Client-side Technologies

Dr. Niveen Nasr El-Den
iTi

Day 5

JavaScript Fundamentals cont.

JavaScript Built-in Objects cont.

Array Object

- Array is actually a special type of object
- Array is a data structure that used to represent list of items
- It has length property:
  - gives the length of the array
  - It is one more than the highest index in the array
- To declare an array use
  - new keyword
  - array literal notation

Array Object

- Using new operator:

  → var colorArray = new Array();

  colorArray [0]="red";

  colorArray [1]="blue";

  colorArray [2]="green";

OR

→ var colorArray = new Array(3);

colorArray [0]="red";

colorArray [1]="blue";

colorArray [2]="green";

OR

→ var colorArray = new Array("red","blue","green");

//this is called dense array where array is populated at the time it is declared

- Use array literal notation

  → var arr = ["apple", "banana", "grapes"];

  → var arr = [, 1, , , "a"];

Array Object Methods

var arr1=new Array(“A”,”B”,”C”);

var arr2 = new Array(1,2,0);

Name	Example	Result
concat	arr1.concat(arr2);	A,B,C,1,2,0 //neither arr1 nor arr2 changed
join	arr1.join() arr1.join(“*”)	A,B,C A*B*C //arr1 not changed
reverse	arr1.reverse()	C,B,A
pop	arr1.pop()	C // and arr1.length becomes 2
push	arr1.push(“D”);	4 // 4 → Length of the array // resulting in : arr1[3]=“D”

Array Object Methods

```
var  arr1=new Array("A","B","C");
var arr2 = new Array(4,2,3,0);
```

Name	Example	Result
shift	arr1.shift();	Returns: A arr1[0] ="B" & arr[1]="C"
unshift	arr1.unshift("D");	arr1[0]="D" //length become 4
slice	arr1.slice(1); arr1.slice(2);	B,C C //arr1 not changed
sort (according to Unicode)	arr2.sort()	0,2,3,4

Other Useful Methods

Method name
toReversed()
toSorted()
toSpliced()
with()
at()
fill()
flat()
indexOf()
include()

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array

Associative Array

- The Arrays That Aren't
  - JavaScript has no pure associative array.
  - Associative array is just like an ordinary array, except that instead of the indices being numbers, they're strings, hence they do not have a length property.
    - The indices are replaced by user defined keys.
  - Although the keys for an associative array have to be strings, the values can be of any data type, including other arrays or associative arrays.
  - Associative array is simply a set of key-value pairs

- The key idea is that every JavaScript object is an associative array which is the most general sort of array you can invent - sometimes this is called a hash or map structure or a dictionary object.

Associative Array

- **Example:**

```
var assocArray = new Array( );
assocArray["one"] = "one";
assocArray[“1"] = "two";
assocArray[“Next Value"] = “Three";
assocArray[“new"] = 2;

for (let i in assocArray)
    console.log(i+”:”+ assocArray[i]);
```

Objects are Associative arrays

Object Object

- Object is the parent of all JavaScript objects, which means that every object you create inherits from it
  - Reminder : the Global object is window object
- To create an object
  - var obj = { }; → preferable way
  - var obj = new Object( );
- Object object has constructor property that used to return the constructor function of the created Object.
- Objects are considered Associative Arrays also called a hash (the keys are strings)

JavaScript uses **Arrays** to represent **indexed** arrays

&

Objects to represent **associative** arrays.

Object Object

```
//old way of creating an object
var obj = new Object();
//new way of creating an object (Literal notation)
//var obj={ };
// adding property to object obj
obj.name = “JavaScript”;//dot notation → preferable approach
//obj[“name”]= “JavaScript”;// subscript notation
```

```
var obj ={
            // adding property to object obj
            name : “JavaScript”,
            //”name” : “JavaScript”,
          };
```

Example!

Date Object

- To obtain and manipulate the day and time in a script.
- The information either takes the value from the user's computer or from a specified date and time
- To create date object:
  var varName = new Date([parameters])
  - Parameters are
    - Year, month, date of the month, hour, minute, second, and milliseconds
  - Example:

    var varName = new Date()

    var varName = new Date(milliseconds)

    var varName = new Date(datestring)

    var varName = new Date(yr, month, date [, hrs, min, sec, msec])

Date Object Number Conventions

Date Attribute	Numeric Range
seconds, minutes	0 - 59
hours	0 - 23
day	0 - 6 (0 = Sunday, 1 = Monday, and so on)
date	1 - 31
month	0 - 11 (0 = January, 1 = February, and so on)
year	0 + number of years since 1900

Date Object

- **The Date object methods fall into these broad categories:**

  1. ***"get"* methods**

     → **for getting date and time values from date objects**

  2. ***"set"* methods**

     → **for setting date and time values in date objects**

  3. ***"to"* methods**

     → **for returning string values from date objects.**

Date Object “get” Methods

```
var   now = new Date ( "November 25,2009");
```

Name	Example	Returned Value
getDate	now.getDate()	25
getMonth	now.getMonth()	10
getFullYear	now.getFullYear()	2009
getDay	now.getDay()	6
getHours	now.getHours()	0
getMinutes	now.getMinutes()	0
getSeconds	now.getSeconds()	0
getTime	now.getTime()	The internal, millisecond representation of a Date object similar to now.valueOf()

Date Object “set” Methods

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

Name	Example
setDate	someDate.setDate(6)
setHours	someDate.setHours(14)
setMinutes	someDate.setMinutes(50)
setMonth	someDate.setMonth(7)
setSeconds	someDate.setSeconds(7)
setTime	someDate.setTime(yesterday.getTime())
setFullYear	someDate.setFullYear(88)

Date Object “to” Methods

var now = new Date ("November 25,2009");

Name	Example	Returned value
toUTCString	now.toUTCString()	Tue, 24 Nov 2009 22:00:00 GMT
toString	now.toString()	'Wed Nov 25 2009 00:00:00 GMT+0200 (Eastern European Standard Time)'
toLocaleString	now.toLocaleString()	11/25/2009, 12:00:00 AM
	now.toLocaleString(‘ar-EG’)	’٢٥/١١/٢٠٠٩ ١٢:٠٠:٠٠ ص’
	now.toLocaleString(‘ar-EG’,arrDate)	11/25/2009, 12:00:00 AM
toLocaleDateString	now.toLocaleDateString()	'11/25/2009'
	now.toLocaleDateString(‘ar-EG’)	‘٢٥/١١/٢٠٠٩'

[weekday: 'long', year: 'numeric', month: 'long', day: 'numeric’]

Date Object

- Hours should be specified using a 24-hour clock.
- The month is always indexed from zero, so that November is month 10.
- The year can also be offset by 1900, so that you can use either of these two forms

```
var NovDate = new Date(90, 10, 23);
        var NovDate = new Date(1990, 10, 23);
```

- For the year 2000 and beyond you must use the second form

```
var NovDate = new Date(2006, 10, 23);
```

- This form may optionally take an additional three integer arguments for the time, so that 1:05 PM on November 23, 1990 is

```
var NovDate2 = new Date(90, 10, 23, 13, 5, 0);
```

Boolean Object

- The Boolean object is used to convert a non-Boolean value to a Boolean value (true or false).
- Everything in the language is either “truthy” or “falsy”
- The rules for truthiness:
  - 0 , ”” , NaN , null , and undefined → falsy
  - Everything else → truthy
- You can convert any value to it’s boolean equivalent by applying “!!” preceding the value
  - Example:

    !!”” ➔ false

    !!123 ➔true
- To create Boolean Object
  - var b = new Boolean(); → false // typeof is Object
  - B = false → false // typeof “boolean”

Boolean Object

- All the following lines of code create Boolean objects with an initial value of false:

```
var myBoolean=new Boolean()
var myBoolean=new Boolean(0)
var myBoolean=new Boolean(null)
var myBoolean=new Boolean(undefined)
var myBoolean=new Boolean("")
var myBoolean=new Boolean(false)
var myBoolean=new Boolean(NaN)
```

- And all the following lines of code create Boolean objects with an initial value of true:

```
var myBoolean=new Boolean(true)
var myBoolean=new Boolean(1)
var myBoolean=new Boolean("false“)
var myBoolean=new Boolean(“anyThing")
```

Browser Object Model

BOM

Browser Engine & JavaScript

- **Browser engine** is a core software component of every major web browser. The primary job of a browser engine is to transform HTML documents and other resources of a web page into an interactive visual representation on a user's device.
  - e.g. Blink, Gecko, webkit etc.

- All Chromium-based browsers use Blink browser engine.
- **JavaScript engine** is a computer program that executes JavaScript (JS) code
  - e.g. V8, spiderMonkey etc.

https://en.wikipedia.org/wiki/Browser_engine

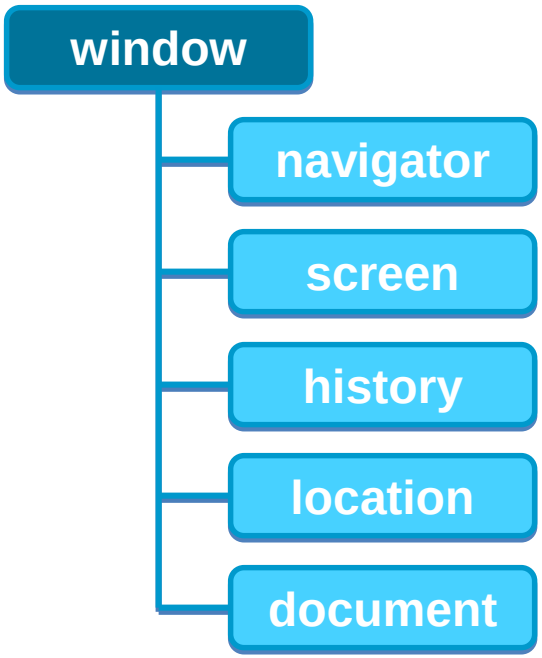
- In 2019, Microsoft announced plans to rebuild the browser as Chromium-based with Blink and V8 engines.

BOM

- BOM Stands for Browser Object Model.
- BOM covers objects which relate to the browser.
- At the top of the **BOM** hierarchy is **window** object. Below that comes the
  - **navigator** object,
  - **screen** object,
  - **history** object,
  - **location** object, and
  - **document** object
    - It is the top level of the **DOM** hierarchy.

  Each object below the window is of equal status. (comes in no particular order).

  They all relate directly to the window object.



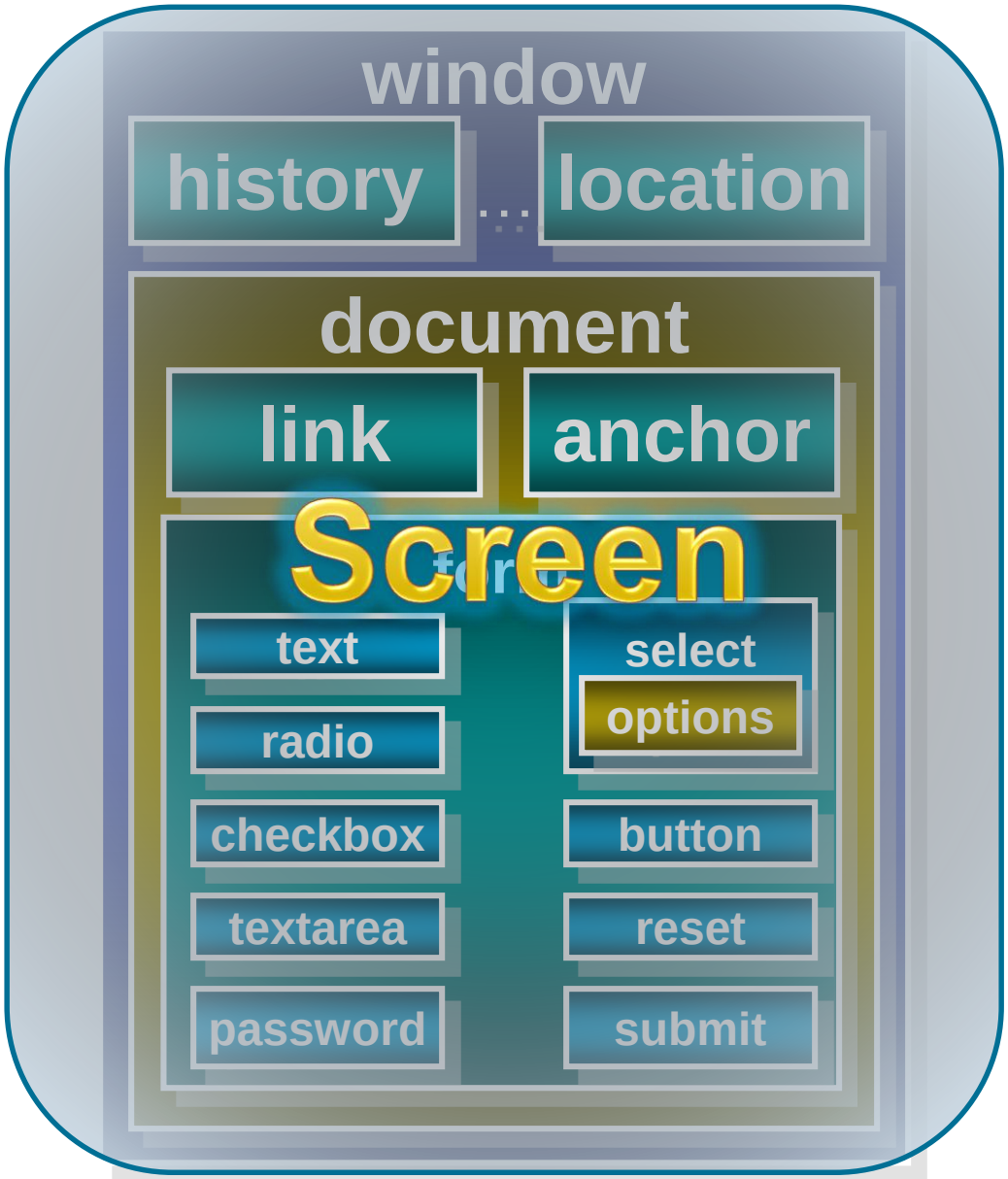
BOM

- Using the **BOM**, developers can move the window, and perform other actions that do not directly relate to the page content.
- For some reason, the **B**rowser **O**bject **M**odel is generally not referred to by its proper name. More often, it's usually wrapped up with the **DOM**.
- In actuality, the **DOM**, which relates to all things pertaining to the document, resides *within* the **BOM**.
- Because no standards exist for the BOM, each browser has its own implementation.

JavaScript Top Object Model Hierarchy

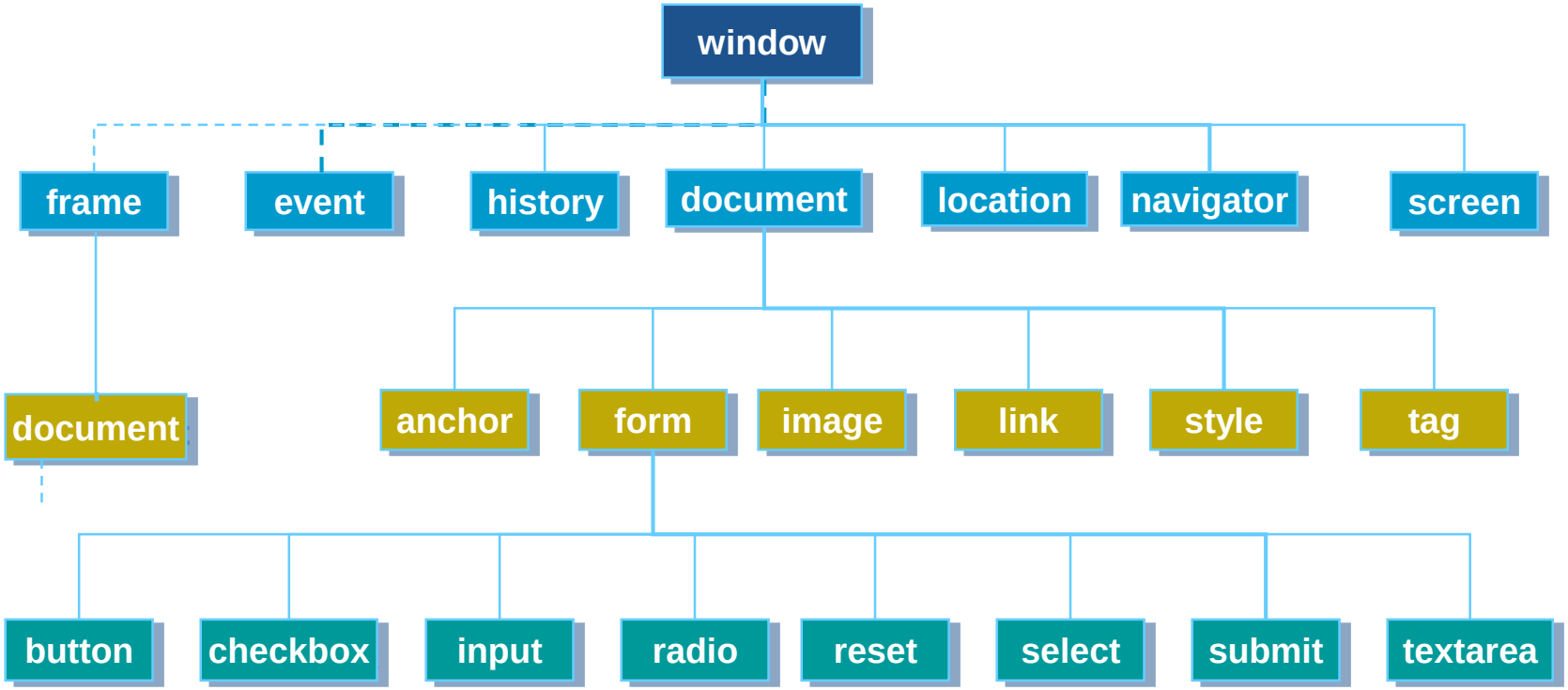
- Every page has the following objects:
  - **window** *:* the top-level object; has properties that apply to the entire window.
  - **navigator** *:* has properties related to the **name** and **version** of the Navigator being used.
  - **document** *:* contains properties based on the **content** of the document, such as title, background color, links, and forms.
  - **location** *:* has properties based on the current **URL**.
  - **history** *:* contains properties representing **URLs** the client has previously **requested**.
  - **screen** : contains information about the visitor's screen.

Browser Model



Model Hierarchy

BOM is a larger representation of everything provided by the browser including any other functionality the browser may expose to JavaScript.



Window

- Window is the top level object in the JavaScript client hierarchy.
- Window is the Global Object
- The Window object represents a browser window.
- Window object has a set of properties & methods.
- Object Model Reference:

  window
- To reference its properties & methods:
  - [window.]property
  - [window.]method

Window Properties

Name	Description	Syntax
document	Reference to the current document object.	window.document
frames	An array referencing all of the frames in the current window.	window.frames[i]
history	Reference to the History object of JavaScript	window.history
navigator	Reference to the browser application	window.navigator
location	Reference to the Location object of JavaScript	window.location

Assignments