Peter Wolf
<Software Developer & Advisor />

JS – Grundkurs

TO LIST – ÜBUNG

HTML – Grundstruktur

```html
<!DOCTYPE html>
<html lang="de">
<head>
  <meta charset="UTF-8">
  <title>ToDo Liste</title>
  <style>
    body { font-family: sans-serif; margin: 2rem; }
    li { margin: 4px 0; }
    button { margin-left: 10px; }
  </style>
</head>
<body>

  <h1>📝 ToDo Liste</h1>

  <!-- Eingabefeld und Buttons -->
  <input id="taskInput" type="text" placeholder="Neue Aufgabe eingeben">
  <button id="addBtn">Hinzufügen</button>
  <button id="clearBtn">Liste löschen</button>

  <!-- Hier erscheinen die Aufgaben -->
  <ul id="taskList"></ul>

  <script src="app.js"></script>
</body>
</html>
```

Peter Wolf
<Software Developer & Advisor />

Javascript Datei

```javascript
// === Schritt 1: Elemente aus dem DOM holen ===
const input = document.getElementById("taskInput");
const addBtn = document.getElementById("addBtn");
const clearBtn = document.getElementById("clearBtn");
const taskList = document.getElementById("taskList");

// === Schritt 2: Aufgabe hinzufügen ===
addBtn.addEventListener("click", addTask);

function addTask() {
  const taskText = input.value.trim(); // Leerzeichen entfernen

  if (taskText === "") {
    alert("Bitte gib eine Aufgabe ein!");
    return;
  }

  // Neues <li> Element erzeugen
  const li = document.createElement("li");
  li.textContent = taskText;

  // Löschen-Button hinzufügen
  const deleteBtn = document.createElement("button");
  deleteBtn.textContent = "Löschen";
  deleteBtn.addEventListener("click", () => {
    li.remove(); // entfernt nur dieses eine Listenelement
  });

  // Button ans <li> anhängen
  li.appendChild(deleteBtn);

  // <li> in die <ul> einfügen
  taskList.appendChild(li);

  // Eingabefeld leeren
  input.value = "";
  input.focus();
}

// === Schritt 3: Gesamte Liste löschen ===
clearBtn.addEventListener("click", () => {
  taskList.innerHTML = ""; // leert die gesamte Liste
});
```

Peter Wolf
<Software Developer & Advisor />

Hinweis und Tips:

Warum `li.remove()` nur das eine `<li>` löscht, auf welches es sich bezieht?

Kurz gesagt:

➡️ **JavaScript weiß, welches `<li>` gemeint ist**, weil die Funktion, die du im Event-Listener definierst, **in ihrem Kontext (Scope)** genau **auf das eine Element zugreifen kann, zu dem sie gehört**.

Das passiert durch ein Konzept namens **Closure** (Abschluss).

CLOSURE – genauer erklärt.

Ein Closure ist wie ein „Gedächtnis“ einer Funktion.

Wenn du in einer Funktion auf eine Variable zugreifst, die **außerhalb** dieser Funktion definiert wurde,
„merkt“ sich JavaScript den Wert, selbst wenn die äußere Funktion längst fertig ist.

In unserem ToDo Beispiel würde es dann „unter der Haube“ so gespeichert werden.

Jede Aufgabe hat ihr eigenes `li` und ihren eigenen Listener

Wenn du drei Aufgaben hinzufügst, passiert folgendes:

Aufgabe	`<li>` Variable	Event-Handler enthält	Ergebnis
„Einkaufen“	li_1	li_1`.remove()`	löscht li_1
„Putzen“	li_2	li_2`.remove()`	löscht li_2
„Kochen“	li_3	li_3`.remove()`	löscht li_3