Relación D

Manipulación del DOM

Nota

Esta relación practica el acceso y modificación del árbol DOM mediante los métodos modernos de selección (`querySelector`, `querySelectorAll`, `getElementById`, `getElementsByClassName`, `getElementsByTagName`), la creación y modificación de nodos, y la navegación por la jerarquía del documento. Se evita el uso de `innerHTML` salvo donde se indique explícitamente, favoreciendo la creación de nodos con `createElement` y `createTextNode`.

Ejercicio 1. Cree una página HTML estática con los siguientes elementos: un `<h1>` con el texto "Estadística Descriptiva", tres párrafos con texto autogenerado (puede usar Emmet: `p>lorem10`), una lista no ordenada con 4 ítems y un `<div>` contenedor con id `"datos"`. Sin modificar el HTML, use JavaScript para:

- Cambiar el texto del `<h1>` accediendo a él por su etiqueta.
- Cambiar el texto del tercer párrafo usando `querySelector`.
- Añadir la clase `"destacado"` al primer elemento de la lista.
- Establecer el atributo `title` del `div#datos` a "Contenedor principal".

Ejercicio 2. Sobre la página del ejercicio anterior, añada mediante JavaScript (sin modificar el HTML) los siguientes nodos al árbol DOM:

- Un nuevo párrafo al final del `<body>` indicando la fecha y hora actuales (use `new Date()`).
- Un nuevo `<li>` al final de la lista con el texto "Ítem generado dinámicamente".
- Un nuevo `<h2>` **antes** del primer párrafo existente (use `insertBefore`).
- Un elemento `<strong>` que envuelva el texto del primer párrafo (cree el nodo `<strong>`, mueva el nodo de texto dentro de él y sustitúyalo con `replaceChild`).

No use `innerHTML` en este ejercicio.

Ejercicio 3. Cree una página con una tabla HTML de 3 filas y 4 columnas con datos numéricos de ejemplo (como una pequeña tabla de datos estadísticos). Usando JavaScript:

- Acceda a todas las celdas de la segunda fila y cambie su color de fondo.
- Añada una nueva fila al final de la tabla con los totales de cada columna, calculados dinámicamente.
- Añada una columna extra con la media de cada fila.
- Asigne a la primera fila (cabecera) un estilo diferenciado.

Ejercicio 4. Usando `querySelectorAll` y el tipo de dato `NodeList`, cree una página con al menos 8 párrafos, varios de ellos con la clase `"importante"` y otros con la clase `"secundario"`. Escriba código JavaScript que:

- Cuente y muestre por consola el número de párrafos de cada clase.
- Asigne un color de texto diferente a cada clase usando la propiedad `style`.

- Convierta el `NodeList` a array con `Array.from()` y use `filter()` para obtener solo los párrafos que contengan la palabra "estadística" en su texto.
- Muestre en el documento el número de párrafos que coinciden con el filtro anterior.

Ejercicio 5. Cree una página con una lista no ordenada de 5 ítems, cada uno con un enlace dentro. Sin seleccionar ningún elemento directamente por id o clase, acceda desde el primer `<li>` a otros nodos usando únicamente propiedades de navegación: `parentNode`, `childNodes`, `firstChild`, `lastChild`, `nextSibling`, `previousSibling`, `firstElementChild`, `nextElementSibling`, etc. Muestre por consola el `nodeName`, `nodeType` y `textContent` de al menos 6 nodos distintos alcanzados mediante navegación.

Ejercicio 6. Cree una función `generarTabla(datos, contenedor)` que reciba un array de objetos y un elemento del DOM, y genere dentro de ese elemento una tabla HTML completa (con cabecera inferida de las claves del primer objeto y filas para cada objeto). Pruébela con el array de asignaturas del ejercicio 13 de la Relación B. No use `innerHTML`: cree cada elemento con `createElement`.

Ejercicio 7. Cree una página con un `<div>` que contenga una serie de elementos `<p>` y `<span>` mezclados. Escriba JavaScript que recorra todos los nodos hijos del `<div>` (incluyendo nodos de texto) y muestre por consola: el tipo de nodo (`Element` / `Text` / otro), el nombre de la etiqueta si es un elemento, y los primeros 20 caracteres de su contenido de texto. Use la propiedad `nodeType` para distinguir los tipos.

Ejercicio 8. Implemente una función `resaltarTexto(texto, contenedor)` que recorra todos los nodos de texto descendientes del `contenedor` dado y reemplace cada aparición de la cadena `texto` por un elemento `<mark>` que la contenga, de modo que quede visualmente resaltada. Pruébela resaltando una palabra clave en varios párrafos de una página.

Pista: use una función recursiva que explore el árbol DOM y `splitText()` sobre los nodos de texto.

Ejercicio 9. Cree una página con tres `<section>`, cada una con un `<h2>` y dos o tres párrafos. Cree también un `<nav>` vacío. Mediante JavaScript, recorra todas las secciones, asígneles un `id` dinámico (`sec-1`, `sec-2`...) y en el `<nav>` genere automáticamente un enlace (`<a>`) por cada sección apuntando a su `id`, con el texto extraído del `<h2>` correspondiente.

Ejercicio 10. Estudie y demuestre la diferencia entre `textContent`, `innerText` e `innerHTML` creando una página con un párrafo que contenga etiquetas HTML en su interior (por ejemplo, `<strong>` y `<em>`). Con JavaScript, muestre por consola el valor de las tres propiedades. A continuación, modifique el contenido del párrafo usando cada propiedad y observe cómo se interpreta (o no) el HTML. Documente las diferencias en comentarios de código.

Ejercicio 11. Cree una página con un `<div id="galeria">` inicialmente vacío y un array de objetos con información de varias distribuciones de probabilidad (nombre, parámetros, descripción breve). Mediante JavaScript, genere dinámicamente dentro de `#galeria` una "tarjeta" por cada distribución, compuesta de un `<article>` con un `<h3>` para el nombre, un `<dl>` para los parámetros (clave-valor) y un `<p>` para la descripción.

Ejercicio 12. Cree una función `clonarSeccion(id)` que:

- Acceda a la sección con el `id` indicado.
- La clone con `cloneNode(true)`.
- Asigne al clon un nuevo id único.
- Modifique el texto del `<h2>` del clon añadiéndole "(copia)" al final.
- Inserte el clon inmediatamente después del original.

Pruébela en la página del ejercicio 9 llamando a la función para clonar cada una de las tres secciones.

Ejercicio 13. Estudie la diferencia entre `remove()`, `removeChild()` y sustitución con `replaceChild()`. Cree una página con una lista de 6 elementos. Mediante JavaScript y sin eventos (solo código que se ejecute al cargar):

- Elimine el tercer elemento con `remove()`.
- Elimine el último elemento con `removeChild()`.
- Sustituya el primer elemento por un nuevo elemento con `replaceChild()`.

Muestre el estado de la lista por consola antes y después de cada operación.

Ejercicio 14. Cree una página con una tabla HTML estática de 5 filas de datos de estudiantes (nombre y nota) y un botón "Ordenar tabla". Al pulsar el botón, el código JavaScript debe ordenar las filas por nota (de mayor a menor) reordenando los nodos `<tr>` directamente en el DOM, usando `appendChild()` o `insertBefore()` dentro del `<tbody>`. No regenere la tabla desde cero.