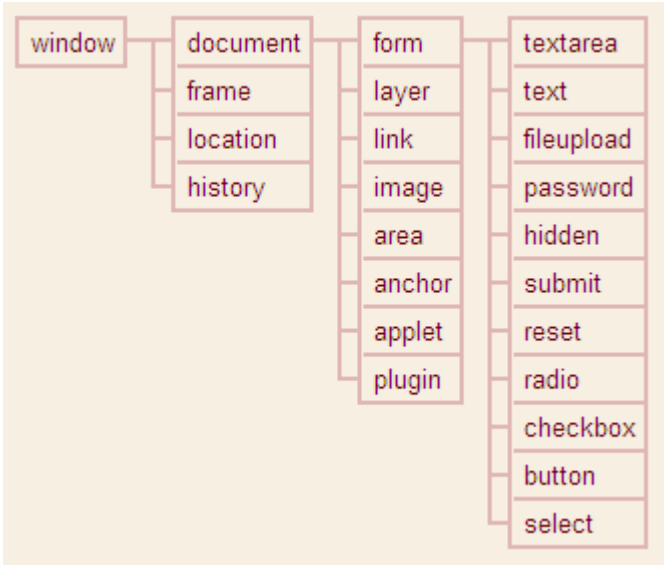
JavaScript

CONTENIDO:



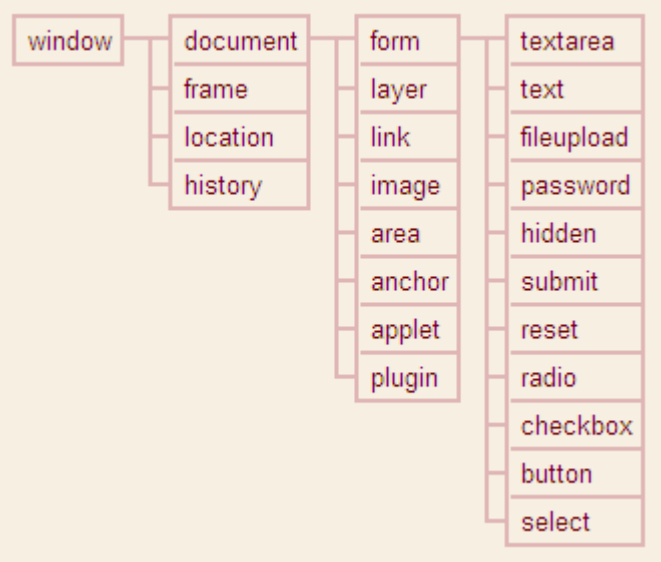
Unidad 10 - Jerarquía Javascript

Cuando se carga un documento en el Navegador, crea un número de objetos JavaScripts con propiedades basadas en el HTML dentro del documento y otra información pertinente. Estos objetos existen en una jerarquía que refleja la estructura de la página HTML.

En esta jerarquía, los descendientes de un objeto son propiedades suyas y a su vez son objetos que pueden tener otros descendientes. Al final de la jerarquía se encuentran solamente las propiedades de los objetos (normalmente, se denominan hojas). Cada una de las propiedades contiene un valor y, en cierto sentido, se pueden considerar como variables en sentido clásico de la programación.

1.– Estructura de la Jerarquía

Cuando nuestro navegador abre una página web, su motor JavaScript la analiza, y crea una estructura de objetos, ordenados jearárquicamente de la siguiente forma:



Cuando se carga un documento en el Navegador, crea un número de objetos JavaScripts con propiedades basadas en el HTML dentro del documento y otra información pertinente. .

Según esta jerarquía, tenemos un objeto principal window (la ventana del navegador). Dentro de window podemos encontrar otros objetos: document, frame, location e history. A su vez, dentro document, encontramos otros objetos, como los enlaces de la página (links), las imágenes (image) o los formularios (form). Por último, dentro del formulario, podemos encontrar sus elementos.

- Todos los objetos tienen propiedades, a las que podemos acceder como nombre_objeto.nombre_propiead. Las propiedades son valores del objeto, por ejemplo, color de fondo. Por tanto, podemos considerar a document como una propiead de window, y a link como una propiedad de document, etc...Podemos ver el valor de las propieadedes, o asignarles un valor nuevo.
- Todos los objetos tienen métodos, acccesibles como nombre_objeto.nombre_metodo('parametro'). Los métodos son funciones o procesos del objeto, que realizan acciones. Por ejemplo, mover una ventana o abirla. Los parámetros dependeránd el método.
- Algunos objetos o propiedadades son arrays. Por ejemplo, el objeto image de document es un array con todas las imágenes. Para cada una, tenemos sus propiedades, como el src, con el nombre del archivo.

2.– El Objeto Windows

Se trata del objeto más alto en la jerarquía del navegador (navigator es un objeto independiente de todos en la jerarquía), pues todos los componentes de una página web están situados dentro de una ventana. El objeto window hace referencia a la ventana actual. Veamos a continuación sus propiedades y sus métodos.

El objeto window es el más importante. A partir de él se pueden crear nuevos objetos window que serán nuevas ventanas ejecutando el navegador. Tales ventanas se pueden controlar desde la ventana padre. Además permite cerrar ventanas, actúa sobre los marcos, puede sacar mensajes (de error u otro tipo), confirmación y entrada de datos, y mantiene una matriz por cada tipo de elemento que puede haber en un documento, formularios, enlaces, imágenes y marcos.

El objeto window se refiere a la ventana actual del navegador, por lo que es el más alto en la jerarquía, ya que todos lo demás objetos están dentro de la ventana.

Propiedades.

location. URL de la dirección actual. Ver. Si la cambiamos, redirigimos a otra página. Por ejemplo, window.location='http://www.aulaclic.es'; cambiaría la página actual por la de aulaclic.

history. Es un array con las páginas visitadas en la ventana actual.

closed. Es un booleano que nos dice si la ventana está cerrada (closed = true) o no (closed = false).

defaultStatus. Cadena que contiene el texto por defecto que aparece en la barra de estado (status bar) del navegador.

frames. Es un array: cada elemento de este array (frames[0], frames[1], ...) es uno de los frames que contiene la ventana. Su orden se asigna según se definen en el documento HTML.

self. Es un nombre alternativo del window actual.

status. String con el mensaje que tiene la barra de estado.

top. Nombre alternativo de la ventana del nivel superior.

window. Igual que self: nombre alternativo del objeto window actual.

opener. Es una referencia al objeto **window** que lo abrió, si la ventana fue abierta usando el método open() que veremos cuando estudiemos los métodos.

Métodos:

close(). Cierra la ventana.

moveBy(x, y). Mueve la ventana, x píxeles en horizontal, y y píxeles en vertical.

moveTo(x, y). Mueve la ventana a las coordenadas dadas.

open(URL, nombre, características). Abre una ventana con la URL indicada, el nombre que le demos, y las características, por ejemplo si muestra la barra de menús, la abrra de herramientas, el alto y ancho, etc... Los parámetros son opcionales.

back(), forward(), navega hacia atrás o hacia alante en el historial.

blur(). Elimina el foco del objeto window actual

Apertura de una ventana

Abrir una ventana a través de JavaScript es tan simple como pasar a una instrucción una dirección URL para que la localice y la cargue:

nuevaVentana=window.open("dirección URL");

Es necesario asignar la nueva ventana a una variable (en este caso a nuevaVentana) para poder operar posteriormente sobre ella. Sin embargo ese no es el nombre como tal de la ventana, por lo que si en un enlace ponemos:

Pulsa

no funcionará.

En realidad, son varios los parámetros que se pasan a una ventana al crearla:

variable=window.open("dirección URL",

"nombre de la Ventana","parámetros de apertura");

donde:

dirección URL es la página que se va a cargar en la nueva ventana.

nombre de la ventana es el nombre que se podrá utilizar posteriormente en los target de los enlaces.

parámetros de apertura es una cadena que contiene los valores para ciertos atributos de la ventana, que son los siguientes:

toolbar,location,directories,status,menubar, scrollbars,resizable.

Cada uno de estos atributos puede tomar los valores YES o NO, o bien, 1 ó 0, respectivamente.

Además, podemos darle la anchura y altura en pixels mediante los atributos width, height.

Para ponerlos todos en la misma cadena se pone el atributo seguido del signo de igual y el valor que tiene. Los atributos se separan entre sí mediante comas. Por ejemplo:

"toolbar=0,location=1,directories=0,resizable=0,width=200,height=100"

Hay que tener cuidado de ponerlos en ese orden y de no dejar espacios entre ellos.

Por ejemplo, abramos una ventana con barra de herramientas, sin posibilidad de escribir una dirección y que no sea redimensionable. En ella vamos a cargar la página "www.romeroesteo.es" y la vamos a llamar "segundaPag"; la altura será de 200 pixeles.

nuevaVentana=window.open("http://www.romeroesteo.es",
"segundaPag",
"toolbar=yes,location=no,resizable=no,height=200");

Cerrar ventanas

Para cerrar una ventana se utiliza el método close() de las mismas:

variable_de_ventana.close()

Si la ventana a cerrar es la propia, entonces tendremos que usar:

window.close()

, o simplemente ,

close();

Ha de tenerse cuidado al cerrar una ventana dentro de un elemento de un formulario ya que si es la propia ventana la que queremos cerrar habremos de especificar

window.close();

y no solo

close();

Esto es así porque dentro de los formularios el objeto por defecto es document no window, por ello, close() solamente cerraría el documento, no la ventana.

Mensajes de alerta, confirmaciones y entradas de datos.

Aunque para cada una de estas tareas se utiliza un método distinto, las veremos juntas ya que son muy similares.

Los mensajes de alerta aparecen con el método alert (mensaje)

El parámetro mensaje es la cadena que queremos que se visualice. Se puede usar para sacar mensajes de error, de ayuda, alertar sobre ciertos eventos, etc.

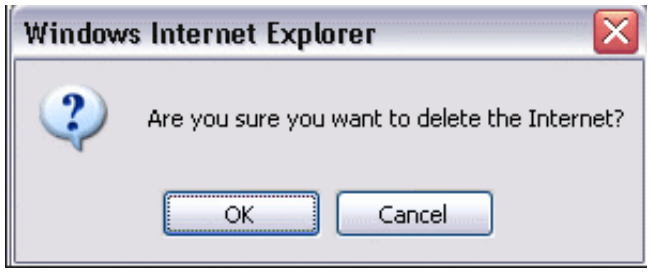
3.– El Objeto Location

Para una confirmación de alguna opción, habremos de utilizar el método

confirm(mensaje);

Este método devuelve un valor lógico true o false por lo

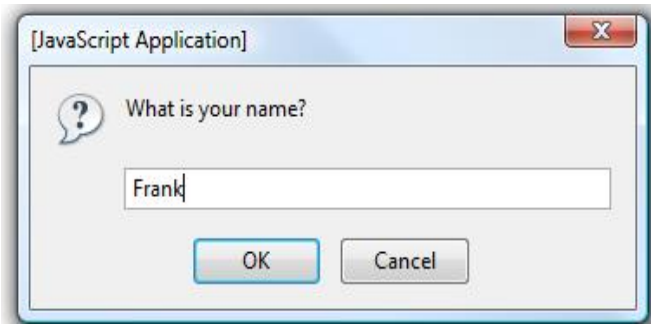
que podrá usarse en una sentencia de asignación o en una expresión lógica.



Por último, la adquisición de datos se hace a través del método

prompt(mensaje, valor_por_defecto)

El parámetro mensaje es obligatorio, pero el valor por defecto es opcional.



Este objeto contiene la URL actual así como algunos datos de interés respecto a esta URL. Su finalidad principal es, por una parte, modificar el objeto location para cambiar a una nueva URL, y extraer los componentes de dicha URL de forma separada para poder trabajar con ellos de forma individual si es el caso. Recordemos que la sintaxis de una URL era:

protocolo://maquina_host[:puerto]/camino_al_recurso

Propiedades

hash. Cadena que contiene el nombre del enlace, dentro de la URL.

host. Cadena que contiene el nombre del servidor y el número del puerto, dentro de la URL.

hostname. Cadena que contiene el nombre de dominio del servidor (o la dirección IP), dentro de la URL.

href. Cadena que contiene la URL completa.

pathname. Cadena que contiene el camino al recurso, dentro de la URL.

port. Cadena que contiene el número de puerto del servidor, dentro de la URL.

protocol. Cadena que contiene el protocolo utilizado (incluyendo los dos puntos), dentro de la URL.

search. Cadena que contiene la información pasada en una llamada a un script, dentro de la URL.

Métodos

reload(). Vuelve a cargar la URL especificada en la propiedad href del objeto location.

replace(cadenaURL). Reemplaza el historial actual mientras carga la URL especificada en cadenaURL.

4.– El Objeto History

Este objeto se encarga de almacenar una lista con los sitios por los que se ha estado navegando, es decir, guarda las referencias de los lugares visitados. Se utiliza, sobre todo, para movernos hacia delante o hacia atrás en dicha lista.

Propiedades

current. Cadena que contiene la URL completa de la entrada actual en el historial.

next. Cadena que contiene la URL completa de la siguiente entrada en el historial.

length. Entero que contiene el número de entradas del historial (i.e., cuántas direcciones han sido visitadas).

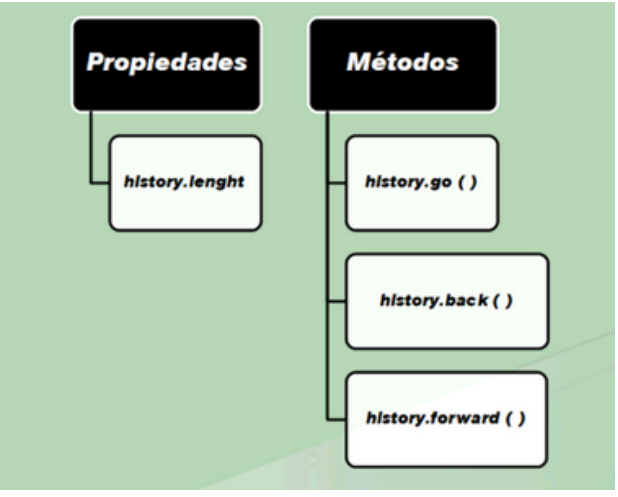
previous. Cadena que contiene la URL completa de la anterior entrada en el historial.

Métodos

back(). Vuelve a cargar la URL del documento anterior dentro del historial.

forward(). Vuelve a cargar la URL del documento siguiente dentro del historial.

go(posicion). Vuelve a cargar la URL del documento especificado por posicion dentro del historial. posicion puede ser un entero, en cuyo caso indica la posición relativa del documento dentro del historial; o puede ser una cadena de caracteres, en cuyo caso representa toda o parte de una URL que esté en el historial.



5.– El Objeto Navigator

Este objeto simplemente nos da información relativa al navegador que esté utilizando el usuario.

Propiedades

appCodeName. Cadena que contiene el nombre del código del cliente.

appName. Cadena que contiene el nombre del cliente.

appVersion. Cadena que contiene información sobre la versión del cliente.

language. Cadena de dos caracteres que contiene información sobre el idioma de la versión del cliente.

mimeTypes. Array que contiene todos los tipos MIME soportados por el cliente. A partir de NS 3.

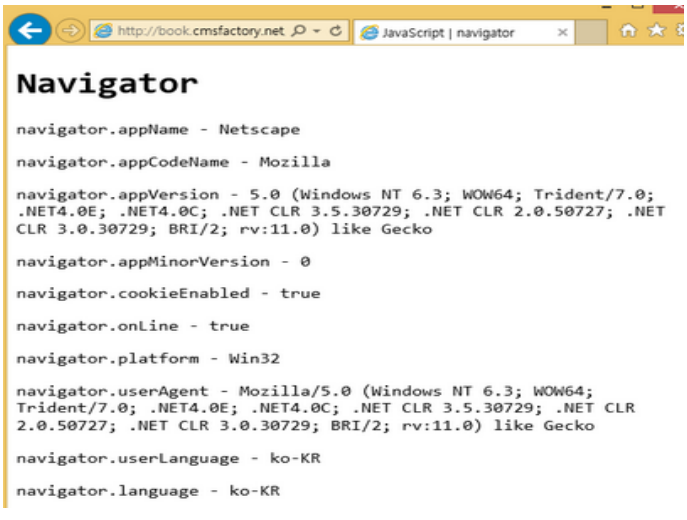
platform. Cadena con la plataforma sobre la que se está ejecutando el programa cliente.

plugins. Array que contiene todos los plug-ins soportados por el cliente. A partir de NS 3.

userAgent. Cadena que contiene la cabecera completa del agente enviada en una petición HTTP. Contiene la información de las propiedades appCodeName y appVersion.

Métodos

javaEnabled(). Devuelve true si el cliente permite la utilización de Java, en caso contrario, devuelve false.



IES MIGUEL
ROMERO ESTEO

Este material forma parte del currículo del módulo Implantación de Aplicaciones WEB correspondiente al ciclo formativo de grado superior "Administración de sistemas informáticos en red".

La información contenida en el magazine ha sido extraída de la Web y, en parte, de la experiencia profesional acumulada durante mis años de docencia.

Por tanto, es de uso público para tareas de docencia y aprendizaje.