

# DOM

Modelo de objeto de Documento

# Páginas web dinámicas

DOM permite a los programadores web acceder y manipular las páginas XHTML como si fueran documentos XML. De hecho, DOM se diseñó originalmente para manipular de forma sencilla los documentos XML

Java / PHP / JavaScript

## 5.1. Árbol de nodos

La siguiente página XHTML sencilla:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head>
```

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

```
<title>Página sencilla</title>
```

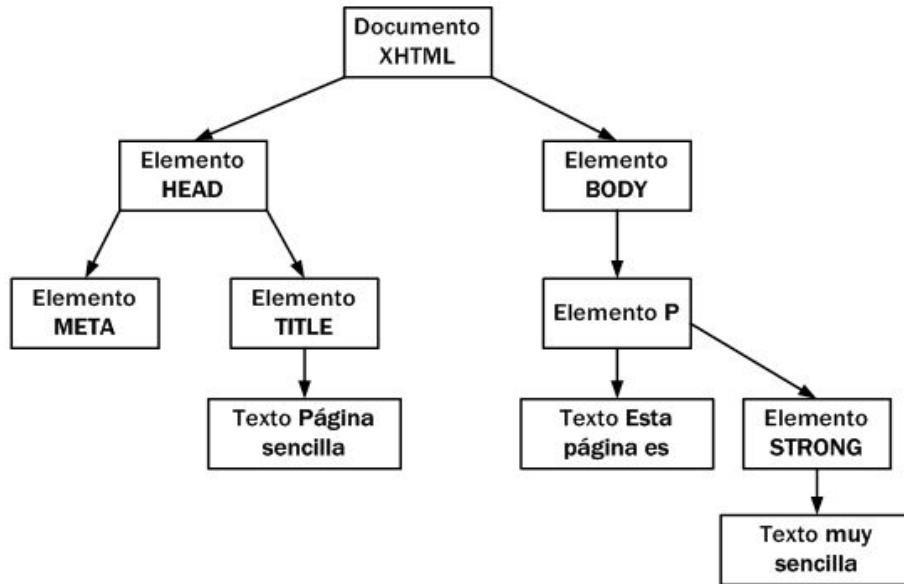
```
</head>
```

```
<body>
```

```
<p>Esta página es <strong>muy sencilla</strong></p>
```

```
</body>
```

```
</html>
```



En el esquema anterior, cada rectángulo representa un nodo DOM y las flechas indican las relaciones entre nodos. Dentro de cada nodo, se ha incluido su tipo (que se verá más adelante) y su contenido.

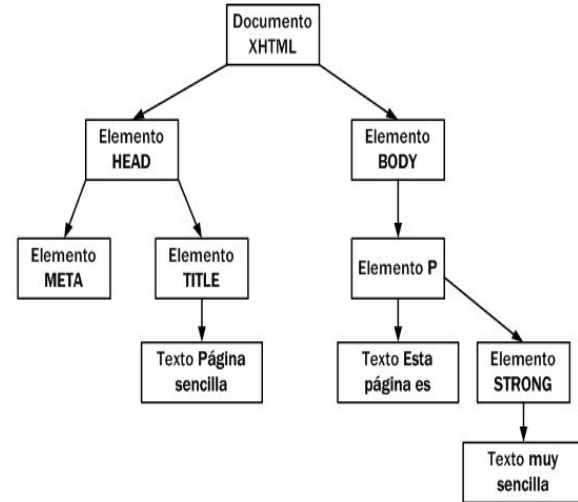
La raíz del árbol de nodos de cualquier página XHTML siempre es la misma: un nodo de tipo especial denominado *"Documento"*.

A partir de ese nodo raíz, cada etiqueta XHTML se transforma en un nodo de tipo "*Elemento*".

La conversión de etiquetas en nodos se realiza de forma jerárquica.

De esta forma, del nodo raíz solamente pueden derivar los nodos HEAD y BODY.

A partir de esta derivación inicial, cada etiqueta XHTML se transforma en un nodo que deriva del nodo correspondiente a su "*etiqueta padre*".



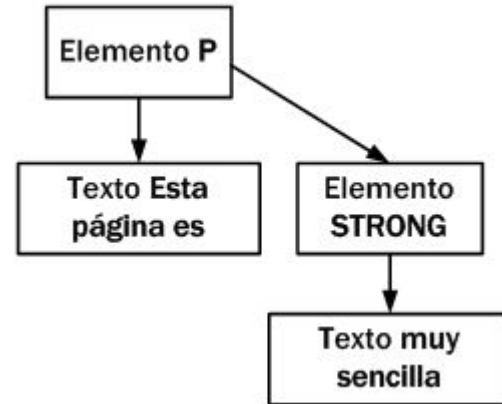
La transformación automática de la página en un árbol de nodos siempre sigue las mismas reglas:

- Las etiquetas XHTML se transforman en dos nodos: el primero es la propia etiqueta y el segundo nodo es hijo del primero y consiste en el contenido textual de la etiqueta.
- Si una etiqueta XHTML se encuentra dentro de otra, se sigue el mismo procedimiento anterior, pero los nodos generados serán nodos hijo de su etiqueta padre.

**<p>Esta página es <strong>muy sencilla</strong></p>**

Genera los siguientes nodos:

- Nodo de tipo *"Elemento"* correspondiente a la etiqueta <p>.
- Nodo de tipo *"Texto"* con el contenido textual de la etiqueta <p>.
- Como el contenido de <p> incluye en su interior otra etiqueta XHTML, la etiqueta interior se transforma en un nodo de tipo *"Elemento"* que representa la etiqueta <strong> y que deriva del nodo anterior.
- El contenido de la etiqueta <strong> genera a su vez otro nodo de tipo *"Texto"* que deriva del nodo generado por <strong>.



## 5.2. Tipos de nodos

- Document, nodo raíz del que derivan todos los demás nodos del árbol.
- Element, representa cada una de las etiquetas XHTML. Se trata del único nodo que puede contener atributos y el único del que pueden derivar otros nodos.
- Attr, se define un nodo de este tipo para representar cada uno de los atributos de las etiquetas XHTML, es decir, uno por cada par atributo=valor.
- Text, nodo que contiene el texto encerrado por una etiqueta XHTML.
- Comment, representa los comentarios incluidos en la página XHTML.

Los otros tipos de nodos existentes que no se van a considerar son DocumentType, CDataSection, DocumentFragment, Entity, EntityReference, ProcessingInstruction y Notation.



## 5.3. Acceso directo a los nodos

No se van a presentar las funciones necesarias para el acceso jerárquico de nodos y se muestran solamente las que permiten acceder de forma directa a los nodos.

Solamente es posible cuando el árbol DOM ha sido construido completamente, es decir, después de que la página XHTML se cargue por completo.

### 5.3.1. `getElementsByTagName()`

### 5.3.2. `getElementByName()`

### 5.3.3. `getElementById()`

## 5.3.1. getElementsByTagName()

La función `getElementsByTagName(nombreEtiqueta)` obtiene todos los elementos de la página XHTML cuya etiqueta sea igual que el parámetro que se le pasa a la función.

El siguiente ejemplo muestra cómo obtener todos los párrafos de una página XHTML:

```
var parrafos = document.getElementsByTagName("p");
```

El valor que se indica delante del nombre de la función (en este caso, `document`) es el nodo a partir del cual se realiza la búsqueda de los elementos. En este caso, como se quieren obtener todos los párrafos de la página, se utiliza el valor **document como punto de partida de la búsqueda**.

El valor que devuelve la función es un array con todos los nodos que cumplen la condición de que su etiqueta coincide con el parámetro proporcionado. **El valor devuelto es un array de nodos DOM**, no un array de cadenas de texto o un array de objetos normales. Por lo tanto, se debe procesar cada valor del array de la forma que se muestra en las siguientes secciones.

De este modo, se puede obtener el primer párrafo de la página de la siguiente manera:

```
var primerParrafo = parrafos[0];
```

De la misma forma, se podrían recorrer todos los párrafos de la página con el siguiente código:

```
for(var i=0; i<parrafos.length; i++) {  
  
    var parrafo = parrafos[i];  
  
}
```

### 5.3.2. `getElementsByName()`

La función `getElementsByName()` es similar a la anterior, pero en este caso se buscan los elementos cuyo atributo `name` sea igual al parámetro proporcionado. En el siguiente ejemplo, se obtiene directamente el único párrafo con el nombre indicado:

```
var parrafoEspecial = document.getElementsByName("especial");
```

```
<p name="prueba">...</p>
```

```
<p name="especial">...</p>
```

```
<p>...</p>
```

### 5.3.3. getElementById()

La función `getElementById()` es la más utilizada cuando se desarrollan aplicaciones web dinámicas. Se trata de la función preferida para acceder directamente a un nodo y poder leer o modificar sus propiedades.

La función `getElementById()` devuelve el elemento XHTML cuyo atributo `id` coincide con el parámetro indicado en la función. Como el atributo `id` debe ser único para cada elemento de una misma página, la función devuelve únicamente el nodo deseado.

```
var cabecera = document.getElementById("cabecera");
```

```
<div id="cabecera">
```

```
  <a href="/" id="logo">...</a>
```

```
</div>
```

La función `getElementById()` es tan importante y tan utilizada en todas las aplicaciones web, que casi todos los ejemplos y ejercicios que siguen la utilizan constantemente.

Internet Explorer 6.0 también interpreta incorrectamente esta función, ya que devuelve también aquellos elementos cuyo atributo `name` coincida con el parámetro proporcionado a la función.

## 5.4. Creación y eliminación de nodos

### 5.4.1. Creación de elementos XHTML simples

Crear y añadir a la página un nuevo elemento XHTML sencillo consta de cuatro pasos diferentes:

1. Creación de un nodo de tipo `Element` que represente al elemento.
2. Creación de un nodo de tipo `Text` que represente el contenido del elemento.
3. Añadir el nodo `Text` como nodo hijo del nodo `Element`.
4. Añadir el nodo `Element` a la página, en forma de nodo hijo del nodo correspondiente al lugar en el que se quiere insertar el elemento.

De este modo, si se quiere añadir un párrafo simple al final de una página XHTML, es necesario incluir el siguiente código JavaScript:

```
// Crear nodo de tipo Element
```

```
var parrafo = document.createElement("p");
```

```
// Crear nodo de tipo Text
```

```
var contenido = document.createTextNode("Hola Mundo!");
```

```
// Añadir el nodo Text como hijo del nodo Element
```

```
parrafo.appendChild(contenido);
```

```
// Añadir el nodo Element como hijo de la pagina
```

```
document.body.appendChild(parrafo);
```



### 5.4.2. Eliminación de nodos

Afortunadamente, eliminar un nodo del árbol DOM de la página es mucho más sencillo que añadirlo. En este caso, solamente es necesario utilizar la función `removeChild()`:

```
var parrafo = document.getElementById("provisional");
```

```
parrafo.parentNode.removeChild(parrafo);
```

```
<p id="provisional">...</p>
```

La función `removeChild()` requiere como parámetro el nodo que se va a eliminar.

## 5.5. Acceso directo a los atributos XHTML

Los atributos XHTML de los elementos de la página se transforman automáticamente en propiedades de los nodos. Para acceder a su valor, simplemente se indica el nombre del atributo XHTML detrás del nombre del nodo.

El siguiente ejemplo obtiene de forma directa la dirección a la que enlaza el enlace:

```
var enlace = document.getElementById("enlace");
```

```
alert(enlace.href); // muestra http://www...com
```

```
<a id="enlace" href="http://www...com">Enlace</a>
```

En el ejemplo anterior, se obtiene el nodo DOM que representa el enlace mediante la función `document.getElementById()`. A continuación, se obtiene el atributo `href` del enlace mediante `enlace.href`. Para obtener por ejemplo el atributo `id`, se utilizaría `enlace.id`.

Las propiedades CSS no son tan fáciles de obtener como los atributos XHTML. Para obtener el valor de cualquier propiedad CSS del nodo, se debe utilizar el atributo style. El siguiente ejemplo obtiene el valor de la propiedad margin de la imagen:

```
var imagen = document.getElementById("imagen");  
  
alert(imagen.style.margin);
```

```

```

Aunque el funcionamiento es homogéneo entre distintos navegadores, los resultados no son exactamente iguales, como muestran las siguientes imágenes que son el resultado de ejecutar el código anterior en distintos navegadores.