Universidad de las Fuerzas Armadas “ESPE”

Programación Integrativa

Stephano Santin

NRC 16496

Manual 2

Paso 1: Crear la estructura de carpetas:

task-manager/

├── css/

│ └── styles.css

├── js/

│ ├── task.js

│ └── task-list.js

├── index.html

└── app.js

css/: Contendrá el archivo de estilos CSS.

js/: Contendrá los archivos JavaScript, uno para cada clase (task.js y task-list.js) y un archivo principal (app.js) para la lógica de la aplicación.

index.html: Es la página principal de la aplicación.

Paso 2: Definición de las clases

Definir la clase Task en js/task.js:

javascript

// js/task.js

```
// Clase Task que representa una tarea individual
class Task {
    constructor(id, description, completed = false) {
        this.id = id;
        this.description = description;
        this.completed = completed;
    }

    complete() {
        this.completed = true;
    }

    edit(newDescription) {
        this.description = newDescription;
    }
}
```

Definir la clase TaskList en js/task-list.js:

javascript

Copiar código

```
// js/task-list.js

// Clase TaskList que maneja una lista de tareas
class TaskList {
```

```
    constructor() {
        this.tasks = [];
        this.currentId = 1; // Id inicial de las tareas
    }

    addTask(description) {
        const task = new Task(this.currentId++, description);
        this.tasks.push(task);
    }

    deleteTask(taskId) {
        this.tasks = this.tasks.filter(task => task.id !== taskId);
    }

    getTask(taskId) {
        return this.tasks.find(task => task.id === taskId);
    }

    getAllTasks() {
        return this.tasks;
    }
}
```

Paso 3: Interfaz de Usuario (UI)

Crear index.html:

html

```
<!-- index.html -->

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Task Manager</title>
    <link rel="stylesheet" href="css/styles.css">
</head>
<body>
    <div class="container">
        <h1>Task Manager</h1>
        <div id="taskList"></div>
        <form id="taskForm">
            <input type="text" id="taskInput" placeholder="Enter task description...">
            <button type="submit">Add Task</button>
        </form>
    </div>
```

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

Paso 4: Estilos CSS

Estilizar la aplicación en css/styles.css:

```
css
/* css/styles.css */

body {
    font-family: Arial, sans-serif;
    background-color: #f0f0f0;
    margin: 0;
    padding: 0;
}

.container {
    max-width: 800px;
    margin: 20px auto;
    background-color: #fff;
    padding: 20px;
```

```
    border-radius: 8px;
    box-shadow: 0px 0px 10px 0px rgba(0, 0, 0, 0.1);
}

.task-item {
    margin-bottom: 10px;
    padding: 10px;
    border: 1px solid #ccc;
    border-radius: 4px;
    display: flex;
    align-items: center;
    justify-content: space-between;
}

.task-item .description {
    margin-right: 10px;
}

.task-item .actions button {
    margin-left: 10px;
}
```

Paso 5: Lógica de la Aplicación

Crear app.js para manejar la lógica de la aplicación:

javascript

```
// app.js

// Creación de una instancia de TaskList
const taskList = new TaskList();

// Referencias a elementos del DOM
const taskForm = document.getElementById('taskForm');
const taskInput = document.getElementById('taskInput');
const taskListContainer = document.getElementById('taskList');

// Función para renderizar las tareas en la UI
function renderTasks() {
    taskListContainer.innerHTML = '';

    taskList.getAllTasks().forEach(task => {
        const taskElement = document.createElement('div');
        taskElement.classList.add('task-item');
        taskElement.innerHTML = `
            <span class="description">${task.description}</span>
            <div class="actions">
                <button onclick="completeTask(${task.id})">Complete</button>
                <button onclick="deleteTask(${task.id})">Delete</button>
            </div>
```

```
        `;
        if (task.completed) {
            taskElement.classList.add('completed');
        }
        taskListContainer.appendChild(taskElement);
    });
}

// Función para añadir una tarea
function addTask(description) {
    taskList.addTask(description);
    renderTasks();
}

// Función para eliminar una tarea
function deleteTask(taskId) {
    taskList.deleteTask(taskId);
    renderTasks();
}

// Función para completar una tarea
function completeTask(taskId) {
    const task = taskList.getTask(taskId);
    task.complete();
```

```
    renderTasks();
}

// Evento submit del formulario para añadir una tarea
taskForm.addEventListener('submit', function(event) {
    event.preventDefault();
    const taskDescription = taskInput.value.trim();
    if (taskDescription !== '') {
        addTask(taskDescription);
        taskInput.value = '';
    }
});

// Inicializar la lista de tareas en la UI
renderTasks();
```