REPASO

¿Qué es el DOM?

El Document Object Model (DOM) es una interfaz de programación que representa los documentos HTML y XML como una estructura de árbol. Permite a los lenguajes de programación, como JavaScript, acceder y manipular el contenido, la estructura y el estilo de las páginas web. Las características clave son:

Estructura de Árbol: El DOM organiza los elementos de una página web en una jerarquía de árbol. Cada elemento, atributo y texto se convierte en un nodo de este.

Independiente del Lenguaje: Aunque se usa comúnmente con JavaScript, el DOM es independiente del lenguaje de programación.

Dinámico: Los cambios realizados en el DOM mediante scripts se reflejan inmediatamente en la visualización de la página web.

Interactivo: Permite la interacción del usuario y los scripts, facilitando tareas como la captura de eventos o la modificación de elementos en la página.

Estandarizado: El World Wide Web Consortium (W3C) le da soporte, asegurando la compatibilidad y consistencia entre diferentes navegadores y plataformas.

En JavaScript se puede acceder al `DOM` mediante la variable global `document`.

Una vez aclarado qué es el DOM, dejaremos algo en claro: Existe una propiedad de cada elemento llamada `innerHTML`, la cual modifica el HTML del mismo y no

es considerada buena práctica. Usaremos la propiedad `textContent`, la cual solo permite texto junto con el método `createElement`.

En los siguientes pasos usaremos `métodos` para obtener y/o modificar los elementos del DOM.

createElement: Es un método en JavaScript utilizado para crear un nuevo elemento HTML. Este método es parte del objeto `document` y permite generar dinámicamente elementos en una página web.

querySelector(): `Retorna` el primer elemento que coincida con un grupo específico de `selectores CSS` dentro del documento. Por ejemplo, si quieres seleccionar el primer elemento con la clase `.my-class` en un documento HTML, deberías utilizar `document.querySelector('.my-class')`. Este método es útil para manipular elementos del DOM basado en sus atributos de estilo, identificadores, clases, etc., permitiendo una interacción dinámica en páginas web.

getElementById(): Es otra forma de seleccionar elementos en un documento HTML. Este método también forma parte del objeto `document` y permite acceder a un elemento específico por su atributo id. El id de un elemento es único dentro de una página, por lo que `getElementById` siempre devuelve como máximo un solo elemento.

appendChild: Se usa para agregar un nuevo nodo al final de los hijos de un elemento padre especificado.

innerHTML: Se usa para imprimir contenido HTML en un nodo seleccionado.

¿Qué es una NodeList y cómo puedo iterar una “lista”?

En JavaScript, un **NodeList** es un objeto que representa una colección de nodos del DOM (Document Object Model). Los NodeList son devueltos por métodos como querySelectorAll() y representan una lista (no un array) de nodos que cumplen con el selector proporcionado.

El método forEach() es una función de los NodeList que permite iterar sobre cada uno de los nodos en la lista y ejecutar una función proporcionada para cada uno de ellos. La ventaja de usar forEach() es que proporciona una forma más concisa y legible de realizar operaciones en cada nodo de la NodeList. Ejemplo:

Html:

```
Unset
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejemplo NodeList y forEach</title>
</head>
<body>
    <ul>
        <li>Elemento 1</li>
        <li>Elemento 2</li>
        <li>Elemento 3</li>
    </ul>
    <script src="mi_script.js"></script>
</body>
</html>
```

Javascript (mi_script.js):

```
Unset
// Obtener una NodeList de todos los elementos <li> dentro de un <ul>
var listaItems = document.querySelectorAll('ul li');

// Utilizar forEach para iterar sobre cada elemento de la NodeList
listaItems.forEach(function(item, indice) {
    // La función se ejecutará para cada elemento en la NodeList

    // Ejemplo: Modificar el texto de cada elemento
    item.textContent = 'Nuevo Elemento ' + (indice + 1);

    // Ejemplo adicional: Agregar un evento click a cada elemento
    item.addEventListener('click', function() {
        alert('Hiciste clic en ' + item.textContent);
    });
});
```

En este ejemplo, querySelectorAll('ul li') devuelve un NodeList que contiene todos los elementos <li> dentro de un <ul>. Luego, forEach() se utiliza para iterar sobre cada elemento de la NodeList, cambiando su texto y agregando un evento click a cada uno.

En resumen, NodeList y forEach() son herramientas útiles para trabajar con colecciones de nodos del DOM, permitiéndote realizar operaciones en cada nodo de manera eficiente y legible.

Repaso forEach()

El método forEach es una función de orden superior en JavaScript que te permite iterar con una función **callback** los elementos de un arreglo de una manera eficiente y concisa sin la necesidad de utilizar bucles tradicionales como for o while.

La función evalúa cada elemento del array, desde el primero hasta el último. El método foreach no retorna nada.

A modo de ejemplo:

```
Unset
const numeros = [1, 12, 23, 8, 5, 7, 31];
const doble = numeros.foEach(cadaElemento => console.log(cadaElemento));
// 1
// 12
// 23
// 8
// 5
// 7
// 31
```