

<script>



JavaScript & Advanced Technologies

JavaScript Built-in objects

ITI – Assiut Branch

Eng. Hany Saad

Jan 2010

Quiz...?



1. JavaScript code is interpreted by:

- a- Web server b- JavaScript Console in the browser c- Java Compiler

2. JavaScript code is executed on:

- a- Server side b- Client side

3. JavaScript Can't access any files on client PC, except cookies.

- a- True b- False

4. What's the result of executing This JS code?

```
<script>
    <Center> My First JS code </Center>
    document.write (“<b> Hello World</b>”);
</script>
```

- a- Will print Hello World in bold letters b- Error in Line 2 c- Error in line 3

Topics will be covered...



1

- JavaScript Objects

2

- JavaScript Built-in Objects

3

- String Object

4

- Number Object

5

- Array Object

6

- Date Object

7

- Math Object

8

- Boolean Object

9

- Regular expression Object

10

- Error Object

JavaScript Objects



❑ JavaScript Objects fall into 4 categories:

1. Custom Objects

- Objects that you, as a JavaScript developer, create and use.

2. Built – in Objects

- Objects that are provided with JavaScript to make your life as a JavaScript developer easier.

3. BOM Objects “**B**rowser **O**bject **M**odel”

- It is a collection of objects that are accessible through the global objects window. The browser objects deal with the characteristic and properties of the web browser.

4. DOM Objects “**D**ocument **O**bject **M**odel”

- Objects provide the foundation for creating dynamic web pages. The DOM provides the ability for a JavaScript script to access, manipulate, and extend the content of a web page dynamically.

JavaScript Built-in Objects



☐ String

☐ RegExp

☐ Number

☐ Error

☐ Array

☐ Object

☐ Date

☐ Math

☐ Boolean

String Object



❑ **Enables us to work with and manipulate strings of text.**

❑ **String Objects have:**

- **One Property:**

- `length` : gives the length of the String.

- **Methods that fall into three categories:**

1. Manipulate the contents of the String
2. Manipulate the appearance of the String
3. Convert the String into an HTML element

String Object (Cont.)



1- Methods manipulating the contents of the String

```
var myStr = "Let's see what happens!";
```

| Method | Description | Example |
|--------------------------------|---|--|
| charAt(index) | Returns the character at the specified index | <pre>document.write(myStr.charAt(0));
//L</pre> |
| indexOf(string) | Returns the position of the first found occurrence of a specified value in a string | <pre>document.write(myStr.indexOf("at"));
//12</pre> |
| lastIndexOf(string) | Returns the position of the last found occurrence of a specified value in a string | <pre>document.write(myStr.lastIndexOf("a"));
//16</pre> |
| substring(index, index) | Extracts the characters from a string, between two specified indices | <pre>document.write(myStr.substring(1, 7));
//et's s
document.write(myStr.substring(2));
//t's see what happens!</pre> |

String Object (Cont.)



1- Methods manipulating the contents of the String (Cont.)

```
var myStr = "Let's see what happens!";
```

| Method | Example | Returned value |
|------------------------|--|--|
| replace(string) | Searches for a match between a substring (or regular expression) and a string, and replaces the matched substring with a new substring | <code>document.write(myStr.replace(/e/, "?"));</code>
<code>//L?t's see what happens!</code> |
| | | <code>document.write(myStr.replace(/e/g, "?"));</code>
<code>//L?t's s?? what happ?ns!</code> |
| concat(string) | Joins two or more strings, and returns a copy of the joined strings | <code>document.write(myStr.concat(" now"));</code>
<code>//Let's see what happens! now</code> |

String Object (Cont.)



2- Methods manipulating the appearance of the String

| Method name | Example | Returned value |
|----------------------|---------------------------|--------------------------------------|
| bold() | "hi".bold() | hi |
| fontcolor() | "hi".fontcolor("green") | hi |
| fontsize() | "hi".fontsize(1) | hi |
| italics() | "hi".italics() | <I>hi</I> |
| strike() | "hi".strike() | <STRIKE>hi</STRIKE> |
| sup() | "hi".sup() | ^{hi} |
| toLowerCase() | "UPPERcase".toLowerCase() | uppercase |
| toUpperCase() | "UPPERcase".toUpperCase() | UPPERCASE |

String Object (Cont.)



2- Methods Converting the String into an HTML element

| Method name | Example | Returned value |
|---------------------|--|---------------------------------|
| link(string) | "Click me".link("linktext")
Or
myStr. link("linktext") | Click me |

❑ String Object Complete Reference:

- http://www.w3schools.com/jsref/jsref_obj_string.asp

Array Object



❑ To declare an array:

`<script>`

```
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");
```

`//OR`

```
var colorArray=[];  
var colorArray=["red","blue","green"];
```

`</script>`

❑ Array Object has One Property:

- **length** : gives the length of the array

Array Object (Cont.)



1- Object Methods

```
var arr1=new Array("A","B","C"); var arr2 = new Array("1","2","0")
```

| Methods | Description | Example |
|-----------------------|--|---|
| concat(array) | Joins two or more arrays, and returns a copy of the joined arrays | <pre>document.write(arr1.concat(arr2));
//A,B,C,1,2,0</pre> |
| join() | Joins all elements of an array into a string | <pre>document.write(arr1.join());
//A,B,C
document.write(arr1.join("*"));
//A*B*C</pre> |
| reverse() | Reverses the order of the elements in an array | <pre>document.write(arr1.reverse());
//C,B,A</pre> |
| pop() | Removes the last element of an array, and returns that element | <pre>document.write(arr1.pop());
//C, and the length becomes 2</pre> |
| push(element) | Adds new elements to the end of an array, and returns the new length | <pre>document.write(arr1.push("D"));
//4 (Length of the array)
//and arr1[3]="D"</pre> |

Array Object (Cont.)



1- Object Methods

```
var arr1=new Array("A","B","C"); var arr2 = new Array(1,2,10)
```

| Methods | Description | Example |
|------------------------------|--|---|
| shift() | Removes the first element of an array, and returns that element | <code>document.write(arr1.shift()); // A</code>
<code>//arr1[0] ="B" & arr[1]="C"</code> |
| unshift(elem
ent) | Adds new elements to the beginning of an array, and returns the new length | <code>document.write(arr1.unshift("D")); //4</code>
<code>//arr1[0]="D"</code> |
| slice(index) | Selects a part of an array, and returns the new array | <code>document.write(arr1.slice(1)); // B,C</code>
<code>document.write(arr1.slice(2)); //C</code> |
| sort() | Sorts the elements of an array alphapitacally | <code>document.write(arr2.sort()) ;//1,10,2</code> |

❑ Array Object Complete Reference:

- http://www.w3schools.com/jsref/jsref_obj_array.asp

Array Object (Cont.)



❑ Associative Arrays: The Arrays That Aren't

- Associative arrays provide let you specify key-value pairs.
- Associative array is just like an ordinary array, except that instead of the indices being numbers, they're strings, which can be a lot easier to remember and reference:
- 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.
- **Syntax:**

```
<script>
```

```
var assocArray = new Array( );  
assocArray["Ahmed"] = "Excellent";  
assocArray["Tareq"] = "pass";
```

```
</script>
```

Date Object



❑ To obtain and manipulate the date and time in a script.

- **Syntax:**

```
<script>
```

```
var myDate= new Date() // holds current date
```

```
//var myDate = new Date(datestring)
```

```
var myDate = new Date("October 13, 1975 11:13:00") ;
```

```
//var myDate = new Date(yr, mon, day)
```

```
var myDate = new Date(79,5,24);
```

```
//var myDate = new Date(yr, mon, day, hrs, min, sec)
```

```
var myDate = new Date(79,5,24,11,33,0);
```

```
</script>
```

Date Object (Cont.)



□ 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 | 1900
90
2000 |

Date Object (Cont.)



❑ 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 (Cont.)



1. get Methods

```
var now = new Date ( "November 25,2006 11:13:55");
```

| Name | Example | Returned Value |
|---------------------|------------------|----------------|
| getDate() | now.getDate() | 25 |
| getMonth() | now.getMonth() | 10 |
| getYear() | now.getYear() | 2006 |
| getDay() | now.getDay() | 0 |
| getHours() | now.getHours() | 11 |
| getMinutes() | now.getMinutes() | 13 |
| getSeconds() | now.getSeconds() | 55 |
| getTime() | now.getTime() | 11:13:55 |

Date Object (Cont.)



2. set Methods

```
var someDate = new Date ();  
var now = new Date ( "November 25,2006 11:13:55");
```

| Name | Example |
|---------------------|---------------------------------|
| setDate(number) | someDate.setDate(25) |
| setHours(number) | someDate.setHours(14) |
| setMinutes(number) | someDate.setMinutes(50) |
| setMonth(number) | someDate.setMonth(7) |
| setSeconds(number) | someDate.setSeconds(7) |
| setTime(TimeString) | someDate.setTime(now.getTime()) |
| setYear(number) | someDate.setYear(88) |

Date Object (Cont.)



3. to Methods

```
var now = new Date ( "November 25,2006 11:13:00");
```

| Name | Example | Returned Value |
|--|-------------------------------|---|
| toGMTString()
toUTCString() | now.toGMTString() | Sat, 25 Nov 2006 09:13:00 UTC |
| toLocaleString() | now.toLocaleString() | 25 نوفمبر, 2006 11:13:00 ص
(Based on date format in your OS) |
| toLocaleTimeString() | now.toLocaleTimeString()
) | 11:13:00 ص |
| toLocaleDateString() | now.toLocaleDateString() | 01 نوفمبر, 2006 |
| toString() | now.toString() | Sat Nov 25 11:13:00
UTC+0200 2006 |
| toDateString() | now.toDateString() | Sun Nov 1 2006 |

□ Date Object Complete Reference:

- http://www.w3schools.com/jsref/jsref_obj_date.asp

Math Object



☐ Allows you to perform common mathematical tasks.

☐ The Math object is a static object.

☐ Math object has:

- Properties (constant values)
- Methods

☐ Example:

```
var circleArea = Math.PI * radius * radius;
```

Math Object(Cont.)



1. Math Object Constants

| Name | Returned value |
|---------------------|-------------------------------------|
| Math.E | Returns Euler's constant |
| Math.PI | Return the value of π (PI) |
| Math.SQRT2 | Returns the square root of 2 |
| Math.SQRT1_2 | Returns the square root of 0.5 |
| Math.LN2 | Returns the natural logarithm of 2 |
| Math.LN10 | Returns the natural logarithm of 10 |
| Math.LOG2E | Returns the log base -2 of E |
| Math.LOG10E | Returns the log base -10 of E |

Math Object(Cont.)



2. Math Object Methods

| Name | Example | Returned value |
|----------------|----------------------------|-----------------------|
| cos(n) | Math.cos(.4) | 0.9210609940028851028 |
| sin(n) | Math.sin(Math.PI) | 0 |
| tan(n) | Math.tan(1.5 *
Math.PI) | infinity |
| acos(n) | Math.acos(.5) | 1.047197551196597631 |
| asin(n) | Math.asin(1) | 1.570796326794896558 |
| atan(n) | Math.atan(.5) | 0.4636476090008060935 |
| exp(n) | Math.exp(8) | 2980.957987041728302 |
| sqrt(n) | Math.sqrt(9801) | 99 |
| log(n) | Math.log(5) | 1.609437912434100282 |

Math Object(Cont.)



2. Math Class Methods(cont.)

| Name | Example | Returned value |
|---------------------|---------------------|----------------|
| max(x,y,...) | Math.max(1 , 700) | 700 |
| min(x,y,...) | Math.min(1 , 700,2) | 1 |
| pow(x,n) | Math.pow(2, 3) | 8 |
| abs(n) | Math.abs(-6.5) | 6.5 |
| random() | Math.random() | .7877896 |
| floor(n) | Math.floor(8.9) | 8 |
| ceil(n) | Math.ceil(8.1) | 9 |
| round(n) | Math.round(.567) | 1 |

❑ Math Object Complete Reference:

- http://www.w3schools.com/jsref/jsref_obj_math.asp

Boolean Object



- ❑ The Boolean object is used to convert a non-Boolean value to a Boolean value (true or false).
- ❑ 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("")  
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("true")  
var myBoolean=new Boolean("false")  
var myBoolean=new Boolean("Richard")
```

Regular expression Object (Self Study)



- ☐ Regular expressions provide a powerful way to search and manipulate text.
- ☐ A Regular Expression is a way of representing a pattern you are looking for in a string.
- ☐ A Regular Expression lets you build patterns using a set of special characters. Depending on whether or not there's a match, appropriate action can be taken.
- ☐ Regular expressions is often used for the purposes of validation.
- ☐ In the validation process; you don't know what exact values the user will enter, but you do know the format they need to use.

Regular expressions(Cont.)



❑ Pattern:

- Mandatory parameter, the regular expression you use to match text.

❑ Mode/Flag:

- Optional parameter, indicates the mode in which the Regular Expression is to be used:

- (i) ignore case.
 - (g) global search.
 - (m) Multiline

<

- Flags can be passed in any combination and/or in any order

Regular expressions(Cont.)



❑ Regular Expression syntax:

- Regular expressions can be created:

→ Explicitly using the RegExp object:

```
var searchPattern = new RegExp("pattern" [ , "flag"]);  
var searchPattern = new RegExp("j.*t", "i");
```

→ Using literal RegExp:

```
var myRegExp = / pattern / [flag] ;  
var myRegExp = /j.*t/i;
```

Mode

**Regular
Expression**

Regular expressions(Cont.)



❑ In the example above:

- `j.*t` is the regular expression pattern. It means, "Match any string that starts with j, ends with t and has zero or more characters in between".
- The asterisk `*` means "zero or more of the preceding".
- the dot `(.)` means "any character".

Regular expressions(Cont.)



❑ Regular Expression Object properties:

- **global:**
 - If this property is false, which is the default, the search stops when the first match is found. Set this to true if you want all matches.
- **ignoreCase:**
 - Case sensitive match or not, defaults to false.
- **multiline:**
 - Search matches that may span over more than one line, defaults to false.
- **lastIndex:**
 - The position at which to start the search, defaults to 0.
- **source:**
 - Contains the regexp pattern.

Regular expressions(Cont.)



❑ Regular Expression Object Methods:

○ **test()**

- returns a boolean (true when there's a match, false otherwise)
- Example:

```
var reg=/j.*t/ ;  
var t= reg.test("Javascript")
```

→false

case sensitive

○ **exec()**

- returns an array of matched strings.
- Example:

```
Var reg=/j.*t/ ig;  
Var str="Jscript is the same of javascript";  
var arr= reg.exec(str);
```

→array lenght will be 2

and arr[0]=«Jscript »

and arr[1]=« javascript »

Regular expressions(Cont.)



❑ String Methods that Accept Regular Expressions as Parameters :

○ **match()**

- returns an array of matches.

○ **search()**

- returns the position of the first match.

○ **replace()**

- allows you to substitute matched text with another string.

○ **split()**

- also accepts a RegExp when splitting a string into array elements.

Regular expressions(Cont.)



❑ RegExp Object patterns Reference:

- http://www.w3schools.com/jsref/jsref_obj_regexp.asp
- <http://regexlib.com/CheatSheet.aspx>

❑ RegExp Library:

- <http://www.regexlib.com/>

❑ Test your Regular Expression:

- <https://regex101.com/#javascript>
- <http://www.regexr.com/>
- <http://regexpal.com/>

Error Object



❑ Whenever an error occurs, an instance of the Error object is created to describe the error.

❑ Error objects can be created in 2 ways:

- Explicitly:

```
var newErrorObj = new Error();
```

- Implicitly:

➔ thrown using the throw statement.

Error Object(Cont.)



❑ Error Object Properties:

| Property | Description |
|--------------------|--|
| description | Plain-language description of error |
| fileName | URI of the file containing the script throwing the error |
| lineNumber | Source code line number of error |
| message | Plain-language description of error (ECMA) |
| name | Error type (ECMA) |
| number | Microsoft proprietary error number |

Error Object(Cont.)



❑ Error constructor:

– `var e = new Error();`

❑ Six additional Error constructor ones exist and they all inherit Error:

| | |
|-----------------------|---|
| EvalError | Raised by eval when used incorrectly |
| RangeError | Numeric value exceeds its range |
| ReferenceError | Invalid reference is used |
| SyntaxError | Used with invalid syntax |
| TypeError | Raised when variable is not the type expected |
| URIError | Raised when <code>encodeURIComponent()</code> or <code>decodeURIComponent()</code> are used incorrectly |

Using ***instanceOf*** when catching the error lets you know if the error is one of these built-in types.

Error Object(Cont.)



❑ Error Object standard Properties:

- **Name:** The name of the error constructor used to create the object
- **Example:**

```
var e = new Error('Oops');
```

- **Message:** Additional error information
- **Example:**

```
var e = new EvalError('jaavcsritp is _not_ how you spell it');  
document.write(e.name); //EvalError  
document.write(e.description); // jaavcsritp is _not_ how you
```

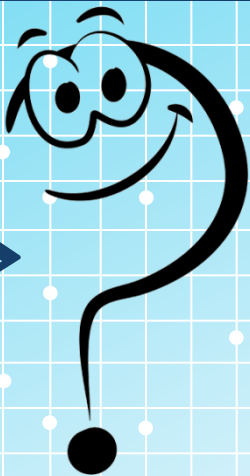
spell it

<script>



JavaScript

</script>

<SCRIPT>  </SCRIPT>

```
<script>document.writeln("Thank  
You!")</script>
```