Acceso al DOM

El Documento

Objetos, métodos y propiedades:

Concepto	Definición
Objetos	Cualquier cosa que pueda guardarse en una variable: desde simples datos hasta elementos complejos (arrays, funciones, fechas, etc.)
Métodos	Forma de trabajar: funciones normalmente predefinidas. Para aplicar un método debemos llamar a la función que lo contiene
Propiedades	Elementos que definen o cambian características de un objeto. Puede ser de sólo lectura o de lectura y escritura

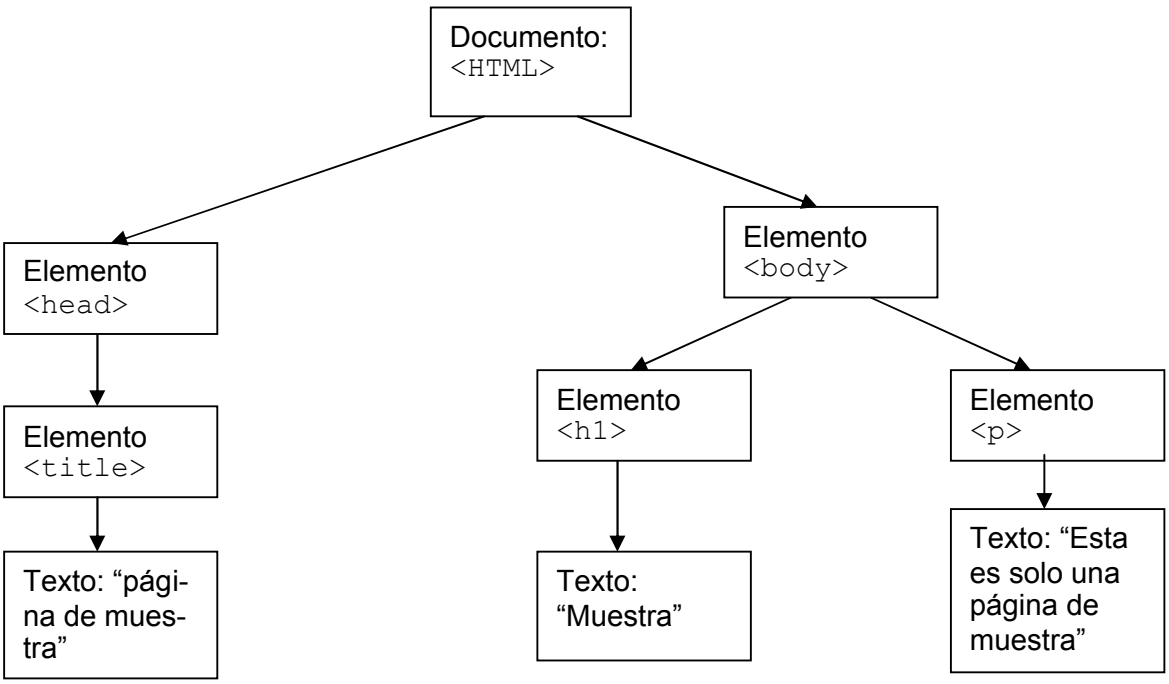
Para acceder a un método o propiedad pondremos el nombre del objeto, seguido de un punto, y después el método o propiedad:

```
n = miarray.lenght;

n = fecha.getDate();
```

Estructura del DOM

Ejemplo de estructura de página sencilla de DOM



El **DOM** es la estructura de la página, la cual está construida de manera **jerárquica** al depender unos elementos de otros.

Cada elemento se denomina **nodo** y estos dependen unos de otros de manera jerárquica, de modo que cada nodo tiene su **nodo padre** y puede tener **nodos hijos**.

Objeto inicial: Es el objeto de que depende todo, este es el objeto **window** que es el navegador. Este se sobreentiende por lo que la página depende del objeto **document**, hijo del anterior, por lo que es este último el que tendremos en cuenta como inicio de la página. El objeto document corresponde a la etiqueta <html>

Tipos de nodos:

Aunque existen 12 tipos de nodos en realidad para manipular las páginas web sólo necesitamos los 5 siguientes:

`document`	Nodo raíz del que derivan los demás
`element`	Cada una de las etiquetas HTML
`attr`	Cada uno de los atributos de las etiquetas HTML
`text`	Texto encerrado en las etiquetas HTML
`comment`	Comentarios de la página HTML

Lectura de elementos de la página:

Para acceder a la lectura de los elementos de la página, los métodos aquí descritos van precedidos siempre de **document.** .

`getElementByTagName(``"etiqueta"`**`)`**	Crea un array con todas las etiquetas marcadas en "etiqueta"
`getElementsByName(``"valor"`**`)`**	Crea un array con todas las etiquetas cuyo atributo `name` sea `"valor"`
`getElementById(``"valor"`**`)`**	Accede al nodo cuya etiqueta `id` sea `"valor"`

Crear una nueva etiqueta

Para crear una nueva etiqueta crearemos los nodos que la contienen: para ello seguiremos los siguientes pasos:

Paso	Código y Explicación:	
1º	Código	**`var`** `escribir ="Nuevo texto."`
	Explicación	Variable con el nuevo texto
2º	Código	**`var`** `etiqueta =` **`document.createElement(`**`"tag"`**`)`**
	Explicación	Crear el nodo de la etiqueta. `"tag"` = etiqueta HTML
3º	Código	**`var`** `texto =` **`document.createTextNode(`**`escribir`**`)`**
	Explicación	Crear el nodo de texto. Podemos insertar una variable con texto o directamente el texto entre comillas
4º	Código	**`var`** `nuevoElemento = etiqueta.` **`appendChild(`**`texto`**`)`**
	Explicación	Insertar el nodo de texto como nodo hijo del nodo etiqueta

Hemos creado un nodo etiqueta que contiene un nodo de texto, es decir una nueva etiqueta con texto, pero ésta no está integrada en la página, sino que la tenemos definida en una variable. Los siguientes pasos son integrar la etiqueta en la página

Insertar o reemplazar un nodo en la página

Después de crear el nodo lo debemos insertar en la página o reemplazarlo por otro ya existente. Una vez creado el nodo siguiendo los cuatro pasos anteriores tenemos varias opciones:

Insertar después de un elemento

Insertar el nodo después de un elemento especificado. Método **`appendChild().`**

Paso	Código y Explicación:	
5º	Código	**`document.getElementById("caja1").appendChild(`**`nuevoElemento`**`)`**
	Explicación	Especificamos el lugar de la página donde queremos introducir el elemento - `document.getElementById("…")` – e insertamos el elemento como hijo del nodo especificado. El nuevo elemento se colocará detrás de los ya existentes dentro de ese nodo.

Insertar antes de un elemento

Insertar el nodo antes de un elemento concreto. Después de crearlo seguiremos con los siguientes pasos. Método **`insertBefore()`**.

Paso	Código y Explicación:	
5º	Código	**`var`** `referencia` **`= document.getElementById("ref")`**
	Explicación	En una variable guardamos el nodo de referencia para insertarlo antes de éste.
6º	Código	**`var`** `padre` **`=`** `referencia`**`.parentNode;`**
	Explicación	En una variable guardamos el nodo padre del nodo de referencia llamado desde el método **`parentNode`**
7º	Código	`padre`**`.insertBefore(`**`nuevoElemento,referencia`**`)`**
	Explicación	Desde el elemento padre insertamos el nuevo elemento mediante el método `insertBefore(… ,… )`; donde el primer elemento del paréntesis será el nuevo elemento a insertar, y el segundo el elemento de referencia.)

Reemplazar un elemento

Para remplazar el nuevo nodo por otro ya existente seguiremos los siguientes pasos. Método `replaceChild()`.

Paso	Código y Explicación:	
5º	Código	**`var`** `viejoElemento` **`= document.getElementById("`**`remplazar`**`")`**
	Explicación	En una variable guardamos el nodo que queremos reemplazar.
6º	Código	**`var padre = viejoElemento.parentNode;`**
	Explicación	Mediante una variable accedemos al elemento padre del nodo que queremos reemplazar
7º	Código	`padre`**`.replaceChild(`**`nuevoElemento,viejoElemento`**`)`**
	Explicación	Desde el elemento padre reemplazamos el nuevo elemento mediante el método `replaceChild(nuevo, viejo )`; donde el primer elemento del paréntesis será el nuevo elemento a insertar, y el segundo el elemento a eliminar.)

Eliminar un nodo

Para eliminar un nodo ya existente, utilizaremos el método **`removeChild()`** con un sólo parámetro: el nodo a eliminar.

Paso	Código y Explicación:	
1º	Código	`var suprimir = document.getElementById("parrafo")`
	Explicación	En una variable guardamos el elemento que queremos eliminar..
2º	Código	`suprimir.parentNode.removeChild(suprimir)`
	Explicación	Llamamos a su elemento padre - `parentNode` - y desde ahi lo remplazamos mediante el método – `removeChild` - sin poner ningún elemento en su lugar.

Cambiar un nodo de sitio

Seguiremos los mismos pasos que para eliminarlo. En los pasos anteriores hemos guardado el nodo en una variable. Esto permite volver a insertarlo en cualquier punto de la página mediante el método **`appendChild()`**

Paso	Código y Explicación:	
1º	Código	`var trasladar = document.getElementById("parrafo")`
	Explicación	guardar en una variable el elemento que se quiere trasladar
2ª	Código	`trasladar.parentNode.removeChild(trasladar)`
	Explicación	Eliminar el elemento que se quiere trasladar
3ª	Código	`document.getElementById("reponer").appendChild(trasladar)`
	Explicación	Colocar el elemento guardado en otro nodo: llamar al otro nodo: - `document.getElementById() -;` colocarlo como hijo. – `appendChild()` -

Copiar un nodo en otro sitio

Para copiar un nodo en otro lugar de la página conservando el original utilizaremos el método **`cloneNode(true)`**.

Paso	Código y Explicación:	
1º	Código	`var elemento = document.getElementById("copia")`
	Explicación	guardar en una variable el elemento que se quiere copiar
2ª	Código	`var copiar = elemento.cloneNode(true)`
	Explicación	Para copiarlo debemos aplicarle el método `cloneNode(true)`, La copia la guardamos en una variable.
3ª	Código	`document.getElementById("padre").appendChild(copiar)`
	Explicación	Procedemos como cuando insertamos un nuevo elemento, colocando el elemento desde su etiqueta padre (también podemos usar el método `insertBefore` para insertarlo antes de otro elemento.

Propiedad innerHTML

La propiedad `innerHTML` permite leer y escribir el contenido de una etiqueta. Lo escrito dentro de la etiqueta es tratado como código HTML, lo que permite escribir directamente etiquetas anidadas

Lectura:

Paso	Código y Explicación:	
1º	Código	**`var lectura = document.getElementById("lee").innerHTML`**
	Explicación	guardar en una variable código HTML del elemento

Escritura:

Paso	Código y Explicación:	
1º	Código	**`document.getElementById("escribe").innerHTML =`** `"<a href='http://www.google.com'>Ir a Google</a>";`
	Explicación	Llamar al elemento de destino - `document.getElementById("escribe")` - aplicar la propiedad - `.innerHTML` - signo igual - `=` - escribir el código nuevo entre comillas. También podemos escribir una variable donde hayamos guardado previamente el código.

Esta operación borra el contenido antiguo que tenía el elemento y lo remplaza por el nuevo.

Si el nuevo contenido es un elemento vacio `""` se borrará el contenido del elemento.

Añadir más contenido a un elemento

Podemos añadir más contenido a un elemento sin borrar el que ya tiene de la siguiente manera:

Paso	Código y Explicación:	
1º	Código	**`var`** `contenido = document.getElementById("elemento");`
	Explicación	Guardamos el elemento en una variable.
2º	Código	`contenido.innerHTML += "<p>Añadir otro párrafo</p>";`
	Explicación	Aplicamos la propiedad **`innnerHTML`** y la suma y asignación **`+=`** del nuevo valor. El nuevo valor asignado se añadirá al ya existente.

Acceso a los atributos

Acceso a atributos: Metodo 1: `.atributo`

El atributo se convierte en una propiedad de su elemento, por lo que no hay más que llamar al elemento y añadirlo mediante - `.atributo` –

Lectura del atributo

Paso	Código y Explicación:	
1º	Código	`var valor = document.getElementById("elemento").href`
	Explicación	Leer el valor del atributo: llamamos al elemento, y le ponemos detrás la referencia del atributo precedida de un punto.

Cambiar o escribir el atributo

Al escribir un atributo con javascript cambiamos el valor que tenía

Paso	Código y Explicación:	
1º	Código	`document.getElementById("navegador").href = "http://yahoo.es/"`
	Explicación	Llamar al nodo que contiene el atributo, un punto y el nombre del atributo. Con el signo igual = le asignamos un nuevo valor, bien sea directamente entre comillas o mediante una variable.

Acceso a atributos: metodo 2: getAttribute() – setAttribute()

Lectura del atributo

Paso	Código y Explicación:	
1º	Código	`var valor = document.getElementById("...").getAttribute("align")`
	Explicación	Llamamos al elemento, y ponemos detrás el método - `getAttribute("...")`-, el parámetro de este método será el nombre del atributo, bien entre comillas o guardado en una variable.

Cambiar o escribir el atributo

Al escribir un atributo con javascript cambiamos el valor que tenía

Paso	Código y Explicación:	
1º	Código	`document.getElementById("...").setAttribute("align","center")`
	Explicación	Llamamos al elemento, y pondemos detrás el método - `setAttribute("... , ...")`, el primer parámetro será el nombre del atributo y el segundo su valor, También podemos escribir variables que contengan estos valores.

Acceso al código CSS

Escritura y reemplazar

La siguiente propiedad reemplaza el código CSS indicado, o lo crea como nuevo si no estaba definido: propiedad `.style.propiedadCSS`

Paso	Código y Explicación:	
1º	Código	`document.getElementById("…").style.fontSize = "1.5em"`
	Explicación	Llamamos al elemento al que queremos aplicar la propiedad; escribimos después - `.style` - seguido de un punto y el nombre de la propiedad CSS; seguimos con el signo igual = y entre comillas después el valor de la propiedad CSS.

Observaciones:

- Si el nombre de la propiedad tiene más de una palabra (separada por guiones en CSS) debe escribirse todo junto y empezando por mayúscula la segunda palabra y siguientes.
- Sólo en el caso de que el valor sea un número (sin indicar medidas) se escribirá sin comillas. Ej : marginLeft = “10px” (con comillas) ZIndex = 2 (sin comillas)

Lectura

- Solo se puede acceder a la lectura de una propiedad CSS mediante este código si previamente se ha escrito con Javascript.

Paso	**Código y Explicación:**	
1º	Código	`Var valor = document.getElementById("…").style.fontSize`
	Explicación	Usando el mismo código anterior podemos guardar en una variable el valor de la propiedad CSS, siempre que éste haya sido escrito antes mediante javascript.