JAVASCRIPT

DOM/Eventos

DOM

- El DOM es una jerarquía de objetos que describen los elementos de la página web que está mostrando el navegador.
- El DOM permite a los programadores web acceder y manipular las páginas HTML.
- Para poder utilizar las capacidades del DOM es necesario transformar en algo visual la sucesión de caracteres que conforman el código HTML.
- DOM transforma todos los documentos HTML en un conjunto de elementos llamados nodos.
- Estos interconectados y que representan los contenidos de las páginas web y las relaciones entre ellos.

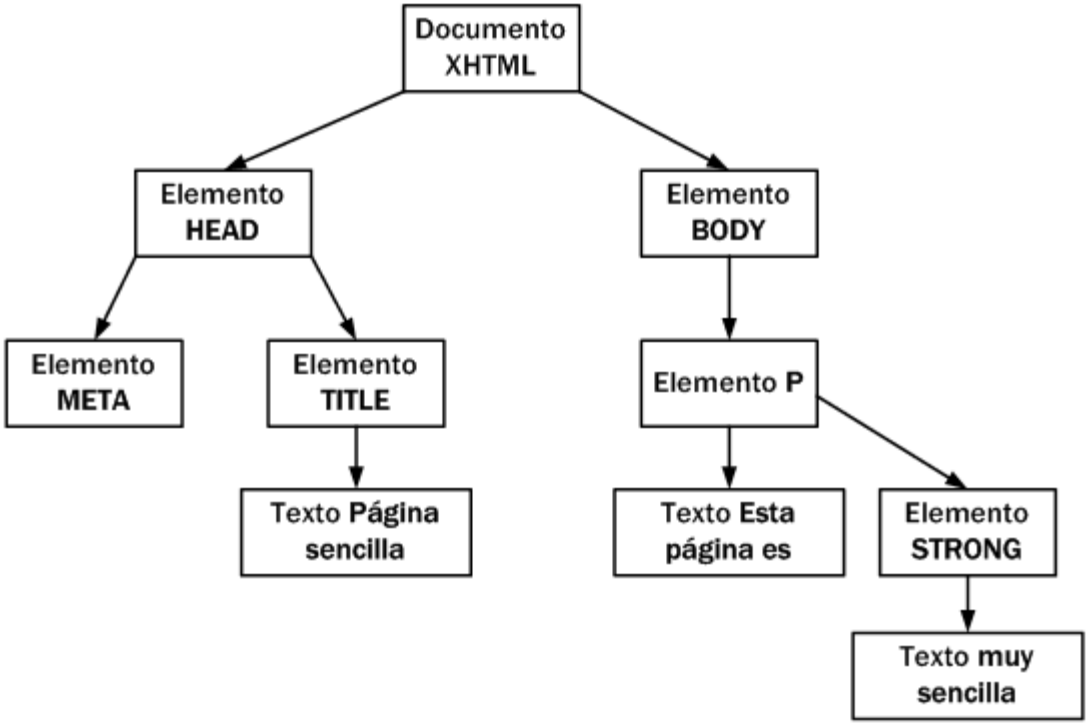
DOM: Ejemplo (I)

- Ejemplo:

```
<html>
<head>
<title>Página sencilla</title>
<meta charset="UTF-8"/>
</head>

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

DOM: Ejemplo (II)



DOM: Estructura

- La raíz del árbol de nodos de cualquier página HTML siempre es la misma:
  - Un nodo de tipo especial denominado **window** (Ventana) que hace referencia al navegador con otro nodo hijo llamado:
  - **document** (Documento o página web).
- A partir de ese nodo raíz, cada etiqueta HTML se transforma en un nodo de tipo "Elemento".
- La conversión de etiquetas en nodos se realiza de forma jerárquica.

Acceso a los nodos del DOM: **getElementsByTagName(etiqueta)**

- Esta función obtiene un array todos los elementos de la etiqueta que se especifica.
- El siguiente ejemplo muestra cómo obtener todos los párrafos de una página HTML:
- var parrafos =document.getElementsByTagName("p");
- Se puede obtener el primer párrafo de la página de la siguiente manera:

```
const parrafos
=document.getElementsByTagName('p');
const primerParrafo = parrafos[0];
```

Acceso a los nodos del DOM
getElementsByClassName(clase)

- Esta función es similar a la anterior, pero en este caso se buscan los elementos cuyo atributo **class** sea igual al parámetro proporcionado.
- Puede ser muy útil cuando queremos hacer una captura de elementos de la misma clase.

Acceso a los nodos del DOM
getElementById(id)

- Permite acceder directamente a un nodo y poder leer o modificar sus propiedades.
- Devuelve el elemento HTML cuyo atributo **id** coincide con el parámetro indicado en la función.

Acceso a los nodos del DOM
querySelector(selector css) y querySelectorAll(selector css)

- **querySelector(selector css):** Selecciona un único elemento basándose en el selector css que especificamos.
- **querySelectorAll(selector css):** Selecciona todos los elementos basándose en el selector css que especificamos.

Creación y eliminación de nodos (I)

- Crear y añadir a la página un nuevo elemento HTML sencillo consta de cuatro pasos diferentes:
  - Creación de un nodo de tipo **Element** que represente al elemento.
  - Darle características.
  - **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 dentro del DOM.

Creación y eliminación de nodos (II)

- **createElement(etiqueta):** Crea un nodo de tipo Element que representa al elemento HTML cuya etiqueta se pasa como parámetro.
- **nodoPadre.appendChild(nodoHijo):** Añade un nodo como hijo de otro nodo.

Creación y eliminación de nodos (III)

- La eliminación es más sencilla.
- Podemos invocar al método **remove** de un nodo.
- Otra manera es mediante la función **removeChild(elemento)**.
- Esta función requiere como parámetro el nodo que se va a eliminar y debe ser invocada desde el elemento padre de ese nodo que se quiere eliminar.

Eventos (I)

- Hasta ahora, todas las aplicaciones y scripts que hemos visto tienen algo en común: Se ejecutan desde la primera instrucción hasta la última de forma secuencial.
- Las aplicaciones web creadas con el lenguaje JavaScript pueden utilizar el modelo de programación basada en eventos.
- JavaScript define numerosos eventos que permiten una interacción completa entre el usuario y las páginas web.

Eventos (II)

- Los eventos hacen posible que los usuarios transmitan información a los programas.
- JavaScript define numerosos eventos que permiten una interacción completa entre el usuario y las páginas web.
- La pulsación de una tecla constituye un evento, así como pinchar o mover el ratón, seleccionar un elemento de un formulario, redimensionar la ventana del navegador, etc.
- JavaScript permite asignar una función a cada uno de los eventos.

Eventos (III)

- El siguiente ejemplo muestra cómo definir un evento click en un botón:

```
<button onclick="calcular()">Haz clic para calcular</button>
```