



Tecnológico de Monterrey

DOM e IA

Eder Jezrael Cantero Moreno.

Construcción de Software y toma de decisiones

Este es una especie de reporte de investigación respecto al uso de herramientas de IA generativa, lo siguiente es totalmente generado con IA.

Preguntas:

¿Qué es el DOM?

¿Qué es un nodo?

¿Cómo se accede a un elemento del DOM?

¿Cómo se modifica un elemento del DOM?

¿Cuál es la diferencia entre getElementby* y querySelector*?

¿Qué es un evento?

¿Qué es un listener?

¿Qué es un callback?

¿Qué es un objeto de evento?

CHAT GPT:

Reporte: Conceptos Claves de Manipulación del DOM en JavaScript

1. ¿Qué es el DOM?

El DOM (Document Object Model) es una representación estructurada de un documento HTML o XML en forma de árbol de nodos, donde cada nodo representa una parte del documento (elementos, atributos, texto, etc.). Este modelo permite a los lenguajes de programación, como JavaScript, manipular la estructura, estilo y contenido de la página web de manera dinámica.

Fuente: [MDN Web Docs - DOM](#)

2. ¿Qué es un nodo?

Un nodo es cada unidad individual dentro del DOM. Puede representar un elemento HTML, un atributo, un texto, un comentario o incluso el propio documento.

Ejemplo de nodos:

Unset

```
<p id="parrafo">Hola, <strong>mundo</strong></p>
```

En este ejemplo hay tres nodos:

1. `<p>` (nodo elemento)
2. `"Hola, "` (nodo de texto)

3. **** (nodo elemento) con su propio nodo de texto "mundo"

Fuente: [MDN Web Docs - Node](#)

3. ¿Cómo se accede a un elemento del DOM?

Se pueden utilizar distintos métodos para acceder a un elemento del DOM:

Ejemplo:

JavaScript

```
let elemento = document.getElementById("miElemento"); //  
Accede por ID  
let parrafo = document.querySelector("p");           //  
Accede al primer <p>  
console.log(parrafo.innerText);
```

Fuente: [MDN Web Docs - Document](#)

4. ¿Cómo se modifica un elemento del DOM?

Un elemento del DOM se puede modificar cambiando su contenido, estilos o atributos.

Ejemplo:

JavaScript

```
let titulo = document.getElementById("titulo");  
titulo.innerText = "Nuevo título"; // Cambia el texto  
titulo.style.color = "red";          // Cambia el color  
titulo.setAttribute("class", "destacado"); // Agrega una  
clase
```

Fuente: [MDN Web Docs - Element.innerText](#)

5. ¿Cuál es la diferencia entre `getElementBy*` y `querySelector*`?

Método	Retorna	Soporte de selectores
<code>getElementById("id")</code>	Un único elemento	Solo por ID
<code>getElementsByName("nombre")</code>	HTMLCollection (array-like)	Solo por nombre
<code>getElementsByClassName("clase")</code>	HTMLCollection (array-like)	Solo por clase
<code>getElementsByTagName("etiqueta")</code>	HTMLCollection (array-like)	Solo por etiqueta
<code>querySelector("selector")</code>	El primer elemento que coincide	Soporta selectores CSS
<code>querySelectorAll("selector")</code>	NodeList (array-like)	Soporta selectores CSS

Ejemplo:

JavaScript

```
document.querySelector(".miClase"); // Devuelve el primer  
elemento con clase 'miClase'  
document.querySelectorAll("p");      // Devuelve todos los  
<p>
```

Fuente: [MDN Web Docs - querySelector](#)

6. ¿Qué es un evento?

Un evento es una acción que ocurre en una página web (clics, teclas presionadas, carga de página, etc.) y que puede ser detectada y manejada con JavaScript.

Ejemplo:

JavaScript

```
document.getElementById("boton").addEventListener("click",  
function() {  
    alert("¡Botón clickeado!");  
});
```

Fuente: [MDN Web Docs - Event](#)

7. ¿Qué es un listener?

Un listener (o event listener) es una función que escucha un evento y se ejecuta cuando dicho evento ocurre.

Ejemplo:

JavaScript

```
document.getElementById("miBoton").addEventListener("mouseover", function() {  
    console.log("El mouse pasó sobre el botón.");  
});
```

Fuente: [MDN Web Docs - EventTarget.addEventListener](#)

8. ¿Qué es un callback?

Un callback es una función que se pasa como argumento a otra función y se ejecuta después de que esa función termine su trabajo.

Ejemplo:

JavaScript

```
function procesarDatos(callback) {  
    console.log("Procesando datos...");  
    callback();  
}  
  
procesarDatos(function() {  
    console.log("Datos procesados.");  
});
```

```
});
```

Fuente: [MDN Web Docs - Callback function](#)

9. ¿Qué es un objeto de evento?

Un objeto de evento es un objeto especial que contiene información sobre un evento, como el tipo de evento, el elemento que lo generó y otras propiedades útiles.

Ejemplo:

JavaScript

```
document.getElementById("inputTexto").addEventListener("key  
press", function(evento) {  
    console.log("Tecla presionada: " + evento.key);  
});
```

En este caso, **evento.key** almacena la tecla presionada.

Fuente: [MDN Web Docs - Event](#)

COPILLOT:

Reporte sobre el DOM y conceptos relacionados

¿Qué es el DOM?

El **DOM (Document Object Model)** es una interfaz de programación para los documentos HTML y XML. Representa la estructura del documento como un árbol de nodos, donde cada nodo corresponde a una parte del documento (como un elemento, atributo o texto). El DOM permite a los lenguajes de programación manipular la estructura, estilo y contenido del documento de manera dinámica.

¿Qué es un nodo?

Un **nodo** es la unidad básica del DOM. Puede representar diferentes partes del documento, como:

- **Elementos:** Etiquetas HTML como `<div>`, `<p>`, etc.
- **Atributos:** Propiedades de los elementos, como `class`, `id`, etc.
- **Texto:** El contenido textual dentro de los elementos.
- **Comentarios:** Comentarios en el código HTML.

¿Cómo se accede a un elemento del DOM?

Para acceder a un elemento del DOM, se pueden usar varios métodos. Aquí hay algunos ejemplos:

- **getElementById:** Accede a un elemento por su ID.

Unset

```
const element = document.getElementById('miElemento');
```

- **getElementsByClassName:** Accede a todos los elementos con una clase específica.

Unset

```
const elements = document.getElementsByClassName('miClase');
```

- **getElementsByTagName:** Accede a todos los elementos con una etiqueta específica.

Unset

```
const elements = document.getElementsByTagName('div');
```

- **querySelector:** Accede al primer elemento que coincide con un selector CSS.

Unset

```
const element = document.querySelector('.miClase');
```

- **querySelectorAll:** Accede a todos los elementos que coinciden con un selector CSS.

Unset

```
const elements = document.querySelectorAll('.miClase');
```

¿Cómo se modifica un elemento del DOM?

Para modificar un elemento del DOM, se pueden usar varias propiedades y métodos. Aquí hay algunos ejemplos:

- **Modificar el contenido de texto:**

Unset

```
const element = document.getElementById('miElemento');  
  
element.textContent = 'Nuevo contenido';
```

- **Modificar el HTML interno:**

Unset

```
const element = document.getElementById('miElemento');
```

```
element.innerHTML = '<p>Nuevo contenido HTML</p>';
```

- **Modificar atributos:**

Unset

```
const element = document.getElementById('miElemento');  
  
element.setAttribute('class', 'nuevaClase');
```

- **Modificar estilos:**

Unset

```
const element = document.getElementById('miElemento');  
  
element.style.color = 'red';
```

¿Cuál es la diferencia entre `getElementBy*` y `querySelector*`?

- **`getElementBy*`:** Métodos como `getElementById`, `getElementsByClassName` y `getElementsByTagName` son más antiguos y específicos. `getElementById` devuelve un único elemento, mientras que `getElementsByClassName` y `getElementsByTagName` devuelven una colección de elementos.
- **`querySelector*`:** Métodos como `querySelector` y `querySelectorAll` son más modernos y flexibles, ya que permiten usar selectores CSS. `querySelector` devuelve el primer elemento que coincide con el selector, mientras que `querySelectorAll` devuelve todos los elementos que coinciden.

¿Qué es un evento?

Un **evento** es una acción o suceso que ocurre en el navegador, como un clic del usuario, el envío de un formulario, o la carga de una página. Los eventos permiten a los desarrolladores ejecutar código en respuesta a estas acciones.

¿Qué es un listener?

Un **listener** (o manejador de eventos) es una función que se ejecuta en respuesta a un evento específico. Se puede agregar un listener a un elemento del DOM usando el método `addEventListener`.

Unset

```
const button = document.getElementById('miBoton');

button.addEventListener('click', function() {

    alert('Botón clickeado');

});
```

¿Qué es un callback?

Un **callback** es una función que se pasa como argumento a otra función y se ejecuta después de que se completa una operación. Los callbacks son comunes en la programación asíncrona y en el manejo de eventos.

Unset

```
function saludar(nombre, callback) {

    console.log('Hola ' + nombre);

    callback();

}

saludar('Juan', function() {

    console.log('Callback ejecutado');

});
```

¿Qué es un objeto de evento?

Un **objeto de evento** es un objeto que contiene información sobre el evento que ocurrió. Se pasa automáticamente a la función manejadora de eventos y puede proporcionar detalles como el tipo de evento, el elemento objetivo, y las coordenadas del cursor.

Unset

```
const button = document.getElementById('miBoton');

button.addEventListener('click', function(event) {

    console.log('Tipo de evento: ' + event.type);

    console.log('Elemento objetivo: ' + event.target);

});
```

Fuentes

- [MDN Web Docs sobre el DOM](#)
- [MDN Web Docs sobre eventos](#)

Espero que esta información te sea útil. ¿Hay algo más en lo que pueda ayudarte?