Manipulación del DOM: selección y manipulación JavaScript

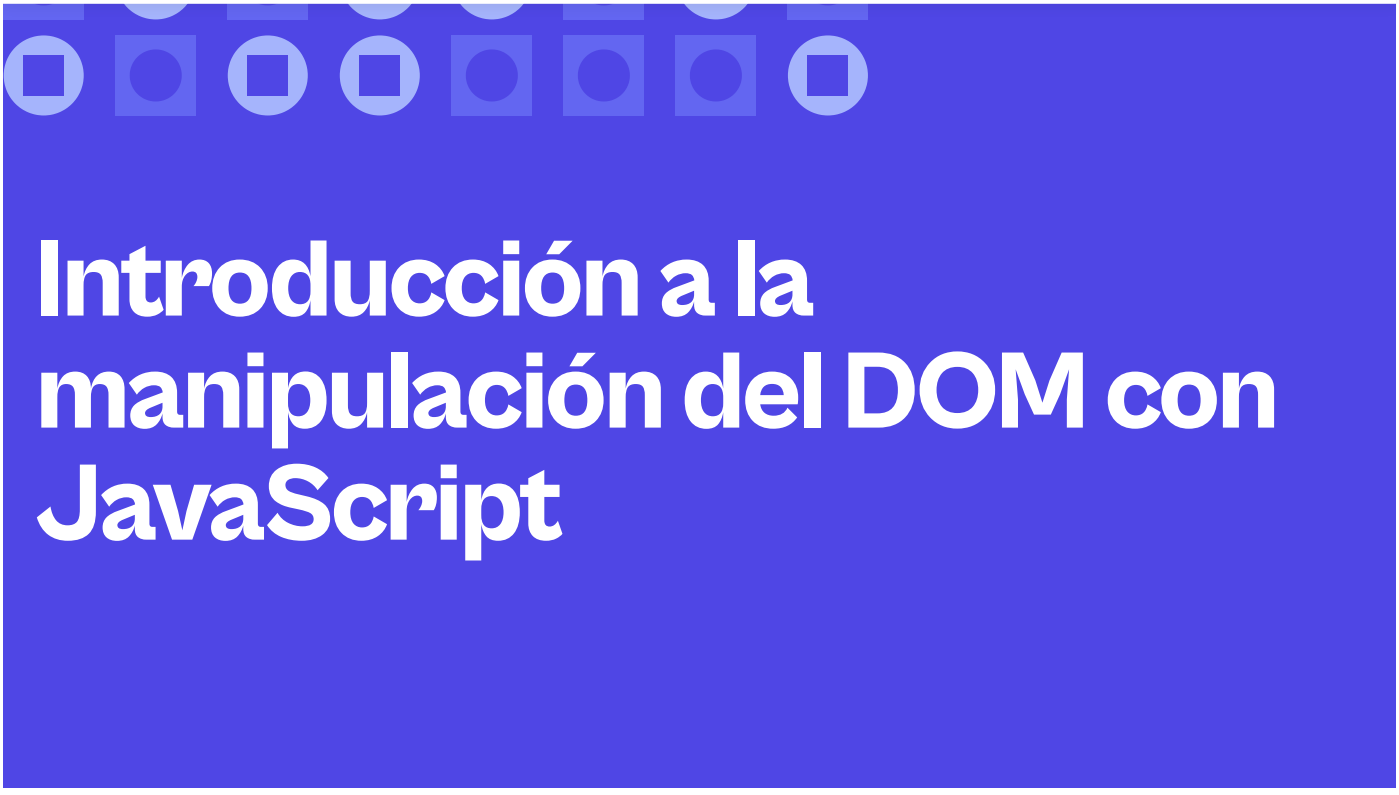
Aprende cómo manipular el DOM utilizando JavaScript

Empezar

Descripción general

En este curso aprenderás cómo seleccionar y manipular elementos en el Document Object Model (DOM) utilizando JavaScript. Descubrirás cómo acceder a elementos HTML, cambiar su contenido, estilos y atributos, así como también cómo crear y eliminar elementos dinámicamente. ¡Conviértete en un experto en la manipulación del DOM!

01 Introducción



01 | Introducción a la manipulación del DOM con JavaScript

¿Qué es el DOM?

El Document Object Model (DOM) es una representación de un documento HTML o XML como un árbol de objetos. Cada elemento, atributo y texto en el documento es representado como un nodo en dicho árbol. El DOM proporciona una interfaz para acceder y manipular estos nodos, lo que permite a los desarrolladores web interactuar con los elementos de una página web y modificar su contenido, estructura y apariencia.

¿Por qué manipular el DOM con JavaScript?

Manipular el DOM con JavaScript nos brinda una forma poderosa de hacer cambios dinámicos en una página web. Podemos crear, eliminar, modificar y mover elementos, así como cambiar su estilo y contenido en respuesta a eventos o acciones del usuario. Esto nos permite crear experiencias interactivas y personalizadas en nuestros sitios web.

Selección de elementos del DOM

Para poder manipular elementos del DOM con JavaScript, primero necesitamos seleccionarlos. Podemos seleccionar elementos de varias formas:

- **Por etiqueta**: Podemos seleccionar todos los elementos que tienen una etiqueta específica, como `<div>`, `<p>`, `<h1>`, etc.
- **Por id**: Cada elemento en el DOM puede tener un atributo `id` único. Podemos seleccionar un elemento por su id utilizando el método `getElementById()`.
- **Por clase**: Podemos seleccionar todos los elementos que tienen una clase específica utilizando el método `getElementsByClassName()`.
- **Por selector CSS**: Podemos utilizar selectores CSS para seleccionar uno o varios elementos. Podemos utilizar los métodos `querySelector()` o `querySelectorAll()` para realizar estas selecciones.

Manipulación de elementos del DOM

Una vez que hemos seleccionado los elementos del DOM que deseamos manipular, podemos utilizar métodos y propiedades para cambiar su contenido, estilo y estructura.

- **Cambiar contenido**: Podemos cambiar el contenido de un elemento modificando su propiedad `innerHTML` o `textContent`. También podemos agregar o eliminar nodos hijos utilizando métodos como `appendChild()` o `removeChild()`.
- **Cambiar estilo**: Podemos cambiar el estilo de un elemento modificando sus propiedades CSS utilizando la propiedad `style`.
- **Modificar atributos**: Podemos cambiar los atributos de un elemento utilizando sus propiedades correspondientes, como `src` para un elemento `<img>`, `href` para un elemento `<a>`, etc.
- **Agregar o eliminar clases**: Podemos agregar o eliminar clases de un elemento utilizando los métodos `classList.add()` y `classList.remove()`.

Manipulación de eventos

Además de manipular el contenido y el estilo del DOM, también podemos manipular eventos, como hacer clic en un botón, mover el mouse sobre un elemento, etc. Podemos utilizar JavaScript para registrar funciones que se ejecutarán cuando ocurra un evento determinado. Esto nos permite hacer que nuestras páginas web respondan de manera interactiva a las acciones del usuario.

Conclusión - Introducción a la manipulación del DOM con JavaScript

En esta introducción a la manipulación del DOM con JavaScript, hemos aprendido los conceptos básicos necesarios para acceder y manipular elementos en el DOM. Hemos visto cómo utilizar la sintaxis de JavaScript para seleccionar elementos del DOM utilizando selectores y cómo modificar el contenido y los atributos de estos elementos. Con estas técnicas, podemos crear interacciones dinámicas en nuestras páginas web y mejorar la experiencia del usuario.

Selección de elementos del DOM utilizando selectores en JavaScript

Introducción

En JavaScript, la selección de elementos del DOM es esencial para poder interactuar y manipular la estructura y contenido de una página web. Para lograr esto, JavaScript proporciona una amplia variedad de selectores que permiten identificar y acceder a elementos específicos del DOM de manera fácil y eficiente.

En este tema, exploraremos los diferentes tipos de selectores disponibles en JavaScript y veremos cómo utilizarlos correctamente para seleccionar los elementos deseados en el DOM.

Tipos de selectores en JavaScript

1. Selectores por etiqueta

Los selectores por etiqueta permiten seleccionar elementos basados en el nombre de su etiqueta HTML. Por ejemplo, si queremos seleccionar todos los elementos `p` en el DOM, podemos utilizar el selector `getElementsByTagName` de la siguiente manera:

```
const elementosP = document.getElementsByTagName('p');
```

2. Selectores por clase

Los selectores por clase permiten seleccionar elementos basados en su atributo `class`. Si queremos seleccionar todos los elementos con la clase `miClase`, podemos utilizar el selector `getElementsByClassName` de la siguiente manera:

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

3. Selectores por ID

Los selectores por ID permiten seleccionar elementos basados en su atributo `id`. Si queremos seleccionar el elemento con el ID `miElemento`, podemos utilizar el selector `getElementById` de la siguiente manera:

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

4. Selectores por atributo

Los selectores por atributo permiten seleccionar elementos basados en un atributo específico. Por ejemplo, si queremos seleccionar todos los elementos que

tienen el atributo `data-toggle`, podemos utilizar el selector `querySelectorAll` de la siguiente manera:

```
const elementosAtributo = document.querySelectorAll('[data-toggle]');
```

5. Selectores por selector CSS

Los selectores por selector CSS permiten seleccionar elementos utilizando cualquier selector CSS válido. Por ejemplo, si queremos seleccionar todos los elementos `input` dentro de un formulario con la clase `miFormulario`, podemos utilizar el selector `querySelectorAll` de la siguiente manera:

```
const elementosSelectorCSS = document.querySelectorAll('.miFormulario input')
```

Manipulación de elementos seleccionados

Una vez que hemos seleccionado los elementos deseados del DOM, podemos manipularlos de diversas maneras utilizando JavaScript. Algunas de las acciones más comunes que podemos realizar incluyen:

- Cambiar el contenido de un elemento utilizando la propiedad `innerHTML`.
- Modificar estilos utilizando la propiedad `style`.
- Agregar o eliminar clases con los métodos `classList.add` y `classList.remove`.
- Añadir, modificar o eliminar atributos utilizando el método `setAttribute` y las propiedades `getAttribute` y `removeAttribute`.

A continuación, se muestra un ejemplo de cómo cambiar el contenido de un elemento seleccionado utilizando JavaScript:

```
 const elemento = document.getElementById('miElemento');
elemento.innerHTML = 'Nuevo contenido';
```

En resumen, JavaScript ofrece una variedad de selectores que nos permiten seleccionar elementos del DOM de manera eficiente. Utilizando estos selectores, podemos manipular y modificar los elementos seleccionados de acuerdo a nuestras necesidades.

Actividad

1. Utiliza el selector adecuado para seleccionar todos los elementos `div` presentes en el DOM y almacénalos en una variable llamada `elementosDiv`.
2. Modifica el contenido de todos los elementos seleccionados en el paso anterior y cámbialo por "Nuevo contenido".
3. Agrega la clase `destacado` a todos los elementos `h1` presentes en el DOM.
4. Elimina el atributo `disabled` de todos los elementos `button` que lo tengan.
5. Crea un selector CSS que seleccione el elemento `span` con el ID `miSpan` y almacénalo en una variable llamada `elementoSpan`.
6. Cambia el estilo del elemento seleccionado en el paso anterior y modifica el color de fondo a rojo.

¡Ponte a prueba y comprueba tus conocimientos seleccionando y manipulando elementos del DOM utilizando selectores en JavaScript!

Conclusión - Selección de elementos del DOM utilizando selectores en JavaScript

En este tema, hemos explorado en profundidad la selección de elementos del DOM utilizando selectores en JavaScript. Hemos aprendido diferentes métodos de selección, como getElementById, getElementsByClassName, getElementsByTagName y querySelectorAll. Además, hemos visto cómo utilizar selectores CSS para seleccionar elementos de forma más precisa. Con estas herramientas, podremos encontrar y manipular cualquier elemento en el DOM de nuestras páginas web.

Manipulación del contenido y atributos del DOM con JavaScript

03 | Manipulación del contenido y atributos del DOM con JavaScript

Introducción

En el desarrollo web, la manipulación del Document Object Model (DOM) es esencial para crear páginas dinámicas y interactivas. El DOM es una representación estructurada de la página web que permite acceder y manipular cada elemento HTML mediante JavaScript. En este tema, aprenderemos cómo manipular el contenido y los atributos del DOM utilizando JavaScript, lo que nos permitirá cambiar y actualizar el contenido de nuestra página.

Manipulando el contenido del DOM

Selección de elementos

El primero paso para manipular el contenido del DOM es seleccionar los elementos que deseamos modificar. Para ello, utilizamos distintos métodos y propiedades de JavaScript:

- `getElementById`: nos permite seleccionar un elemento a través de su identificador único.
- `getElementsByClassName`: permite seleccionar uno o varios elementos mediante su clase.
- `getElementsByTagName`: permite seleccionar uno o varios elementos mediante su etiqueta.

- `querySelector`: nos permite seleccionar un elemento utilizando un selector CSS.
- `querySelectorAll`: nos permite seleccionar varios elementos utilizando un selector CSS.

Modificación del contenido

Una vez que hemos seleccionado los elementos, podemos utilizar diferentes métodos y propiedades para modificar su contenido:

- `innerText`: nos permite cambiar el texto contenido dentro de un elemento, incluyendo cualquier etiqueta HTML.
- `innerHTML`: nos permite cambiar el contenido HTML dentro de un elemento, incluyendo etiquetas y atributos.
- `textContent`: nos permite cambiar el texto contenido dentro de un elemento, pero no interpreta las etiquetas HTML.
- `setAttribute`: nos permite añadir o modificar un atributo de un elemento.
- `removeAttribute`: nos permite eliminar un atributo de un elemento.

Manipulando los atributos del DOM

Accediendo a los atributos

Podemos acceder a los atributos de un elemento utilizando la propiedad `attributes`, que devuelve una lista de todos los atributos del elemento. También podemos utilizar métodos específicos para acceder a atributos específicos:

- `getAttribute`: nos permite obtener el valor de un atributo específico.
- `hasAttribute`: nos permite saber si un elemento tiene un atributo específico.

- `classList`: nos permite acceder a la lista de clases de un elemento.

Modificación de los atributos

Además de acceder a los atributos, también podemos modificarlos utilizando diferentes métodos:

- `setAttribute`: nos permite añadir o modificar un atributo de un elemento.
- `removeAttribute`: nos permite eliminar un atributo de un elemento.
- `classList.add`: nos permite añadir una clase a un elemento.
- `classList.remove`: nos permite eliminar una clase de un elemento.
- `classList.toggle`: nos permite alternar la presencia de una clase en un elemento.

Conclusiones

La manipulación del contenido y los atributos del DOM con JavaScript nos permite crear páginas web más dinámicas e interactivas. Mediante la selección y modificación de elementos del DOM, podemos actualizar el contenido de nuestra página de forma dinámica, agregar o eliminar elementos, y realizar todo tipo de interacciones con el usuario. En resumen, la manipulación del DOM con JavaScript es una habilidad fundamental para todo desarrollador web.

En esta parte del curso, hemos visto cómo manipular el contenido y los atributos del DOM utilizando JavaScript. Hemos aprendido a modificar el texto y el HTML de los elementos, así como los valores de sus atributos. Además, hemos explorado diferentes métodos para agregar, eliminar y reemplazar elementos en el DOM. Con esta capacidad de manipulación, podemos crear interfaces dinámicas y personalizadas en nuestras páginas web.

Ejercicios Practicos

Pongamos en práctica tus conocimientos

En esta lección, pondremos la teoría en práctica a través de actividades prácticas. Haga clic en los elementos a continuación para verificar cada ejercicio y desarrollar habilidades prácticas que lo ayudarán a tener éxito en el tema.

Crear un elemento HTML

Escriba un código JavaScript que cree un elemento HTML con el texto 'Hola mundo' y lo agregue al final del elemento con el id 'container'.

Seleccionar elementos por clase

Escriba un código JavaScript que seleccione todos los elementos con la clase 'item' y agregue la clase 'selected' a cada uno de ellos.

Cambiar el contenido de un elemento

Escriba un código JavaScript que cambie el texto del elemento con el id 'title' por 'Nuevo título'.

Resumen

Repasemos lo que acabamos de ver hasta ahora

05 | Resumen

- ✓ En esta introducción a la manipulación del DOM con JavaScript, hemos aprendido los conceptos básicos necesarios para acceder y manipular elementos en el DOM. Hemos visto cómo utilizar la sintaxis de JavaScript para seleccionar elementos del DOM utilizando selectores y cómo modificar el contenido y los atributos de estos elementos. Con estas técnicas, podemos crear interacciones dinámicas en nuestras páginas web y mejorar la experiencia del usuario.
- ✓ En este tema, hemos explorado en profundidad la selección de elementos del DOM utilizando selectores en JavaScript. Hemos aprendido diferentes métodos de selección, como getElementById, getElementsByClassName, getElementsByTagName y querySelectorAll. Además, hemos visto cómo utilizar selectores CSS para seleccionar elementos de forma más precisa. Con estas

herramientas, podremos encontrar y manipular cualquier elemento en el DOM de nuestras páginas web.

- ✓ En esta parte del curso, hemos visto cómo manipular el contenido y los atributos del DOM utilizando JavaScript. Hemos aprendido a modificar el texto y el HTML de los elementos, así como los valores de sus atributos. Además, hemos explorado diferentes métodos para agregar, eliminar y reemplazar elementos en el DOM. Con esta capacidad de manipulación, podemos crear interfaces dinámicas y personalizadas en nuestras páginas web.

Prueba

Comprueba tus conocimientos respondiendo unas preguntas

06 | Prueba

Pregunta 1/6

¿Cuál es la forma correcta de seleccionar un elemento del DOM utilizando un selector de clase en JavaScript?

- ◯ document.querySelector('#mi-clase')
- ◯ document.querySelector('.mi-clase')
- ◯ document.querySelector('.mi-clase')

Pregunta 2/6

¿Qué método se puede utilizar para seleccionar múltiples elementos del DOM utilizando un selector de clase en JavaScript?

- ◯ document.getElementsByName('.mi-clase')
- ◯ document.querySelector('.mi-clase')
- ◯ document.querySelectorAll('.mi-clase')

Pregunta 3/6

¿Cuál es la forma correcta de cambiar el contenido de un elemento del DOM utilizando JavaScript?

- ◯ document.getElementById('mi-elemento').textContent = 'Nuevo contenido';
- ◯ document.getElementById('mi-elemento').innerHTML = 'Nuevo contenido';
- ◯ document.getElementById('mi-elemento').innerText = 'Nuevo contenido';

Pregunta 4/6

¿Cómo se puede obtener el valor de un atributo de un elemento del DOM utilizando JavaScript?

- document.querySelector('.mi-elemento').value
- document.querySelector('.mi-elemento').getAttribute('mi-atributo');
- document.querySelector('.mi-elemento').setAttribute('mi-atributo', 'valor');

Pregunta 5/6

¿Cuál es la forma correcta de agregar una clase a un elemento del DOM utilizando JavaScript?

- document.querySelector('.mi-elemento').classList.add('mi-clase');
- document.querySelector('.mi-elemento').classList.remove('mi-clase');
- document.querySelector('.mi-elemento').classList.toggle('mi-clase');

Pregunta 6/6

¿Cómo se puede comprobar si un elemento del DOM tiene una determinada clase utilizando JavaScript?

- document.querySelector('.mi-elemento').classList.contains('mi-clase');
- document.querySelector('.mi-elemento').classList.add('mi-clase');
- document.querySelector('.mi-elemento').classList.remove('mi-clase');

Entregar

Conclusión

Felicidades!

¡Felicitaciones por completar este curso! Has dado un paso importante para desbloquear todo tu potencial. Completar este curso no se trata solo de adquirir conocimientos; se trata de poner ese conocimiento en práctica y tener un impacto positivo en el mundo que te rodea.

Comparte este curso