Curso desarrollo web Front-End

Germán Caballero Rodríguez <germanux@gmail.com>

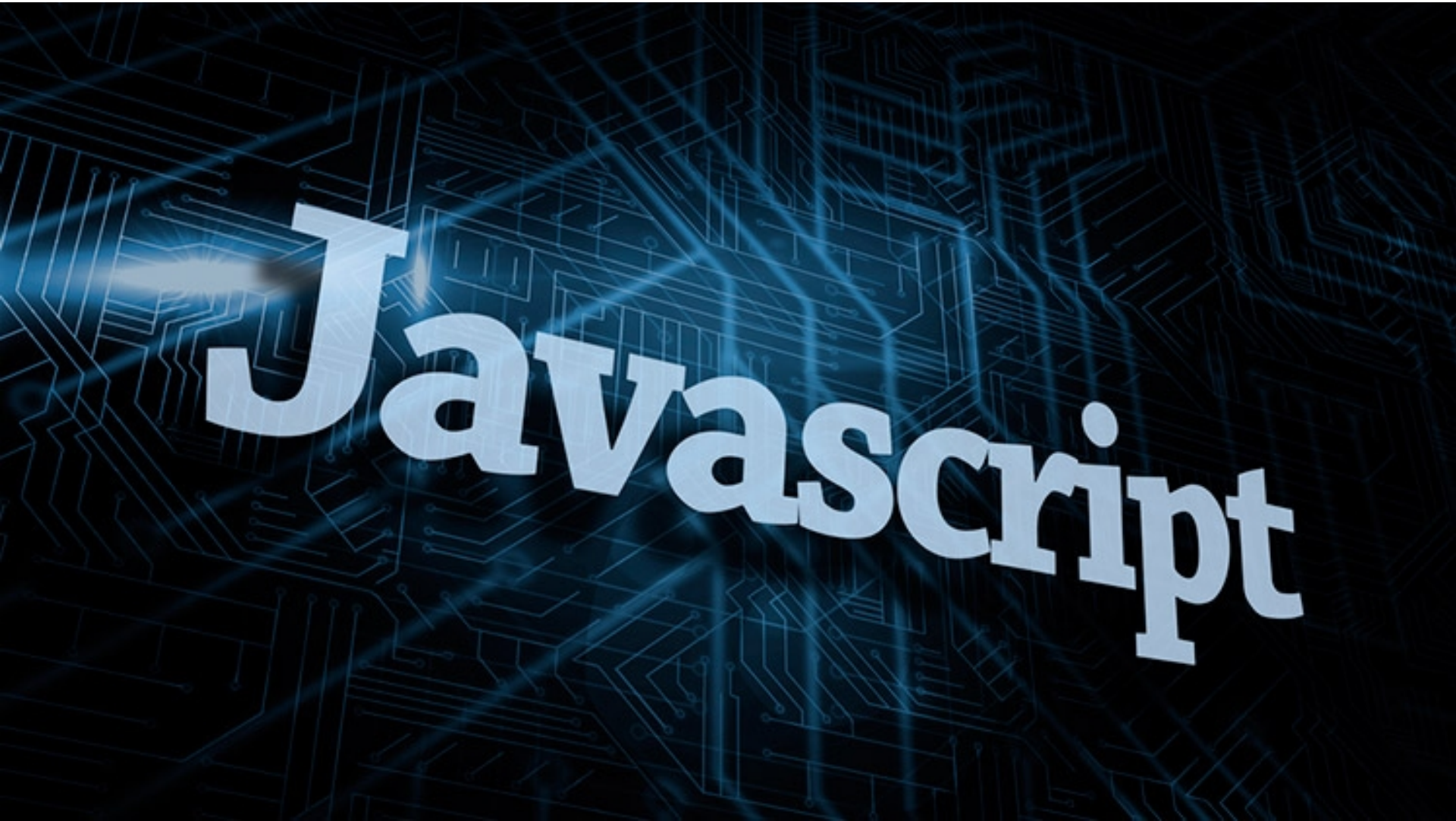
Generation SPAIN

Cámaras Fundación INCYDE

Unión Europea
Fondo Social Europeo

El FSE invierte en tu futuro

Programación básica con JS - II



INDICE

Definición de una función

- Antes de poder usar una función en el código de la página, se la debe definir.
- se debe indicar qué operaciones son las que de
- be hacer la función. La definición de la
- función es:

```
function nombredelafunción(argumento1, argumento2,...)
{
 instrucciones que debe realizar la función
}
```

Mensajes

- Se trata de ventanas que desde el código se lanzan al usuario para hacer que éste reaccione ante una situación o nos informe ante una duda.
- Realmente todos los mensajes se obtienen a través del objeto window .

alert

- Saca un mensaje por la pantalla el cual sólo deja la posibilidad de aceptarle.

```
alert(texto_del_mensaje);
```

prompt

- En este caso se trata de una ventana que pide entrar datos al usuario.
- De modo que esta función devuelve un valor que se puede usar en el código si es asignado a una variable.

```
prompt(texto_del_mensaje,valor_por_defecto);
```

```
respuesta=prompt("¿Qué quieres hacer?","comer");
```

Si el usuario pulsa Cancelar, la función prompt devuelve el valor nulo (null).

confirm

- Saca un mensaje de confirmación el cual suele tener dos botones: Aceptar y Cancelar.
- Sintaxis:

```
confirm(texto_del_mensaje)
```

Funciones predefinidas

- JavaScript trae consigo muchas funciones predefinidas.
- Señalamos aquí algunas de las más importantes:

eval(textoCódigo).

- La función eval tiene un único parámetro que es una cadena de texto.
- Esta función hace que el texto sea interpretado como si fuera código normal de JavaScript. Ejemplo:

```
eval("alert('Hola')");
```

parseInt(textoNúmero, base).

- Convierte el texto (que debe tener cifras numéricas) a formato de número.
- El segundo parámetro es opcional y representa la base del número. ejemplo:

```
alert(parseInt("110011",2));//Sale 51
```

parseFloat(textoNúmero).

- Convierte el texto (que debe tener cifras numéricas) a formato de número con decimales.

escape(texto)

- Muestra el código ASCII de los símbolos del texto.
- Cada número en el resultado va precedido del símbolo % y el código ASCII sale en forma Hexadecimal.

isNaN(expresión)

- Devuelve true si la expresión tiene un contenido no numérico.

Ejercicio 6

- Crea tres páginas HTML sólo con código JavaScript:
- <html>
  - <head></head>
  - <body>
    - <script> /* CODIGO JS */ </script>
  - <body>
- <html>

Ejercicio 6

- Ejercicio 6.1.html) Se cargan por teclado tres números distintos. Mostrar por pantalla el mayor de ellos.
- Ejercicio 6.2.html) Se ingresa por teclado un valor entero, mostrar una leyenda que indique si el número es positivo, cero o negativo.
- Ejercicio 6.3.html) Del mayor de ellos, mostrar si es +, - ó 0.

objetos

- Los objetos son una de las bases fundamentales de JavaScript.
- Un objeto es una agrupación de variables, que en ese caso se llaman propiedades, y de funciones, las cuales se llaman métodos.
- Las propiedades definen las características de los objetos y los métodos las operaciones que podemos hacer con él.
- JavaScript posee muchos objetos predefinidos y también permite crear nuestros propios objetos.

propiedades

- Como ya se comentó anteriormente, los objetos poseen propiedades asociadas, para acceder a ellas se utiliza el punto, en esta forma:

```
objeto.propiedad
```

```
miordenador.marca="HP";
miordenador.procesador="pentium III 900 Mhz"
miordenador.ram=64;
```

métodos

- Los métodos son funciones asociadas a los objetos. Su uso es:

```
objeto.metodo()
```

operación new

- La instrucción new sirve para crear nuevos objetos en el tiempo de ejecución del código JavaScript.
- Ejemplo:

```
miordenador = new ordenador("HP", "Pentium III", 64);
```

objetos predefinidos

- El objeto string sirve para manejar cadenas de texto.
- Cada vez que creamos una variable de cadena, en realidad estamos creando una variable de tipo string.
- Por lo tanto no será necesaria su declaración.

Métodos de String

- charAt(n).
  - Muestra el carácter situado en la posición n de la cadena.
- indexOf(cadenaInterna,inicio).
  - Devuelve la posición de la cadena interna en el texto, teniendo en cuenta que el primer carácter es el número 0.
  - El primer parámetro es el texto que se busca; el segundo es opcional e indica desde qué posición del texto comenzamos a buscar.
  - Si no se encuentra la cadena interna, se devuelve el valor –1. Ejemplo:

```
var cadena="Miguel Indurain"
var subcadena="Indurain"
alert(cadena.indexOf(subcadena))
/*El resultado será 7, ya que la primera posición del
  texto (en este caso la 'M') es la 0.
```

substring(x,y).

- Muestra el fragmento de texto que va desde la posición x a la posición y. Ejemplo:

```
 var cadena="Miguelón Indurain";
 var subcadena="Indurain";
 alert(cadena.substring(5,11));
//Sale lón In
```

Mas de string

- **toLowerCase().** Convierte la cadena a minúsculas.
- **toUpperCase().**Convierte la cadena a mayúsculas.

Propiedades de string

- **length.** Almacena el tamaño de la cadena de texto.

Math

- El objeto Math tiene propiedades y métodos que representan valores m
  - **abs(*n*).** Calcula el valor absoluto de n.
  - **acos(*n*).** Calcula el arco coseno de n.
  - **exp(*n*).** Calcula e^n.
  - **floor(*n*).** Redondea n a su valor inferior.
  - **log(*n*).** Calcula el logaritmo de n.
  - **max(x,y).** Devuelve el mayor valor de x o y.
  - **min(x,y).** Devuelve el menor valor de x o y.
  - **pow(x,y).** Devuelve el x^y.
  - **random().** Genera un número aleatorio entre 0 y 1. Ejemplo:

propiedades de math

- **E.** Devuelve la constante de Euler o número **e.**
- **LN2.** Devuelve el logaritmo neperiano de 2.
- **LN10.** Devuelve el logaritmo neperiano de 10.
- **LOG2E.** Logaritmo en base 2 de **e**.
- **LOG10E.** Logaritmo en base 10 de **e.**
- **PI.** Devuelve el número PI.

objeto Date()

- Este objeto representa fechas y horas.
- JavaScript no permite trabajar con fechas anteriores a 1970, ya que desde ese momento es cuando comienza a contar las fechas en milisegundos.
- En los meses, el mes 0 será Enero, y el mes 11 es Diciembre.
- Los días de la semana y del mes se cuentan desde el número 1 (Jueves = 4).

Algunos metodos

- **getHours().** Devuelve la hora.
- **getMinutes().** Devuelve los minutos.
- **getMonth().** Devuelve el mes (con números del 0 al 11).
- **getSeconds().** Devuelve los segundos.
- **getTime().** Devuelve el número de milisegundos de la fecha, empezan contar desde 1970.
- **getTimezoneOffset().** Devuelve la diferencia en minutos entre la zona ho actual y la hora solar (GMT).
- **getYear().** Devuelve el año.
- **setDate(*valor*).** Establece el día del mes.
- **setFullYear(*valor*).** Establece el año (con cuatro cifras).
- **setHours(*valor*).** Establece la hora.

ej

```
fecha=new Date();
//crea un nuevo objeto de fecha cuyo valor inicial serán
//la fecha y hora actuales
```

```
fecha=new Date(año, mes, día, hora, minutos, segundos”);
```

objeto array

- Un array es un conjunto de datos agrupados.
- Para acceder a cada elemento individual de el array se usa un número de índice, el primer elemento tendrá el índice 0.
- En el caso de los arrays de JavaScript, su uso es muy eficaz y mucho más libre que en los lenguajes formales (como Pascal por ejemplo).

objeto array

Para crear un array se hace:

```
nombrearray = new Array()
```

Esto crea un array de tamaño indeterminado.

objeto array

Para rellenar los valores del array:

```
nombrearray[0] = valor;
nombrearray[1] = valor;
...
nombrearray[n] = valor;
```

objeto array

- Se puede especificar el tamaño:

```
nombreArray= new Array(tamaño)
```

O incluso asignar valores en la propia creación del array. Ejemplo:

```
equipos= new Array("Real Madrid", "F. C. Barcelona");
```

Flexibilidad del array

Además un array puede tener distintos tipos de datos:

```
miOrdenador = new Array("HP", "Pentium III", 64);
```

Y cada elemento de un array puede ser otro array:

```
elemento = new Array(8);
elemento[3] = new Array(5);
```

métodos del objeto array

- **concat(*array*).** Agrupa dos arrays. Disponible desde la versión 1.2. Ejemplo:

```
a = new Array(12, 3, 5);
b = new Array("Hola", "Adios");
c = a.concat(b);
//c es el array (12, 3, 5, "Hola", "Adios")
```

- **join().** Saca una cadena de texto que contiene todos los elementos del array:

```
a = new Array("Rojo", "Azul", "Verde");
b = a.join();
//b contiene "Rojo,Azul,Verde"
```

- **reverse().** Invierte el orden del los elementos de un array. El primero pasa a ser el último y viceversa.

- **Sort().** Obtiene la matriz de manera ordenada.

Propiedades del objeto Array

- **length.** Cuenta el número de elementos del array.

Objetos del navegador

- Los objetos del navegador son todos aquellos objetos que el navegador pone a nuestra disposición para poder modificar los elementos de las páginas.

propiedades

- **appCodeName**. Nombre del código del navegador. Todos los navegadores devuelven la cadena **Mozilla**. Por lo que esta propiedad no es interesante.
- **appName**. Nombre del navegador. Podrá ser: **Microsoft Internet Explorer** o **Netscape.**
- **appVersion**. Versión del navegador. La cadena que devuelve esta propiedad indica la versión completa. Ejemplos:

- **language** o **browserLanguage. language** es una propiedad que s funciona con Netscape, mientras que **browserLanguage** sólo funciona c Explorer. Pero ambas devuelven el lenguaje del navegador. En ambos casos dos primeras letras que devuelven son el lenguaje del navegador (es=Espa en=Inglés, por ejemplo).

- **platform**. Contiene el tipo de sistema operativo del ordenador del usua **Win32** indicaría un sistema Windows 95/98/Me/NT/2000. Otros valores s **Win16** (Windows 3.1 y anteriores) **Mac68x** (McIntosh clásico) **MacP**

screen

- Permite acceder a las propiedades de la pantalla del usuario

propiedades

- **width** y **height.** Respectivamente, anchura y altura total de la pantalla.
- **availWidth** y **availHeight.** Respectivamente, anchura y altura disponible en pantalla tras restar la barra de tareas del sistema operativo. Esta medida no muy exacta ya que no tiene en cuenta la personalización del usuario.
- **colorDepth.** Número de bits por píxel que utiliza la pantalla.

window

- Este objeto, representa a la ventana en la cual se ve la página web.
- En el caso de que la página tenga marcos, se generan tantos objetos window como marcos haya

propiedades

- **closed.** Valor booleano que indica si una ventana ha sido cerrada.
- **defaultStatus.** Texto que la barra de estado mostrará cuando se cargue página web (texto por defecto de la barra de estado).
- **frames.** Array que representa a todos los marcos de la ventana.

window

- **history.** Es un objeto (se verá más adelante) que representa las enlaces a l páginas a las que el visitante había accedido antes de llegar a la ventana actual.
- **location**. Objeto que almacena información sobre el URL de la página actual.
- **name.** El nombre de la ventana.
- **parent.** Ventana "padre" de la actual. Si la actual es un marco, parent será página con etiqueta <FRAMESET>.
- **self**. Se refiere a la propia ventana activa.
- **top.** Ventana superior del navegador.
- **status**. Mensaje de la barra de estado.
- **window.** Igual que **self**.

location

- Objeto incluido dentro del objeto window.
- Almacena información sobre la localización de la página de la ventana y por tanto permite cambiar dinámicamente la página web que se está mostrando.

propiedades

- **href**. Especifica la dirección URL de la ventana (por ejemplo: *http://www.uva.es/problemas/ex1.htm#marca1"*)
- **hostname**. Especifica la parte del URL en la que va el nombre del host *(www.uva.es)*
- **host**. Idéntico al anterior, sólo que al final se indica el número de puerto utilizado *(www.uva.es:80)*

- **pathname**. Especifica la parte del URL en la que va la ruta al recurso (en el ejemplo: */problemas/ex1,htm"*)
- **port**. Puerto utilizado para mostrar la página (generalmente el 80).
- **hash**. Indica qué marcador de la página se utilizó al abrir la misma, si no se usó ninguno aparece vacío (en el ejemplo sería *marca1*).
- **protocol**. Parte referida al protocolo de la URL (en el ejemplo *http*).
- **search**. La parte de una URL que va detrás del signo **?**. Sólo ciertas páginas llevan este signo (en concreto las páginas que llaman a CGIs).

nétodos

- **replace(*URL*)**. Carga una nueva página en la ventana actual indicando su URL entre comillas y además también reemplaza a la página anterior en la lista del historial. Esto último es la única diferencia entre usar este método y cambiar la dirección directamente en la propiedad **href** del objeto **location.**
- **reload().** Hace que se vuelva a recargar la página.

document

- Este objeto representa al contenido de una página web.
- Está incluido dentro del objeto window.

propiedades

- **bgColor**. Color del fondo
- **fgColor**. Color del texto.
- **linkColor**. Color de los enlaces normales.
- **vlinkColor**. Color de los enlaces visitados
- **alinkColor**. Color de los enlaces activos.

- **links.** Array que contiene todos los enlaces que usan el atributo HREF. El orden de los enlaces en la matriz es el orden de uso en el código (el primer enlace en el código será **document.links[0].**) Ejemplo:

```
document.links[0].href ="http://www.hola.com"
//hace que el primer enlace del documento apunte a la dirección www.hola.com
```

- **lasModified**. Fecha de última actualización del documento en forma de cadena de texto. Algunos servidores no proporcionan este dato.

- **URL**. URL del documento. Es idéntico a utilizar **location.href**.

- **cookie**. Escribe o lee el archivo de cookies de la página web. Un archivo de cookies sirve para guardar información sobre el usuario en su propia máquina, con esta propiedad se permite hacerlo

métodos document

- **clear()**. Borra el documento.
- **write(*textoHTML*)**. Escribe el texto indicado en el documento. Ese texto puede contener si se desea etiquetas HTML
- **writeln(*textoHTML*)**. Lo mismo que la anterior, sólo que esta añade un salto de línez tras escribir el texto.
- **close()**. Cierra el documento.

history

- Objeto que representa a las direcciones de las páginas que se almacenan en el historial del navegador.
- Este objeto está dentro del objeto window.

Objeto window.history

propiedades

- **length**. Número de páginas almacenadas actualmente en el historial.

métodos

- **back()**. Hace que la ventana muestre la página visitada anteriormente.
- **forward()**. Hace que la ventana muestre la página siguiente.
- **go(*nº*).** Hace que se muestre la página del historial indicada con un número. De modo que **history.go(-1)** muestra la página anterior y **history.go(-3)** hace que se muestre la página antepenúltima.

image

- Objeto que representa una imagen en el documento definida con la etiqueta HTML <IMG> o con el código JavaScript new Image.
- Todas las imágenes del documento están contenidas en la matriz de imágenes document.images.

propiedades

Son las mismas que los atributos de la etiqueta IMG del HTML, por eso sólo se comentan:

- **border**
- **height**
- **hspace**
- **lowsrc**
- **name**
- **src**
- **vspace**
- **width**

Ejemplo:

```
var imagenAtras = new image(120,87);
//nueva imagen con ancho=120 y altura=87
imagenAtras.src="dibujogris.gif";
//la imagen muestra el archivo dibujogris.gif
```

Ejercicio

- Crear un script para rellenar alumnos: que 1º pida el número de alumnos, luego vaya pidiendo nombre y fecha de nacimiento x mensajes, los vaya guardando en un array, y por último muestre una tabla con el resultado, por ejemplo:

Nº de alumnos: 13

Nombre	Fecha de nacimiento
Alumno numero uno	10-12-1980
Alumno numero dos	11-03-1975
...	...

Ejercicio

- Crea una página en la que se muestren 5 imágenes de diferentes tamaños.
- Además que pida al usuario mediante un mensaje una url local con una imagen que añadir
- Que borre la página y genere dinámicamente un documento con el siguiente formato todo con JavaScript (salvo los CSS):
- Emplear estilos e IDs:
  - Los infos que sean una clase
    - Los datos otra
  - Las figuras otra clase
    - Con formatos de bordes
    - Echos con CSS:
    - Márgenes de 4 a 10 px...
    - Padding de 2 px
    - Colores Negro/Naranja en HEX

Cada información como si fuera un artículo:

- Información del navegador: lenguaje, versión...
- Información de la pantalla: ancho, alto...
- Información de la ventana: ancho, alto...

Listado de imágenes con el formato de figura:

Nombre de la imagen
IMAGEN
Info imagen (ancho, alto...)