💻 Slide Deck: The DOM Manipulation Toolkit

Slide 1: Introduction

Title: Mastering JavaScript DOM

Subtitle: The 3 Tools You Need to Build Anything (Calculators, To-Do Lists, Games)

Instructor: Muhammad Sarim

Slide 2: Lab #1 - The "Digital Garden"

Topic: Creating & Placing Elements (`createElement`, `appendChild`)

📌 Definitions

- **`document.createElement("tag")`**: The "Factory." It manufactures a brand new HTML element (like an `h1` or `div`) in the computer's memory. It is invisible at this stage.
- **`parent.appendChild(child)`**: The "Delivery Truck." It takes the invisible element from memory and physically attaches it to the webpage so users can see it.

💡 The Analogy

- **The Factory:** You build a car in a factory (Create Element).
- **The Showroom:** Nobody sees the car until you ship it to the showroom (Append Child).

💻 Code Example

JavaScript
```
function plantFlower() {
  // 1. Manufacture the H1 (Invisible)
  const flower = document.createElement("h1");

  // 2. Decorate it
  flower.innerText = " 🌹 ";

  // 3. Ship it to the Garden Div
  const garden = document.getElementById("garden");
  garden.appendChild(flower);
}
```

Slide 3: Lab #2 - The "Name Badge Maker"

Topic: Moving Data (`value` vs `innerText`)

📌 Definitions

- **`input.value`**: This is how we grab text that a **user has typed** into a form box.
- **`element.innerText`**: This is how we write text inside a regular HTML tag (like a `p`, `h1`, or `div`).

💡 The Analogy

- **The Bucket (Input):** The input box is a bucket where the user pours water (data).
- **The Cup (Element):** We pour the water *from* the bucket (`.value`) *into* the cup (`.innerText`).

💻 Code Example

JavaScript
```javascript
function makeBadge() {
    // 1. Get text FROM the input box
    const nameInput = document.getElementById("nameInput");
    const userName = nameInput.value;

    // 2. Create a new Paragraph
    const badge = document.createElement("p");

    // 3. Put text INTO the paragraph
    badge.innerText = "Hello, " + userName;

    // 4. Show it on screen
    document.body.appendChild(badge);
}
```

Slide 4: Lab #3 - The "Ghost Hunter"

Topic: Interaction & Deletion (`this`, `remove`)

📌 Definitions

- **`this`**: A special keyword. When used on a button, it means "ME! The specific button that was just clicked."
- **`element.remove()`**: A method that completely deletes an HTML element from the page.

💡 The Analogy

- **Self-Destruct Button:** Imagine a button that says "Do Not Press." When you press it, the button itself explodes. That is `this.remove()`.

💻 Code Example

HTML
```html
<button onclick="killGhost(this)">👻 Ghost 1</button>
<button onclick="killGhost(this)">👻 Ghost 2</button>

<script>
  function killGhost(element) {
    // 'element' is the button we clicked
    element.remove();
  }
</script>
```

🚀 The Final Code (For Class Demo)

HTML
```
<!DOCTYPE html>
<html>
<head>
    <style>
        body { font-family: sans-serif; padding: 20px; }
        .todo-item {
            padding: 10px;
            border-bottom: 1px solid #ddd;
            display: flex;
            justify-content: space-between;
        }
        .delete-btn { background: red; color: white; border: none; cursor: pointer; }
    </style>
</head>
<body>

    <h2>My To-Do List</h2>
    <input type="text" id="myInput" placeholder="Task...">
    <button onclick="addTask()">Add Task</button>

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

    <script>
        function addTask() {
            // STEP 1: Get Data (Lab 2)
            const input = document.getElementById("myInput");
            const text = input.value;

            // STEP 2: Create Element (Lab 1)
            const newItem = document.createElement("div");
            newItem.className = "todo-item";

            // We put the text AND a delete button inside the new item
            // Notice: We use 'this.parentElement' to delete the whole row, not just the button!
            newItem.innerHTML = `
                <span>${text}</span>
                <button class="delete-btn" onclick="deleteTask(this)">Delete</button>
            `;

            // STEP 3: Append (Lab 1)
            const list = document.getElementById("container");
            list.appendChild(newItem);

            // Cleanup
            input.value = "";
        }

        function deleteTask(btn) {
            // STEP 4: Remove (Lab 3)
            // We delete the PARENT of the button (the whole div), not just the button
            btn.parentElement.remove();
        }
    </script>

</body>
</html>
```