

## Scripting the Web

# Client Side ECMAScript & Document Object Model

J0  
1

## JavaScript History

- Java was born as “**LiveScript**” at the beginning of the 94’s.
- Name changed into JavaScript (name owned by Netscape)
- Microsoft responds with Vbscript
- Microsoft introduces JScript (dialect of Javascript)
- A standard is defined: ECMAScript (ECMA-262, ISO-16262)

J0  
2

# JavaScript Myths

JavaScript is NOT simple

**Simple tasks are indeed simple**

JavaScript is NOT Java

|                 | Java | JavaScript |
|-----------------|------|------------|
| Browser Control | NO   | YES        |
| Networking      | YES  | NO         |
| Graphics        | YES  | Partial    |

J0  
3

# JavaScript is...

Scripted (not compiled)

Powerful

Object-based

Cross-Platform

Client and Server

J0  
4

## JavaScript allows...

Dynamic Web Sites  
Dynamic HTML (DHTML)  
Interactive Pages/Forms  
Server-Side CGI Functionality  
Application Development

J0  
5

## JavaScript can...

Build Objects  
Use Events  
Enforce Security  
Embed or Componentize

J0  
6

# Base

- Syntax is C-like (C++-like, Java-like)  
case-sensitive,  
statements end with (optional) semicolon ;  
//comment /\*comment\*/  
operators (=, \*, +, ++, +=, !=, ==, &&, ...)
- Basic data types  
integer, floating point, strings (more later)
- Loosely typed variables (Basic-like) `var x=3;`

JO  
7

## Statements

- `if (expression) {statements} else {statements}`
- `switch (expression) {  
    case value: statements; break;  
    ...  
    default: statements; break;  
}`
- `while (expression) {statements}`
- `do (expression) while {statements}`
- `for (initialize ; test ; increment) {statements}`

J0  
8

Between `<SCRIPT>` and `</SCRIPT>` tags

Between `<SERVER>` and `</SERVER>` tags

In a `<SCRIPT SRC="url"></SCRIPT>` tag

In an event handler:

```
<INPUT TYPE="button" VALUE="Ok"
      onClick="js code">
```

```
<B onMouseOver="Jscore">hello<B>
```

J0  
9

`a="foo"; b='tball'`

Useful methods:

`a+b => football`

`a<b => true`

`a.charAt(0) => f`

`indexOf(substring), lastIndexOf(substring)`

`charCodeAt(n),fromCharCode(value,...)`

`concat(value,...),slice(start,end)`

`toLowerCase(), toUpperCase()`

`replace(regex,string), search(regex)`

J0  
10

`a="foo";`

TAG-related methods:

`a.bold()` => `<B>foo</B>`

`big()`, `blink()`, `fontcolor()`, `fontsize()`, `small()`,

`strike()`, `sup()`

`anchor()`, `link()`



J0  
11

`function f(x) {return x*x}`

---

`function add(x,y) {return x+y};`

`function multiply(x,y) {return x*y};`

`function operate(op,x,y) {return op(x,y)};`

`operate(add,3,2); => 5`

J0  
12

Core

Example

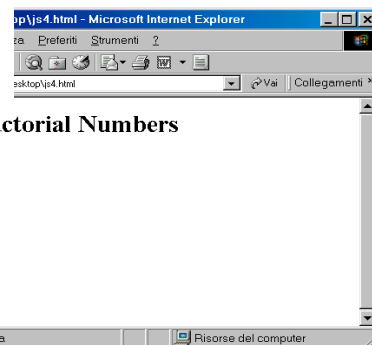
```
<HTML>
<HEAD>
<SCRIPT>
function fact(n) {
    if (n==1) return n;
    return n*fact(n-1);
}
</SCRIPT>
</HEAD>
...
```

J0  
13

Core

Example

```
<BODY>
<H2>Table of Factorial Numbers </H2>
<SCRIPT>
for (i=1; i<10; i++) {
    document.write(i+"!="+fact(i));
    document.write("<BR>");
}
</SCRIPT>
</BODY>
</HTML>
```

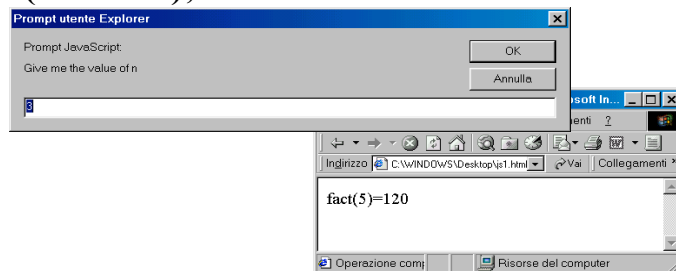


J0  
14

Core

## Example

```
<BODY>
<SCRIPT>
n=window.prompt("Give me the value of n",3)
document.write("fact("+n+")="+fact(n));
document.write("<BR>");
</SCRIPT>
</BODY>
</HTML>
```



J0  
15

Core

## Objects

**Object:** A data structure with methods;  
a special method is the “constructor”.

```
function Rectangle(w, h) {
  this.width=w;
  this.height=h;
  this.area=function() {return this.width*this.height}
}
```

`a=new Rectangle(3,4);`   `a.area() => 12`   `a.width => 3`

J0  
16

Actually, JavaScript does NOT have classes and inheritance.

Moreover, the approach we have shown is not the most efficient in terms of memory allocation.

It would be better to use the “prototype” feature, which can be considered a STATIC object

```
Rectangle.prototype.area=function(){return this.w*this.h}
```

JO  
17

```
a = new Array()
a[0]=3; a[1]="hello"; a[10]=new Rectangle(2,2);
a.length() => 11
```

Arrays can be

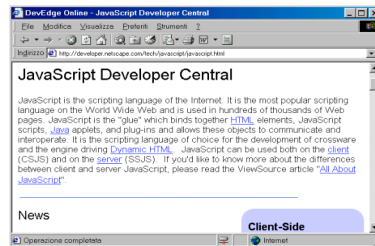
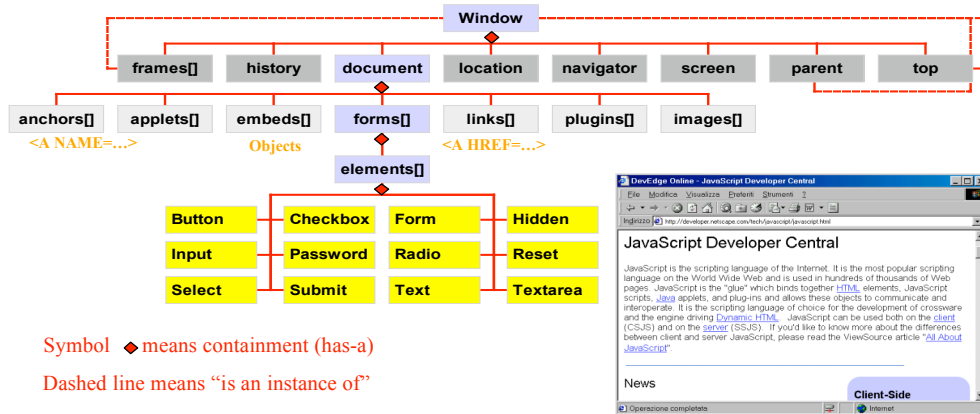
**SPARSE, INHOMOGENEOUS , ASSOCIATIVE**

```
a["name"]="Jaric"
```

```
z=new Rectangle(3,4);  z["width"] ⇔ z.width
```

JO  
18

# DOM Object hierarchy



J0  
19

## DOM Window

*"A web browser window or frame"*

### Main properties

#### Objects

history  
frames[]  
document  
location  
navigator  
screen  
parent – top

#### Other properties

status – defaultStatus  
name

J0  
20

DOM

## Window

### Main methods

alert(), prompt(), confirm()

focus(), blur()

moveBy(), moveTo()

resizeBy(), resizeTo()

scroll(), scrollBy(), scrollTo()

setInterval(), clearInterval()

setTimeout(), clearTimeout()

J0  
21

DOM

## Screen

*“Information about the display”*

### Main properties

availHeight, availWidth

height, width

colorDepth, pixelDepth

hash

J0  
22

DOM

## Navigator

*“Information about the browser in use”*

### Main properties

appName  
appVersion  
Platform

### Main methods

javaEnabled()

### Other properties

Info on available plugins, but only in Netscape Navigator!

J0  
23

DOM

## History

*“The URL history of the browser”*

### Main properties

length

### Main methods

back()  
forward()  
go(+/-n)  
go(target\_substring)

J0  
24

DOM

## Location

*“The specification of the current URL”*

### Main properties

href  
protocol, hostname, port  
search  
hash

### Main methods

reload()  
replace()

J0  
25

DOM

## Document

*“An HTML document”*

### Main properties

#### Arrays of Component Objects

anchors[]  
applets[]  
embeds[]  
forms[]  
links[]  
plugins[]

### Other properties

bgColor, fgColor, linkColor, vlinkColor  
lastModified  
title, URL, referrer, cookie

### Main methods

open()  
close()  
clear()  
write()

J0  
26

*“An image embedded in an HTML document”*

### Main properties

border *[width in pixels]*

height

width

src *[URL of the image to be displayed]*

J0  
27

|             |                                |                                      |
|-------------|--------------------------------|--------------------------------------|
| onClick     | User clicks once. (*)          | Link, button                         |
| onDbClick   | User clicks twice              | Document, Image, Link, button        |
| onMouseDown | User presses mouse button (*)  | Document, Image, Link, button        |
| onMouseUp   | User releases mouse button (*) | Document, Image, Link, button        |
| onMouseOver | Mouse moves over element       | Link, Image, Layer                   |
| onMouseOut  | Mouse moves off element        | Link, Image, Layer                   |
| onKeyDown   | User presses key (*)           | Document, Image, Link, Text elements |
| onKeyUp     | User releases key              | Document, Image, Link, Text elements |
| onKeyPress  | KeyDown+KeyUp (*)              | Document, Image, Link, Text elements |

J0  
28

(\*) Return false to cancel default action

DOM

Events

|          |                                                    |                                        |
|----------|----------------------------------------------------|----------------------------------------|
| onFocus  | Element gains focus                                | TextElement, Window, all form elements |
| onBlur   | Element loses focus                                | TextElement, Window, all form elements |
| onChange | User selects/deselects a text and moves focus away | Select, text input elements            |
| onError  | Error while loading image                          | Image                                  |
| onAbort  | Loading interrupted                                | Image                                  |
| onLoad   | Document or image finishes loading                 | Window, Image                          |
| onUnload | Document is unloaded                               | Window                                 |
| onResize | Window is resized                                  | Window                                 |
| onReset  | Form reset requested (*)                           | Form                                   |
| onSubmit | Form submission requested(*)                       | Form                                   |

JO

29

(\*) Return false to cancel default action

DOM

Input

Properties

Methods

form

length

blur()

focus()

click()

select()

Objects

Event Handlers

|            |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Button     |   |   |   | X |   | X |   |   | X | X | X | X | X |   | X |   | X | X |
| Checkbox   | X | X |   | X |   | X |   |   | X | X | X | X | X |   | X |   | X | X |
| Radio      | X | X |   | X |   | X |   |   | X | X | X | X | X |   | X |   | X | X |
| Reset      |   |   |   | X |   | X |   |   | X | X | X | X | X |   | X |   | X | X |
| Submit     |   |   |   | X |   | X |   |   | X | X | X | X | X |   | X |   | X | X |
| Text       |   |   | X | X |   | X |   |   | X | X | X |   | X | X | X | X |   | X |
| Textarea   |   |   | X | X |   | X |   |   | X | X | X |   | X | X | X | X |   | X |
| Password   |   |   | X | X |   | X |   |   | X | X | X |   | X | X | X | X |   | X |
| FileUpload |   |   | X | X |   | X |   |   | X | X | X |   | X | X | X | X |   | X |
| Select     |   |   |   | X | X | X | X | X | X |   | X | X | X |   | X | X |   | X |
| Hidden     |   |   |   | X |   | X |   |   | X | X |   |   |   |   |   |   |   |   |

Properties

checked

defaultChecked

name

options[]

selectedIndex

type

value

onblur

onchange

onclick

onfocus

DOM

## Form

*"An HTML input form"*

### Main properties

action *[destination URL]*

method *[get/post]*

name *[name of Form]*

name *[destination Window]*

### Main methods

reset()

submit()

Elements[] *[list ; of contained elements]*

J0  
31

DOM

## Events

```
<HTML>
<HEAD>
<TITLE>Form Example</TITLE>
<SCRIPT LANGUAGE="JavaScript1.2">
function setColor() {
    var choice;

    choice = document.colorForm.color.selectedIndex;

    switch(choice) {
        case 0: document.bgColor = "FF0000"; break;
        case 1: document.bgColor = "00FF00"; break;
        case 2: document.bgColor = "0000FF"; break;
        case 3: document.bgColor = "FFFFFF"; break;
        case 4: document.bgColor = "FFFF00"; break;
        case 5: document.bgColor = "FF00FF"; break;
    }
}
</SCRIPT>
```

J0  
32



DOM

## Events

```
<BODY>
<CENTER><H1>Color Changer</H1></CENTER>
<BR><BR>
Select Your Favorite Background Color:
<FORM NAME="colorForm">
<SELECT NAME="color" onChange=setColor() >
  <OPTION VALUE="red">Red
  <OPTION VALUE="blue">Blue
  <OPTION VALUE="yellow">Yellow
  <OPTION VALUE="purple">Purple
  <OPTION VALUE="green">Green
  <OPTION VALUE="white">White
</SELECT>
</FORM>

</BODY>
</HTML>
```

J0  
33

## A more complex example -1

**PERSONAL INFORMATION**

*Fields marked with an asterisk (\*) must be entered.*

\* Family Name:

Company Name:

NOTE: We replace the usual Submit button with a "Display" that acts locally, by calling some code to display what was typed in.

Please enter a Company Name    Risorse del computer

A simple data entry  
validation page

**Data Entered:**

Last Name:

Company Name:

Operazione con    Risorse del computer

J0  
34

## A more complex example -2

Start of file “FormValidation.html”

```
<HTML>
<HEAD>
<TITLE>Data Form Validation Example</TITLE>

<SCRIPT LANGUAGE="JavaScript1.1" SRC="FormCheck.js"></SCRIPT>
```

Load file “FormCheck.js”,  
which contains several JavaScript functions

J0  
35

## A more complex example -3

```
function isEmpty(s)
{ return ((s == null) || (s.length == 0))
}
```

Check that the string  
“s” is not empty

```
function warnEmpty (theField, s)
{
  var mPrefix = "You did not enter a value into the ";
  var mSuffix = " field. This is a required field. Please enter it now.";
  theField.focus();
  alert(mPrefix + s + mSuffix);
  return false;
}
```

Issue a warning  
message

All this is contained in the file “FormCheck.js”

J0  
36

## A more complex example -4

```
function promptEntry (s)
{ window.status = "Please enter a " + s;
}

function validatePersonalInfo(form)
{ return (
  checkString(form.elements["LastName"],sLastName)
)
}

function checkString (theField, s)
{
  if (isEmpty(theField.value)) return warnEmpty (theField, s);
  else return true;
}
```

Type a message in the status bar

Validate the form  
(should run over all fields  
And perform suitable checks)

Check that “theField”  
is not empty

All this is contained in the file “FormCheck.js”

J0  
37

## A more complex example -5

```
<SCRIPT>
var sCompany="Company Name"; var sLastName="Last Name"; var
form="PersonalInfo";

function displayPersonalInfo(form)
{ var outputTable = "<HTML><HEAD><TITLE>Results</TITLE></HEAD>" +
  "<BODY><H1>Data Entered:</H1><TABLE BORDER=1>" +
  "<TR><TD>" + sLastName + "</TD><TD>" + form.elements["LastName"].value +
  "</TD></TR>" +
  "<TR><TD>" + sCompany + "</TD><TD>" + form.elements["Company"].value +
  "</TD></TR></TABLE><FORM>" +
  "<INPUT TYPE='BUTTON' NAME='Back' VALUE='Back'"
  onClick='history.back()'> </FORM></BODY></HTML>"
  document.writeln(outputTable)
  document.close()
  return true
} </SCRIPT>
</HEAD>
```

Global variables

Value-printing  
function

Add a Button to go  
back in history

End of “HEAD” portion of “FormValidation.html”

J0  
38

## A more complex example -6

```
<BODY BGCOLOR="#ffffff">
<CENTER><H2>PERSONAL INFORMATION </H2></CENTER>
<P><P><I>Fields marked with an asterisk (*) must be entered.</I>
<FORM NAME="PersonalInfo">
<TABLE>
<TR>
<TD>* Family Name:</TD>
<TD><INPUT TYPE="text" NAME="LastName"
onFocus="promptEntry(sLastName)"
onChange="checkString(this,sLastName)" ></TD>
</TR>
<TR>
<TD>Company Name:</TD>
<TD><INPUT TYPE="text" NAME="Company"
onFocus="promptEntry(sCompany)"></TD>
</TR>
```

First Field

Second Field

Start of "BODY" portion of "FormValidation.html"

J0  
39

## A more complex example -7

```
<TR>
<TD>
<INPUT TYPE="BUTTON" NAME="fakeSubmit" VALUE="Display"
onClick="if (validatePersonalInfo(this.form)) displayPersonalInfo(this.form); ">
</TD>
<TD><INPUT TYPE = "reset" VALUE = "Reset">
</TD>
</TR>
</TABLE>
<P> NOTE: We replace the usual Submit button with a "Display" that acts locally,
<BR>by calling some code to display what was typed in.
</FORM>
</BODY>
</HTML>
```

First Button

Second Button

End of file "FormValidation.html"

J0  
40

DOM

## Applet

*“An applet embedded in a Web page”*

### Properties

Same as the public fields  
of the Java applet

### Methods

Same as the public methods  
of the Java applet

J0  
41

## LiveConnect

A two-faced, (**Netscape-only**) technology to let JavaScript  
interact with Java, so that:

- A JavaScript script can control and coordinate Java applets, and let Java applets interact with plugins.
- A Java Applet can execute JavaScript code.

J0  
42

## Server-Side JavaScript

### Not discussed here!

A substitute for CGI.

*Server-dependent* technology to process the Web page *before* passing it to the client.

(The Netscape SSJS object model is different from the Microsoft ASP object model, although JavaScript can be used as SSLanguage for ASP)

See

[http://developer.netscape.com/viewsource/husted\\_js/husted\\_js.html](http://developer.netscape.com/viewsource/husted_js/husted_js.html)

J0  
43

## References

### Standard ECMA-262 ECMAScript Language Specification:

<http://www.ecma.ch/ecma1/STAND/ECMA-262.HTM> ()

### Tutorials and references:

- <http://developer.netscape.com/tech/javascript/javascript.html>
- <http://www.sauronsoftware.it/jsreference>

### Books:

- D.Flanagan “Javascript. The definitive guide” O’Reilly.
- D.Goodman “Dynamic HTML. The definitive reference” O’Reilly

J0  
44

## Java-JavaScript interaction: JSObject

*JSObject* allows Java to manipulate objects that are defined in JavaScript.

Values passed from Java to JavaScript are converted as follows:

*JSObject* is converted to the original JavaScript object.

Any **other Java object** is converted to a JavaScript wrapper, which can be used to access methods and fields of the Java object.

Converting this wrapper to a string will call the **toString** method on the original object, converting to a number will call the **floatValue** method if possible and fail otherwise.

Converting to a boolean will try to call the **booleanValue** method in the same way.

**Java arrays** are wrapped with a JavaScript object that understands *array.length* and *array[index]*.

A Java **boolean** is converted to a JavaScript boolean.

Java **byte, char, short, int, long, float, and double** are converted to JavaScript numbers.

J0  
45

**Note** If you call a Java method from JavaScript, this conversion happens automatically--you can pass in "int" argument and it works.

## Java-JavaScript interaction: JSObject

Values passed from JavaScript to Java are converted as follows:

Objects that are wrappers around Java objects are unwrapped.

**Other objects** are wrapped with a *JSObject*.

**Strings, numbers, and booleans** are converted to String, Float, and Boolean objects respectively.

Examples

(String) window.getMember("name")

(JSObject) window.getMember("document")

J0  
46

## Java-JavaScript interaction: JSObject

The netscape.javascript.JSObject class has the following **methods**:

| Method              | Description                                        |
|---------------------|----------------------------------------------------|
| <b>Call</b>         | Calls a JavaScript method                          |
| <b>Eval</b>         | Evaluates a JavaScript expression                  |
| <b>getMember</b>    | Retrieves a named member of a JavaScript object    |
| <b>getSlot</b>      | Retrieves an indexed member of a JavaScript object |
| <b>removeMember</b> | Removes a named member of a JavaScript object      |
| <b>setMember</b>    | Sets a named member of a JavaScript object         |
| <b>setSlot</b>      | Sets an indexed member of a JavaScript object      |
| <b>toString</b>     | Converts a JSObject to a string                    |

The netscape.javascript.JSObject class has the following **static methods**:

**getWindow** Gets a JSObject for the window containing the given applet

J0  
47

## Java-JavaScript interaction: JavaScript side

```
<HTML>
<head>
  <script>
    function g(x){return x+x}
    function f(){return "Called f()";}
  </script>
</head>
<body>
  <script type="text/JavaScript">
    <!-- //hide script from old browsers
    document.write("<h2>JavaScript is enabled.</h2>")
  // end hiding contents from old browsers -->
  </script>
  <noscript><h2>JavaScript is not enabled, or your browser has
  restricted this file from showing active
  content.</h2></noscript>
```

J0  
48

## Java-JavaScript interaction: JavaScript side

```
<script>
  <!-- //hide script from old browsers
  var jEnabled = navigator.javaEnabled();
  if (jEnabled){
    document.write("<h2>JAVA is enabled.</h2>")
  }else{
    document.write("<h2>JAVA is <i>NOT</i> enabled.</h2>")
  }
  // end hiding contents from old browsers -->
</script>

<APPLET code="javascript.MyApplet.class" name="app"
codebase="classes/" align="baseline"
width="200" height="200"
MAYSCRIPT>
<PARAM NAME="param" VALUE="1">
  If your browser is blocking the content, please click on the
  bar above.
</APPLET>
```

J0  
49

## Java-JavaScript interaction: JavaScript side

```
<script language="Javascript">
document.write(f());
</script>
<script language="Javascript">
document.write(app.comment);
app.r=255;
document.write(app.square("3"));
</script>
<form>
  <input name="ChangeColorButton" value="Change color"
type="button"
onclick="app.r=(app.r+100)%256;app.repaint()"/>
</form>
<form>
  <input title="writeButton" value="write on console"
type="button" onclick='java.lang.System.out.println("a java
message");'/>
</form>
</body>
</html>
```

J0  
50

## Java-JavaScript interaction: applet side

```
package javascript;
import netscape.javascript.*;
import java.applet.*;
import java.awt.*;
public class MyApplet extends Applet {
    private JSObject mainWindow;
    private JSObject pageDoc;
    private JSObject location;

    private String s;
    public String comment="instanceVarContent";
    public void init() {
        System.out.println("initing");
        mainWindow = JSObject.getWindow(this);
        pageDoc = (JSObject) mainWindow.getMember("document");
        location = (JSObject) mainWindow.getMember("location");
        s = (String) location.getMember("href"); // document.location.href
    }
}
```

J0  
51

## Java-JavaScript interaction: applet side

```
public int r=0;
public int g=255;
public int b=0;

public void start(){
    s=(String)mainWindow.call("f",null);
    String[] stringArgs = new String[1];
    stringArgs[0] = "5";
    s=(String)mainWindow.call("g", stringArgs);
    System.out.println (" Calling g returned "+s);
}

public void paint(Graphics gra) {
    if (s==null) s="NULL";
    gra.setColor(new Color(r,g,b));
    Dimension d=this.getSize();
    gra.fillRect(0,0,d.width,d.height);
    gra.setColor(new Color(0,0,0));
    gra.drawString("VERSION 1",80,80);
    gra.drawString(s,30,30);
}
```

J0  
52

## Java-JavaScript interaction: applet side

```
void changeColor(String s) {  
    int x=Integer.parseInt(s);  
    r=x;  
    this.repaint();  
}  
  
public String square(String sx) {  
    int x=Integer.parseInt(sx);  
    return new Integer(x*x).toString();  
}  
  
}
```

J0  
53