# DOM Manipulation

Using JavaScript you can create dynamic HTML (A **dynamic** programming language is one in which operations are otherwise done at compile-time can be also done at run-time.)

**D**ocument **O**bject **M**odel of HTML page (constructs a tree of objects…[see w3schools example](https://www.w3schools.com/js/js_htmldom.asp)).

Think about every piece on the tree as an *object with properties and methods*. Using dot notation, you can access properties of or apply methods to objects.

The object is **document**, so:

* document**.getElementById** is the object.**METHOD**
* When you add **.innerHTML** (document.getElementById(“”).innerHTML = ….) you are accessing an element and changing a **PROPERTY** of that element.

With JavaScript you can make these adjustments to the DOM:

**1.Change HTML elements (<body>, <title>, <a>, <p>, <h1>)**

First find the element(s), then change it(them).

Example 1

var parent1 = document.getElementById("div1");

var child1 = document.getElementById("p1");

parent1.removeChild(child1);

(used dot notation to access an object(parent1), then apply a method to it(removeChild). We removed the child1 object from the parent1 object.)

Methods to FIND elements: usually these elements have been identified as an object(parent1) with a unique id(“div1”) so *var parent1 = document.getElementById("div1");*

* document.getElementByID(“id”)-- document.getElementById(“div1”) would find <div id=“div1”>
* document.getElementByTagName(“name”)-- document.getElementByTagName(“p”)[0] would find all <p>called an HTML collection(like an array). Then it selects the first [0] <p>, using a specific index number to get a specific element. Could be “div”, “body”, “img”, “a”, etc…
* document.getElementByClassName(“name”) used mostly in CSS-- document.getElementByClassName(“intro”) would find all tags that have <*element* class=”intro”>

Methods to add or remove elements. You can add/remove elements to the document, or you can be more specific and add/remove elements to *a parent object inside the document*… *parent1.removeChild(child2); see the full example in example 1 above*

* document.createElement(element)--document.createElement(“p”) creates a <p> element and adds it to the HTML document

Example 2

create a new <p> element with appended text:

var newP = document.createElement("p");

var newText = document.createTextNode("This is the text inside.");

newP.appendChild(newText);

//what you get is HTML <p> This is the text inside.</p>

* document.removeChild(element)--document.removeChild(“p”) removes a <p> element, you would need more specific identifiers here unless you only have one <p> element in the doc
* document.appendChild(element)--would add the specified element to the end of the HTML doc or parent element --see example 2 above
* document.replaceChild(new, old)
* document.insertBefore(elementToBeInserted, ElementToBeInsertedBefore)

Example 3

create a new <p> element with appended text:

var newP = document.createElement("p");

var newText = document.createTextNode("This is the text inside.");

newP.appendChild(newText);

Insert new<p> before <button id="myBtn">

var divElement = document.getElementById("div1");

var btnChild = document.getElementById("myBtn2");

divElement.insertBefore(newP, btnChild);

* document.write(text)

**2.Change HTML attributes** by accessing HTML DOM **PROPERTIES** (href, src, width and height). Using JS we can access and change the values of an HTML element. *In every case, first find the element using document.getElementBy\_\_\_then use dot notation to change the attribute.*

* .innerHTML = “*new HTML content*”—document.getElementById(“title”).innerHTML = “This is a newTitle!”; ( accesses text inside the <h1 id=”title”> and changes it to “…newTitle!”)
* .attribute = new value-- document.getElementById(“pic1”).src = “newPicture.jpg”;
* .style.*property* = new style— add a CSS style change with onclick event

<button id="myBtn" onclick="document.getElementById('myBtn').style.color = 'white'">Text Here</button>

* (method) .setAttribute(attribute, value)—similar to element.attribute above. document.getElementById(“pic1”).setAttribute(src, “newPicture.jpg”);

**3.Change CSS styles** (like color, fontFamily, fontSize)

document.getElementById('myBtn').style.color = 'white';

**4.Create or React to HTML Events** (for a review of these see [w3schools](https://www.w3schools.com/tags/ref_eventattributes.asp))

document.getElementById(id).onclick = function() { code}

Example 4

document.getElementById("myBtn2").addEventListener("click", function addIt() {

var newPara = document.createElement("p");

var newText = document.createTextNode("Correct answer: 1, 3, 5, 19, 67, 89.");

newPara.appendChild(newText);

var parentElement = document.getElementById("div1");

parentElement.appendChild(newPara);