

Tema 06: Introducción a JavaScript

**Lenguajes de Marcas y SGI
Administración de Sistemas Informáticos en Red
IES Los Manantiales**

¿Qué es JavaScript?

JavaScript es un lenguaje de programación incrustado en una pagina web que se ejecuta en el navegador web del cliente.

Es utilizado para crear páginas web dinámicas, incluyendo animaciones, validación de formularios, cálculos simples, acciones sobre botones, creación de menús, etc.

JavaScript es una marca registrada por Sun Microsystems.

¿Qué es JavaScript?

A principios de los 90, los usuarios se conectaban a Internet a velocidades muy bajas (28.8 kbps).

Surge la necesidad de desarrollar un lenguaje de programación que se ejecutara en el navegador del usuario. Así, si un usuario no correctamente un formulario, no era necesario volver a mandarlo a través de la red. La validación se realiza previamente en el navegador, mediante JavaScript, antes de enviarlo al servidor.

Aunque existen otros lenguajes de script, JavaScript es el más usado en la actualidad.

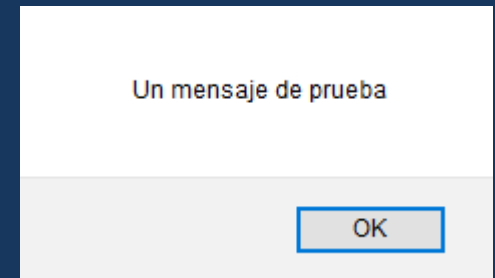
JavaScript no es Java

| Java | JavaScript |
|----------------------------------|-------------------------|
| Compilado | Interpretado |
| Necesita entorno de programación | Bloc de notas |
| Orientado a objetos | Orientado a eventos |
| Propósito general | Orientado a páginas web |
| Fuertemente tipado | Débilmente tipado |
| Potente, robusto y seguro | Sencillo y básico |
| Puede acceder a HD | No accede al HW |

Cómo incluir JavaScript en documentos HTML

1. Incluir JavaScript en el mismo documento HTML.

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <title>Ejemplo JavaScript en documento</title>
    <meta charset="utf-8">
    <script type="text/javascript">
      alert("Un mensaje de prueba");
    </script>
  </head>
  <body>
    <p>Un párrafo de texto.</p>
  </body>
</html>
```



Ejemplo

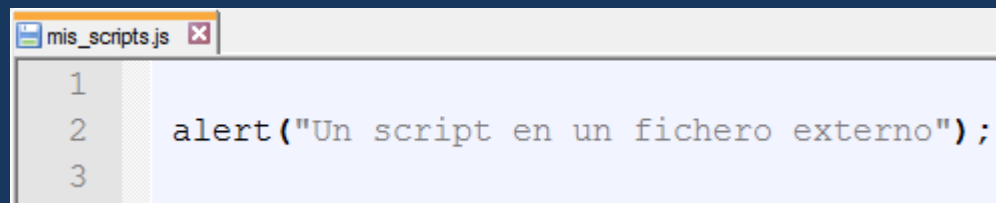
Cómo incluir JavaScript en documentos HTML

2. Definir JavaScript en un fichero externo

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <title>Ejemplo JavaScript en documento externo</title>
    <meta charset="utf-8">
    <script type="text/javascript" src="mis_scripts.js">
    </script>
  </head>

  <body>
    <p>Un párrafo de texto.</p>
  </body>
</html>
```

Ejemplo



The screenshot shows a code editor window titled 'mis_scripts.js'. The code inside is as follows:

```
1
2  alert("Un script en un fichero externo");
3
```

Un script en un fichero externo

OK

Cómo incluir JavaScript en documentos HTML

3. Incluir JavaScript en elementos HTML

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <title>Ejemplo JavaScript en párrafo</title>
    <meta charset="utf-8">
  </head>

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

Ejemplo

La etiqueta noscript

Se utiliza cuando el navegador no soporta JavaScript o cuando no lo tiene activado.

```
<noscript>
  Para utilizar las funcionalidades completas de este sitio
  es necesario tener JavaScript habilitado. Aquí están las
  <a href="http://www.enable-javascript.com/es/" target="_blank">
  instrucciones para habilitar JavaScript en tu navegador web</a>.
</noscript>
```

<http://www.enable-javascript.com/es/>

Ejemplo

Glosario básico

Script: cada uno de los programas, aplicaciones o trozos de código creados con JavaScript.

Sentencia: cada una de las instrucciones que forman el script.

Palabras reservadas: palabras que forman el lenguaje JavaScript, por lo que no pueden ser utilizadas libremente. Cada lenguaje de programación tiene las suyas:

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

Sintaxis

La sintaxis de un lenguaje se define como el conjunto de reglas que debemos seguir para escribir programas en ese lenguaje:

- No se tienen en cuenta espacios en blanco ni nuevas líneas.
- Sensible a mayúsculas.
- No se define el tipo de las variables.
- No es necesario acabar las sentencias con ; (recomendable)
- Se pueden incluir comentarios (// una línea o /*varias*/)

El primer Script.

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <title>El primer script</title>
    <meta charset="utf-8">
    <script type="text/javascript">
      alert("Hola Mundo");
    </script>
  </head>

  <body>
    <p>Esta página contiene el primer script.</p>
  </body>
</html>
```

Ejercicio 1.

Crea una página html con scripts para que:

1. Todo el código JavaScript se encuentre en un archivo externo llamado `codigo.js` y 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. Añade a la página Html un mensaje de aviso para los navegadores que no tengan activado el soporte de JavaScript.

Variables

- Qué es una variable?
- Declaración de variables (var, recomendable)
- Inicialización
- Nombres de variables (caracteres, números, \$ y _).
No puede empezar por un número.

```
resultado = numero_1 + numero_2;
```

```
var numero_1;  
var numero_2;
```

```
numero_1 = 3;  
numero_2 = 1;
```

Tipos de variables

- Numéricas
- Cadenas de caracteres
- Arrays
- Booleanos

Tipos de variables

Numéricas.

Aunque en JavaScript las variables no tienen un tipo definido, la forma en que asignamos valores depende del tipo de datos que contiene.

Las variables numéricas sirven para guardar valores enteros o decimales, con los que se realizarán operaciones matemáticas.

```
var iva = 16;           // variable tipo entero  
var total = 234.65;    // variable tipo decimal
```

Tipos de variables

Cadenas de caracteres.

Las cadenas se utilizan para guardar caracteres, palabras o frases de texto completas. Su valor se entre comillas dobles o simples.

```
var mensaje = "Bienvenido a nuestro sitio web";  
var nombreProducto = 'Producto ABC';  
var letraSeleccionada = 'c';
```


Tipos de variables

Cadenas de caracteres. Cadenas que contienen comillas

Para conseguir que una cadena contenga un trozo de texto entre comillas (como cuando hacemos una cita textual) alternaremos el uso de comillas simples o dobles.

```
/* El contenido de texto1 tiene comillas simples, por lo que  
se encierra con comillas dobles */  
var texto1 = "Una frase con 'comillas simples' dentro";  
  
/* El contenido de texto2 tiene comillas dobles, por lo que  
se encierra con comillas simples */  
var texto2 = 'Una frase con "comillas dobles" dentro';
```

Tipos de variables

Cadenas de caracteres. Otros problemas.

Si queremos incluir un carácter especial dentro de una cadena de caracteres utilizaremos los llamados caracteres de escape:

| Si se quiere incluir.. | Se debe incluir... |
|------------------------|--------------------|
| Una nueva línea | \n |
| Un tabulador | \t |
| Una comilla simple | \' |
| Una comilla doble | \" |
| Una barra inclinada | \\ |

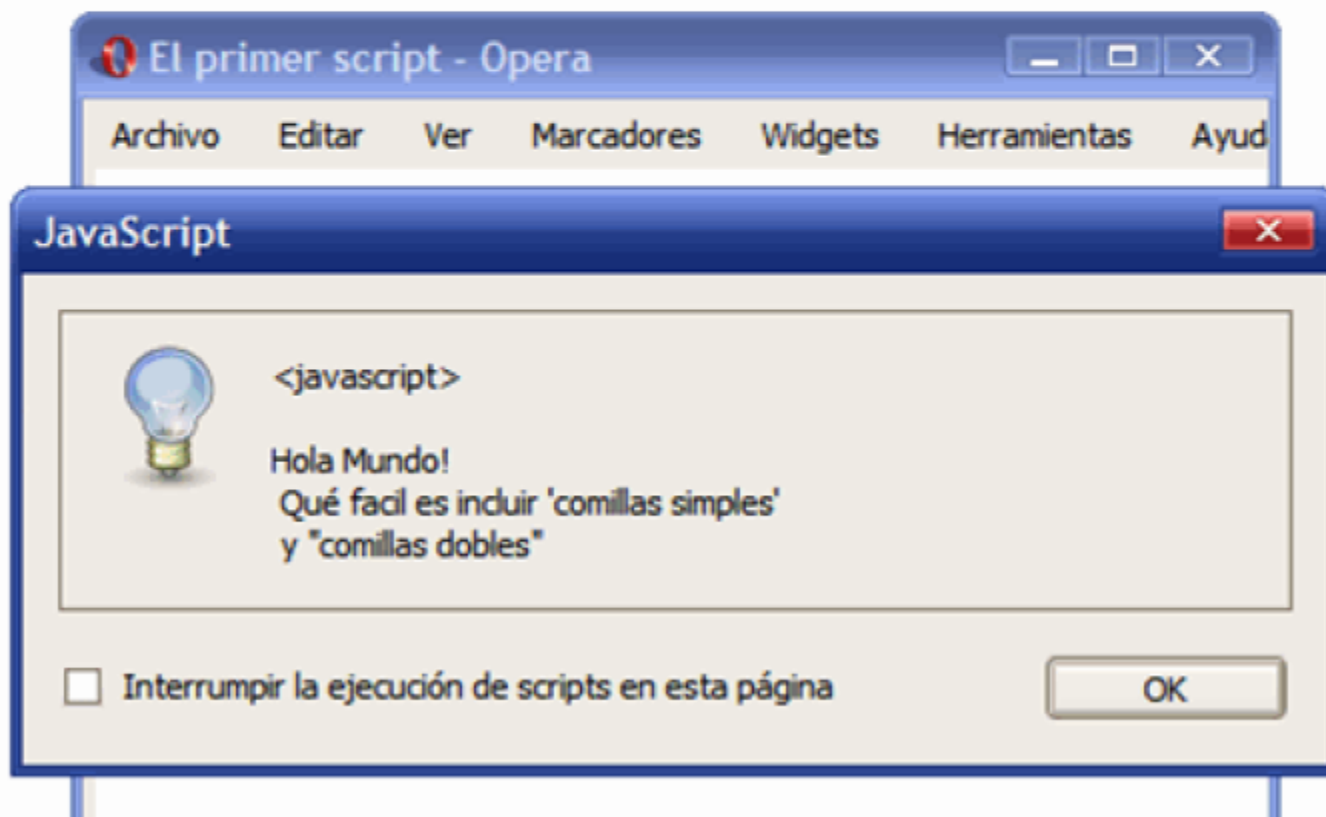
```
var texto1 = 'Una frase con \'comillas simples\' dentro';
```

```
var texto2 = "Una frase con \"comillas dobles\" dentro";
```

Ejercicio 2

Modificar el primer script del capítulo anterior 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:



Solución Ejercicio 2

```
<script type="text/javascript">  
    var mensaje ="Hola Mundo! \n Qué fácil es incluir 'comillas simples' \n y \"comillas dobles\".\";   
    alert(mensaje);  
</script>
```

Hola Mundo!
Qué fácil es incluir 'comillas simples'
y "comillas dobles".

Aceptar

Tipos de variables

Arrays.

Un array o vector es una colección de elementos del mismo tipo que se disponen de forma conjunta en memoria y a las que se accede a través de un índice.

Aunque en cualquier lenguaje de programación los elementos que se almacenan en una array deben ser del mismo tipo, JavaScript permite que no lo sean, por lo que podríamos almacenar elementos de diferente naturaleza.

Se utilizan cuando necesitamos guardar muchos datos similares, evitando así usar muchas variables.

Tipos de variables

Arrays.

```
var telefonos = Array(5);

telefonos[0]="666001120";
telefonos[1]="666001121";
telefonos[2]="666001122";
telefonos[3]="666001123";
telefonos[4]="666001124";

document.write(telefonos[i]);
document.write("</br>");
```

```
var alturas = Array(5);
```

```
alturas[0]=1.81;
alturas[1]=1.75;
alturas[2]=1.68;
alturas[3]=1.84;
alturas[4]=1.79;
```

```
for (i=0;i<5;i++)
{
    document.write(alturas[i]);
    document.write("</br>");
}
```

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

Tipos de variables

Booleanos.

Las variables de tipo boolean son un tipo especial que se utilizan para almacenar valores *true* o *false*.

Sirven para la evaluación de condiciones y no pueden contener otros valores.

```
var clienteRegistrado = false;  
var ivaIncluido = true;
```

Ejercicio 3

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

Muestra por pantalla los doce meses utilizando la función `document.write()` repetidamente, escribiendo cada mes en una línea diferente.

Ejercicio 3. Solución

```
<script type="text/javascript">
    var meses = ["Enero", "Febrero", "Marzo", "Abril", "Mayo", "Junio",
    "Julio", "Agosto", "Septiembre", "Octubre", "Noviembre", "Diciembre"];

    document.write(meses[0] + "<br>");
    document.write(meses[1] + "<br>");
    document.write(meses[2] + "<br>");
    document.write(meses[3] + "<br>");
    document.write(meses[4] + "<br>");
    document.write(meses[5] + "<br>");
    document.write(meses[6] + "<br>");
    document.write(meses[7] + "<br>");
    document.write(meses[8] + "<br>");
    document.write(meses[9] + "<br>");
    document.write(meses[10] + "<br>");
    document.write(meses[11] + "<br>");
</script>
```

Enero
Febrero
Marzo
Abril
Mayo
Junio
Julio
Agosto
Septiembre
Octubre
Noviembre
Diciembre

Ejercicio 03. Los meses del año

Operadores

Permiten manipular el valor de las variables, realizar operaciones matemáticas con sus valores o realizar comparaciones.

Asignación (=)

Se utiliza para guardar un valor específico en una variable.

```
var numero1 = 3;  
var numero2 = 4;
```

Operadores

Matemáticos (+,-,*, / y %)

Realizan operaciones matemáticas convencionales.

```
var numero1 = 10;
var numero2 = 5;

resultado = numero1 / numero2; // resultado = 2
resultado = 3 + numero1;       // resultado = 13
resultado = numero2 - 4;       // resultado = 1
resultado = numero1 * numero 2; // resultado = 50

var numero1 = 10;
var numero2 = 5;
resultado = numero1 % numero2; // resultado = 0

numero1 = 9;
numero2 = 5;
resultado = numero1 % numero2; // resultado = 4
```

Operadores

Incremento/Decremento (++/--)

Se utilizan en variables numéricas para incrementar y decrementar en una unidad su valor.

```
var numero = 5;  
++numero;  
alert(numero); // numero = 6
```



```
var numero = 5;  
numero = numero + 1;  
alert(numero); // numero = 6
```

```
var numero = 5;  
--numero;  
alert(numero); // numero = 4
```



```
var numero = 5;  
numero = numero - 1;  
alert(numero); // numero = 4
```

```
var numero = 5;  
numero++;  
alert(numero); // numero = 6
```

Operadores

Lógicos (!, && y ||)

Son imprescindibles cuando tenemos que tomar decisiones complejas que dependen de varias condiciones. Trabajan con variables booleanas.

Negación (!)

```
var visible = true;  
alert(!visible); // Muestra "false" y no "true"
```

```
var cantidad = 0;  
vacio = !cantidad; // vacio = true
```

Operadores

Lógicos (!, && y ||)

AND (&&)

Se tienen que cumplir las dos condiciones para que el resultado de AND sea verdadero.

```
var valor1 = true;  
var valor2 = false;  
resultado = valor1 && valor2; // resultado = false  
  
valor1 = true;  
valor2 = true;  
resultado = valor1 && valor2; // resultado = true
```

Operadores

Lógicos (!, && y ||)

OR(||)

Se tiene que cumplir al menos una de las condiciones para que el resultado de OR sea verdadero.

```
var valor1 = true;  
var valor2 = false;  
resultado = valor1 || valor2; // resultado = true  
  
valor1 = false;  
valor2 = false;  
resultado = valor1 || valor2; // resultado = false
```

Operadores

Matemáticos (+,-,*, / y %)

Operadores matemáticos combinados con asignación:

```
var numero1 = 5;  
numero1 += 3; // numero1 = numero1 + 3 = 8  
numero1 -= 1; // numero1 = numero1 - 1 = 4  
numero1 *= 2; // numero1 = numero1 * 2 = 10  
numero1 /= 5; // numero1 = numero1 / 5 = 1  
numero1 %= 4; // numero1 = numero1 % 4 = 1
```


Operadores

Relacionales (<, >, >=, <=, == y !=)

Devuelven valores booleanos.

```
var numero1 = 3;  
var numero2 = 5;  
resultado = numero1 > numero2; // resultado = false  
resultado = numero1 < numero2; // resultado = true  
  
numero1 = 5;  
numero2 = 5;  
resultado = numero1 >= numero2; // resultado = true  
resultado = numero1 <= numero2; // resultado = true  
resultado = numero1 == numero2; // resultado = true  
resultado = numero1 != numero2; // resultado = false
```

Operadores

Relacionales (<, >, >=, <=, == y !=)

Devuelven valores booleanos.

```
// El operador "=" asigna valores  
var numero1 = 5;  
resultado = numero1 = 3; // numero1 = 3 y resultado = 3  
  
// El operador "==" compara variables  
var numero1 = 5;  
resultado = numero1 == 3; // numero1 = 5 y resultado = false
```

Operadores

Relacionales (<, >, >=, <=, == y !=)

Comparando cadenas de caracteres.

```
var texto1 = "hola";  
var texto2 = "hola";  
var texto3 = "adios";
```

```
resultado = texto1 == texto3; // resultado = false  
resultado = texto1 != texto2; // resultado = false  
resultado = texto3 >= texto2; // resultado = false
```

Ejemplo de JavaScript.

```
<head>
  <script language=javascript> </script>
</head>

<body>
  <h2>Calcular la media de tres números</h2>
  <br>
  <form name="f1">
    <input type="text" name="n1" value="0" size="5"/><br>
    <input type="text" name="n2" value="0" size="5"/><br>
    <input type="text" name="n3" value="0" size="5"/><br><br>
    <input type="submit" value="Calcular Media" onclick="
      suma=parseInt(f1.n1.value)+parseInt(f1.n2.value)+parseInt(f1.n3.value) ;
      media=suma/3;
      alert('La media es.: ' + media);">
  </form>
</body>
```

Funciones básicas de JavaScript.

Funciones útiles para números.

```
var numero1 = 4564.34567;  
numero1.toFixed(2); // 4564.35  
numero1.toFixed(6); // 4564.345670  
numero1.toFixed(); // 4564
```

Ajusta decimales

Introducir datos desde teclado con prompt

```
<h1> Lee tres números y súmalos</h1>
<form name="f1">

    <input type="submit" value="Calcular" onclick="

var suma=0;
    n=prompt('Introduce número');
    suma=suma+parseInt(n);

    n=prompt('Introduce número');
    suma=suma+parseInt(n);

    n=prompt('Introduce número');
    suma=suma+parseInt(n);

    alert('El resultado es...' + suma) ">

</form>
```

Ejercicios. Operadores

Estructuras de control de flujo

Alteran el flujo secuencial de las instrucciones, decidiendo la siguiente instrucción atendiendo a una condición.

- Alternativas: dependen de una condición.
 - Estructura if
 - Estructura if...else
 - Switch
- Repetitivas: se repiten un número determinado de veces.
 - Estructura for
 - Estructura for...in
 - While
 - Do..While

Instrucciones alternativas. Estructura if.

```
if(condicion) {  
    ...  
}
```

```
var mostrarMensaje = true;  
  
if(mostrarMensaje == true) {  
    alert("Hola Mundo");  
}
```

```
var mostrarMensaje = true;  
  
if(mostrarMensaje) {  
    alert("Hola Mundo");  
}
```

```
if(!mostrado && usuarioPermiteMensajes) {  
    alert("Es la primera vez que se muestra el mensaje");  
}
```

Instrucciones alternativas. Estructura if..else

```
if(condicion) {  
    ...  
}  
else {
```

```
if(edad >= 18) {  
    alert("Eres mayor de edad");  
}  
else {  
    alert("Todavía eres menor de edad");  
}
```

Instrucciones alternativas. If..else if

```
if(edad < 12) {  
    alert("Todavía eres muy pequeño");  
}  
else if(edad < 19) {  
    alert("Eres un adolescente");  
}  
else if(edad < 35) {  
    alert("Aun sigues siendo joven");  
}  
else {  
    alert("Piensa en cuidarte un poco más");  
}
```

Instrucciones alternativas. Estructura Switch.

```
switch(variable) {  
    case valor_1:  
        ...  
        break;  
    case valor_2:  
        ...  
        break;  
    ...  
    case valor_n:  
        ...  
        break;  
    default:  
        ...  
        break;  
}
```

```
switch(numero) {  
    case 5:  
        ...  
        break;  
    case 8:  
        ...  
        break;  
    case 20:  
        ...  
        break;  
    default:  
        ...  
        break;  
}
```

Instrucciones repetitivas. Estructura for.

```
for(inicializacion; condicion; actualizacion) {  
    ...  
}
```

```
for(var i = 0; i < 5; i++) {  
    alert(mensaje);  
}
```

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

```
for(var i=0; i<7; i++) {  
    alert(dias[i]);  
}
```

Ejercicio 3 (versión con for)

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

```
    var meses = ["Enero", "Febrero", "Marzo", "Abril", "Mayo", "Junio",  
    "Julio", "Agosto", "Septiembre", "Octubre", "Noviembre", "Diciembre"];
```

```
    for (i=0;i<12;i++)
```

```
    {
```

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

```
    }
```

```
</script>
```

Enero
Febrero
Marzo
Abril
Mayo
Junio
Julio
Agosto
Septiembre
Octubre
Noviembre
Diciembre

Ejercicio 03. Los meses del año

Instrucciones repetitivas. Estructura while

```
while(condicion) {  
    ...  
}
```

```
var resultado = 0;  
var numero = 100;  
var i = 0;  
  
while(i <= numero) {  
    resultado += i;  
    i++;  
}  
  
alert(resultado);
```

Instrucciones repetitivas. Estructura do..while

```
do {  
    ...  
} while(condicion);
```

```
var resultado = 1;  
var numero = 5;  
  
do {  
    resultado *= numero;  
    numero--;  
} while(numero > 0);  
  
alert(resultado);
```


Ejercicios.

Funciones básicas de JavaScript.

Funciones útiles para cadenas de texto.

```
var mensaje = "Hola Mundo";  
var numeroLetras = mensaje.length; // numeroLetras = 10
```

Longitud de cadena

```
var mensaje1 = "Hola";  
var mensaje2 = " Mundo";  
var mensaje = mensaje1 + mensaje2; // mensaje = "HoLa Mundo"
```

Concatenación

```
var mensaje1 = "Hola";  
var mensaje2 = mensaje1.concat(" Mundo"); // mensaje2 = "HoLa Mundo"
```

Concatenación con una variable

Funciones básicas de JavaScript.

Funciones útiles para cadenas de texto.

```
var mensaje1 = "Hola";  
var mensaje2 = "Mundo";  
var mensaje = mensaje1 + " " + mensaje2; // mensaje = "HoLa Mundo"
```

Concatenación

```
var mensaje1 = "Hola";  
var mensaje2 = mensaje1.toUpperCase(); // mensaje2 = "HOLA"
```

Mayúsculas

```
var mensaje1 = "HoLa";  
var mensaje2 = mensaje1.toLowerCase(); // mensaje2 = "hoLa"
```

Minúsculas

Funciones básicas de JavaScript.

Funciones útiles para números.

```
var numero1 = 0;  
var numero2 = 0;  
alert(numero1/numero2); // se muestra el valor NaN
```

Indica "Not a Number"

```
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);  
}
```

Permite proteger el programa de posibles fallos.

Funciones en JavaScript

Cuando se realizan aplicaciones complejas, es muy habitual utilizar muchas veces las mismas instrucciones.

Cuando las instrucciones se repiten muchas veces el diseño se complica ya que:

- código muy largo difícil de controlar
- alta probabilidad de errores
- si hay que modificar esas instrucciones que se repiten varias veces, habrá que realizar muchos cambios.

Divide y vencerás!!

Definición de Función

Una función es un conjunto de instrucciones que se agrupan y que realizan una tarea concreta, pudiendo ser utilizada todas las veces que queramos.

Las funciones tienen un nombre y se ejecutan al ser llamadas desde el programa.

Pueden recibir parámetros que serán utilizados para realizar cálculos, operaciones, etc.

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

Parámetros (Argumentos)

```
// Definición de la función  
function suma_y_muestra(primerNumero, segundoNumero) {  
    var resultado = primerNumero + segundoNumero;  
    alert("El resultado es " + resultado);  
}
```

```
// Declaración de las variables  
var numero1 = 3;  
var numero2 = 5;
```

```
// Llamada a la función  
suma_y_muestra(numero1, numero2);
```

Valores de retorno

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

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


Ámbito de las Variables

El ámbito de una variable es la zona del programa en la que se define la variable y donde se podrá utilizar.

Variables locales y globales.

```
function creaMensaje() {  
    var mensaje = "Mensaje de prueba";  
}  
creaMensaje();  
alert(mensaje);
```

Si eliminamos la palabra var de la definición de la variable, esta se convertirá en variable global.

Ejemplo

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

<body>
  <h2>Galería de Funciones</h2>
  <br>
  <form name="f1">
    Base.....: <input type="text" name="n1" value="" size="5"/><br>
    Exponente...:<input type="text" name="n2" value="" size="5"/><br>
    .....
    <input type="submit" value="Calcular Potencia" onclick="

base=parseInt(f1.n1.value);
exponente=parseInt(f1.n2.value);
resultado=potencia(base,exponente);
alert('La potencia es..: ' + resultado);
```

```
function potencia(b,e) {
  var valor=1;
  if (e!=0) {
    for(i=0;i<e;i++) {
      valor=valor*b;
    }
  }
  return valor;}

```

Ejercicios

Eventos en JavaScript

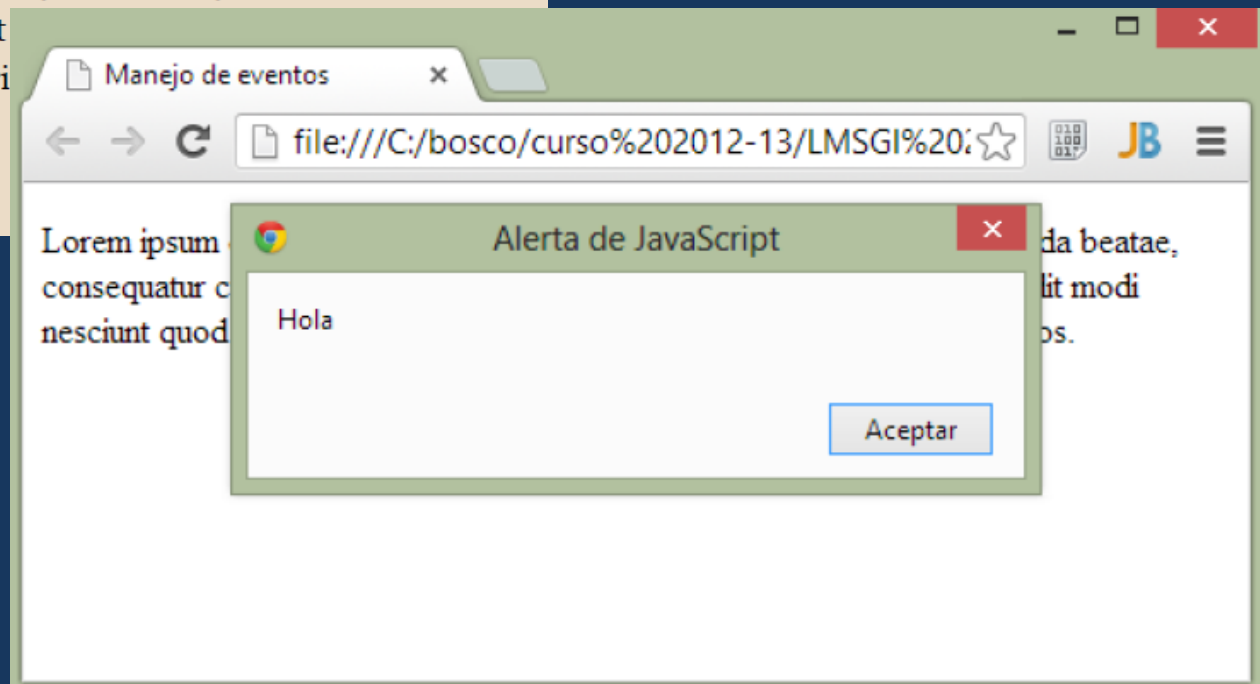
El lenguaje HTML nos permite asociar código JavaScript a diferentes comportamientos del usuario. De esta forma, cuando el usuario realiza una acción, podemos ejecutar una función JavaScript que realizará una determinada tarea. A este comportamiento se le llama Evento.

Existen muchos eventos, como pueden ser pinchar o mover el ratón, pulsar una tecla, cerrar una ventana, etc.

Aunque existen muchos tipos, nosotros sólo vamos a conocer los eventos que pertenecen al modelo básico, ya que son compatibles con todos los navegadores.

Ejemplo de Evento

```
<!doctype html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Manejo de eventos</title>
</head>
<body>
<p onclick="alert('Hola');">Lorem ipsum dolor sit amet, consectetur
elit.
  Aperiam assumenda beatae, consequatur cumque debitis
  delectus et explicabo, fugiat fugit
  nesciunt quod quos reprehenderit
  Dignissimos.</p>
</body>
</html>
```



Tipos de eventos

provocados por el uso del ratón

| evento | ocurre cuando... |
|--------------------------|--|
| <code>onclick</code> | el usuario hace clic en el elemento |
| <code>ondblclick</code> | el usuario hace doble clic en el elemento |
| <code>onmousedown</code> | el usuario pulsa el ratón sobre el elemento. Al soltar se produce <code>onmouseup</code> |
| <code>onmouseup</code> | el usuario suelta el ratón tras haberle pulsado (<code>onmousedown</code>) |
| <code>onmouseover</code> | el usuario mueve el ratón sobre el elemento |
| <code>onmouseout</code> | el usuario mueve el ratón hacia fuera del elemento |

provocados por el teclado

| evento | ocurre cuando... |
|-------------------------|--|
| <code>onkeypress</code> | el usuario pulsa una tecla |
| <code>onkeydown</code> | el usuario está pulsando una tecla |
| <code>onkeyup</code> | el usuario suelta la tecla que estaba pulsando |

Tipos de eventos

provocados por la acción en el propio navegador

| evento | ocurre cuando... |
|-----------------|---|
| onabort | se cancela la carga del elemento |
| onerror | si la carga del elemento no se ha hecho correctamente |
| onload | cuando el elemento se ha terminado de cargar |
| onresize | el tamaño del documento cambia (porque el usuario modifica el tamaño de la ventana) |
| onscroll | cuando el usuario se desplaza por el elemento |
| onunload | cuando se abandona la carga de la página (porque el usuario se va a otra o cierra la ventana) |

Tipos de eventos

eventos de formulario

Presentes sólo en los controles de formulario

| evento | ocurre cuando... |
|-----------------|---|
| onblur | cuando el control del formulario pierde el foco (el usuario abandona el cuadro de texto, casilla de verificación,...) |
| onchange | se ha modificado el contenido del cuadro |
| onfocus | el control del formulario obtiene el foco |
| onreset | se restablece (mediante botón <i>reset</i>) el contenido del formulario |
| onselect | cuando se selecciona texto dentro del control del formulario |
| onsubmit | el usuario pulsa el botón de enviar |

Ejemplos de Eventos

Manejadores de eventos

Un manejador de evento es una función o código JavaScript que se ejecuta al producirse dicho evento.

Existen varias formas de indicar los manejadores:

- a) Manejadores de eventos como atributos HTML.
- b) Manejadores de eventos y variable this.
- c) Manejadores de eventos como funciones externas.

a.- Manejadores de eventos atributos HTML

El código a ejecutar se indica en un atributo del elemento que provoca el evento.

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

```
<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>
```

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

```
...  
</body>
```

b.- Manejadores de eventos y variable this.

La variable `this` sirve para hacer referencia al elemento que ha provocado el evento. En el siguiente ejemplo vemos un evento que cambia el color del borde al pasar por encima.

```
<div id="contenidos" style="width:150px; height:60px; border:thin solid silver"
onmouseover="document.getElementById('contenidos').style.borderColor='black';"
onmouseout="document.getElementById('contenidos').style.borderColor='silver';">
    Sección de contenidos...
</div>
```

```
<div id="contenidos" style="width:150px; height:60px; border:thin solid silver"
onmouseover="this.style.borderColor='black';"
onmouseout="this.style.borderColor='silver';">
    Sección de contenidos...
</div>
```

c.- Manejadores de eventos como funciones externas.

Se definen funciones independientes que serán llamadas al desencadenar el evento. No pueden usar la variable `this`, por lo que debemos pasar el elemento como parámetro.

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

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

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

Ejercicio con Eventos

Crea una página web en la que se incluyan, en los elementos que creas conveniente, los siguientes eventos:

- onclick
- ondblclick
- onchange
- onkeypress
- onfocus
- onmouseover
- onmouseout
- onresize
- onload
- onunload

Una vez terminado el ejercicio deberás publicarlo en tu dominio.