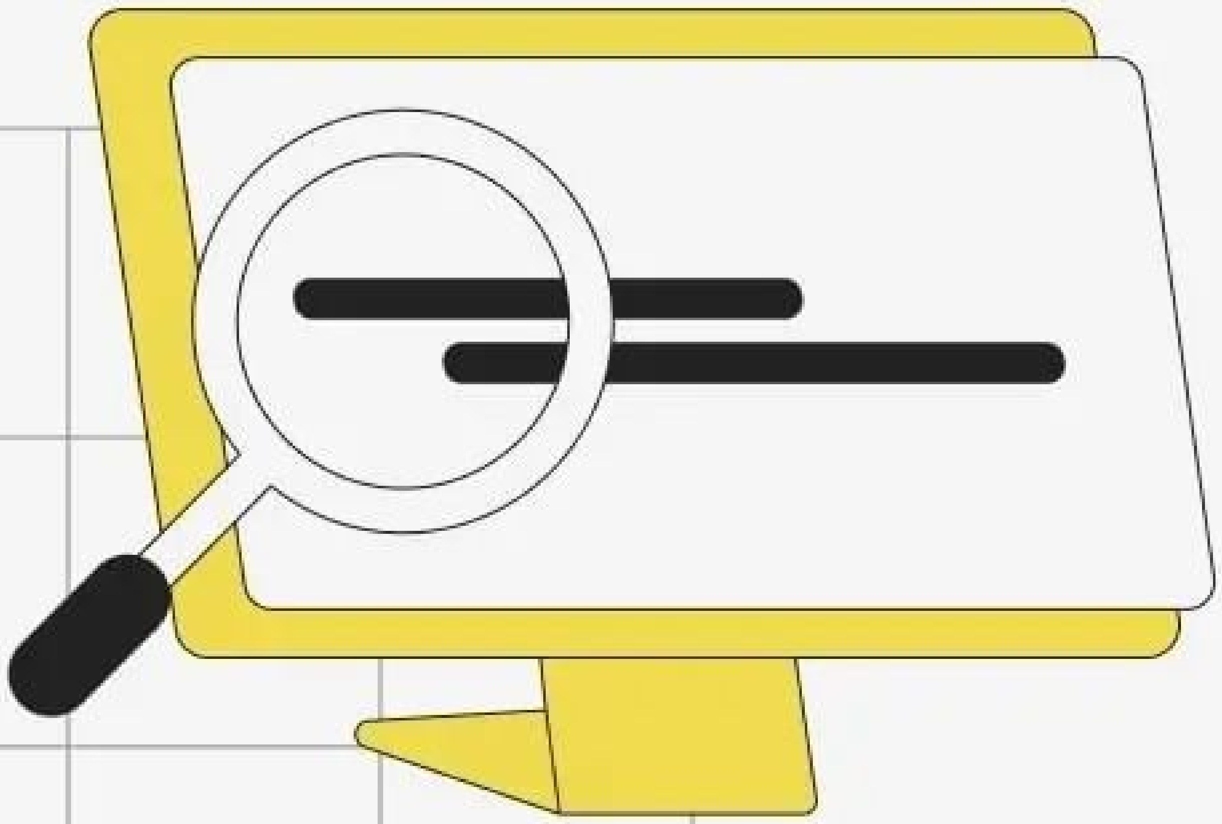
@david.webdev

JS The Ultimate DOM Guide



Follow For More

Web Development | Design | Coding

Swipe

Intro

What Is The DOM?

Simply put, the **Document Object Model (DOM)** is a method of **translating HTML** elements into a form that **JavaScript** can understand and **interact** with using different **methods**.

Why is the DOM important? Because **JavaScript** needs to understand the web page in order to **work** with it and **manipulate it** to turn it into a **dynamic web page**.

JavaScript can interact with the page through the **DOM API** (Application Programming Interface), which is a **set** of **properties** and **methods** used to **access** and **modify** information about our document and specific elements inside it.

Let's take a closer look and examine the DOM!

Swipe

🔍 Examining The DOM

These are just a few of the **operations** you can run in your **JavaScript** file to **examine** the **HTML document:**

`console.dir(document)`	Lets you view the entire DOM Tree Structure!
`console.log(document.title)`	Outputs the title of the current document.
`console.log(document.URL)`	Outputs the full URL of the current web page.
`console.log(document.lastModified)`	Show the date and time of the last modification.
`console.log(document.head)`	Outputs the whole head tag and all its children.
`console.log(document.all)`	Outputs all of the HTML tags on the web page.

Next let's take a look at accessing DOM elements!

Next

Selecting Elements

When we say “**Element**” we refer to a **HTML tag** that is present in our **markup**. Using **JavaScript**, we can access these elements in our code, **store** them in **variables** and **conduct operations** on them:

```
// stores the element with the specified ID
const el1 = document.getElementById('myId')

// stores all elements with the specified class name
const el2 = document.getElementsByClassName('myClass')

// stores all elements that have the specified tag name
const el3 = document.getElementsByTagName('div')

// stores the first element with the specified id / class / tag
const el4 = document.querySelector('#myId')

// stores all the elements with the specified id / class / tag
const el5 = document.querySelectorAll('.myClass')
```

The **querySelector method** is a more **flexible** and universal way of **selecting elements**, because you can **pass in anything** you wish to **select**, be it an **id, class or tag name.**

Next

Manipulating Text

When working with text, there are **three main methods** that you will be using **frequently** in order to make your changes on the web page, **textContent**, **innerText** and **innerHtml**:

```
<p class="my-description">I am a DOM pro!</p>
```

Let's say we have the above HTML paragraph on our page, we can use the following methods to work with it's content:

1. The textContent Method

```
// selecting the element and seeing it's text content
const myDescription = document.querySelector('.my-description');
console.log(myDescription.textContent) // 'I am a DOM pro!'

// updating the text content
myDescription.textContent = 'I am a DOM beginner :(';
console.log(myDescription.textContent) // 'I am a DOM beginner :('
```

2. The innerHtml Method

```
// using innerHtml to add a new tag inside our paragraph!
myDescription.innerHtml += '<span>Woah</span>';
console.log(myDescription) //'I am a DOM pro!<span>Woah</span>'
```

3. The innerText Method

This one is very **similar** to the **textContent** method, the difference being that **innerText** only works on **human-readable elements**!

Next

Changing Styles

There are a number of things we can do to **manipulate** the **styles** of existing **HTML Elements**, here's how:

1. Changing Styles Directly

```
const myCard = document.querySelector('.card');
myCard.style.backgroundColor = 'orange';
```

By accessing the **.style** property you can **modify any CSS** of the selected **HTML Element** and give it any **value** you wish.

2. Updating The Class List

A very useful method to **edit** styles is to update the element's **class list**, you can **add**, **remove** or **toggle** classes that you have **defined** in your **CSS file** on the selected **HTML Element** like so:

```
const myCard = document.querySelector('.card');

// add a class to an element
myCard.classList.add('small-card');

console.log(Array.from(myCard.classList))
// ['card', 'small-card']

// remove a class from an element
myCard.classList.remove('card');
console.log(Array.from(myCard.classList))
// ['small-card']

// toggle a class on an element
myCard.classList.toggle('small-card');
```

Next

Traversing The DOM

Traversing the DOM simply means to **move** up and down in the **DOM** structure, navigating through **children** and **parent elements**:

```javascript
// select the HTML element
const myCard = document.querySelector('.card');

// access the children of an element
console.log(myCard.children);

// access the parent element of an element
console.log(myCard.parentElement);

// access the first or last child of an element
console.log(myCard.firstElementChild);
console.log(myCard.lastElementChild);

// access the siblings of an element
console.log(myCard.nextElementSibling);
console.log(myCard.previousElementSibling);
```

Next let's see how to create new elements in the DOM!

Next

🆕 Creating New Elements

When creating **dynamic web pages**, you will often need to **generate** new elements based on the user's **actions**, this is **how you can do it:**

```javascript
const cardsContainer = document.querySelector('.cards');
const newCard = document.createElement('div');
const cardText = document.createTextNode('My New Card!');

// set a class for the new element
newCard.className = 'card';

// set an attribute for the new element
newCard.setAttribute('title', 'My card is cool');

// add the text node we created to the tag
newCard.appendChild(cardText);
console.log(newCard);
// <div class="card" title="My card is cool">My New Card!</div>

// add the new card to the parent container element
cardsContainer.appendChild(newCard);
```

That's all! You've just created a new **HTML Element** using the **JavaScript DOM API** and have successfully **added** it to the page! 🎉

Next

david.webdev

Was this useful?

Follow me for more posts like this

Let me know with a comment

Share it with others

Let me know you liked it!

Save it for later