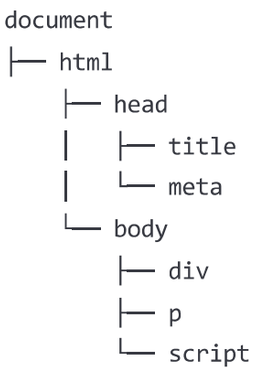
Guía Completa de JavaScript DOM - Estudio y Práctica

Índice

¿Qué es el DOM?

El **DOM (Document Object Model)** es una representación estructurada del documento HTML como un árbol de objetos. JavaScript puede acceder y modificar todos los elementos, atributos y contenido de una página web a través del DOM.

Estructura del DOM



Objeto Document

El objeto `document` es el punto de entrada principal para acceder al DOM.

Propiedades Importantes

```javascript
// Información del documento
console.log(document.title);           // Título de la página
console.log(document.URL);             // URL completa
console.log(document.domain);          // Dominio
console.log(document.documentElement); // Elemento <html>
console.log(document.head);            // Elemento <head>
console.log(document.body);            // Elemento <body>
```

Métodos de Document

javascript

```javascript
// Crear elementos
document.createElement('div');
document.createTextNode('Texto');

// Escribir contenido
document.write('<h1>Hola Mundo</h1>');
document.writeln('<p>Párrafo con salto de línea</p>');
```

Selección de Elementos

Por ID

javascript

```javascript
// Seleccionar por ID (devuelve un elemento o null)
const elemento = document.getElementById('miId');
```

Por Clase

javascript

```javascript
// Seleccionar por clase (devuelve HTMLCollection)
const elementos = document.getElementsByClassName('miClase');
const primerElemento = elementos[0];

// Convertir a array para usar métodos de array
const arrayElementos = Array.from(elementos);
```

Por Etiqueta

javascript

```javascript
// Seleccionar por nombre de etiqueta
const parrafos = document.getElementsByTagName('p');
const todasLasImgs = document.getElementsByTagName('img');
```

Por Atributo Name

javascript

```javascript
// Seleccionar por atributo name (común en formularios)
const inputs = document.getElementsByName('username');
```

Query Selectors (Más Modernos)

javascript

```javascript
// Seleccionar el primer elemento que coincida
const elemento = document.querySelector('#miId');
const primerParrafo = document.querySelector('p');
const primerClase = document.querySelector('.miClase');

// Seleccionar todos los elementos que coincidan
const todosLosP = document.querySelectorAll('p');
const todasLasClases = document.querySelectorAll('.miClase');
```

Selectores CSS Avanzados

javascript

```javascript
// Combinadores
document.querySelector('div > p');        // p hijo directo de div
document.querySelector('div p');          // p descendiente de div
document.querySelector('div + p');        // p hermano adyacente de div
document.querySelector('div ~ p');        // p hermano general de div

// Pseudo-selectores
document.querySelector('p:first-child');
document.querySelector('p:last-child');
document.querySelector('p:nth-child(2)');
document.querySelector('input:checked');
document.querySelector('a:hover');

// Selectores de atributos
document.querySelector('[data-id="123"]');
document.querySelector('input[type="email"]');
document.querySelector('a[href^="https"]');
document.querySelector('img[alt$=".jpg"]');
```

Manipulación de Elementos

Contenido de Elementos

javascript

```javascript
const elemento = document.getElementById('miDiv');

// Texto plano (sin HTML)
elemento.textContent = 'Nuevo texto';
console.log(elemento.textContent);

// HTML interno
elemento.innerHTML = '<strong>Texto en negrita</strong>';
console.log(elemento.innerHTML);

// HTML completo (incluyendo el elemento)
console.log(elemento.outerHTML);

// Texto visible (respeta el CSS)
console.log(elemento.innerText);
```

Atributos

```javascript
const img = document.querySelector('img');

// Obtener atributos
const src = img.getAttribute('src');
const alt = img.getAttribute('alt');

// Establecer atributos
img.setAttribute('src', 'nueva-imagen.jpg');
img.setAttribute('alt', 'Nueva descripción');

// Verificar si existe un atributo
if (img.hasAttribute('title')) {
    console.log('Tiene atributo title');
}

// Eliminar atributos
img.removeAttribute('title');

// Atributos como propiedades (más directo)
img.src = 'otra-imagen.jpg';
img.alt = 'Otra descripción';
```

Clases CSS

```javascript
const elemento = document.querySelector('.miElemento');

// Agregar clase
elemento.classList.add('nuevaClase');
elemento.classList.add('clase1', 'clase2', 'clase3');

// Eliminar clase
elemento.classList.remove('viejaClase');

// Alternar clase (toggle)
elemento.classList.toggle('activo');

// Verificar si tiene una clase
if (elemento.classList.contains('activo')) {
    console.log('El elemento está activo');
}

// Reemplazar clase
elemento.classList.replace('viejaClase', 'nuevaClase');

// Obtener todas las clases
console.log(elemento.classList.toString());
```

Navegación por el DOM

```javascript
const elemento = document.querySelector('#miElemento');

// Elementos padre
console.log(elemento.parentNode);       // Nodo padre (puede ser texto)
console.log(elemento.parentElement);    // Elemento padre

// Elementos hijos
console.log(elemento.childNodes);       // Todos los nodos hijos (incluye texto)
console.log(elemento.children);         // Solo elementos hijos
console.log(elemento.firstChild);       // Primer nodo hijo
console.log(elemento.firstElementChild); // Primer elemento hijo
console.log(elemento.lastChild);        // Último nodo hijo
console.log(elemento.lastElementChild); // Último elemento hijo

// Hermanos
console.log(elemento.nextSibling);      // Siguiente nodo hermano
console.log(elemento.nextElementSibling); // Siguiente elemento hermano
console.log(elemento.previousSibling);  // Nodo hermano anterior
console.log(elemento.previousElementSibling); // Elemento hermano anterior
```

Eventos

Formas de Agregar Eventos

1. Inline (en HTML) - No recomendado

```html
<button onclick="miFuncion()">Clic aquí</button>
```

2. Asignación directa

```javascript
const boton = document.getElementById('miBoton');
boton.onclick = function() {
    console.log('Botón clickeado');
};

// Con arrow function
boton.onclick = () => {
    console.log('Botón clickeado');
};
```

3. addEventListener (Recomendado)

```javascript
const boton = document.getElementById('miBoton');

boton.addEventListener('click', function(event) {
    console.log('Botón clickeado');
    console.log(event); // Objeto del evento
});

// Con arrow function
boton.addEventListener('click', (event) => {
    console.log('Botón clickeado');
});
```

Tipos de Eventos Comunes

Eventos del Mouse

```javascript
elemento.addEventListener('click', handler);      // Clic
elemento.addEventListener('dblclick', handler);   // Doble clic
elemento.addEventListener('mousedown', handler);  // Botón presionado
elemento.addEventListener('mouseup', handler);    // Botón liberado
elemento.addEventListener('mouseover', handler);  // Mouse sobre elemento
elemento.addEventListener('mouseout', handler);   // Mouse fuera del elemento
elemento.addEventListener('mouseenter', handler); // Mouse entra (no burbujea)
elemento.addEventListener('mouseleave', handler); // Mouse sale (no burbujea)
elemento.addEventListener('mousemove', handler);  // Movimiento del mouse
```

Eventos del Teclado

```javascript
elemento.addEventListener('keydown', handler);    // Tecla presionada
elemento.addEventListener('keyup', handler);      // Tecla liberada
elemento.addEventListener('keypress', handler);   // Tecla presionada (deprecated)

// Ejemplo de manejo de teclas
input.addEventListener('keydown', (e) => {
    console.log(e.key);        // Tecla presionada
    console.log(e.code);       // Código de la tecla
    console.log(e.keyCode);    // Código numérico (deprecated)

    if (e.key === 'Enter') {
        console.log('Enter presionado');
    }

    if (e.ctrlKey && e.key === 's') {
        e.preventDefault(); // Prevenir guardar del navegador
        console.log('Ctrl+S presionado');
    }
});
```

Eventos de Formulario

javascript

```javascript
form.addEventListener('submit', handler);
input.addEventListener('change', handler);
input.addEventListener('input', handler);
input.addEventListener('focus', handler);
input.addEventListener('blur', handler);
```

Eventos de Ventana

javascript

```javascript
window.addEventListener('load', handler);      // Página completamente cargada
window.addEventListener('resize', handler);    // Ventana redimensionada
window.addEventListener('scroll', handler);    // Scroll de la página
document.addEventListener('DOMContentLoaded', handler); // DOM cargado
```

Objeto Event

javascript

```javascript
elemento.addEventListener('click', (event) => {
    // Propiedades comunes del evento
    console.log(event.type);           // Tipo de evento ('click')
    console.log(event.target);         // Elemento que disparó el evento
    console.log(event.currentTarget);  // Elemento al que está asociado el listener
    console.log(event.timeStamp);      // Momento del evento

    // Para eventos del mouse
    console.log(event.clientX, event.clientY); // Posición relativa al viewport
    console.log(event.pageX, event.pageY);     // Posición relativa a la página
    console.log(event.screenX, event.screenY); // Posición relativa a la pantalla

    // Métodos del evento
    event.preventDefault();  // Prevenir acción por defecto
    event.stopPropagation(); // Detener propagación del evento
});
```

Event Bubbling y Capturing

javascript

```javascript
// Event Bubbling (por defecto): de hijo a padre
padre.addEventListener('click', () => console.log('Padre'));
hijo.addEventListener('click', () => console.log('Hijo'));
// Al hacer clic en hijo: "Hijo", "Padre"

// Event Capturing: de padre a hijo
padre.addEventListener('click', () => console.log('Padre'), true);
hijo.addEventListener('click', () => console.log('Hijo'), true);
// Al hacer clic en hijo: "Padre", "Hijo"

// Detener propagación
hijo.addEventListener('click', (e) => {
    console.log('Hijo');
    e.stopPropagation(); // Solo se ejecuta este handler
});
```

Delegación de Eventos

javascript

```javascript
// En lugar de agregar listeners a muchos elementos...
const lista = document.getElementById('miLista');

lista.addEventListener('click', (e) => {
    if (e.target.tagName === 'LI') {
        console.log('Clic en item:', e.target.textContent);
    }
});

// Esto funciona incluso para elementos agregados dinámicamente
```

Creación y Eliminación de Elementos

Crear Elementos

javascript

```javascript
// Crear elemento
const nuevoDiv = document.createElement('div');
nuevoDiv.textContent = 'Nuevo contenido';
nuevoDiv.className = 'mi-clase';
nuevoDiv.id = 'nuevo-id';

// Crear nodo de texto
const textoNodo = document.createTextNode('Solo texto');

// Crear fragmento (para múltiples elementos)
const fragmento = document.createDocumentFragment();
for (let i = 0; i < 5; i++) {
    const li = document.createElement('li');
    li.textContent = `Item ${i + 1}`;
    fragmento.appendChild(li);
}
```

Insertar Elementos

javascript

```javascript
const padre = document.getElementById('contenedor');
const nuevoElemento = document.createElement('p');
nuevoElemento.textContent = 'Nuevo párrafo';

// Agregar al final
padre.appendChild(nuevoElemento);

// Insertar antes de un elemento específico
const referencia = document.getElementById('referencia');
padre.insertBefore(nuevoElemento, referencia);

// Métodos modernos de inserción
padre.prepend(nuevoElemento);      // Al principio
padre.append(nuevoElemento);       // Al final
referencia.before(nuevoElemento);  // Antes del elemento
referencia.after(nuevoElemento);   // Después del elemento

// Insertar HTML directamente
padre.insertAdjacentHTML('beforebegin', '<p>Antes del elemento</p>');
padre.insertAdjacentHTML('afterbegin', '<p>Al inicio del contenido</p>');
padre.insertAdjacentHTML('beforeend', '<p>Al final del contenido</p>');
padre.insertAdjacentHTML('afterend', '<p>Después del elemento</p>');
```

Eliminar Elementos

javascript

```javascript
const elemento = document.getElementById('eliminar');

// Método clásico
elemento.parentNode.removeChild(elemento);

// Método moderno
elemento.remove();

// Eliminar todos los hijos
const contenedor = document.getElementById('contenedor');
contenedor.innerHTML = ''; // Rápido pero no remueve listeners

// Forma correcta de eliminar todos los hijos
while (contenedor.firstChild) {
    contenedor.removeChild(contenedor.firstChild);
}
```

Clonar Elementos

```javascript
const original = document.getElementById('original');

// Clon superficial (sin hijos)
const clonSuperficial = original.cloneNode(false);

// Clon profundo (con todos los hijos)
const clonProfundo = original.cloneNode(true);

document.body.appendChild(clonProfundo);
```

Estilos CSS

Estilos Inline

```javascript
const elemento = document.getElementById('miElemento');

// Establecer estilos individuales
elemento.style.color = 'red';
elemento.style.backgroundColor = 'blue';
elemento.style.fontSize = '20px';
elemento.style.marginTop = '10px';

// Establecer múltiples estilos
elemento.style.cssText = 'color: red; background-color: blue; font-size: 20px;';

// Obtener estilos
console.log(elemento.style.color);
```

Estilos Computados

```javascript
const elemento = document.getElementById('miElemento');

// Obtener estilos computados (incluyendo CSS externo)
const estilos = window.getComputedStyle(elemento);
console.log(estilos.color);
console.log(estilos.fontSize);
console.log(estilos.marginTop);

// Con pseudo-elementos
const estilosBefore = window.getComputedStyle(elemento, ':before');
console.log(estilosBefore.content);
```

Propiedades de Dimensión y Posición

```javascript
const elemento = document.getElementById('miElemento');

// Dimensiones del elemento
console.log(elemento.offsetWidth);    // Ancho total (incluye padding y border)
console.log(elemento.offsetHeight);   // Alto total
console.log(elemento.clientWidth);    // Ancho sin border
console.log(elemento.clientHeight);   // Alto sin border
console.log(elemento.scrollWidth);    // Ancho del contenido scrolleable
console.log(elemento.scrollHeight);   // Alto del contenido scrolleable

// Posición del elemento
console.log(elemento.offsetLeft);     // Posición X relativa al offsetParent
console.log(elemento.offsetTop);      // Posición Y relativa al offsetParent
console.log(elemento.offsetParent);   // Elemento padre posicionado

// Scroll
console.log(elemento.scrollLeft);     // Cantidad de scroll horizontal
console.log(elemento.scrollTop);      // Cantidad de scroll vertical

// Establecer scroll
elemento.scrollLeft = 100;
elemento.scrollTop = 200;

// Scroll suave
elemento.scrollTo({
    top: 100,
    left: 0,
    behavior: 'smooth'
});
```

Formularios

Acceso a Formularios

```javascript
// Por índice
const formulario = document.forms[0];

// Por nombre o ID
const formulario = document.forms['miFormulario'];
const formulario = document.forms.miFormulario;

// Con querySelector
const formulario = document.querySelector('#miFormulario');
```

Elementos de Formulario

javascript

```javascript
const formulario = document.getElementById('miFormulario');

// Acceder a elementos por name
const input = formulario.elements['username'];
const input = formulario.elements.username;

// Por índice
const primerInput = formulario.elements[0];

// Todos los elementos
console.log(formulario.elements.length);
```

Valores de Inputs

javascript

```javascript
// Text, email, password, etc.
const textInput = document.getElementById('texto');
console.log(textInput.value);
textInput.value = 'Nuevo valor';

// Checkbox
const checkbox = document.getElementById('acepto');
console.log(checkbox.checked); // true/false
checkbox.checked = true;

// Radio buttons
const radios = document.querySelectorAll('input[name="genero"]');
radios.forEach(radio => {
    if (radio.checked) {
        console.log('Seleccionado:', radio.value);
    }
});

// Select
const select = document.getElementById('pais');
console.log(select.value);
console.log(select.selectedIndex);
console.log(select.options[select.selectedIndex].text);

// Select múltiple
const selectMultiple = document.getElementById('hobbies');
const seleccionados = Array.from(selectMultiple.options)
    .filter(option => option.selected)
    .map(option => option.value);
```

Validación de Formularios

```javascript
const formulario = document.getElementById('miFormulario');
const email = document.getElementById('email');

// Validación HTML5
console.log(email.validity.valid);        // ¿Es válido?
console.log(email.validity.valueMissing); // ¿Está vacío?
console.log(email.validity.typeMismatch); // ¿Tipo incorrecto?
console.log(email.validationMessage);     // Mensaje de error

// Validación personalizada
email.addEventListener('input', () => {
    if (email.value.includes('@')) {
        email.setCustomValidity(''); // Válido
    } else {
        email.setCustomValidity('Debe contener @');
    }
});

// Evento de envío
formulario.addEventListener('submit', (e) => {
    if (!formulario.checkValidity()) {
        e.preventDefault(); // Prevenir envío si no es válido
        console.log('Formulario inválido');
    }
});
```

Ejercicios Prácticos

Ejercicio 1: Lista de Tareas

Objetivo: Crear una lista de tareas interactiva

```html
<!-- HTML base -->
<div id="todo-app">
    <input type="text" id="todo-input" placeholder="Nueva tarea...">
    <button id="add-btn">Agregar</button>
    <ul id="todo-list"></ul>
</div>
```

```javascript
// Tu código aquí
const input = document.getElementById('todo-input');
const addBtn = document.getElementById('add-btn');
const list = document.getElementById('todo-list');

function addTodo() {
    const text = input.value.trim();
    if (text === '') return;

    const li = document.createElement('li');
    li.innerHTML = `
        <span>${text}</span>
        <button class="delete-btn">Eliminar</button>
    `;

    list.appendChild(li);
    input.value = '';
}

addBtn.addEventListener('click', addTodo);
input.addEventListener('keypress', (e) => {
    if (e.key === 'Enter') addTodo();
});

list.addEventListener('click', (e) => {
    if (e.target.classList.contains('delete-btn')) {
        e.target.parentElement.remove();
    }
});
```

Ejercicio 2: Cambiar Tema

Objetivo: Implementar cambio de tema claro/oscuro

```javascript
const themeToggle = document.getElementById('theme-toggle');
const body = document.body;

themeToggle.addEventListener('click', () => {
    body.classList.toggle('dark-theme');

    const isDark = body.classList.contains('dark-theme');
    themeToggle.textContent = isDark ? '☀️' : '🌙';

    // Guardar preferencia
    localStorage.setItem('theme', isDark ? 'dark' : 'light');
});

// Cargar tema guardado
const savedTheme = localStorage.getItem('theme');
if (savedTheme === 'dark') {
    body.classList.add('dark-theme');
    themeToggle.textContent = '☀️';
}
```

Ejercicio 3: Formulario de Contacto

Objetivo: Validar y procesar formulario

```javascript
```

```
const form = document.getElementById('contact-form');
const inputs = form.querySelectorAll('input, textarea');

inputs.forEach(input => {
    input.addEventListener('blur', validateField);
    input.addEventListener('input', clearError);
});

function validateField(e) {
    const field = e.target;
    const value = field.value.trim();

    clearError(e);

    if (field.hasAttribute('required') && value === '') {
        showError(field, 'Este campo es obligatorio');
        return false;
    }

    if (field.type === 'email' && !isValidEmail(value)) {
        showError(field, 'Ingrese un email válido');
        return false;
    }

    return true;
}

function showError(field, message) {
    field.classList.add('error');
    let errorDiv = field.nextElementSibling;
    if (!errorDiv || !errorDiv.classList.contains('error-message')) {
        errorDiv = document.createElement('div');
        errorDiv.className = 'error-message';
        field.parentNode.insertBefore(errorDiv, field.nextSibling);
    }
    errorDiv.textContent = message;
}

function clearError(e) {
    const field = e.target;
    field.classList.remove('error');
    const errorDiv = field.nextElementSibling;
    if (errorDiv && errorDiv.classList.contains('error-message')) {
        errorDiv.remove();
    }
}

function isValidEmail(email) {
    return /^[^\s@]+@[^\s@]+\.[^\s@]+$/.test(email);
}

form.addEventListener('submit', (e) => {
    e.preventDefault();

    let isValid = true;
    inputs.forEach(input => {
```

```
        if (!validateField({target: input})) {
            isValid = false;
        }
    });

    if (isValid) {
        console.log('Formulario válido - procesar envío');
        // Aquí procesarías el formulario
    }
});
```

Proyectos de Práctica

Proyecto 1: Calculadora

Funcionalidades:

- Operaciones básicas (+, -, *, /)
- Historial de operaciones
- Teclas del teclado

Proyecto 2: Galería de Imágenes

Funcionalidades:

- Visualización de thumbnails
- Modal para imagen completa
- Navegación con flechas del teclado

Proyecto 3: Quiz Interactivo

Funcionalidades:

- Preguntas de opción múltiple
- Puntuación en tiempo real
- Resultado final con estadísticas

Proyecto 4: Dashboard Personal

Funcionalidades:

- Widgets arrastrable
- Guardar configuración en localStorage
- Diferentes tipos de contenido (reloj, clima, notas)

Consejos y Mejores Prácticas

Performance

1. **Minimizar accesos al DOM**: Cachea referencias a elementos
2. **Usar DocumentFragment**: Para múltiples inserciones
3. **Event Delegation**: Para muchos elementos similares
4. **Debounce/Throttle**: Para eventos frecuentes como scroll/resize

Buenas Prácticas

1. **Separar HTML, CSS y JS**: Mantén el código organizado
2. **Usar addEventListener**: En lugar de propiedades onclick
3. **Validar elementos**: Verificar que existen antes de usarlos
4. **Nombres descriptivos**: Para variables y funciones
5. **Comentar código complejo**: Especialmente manipulaciones DOM complejas

Debugging

1. **console.log()**: Para verificar valores
2. **Inspeccionar elementos**: DevTools del navegador
3. **Breakpoints**: En el debugger del navegador
4. **console.dir()**: Para explorar objetos DOM

Recursos Adicionales

Referencias Online

- MDN Web Docs: https://developer.mozilla.org/
- W3Schools: https://www.w3schools.com/js/
- JavaScript.info: https://javascript.info/

Herramientas

- DevTools del navegador
- VSCode con extensiones de JavaScript
- Linters como ESLint

Práctica

- Codepen.io: Para experimentos rápidos
- JSFiddle: Para compartir código
- GitHub: Para proyectos completos

¡Practica estos conceptos creando proyectos reales y experimentando con diferentes técnicas de manipulación del DOM!