**DOM**

# HTML DOM Methods

HTML DOM methods are **actions** you can perform (on HTML Elements).

HTML DOM properties are **values** (of HTML Elements) that you can set or change.

The DOM Programming Interface

The HTML DOM can be accessed with JavaScript (and with other programming languages).

In the DOM, all HTML elements are defined as **objects**.

The programming interface is the properties and methods of each object.

A **property** is a value that you can get or set (like changing the content of an HTML element).

A **method** is an action you can do (like add or deleting an HTML element).

Example

The following example changes the content (the innerHTML) of the <p> element with id="demo":

Example

<html>  
<body>  
  
<p id="demo"></p>  
  
<script>  
document.getElementById("demo").innerHTML = "Hello World!";  
</script>  
  
</body>  
</html>

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_dom_method)

In the example above, getElementById is a **method**, while innerHTML is a **property**.

The getElementById Method

The most common way to access an HTML element is to use the id of the element.

In the example above the getElementById method used id="demo" to find the element.

The innerHTML Property

The easiest way to get the content of an element is by using the innerHTML property.

The innerHTML property is useful for getting or replacing the content of HTML elements.

The innerHTML property can be used to get or change any HTML element, including <html> and <body>.

DOM Document

The HTML DOM document object is the owner of all other objects in your web page.

The HTML DOM Document Object

The document object represents your web page.

If you want to access any element in an HTML page, you always start with accessing the document object.

Below are some examples of how you can use the document object to access and manipulate HTML.

Finding HTML Elements

|  |  |
| --- | --- |
| **Method** | **Description** |
| document.getElementById(*id*) | Find an element by element id |
| document.getElementsByTagName(*name*) | Find elements by tag name |
| document.getElementsByClassName(*name*) | Find elements by class name |

Changing HTML Elements

|  |  |
| --- | --- |
| **Property** | **Description** |
| *element*.innerHTML =  *new html content* | Change the inner HTML of an element |
| *element*.*attribute = new value* | Change the attribute value of an HTML element |
| *element*.style.*property = new style* | Change the style of an HTML element |
| **Method** | **Description** |
| *element*.setAttribute*(attribute, value)* | Change the attribute value of an HTML element |

Adding and Deleting Elements

|  |  |
| --- | --- |
| **Method** | **Description** |
| document.createElement(*element*) | Create an HTML element |
| document.removeChild(*element*) | Remove an HTML element |
| document.appendChild(*element*) | Add an HTML element |
| document.replaceChild(*new, old*) | Replace an HTML element |
| document.write(*text*) | Write into the HTML output stream |

ADVERTISEMENT

Adding Events Handlers

|  |  |
| --- | --- |
| **Method** | **Description** |
| document.getElementById(*id*).onclick = function(){*code*} | Adding event handler code to an onclick event |

# JavaScript HTML DOM Elements

[❮ Previous](https://www.w3schools.com/js/js_htmldom_document.asp)[Next ❯](https://www.w3schools.com/js/js_htmldom_html.asp)

This page teaches you how to find and access HTML elements in an HTML page.

## Finding HTML Elements

Often, with JavaScript, you want to manipulate HTML elements.

To do so, you have to find the elements first. There are several ways to do this:

* Finding HTML elements by id
* Finding HTML elements by tag name
* Finding HTML elements by class name
* Finding HTML elements by CSS selectors
* Finding HTML elements by HTML object collections

## Finding HTML Element by Id

The easiest way to find an HTML element in the DOM, is by using the element id.

This example finds the element with id="intro":

### Example

const element = document.getElementById("intro");

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_dom_getelementbyid)

If the element is found, the method will return the element as an object (in element).

If the element is not found, element will contain null.

## Finding HTML Elements by Tag Name

This example finds all <p> elements:

### Example

const element = document.getElementsByTagName("p");

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_dom_getelementsbytagname2)

This example finds the element with id="main", and then finds all <p> elements inside "main":

### Example

const x = document.getElementById("main");  
const y = x.getElementsByTagName("p");

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_dom_getelementsbytagname)

ADVERTISEMENT

## Finding HTML Elements by Class Name

If you want to find all HTML elements with the same class name, use getElementsByClassName().

This example returns a list of all elements with class="intro".

### Example

const x = document.getElementsByClassName("intro");

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_dom_getelementsbyclassname)

## Finding HTML Elements by CSS Selectors

If you want to find all HTML elements that match a specified CSS selector (id, class names, types, attributes, values of attributes, etc), use the querySelectorAll() method.

This example returns a list of all <p> elements with class="intro".

### Example

const x = document.querySelectorAll("p.intro");

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_dom_queryselectorall)

## Finding HTML Elements by HTML Object Collections

This example finds the form element with id="frm1", in the forms collection, and displays all element values:

### Example

const x = document.forms["frm1"];  
let text = "";  
for (let i = 0; i < x.length; i++) {  
  text += x.elements[i].value + "<br>";  
}  
document.getElementById("demo").innerHTML = text;