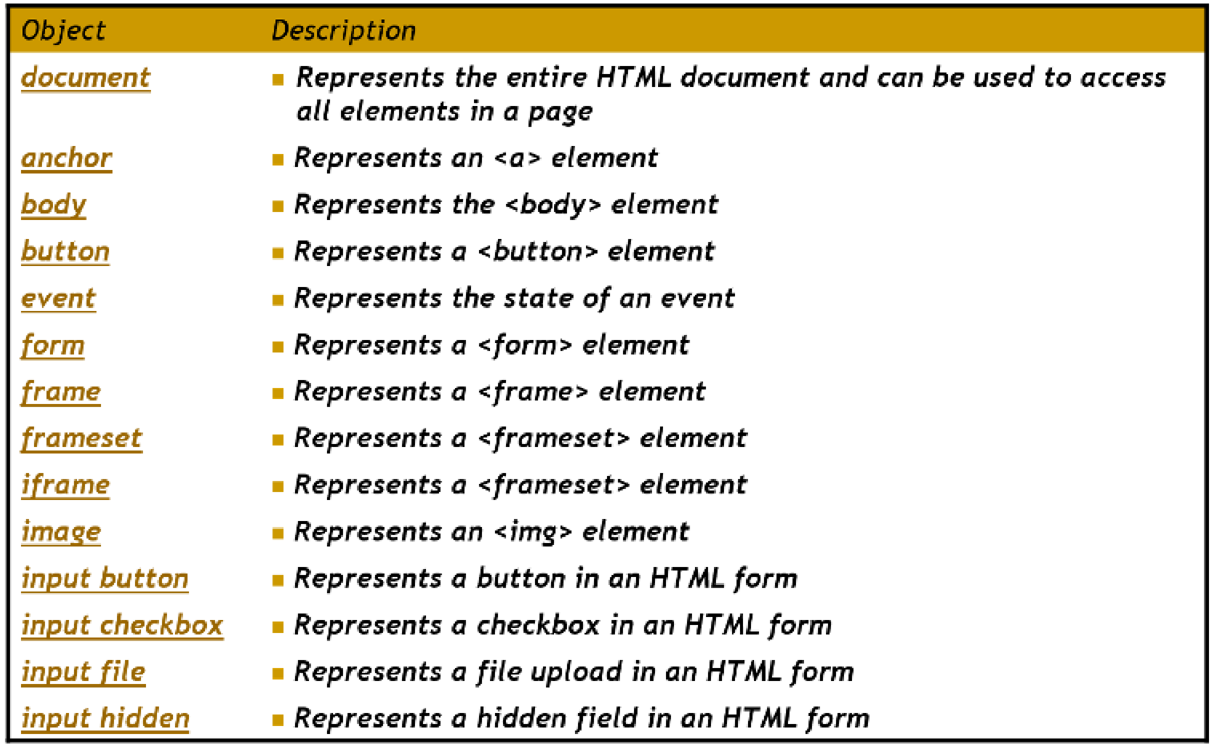


Object	Description
document	▪ Represents the entire HTML document and can be used to access all elements in a page
anchor	▪ Represents an <a> element
body	▪ Represents the <body> element
button	▪ Represents a <button> element
event	▪ Represents the state of an event
form	▪ Represents a <form> element
frame	▪ Represents a <frame> element
frameset	▪ Represents a <frameset> element
iframe	▪ Represents a <frameset> element
image	▪ Represents an <img> element
input button	▪ Represents a button in an HTML form
input checkbox	▪ Represents a checkbox in an HTML form
input file	▪ Represents a file upload in an HTML form
input hidden	▪ Represents a hidden field in an HTML form

W3Schools (no dated). *JavaScript HTML DOM Objects*. http://www.w3schools.com/js/js_obj_htmldom.asp

7.3.6. HTML DOM Objects - more

7.3.6. HTML DOM Objects - more



Object	Description
document	■ Represents the entire HTML document and can be used to access all elements in a page
anchor	■ Represents an <a> element
body	■ Represents the <body> element
button	■ Represents a <button> element
event	■ Represents the state of an event
form	■ Represents a <form> element
frame	■ Represents a <frame> element
frameset	■ Represents a <frameset> element
iframe	■ Represents a <frameset> element
image	■ Represents an <img> element
input button	■ Represents a button in an HTML form
input checkbox	■ Represents a checkbox in an HTML form
input file	■ Represents a file upload in an HTML form
input hidden	■ Represents a hidden field in an HTML form

W3Schools (no dated). *JavaScript HTML DOM Objects*. http://www.w3schools.com/js/js_obj_htmldom.asp

7.4. Event

What is an Event?

- the time at which something happens, referred to by a name
- such as :
    - after a page is loaded - referred as onload in Javascript
    - when an element is clicked - onclick

Can be referred to as:

- HTML element's attribute -
  `<h1 onclick="alert("you click me?")">Heading</h1>`
- an object's property -
  `document.querySelector("#id").onclick`

An event handler

- a piece of Javascript, usually a function
- attached to an object on the page for an event
- tell the object how to react when that event occurs

This is the 4th way to run Javascript - event-driven, other than External, Embedded and Inline Javascript.

7.4.1. Set up handler in HTML | **w3schools DOM events - https://www.w3schools.com/jsref/dom_obj_event.asp**

7.4.1. Set up handler in HTML

Assign handler Javascript code to the event attribute

```
<h1 onclick=“alert(“you click me?”);”>Heading</h1>
<img id="imgid" src="..." alt="" onclick="myHandler(inarg)">
<img id="imgid" src="..." alt="" onclick="myHandler(this.id)">
```

Can put many JS statements inside the "", although not a good practice. Usually use a function call, e.g. myHandler(inarg), therefore easy to pass arguments into the function.

Disadvantage is not easy to maintain. Need to find and edit the HTML elements in case of change.

Another commonly seen but not recommended way

```
<a href="javascript:alert('click me?');">Try click me</a>
```

7.4.2. Set up handler in Javascript | **Example - http://courses.cs.cityu.edu.hk/cs2204/example/html/32-EventHandler1.html**

7.4.2. Set up handler in Javascript

Find the object, assign a function to the event property

```
window.onload = init;
document.querySelector(".myclass").onmouseover=show
```

Note that the function does not have ( ), otherwise it becomes a direct call and executes the function. This is incorrect because in setting up handler, it should not be called at that time, only when the event occurs.

Since the () cannot be used, there is no way to pass parameters. May try to use annonymous function with variable inside the same or global scope

```
var i="before";
document.querySelector(".myclass").onmouseover=function()
{alert(i);};
i="after";
```

but using variable for passing parameter is error prone and not a good practice

Find the object, use the addEventListener() instead of event property

```
document.querySelector(".myclass").addEventListener("mouseover",
function() {alert("mouseover now");};
```

Strength & limitation :

- no need to edit HTML during setup or change
- can be used in pure Javascript applications
- by using external JS, can apply standard event handlers to common components in different Web pages
- more difficult to pass parameters to event handler
- be careful with timing problem of page load; see example below : 32-EventHandler3.html; always use

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

7.4.3. Object this | example - http://courses.cs.cityu.edu.hk/cs2204/example/html/32-EventHandler2.html | timing problem - http://courses.cs.cityu.edu.hk/cs2204/example/html/32-EventHandler3.html

7.4.3. Object this

Sometimes, the best way to pass information to the event handler is to use the object that triggers the event because the same event handler may be used for many similar objects, e.g. we are using the same event handler for a number of buttons and need to know which one has been clicked. If different id's are used then they could be used as argument to call the event handler function.

Instead of typing in different id's, access of the clicked button will be very useful. This can be done through the use of the this object. Inside the event handler

```
document.querySelector("").onclick=evthdler;
function evthdler() {
...
alert(this.id);
...
}
```

this can alert out the id (or any attribute) of the clicked object.

There is a tricky point about this method. If the event handler is set up in HTML, the this object is window instead of the clicked object! To access the object in HTML set up, pass the this object as argument explicitly

```
<img id="myid" src="..." alt="..." onclick="evthdler(this)">
function evthdler(obj) {
...
alert(obj.id);
...
}
```

7.4.4. Event Cancelling

As shown in the form reset example below, there are situations where after event handling, the event should not go on, e.g. the form should not be reset. This is known as Event Cancelling. To do this, simply return a Boolean from the event handling function. A false value will cancel the event while true will have the event continues.

Again, different ways of event handler set up would have slight difference in event cancelling

- for set up in Javascript, a return of true or false will do

  ```
  obj.onclick=function( ) {... ; return false;}
  ```

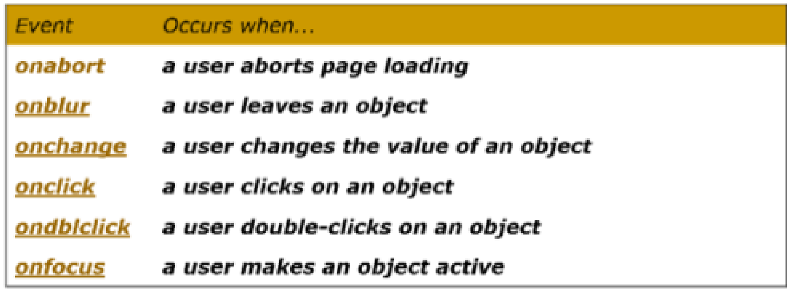
- in HTML, need to use an explicit return in the event attribute, e.g. in the form reset example

```
<form action="#" method="get" onreset="return check();";>
```

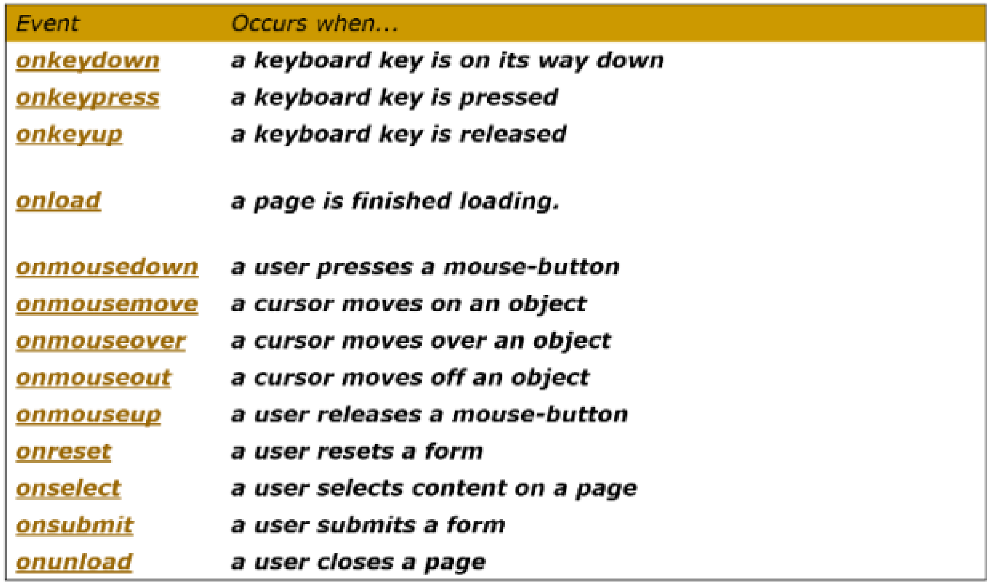
note that setting up the event handling for the form is better than that for the reset button's onclick event.

7.4.5. More events | Reset form example - http://courses.cs.cityu.edu.hk/cs2204/example/html/32-ResetForm.html

7.4.5. More events



Event	Occurs when...
onabort	a user aborts page loading
onblur	a user leaves an object
onchange	a user changes the value of an object
onclick	a user clicks on an object
ondblclick	a user double-clicks on an object
onfocus	a user makes an object active



Event	Occurs when...
onkeydown	a keyboard key is on its way down
onkeypress	a keyboard key is pressed
onkeyup	a keyboard key is released
onload	a page is finished loading.
onmousedown	a user presses a mouse-button
onmousemove	a cursor moves on an object
onmouseover	a cursor moves over an object
onmouseout	a cursor moves off an object
onmouseup	a user releases a mouse-button
onreset	a user resets a form
onselect	a user selects content on a page
onsubmit	a user submits a form
onunload	a user closes a page

| w3schools events - https://www.w3schools.com/jsref/dom_obj_event.asp

7.5. Conclusions

Extension readings:

- Text book chapter 11
- Javascript guide from Mozilla/
- http://www.w3schools.com/jsref/