Diseño y Programación Web

Tema 5

JavaScript

Enrique Moguel
enrique@unex.es

Índice

JavaScript

- **JavaScript** es una lenguaje de programación interpretado.
  - Basado en ECMAScript,
  - Orientado a objetos,
  - Multiplataforma.

- Se utiliza principalmente como parte del cliente.
  - Aunque también se puede usar en el lado servidor.

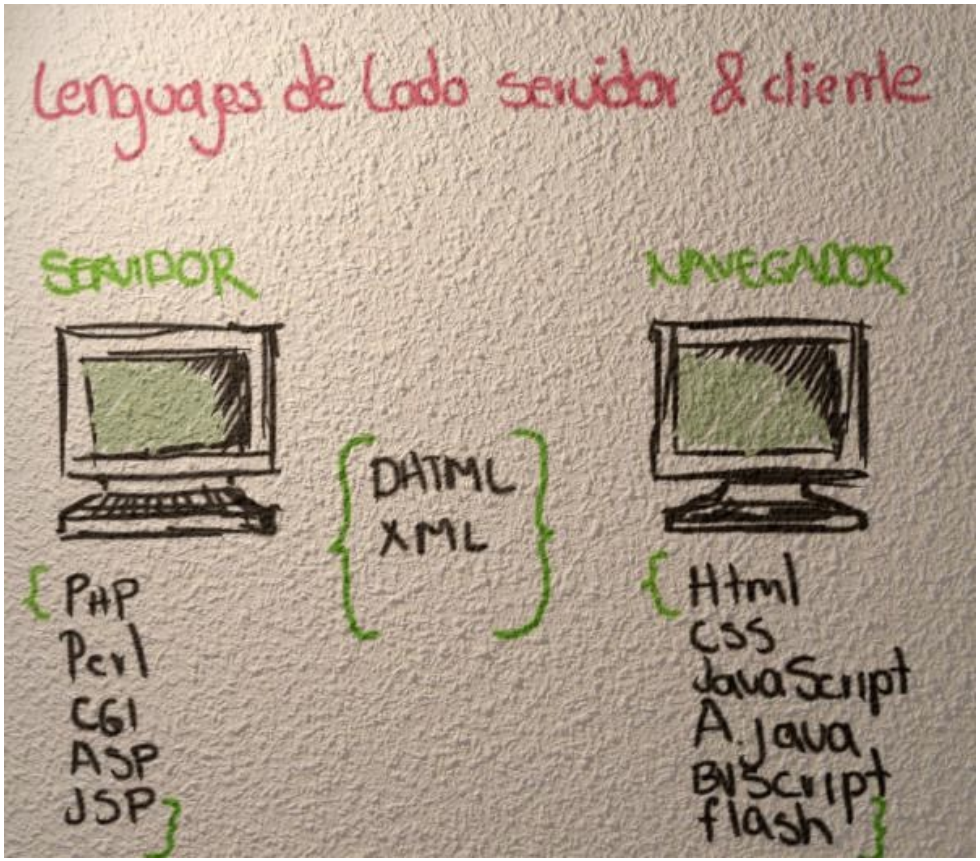
JavaScript

- Desde 2012 todos los navegadores modernos soportan al menos ECMAScript 5.1.

- Sintaxis similar a C o Java.
  - No tiene relación con Java.
  - Débilmente tipado.
  - Orientación a objetos basado en prototipos, no en clases.
  - Herencia dinámica.
  - Sensible a mayúsculas.

JavaScript

- **JavaScript** (abreviado comúnmente a "*JS*") es un lenguaje de programación **interpretado**. Se define como **orientado a objetos**, **basado en prototipos**, **imperativo**, **débilmente tipado** y **dinámico**.



JavaScript

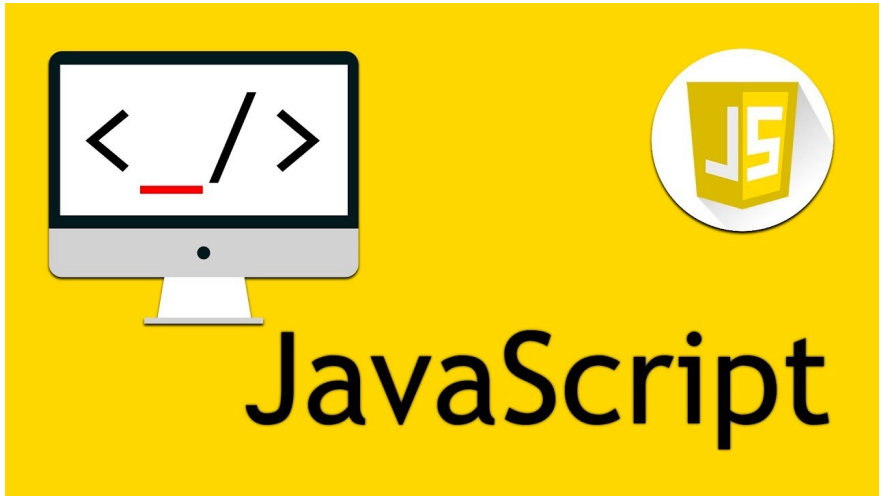
Aplicaciones Web:

Aunque los inicios de Internet se remontan a los años sesenta, no ha sido hasta los años noventa cuando, gracias a la Web, se ha extendido su uso por todo el mundo. **En pocos años la Web ha evolucionado enormemente**: se ha pasado de páginas sencillas, con pocas imágenes y contenidos estáticos a páginas complejas con contenidos dinámicos que provienen de bases de datos, lo que permite la creación de "*aplicaciones web*". De forma breve, una aplicación web se puede definir como una **aplicación en la cual un usuario por medio de un navegador realiza peticiones a una aplicación remota accesible a través de Internet** (o a través de una intranet) **y que recibe una respuesta que se muestra en el propio navegador**.

JavaScript

¿Para qué podemos usar JavaScript?

- para crear **sitios web** que respondan a la interacción del usuario;
- para crear **aplicaciones** y **juegos** (por ejemplo, el blackjack);
- para **acceder a información** en Internet (por ejemplo, encontrar las palabras más populares en Twitter según el tema);
- para **organizar y presentar datos** (por ejemplo, automatizar el trabajo de las hojas de cálculo o para visualización de datos).



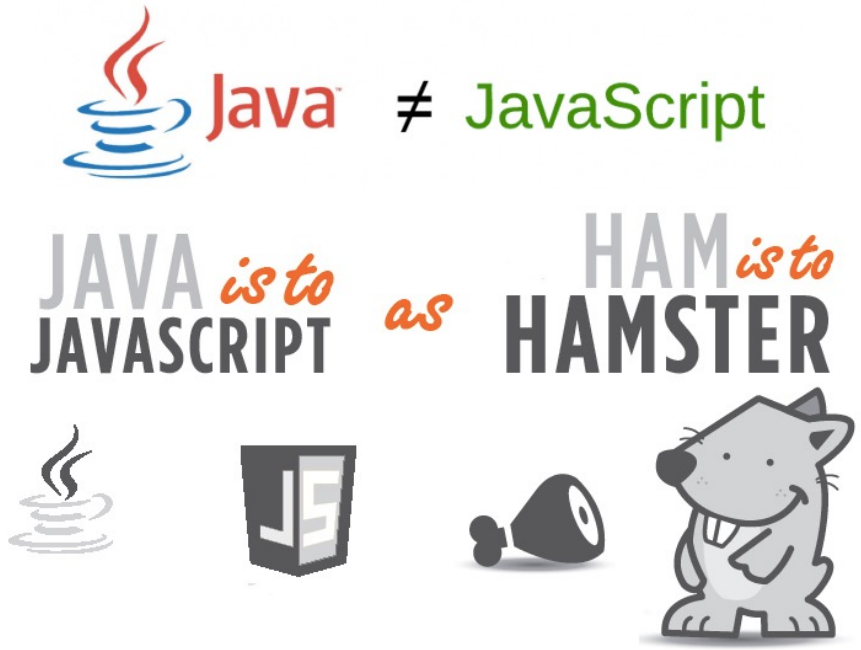
JavaScript

Por **seguridad**, los scripts sólo se pueden ejecutar dentro del navegador y con ciertas limitaciones:

- No pueden comunicarse con recursos que no pertenezcan al mismo dominio desde el que se descargó el script;
- No pueden cerrar ventanas que no hayan abierto esos mismos scripts;
- No pueden acceder al sistema de ficheros, ni para leer ni para escribir;
- No pueden acceder a las preferencias del navegador;
- Si la ejecución de un script dura demasiado tiempo el navegador informa al usuario de que el script está consumiendo demasiados recursos y le da la posibilidad de detener su ejecución;
- Esto podría ocurrir por un error de programación del script o alguno malicioso;
- Es posible firmar digitalmente los scripts para que el usuario permita realizar algunas de esas acciones.

Otros datos de interés

- JavaScript fue desarrollado originalmente por **Brendan Eich** de Netscape.
- JAVASCRIPT es una marca registrada de **Oracle Corporation**. Es usada con licencia por los productos creados por *Netscape Communications* y entidades actuales como la *Fundación Mozilla*.
- **JavaScript se ha convertido en uno de los lenguajes de programación más populares en internet**.



Insertar JS en HTML

```
<script>
        function (…) {
                // Inserta tu código aquí
        }
</script>
```

```
<script src=”myscript.js">
```

Primeros ejemplos

Ejemplo 01: Con JavaScript podremos realizar cambios en el HTML, en el CSS, atributos, etc.

Ejemplo 02: También podremos realizar validación de campos.

Entre infinidad de funcionalidad más que iremos aprendiendo…

Sintaxis

Principalmente, la sintaxis de JavaScript se compone en:

- **Tipos de datos**;
- **Operadores**;
- **Expresiones**;
- **Palabras reservadas**;
- Y **Comentarios**.

Tipos de datos

```
var length = 15;                                          // Number
var name = “Juan”;                                        // String
var cars = [“Seat”, “Volvo”, “BMW”];                      // Array
var person = {firstName:”Juan”, lastName:”Sanz”};         // Object
var person = new Object (“Juan”, “Sanz”, 59);             // Object constructor
Var person = new Object                                   // Object constructor
      person.firstName = ‘Juan’;
      person [lastName] = ‘Sanz’;

var x;                    // x es undefined
var x = 5;                // x cambia a Number
var x = “John”;           // x cambia a String


var x = 16 + 4 + “Volvo”;
¿SALIDA?

var x = “Volvo” + 16 + 4;
¿SALIDA?
```

Tipos de datos

```
var x1 = 34.00;      // Number con decimales
var x2 = 34;         // Number sin decimales

var x = true;
var y = false;

var cars = [“Seat”, “Volvo”, “BMW”];
var person = {firstName:”Juan”, lastName:”Sanz”, age:50, eyeColor:”azul”};


typeof “John”                    // Devuelve string
typeof 3.14                      // Devuelve number
typeof false                     // Devuelve boolean
typeof [1,2,3,4]                 // Devuelve object
typeof {name:’John’, age:34}     // Devuelve object
```

Operadores

Operador	Descripción	Valor de y	Operación	Valor de x
+	Suma	5	x = y + 2	7
-	Resta	5	x = y - 2	3
*	Multiplicación	5	x = y * 2	10
/	División	5	x = y / 2	2.5
%	Resto división	5	x = y % 2	1
--	Decrementar	Dado y=5		
		4	y--	
		4	x = --y	4
		4	x = y--	5
++	Incrementar	Dado y=5		
		6	y++	
		6	x = ++y	6
		6	x = y++	5

Expresiones

Operator	Description	Comparing	Returns
==	equal to	x == 8	false
		x == 5	true
===	equal value and equal type	x === "5"	false
		x === 5	true
!=	not equal	x != 8	true
!==	not equal value or not equal type	x !== "5"	true
		x !== 5	false
>	greater than	x > 8	false
<	less than	x < 8	true
>=	greater than or equal to	x >= 8	false
<=	less than or equal to	x <= 8	*true*

Operator	Description	Example
&&	and	(x < 10 && y > 1) is true
||	or	(x == 5 || y == 5) is false
!	not	!(x == y) is true

Expresiones regulares

Cualquier letra en minuscula	[a-z]
Entero	^(?:\+|-)?\d+$
Correo electrónico	/[\w-\.]{3,}@([\w-]{2,}\.)*([\w-]{2,}\.)[\w-]{2,4}/
URL	^(ht|f)tp(s?)\:\/\/[0-9a-zA-Z]([-.\w]*[0-9a-zA-Z])*(:(0-9)*)*(\/?)([a-zA-Z0-9\-\.\?\,\'\/\\\+&%\$#_]*)?$
Contraseña segura	(?!^[0-9]*$)(?!^[a-zA-Z]*$)^([a-zA-Z0-9]{8,10})$ (Entre 8 y 10 caracteres, por lo menos un digito y un alfanumérico, y no puede contener caracteres espaciales)
Fecha	^\d{1,2}\/\d{1,2}\/\d{2,4}$ (Por ejemplo 01/01/2007)
Hora	^(0[1-9]|1\d|2[0-3]):([0-5]\d):([0-5]\d)$ (Por ejemplo 10:45:23)
Número tarjeta de crédito	^((67\d{2})|(4\d{3})|(5[1-5]\d{2})|(6011))(-?\s?\d{4}){3}|(3[4,7])\ d{2}-?\s?\d{6}-?\s?\d{5}$
Número teléfono	^[0-9]{2,3}-? ?[0-9]{6,7}$
Código postal	^([1-9]{2}|[0-9][1-9]|[1-9][0-9])[0-9]{3}$
Certificado Identificación Fiscal	^(X(-|\.)?0?\d{7}(-|\.)?[A-Z]|[A-Z](-|\.)?\d{7}(-|\.)? [0-9A-Z]|\d{8}(-|\.)?[A-Z])$

Palabras reservadas

break,
case,
catch,
continue,
default,
delete,
do,
else,
finally,
for,
function,
if,
in,
instanceof,
new,
return,
switch,
this,
throw,
try,
typeof,
var,
void,
while,
with

etc.

Comentarios

```
// Comentario de línea simple


/*
Comentario multilínea
...
...
*/
```

Objetos

Object	Properties	Methods
	car.name = Fiat car.model = 500 car.weight = 850kg car.color = white	car.start() car.drive() car.brake() car.stop()

```
var car = {type:"Fiat", model:500, weight:850, color:"white”
          function start () { … },
          function drive () { … },
          …
          };
```

Funciones

```
function nombre (parámetro1, parámetro2, parámetro3) {
        código a ejecutar
};
```

```
// Ejemplo de función
function suma_y_muestra (número1, número2) {
        resultado = número1 + número2;
        alert ("El resultado es " + resultado);
};
```

Eventos

Evento	Descripción	Elementos para los que está definido
onblur	Deseleccionar el elemento	<button>, <input>, <label>, <select>, <textarea>, <body>
onchange	Deseleccionar un elemento que se ha modificado	<input>, <select>, <textarea>
onclick	Pinchar y soltar el ratón	Todos los elementos
ondblclick	Pinchar dos veces seguidas con el ratón	Todos los elementos
onfocus	Seleccionar un elemento	<button>, <input>, <label>, <select>, <textarea>, <body>
onkeydown	Pulsar una tecla (sin soltar)	Elementos de formulario y <body>
onkeypress	Pulsar una tecla	Elementos de formulario y <body>
onkeyup	Soltar una tecla pulsada	Elementos de formulario y <body>
onload	La página se ha cargado completamente	<body>
onmousedown	Pulsar (sin soltar) un botón del ratón	Todos los elementos
onmousemove	Mover el ratón	Todos los elementos
onmouseout	El ratón *"sale"* del elemento (pasa por encima de otro elemento)	Todos los elementos
onmouseover	El ratón *"entra"* en el elemento (pasa por encima del elemento)	Todos los elementos
onmouseup	Soltar el botón que estaba pulsado en el ratón	Todos los elementos
onreset	Inicializar el formulario (borrar todos sus datos)	<form>
onresize	Se ha modificado el tamaño de la ventana del navegador	<body>
onselect	Seleccionar un texto	<input>, <textarea>
onsubmit	Enviar el formulario	<form>
onunload	Se abandona la página (por ejemplo al cerrar el navegador)	<body>

Métodos

- **Boolean;**
- **String;**
- **Number;**
- **Math;**
- **Date;**
- **Array.**

Métodos: **Boolean**

```
Boolean (10 > 9)    // returns true

var x = 0;
Boolean (x);        // returns false

var x = 1;
Boolean (x);        // returns true
```

Métodos: **String**

```
var text = “Él se llama ’Juan’ “;
var text =  ‘Él se llama ”Juan” ‘;

// Longitud de cadena de caracteres
var txt = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
var sln = txt.length;

// Concatenación de textos
var text1 = "Hello";
var text2 = "World";
text3 = text1.concat("  ",text2);

var str = "HELLO WORLD";
str[0];                          // devuelve H

var txt = "a,b,c,d,e";       // String
txt.split(",");                  // Split con comas
```

Métodos: **Number**

```
var x = 123e5;       // 12300000
var y = 123e-5;      // 0.00123

var x = 0xFF;        // x tiene valor igual a 255

var myNumber = 2;
while (myNumber != Infinity) {        // Ejecuta hasta el infinito
   myNumber = myNumber * myNumber;
}

var x = 9.656;
x.toFixed (0);              // returns 10
x.toFixed (2);              // returns 9.66
x.toFixed (4);              // returns 9.6560

var x = 9.656;
x.toPrecision ();           // returns 9.656
x.toPrecision (2);          // returns 9.7
x.toPrecision (4);          // returns 9.656
```

Métodos: **Math**

```
Math.random ();          // Devuelve un número aleatorio

Math.min (0, 150, 30, 20, -8);      // returns -8
Math.max (0, 150, 30, 20, -8);      // returns 150

Math.round (4.7);        // Redondea a returns 5
Math.round (4.4);        // Redondea a returns 4

Math.floor (4.7);          // returns 4

// Devuelve un número aleatorio entre 0 y 10
Math.floor (Math.random() * 11);
```

Métodos: **Date**

```
// Inicializamos
new Date() ;
new Date(year, month, day, hours, minutes, seconds, milliseconds);

// ISO Dates
var d = new Date("2015-03-25");
var d = new Date("2015-03-25T12:00:00");
var d = new Date("2015-03-25 12:00:00");

// Long Dates
var d = new Date("25 Mar 2015");
var d = new Date("2015, JANUARY, 25.");

// Short Dates
var d = new Date("03-25-2015");
var d = new Date("2015/03/25");

// Full format
var d = new Date("Wed Mar 25 2015 09:56:24 GMT+0100 (W. Europe Std. Time)");
```

Métodos: **Date**

```
// En qué año estamos.
var d = new Date();
document.getElementById("demo").innerHTML = d.getFullYear();
```

```
// En qué día estamos.
var d = new Date();
var days = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",
"Friday”, "Saturday"];
document.getElementById("demo").innerHTML = days[d.getDay()];
```

```
// Tranformar fecha a Milisegundos
var msec = Date.parse("March 21, 2012");
document.getElementById("demo").innerHTML = msec;
```

Métodos: **Array**

```
var points = new Array (40, 100, 1, 5, 25, 10);          // Bad
var points = [40, 100, 1, 5, 25, 10];                    // Good

var cars = [”Seat", "Volvo", "BMW"];
document.getElementById("demo").innerHTML = cars[0];

var fruits = [”Plátano", ”Naranja", ”Manzana", "Mango"];
fruits.length;                    // La longitud del array es 4

var fruits = ["Plátano", "Naranja", "Manzana", "Mango"];
fruits.pop();                 // Elimina el último elemento del array
fruits.push("Kiwi");     //  Añade nuevo elemento al array

var fruits = ["Plátano", "Naranja", "Manzana", "Mango"];
fruits[0] = "Kiwi";       // Cambia el primer elemento del array

var fruits = ["Plátano", "Naranja", "Manzana", "Mango"];
delete fruits[0];         // Cambia el primer elemento a undefined
```

Sentencias

```
if (hour < 12) {
    greeting = ”Buenos días.";
} else {
    greeting = ”Buenas tardes.";
}
```

```
for (i = 0; i < 5; i++) {
    text += ”El número es " + i + "<br>";
}
```

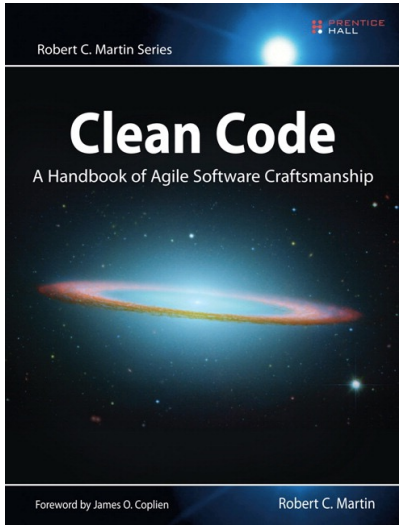
```
while (i < 10) {
    text += ”El número es " + i;
    i++;
}
```

```
switch (new Date().getDay()) {
    case 0:
        day = ”Lunes";
        break;
    case 1:
        day = ”Martes";
        break;
    case 2:
        day = ”Miércoles";
        break;
    case 3:
        day = ”Jueves";
        break;
    …
}
```

```
window.alert(5 + 6);
console.log(5 + 6);
```

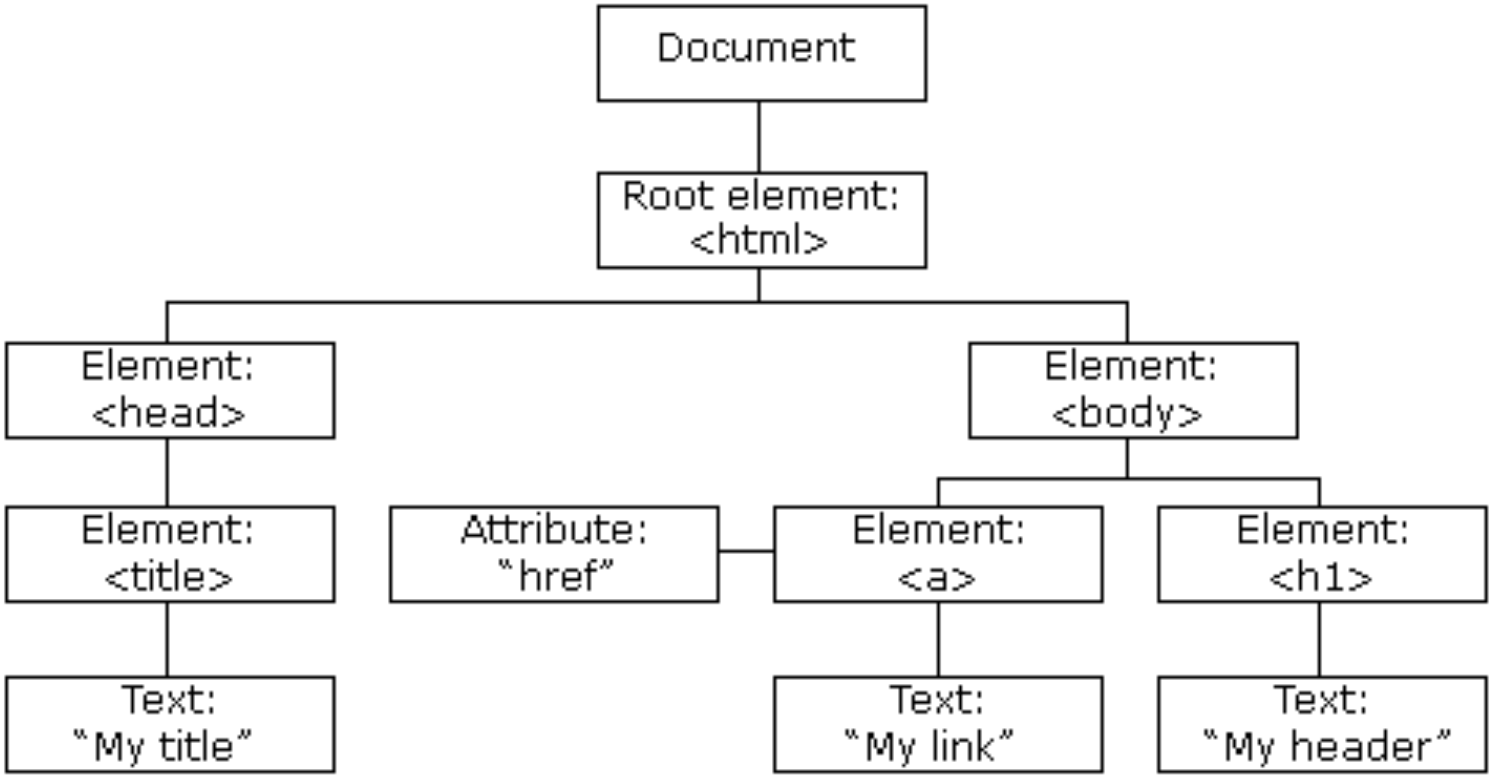
Buenas prácticas

- Minimiza el uso de variables globales;
- Todas las variables utilizadas en una función deben ser declaradas como variables locales.
- Es una buena práctica de codificación el inicializar las variables cuando se declaran.
- Trata los tipos de datos Numbers, Strings y Booleans como datos primitivos. NO como objetos.
- var x = ”Juan";
- var y = new String (”Juan");
- (x === y) // es FALSE porque **x** es un STRING e **y** es un OBJECT
- etc.



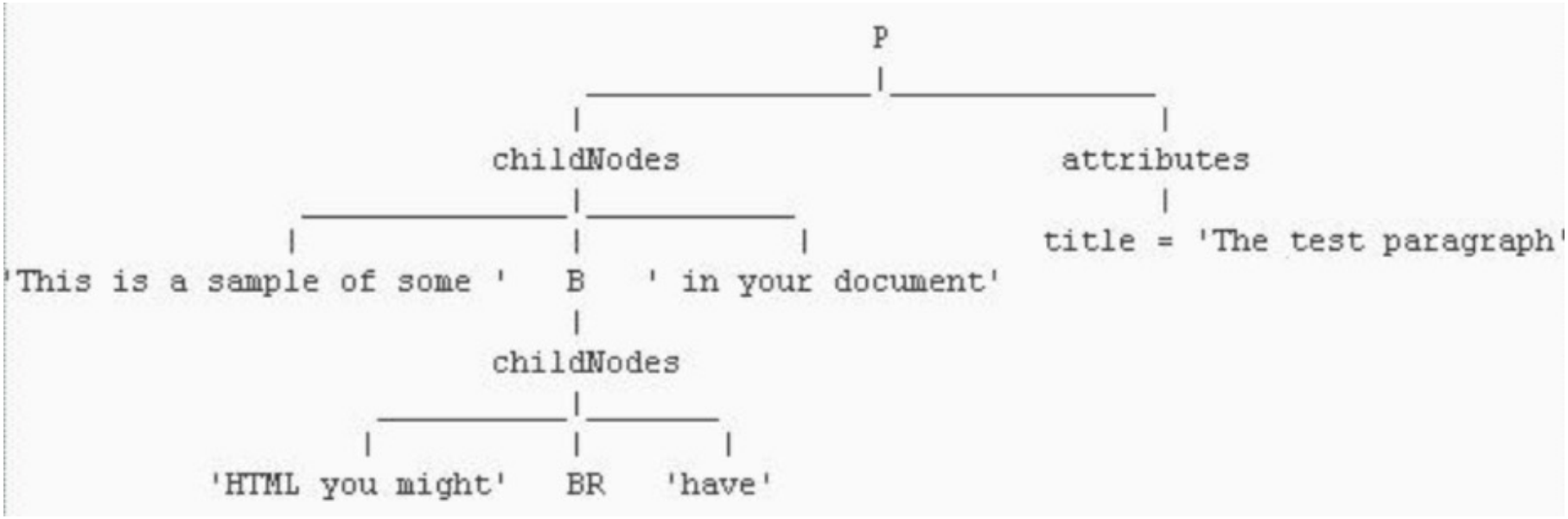
DOM (Document Object Model)

Cuando se carga una página web, el navegador crea un **D**ocument **O**bject **M**odel de la página.



DOM (Document Object Model)

This is a sample of some HTML you might **
** have in your document



DOM (Document Object Model)

El DOM HTML se puede acceder con JavaScript (y con otros lenguajes de programación).

En el DOM, todos los elementos HTML se definen como objetos:

- **objects;**
- **properties;**
- **methods;**
- **events.**

DOM (Document Object Model)

Method	Description
document.getElementById()	Find an element by element id
document.getElementsByTagName()	Find elements by tag name
document.getElementsByClassName()	Find elements by class name

---

Method	Description
element.innerHTML=	Change the inner HTML of an element
element.attribute=	Change the attribute of an HTML element
element.setAttribute(*attribute,value*)	Change the attribute of an HTML element
element.style.*property*=	Change the style of an HTML element

https://www.w3schools.com/jsref/prop_html_innerhtml.asp

DOM (Document Object Model)

- El objeto **document**
  - Es la raíz del documento HTML.
  - Permite escribir texto en la salida:

    document.write (“Hello World!”);

  - Permite escribir texto con formato en la salida:

    document.write (“<h1>Hello World!</h1>”);

  - Permite obtener el código HTML que contiene otro elemento:

```
<a name="first">First anchor</a><br/>
<a name="second">Second anchor</a><br/>
<a name="third">Third anchor</a><br />
<script type="text/javascript">
    document.write(document.anchors[0].innerHTML)
</script>
```

DOM (Document Object Model)

- Permite acceder a un elemento de una colección:

```
<form id="Form1" name="Form1">
    Your name: <input type="text">
</form>
<form id="Form2" name="Form2">
    Your car: <input type="text">
</form>

<script type="text/javascript">
      document.write("<p>The first form's name is: " +
      document.forms[0].name + "</p>")
      document.write("<p>The first form's name is: " +
      document.getElementById("Form1").name + "</p>”);
</script>
```

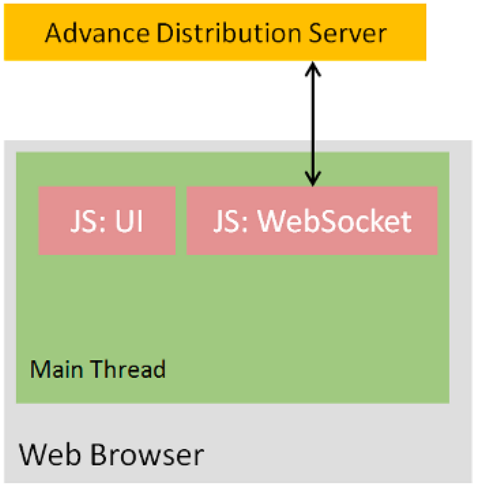
Más sobre el DOM: https://www.w3schools.com/jsref/obj_navigator.asp

WebWorkers

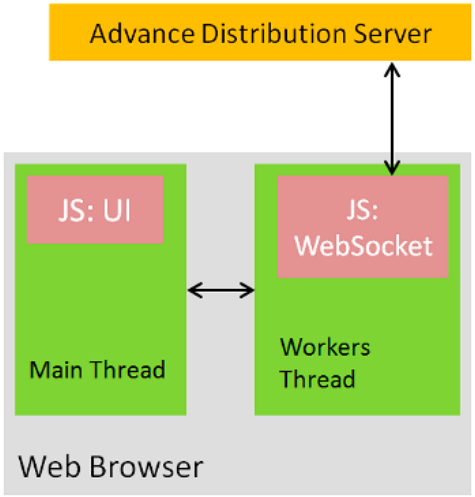
Por lo general, se usan para ejecutar scripts e segundo plano. Son procesos que se ejecutan en segundo plano sin afectar al rendimiento del sitio Web.

Ejemplo 5 del tema HTML5.

Post interesante.



Default Brower Thread diagram



Workers Thread diagram

Herramientas

HTML KickStart. Código de referencia:
http://www.99lime.com/elements/

99Lime UIKIT. Librería para elementos UI:
http://www.99lime.com/uikit/

CreateJS. Algunas librerías y herramientas de interés:
https://www.createjs.com/

Apache Cordova. Entorno de desarrollo para aplicaciones móviles:
https://cordova.apache.org/

Librerías

- **jQuery**: es la librería más utilizada por su fácil manejo en el tratamiento de los objetos del DOM ya que nos permite abstraernos completamente. Muy aconsejable para la integración de efectos y animaciones personalizadas.

- **Mootools**: este framework está enfocado a la orientación de objetos como sus siglas indican (My object oriented tools). Cuenta con un componente avanzado de efectos y con transiciones optimizadas y utilizadas por multitud de desarrolladores Flash.

- **Prototype**: es un framework escrito en JavaScript para el desarrollo sencillo y dinámico de aplicaciones web. Su potencial es aprovechado al máximo cuando se desarrolla con *Ruby On Rails*.

- **Yahoo! UI Library**: bibliotecas utilizadas para la construcción de aplicaciones enriquecidas **RIA** (*rich Internet applications*) o aplicaciones web que tienen la mayoría de las características de las aplicaciones de escritorio tradicionales.

Librerías

- **Modernizr**: esta librería proporciona la misma experiencia de usurio a todos los visitantes del sitio web independientemente de su navegador. Modernizr no añade ninguna funcionalidad al navegador. En cambio, sólo averigua si la funcionalidad que estás intentando implementar responde en dicho navegador. Esto nos permite experimentar con las nuevas características de HTML5 y CSS3 sin preocuparnos por restar experiencia de usuario o que la página no se renderice adecuadamente.

- **Dojo**: es un framework que contiene APIs y widgets para facilitar el desarrollo de aplicaciones Web que utilicen tecnología AJAX. Es de código abierto y se puede descargar de forma gratuita en su página oficial. Cuenta con el patrocinio de *IBM*, *Google*, *AOL* y *Nexaweb*.

Librerías

- Validación de formularios: **wForms**, **Validanguage**, **LiveValidation**, **qForms.**

- Visualización y efectos de imagenes: **JS charts** (gráficos de barras, circulares y de líneas simples), **Gráfico** (10 tipos de gráficos, de barras, de area, de línea y barras horizontales...), **Canvas 3D JS Library** (objetos 3D), **CanvasXpress** (gráficos tridimensionales), **Raphaël** (gráficos vectoriales con SVG y VML), **ImageFX** (efectos sobre imágenes con CANVAS), **Reflection.js** (efectos de reflexión), **PaintbrushJS** (efectos sobre imágenes similares a los que podemos encontrar en aplicaciones como Instagram o picplz).

- Manejo de cadenas y funciones matemáticas: **Date.js** (funciones complejas de fechas), **Sylvester** (vectores y arrays en varias dimensiones), **XRegEx** (expresiones regulares), **JavaScript Url Library** (manipulación de direcciones o url).

- Uso de cualquier fuente en un sitio web: **typeface.js**, **Cufón**.

Librerías

- Chart.js:
  https://www.chartjs.org/

  https://www.chartjs.org/docs/latest/samples/other-charts/radar-skip-points.html

- D3.js:
  https://d3js.org/

  https://observablehq.com/@d3/bar-chart-race