**Ch. 11 – DOM**

You can have inline, internal and external JS similar to CSS. You want to apply a CSS style sheet in the beginning of your html document so the CSS styles are always applied. You want to apply your JS sheet later in the document so all the HTML and CSS files are loaded before the JS is applied.

Best practice is to place your JS sheet right before closing body tag of HTML document.

DOM (document object tree) formats your code into a tree. Similar level parts are siblings if they share the same parent. The head and body element are siblings who share the HTML parent.

Objects in DOM can have properties and methods.

Properties: describe the object (color of car, # of seats)

Can have get and set functions to obtain or change different properties

Don’t necessarily need () at end of these functions.

Car.color; (gets the color) Car.door(0); (sets the # of doors to 0

Difference between get and set is if there is a value in the ()

Whereas CSS-properties are usually hyphenated, JS properties are camelCased.

\*\*\***Values in JS must be written as strings (in quotes)**

Methods: things the object can do (drive, turn AC on)

Car.drive(); (method is called which makes the car drive)

Methods usually end with ()

**\*\*\*Method is something an object can do. Function can be used independent of an object**

**How to select different elements:**

querySelector(“tag/.Class/#ID”) – only returns 1 item (first in HTML code) with specified tag/Class/ID

querySelectorAll(“tag/.Class/#ID”) – returns an array of all items with the specified tag/Class/ID

firstElementChild.lastElementChild

getElementsByTagName(“tagName”) – returns array of all elements with tag name tagName

You can specify a certain index with getElementsByTagName(“tagName”)[specificIndex]

getElementsByClassName(“className”) – returns array of all elements with class name className

getElementByID(“iDName”) – returns one item with ID name iDName. Only returns 1 element as ID is unique. No brackets are needed when using this.

**Separation of concerns:**

It is good practice to have HTML used for content, CSS for styles and JS for behavior. In order to keep the separation of concerns true, you can use JS to change the classes of your elements and have different classes styled differently using your CSS. This prevents you from having to style your elements using JS.

.classList – displays the classes of a querySelector

.classList.add(“className”) – adds a class to the querySelector

.classList.remove(“className”) – removes a class from the querySelector

classList.toggle(“className”) – Adds a class if it’s not there/removes a class if it’s there from a querySelector.

**HTML - get and set functions:**

document.querySelector(“elementName”).innerHTML;

Returns the HTML of elementName.

This is more inclusive than textContent as it returns things like <b></b> or <i></i> as well.

You can change the HTML code by adding this to the end = “htmlCode”;

document.querySelector(“elementName”).textContent;

Returns only the text of elementName

document.querySelector(“elementName”).attributes;

Returns all attributes of given element.

document.querySelector(“elementName”).getAttribute(“attributeName”);

returns value of requested attribute of given element

document.querySelector(“elementName”).setAttribute(“attributeName”, “newValue”);

sets value of requested attribute of given element to newValue

**CSS – set functions:**

document.querySelector(“elementName”).style.cssStyleAttribute = “value”;

changes the cssAttribute to “value” of elementName