


DESARROLLO WEB EN ENTORNO CLIENTE





CAPÍTULO 2: Introducción al lenguaje JavaScript

Juan Manuel Vara Mesa
Marcos López Sanz
David Granada
Emanuel Irrazábal
Jesús Javier Jiménez Hernández
Jenifer Verde Marín


Características de JavaScript

- ¿Qué es JavaScript?
 - Lenguaje de programación interpretado utilizado fundamentalmente para dotar de comportamiento dinámico a las páginas web.
 - Cualquier navegador web actual incorpora un intérprete para código JavaScript. 

Características de JavaScript

- Su sintaxis se asemeja a la de C++ y Java. 
- Sus objetos utilizan herencia basada en prototipos. 
- Es un lenguaje débilmente tipado. 
- Todas sus variables son globales. 

“Hola mundo” con JavaScript

- La forma más inmediata de empezar a experimentar con JavaScript es escribir secuencias de comandos simples.
- Es necesario sólo un navegador web y un editor de texto. 

También se puede utilizar: BlueFish, Aptana Estudio, JavaScript Utility Suite, Macromedia DreamWeaver, Sublime Text2, NetBeans. Otros IDE's JavaScript:-->



“Hola mundo” con JavaScript

- Hay dos formas de embeber el código JavaScript en una página HTML:

1. Incluirlo directamente en la página HTML mediante la etiqueta <script>.

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv= "content-type" content="text/html;
      charset=utf-8">
    <title>Hola Mundo</title>
  </head>
  <body>
    <script>
      alert('Hola mundo en JavaScript')
    </script>
  </body>
</html>
```

Recordamos
error en alert
(`'Hola mundo en
JavaScript'`);

“Hola mundo” con JavaScript

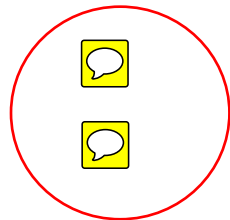
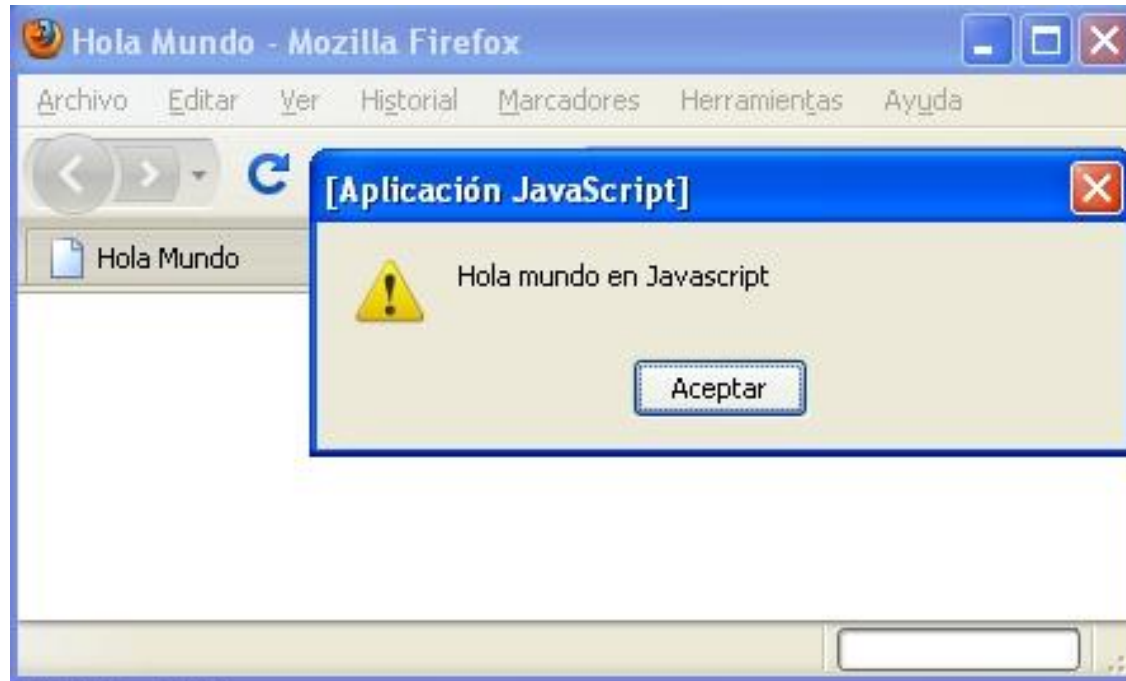
2. Utilizar el atributo `src` de la etiqueta `<script>` para especificar el fichero que contiene el código JavaScript.



```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="content-type" content="text/html;
      charset=utf-8">
    <title>Hola Mundo</title>
    <script type="text/javascript" src="HolaMundo.js">
    </script>
  </head>
  <body></body>
</html>
```

El fichero `HolaMundo.js` debe contener:
`alert('Hola Mundo en JavaScript')`

“Hola mundo” con JavaScript




El lenguaje JavaScript: sintaxis

- La **sintaxis** de JavaScript es muy similar a la de Java o C++.
- Especifica aspectos como:
 - Definición de comentarios.
 - Nombre de las variables.
 - Separación entre las diferentes instrucciones del código.
 - Etc.


Doble barra // y
/* barra y
asterisco
finalizado en */

El lenguaje JavaScript: sintaxis

- Mayúsculas y minúsculas:
 - El lenguaje distingue entre mayúsculas y minúsculas, a diferencia de por ejemplo HTML.
 - No es lo mismo utilizar `alert()` que `Alert()`. 

El lenguaje JavaScript: sintaxis

■ Comentarios en el código:

- Los comentarios no se interpretan por el navegador.
- Existen dos formas de insertar comentarios:
 - Doble barra (//) – Se comenta una sola línea de código.
 - Barra y asterisco (/ * al inicio y */ al final) – Se comentan varias líneas de código. 



El lenguaje JavaScript: sintaxis

- Comentarios en el código – Ejemplo:

```
<script type="text/javascript">  
    // Este modo permite comentar una sola línea  
    /* Este modo permite realizar  
    comentarios de  
    varias líneas */  
</script>
```



El lenguaje JavaScript: sintaxis

- **Tabulación y saltos de línea:**
 - JavaScript ignora los espacios, las tabulaciones y los saltos de línea con algunas excepciones. 📌
 - Emplear la tabulación y los saltos de línea mejora la presentación y la legibilidad del código.

El lenguaje JavaScript: sintaxis



■ Tabulación y saltos de línea – Diferencias:




```
<script
type="text/javascript">va
r i,j=0;
for (i=0;i<5;i++){
alert("Variable i: "+i);
for (j=0;j<5;j++){ if
(i%2==0){
document.write
(i + "-" + j +
"<br>");}}}</script>
```

```
<script type="text/javascript">
var i,j=0;
for (i=0;i<5;i++){
  alert("Variable i: "+i;
  for (j=0;j<5;j++){
    if (i%2==0){
      document.write(i + "-" + j + "<br>");
    }
  }
}
</script>
```

Falta cerrar
parentesis)

El lenguaje JavaScript: **sintaxis**

- **El punto y coma:**
 - Se suele insertar un signo de punto y coma (;) al final de cada instrucción de JavaScript.
 - Su utilidad es separar y diferenciar cada instrucción.
 - Se puede omitir si cada instrucción se encuentra en una línea independiente (la omisión del punto y coma no es una buena práctica de programación). 

El lenguaje JavaScript: sintaxis

■ Palabras reservadas:

- Algunas palabras no se pueden utilizar para definir nombres de variables, funciones o etiquetas.
- Es aconsejable no utilizar tampoco las palabras reservadas para futuras versiones de JavaScript.
- En www.ecmascript.org es posible consultar todas las palabras reservadas de JavaScript.





Tipos de datos

- Los tipos de datos especifican qué tipo de valor se guardará en una determinada variable.
- Los tres tipos de datos primitivos de JavaScript son:
 - Números.
 - Cadenas de texto.
 - Valores booleanos.


Tipos de datos

■ Números:

- En JavaScript existe sólo un tipo de dato numérico.
- Todos los números se representan a través del formato de punto flotante de 64 bits. 
- Este formato es el llamado `double` en los lenguajes Java o C++. 

Tipos de datos

- Cadenas de texto:

- El tipo de datos para representar cadenas de texto se llama `string`.
- Se pueden representar letras, dígitos, signos de puntuación o cualquier otro carácter de Unicode.
- La cadena de caracteres se debe definir entre comillas dobles o comillas simples. 

Tipos de datos

■ Cadenas de texto - Secuencias de escape:

Secuencia de escape	Resultado
\\	Barra invertida
\'	Comilla simple
\"	Comillas dobles
\n	Salto de línea
\t	Tabulación horizontal
\v	Tabulación vertical
\f	Salto de página
\r	Retorno de carro
\b	Retroceso

Tipos de datos

- **Valores booleanos:**
 - También conocido como valor lógico.
 - Sólo admite dos valores: `true` o `false`.
 - Es muy útil a la hora de evaluar expresiones lógicas o verificar condiciones.



Variables

- Se pueden definir como zonas de la memoria de un ordenador que se identifican con un nombre y en las cuales se almacenan ciertos datos.
- El desarrollo de un script conlleva:
 - Declaración de variables.
 - Inicialización de variables.

Variables

- Declaración de variables:
 - Se declaran mediante la palabra clave `var` seguida por el nombre que se quiera dar a la variable.
 - `var mi_variable;`
 - Es posible declarar más de una variable en una sola línea.
 - `var mi_variable1, mi_variable2;`

Variables

■ Inicialización de variables:

○ Se puede asignar un valor a una variable de tres formas:

- Asignación directa de un valor concreto.
- Asignación indirecta a través de un cálculo en el que se implican a otras variables o constantes.
- Asignación a través de la solicitud del valor al usuario del programa.

○ Ejemplos:

- `var mi_variable_1 = 30;`
- `var mi_variable_2 = mi_variable_1 + 10;`
- `var mi_variable_3 = prompt('Introduce un valor:');`





Operadores

- JavaScript utiliza principalmente cinco tipo de operadores:
 - Aritméticos.
 - Lógicos.
 - De asignación.
 - De comparación.
 - Condicionales.



Operadores




- **Operadores aritméticos:**
 - Permiten realizar cálculos elementales entre variables numéricas.

Operador	Nombre
+	Suma
-	Resta
*	Multiplicación
/	División
%	Módulo 
++	Incremento 
--	Decremento 

Operadores


- **Operadores lógicos:**


- Combinan diferentes expresiones lógicas con el fin de evaluar si el resultado de dicha combinación es verdadero o falso.

Operador	Nombre	
&&	Y	
	O	
!	No	

Operadores

■ Operadores de asignación:


- Permiten obtener métodos abreviados para evitar escribir dos veces la variable que se encuentra a la izquierda del operador. 


Operador	Nombre
+=	Suma y asigna
-=	Resta y asigna 
*=	Multiplica y asigna
/ *	Divide y asigna
%=	Módulo y asigna

Operadores

■ Operadores de comparación:

- Permiten comparar todo tipo de variables y devuelve un valor booleano. 

Operador	Nombre
<	Menor que
<=	Menor o igual que
==	Igual
>	Mayor que
>=	Mayor o igual que
!=	Diferente
===	Estrictamente igual
 !=	Estrictamente diferente



Operadores

■ Operadores condicionales:

- Permite indicar al navegador que ejecute una acción en concreto después de evaluar una expresión.

Operador	Nombre
?:	Condicional



- Ejemplo:

```
<script type="text/javascript">
  var dividendo = prompt("Introduce el dividendo: ");
  var divisor = prompt("Introduce el divisor: ");
  var resultado;
  divisor != 0 ? resultado = dividendo/divisor :
    alert("No es posible la división por cero");
  alert("El resultado es: " + resultado);
</script>
```



Sentencias condicionales

- Con las sentencias condicionales se puede gestionar la toma de decisiones y el posterior resultado por parte del navegador.
- Dichas sentencias evalúan condiciones y ejecutan ciertas instrucciones en base al resultado de la condición.
- Las sentencias condicionales en JavaScript son:
 - if.
 - switch.
 - while.
 - for.

Sentencias condicionales

- **if** – sintaxis (1):  

```
if (expresión) {  
    instrucciones  
}
```

Sentencias condicionales

- **if** – sintaxis (2): 

```
if (expresión) {  
    instrucciones_si_true  
} else {  
    instrucciones_si_false  
}
```


Sentencias condicionales

- **if** – sintaxis (3):




```
if (expresión_1) {  
    instrucciones_1  
} else if (expresión_2) {  
    instrucciones_2  
} else {  
    instrucciones_3  
}
```



Sentencias condicionales

■ switch – sintaxis:

```
switch (expresión){  
  case valor1:  
    instrucciones a ejecutar si expresión = valor1  
    break;  
  case valor2:  
    instrucciones a ejecutar si expresión = valor2  
    break;  
  case valor3:  
    instrucciones a ejecutar si expresión = valor3  
    break  
  default:  
    instrucciones a ejecutar si expresión es diferente a  
    los valores anteriores  
}
```




Sentencias condicionales

- **while** – sintaxis:

(1)

```
while (expresión) {  
    instrucciones  
}
```

(2)






```
do {  
    instrucciones  
} while (expresión)
```



Sentencias condicionales

- **for** – sintaxis:

```
for(valor_inicial_variable;  
expresión_condicional;  
incremento_o_decremento_de_la_variable)   
{  
    cuerpo_del_bucle  
}  
  

```