DWEC
TEMA 4

EL DOM Y EL BOM

P E G O
IES Enric Valor

GENERALITAT VALENCIANA
CONSELLERIA D'EDUCACIÓ, CULTURA I ESPORT

Unió Europea
Fons Social Europeu
El FSE inverteix en el teu futur

ÍNDICE

4.1 INTRODUCCIÓN

El llamado DOM (Document Object Model) es un modelo que permite tratar un documento Web XHTML como si fuera XML, navegando por los nodos existentes que forman la página, pudiendo manipular sus atributos e incluso crear nuevos elementos.

Usando Javascript para navegar en el DOM podemos acceder a todos los elementos XHTML de una página. Esto nos permite cambiar dinámicamente el aspecto de nuestras páginas Web.

4.1 INTRODUCCIÓN

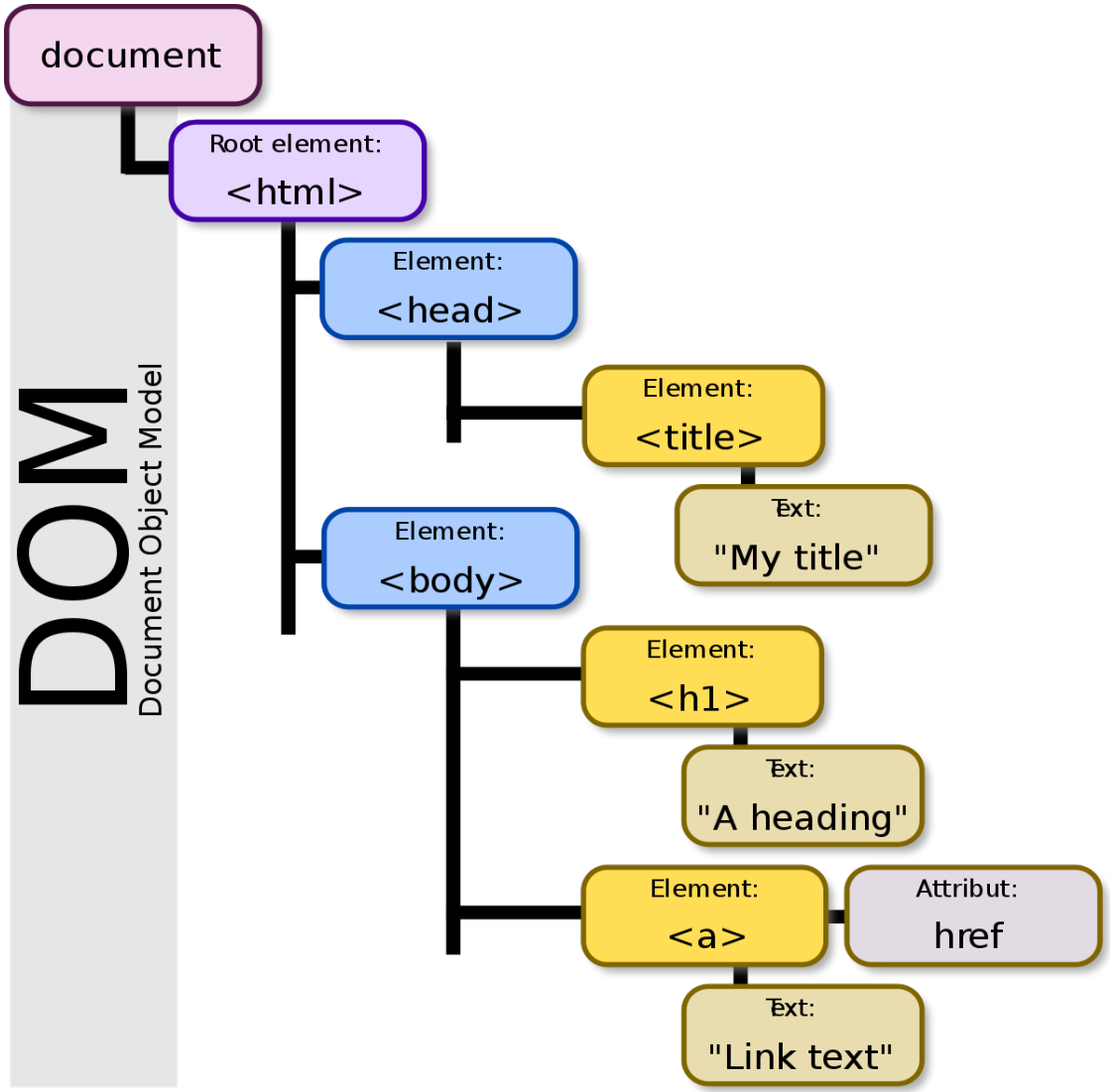
Para más información general de DOM:

https://es.wikipedia.org/wiki/Document_Object_Model

http://www.w3schools.com/js/js_htmldom.asp

https://developer.mozilla.org/es/docs/Web/API/Document_Object_Model/Introduction

4.1 INTRODUCCIÓN



4.2 FUNCIONES PARA LOCALIZAR ELEMENTOS

Cuando obtengamos algún elemento con las funciones que estudiaremos más adelante, podemos manipular los atributos de dicho elemento de la siguiente forma.

```
let elemento=document.getElementById("miElemento");
elemento.innerHTML="El html interno a cambiar de ese elemento";
```

Los atributos disponibles a modificar pueden depender de cada elemento.

4.2 FUNCIONES PARA LOCALIZAR ELEMENTOS

Funciones de Javascript para localizar elementos en el DOM:

- **getElementById(identificador)**

Esta función devuelve un elemento DOM del subárbol cuya identificador sea el indicado en la cadena “identificador”.
http://www.w3schools.com/jsref/met_document_getelementbyid.asp

- **getElementsByTagName(etiqueta)**

Esta función devuelve una array con todos los elementos DOM del subárbol cuya etiqueta XHTML sea la indicada en la cadena “etiqueta”.
http://www.w3schools.com/jsref/met_document_getelementsbytagname.asp

4.2 FUNCIONES PARA LOCALIZAR ELEMENTOS

Funciones de Javascript para localizar elementos en el DOM:

- **getElementsByName(nombre)**

Esta función devuelve una array con todos los elementos DOM del subárbol cuya atributo name sea el indicado en la cadena “nombre”.
http://www.w3schools.com/jsref/met_doc_getelementsbyname.asp

- **document.querySelector(selectores)**

Devuelve el primer elemento del documento (utilizando un recorrido primero en profundidad pre ordenado de los nodos del documento) que coincida con el grupo especificado de selectores.
Selectores es una cadena de caracteres que contiene uno o más selectores CSS separados por coma

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

4.2 FUNCIONES PARA LOCALIZAR ELEMENTOS

Ejemplo de document.querySelector(selectores) Parte 1

```
<!DOCTYPE html>
<html>
<body>

<h3 class="ejemplo">introduzca una expresi&oacute;n num&eacute;rica</h3>

<label for="texto">Introducir datos:</label>
<input type="text" id="texto">

<button>Ejecutar expresi&oacute;n</button>

<p class="ejemplo">Un p&aacute;rrafo de la clase ejemplo.</p>

<p>otro p&aacute;rrafo.</p>

<button onclick="laFuncion()">Cambiar color</button>

<script>
function laFuncion() {
    document.querySelector(".ejemplo").style.backgroundColor = "red";
    document.querySelector("p").innerHTML = "rojo";
}
```

4.2 FUNCIONES PARA LOCALIZAR ELEMENTOS

Ejemplo de document.querySelector(selectores)

Parte 2.

```
var input = document.querySelector('input');
var boton = document.querySelector('button');
var parrafo = document.querySelector('p');
boton.onclick = function() {
    var dato = input.value;
    parrafo.innerHTML = eval(dato);
}

</script>

</body>
</html>
```

Reproducir el ejemplo y ver el funcionamiento.

4.3 EVENTOS DEL DOM

Ya se ha estudiado en este curso como aplicar eventos al documento HTML. En este tema vamos a profundizar más en estos comceptos.
JavaScript puede reaccionar a cualquier evento que ocurra en una página web.
Para separar el JS del html en su totalidad debemos de utilizar el método addEventListener()

```
<!DOCTYPE html>
<html>
<body>
<button id="unboton">Dale</button><p id="texto"></p>

<script>
function MuestraFecha() {
        document.getElementById("texto").innerHTML = Date();
}
document.getElementById("unboton").addEventListener("click",MuestraFecha);
</script>

</body>
</html>
```

4.4 FUNCIONES PARA CREAR / ELIMINAR NODOS

Nodos del DOM

Cuando el navegador recibe la página, la interpreta y va creando una estructura arborescente con los elementos recibidos llamados DOM.

NOTA: hasta que el navegador no haya construido totalmente el árbol (DOM) cargándose la página por completo, no será posible acceder hasta él.

4.4 FUNCIONES PARA CREAR / ELIMINAR NODOS

Nodos del DOM

En el DOM existen muchos tipos de nodos como Attr, CDataSecction, Comment, DocumentFragment…

Generalmente los programadores actuaremos sobre los:

- **Attr.** Representa los atributos de las etiquetas.
- **Document**. El más importante que construye la raíz y de los que derivan los demás nodos.
- **Element**: Cada etiqueta tendrá un nodo tipo element.
- **Text.** En este nodo se almacena el texto de la etiqueta.

4.4 FUNCIONES PARA CREAR / ELIMINAR NODOS

Nodos del DOM

En el DOM existen muchos tipos de nodos como Attr, CDataSecction, Comment, DocumentFragment…

Generalmente los programadores actuaremos sobre los:

- **Attr.** Representa los atributos de las etiquetas.
- **Document**. El más importante que construye la raíz y de los que derivan los demás nodos.
- **Element**: Cada etiqueta tendrá un nodo tipo element.
- **Text.** En este nodo se almacena el texto de la etiqueta.

4.4 FUNCIONES PARA CREAR / ELIMINAR NODOS

Creación de Nodos

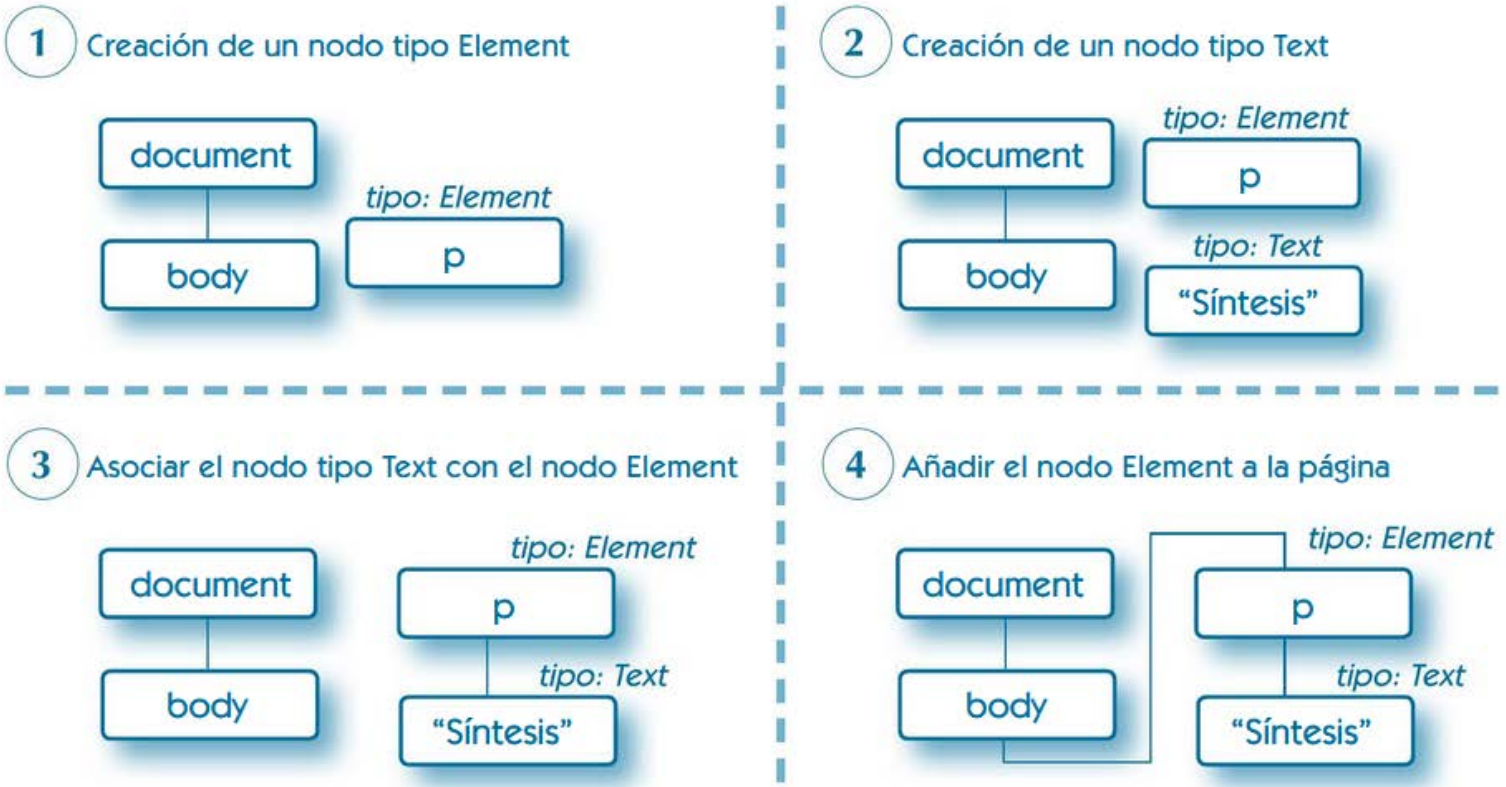
Como se puede ver , los pasos para crear un nuevo nodo son los siguientes :

- **Paso 1.** Crear nuevo nodo de tipo element
- **Paso 2**. Crear un nuevo nodo de tipo texto.
- **Paso 3.** Vincular el nodo terxto a nodo element.
- **Paso 4.** Vincular el nodo Element a la página de tal manera que el nodo esté en el lugar correspondiente.

4.4 FUNCIONES PARA CREAR / ELIMINAR NODOS

Creación de Nodos

Proceso gráfico:



4.4 FUNCIONES PARA CREAR / ELIMINAR NODOS

Creación de Nodos

Código de los pasos de creación del nodo anteriores:

```
// Crear nodo de tipo Element
var parrafo = document.createElement("p");
// Crear nodo de tipo Text
var contenido = document.createTextNode("Síntesis");
// Vincular el nodo Text como hijo del nodo Element
parrafo.appendChild(contenido);
// Vincular el nodo Element como hijo de la página
document.body.appendChild(parrafo);
```

4.4 FUNCIONES PARA CREAR / ELIMINAR NODOS

Eliminación de un nodos

Para eliminar un nodo debemos de realizarlo con la función :

removeChild()

Para eso debemos de situarnos en el padre utilizando la función:

parentNode

Ejemplo:

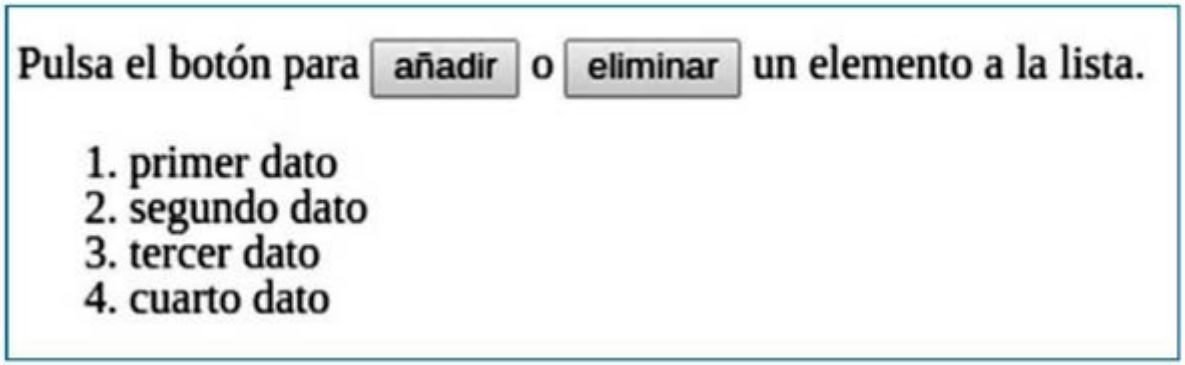
```
<p id="editorial">Editorial S&iacute;ntesis</p>
```

```
var pEditorial = document.getElementById("editorial");
pEditorial.parentNode.removeChild(pEditorial);
```

4.5 EJERCICIO

Ejercicio:

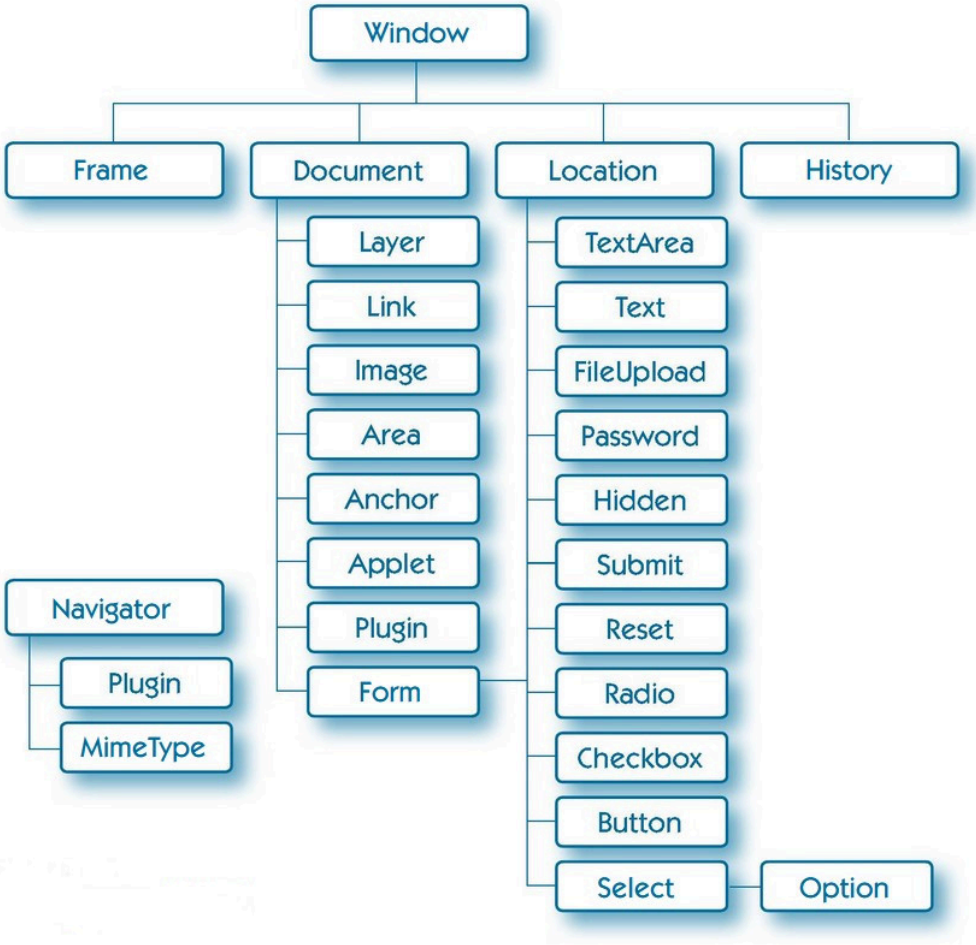
En el siguiente ejercicio se va a mostrar una lista de elementos y dos botones: añadir y eliminar. Pulsando el botón de añadir, se podrá dar de alta un nuevo elemento a la lista y pulsando eliminar , se eliminará el último elemento de la lista.



4.6 BOM

El BOM (Browser Object Model):

Fue implementado por los navegadores para que JS puediese hacer uso de sus métodos y propiedades de forma uniforme



4.6 BOM

El BOM (Browser Object Model):

Como hemos apreciado con el DOM podemos acceder a los elementos del documento, solo del documento, mientas que con el BOM podemos acceder a elementos del navegador.

Ejemplo :

```
<!DOCTYPE html>
<html>
<body>
<p id="dimension"></p>
<script>
var ancho = window.innerWidth;
var alto = window.innerHeight;
document.getElementById("dimension").innerHTML = "Ancho: " + ancho + "
--- Alto: " + alto ;
</script>
</body>
</html>
```

4.6 BOM

El BOM (Browser Object Model):

Los objetos más comunes del BOM son:

- **window :** representa la ventana del navegador en la que se muestra el documento.
- **location:** Devuelve información de la configuración de la página i el navegador.
- **history** : Se puede navegar por el histórico de la navegación igual que se hace como la aplicación del navegador:
  - **history.back()**
  - **history.forward()**
- **navigator**: tiene las siguientes propiedades**:**
  - **navigator.appName**
  - **navigator.appCodeName**
  - **navigator.platform**

https://www.w3schools.com/js/js_window.asp