

Teoria-JavaScript.pdf



ShadowedWarrior



Lenguajes de marcas y sistemas de gestión de información



1º Desarrollo de Aplicaciones Multiplataforma



Estudios España



Accede al documento original

antes



**Descarga sin publi
con 1 coin**



Después

WUOLAH



HASTA
-40%

Teoría JavaScript

Código

```
<script type="text/javascript">
```

```
// Zona de mensajes
```

```
alert('¡Esto es una "alert" de JavaScript!')
console.log('Mensaje de log de la consola (console.log)')
console.info('Mensaje de información (console.info)')
console.error('Mensaje de error (console.error)')
console.warn('Mensaje de warning (console.warn)')
```

```
// Zona de depuración
```

```
//-----
// debugger // El debugger es utilizado para hacer una
depuración de código en la página web
// console.log('Log después de hacer uso de debugger')
```

```
//-----
// Zona de variables
```

```
//-----
// const // Variables que durante la ejecución de nuestra
página no van a cambiar de valor
```

```
//-----
// let // Variables que pueden cambiar de valor durante la
ejecución
//      // pero deben ser leídas para poder ser utilizadas
// var // Variables que pueden cambiar de valor durante la
ejecución
//      // y pueden no haber sido leídas antes para ser
utilizadas, pero saldrán como "undefined"
```

```
//-----
// Variable global (se puede utilizar en cualquier momento)
```

```
const NOMBRE = 'David';
let primerapellido = 'Solera';
var segundoapellido = 'Romero';
```

```
// En este caso 'NOMBRE' daría fallo y no cambiaría el nombre
al ser
```

```
// una variable const, que no puede ser modificada en la
ejecución
```

```
// NOMBRE = 'Juan';
primerapellido = 'Sanchez';
segundoapellido = 'Carricondo';
```

NEXT-LEVEL AI PC

No más "mi portátil no carga en clase", "se queda colgado", "el ventilador ruge".
Este lo llevas, lo abres, y ¡boom! listo para salvar la mañana. Y sí, con ofertas para ti.

[Ver ofertas](#)

```

// Variables locales (se define dentro de una condición)
if (true) {
    var condicion = "Hello World!"
    console.log(condicion)
}

function juan() {
    let condicion = "Bye World!"
    console.log(condicion)
}

//-----

// Operaciones
// + Suma
console.log(2 + 2)
// - Resta
console.log(4 - 2)
// / División
console.log(6 / 2)
// * Multiplicación
console.log(3 * 2)
// < Menor que
console.log(10 < 20)
// > Mayor que
console.log(40 > 20)
// <= Menor o igual que
console.log(20 <= 21)
// >= Mayor o igual que
console.log(41 >= 40)
// == o === Igual que (el triple igual también identifica que
sean del mismo tipo de variable mientras que el doble no lo hace)
console.log(20 === '20')
console.log(20 == 20)
// != No es igual que
console.log(30 != 20)

// typeof / Tipo de variable que se está utilizando
let num = 20
console.log(typeof num)
// NaN / Variable con número y texto
const notnum = 2 + 'pepe';
console.log(typeof notnum)
// isNaN() / Comprobación de que la variable no es un número
console.log(isNaN(NaN))
console.log(isNaN(num))
console.log(isNaN(notnum))
console.log(isNaN('pepe'))
// toString() / Cambiar variable a string
let numero = 100.1676
console.log(typeof numero)

```

```

console.log(typeof numero.toString())
// .toFixed() / Mostrar o redondear decimales de una variable
console.log(numero.toFixed(3))

// Ejercicio práctico 1
// Una cena en la que hay que pagar 102€ y se reparten entre 6
personas

const TOTALL = 102
let PERSONAS = 6
let TOTAL_PERSONA = TOTALL / PERSONAS
console.log(TOTAL_PERSONA)

// Ejercicio práctico 2
// Carrito de la compra donde hay varios productos que sumar
// Aplicar 10% de descuento por ser su primera compra

const MOVIL = 300;
const CASCOS = 30;
const FUNDA = 10;
let DESCUENTO = 0.1;
let TOTALLL = (MOVIL + CASCOS + FUNDA);
let SUBTOTAL = (TOTALLL * DESCUENTO);
let TOTAL_COMPRA = (TOTALLL - SUBTOTAL);
console.log(TOTAL_COMPRA);

// Strings
// Concatenar texto // Con '+'
console.log('Hello ' + 'World')
// template literals // Para no concatenar con el más utilizar
`${variable}`
const name = 'Juana'
console.log(`Hello ${name}`)
// typeof // Tipo de variable
console.log(typeof name)
// .length // Longitud
console.log(name.length)
// .includes() // Comprobar si incluye el texto indicado
console.log(name.includes('Ju'))
console.log(name.includes('Cr'))
// .slice(start, end) // Cortar el texto en los caracteres
indicados
console.log(name.slice(0,3))
// .replace('este texto', 'por este otro') //
console.log(name.replace('ana', 'an'))
console.log(name.replace('Ju', 'Jinc'))
// .trim() // Limpia los valores de espacio y extraños que se
encuentren
// ----- // al principio y final de la entrada de texto (no
limpia los de por medio)
const prueba = '      Pero      esto no sé por qué ocurre      '

```



LA NUESTRA DURA MÁS



```
console.log(prueba.trim())
// .split(',') // Nos separa el texto según el separador que le
indiquemos
// ----- // (como si fuese un cut de Linux) y nos otorga
un Array
const separacion = 'Calle Gallón, 3, 2ª planta'
console.log(separacion.split(','))

// Ejercicio práctico 1
// Texto de una newsletter donde dan unas variables dinámicas
que vienen del usuario
// Usamos comillas invertidas ``" para introducir variables en
un texto
const Nombre = 'David'
const Juego = 'Red Dead Redemption 2'
const Descuento = 20
const newsletter = `Hola ${Nombre}
Como regalo de bienvenida te vamos a
ofrecer un descuento del ${Descuento}%
en el videojuego ${Juego}
Disfrútalo, un saludo.`
// Con querySelector accedemos a todos los nodos, y
seleccionamos la clase <p></p> que hemos
// creado al inicio del HTML y le ponemos como .innerHTML la
variable con el contenido deseado
document.querySelector('.newsletter-text').innerHTML =
newsletter

// Booleanos
// Admite "true" y "false"
// Nos ayuda a detectar estados dentro de la aplicación,
básicamente hace comprobaciones
// Aclaración: "Boolean()" es un objeto
// Para crear simplemente un booleano, podemos hacer esto:
let ejemplobooleano = true;
let booleanoejemplo = false;

// Valores que nos devuelven "true"
console.log(Boolean(1))
console.log(Boolean('Juan'))
console.log(Boolean(3.14))
console.log(Boolean(100 > 5))
console.log(Boolean(1 < 100))
console.log(Boolean('1' == 1))

// Valores que nos devuelven "false"
console.log(Boolean(0))
console.log(Boolean(-0))
console.log(Boolean(""))
console.log(Boolean(NaN))
console.log(Boolean(null))
```

WUOLAH

```

console.log(Boolean(undefined))
console.log(Boolean('1' === 1))

// Ejercicio práctico 1
// Mostrar una información al usuario dependiendo si está
logueado o no
// Si está conectado: "Haz click aquí para ver el artículo"
// Si no está conectado: "Debes registrate para leer este
artículo"

const USER_LOGGED = false;
let TEXTO = document.querySelector('.bloque-texto')
if (USER_LOGGED == true) {
    TEXTO.innerHTML = "Haz click aquí para ver el contenido."
} else {
    TEXTO.innerHTML = "Debes registrate para leer este
artículo."
}

// Lo que hacemos con lo de arriba es crear una variable
diciendo que no está logeado (o que lo está)
// para luego utilizar un bloque de texto creado arriba en un
<p></p> (bloque-texto) con el selector document.querySelector
// Dentro de un if indicamos que si está logeado, cambie la
variable (que se hace así: variable.innerHTML) y hacemos
// que muestre lo que queremos que se vea por pantalla
// Hacemos lo mismo con el else, ponemos lo que se muestre en
caso de que no esté logeado
// Tenemos una solución más fácil aquí |
var usuarioLogueado = false;

if (usuarioLogueado == true) {
    document.write('<p>Haz clic aquí para ver el
contenido.</p>');
} else {
    document.write('<p>Debes registrarte para leer este
artículo.</p>');
}

// Creamos una variable igual que antes y directamente en el
if, si es true, crea el <p></p>
// y si es el caso de que no es true, lo crea en el else
// document.write sirve para escribir en el documento. Ambas
soluciones son válidas.

// Null y Undefined

// undefined es de tipo undefined
// undefined quiere decir que hay una ausencia de valor (aún
así puede que la variable esté definida)
console.log(a)

```


OFERTAS
BLACK FRIDAY

msi®

Esta oferta

es como una doble victoria
tuya: disfrútala ahora porque
no pasa dos veces.

Ver ofertas



HASTA
-40%

Tu viejo portátil ya dio lo que tenía que dar. Pásate a MSI: rápido, potente y sin dramas. Lo enciendes y estás listo para todo. Aprovecha las ofertas y despédete del modo “se cuelga cada dos por tres”.

```

    var a = 'Variable de prueba' // También sirven las variables
let (solo que nos soltará un error directamente si trata de leerla
antes)
    console.log(a)

    // null es de tipo objeto
    // null quiere decir que la variable está definida pero que su
valor es nulo
    console.log(c)
    var c = null
    console.log(c)

    // Funciones
    // Las funciones son tareas que tienen cumplen 1 o más
objetivos y en ellas se crea un nuevo entorno
    // Podemos utilizar variables globales aquí y las locales que
definamos que este entorno
    function sayHello() {
        document.write('<p>¡Sean todos bienvenidos!</p>')
    }
    sayHello()

    var num1 = 30;
    var num2 = 21;
    function suma() {
        document.write(num1 + num2)
    }
    suma()
    // Definimos la función y luego la ejecutamos así:
nombreFuncion()

    // Parámetros
    // Pueden ser recibidos aunque no es obligatorio
    // Puedes poner todos los parámetros que quieras
    // Puedes usarlo tantas veces como quieras para obtener
diferentes resultados
    function sum(a, b, c, d) {
        console.log(a + b)
    }
    sum(num1, num2)
    sum(7, 9)

    // Podemos hacer que diga algo por defecto y si se le introduce
un parámetro lo reemplaza como en este caso
    function hola(name = 'World') {
        console.log('Hello ' + name)
    }
    hola('Juan')
    hola()

    // Retorno de valor

```


Portátiles desde
549€



msi

BLACK FRIDAY

```
function retorno(name = 'World') {
    return `Hello ${name}`;
}
console.log(retorno('David'))
console.log(retorno())

// Funciones anónimas
// No tienen nombre y pueden ser asociadas a una variable
var sayhello = function(name = 'World') {
    return `Hello ${name}`
}
console.log(sayhello('Pepe'))
console.log(sayhello())

// Funciones flechas / arrow
// No tienen nombre con una sintaxis reducida. También se
asocian a variables
const mensaje = 'Texto de ejemplo';
var anonima = (name) => `Hello ${name}` // De esta forma ya
hemos creado la función arrow
var anonymous = (name) => {

    return `Bye ${name}`
} // Función arrow usando el return y corchetes (si se
necesitan más líneas de código)

console.log(anonima('Pedro'))
console.log(anonymous('Rubén'))

// Aplicar lógica con JavaScript (visto de varias formas)
(podemos utilizar diagramas para entender la lógica mejor)
// Quiero irme de viaje - ¿Puedo? + Sí, lo hago - No, ahorraré
más

// Condicionales (if, else if, else)
// Comparar con ==, ===, &&, ||
const VUELO = 200;
const HOTEL = 50;
const TOUR = 40;
let TOTAL = VUELO + HOTEL + TOUR;
let PRESUPUESTO = 190;

if (PRESUPUESTO > TOTAL) {
    console.log('Puedo irme de viaje')
} else if (PRESUPUESTO === TOTAL) {
    console.log('Voy justo de dinero, tengo un presupuesto
ajustado')
} else {
    console.log('No puedo irme de viaje, tengo que ahorrar')
}
```



VER OFERTAS

WUOLAH

```

        // Operadores ternarios (podemos usarlo cuando tenemos 2
posibles soluciones)
        // Comprueban una condición y dan una respuesta o bien positiva
o bien negativa
        // (Reutilizo variables anteriores)
        const TERNARIO = PRESUPUESTO >= TOTAL
            ? console.log('Sí, me voy de viaje') // ? Respuesta
positiva
            : console.log('No, no me voy de viaje') // Respuesta
negativa

```

Todo este código se encuentra en → **Pruebas Web** → **Código JS en HTML**

La diferencia entre Java y JavaScript es que Java es un lenguaje de programación general y JavaScript es uno específico.

En **Java** puedes crear **aplicaciones para diversos fines**: para móviles, para escritorio, para páginas web, etc.

JavaScript está destinado a ser **ejecutado** en un **navegador** y por lo tanto, está orientado totalmente a la web. También es un **lenguaje interpretado con respuesta inmediata**, que añade **interactividad a la web** y **efectos visuales dinámicos**. Puede ser **utilizado** tanto de forma **local** como del **lado del servidor**.

jQuery es una librería de código JavaScript de código abierto, por lo que se nos permite utilizarlo en los proyectos.

Podemos añadir JavaScript en el documento **HTML directamente** o en un **documento aparte**, que es como lo utilizaremos.

En el <head> </head> o en <body> </body>

```

<script>
    alert("Hola alumnos");
</script>

```

En documento aparte, escribimos en el <head> </head>

<script src="ruta/nombreArchivo.js"></script>

Incluso podemos utilizar los dos métodos a la vez, pero al hacer uso de un **archivo independiente**, tenemos dos **ventajas**:

- **Organización extra en el código**
- **Una mayor reutilización de código**

Ejemplo sencillo de JQuery

Tenemos descargado un archivo JQuery listo para ser enlazado con HTML

Hacemos uso de `<script>`

```
<script src="video3pi/Jquery/jquery-1.6.3.min.js"></script>
```

Y luego llamamos a unas funciones que se encuentran en su interior

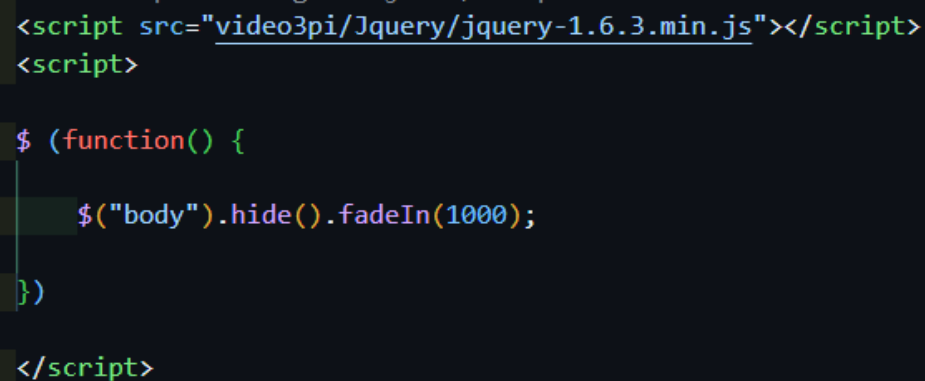
```
<script>
```

```
$ (function() {
```

```
    $("body").hide().fadeIn(1000);
```

```
})
```

```
</script>
```



```
<script src="video3pi/Jquery/jquery-1.6.3.min.js"></script>
<script>

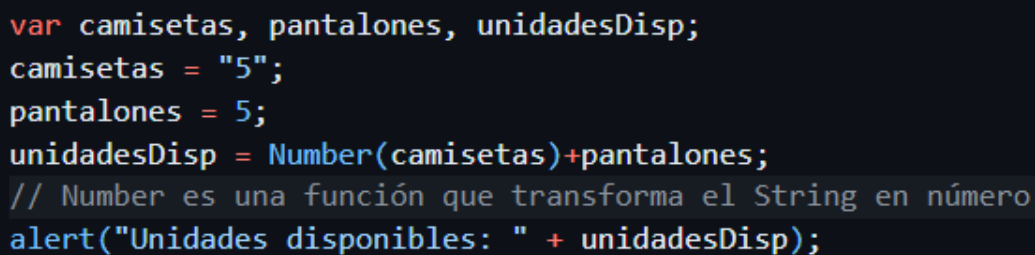
$ (function() {

    $("body").hide().fadeIn(1000);

})

</script>
```

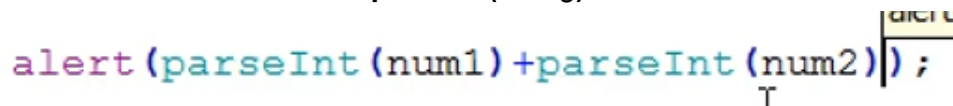
Ejemplo de código JavaScript



```
var camisetas, pantalones, unidadesDisp;
camisetas = "5";
pantalones = 5;
unidadesDisp = Number(camisetas)+pantalones;
// Number es una función que transforma el String en número
alert("Unidades disponibles: " + unidadesDisp);
```

La función **Number(String)** sirve para pasar un String a número entero.

También nos sirve la función **parseInt(String)**



```
alert (parseInt (num1)+parseInt (num2));
```

Si no puede pasarlo a entero, devolverá un **isNaN** (is Not a Number)

HASTA
-40%

Escribiremos nuestro nombre y apellidos en una etiqueta `<p> </p>` con JavaScript. Para añadir cualquier **etiqueta en HTML** simplemente la **metemos entre comillas**, como en el **ejemplo** a continuación:

```
// Programa almacena nuestro nombre y apellidos
// y posteriormente los muestra en la web
var nombre = "David", apellidos = "Solera Romero";
document.write("<p>"); // Abrimos una etiqueta párrafo
    document.write(nombre + " " + apellidos); // Contenido párrafo
document.write("</p>"); // Cerramos el párrafo
```

Ahora usaremos la función **prompt**, que realiza la función de entrada de datos en nuestra página web, una **petición de datos**. Repetimos lo anterior usando **prompt**. Si después introducimos una coma, dejamos un texto por defecto

```
nombre=prompt("Introduce tu nombre por favor","Escribe aquí el nombre");
```

```
// Programa que pide nuestro nombre y apellidos
// y posteriormente los muestra en la web
var nombre = prompt("Introduce tu nombre");
var apellidos = prompt("Introduce tus apellidos");
document.write("<p>"); // Abrimos una etiqueta párrafo
    document.write(nombre + " " + apellidos); // Contenido párrafo
document.write("</p>"); // Cerramos el párrafo
```

Ahora nos aparece una ventana emergente esperando la introducción de datos

Operadores de **incremento y decremento** en JavaScript

Operador	Qué hace	Sintaxis
+=	Incrementa en x el valor de una variable	Record+=10
-=	Decrementa en x el valor de una variable	Record-=7
=	Multiplica en x el valor de una variable	Record=3
/=	Divide en x el valor de una variable	Record/=4
++	Incrementa en 1 el valor de una variable	Record++
--	Decrementa en 1 el valor de una variable	Record--

Ver ofertas



Ejemplo de uso del operador -= restando edad:

```
var edad = prompt("Introduce tu edad");  
var resta = edad-=2;  
document.write("<p>"); // Abrimos una etiqueta párrafo  
document.write("Tenías " + resta + " años en 2021");  
document.write("</p>"); // Cerramos el párrafo
```

También podemos incluir saltos de líneas con: "
"

```
// var resta = edad-2; // También se modifica la variable edad  
var resta = edad-2;  
document.write("<p>"); // Abrimos una etiqueta párrafo  
document.write("Bienvenido al curso " + nombre + " " + apellidos + ", tenías " + resta  
+ " años en 2021" + "<br>" + "Ahora mismo tienes " + edad + " años."); // Contenido párrafo  
document.write("</p>"); // Cerramos el párrafo
```


Arrays en JavaScript

Los valores almacenados **no tienen por qué ser del mismo tipo** y podemos almacenar múltiples valores

Tenemos dos formas de declarar un Array en JavaScript:

```
var artículos=["zapatillas","pantalón","camiseta","calcetines"];
```

```
var artículos=new Array("zapatillas","pantalón","camiseta","calcetines");
```

Accedemos a las posiciones del Array igual que en Java:

```
alert(artículos[i]);
```

En JavaScript tenemos las siguientes propiedades y métodos para tratar **Arrays**
Propiedades

artículos.length; → Devuelve la longitud del Array

Métodos

artículos.push("balón"); → Agrega elementos al final

artículos.push("balón","raqueta","bolsa"); → Agrega elementos al final

artículos.unshift("balón"); → Agrega elementos al principio

artículos.unshift("balón","raqueta","bolsa"); → Agrega elementos al principio

artículos.pop(); → Elimina elementos al final del Array

artículos.shift(); → Elimina elementos al principio del Array

```
// Arrays en JavaScript
var articulos = ["Balón", "Zapatos", "Camiseta", "Pantalones", "Calcetines"]; // Se puede
var articulos = new Array("Balón", "Zapatos", "Camiseta", "Pantalones", "Calcetines");

document.write("<p> El primer artículo es <strong>")
document.write(articulos[0] + "</strong> </p>");
```

Organiza tu futuro: estudia hoy para destacar mañana.

En Carpe Diem te esperan cursos adaptados a ti.
Una forma fácil y real de avanzar profesionalmente.

¡Quiero formarme!



Ejemplo de uso de Arrays

```
// Arrays en JavaScript
var articulos = ["Balón", "Zapatos", "Camiseta", "Pantalones", "Calcetines"]; // Se puede declarar de las dos formas
var articulos = new Array("Balón", "Zapatos", "Camiseta", "Pantalones", "Calcetines"); // Se puede declarar de las dos formas

document.write("<p> El primer artículo es <strong>")
document.write(articulos[0] + "</strong> </br>");
document.write("El último artículo es <strong>" + articulos[articulos.length-1] + "</strong></p>"); // Vemos el último elemento
articulos.unshift("Pepito"); // Añadimos al principio
articulos.push(prompt("Indica qué elemento añadir")); // Añadimos al final
alert(articulos[0] + " y " + articulos[articulos.length-1]); // Los leemos
articulos.shift(); // Eliminamos el primer elemento
articulos.pop(); // Eliminamos el último elemento
alert(articulos[0] + " y " + articulos[articulos.length-1]); // Leemos de vuelta
```

WUOLAH

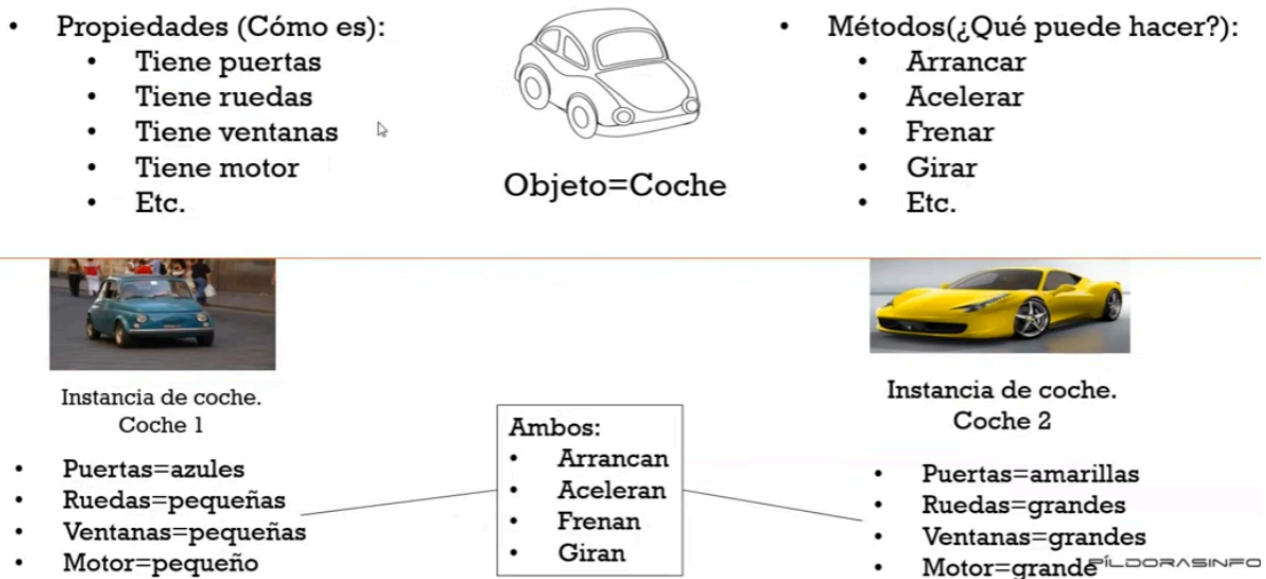
POO

JavaScript es un lenguaje orientado a objetos → un estilo de programación basado en clases que contienen objetos, donde **cada objeto** tiene sus **propias características y funcionalidades propias**

Las características son denominadas **atributos o propiedades**

Las funcionalidades son denominadas **métodos**

Ejemplo:



En **JavaScript**, un objeto podría ser un **botón**, el cual tiene propiedades y métodos, por lo cual **podemos modificar sus propiedades y utilizar esos métodos** según lo que queramos hacer.

Podemos cambiar sus propiedades y usar sus métodos con la nomenclatura del punto → Ancho de un botón → **boton.style.width="500px";**

Trabajando con un botón

Añadimos un botón en HTML, con input type="button"

```
<input type="button" id="boton1">
```

A continuación almacenamos el botón en una variable en JavaScript

document es la clase que identifica al documento, y al poner un **punto**, tenemos los diferentes métodos como **getElementById**

```
var miboton = document.getElementById("boton1");
```

Ya podemos modificar sus valores con .style.width/height="10px";

```
miboton.style.width="250px";  
miboton.style.height="250px";
```

Se vería así en la página web



Igualmente podemos cambiarle el color

```
miboton.style.background="red";
```

Utilizamos el método focus() para que aparezca preseleccionado el botón

Creamos otro cuadro para notar la diferencia



Condicionales

Los condicionales funcionan exactamente igual que en **Java**, incluso con la misma sintaxis que en este otro lenguaje.

```
if (condición) {
```

```
    //código a ejecutar si la condición es verdad
```

```
}
```

```
else{
```

```
    //código a ejecutar si la acción no es verdad
```

```
}
```

HASTA
-40%

Comparadores de JavaScript para if, for, y while

Operadores de
comparación

Operador	Significado
==	Igual que...
!=	Diferente que...
>	Mayor que...
<	Menor que...
>=	Mayor o igual que...
<=	Menor o igual que...
===	Estrictamente igual que (comprueba también tipos)
!==	Estrictamente diferente que.... (comprueba tipos)

Operadores lógicos

Operadores
lógicos

Operador	Significado
&&	Y lógico
	O lógico

&& es AND y || es OR

Para hacer uso de else if:

```

if(colorWeb=="Azul"){
    cuerpo.style.backgroundColor="blue";
}
else if(colorWeb=="Rojo"){
    cuerpo.style.backgroundColor="red";
}
else if(colorWeb=="Verde"){
    cuerpo.style.backgroundColor="green";
}
else{
    alert("Ese color no está contemplado");
}

```

Para resolver problemas de mayúsculas, tenemos los métodos de la clase **String** los cuales son **toLowerCase()** o **toUpperCase()**, que van después del punto

```

var colorWeb=prompt("Introduce el color de fondo. Azul/Verde/Rojo").toLowerCase();

```



Para comprobar si una variable no es un número utilizamos la función **isNaN(variable)**, y para comprobar que es un número podemos usar el operador de lo contrario ! → **!isNaN(variable)**

```
if(!isNaN(num1) && !isNaN(num2)){
```

En el ejemplo indicamos que si **num1** es un número y **num2** es un número se ejecute el programa

Bucles

Mismos bucles que en Java:

Determinados → **For**

Indeterminados → **While, Do While**

Utilizamos la misma sintaxis que en Java

```
While (condición){
```

```
    //código a repetir mientras la condición del bucle sea cierta
```

```
}
```

Mini Ejemplo con bucle While

```
var meses=["Enero", "Febrero", "Marzo", "Abril", "Mayo", "Junio", "Julio",  
"Agosto", "Septiembre", "Octubre", "Noviembre", "Diciembre"];  
  
var contador=0;  
  
while(contador<meses.length){  
    document.write(meses[contador]);  
  
    contador++;  
}
```

Creamos un array y un contador, y recorremos con el bucle while el array hasta la longitud del array mientras escribimos en el documento mes a mes.

Sintaxis **Do While** (igual que en Java)

```
Do{  
    //código a repetir mientras la condición del bucle sea cierta  
}while (condición);
```

Para matemáticas tenemos la clase **Math** en JavaScript

Utilizaremos el método **Math.random()** que nos devolverá un número aleatorio entre 0 y 1. Si queremos entre 0 y un número concreto, multiplicamos por ese número.

Además, redondeamos para que no sea un número float

```
var aleatorio=Math.round(Math.random()*100);
```

Vamos a hacer un programa para adivinar un número entre 0 y 100.

```
var num = Math.round(Math.random()*100);  
var minum, intentos;  
intentos = 0;  
do {  
    minum = prompt("Introduce un número entre 0 y 100");  
    do {  
        if (minum < 0 || minum > 100 || isNaN(minum)){  
            minum=prompt("Introduce un número válido");  
        } } while (minum < 0 || minum > 100 || isNaN(minum));  
    intentos++;  
    if (minum > num) {  
        alert("El número introducido es mayor que el generado");  
    } else if (minum < num) {  
        alert("El número introducido es menor que el generado");  
    }  
} while (minum!=num);  
alert("¡Enhorabuena! Has adivinado el número aleatorio");  
alert("Lo has conseguido en " + Number(intentos) + " intentos");
```



Bucle For (igual que en java)

```
for(inicio;condición;incremento/decremento){
```

//código a repetir mientras la condición del bucle sea cierta

```
}
```

Incremento → i++

Decremento → i--

Incremento de 2 en 2 → i+=2

Ejemplo while y for

Rellenamos un Array con un bucle while

```
while (persona!="salir") {

    persona=prompt("Introduce trabajador");

    Trabajadores[contador]=persona;

    contador++;

}
```

Borramos el último elemento, que sería **salir**, puesto que es la condición del while

```
Trabajadores.pop();
```

Y luego escribimos el contenido en el documento

```
for(var i=0;i<Trabajadores.length;i++){

    document.write(Trabajadores[i] + "<br/>");

}
```




LA NUESTRA DURA MÁS

Nuestra tecnología y
nuestra garantía, granuja...



webuy.com

Funciones (están muy ligadas a los eventos)

Es básicamente un conjunto de instrucciones que pueden ser reutilizadas (como si fuese un comando), que realizan una tarea específica aceptando y devolviendo argumentos.

Sintaxis

```
function nombre_función(){  
  
    //Código a ejecutar por la función  
  
}
```

Para llamar a la función, usamos **nombre_funcion()**;

Ejemplo:

JS

```
// Día de hoy  
function dia_de_hoy() {  
  
    var hoy = new Date();  
    //document.write(hoy);  
    document.write(hoy.toString());  
  
}
```

HTML

```
  
<p>Hoy es día: <script>dia_de_hoy();</script></p>
```

Se les pueden pasar **argumentos** a las **funciones**, indicando **entre los paréntesis cuántos argumentos recibirá**, para **luego ser tratados** en ese conjunto de instrucciones definidas. Tantos argumentos como la función tenga definidos

```
function suma( num1, num2 ){  
  
    var resultado=num1+num2;  
  
}
```

Para utilizar esta función → **suma(5, 10)**;

Ejemplo

JS

```
// Muestra nombre  
  
function saludo(nombre){  
  
    document.write("hola " + nombre);  
  
}  
var minombre=prompt("Introduce tu nombre");
```

HTML

```
<p>Hoy es día: <script>dia_de_hoy();</script></p>  
<p>Además, <script>saludo(minombre);</script></p>
```


Introducción a eventos y su utilidad en las funciones

Los eventos son desencadenantes de una acción; cuando ocurre un evento específico, se ejecutará la acción ligada a ese evento. Esto nos permite interactuar de forma directa con la web.

Eventos más importantes de JavaScript

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>

HASTA
-40%

NEXT-LEVEL AI PC

No más "mi portátil no carga en clase", "se queda colgado", "el ventilador ruge". Este lo llevas, lo abres, y ¡boom! listo para salvar la mañana. Y sí, con ofertas para ti.

Vamos a probar un **evento básico con una imagen**. Ejemplo:

Nota: Esta forma de implementar los eventos está desaconsejada

Evento relacionado: Hacer click

Esto es lo que ocurrirá al hacer click:

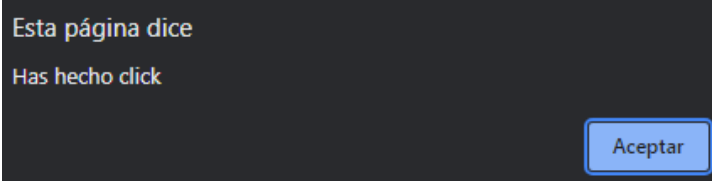
```
function aviso() {  
    alert("Has hecho click");  
}
```

Vamos a la etiqueta de la imagen y añadimos **onclick="nombreFuncion"**

También podríamos usar ondblclick (doble click) o onmouseover (ratón por encima)

```
 </br>
```

Al hacer click nos enseña una ventana alert



Ejemplo video 25

Tenemos una **tabla HTML** que representa una **calculadora** (números del 1 al 9, operadores, un igual y una coma). Cada botón está identificado con un **id** y una **class**, al igual que cada botón provoca un evento.

Cambiamos la información de la pantalla de la calculadora cambiando su valor en la función preparada para ello:

```
var cifra = "";  
function calculadora(parametro) {  
    document.getElementById("display").value = cifra + parámetro;  
    cifra = document.getElementById("display").value;  
}
```

Al hacer clic en cada botón provocamos una acción que **escribe el número** presionado en la pantalla de la calculadora, y **reutilizamos** esa acción para **todos los números**, simplemente **cambiando el parámetro** pasado.

Al hacer clic en un número, **se guardan todos los números** escritos y se vuelven a escribir junto al número presionado para **no sobrescribir la información**.

Igualmente podemos hacer que **aparezca un 0 de forma predeterminada** situando este código tras haber cargado la web:

```
<script>  
  
document.getElementById("display").value=0;  
  
</script>
```

Utilizamos esta función / acción cuando le demos a **sumar “+”**

```
function suma () {  
  
    acumulado=acumulado+parseInt(cifra);  
    document.getElementById("display").value=acumulado;  
  
    cifra="";  
}
```

Acumula el número introducido en **acumulado** y lo va mostrando por pantalla mientras resetea el número **cifra** para que se introduzca otro diferente.