



Departamento de Lenguajes y
Sistemas Informáticos



Universitat d'Alacant
Universidad de Alicante

Modelo de objetos de documento

Programación en Internet
Curso 2009-2010

Programación en Internet – Curso 2009-2010

Índice

- Introducción
- DOM
- Cómo acceder a un formulario
- BOM

Introducción

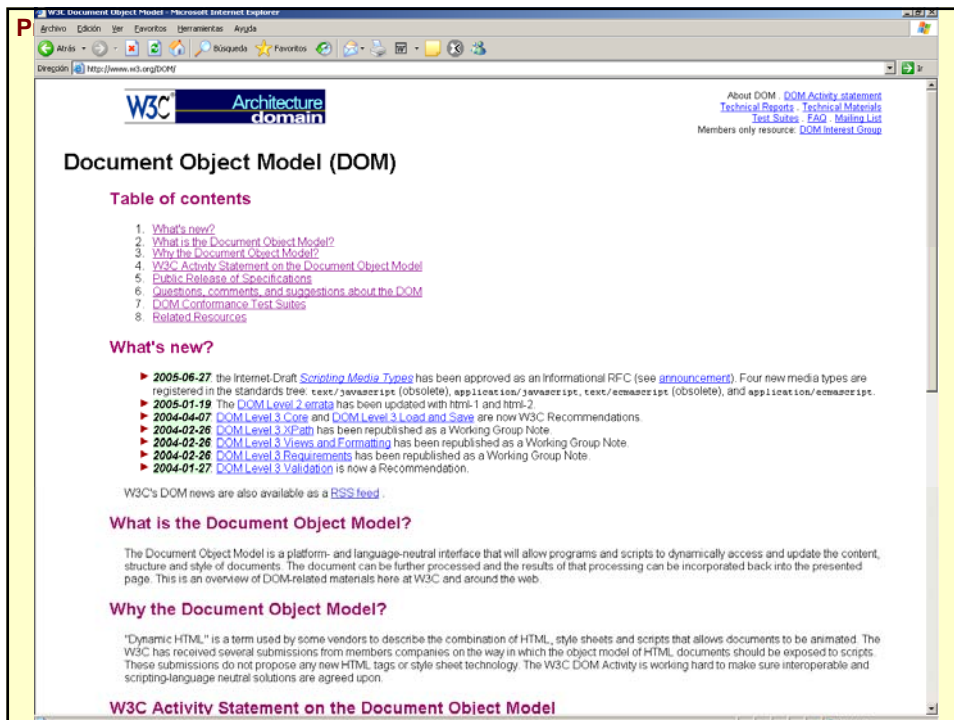
- *Document Object Model (DOM)*
 - Permite acceder a los elementos HTML de un documento → No confundir con los lenguajes de *script*
- *Browser Object Model (BOM)*
 - *Navigator objects*
 - Permite acceder a elementos del navegador
- Incompatibilidades entre distintos navegadores
- Jerarquía objetos ⇔ Estructura documento

Introducción

- Ventajas:
 - Permite manipular las páginas web en el cliente sin necesidad de procesamiento en el lado del servidor

DOM

- *Document Object Model*
- Especificación de W3C
- Mecanismo para que desde cualquier lenguaje de programación se pueda manipular un documento XML o HTML
- Básicamente:
 - Crea una estructura lógica para el documento (llamada “árbol nodal”)
 - Proporciona un método para leer y manipular el documento



Programación en Internet – Curso 2009-2010

DOM

- Versiones de DOM:
 - Nivel 0
 - Nivel 1
 - *Document Object Model Level 1* (1/10/1998)
 - *Document Object Model Level 1 (Second Edition)* (29/9/2000)
 - Nivel 2
 - *Document Object Model Level 2 Core* (13/11/2000)
 - *Document Object Model Level 2 Views*
 - *Document Object Model Level 2 Events*
 - *Document Object Model Level 2 Style*
 - *Document Object Model Level 2 Traversal and Range*
 - *Document Object Model Level 2 HTML*
 - Nivel 3
 - *Document Object Model Level 3 Core* (7/4/2004)
 - *Document Object Model Level 3 Load and Save*
 - *Document Object Model Level 3 Validation*

Programación en Internet – Curso 2009-2010

DOM

- What does your user agent claim to support?
 - <http://www.w3.org/2003/02/06-dom-support.html>

W3C Document Object Model Recommendations: support claims - Windows Internet Explorer

http://www.w3.org/2003/02/06-dom-support.html

DOM Module	DOM Level 1	DOM Level 2	DOM Level 3
Core: basic methods (Level 1 and 2) and extensions for XML Namespaces (Level 2 only)	-	2000	2009
XML: extensions for XML 1.0	1999	2000	2009
HTML: extensions for HTML 4.0x (Level 1 and 2) and support of XHTML 1.0 (Level 2 only)	supported	2003	N/A
Views: used with the Level 2 CSS and UIEvents DOM modules	N/A	2000	N/A
StyleSheets: association between a style sheet and a document	N/A	2000	N/A
CSS: extensions for cascading style sheets	N/A	2000	N/A
CSS2: extensions for Cascading Style Sheets Level 2	N/A	2000	N/A
Events: generic events system	N/A	2000	N/A
UIEvents: basic user interface events	N/A	2000	N/A
MouseEvents: mouse device events	N/A	2000	N/A
MutationEvents: events for mutations in a DOM tree	N/A	2000	N/A
HTMLEvents: HTML 4.01 events	N/A	2000	N/A
Range: extensions to manipulate a range in a DOM tree	N/A	2000	N/A
Traversal: Alternative traversal methods of a DOM tree	N/A	2000	N/A
LS: Loading a document into a DOM tree	N/A	N/A	2009
LS-Async: Asynchronous loading of a document into a DOM tree	N/A	N/A	2009
Validation: Schema-oriented modification of a DOM tree	N/A	N/A	2009

This page *does not*

W3C Document Object Model Recommendations: support claims - Mozilla Firefox

http://www.w3.org/2003/02/06-dom-support.html

DOM Module	DOM Level 1	DOM Level 2	DOM Level 3
Core: basic methods (Level 1 and 2) and extensions for XML Namespaces (Level 2 only)	-	supported	2009
XML: extensions for XML 1.0	supported	supported	2009
HTML: extensions for HTML 4.0x (Level 1 and 2) and support of XHTML 1.0 (Level 2 only)	supported	supported	N/A
Views: used with the Level 2 CSS and UIEvents DOM modules	N/A	supported	N/A
StyleSheets: association between a style sheet and a document	N/A	supported	N/A
CSS: extensions for cascading style sheets	N/A	supported	N/A
CSS2: extensions for Cascading Style Sheets Level 2	N/A	supported	N/A
Events: generic events system	N/A	supported	N/A
UIEvents: basic user interface events	N/A	supported	N/A
MouseEvents: mouse device events	N/A	supported	N/A
MutationEvents: events for mutations in a DOM tree	N/A	2000	N/A
HTMLEvents: HTML 4.01 events	N/A	supported	N/A
Range: extensions to manipulate a range in a DOM tree	N/A	supported	N/A
Traversal: Alternative traversal methods of a DOM tree	N/A	2000	N/A
LS: Loading a document into a DOM tree	N/A	N/A	2009
LS-Async: Asynchronous loading of a document into a DOM tree	N/A	N/A	2009
Validation: Schema-oriented modification of a DOM tree	N/A	N/A	2009

Terminado

DOM

- DOM trata un documento HTML (XML) como una jerarquía de nodos
 - Distintos nodos y cada uno tiene permitido solo ciertos hijos de ciertos tipos de nodo
 - Existen algunos tipos de nodo que son hoja → No pueden tener hijos

DOM

- Document:
 - Element (máximo uno)
 - ProcessingInstruction
 - Comment
 - DocumentType (máximo uno)
- DocumentFragment:
 - Element
 - ProcessingInstruction
 - Comment
 - Text
 - CDATASection
 - EntityReference

Programación en Internet – Curso 2009-2010

DOM

- DocumentType: sin hijos
- EntityReference:
 - Element
 - ProcessingInstruction
 - Comment
 - Text
 - CDATASection
 - EntityReference

Programación en Internet – Curso 2009-2010

DOM

- Element:
 - Element
 - Text
 - Comment
 - ProcessingInstruction
 - CDATASection
 - EntityReference
- Attr:
 - Text
 - EntityReference

Programación en Internet – Curso 2009-2010

DOM

- ProcessingInstruction: sin hijos
- Comment: sin hijos
- Text: sin hijos
- CDATASection: sin hijos
- Entity:
 - Element
 - ProcessingInstruction
 - Comment
 - Text
 - CDATASection
 - EntityReference
- Notation: sin hijos

Programación en Internet – Curso 2009-2010

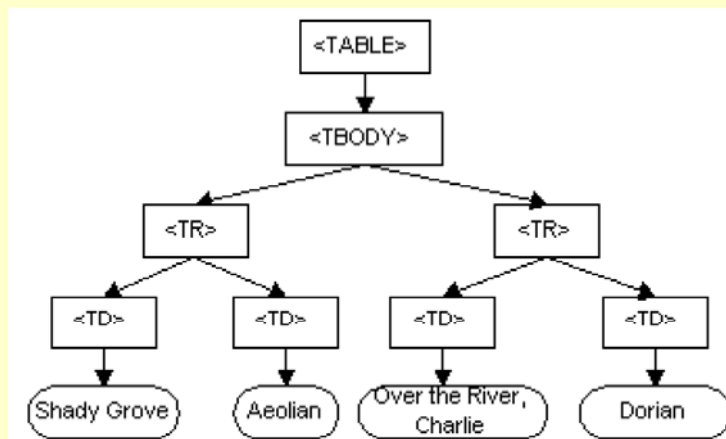
DOM

- Ejemplo:

```
<TABLE>
<TBODY>
<TR>
<TD>Shady Grove</TD>
<TD>Aeolian</TD>
</TR>
<TR>
<TD>Over the River, Charlie</TD>
<TD>Dorian</TD>
</TR>
</TBODY>
</TABLE>
```

Fuente: Document Object Model (DOM) Level 1 Specification, W3C Recommendation 1 October, 1998

DOM



Fuente: Document Object Model (DOM) Level 1 Specification, W3C Recommendation 1 October, 1998

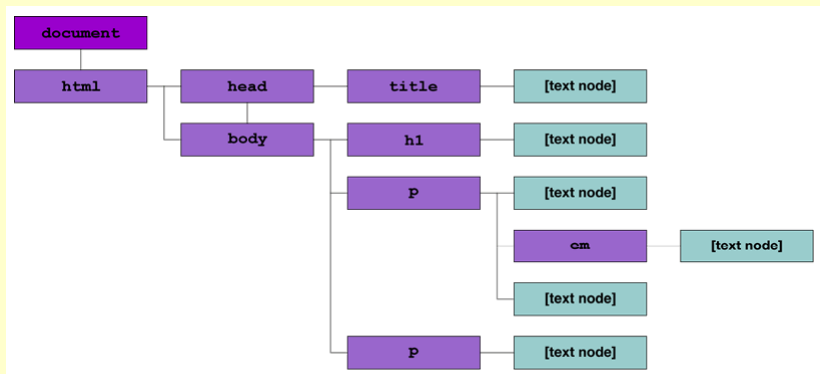
DOM

- Ejemplo:

```
<html>
<head>
  <title>This is a Document!</title>
</head>
<body>
  <h1>This is a header!</h1>
  <p id="excitingText">This is a paragraph!
  <em>Excitement</em>!</p>
  <p>This is also a paragraph, but it's not nearly as
  exciting as the last one.</p>
</body>
</html>
```

Fuente: DevOpera, 46: Traversing the DOM, 3 February 2009

DOM



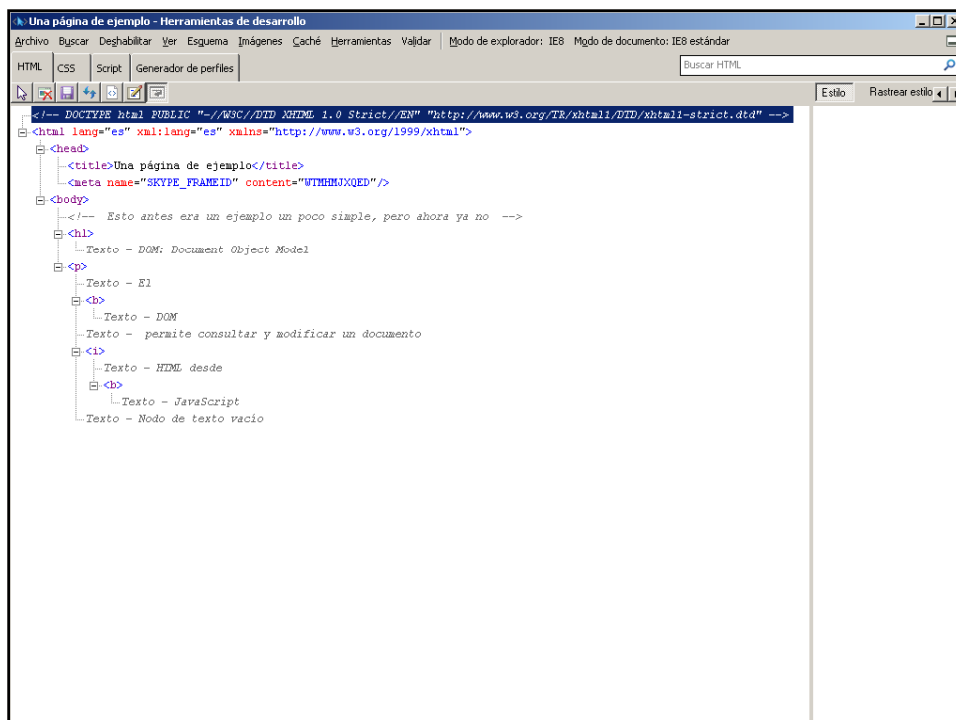
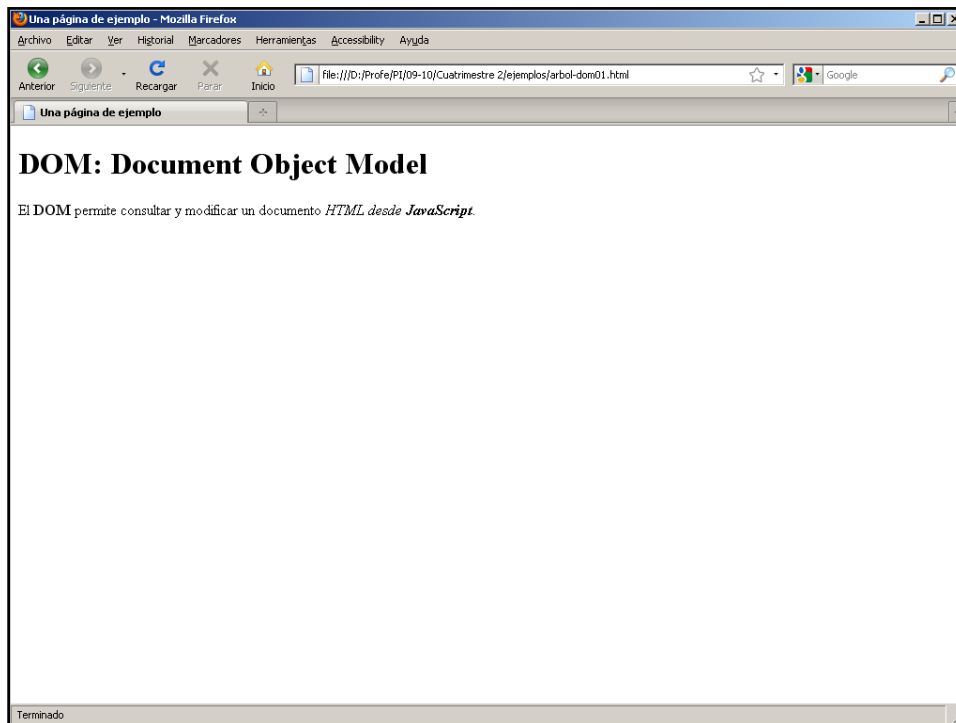
Fuente: DevOpera, 46: Traversing the DOM, 3 February 2009

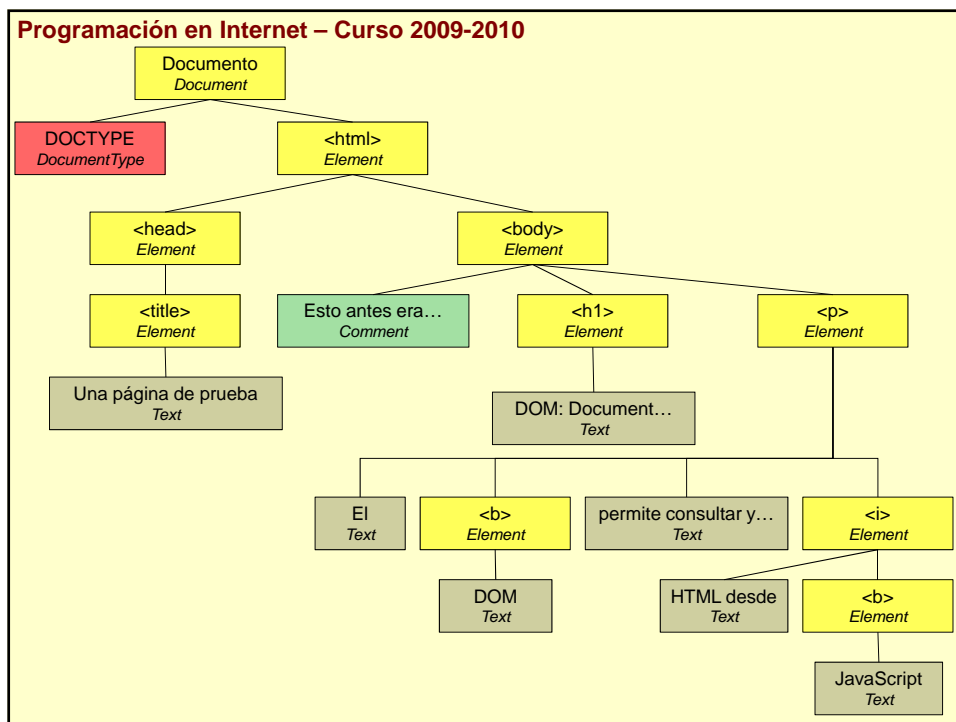
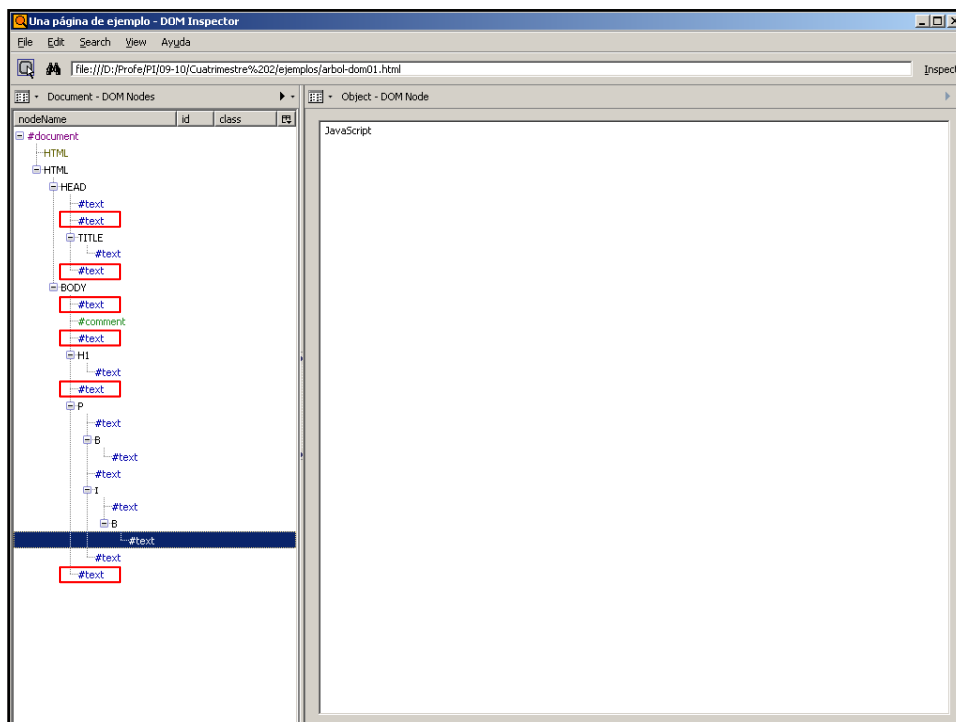
DOM

• Ejemplo:

```

<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="es"
  lang="es">
<head>
<title>Una página de ejemplo</title>
</head>
<body>
<!-- Esto antes era un ejemplo un poco simple, pero ahora ya no
-->
<h1>DOM: Document Object Model</h1>
<p>
El <b>DOM</b> permite consultar y modificar un documento
<i>HTML desde <b>JavaScript</b></i>
</p>
</body>
</html>
    
```





DOM Level 0

- Nivel 0:
 - No es una recomendación (no existe como tal)
 - "DOM Level 0" refiere una mezcla de funcionalidades para trabajar con documentos HTML
 - Ofrecido desde Netscape Navigator version 3.0 y Microsoft Internet Explorer version 3.0

DOM Level 0

- `innerHTML`:
 - Asigna o devuelve el código HTML entre las etiquetas de inicio y final del objeto
 - Como no existe una especificación formal de esta propiedad, su implementación difiere
 - Nunca se debería de utilizar para escribir partes de una tabla → Usar los métodos de W3C DOM

DOM Level 0

- Ejemplo: eliminar el contenido entero de una página

```
// Sustituye el contenido de la etiqueta body
// por una cadena vacía
document.body.innerHTML = "";
```

DOM Level 0

- Ejemplo: texto que cambia al desplazar el puntero del ratón

```
<p onmouseover="this.innerHTML='<b>Mouse
out to change back.</b>'"
  onmouseout="this.innerHTML='<i>Mouse
over again to change.</i>'">
<i>Mouse over this text to change it.</i>
</p>
```

DOM Level 1

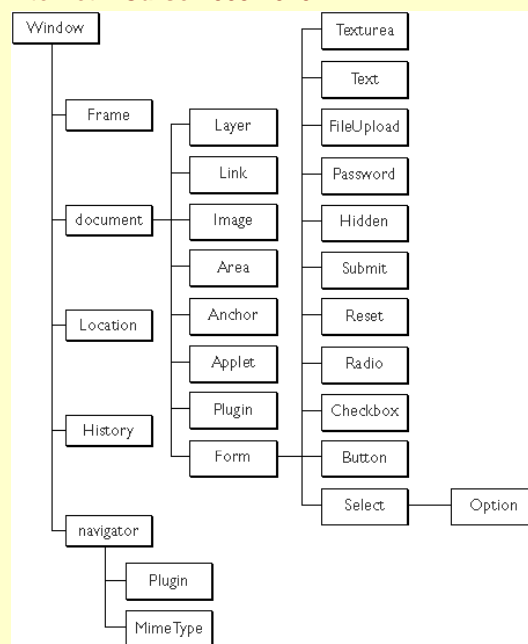
- Nivel 1
 - *Document Object Model Level 1* (1/10/1998): definición de las estructuras de datos e interfaces básicos del modelo DOM
 - *Document Object Model Level 1 (Second Edition)* (29/9/2000)
 - Dos partes:
 - *Core*: conjunto fundamental de interfaces de bajo nivel que representan cualquier documento estructurado e interfaces extendidos para representar documentos XML
 - *HTML*: proporciona interfaces adicionales de alto nivel para un acceso más apropiado a documentos HTML

DOM Level 1

- Cada nodo es de tipo `Node`, pero existen subtipos: `Document`, `DocumentType`, `Element`, `Text`, `Comment`, ...
- Atributos son nodos del tipo `Attr`, pero no aparecen en el árbol que representa el documento (hay que acceder a través de métodos del nodo)

DOM Level 1

- Objetos=propiedades+métodos+eventos
- Descendientes son propiedades y objetos a la vez



DOM Level 1

- `document` \Leftrightarrow `<body>...</body>`
- **Propiedades:**
 - `alinkColor`, `bgColor`, `fgColor`,
`linkColor`, `vlinkColor`
 - `forms`, `images`, `title`
 - `lastModified`, `referrer`
- **Métodos:**
 - `write`, `writeln`

DOM Level 1

- **Eventos:**
`onAbort`, `onBlur`, `onChange`, `onClick`,
`ondblclick`, `ondragdrop`, `onerror`,
`onfocus`, `onkeydown`, `onkeypress`,
`onkeyup`, `onload`, `onmousedown`,
`onmousemove`, `onmouseout`,
`onmouseover`, `onmouseup`, `onmove`,
`onreset`, `onresize`, `onselect`,
`onsubmit`, `onunload`

Programación en Internet – Curso 2009-2010

DOM Level 1

- **onClick**
 - Se invoca al recibir un click el objeto
- **onChange**
 - Se invoca al cambiar el valor o estado del control
- **onBlur y onFocus**
 - ‘Salta’ al obtener (Focus) o perder (Blur) el enfoque el control correspondiente.

Programación en Internet – Curso 2009-2010

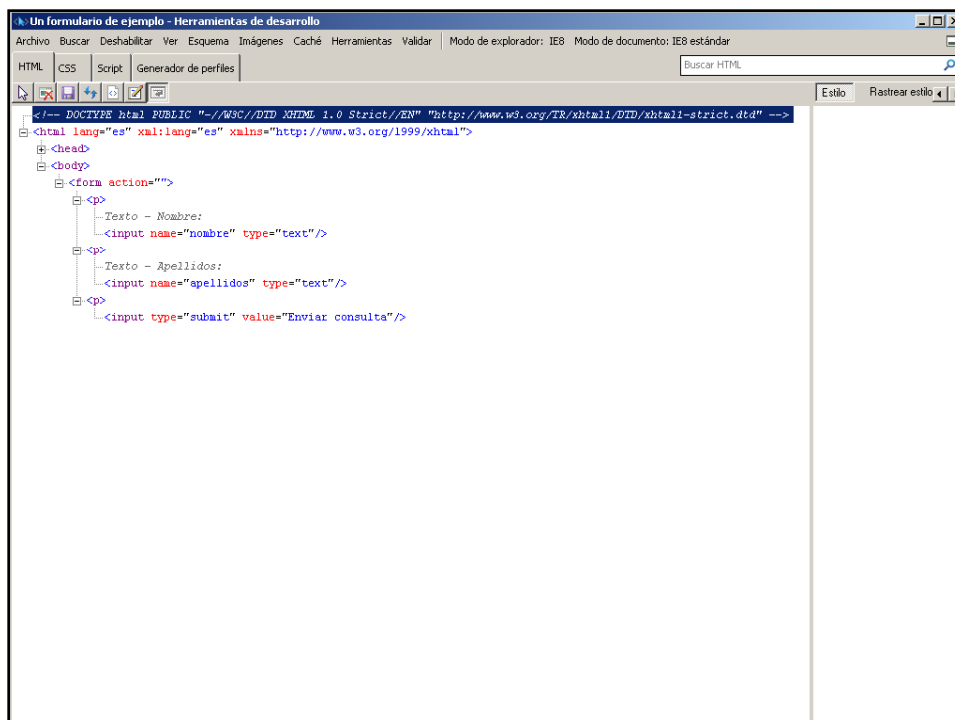
DOM Level 1

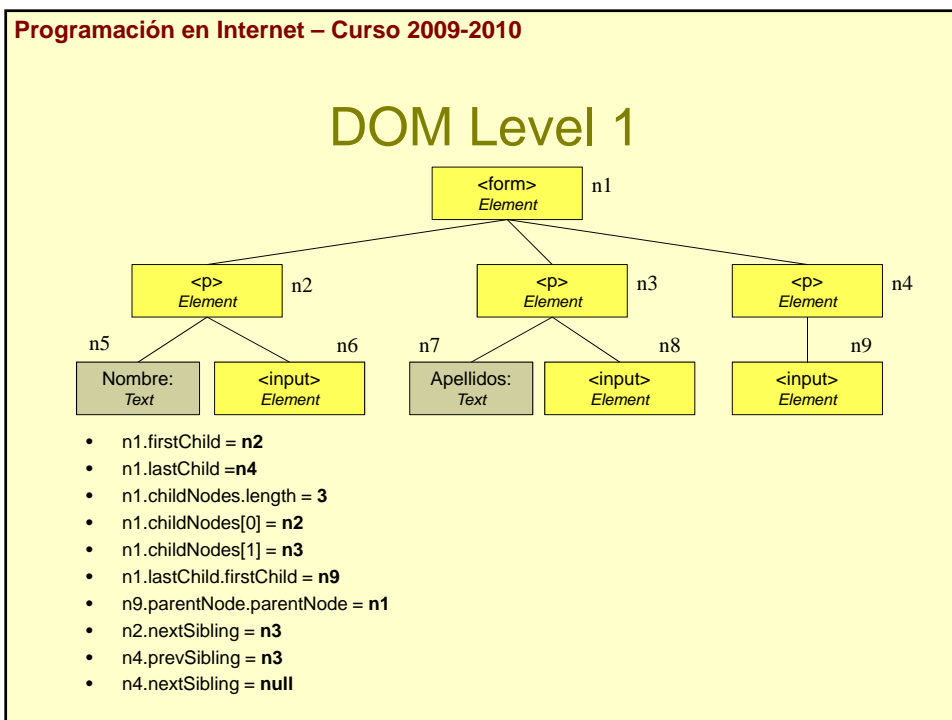
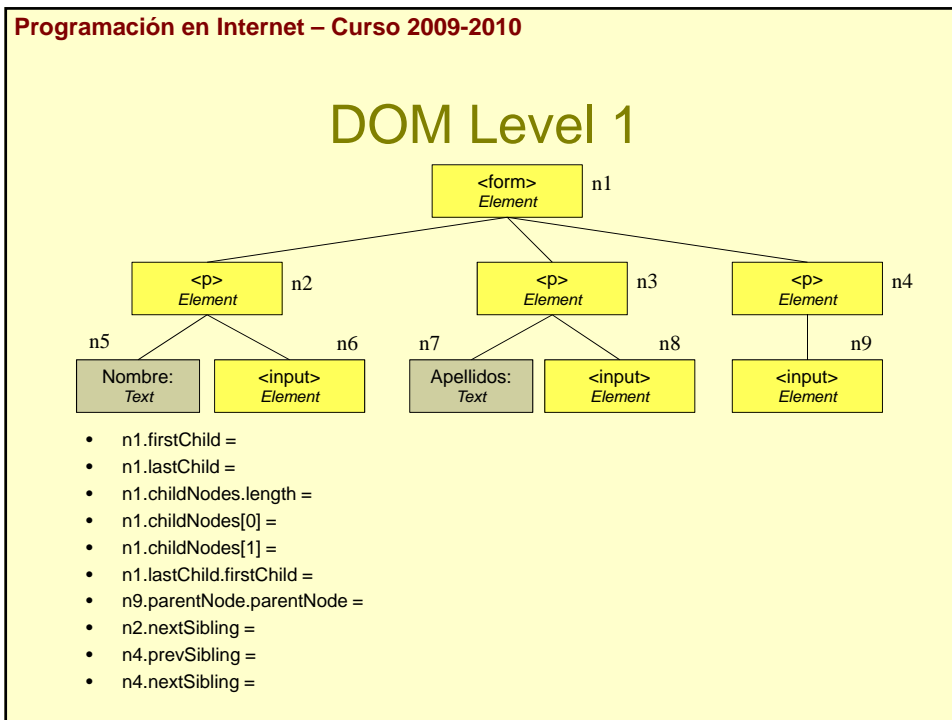
- Cada nodo posee un conjunto de propiedades que lo relacionan con sus “familiares”:
 - `childNodes`
 - `firstChild`
 - `lastChild`
 - `parentNode`
 - `nextSibling`
 - `prevSibling`

DOM Level 1

- Ejemplo:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="es" lang="es">
<head>
<title>Un formulario de ejemplo</title>
</head>
<body>
<form action="">
<p>
Nombre: <input type="text" name="nombre" /></p>
<p>
Apellidos: <input type="text" name="apellidos" /></p>
<p>
<input type="submit" /></p>
</form>
</body>
</html>
```





Programación en Internet – Curso 2009-2010

DOM Level 1

- `getElementById("elementID")`
 - Útil para recuperar un elemento concreto
`var element = document.getElementById("myTable")`
- `getElementsByName("name")`:
 - Útil para recuperar los controles de un formulario con un mismo nombre
- `getElementsByTagName("tagName")`:
 - Útil para recuperar un conjunto de elementos de un mismo tipo
`var images = document.getElementsByTagName("img")`
 - El valor especial "*" representa todas las etiquetas
- `getAttribute("attrName")`:
 - `var element = document.getElementById("myTable").getAttribute("width");`

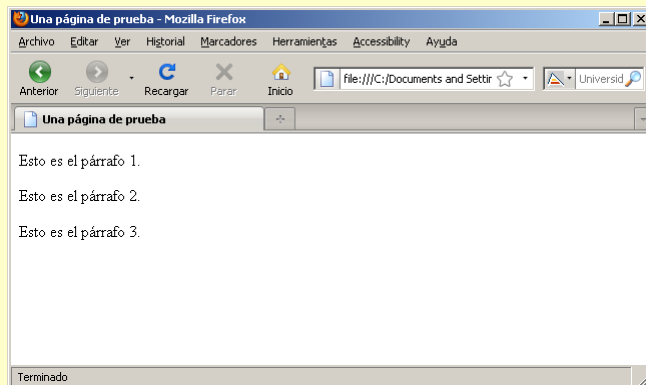
Programación en Internet – Curso 2009-2010

DOM Level 1

```
// HTML:  
// <div id="d"><p>Esto es un párrafo.</p>  
// <p>Esto es otro párrafo</p>  
// </div>  
  
d = document.getElementById("d");  
alert(d.innerHTML);
```

DOM Level 1

- Cómo acceder a un elemento:
 - Queremos mostrar el texto del segundo párrafo



DOM Level 1

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="es" lang="es">
<head>
<title>Una página de prueba</title>
<script type="text/javascript">
function mostrar() {
  // Muestra el contenido del segundo párrafo de tres formas distintas
}
</script>
</head>
<body onload="mostrar()"><p id="p1">Esto es el párrafo 1.</p><p
  id="p2">Esto es el párrafo 2.</p><p id="p3">Esto es el párrafo
  3.</p></body>
</html>
```

DOM Level 1

```
#document
  xml version="1.0" encoding="iso-8859-1"
  html xmlns="http://www.w3.org/1999/xhtml" xml:lang="es" lang="es"
  + head
  - body onload="mostrar()"
    p id="p1"
      #text Esto es el párrafo 1.
    p id="p2"
      #text Esto es el párrafo 2.
    p id="p3"
      #text Esto es el párrafo 3.
```

DOM Level 1

```
<script type="text/javascript">
function mostrar() {
  var b, p;
  // El body de la página
  b = document.getElementsByTagName("body")[0];
  // El segundo párrafo es el primer hermano del primer párrafo
  p = b.firstChild.nextSibling;
  alert(p.firstChild.nodeValue);

  // El segundo párrafo es el segundo hijo
  p = b.childNodes[1];
  alert(p.firstChild.nodeValue);

  // El segundo párrafo tiene el id "p2"
  p = document.getElementById("p2");
  alert(p.firstChild.nodeValue);
}
</script>
```

DOM Level 1

```
<body onload="mostrar()">
<p id="p1">
Esto es el párrafo 1.
</p>
<p id="p2">
Esto es el párrafo 2.
</p>
<p id="p3">
Esto es el párrafo 3.
</p>
</body>
</html>
```

DOM Level 1

```
#document
  xml version="1.0" encoding="iso-8859-1"
  html xmlns="http://www.w3.org/1999/xhtml" xml:lang="es" lang="es"
  head
  body onload="mostrar()"
    #text \r\n
    p id="p1"
      #text Esto es el párrafo 1.
    #text \r\n
    p id="p2"
      #text Esto es el párrafo 2.
    #text \r\n
    p id="p3"
      #text Esto es el párrafo 3.
    #text \r\n
    #text \r\n
```


DOM Level 1

- `createElement("tagName")`
 - `var element = document.createElement("tbody")`
- `appendChild(element)`
 - `element.appendChild(newChild);`
- `replaceChild(element, element)`
 - `element.replaceChild(newElement, oldElement);`
- `setAttribute(name, value)`
 - `document.getElementById("myTable").setAttribute("width", 300);`

DOM Level 1

- **Cómo acceder al texto de una etiqueta:**

```
// Crea un nuevo elemento de tipo párrafo
var p = document.createElement("p");
```

```
// Obtiene un párrafo de un documento
var p = document.getElementById("p1");
```

Programación en Internet – Curso 2009-2010

DOM Level 1

```
// Incorrecto: esto es DOM 0, no es estándar
p.innerHTML = txt;
// Correcto, es del estándar, pero en Internet Explorer
  no funciona
p.textContent = txt;
// Incorrecto, no es del estándar, es sólo de Internet
  Explorer (y algún otro), pero en el resto no funciona
p.text = txt;
// Incorrecto, no es del estándar, es sólo de Internet
  Explorer (y algún otro), pero en el resto no funciona
p.innerText = txt;
// Sólo tiene sentido con los siguientes tipos de nodo:
  CDATASection, Comment, ProcessingInstruction, Text
// Con un párrafo no debe funcionar
p.nodeValue = txt;
```

Programación en Internet – Curso 2009-2010

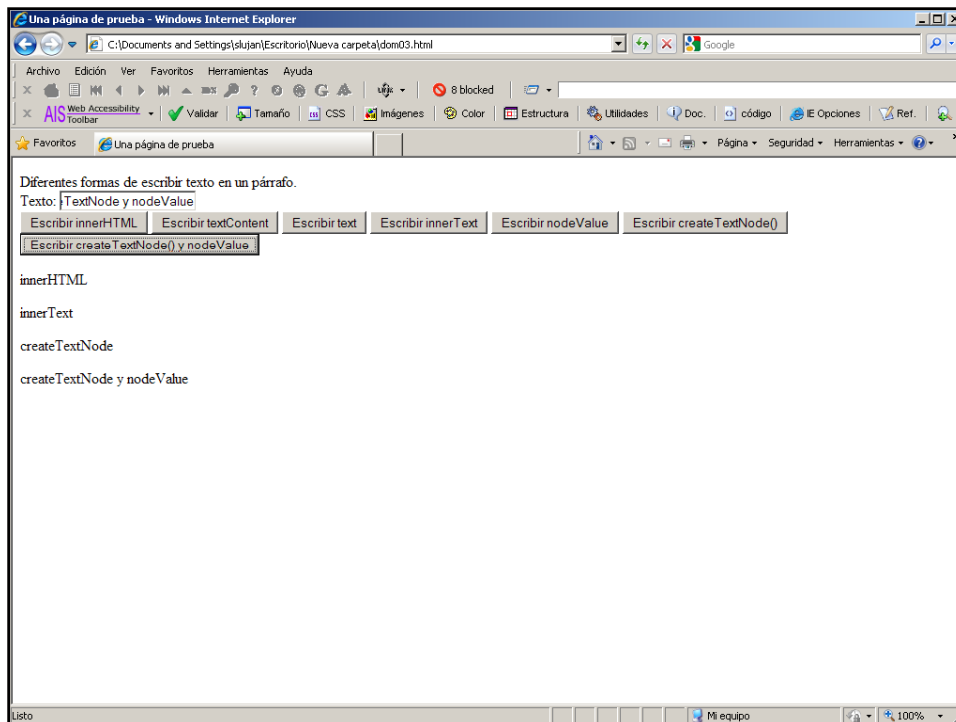
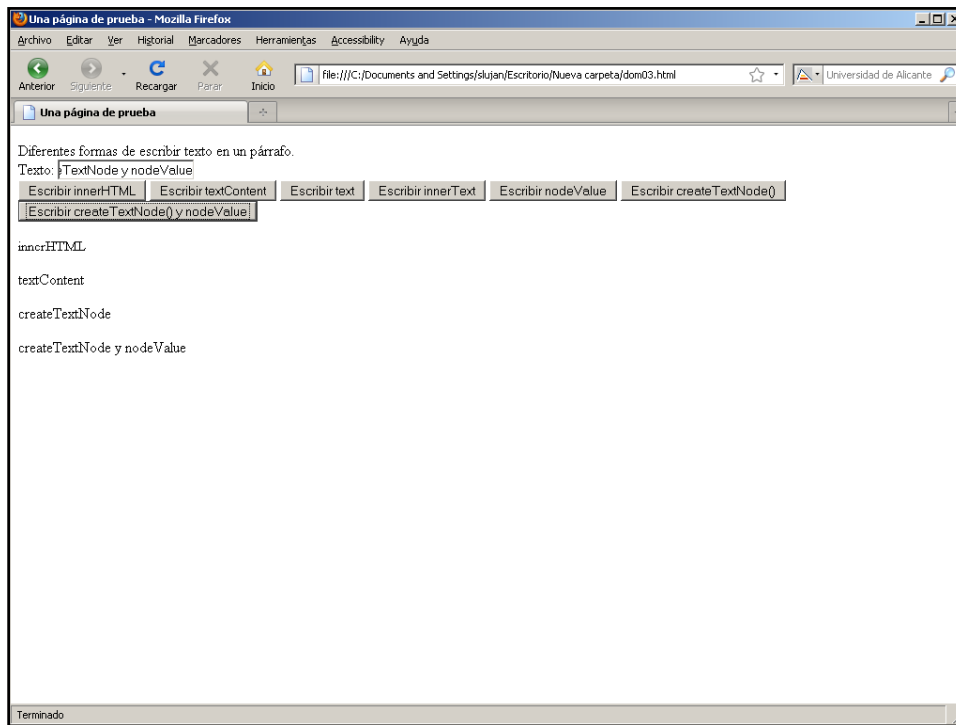
DOM Level 1

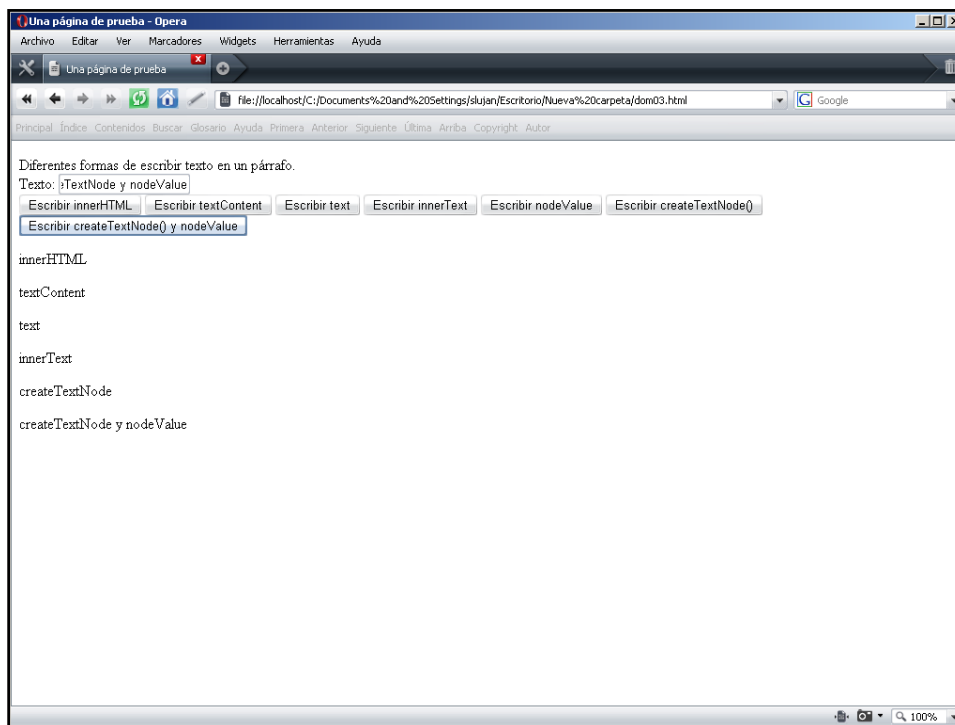
- ¿Cómo se tiene que hacer?

```
var p = document.createElement("p");

// Correcto, parece que funciona en todos
p.appendChild(document.createTextNode(txt));

// Correcto, parece que funciona en todos
var tn = document.createTextNode("");
tn.nodeValue = txt;
p.appendChild(tn);
```





Programación en Internet – Curso 2009-2010

DOM Level 2 - 3

- Versiones de DOM:
 - Nivel 2
 - *Document Object Model Level 2 Core* (13/11/2000)
 - *Document Object Model Level 2 Views*: permite mediante programas acceder dinámicamente a la representación de un documento
 - *Document Object Model Level 2 Events*: proporciona a los programas un sistema genérico de eventos
 - *Document Object Model Level 2 Style*: permite a los programas el acceso y la modificación dinámica de las hojas de estilo asociadas
 - *Document Object Model Level 2 Traversal and Range*: permite a los programas la identificación y el recorrido de rangos de documentos
 - *Document Object Model Level 2 HTML*: permite a los programas el acceso y la modificación dinámica de documentos HTML
 - Nivel 3
 - *Document Object Model Level 3 Core* (7/4/2004)
 - *Document Object Model Level 3 Load and Save*
 - *Document Object Model Level 3 Validation*

Cómo acceder a un formulario

- Tres formas:
 - `document.forms[n]`
 - `document.forms["miForm"]`
 - `document.miForm`
- Estructura general de acceso a un control:
 - `document.nForm.nControl.propiedad`

Cómo acceder a un formulario

- Campos de verificación (checkbox)
 - `checked`
 - `defaultChecked`
 - `value`
- Campos excluyentes (radio)
 - `checked`
 - `defaultChecked`
 - `value`

Cómo acceder a un formulario

- Campos de texto y áreas de texto
 - `defaultValue`
 - `value`
- Listas de selección:
 - `length`
 - `options` → `selected`, `text`, `value`
 - `selectedIndex`

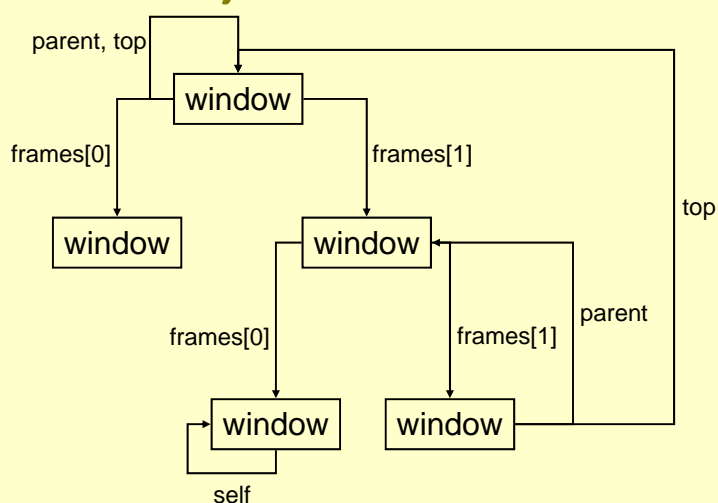
BOM

- *Browser Object Model*
- En proceso de estandarización por W3C:
 - Window Object 1.0 W3C Working Draft 07 April 2006
- Objetos:
 - `window`
 - `history`
 - `location`
 - `navigator`
- Raíz: `window` (se puede eliminar)
 - `window.navigator.appName`
 - `navigator.appName`

Objeto window

- Objeto principal
- Ventana (body) o marco (frameset)
- Propiedades (algunas son objetos):
 - document, frames, history, location
 - parent, self, top
 - status y defaultStatus (texto de la barra de estado de la ventana del navegador)

Objeto window



Objeto window

- Cada marco es un objeto window
- parent devuelve el objeto window padre en la jerarquía de objetos
- top devuelve el objeto window superior en la jerarquía de antecesores
- Para el objeto window raíz o principal, parent y top apuntan a sí mismo → Útil para saber si hay marcos

Objeto window

- Métodos
 - alert(mensaje)
 - clearInterval(intervalID)
 - clearTimeout(timerID)
 - confirm(mensaje) →
Ok(true)/Cancel(false)
 - prompt(mensaje[, valorInicial])
 - close
 - open(URL, nombre[, características])
 - setInterval(expresión, milisegundos)
 - setTimeout(expresión, milisegundos)

Objeto window

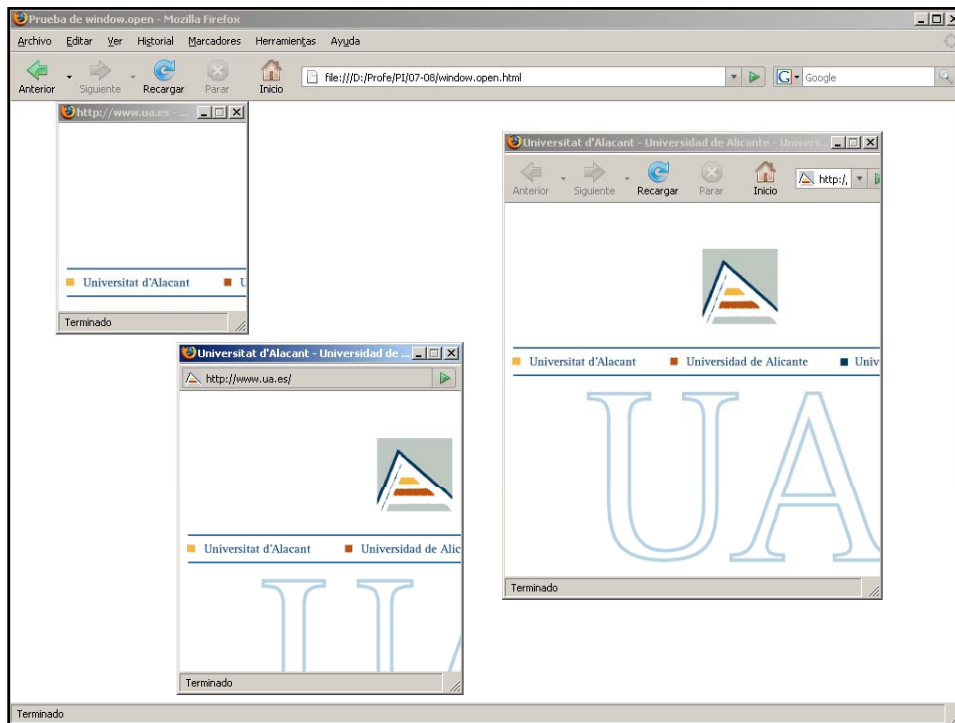
Parámetro	Tipo valor	Descripción
toolbar	boolean	display a toolbar
location	boolean	display the location text box
directories	boolean	display the special link buttons
status	boolean	display a status bar
menubar	boolean	display the menus at the top of the window
scrollbars	boolean	display scrollbars if the document is larger than the window
resizable	boolean	allow the window to be resized
width	integer	the width of the window (in pixels)
height	integer	the height of the window (in pixels)
top	integer	the top position of the window (in pixels)
left	integer	the left position of the window (in pixels)

Objeto window

```

<html>
<head>
<title>Prueba de window.open</title>
<script type="text/javascript">
window.open("http://www.ua.es/", "ventana1", "width=200,
height=200, toolbar=no, menubar=no, location=no,
directories=no");
window.open("http://www.ua.es/", "ventana2", "width=300,
height=300, location=yes");
window.open("http://www.ua.es/", "ventana3", "width=400,
height=400, menubar=no, toolbar=yes, location=yes");
</script>
</head>
<body>
</body>
</html>

```



Programación en Internet – Curso 2009-2010

Objeto history

- Array de URLs visitadas en la ventana actual
- Métodos:
 - back: página visitada anteriormente.
 - forward: siguiente página del histórico
 - go(n): ir a la página 'n' del histórico

Objeto location

- Contiene la URL del objeto window
- Propiedades:
 - host, hostname, href, pathname, port, protocol, search, hash
- Métodos:
 - reload, replace

Objeto navigator

- Información sobre el navegador (el programa cliente)
- Propiedades:
 - appName, appVersion, platform

Programación en Internet – Curso 2009-2010

Objeto navigator

```
<html>
<head>
<title>Ejemplo de uso del objeto location</title>
</head>
<body>
<p>
<script type="text/javascript">
document.writeln("navigator.appCodeName = " + navigator.appCodeName);
document.writeln("<br />");
document.writeln("navigator.appName = " + navigator.appName);
document.writeln("<br />");
document.writeln("navigator.appVersion = " + navigator.appVersion);
document.writeln("<br />");
document.writeln("navigator.language = " + navigator.language);
document.writeln("<br />");
document.writeln("navigator.mimeTypes = " + navigator.mimeTypes);
document.writeln("<br />");
```

Programación en Internet – Curso 2009-2010

Objeto navigator

```
document.writeln("navigator.platform = " + navigator.platform);
document.writeln("<br />");
document.writeln("navigator.plugins = " + navigator.plugins);
document.writeln("<br />");
document.writeln("navigator.userAgent = " + navigator.userAgent);
document.writeln("<br />");
</script>
</p>
</body>
</html>
```

