

The Web, revisited

The interactive WEB

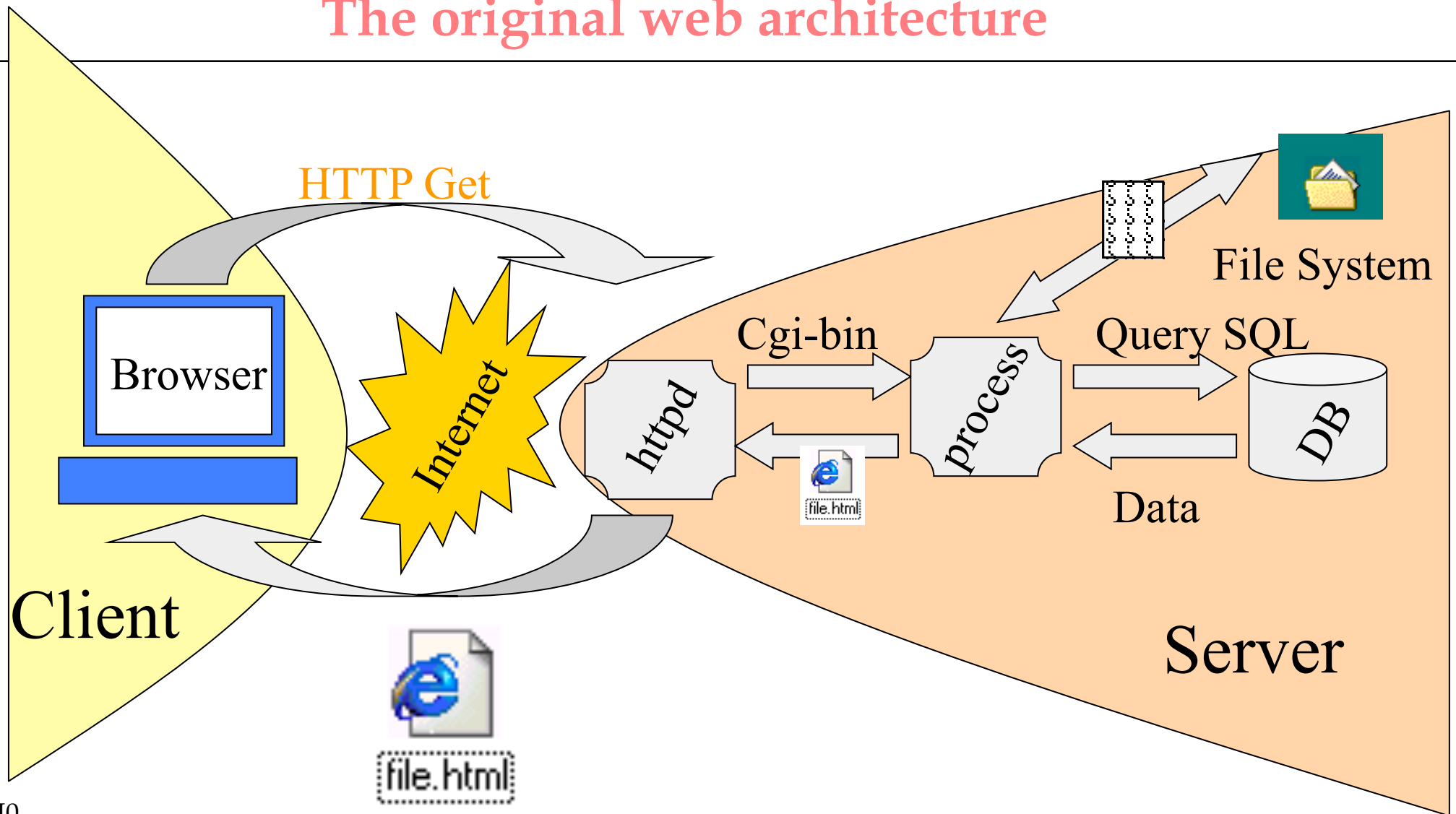
marco.ronchetti@unitn.it

The old web: 1994

HTML pages (hyperlinks)
+ static graphics
+ cgi (active engines)
+ some separated dynamic graphics
(Applets)

HTTP is a stateless protocol: cookies

The original web architecture



Evolution of the web: 1

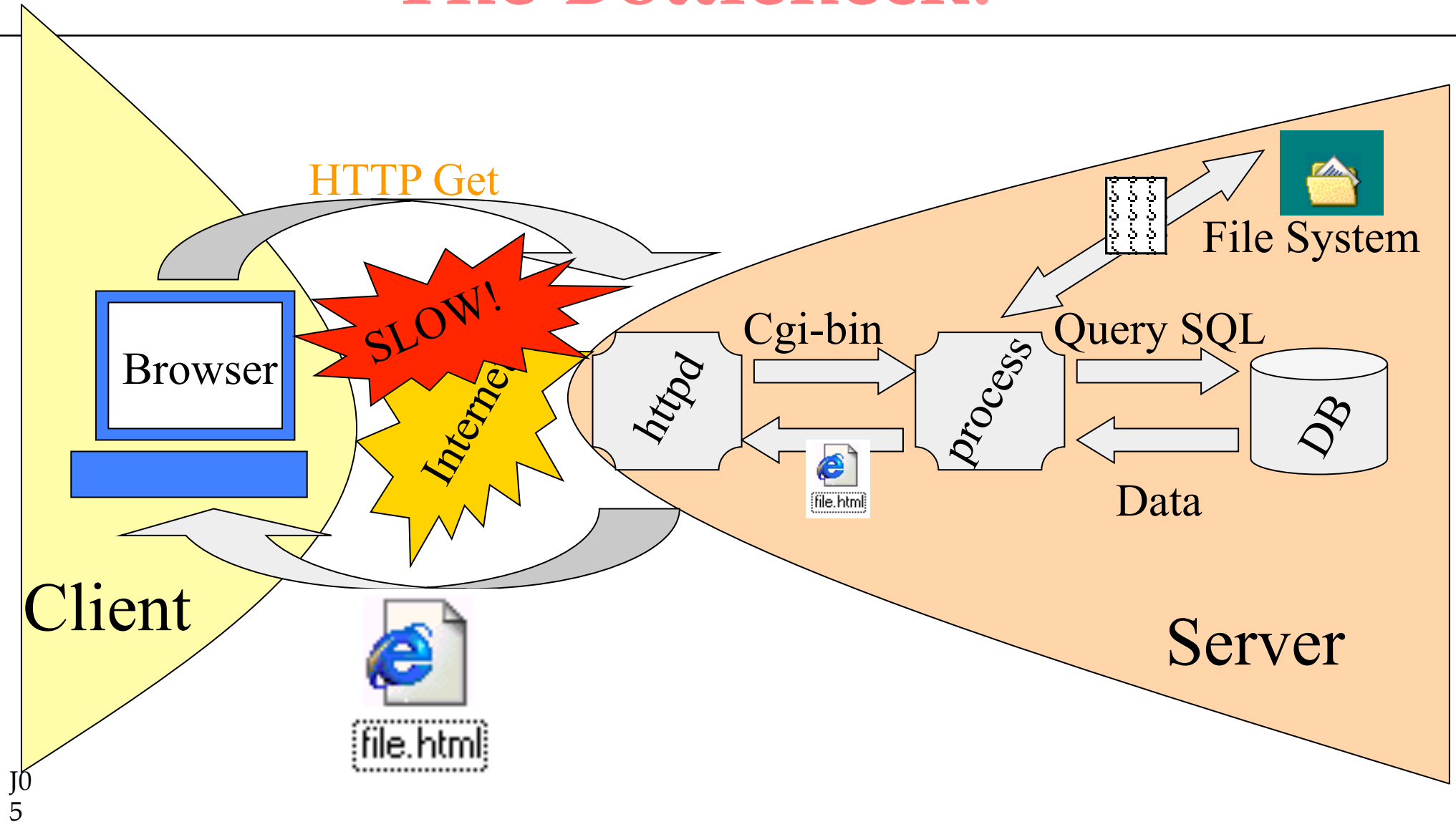
Better dynamic engines

Servlets, ASP, JSP (+ Php, Perl, Python...)

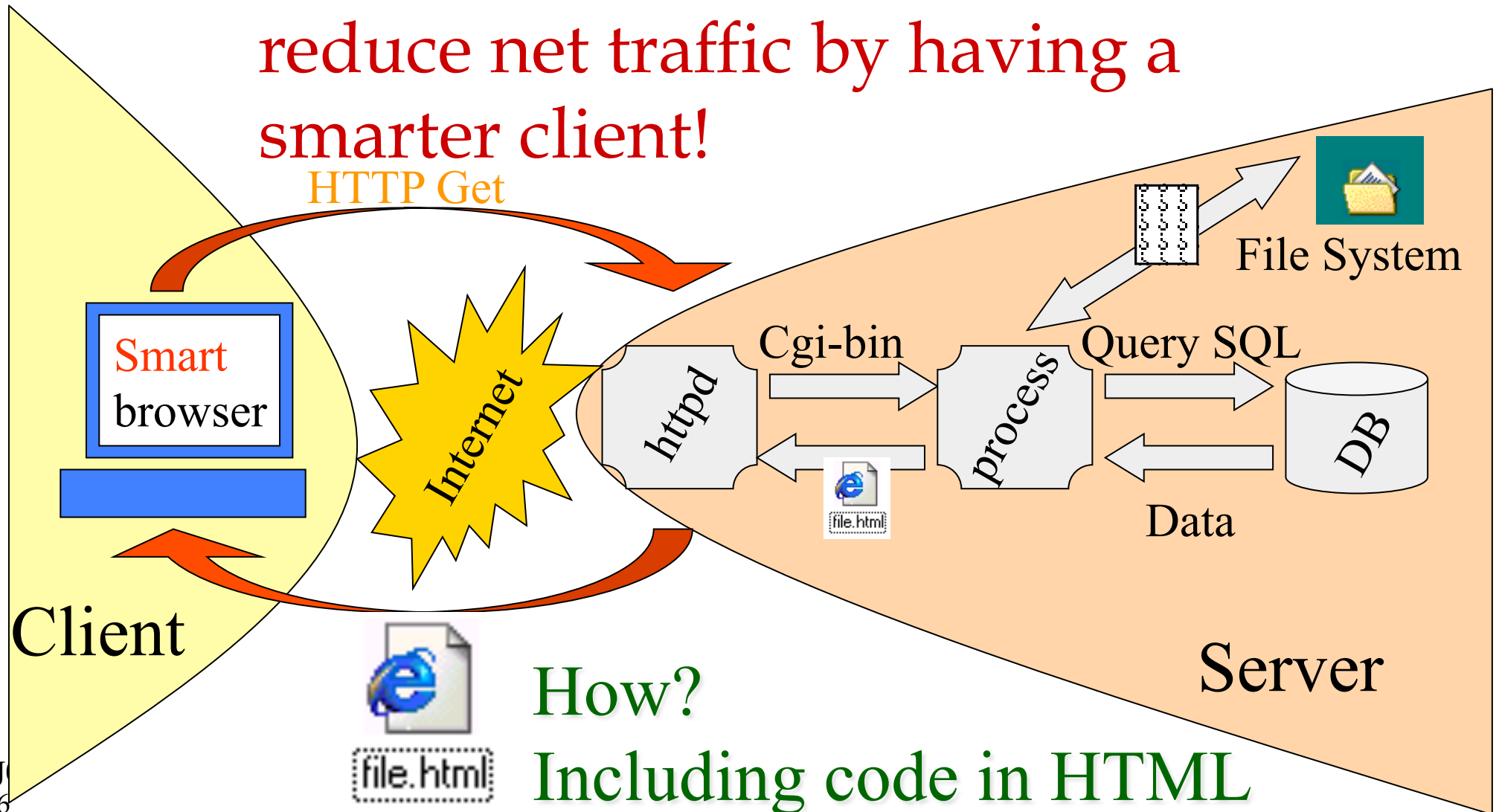
Better Server-side organization

EJB, frameworks (Struts, Hybernate, Spring)

The Bottleneck!



The solution:



Evolution of the web: 2

Better control of the browser

Javascript + DOM

Applet-Javascript interaction

Better separation of content and presentation

CSS (DHTML=HTML4+Javascript+DOM
+CSS)

XML+XSLT, Cocoon (XHTML)

Evolution of the web: 3

Better construction of interfaces (widgets)

.Net

Java Server Faces

Scripting the Web

Client Side ECMAScript & Document Object Model

JavaScript History

- JavaScript 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)

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

JavaScript is...

Scripted (not compiled)

Powerful

Object-based

Cross-Platform

Client and Server

JavaScript allows...

Dynamic Web Sites

Dynamic HTML (DHTML)

Interactive Pages/Forms

Server-Side CGI Functionality

Application Development

JavaScript can...

Build Objects

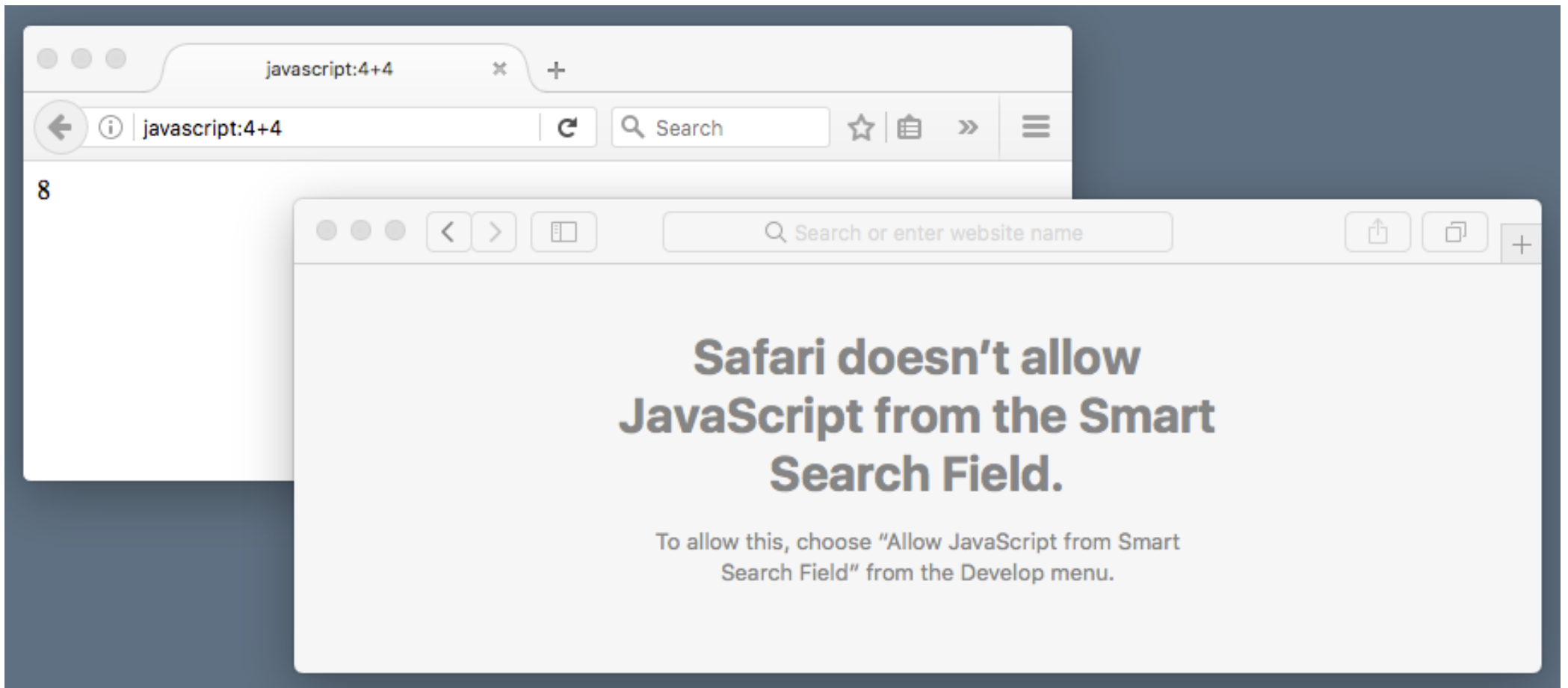
Use Events

Enforce Security

Embed or Componentize

Javascript in the url field

Firefox



10
15

Not working any more in Chrome

- ```
/*comment*/
```

- `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}`

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
```

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)

a="foo";

TAG-related methods:

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

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

strike(), sup()

anchor(), link()



```
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
```

**<HTML>**

**<HEAD>**

**<SCRIPT>**

```
function fact(n) {
 if (n==1) return n;
 return n*fact(n-1);
}
```

**</SCRIPT>**

**</HEAD>**

**...**

**<BODY>**

**<H2>Table of Factorial Numbers </H2>**

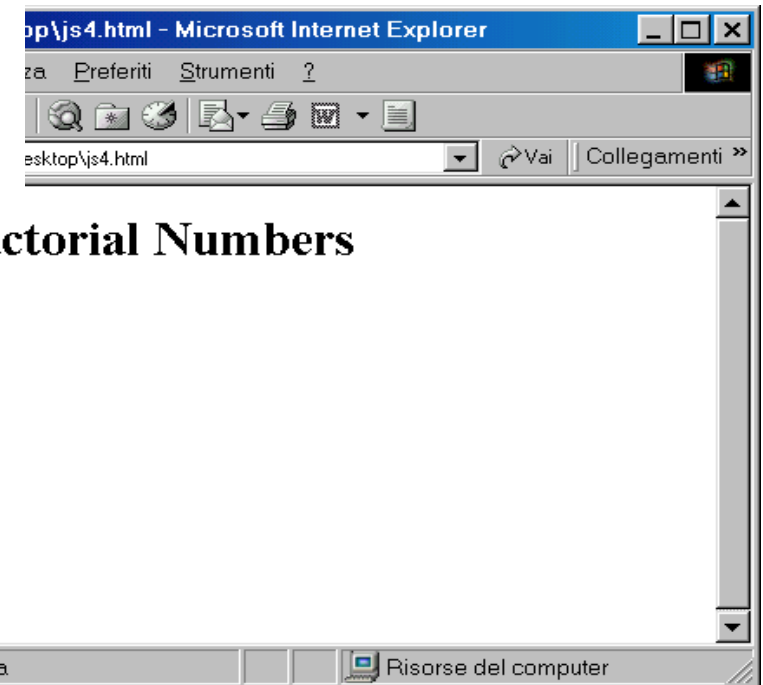
**<SCRIPT>**

```
for (i=1; i<10; i++) {
 document.write(i+"!="+fact(i));
 document.write("
");
}
```

**</SCRIPT>**

**</BODY>**

**</HTML>**



**<BODY>**

**<SCRIPT>**

**n=window.prompt("Give me the value of n",3)**

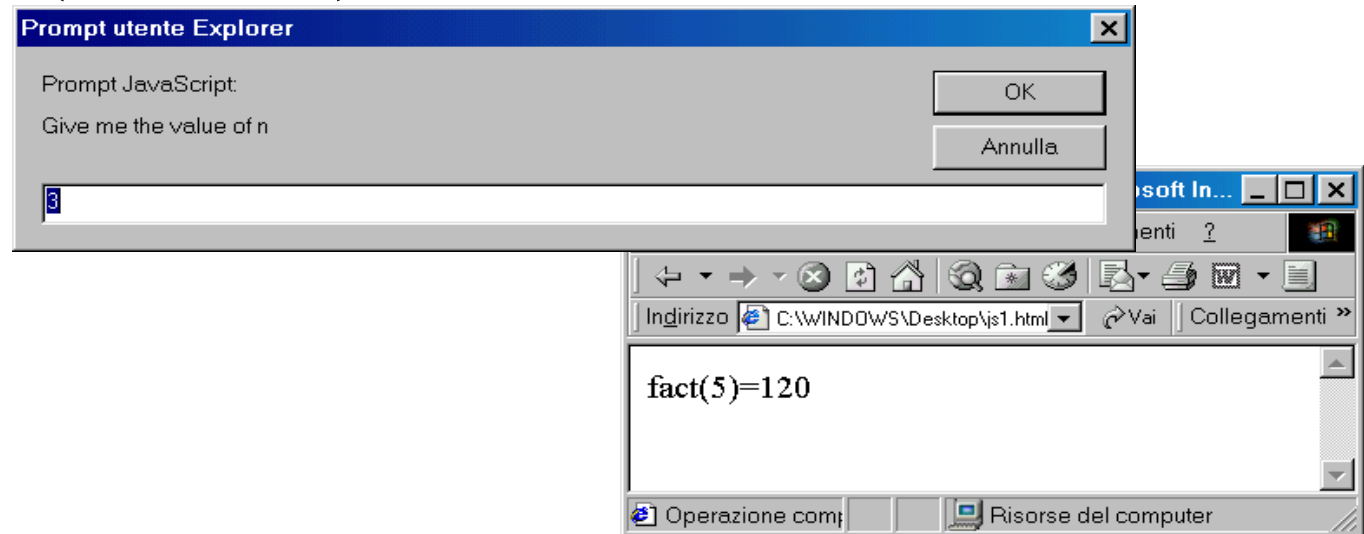
**document.write("fact("+n+")="+fact(n));**

**document.write("<BR>");**

**</SCRIPT>**

**</BODY>**

**</HTML>**



**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}
}
```

**Instance variables**

**method**

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

Actually, JavaScript does NOT have traditional 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}
```

```
a = new Array()
```

```
a[0]=3; a[1]="hello"; a[10]=new Rectangle(2,2);
```

```
a.length() => 11
```

Arrays can be

**SPARSE, INHOMOGENEOUS, ASSOCIATIVE**

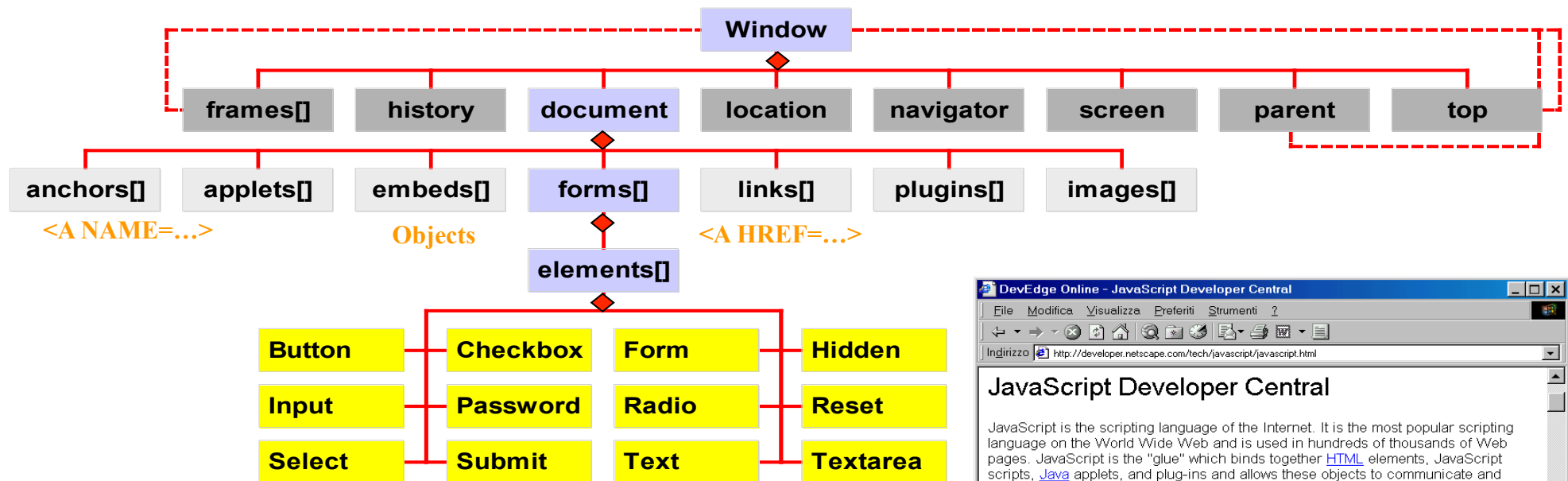
```
a["name"]="Jaric"
```

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

# DOM Object hierarchy

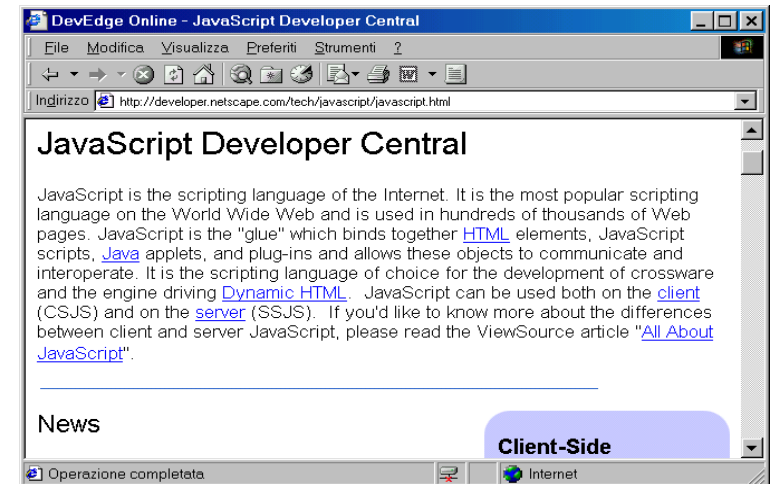


**The Most Important Slide**



Symbol  $\blacklozenge$  means containment (has-a)

Dashed line means “is an instance of”



*“A web browser window or frame”*

## Main properties

### Objects

history

frames[]

document

location

navigator

screen

parent – top

### Other properties

status – defaultStatus

name

## Main methods

alert(), prompt(), confirm()

focus(), blur()

moveBy(), moveTo()

resizeBy(), resizeTo()

scroll(), scrollBy(), scrollTo()

setInterval(), clearInterval()

setTimeout(), clearTimeout()

*“Information about the display”*

### Main properties

availHeight, availWidth

height, width

colorDepth, pixelDepth

hash

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!

*“The URL history of the browser”*

## Main properties

length

## Main methods

back()

forward()

go(+/-n)

go(target\_substring)

*“The specification of the current URL”*

## Main properties

href

protocol, hostname, port

search

hash

## Main methods

reload()

replace()

*“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()

*“An image embedded in an HTML document”*

## Main properties

border *[width in pixels]*

height

width

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

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

(\*) Return false to cancel default action

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

(\*) Return false to cancel default action

DOM

Properties

Input

Methods

Objects

|            |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 |   |   |   |   |   |   |   |   |

Event  
Handlers

Properties

checked  
defaultChecked

name

options[]

selectedIndex

type

value

onblur

onchange

onclick

onfocus

*“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]*

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>
```



**<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="green">Green**

**<OPTION VALUE="blue">Blue**

**<OPTION VALUE="white">White**

**<OPTION VALUE="yellow">Yellow**

**<OPTION**

**VALUE="purple">Purple**

**</SELECT>**

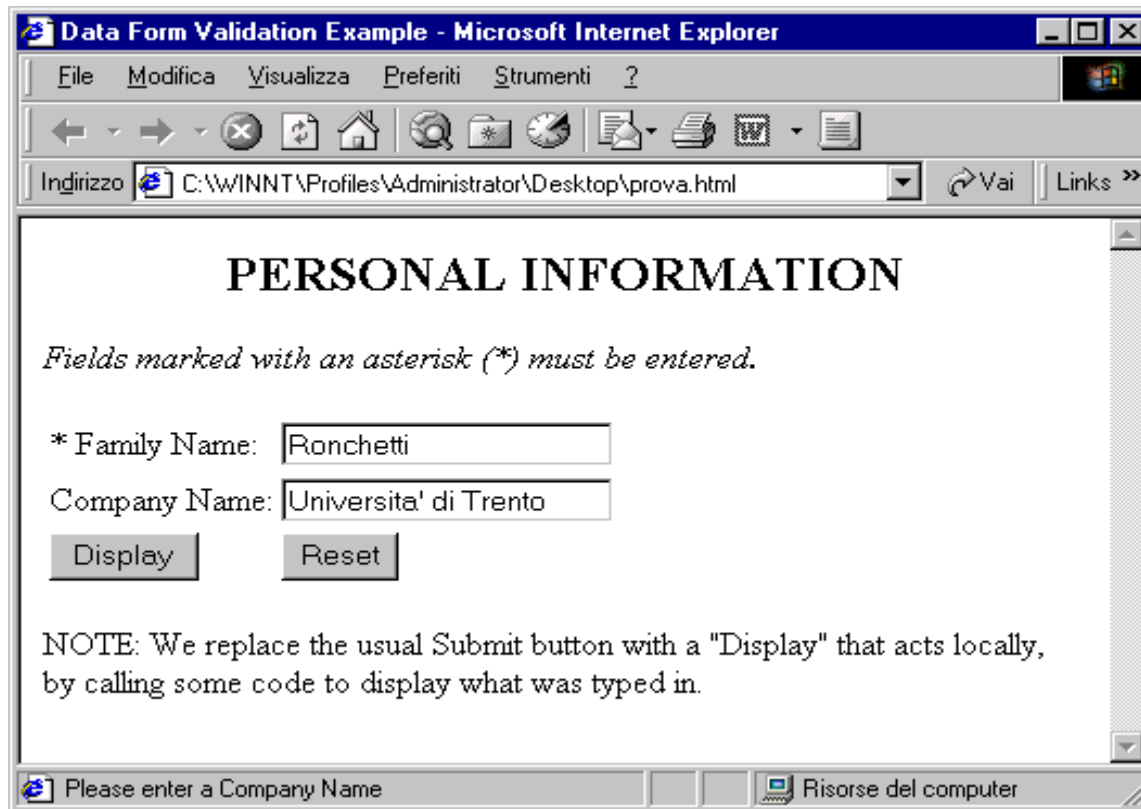
**</FORM>**

**</BODY>**

**</HTML>**

# A more complex example -1

A simple data entry validation page



**Data Form Validation Example - Microsoft Internet Explorer**

File Modifica Visualizza Preferiti Strumenti ?

Indirizzo C:\WINNT\Profiles\Administrator\Desktop\prova.html Vai Links »

## PERSONAL INFORMATION

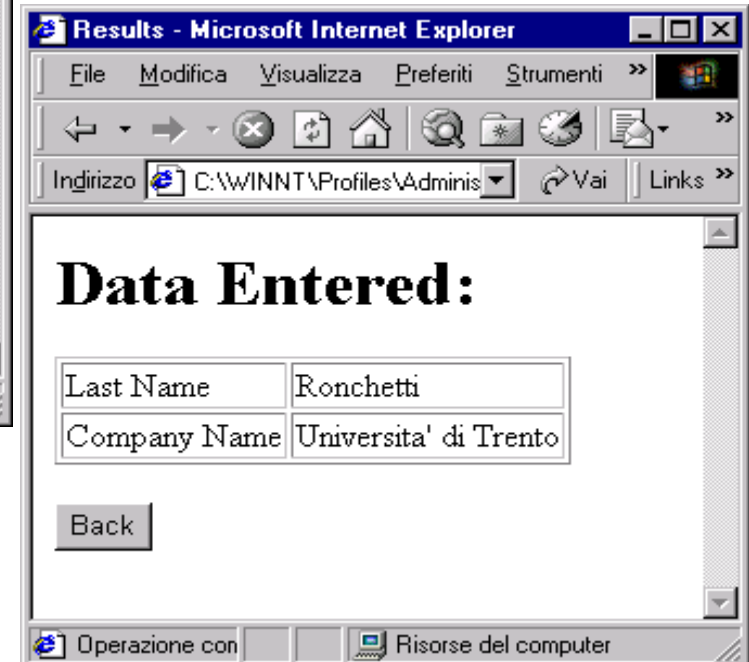
*Fields marked with an asterisk (\*) must be entered.*

\* 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



**Results - Microsoft Internet Explorer**

File Modifica Visualizza Preferiti Strumenti »

Indirizzo C:\WINNT\Profiles\Adminis... Vai Links »

## Data Entered:

|              |                       |
|--------------|-----------------------|
| Last Name    | Ronchetti             |
| Company Name | Universita' di Trento |

Operazione con Risorse del computer

# 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

# 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”

# A more complex example -4

---

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

Type a message in the status bar

```
function validatePersonalInfo(form)
```

Validate the form

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

(should run over all fields  
And perform suitable checks)

```
function checkString (theField, s)
```

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

Check that “theField”  
is not empty

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

# A more complex example -5

---

**<SCRIPT>**

Global variables

```
var sCompany="Company Name"; var sLastName="Last Name"; var
 form="PersonallInfo";
```

Value-printing  
function

**function displayPersonallInfo(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>
```

Add a Button to go  
back in history

**</HEAD>**

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

# 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="PersonallInfo">
<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”

# 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,

by calling some code to display what was typed in.
</FORM>
</BODY>
</HTML>
```

First Button

Second Button

End of file “FormValidation.html”

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

# Layout engines

---

A **web browser engine** (sometimes called **layout engine** or **rendering engine**) is a software component that takes

- marked up content (such as HTML, XML, image files, etc.) and
- formatting information (such as CSS, XSL, etc.)

and displays the formatted content on the screen.

It draws onto the content area of a window, which is displayed on a monitor or a printer.

- Blink (Chromium)
- Gecko (Mozilla)
- Trident (IE)
- Webkit (Safari, Android)

See:

[https://en.wikipedia.org/wiki/Comparison\\_of\\_layout\\_engines\\_\(ECMAScript\)](https://en.wikipedia.org/wiki/Comparison_of_layout_engines_(ECMAScript))

# ECMAScript Engine

---

An ECMAScript engine is a program that  
**executes source code written in a version  
of the ECMAScript** language standard

See [https://en.wikipedia.org/wiki/List\\_of\\_ECMAScript\\_engines](https://en.wikipedia.org/wiki/List_of_ECMAScript_engines)

## Examples:

- V8 (chrome)
- SpiderMonkey (Mozilla)
- Chakra (I.E.)
- Squirrelfish/Nitro (Apple)
- Nashorn (Oracle – JDK)

# References

---

## **Standard ECMA-262 ECMAScript Language Specification:**

[http://www.ecma-international.org/publications/standards/  
Ecma-262.htm](http://www.ecma-international.org/publications/standards/Ecma-262.htm)

## **Books:**

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

# Server-Side JavaScript

---

A substitute for CGI.

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

(An approach which started long ago (Netscape SSJS))

Then mostly forgotten, later revived by Rhino (a bridge between JS and Java) and recently by Node.js

# Node.js

---

Node.js is an open-source, cross-platform JavaScript run-time environment for executing JavaScript code server-side. Node.js has an event-driven architecture capable of asynchronous I/O.

Optimize throughput and scalability

- in Web applications with many input/output operations,
- for real-time Web applications

<https://www.w3schools.com/nodejs/default.asp>