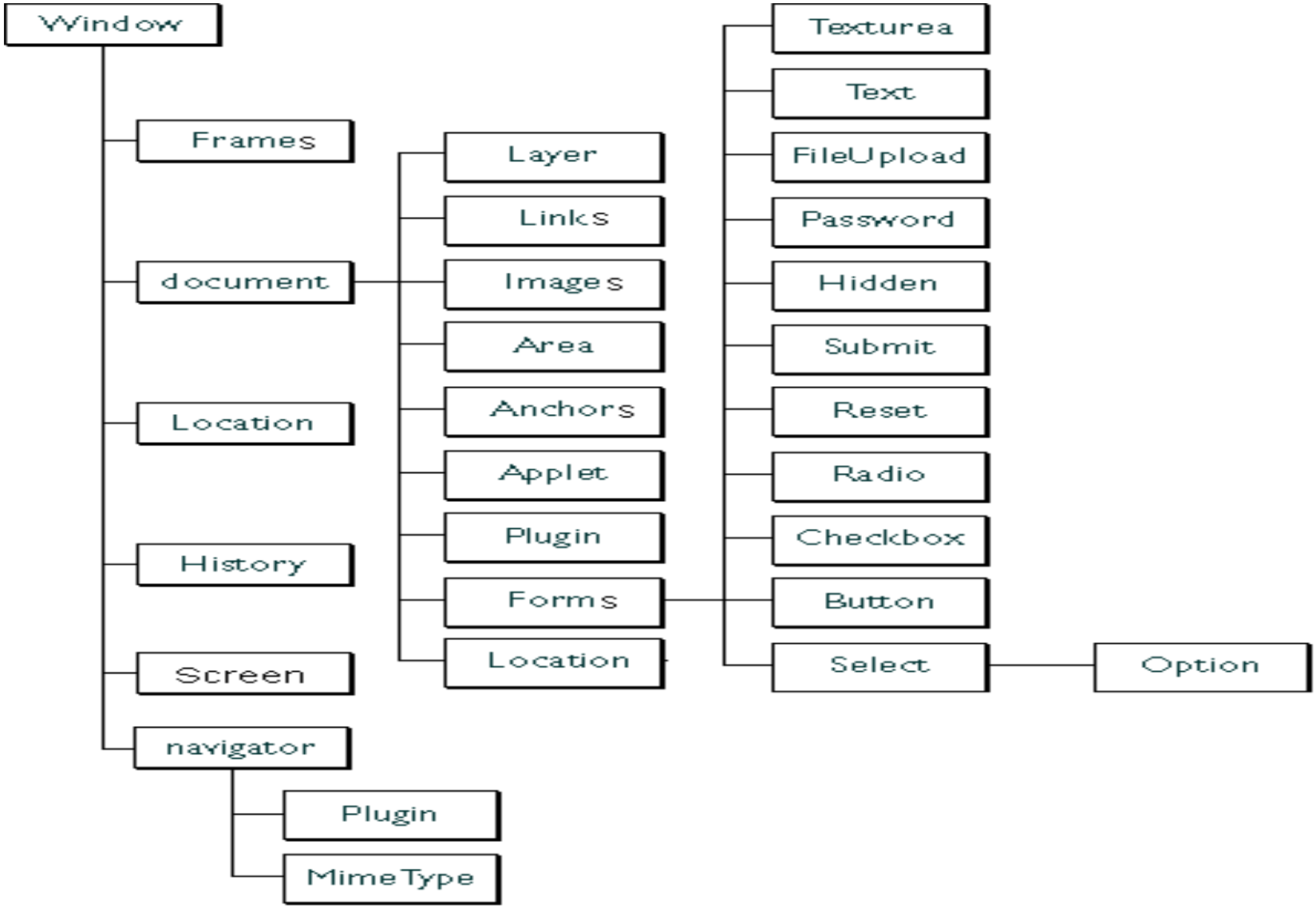
BOM

Browser Object Model

Introducción

- El BOM o Browser Object Model, permite acceder y modificar las propiedades de las ventanas del propio navegador.
- Es posible redimensionar y mover la ventana del navegador, modificar el texto que se muestra en la barra de estado y realizar muchas otras manipulaciones no relacionadas con el contenido de la página HTML.
- El mayor inconveniente de BOM es que, al contrario de lo que sucede con DOM, ninguna entidad se encarga de estandarizarlo o definir unos mínimos de interoperabilidad entre navegadores.

Jerarquía de objetos que forman el BOM



Objeto window – propiedades

- frames: array de frames.
- length: número de frames.
- name: nombre de la ventana (creada por ejemplo con open()).
- parent: ventana padre.
- top: ventana del nivel superior.
- status: texto de la barra de estado (para un mensaje temporal).
- defaultStatus: texto por defecto de la barra de estado.

Objeto window – frames

- Si una página emplea frames, cada uno de ellos se almacena en el array frames, que puede ser accedido numéricamente (window.frames[0]) o, si se ha indicado un nombre al frame, mediante su nombre (window.frames["nombre del frame"]).
- Como todos los demás objetos heredan directa o indirectamente del objeto window, no es necesario indicarlo de forma explícita en el código JavaScript. En otras palabras:

  window.frames[0] == frames[0]

  window.document == document == document (DOM)

Objeto window – métodos

- alert(mensaje): abre una ventana con un mensaje de alerta.
- confirm(mensaje): similar.

  var confirmar= window.confirm(‘Seguir jugando?');
- prompt(mensaje, [Texto por defecto]) : entrada de texto en una ventana de dialogo.

  var texto = window.prompt(‘Introduce un texto');

  document.write('
 Has introducido ' + texto);
- open(url, nombre, [propiedades]): abre una ventana.

  var newwin = window.open("http://www.san.gva.es", "nueva", "width=400, height=300");
- close(): cierra la ventana.

  newwin.close();

Objeto window – espera

- El método setTimeout() permite ejecutar una función al transcurrir un determinado periodo de tiempo.
- Cuando se establece una cuenta atrás, la función setTimeout() devuelve el identificador de esa nueva cuenta atrás. Empleando ese identificador y la función clearTimeout() es posible impedir que se ejecute el código pendiente.

```
function muestraMensaje() {
alert("Han transcurrido 3 segundos desde que me programaron");
}
var id = setTimeout(muestraMensaje, 3000);
// Antes de que transcurran 3 segundos, se decide eliminar la
   ejecución pendiente
clearTimeout(id);
```

Objeto window – repetición

- El método setInterval() permite establecer la ejecución periódica y repetitiva de una función.
- También existe un método que permite eliminar una repetición periódica y que en este caso se denomina clearInterval():

```
function muestraMensaje() {
alert("Este mensaje se muestra cada segundo");
}
var id = setInterval(muestraMensaje, 1000);
// Despues de ejecutarse un determinado número de veces, se elimina
   el intervalo
clearInterval(id);
```

Objeto window – manipular

- Representa la ventana completa del navegador.
- BOM define cuatro métodos para manipular el tamaño y la posición de la ventana:
  - moveBy(x, y) desplaza la posición de la ventana x píxel hacia la derecha y y píxel hacia abajo. Se permiten desplazamientos negativos para mover la ventana hacia la izquierda o hacia arriba.
  - moveTo(x, y) desplaza la ventana del navegador hasta que la esquina superior izquierda se encuentre en la posición (x, y) de la pantalla del usuario. Se permiten desplazamientos negativos, aunque ello suponga que parte de la ventana no se visualiza en la pantalla.
  - resizeBy(x, y) redimensiona la ventana del navegador de forma que su nueva anchura sea igual a (anchura_anterior + x) y su nueva altura sea igual a (altura_anterior + y). Se pueden emplear valores negativos para reducir la anchura y/o altura de la ventana.
  - resizeTo(x, y) redimensiona la ventana del navegador hasta que su anchura sea igual a x y su altura sea igual a y. No se permiten valores negativos.

Objeto window – averiguar

- Para averiguar la posición y tamaño actual de la ventana (la ausencia de un estándar para BOM provoca que cada navegador implemente su propio método):
  - Internet Explorer proporciona las propiedades window.screenLeft y window.screenTop para obtener las coordenadas de la posición de la ventana. No es posible obtener el tamaño de la ventana completa, sino solamente del área visible de la página (es decir, sin barra de estado ni menús). Las propiedades que proporcionan estas dimensiones son document.body.offsetWidth y document.body.offsetHeight.
  - Los navegadores de la familia Mozilla, Safari y Opera proporcionan las propiedades window.screenX y window.screenY para obtener la posición de la ventana. El tamaño de la zona visible de la ventana se obtiene mediante window.innerWidth y window.innerHeight, mientras que el tamaño total de la ventana se obtiene mediante window.outerWidth y window.outerHeight.

Objeto document

- El objeto document es el único que pertenece tanto al DOM como al BOM. Desde el punto de vista del BOM, el objeto document proporciona información sobre la propia página HTML.

Objeto document – propiedades

- title: texto de la etiqueta <title> (título del documento).
- bgColor y fgColor: Color del fondo y del texto del documento.
- linkColor, alinkColor, vlinkColor: Colores de los enlaces.
- cookie: Cadena con el valor del cookie asociado al documento.
- lastModified: la fecha de la última modificación de la página
- referrer: la URL desde la que se accedió a la página (es decir, la página anterior en el array history)
- URL: la URL de la página actual del navegador

Las propiedades son de lectura y escritura, por lo que además de obtener su valor, se puede establecer de forma directa:

```
// modificar el título de la página
document.title = "Nuevo titulo";
// llevar al usuario a otra página diferente
document.URL = "http://nueva_pagina";
```

Objeto document – arrays

- El objeto document contiene varios arrays con información sobre algunos elementos de la página:
  - Anchors: contiene todas las "anclas" de la página (los enlaces de tipo)
  - Applets: contiene todos los applets de la página
  - Embeds: contiene todos los objetos embebidos en la página mediante la etiqueta <embed>
  - Forms: contiene todos los formularios de la página
  - Images: contiene todas las imágenes de la página
  - Links: contiene todos los enlaces de la página (los elementos de tipo)

  Los elementos de cada array del objeto document se pueden acceder mediante su índice numérico o mediante el nombre del elemento en la página HTML.

Objeto document – métodos

- clear(): borra el documento.
- write(..) y writeln(..): añade texto al documento.
- open(..): abre un bloque de escritura para añadir nuevos contenidos (una vez cargado todo el documento HTML en el browser, para añadir contenidos con document.write es necesario abrir un bloque, en caso contrario podría no mostrarse).
- close(): cierra el bloque de escritura, visualizando el nuevo contenido (creado con document.write).

  nota: aunque no utilicemos los métodos open() y close(), los navegadores lo soportan.

```
...
document.open();
document.write("<p>nuevoEl contenido anterior ha sido
borrado por motivos de ....");
document.write("<p> Para volver a visualizarlo, pulsa
el botón de recargar");
document.close();
....
```

DOM Y BOM

- Para acceder a los elementos de la página se pueden emplear las funciones DOM o los objetos de BOM:
  - Párrafo: document.getElementsByTagName("p") (DOM)
  - Enlace: document.links[0]
  - Imagen: document.images[0] o document.images["logotipo"]
  - Formulario: document.forms[0] o document.forms["consultas"]

- Una vez obtenida la referencia al elemento, se puede acceder al valor de sus atributos HTML utilizando las propiedades de DOM. De esta forma, el método del formulario se obtiene mediante document.forms["consultas"].method y la ruta de la imagen es document.images[0].src.

Objeto location – propiedades (I)

- El objeto location representa la URL de la página HTML que se muestra en la ventana del navegador . Debido a la falta de estandarización, location es una propiedad tanto del objeto window como del objeto document.
- Propiedades:
  - hash: el contenido de la URL que se encuentra después del signo # (para los enlaces de las anclas)
    http://www.ejemplo.com/ruta1/ruta2/pagina.html#seccion
    hash = #seccion
  - host: el nombre del servidor
    http://www.ejemplo.com/ruta1/ruta2/pagina.html#seccion
    host = www.ejemplo.com
  - hostname: la mayoría de las veces coincide con host, aunque en ocasiones, se eliminan las www del principio
    http://www.ejemplo.com/ruta1/ruta2/pagina.html#seccion
    hostname = www.ejemplo.com
  - href: la URL completa de la página actual
    http://www.ejemplo.com/ruta1/ruta2/pagina.html#seccion
    URL = http://www.ejemplo.com/ruta1/ruta2/pagina.html#seccion

Objeto location – propiedades (II)

- pathname: todo el contenido que se encuentra después del host
  http://www.ejemplo.com/ruta1/ruta2/pagina.html#seccion
  pathname = /ruta1/ruta2/pagina.html
- port: si se especifica en la URL, el puerto accedido
  http://www.ejemplo.com:8080/ruta1/ruta2/pagina.html#seccion
  port = 8080
  La mayoría de URL no proporcionan un puerto, por lo que su contenido es vacío
  http://www.ejemplo.com/ruta1/ruta2/pagina.html#seccion
  port = (vacío)
- protocol: el protocolo empleado por la URL, es decir, todo lo que se encuentra antes de las dos barras inclinadas //
  http://www.ejemplo.com/ruta1/ruta2/pagina.html#seccion
  protocol = http:
- search: todo el contenido que se encuentra tras el símbolo ?, es decir, la consulta o *"query string"*
  http://www.ejemplo.com/pagina.php?variable1=valor1&variable2=valor2
  search = ?variable1=valor1&variable2=valor2

Objeto location – métodos

- *// Método assign()*

  location.assign("http://www.ejemplo.com");

  // Equivalente a location.href = http://www.ejemplo.com

- *// Método replace()*

  location.replace("http://www.ejemplo.com");

  // Similar a assign(), salvo que se borra la página actual del array history del navegador

- *// Método reload()*

  location.reload(true);

  / Recarga la página. Si el argumento es true, se carga la página desde el servidor. Si es false, se carga desde la cache del navegador */*

Objeto history

- Historial de la páginas visitadas. Se guarda en forma de array.
- Propiedades:
  - length: longitud del historial.
- Métodos:
  - back(): carga el url anterior.

    window.history.back();
  - forward(): carga el url posterior.
  - go(n): carga el url situado dentro del array n lugares a la derecha (si n es positivo), o n lugares a la izquierda (si n es negativo), tomando como referencia la posición del url actual:

    window.history.go(-2);

Objeto screen

- El objeto screen se utiliza para obtener información sobre la pantalla del usuario.
- Propiedades:
  - availHeight: altura de pantalla disponible para las ventanas
  - availWidth: anchura de pantalla disponible para las ventanas
  - colorDepth: profundidad de color de la pantalla (32 bits normalmente)
  - height: altura total de la pantalla en píxel
  - width: anchura total de la pantalla en píxel

El siguiente ejemplo redimensiona una nueva ventana al tamaño máximo posible según la pantalla del usuario:

```
window.moveTo(0, 0);
window.resizeTo(screen.availWidth, screen.availHeight);
```

Objeto navigator – propiedades (I)

- Permite obtener información sobre el propio navegador. En Internet Explorer, el objeto navigator también se puede acceder a través del objeto clientInformation.
- Propiedades:
  - appCodeName: cadena que representa el nombre del navegador (normalmente es Mozilla)
  - appName: cadena que representa el nombre oficial del navegador
  - appMinorVersion: (sólo Internet Explorer) cadena que representa información extra sobre la versión del navegador
  - appVersion: cadena que representa la versión del navegador
  - browserLanguage: cadena que representa el idioma del navegador
  - cookieEnabled : boolean que indica si las cookies están habilitadas
  - cpuClass: (sólo Internet Explorer) cadena que representa el tipo de CPU del usuario ("x86“, "68K", "PPC", "Alpha", "Other")
  - javaEnabled(): boolean que indica si Java está habilitado
  - Language: cadena que representa el idioma del navegador

Objeto navigator – propiedades (II)

- mimeTypes: array de los tipos MIME registrados por el navegador
- onLine: (sólo Internet Explorer) boolean que indica si el navegador está conectado a Internet
- oscpu: (sólo Firefox) cadena que representa el sistema operativo o la CPU
- platform: cadena que representa la plataforma sobre la que se ejecuta el navegador
- plugins: array con la lista de plugins instalados en el navegador
- preference(): (sólo Firefox) método empleado para establecer preferencias en el navegador
- product: cadena que representa el nombre del producto (normalmente, es Gecko)
- productSub: cadena que representa información adicional sobre el producto (normalmente, la versión del motor Gecko)
- securityPolicy: sólo Firefox
- userAgent: información que envía el browser al servidor.

  Mozilla/4.0 (compatible; MSIE 5.01; Windows NT 5.0)

Objeto imagen

- Imágenes incluidas en el documento a partir de la directiva <IMG>.
- Se acceden a ellas a partir del array images de document.
- Propiedades:
  - name: nombre de la imagen.
  - src: dirección url de la imagen.
  - width / height: ancho / alto de la imagen.
  - hspace / vspace: espacio horizontal/vertical que mantiene la imagen con los elementos circundantes.

```
<img name="logotipo" src= "logo.jpg" height= "300" >
...
document.images[0].height = 400;
document.images['logotipo'].height = 400;
document.images.logotipo.height = 400;
```

Objeto link

- Enlaces hipertexto incluidos en el documento, creados a partir de la directiva <A>..</A>.
- Se acceden a partir del array links de document.
- Propiedades:
  - hostname: nombre de la máquina de la url.
  - href: contenido total de la url.
  - protocol: protocolo de la url.
  - port: puerto.
  - target: nombre del frame donde se visualizará el contenido del documento apuntado en la url.

```
for (var i=0; i<document.links.length; i++)
   document.links[i].target = "frame1";
```

Objeto forms

- Formularios incluidos en el documentos a través de la directiva <FORM> ... </FORM>.
- Se acceden a ellos partir del array forms de document.
- Propiedades:
  - action: URL donde se enviarán los datos del formulario.
  - elements: array que contiene todos los elementos del formulario: entradas texto, cajas de selección, botones de radio, áreas de texto, combo box, etc.
  - encoding: tipo de codificación de los datos.
  - method: método de envío (GET o POST).
  - target: frame donde se visualizará el resultado.

Objeto form – eventos

- onreset() y onsubmit(): manejadores de los eventos generados tras pulsar el botón de reset y de submit, respectivamente.

```
function CompruebaNombre() {
    a = document.forms[0].elements[0].value;
    // Las dos siguientes son equivalentes a la primera:
    a = document.forms["formulario1"].elements[“nombre"].value;
    a = document.forms.formulario1.nombre.value;
}
document.forms.formulario1.onsubmit=function(){
    alert('Mensaje2');
}
...
<form name=“formulario1” action='' onSubmit="alert('Mensaje1');">
Identificate:<input type=“text” name=“nombre”><br>
<input type=“”button” value=“ok” name=“confirma”
    onClick=“”CompruebaNombre();”><br>
<input type="submit"></form>
```

Objeto elements

- Elementos incluidos dentro del formulario. Se acceden a ellos a través del vector elements del objeto form, o a través de su nombre (name).
- Propiedades:
  - form: referencia al formulario al que pertenece.
  - type: tipo de objeto.
  - value: valor asociado.
  - Otros particulares de cada tipo de elemento ...
- Manejadores de eventos:
  - onfocus(), onblur(): generales a todos los elementos.
  - onchange(): sólo para Password, Text, Textarea y Select.
  - onclick(): Button, Checkbox, Radio, Reset, Submit.
  - ondblclick(): Button, Reset, Submit.

Cookies (I)

- En javascript, se accede a los datos del cookie a partir de la propiedad cookie de document:

```
var micookie = document.cookie;
alert(micookie);
document.cookie = “nombre=jorge;apellidos=barbera”;
```

- Para que se grabe en disco debe fijarse una fecha de expiración. En caso contrario, sólo se mantiene temporalmente en memoria. Para fijar dicha fecha se utiliza la variable expires.

```
function AddDataCookie(variable, dato, miliseg_vida) {
    if ( miliseg_vida ) {
        var fecha = new Date();
        fecha.setTime(fecha.getTime() + miliseg_vida);
        document.cookie = (variable + '=' + escape(dato) +
        ';expires=' + fecha.toGMTString());
    }
    else
        document.cookie = variable + '=' + escape(dato);
}
```

Cookies (II)

```
function GetDataCookie(variable) {
    var pos_ini, pos_fin, pos, mycook = document.cookie;
    if (!mycook) return "";
            pos = mycook.indexOf(variable);
            if ( pos == -1 ) return "";
            pos_ini = mycook.indexOf("=", pos);
            if ( pos_ini != (pos + variable.length) ) return "";
            pos_fin = mycook.indexOf(";", pos_ini) - 1;
            if (pos_fin == -2) pos_fin = mycook.length - 1;
            return unescape( mycook.substr(pos_ini+1,(pos_fin-pos_ini)) );
}
function DeleteVarCookie(variable) {
    var fecha_cad = new Date();
    fecha_cad.setTime(fecha_cad.getTime() - 1000);
    document.cookie = variable + "=NULL;expires=" +
    fecha_cad.toGMTString();
}
```

Enlaces

- Utilizando los métodos del objeto window, junto con funciones javascript que se ejecutan con temporizadores sincronizados con un video, obtenemos:

  http://www.fly-a-balloon.be

- Propiedades y métodos que soporta cada navegador:

  http://help.dottoro.com/ljrobjtw.php