JavaScript

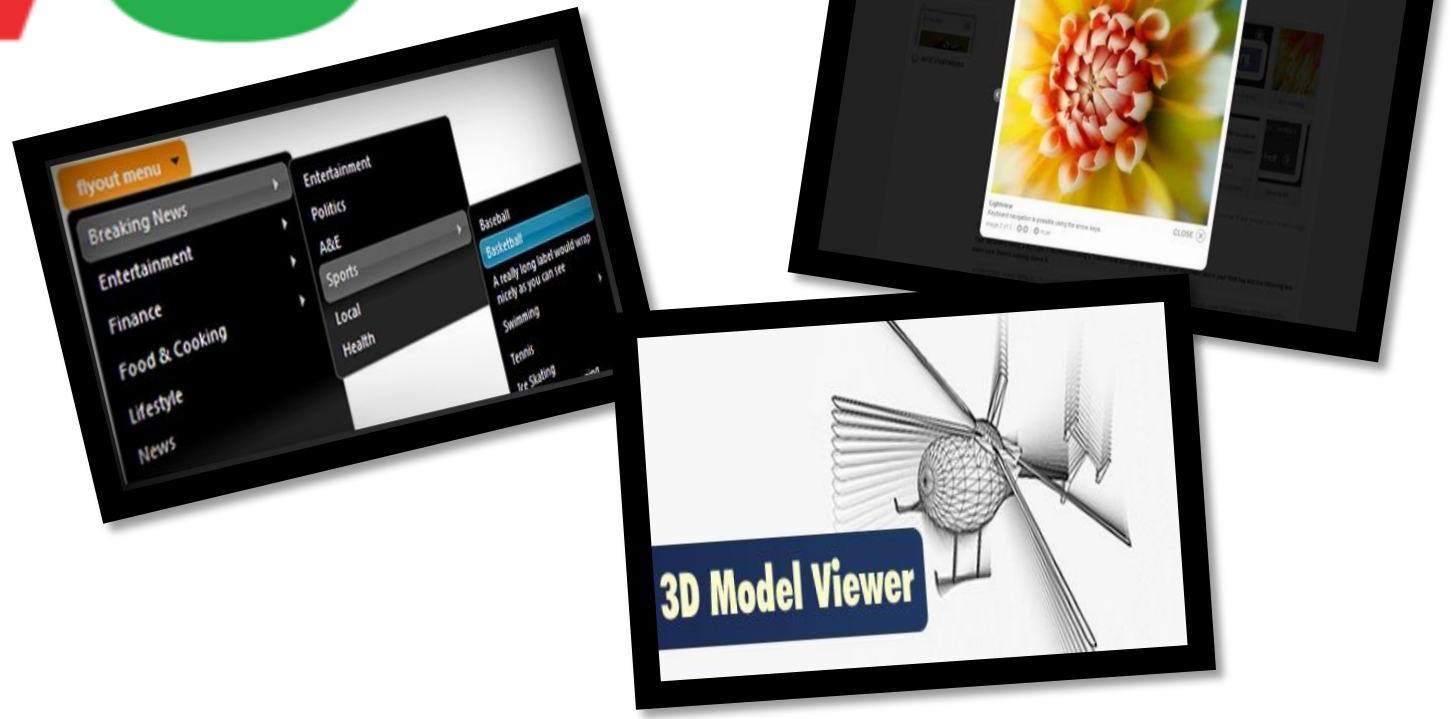
The leading client-side scripting language

Eman Fathi
Professional Development Department
Information Technology Institute

COURSE CONTENTS

- DAY 1 : CORE JS
- DAY 2 : CORE JS
- DAY 3 : BROWSER OBJECT MODEL
- DAY 4 : DOCUMENT OBJECT MODEL
- DAY 5 : OBJECT ORIENTED JS
- DAY 6 : JQUERY
- DAY 7 : JQUERY
- DAY8 : AJAX

Why?



Web application modules consist of

HTML + CSS + JavaScript

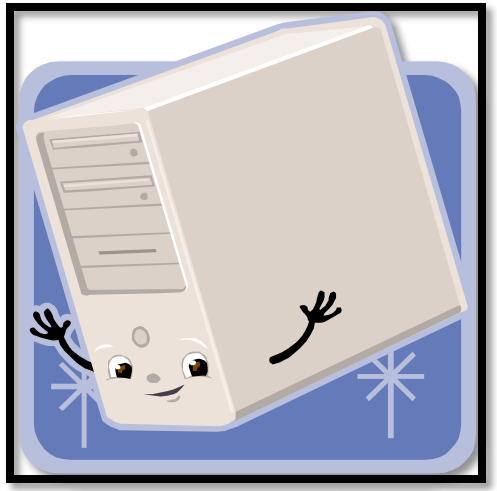
It is the language that people use without bothering to learn it first.

We need to turn them into developers who think of applications like this.

How Does the Web Work?

http

html



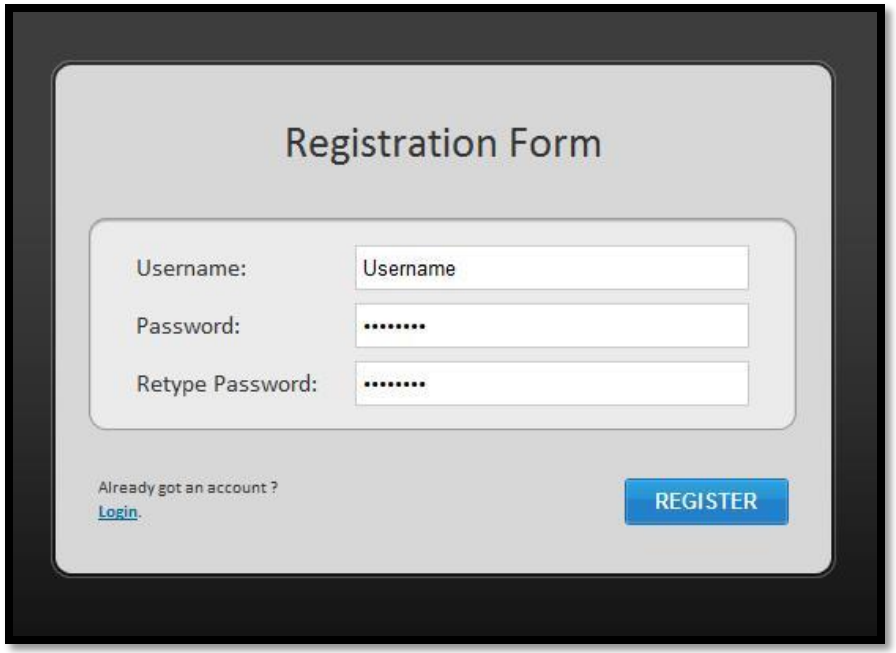
Client

Server


.html

Static Web Page

Traffic



Send Data

Send Page

If(userName==“”)

If(Pass==“”)

If(ConfPass==“”)

If(Pass==CnfPass)

JAVASCRIPT HISTORY

JAVASCRIPT HISTORY

Brendan Eich convinced his boss at Netscape that the Navigator browser should have its own **scripting language**, and that only a new language would do, a new language designed and implemented in big hurry, and that no existing language should be considered for that role.

Brendan Eich

JAVASCRIPT HISTORY

- JavaScript language first became available in the web browser Netscape Navigator 2,it was called **LiveScript**.
- Since Java was the hot technology of the time, Netscape decided that **JavaScript** sounded more exciting.
- Microsoft decided to add their own brand of JavaScript to Internet Explorer, which they named **JScript**.

ECMAScript Releases

1995 **JavaScript** is born as LiveScript

1997 **ECMAScript** standard is established

1999 **ES3** comes out and IE5 is all the rage

2000 XMLHttpRequest, a.k.a. AJAX, gains popularity

2009 **ES5** comes out and standard JSON

2015 **ES6**/ECMAScript2015 comes out

2016 **ES7**/ECMAScript2016 comes out

2017 ES.Next

JavaScript !== Java

JAVA VS. JAVASCRIPT

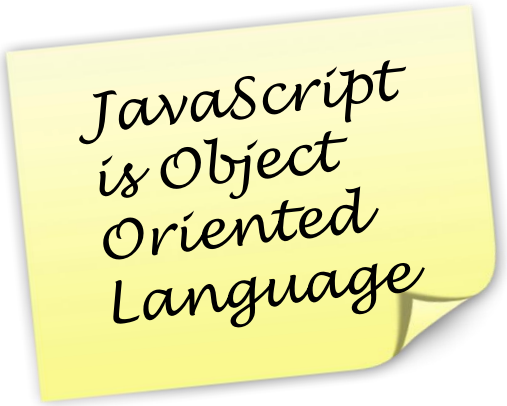
JAVASCRIPT	JAVA
Interpreted (not compiled) by client.	Compiled on server before execution on client.
Object-based. Code uses built-in, extensible objects, but no classes or inheritance.	Object-oriented. Applets consist of object classes with inheritance.
Code integrated with, and embedded in, HTML No Code Hiding.	Applets distinct from HTML (accessed from HTML pages).
Variable data types not declared (loose typing) Var.	Variable data types must be declared (strong typing).

WHAT IS JAVASCRIPT ?

- JavaScript is a **scripting** language.
- JavaScript works on the **client side**.
- JavaScript is designed for use on Web pages and is closely integrated with HTML.
- JavaScript statements embedded in an HTML page can recognize and respond to **User Events**.

JAVASCRIPT FEATURES

- Case sensitive
- Object-based
- Event-Driven
- Running inside hosting environment
- Browser-Dependent
- Interpreted language



WHAT CAN'T JAVASCRIPT DO?

- Directly access files on the user's system or the client-side LAN.
- The only exception is the access to the browser's cookie files.
- Directly access files on the Web server.

However, ***the biggest JavaScript limitation***:
the user can disable JavaScript!

HOW TO EMBED JAVASCRIPT

- We can write JavaScript

1. Anywhere in the html file between <script></script> tags.

```
<HEAD>
        <TITLE>A Simple Document</TITLE>
        <script language="javaScript">
                document.write ("Hello world")
        </script >
</HEAD>
<BODY>
        <p>Page content</p>
        <script>
                document.write (" welcome to JavaScript world")
        </script>
</BODY>
```

HOW TO EMBED JAVASCRIPT *CONT.*

2. As the value of the event handler attributes.

```
<HEAD>
     <TITLE>A Simple Document</TITLE>
</HEAD>
<BODY>
    <p> We can write it at the event handlers </p>
     <a href="try1.htm" onclick= "alert('Hello world') ">
     click here to run JavaScript code
      </a>
</BODY>
```

HOW TO EMBED JAVASCRIPT *CONT.*

3. In an external file and refer to it using the SRC attribute.

```
<HEAD>
  <TITLE>A Simple Document</TITLE>
  <script src= ”MyJavascripFile.js"></script>
</HEAD>
<BODY>
<p>We can refer to JavaScript statements in another file.</p>
</BODY>
```

VARIABLES
(DECLARATION, ASSIGNMENT & LIFETIME)

- **Declaration:**

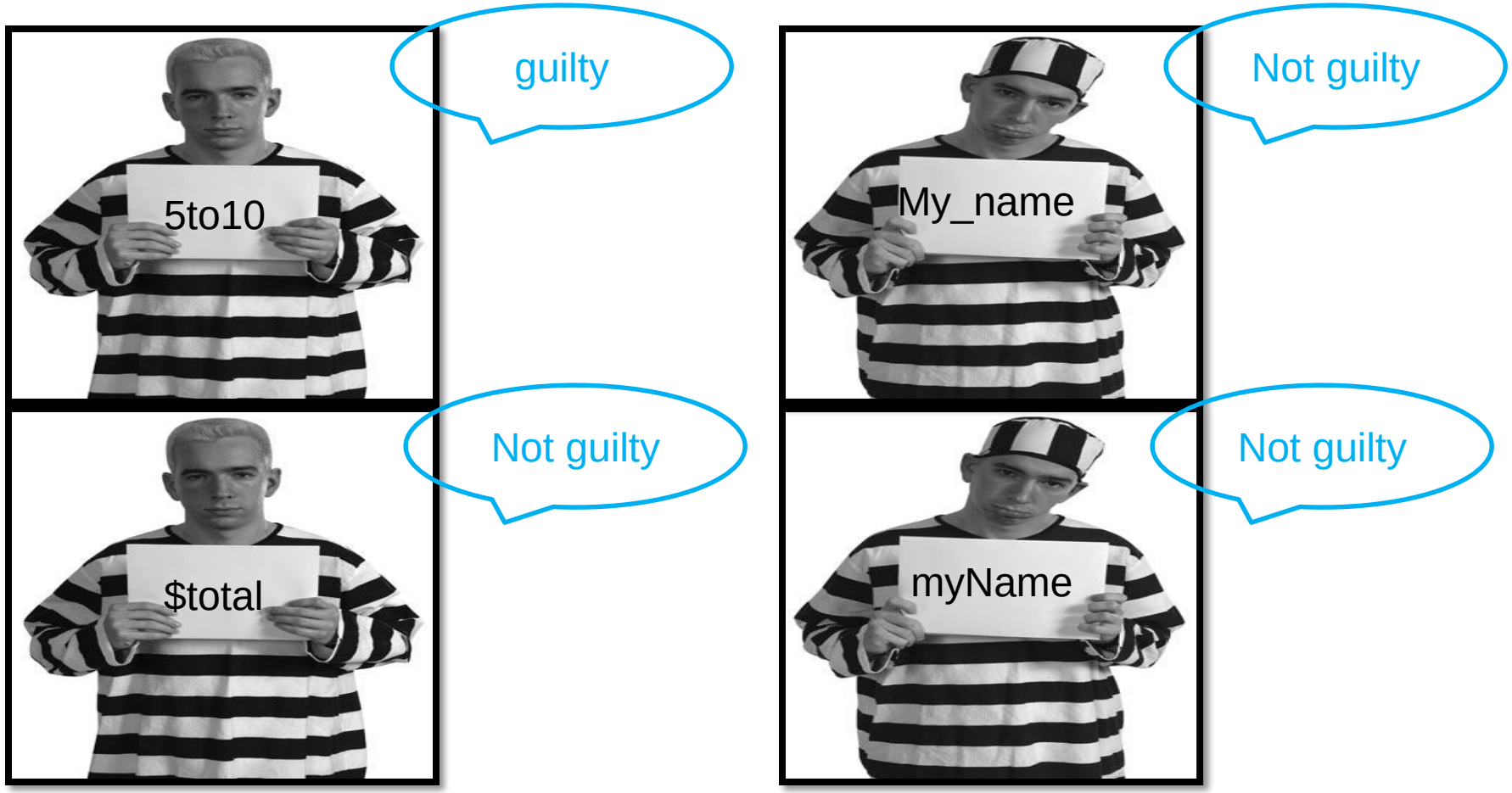
1. **Types**

- String
- Number
- Boolean
- null and undefined

VARIABLES (DECLARATION, ASSIGNMENT & LIFETIME)

- What's In a name ?
- An identifier must be at least one character in length.
- The first character in an identifier must be a letter, an underscore _ , or a dollar sign $.
- Each character after the first character can be a letter, an underscore _, a dollar sign ($), or a number.
- Spaces and special characters other than _ and $ are not allowed in any part of an identifier

VARIABLES
(DECLARATION, ASSIGNMENT & LIFETIME)



VARIABLES
(DECLARATION, ASSIGNMENT & LIFETIME)

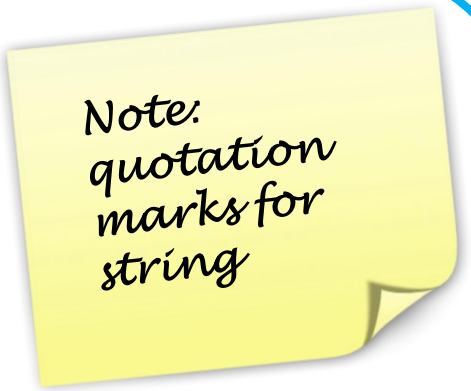
Assignment:

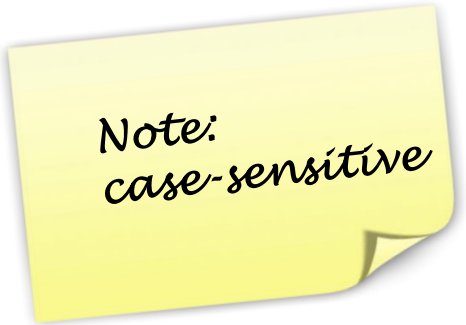
- var month = "June";

-month = "June";

-var num;
-var num = 9;

-var nextWeek;
-var NextWeek;

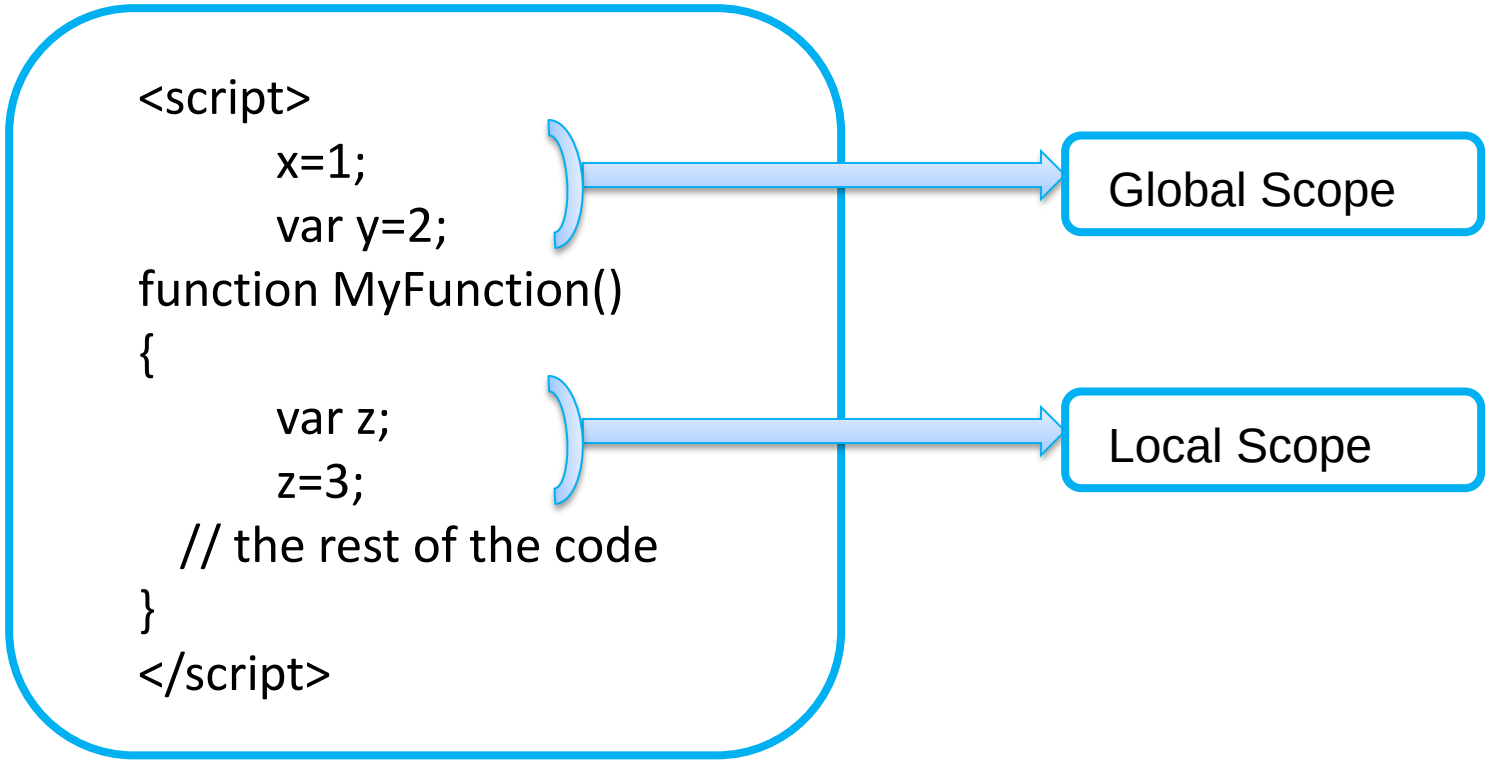




VARIABLES
(DECLARATION, ASSIGNMENT & LIFETIME)

- Lifetime:

- Local Scope
- Global Scope



USER-DEFINED FUNCTIONS

```
function dosomething(x)
{
        //statements
}

dosomething("hello");
```

Function parameters

Function call

```
function sayHi()
{
        //statements
        return "hi"
}

z= sayHi();
```

The value of z is "hi"

OPERATORS

- JavaScript supports:

  1- Binary operators:

  Require two operands in the expression such as x+2

  2- Unary operators:

  Requires one operand such as x++

  3- Ternary operators:

  Requires three operands such as (?) operator

1. ARITHMETIC OPERATORS

+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus arithmetic
++	Increment
--	Decrement

2. BITWISE OPERATORS

Bitwise AND	&
Bitwise OR	|
Bitwise XOR	^
Bitwise NOT	~
Bitwise Left Shift	<<
Bitwise Right Shift	>>

3. ASSIGNMENT OPERATORS

(x = 10 and y =5)

ASSIGNMENTS	WHAT IT DOES
x = y	Sets x to the value of y
x += y	x = x + y (15)
x -= y	x = x – y (5)
x *= y	x = x * y (50)
x /= y	x = x / y (2)
x %= y	x = x % y (0)

4. COMPARISON OPERATORS

<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Equality
!=	Inequality
===	Strict Equality
!==	Strict Inequality

5. LOGICAL OPERATORS

Operator	What it does
&&	Logical "AND" – returns true when both operands are true; otherwise it returns false
||	Logical "OR" – returns true if either operand is true. It only returns false when both operands are false
!	Logical "NOT"—returns true if the operand is false and false if the operand is true. This is a unary operator and precedes the operand

6. STRING OPERATORS

- + operator
- Example:

```
<script>
   A=“Welcome ”
   B=“Ahmed”
   C=A+B
   document.write (c)
   // the result will be “WelcomeAhmed”
</script>
```

7. SPECIAL OPERATORS

(Conditional operator)

(test_Condition) **?** if true: if false

```
<script>
        temp=120
        newvar=(temp>100) ? ”red” : “blue”
        // the value of  newvar will be “red”
        Temp=20
        newvar=(temp>100) ? ”red” : “blue”
        // the value of  newvar will be “blue”
</script>
```

CONTROL STATEMENTS

Conditional Statements

a) if….else

```
if (condition is true)
{
        do something;
}
else
{
      do something else;
}
```

b) switch / case

```
switch (expression)
{
        case label1:
                statements
        break
        case label2:
                statements
        break
        default :
}
```

CONTROL STATEMENTS

Loop Statements

a) **For**

```
for ( var i=0 ;i<10;i++)
{
    document.write("this is number" + i)
}
```

b) **while**

```
while (condition)
{
     statements
}
```

c) **do…while**

```
do
{
    statements
}while(condition)
```

JAVASCRIPT GUIDELINE *(CONT.)*

Multiple line comments preceded by /* and ended by */

→ /* This is a comment block.
It contains several lines */
document.write("Hello World!");

Single line Comment are preceded by a double-slash (//)

→ //this is a comment
document.write("Hello World!");

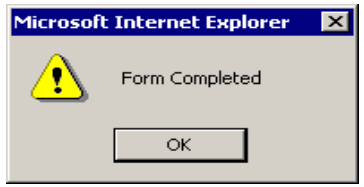
**Special characters are preceded by **

→document.write ("You \\ me are singing!");

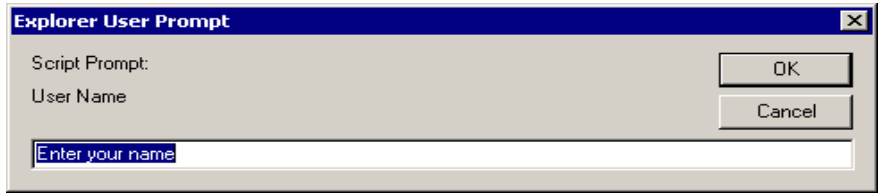
Dialog Boxes in JavaScript

DIALOG BOXES IN JAVASCRIPT

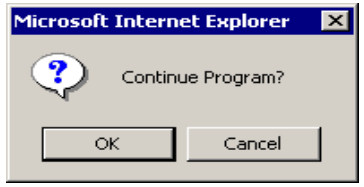
- alert dialog box



- prompt dialog box



- confirm dialog box

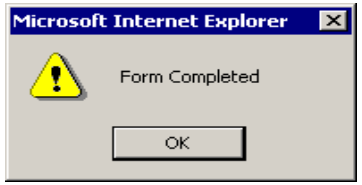


ALERT

- The simplest way to direct output to a dialog box.

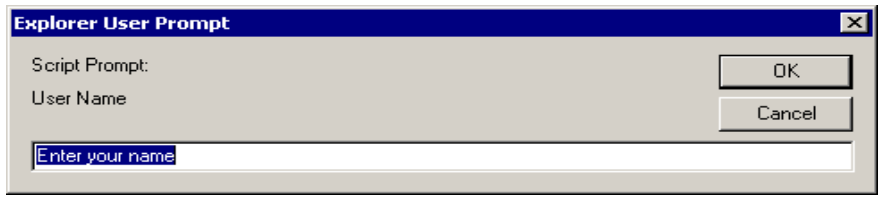
- The script
  alert("Click Ok to continue.");

  and HTML holding the script will not continue or execute until the user clicks the OK button.



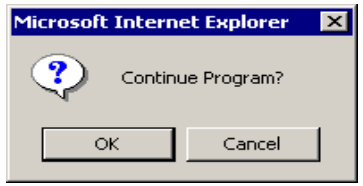
PROMPT

- The simplest way to interact with the user.
- The user needs to fill in a field and then press *OK* or *Cancel* button.
- When you press *OK* the value you typed in the field is returned.
- when you press *Cancel* the value null is returned.
- The prompt() method requires 2 parameters:
    1. text to be displayed,
    2. the default data in the entry field.



CONFIRM

- Confirm displays a dialog box with two buttons: OK and Cancel.
- If the user clicks on *OK* it will return true.
- If the user clicks on the *Cancel* it will return false.



JavaScript Built-in functions

PARSEINT() AND PARSEFLOAT(): CONVERTS TEXT TO A NUMBER

ParseInt()

```
parseInt("3")        //returns 3
parseInt(“3a”)       //returns 3
parseInt(“a3”)       //returns NaN
```

ParseFloat()

```
parseFloat("3.55")        //returns 3.55
parseFloat(“3.55a”)       //returns 3.55
parseFloat(“a3.55”)       //returns NaN
```

JAVASCRIPT OBJECTS

- Language Objects
- Browser Objects
- HTML Objects

JAVASCRIPT BUILT-IN OBJECTS (LANGUAGE OBJECTS)

- String
- Array
- Date
- Math
- Boolean

1. 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:
    - Manipulate the contents of the String
    - Manipulate the appearance of the String

1. STRING OBJECT CONT.

var myStr =new String("Let's see what happens!");

I. Methods manipulating the contents of the String

Method name	Example	Returned value
charAt	myStr.charAt(0)	L
indexOf	myStr.indexOf("at")	12
lastIndexOf	myStr.lastIndexOf("a")	16
substring	myStr.substring(0, 7)	Let's s
replace	myStr.replace(/e/,”?”)	L?t's see what happens!
	myStr.replace(/e/g,”?”);	L?t's s?? what happ?ns!
concat	myStr.concat(“ now”);	Let's see what happens! now

1. STRING OBJECT

CONT.

II. Methods manipulating the appearance of the String

Name	Example	Returned value
big	“hi".big()	<BIG>hi</BIG>
bold	"hi".bold()	<B>hi</B>
fontcolor	"hi".fontcolor("green")	<FONT COLOR="green“>hi</FONT>
fontsize	"hi".fontsize(-1)	<FONT SIZE="-1">hi</FONT>
italics	"hi".italics()	<I>hi</I>
small	"hi".small()	<SMALL>hi</SMALL>
strike	"hi".strike()	<STRIKE>hi</STRIKE>
sup	"hi".sup()	^{hi}
toLowerCase	"UPPERcase".toLowerCase()	uppercase
toUpperCase	"UPPERcase".toUpperCase()	UPPERCASE

2. ARRAY OBJECT

- To declare an array:

  → 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");

  var colorarray=["red","blue","green"];

- Array Object has:

  - One Property length : gives the length of the array

2. ARRAY OBJECT CONT.

- **Array 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
join	arr1.join() arr1.join(“*”)	A,B,C A*B*C
reverse (alphabetically)	arr1.reverse()	C,B,A
pop	arr1.pop()	C -> Length = 2
push	arr1.push(“D”);	arr1[3]=“D” -> Length = 3

2. ARRAY OBJECT CONT.

- **Array Methods**

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

Names	Example	Result
shift	arr1.shift();	Returns: A arr1[0] =“B” & arr1[1]=“C”
unshift	arr1.unshift(“D”);	arr1[0]=“D”
slice	arr1.slice(1); arr1.slice(2);	B,C C
sort (alphabetically)	arr2.sort()	0,1,2

3. 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)

3. 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.

3. DATE OBJECT CONT.

- var now = new Date ("November 25,2006");
- 1. *"get"* Methods

Name	Example	Returned Value
getDate	now.getDate()	25
getMonth	now.getMonth()	10
getYear	now.getYear()	2006
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

3. DATE OBJECT CONT.

- var now = new Date ("November 25,2006");
- 2. *" set"* Methods

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())
setYear	someDate.setYear(88)

3. DATE OBJECT

CONT.

- var now = new Date ("November 25,2006");
- 2. ″ *set*″ Methods

Name	Example	Returned value
toGMTString	now.toGMTString()	Tues, Sept 21 1999 14:28:15 GMT
toLocaleString	now.toLocaleString()	9/21/99 14:28:15

3. DATE OBJECT CONT.

- Notes:
  1. Hours should be specified using a 24-hour clock.
  2. The month is always indexed from zero, so that November is month 10.
  3. 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);
```

  4. For the year 2000 and beyond you must use the second form

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

  5. 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);
```

4. MATH OBJECT

- Allows you to perform common mathematical tasks.

- The Math object is a ***static*** *object*.

- Math object has:

I- Properties (constant values)

II- Methods

- Example:

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

4. MATH OBJECT CONT.

I. Math Object Properties

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

4. MATH OBJECT CONT.

II. Math Object Properties

Name	Example	Returned value
Abs	Math.abs(-6.5)	6.5
Acos	Math.acos(.5)	1.04719755119659763
Asin	Math.asin(1)	1.5707963267948966
Atan	Math.atan(.5)	0.4636476090008060935
Floor	Math.floor(8.9)	8
Ceil	Math.ceil(7.6)	8
Cos	Math.cos(.4)	0.9210609940028851028
Exp	Math.exp(8)	2980.957987041728302
Log	Math.log(5)	1.6094379124341002 82

4. MATH OBJECT CONT.

II. Math Object Properties

Name	Example	Returned value
max	Math.max(1 , 700)	700
min	Math.min(1 , 700)	1
pow	Math.pow(6, 2)	36
random	Math.random()	.7877896
round	Math.round(.567)	1
sin	Math.sin(Math.PI)	0
sqrt	Math.sqrt(9801)	99
tan	Math.tan(1.5 * Math.PI)	INF (infinity)

5. 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);
```

- 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(“Ali");
```

Regular Expressions

Thank You