Modelo de objetos con lenguaje Script.

El Modelo de Objetos de Documento (DOM - Document Object Model -) describe cómo todos los elementos en una página HTML, tales como campos de entrada, imágenes, etc, se relacionan con la estructura más alta: el propio documento. Llamando al elemento por su nombre correcto DOM, podemos influir en él mediante un lenguaje de programación, e,g. Javascript, Java, etc.

|  |  |
| --- | --- |
| A description... | La gran difusión de Ajax y Javascript es notable en muchos proyectos en Internet. Sin embargo hay que considerar que algunos navegadores tienen deshabilitado Javascript por defecto. Lo cual hace necesario que se habilite en cada máquina cliente.  Para que estos códigos funcionen en Mozilla Firefox es necesario  1. Selecciona “Herramientas” 2. Selecciona “Opciones” 3. Selecciona “Contenido” 4. Selecciona la casillla de ‘Habilitar JavaScript’ y darle clic en OK |

write

Permite escribir texto html desde Javascript. El texto se escribe directamente el body del html.

Es la forma más fácil de escribir HTML desde Javascript, lo cual no quiere decir que es la mejor manera de escribir código HTML desde Javascript o incluso que va a trabajar en todas las situaciones. De hecho, el método document.write es muy limitado en lo que puedes hacer con él.

El lector debe considerar que document.write sólo funciona con las página que el navegador procesa como HTML.

Ejemplo: Imprimir un mensaje y abrir una nueva ventana mediante DOM

* Las cadenas en Javascript son objetos y por lo tanto tienen métodos y atributos. En las líneas 4 y 5 se presenta ejemplos de ellos que facilmente pueden deducir su finalidad.
* La línea 6 imprime la cadena en el body.
* La línea 8 abre una página nueva, la función open tiene como parámetros la página que abrira, y el lugar donde lo hará en este caso \_self sustituye la página actual

<body>

<script language='JavaScript'>

var mensaje = '¡Hola mis estimados y finos alumnos!. Buen dia';

mensaje = mensaje.toUpperCase().bold().italics();

mensaje = mensaje.fontsize(6).fontcolor('red');

document.write(mensaje);

if ( confirm('Abrir HomePage del curso?') )

window.open('http://www.prograweb.com.mx/pweb/index.html',

'\_self');

</script>

</body>

Ejemplo: Imprimir un saludo en la pantalla dependiente de la hora de la máquina cliente

* En la línea 4 hacemos una instancia de la clase Date.
* getHours extrae la hora del lado cliente (Línea 6)
* Escribimos en el documento el mensaje: document.write( saludo() )

<head>

<script language='javascript'>

function saludo() {

var horario = new Date()

var hora = horario.getHours();

if( hora < 12 ) { var mensaje = 'Buenos dias!';}

else if (hora >= 12 and hora <= 19) {

var mensaje = 'Buenas tardes!';

} else {

var mensaje = 'Buenas noches!';

}

mensaje = '¡Hola Mundo. ' + mensaje;

return mensaje;

}

</script>

</head>

<body>

<script language='javascript'>

document.write( saludo() );

</script>

</body>

**getElementById**

Permite obtener la referencia a un elemento de la página mediante el id de dicho elemento.

Una vez obtenida la referencia al elemento, podemos acceder a todas sus propiedades

<head>

<title>getElementById</title>

<script type='text/javascript'>

function cambiarColor( color ) {

var elemento = document.getElementById('parrafo');

elemento.style.font = 'bold italic 16 pt Verdana';

elemento.style.color = color;

}

</script>

</head>

<body>

<p id='parrafo'>Mis colores favoritos. ¿Adivina por qué?</p>

<button onclick = 'cambiarColor('blue');'>Azul</button>

<button onclick = 'cambiarColor('red');'>Rojo</button>

</body>

A description...

**innerHTML**

Sirve para obtener o modificar el HTML del documento HTML. Se utiliza la propiedad InnerHtml para modificar mediante programación el contenido interno entre las etiquetas de apertura y cierre de un control de servidor HTML.

La propiedad InnerHtml no codifica automáticamente los caracteres especiales en entidades HTML, es el programador el responsable de hacerlo.

Ejemplo: Crear una lista con viñetas no ordena en forma dinámica

* En la línea 4 creamos un vector de días de la semana.
* En la línea 11 modificamos el contenido del elemento llamado contenedor.

<body>

<div id='contenedor' name='contenedor'></div>

<script language='javascript' type='text/javascript'>

dias = new Array('Lunes','Martes','Miércoles','Jueves',

'Viernes', 'Sábado', 'Domingo');

HTML = '<ul>';

for(i = 0; i < dias.length; i++) {

HTML += '<li>' + dias[i] + '</li>';

}

HTML += '</ul>';

document.getElementById('contenedor').innerHTML = HTML;

</script>

</body>

**Ejemplo: Agregar elementos a un documento**

HTML

* Los elementos son agregados mediante el evento onclick (Línea 7)
* La línea 8 declara un elemento div vacio cuya función es servir de contenedor

Javascript

* En la línea 5 se crea un objeto de tipo div
* El objeto nuevoElemento hereda todas las características de div, por ejemplo innerHTML, color, fontSize.
* Se crea una instancia del contenedor
* Para agregar los elementos al contenedor usamos appenChild

A description...

Javascript

var n = 0;

function sumarHijo() {

n++;

if( n < 6) {

var nuevoElemento = document.createElement('div');

if(n <= 5) {

nuevoElemento.innerHTML = 'mi color favorito';

nuevoElemento.style.color = 'red';

} else if ( n == 6) {

nuevoElemento.innerHTML = 'No abuses';

nuevoElemento.style.color = 'blue';

nuevoElemento.style.fontSize = '40px';

}

var contenedor = document.getElementById('contenedor');

contenedor.appendChild(nuevoElemento);

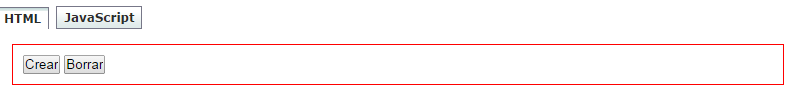
}

}

A description...

**Ejemplo: Cargar la ruta a un archivo**

* Se asignan las características de un *fieldset* en **fi** (Línea 4)
* Creamos un contenedor de tipo *div* (Línea 5)
* Las líneas 9 y 15 crean dos objetos de tipo input
* Mediante *insertBefore* insertamos el contenedor antes del botón crear (Línea 7)
* Asignamos al evento *onclick* del botón la función borrar y el pasamos como parámetro el nombre del propio botón
* Usamos *removeChild* para eliminar el *div* y su contenido (Línea 25)



**Javascript**

var num = 0;

function crear(objeto) {

num++;

var fi = document.getElementById('contenedor');

var contenedor = document.createElement('div');

contenedor.id = 'div' + num;

contenedor.className = 'codigoCorto';

fi.insertBefore(contenedor, objeto);

var elemento = document.createElement('input');

elemento.type = 'file';

elemento.name = 'archivo' + num;

elemento.size = 40;

contenedor.appendChild(elemento);

}

function borrar() {

var elemento = document.getElementById('contenedor');

while (elemento.firstChild) {

elemento.removeChild(elemento.firstChild);

}

}

**Ejemplo: Mostrar los objetos interconstruidos de Javascript**

* Se presentan los siguientes objetos: *document, even, history, location, navigator, screen* y *windows*
* La función eval evalúa una cadena de código JavaScript sin referencia a ningún objeto en particular (Línea 2). El objetivo de *Object* es este caso es crear un arreglo atributos, métodos y parámetros del objeto interconstruido.
* La línea 9 desarrolla un *array* asociativo con los nombres de los métodos del objeto interconstruido.
* En la línea 19 asignamos al atributo innerHTML el contenido.

![A description...](data:None;base64,)

**Javascript**

function propiedades( objeto ){

object = new Object(eval( objeto ));

var cadena = '<table class='tabla'>';

cadena += 'Propiedades de ' + objeto;

cadena += '<th>Número</th>';

cadena += '<th>Propiedad</th>';

cadena += '<th>Valor</th>';

var i = 0;

for(p in object){

cadena += '<tr>';

cadena += '<td>' + i + '</td>';

cadena += '<td>' + p + '</td>';

cadena += '<td>' + object[p] + '</td>';

cadena += '</tr>';

i++;

}

cadena += '</table>';

var elemento = document.getElementById('contenedor');

elemento.innerHTML = cadena;

}

**Ejemplo: Mostrar los objetos interconstruidos de Javascript**

* Se presentan los siguientes objetos: document, even, history, location, navigator, screen y windows
* La función eval evalúa una cadena de código JavaScript sin referencia a ningún objeto en particular (Línea 2). El objetivo de Object es este caso es crear un arreglo atributos, métodos y parámetros del objeto interconstruido.
* La línea 9 desarrolla un array asociativo con los nombres de los métodos del objeto interconstruido.
* En la línea 19 asignamos al atributo innerHTML el contenido.

![A description...](data:None;base64,)

HTML

<head>

<title>Propiedad de los Objetos</title>

<link href='0402ObjectPropiedades.css' rel='stylesheet' type='text/css' />

<meta http-equiv='content-type' content='text/html; charset=iso-8859-1' />

<script type='text/javascript' src='0402ObjectPropiedades.js'></script>

</head>

<body>

<form name='forma' method='post' action='#'>

<div align='center'>

Objeto:

<select name='objetos' id='objetos'

onChange='propiedades(window.document.forms[0][0].value)'>

<option value='document'>document</option>

<option value='event'>event</option>

<option value='history'>history</option>

<option value='location'>location</option>

<option value='navigator'>navigator</option>

<option value='screen'>screen</option>

<option value='window'>window</option>

</select>

</div>

</form>

<div id='contenedor' class='centrado'></div>

Ejemplo: Efecto Row Rollover Table

HTML

<head>

<meta http-equiv='Content-Type' content='text/html; charset=iso-8859-1' />

<title>Row Rollover table</title>

<script type='text/javascript' src='0402tableruler.js'></script>

<script type='text/javascript'>

window.onload = function(){tableruler();}

</script>

<link href='0402tableruler.css' rel='stylesheet' type='text/css' />

</head>

<body>

<table class='ruler' summary='Campeones FIFA'>

<caption>Campeones FIFA (1930 - 2022)</caption>

<thead>

<tr>

<th>Año</th>

<th>Lugar</th>

<th>Campeón</th>

<th>Marcador</th>

<th>Subcampeón</th>

</tr>

</thead>

<tfoot>

<tr>

<td colspan='5'>Última actualización: 2010-9-9</td>

</tr>

</tfoot>

<tbody>

<tr>

<th>1930</th>

<td>Uruguay</td>

<td>Uruguay</td>

<th>4-2</th>

<td>Argentina</td>

</tr>

</tbody>

</table>

</body>

**Javascript**

/\* tableruler() written by Chris Heilmann \*/

function tableruler() {

if (document.getElementById && document.createTextNode){

var tables = document.getElementsByTagName('table');

for (var i = 0; i < tables.length; i++){

if(tables[i].className == 'ruler'){

var trs = tables[i].getElementsByTagName('tr');

for(var j = 0;j < trs.length; j++) {

if(trs[j].parentNode.nodeName == 'TBODY') {

trs[j].onmouseover = function(){

this.className='ruled';

return false;

}

trs[j].onmouseout = function(){

this.className='';

return false;

}

}

} //FOR

}

}

}

}

**CSS**

body{

font-family:Arial,Helvetica,Sans-serif;

font-size:80%;

}

caption{

padding-bottom: 5px;

font-weight: bold;

}

thead th,tfoot td{

background:#ddd;

margin: 10px;

padding: 10px;

}

tr.ruled{

background:#9cf;

margin: 10px;

padding: 10px;

}

table{

border:1px solid #000;

border-collapse:collapse;

padding: 10px;

margin: 0 auto;

}

th,td{

border:1px solid #000;

border-collapse:collapse;

}

#mytable tr.ruled{

background:#333;

color:#9F9;

}