DOM: Document Object Model (Modelo de Objetos del Documento)

El DOM es una representación estructurada de un documento HTML o XML en forma de árbol de nodos. Los nodos pueden ser de diferentes tipos:

- Elementos: Representan las etiquetas HTML. Por ejemplo, <a>, <p>,
- Atributos: Son las características de los elementos HTML. Por ejemplo, id, src, class, name, …
- Texto: Las descripciones de texto dentro un elemento. Por ejemplo dentro de un <p>
- Comentarios: Representan los comentarios del código HTML.
- Document: El nodo raíz que representa todo el documento.

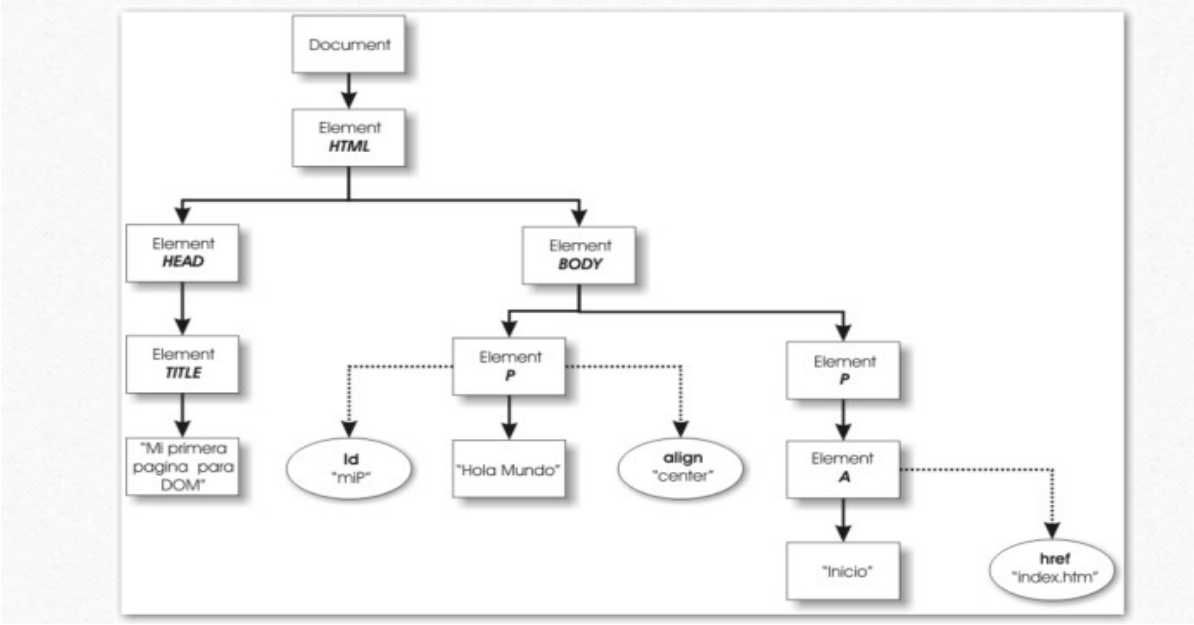
DOM permite a Js acceder y manipular el contenido de una página web de manera dinámica. Proporciona un estándar de cómo obtener, modificar, añadir o eliminar elementos HTML. Por ejemplo, crear un botón nuevo, añadir una fila a una tabla, cambiar el contenido de un etiqueta…Para ello tendremos que manipular los nodos.

IMPORTANTE: El orden de los nodos es crucial

Ejemplo:

```
<HTML>
     <HEAD>
          <TITLE>Mi primera página para DOM</TITLE>
     </HEAD>
     <BODY>
      <P id=”primera” align=”center”>Hola Mundo!</P>
      <P id=”segunda”>
          <A href=”index.htm”>Inicio</A>
      </P>
     </BODY>
</HTML>
```

DOM asociado:



Podemos acceder a los elementos del DOM:

- Por su ID: let myID = document.getElementById(“idMiBoton”);
- Por su etiqueta: let myEnlace = document.getElementsByTagName(“a”)[0];
- Por su name: let myLibro = document.getElementsByName(“libros”)[0];
- Por su nombre de clase let myFilas = document.getElementsByClassName(“row”);

Una vez obtenida la referencia a un nodo, podemos obtener sus propiedades:

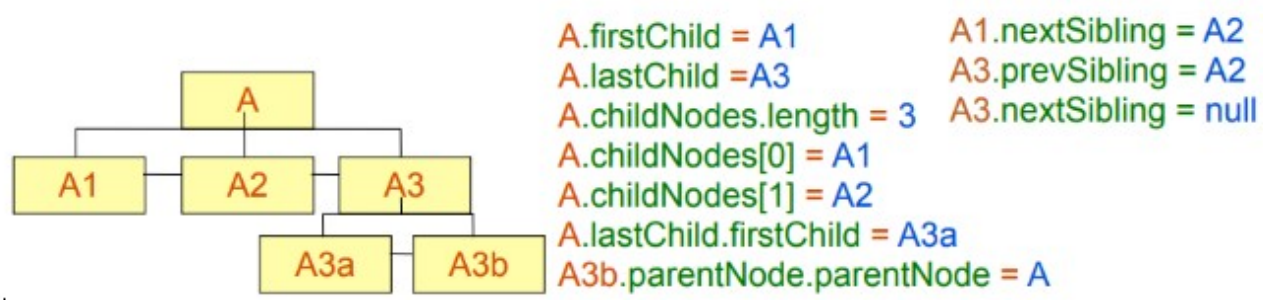
- **nodeType**: constante entera que representa el tipo del nodo
- **nodeName**: nombre
- **nodeValue**: valor

Tipo de nodo	nodeType	nodeName	nodeValue
Etiqueta	1 (Node.ELEMENT_NODE)	Nombre de la etiqueta sin los “<>” y en máyúsc.	null
Texto	3 (Node.TEXT_NODE)	#text	Texto del nodo
Comentario	8 (Node.COMMENT_NODE)	#comment	Texto del comentario
DOCTYPE	10(Node.DOCUMENT_TYPE_NODE)	Nombre de la etiq. raíz del DOCTYPE	null
Documento	9 (Node.DOCUMENT_NODE)	#document	null

Acceso a un nodo desde otro

Cada nodo tiene una seria de propiedades que reflejan el “parentesco” con otros. Algunas de las cuales son:

- **ChildNodes / Children:** Array con los nodos hijos
- **FirstChild / FirstElementChild:** Primer nodo hijo
- **LastChild / lastElementChild**: Último nodo hijo
- **ParentNode / parentElement:** Nodo padre
- **NextSibling / nextElementSibling:** siguiente hermano al mismo nivel
- **PrevSibling / previousElementSibling:** hermano anterior.



Veamos diferencias:

- childNodes vs children
  - **childNodes** devuelve un array con todos los nodos hijos (elementos, texto, comentarios..) sin importar su tipo. Esto significa que en la lista de childNodes encontrarás:
    - Elementos (, <p>, , etc.)
    - Nodos de texto (espacios en blanco, saltos de línea, texto)
    - Comentarios
    - Otros nodos como nodos de procesamiento.
  - **children** devuelve un array con todos los nodos hijos de tipo elemento.

```
//HTML
<div id="miElemento">
                Texto
                  <p>Un párrafo</p>
                  <!-- Un comentario -->
                  <span>Un span</span>
</div>

// JavaScript
var miElemento = document.getElementById('miElemento');

console.log(miElemento.childNodes);   // ¿cuántos nodos tiene?
console.log(miElemento.children);   // Muestra una HTMLCollection con 2 nodos:  <p> y <span>
```

- firstChild vs firstElementChild
  - **firstChild**: primer hijo de un nodo
  - **firstElementChild**: primer hijo de tipo Element.
- lastChild vs lastElementChild
  - **lastChild**: último hijo de un nodo
  - **lastElementChild**: último hijo de tipo Element.
- previousSibling vs previousElementSibling
  - **previousSibling**: anterior hijo de un nodo
  - **previousElementSibling**: anterior hijo de tipo Element.
- nextSibling vs nextElementSibling
  - **nextSibling**: siguiente hijo de un nodo
  - **nextElementSibling**: siguiente hijo de tipo Element.
- parentNode vs parentElement
  - **parentNode**: padre del nodo
  - **parentElement**: padre del nodo de tipo Element.

Cuidado con los espacios en blanco: se interpretan como nodos de texto.

Modificar el contenido o atributos de un elemento: setAttribute, nodeValue, innerHTML, textContent

```
<input type="button" value="Enviar" id="b1">
<p id="parrafo" >TEXTO</p>


<script language="JavaScript">
    let p = document.getElementById("parrafo");
    let boton = document.getElementById(“b1”);
    boton.setAttribute(“value”, “Enviado”);     // modifico el atributo value del botón
    p.setAttribute("align","right");              // modifico el atributo align de la etiqueta <p>
    p.firstChild.nodeValue="He cambiado el texto del p";
    p.innerHTML=”Otro texto”;
    p.textContent =”Nuevo texto”;
</script>
```

NOTA: innerHTML interpreta el contenido HTML, mientras que textContent trata el contenido como texto plano.

Crear nuevos nodos

- **document.createElement(“etiqueta”):** Crea un nodo etiqueta. Se le pasa el nombre de la etiqueta sin “<>”.
- **document.createTextNode(“texto”):** Crea un nodo de texto, con el contenido especificado.

Modificar nodos

Una vez creado un nodo es imprescindible agregarlo al DOM. Se puede insertar de varias maneras:

- padre.**appendChild**(nuevoHijo) : Añade al final de todos los hijos actuales del padre el nuevoHijo.
- padre.**insertBefore**(nuevoHijo, hijoReferencia): Inserta el nuevoHijo justo antes del hijoReferencia.

Para eliminar un nodo se puede hacer de dos formas:

- padre.**removeChild**(hijoABorrar): Se borra el hijoABorrar que pertenece al padre.
- nodo.remove(): Se elimina directamente sin referenciar al padre.

Para reemplazar un nodo hijo existente por un nuevo nodo:

- padre.**replaceChild**(nuevoHijo, hijoAntiguo): reemplaza un hijo por otro nuevo

IMPORTANTE: Hay que insertar los nodos creados en el lugar apropiado del árbol:

```
<body id="cuerpo">
<script language="JavaScript">
  var par = document.createElement("p");
  var texto = document.createTextNode("Yo antes no existía!");
  par.appendChild(texto);
  document.getElementById("cuerpo").appendChild(par);
</script>
</body>
```

¿Qué hace este código?

Recordatorio:

```
let formularios = document.forms;

let imagenes = document.images;

let links = document.links;

let body = document.body;
```

Ejercicios

1. Dado el siguiente código:

```
<div id="myDIV">
        <p>First p element</p>
        <p>Second p element</p>
</div>
```

   - Poner el fondo del primer <p> de color amarillo.
   - Poner el fondo del segundo <p> de color naranja.
   - Poner el color rojo a todos los elementos p

2. Dado el siguiente código, añadir un tercer elemento <p> al final, con el texto: “Soy el nuevo” y añadirle un id.

```
<div id="div1">
        <p id="p1">Soy el primer parrafo</p>
        <p id="p2">Soy el segundo parrafo</p>
</div>
```

3. Añadir otro elemento <p> con el texto: “Soy el primero ahora”, delante del primer p.
4. Dado este código, eliminar el primer elemento:

```
<p>Soy el primero ahora</p>
<p id="p1">Soy el primer parrafo</p>
<p id="p2">Soy el segundo parrafo</p>
<p id="p3">Soy el nuevo</p>
```

5. Dado el código anterior, reemplazar el primer <p> por otro elemento <p> con el texto: ‘reemplazado’.
6. Crear una página HTML vacía, sólo con la etiqueta body. Añadir un párrafo (<p>) con un texto a la página HTML.
7. Crear 3 párrafos con sus id individuales. Desde JavaScript borrar el segundo de los párrafos.
8. Crear una página HTML con un enlace con con color: #1a73e8 y con letra negrita (Font-weight:bold). Mostrar un mensaje alert con el contenido de su atributo href, color y font-weight.
9. Dada una página HTML con un párrafo <p>, crear una clase de estilo llamado parrafo1 con fondo de color a #f0f0f0 y con el texto centrado. Asignarle la clase de estilo al parrafo desde JavaScript.

10.- Dada la siguiente lista en html: Añadir un botón que al pulsarse añada un elemento nuevo li a la lista.

```
<ul id="lista">
        <li>Lorem ipsum dolor sit amet</li>
        <li>Consectetuer adipiscing elit</li>
        <li>Sed mattis enim vitae orci</li>
        <li>Phasellus libero</li>
        <li>Maecenas nisl arcu</li>
</ul>
```

11.- Realizar los apartados siguientes creando todo el código desde JavaScript:
A. Dado un array con las estaciones del año, crear una lista con cada uno de los valores del array. Crear también un titulo en h2 ‘Estaciones del año’. (Utilizar createElement).
B. Dado un array con los continentes, crear ahora la lista utilizando innerHTML.

12.- Crear una página HTML con 3 párrafos con <p> con los siguientes id: ‘contenidos_1’, ‘contenidos_2’ y ‘contenidos_3’. Detrás de cada párrafo, poner un enlace <a> con el texto ‘Ocultar Contenido’ y con sus id: ‘enlace_1’, ‘enlace_2’ y ‘enlace_3’.
Realizar un programa que desde código JavaScript, al pulsar un enlace se ocultará el texto del párrafo correspondiente. Cuando se oculte, el texto del enlace deberá mostrar ‘Mostrar Contenido’ y al pulsarlo, se visualizará el párrafo volviendo a mostrar el enlace el texto ‘ocultar contenido’.

13.- Crear una página HTML con un botón dentro de una etiqueta section (<section id="ContentFormulario">) de tal forma que al pulsarlo, desde JavaScript, se genere un formulario. Dicho formulario debe tener los siguientes atributos: una anchura de 300px, un action a la página de google y el method será get. Además, el formulario debe contener:

- Un input de tipo text para el nombre, con el atributo placeholder ‘Nombres’ y estilo: width:100%;margin: 10px 0px;padding: 5px
- Un input de tipo text para los apellidos, con el atributo placeholder ‘Apellidos’ y estilo: width:100%;margin: 10px 0px;padding: 5px
- Un input de tipo text para el email, con el atributo placeholder ‘Email’ y estilo: width:100%;margin: 10px 0px;padding: 5px
- Un input de tipo text para el asunto, con el atributo placeholder ‘Asunto’ y estilo: width:100%;margin: 10px 0px;padding: 5px
- Un input de tipo text para el Mensaje, con el atributo placeholder ‘Mensaje’ y estilo: width:100%; height:200px;margin: 10px 0px;padding: 5px
- Un botón con el valor ‘Enviar’ con estilo width:100px;margin: 10px 0px; padding: 10px; background: #F05133; color:#fff; border: solid 1px #000; y con un mensaje de alert cuando se pulse el botón.

14.- Crear una página HTML con una lista de 10 elementos (ejemplo lorem), y un botón. Al pulsarlo, preguntar al usuario con mensajes de prompt, un texto a introducir y una posición del 1 al 10. Colocar el texto en la posición indicada. (Actualiza el texto de ese elemento, no se modifica la longitud de la lista). Si la posición no es correcta, indicadlo.

15.- Dada una página con un botón y dos párrafos de texto, poner el fondo del segundo párrafo en color rojo desde código JavaScript. LOS PARRAFOS NO TIENEN ID.

16.- Crear una tabla desde JavaScript con 2 filas y dos columnas. El texto de cada celda será: ‘Posición: ij’, donde i es la fila y la j la columna. La primera columna tendrá un fondo de color rojo, la segunda columna tendrá un fondo amarillo y el texto de esta columna está en negrita.

17.- A partir del siguiente código HTML:

```
<h1>Tabla HTML dibujada con JS</h1>
<table>
    <thead>
        <tr>
            <th>Nombre</th>
            <th>Precio</th>
            <th>Código</th>
        </tr>
    </thead>
    <tbody id="cuerpoTabla">
    </tbody>
</table>
```

Crear desde javascript las filas (tr) y columnas (td) de dicha tabla dentro del cuerpo de la tabla (tbody) de la siguiente forma: Crear un **array** de 4 elementos, donde cada elemento será un **objeto literal** con los campos: id, nombre, precio y código. Cada objeto literal tendrá los datos de una fila. El id del objeto tendrá que asignárselo al id de la fila (<tr>).

18.- Sobre el ejercicio anterior, cuando pulsemos el botón, añadir desde javascript una columna nueva, que será un checkbox. Cada checkbox tiene un id diferente: “checkbox1”, “checkbox2”,… y un name con el valor “marcar.”

19.- Continuamos con el ejercicio anterior. En la página HTML, debajo de la tabla, habrá 3 botones:

• Uno que seleccione todos los checkbox de la tabla
• Otro que deseleccione todos los checks marcados de la tabla
• Otro que elimine aquellas filas que estén marcadas con el checkbox.

20.- Continuamos con el ejercicio anterior. Encima de la tabla, incluir el siguiente código HTML:

```
<div>
    <input type="text" id="nombreN" placeholder="Por favor escriba el nombre del producto">
    <input type="text" id="precioN" placeholder="Por favor escriba su precio">
    <input type="text" id="codigoN" placeholder="Por favor escriba su codigo">
    <input type="button" id="add" value="Añadir">
</div>
```

Los 3 input de tipo text permitirán poder almacenar el nombre, el precio y el código de un nuevo producto. También hay un botón que cuando se pulse, cree una nueva fila en la tabla con los datos introducidos.

21.- Continuamos con el ejercicio anterior para que cuando se pase por una fila de la tabla, ponga su fondo de color amarillo.

22.- Continuamos con el ejercicio anterior para que el producto sea un enlace, cuyo atributo href sea ‘#’.