JavaScript (JS) is commonly used for manipulating the Document Object Model (DOM), which represents the structure of a web page. DOM manipulation involves accessing and modifying HTML elements, attributes, and content dynamically using JavaScript. Here's a basic overview of how DOM manipulation works in JavaScript:

1. **Accessing DOM Elements**: You can access DOM elements using methods like `getElementById`, `getElementsByClassName`, `getElementsByTagName`, `querySelector`, and `querySelectorAll`. For example:

```javascript
// Accessing an element by ID
var element = document.getElementById("myElement");

// Accessing elements by class name
var elements = document.getElementsByClassName("myClass");

// Accessing elements by tag name
var elements = document.getElementsByTagName("div");

// Accessing elements using CSS selectors
var element = document.querySelector("#myElement");
var elements = document.querySelectorAll(".myClass");
```

2. **Modifying DOM Elements**: Once you have accessed an element, you can modify its properties, attributes, and content. Common properties and methods include `innerHTML`, `innerText`, `textContent`, `setAttribute`, `style`, `classList`, etc. For example:

```javascript
// Modifying content
element.innerHTML = "New content";

// Modifying text
element.innerText = "New text";

// Modifying attributes
element.setAttribute("class", "newClass");

// Modifying CSS styles
element.style.color = "red";

// Adding or removing classes
element.classList.add("newClass");
element.classList.remove("oldClass");
```

3. **Creating and Appending Elements**: You can create new DOM elements dynamically using `document.createElement` and then append them to existing elements using methods like `appendChild` or `insertBefore`. For example:

```javascript
// Creating a new element
var newElement = document.createElement("div");

// Appending it to an existing element
parentElement.appendChild(newElement);

// Inserting it before another element
parentElement.insertBefore(newElement, existingElement);
```

4. **Event Handling**: You can attach event listeners to DOM elements to handle user interactions. Common events include `click`, `mouseover`, `submit`, etc. For example:

```javascript
// Adding an event listener
element.addEventListener("click", function() {
   // Your event handling code here
});
```

5. **Manipulating CSS Classes**: You can dynamically add, remove, or toggle CSS classes on elements to change their appearance or behavior. This is typically done using the `classList` property. For example:

```javascript
// Adding a class
element.classList.add("active");

// Removing a class
element.classList.remove("inactive");

// Toggling a class
element.classList.toggle("highlight");
```

These are just the basics of DOM manipulation in JavaScript. As you work with JavaScript more, you'll encounter more advanced techniques and libraries (like jQuery) that can simplify and streamline DOM manipulation tasks.