enfocaT

fundación esplai

Javascript

SOC Servei d'Ocupació de Catalunya

Generalitat de Catalunya

Ministerio de Trabajo y Economía Social SEPE

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

¿Qué es javaScript?

- JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas.
- Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario.
- JavaScript es un lenguaje interpretado, por lo que no es necesario compilar los programas para ejecutarlos.
- JavaScript no tiene ninguna relación directa con Java. JavaScript es una marca registrada de la empresa Sun Microsystems.
- Se inserta dentro del código HTML

Historia

- A principios de los 90's la conexión eran con modems con velocidad de 28.8 kbps. → se comenzaban a desarrollar las primeras aplicaciones web (formularios)

- Se requería que la validación de formularios se realizara en el lado del cliente, mostrando los errores existentes.

- Brendan Eich (programador de Netscape) desarrolló el lenguaje LiveScript.

- Netscape firmó una alianza con Sun y cambió el nombre por el de JavaScript. → Nombre elegido por marketing, puesto que Java era la palabra de moda en el mundo informático.

¿Qué podemos hacer con javascript?

- Poner código en páginas HTML
- Reaccionar a eventos del cliente
  - Imágenes que cambian cuando pasamos el ratón por encima
  - Crear cookies
  - Validar formularios
  - Detectar el navegador y la resolución de la pantalla de un cliente (y cargarle una página)

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Cómo incluir JavaScript en documentos HTML

- **1. Incluir en el mismo documento HTML**
  - Se encierra entre etiquetas <script> y se incluye en cualquier parte del documento.
  - Pero se recomienda poner en la cabecera

```
<script type="text/javascript">
    alert("Un mensaje de prueba");
</script>
```

  - Es necesario añadir el atributo *Type* para que sean valores estándares.

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

- **2. Definir Javascript en un archivo externo**
  - Se enlazan mediante la etiqueta <script>
  - Cada documento html puede enlazar tantos archivos javascript como necesite.

- Archivo 'codigo.js':

```
alert("Un mensaje de prueba");
```

- Documento XHTML:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head>
      <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
      <title>Ejemplo de código JavaScript en el propio documento</title>
      <script type="text/javascript" src="/js/codigo.js">
      </script>
</head>
<body>
      <p>Un párrafo de texto.</p>
</body>
</html>
```

- También se requiere definir el atributo **src,** que indica la URL correspondiente al archivo javascript que se quiere enlazar.

- Cada etiqueta <script> solo puede enlazar un único archivo, pero se pueden incluir varias etiquetas <scripts> como se requieran.

- Los archivos de tipo javascript tienen extensión **.js**

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

- **3. Incluir javascript en los elementos HTML**
  - Permite incluir trozos de javascript dentro del codigo html de la página.

```
<html >
<head>
    <title>Ejemplo de código JavaScript</title>
</head>
<body>
    <p onclick="alert('Un mensaje de prueba')">Un párrafo de texto.</p>
</body>
</html>
```

- Este método solo se utiliza para definir algunos eventos.

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

- Javascript puede manipular los objetos creados en HTML (y los suyos):

Table 1-1	Creating and Working with Objects	
Object	*HTML Tag*	*JavaScript*
Web page	<BODY> . . . </BODY>	document
Image	<IMG NAME="myImage">	document.myImage
HTML form	<FORM name="myForm"> . . . </FORM>	document.myForm
Button	<INPUT TYPE="button" NAME="myButton">	document.myForm.myButton

- Dónde colocar los scripts:
  - En el <head> si queremos que estén disponibles en toda la página
  - En el <body> si queremos que se ejecuten cuando se carga la página
  - En un fichero externo para la reutilización de código: <script src="xxx.js"> </script>

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Sintaxis básica de Javascript

- No necesitamos terminadores de línea (bye ;)
- Comentarios : // Esto es un comentario
- Variables : var nombre; var nombre = “Pepe”
- Javascript es “case sensitive”: var pepe / var Pepe diferentes !

```
<!DOCTYPE html>
<html>
  <body>
    <h3>My favorite subject</h3>
    <p id="demo"></p>
    <script>
      var subject, Subject;
      subject = "Java";
      Subject = "Maths";
      document.getElementById("demo").innerHTML = subject;
    </script>
  </body>
</html>
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Declaración de variables.

En cuanto a las variables en JavaScript decir que no se les asigna un tipo predefinido. En JavaScript el tipo de las variables dependen del valor que contengan las mismas en cada momento.

Por tanto se realiza una conversión automática de tipos.

JavaScript reconoce los siguientes tipos de valores:

- **1. Números**: enteros y reales.
- **2. Valores booleanos**: true y false.
- **3. Strings.**
- **4. El valor null**.
- **5. Los objetos**: Creados por el programador o predefinidos por el lenguaje. Dado que no existen tipos de variables a priori, no hemos de especificar el tipo de variable cuando la declaramos.

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

- Escribir texto en pantalla
  Document.write("Esto sale en la propia página web")

- Crear ventanas de confirmación: confirm()

  Confirm("¿Está seguro de que quiere hacer esto?")

- Crear popups: alert()

  alert("Esta es una ventana de aviso")

La sentencia *if*.

La sentencia **if** tiene la forma:

if (*Condición*) Instrucción 1 o bloque de instrucciones; [**else** Instrucción 2 o bloque de instrucciones;]

Los paréntesis asociados que delimitan la condición no son opcionales.

{ Instrucción 1; Instrucción 2; ... Instrucción N; }

La sentencia *while*.

La sentencia **while** tiene la forma:

while (*Condición*) Instrucción o bloque de instrucciones.

Los paréntesis no son opcionales. Si se cumple la condición se ejecute la instrucción o el bloque de instrucciones y se repite el proceso.

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

La sentencia *for*.

En cuando a dicha sentencia, en JavaScript podemos distinguir dos variantes:

El bucle *for* "clásico".

- Este bucle, como a continuación podremos ver, tiene una sintaxis muy parecida a la de C/C++.

for (*[inicialización]; [condición]; [expresión]*) Instrucción o bloque de instrucciones;

En esta sintaxis:

Inicialización: Crea la variable contador y le da un valor inicial. Condición: lo que se debe cumplir para que el bucle se ejecute. Depende de la variable índice.

Actualización: Actualiza el valor de la variable índice.

El equivalente de esta expresión con while es:

inicialización; while (condición) { Instrucción 1; Instrucción 2; ... Instrucción N; expresión; }

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

El bucle *for/in*.

Esta estructura itera una variable *var* sobre todas las propiedades de un objeto *obj* que se le pasa. Así para cada valor de var se ejecutaran las sentencias del bucle. Por lo tanto, el bucle tendrá tantas iteraciones como propiedades el objeto y en cada iteración la variable tendrá el valor de la propiedad del objeto correspondiente con dicha iteración. Su sintaxis es:

for (*var* in obj) Instrucción o bloque de instrucciones;

La sentencia *break*.

La sentencia break se puede colocar dentro de un bucle o bucles anidados. Cuando se ejecuta la sentencia break se abandona el bucle más interno. A todos los efectos la sentencia break actúa como un salto a la instrucción siguiente al bucle en el que se ejecuta.

La sentencia *continue*.

La sentencia continue, no abandona el bucle si no hace que se ejecute la siguiente iteración. En el bucle while la ejecución del continue hace que el flujo del programa salte a la condición. En el bucle for la ejecución del continue hace que la expresión de incremento, para después continuar normalmente con la condición. Es decir, la ejecución del continue evita que se ejecute el resto del cuerpo del bucle.

La sentencia *switch*.

Hace que se seleccione un grupo de sentencias entre varias

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Definición de funciones.

La instrucción function permite la definición de funciones. Después de esta palabra reservada se coloca el nombre de la función seguido de una lista de argumentos delimitados por paréntesis y separados por comas.

```
function nombre (param1, param2,..., paramn){ instrucciones en JavaScript; }
```

La sentencia *return*.

La sentencia return es la que permite devolver el resultado de una función. En el ejemplo que se verá a continuación, se muestra una función que devuelve el área de un cuadrado de lado l.

```
function Area(lado){ return lado*lado; }
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Eventos

- Interacción del usuario con la página web (pasar el ratón, aceptar un formulario, etc…)
- Javascript permite ejecutar código cuando ocurra un evento:

```
<form method="post" action="xxx.htm“ onsubmit="return checkForm()">

<input type="text" size="30" id="email" onchange="checkEmail()">

<a href=“a.php" onmouseover="alert('An onMouseOver event');return false">
        <img src="w3schools.gif" width="100" height="30">
    </a>
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Objetos

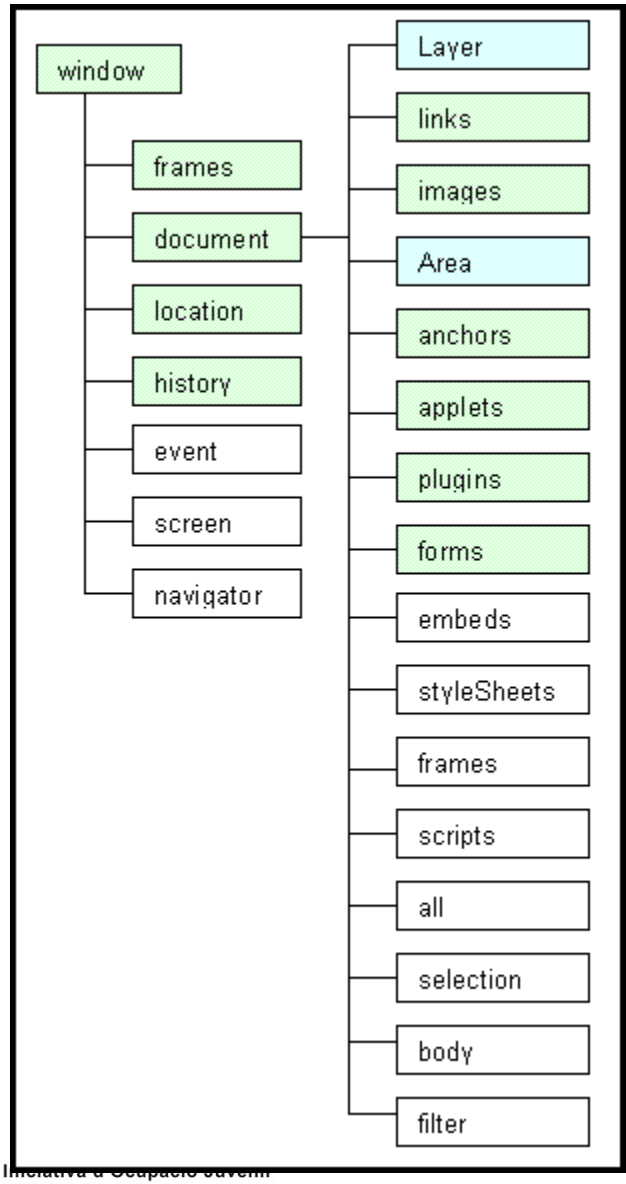
- Javascript es un lenguaje orientado a objetos
- Puede emplear tanto los objetos HTML como sus propios objetos (y algunos más del navegador como window, screen, location)

  var txt="HelloWorld!
  document.write(**txt.length**)

- Objetos HTML → DOM (DocumentObjectModel)
  document.body.style.background="#FFCC80

  DOM: Define la estructura lógica de los documentos y el modo en que se accede y manipula un documento

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur



- Para acceder a una propiedad de un objeto del modelo se utiliza la siguiente nomenclatura:

nombre_objeto.nombre_propiedad

- Para acceder a un método, se utiliza una sintáxis similar:

nombre_objeto.nombre_metodo()

- Entre paréntesis se le pasan al método los argumentos necesarios para su ejecución.

- Un objeto de JavaScript es, básicamente un *array*. Esto quiere decir que es posible acceder a las propiedades del objeto utilizando también la sintaxis siguiente:
- nombre_objeto["nombre_propiedad"]

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

// Define a variable called cssName and a message
// called resolutionInfo
Var cssName;
Var resolutionInfo;

// If the width of the screen is less than 650 pixels
if( screen.availWidth< 650 )
{
        // define the style Variable as the low-resolution style
        cssName= 'lowres.css';

        // Or if the width of the screen is less than 1000 pixels
}
else
{
        if( screen.availWidth> 1000 )
        {
                // define the style Variable as the high-resolution style
                cssName= 'highres.css';
                // Otherwise
                } else{
                // define the style Variable as the mid-resolution style
                cssName= 'lowres.css';
        }
}
document.write( '<link rel="StyleSheet" href="' +cssName+ '" type="text/css" />' );

</script>
```

Etiqueta noscript

- Es habitual que si la página requiere JavaScript para su correcto funcionamiento, se incluya un mensaje de aviso al usuario indicándole que debería activar javascript para disfrutar completamente de la página.

- El lenguaje HTML define la etiqueta **<noscript>** para mostrar un mensaje al usuario cuando su navegador no puede ejecutar JavaScript.

- El siguiente código muestra un ejemplo del uso de la etiqueta <noscript>:

```
<body>
<noscript>
    <p>Bienvenido a Mi Sitio</p>
    <p>La página que estás viendo requiere para su funcionamiento el uso de JavaScript. Si lo has deshabilitado intencionadamente, por favor vuelve a activarlo.</p>
</noscript>
</body>
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Primer script

```
<html>
<head>
<title>El primer script</title>
<script type="text/javascript">
    alert("Hola Mundo!");
</script>
</head>
<body>
    <p>Esta página contiene el primer script</p>
</body>
</html>
```

- La instrucción **alert**() permite mostrar un mensaje en la pantalla del usuario.

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur


El primer script - Microsoft Internet Explorer
Archivo
Edición
Ver
Favoritos
Herramientas
Vínculos
Microsoft Internet Explo...
Hola Mundo!
Aceptar
Intranet local

Ejercicio 1.

- Modificar el primer script para que:
  - 1. Todo el código JavaScript se encuentre en un archivo externo llamado codigo.js y el script siga funcionando de la misma manera.
  - 2. Después del primer mensaje, se debe mostrar otro mensaje que diga *"Soy el primer script"*
  - 3. Añadir algunos comentarios que expliquen el funcionamiento del código
  - 4. Añadir en la página HTML un mensaje de aviso para los navegadores que no tengan activado el soporte de JavaScript

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Programación básica

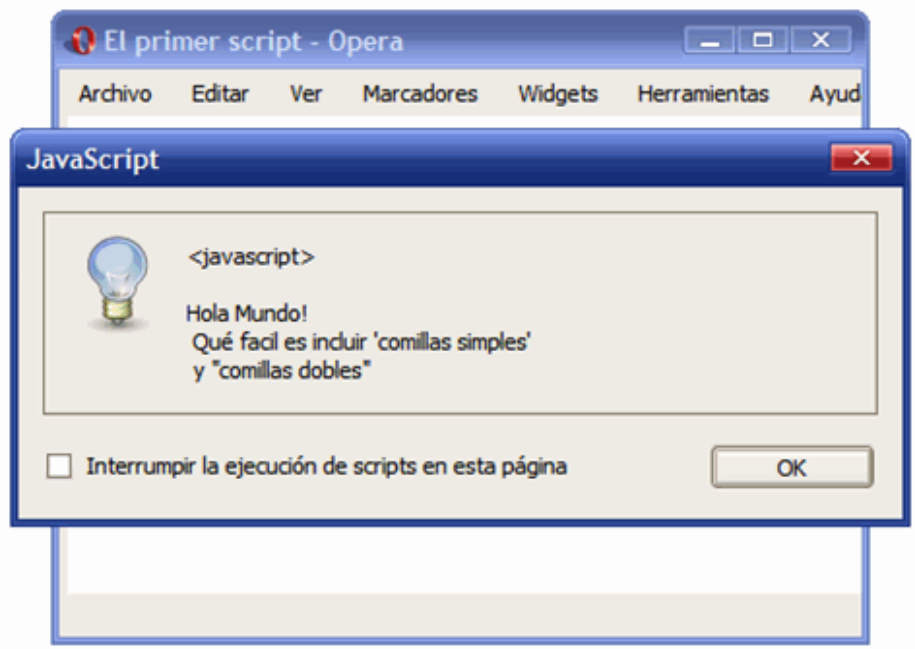
- Variables
  - Se crean mediante la palabra reservada **var** (aunque no es necesario declararlas)
  - Ejemplo:
    - var numero_1=3;
    - var numero_2=1;
    - Var resultado=numero_1+ numero_2;

- Solo puede tener letras, números, simbolos $, y _
- Tipos:
  - Numéricas
  - Cadenas de texto

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Ejercicio 2

- Modificar el primer script para que:
- 1. El mensaje que se muestra al usuario se almacene en una variable llamada mensaje y el funcionamiento del script sea el mismo.
- 2. El mensaje mostrado sea el de la siguiente imagen



Arrays

- var nombre_array = [valor1, valor2, ..., valorN];
- var dias = ["Lunes", "Martes", "Miércoles", "Jueves", "Viernes", "Sábado", "Domingo"];
- var diaSeleccionado = dias[0]; *// diaSeleccionado = "Lunes“*
- var otroDia = dias[5]; *// otroDia = "Sábado"*

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Ejercicio 3

- Crear un array llamado meses y que almacene el nombre de los doce meses del año.

- Mostrar por pantalla los doce nombres utilizando la función alert().

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Operadores

- Asignación
- Incremento y decremento
  - ++numero; numero++;
  - --numero; numero--;
- Lógicos
  - Negación
    - Var visible=true;
    - Alert(!visible); //muestra “false”
  - AND (&&)
  - OR (||)
- Aritméticos, +, -, *, /, %
  - numero1 +=3; //numero1=numero1+3=8
- Relacionales
  - >,<,>=,<=, ==,!=

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Estructuras de control de flujo

- If(condicion){
  }
- if(condicion) {
  ..
  }
  else {
  }

- for(inicializacion; condicion; actualizacion) {

...

}

- var dias = ["Lunes", "Martes", "Miércoles", "Jueves", "Viernes", "Sábado", "Domingo"];

- for(i in dias) {
  alert(dias[i]);
  }

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Funciones y propiedades básicas de javascript

- **Funciones para cadenas de texto**
  - **length.** Calcula la longitud de cadena de texto
    - `Var mensaje="hola mundo"`
    - `Numeroletras=mensaje.length`
  - **+**, se usa para concatenar cadenas de texto
  - **concat()**
    - `Mensaje=mensaje1.concat("holita");`
  - **toUpperCase().** Convierte a mayúsculas
  - **toLowerCase().** Convierte a minúsculas
  - **charAt(posicion).** Obtiene el carácter que se encuentra en la posicion indicada.
    - `Letra=mensaje.charAt(2);  //letra l`
  - **indexOf(carácter)**. Calcula la posición del caracter
    - `Posicion=mensaje.indexOf('a'); //posicion 3`
  - **substring(inicio, final).** Extrae una porción de la cadena de texto.
  - **split(separador)**. Convierte una cadena en un arreglo.
    - `Palabras=mensaje.split(" ");`
    - `//palabras={"hola", "mundo", "soy", "una", "cadena", "de", "texto"};`

- Join(separador)
  - Funcion contraria a split()
- Pop()
  - Elimina el último elemento de un arreglo y lo devuelve

```
Var array=[1,2,3]
var ultimo = array.pop();
// ahora array = [1, 2], ultimo = 3
```

- Push()
  - Añade un elemento al final del arreglo

```
array.push(3);
// ahora array = [1, 2, 3]
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

- shift(), elimina el primer elemento del array y lo devuelve. El array original se ve modificado y su longitud disminuida en 1 elemento.

```
var array = [1, 2, 3];
var primero = array.shift();
// ahora array = [2, 3], primero = 1
```

- unshift(), añade un elemento al principio del array. El array original se modifica y aumenta su longitud en 1 elemento. (También es posible añadir más de un elemento a la vez)

```
var array = [1, 2, 3];
array.unshift(0);
// ahora array = [0, 1, 2, 3]
```

- reverse(), modifica un array colocando sus elementos en el orden inverso a su posición original:

```
var array = [1, 2, 3];
array.reverse();
// ahora array = [3, 2, 1]
```

- Completar las condiciones de los if del siguiente script para que los mensajes de los alert() se muestren siempre de forma correcta:

- var numero1 = 5;
- var numero2 = 8;

- if(...) {
  - alert("numero1 no es mayor que numero2");
- }

- if(...) {
  - alert("numero2 es positivo");
- }

- if(...) {
  - alert("numero1 es negativo o distinto de cero");
- }

- if(...) {
  - alert("Incrementar en 1 unidad el valor de numero1 no lo hace mayor o igual
  - que numero2");
- }

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Funciones útiles para números

- NaN
  - Valor numérico no definido. Ejemplo la division entre 0

- isNan()
  - Permite proteger de posibles valores numéricos no definidos
  - var numero1 = 0;
  - var numero2 = 0;
  - if(isNaN(numero1/numero2)) {
    - alert("La división no está definida para los números indicados");
  - }
  - else {
    - alert("La división es igual a => " + numero1/numero2);
  - }

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Programación avanzada

- Funciones

```
function nombre_funcion() {
        ...
}

function suma_y_muestra() {
    resultado = numero1 + numero2;
    alert("El resultado es " + resultado);
}
var resultado;
var numero1 = 3;
var numero2 = 5;
suma_y_muestra();
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

```
// Definición de la función
function calculaPrecioTotal(precio) {
    var impuestos = 1.16;
    var gastosEnvio = 10;
    var precioTotal = ( precio * impuestos ) + gastosEnvio;
}

// Llamada a la función
calculaPrecioTotal(23.34);


function calculaPrecioTotal(precio, porcentajeImpuestos) {
    var gastosEnvio = 10;
    var precioConImpuestos = (1 + porcentajeImpuestos/100) * precio;
    var precioTotal = precioConImpuestos + gastosEnvio;
    return precioTotal.toFixed(2);;
}

var precioTotal = calculaPrecioTotal(23.34, 16);
var otroPrecioTotal = calculaPrecioTotal(15.20, 4);
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

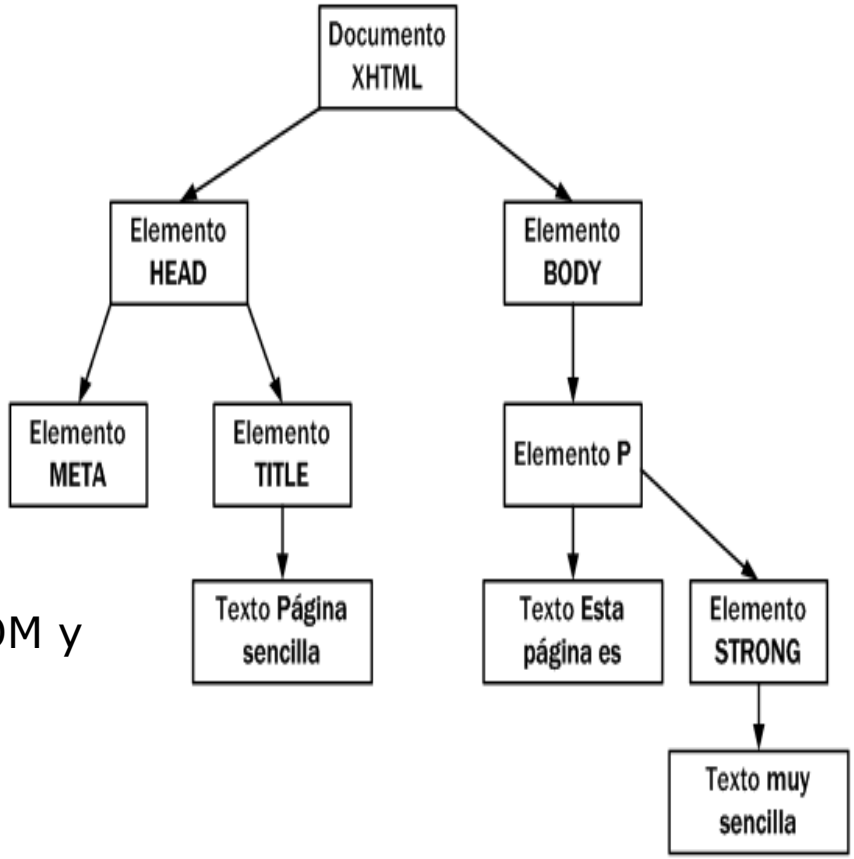
Tema: DOM

- Document Object Model o DOM.
- Permite acceder y manipular las páginas HTML. Acceder al valor almacenado por un elemento de html, crear elementos dinámicos, que aparezcan, que desaparezcan, etc.
- DOM transforma todos los documentos HTML en un conjunto de elementos llamados nodos, que están interconectados y que representan los contenidos de las páginas web y las relaciones entre ellos. → **Arbol de nodos**

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Ejemplo:

```
<html">
<head>
  <meta….>  </meta>
<title>Página sencilla</title>
</head>
<body>
<p>Esta página es <strong>muy
sencilla</strong></p>
</body>
</html>
```



Cara rectángulo representa un nodo DOM y las flechas indican las relaciones entre nodos.

La raíz del árbol de nodos siempre es la misma: un nodo de tipo especial denominado *"Documento"*.

- Cada etiqueta, se transforma en un nodo de tipo “Elemento”

- Así, la siguiente etiqueta HTML:
  - <title>Página sencilla</title>
  - Genera los siguientes dos nodos

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Tipos de nodos

- La especificación completa de DOM define 12 tipos de nodos, aunque las páginas HTML habituales se pueden manipular manejando solamente cuatro o cinco tipos de nodos:

  - **Document,** nodo raíz del que derivan todos los demás nodos del árbol.
  - **Element,** representa cada una de las etiquetas HTML. Se trata del único nodo que puede contener atributos y el único del que pueden derivar otros nodos.
  - **Attr,** se define un nodo de este tipo para representar cada uno de los atributos de las etiquetas HTML, es decir, uno por cada par atributo=valor.
  - **Text,** nodo que contiene el texto encerrado por una etiqueta HTML.
  - **Comment,** representa los comentarios incluidos en la página HTML.

- Otros tipos de nodos existentes son:
  - DocumentType, CDataSection,
  - DocumentFragment, Entity,
  - EntityReference, ProcessingInstruction y Notation

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

getElementsByTagName()

- Obtiene todos los elementos de la página HTML cuya etiqueta sea igual que el parámetro que se le pasa a la función.

- Ejemplo: Como obtener todos los párrafos de una página HTML

  var parrafos = document.getElementsByTagName("p");

  - Document: es el nombre de nodo a partir del cual se realiza la búsqueda de elementos.

  - El valor que devuelve la función es un arreglo con todos los nodos que cumplen la condición de que su etiqueta coincide con el parámetro proporcionado.

  - Se procesa de la siguiente manera:

  Var primerparrafo=parrafo[0];

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

- Para recorrer todos los párrafos de la página se utilizaría:

```
for(var i=0; i<parrafos.length; i++) {
    var parrafo = parrafos[i];
}
```

- La función getElementsByTagName() se puede aplicar de forma recursiva sobre cada uno de los nodos devueltos por la función. En el siguiente ejemplo, se obtienen todos los enlaces del primer párrafo de la página:

```
var parrafos = document.getElementsByTagName("p");
var primerParrafo = parrafos[0];
var enlaces = primerParrafo.getElementsByTagName("a");
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

getElementsByName()

- Se buscan los elementos cuyo atributo **name** sea igual al parámetro proporcionado.

- Ejemplo: Obtener el único párrafo con el nombre indicado.

```
var parrafoEspecial = document.getElementsByName("especial");
<p name="prueba">...</p>
<p name="especial">...</p>
<p>...</p>
```

getElementById()

- Se utilizan para desarrollo de aplicaciones web dinámicas.
- Se puede acceder directamente a un nodo y poder leer o modificar sus propiedades.
- getElementById() devuelve el elemento HTML cuyo atributo **id** coincide con el parámetro indicado en la función. Cada **id** es único para cada elemento de la página, por lo que la función devuelve solo el nodo deseado.

```
var cabecera = document.getElementById("cabecera");
```

```
<div id="cabecera">
    <a href="/" id="logo">...</a>
</div>
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Acceso directo a los atributos HTML

- Una vez que se ha accedido a un nodo, el siguiente paso natural consiste en acceder y/o modificar sus atributos y propiedades
- Simplemente se indica el nombre del atributo detrás del nombre del nodo.

```
var enlace = document.getElementById("enlace");
alert(enlace.href); // muestra http://www...com

<a id="enlace" href="http://www...com">Enlace</a>
```

- Las propiedades CSS no son tan fáciles de obtener como los atributos HTML. Para obtener el valor de cualquier propiedad CSS del nodo, se debe utilizar el atributo ***style***.

- El siguiente ejemplo obtiene el valor de la propiedad ***margin*** de la imagen:

```
var imagen = document.getElementById("imagen");
alert(imagen.style.margin);

<img id="imagen" style="margin:0; border:0;" src="logo.png" />
```

- Si el nombre de una propiedad CSS es compuesto, se accede a su valor modificando ligeramente su nombre:

```
var parrafo = document.getElementById("parrafo");
alert(parrafo.style.fontWeight); // muestra "bold"

<p id="parrafo" style="font-weight: bold;">...</p>
```

- La transformación del nombre de las propiedades CSS compuestas consiste en eliminar todos los guiones medios (-) y escribir en mayúscula la letra siguiente a cada guión medio.

- A continuación se muestran algunos ejemplos:
  - font-weight se transforma en fontWeight
  - line-height se transforma en lineHeight
  - border-top-style se transforma en borderTopStyle
  - list-style-image se transforma en listStyleImage

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Ejercicios sobre DOM

- 1. A partir de la página web proporcionada y utilizando las funciones DOM, mostrar por pantalla la siguiente información:
  - 1. Número de enlaces de la página
  - 2. Dirección a la que enlaza el penúltimo enlace
  - 3. Numero de enlaces que enlazan a http://prueba
  - 4. Número de enlaces del tercer párrafo

- 2. Completar el código JavaScript proporcionado para que cuando se de click sobre el enlace se muestre completo el contenido de texto.
- Además, el enlace debe dejar de mostrarse después de pulsarlo por primera vez.
- La acción de pinchar sobre un enlace forma parte de los "Eventos" de JavaScript que se ven en el siguiente capítulo. En este ejercicio, sólo se debe saber que al pinchar sobre el enlace, se ejecuta la función llamada muestra()

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Eventos

- Eventos definidos por 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>
onselect	Seleccionar un texto	<input>, <textarea>
onsubmit	Enviar el formulario	<form>
onunload	Se abandona la página (por ejemplo al cerrar el navegador)	<body>

- Los eventos más utilizados en las aplicaciones web tradicionales son:
  - **onload** para esperar a que se cargue la página por completo.
  - **onclick**, **onmouseover**, **onmouseout** para controlar el ratón
  - **onsubmit** para controlar el envío de los formularios.

Manejadores de eventos

- Las funciones o código JavaScript que se definen para cada evento se denominan *"manejador de eventos"* y como JavaScript es un lenguaje muy flexible, existen varias formas diferentes de indicar los manejadores:

  - Manejadores como atributos de los elementos HTML.
  - Manejadores como funciones JavaScript externas.
  - Manejadores *"semánticos"*

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

a) Manejadores de eventos como atributos

- Se trata del método más sencillo y a la vez *menos profesional* de indicar el código JavaScript que se debe ejecutar cuando se produzca un evento. En este caso, el código se incluye en un atributo del propio elemento HTML.
- En el siguiente ejemplo, se quiere mostrar un mensaje cuando el usuario pinche con el ratón sobre un botón:

```
<input type="button" value="Pinchame y verás"
onclick="alert('Gracias por pinchar');" />
```

- En este método, se definen atributos HTML con el mismo nombre que los eventos que se quieren manejar. El ejemplo anterior sólo quiere controlar el evento de pinchar con el ratón, cuyo nombre es **onclick**. Así, el elemento HTML para el que se quiere definir este evento, debe incluir un atributo llamado **onclick**.

- En este otro ejemplo, cuando el usuario pincha sobre el elemento se muestra un mensaje y cuando el usuario pasa el ratón por encima del elemento, se muestra otro mensaje:

```
<div onclick="alert('Has pinchado con el ratón');" onmouseover="alert('Acabas de pasar el ratón por encima');">
    Puedes pinchar sobre este elemento o simplemente pasar el ratón por encima
</div>
```

- Este otro ejemplo incluye una de las instrucciones más utilizadas en las aplicaciones JavaScript más antiguas:

```
<body onload="alert('La página se ha cargado completamente');">
  ...
</body>
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

b) Manejadores de eventos como funciones externas

- Siguiendo con el ejemplo anterior que muestra un mensaje al pinchar sobre un botón:

```
<input type="button" value="Pinchame y verás" onclick="alert('Gracias por
                pinchar');" />
```

- Utilizando funciones externas se puede transformar en:

```
function muestraMensaje() {
    alert('Gracias por pinchar');
    }

<input type="button" value="Pinchame y verás" onclick="muestraMensaje()" />
```

- Esta técnica consiste en extraer todas las instrucciones de JavaScript y agruparlas en una función externa. Una vez definida la función, en el atributo del elemento XHTML se incluye el nombre de la función, para indicar que es la función que se ejecuta cuando se produce el evento.

- La llamada a la función se realiza de la forma habitual, indicando su nombre seguido de los paréntesis y de forma opcional, incluyendo todos los argumentos y parámetros que se necesiten

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

- El principal inconveniente de este método es que en las funciones externas no se puede seguir utilizando la variable ***this*** y por tanto, es necesario pasar esta variable como parámetro a la función:

```
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;
                }
        }
```

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

    Sección de contenidos...

</div>
```

c) Manejadores de eventos semánticos

- Esta técnica es una evolución del método de las funciones externas, ya que se basa en utilizar las propiedades DOM de los elementos HTML para asignar todas las funciones externas que actúan de manejadores de eventos. Así, el siguiente ejemplo:

```
<input id="pinchable" type="button" value="Pinchame y verás"
                onclick="alert('Gracias por pinchar');" />
```

Se puede transformar en:

```
// Función externa
    function muestraMensaje() {
                alert('Gracias por pinchar');
        }
// Asignar la función externa al elemento
    document.getElementById("pinchable").onclick = muestraMensaje;

// Elemento XHTML
<input id="pinchable" type="button" value="Pinchame y verás" />
```

Formularios

- **Propiedades básicas de formularios y elementos**
- JavaScript dispone de numerosas propiedades y funciones que facilitan la programación de aplicaciones que manejan formularios.

- En primer lugar, cuando se carga una página web, el navegador crea automáticamente un array llamado forms y que contiene la referencia a todos los formularios de la página.

- Para acceder al array forms, se utiliza el objeto document, por lo que document.forms es el array que contiene todos los formularios de la página.

- Como se trata de un array, el acceso a cada formulario se realiza con la misma sintaxis de los arrays.
- La siguiente instrucción accede al primer formulario de la página:
  - document.forms[0];

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Array Elements

- Además del array de formularios, el navegador crea automáticamente un array llamado elements por cada uno de los formularios de la página.

- Cada array elements contiene la referencia a todos los elementos (cuadros de texto, botones, listas desplegables, etc.) de ese formulario. Utilizando la sintaxis de los arrays, la siguiente instrucción obtiene el primer elemento del primer formulario de la página:

  - document.forms[0].elements[0];

- La sintaxis de los arrays no siempre es tan concisa. El siguiente ejemplo muestra cómo obtener directamente el último elemento del primer formulario de la página:

  - document.forms[0].elements[document.forms[0].elements.length-1];

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

- Sin embargo, debería evitarse el acceso a los formularios de una página mediante el array document.forms.

- Una forma de evitar los problemas del método anterior consiste en acceder a los formularios de una página a través de su nombre (atributo name) o a través de su atributo id.

- El objeto document permite acceder directamente a cualquier formulario mediante su atributo name:

```
var formularioPrincipal = document.formulario;
var formularioSecundario = document.otro_formulario;

<form name="formulario" >
...
</form>

<form name="otro_formulario" >
...
</form>
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Accediendo a los elementos del formulario

- Los elementos de los formularios también se pueden acceder directamente mediante su atributo name:

```
var formularioPrincipal = document.formulario;
var primerElemento = document.formulario.elemento;

<form name="formulario">
    <input type="text" name="elemento" />
</form>
```

- También se puede acceder a los formularios y a sus elementos utilizando las funciones DOM de acceso directo a los nodos.
- El siguiente ejemplo utiliza la habitual función document.getElementById() para acceder de forma directa a un formulario y a uno de sus elementos:

```
var formularioPrincipal = document.getElementById("formulario");
var primerElemento = document.getElementById("elemento");

<form name="formulario" id="formulario" >
    <input type="text" name="elemento" id="elemento" />
</form>
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

- Independientemente del método utilizado para obtener la referencia a un elemento de formulario, cada elemento dispone de las siguientes propiedades útiles para el desarrollo de las aplicaciones:
  - type: indica el tipo de elemento que se trata. Para los elementos de tipo <input> (text, button, checkbox, etc.) coincide con el valor de su atributo type. Para las listas desplegables normales (elemento <select>) su valor es select-one, lo que permite diferenciarlas de las listas que permiten seleccionar varios elementos a la vez y cuyo tipo es select-multiple. Por último, en los elementos de tipo <textarea>, el valor de type es textarea.

  - form: es una referencia directa al formulario al que pertenece el elemento. Así, para acceder al formulario de un elemento, se puede utilizar document.getElementById("id_del_elemento").form

  - name: obtiene el valor del atributo name de XHTML. Solamente se puede leer su valor, por lo que no se puede modificar.

  - value: permite leer y modificar el valor del atributo value de XHTML. Para los campos de texto (<input type="text"> y <textarea>) obtiene el texto que ha escrito el usuario. Para los botones obtiene el texto que se muestra en el botón.

- Por último, los eventos más utilizados en el manejo de los formularios son los siguientes:
  - onclick: evento que se produce cuando se pincha con el ratón sobre un elemento.
  - onchange: evento que se produce cuando el usuario cambia el valor de un elemento de texto (<input type="text"> o <textarea>).
  - onfocus: evento que se produce cuando el usuario selecciona un elemento del formulario.
  - onblur: evento complementario de onfocus, ya que se produce cuando el usuario ha *deseleccionado* un elemento por haber seleccionado otro elemento delformulario. Técnicamente, se dice que el elemento anterior *"ha perdido el foco"*.

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

1. Obtener el valor de los campos de formulario

- 1. cuadro de texto y textarea

```
<input type="text" id="texto" />
var valor = document.getElementById("texto").value;

<textarea id="parrafo"></textarea>
var valor = document.getElementById("parrafo").value;
```

- 2. Radiobutton

```
<input type="radio" value="si" name="pregunta" id="pregunta_si"/> SI
<input type="radio" value="no" name="pregunta" id="pregunta_no"/> NO
<input type="radio" value="nsnc" name="pregunta" id="pregunta_nsnc"/> NS/NC
```

El siguiente código permite determinar si cada *radiobutton* ha sido seleccionado o no:

```
var elementos = document.getElementsByName("pregunta");
for(var i=0; i<elementos.length; i++) {
    alert(" Elemento: " + elementos[i].value + "\n Seleccionado: " +
    elementos[i].checked);
}
```

- 3. Las listas desplegables (<select>) son los elementos en los que es más difícil obtener su valor. Si se dispone de una lista desplegable como la siguiente:

```
<select id="opciones" name="opciones">
    <option value="1">Primer valor</option>
    <option value="2">Segundo valor</option>
    <option value="3">Tercer valor</option>
    <option value="4">Cuarto valor</option>
</select>
```

  - En general, lo que se requiere es obtener el valor del atributo value de la opción (<option>) seleccionada por el usuario.
  - Obtener este valor no es sencillo, ya que se deben realizar una serie de pasos. Además, para obtener el valor seleccionado, deben utilizarse las siguientes propiedades:
    - options, es un array creado automáticamente por el navegador para cada lista desplegable y que contiene la referencia a todas las opciones de esa lista.
    - De esta forma, la primera opción de una lista se puede obtener mediante document.getElementById("id_de_la_lista").options[0].
    - selectedIndex, cuando el usuario selecciona una opción, el navegador actualiza automáticamente el valor de esta propiedad, que guarda el índice de la opción seleccionada.

```
// Obtener la referencia a la lista
var lista = document.getElementById("opciones");

// Obtener el índice de la opción que se ha seleccionado
var indiceSeleccionado = lista.selectedIndex;

// Con el índice y el array "options", obtener la opción seleccionada
var opcionSeleccionada = lista.options[indiceSeleccionado];

// Obtener el valor y el texto de la opción seleccionada
var textoSeleccionado = opcionSeleccionada.text;
var valorSeleccionado = opcionSeleccionada.value;

alert("Opción seleccionada: " + textoSeleccionado + "\n Valor de la opción: " + valorSeleccionado);
```

No obstante, normalmente se abrevian todos los pasos necesarios en una única instrucción:

```
var lista = document.getElementById("opciones");

// Obtener el valor de la opción seleccionada
var valorSeleccionado = lista.options[lista.selectedIndex].value;
// Obtener el texto que muestra la opción seleccionada
var valorSeleccionado = lista.options[lista.selectedIndex].text;
```

Establecer el foco en un elemento

Se utiliza la función focus(). El siguiente ejemplo asigna el foco a un elemento de formulario cuyo atributo id es igual a primero:

```
document.getElementById("primero").focus();
```

```
<form id="formulario" action="#">
        <input type="text" id="primero" />
</form>
```

Ampliando el ejemplo anterior, se puede asignar automáticamente el foco del programa al primer elemento del primer formulario de la página, independientemente del id del formulario y de los elementos:

```
if(document.forms.length > 0) {
        for(var i=0; i < document.forms[0].elements.length; i++) {
                var campo = document.forms[0].elements[i];
                if(campo.type != "hidden") {
                        campo.focus();
                        break;
                }
        }
}
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Limitar el máximo caracteres

```
function limita(maximoCaracteres) {
    var elemento = document.getElementById("texto");
  if(elemento.value.length >= maximoCaracteres ) {
        return false;
  }
  else {
        return true;
        }
  }
```

```
<textarea id="texto" onkeypress="return limita(100);"></textarea>
```

Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Validar un campo de texto con valores numéricos

```
valor = document.getElementById("campo").value;

if( isNaN(valor) ) {
    return false;
}
```

enfocaT
fundaciónesplai
SOC Servei d'Ocupació de Catalunya
Generalitat de Catalunya
Ministerio de Trabajo y Economía Social SEPE
Iniciativa d'Ocupació Juvenil
Unió Europea
Fons Social Europeu
L'FSE inverteix en el teu futur

Validar que se ha seleccionado una opcion de una lista

```
indice = document.getElementById("opciones").selectedIndex;


if( indice == null || indice == 0 ) {
        return false;
}


<select id="opciones" name="opciones">
        <option value="">- Selecciona un valor -</option>
        <option value="1">Primer valor</option>
        <option value="2">Segundo valor</option>
        <option value="3">Tercer valor</option>
</select>
```

A partir de la propiedad selectedIndex, se comprueba si el índice de la opción seleccionada es válido y además es distinto de cero.

La primera opción de la lista (- Selecciona un valor –) no es válida, por lo que no se permite el valor 0 para esta propiedad selectedIndex.

Validar una fecha

```
    var ano = document.getElementById("ano").value;
    var mes = document.getElementById("mes").value;
    var dia = document.getElementById("dia").value;
    valor = new Date(ano, mes, dia);

if( !isNaN(valor) ) {
    return false;
}
```