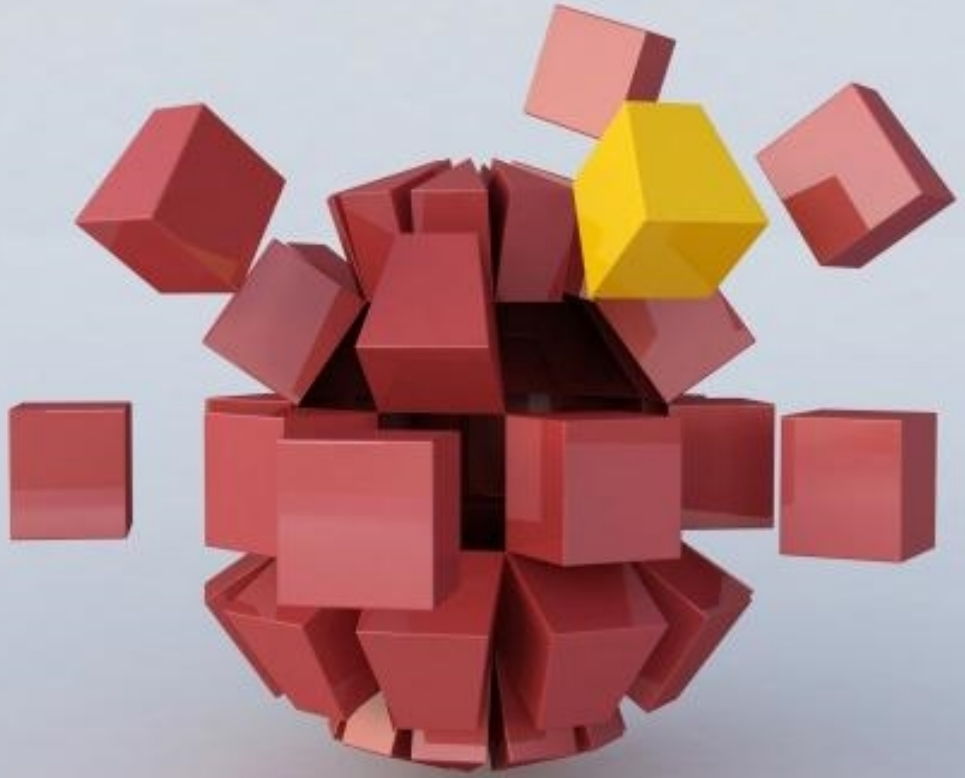
TEMA 4: Objetos Predefinidos de JavaScript II



DESARROLLO WEB EN ENTORNO CLIENTE

TEMA 4. Objetos predefinidos de Javascript II

I. **Objeto Window**
   - **Document**
   - **History**
   - **Location**
   - **Screen**
   - **Navigator**

Interacción de los objetos con el navegador

- Además de los objetos presentados anteriormente, existe otro tipo de objetos que permiten **manipular** diferentes **características** del **navegador** en sí mismo.

Objeto Document

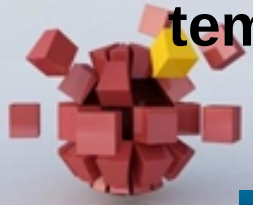
- **BOM (Browser Object Model) ó DOM Level 0**
  - Internet Explorer y Netscape Navigator introdujeron el concepto de *Browser Object Model* o BOM, que permite acceder y modificar las propiedades de las ventanas del propio navegador
  - El modelo BOM lo constituyen el objeto window y todas sus propiedades: Histoty, Location, Screen, Navigator
  - El mayor inconveniente de BOM es que ninguna entidad se encarga de estandarizarlo o definir unos mínimos de interoperabilidad entre navegadores

Objeto Document

- **El objeto Document representa el documento html de la ventana actual. W3C ha definido el estándar DOM para estándarizar las propiedades y métodos del objeto document y asegurar la interoperabilidad entre navegadores**
  - **Este estándar se divide en 4 partes**
    - DOM Level 1
    - DOM Level 2
    - DOM Level 3
    - DOM Level 4

document.getElementById("id");

- **SIEMPRE accederemos y modificaremos el objeto document mediante el estándar DOM que veremos en temas siguientes.**



http://www.w3schools.com/jsref/dom_obj_document.asp

DOM – Actividad 1

- Realiza una página html con un formulario y un script Javascrip para realizar lo siguiente:
  - Un campo de texto con un valor inicial «Texto inicial»
  - Un botón tal que al pulsarlo muestre en un alert el valor del campo de texto.
  - Otro botón tal que al pulsarlo cambie el valor del campo a uno solicitado al usuario mediante la función prompt.
  - Consulta en http://www.w3schools.com/jsref/default.asp las propiedades de DOM input text acceder para obtener el valor seleccionado.

DOM – Actividad 2

- Realiza una página html con un formulario y un script Javascrip para realizar lo siguiente:
  - Una lista desplegable con 3 valores de colores
  - Un botón tal que al pulsarlo muestre en un alert el valor del la lista desplegable que está seleccionado.
  - Consulta en http://www.w3schools.com/jsref/default.asp las propiedades de DOM select y DOM option a las que debes acceder para obtener el valor seleccionado.

DOM – Actividad 3

- Realiza una página html con un formulario y un script Javascrip para realizar los siguiente:
  - oUna lista desplegable con 4 opciones para ciudades
  - oUn campo de texto inicialmente vacío
  - oUn botón tal que al pulsarlo coja el valor seleccionado de la lista desplegable y lo muestre en el campo de texto.
  - oUtiliza el método getElementById para acceder a los elementos.
  - oConsulta en http://www.w3schools.com/jsref/default.asp las propiedades de DOM select y DOM option a las que debes acceder para obtener el valor seleccionado.

Document.write

- El método write nos permite generar código html dinámicamente
  - document.write(' El resultado es ');
- Sólo se debe usar si el usuario no va a interaccionar más con el nuevo html y no vamos a necesitar acceder a él
  - Origina el problema del documento cerrado, muy díficil de gestionar

Actividades

- **En el aula virtual Ejercicios 1**

Objeto Window

- Representa la ventana actual que contiene el navegador
- Es el objeto raíz de la jerarquía de objetos de JavaScript.
- Permite gestionar múltiples ventanas.
- Es un objeto implícito, con lo cual no es necesario nombrarlo para acceder a sus objetos propiedades
  - window.document ó simplemente document

Objeto Window

Métodos		
alert()	forward()	setinterval()
back()	home()	setTimeOut()
blur()	moveTo()	scrollBy()
close()	open()	scrollTo()
confirm()	print()	stop()
find()	prompt()	setinterval()
focus()	resizeTo()	setTimeOut()

Propiedades		
closed	location	pageYoffset
defaultStatus	locationbar	parent
document	menubar	personalbar
frames	name	scrollbars
history	opener	self
innerHeight	outerHeight	status
innerWidth	outerWidth	toolbar
length	pageXoffset	top

Métodos del objeto Window

Método	Descripción
alert(*mensaje*)	Muestra un mensaje informativo no modal e impide cualquier interacción con el documento hasta que el usuario pulse **Aceptar**.
confirm(*mensaje*)	Muestra un mensaje no modal y solicita confirmación presentando los botones **Cancelar** y **Aceptar**. Devuelve un booleano que indica qué botón se ha pulsado (true si **Aceptar** o false si **Cancelar**).
prompt(*mensaje,valor*)	Muestra un mensaje no modal solicitando al usuario un dato. El método devuelve ese dato (en forma de cadena). Mediante el argumeno *valor* podemos proponer un dato predeterminado.
showModalDialog(*url*)	Muestra una ventana emergente (*popup*) modal con el contenido del url especificado. Tenga en cuenta que la mayoría de los navegadores interceptan los popups impidiendo que se visualicen (por el abuso que tradicionalmente se ha hecho de ellos con motivos comerciales).

Gestión de las ventanas

- **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.**

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.**
    - <a href= … target=…>

Gestión de las ventanas

- **Abrir y cerrar nuevas ventanas:**
  - **Con JavaScript es posible abrir una ventana vacía mediante el método `open()`:**
    - `nuevaVentana = window.open();`
  - **De este modo la variable llamada `nuevaVentana` contendrá una referencia a la ventana creada.**

Gestión de las ventanas

- **Abrir y cerrar nuevas ventanas:**
  - **El método open( )cuenta con tres parámetros:**
    - URL.
    - Nombre de la ventana.
    - Colección de atributos que definen la apariencia de la ventana.
  - **Ejemplo:**

```
nuevaVentana = window.open(“http://www.misitioWeb.com/ads”,
“Publicidad”, “height=100, widht=100”);
```

Gestión de las ventanas

- **Un ejemplo completo:**

Usamos write sólo porque es una ventana informativa. El usuario no tiene que interaccionar con ella, sino tendremos un html que pasaremos como url a la nueva ventana

```
<html><head></head><body>
 <h1> Ejemplo de Apariencia de una Ventana</h1>
 <br><input type="Button" value="Abre una Ventana" onclick=”
  myWindow1=window.open('', 'Nueva Ventana', 'width=300,   height=200');
  myWindow1.document.write('<html>');
  myWindow1.document.write('<head>');
  myWindow1.document.write('<title>Ventana Test</title>');
  myWindow1.document.write('</head>');
  myWindow1.document.write('<body>');
  myWindow1.document.writeln('Se usan las propiedades: ');
  myWindow1.document.write('<li>height=200</li> <li>width=300</li>');
  myWindow1.document.write('</body>');
  myWindow1.document.write('</html>');"/>
</body></html>
```

AbrirVentana.html

Gestión de las ventanas

- **Un ejemplo completo:**

Si el usuario va a interaccionar con la ventana secundaria o su contenido es dinámico, abriré en ella un html creado previamente. **No usaré write**.

```
<html>
<head></head>
<body>
 <h1> Ejemplo de Apariencia de una Ventana</h1>
 <br><input type="Button" value="Abre una Ventana" onclick=”
  myWindow1=window.open(‘EligeColor.html', 'Nueva Ventana', 'width=300,
height=200');
</body></html>
```

Gestión de las ventanas

- **Para cerrar una ventana se puede invocar el método `close()`:**

```
<input type=button value=Cerrar
onClick=window.close();/>
```

Gestión de las ventanas

- **Se pueden abrir múltiples ventanas**
  - **Suele ser bastante molesto para el usuario**
  - **Basta con usar un bucle for**

MultiplesVentanas.html

Gestión de las ventanas

- **Apariencia de las ventanas:**
  - **Con las propiedades height y width podemos establecer el alto y ancho de las ventanas.**
  - **Podemos utilizar los métodos moveTo() y moveBy() para cambiar la posición de la ventana.**
  - **Como la resolución puede variar es mejor especificar posiciones relativas definiendo % de screen.height y screen.width para estos métodos.**

Gestión de las ventanas

- **Comunicación entre ventanas:**
  - **Desde una ventana se pueden abrir o cerrar nuevas ventanas.**
  - **La primera se denomina ventana principal, mientras que las segundas se denominan ventanas secundarias.**
  - **Desde la ventana principal se puede acceder a las ventanas secundarias.**

Gestión de las ventanas

- **Comunicación entre ventanas:**
  - **En el siguiente ejemplo se muestra cómo acceder a una ventana secundaria:**

ComunicacionEntreVentanas.html

```
<html><head></head><body>
  <script>
   var ventanaSecundaria = window.open("", "ventanaSec”, "width=500,
   height=500");
  </script>
  <h1> Comunicaci&oacute;n entre ventanas </h1><br>
  <form name=formulario>
   <input type=text id=‘url’ size=50 value="http://www.">
   <input type=button value="Mostrar URL en ventana secundaria"
  onclick="ventanaSecundaria.location =
document.getElementById(‘url’).value;">
  </form></body></html>
```

Gestión de las ventanas

- **Comunicación entre ventanas:**
  - **Desde una ventana secundaria podemos acceder a propiedades y métodos de la ventana principal con**
    - **window.opener**
    - Podemos escribir código como:

```
window.opener.document.getElementById('color').value=‘rojo’;
```

Actividad

- **Realiza un html que contenga una lista desplegable con 3 valores de colores**
- **Cuando el usuario seleccione un color y pulse un botón llamado “selecciona color”, abrirás una ventana secundaria con un html previamente creado,**
- **El html de la ventana secundaria tendrá un campo de texto, y en su campo de texto debe aparecer el color seleccionado en la ventana principal.**

Actividad

- **Realiza un html que contenga un botón llamado “selecciona color” y un campo de texto inicialmente vacio.**
- **Cuando el usuario el botón llamado “selecciona color”, abrirás una ventana secundaria con un html previamente creado.**
- **El html de la ventana secundaria tendrá una lista desplegable con 3 valores de colores y una opción que vendrá marcada por defecto con valor –seleccione color-**
- **Cuando el usuario seleccione un color de la lista desplegable de la ventana secundaria, esté aparecerá en el campo de texto de la ventana principal.**

El objeto Navigator

- Permite identificar las características de la plataforma sobre la cual se ejecuta la aplicación web. Ejemplo:
  - Tipo de navegador.
  - Versión del navegador.
  - Sistema operativo

.

Métodos
javaEnable()

Propiedades
appCodeName
appName
appVersión
cookieEnable
platform
userAgent

http://www.useragentstring.com/pages/useragentstring.php

El objeto screen

- Corresponde a la pantalla utilizada por el usuario.
- Todas sus propiedades son solamente de lectura.
- Se puede utilizar
  - Para cargar un diseño (css) adecuado a nuestra pantalla
  - Adaptar posición y tamaño de las ventanas emergentes

Propiedades
availHeight
availWidth
colorDepth
height
pixelDepth
width

El objeto History

- Almacena las referencias de las páginas web visitadas.
- Las referencias se guardan en una lista utilizada principalmente para desplazarse entre dichas páginas web.
- No es posible acceder a los nombres de las URL, ya que es información privada.

Métodos
back()
forward()
go()

Propiedades
current
length
next
previous

El objeto Location

- Corresponde a la URL de la página web en uso.
- Su principal función es la de consultar las diferentes partes que forman una URL como por ejemplo:
  - El dominio.
  - El protocolo.
  - El puerto.

Métodos
assign()
reload()
replace()

Propiedades
hash
host
hostname
href
pathname
port
protocol
search

Actividades

- **En el aula virtual Ejercicios 2**