



dev Senior

www.devseniorcode.com



✓ Guía para el Proyecto: Lista de Tareas Interactiva

Objetivo: Construir una página web dinámica que permita agregar, completar y eliminar tareas, aplicando HTML, CSS y JavaScript.

www.devseniorcode.com



Estructura del Proyecto

```
proyecto-todo/  
|— index.html  
|— style.css  
└— script.js
```





Paso 1: Crear la estructura HTML

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Lista de Tareas</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <header>
    <h1>Lista de Tareas</h1>
  </header>
  <main>
    <section class="todo-app">
      <input type="text" id="taskInput" placeholder="Escribe una tarea...">
      <button id="addBtn">Agregar</button>
      <ul id="taskList"></ul>
    </section>
  </main>
  <script src="script.js"></script>
</body>
</html>
```



Explicación

`<!DOCTYPE html>` → indica que usamos HTML5

`<header>` y `<main>` → etiquetas semánticas

`<input>` → donde el usuario escribe la tarea

`<button>` → botón para agregar tareas

`` → lista donde se mostrarán las tareas

`<script src="script.js">` → enlazamos el archivo JS



Captura esperada:

Una página con un título, un input, un botón y un espacio para la lista.



dev
Senior



Paso 2: Diseñar con CSS

```
body {
  font-family: Arial, sans-serif;
  background: #f4f4f4;
  margin: 0;
  padding: 0;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}

.todo-app {
  background: #fff;
  padding: 20px;
  border-radius: 10px;
  box-shadow: 0 0 10px rgba(0,0,0,0.1);
  width: 300px;
  text-align: center;
}

input {
  width: 70%;
  padding: 10px;
  margin-bottom: 10px;
}

button {
  padding: 10px;
  background: #28a745;
  color: #fff;
  border: none;
  cursor: pointer;
}
```



```
button:hover {
  background: #218838;
}

ul {
  list-style: none;
  padding: 0;
}

li {
  display: flex;
  justify-content: space-between;
  padding: 8px;
  background: #f9f9f9;
  margin-bottom: 5px;
  border-radius: 5px;
}

li.completed {
  text-decoration: line-through;
  color: gray;
}
```

Explicación

- ✓ Centramos la app en la pantalla con Flexbox.
- ✓ Aplicamos bordes redondeados y sombras para estética.
- ✓ Clase `.completed` → marca tareas terminadas.



Captura esperada:

Caja blanca centrada con un input, botón verde y lista limpia.





Paso 3: Programar la lógica en JavaScript

```
// Selección de elementos
const addBtn = document.getElementById('addBtn');
const taskInput = document.getElementById('taskInput');
const taskList = document.getElementById('taskList');

// Función para agregar tarea
function addTask() {
  const taskText = taskInput.value.trim();

  if (taskText === "") {
    alert("Por favor escribe una tarea.");
    return;
  }

  // Crear elemento li
  const li = document.createElement('li');
  li.textContent = taskText;

  // Botón completar
  const completeBtn = document.createElement('button');
  completeBtn.textContent = "✓";
  completeBtn.style.marginRight = "10px";
  completeBtn.addEventListener('click', () => {
    li.classList.toggle('completed');
  });
```

```
// Botón eliminar
const deleteBtn = document.createElement('button');
deleteBtn.textContent = "✕";
deleteBtn.addEventListener('click', () => {
  taskList.removeChild(li);
});

li.prepend(completeBtn);
li.appendChild(deleteBtn);
taskList.appendChild(li);

// Limpiar input
taskInput.value = "";
}

// Evento click
addBtn.addEventListener('click', addTask);

// Evento Enter
taskInput.addEventListener('keypress', (e) => {
  if (e.key === "Enter") {
    addTask();
  }
});
```


Explicación

- ✓ `document.getElementById` → seleccionamos elementos del DOM
- ✓ `trim()` → eliminamos espacios innecesarios
- ✓ `createElement('li')` → creamos elementos dinámicamente
- ✓ `addEventListener` → escuchamos eventos (click, Enter)
- ✓ `toggle('completed')` → marca o desmarca tareas



Captura esperada:

La app funcionando: puedes agregar, marcar y eliminar tareas.



dev
Senior

Cierre

¡Felicitaciones, equipo!

Hoy cerramos el **Módulo 1: Fundamentos de JavaScript**, el primer gran paso en su camino hacia convertirse en **Desarrolladores Full Stack**.

En estas semanas:

- ✓ Aprendieron los **conceptos base** del lenguaje JavaScript.
- ✓ Dominaron **tipos de datos, operadores, estructuras de control y funciones**.
- ✓ Aplicaron **buenas prácticas** y desarrollaron **proyectos prácticos** para sentar las bases del desarrollo web.

Recuerden: **JavaScript es el corazón del desarrollo moderno** y todo lo que viene (Node.js, Express, MongoDB, Angular) se apoyará en lo que ya aprendieron aquí. Este módulo no es un punto final, **es el inicio de su transformación**. Cada línea de código que escriban los acerca más a su meta: convertirse en **líderes Full Stack del mercado tecnológico**.

¡Prepárense para el **Módulo 2**, donde llevaremos su lógica al siguiente nivel y comenzaremos a construir aplicaciones dinámicas, robustas y escalables!

Gracias por su esfuerzo, compromiso y dedicación. ¡Lo mejor está por venir! 🚀

dev Senior

