Chapter 6

DOM Manipulation

**Example: dom.html**

<!DOCTYPE html>

<html>

<head>

<title>DOM Introduction</title>

</head>

<body>

<h1>Hello, DOM!</h1>

<p>This is a sample page.</p>

</body>

</html>

**Syntax**

document.getElementById(id);

**Example**

<p id="greeting">Hello, World!</p>

To select this <p> element and change its text content:

let element = document.getElementById('greeting');

element.textContent = 'Welcome to the DOM!';

**Another** **Example**

<p id="intro">Welcome to the DOM!</p>

<script>

let element = document.getElementById('intro');

console.log(element.textContent); // Output: Welcome to the DOM!

</script>

**querySelector()**

**Syntax**

document.querySelector(selector);

**Example 1: Selecting by ID**

<p id="greeting">Hello, World!</p>

To select this element by ID:

let element = document.querySelector('#greeting');

element.textContent = 'Welcome to the DOM!';

**Example 2: Selecting by Class**

<div class="container">This is a container</div>

<div class="container">This is another container</div>

To select the first element with the class container:

let element = document.querySelector('.container');

element.style.backgroundColor = 'lightblue';

**Example 3: Selecting by Attribute**

<input type="text" name="username" placeholder="Enter username">

<input type="password" name="password" placeholder="Enter password">

To select the input element with the name username:

let inputElement = document.querySelector('input[name="username"]');

inputElement.style.borderColor = 'green';

**Example 4: Selecting by Multiple Criteria**

<button class="btn primary" id="submit">Submit</button>

To select this button using both ID and class:

let button = document.querySelector('#submit.primary');

button.style.color = 'white';

button.style.backgroundColor = 'blue';

**a. innerHTML**

**Syntax:**

element.innerHTML = "New HTML content";

**Example 1: Modifying HTML content using innerHTML**

<div id="container">

<h2>Old Heading</h2>

<p>This is an old paragraph.</p>

</div>

<script>

let container = document.getElementById("container");

container.innerHTML = "<h2>New Heading</h2><p>This is a new paragraph with <strong>HTML</strong> content.</p>";

</script>

**b. textContent**

**Syntax**

element.textContent = "New text content";

**Example 2: Modifying text content using textContent**

<p id="message">Old message</p>

<script>

let message = document.getElementById("message");

message.textContent = "This is the new message text!";

</script>

**c. innerText**

**Syntax**

element.innerText = "New visible text content";

**Example 3: Modifying visible text using innerText**

<p id="info" style="display:none;">This text is hidden</p>

<script>

let info = document.getElementById("info");

info.innerText = "This text is now visible!";

</script>

* Here, the content of the <p> element is modified, but because the display: none style is applied initially, innerText would not retrieve the text if the element is hidden.

**a. setAttribute()**

**Syntax**

element.setAttribute(attributeName, attributeValue);

**Example 4: Modifying an attribute using setAttribute()**

<a id="link" href="https://example.com">Visit Example</a>

<script>

let link = document.getElementById("link");

link.setAttribute("href", "https://new-url.com");

link.setAttribute("target", "\_blank");

</script>

**b. getAttribute()**

**Syntax:**

let value = element.getAttribute(attributeName);

**Example 5: Retrieving an attribute using getAttribute()**

<a id="link" href="https://example.com">Visit Example</a>

<script>

let link = document.getElementById("link");

let hrefValue = link.getAttribute("href");

console.log(hrefValue); // Outputs: "https://example.com"

</script>

**c. removeAttribute()**

**Syntax**

element.removeAttribute(attributeName);

**Example 6: Removing an attribute using removeAttribute()**

<a id="link" href="https://example.com" target="\_blank">Visit Example</a>

<script>

let link = document.getElementById("link");

link.removeAttribute("target");

</script>

**Example 7: Modifying text input values**

<input type="text" id="username" value="Old Username">

<script>

let usernameInput = document.getElementById("username");

usernameInput.value = "New Username";

</script>

**Example 8: Modifying checkbox checked status**

<input type="checkbox" id="subscribe" checked>

<script>

let checkbox = document.getElementById("subscribe");

checkbox.checked = false; // Unchecks the checkbox

</script>

**c. Modifying Select Dropdowns**

**Example 9: Modifying select dropdown value**

<select id="colors">

<option value="red">Red</option>

<option value="blue">Blue</option>

<option value="green">Green</option>

</select>

<script>

let selectElement = document.getElementById("colors");

selectElement.value = "blue"; // Selects the "Blue" option

</script>

**a. document.createElement()**

**Syntax**

let newElement = document.createElement(tagName);

**Example 1: Creating and appending a new element**

<div id="container"></div>

<script>

// Create a new <p> element

let newParagraph = document.createElement("p");

// Set the text content of the new element

newParagraph.textContent = "This is a dynamically created paragraph.";

// Append the new element to the container

let container = document.getElementById("container");

container.appendChild(newParagraph);

</script>

**b. Adding Text Using createTextNode()**

**Syntax:**

let textNode = document.createTextNode("Your text here");

**Example 2: Creating a text node**

<div id="text-container"></div>

<script>

// Create a new <p> element

let paragraph = document.createElement("p");

// Create a text node

let text = document.createTextNode("This is plain text added to the paragraph.");

// Append the text node to the <p> element

paragraph.appendChild(text);

// Add the <p> element to the DOM

let container = document.getElementById("text-container");

container.appendChild(paragraph);

</script>

**Example 3: Using append() and appendChild()**

<ul id="list"></ul>

<script>

// Create a new list item

let listItem = document.createElement("li");

listItem.textContent = "List Item 1";

// Append the list item

let list = document.getElementById("list");

list.appendChild(listItem);

// Using append() to add another item with text

list.append("List Item 2");

</script>

**Syntax:**

parent.insertBefore(newNode, referenceNode);

**Example 4: Using insertBefore()**

<ul id="items">

<li id="item1">Item 1</li>

</ul>

<script>

// Create a new list item

let newItem = document.createElement("li");

newItem.textContent = "Item 0";

// Insert the new item before the first item

let list = document.getElementById("items");

let firstItem = document.getElementById("item1");

list.insertBefore(newItem, firstItem);

</script>

**e. Using insertAdjacentHTML()**

**Syntax**

element.insertAdjacentHTML(position, htmlString);

**Example 5: Using insertAdjacentHTML()**

<div id="content"></div>

<script>

let content = document.getElementById("content");

// Insert HTML at different positions

content.insertAdjacentHTML("beforeend", "<p>Added at the end</p>");

content.insertAdjacentHTML("afterbegin", "<p>Added at the beginning</p>");

</script>

**Example 6: Removing a child element**

<div id="container">

<p id="paragraph">This is a paragraph to remove.</p>

</div>

<script>

let container = document.getElementById("container");

let paragraph = document.getElementById("paragraph");

// Remove the paragraph

container.removeChild(paragraph);

</script>

**b. remove()**

**Syntax:**

element.remove();

**Example 7: Removing an element using remove()**

<div id="box">This box will be removed.</div>

<script>

let box = document.getElementById("box");

// Remove the element directly

box.remove();

</script>

**Example 8: Clearing all children**

<div id="content">

<p>Child 1</p>

<p>Child 2</p>

<p>Child 3</p>

</div>

<script>

let content = document.getElementById("content");

// Clear all child nodes

content.innerHTML = "";

</script>

**Practical Example**

Here’s a real-world example of creating and removing elements dynamically:

<div id="todo-list">

<h3>Todo List</h3>

<ul id="tasks"></ul>

<button id="add-task">Add Task</button>

</div>

<script>

let taskList = document.getElementById("tasks");

let addTaskButton = document.getElementById("add-task");

// Add a new task

addTaskButton.addEventListener("click", function () {

let newTask = document.createElement("li");

newTask.textContent = "New Task";

// Add a remove button for each task

let removeButton = document.createElement("button");

removeButton.textContent = "Remove";

removeButton.addEventListener("click", function () {

newTask.remove();

});

newTask.appendChild(removeButton);

taskList.appendChild(newTask);

});

</script>

**Examples of Inline Styles**

**Example 1: Changing the Background Color of a Button**

<button id="myButton">Click Me</button>

<script>

let button = document.getElementById("myButton");

// Change the background color using inline styles

button.style.backgroundColor = "blue";

button.style.color = "white";

button.style.fontSize = "20px"; // Font size

</script>

**Example 2: Dynamically Adjusting Styles on Event**

<div id="box" style="width: 100px; height: 100px; background-color: red;"></div>

<button id="resizeButton">Resize Box</button>

<script>

let box = document.getElementById("box");

let resizeButton = document.getElementById("resizeButton");

resizeButton.addEventListener("click", function () {

box.style.width = "200px";

box.style.height = "200px";

box.style.backgroundColor = "green";

});

</script>

**Examples of Using classList**

**Example 3: Adding and Removing Classes Dynamically**

<div id="text" class="normal-text">Hello, World!</div>

<button id="boldButton">Make Bold</button>

<button id="normalButton">Make Normal</button>

<style>

.normal-text {

font-weight: normal;

color: black;

}

.bold-text {

font-weight: bold;

color: blue;

}

</style>

<script>

let text = document.getElementById("text");

let boldButton = document.getElementById("boldButton");

let normalButton = document.getElementById("normalButton");

boldButton.addEventListener("click", function () {

text.classList.add("bold-text");

text.classList.remove("normal-text");

});

normalButton.addEventListener("click", function () {

text.classList.add("normal-text");

text.classList.remove("bold-text");

});

</script>

**Example 4: Toggling Classes**

<div id="box" class="box"></div>

<button id="toggleButton">Toggle Highlight</button>

<style>

.box {

width: 100px;

height: 100px;

background-color: gray;

}

.highlight {

background-color: yellow;

border: 2px solid black;

}

</style>

<script>

let box = document.getElementById("box");

let toggleButton = document.getElementById("toggleButton");

toggleButton.addEventListener("click", function () {

box.classList.toggle("highlight");

});

</script>

**Example 5: Checking Class Presence with contains**

<div id="info" class="hidden">This is hidden content.</div>

<button id="showButton">Show Content</button>

<style>

.hidden {

display: none;

}

.visible {

display: block;

color: green;

}

</style>

<script>

let info = document.getElementById("info");

let showButton = document.getElementById("showButton");

showButton.addEventListener("click", function () {

if (info.classList.contains("hidden")) {

info.classList.remove("hidden");

info.classList.add("visible");

}

});

</script>

**Practical Example: Interactive Style Toggle**

<div id="circle" class="circle"></div>

<button id="changeStyle">Change Style</button>

<style>

.circle {

width: 100px;

height: 100px;

border-radius: 50%;

background-color: red;

}

.square {

border-radius: 0;

background-color: green;

}

</style>

<script>

let circle = document.getElementById("circle");

let changeStyleButton = document.getElementById("changeStyle");

changeStyleButton.addEventListener("click", function () {

if (circle.classList.contains("circle")) {

circle.classList.remove("circle");

circle.classList.add("square");

} else {

circle.classList.remove("square");

circle.classList.add("circle");

}

});

</script>

**Examples of Parent Node Traversal**

**Example 1**: Accessing the Parent Node

html

Copy code

<div id="parent">

<p id="child">This is a child element.</p>

</div>

<script>

let child = document.getElementById("child");

let parent = child.parentNode;

console.log(parent); // Logs the <div id="parent"> element

</script>

**Example 2**: Changing the Parent Node's Style

<div id="parent" style="padding: 20px; background-color: lightgray;">

<p id="child">This is a child element.</p>

</div>

<script>

let child = document.getElementById("child");

let parent = child.parentNode;

parent.style.border = "2px solid black"; // Adds a border to the parent <div>

</script>

**Examples of Child Node Traversal**

**Example 3**: Accessing All Child Nodes

<ul id="list">

<li>Item 1</li>

<li>Item 2</li>

<li>Item 3</li>

</ul>

<script>

let list = document.getElementById("list");

let children = list.childNodes; // Includes text nodes and elements

console.log(children); // Logs a NodeList including text and <li> elements

</script>

**Example 4**: Accessing Only Child Elements

<ul id="list">

<li>Item 1</li>

<li>Item 2</li>

<li>Item 3</li>

</ul>

<script>

let list = document.getElementById("list");

let children = list.children; // Includes only <li> elements

console.log(children); // Logs an HTMLCollection of <li> elements

</script>

**Example 5**: Highlighting the First Child Element

<ul id="list">

<li>Item 1</li>

<li>Item 2</li>

<li>Item 3</li>

</ul>

<script>

let list = document.getElementById("list");

let firstChild = list.firstElementChild;

firstChild.style.color = "red"; // Changes the color of "Item 1" to red

</script>

**Examples of Sibling Node Traversal**

**Example 6**: Accessing the Next Sibling Node

<ul id="list">

<li id="item1">Item 1</li>

<li id="item2">Item 2</li>

<li id="item3">Item 3</li>

</ul>

<script>

let item1 = document.getElementById("item1");

let nextSibling = item1.nextElementSibling;

console.log(nextSibling); // Logs the <li> element with "Item 2"

</script>

**Example 7**: Changing the Style of a Sibling Node

<ul id="list">

<li id="item1">Item 1</li>

<li id="item2">Item 2</li>

<li id="item3">Item 3</li>

</ul>

<script>

let item2 = document.getElementById("item2");

let prevSibling = item2.previousElementSibling;

prevSibling.style.fontWeight = "bold"; // Makes "Item 1" bold

</script>

**Example 8**: Accessing a Grandchild Node

<div id="grandparent">

<div id="parent">

<p id="child">Grandchild Element</p>

</div>

</div>

<script>

let grandparent = document.getElementById("grandparent");

let grandchild = grandparent.firstElementChild.firstElementChild;

console.log(grandchild); // Logs the <p id="child"> element

</script>

**5. Practical Application: Highlight Active Navigation Link**

<ul id="navbar">

<li class="nav-item active">Home</li>

<li class="nav-item">About</li>

<li class="nav-item">Contact</li>

</ul>

<script>

let activeItem = document.querySelector(".active");

let nextItem = activeItem.nextElementSibling;

// Highlight the next navigation item

nextItem.style.color = "blue";

nextItem.style.fontWeight = "bold";

</script>

**Example: Accessing Various Form Elements**

<form id="userForm">

<input type="text" id="username" placeholder="Username" />

<textarea id="comments" placeholder="Comments"></textarea>

<select id="gender">

<option value="male">Male</option>

<option value="female">Female</option>

</select>

<input type="checkbox" id="subscribe" /> Subscribe to Newsletter

<input type="radio" name="role" value="admin" id="adminRole" /> Admin

<input type="radio" name="role" value="user" id="userRole" checked /> User

<button type="submit" id="submitBtn">Submit</button>

</form>

<script>

const username = document.getElementById("username");

const comments = document.getElementById("comments");

const gender = document.getElementById("gender");

const subscribe = document.getElementById("subscribe");

const adminRole = document.getElementById("adminRole");

const userRole = document.getElementById("userRole");

console.log(username, comments, gender, subscribe, adminRole, userRole);

</script>

**2. Reading and Updating Input Values**

**Text Input and Textarea**

Use the value property for input and textarea.

<form id="form1">

<input type="text" id="username" value="JohnDoe" />

<textarea id="comments">Enter your comments...</textarea>

</form>

<script>

const username = document.getElementById("username");

const comments = document.getElementById("comments");

console.log(username.value); // Output: JohnDoe

console.log(comments.value); // Output: Enter your comments...

// Update values

username.value = "JaneDoe";

comments.value = "Updated comments";

</script>

**Select Element**

<select id="gender">

<option value="male">Male</option>

<option value="female" selected>Female</option>

</select>

<script>

const gender = document.getElementById("gender");

console.log(gender.value); // Output: female

console.log(gender.options[gender.selectedIndex].text); // Output: Female

</script>

**Checkbox and Radio Buttons**

<input type="checkbox" id="subscribe" checked />

<input type="radio" name="role" id="adminRole" value="admin" />

<input type="radio" name="role" id="userRole" value="user" checked />

<script>

const subscribe = document.getElementById("subscribe");

const adminRole = document.getElementById("adminRole");

const userRole = document.getElementById("userRole");

console.log(subscribe.checked); // Output: true

console.log(adminRole.checked); // Output: false

console.log(userRole.checked); // Output: true

</script>

**3. Handling User Input Events**

**Real-Time Validation**

<form>

<input type="text" id="username" placeholder="Username (min 5 chars)" />

<p id="error" style="color: red; display: none;">Username must be at least 5 characters long!</p>

</form>

<script>

const username = document.getElementById("username");

const error = document.getElementById("error");

username.addEventListener("input", () => {

if (username.value.length < 5) {

error.style.display = "block";

} else {

error.style.display = "none";

}

});

</script>

**Handling Select and Checkbox Changes**

<form>

<select id="gender">

<option value="male">Male</option>

<option value="female">Female</option>

</select>

<input type="checkbox" id="subscribe" /> Subscribe

</form>

<script>

const gender = document.getElementById("gender");

const subscribe = document.getElementById("subscribe");

gender.addEventListener("change", () => {

console.log("Selected Gender:", gender.value);

});

subscribe.addEventListener("change", () => {

console.log("Subscribed:", subscribe.checked);

});

</script>

**4. Adding and Removing Form Elements Dynamically**

**Adding Elements**

<form id="form1">

<button type="button" id="addInput">Add Input</button>

</form>

<script>

const form = document.getElementById("form1");

const addInput = document.getElementById("addInput");

addInput.addEventListener("click", () => {

const newInput = document.createElement("input");

newInput.type = "text";

newInput.placeholder = "New Input";

form.appendChild(newInput);

});

</script>

**Removing Elements**

<button id="removeLast">Remove Last Input</button>

<script>

const removeLast = document.getElementById("removeLast");

removeLast.addEventListener("click", () => {

const inputs = document.querySelectorAll("input");

if (inputs.length > 0) {

inputs[inputs.length - 1].remove();

}

});

</script>

**5. Form Submission and Data Handling**

**Prevent Default Submission**

<form id="userForm">

<input type="text" id="username" placeholder="Enter username" />

<button type="submit">Submit</button>

</form>

<p id="result"></p>

<script>

const form = document.getElementById("userForm");

const result = document.getElementById("result");

form.addEventListener("submit", (event) => {

event.preventDefault();

const username = document.getElementById("username").value;

result.textContent = `Submitted Username: ${username}`;

});

</script>

**6. Validation and Error Handling**

**Custom Validation**

<form id="userForm">

<input type="email" id="email" placeholder="Enter email" />

<button type="submit">Submit</button>

</form>

<p id="error" style="color: red; display: none;">Invalid email!</p>

<script>

const form = document.getElementById("userForm");

const email = document.getElementById("email");

const error = document.getElementById("error");

form.addEventListener("submit", (event) => {

if (!email.value.includes("@")) {

event.preventDefault();

error.style.display = "block";

} else {

error.style.display = "none";

alert("Form submitted successfully!");

}

});

</script>

**7. Advanced Techniques**

**Disabling Form Elements**

<form>

<input type="text" id="username" placeholder="Username" />

<button type="button" id="toggleDisable">Disable/Enable</button>

</form>

<script>

const username = document.getElementById("username");

const toggleDisable = document.getElementById("toggleDisable");

toggleDisable.addEventListener("click", () => {

username.disabled = !username.disabled;

});

</script>

**Syntax**

element.addEventListener(eventType, eventHandler, useCapture);

**Handling Events: Examples**

**Click Event**

<button id="myButton">Click Me</button>

<script>

const button = document.getElementById("myButton");

button.addEventListener("click", () => {

alert("Button was clicked!");

});

</script>

**Mouse Events**

<div id="hoverDiv" style="width: 200px; height: 100px; background-color: lightblue;">

Hover over me!

</div>

<script>

const hoverDiv = document.getElementById("hoverDiv");

hoverDiv.addEventListener("mouseover", () => {

hoverDiv.style.backgroundColor = "lightgreen";

});

hoverDiv.addEventListener("mouseout", () => {

hoverDiv.style.backgroundColor = "lightblue";

});

</script>

**Keyboard Events**

<input type="text" id="nameInput" placeholder="Type your name" />

<script>

const nameInput = document.getElementById("nameInput");

nameInput.addEventListener("keydown", (event) => {

console.log(`Key Pressed: ${event.key}`);

});

nameInput.addEventListener("keyup", () => {

console.log("Key released.");

});

</script>

**Example: Accessing Event Properties**

<button id="eventButton">Click Me</button>

<script>

const eventButton = document.getElementById("eventButton");

eventButton.addEventListener("click", (event) => {

console.log("Event Type:", event.type); // Output: click

console.log("Target Element:", event.target); // Output: <button>

console.log("Mouse X:", event.clientX, "Mouse Y:", event.clientY); // Mouse position

});

</script>

**Example: Adding and Removing Listeners**

<button id="toggleButton">Enable/Disable Alert</button>

<script>

const toggleButton = document.getElementById("toggleButton");

function showAlert() {

alert("Event Triggered!");

}

let enabled = true;

toggleButton.addEventListener("click", () => {

if (enabled) {

toggleButton.removeEventListener("click", showAlert);

toggleButton.textContent = "Enable Alert";

} else {

toggleButton.addEventListener("click", showAlert);

toggleButton.textContent = "Disable Alert";

}

enabled = !enabled;

});

toggleButton.addEventListener("click", showAlert);

</script>

**Example: Event Bubbling**

<div id="outer" style="padding: 20px; background-color: lightgray;">

Outer Div

<div id="inner" style="padding: 20px; background-color: lightblue;">Inner Div</div>

</div>

<script>

const outer = document.getElementById("outer");

const inner = document.getElementById("inner");

outer.addEventListener("click", () => {

alert("Outer Div Clicked");

});

inner.addEventListener("click", (event) => {

alert("Inner Div Clicked");

event.stopPropagation(); // Prevents bubbling to outer div

});

</script>

**Example: Delegation for Dynamic Buttons**

<div id="buttonContainer">

<button>Button 1</button>

<button>Button 2</button>

</div>

<script>

const container = document.getElementById("buttonContainer");

container.addEventListener("click", (event) => {

if (event.target.tagName === "BUTTON") {

alert(`${event.target.textContent} Clicked`);

}

});

</script>

**Advanced Examples**

**Double Click Event**

<button id="doubleClickButton">Double Click Me</button>

<script>

const doubleClickButton = document.getElementById("doubleClickButton");

doubleClickButton.addEventListener("dblclick", () => {

alert("Button was double-clicked!");

});

</script>

**Custom Events**

<button id="customEventButton">Trigger Custom Event</button>

<script>

const button = document.getElementById("customEventButton");

button.addEventListener("myCustomEvent", (event) => {

alert(`Custom Event Triggered with Data: ${event.detail}`);

});

button.addEventListener("click", () => {

const customEvent = new CustomEvent("myCustomEvent", { detail: "Hello World!" });

button.dispatchEvent(customEvent);

});

</script>