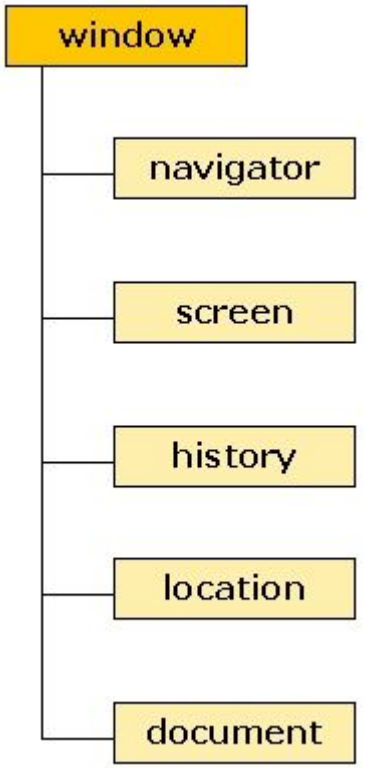
OBJETOS DEL NAVEGADOR EN JAVASCRIPT

UT3.- OBJETOS PREDEFINIDOS

Susana López Luengo

Objetos del navegador (BOM)

- El Browser Object Model (BOM) permite a JavaScript comunicarse con el navegador web Browser Object Model (BOM).
- La mayoría de los navegadores web modernos han implementado la mayoría los mismos métodos y propiedades para la interacción JavaScript
- Estos son conocidos como las propiedades y métodos de BOM



Objetos del navegador (BOM)

objeto Window

- El objeto window representa la ventana que contiene un documento DOM
- En los navegadores con pestañas, cada una contiene su propio objeto window
- Esto significa que el objeto window no se comparte entre diferentes pestañas de la misma ventana del navegador
- Algunos métodos, como **window.resizeTo** y **window.resizeBy** se aplican sobre toda la ventana del navegador y no sobre una pestaña específica a la que pertenece el objeto

Objetos del navegador (BOM)

propiedades del objeto Window

- **navigator** Devuelve el **objeto Navigator** de la ventana
- **screen** Devuelve el **objeto screen** de la ventana
- **history** Devuelve el **objeto history** de la ventana
- **location** Devuelve el **objeto de location** de la ventana
- **document** Devuelve el **objeto de document** de la ventana
- **length** Devuelve el número de elementos <iframe> en la ventana actual
- **name** Fija o devuelve el nombre de la ventana
- **opener** Devuelve una referencia a la ventana que ha creado la ventana
- **closed** Devuelve si una ventana ha sido cerrada o no
- **defaultStatus** Fija o devuelve el texto por defecto de la barra de estado de una ventana

Objetos del navegador (BOM)

propiedades del objeto Window

- **outerHeight** Devuelve la altura de la ventana, incluyendo barras de herramientas y de scroll
- **outerWidth** Devuelve la anchura de la ventana, incluyendo barras de herramientas y de scroll
- **pageXOffset** y **scrollX** Devuelve en píxeles del documento actual que ha sido movido en horizontal de la esquina izquierda de la ventana
- **pageYOffset** y **scrollY** Devuelve en píxeles del documento actual que ha sido movido en vertical de la esquina izquierda de la ventana
- **screenLeft** y **screenX** Devuelven la coordenada horizontal de la ventana relativa a la pantalla
- **screenTop** y **screenY** Devuelven la coordenada vertical de la ventana relativa a la pantalla

Objetos del navegador (BOM)

propiedades del objeto Window

- **parent** Devuelve la ventana padre de la ventana actual
- **self** Devuelve la ventana actual
- **status** Fija o devuelve el texto de la ventana de estado de la ventana
- **top** Devuelve la parte más alta de la ventana de navegación
- **localStorage** permite guardar pares clave/valor en el navegador web sin fecha de caducidad
- **sessionStorage** permite guardar pares clave/valor en el navegador web para la sesión actual
- **frameElement** Devuelve el elemento <iframe> en el que la ventana actual ha sido insertada
- **frames** Devuelve todos los elementos <iframe> de la ventana actual elements in the current window

Gestión de las ventanas

Abrir y cerrar nuevas ventanas

- Es una operación muy común en las páginas web.
- En algunas ocasiones se abren sin que el usuario haga algo.
- HTML permite abrir nuevas ventanas pero no permite ningún control posterior sobre ellas.
- JavaScript permite gestionar diferentes aspectos relacionados con las ventanas como por ejemplo abrir nuevas ventanas al presionar un botón
- Cada una de estas ventanas tiene un tamaño, posición y estilo diferente.
- Estas ventanas emergentes suelen tener un contenido dinámico.

Objetos del navegador (BOM)

Abrir ventanas

open() Abre una nueva ventana o pestaña

variable=window.open(“direccion URL”,”nombre de la ventana”,”parametros 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.
  - Podemos definir width y height en pixels

Objetos del navegador (BOM)

métodos del objeto Window

- **Ejemplo open()**

Vamos a abrir una ventana con barra de herramientas, sin posibilidad de escribir una dirección y que no sea redimensionable. En ella vamos a cargar la página “educa.madrid.org” y la vamos a llamar “educamadrid”. La altura será de 300 pixeles

```
nuevaVentana=window.open("http://educa.madrid.org","educamadrid","toolbar=yes,location=no,resizable=no,height=300" );
```

Ejemplo Abrireducamadrid.html

Objetos del navegador (BOM)

métodos del objeto Window

- **Crear una ventana con open()**

Vamos a crear una ventana con el código HTML que queramos:

```
miVentana=window.open("","Ventana nueva");
miVentana.document.write('<html>');
miVentana.document.write('<head> </head>');
miVentana.document.write('<body>');
miVentana.document.write('<h1> MI WEB </H1> ');
miVentana.document.write('</body> </html>');
```

Objetos del navegador (BOM)

métodos del objeto Window

- **close()** Cierra la ventana actual
- **stop()** Para la carga de la página
- **focus()** Fija el foco de la ventana actual
- **blur()** Quita el foco de la ventana actual
- **print()** Imprime el contenido de la ventana actual

https://www.w3schools.com/jsref/obj_window.asp

Objetos del navegador (BOM)

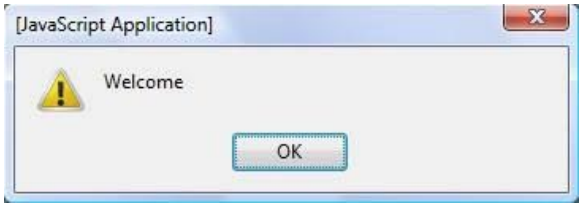
métodos del objeto Window

- **moveBy()** Mueve una ventana de forma relativa a su posición
- **moveTo()** Mueve una ventana a la posición indicada
- **resizeBy()** Cambia el tamaño de la ventana por los pixeles especificados
- **resizeTo()** Cambia el tamaño de la ventana por la altura y anchura especificada
- **scrollBy()** Desplaza el documento por el número especificado de píxeles
- **scrollTo()** Desplaza el documento a la coordenada especificada

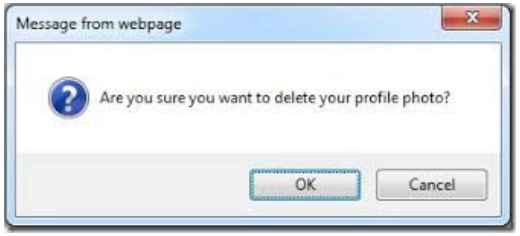
Objetos del navegador (BOM)

métodos para abrir ventanas de diálogo del objeto Window

- **alert()** Muestra una ventana de diálogo con un mensaje y el botón OK



- **confirm()** Muestra una ventana de diálogo con un mensaje y un botón OK y otro Cancel



- **prompt()** Muestra una ventana de diálogo con una caja de texto para introducir información

Objetos del navegador (BOM)

métodos para abrir ventanas de diálogo del objeto Window

- **setInterval()** Llama a una función cada vez que pasa un intervalo de tiempo (en milisegundos)
- **clearInterval()** Quita el contador fijado con setInterval()
- **setTimeout()** Llama a una función después de pasar un intervalo de tiempo (en milisegundos)
- **clearTimeout()** Quita un contador fijado con setTimeout()
- **getComputedStyle()** Devuelve los estilos CSS aplicados a un elemento
- **getSelection()** Devuelve una selección de objetos que representa el texto seleccionado por el usuario
- **matchMedia()** Devuelve una lista de objetos MediaQuery que representan la cadena media query CSS especificada

Objetos del navegador (BOM)

objeto Navigator

- Este objeto contiene información sobre el navegador
  - Tipo de navegador.
  - Versión del navegador.
  - Sistema operativo
- Se suele utilizar para obtener este tipo de información, y en base al resultado, tomar una decisión sobre qué tipo de código ejecutar.

Objetos del navegador (BOM)

propiedades del objeto navigator

- **appCodeName** Devuelve el nombre en código del navegador
- **appName** Devuelve el nombre del navegador
- **appVersion** Devuelve información de la versión del navegador
- **cookieEnabled** Comprueba si están habilitadas las cookies
- **geolocation** Devuelve un objeto de Geolocalización para indicar la localización del usuario
- **language** Devuelve el idioma del navegador
- **onLine** Determina si el navegador tiene conexión a internet
- **platform** Devuelve para que plataforma está compilado el navegador
- **product** Devuelve el motor de renderizado del navegador
- **userAgent** Devuelve la cabecera del agente de usuario enviado por el navegador al servidor

Objetos del navegador (BOM)

algunos métodos del objeto navigator

- **javaEnabled()** Indica si el navegador tiene habilitado java
- **vibrate()** Causa vibración en el dispositivo que la soporta. No hace nada si el soporte para vibración no está disponible.
- **getVRDisplays()** Devuelve un array de objetos VRDisplay que representan cualquier dispositivo VR conectado al ordenador que esté disponible

Objetos del navegador (BOM)

Objeto Screen

- Corresponde a la pantalla utilizada por el usuario
- Todas sus propiedades son solamente de lectura.
- Podemos consultar sus propiedades, nunca modificarlas
- No tiene métodos

Objetos del navegador (BOM)

propiedades del objeto Screen

- **availHeight** Devuelve la altura de la pantalla(excluyendo el menú de la ventana)
- **availWidth** Devuelve la anchura de la pantalla(excluyendo el menú de la ventana)
- **colorDepth** Devuelve la profundidad en bits del la paleta de colores para mostrar imágenes
- **height** Devuelve la altura total de la pantalla
- **pixelDepth** Devuelve la resolución de color (en bits por pixel) de la pantalla
- **width** Devuelve la anchura total de la pantalla
- **orientation**: Devuelve la orientación de la pantalla

Objetos del navegador (BOM)

objeto History

- El objeto history contiene las URLs visitados por el usuario
- Este objeto es parte de los objetos window y se accede a través de la propiedad window.history
- **history.length** Propiedad que devuelve el número de URLs del historial
- **history.back()** Método que carga la URL anterior del historial
- **history.forward()** Método que carga la siguiente URL del historial
- **history.go()** Método que carga una URL específica del historial

Browser Object Model

objeto Location

- Corresponde a la URL de la página web en uso.
- Su principal función es la de consultar las diferentes partes que forman la URL: Dominio, protocolo, puerto
- Métodos
  - **assign()** Carga un nuevo documento
  - **reload()** Recarga el documento actual
  - **replace()** Reemplaza el documento actual por uno nuevo

Browser Object Model

propiedades del objeto Location

- **hash** Fija o devuelve la parte del ancla (#) de una URL
- **host** Fija o devuelve el hostname y el puerto de una URL
- **hostname** Fija o devuelve el hostname de una URL
- **href** Fija o devuelve la URL entera
  origin Devuelve el protocolo, hostname y el puerto de una URL
  pathname Fija o devuelve la ruta de una URL
  port Fija o devuelve el puerto de una URL
  protocol Fija o devuelve el protocolo de una URL
  search Fija o devuelve la cadena de consulta (?) de una URL

Práctica guiada 1-2

UT03-Pr02ObjetosPredef hasta ej. 7

Generación de elementos HTML desde código

- Uno de los principales objetivos de JavaScript es convertir un documento **HTML estático** en una aplicación **web dinámica**.
- Por ejemplo, es posible ejecutar instrucciones que crean **nuevas ventanas con contenido propio**, en lugar de mostrar dicho contenido en la ventana activa.
- **Cada ventana** de un navegador **presenta un documento HTML** y es representada por un **objeto window** que contiene un **subobjeto document**.
- El objeto document contiene a su vez una serie de objetos que representan todo el contenido del documento HTML, como el **texto, imágenes, enlaces, formularios, tablas, etc.**

Generación de elementos HTML desde código

Se puede crear y manipular todo tipo de objetos:

```
<script type="text/javascript">
  document.write("<form name=\"cambiacolor\">");
  document.write("<b>Selecciona un color para el fondo de página:</b><br>");
  document.write("<select name=\"color\">");
  document.write("<option value=\"red\">Rojo</option>");
  document.write("<option value=\"blue\">Azul</option>");
  document.write("<option value=\"yellow\">Amarillo</option>");
  document.write("<option value=\"green\">Verde</option>");
  document.write("</select>");
  document.write("<input type=\"button\" value=\"Modifica el color\"onclick=\"document.bgColor=document.cambiacolor.color.value\">");
  document.write("</form>");
</script>
```

Generación de elementos HTML desde código

A partir del script anterior se obtiene la siguiente página web dinámica:

Marcos. Aplicaciones prácticas

- Es posible dividir la ventana de una aplicación web en dos o más partes independientes.
- Con JavaScript se puede interactuar entre estos sectores independientes.
- Dichos sectores se denominan **marcos - iframe**.
- Algunas páginas web presentan una estructura en la cual una parte permanece fija mientras que otra va cambiando.
- Este efecto se produce creando diferentes páginas web y posicionando cada una de ellas en un marco diferente.

Marcos. Aplicaciones prácticas

- La página web de la API de Java podemos ver la ventana dividida en 3 partes. Estos marcos interactúan entre ellos

Marcos. Aplicaciones prácticas

Los marcos se definen utilizando HTML. Estos presentan la ventaja de poder crear páginas en las que es posible mantener elementos como botones, enlaces, imágenes, etc.

Con JavaScript podemos manipular los marcos y realizar una interacción entre ellos.

<iframe>: esta etiqueta define las características de un marco.

<frame> y <frameset> no son soportadas en HTML5

https://www.w3schools.com/html/html_iframe.asp

WEBGRAFÍA Y ENLACES DE INTERÉS

- https://www.w3schools.com/js/
- Curso Desarrollo de aplicaciones con HTML,node.js y Javascript. UPM. Miriadax
- https://www.aprenderaprogramar.com


ESCUELA PÚBLICA:
DE TOD@S
PARA TOD@S
Vallecas