

Guía Teórica N°01

Introducción Teórica – Términos Fundamentales

Ya es hora de seguir profundizando en una de las tecnologías más reconocidas actualmente en el mundo de la Computación, JavaScript.

En las próximas clases estaremos aprendiendo todo lo relacionado con la manipulación directa del Modelo de Objetos del Documento o mejor conocido como *DOM* (Document Object Model). Pero antes de empezar a escribir nuestras primeras líneas de código, debemos conocer algunos términos muy importantes que nos ayudarán a entender a la perfección lo que haremos a nivel de código; ¿qué es el DOM?, ¿qué es un Nodo?, ¿qué tipos de Nodos existen?, entre muchas otras preguntas que aclararemos en esta guía teórica. Así que, ¡VAMOS A ELLO!

Document Object Model:

El Modelo de Objetos del Documento (DOM) es una API de programación para documentos HTML y XML. Define la estructura lógica de los documentos y la forma en que se accede y se manipula un documento. En la especificación DOM, el término "documento" se usa en un sentido amplio; cada vez más, XML se usa como una forma de representar muchos tipos diferentes de información que pueden almacenarse en diversos sistemas, y gran parte de esto se consideraría tradicionalmente como datos en lugar de documentos. Sin embargo, XML presenta estos datos como documentos y el DOM se puede utilizar para gestionar estos datos.

Con el Modelo de objetos de documento, los programadores pueden crear y construir documentos, navegar por su estructura y agregar, modificar o eliminar elementos y contenido.

Todo lo que se encuentre en un documento HTML o XML se puede acceder, modificar, eliminar o agregar utilizando el Modelo de objetos del Documento, con algunas excepciones; en particular, las interfaces DOM para el subconjunto interno y el subconjunto externo aún no se han especificado.

Cabe destacar que el DOM no pertenece a un Lenguaje de Programación en sí, es decir, cada lenguaje implementa sus propias instrucciones para la manipulación directa del documento. Como ya es de saber, en este curso aprenderemos las funciones y métodos que nos brinda JavaScript para cumplir esta tarea.

¿Qué es un Nodo?

Una de las tareas habituales en la programación de aplicaciones web con JavaScript consiste en la manipulación de las páginas web. De esta forma, es habitual obtener el valor almacenado por algunos elementos (por ejemplo, los elementos de un formulario), crear un elemento (párrafos, <div>, etc.) de forma dinámica y añadirlo a la página, aplicar una animación a un elemento (que aparezca/desaparezca, que se desplace, entre otros).


Todas estas tareas habituales son muy sencillas de realizar gracias a DOM. Sin embargo, para poder utilizar las utilidades de DOM, es necesario "*transformar*" la página original. Una página HTML normal no es más que una sucesión de caracteres, por lo que es un formato muy difícil de manipular. Por ello, los navegadores web transforman automáticamente todas las páginas web en una estructura más eficiente de manipular.

Esta transformación la realizan todos los navegadores de forma automática y nos permite utilizar las herramientas de DOM de forma muy sencilla. El motivo por el que se muestra el funcionamiento de esta transformación interna es que condiciona el comportamiento de DOM y, por tanto, la forma en la que se manipulan las páginas.

DOM transforma todos los documentos XHTML en un conjunto de elementos llamados ***nodos***, que están interconectados y que representan los contenidos de las páginas web y las relaciones entre ellos. Por su aspecto, la unión de todos los nodos se llama "***árbol de nodos***".

Árbol de nodos:

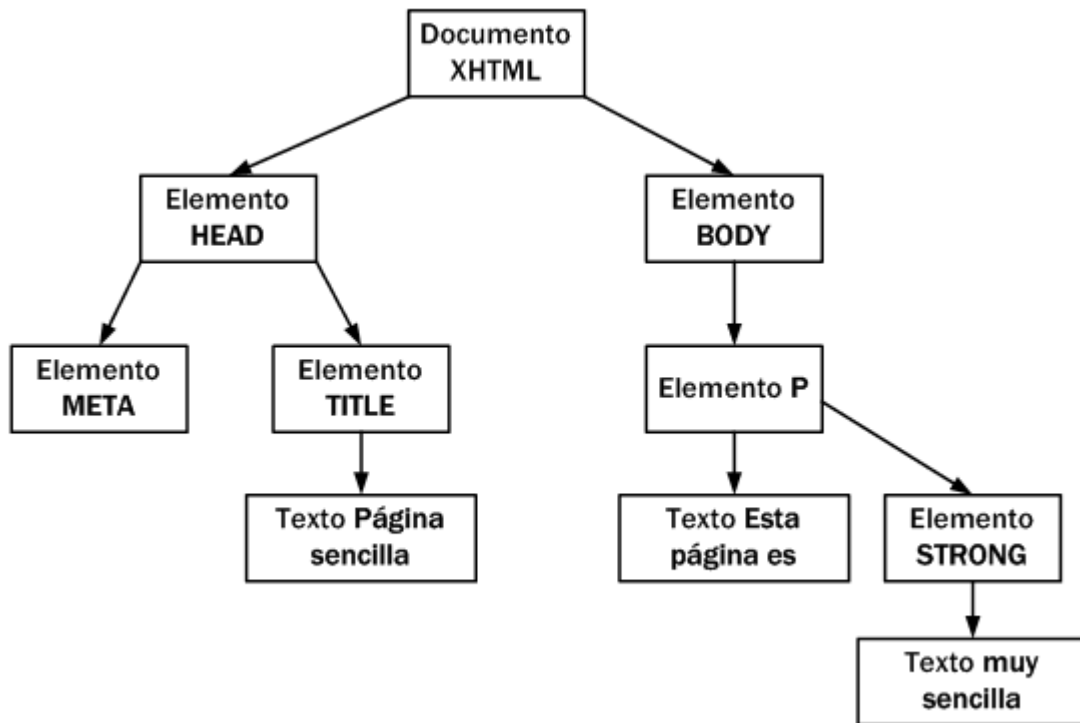
La siguiente página XHTML sencilla:



```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta name="author" content="@educa2ucv"/>
  <title>Página sencilla</title>
</head>

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

Se transforma en el siguiente árbol de nodos:



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 *"Document"*.

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 de las etiquetas XHTML habituales genera dos nodos: el primero es el nodo de tipo "*Elemento*" (correspondiente a la propia etiqueta XHTML) y el segundo es un nodo de tipo "*Texto*" que contiene el texto encerrado por esa etiqueta XHTML.

Tipos de nodos:

La especificación completa de DOM define doce (12) tipos de nodos, aunque las páginas XHTML habituales se pueden manipular manejando solamente cuatro o cinco 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**.

¡Nos vemos en la próxima!

Ha sido mucha teoría, ¿no crees?

Ahora es turno de conocer aquellas instrucciones que nos ofrece JavaScript para la manipulación del Modelo Objetos del Documento. Para ello, en las próximas guías veremos un resumen de cada una de las instrucciones que vayamos aprendiendo en cada clase.