



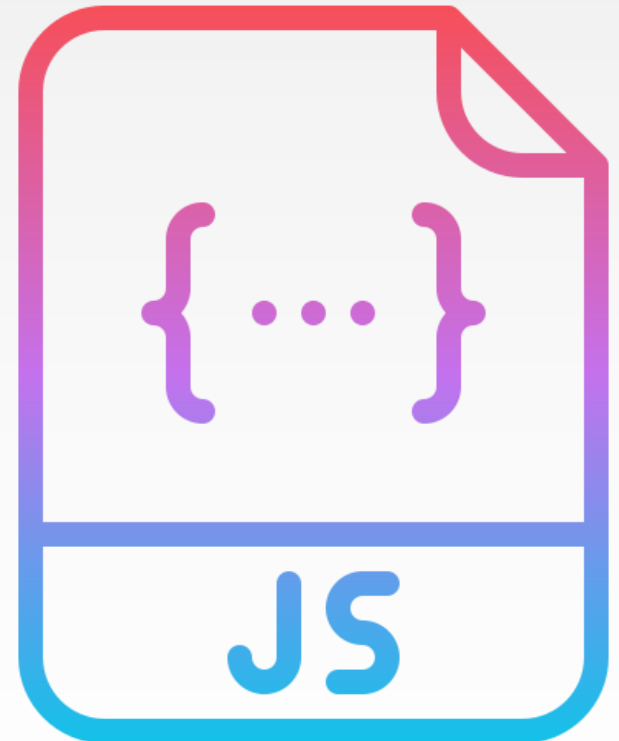
Introducción a JavaScript

FUNDAMENTOS DE DESARROLLO WEB

Curso académico
2023 – 2024

Índice

1. Introducción
2. Lenguaje JavaScript
3. Inclusión de JavaScript en una web
4. Árbol BOM (Browser Object Model)
5. Árbol DOM (Document Object Model)
6. Manejadores de eventos
7. Algunos métodos
8. Formularios
9. Menús



1. Introducción

■ ORIGEN

- Creado en sólo 10 días en mayo de 1.995, por Brendan Eich cuando trabajaba en Netscape. Actualmente en Mozilla.
- Nombre original: Mocha.
- Septiembre de 1.995: LiveScript.
- Diciembre de 1.995: JavaScript.
- Estandarizado bajo el nombre de ECMAScript (1.996-1.999).
- Creación de páginas web dinámicas.
 - Se actualizan cuando el usuario hace peticiones, navegando por las páginas o actualizando su contenido, de forma que incorporan todo tipo de efectos visuales (texto, imágenes, animaciones, ventanas con mensajes de avisos...).

1. Introducción

■ CARACTERÍSTICAS

- Lenguaje de programación interpretado.
- Débilmente tipado.
- Basado en Objetos.
- De propósito general, pero utilizado habitualmente en navegadores Web (client-side JavaScript).
- Sentencias de control con sintaxis similar a *Java* o *C*.
 - if, if else, switch
 - for, while, do while
 - return, break, continue

2. Lenguaje JavaScript

■ SINTAXIS DEL LENGUAJE

- Es *case sensitive*
 - Distingue entre mayúsculas y minúsculas.
- No se define el tipo de las variables. A diferencia de otros lenguajes de programación, una misma variable puede almacenar diferentes tipos de datos.
- Cada sentencia acaba con el carácter ; aunque no es obligatorio.
- Comentarios

// Esto es un comentario de una línea.

/ Esto es un comentario
de varias líneas. */*

2. Lenguaje JavaScript

■ VARIABLES

- Identificador \equiv nombre de la variable
 - Sólo puede estar formado por letras, números y los símbolos '\$' y '_'. El primer carácter no puede ser un número.
 - NO puede ser una palabra reservada del lenguaje.
- Declaración con la palabra reservada 'var'.

var identificador; *¡¡¡NO se indica el TIPO!!!*

- Inicialización

identificador = valor;

- Ejemplo (declaración e inicialización en la misma sentencia).

```
var num1 = 2;  
var num2 = 3;  
var suma = num1 + num2;
```

2. Lenguaje JavaScript

■ VARIABLES

■ Formas de declarar una variable:

- 1) Con la palabra reservada **var**
- 2) Con la palabra reservada **let**
- 3) Con la palabra reservada **const**
- 4) Sin usar ninguna palabra reservada

- *var* se usa para crear variables globales, mientras que *let* y *const* pueden crear variables de bloque.
- Con *let* y *const* no se puede re-declarar una variable, mientras que con *var* sí.

```
>> var x = "¡Hola!"  
  
>> var x = 10
```

```
>> let x = "¡Hola!";let x = 10;
```

❗ Uncaught SyntaxError: redeclaration of let x
note: Previously declared at line 1, column 4

2. Lenguaje JavaScript

Palabras Reservadas					
abstract	debugger	final	instanceof	protected	transient
boolean	default	finally	int	public	true
break	delete	float	interface	return	try
byte	do	for	let	short	typeof
case	double	function	long	static	var
catch	else	goto	native	super	void
char	enum	if	new	switch	volatile
class	export	implements	null	synchronized	while
const	extends	import	package	this	with
continue	false	in	private	throw	yield

2. Lenguaje JavaScript

- TIPOS DE DATOS

- Simples

- Numéricos (*Number*)
 - Cadenas de caracteres (*String*)
 - Booleanos (*Boolean*)
 - Null
 - Undefined

- Objetos

- Arrays
 - Expresiones regulares
 - Funciones
 - Objetos

2. Lenguaje JavaScript

■ TIPOS DE DATOS

■ Numéricos

- Valores enteros o decimales.
- Dos valores especiales: *NaN*, *Infinite*.

■ Cadenas de texto o *strings*.

- Encerradas entre comillas dobles o simples.

var c1 = "Esto es una cadena de texto.";

var c2 = 'Esto también es una cadena de texto.';

- Para incluir caracteres especiales se utilizan los siguientes caracteres de escape.

Nueva línea	Tabulador	Comilla simple	Comilla doble	Barra inclinada
<code>\n</code>	<code>\t</code>	<code>\'</code>	<code>\"</code>	<code>\\</code>

2. Lenguaje JavaScript

■ TIPOS DE DATOS

■ Booleanos

- Dos valores: *true*, *false*

■ Arrays

- Conjunto de variables. Pueden ser de cualquier tipo.
- Un *string* es un array de caracteres.
- Declaración e inicialización:

```
var nombre_array = [valor1, valor2, ..., valorN];
```

■ Ejemplo:

```
var diasSemana = ["Lunes", "Martes", "Miércoles", "Jueves", "Viernes",  
                 "Sábado", "Domingo"];  
var inicioSemana = diasSemana[0]; // inicioSemana = "Lunes"
```




2. Lenguaje JavaScript

■ OPERADORES

Asignación	Incremento/ Decremento	De comparación	Matemáticos	Lógicos
=	++	===	+ (suma números)	&& (and)
+=	--	!==	+ (concatena strings)	 (or)
-=		<	- (resta)	! (negación)
*=	Prefijo (se incrementa antes de la operación)	<=	* (multiplicación)	
/=		>	/ (división)	
%=	Sufijo (se incrementa después)	>=	% (módulo)	

2. Lenguaje JavaScript

■ PUERTAS LÓGICAS

<p>NOT</p>  <p>A — Y</p>	<table><tr><th>A</th><th>Y</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	Y	0	1	1	0	<p>Negación</p> <p>$Y = !A$</p>									
A	Y																
0	1																
1	0																
<p>OR</p>  <p>A — Y B —</p>	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Y	0	0	0	0	1	1	1	0	1	1	1	1	<p>Suma</p> <p>$Y = A B$</p>
A	B	Y															
0	0	0															
0	1	1															
1	0	1															
1	1	1															
<p>AND</p>  <p>A — Y B —</p>	<table><tr><th>A</th><th>B</th><th>Y</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	Y	0	0	0	0	1	0	1	0	0	1	1	1	<p>Multiplicación</p> <p>$Y = A \&\& B$</p>
A	B	Y															
0	0	0															
0	1	0															
1	0	0															
1	1	1															

2. Lenguaje JavaScript

- SENTENCIAS DE CONTROL DE FLUJO

- if ... else

```
if ( condición ) {    // 0, “ ” y null equivalen a false  
    // Instrucciones  
}  
else {  
    // Instrucciones  
}
```

- for

```
for ( inicialización; condición; actualización ) {  
    // Instrucciones  
}
```

2. Lenguaje JavaScript

- SENTENCIAS DE CONTROL DE FLUJO

- Ejemplo *if ... else*

```
>> var nota = 6

>> if(nota >= 5){
    console.log("Aprobado");
}else{
    console.log("Suspenso");
}

Aprobado
```

- Ejemplo *for*

```
>> for ( var num = 0; num < 10; num ++ ) {
    console.log(num);
}
```

2. Lenguaje JavaScript

■ SENTENCIAS DE CONTROL DE FLUJO

■ for ... in

```
for ( índice in array ) {  
    // Instrucciones  
}
```

```
>> var finSemana = ["Sábado", "Domingo"];  
>> for (index in finSemana){  
    console.log(finSemana[index]);  
}
```

Sábado

Domingo

■ while, do ... while

```
while( condición ){  
    // Instrucciones  
}
```

```
do {  
    // Instrucciones  
} while( condición )
```

```
>> while (true) {  
    console.log("Bucle infinito");  
}  
  
do{  
    console.log("Bucle infinito");  
}  
while (true)
```


2. Lenguaje JavaScript

- SENTENCIAS DE CONTROL DE FLUJO

- Ejemplo *for*

```
>> for(var count = 0; count < 5; count ++){  
    console.log(count);  
}
```

- Ejemplo *while*

```
>> var count = 0;  
    while(count < 5){  
        console.log(count);  
        count ++;  
    }
```

2. Lenguaje JavaScript

- SENTENCIAS DE CONTROL DE FLUJO

- switch

```
switch ( expresión ) { // La expresión devuelve un numero,  
    // un valor lógico o un string  
    case valor1:  
        // Instrucciones caso 1  
    break; // para acabar el switch  
    case valor2:  
        // Instrucciones caso 2  
    break;  
    default: // opcional  
        // Instrucciones si no se diera ningún caso  
}
```

2. Lenguaje JavaScript

■ SENTENCIAS DE CONTROL DE FLUJO

■ Ejemplo *switch*

```
>> ▼ var nota = 1;
    switch (nota){
      case 1:
        console.log("¡Has suspendido :(!");
        break;
      case 4:
        console.log("¡Casi apruebas :(!");
        break;
      case 5:
        console.log("¡Has aprobado, por los pelos...!");
        break;
      case 9:
        console.log("¡¡Enhorabuena, has sacado un sobresaliente!!");
        break;
      default:
        console.log("¡¡¡Error: valor no válido!!!");
    }

¡Has suspendido :(!
```

2. Lenguaje JavaScript

- SENTENCIAS DE CONTROL DE FLUJO

- Excepciones `try ... catch`

```
try {  
    // Código a ejecutar  
}  
catch(error) {  
    // Gestión del error  
}
```

- Para lanzar la excepción: *throw exception*

<https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Sentencias/throw>

2. Lenguaje JavaScript

■ FUNCIONES

- Bloque de código (con parámetros) asociado a un nombre. La función se invoca (o ejecuta) por el nombre y devuelve un valor como resultado.
- Las funciones permiten crear operaciones de alto nivel. Se denominan también abstracciones o encapsulaciones de código.
- Se definen con la palabra reservada ***function*** seguida del nombre y los parámetros entre paréntesis. Y a continuación el bloque de código entre llaves {}, que termina con la sentencia ***return***.
- En la invocación se deben asignar valores concretos a los parámetros.
- La función representa el valor resultante de su ejecución. Si no hay un ***return***, acaba y devuelve ***undefined***.

<https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Funciones>

2. Lenguaje JavaScript

■ FUNCIONES

■ Sintaxis

```
function nombre_funcion ( arg1, arg2, ...){  
    // instrucciones  
    return resultado;  
}
```

■ Ejemplo

```
>> function multiplica(a,b){  
    return a*b ;  
}  
multiplica(3,2);  
← 6
```

2. Lenguaje JavaScript

■ FUNCIONES

■ **alert(), confirm(), prompt()**

- Reciben como argumento un mensaje que sacan como un cuadro de diálogo o cajas PopUp.

The image displays three distinct JavaScript dialog boxes, each with a white background and a light gray border. The first dialog box on the left, created by `alert()`, contains the text 'Esta página dice' and 'Cuidado hay JavaScript' with a single blue 'Aceptar' button at the bottom right. The middle dialog box, created by `confirm()`, contains the text 'Esta página dice' and 'Cerrar ventana', featuring a blue 'Aceptar' button and a white 'Cancelar' button with a blue border at the bottom. The third dialog box on the right, created by `prompt()`, contains the text 'Esta página dice' and 'Entrar clave:', followed by a text input field and two buttons at the bottom: a blue 'Aceptar' button and a white 'Cancelar' button with a blue border.

3. Inclusión de JavaScript en una web

- Hay tres formas de incluir JavaScript en un documento HTML:

- 1ª Dentro de las propias etiquetas de HTML.

`<p onclick = "alert('¡¡Gracias por pulsar!!')"> PÚLSAME </p>`

- 2ª Escribir el código JavaScript en el documento HTML dentro de la etiqueta `<head>`. También puede ir en el *body*.

```
<head>  
  <script type = "text/javascript">  
    document.write("Esto es una sentencia de JavaScript")  
  </script>  
</head>
```

- 3ª Definir JavaScript en un archivo externo.

```
<head>  
  <script type = "text/javascript" src = "archivo.js"> </script>  
</head>
```


3. Inclusión de JavaScript en una web

- En el caso de que el navegador NO soporte JavaScript o no lo tenga activado se puede indicar con la etiqueta `<noscript>`. Definido como elemento de bloque.

```
<body>
```

```
  <noscript>
```

```
    <p>
```

Esta página requiere JavaScript para su correcto funcionamiento. Compruebe si JavaScript está deshabilitado en el navegador.

```
    </p>
```

```
  </noscript>
```

```
</body>
```

Ejercicio 1

- Completar los condicionales *if()* del siguiente script para que los mensajes *alert()* se muestren siempre de forma correcta.

```
var numero1 = 5;
var numero2 = 8;

if(...) {
    alert("El numero1 es menor que el numero2");
}
if(...) {
    alert("El numero2 es positivo");
}
if(...) {
    alert("El numero1 es negativo o distinto de cero");
}
if(...) {
    alert("Incrementar en 1 el numero1 no lo
        hace mayor o igual que el numero2");
}
```

Ejercicio 1: Solución

```
<!DOCTYPE html>
<html lang = "es">
  <head>
    <meta charset="utf-8">
    <title>JavaScript</title>
    <script type="text/javascript">
      var numero1 = 5;
      var numero2 = 8;

      if(numero1 < numero2) {
        alert("El numero1 es menor que el numero2");
      }
      if(numero2 >= 0) {
        alert("El numero2 es positivo");
      }
      if(numero1 < 0 || numero1 != 0) {
        alert("El numero1 es negativo o distinto de cero");
      }
      if(++numero1 <= numero2) {
        alert("Incrementar en 1 unidad el valor de numero1 no lo hace mayor o igual que numero2");
      }
    </script>
  </head>

  <body>
    <p>Ejemplo de uso de la estructura de control IF</p>
  </body>
</html>
```

Ejercicio 2

- Crear una página web que enlace un archivo de JavaScript, cuyo objetivo es pedir por teclado algunos números (usar la función *prompt()*) para realizar las siguientes operaciones:
 - **Multiplica:** Crea una función que realice la operación numA multiplicado por numB y ponga en la web el resultado (usar la función *document.write()*). Solicita los dos números y después formatea en la web el texto:

numA x numB igual a resultado

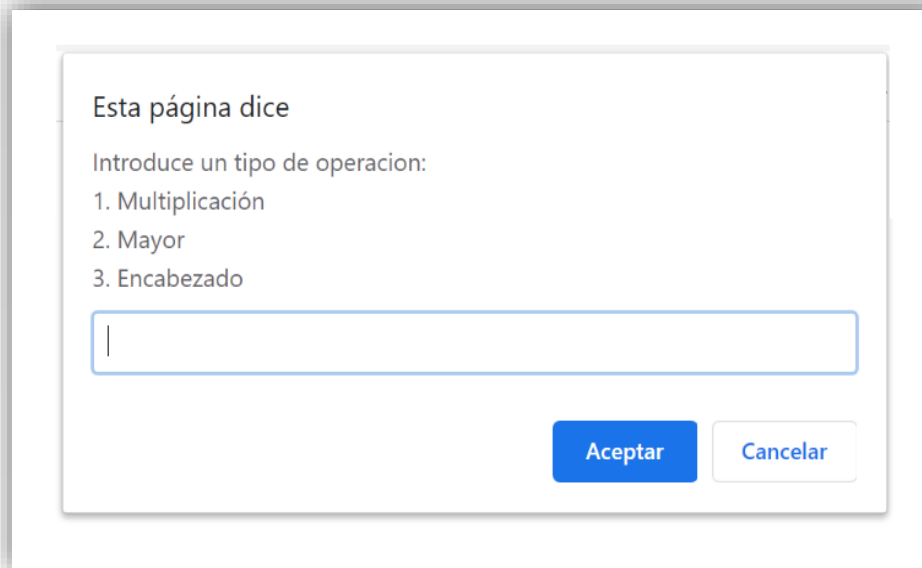
- **Mayor:** Crea una función que solicite dos números y compare y muestre por pantalla cuál de los dos es mayor. Formatea la salida para que muestre:

numA es mayor que numB

- **Encabezado:** Crea una función que solicite un texto y un número, y escriba el texto en el formato de encabezado que le indique la variable número.

Ejercicio 2

- Para seleccionar cada una de estas operaciones podéis usar un *prompt()*, como se muestra en la siguiente imagen, de manera que si introduzco un 1 hará la multiplicación, si pongo un 2 calculará el mayor y con 3 me pondrá un texto con el tamaño del encabezado indicado.



A screenshot of a JavaScript `prompt()` dialog box. The dialog has a white background and a thin gray border. It contains the following text: "Esta página dice" at the top, followed by "Introduce un tipo de operacion:". Below this is a numbered list: "1. Multiplicación", "2. Mayor", and "3. Encabezado". Under the list is a text input field with a blue border and a vertical cursor. At the bottom right are two buttons: a blue "Aceptar" button and a white "Cancelar" button with a gray border.

Ejercicio 2: Solución

■ Archivo .js

```
// SOLUCIÓN EJERCICIO 2
var selection = prompt("Introduzca el tipo de operación a realizar: \n 1.Multiplicación \n 2.Mayor \n 3.Encabezado");

switch (selection){
    case '1':
        Multiplicar();
        break;
    case '2':
        Mayor();
        break;
    case '3':
        Encabezado();
        break;
    default:
        alert(";;;Error: valor introducido no válido!!!");
}

function Multiplicar(){
    var num1 = prompt("Introduzca el primer número:");
    var num2 = prompt("Introduzca el segundo número:");
    num1 = parseInt(num1);
    num2 = parseInt(num2);

    /*También puede hacerse con el operador + para indicar que es un número y no un String
    var num1 = +prompt("Introduzca el primer número");
    var num2 = +prompt("Introduzca el segundo número");*/

    var resultado = num1*num2;

    if(isNaN(num1) || isNaN(num2)){
        alert(";;;Error: valor introducido no válido!!!");
    } else{
        document.write(num1 + ' x ' + num2 + ' = ' + resultado);
    }
}
```

Ejercicio 2: Solución

- Archivo .js
 - Continuación

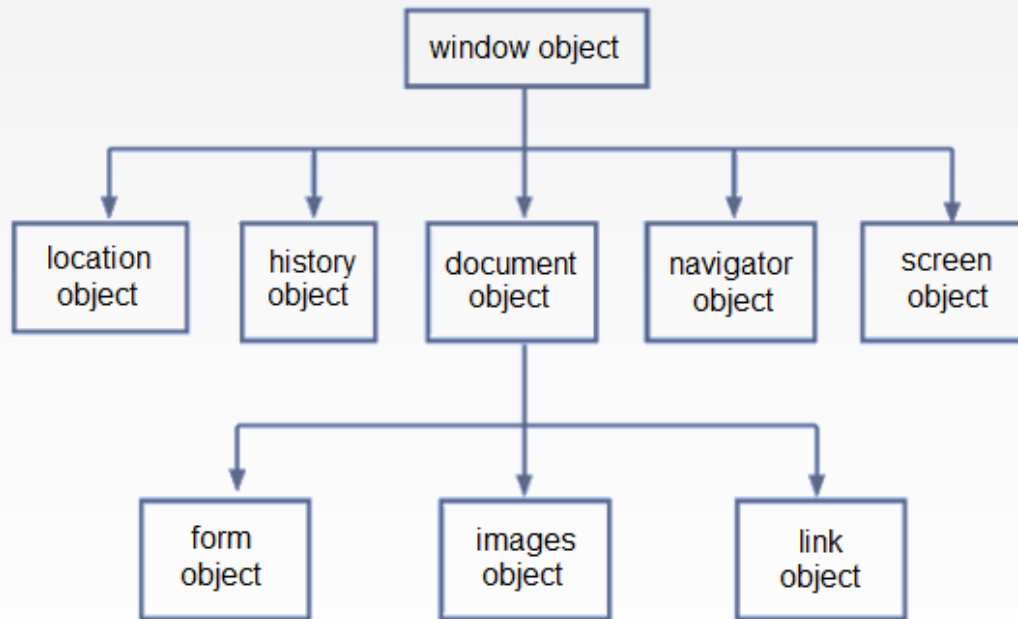
```
function Mayor() {
    var num1 = +prompt("Introduzca el primer número:");
    var num2 = +prompt("Introduzca el segundo número:");
    if (num1 > num2) {
        document.write(num1 + ' es mayor que ' + num2);
    } else if (num2 > num1) {
        document.write(num2 + ' es mayor que ' + num1);
    } else if (num1 === num2) {
        document.write(num2 + ' es igual a ' + num1);
    } else {
        alert(";;;Error: valor introducido no válido!!!");
    }
}

function Encabezado() {
    var texto = prompt("Introduzca un texto para el encabezado:");
    var num = +prompt("Introduzca un número para el tamaño entre 1 y 6:");

    if (isNaN(num)) {
        alert(";;;Error: valor introducido no válido!!!");
    }
    else if (num < 1 || num > 6) {
        alert(";;;Error: tamaño de encabezado no válido!!!");
    } else {
        document.write("<h" + num + ">" + texto + "</h" + num + ">");
    }
}
```

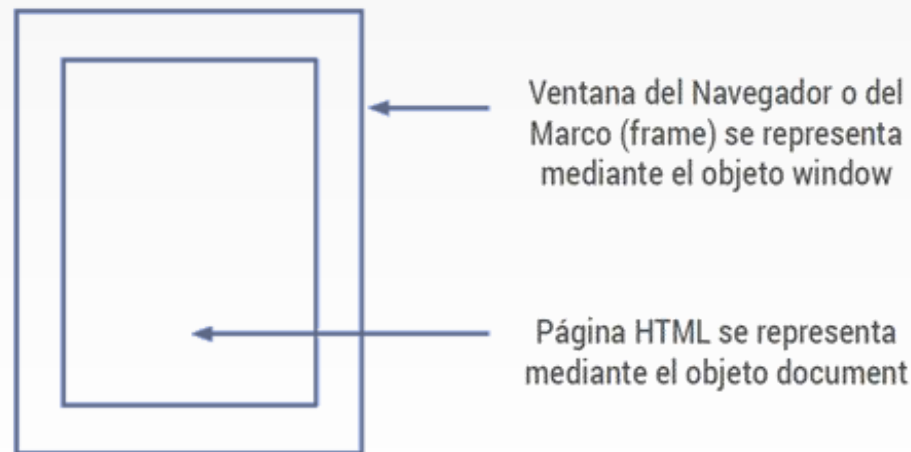
4. Árbol BOM (Browser Object Model)

- Objetos característicos que permiten interactuar con el navegador.



4. Árbol BOM (Browser Object Model)

- El objeto ***window*** es el de más alto nivel, contiene las propiedades de la ventana y en el supuesto de trabajar con marcos (*frames*), se genera un objeto *window* para cada uno.
- El objeto ***document*** contiene todas las propiedades del documento actual, como son su color de fondo, enlaces, imágenes, etc.



4. Árbol BOM (Browser Object Model)

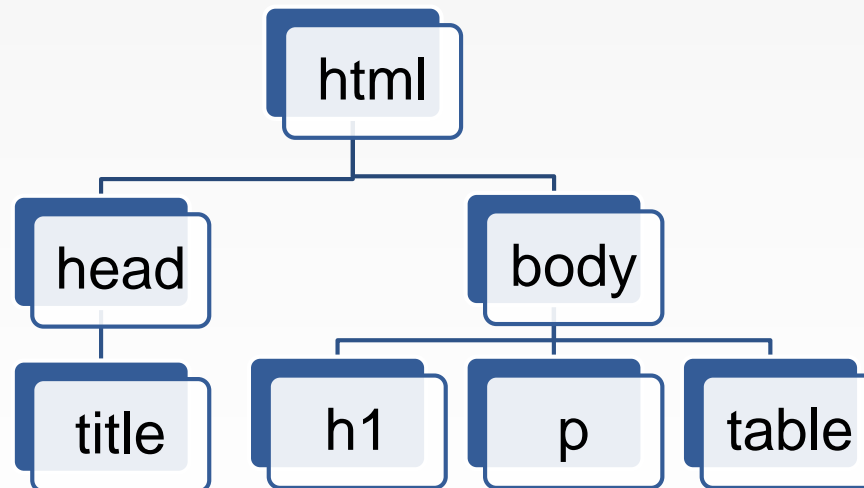
- El objeto ***location*** contiene toda la información sobre la URL que se está visualizando, así como todos los detalles de esa dirección (puerto, protocolo,...).
- El objeto ***history*** contiene información sobre los enlaces que el usuario ha visitado. Se usa principalmente para generar botones de avance y retroceso.
- El objeto ***navigator*** permite obtener información del navegador con el que se está visualizando la página.
- El objeto ***screen*** permite obtener información sobre la resolución de la pantalla.
 - Ejemplo objeto *screen*:

4. Árbol BOM (Browser Object Model)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset = "utf-8">
    <title> Ejemplo Objeto screen</title>
  </head>
  <body>
    <h2> Propiedades de mi pantalla </h2>
    <script type = "text/javascript">
      switch(screen.colorDepth) {
        case 1: case 4: document.bgColor = "gray"; break;
        case 8: case 15: case 16: document.bgColor = "blue"; break;
        case 24: case 32: document.bgColor = "lightgreen"; break;
        default: document.bgColor = "white";
      }
      document.write("Su pantalla soporta color de " + screen.colorDepth + " bits.");
      document.write(" La altura es de " + screen.height + " px,");
      document.write(" y la anchura son " + screen.width + " px.");
    </script>
  </body>
</html>
```

5. Árbol DOM (Document Object Model)

- Es una interfaz de programación para los documentos HTML.
- Define los objetos y propiedades de los elementos HTML y los métodos para acceder a ellos.
- Los objetos DOM permiten inspeccionar y modificar los elementos HTML.



5. Árbol DOM (Document Object Model)

- MÉTODOS PARA ACCEDER A ELEMENTOS DOM
 - *getElementById("my_id")*
 - Devuelve el objeto DOM con el identificador buscado o *null* si no lo encuentra
 - *getElementsByTagName("my_name")*
 - *getElementsByTagName("my_tag")*
 - *getElementsByClassName("my_class")*
 - Devuelven una matriz de objetos
- Lista de propiedades y métodos del objeto *document*
https://www.w3schools.com/jsref/dom_obj_document.asp
- Ejemplos de uso: https://www.w3schools.com/js/js_htmlDOM_elements.asp

5. Árbol DOM (Document Object Model)

- MÉTODOS PARA CAMBIAR O MODIFICAR ELEMENTOS
 - *document.write(string)*: para escribir en el documento (página web).
 - *element.getAttribute(attribute)*: devuelve el valor del atributo.
 - *element.setAttribute(attribute, value)*: permite modificar el valor del atributo, o añadir un nuevo atributo.
- PROPIEDADES
 - *element.innerHTML*: para cambiar el contenido del elemento.
 - *element.style.property*: para cambiar el estilo del elemento.
 - *element.attribute*: para cambiar un atributo del elemento.

6. Manejadores de Eventos

■ EVENTOS

- Los eventos permiten ejecutar acciones cuando el usuario realiza una determinada acción sobre la página web.
- Algunos de los más comunes son pasar el ratón o *clicar* sobre un elemento, cerrar la página o cambiar el contenido de un *input* en un formulario.
- Se definen con atributos con nombres especiales de elementos HTML.
- El valor asignado al atributo es código JavaScript ejecutado al ocurrir el evento.
- https://www.w3schools.com/js/js_htmldom_events.asp

6. Manejadores de Eventos

■ TIPOS

- Atributos de las etiquetas HTML

```
<input type="button" value="Púlsame" onclick="alert('Gracias por pulsarme');" >
```

- Funciones en código JavaScript externo

```
function muestra() {alert('Gracias');}
```

```
<input type="button" value="Púlsame" onclick="muestra()" >
```

- Semánticos

```
function muestra() {alert('Gracias');} // Función externa
```

```
// Asignar la función externa al elemento
```

```
document.getElementById("boton").onclick = muestra;
```

```
// Elemento HTML
```

```
<input id="boton" type="button" value="Púlsame" >
```


6. Manejadores de Eventos

■ EJEMPLO

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset = "utf-8">
  </head>
  <body>
    <h1 id="id1">My Heading 1</h1>
    <button onclick="document.getElementById('id1').style.color = 'blue'">
      Click Me!
    </button>
  </body>
</html>
```

My Heading 1

Click Me!

My Heading 1

Click Me!

6. Manejadores de Eventos

■ EJEMPLO CON IMAGEN

```
<!DOCTYPE html>

<html>
  <head>
    <meta charset = "utf-8">
    <title>Evento con imagen</title>
    <style> body{text-align:center;} </style>
  </head>

  <body>
    <h1>Eventos HTML con imágenes</h1>

  </body>
</html>
```

Eventos HTML con imágenes



6. Manejadores de Eventos

■ EVENTOS

- También existen eventos no producidos directamente por una acción del usuario.
 - Para detectar la carga de la página web se usa:

<body onload = "miFuncion()">

- Métodos para ejecutar una acción cada cierto tiempo:
 - `setInterval()`
 - `clearInterval()`
 - `setTimeout()`
 - El método `setInterval()` llama a la función que se le pasa como argumento indefinidamente, cada cierto tiempo especificado (en milisegundos).
 - El método `clearInterval()` detiene la ejecución del método `setInterval()`.
 - El método `setTimeout()` es igual que `setInterval()` pero se ejecuta solo una vez.

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

6. Manejadores de Eventos

■ EVENTOS

```
>> setInterval(function(){ alert("Hello"); }, 2000);
```

```
>> setInterval(hola, 2000);
```

```
function hola(){  
    alert("Hello");  
}
```

```
>> var myVar = setInterval(hola, 2000);
```

```
function stopHola() {  
    clearInterval(myVar);  
}
```

6. Manejadores de Eventos

■ EJERCICIO

Ejemplo de setInterval

00:00:10

Activar

Parar

Ejemplo de setInterval

00:00:02

Activar

Parar

Ejemplo de setInterval

¡BOOM!

Activar

Parar

6. Manejadores de Eventos

■ EJERCICIO

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset = "utf-8">
    <title> setInterval </title>
    <link rel = "icon" type = "image/svg" href = "bomb.svg">
    <style>
      .texto{
        color:red;
        font-size:50pt;
      }
    </style>
  </head>
  <body>
    <h1>Ejemplo de setInterval</h1>
    <p class="texto" id="reloj">00:00:10</p>
    <div><button onclick="activar()">Activar</button>
    <button onclick="desactivar()">Parar</button></div>
  </body>
</html>
```

Ejercicio - Solución

```
var t = 10;
var intervalo;
function restar() {
    if(t==0) {
        document.getElementById('reloj').innerHTML = "¡BOOM!";
        clearInterval(intervalo);
    } else {
        t--;
        document.getElementById('reloj').innerHTML = "00:00:0"+t;
    }
}
function activar() {
    intervalo = setInterval(restar,1000);
}
function desactivar() {
    clearInterval(intervalo);
}
```

6. Manejadores de 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>

6. Manejadores de Eventos

Evento	Descripción	Elementos
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>

6. Manejadores de Eventos

■ EJEMPLO

```
<!DOCTYPE html>
<html>
  <head>
    <title>Ejemplo mouseover</title>
    <script type='text/javascript'>
      function mOver(obj) {
        obj.innerHTML = "¡¡¡Gracias!!!";
      }
      function mOut(obj) {
        obj.innerHTML = "Pasa el ratón ;)";
      }
    </script>
  </head>
  <body>
    <div id = 'my_div' onmouseover="mOver(this)" onmouseout="mOut(this)"
      style="background-color:cyan; width:120px; height:20px; padding:40px;">
      Pasa el ratón ;)
    </div>
  </body>
</html>
```

Pasa el ratón ;)

¡¡¡Gracias!!!

6. Manejadores de Eventos

■ EJERCICIO GALERÍA

- Crear una página web que muestre una galería de fotos con dos botones, *Siguiente* y *Anterior*, para ir pasando las imágenes.
- Se puede usar el ‘truco’ de llamar a todas las imágenes igual salvo un índice
 - Por ejemplo: foto1.jpg, foto2.jpg, foto3.jpg, ...



6. Manejadores de Eventos

■ EJERCICIO GALERÍA

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset = "utf-8">
    <title>Galería de Motos</title>
    <link rel="stylesheet" type="text/css" href="styles.css">
    <script type = "text/javascript" src="scripts.js"></script>
  </head>
  <body>
    <div id="gallery">
      <button id="prevButton" onclick="prev()">Anterior</button>
      
      <button id="nextButton" onclick="next()">Siguiete</button>
    </div>
  </body>
</html>
```

6. Manejadores de Eventos

■ EJERCICIO GALERÍA

```
// Código JavaScript
var currentMoto = 1;

function prev(){
    currentMoto--;
    if (currentMoto !== 0) {
        document.getElementById("image").src = "images/moto0" + currentMoto + ".jpeg";
        //También puede hacerse así:
        //document.getElementById("image").setAttribute("src","images/moto0"+currentMoto+".jpeg");
    } else {
        currentMoto++;
    }
}

function next(){
    currentMoto++;
    if (currentMoto !== 6) { //5 imágenes
        document.getElementById("image").src = "images/moto0" + currentMoto + ".jpeg";
    } else {
        currentMoto--;
    }
}
```

6. Manejadores de Eventos

■ EJERCICIO GALERÍA

```

/* Archivo CSS */

body {
    text-align: center;
}

#gallery {
    text-align: center;
    display: inline-flex;
    flex-flow: row nowrap;
    align-items: center;
    margin: 30px;
}

button {
    width: 80px;
    height: 30px;
    margin: 10px;
    background-color: paleturquoise;
    transition: height 200ms, background-color 1000ms ease;
    /* -webkit-transition: background-color 2s ease-out;
    -moz-transition: background-color 2s ease-out;
    -o-transition: background-color 2s ease-out; */
}

button:hover {
    height: 50px;
    background-color: orange;
}

#image {
    width: 300px;
    height: 200px;
}

```

7. Algunos Métodos

■ PARA EL TIPO STRING

Método	Descripción	Ejemplo
<code>toUpperCase()</code>	Convierte los caracteres a mayúsculas	<code>'hola'.toUpperCase()</code>
<code>toLowerCase()</code>	Convierte los caracteres a minúsculas	<code>'HOLA'.toLowerCase()</code>
<code>charAt(position)</code>	Devuelve el carácter de esa posición	<code>'hola'.charAt(2)</code>
<code>indexOf(char)</code>	Devuelve la posición en la que está el carácter indicado. Si no está devuelve -1, y si está varias veces su primera aparición	<code>'mañana'.indexOf('a')</code>
<code>lastIndexOf(char)</code>	Devuelve la última posición en la que se encuentra el carácter. Si no está devuelve -1	<code>'mañana'.lastIndexOf('a')</code>

7. Algunos Métodos

■ PARA EL TIPO STRING

Método	Descripción	Ejemplo
<code>substring(inicio,fin)</code>	Devuelve un trozo del string	<code>'hola'.substring(1,3)</code>
<code>concat(" ")</code>	Concatena cadenas de caracteres	<code>'pinta'.concat('labios')</code>
<code>parseInt(string)</code>	Convierte string a number. El string se interpreta como un entero. Por defecto en base 10	<code>parseInt('10') => 10</code> <code>parseInt('10.45') => 10</code>
<code>parseFloat(string)</code>	Convierte string a number. String se interpreta como n° en coma flotante	<code>parseFloat("1e2") => 100</code> <code>parseFloat("1.9") => 1.9</code>

Propiedad	Descripción	Ejemplo
<code>length</code>	Devuelve el número de caracteres del string	<code>'hola'.length</code>

7. Algunos Métodos

■ PARA EL TIPO ARRAY

Método	Descripción	Ejemplo
join(separador)	Une los elementos del array para formar un string. Se usa un separador para unir los elementos de la cadena	<pre>var a = ["hola", "mundo"] var b = a.join(" ")</pre>
concat()	Concatena elementos de varios arrays	<pre>array1.concat(array2)</pre>
pop()	Suprime el último elemento del array y lo mete en la variable seleccionada	<pre>var a = [1, 2, 3] var b = a.pop()</pre>
push()	Agrega un elemento (o varios) al array	<pre>a.push(4)</pre>
shift()	suprime el primer elemento del array y lo mete en la variable seleccionada	<pre>var a = [1, 2, 3] var b = a.shift()</pre>
unshift()	Agrega un elemento (o varios) al principio del array	<pre>var a = [1, 2, 3] a.unshift(0)</pre>

7. Algunos Métodos

■ PARA EL TIPO ARRAY

Método	Descripción	Ejemplo
<code>reverse()</code>	Coloca los elementos de un array en su orden inverso	<code>a.reverse()</code>

Propiedad	Descripción	Ejemplo
<code>length</code>	Devuelve el número de elementos dentro del array	<code>a.length</code>

7. Algunos Métodos

■ PARA EL TIPO NUMBER

Método	Descripción	Ejemplo
isNaN()	Protege el código de valores no numéricos	var num1 = 0; var num2 = 0; if(isNaN(num1/num2)) {...} else {...}
toFixed(digitos)	Fija el número de decimales que tiene que resolver una operación y redondea si es necesario. Devuelve un string	var num1 = 4564.34567; num1.toFixed(2); // 4564.35
toString(base)	Convierte un número a string con la base indicada. Si no se pone, por defecto es base 10	(31).toString(2) => "11111" (31).toString(10) => "31"

7. Algunos Métodos

■ OBJETO Math

Métodos		Constantes
Math.random()	Math.floor()	Math.PI
Math.pow()	Math.round()	Math.E
Math.sqrt()	Math.abs()	Math.SQRT2
Math.min()	Math.log()	Math.LN2
Math.max()	Math.exp()	Math.LN10
Math.ceil()	Math.sin()	...

```

>> Math.round(Math.random()*100)
< 43

>> Math.pow(3,2)
< 9

>> Math.PI
< 3.141592653589793
  
```

7. Algunos Métodos

■ OBJETO **Audio**

Métodos	
play()	Reproduce el audio
pause()	Detiene el audio
load()	Recarga el audio

Propiedades	
currentTime	Coloca la reproducción en el segundo indicado

7. Algunos Métodos

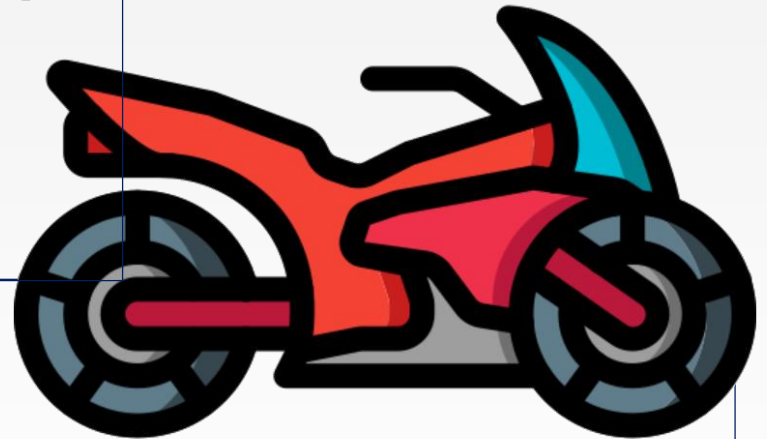
■ OBJETO Audio

■ Ejemplo

```
var sonido = new Audio("moto.mp3");

function playAudio() {
    sonido.play();
}

function pausarAudio() {
    sonido.pause();
}
```



```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="utf-8">
    <title>Ejemplo Audio</title>
    <script type="text/javascript" src="moto.js"></script>
  </head>
  <body>
    
  </body>
</html>
```

7. Algunos Métodos

■ OBJETO **Audio**

■ Webs de descarga de audios

- <https://www.sshhtt.com/>
- <https://www.freeaudiolibrary.com/es/>
- <https://www.elongsound.com/>
- <http://www.flashkit.com/soundfx/>
- <http://creativesounddesign.com/the-recordist-free-sound-effects/>
- <https://bigsoundbank.com/>
- <https://www.freesfx.co.uk//default.aspx>

7. Algunos Métodos

■ OBJETO Date

Métodos	
getDay()	Devuelve el día de la semana
getDate()	Devuelve el día del mes
getMonth()	Devuelve el mes
getFullYear()	Devuelve el año
getHours()	Devuelve la hora
getMinutes()	Devuelve los minutos
getSeconds()	Devuelve los segundos

```

>> var fecha = new Date()
>> fecha
< ▶ Date Mon Mar 22 2021 18:39:53
>> fecha.getDate()
< 22
>> fecha.getHours()
< 18
  
```


7. Algunos Métodos

■ OBJETO Date

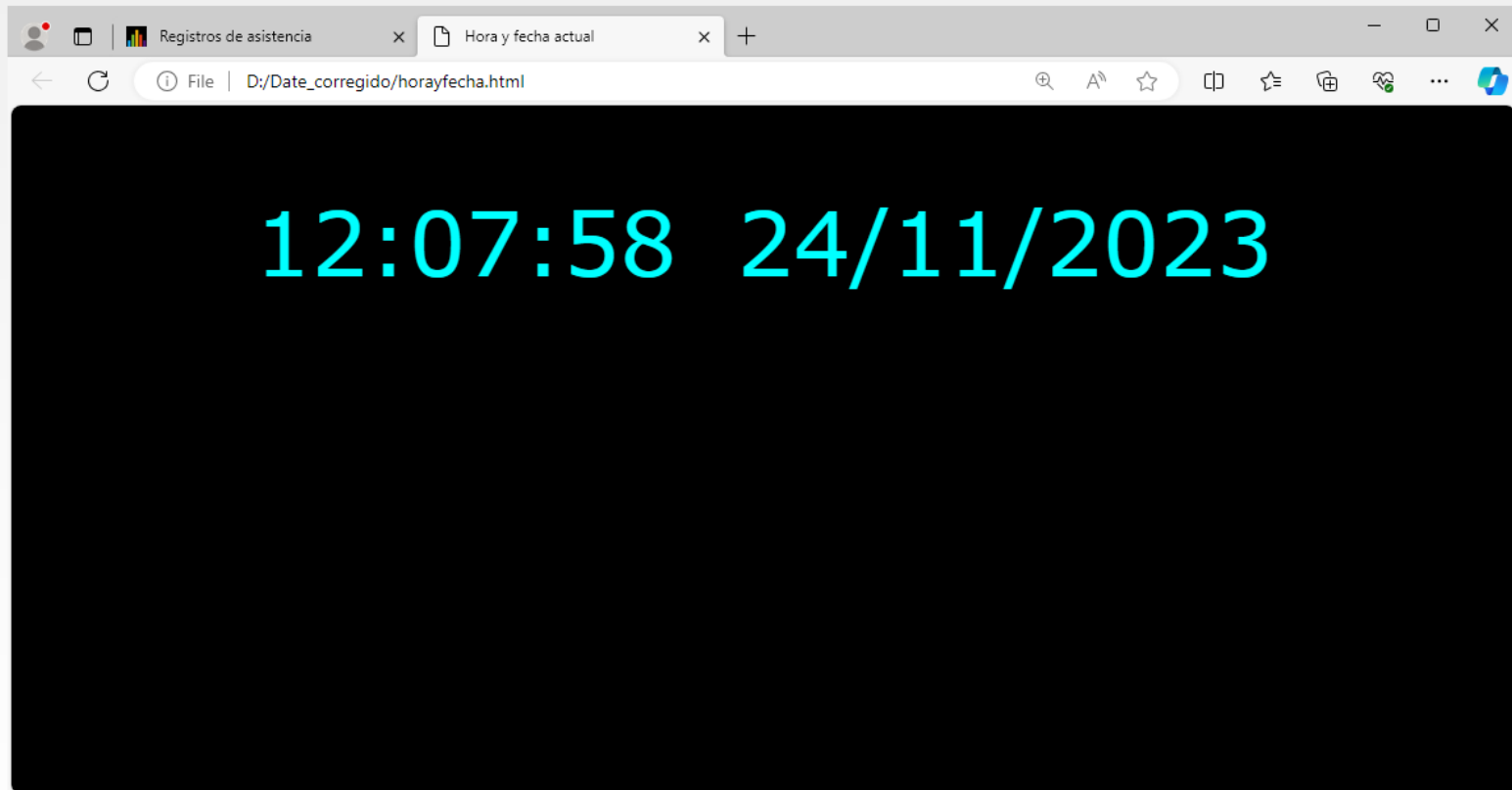
■ Ejemplo

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset = "utf-8">
    <title> Date </title>
  </head>
  <body>
    <h2> La fecha y hora de hoy son: </h2>

    <div id = "fecha"> </div>

    <script type="text/javascript">
      var fechaActual = document.getElementById("fecha");
      fechaActual.innerHTML = new Date();
    </script>
  </body>
</html>
```

Ejercicio Objeto Date



Ejercicio Objeto Date: Solución

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hora y fecha actual</title>
    <meta charset = "utf-8">
    <script src="date.js" type="text/javascript"></script>
    <style type="text/css">
      *{ background-color: black;}
      #fecha_hora{
        display: flex;
        justify-content: center;
        color: cyan;
        font-family: Verdana, Geneva, sans-serif;
        font-size: 3.5em;
      }
      #reloj, #fecha{padding: 10vh 2vw;}
    </style>
  </head>
  <body onload="main()" >
    <div id="fecha_hora">
      <div id="reloj"></div>
      <div id="fecha"></div>
    </div>
  </body>
</html>
```

Ejercicio Objeto Date: Solución

```
function main(){
    setInterval(tiempo, 1000);
}

function tiempo(){
    var fecha = new Date();
    var hora = fecha.getHours();
    var min = fecha.getMinutes();
    var seg = fecha.getSeconds();
    var dia = fecha.getDate();
    var mes = fecha.getMonth()+1;
    var year = fecha.getFullYear();

    if(seg < 10){seg = "0" + seg;}
    if(min < 10){min = "0" + min;}
    if(hora < 10){hora = "0" + hora;}
    if(dia < 10){dia = "0" + dia;}
    if(mes < 10){mes = "0" + mes;}

    document.getElementById('reloj').innerHTML = hora + ":" + min + ":" + seg;
    document.getElementById('fecha').innerHTML = dia + "/" + mes + "/" + year;
}
```

Atributos *async* y *defer*

■ Diferencia entre los atributos *async* y *defer*

■ **<script>**

- El análisis HTML se detiene, se descarga el archivo (si es un script externo), se ejecuta el script y después se reanuda el análisis HTML.

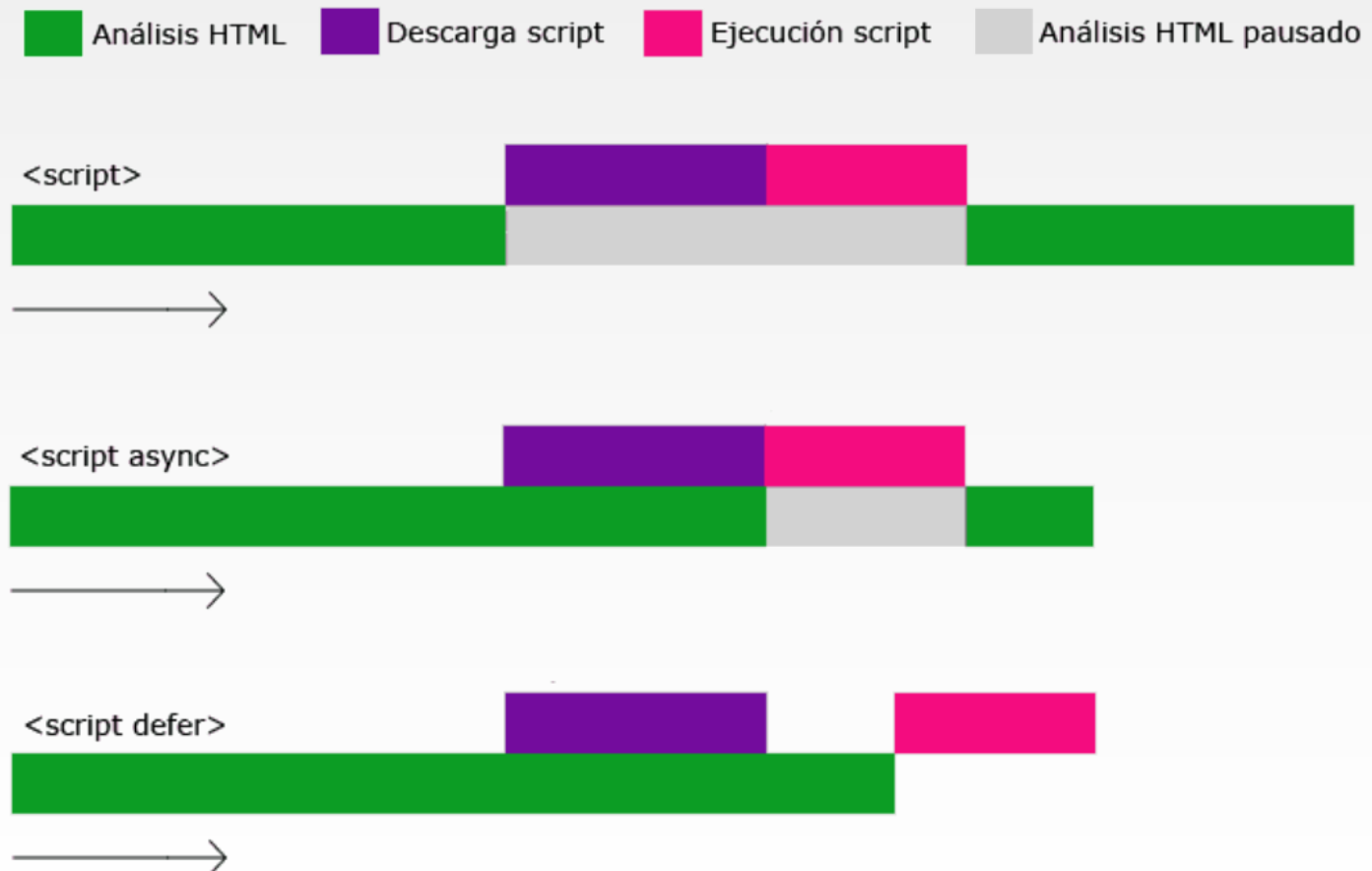
■ **<script async>**

- El script se descarga de forma asíncrona, es decir, sin detener el análisis HTML, pero una vez descargado, sí se detiene para ejecutar el script. Tras la ejecución se reanuda el análisis HTML. No se garantiza la ejecución de los scripts asíncronos en el mismo orden en el que aparecen en el documento.

■ **<script defer>**

- El script se descarga de forma asíncrona, en paralelo con el análisis HTML, y además su ejecución es diferida hasta que termine el análisis HTML. No hay bloqueo en el renderizado HTML. La ejecución de todos los scripts diferidos se realiza en el mismo orden en el que aparecen en el documento.

Atributos *async* y *defer*



Fuente: <https://cybmeta.com/diferencia-async-y-defer>

Formularios

8. Formularios

■ Introducción

- Sección de un documento que los usuarios ‘rellenan’ antes de enviarlo a un agente (servidor web, etc.) para su procesamiento.
- El procesamiento se lleva a cabo por programas ubicados en el servidor (PHP, Java, ASP, etc.).
- Objeto form: *document.forms*.
 - Cada formulario es un array de elementos.
 - Al cargar la página web el navegador crea automáticamente un array llamado ‘forms’ que contiene la referencia a todos los formularios de la página.
 - *document.forms[i].elements[j]* // $i=0\dots n$, $j=0,\dots m$

8. Formularios

■ Sintaxis

- Entre las etiquetas **<form>** y **</form>** se añaden tantos campos de entrada **<inputs>** como sea necesario.
 - <form> Elemento de bloque
 - <input> Elemento de línea

■ Ejemplo

```
<form>  
    <input type="text" id="nombre">  
</form>
```

8. Formularios

- Sintaxis

- Atributo *action*

- URL que va a ser llamada cuando se envíen los datos del formulario.

- Atributo *method*

- Método de envío al servidor. Dos valores *GET* (valor por defecto) y *POST*.
 - Los datos se envían como una lista de parámetros clave/valor (*name/value*).

- Ejemplo

```
<form method = "post" action = "web.php">
```

8. Formularios

■ Sintaxis

- Existen diferentes tipos de *inputs*:
 - Campos de texto
 - Radio Buttons
 - Checkboxes
 - Selectores
- Otros tipos incorporados en HTML5:
 - [W3Schools](#)
 - [W3C](#)
 - Los formularios se comprueban y validan de forma automática y nativa (no hace falta JavaScript).

8. Formularios

■ Elemento <input>

■ Atributo *type*

- Tipo o clase de información de entrada esperada por parte del usuario.
- Valor por defecto: texto.

Tipos		
text	tel	time
password	number	email
submit	range	radio
reset	date	checkbox
search	week	file
url	month	image

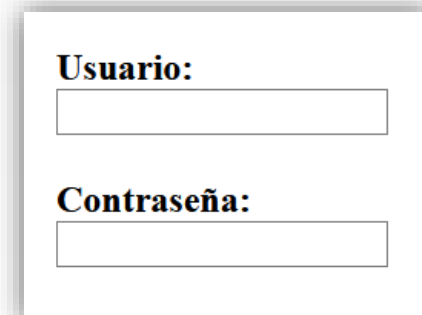
```
<input type="date" id="fecha"> <br>
```

```
<input type="search" placeholder="Escriba su búsqueda">
```

8. Formularios

- Inputs – Campos de texto
 - Para introducir textos cortos de sólo una línea.
 - Texto normal: **type = "text"**
 - Password: **type = "password"** (oculta los caracteres que se escriben).
 - Atributo *required*
 - Para hacer un campo obligatorio en el formulario.
 - Ejemplo

```
<form>  
  <b>Usuario:</b><br>  
  <input type="text" id="usuario"><br>  
  <b>Contraseña:</b><br>  
  <input type="password" id="passwd">  
</form>
```



8. Formularios

- Inputs – Campos de texto
 - Para introducir textos más largos o un área de texto.
 - Etiqueta **<textarea>**
 - Ejemplo

```
<b>Comentario:</b><br>  
<textarea rows="5" cols="50">  
    Escriba aquí su comentario  
</textarea>
```

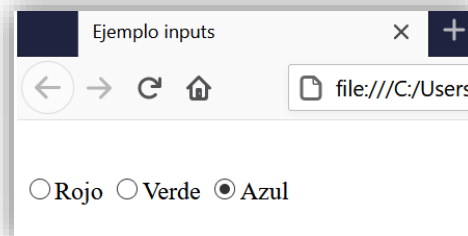
Comentario:

8. Formularios

- Inputs – *Radio buttons*
 - Para seleccionar entre varias opciones.
 - Sólo se puede seleccionar una.

- Ejemplo

```
<input type="radio" id="red"><i>Rojo</i>  
<input type="radio" id="green"><i>Verde</i>  
<input type="radio" id="blue"><i>Azul</i>
```



8. Formularios

■ Inputs – *Checkboxes*

- Para seleccionar entre varias opciones.
 - Se puede seleccionar más de una opción.

■ Ejemplo

```
<input type="checkbox" id = "check1" >  
  <i>Opción 1</i><br>  
<input type="checkbox" id = "check2" >  
  <i>Opción 2</i><br>
```

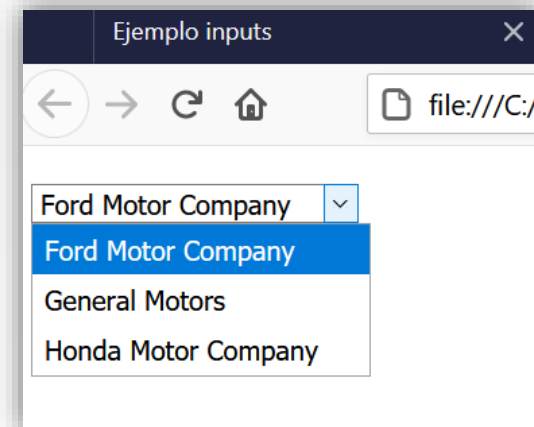
- ☐ Opción 1
☒ Opción 2

8. Formularios

■ Inputs – *Select*

- Desplegable con varias opciones.
- Ejemplo
 - Atributo *value*: valor que se enviará al servidor si se selecciona esa opción.

```
<select id = "company">
  <option value="ford">
    Ford Motor Company
  </option>
  <option value="generalm">
    General Motors
  </option>
  <option value="honda">
    Honda Motor Company
  </option>
</select>
```

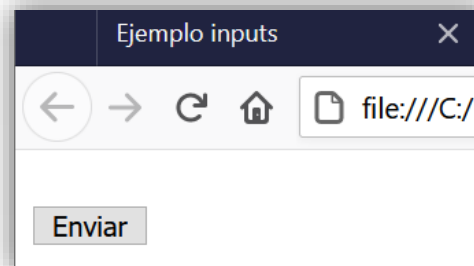


8. Formularios

- Inputs – Tipo *button*

- Input tipo botón que se puede utilizar para realizar una acción.
- Ejemplo

```
<input type="button" value="Enviar" onclick="test()">
```



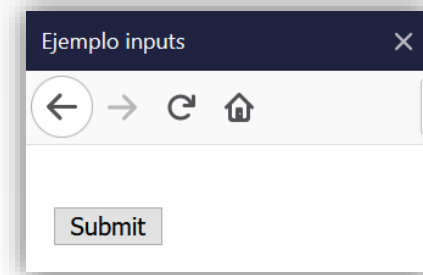
8. Formularios

■ Inputs – Tipo *submit*

- Envía la información del formulario a la página especificada por el atributo *action* del formulario o etiqueta `<form>`.

■ Ejemplo

```
<form name = "pruebaFormulario" action = "prueba.php">  
    ...  
    <input type="submit" value="Submit">  
</form>
```



8. Formularios

- Propiedad ***value***
 - Indica el valor actual del elemento, valor introducido en el formulario.
- Propiedad ***checked***
 - Informa sobre el estado del *checkbox* o *radio buttons*. Puede ser *true* o *false*.
- Más información
 - [Cómo obtener el valor de los campos del formulario](#)

8. Formularios

■ Ejercicio

Cálculo de Áreas

Introduzca la base y la altura del triángulo que quiera calcular el área, o bien el radio del círculo. Para introducir números decimales hay que utilizar el 'punto decimal', no la coma.

Base del triángulo :

Altura del triángulo:

Calcular ÁREA

El área del triángulo es:

Radio del círculo:

Calcular ÁREA

El área del círculo es:

Solución

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset = "utf-8" />
    <title>Ejemplo uso formularios</title>
    <link href = "estiloT.css" rel="stylesheet" type="text/css"/>
    <script type = "text/JavaScript" src = "areas.js"></script>
  </head>
  <body>
    <header>Cálculo de Áreas</header><hr>
    <p>Introduzca la base y la altura del triángulo que quiera calcular el área,
      o bien el radio del círculo. Para introducir números decimales hay que utilizar
      el 'punto decimal', no la coma.</p><hr>
    <main>
      <section>
        <form>
          Base del triángulo : <input size="4" id="base" placeholder="0.0"/><br>
          Altura del triángulo: <input size="4" id="altura" placeholder="0.0"/><hr>
          <input type="button" value = "Calcular ÁREA" onclick = "calculaAreaT()"><hr>
          El área del triángulo es: <input type="text" size="6" id = "resultado_T" readonly>
          <input type="reset" value="Borrar">
        </form>
      </section>
      <section>
        <form>
          Radio del círculo: <input size="4" id="radio" placeholder="0.0"/><hr>
          <input type="button" value = "Calcular ÁREA" onclick = "calculaAreaCirculo()"><hr>
          El área del círculo es: <input type="text" size="6" id = "resultado_C" disabled><hr>
          <input type="reset" value="Borrar">
        </form>
      </section>
    </main>
  </body>
</html>
```

Solución

```
// Cálculo del área de un triángulo
function calculaAreaT() {

    var base = document.getElementById("base").value;
    var altura = document.getElementById("altura").value;
    var resultado = parseFloat(base)*parseFloat(altura)/2;

    document.getElementById("resultado_T").value = resultado;
}

// Cálculo del área de un círculo
function calculaAreaCirculo(){

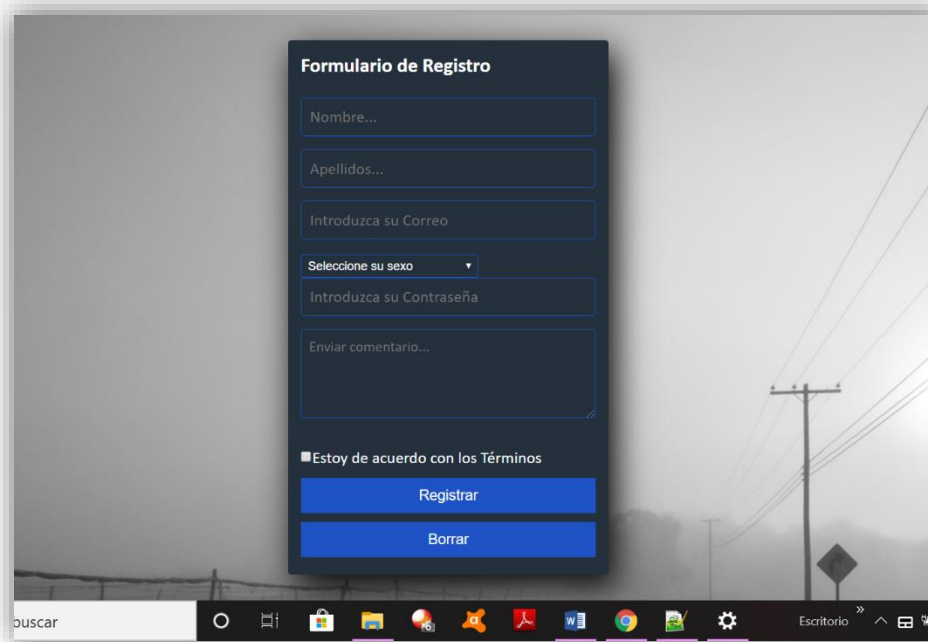
    var radio = document.getElementById("radio").value;
    var resultado = Math.PI*parseFloat(Math.pow(radio,2));

    document.getElementById("resultado_C").value = resultado;
}
```

8. Formularios

■ Ejercicio

- Crear el siguiente formulario de manera que devuelva una alerta en el caso de registrarse correctamente mostrando los datos introducidos. Si por el contrario se deja algún campo en blanco o si alguno de los datos introducidos no es correcto mostrará un mensaje de error. También sacará una alerta en caso de poner una contraseña con menos de seis caracteres, así como si no se aceptan los términos.



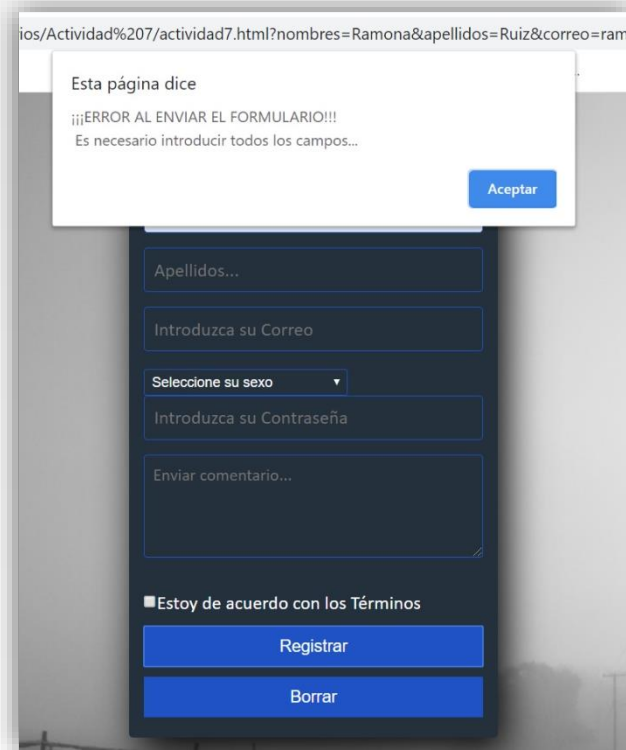
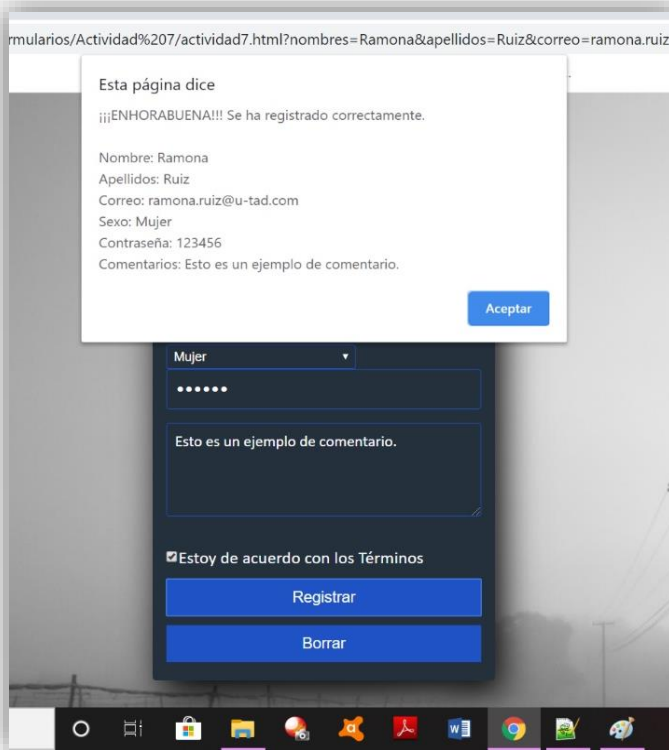
The image shows a web browser window with a registration form titled "Formulario de Registro". The form is displayed over a background image of a foggy street with power lines. The form fields include:

- Nombre...
- Apellidos...
- Introduzca su Correo
- Seleccione su sexo (dropdown menu)
- Introduzca su Contraseña
- Enviar comentario...

Below the form fields, there is a checkbox labeled "Estoy de acuerdo con los Términos". At the bottom of the form, there are two buttons: "Registrar" and "Borrar". The browser's taskbar is visible at the bottom, showing various application icons and the system clock.

8. Formularios

■ Ejercicio



Menús

*Menú dots 

*Menú hamburguer 

9. Menús

- Menús

- Existen numerosos tipos de menús, y numerosas formas distintas de realizarlos.
- Aquí se pueden encontrar diferentes ejemplos:
 - <https://www.w3schools.com/howto/default.asp>

9. Menús

- Menú *dots*



9. Menús

■ Menú *dots*

```
<!DOCTYPE html>
<html lang = "es">
  <head>
    <title>Menú dots</title>
    <link rel="stylesheet" href="menu_dots.css">
    <script src = "menu_dots.js"></script>
  </head>
  <body onload="inicio()">
    <header>
      
    </header>
    <nav style="text-align:center">
      <span class="dot" id="id1" onclick="currentImage(1)"></span>
      <span class="dot" id="id2" onclick="currentImage(2)"></span>
      <span class="dot" id="id3" onclick="currentImage(3)"></span>
      <span class="dot" id="id4" onclick="currentImage(4)"></span>
      <span class="dot" id="id5" onclick="currentImage(5)"></span>
    </nav>
  </body>
</html>
```

9. Menús

■ Menú *dots*

```
//MENÚ DOTS JAVASCRIPT
var index = 1;

function inicio(){
    document.getElementById('imagen').src = "images/number" +index+ ".png";
    document.getElementById('id'+index).style.backgroundColor='black';

    if(index>1){
        document.getElementById('id'+(index-1)).style.backgroundColor='cyan';
    }else{
        document.getElementById('id5').style.backgroundColor='cyan';
    }
    index++;

    if (index === 6 ) {
        index = 1;
    }
    setTimeout(inicio, 2000);

function currentImage(n){
    document.getElementById('imagen').src = "images/number"+n+".png";
    document.getElementById('id'+(index-1)).style.backgroundColor='cyan';
    document.getElementById('id'+n).style.backgroundColor='black';
    index = n;
}
```

```
/*MENÚ DOTS CSS*/

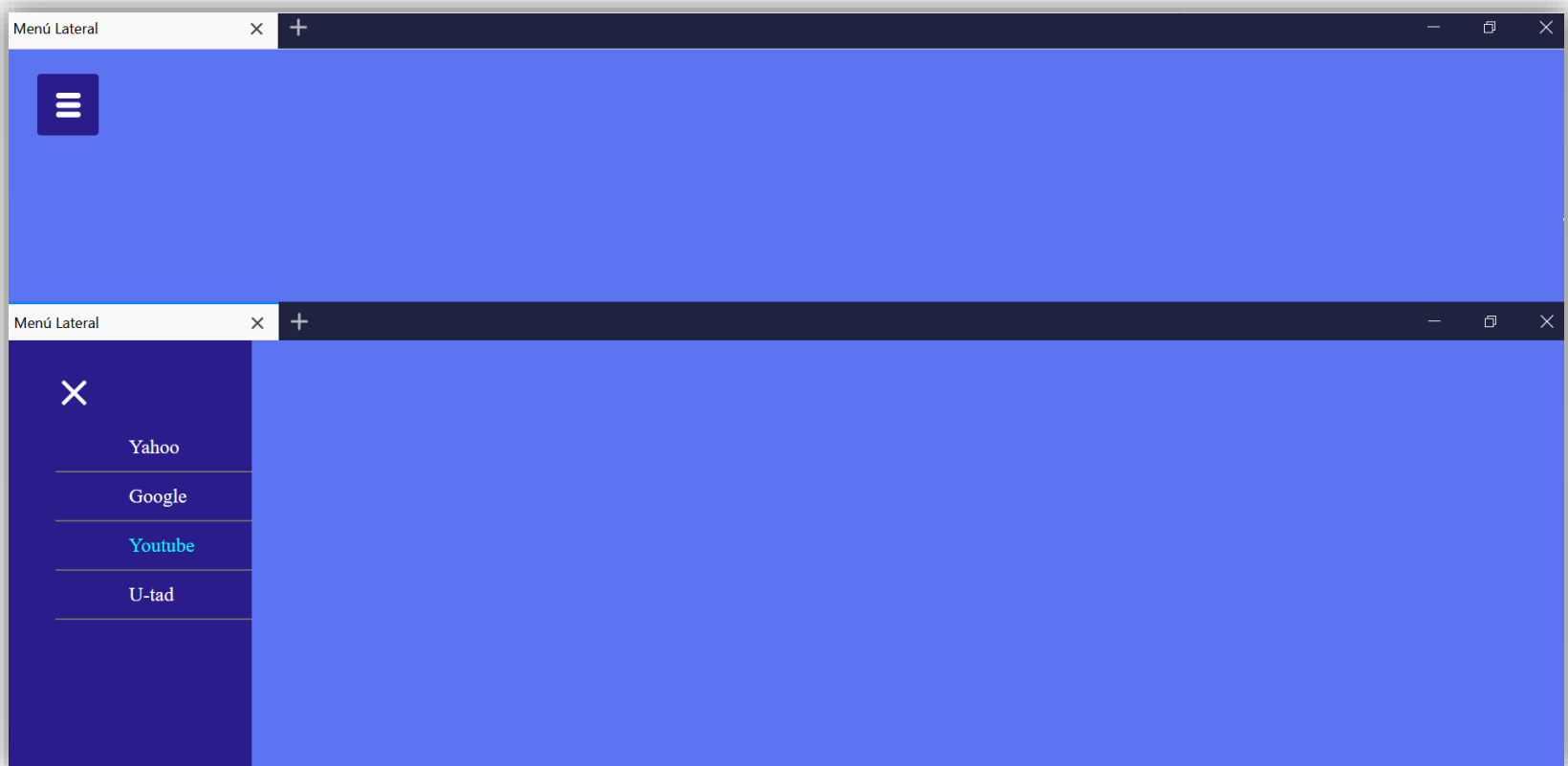
body{
    background-color: #3287FA;
    margin: 30px;
}

header{
    display: flex;
    justify-content: center;
    margin: 10px;
}

.dot {
    cursor: pointer;
    height: 15px;
    width: 15px;
    margin: 0 2px;
    background-color: cyan;
    border-radius: 50%;
    display: inline-block;
}
```

9. Menús

- Menú *hamburguer*



9. Menús

■ Menú *hamburguer*

```
<!DOCTYPE html>
<html>
  <head>
    <title>Menú Lateral</title>
    <link type= "text/css" rel="stylesheet" href="menu.css">
    <script type="text/javascript" src="menu.js" defer></script>
  </head>
  <body>
    <div id="sideNav">
      <nav>
        <ul>
          <li><a href="https://es.yahoo.com/" target="_blank">Yahoo</a></li>
          <li><a href="https://www.google.com/" target="_blank">Google</a></li>
          <li><a href="https://www.youtube.com/" target="_blank">Youtube</a></li>
          <li><a href="https://www.u-tad.com/" target="_blank">U-tad</a></li>
        </ul>
      </nav>
    </div>
    <div id="boton" onclick = "abrirMenu()">
      
    </div>
  </body>
</html>
```


9. Menús

■ Menú *hamburguer*

```
/* MENU LATERAL CSS*/
body{
    background-color: #5C73F2;
}
#sideNav{
    width: 200px;
    height: 100vh;
    position: fixed;
    left: -250px;
    top: 0;
    background: #2B1C8C;
    z-index: 1;
    transition: 0.5s;
}
ul{
    margin-top: 80px;
}
li{
    list-style: none;
    margin: 10px 60px;
}
a{
    text-decoration: none;
    color: white;
}

a:hover{
    color: cyan;
}
#boton{
    width: 50px;
    height: 50px;
    background: #2B1C8C;
    text-align: center;
    position: fixed;
    left: 30px;
    top: 20px;
    border-radius: 3px;
    z-index: 2;
    cursor: pointer;
    display: block;
}

img{
    width: 20px;
    margin-top: 15px;
}
```

9. Menús

- Menú
hamburguer

```
// MENÚ LATERAL JAVASCRIPT

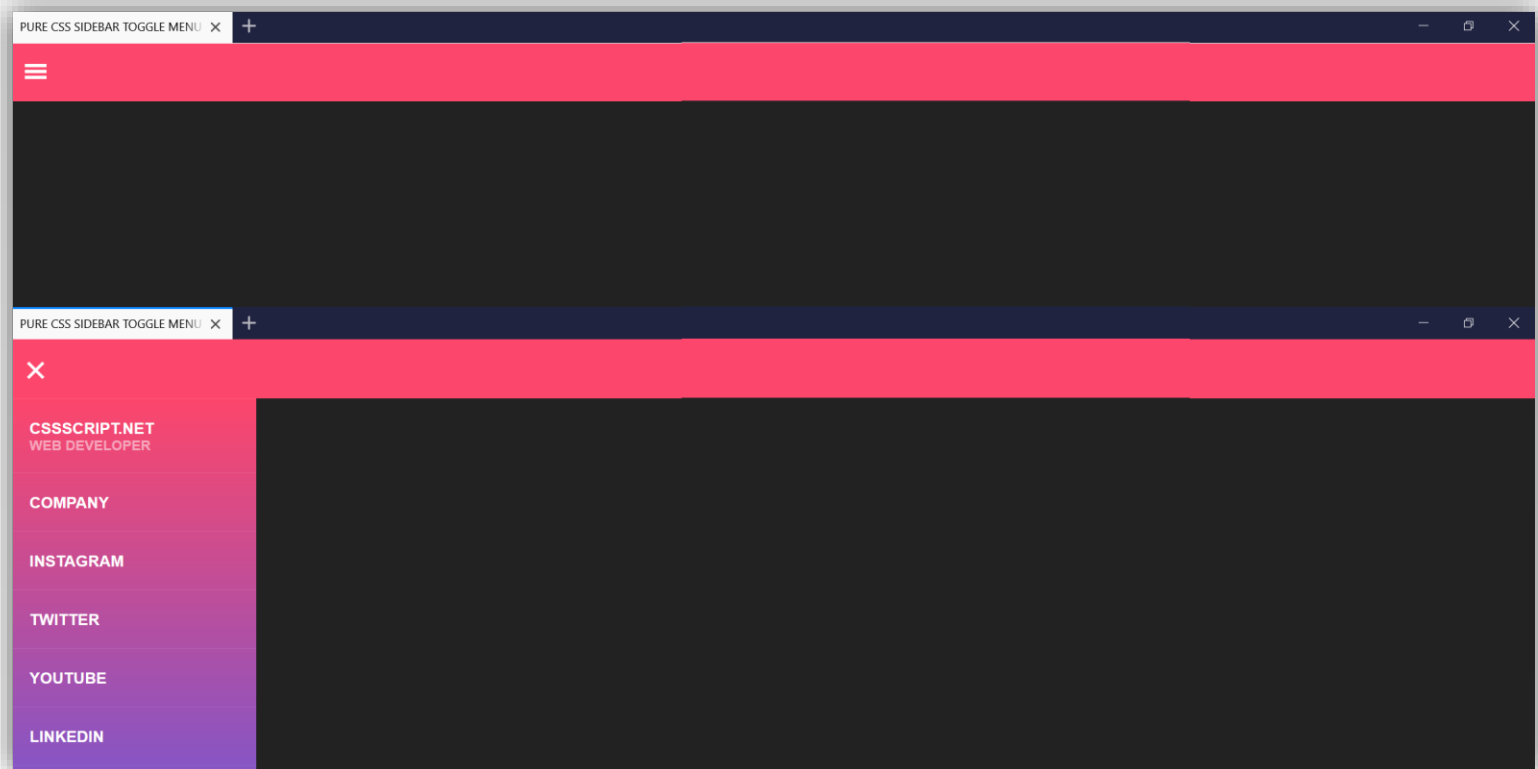
var boton = document.getElementById("boton");
var menu = document.getElementById("menu");
var sideNav = document.getElementById("sideNav");

sideNav.style.left = "-250px";

function abrirMenu() {
    if(sideNav.style.left === "-250px") {
        sideNav.style.left = "0";
        menu.src = "imagen/close.png";
    }
    else {
        sideNav.style.left = "-250px";
        menu.src = "imagen/menu.png";
    }
}
```

9. Menús

- Menú *hamburguer* (sólo con CSS)



Autora: [Jelena Jovanovic](#)

INTRODUCCIÓN A JAVASCRIPT

Grado en Ingeniería del Software

Asignatura: Fundamentos de Desarrollo Web

Curso: 2023 – 2024

ramona.ruiz@u-tad.com