Indice

- Historia
- Inclusion en un xhtml
- comentarios
- variables
- Tipos de datos
    - Cadenas de caracteres
    - Booleanos
    - Objetos
    - Nulos
    - Arrays
- Funciones
- Estructura de Control: IF
- Alert, confirm y prompt

Historia

netscape crea como lenguaje de script > javascript.

No tiene nada ver con java de Sun su denominaciòn responde a una cuestión de mktng, en ese entonces java era un lenguaje muy popular.

internet explorer crea el Jscript, para evitar una guerra de navegadores netscape envió su lenguaje a que fuese estandarizado, asi entra el organismo ECMA (european computer manufacturers asociation)

El verdadero nombre del lenguaje es ECMAScript

Inclusion en un xhtml

interno:

<script language="JavaScript">

<!--

//-->
</script>

externo:
<script type=”text/javascript” src=”path/archivo.js”></script>

o también puede meterse adentro del código
<elementohtml evento=”javascript:nombreFuncion()”></element

comentarios

// **Comentario** de una línea

/*
Este es un **comentario**
por **bloque**. Sin importar
la cantidad de líneas.
*/

primeros scripts:

variables:

Una variable es un espacio en memoria donde se almacena un dato, un espacio donde podemos guardar cualquier tipo de información que necesitemos para realizar las acciones de nuestros programas.
Javascript cuenta con la palabra "var" que utilizaremos cuando queramos declarar una o varias variables. Como es lógico, se utiliza esa palabra para definir la variable antes de utilizarla.
En su declaración deben comenzar por un carácter alfabético o el subrayado. No podemos utilizar caracteres raros como el signo +, un espacio o un $.

Ejemplo:

var operando1
var operando2

También se puede asignar un valor a la variable cuando se está declarando

var operando1 = 23

Tipos de datos

Cadenas de caracteres

Són datos de texto y deben estar delimitados por comillas simples o dobles, pueden incluir caracteres especiales cómo ' (comilla) " (comilla doble), n (salto de línea), t (tabulador)...

<script>
var = "Holanamigo";
</script>

Booleanos

Són datos de bool, sus valores significan verdadero (true) y falso (false), se escriben sin comillas

<script>
var = true;
</script>

Objetos

Són conjuntos de variables y funciones definidos previamente por el lenguaje (objetos predefinidos) o por el usuario, són ejemplos de objetos predefinidos los objetos Image y Array, para definirlas, usaremos el prefijo new y por el uso de paréntesis:

<script>
var = new Image();
</script>

Nulos

Són datos vacios, se producen cuando se ha definido una variable como null para borrarla:

<script>
var = null;
</script>

Arrays:

Creación de Arrays javascript

El primer paso para utilizar un array es crearlo. Para ello utilizamos un objeto Javascript ya implementado en el navegador. Veremos en adelante un tema para explicar lo que es la orientación a objetos, aunque no será necesario para poder entender el uso de los arrays. Esta es la sentencia para crear un objeto array:

var miArray = new Array()

Esto crea un array en la página que esta ejecutándose. El array se crea sin ningún contenido,

es decir, no tendrá ninguna casilla o compartimiento creado. También podemos crear el array Javascript especificando el número de compartimentos que va a tener.

```
var miArray = new Array(10)
```

En este caso indicamos que el array va a tener 10 posiciones, es decir, 10 casillas donde guardar datos.

Es importante que nos fijemos que la palabra Array en código Javascript se escribe con la primera letra en mayúscula. Como en Javascript las mayúsculas y minúsculas si que importan, si lo escribimos en minúscula no funcionará.

Tanto se indique o no el número de casillas del **array javascript**, podemos introducir en el array cualquier dato. Si la casilla está creada se introduce simplemente y si la casilla no estaba creada se crea y luego se introduce el dato, con lo que el resultado final es el mismo. Esta creación de casillas es dinámica y se produce al mismo tiempo que los scripts se ejecutan. Veamos a continuación cómo introducir valores en nuestros arrays.

```
miArray[0] = 290
miArray[1] = 97
miArray[2] = 127
```

Se introducen indicando entre corchetes el índice de la posición donde queríamos guardar el dato. En este caso introducimos 290 en la posición 0, 97 en la posición 1 y 127 en la 2.
Los arrays en Javascript empiezan siempre en la posición 0, así que un array que tenga por ejemplo 10 posiciones, tendrá casillas de la 0 a la 9. Para recoger datos de un array lo hacemos igual: poniendo entre corchetes el índice de la posición a la que queremos acceder. Veamos cómo se imprimiría en la pantalla el contenido de un array.

```
var miArray = new Array(3)



miArray[0] = 155
miArray[1] = 4
miArray[2] = 499

for (i=0;i<3;i++){
        document.write("Posición " + i + " del array: " + miArray[i])
        document.write("<br>")
}
```

Declaración e inicialización resumida de Arrays

En Javascript tenemos a nuestra disposición una manera resumida de declarar un array y cargar valores en un mismo paso. Fijémonos en el código siguiente:
var arrayRapido = [12,45,"array inicializado en su declaración"]
Como se puede ver, se está definiendo una variable llamada arrayRapido y estamos indicando en los corchetes varios valores separados por comas. Esto es lo mismo que haver declarado el array con la función Array() y luego haberle cargado los valores uno a uno.

Funciones

Qué es una función

A la hora de hacer un programa ligeramente grande existen determinados procesos que se pueden concebir de forma independiente, y que son más sencillos de resolver que el problema entero. Además, estos suelen ser realizados repetidas veces a lo largo de la ejecución del programa. Estos procesos se pueden agrupar en una función, definida para que no tengamos que repetir una y otra vez ese código en nuestros scripts, sino que simplemente llamamos a la función y ella se encarga de hacer todo lo que debe.

Así que podemos ver una función como una serie de instrucciones que englobamos dentro de un mismo proceso. Este proceso se podrá luego ejecutar desde cualquier otro sitio con solo llamarlo. Por ejemplo, en una página web puede haber una función para cambiar el color del fondo y desde cualquier punto de la página podríamos llamarla para que nos cambie el color cuando lo deseemos.

Cómo se escribe una función

Una función se debe definir con una sintaxis especial que vamos a conocer a continuación.

```
function nombreFuncion (){
        instrucciones de la función
        ...
}
```

Veamos un ejemplo de función para escribir en la página un mensaje de bienvenida dentro de etiquetas <H1> para que quede más resaltado.

```
function escribirBienvenida(){
```

```
        document.write("<H1>Hola a todos</H1>")
}
```

Cómo llamar a una función

Para ejecutar una función la tenemos que invocar en cualquier parte de la página. Con eso conseguiremos que se ejecuten todas las instrucciones que tiene la función entre las dos llaves.
Para ejecutar la función utilizamos su nombre seguido de los paréntesis. Por ejemplo, así llamaríamos a la función escribirBienvenida() que acabamos de crear.

```
escribirBienvenida()
```

Estructura de Control: IF

F es una estructura de control utilizada para **tomar decisiones**. Es un condicional que sirve para realizar unas u otras operaciones en función de una expresión. Funciona de la siguiente manera, primero se evalúa una expresión, si da resultado positivo se realizan las acciones relacionadas con el caso positivo.

La sintaxis de la estructura IF es la siguiente.

```
if (expresión) {
  //acciones a realizar en caso positivo
  //...
}
```

Opcionalmente se pueden indicar acciones a realizar en caso de que la evaluación de la sentencia devuelva resultados negativos.

```
if (expresión) {
  //acciones a realizar en caso positivo
  //...
} else {
  //acciones a realizar en caso negativo
  //...
}
```

Ejemplo:

```
var llueve = true;
```

```
if (llueve){
alert("Cae agua");
}else{
alert("salio el sol")
}
```

Alert, confirm y prompt

alert
Crea una caja de diálogo con un icono de peligro amarillo, un botón 'Aceptar' y un texto definido por el parámetro enviado a la función.

```
<script>
alert("Ha ocurrido un error");
</script>
```
Las alertas, nos serán útiles para transmitir información al usuario tal como errores ocurridos en la navegación, problemas en el rellenado de un formulario...

Confirm
Crea una caja de confirmación con un icono de interrogación, botones Aceptar y Cancelar y un texto definido por el parámetro enviado a la función.

Devuelve 1 cuando el usuario abandona el diálogo pulsando Aceptar y 0 si lo hace pulsando Cancelar o el aspa de cerrar.

```
<script>
if(confirm('¿Seguro que ha leido las condiciones del contrato?'))this.form.submit();
</script>
```

Será útil para recibir información del usuario en tiempo de ejecución (al pulsar un botón, al pasar el mouse por un lugar...)

Prompt
Muestra un diálogo de campo de formulario con botones Aceptar y Cancelar, un texto definido por el primer parámetro enviado a la función y un input de texto con valor predeterminado definido por el segundo parámetro.

La función devuelve el valor insertado en el campo de formulario si el usuario pulsa en Aceptar o null si pulsa Cancelar o el aspa de cerrar.

```
<script>
nombre = prompt('Introduce tu nombre','[ nombre del usuario ]');
</script>
```

Prompt nos será útil sobre todo para recojer datos del usuario para utilizar en nuestro script en tiempo de ejecución.