

# Introduction to JavaScript

Prof. Ing. Andrea Omicini  
II Facoltà di Ingegneria, Cesena  
Alma Mater Studiorum, Università di Bologna  
[andrea.omicini@unibo.it](mailto:andrea.omicini@unibo.it)

# Documents and computation

## 👁 HTML

- 👁 Language for the description of documents
- 👁 Information-oriented
- 👁 Document mobility
  - 👁 Distributed information

## 👁 How to distribute computation using the Web?

- 👁 Associating mobile code to HTML pages
  - 👁 Applet Java
  - 👁 JavaScript

# JavaScript vs. Java Applet

- 👁 Specialisation on the "client as browser" model
- 👁 Dynamics
- 👁 "Lightness"
- 👁 Regular Expressions agile management
  - 👁 Perl-like
- 👁 Weakly typed
  - 👁 easy prototyping
- 👁 Inheritance and objects
  - 👁 prototype vs. class
- 👁 ...

# Myths

- 👁 JavaScript is similar to Java
  - 👁 A little...
- 👁 JavaScript is simple
  - 👁 It is easily usable without training
- 👁 JavaScript runs on every browser
  - 👁 Yes, of course, thankyouverymuch
    - 👁 but it can have specific quirks on specific browsers
  - 👁 Versions, IE vs Mozilla (Netscape) vs Opera vs ...
  - 👁 ECMA
    - 👁 <http://www.ecma-international.org/>

# Standard

- 👁 ECMA 262

  - 👁 ISO 16262

  - 👁 ECMAScript

    - 👁 JavaScript, Jscript

<http://www.ecma-international.org/publications/standards/ECMA-262.HTM>

<http://www.ecma-international.org/publications/files/ecma-st/ECMA-262.pdf>

- 👁 ECMA 357

  - 👁 E4X

    - 👁 ECMAScript for XML

<http://www.ecma-international.org/publications/standards/ECMA-357.HTM>

<http://www.ecma-international.org/publications/files/ecma-st/ECMA-357.pdf>

# JavaScript

- 👁 Object-oriented / Functional language
  - 👁 Model
  - 👁 Syntactic details
- 👁 Client side
  - 👁 Browser integration
- 👁 Server side
  - 👁 We are not interested
- 👁 Embedded
  - 👁 I have not heard about that in a long time...

# Example – XHTML

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
  <head>
    <meta http-equiv="content-type" content="text/html; charset=utf-8" />
    <title>...</title>
    <link href="style.css" rel="stylesheet" type="text/css" media="screen" />
    <script type="text/javascript" src="command.js"></script>
  </head>
  <body class="papers">
    ...
    <form action="" method="get">
      ...
      <input type="button" value="BibTeX" class="bibBtn" onclick="showBib('volume');">
      ...
```

# Example – JavaScript

```
absURL = "abs/";  
bibURL = "bib/";  
pdfURL = "pdf/";  
  
function showAbs(key) {  
    abstractWin=window.open(absURL+key+".html", "abstractWindow",  
"resizable=yes,dependent=yes,height=150,width=600,location=no,menubar=no  
,scrollbars=yes,status=no,toolbar=no");  
    abstractWin.focus();  
}  
function showBib(key) {  
    bibtexWin=window.open(bibURL+key+".html", "bibtexWindow",  
"resizable=yes,dependent=yes,height=300,width=600,location=no,menubar=no  
,scrollbars=yes,status=no,toolbar=no");  
    bibtexWin.focus();  
}  
function showPDF(key) {  
    top.location.href=pdfURL+key+".pdf";  
}  
...
```

# What does JavaScript do?

- 👁 Document content and presentation control
  - 👁 The document object
  - 👁 DOM
- 👁 Browser control
  - 👁 The window object
- 👁 Form management
  - 👁 The Form, Button, ... objects
- 👁 User interaction
  - 👁 Events management
  - 👁 Interaction state management
    - 👁 Cookies

# Structure of the language

- Case sensitive
  - It is a problem using HTML
- Separators
  - Spaces, line breaks, tabs, ...
- Semicolon
  - Optional, but please use it
- Comments
  - Similar to C, C++ e Java
  - Use `//` for single line and `/* ... */` for multiline
- Keywords

# Data types

- 👁 Primitive types
  - 👁 Number, string and boolean
- 👁 Objects
  - 👁 General e special
    - 👁 window, document, Data, RegExp, ...
- 👁 Array
- 👁 Functions
- 👁 E4X adds XML-like data types

# Numbers

- Integer and real numbers as IEEE 8 byte
  - Only double-precision numbers
- The `Math` object
  - Library of mathematical functions
- Special values
  - `Infinity`
  - `NaN`
  - ...

# Strings

- No char type
- Quotes and double quotes
  - they are equal
  - pay attention with (X)HTML
- Concatenation
  - and many other “classic” operators
  - Wrapper `string`
    - Virtual “library”, à la Java (static functions)

# Boolean

- `false` and `true`
  - As strings
- Automatically converted in 0 and 1
  - Numbers
  - Whenever needed...

# Primitive types and references

- 👁 Assignment

- 👁 Between non-primitive types
  - 👁 References are shared

- 👁 Example

```
var a = [1,2,3];  
var b = a;  
a[0] = 99;  
alert(b);
```

- 👁 what does that do?

- 👁 Try it! (IE, Mozilla, Opera, Safari/Konqueror)

```
javascript: var a = [1,2,3]; var b = a; a[0] = 99; alert(b);
```

- 👁 what is the output?

# Variables and scope

- 👁 Keyword `var`
  - 👁 Used or not...
- 👁 Scopes
  - 👁 Global
    - 👁 Global object
  - 👁 Local
    - 👁 Execution context
    - 👁 No blocks
- 👁 Web
  - 👁 documents and windows are new context in addition to "classic" scopes

# Expressions and operators

- We won't present them
  - Similar to C, C++, Java, more or less...
  - Please help yourself...
- There's all sort of them
- Note
  - `typeof`
    - a kind of "reflexive" operator

# Control instructions

## • Selection

- `if, if/else, else if`

- `switch`

## • Iteration

- `while, do/while, for, for/in`

## • Function

- `function, return`

- ...

# Functions

- 👁 First class objects

- 👁 Parameters

- 👁 Lambda expression, closures

- 👁 Examples

```
function square(x) {return x*x;}  
var square = new Function("x", "return x*x;");  
var square = function(x) {return x*x;};
```

- 👁 Function objects and properties

- 👁 The "call" object

- 👁 arguments, caller

- 👁 length and arity

- 👁 apply and call

# Objects

- Collections of properties

  - with names

- The `new` operator

  - `var paper = new Object();`

- Definition of / access to properties

  - `paper.title = "JavaScript -- Ohboy!!!";`

- Enumeration

  - `for/in`

- Methods

  - Properties like any other

- Prototypes

  - Not (only) classes and inheritance

  - In the 3rd standard, class and prototype properties...

# Array

- As objects...

- `var arr = new Array(1,2,3,4,5);`

- Classic access

- `var four = arr[3];`

- `var arr = [[2,3],[true,false],["boh",'mah']];`

- Fragmented and dynamic

- you can do everything you want...

- Wrapper Array

# Regular Expressions

- Excellent to manage text
  - User input
- The `RegExp` object
- A lot of relevant details...
  - Ok, let's move on!

# Browser integration

- 👁 The window object
  - 👁 Window as a global execution context
    - 👁 `var foo` and `window.foo` are the same
- 👁 Client-side object hierarchy
  - 👁 The window object contains
    - 👁 `document`, `location`, `frames[]`, `forms[]`, ...
- 👁 Event model
  - 👁 Event managers associated to (X)HTML tags

# The SCRIPT tag

```
...  
<html>  
<head>  
...  
<script type="text/javascript" language="JavaScript">  
<!-- hide to very old browsers  
    javascript code  
-->  
</script>  
</head>  
...
```

# Windows management

- You can control almost everything...
  - but you need to study a little
  - so it is better to start from existing examples...
- A window objects hierarchy
  - screen, navigator, document, ...

```
function showBib(key) {
  bibtexWin=window.open(bibURL+key+".html", "bibtexWindow",
    "resizable=yes,dependent=yes,height=300,width=600,location=no,menubar=no,
    scrollbars=yes,status=no,toolbar=no");
  bibtexWin.focus();
}
function showPDF(key) {
  top.location.href=pdfURL+key+".pdf";
}
```

# DOM

- ① Standardize the Document Object Model...
  - ① It has been done in theory, but in practice there are still some problems
    - ① especially using frames
- ① To dynamically generate objects...  
`document.write(), writeln(), open(), close()`

# Events

- 👁 Event managers

- 👁 `onChange`, `onClick`, `onMouseDown`, `onSubmit`, ...

- 👁 Problem

- 👁 to define a set of common events between IE and the other browsers...

- 👁 it has been tried...

- 👁 Managers as HTML attributes

...

```
<form action="" method="get">
```

...

```
<input type="button" value="BibTeX" class="bibBtn"
onclick="showBib('volume');">
```

...

# HTML and Forms

- Every (X)HTML element can have an identifier
  - The `id` attribute (once called `name`)
- The `Form` object
  - Modules as elements of `document.forms[]`
  - Input elements as elements of `document.forms[] .elements[]`
  - Associative access using the `name/id` name
- The `onSubmit()` and `reset()` methods
  - If `onSubmit()` returns `false`, data are not sent
    - A crystal-clear example of “distributed computation”...

# Security

- 👁️ Implicit

- 👁️ No access to the local file system
- 👁️ No direct network functions

- 👁️ Explicit

- 👁️ Restricted or privilege based functionality
- 👁️ "From the same origin" rule
- 👁️ Signed script

# JavaScript in a few hours?

- Tutorial on the Internet
  - Course website
  - or <http://www.google.it>, search: JavaScript Tutorial
- Example
  - <http://www.pageresource.com/jscript/>
    - tutorial page
  - <http://academ.hvcc.edu/~kantopet/old/javascript/>
- Books
  - "JavaScript: The Definitive Guide"  
(David Flanagan, O'Reilly/Apogeo)
  - or anything you like...