**JavaScript Tutorial: Navigation The DOM | Web Development Tutorials #56**

In this tutorial, we are going to learn how we can do DOM manipulation in JavaScript. Let us start by making a new file as *tut56.html* and then add the boilerplate to get the HTML code. Then give the title as **Manipulating DOM**under the <title> tag.

The HTML DOM (Document Object Model) is a standard **object** model and **programming interface** for HTML. It defines:

* The HTML elements as objects
* The properties of all HTML elements
* The methods to access all HTML elements
* The events for all HTML elements

In other words: The HTML DOM is a standard for how to **get, change, add**, or **delete** HTML elements.

Let us begin with our HTML code as follows-

<div id="main" class="container">

<ul id="nav">

<li>Home</li>

<li>About</li>

<li>Services</li>

<li>More About Us</li>

<li>Contact Us</li>

</ul>

</div>

Copy

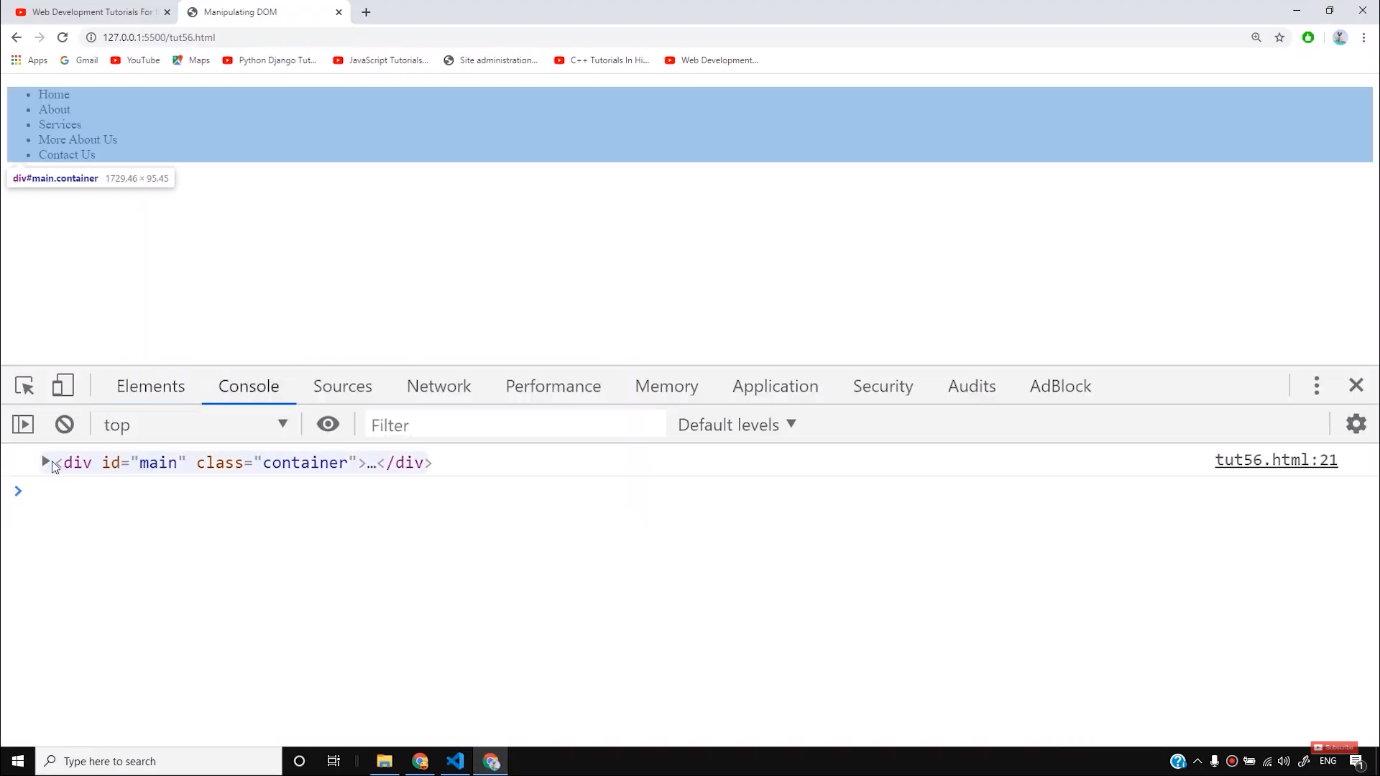
* The most common way to access an HTML element is to use the *id*of the element. For example, if we write as follows-

let main = document.getElementById('main');

console.log(main);

Copy

By writing **getElementById,**we can target the HTML through its id. Therefore, the result of the above code will look like-



Let us see another example. I want to target the elements under the id *nav,*we can write this as follows-

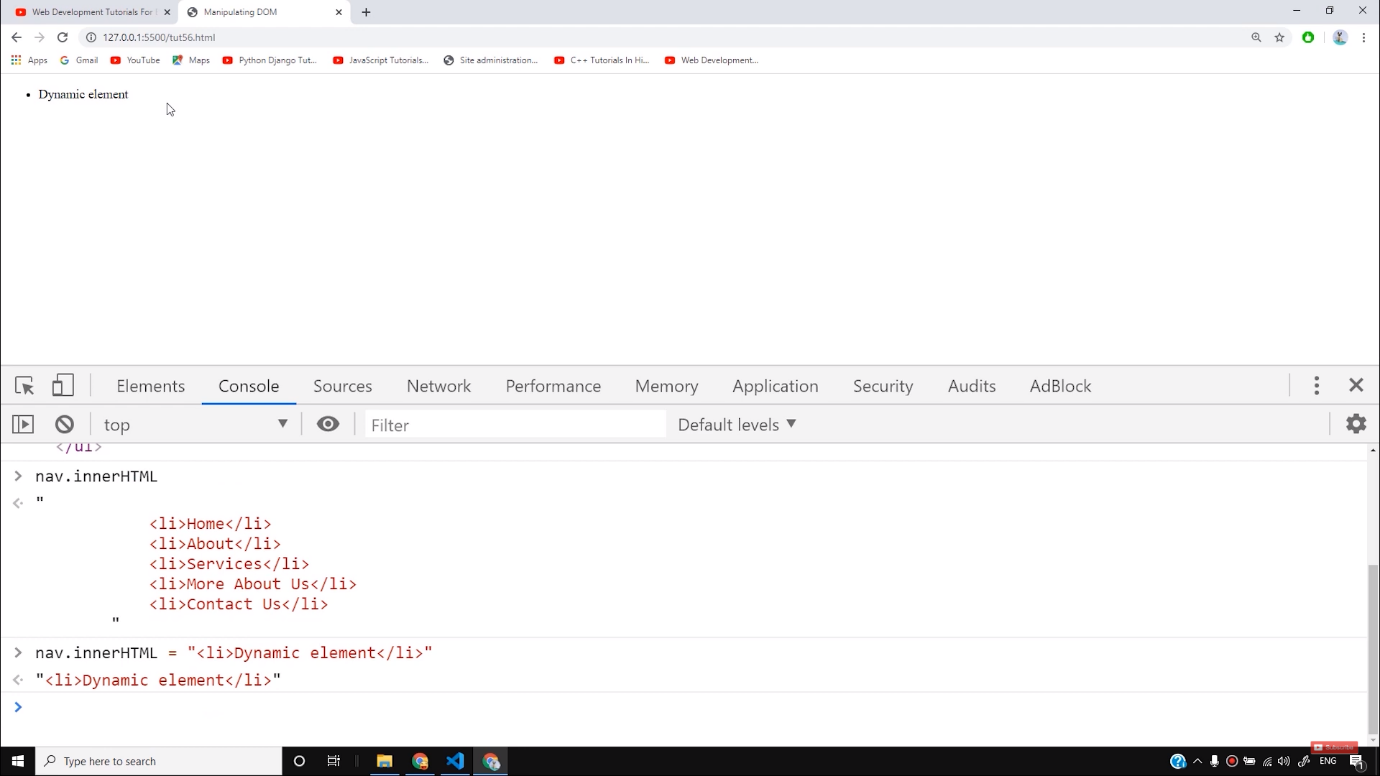
let nav = document.getElementById('nav');

console.log(nav);

Copy

For DOM to work, keep in mind that while selecting the elements with its id, there should be a unique id for every element in the HTML code. You cannot work with two same ids for different elements otherwise it will create a problem.

The easiest way to get the content of an element is by using the **innerHTML property.**It is useful for getting or replacing the content of HTML elements. The example is shown below-



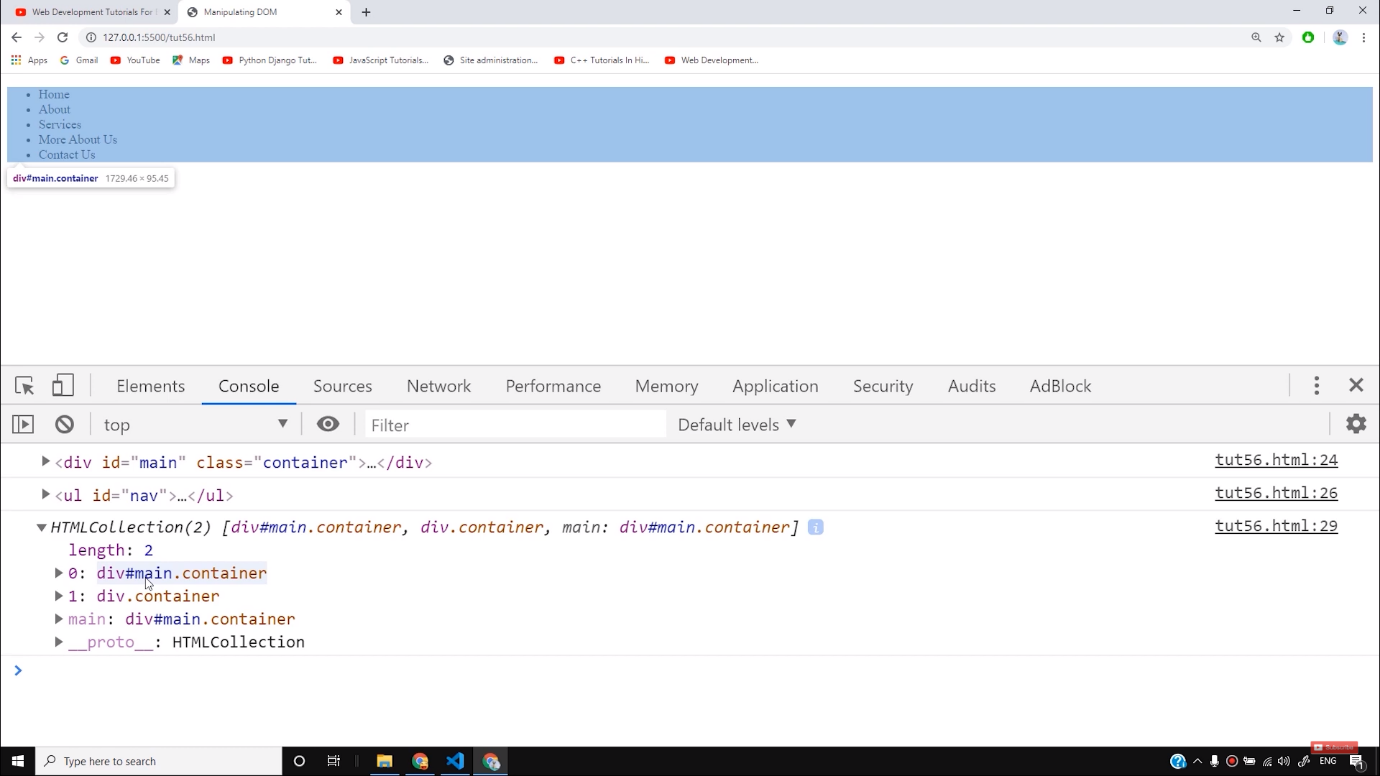
* However, we can also extract the elements from HTML with the help of **class**name. If we want to find all the elements with same class name, use **getElementsByClassName().**Let us understand this with an example. If we write as follows-

let containers = document.getElementsByClassName('container');

console.log(containers);

Copy

The output of the above code will be as follows-



Here, we can see two container class as have made two container classes. But if we write as **container(0)** or **container(1)**then we will get the first and second containers respectively.

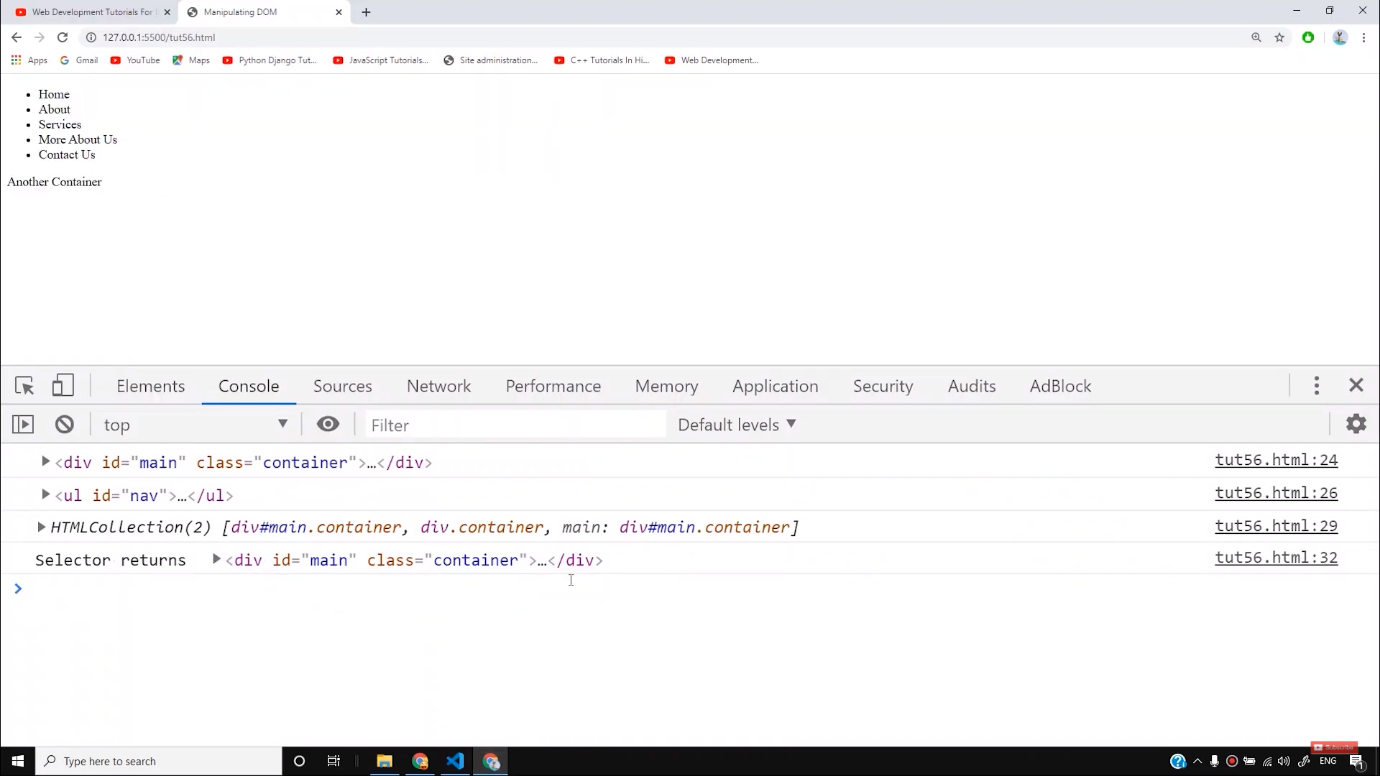
* If you want to find all HTML elements that match a specified CSS selector (id, class names, types, attributes, values of attributes, etc), use the **querySelector()**  method. For example, if we write-

let sel = document.querySelector('container’);

console.log("Selector returns ", sel)

Copy

In this example, the first element which matches with the selector will be returned as follows-



* We can use all the selectors that we have learned till now. We can also use **querySelectorAll()**to target the elements. For example, if we want to select all the ids, we can write as follows-

let sel = document.querySelectorAll('#nav');

console.log("Selector returns ", sel)

Copy

In the upcoming tutorials, we will see some more selectors. Till then practice these because they are the most important selectors that are mostly used.

**Code as described/written in the video**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Manipulating DOM</title>

</head>

<body>

<div id="main" class="container">

<ul id="nav">

<li>Home</li>

<li>About</li>

<li>Services</li>

<li>More About Us</li>

<li>Contact Us</li>

</ul>

</div>

<div class="container">

Another Container

</div>

<script>

let main = document.getElementById('main');

console.log(main);

let nav = document.getElementById('nav');

console.log(nav);

let containers = document.getElementsByClassName('container');

console.log(containers);

// let sel = document.querySelector('#nav>li');

// console.log("Selector returns ", sel)

let sel = document.querySelectorAll('#nav>li');

console.log("Selector returns ", sel)

</script>

</body>

</html>

# JavaScript HTML DOM Navigation

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

With the HTML DOM, you can navigate the node tree using node relationships.

## **DOM Nodes**

According to the W3C HTML DOM standard, everything in an HTML document is a node:

* The entire document is a document node
* Every HTML element is an element node
* The text inside HTML elements are text nodes
* Every HTML attribute is an attribute node (deprecated)
* All comments are comment nodes



With the HTML DOM, all nodes in the node tree can be accessed by JavaScript.

New nodes can be created, and all nodes can be modified or deleted.

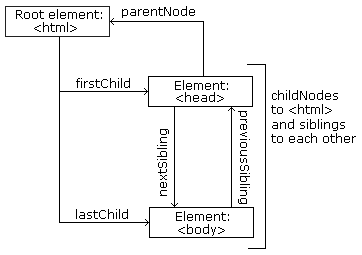
## **Node Relationships**

The nodes in the node tree have a hierarchical relationship to each other.

The terms parent, child, and sibling are used to describe the relationships.

* In a node tree, the top node is called the root (or root node)
* Every node has exactly one parent, except the root (which has no parent)
* A node can have a number of children
* Siblings (brothers or sisters) are nodes with the same parent

<html>  
  
  <head>  
    <title>DOM Tutorial</title>  
  </head>  
  
  <body>  
    <h1>DOM Lesson one</h1>  
    <p>Hello world!</p>  
  </body>  
  
</html>



From the HTML above you can read:

* <html> is the root node
* <html> has no parents
* <html> is the parent of <head> and <body>
* <head> is the first child of <html>
* <body> is the last child of <html>

and:

* <head> has one child: <title>
* <title> has one child (a text node): "DOM Tutorial"
* <body> has two children: <h1> and <p>
* <h1> has one child: "DOM Lesson one"
* <p> has one child: "Hello world!"
* <h1> and <p> are siblings

ADVERTISEMENT

## **Navigating Between Nodes**

You can use the following node properties to navigate between nodes with JavaScript:

* parentNode
* childNodes[nodenumber]
* firstChild
* lastChild
* nextSibling
* previousSibling

## **Child Nodes and Node Values**

A common error in DOM processing is to expect an element node to contain text.

### **Example:**

<title id="demo">DOM Tutorial</title>

The element node <title> (in the example above) does **not** contain text.

It contains a **text node** with the value "DOM Tutorial".

The value of the text node can be accessed by the node's innerHTML property:

myTitle = document.getElementById("demo").innerHTML;

Accessing the innerHTML property is the same as accessing the nodeValue of the first child:

myTitle = document.getElementById("demo").firstChild.nodeValue;

Accessing the first child can also be done like this:

myTitle = document.getElementById("demo").childNodes[0].nodeValue;

All the (3) following examples retrieves the text of an <h1> element and copies it into a <p> element:

### **Example**

<html>  
<body>  
  
<h1 id="id01">My First Page</h1>  
<p id="id02"></p>  
  
<script>  
document.getElementById("id02").innerHTML = document.getElementById("id01").innerHTML;  
</script>  
  
</body>  
</html>

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_dom_nav_innerhtml1)

### **Example**

<html>  
<body>  
  
<h1 id="id01">My First Page</h1>  
<p id="id02"></p>  
  
<script>  
document.getElementById("id02").innerHTML = document.getElementById("id01").firstChild.nodeValue;  
</script>  
  
</body>  
</html>

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_dom_nav_innerhtml2)

### **Example**

<html>  
<body>  
  
<h1 id="id01">My First Page</h1>  
<p id="id02">Hello!</p>  
  
<script>  
document.getElementById("id02").innerHTML = document.getElementById("id01").childNodes[0].nodeValue;  
</script>  
  
</body>  
</html>

[Try it Yourself »](https://www.w3schools.com/js/tryit.asp?filename=tryjs_dom_nav_innerhtml3)

## **InnerHTML**

In this tutorial we use the innerHTML property to retrieve the content of an HTML element.

However, learning the other methods above is useful for understanding the tree structure and the navigation of the DOM.

## **DOM Root Nodes**

There are two special properties that allow access to the full document:

* document.body - The body of the document
* document.documentElement - The full document

### **Example**

<html>  
<body>  
  
<h2>JavaScript HTMLDOM</h2>  
<p>Displaying document.body</p>  
  
<p id="demo"></p>  
  
<script>  
document.getElementById("demo").innerHTML = document.body.innerHTML;  
</script>  
  
</body>  
</html>

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

### **Example**

<html>  
<body>  
  
<h2>JavaScript HTMLDOM</h2>  
<p>Displaying document.documentElement</p>  
  
<p id="demo"></p>  
  
<script>  
document.getElementById("demo").innerHTML = document.documentElement.innerHTML;  
</script>  
  
</body>  
</html>

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

## **The nodeName Property**

The nodeName property specifies the name of a node.

* nodeName is read-only
* nodeName of an element node is the same as the tag name
* nodeName of an attribute node is the attribute name
* nodeName of a text node is always #text
* nodeName of the document node is always #document

### **Example**

<h1 id="id01">My First Page</h1>  
<p id="id02"></p>  
  
<script>  
document.getElementById("id02").innerHTML = document.getElementById("id01").nodeName;  
</script>

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

**Note:** nodeName always contains the uppercase tag name of an HTML element.

## **The nodeValue Property**

The nodeValue property specifies the value of a node.

* nodeValue for element nodes is null
* nodeValue for text nodes is the text itself
* nodeValue for attribute nodes is the attribute value

## **The nodeType Property**

The nodeType property is read only. It returns the type of a node.

### **Example**

<h1 id="id01">My First Page</h1>  
<p id="id02"></p>  
  
<script>  
document.getElementById("id02").innerHTML = document.getElementById("id01").nodeType;  
</script>

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

The most important nodeType properties are:

|  |  |  |
| --- | --- | --- |
| **Node** | **Type** | **Example** |
| ELEMENT\_NODE | 1 | <h1 class="heading">W3Schools</h1> |
| ATTRIBUTE\_NODE | 2 | class = "heading" (deprecated) |
| TEXT\_NODE | 3 | W3Schools |
| COMMENT\_NODE | 8 | <!-- This is a comment --> |
| DOCUMENT\_NODE | 9 | The HTML document itself (the parent of <html>) |
| DOCUMENT\_TYPE\_NODE | 10 | <!Doctype html> |