

Modelando Comportamiento Javascript

JS

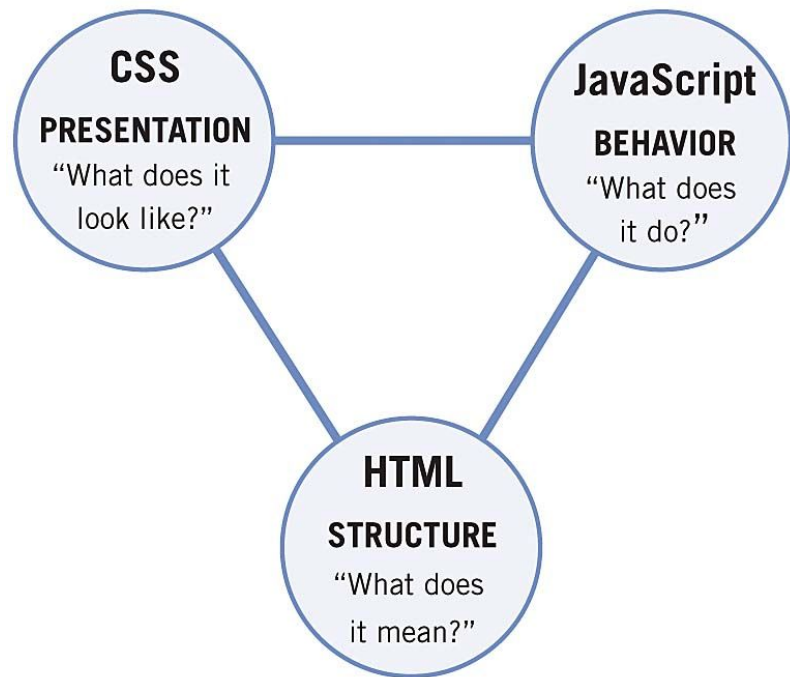
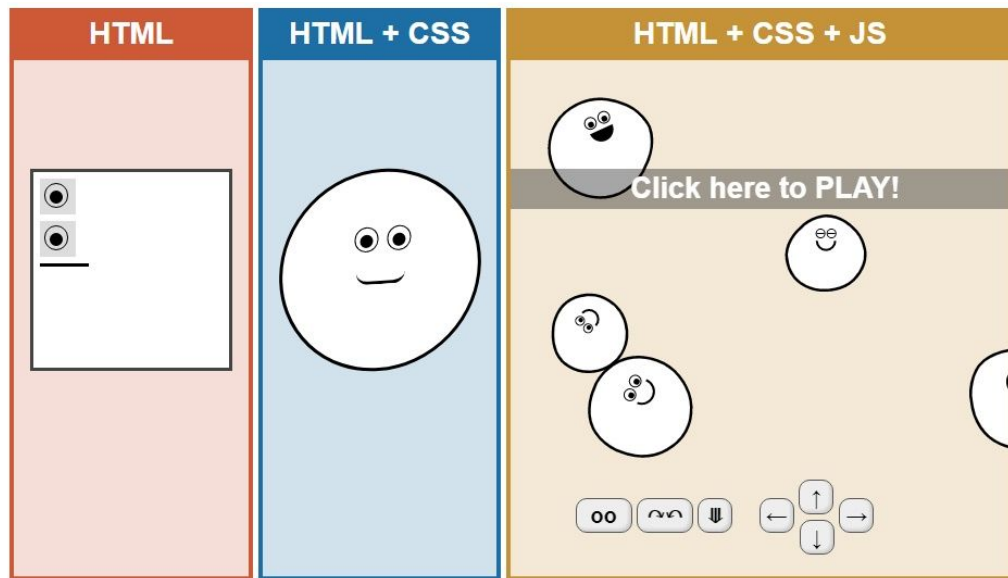


upna

Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

Iñigo Ezcurdia

Lenguajes base de la web - HTML & CSS & JavaScript



Javascript - Introducción

- Desarrollado en los 90 por Brendan Eich, cofundador de Mozilla.
- NO tiene nada que ver con el lenguaje Java.

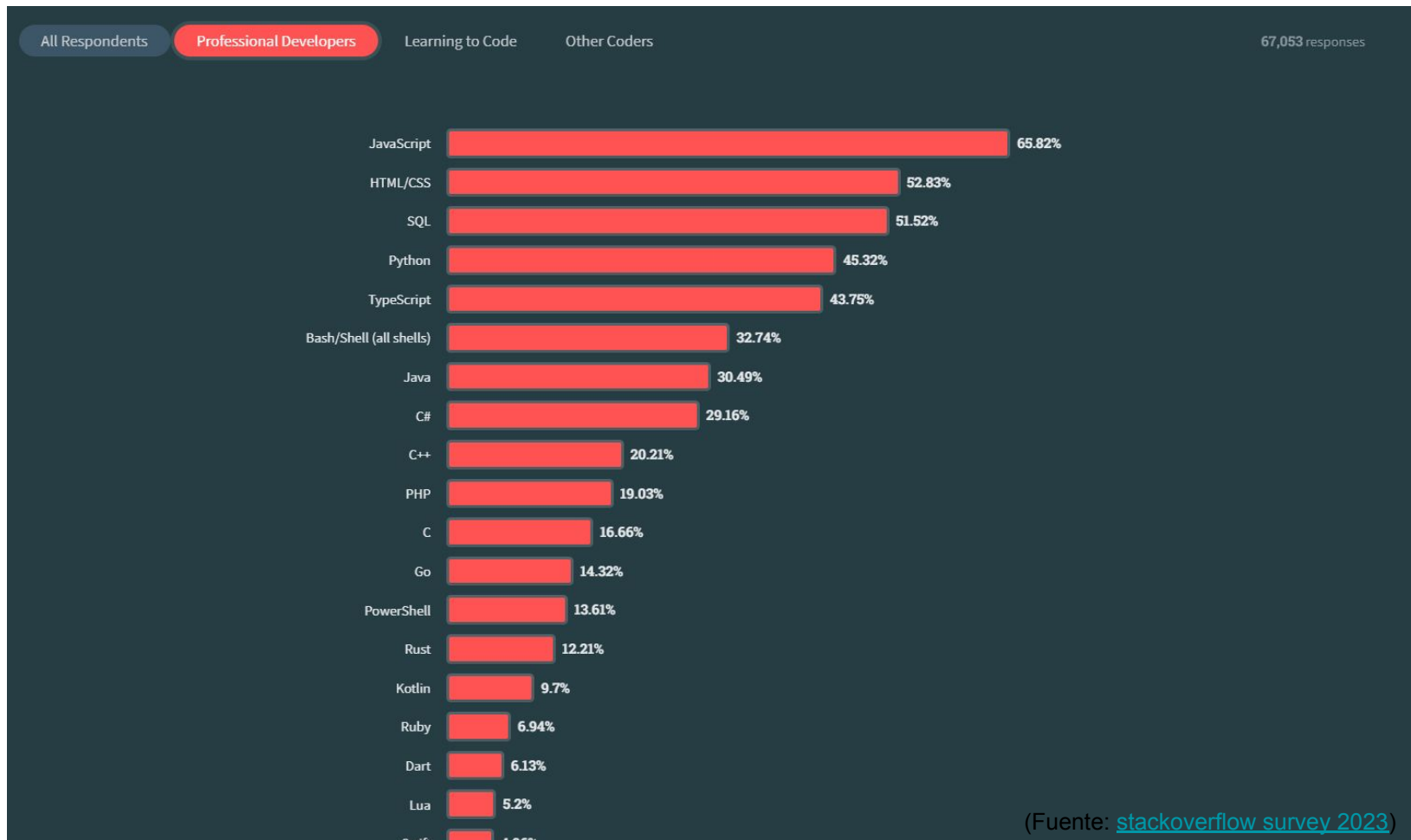
JAVA es a **JAVASCRIPT** lo que **PALO** es a **PALOMA**



- Al nivel al que vamos a trabajar, para nosotros es lo mismo:
 - JavaScript
 - Mocha
 - LiveScript
 - ECMAScript
- Se concibe inicialmente para ser ejecutado del lado del cliente.
- Actualmente existen herramientas del lado del servidor:
 - MongoDB
 - NodeJS & NPM
 - ...

Javascript - Popularidad

Líder por 11º año consecutivo

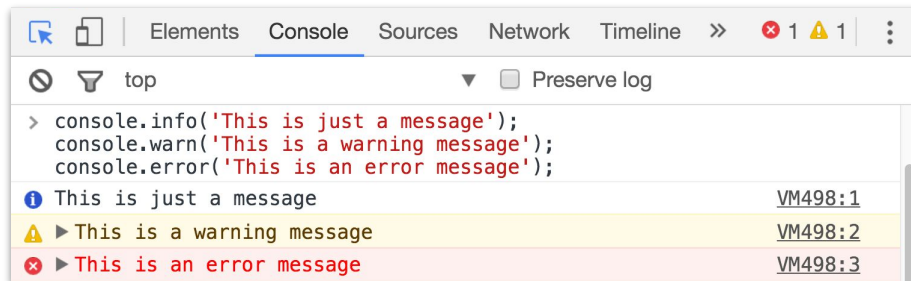


(Fuente: [stackoverflow survey 2023](#))

Javascript - Antes de empezar

- **Débilmente tipado:** No indicamos el tipo de variable al declararla
- **Interpretado:** No hace falta generar ficheros compilados. El navegador interpreta el código.
- Tiene acceso a todos los elementos del documento HTML

- **errores** y **warnings** por **consola** (F12->Consola)



- Podemos utilizar cualquier editor de texto para programar en Javascript (Editor de [snippets](#) en google chrome dev tools)
- Se ejecuta en el mismo momento en el que el navegador llega a interpretarlo.
- <https://jsfiddle.net/> Es un buen lugar para trastear puntualmente con ideas Javascript.

Javascript - ¿Qué aspecto tiene?

- Sin haberlo utilizado nunca... Parece que entendemos buena parte, ¿No?

```
542 <script>
543 function authenticateUser(username, password) {
544     var accounts = apiService.sql(
545         "SELECT * FROM users"
546     );
547
548     for (var i = 0; i < accounts.length; i++) {
549         var account = accounts[i];
550         if (account.username === username &&
551             account.password === password)
552         {
553             return true;
554         }
555     }
556     if ("true" === "true") {
557         return false;
558     }
559 }
560
561 $('#login').click(function() {
562     var username = $('#username').val();
563     var password = $('#password').val();
564
565     var authenticated = authenticateUser(username, password);
566
567     if (authenticated === true) {
568         $.cookie('loggedin', 'yes', { expires: 1 });
569     } else if (authenticated === false) {
570         $("#error_message").show();
571     }
572 });
573 </script>
574
```

Javascript - Inclusión en HTML

- Al igual que en CSS, tenemos tres maneras distintas de incluir Javascript en nuestros documentos HTML.

1. Entre las etiquetas `<script></script>`
2. Incluyendo ficheros externos .js (recomendado)
3. En elementos inline

```
<!DOCTYPE html>
<html>
<head>
  <title>Enlazar JS</title>
  <script type="text/javascript">
    document.write("Hola mundo");
  </script>
  <script type="text/javascript" src="scripts/miFichero.js"></script>
  <noscript>
    Tu navegador no soporta, o tiene desactivado, Javascript.
  </noscript>
</head>
<body>
  <p onclick="alert('Hola mundo')">Párrafo con sorpresa</p>
</body>
</html>
```

Además es importante decidir dónde ubicarlo, ya que se ejecuta en el momento en el que es parseado por el navegador.



Si nuestro código javascript utiliza un elemento HTML posterior, que todavía no se ha parseado = problemas

Javascript - Sintaxis - Comentarios y Variables

- Toda sentencia termina en ;
- Espacios en blanco se ignoran
- Comentarios
 - **//** para una línea
 - **/*** para multilínea ***/**
- Declaración de **variables**
 - Explícita/implícita
 - Mediante “var” o “let”

Tengo alcance únicamente local!
 - El primer carácter no numérico
 - NO indicamos su tipo
 - Se decide en tiempo de ejecución
 - Puede variar a lo largo de la ejecución
- **Constantes** mediante “const”

```
//Comentario de una línea

/* Comentario que
puede extenderse
a lo largo de varias líneas */

var variableExplícita = 8;

variableImplícita = "no necesito el var para declararme";
variableImplícita = 99; //Puedo cambiar de tipo!

const PI = 3.14;|
```


Javascript - Sintaxis - Tipos de las Variables

- Tienen tipos, aunque no los declaremos y puedan variar...

```
// Numeros
var tipoNumerico_entero = 55;
var tipoNumerico_decimal = 0.05;

// Booleanos
var boolean = true || false;

// Cadenas
var cadena = "Soy una \"cadena\" de texto\n";
var cadena = 'Tambien funciona con "comillas" simples';

// Objetos
var miCoche = new Object(); // Se crean con "new"
miCoche.color = "Rojo";
miCoche.year = 2005;
miCoche.mola = true;
```

```
// Array: Una estructura de multiples valores
var miArray = [55, 'Mezclo tipos!', false, false, null]
console.log(miArray)
```

```
► (5) [55, "Mezclo tipos!", false, false, null]
```

```
1  /* Undefined: Variables que han sido declaradas
2     pero no inicializadas*/
3  var variable;
4  console.log(variable);
5  console.log(typeof(variable));
6
7
8  // NULL: Es de tipo objeto. Es un valor, valor nulo
9  var variable2 = null;
10 console.log(variable2);
11 console.log(typeof(variable2));
```

```
undefined
```

```
undefined
```

```
null
```

```
object
```

typeof(variable) para consultar el tipo de una variable

```
> var a = "soy string";
< undefined
> typeof(a)
< "string"
```

Javascript - Sintaxis - Operadores

- Los de siempre, como (casi)siempre...

Category	Operator	Name/Description	Example	Result
Arithmetic	+	Addition	3+2	5
	-	Subtraction	3-2	1
	*	Multiplication	3*2	6
	/	Division	10/5	2
	%	Modulus	10%5	0
	++	Increment and then return value	X=3; ++X	4
		Return value and then increment	X=3; X++	3
	--	Decrement and then return value	X=3; --X	2
		Return value and then decrement	X=3; X--	3
Logical	&&	Logical “and” evaluates to true when both operands are true	3>2 && 5>3	False
		Logical “or” evaluates to true when either operand is true	3>1 2>5	True
	!	Logical “not” evaluates to true if the operand is false	3!=2	True
Comparison	==	Equal	5==9	False
	!=	Not equal	6!=4	True
	<	Less than	3<2	False
	<=	Less than or equal	5<=2	False
	>	Greater than	4>3	True
	>=	Greater than or equal	4>=4	True
String	+	Concatenation(join two strings together)	“A”+“BC”	ABC

⚠ Dividir entre 0 da como resultado “infinity”, no genera una excepción.

Javascript - Sintaxis - Estructuras de Control

- Las de siempre...

([foreach](#) disponible para elemento iterables)

IF-ELSE

```
if( myVariable == 2 ) {  
  myVariable = 1;  
} else if( myVariable == 5 ){  
  myVariable = 3;  
} else {  
  myVariable = 4;  
}
```

FOR

```
for( starting_initialise; continue_as_long_as_condition; do_this_each_time )  
  
for( var myVariable = 1; myVariable <= 5; myVariable++ ){  
  myArray[myVariable] = 1;  
}
```

FOR-IN

```
for( var myVariable in anObjectOrArray ) {  
  
for( var myVariable in document ) {  
  console.log(myVariable);  
}
```

WHILE

```
var myVariable = 1;  
while( myVariable <= 5 ) {  
  myArray[myVariable] = 1;  
  myVariable++;  
}
```

DO-WHILE

```
var myVariable = 1;  
do {  
  myArray[myVariable] = 1;  
  myVariable++;  
} while( myVariable <= 5 );
```

SWITCH

```
switch(myVar) {  
  case 1:  
    //if myVar is 1 this is executed  
  case 'sample':  
    //if myVar is 'sample' or 1 this is executed  
  case false:  
    //if myVar is false or sample or 1 this is  
    executed  
  default:  
    //if myVar does not satisfy any case  
    //this is executed  
}
```

Javascript - Sintaxis - Funciones

- Las declaramos mediante function.
- Pueden tener distintos argumentos de entrada
- Pueden devolver o no salidas.
- Pueden ser recursivas
- Pueden anidarse

```
var square = function(number) {  
    return number * number;  
};  
var x = square(4);
```

```
var num3 = 12;  
function multiply3(num1,num2) {  
    return num1 * num2 * num3;  
}
```

```
var factorial = function fac(n) {  
    return n < 2 ? 1 : n * fac(n - 1);  
};  
  
console.log(factorial(3));
```

```
function getScore() {  
    var num1 = 2,  
        num2 = 3;  
  
    function add() {  
        return name + ' scored ' + (num1 + num2);  
    }  
  
    return add();  
}
```



Ejercicio de ejemplo

- Pedir un número al usuario/a e imprimir por consola si este es par o impar.
 - `prompt()`

An embedded page at local-ntp says

Dame un número y te digo si es par o impar!

Cancel OK

JavaScript - Orientado a Objetos

Clase: Persona
Atributo: primerNombre
Método: diHola

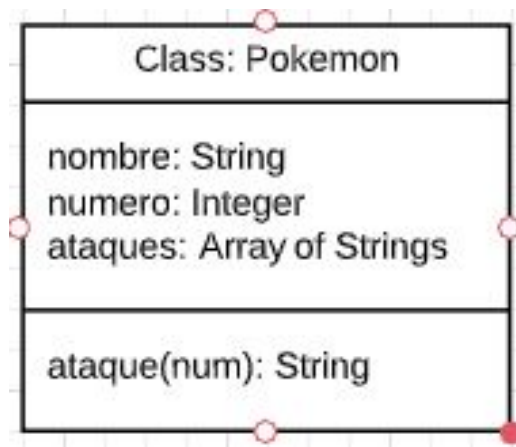


```
function Persona(primerNombre) {  
  this.primerNombre = primerNombre;  
}  
  
Persona.prototype.diHola = function() {  
  alert ('Hola, Soy ' + this.primerNombre);  
};  
  
// Creación de dos objetos de tipo Persona  
var persona1 = new Persona("Alicia");  
var persona2 = new Persona("Sebastian");  
  
// Llamadas al método diHola de la clase Persona.  
persona1.diHola(); // muestra "Hola, Soy Alicia"  
persona2.diHola(); // muestra "Hola, Soy Sebastian"
```

Alternativamente también podemos hacer uso de [clases y herencia](#) que quizás nos suene de otros lenguajes...

Un pequeño reto javascript

1. Crea una clase llamada "Pokemon", que tenga como atributos "nombre", "número", y "ataques".
2. Añádele un método llamado "ataque", que tenga como argumento de entrada un nº entero y que devuelva una frase explicando el ataque realizado.
3. Crea un objeto de tipo "Pokemon" y realiza un ataque (Escoge el pokemon que quieras)



```
var pikachu = new Pokemon ("Pikachu", 25, ["placaje", "rayo", "gruñido"]);
pikachu.ataque(2);
```

```
< "Pikachu utiliza el ataque gruñido"
```



Un pequeño reto javascript - Posible solución

?



Javascript - Manipulación del DOM

- Javascript puede:
 - Modificar todos los elementos HTML de una página
 - Modificar todos los atributos HTML de una página
 - Cambiar estilos CSS de una página
 - Eliminar elementos y atributos HTML de una página
 - Añadir nuevos elementos y atributos HTML de una página
 - Reaccionar a eventos HTML
 - Disparar eventos HTML
- “*document*” es la raíz el documento HTML.

Javascript - Manipulación del DOM

Encontrar elementos

```
// Encontrar un elemento por su ID
var myElement = document.getElementById("intro");

// Encontrar todos los elementos de una etiqueta concreta
var x = myElement.getElementsByTagName("p");

// Encontrar todos los elementos con una clase concreta
var x = document.getElementsByClassName("intro");

// Encontrar todos los elementos mediante un selector CSS
var x = document.querySelectorAll("p.intro");
```

Modificar elementos

```
// Cambiar el contenido HTML de un elemento
element.innerHTML = new html content
document.getElementById("demo").innerHTML = "Paragraph changed!";

// Consultar/Cambiar atributos de un elemento
element.attribute = new value
element.setAttribute(attribute, value)
document.getElementsByTagName("H1")[0].setAttribute("class", "democlass");

// Cambiar el estilo CSS de un elemento
element.style.property = new style
document.getElementById("myH1").style.color = "red";
```

Añadir/eliminar elementos

```
// Crear un elemento HTML
document.createElement(element)    var btn = document.createElement("BUTTON");

// Retirar un elemento HTML
document.removeChild(element)      list.removeChild(list.childNodes[0]);

// Concatenar un elemento HTML
document.appendChild(element)      document.getElementById("myList").appendChild(document.createElement("LI"));

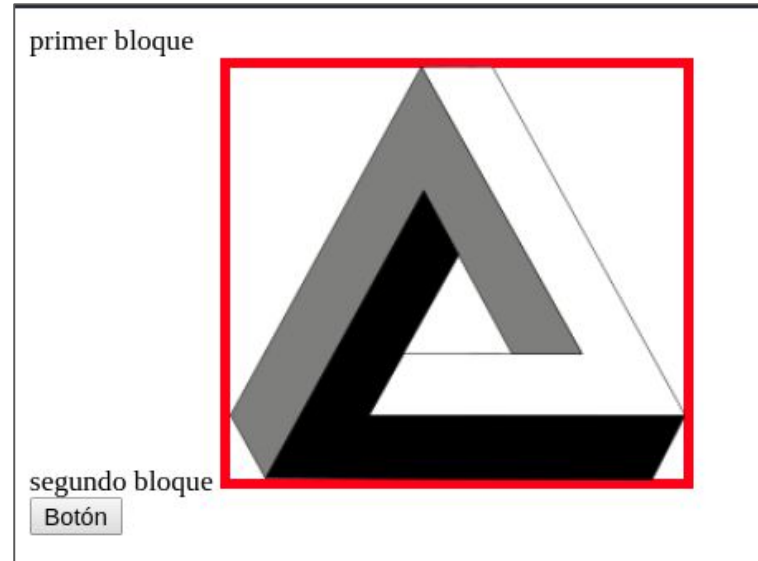
// Escribir directamente en el documento
document.write(text)               document.write("Hello World!");
```



Ejercicios de ejemplo

- En un documento HTML, encontrar un elemento con id="contenido"
- Introducir en él una imagen
- Aplicar un borde rojo de 5px a la imagen

<https://jsfiddle.net/yu16gnjk/>



Javascript - Eventos

- Javascript se ejecuta secuencialmente.
- Gracias a los eventos podemos definir manejadores que se ejecuten **al dispararse un evento**.
 - Pulsar una tecla, redimensionar la ventana, cambiar el foco, mover el ratón, enviar formulario...

Eventos como atributos en HTML

onload, onclick, onmouseover, onmouseout, onsubmit

```
<input type="button" value="boton con evento" onclick="alert('Evento onclick');" />
```

```
<div style="width:150px; height:60px; border:thin solid silver"
onmouseover="resalta(this)"
onmouseout="resalta(this)"
>
  Contenido.
</div>
```

```
<input type="button" value="boton con evento" onclick="saludo()" />
```

```
function saludo() {
  document.write("Hola");
}
```

```
function resalta(elemento) {
  switch(elemento.style.borderColor) {
    case 'silver':
    case 'silver silver silver silver':
    case '#c0c0c0':
      elemento.style.borderColor = 'black';
      break;
    case 'black':
    case 'black black black black':
    case '#000000':
      elemento.style.borderColor = 'silver';
      break;
  }
}
```

Javascript - Eventos

El nombre de la función manejadora no se pone con () cuando es un manejador!

Eventos definidos desde Javascript

```
<script>
  function saludo() {
    document.write("Hola");
  }

  // Asignar la función externa al elemento
  document.getElementById("pinchable").onclick = saludo;
</script>

<input id="pinchable" type="button" value="Input que puedes pinchar" />
```

Debemos asegurarnos de que el elemento al que aludimos ya ha sido cargado...

En ocasiones esto implica esperar a que se cargue toda la página antes de definir el evento:

```
window.onload = function() {
  document.getElementById("pinchable").onclick = saludo;
}
```

Podemos emplear parámetros en los manejadores mediante [addEventListener](#) o mediante [bind](#).



Ejercicios de ejemplo

- Aplicar a un botón existente la siguiente funcionalidad:
 - Al pasar el ratón por encima imprime por consola el mensaje “Clíckame ya!”
 - Cada vez que se pulsa se incrementa un contador y se imprime su valor en el documento HTML

<https://jsfiddle.net/pxcfw64k/>

Javascript - Objetos Predefinidos

- Hay una serie de objetos predefinidos en Javascript que pueden sernos muy útiles
 - Array
 - String
 - Date
 - Math
 - Objetos del navegador...

Javascript - Objetos Predefinidos - Array

Con el objeto Array

```
// Definición básica
var zapatos = new Array();

// Reservando N posiciones
var coches = new Array(4);

// Dando valores
var marcas = new Array('HP', 'Microsoft', 'Lenovo');
```

Sin el objeto Array

```
var dias = [];

var dias2 = ['lunes', 'martes', 'miércoles', 'jueves'];
```

- Length devuelve el tamaño del array (es editable)
- Las posiciones van desde 0 hasta (length-1)

```
var dias = ['lunes', 'martes', 'miercoles', 'jueves', 'viernes', 'sabado', 'domingo'];

// Recorrido de un array con la propiedad length
for(var i=0; i<dias.length; i++) {
    console.log("Días actual: " + dias[i]);
}

// Añadimos un nuevo elemento, modificando su tamaño.
dias.length = 10; // Añado 3 elementos de tipo undefined

// Eliminamos elementos
dias.length = 6; // De lunes a sabado
```

Link: [Métodos de Array](#)

Javascript - Objetos Predefinidos - String

```
var miCadena = "Hola"; = var miCadenaObjeto = new String("Hola");
```

También tiene la propiedad "length".

Métodos

[charAt\(\)](#)

[charCodeAt\(\)](#)

[concat\(\)](#)

[constructor](#)[endsWith\(\)](#)

[fromCharCode\(\)](#)

[includes\(\)](#)

[indexOf\(\)](#)

[lastIndexOf\(\)](#)

[length](#)[localeCompare\(\)](#)

[match\(\)](#)

[prototype](#)[repeat\(\)](#)

[replace\(\)](#)

[search\(\)](#)

[slice\(\)](#)

[split\(\)](#)

[startsWith\(\)](#)

[substr\(\)](#)

[substring\(\)](#)

[toLocaleLowerCase\(\)](#)

[toLocaleUpperCase\(\)](#)

[toLowerCase\(\)](#)

[toString\(\)](#)

[toUpperCase\(\)](#)

[trim\(\)](#)

[valueOf\(\)](#)

Javascript - Objetos Predefinidos - Math

Existe de manera implícita. No hace falta crearlo.

Propiedades

- Math.E
- Math.PI
- Math.SQRT1_2
- Math.SQRT2

Métodos

abs	Absolute value
sin, cos, tan	Standard trigonometric functions; argument in radians
acos, asin, atan, atan2	Inverse trigonometric functions; return values in radians
exp, log	Exponential and natural logarithm, base e
ceil	Returns least integer greater than or equal to argument
floor	Returns greatest integer less than or equal to argument
min, max	Returns greater or lesser (respectively) of two arguments
pow	Exponential; first argument is base, second is exponent
random	Returns a random number between 0 and 1.
round	Rounds argument to nearest integer
sqrt	Square root

```
> var res = Math.min(5,Math.PI);  
   console.log("El minimo entre 5 y PI es: "+res);  
El minimo entre 5 y PI es: 3.141592653589793
```

Javascript - Objetos Predefinidos - Date

- Para fechas y horas
- Los meses van de 0 (enero) a 11 (diciembre)

```
new Date();  
new Date(value);  
new Date(dateString);  
new Date(year, month [, day, hour, minute, second, millisecond])
```

```
fecha = new Date("May 20, 2013 19:00:00");  
fecha = new Date(2013, 6, 25);  
fecha = new Date(2013, 6, 25, 19, 0, 0);
```

- **Date.now()** devuelve cuantos milisegundos han transcurrido desde el 1 de Enero de 1970. Podemos utilizar esta información en el constructor.

```
var d = new Date(Date.now());  
// Converting the number of millisecond in date string  
a = d.toString()  
// Printing the current date  
document.write("The current date is: " + a)
```

The current date is: Mon Feb 27 2023 10:22:06 GMT+01

Javascript - Objetos Predefinidos - Date

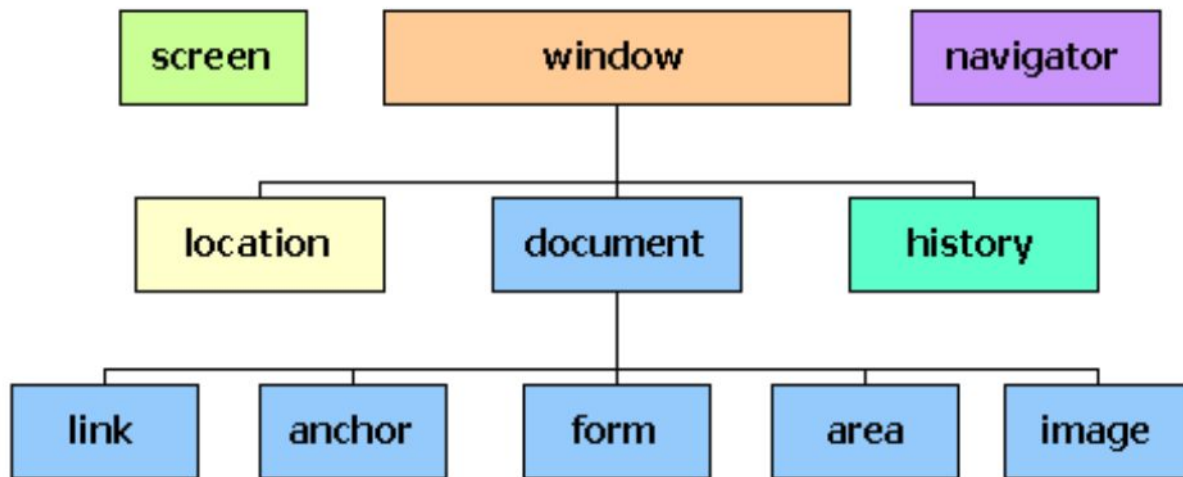
GETTERS

<code>.getDate()</code>	1..31
<code>.getDay()</code>	0..6 (sun..sat)
<code>.getFullYear()</code>	2014
<code>.getMonth()</code>	0..11
<code>.getHours()</code>	
<code>.getMinutes()</code>	
<code>.getSeconds()</code>	
<code>.getMilliseconds()</code>	
<code>.getTime()</code>	ms since epoch
<code>.getTimezoneOffset()</code>	

SETTERS

<code>.setDate (val)</code>
<code>.setDay (val)</code>
<code>.setFullYear (val)</code>
<code>.setMonth (val)</code>
<code>.setHours (val)</code>
<code>.setMinutes (val)</code>
<code>.setSeconds (val)</code>
<code>.setMilliseconds (val)</code>
<code>.setTime (val)</code>
<code>.setTimezoneOffset (val)</code>

Javascript - Objetos Del Navegador



- Un objeto está dentro del otro, como una propiedad más.
- “window” se entiende como la raíz. `window.document.write() = document.write()`

Javascript - Objetos Del Navegador

window

- Ventana del navegador
- Incluye los objetos referentes a la barra de tareas, el documento o la secuencia de direcciones de la última sesión.
- Entre otras cosas, con él podemos
 - Enviar alertas (alert) y diálogos (prompt y confirm)
 - Ejecutar porciones de código periódicamente (setInterval y setTimeout)
 - Cerrar la ventana (close) o abrir una nueva (open)
 - Cambiar el tamaño de la ventana (resizeBy)
 - Mover la ubicación del scroll (scrollBy y scrollTo)
- obtener el historial de navegación de la ventana (history)
 - obtener el tamaño de la ventana (innerHeight, innerWidth, outerHeight...)
 - consultar información sobre la pantalla y el navegador (screen, navigator)
 - conocer la ubicación del scroll de la ventana (scrollX, scrollY)
 - Acceder al almacenamiento del navegador (localStorage)

Métodos

Propiedades

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

Javascript - Window.localStorage

- Podemos almacenar información clave-valor en el almacenamiento local del usuario!
- Esa información se almacena hasta que el usuario la borre explícitamente desde el navegador
- Para guardar información perecedera cuando se cierre el navegador: **sessionStorage**

```
localStorage.setItem('myCat', 'Tom');  
  
var cat = localStorage.getItem('myCat');  
  
localStorage.removeItem('myCat');  
  
// Clear all items  
localStorage.clear();
```

```
if (sessionStorage.clickcount) {  
    sessionStorage.clickcount = Number(sessionStorage.clickcount) + 1;  
} else {  
    sessionStorage.clickcount = 1;  
}  
document.getElementById("result").innerHTML = "You have clicked the button " +  
sessionStorage.clickcount + " time(s) in this session.";
```



Ejercicios de ejemplo

- Un HTML en el que:
 - Cada segundo se actualice y muestre en pantalla la hora actual
 - A los 10 segundos abra otra ventana a <http://www.unavarra.es>
 - Guarde en almacenamiento local un elemento “unavarra” con la hora a la que se ha abierto unavarra.es



Ejercicios de ejemplo

- Un HTML en el que:
 - Cada segundo se actualice la hora actual
 - A los 10 segundos abra otra ventana a <http://www.unavarra.es>
 - Guarde en almacenamiento local un elemento “unavarra” con la hora a la que se ha abierto unavarra.es



Javascript - Formularios

- “.value” para obtener información de un formulario

```
<form name="formuA">
  <input type="text" name="nombre" id="nombre"/>
  <input type="button" value="Aceptar" onclick="saludar()">
</form>

<div id="resultado">
</div>
```

```
saludar = function(){
  //Opcion 1; Mediante el identificador del input
  var nombre = document.getElementById("nombre").value;

  //Opcion 2: Mediante el objeto forms
  nombre = document.forms[0]["nombre"].value

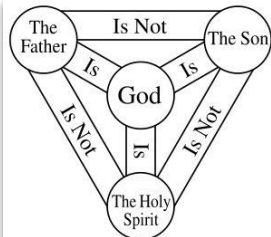
  if (nombre == "") {
    alert("Error: nombre vacío");
    return false;
  }

  document.getElementById("resultado").innerHTML = "Hola, " + nombre;
  return true;
}
```

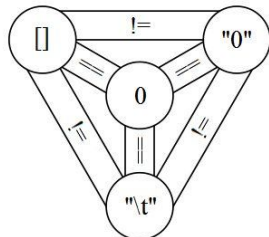
Javascript - WTF

```
var a = [1,2,3];  
var b = [1,2,3];  
var c = '1,2,3';
```

```
a == c; // true
b == c; // true
a == b; // false
```



Christianity



JavaScript

@hsiohs

Console Search Emula

  <top frame>

```
> 0 > null
```

false

```
> 0 >= null
```

true

```
> 0 == null
```

false

```
> 0 <= null
```

true

```
> 0 < null
```

false

> FML

```
> var a=0.1
```

undefined

```
> a=a+a+a
```

0,300000000000000000000004

$$> (a-0.3) == 0$$

false

>

```
> '5' - 3
2          // weak typing + implicit conversions * headaches
> '5' + 3
'53'      // Because we all love consistency
> '5' - '4'
1          // string - string * integer. What?
> '5' ++ '5'
'55'
> 'foo' ++ 'foo'
'fooNaN' // Marvelous.
> '5' + - '2'
'5-2'
> '5' + - + - - + - - + + - + - + - - - '2'
'52'      // Apparently it's ok

> var x * 3;
> '5' + x - x
50
> '5' - x + x
5          // Because fuck math
```

[illegible]

```
alert(1);
```

<http://www.jsfuck.com>

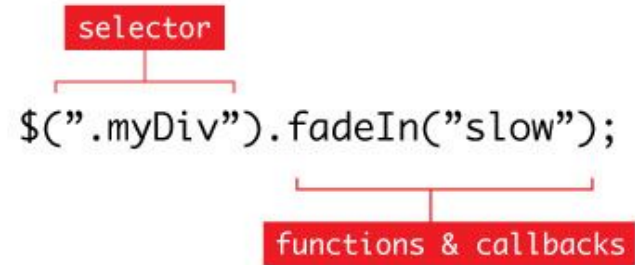
Modelando Comportamiento



<https://api.jquery.com/>

jQuery - ¿Qué es?

- jQuery es un framework Javascript, gratuito, abierto y “ligero”.
- “*Write less, do more*” : Encapsula y simplifica tareas habituales que normalmente se harían en múltiples líneas de código.
- Simplifica
 - La manipulación del DOM y CSS, efectos y animaciones.
 - El uso de AJAX y fetch
 - La gestión de eventos



The diagram illustrates the components of the jQuery code snippet `$(".myDiv").fadeIn("slow");`. A red bracket above the text `$(".myDiv")` points to a red box labeled "selector". Another red bracket below the text `.fadeIn("slow")` points to a red box labeled "functions & callbacks".

```
graph TD; selector[selector] --- S$(".myDiv"); S --- F(".fadeIn(\"slow\")"); F --- callbacks[functions & callbacks];
```

`$(".myDiv").fadeIn("slow");`

jQuery - Ejemplos

- Obtener elementos del documento HTML

```
var listItems = $( 'ol li' );  
var claseItem = $( '.clase' );  
var identificadorItem = $( '#identificador' );
```

- Esperar a la carga de un elemento HTML:

```
$( document ).ready(function() {  
    console.log( 'HTML cargado al completo!' );  
});
```

- Mostrar/ocultar un elemento al hacer click en otro.

```
$( "#clickme" ).click(function() {  
    $( "#book" ).toggle( "slow" );  
});
```

```
<div id="clickme">  
    Click here  
</div>  

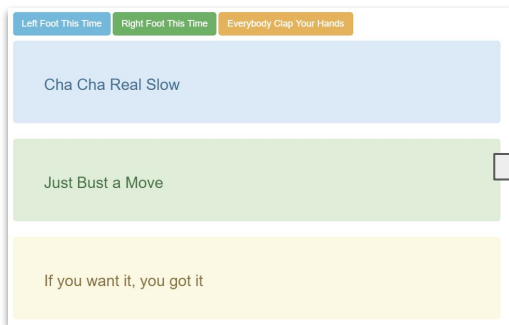
```

<https://jsfiddle.net/fmskycbt/>

jQuery - Ejemplos 2

- Animaciones (<https://jsfiddle.net/L7CnR/41/>)

```
$(document).ready(function () {  
  
    $("#top").click(function () {  
        $("#watch1").animate({  
            borderRadius: 100,  
            paddingBottom: 24,  
            paddingTop: 24  
        }, 1000);  
    });  
  
    $("#middle").click(function () {  
        $("#watch2").animate({  
            fontSize: 75  
        }, 2000);  
    });  
  
    $("#bottom").click(function () {  
        $("#watch3").animate({  
            paddingLeft: 300  
        }, 1000, "swing");  
    });  
});
```



jQuery - ¿Cómo incluirlo?

- En el header (Cuidado con el orden de carga de los elementos!)
- Desde un CDN (Content Delivery Network):
 - [De google](#) por ejemplo

```
<head>
...
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
...
</head>
```

- [Descargándolo](#), guardándolo en nuestro servidor y añadiéndolo como recurso.

```
<head>
...
<script src="./js/jquery-3.7.1.min.js"></script>
...
</head>
```


JavaScript y las 1001 librerías

three.js

<https://threejs.org/>

Animaciones 3D



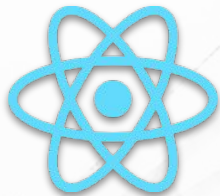
<https://www.chartjs.org/>

Graficas, tablas,
presentación de datos



<https://d3js.org/>

Presentación de
datos



<https://reactjs.org/>

Creación de interfaces



<https://angularjs.org/>

Front-end framework



jsPDF

<https://parall.ax/products/jspdf>

Creador de PDF

∞ más...