# 4. Objetos del DOM

## Acceso y cambio de propiedades

Desarrollo Web en entorno cliente

IES Clara del Rey

# El DOM

DOM son las siglas de **Document Object Model** o Modelo de Objetos del Documento. Es un conjunto de objetos que modelizan cada uno de los elementos que aparecen en el documento HTML visible.

Permite controlar el documento de manera global y los distintos elementos que se encuentran en la página, hacer todo tipo de efectos y dinamismos, y responder a las acciones del usuario.

Los desarrolladores Javascript usan el DOM para controlar programáticamente los elementos que hay en la página. Por ejemplo, permite crear e insertar nuevos elementos en la página dinámicamente, hacer que ciertos elementos se reaccionen cuando el usuario interaccione con ellos, etc.

# Acceso a los elementos del DOM (I)

- Por su identificador: **getElementById**

El *id* debe ser único

let elem = document.getElementById('elem');

- Por un selector CSS dado o pseudoclases: **querySelectorAll**

let elements = document.querySelectorAll('ul > li:last-child');

let elements = document.querySelectorAll(':hover');

- El primer elemento de un selector CSS: **querySelector**

Lista de selectores CSS

# Acceso a los elementos del DOM (II)

Permiten buscar nodos por una etiqueta, una clase, etc

Devuelven una colección de elementos que cumplen con la búsqueda

- Por la etiqueta html: **getElementsByTagName(tag)**
- Por una clase CSS: **getElementsByClassName(className)**
- Por el atributo *name*: **getElementsByName(name)**

# Contenido de los objetos HTML

- **innerHTML**: permite obtener el HTML dentro del elemento como un string. También permite modificarlo.

**Cuidado: "innerHTML+=" hace una sobrescritura completa**

- **outerHTML**: contiene el HTML completo del elemento. No cambia el elemento en el que estamos escribiendo, en su lugar, coloca el nuevo HTML.

```
alert(elem.outerHTML); // escribe <div id="elem">Hola <b>Mu
```

- **textContent**: proporciona acceso al texto dentro del elemento: solo texto, menos todas las . Permite escribir texto de "forma segura".
- **innerText**: guarda el contenido sólo texto del elemento y de los descendientes

# Otras propiedades / atributos

- **hidden**: el elemento es visible o no

Funciona igual que style="display:none"

- **value**: para objetos input, select y textarea
- **href**: para <a href="...">
- **checked**

Las propiedades DOM no siempre son strings. Por ejemplo, la propiedad input.checked (para casillas de verificación) es un booleano.

# Estilos CSS

Se puede utilizar **style** como atributo o como propiedad. El atributo *style* es un string, pero la propiedad *style* es un objeto.

```
<div id="div" style="color:red;font-size:120%">Hola</div>

<script>
  // string
  alert(div.getAttribute('style'));
      // devuelve color:red;font-size:120%

  // object
  alert(div.style); // [object CSSStyleDeclaration]
  alert(div.style.color); // red
</script>
```

# Asignación de estilos

Por lo general, hay dos formas de dar estilo a un elemento:

1. Crear una clase css y agregarla: 
2. Escribir las propiedades directamente en style: .

JavaScript puede modificar ambos, clases y las propiedades de style. Es preferible manejar las clases css en lugar de style, excepto si las clases no permiten manejarlo (cálculos dinámicos de posición, por ejemplo)

# Estilos con clases

- **className** y **classList**
- **className** corresponde al atributo “class”. Asignarle un valor reemplaza la cadena de clases.
- **classList** objeto especial con métodos para agregar, eliminar y alternar (add/remove/toggle) una sola clase de la lista.
  - elem.classList.add/remove(“class”): añade o elimina la clase
  - elem.classList.toggle(“class”): la añade si no existe, y si existe la elimina
  - elem.classList.contains(“class”): devuelve true/false si la contiene

# Estilos con propiedades

elem.style es un objeto que corresponde a lo escrito en el atributo "style" del elemento HTML

Para propiedades individuales, con elem.style, si son de varias palabras se usa *camelCase*

- background-color => elem.style.backgroundColor
- z-index => elem.style.zIndex
- border-left-width => elem.style.borderLeftWidth

Para quitar una propiedad tenemos el método
**elem.style.removeProperty('style property')**

# Asignación de propiedades CSS con métodos

- Con **setAttribute(nombre, valor)**

Permite asignar cualquier atributo a un objeto HTML. Para estilos, se usa el atributo *style*

```
div.setAttribute('style', 'color: red...')
```

- Con **setProperty(nombreDePropiedadCSS, valorDeLaPropiedad)**

Asigna o modifica una propiedad de un bloque de estilos CSS

```
r.style.setProperty('color', 'lightblue');
```