

## Task 9: To-Do List Application with Local Storage

### Objective:

Create a to-do list application that allows users to add tasks, remove tasks, and display tasks. The tasks should be stored in the browser's local storage, so they persist across page reloads. The application should support the following operations:

- Add a new task
- Remove an existing task
- Display all tasks

### Pre-requisites:

- Basic JavaScript (variables, functions, arrays)
- DOM Manipulation
- Event Listeners
- Local Storage API
- `JSON.stringify` and `JSON.parse`

### Concepts Covered:

- Storing and retrieving data from local storage
- DOM manipulation
- Event handling
- JSON serialization and deserialization

### Setup:

#### Install Node.js:

- Ensure Node.js is installed on your machine. You can download it from [nodejs.org](https://nodejs.org).

### Tasks:

#### 1. Define the `todoList` Module:

- **Task:**
  - Define a module named `todoList`.
  - The `todoList` module should encapsulate an array `tasks` initialized with the tasks stored in local storage (if any).
  - The module should expose the following public methods:
    - `addTask(task)` : Adds a new task to the list and updates local storage.
    - `removeTask(taskIndex)` : Removes a task by its index from the list and updates local storage.
    - `getTasks()` : Returns the current list of tasks.
  - Ensure that the `tasks` array is not directly accessible from outside the module.
- **Outcome:**

- The module should manage tasks correctly and interact with local storage.

## 2. Create HTML Structure:

### ◦ Task:

- Create an HTML file named `index.html` with the following structure:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>To-Do List</title>
</head>
<body>
  <h1>To-Do List</h1>
  <input type="text" id="taskInput" placeholder="New task">
  <button id="addTaskButton">Add Task</button>
  <ul id="taskList"></ul>
  <script src="script.js"></script>
  <script src="app.js"></script>
</body>
</html>
```

### ◦ Outcome:

- Ensure the HTML structure is correct and includes elements for inputting and displaying tasks.

## 3. Implement UI Interaction in `app.js` :

### ◦ Task:

- Create a JavaScript file named `app.js` .
- In `app.js` , implement the following functionality:
  - Select the input element, add button, and task list.
  - Add an event listener to the add button to add a new task using `todoList.addTask` .
  - Add an event listener to the task list to remove a task when clicked.
  - Display the list of tasks on the webpage and update it dynamically as tasks are added or removed.

### ◦ Outcome:

- Ensure the UI interacts correctly with the `todoList` module and updates dynamically.

## Instructions:

### • Perform the following tasks:

- Write the required code in `index.html` , `script.js` , and `app.js` .
- Open `index.html` in a web browser to ensure the code executes without errors and demonstrates the use of basic JavaScript concepts for DOM manipulation and local storage interaction.









### Example Input:

1. Actions: Add "Buy groceries", add "Walk the dog", remove "Buy groceries"

### Expected Output:

1. After adding "Buy groceries": ["Buy groceries"]
2. After adding "Walk the dog": ["Buy groceries", "Walk the dog"]
3. After removing "Buy groceries": ["Walk the dog"]

### Resources:

-  **Grammar and types - JavaScript | MDN**  
This chapter discusses JavaScript's basic grammar, variable declarations, data types and li...  
 [https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Grammar\\_and\\_types#declarations](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Grammar_and_types#declarations)
-  **EventTarget: addEventListener() method - Web APIs | MDN**  
The addEventListener() method of the EventTarget interface sets up a function that will b...  
 <https://developer.mozilla.org/en-US/docs/Web/API/EventTarget/addEventListener>
-  **Document Object Model (DOM) - Web APIs | MDN**  
The Document Object Model (DOM) connects web pages to scripts or programming langu...  
 [https://developer.mozilla.org/en-US/docs/Web/API/Document\\_Object\\_Model](https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model)
-  **Window: localStorage property - Web APIs | MDN**  
The localStorage read-only property of the window interface allows you to access a Storag...  
 <https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage>

### Videos:

# Using Local Storage to store objects in JavaScript

dcode

JS

dc dcode 6:58 min 161,594 Views 2,545 Likes

## GitHub Instructions:

### 1. Open in Visual Studio Code:

- After clicking on the "Open in Visual Studio Code" button from the GitHub Classroom confirmation page, Visual Studio Code (VSCode) will open the repository directly.
- If prompted, select "Open" or "Allow" to open the repository in VSCode.

### 2. Open the Terminal in VSCode:

- In VSCode, open a terminal by selecting Terminal > New Terminal from the top menu.

### 3. Complete the Task:

- In VSCode, write your solution in the `index.html`, `script.js`, and `app.js` files.

### 4. Run and Test Your Code:

- Open `index.html` in a web browser to ensure it works correctly.

### 5. Commit Your Changes:

- In the VSCode terminal, add your changes to git:

```
git add index.html script.js app.js
```

- Commit your changes with a meaningful message:

```
git commit -m "Completed task 9"
```

### 6. Push Your Changes to Your Repository:

- Push your changes to your forked repository:

```
git push origin main
```

## 7. Create a Pull Request:

- Go to your repository on GitHub.
- Click on the "Pull Requests" tab.
- Click the "New Pull Request" button.
- Ensure the base repository is the original template repository and the base branch is `main`.
- Ensure the head repository is your forked repository and the compare branch is `main`.
- Click "Create Pull Request".
- Add a title and description for your pull request and submit it.

## Summary of Commands:

```
# Open in Visual Studio Code

# Open terminal in VSCode

# Complete the task by editing index.html, script.js, and app.js

# Add, commit, and push your changes
git add index.html script.js app.js
git commit -m "Completed task 9"
git push origin main

# Create a pull request on GitHub
```

## Need Help?



### Task 9 solution

Code Hints — `script.js`: — Ensure your JavaScript module manages tasks correctly and interac...



w3o NFThing Last Edited 7/3/2024