TEMA 3. OBJETOS PREDEFINIDOS DE JAVASCRIPT.

Objetivos

- Conocer las principales objetos predefinidos del lenguaje Javascript.
- Comprender las propiedades y métodos de los principales objetos del lenguaje.
- Aprender a generar HTML desde Javascript.
- Dominar el uso de marcos de HTML y realizar interacciones entre ellos con Javascript.
- Manipular y gestionar la creación y apariencia de ventanas en el navegador y la comunicación entre ellas.

Contenidos

3.1.- Introducción a los objetos predefinidos en Javascript.

Javascript dispone de una serie de objetos predefinidos como ventana, documento, formulario, ubicación, marco,... Estos objetos son elementos programables que podemos cambiar sus propiedades y realizar tareas a través de sus métodos o ejecutar un evento relacionado con el objeto.
Las propiedades, métodos y eventos son conceptos fundamentales para comprender el funcionamiento del objeto.

Las **propiedades** son características del objeto. Por ejemplo el Color del objeto página es una propiedad (bgColor).

Los **métodos** son funciones que puede realizar el objeto. Por ejemplo sqrt() es método del objeto Math que permite calcular la raíz cuadrada de un número.

Un **evento** es una situación que puede suceder sobre un objeto. Por ejemplo pulsar sobre una imagen.

3.2.- Objetos nativos de Javascript.

Podemos crear un objeto con la palabra clave **new.**

```
var mi_objeto = new Object();
```

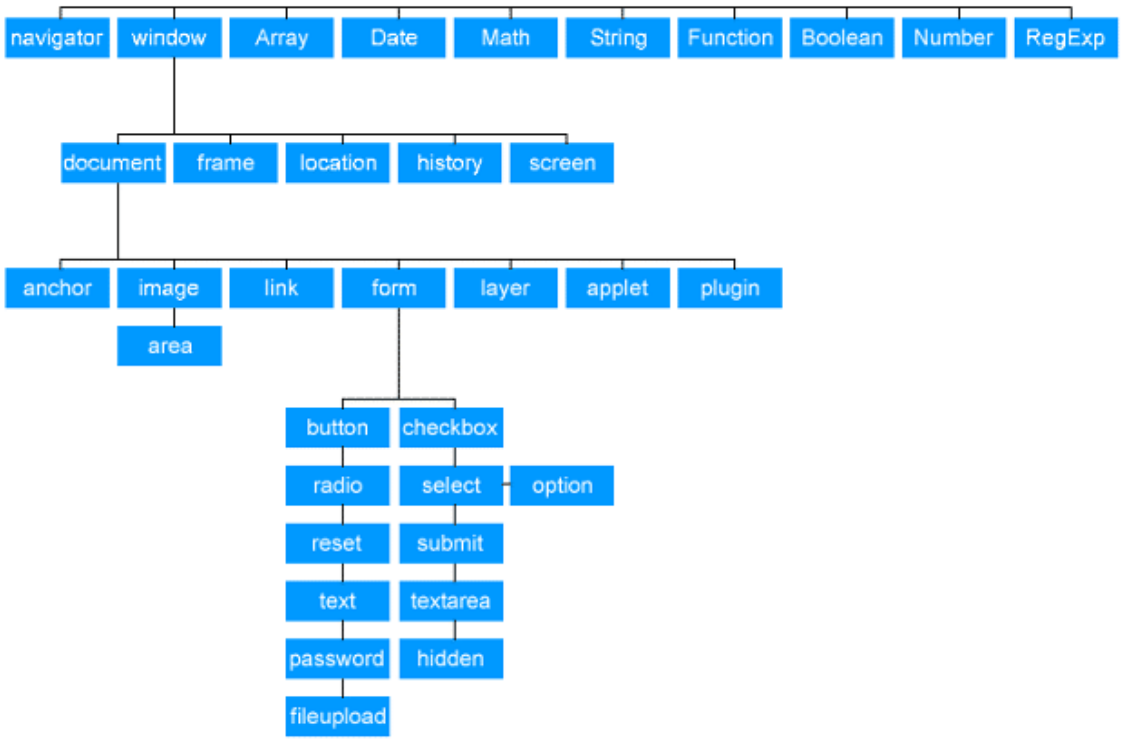
Podemos acceder a las propiedades del objeto utilizando

```
mi_objeto.nombre_propiedad;
```

Podemos ejecutar métodos de un objeto utilizando

```
mi_objeto.nombre_método([parámetros]);
```

Los objetos en Javascript se organizan de modo jerárquico siendo **window** el objeto más alto ya que representa la ventana del navegador. Al mismo nivel se encuentran oreos objetos predefinidos.



a) Objeto Date.

Permite realizar controles relacionados con el tiempo en aplicaciones web. No tiene propiedades. Tiene métodos divididos en tres grupos:

- **Métodos de lectura**. Utilizan prefijo **get** y permiten consultar un objeto de tipo Date.
- **Métodos de escritura**. Utilizan prefijo **set** y permiten inicializar partes de un objeto de tipo Date.
- **Métodos de conversión**. Convierten objetos de tipo Date en cadenas de texto o en milisegundos.

Método	Descripción
`getDate()`	Devuelve el número correspondiente al día del mes entre 1 y 31.
`getDay()`	Devuelve el número correspondiente al día de la semana entre 0 (domingo) y 6 (sábado).
`getFullYear()`	Devuelve un número de 4 cifras correspondiente al año.
`getHours()`	Devuelve la hora entre 0 y 23.
`getMilliseconds()`	Devuelve los milisegundos entre 0 y 9999.
`getMinutes()`	Devuelve los minutos entre 0 y 59.
`getMonth()`	Devuelve el mes entre 0 (enero) y 11 (diciembre).
`getSeconds()`	Devuelve los segundos entre 0 y 59.
`getTime()`	Devuelve el número de milisegundos transcurridos desde el 1 de enero de 1970.

parse()	Analiza una fecha y devuelve el número de milisegundos pasados desde el 1 de enero de 1970 hasta la fecha analizada.
setDate()	Establece el valor del día del mes.
setFullYear()	Establece el valor del año.
setHours()	Establece el valor de la hora.
setMilliseconds()	Establece el valor de los milisegundos.
setMinutes()	Establece el valor de los minutos.
setMonth()	Establece el valor del mes.
setSeconds()	Establece el valor de los segundos.
setTime()	Establece una fecha, agregando o sustrayendo un número específico de milisegundos hasta o desde la medianoche del 1 de enero de 1970.
toLocaleDateString()	Convierte la fecha del objeto Date en una cadena de caracteres, según las convenciones locales.
toLocaleTimeString()	Devuelve la parte del tiempo de un objeto Date en una cadena según las convenciones locales.

Podemos instanciar un objeto de tipo Date con la fecha actual o con una fecha cualquiera.

El objeto Date permite manipular fechas y horas. La información horaria se almacena del siguiente modo:

- Año con 4 digitos.
- Mes en un array de 0 a 11
- Día de 1 a 31
- Día de la semana en un array de 0 a 6
- Hora de 0 a 23
- Minutos de 0 a 59
- Segundos de 0 a 59
- Milisegundos de 0 a 999
- Uso horario como desplazamiento en horas respecto al meridiano Greenwich.

Podemos crear un objeto Date del siguiente modo:

```
var fecha = new Date();      // Almacena la fecha actual
```

Podemos crear una fecha con las 8:30:15 del día 15 de Marzo de 2013:

```
var fecha = new Date(2013, 2, 15, 8, 30, 15);
```

Ejemplo:

```
var fecha_actual = new Date();
var fecha_nac = new Date(1989, 10, 15);
var tiempo = fecha_actual-fecha_nac;
document.write("Fecha Actual: "+fecha_actual+"<br>");
document.write("Fecha Nac: "+fecha_nac+"<br>");
document.write("Le edad es "+
Math.floor(tiempo/(12*31*24*60*60*1000))+" años<br>");
```

```
document.write("Año: "+fecha_actual.getFullYear()+"<br>");
document.write("Mes: "+fecha_actual.getMonth()+"<br>");
document.write("Día: "+fecha_actual.getDate()+"<br>");
document.write("Día Semana: "+fecha_actual.getDay()+"<br>");
document.write("Horas: "+fecha_actual.getHours()+"<br>");
document.write("Minutos: "+fecha_actual.getMinutes()+"<br>");
document.write("Segundos: "+fecha_actual.getSeconds()+"<br>");

fecha_nac.setFullYear(1970);
fecha_nac.setMonth(1);
fecha_nac.setDate(24);
fecha_nac.setHours(15);
fecha_nac.setMinutes(5);
fecha_nac.setSeconds(17);
document.write("Año: "+fecha_nac.getFullYear()+"<br>");
document.write("Mes: "+fecha_nac.getMonth()+"<br>");
document.write("Día: "+fecha_nac.getDate()+"<br>");
document.write("Día Semana: "+fecha_nac.getDay()+"<br>");
document.write("Horas: "+fecha_nac.getHours()+"<br>");
document.write("Minutos: "+fecha_nac.getMinutes()+"<br>");
document.write("Segundos: "+fecha_nac.getSeconds()+"<br>");

document.write("Fecha Completa: "+fecha_nac.toLocaleDateString()+"<br>");
document.write("Hora Completa:  "+fecha_nac.toLocaleTimeString()+"<br>");
```

Función SetTimeout().

Permite definir la función que se ejecutará en el momento en que pase un tiempo determinado.

```
var mitiempo = setTimeout("nombrefunción", lapsomiliseg);
```

La función **`clearTimeout(mitiempo)`** provoca que la función no vuelva a ser llamada.

Función setInterval().

Se ejecuta una función indefinidamente cada cierto intervalo de tiempo.

```
var mitiempo = setInterval("nombrefunción", lapsomiliseg);
```

La función **`clearInterval(mitiempo)`** provoca que se interrumpa el proceso.

Ejemplo:

```
var tiempo_aviso = setTimeout("alert('Han pasado 5 seg.')", 5000);
document.write("<p onclick='clearTimeout(tiempo_aviso)'>Parar</p>");
var tiempo_aviso_rep = setInterval("alert('Han pasado 10 seg.')",
10000);
document.write("<p onclick='clearInterval(tiempo_aviso_rep)'>Parar
Repeticiones</p>");
```

b) Objeto Math.

Permite hacer operaciones matemáticas con Javascript. Las propiedades almacenan una serie de Constantes que son útiles para realizar cálculos matemáticos.

Propiedad	Descripción
E	Devuelve la constante de Euler.
LN2	Devuelve el logaritmo natural de 2.
LN10	Devuelve el logaritmo natural de 10.
LOG2E	Devuelve el logaritmo de E en base 2.
LOG10E	Devuelve el logaritmo de E en base 10.
PI	Devuelve el valor de Pi.
SQRT1_2	Devuelve la raíz cuadrada de 1/2.
SQRT2	Devuelve la raíz cuadrada de 2.

Para utilizar los métodos del objeto Math no necesitamos usar el constructor new. Los datos para operar con los métodos los debemos pasar como argumentos del método.

Método	Descripción
abs()	Devuelve el valor absoluto.
acos()	Devuelve el arcocoseno.
asin()	Devuelve el arcoseno.
atan()	Devuelve el arcotangente.
ceil()	Devuelve el número entero superior.
cos()	Devuelve el coseno.
exp()	Devuelve el exponencial.
floor()	Devuelve el número entero inferior.
log()	Devuelve el logaritmo natural.
max()	Devuelve el número máximo entre los números pasados como argumento.
min()	Devuelve el número mínimo entre los números pasados como argumento.
pow()	Devuelve el resultado de un número elevado a una potencia pasada como argumento.
random()	Devuelve un número aleatorio entre 0 y 1.
round()	Redondea un número al número entero más próximo.
sin()	Devuelve el seno.
sqrt()	Devuelve la raíz cuadrada.
tan()	Devuelve la tangente.

Ejemplo: Calcular el área de un círculo. area = PI * radio^2

```
var radio = prompt("Introduce el radio");
var area = Math.PI * Math.pow(radio, 2);
document.write("El area del circulo de radio "+radio+" es
"+Math.round(area*100)/100);
```

Ejemplo: Máximo de varios valores.

```
var maximo = Math.max(10, 14, 12, 25, 17, 13, 2);
document.write("El máximo es "+maximo);
```

Ejemplo: Simular el lanzamiento de un dado.

```
var dado = parseInt((Math.random()*6)+1);
document.write("El dado vale "+dado);
```

c) Objeto Number.

El objeto Number tiene propiedades que indican si un valor es número o los valores máximo y mínimo permitido.

Propiedad	Descripción
MAX_VALUE	Devuelve el mayor número posible en JavaScript.
MIN_VALUE	Devuelve el menor número posible en JavaScript.
NaN	Representa el valor especial Not a Number.

Los métodos del objeto Number permiten formatear el número pasado como parámetro.

Método	Descripción
toFixed()	Formatea el número con la cantidad de dígitos decimales que pasemos como parámetro.
toPrecision()	Formatea el número con la longitud que pasemos como parámetro.

Ejemplo:

```
document.write("Valor Mínimo en Javascript:"+Number.MIN_VALUE+"<br>");
document.write("Valor Máximo en Javascript:"+Number.MAX_VALUE+"<br>");
var sueldo = 1452.46098;
document.write("El sueldo vale "+sueldo.toFixed(2)+"<br>");
document.write("El sueldo vale "+sueldo.toPrecision(5)+"<br>");
```

d) Objeto String.

El objeto String permite manipular cadenas de texto. El objeto dispone de una propiedad:

Propiedad	Descripción
length	Corresponde a la longitud de una cadena de texto.

El objeto String dispone de multitud de métodos para manipular cadenas de texto.

Método	Descripción
anchor()	Devuelve una cadena convertida en un ancla de HTML.
big()	Aumenta el tamaño de una cadena.
blink()	Crea el efecto de una cadena parpadeando.
bold()	Muestra una cadena en negrita.
charAt()	Permite acceder a un carácter en concreto de una cadena.
charCodeAt()	Devuelve un carácter en concreto en su formato Unicode.
concat()	Concatena dos o más cadenas y devuelve una nueva con dicha concatenación.
fixed()	Convierte una cadena en una cadena con fuente monoespaciada.
fontcolor()	Modifica el color de la fuente de una cadena.
fontsize()	Modifica el tamaño de la fuente de una cadena.
fromCharCode()	Convierte valores Unicode en caracteres.
indexOf()	Devuelve la posición de la primera ocurrencia del carácter pasado como parámetro.
italics()	Muestra una cadena en cursiva.
lastIndexOf()	Devuelve la posición de la última ocurrencia del carácter pasado como parámetro.
link()	Muestra una cadena como un hipervínculo HTML con el enlace le pasemos como parámetro.
match()	Busca una coincidencia en una cadena y devuelve todas las coincidencias encontradas.
replace()	Busca una coincidencia en una cadena y si existe, la remplaza por otra cadena pasada como parámetro.
search()	Busca una coincidencia en una cadena y devuelve la posición de la coincidencia.
slice()	Extrae una parte de una cadena en base a los parámetros que indiquemos como índices de inicio y final.
small()	Disminuye el tamaño de una cadena.
split()	Corta una cadena en base a un separador que pasamos como parámetro.
strike()	Muestra una cadena tachada.
sub()	Muestra una cadena como subíndice.
substr()	Devuelve una subcadena en base a un índice y longitud pasados como parámetros.
substring()	Devuelve una subcadena en base a un índice de inicio y de final pasados como parámetros.
sup()	Muestra una cadena como superíndice.
toLowerCase()	Convierte una cadena en minúsculas.
toUpperCase()	Convierte una cadena en mayúsculas.

Ejemplo:

```
var texto = "Este es un texto en Javascript";
document.write("Longitud del texto es: "+texto.length+" caracteres");
```

Ejemplo:

```
var texto = "Esto es una cadena de caracteres";
document.write(texto+"<br>");
document.write(texto.big()+"<br>");
document.write(texto.blink()+"<br>");
document.write(texto.bold()+"<br>");

document.write(texto.charAt(3)+"<br>");
document.write(texto.charCodeAt(10)+"<br>");

var texto1 = "Esto es ";
var texto2 = "una prueba";
var resultado;
resultado = texto1.concat(texto2);
document.write(resultado+"<br>");

document.write(texto.fixed()+"<br>");
document.write(texto.fontcolor('#FF0000')+"<br>");
document.write(texto.fontsize(24)+"<br>");
document.write(String.fromCharCode(97)+"<br>");

document.write(texto.indexOf('a')+"<br>");
document.write(texto.lastIndexOf('a')+"<br>");

document.write(texto.italics()+"<br>");
document.write(texto.link('http://www.google.es')+"<br>");

document.write(texto.match(/es/gi)+"<br>");
// Devuelve array con todas las ocurrencias encontradas
// Entre / la cadena que queremos buscar.
// g busca en toda la cadena
// i no distingue entre mayúsculas y minúsculas.

document.write(texto.replace(/una cadena/gi, 'un texto')+"<br>");
document.write(texto.search('caracteres')+"<br>");
// Empieza en 0
document.write(texto.slice(5)+"<br>");
// Sin segundo parámetro extrae hasta el final
document.write(texto.slice(5, 18)+"<br>");
//Inicio y fin sin extraer el caracter final

document.write(texto.small()+"<br>");
document.write(texto.strike()+"<br>");
document.write("Texto: "+texto.sup()+"<br>");
document.write("Texto: "+texto.sub()+"<br>");
```

```
var cadena ="Grado de Ciclo Superior";
document.write(cadena.toLowerCase()+"<br>");
document.write(cadena.toUpperCase()+"<br>");

var resultado = cadena.split(" ");
// Introduce cada palabra en una posición del array (espacio)
document.write(resultado+"<br>");

document.write(cadena.substr(9)+"<br>");
// Sin segundo parámetro extrae hasta el final
document.write(cadena.substr(9,5)+"<br>");
// Parámetros Inicio y Número de Caracteres
document.write(cadena.substring(9)+"<br>");
// Sin segundo parámetro extrae hasta el final
document.write(cadena.substring(9,14)+"<br>");
// Parámetros Inicio y Fin. No extrae el Carácter final
```

3.3.- Interacción de los objetos con el navegador.

Existen objetos que permiten el control del navegador como mostrar mensajes en la barra de estado o la creación de nuevas ventanas.

a) Objeto Navigator.

El objeto Navigator permite identificar las características la plataforma sobre la que se ejecuta la aplicación. Esta información permite tomar decisiones en la aplicación dependiendo de la información obtenida y adaptar fragmentos de código dependiendo del navegador utilizado.

Las propiedades del objeto Navigator son:

Propiedad	Descripción
`cookieEnable`	Determina si las *cookies* están habilitadas o no.
`platform`	Devuelve la plataforma sobre la cual se está ejecutando el navegador.
`userAgent`	Devuelve una información completa sobre el agente de usuario, el cual es normalmente el navegador.

Los métodos del objeto Navigator son:

Método	Descripción
`javaEnable()`	Informa si el navegador está habilitado para soportar la ejecución de programas escritos en Java.

La propiedad plataform devuelve la plataforma sobre la que se ejecuta el navegador. Posibles valores son:

- HP-UX
- Linux i686
- Linux armv7l
- Mac68K
- MacPPC
- MacIntel
- SunOS
- Win16
- Win32
- WinCE
- Etc..

La propiedad userAgent devuelve completa información del navegador utilizado. Determinamos el navegador utilizado si en la información que devuelve la propiedad contiene una determinada palabra:

Palabra	Navegador
Chrome	Google Chrome
Firefox	Mozilla Firefox
Safari	Safari
Opera	Opera
Trident	Internet Explorer
MSIE	Internet Explorer
Edge	Microsoft Edge

Ejemplo:

```
<script type="text/javascript">
 if (navigator.javaEnabled()){
    document.write("Navegador soporta Applets Java<br>");
 } else {
    document.write("Navegador NO soporta Applets Java<br>");
 }
 document.write("Cookies Habilitadas: " + navigator.cookieEnabled + "<br>");
 document.write("Lenguaje del Navegador: " + navigator.language + "<br>");
 document.write("Platforma Cliente: " + navigator.platform + "<br>");
 document.write("Datos Navegador: " + navigator.userAgent + "<br>");
</script>
```

Ejemplo: Detectar el navegador con el que accedemos a la web.

```
<script type="text/javascript">
      var agente = window.navigator.userAgent;
      var navegadores = [["Chrome","Google Chrome"],
                         ["Firefox","Mozilla Firefox"],
                         ["Safari","Safari"],
                         ["Opera","Opera"],
                         ["Trident","Internet Explorer"],
                         ["MSIE","Internet Explorer"],
                         ["Edge","Microsoft Edge"]];

      for(var i in navegadores){
        if(agente.indexOf( navegadores[i][0]) != -1 ){
                  document.write(navegadores[i][1]);
                  break;
        }
      }
</script>
```

b) Objeto Screen.

El objeto Screen corresponde con la pantalla utilizada por el usuario. Posee propiedades de lectura y ningún método.

Propiedad	Descripción
availHeight	Corresponde a la altura disponible de la pantalla para el uso de ventanas.
availWidth	Corresponde a la anchura disponible de la pantalla para el uso de ventanas.
colorDepth	Corresponde al número de colores que puede representar la pantalla.
height	Corresponde a la altura total de la pantalla.
pixelDepth	Corresponde a la resolución de la pantalla expresada en bits por píxel.
width	Corresponde a la anchura total de la pantalla.

La altura y anchura disponible es menor que la altura y anchura total de pantalla ya que el Sistema Operativo utiliza parte de la pantalla para la barra de tareas, menús, etc...

Ejemplo:

```
<script type="text/javascript">
      document.write("Altura total: " + screen.height + "<br>");
      document.write("Anchura total: " + screen.width + "<br>");
      document.write("Altura disponible: "+screen.availHeight+"<br>");
      document.write("Anchura disponible: "+screen.availWidth+"<br>");
</script>
```

c) Objeto Window.

El objeto Window permite gestionar la ventana del navegador. El objeto Window dispone de las siguiente propiedades:

Propiedad	Descripción
closed	Corresponde al valor booleano que indica si la ventana está cerrada o no.
defaultStatus	Corresponde a la cadena de texto de la barra de estado del navegador.
document	Corresponde al documento actual de la ventana.
frames	Corresponde al conjunto de marcos de la ventana.
history	Corresponde al conjunto de elementos que representan las URL visitadas.
innerHeight	Corresponde a la altura utilizable de la ventana.
innerWidth	Corresponde al ancho utilizable de la ventana.
length	Corresponde al número de frames de la ventana.
location	Corresponde a la URL de la barra de direcciones.
locationbar	Corresponde a la barra de direcciones del navegador.
menubar	Corresponde a la barra de menú del navegador.
name	Corresponde al nombre de la ventana.
opener	Corresponde a la referencia al objeto Window que haya abierto una ventana nueva.
outerHeight	Corresponde a la altura exterior de la página.
outerWidth	Corresponde al ancho exterior de la página.

Propiedad	Descripción
pageXoffset	Corresponde a la posición horizontal de la ventana.
pageYoffset	Corresponde a la posición vertical de la ventana.
parent	Corresponde a la referencia al objeto Window que contiene los marcos de una página de marcos.
personalbar	Corresponde a la barra de herramientas personal.
scrollbars	Corresponde a las barras de desplazamiento vertical y horizontal.
self	Corresponde a la ventana actual.
status	Corresponde a la cadena con el mensaje que contiene la barra de estado.
toolbar	Corresponde a la barra de herramientas del navegador.
top	Corresponde a la ventana de nivel superior.

El objeto Window dispone de los siguientes métodos:

Método	Descripción
alert()	Genera un cuadro de diálogo con un mensaje y un botón *Aceptar*.
back()	Regresa a la página anterior según el historial.
blur()	Desactiva una página.
close()	Cierra una página.
confirm()	Genera un cuadro de diálogo con los botones *Aceptar* y *Cancelar*.
find()	Realiza una búsqueda de texto en una página.
focus()	Activa una ventana.
forward()	Avanza una página según el historial.
home()	Carga la página definida como página por defecto del navegador.
moveTo()	Mueve la ventana activa.
open()	Abre una nueva ventana.
print()	Imprime una página.
prompt()	Genera un cuadro de diálogo con un cuadro de texto para que el usuario introduzca valores.
resizeTo()	Modifica el tamaño de una ventana.
setInterval()	Evalúa una expresión después de un tiempo especificado.
setTimeOut()	Inicia un registro de tiempo.
scrollBy()	Mueve el contenido de una ventana en función de una cantidad especificada en píxeles.
scrollTo()	Mueve el contenido de una ventana especificando una nueva posición de la esquina superior izquierda.
stop()	Detiene una página.

Modelo BOM (Browser Object Model) permite realizar diversas acciones sobre los elementos de una página, como redimensionar y mover la ventana del navegador. Inconveniente de BOM es que no está estandarizado entre los distintos tipos de navegadores.

Las principales porpiedades del objeto window son:

- **screenX, screenLeft**: Estas dos propiedades indican la distancia en píxeles de la ventana al borde izquierdo de la pantalla.
- **screenY, screenTop**: Estas dos propiedades indican la distancia en píxeles de la ventana al borde superior de la pantalla.
- **innerWidth, body.clientWidth**: Estas dos propiedades indican la anchura del área visible del documento.
- **innerHeight, body.clientHeight**: Estas dos propiedades indican la altura del área visible del documento.
- **outerWidth:** Indica la anchura total de la ventana del navegador. En Internet Explorer no existe esta propiedad.
- **outerHeight**: Indica la altura total de la ventana del navegador. En Internet Explorer no existe esta propiedad.
- **defaultStatus**: Texto que se muestra por defecto en la barra de estado.
- **closed():** Verdadero si la ventana está cerrada.

Los principales métodos del objeto window son:

- **focus():** Permite forzar el foco de la ventana.
- **blur():** Quita el foco de la ventana.
- **close():** Cierra la ventana del navegador.
- **home():** Abre la página de inicio.
- **print():** Abre el cuadro de diálogo de la impresora.
- **stop():** Detiene la carga de la página.
- **moveBy(x,y):** Desplaza la ventana x píxeles a la derecha y y píxeles hacia abajo. Si los números son negativos se desplaza hacia la izquierda y hacia arriba.
- **moveTo(x,y):** Desplaza la ventana del navegador a la posición x,y de la pantalla.
- **resizeBy(x,y):** Redimensiona la ventana del navegador sumando a la anchura anterior el valor x y a la altura anterior el valor y. Si los números son negativos se reducen la anchura y la altura.
- **resizeTo(x,y):** Redimensiona la ventana del navegador a una anchura x y altura y.
- **scrollTo(x,y):** Desplaza el contenido de la ventana horizontal o vertical en valores absolutos.
- **scrollBy(x,y):** Desplaza el contenido de la ventana horizontal o vertical en valores relativos.

Ejemplo: (Ejecutar en el Servidor)

```
<script type="text/javascript">
	var ventana;
	function abrir(){
		ventana=window.open('','', 'width=100,height=100');
		ventana.focus();
	}
	function tam(){
		ventana.resizeTo(250, 250);
		ventana.focus();
	}
```

```
	function aumentatam(){
		ventana.resizeBy(20, 20);
		ventana.focus();
	}
	function disminuyetam(){
		ventana.resizeBy(-20, -20);
		ventana.focus();
	}

	function pos(){
		ventana.moveTo(100, 100);
		ventana.focus();
	}

	function aumentapos(){
		ventana.moveBy(20, 20);
		ventana.focus();
	}
	function disminuyepos(){
		ventana.moveBy(-20, -20);
		ventana.focus();
	}

	function estado(){
		if (ventana.closed)
			alert("Ventana Cerrada");
		else
			alert("Ventana Abierta");
	}

	function imprimir(){
		ventana.print();
		ventana.focus();
	}

	function mensaje(texto){
		ventana.document.write(texto+"<br>");
		ventana.focus();
	}
</script>
<input type="button" value="ABRIR" onclick="abrir();">
<input type="button" value="CERRAR" onclick="ventana.close();">
<input type="button" value="CERRADA?" onclick="estado();">
<input type="button" value="TAM" onclick="tam();">
<input type="button" value="TAM +" onclick="aumentatam();">
<input type="button" value="TAM -" onclick="disminuyetam();">
<input type="button" value="POS" onclick="pos();">
<input type="button" value="POS +" onclick="aumentapos();">
<input type="button" value="POS -" onclick="disminuyepos();">
<input type="button" value="QUITAR FOCO" onclick="ventana.blur();">
<input type="button" value="PONER FOCO" onclick="ventana.focus();">
<input type="button" value="IMPRIMIR" onclick="imprimir();">
<input type="button" value="MENSAJE" onclick="mensaje('AulaCampus');">
```

Ejemplo: (Ejecutar en el Servidor)

```
<html>
<head>
<script>
function ventana(){
	var navegador = ((window.navigator.userAgent.indexOf("Trident") != -1)
			||(window.navigator.userAgent.indexOf("MSIE") != -1));
	if (navegador) {
		document.write("<br>PosX: " + window.screenLeft);
		document.write("<br>PosY: " + window.screenTop + "</p>");
		document.write("<br>Ancho: " + document.body.clientWidth);
		document.write("<br>AltoY: " + document.body.clientHeight +
"</p>");
	} else {
		document.write("<br>PosX: " + window.screenX);
		document.write("<br>PosY: " + window.screenY + "</p>");
		document.write("<br>Ancho: " + window.innerWidth);
		document.write("<br>AltoY: " + window.innerHeight + "</p>");
	}
}
</script>
</head>
<body>

<input type="button" value="Datos Ventana" onclick="ventana()">

</body>
</html>
```

Ejemplo: (Ejecutar en el Servidor)

```
<script type="text/javascript">
	function ventanaCondiciones() {
		var tipoNav = 0;
		var navegador = navigator.userAgent;
		if (navegador.indexOf('MSIE') !=-1) {
			tipoNav = 1;
		}
		if (tipoNav == 1) {
			posX = window.screenLeft;
			posY = window.screenTop;
			ancho = document.body.clientWidth;
			alto = document.body.clientHeight;
		} else {
			posX = window.screenX;
			posY = window.screenY;
			ancho = window.innerWidth;
			alto = window.innerHeight;
		}
		anchopop = 400;
		altopop = 200;
		posXpop = posX+((ancho-anchopop)/2);
		posYpop = posY+((alto-altopop)/2);
```

```
      window.open("condiciones.html","Condiciones","left="+posXpop+",
top="+posYpop+", width="+anchopop+", height="+altopop);
      }
</script>
<input type="button" value="Condiciones de Uso"
onclick="ventanaCondiciones();">
```

condiciones.html

```
<b>Condiciones Legales.</b><br>
Estas son las condiciones legales de uso. Estas son las condiciones
legales de uso. Estas son las condiciones legales de uso. Estas son
las condiciones legales de uso. Estas son las condiciones legales de
uso. Estas son las condiciones legales de uso.
```

d) Objeto Document.

El objeto Document se refiere a las páginas que se cargan en el navegador. Con este objeto podemos manipular las propiedades y el contenido de los elementos de las páginas web.

Las principales propiedades del objeto document son:

Propiedad	Descripción
alinkColor	Corresponde al color de los vínculos activos de la página.
anchors	Corresponde a los puntos de anclaje (etiquetas <a name>) del documento.
applets	Corresponde a los *applets* (etiquetas <applet>) Java del documento.
bgColor	Corresponde al color de fondo del documento.
cookie	Corresponde a un fichero guardado en el equipo del cliente del navegador con información sobre el usuario.
domain	Corresponde al nombre del dominio por defecto para el documento.
embeds	Corresponde a los objetos embebidos (etiqueta <embed>) en el documento.
fgColor	Corresponde al color del texto del documento.
forms	Corresponde a los formularios (etiqueta <form>) del documento.
images	Corresponde a las imágenes (etiqueta <images>) del documento.
lastModified	Corresponde a la última fecha en la que se modificó el documento.
layers	Corresponde a las capas (etiqueta <layer>) del documento.
linkColor	Corresponde al color de los enlaces aún no visitados.
links	Corresponde a los vínculos (etiqueta <a href>) del documento.
plugins	Corresponde a las referencias y llamadas de los plugins del documento.
referrer	Corresponde a la dirección del documento usado para ir al documento actual.
title	Corresponde al título (etiqueta <title>) del documento.
URL	Corresponde a la dirección del documento.
vlinkColor	Corresponde al color de los enlaces visitados.

Los principales métodos del objeto document son:

Método	Descripción
captureEvents()	Intercepta un evento para que pueda ser manipulado por el documento.
close()	Cierra el documento activo.
getSelection()	Devuelve el texto seleccionado en el documento.
handleEvent()	Activa el manejador del evento especificado.
home()	Carga la página de inicio.
open()	Activa el documento.
releaseEvents()	Libera los eventos que han sido interceptados.
routeEvents()	Intercepta un evento y lo pasa a lo largo de la jerarquía del objeto que lo lanzó.
write()	Escribe datos en el documento.
writeln()	Escribe datos además de un salto de línea en el documento.

Ejemplo:

```
<html>
<head>
  <title> Ejemplo del Objeto document</title>
</head>
  <body>
    <img name="imagen" src="" border="0">
    <script type="text/javaScript">
      var imagenes = new Array("1.jpg","2.jpg","3.jpg","4.jpg");
      var pos = Math.round(Math.random()*3);
          document.images["imagen"].src = imagenes[pos];
  </script>
</body>
</html>
```

Ejemplo:

```
<html>
  <head>
    <title> Ejemplo del Objeto document</title>
  </head>
  <body>
    <p onclick="location.reload(true);">Actualizar</p>
    <p onclick="document.bgColor='orange';">Color Fondo</p>
    <h1><a name="mienlace" href="">Enlace</a></h1>
    <script type="text/javaScript">
      var enlaces = new Array("http://www.yahoo.es",
                              "http://www.google.es",
                              "http://www.elcorteingles.es/",
                              "http://www.rtve.es");
```

```
        var pos = Math.round(Math.random()*3);
        document.links[0].href = enlaces[pos];
      </script>
    </body>
</html>
```

e) Objeto History.

El objeto History almacena las referencias de todos los sitios web visitados, pudiendo desplazarnos adelanbte o atrás. No podemos acceder a los nombres de las URLs accedidas pues es información privada del usuario.

La principal propiedad del objeto history es:

Propiedad	Descripción
length	Corresponde al número de páginas que han sido visitadas.

Los métodos del objeto history son:

Método	Descripción
back()	Carga la URL del documento anterior del historial.
forward()	Carga la URL del documento siguiente del historial.
go()	Carga la URL del documento especificado por el índice que pasamos como parámetro dentro del historial.

Ejemplo:

```
<html>
  <head>
    <title> Ejemplo del Objeto history</title>
  </head>
  <body>
    <script type="text/javaScript">
      document.write(history.length);
    </script>
    <button onclick="history.go(2)">Saltar</button>
    <button onclick="history.back()">Anterior</button>
    <button onclick="history.forward()">Siguiente</button>
  </body>
</html>
```

e) Objeto location.

El objeto location corresponde con la URL de la página web en uso. Las propiedades permiten acceder a las diferentes partes de la URL y sus métodos permiten recargar la página, cargar una página nueva o reemplazar una por otra.

Principales propiedades del obejto location:

Propiedad	Descripción
hash	Corresponde a la cadena que representa el anclaje de la URL. Es decir, la parte de la URL que va después de la etiqueta #.
host	Corresponde a la cadena que representa el nombre del dominio del servidor y el número del puerto dentro de la URL.
hostname	Corresponde a la cadena que representa el nombre del dominio del servidor dentro de la URL.
href	Corresponde a la URL completa.
pathname	Corresponde a la cadena que sigue al nombre del servidor.
port	Corresponde al número de puerto de la URL.
protocol	Corresponde al protocolo utilizado por la página web.
search	Corresponde a la cadena de búsqueda que se encuentra después de un signo de interrogación de la URL.

Principales métodos del objeto location:

Método	Descripción
assign()	Carga un nuevo documento.
reload()	Carga de nuevo el documento actual.
replace()	Sustituye la URL del documento actual por otra URL.

Ejemplo:

```
<script type="text/javascript">
  document.write("Nombre del Dominio de la URL: "
    +location.hostname+"<br>");
  document.write("URL Completa: "+location.href+"<br>");
  document.write("Carpeta de la URL: "+location.pathname+"<br>");
  document.write("Protocolo de la URL: "+location.protocol+"<br>");
</script>
<button onclick="location.reload()">Recargar</button>
<button
onclick="location.replace('http://www.yahoo.es/')">Yahoo</button>
```

3.4.- Generación de Elementos HTML desde el código.

Mediante el método document.write() podemos escribir un resultado por pantalla. También podemos escribir texto en una nueva ventana emergente.

Ejemplo:

```
<button onclick="var nuevaventana = window.open();
      nuevaventana.document.write('Esta es una nueva ventana');">
Abrir Ventana
</button>
```

3.5.- Aplicaciones prácticas de los marcos.

Existe la posibilidad de dividir la ventana en dos o más partes independientes llamados marcos y desde javascript interactuar con ellos.

Los marcos se crean desde HTML con la etiqueta <frameset>, indicando el numero de marcos, la posición y el tamaño. Utilizaremos los atributos rows y cols para definir la posición de cada uno de los marcos. La etiqueta <frame> define las características de cada uno de ellos.

Los atributos de la etiqueta <frame> son:

Atributo	Descripción
frameborder	Define si mostrar o no el borde del marco.
marginheight	Permite cambiar los márgenes verticales del marco.
marginwidth	Permite cambiar los márgenes horizontales del marco.
name	Asigna un nombre al marco.
noresize	Evita que el usuario pueda modificar el tamaño del marco.
scrolling	Permite elegir si posiciona o no una barra de desplazamiento en el marco.
src	Indica la URL del documento HTML que contendrá el marco.

Ejemplo: Realizar dos divisiones verticales (cols) del 20% y del 80%. La primera división vertical queda dividida en dos divisiones horizontales (rows) del 30% y 70%. Ningún marco puede cambiar su tamaño (noresize).

marcos.html

```
<frameset cols="20%,80%">
      <frameset rows="30%,70%">
            <frame name="superior" src="superior.html" noresize>
            <frame name="inferior" src="inferior.html" noresize>
      </frameset>
      <frame name="principal" src="principal.html" noresize>
</frameset>
```

superior.html

```
<button onclick="parent.superior.document.bgColor='red'">
	Color Superior
</button><br>
<button onclick="parent.inferior.document.write('Marco Inferior')">
	Escribir Inferior
</button><br>
<button
onclick="parent.principal.location.replace('http://www.elcorteingles.es')">
	Cargar Web Principal
</button>
```

inferior.html

```
Sin código
```

principal.html

```
Sin código
```

Otra forma de acceder a los marcos de la página es mediante el objeto frames de javascript que almacena un array de los marcos de la página. La página **superior.html** quedaría del siguiente modo:

```
<button onclick="parent.frames['superior'].document.bgColor='red'">
	Color Superior
</button><br>
<button onclick="parent.frames['inferior'].document.write('Marco inferior')">
	Escribir Inferior
</button><br>
<button
onclick="parent.frames['principal'].location.replace('http://www.elcorteingle
s.es')">
	Cargar Web Principal
</button>
```

HTML en sus últimas versiones, no utiliza las frames y en su lugar ofrece la posibilidad de usar <iframe> para cargar contenido en un marco.

Ejemplo: Crear un <iframe> en la página principal de 640px x 360px y que muestre una web externa. Acceder al iframe y cargar otra página.

iframe.html

```
<iframe id="marco" width="640" height="360" frameborder="0"
src="http://www.elcorteingles.es"></iframe>
<p>Esta es la Pagina Principal</p>
<button onclick=" document.getElementById('marco').src='http://www.sony.es'">
Cargar Sony</button>
```

Ejemplo: Crear un <iframe> en la página principal de 640px x 360px y que permita cargar una página, quitar la página mostrada, cambiar color de fondo del iframe y mostrar un texto.

iframeContenido.html

```
<iframe id="marco" width="640" height="360" frameborder="0"></iframe>
<p>Esta es la Pagina Principal</p>
<button onclick="
document.getElementById('marco').src='http://www.sony.es'">
      Cargar Pagina
</button>
<button onclick=" document.getElementById('marco').src=''">
      Descargar Pagina
</button>
<button onclick="
document.getElementById('marco').contentDocument.body.style.background
Color='red'">
      Color Fondo
</button>
<button onclick="
document.getElementById('marco').contentDocument.body.innerHTML='Esto
es un texto dentro del iframe'">
      Escribir Texto
</button>
```

3.6.- Gestión de las ventanas.

Podemos crear una nueva ventana vacía con el método open(), el cual devuelve una referencia a dicha ventana.

```
nuevaVentana = window.open();
```

La variable nuevaVentana contiene la referencia a la ventana creada, pudiendo utilizar dicha referencia para manipular la ventana. El método open() cuenta con tres parámetros:

- URL que contendrá.
- Nombre de la ventana.
- Atributos que definen la ventana.

El método close() permite cerrar la ventana abierta con el método open().

```
nuevaVentana.window.close();
```

Ejemplo:

```
<button onclick="miVentana=window.open('http://www.elcorteingles.es',
      'Publicidad','top=100, left=100, width=400, height=400');">
Abrir Ventana
</button><br>
<button onclick="miVentana.window.close();">
      Cerrar Ventana
</button>
```

Ejemplo:

```
<button onclick="miVentana = window.open('','Publicidad',
            'top=100, left=100, width=400, height=400');">
      Abrir Ventana
</button><br>
<button onclick="miVentana.document.write('Este texto esta escrito en
la ventana emergente');">
      Escribir en  Ventana
</button><br>
<button onclick="miVentana.window.close();">
      Cerrar Ventana
</button>
```

La ventana principal puede cerrar ventanas secundarias pero no al revés. El atributo de la ventana window.opener() permite acceder a los métodos y propiedades de la ventana.

Ejemplo:

Fichero: v1.html

```
<script type="text/javascript">
      var miventana= window.open("v2.html","","width=200,height=200");
</script>
```

Fichero: v2.html

```
<html>
<head>
      <title>Ventana Secundaria</title>
</head>
<body>
      <input type="button"
            onClick="window.opener.document.bgColor='yellow'"
            value="Amarillo">
      <input type="button"
            onClick="window.opener.document.bgColor='red'"
            value="Rojo">
      <input type="button"
            onClick="window.opener.document.bgColor='white'"
             value="Blanco">
</body>
</html>
```