**федеральное государственное автономное образовательное учреждение высшего образования**

**МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Факультет информационных технологий**

***Кафедра Информатики и информационных технологий***

**ЛАБОРАТОРНАЯ РАБОТА № \_1**

**Дисциплина:** Функциональное программирование

**Тема:** Применение функционального программирования в JS

**Выполнил: студент группы**  **221-3711**

**Закалкин А.А.**

(Фамилия И.О.)

**Дата, подпись**  13.03.25

(Дата) (Подпись)

**Проверил:**

(Фамилия И.О., степень, звание) **(Оценка)**

**Дата, подпись**

(Дата) (Подпись)

**Замечания:**

**Москва 2025**

**Лабораторная работа № 2.**

**Применение функционального программирования в JS.**

**Цель:**

Применить принципы функционального программирования для разработки  небольшого веб-приложения.

**Задание:**

Разработайте веб-приложение "Список задач", которое позволяет пользователю:

* Добавлять новые задачи.
* Отмечать задачи как выполненные.
* Удалять задачи.
* Фильтровать задачи по статусу (выполненные/невыполненные).

**Требования:**

* Используйте принципы функционального программирования, такие как иммутабельность данных и чистые функции.
* Используйте функции высшего порядка для обработки списка задач.
* Веб-приложение должно быть реализовано с использованием HTML, CSS и JavaScript.
* Интерфейс должен быть интуитивно понятным и удобным для пользователя.

**Ход работы:**

**saveTasks()**

Сохраняет текущий список задач в localStorage, чтобы они сохранялись после перезагрузки страницы

**renderTasks()**

Фильтрует задачи в зависимости от выбранного режима (все, активные, выполненные) и обновляет интерфейс, отображая задачи в списке

**taskToHTML(task, index)**

Формирует HTML-разметку для задачи, включая текст, статус выполнения и кнопки управления (выполнить/отменить и удалить)

**addTask()**

Добавляет новую задачу в список, если введен текст, обновляет localStorage и перерисовывает интерфейс

**toggleTask(index)**

Меняет статус выполнения задачи (выполнена/не выполнена), обновляет localStorage и интерфейс

**deleteTask(index)**

Удаляет задачу по индексу, обновляет localStorage и интерфейс

**filterTasks(newFilter)**

Изменяет текущий фильтр задач (все, активные, выполненные) и вызывает renderTasks()

Код программы:

<!DOCTYPE html>

<html lang="ru">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Список задач</title>

    <style>

        body {

            font-family: Arial, sans-serif;

            max-width: 400px;

            margin: 20px auto;

            text-align: center;

        }

        ul {

            list-style: none;

            padding: 0;

        }

        li {

            display: flex;

            justify-content: space-between;

            align-items: center;

            padding: 8px;

            border-bottom: 1px solid #ddd;

        }

        .completed {

            text-decoration: line-through;

            color: gray;

        }

        .button-complete {

            margin-left: auto;

            margin-right: 10px;

        }

        .button-filters {

            margin-top: 10px;

        }

    </style>

</head>

<body>

    <h1>Список задач</h1>

    <input type="text" id="taskInput" placeholder="Введите задачу">

    <button onclick="addTask()">Добавить</button>

    <div class="button-filters">

        <button onclick="filterTasks('all')">Все</button>

        <button onclick="filterTasks('active')">Активные</button>

        <button onclick="filterTasks('completed')">Выполненные</button>

    </div>

    <ul id="taskList"></ul>

    <script>

        let tasks = JSON.parse(localStorage.getItem("tasks")) || [];

        let filter = 'all';

        const saveTasks = () => {

            localStorage.setItem("tasks", JSON.stringify(tasks));

        };

        const renderTasks = () => {

            const filteredTasks = tasks.filter(task => {

                if (filter === 'active') return !task.completed;

                if (filter === 'completed') return task.completed;

                return true;

            });

            document.getElementById("taskList").innerHTML = filteredTasks.map(taskToHTML).join('');

        };

        const taskToHTML = (task, index) => `

            <li>

                <span class="${task.completed ? 'completed' : ''}">${task.text}</span>

                <button class="button-complete" onclick="toggleTask(${index})">${task.completed ? 'Отменить' : 'Выполнить'}</button>

                <button onclick="deleteTask(${index})">Удалить</button>

            </li>

        `;

        const addTask = () => {

            const input = document.getElementById("taskInput");

            const text = input.value.trim();

            if (text) {

                tasks = [...tasks, { text, completed: false }];

                input.value = "";

                saveTasks();

                renderTasks();

            }

        };

        const toggleTask = (index) => {

            tasks = tasks.map((task, i) => i === index ? { ...task, completed: !task.completed } : task);

            saveTasks();

            renderTasks();

        };

        const deleteTask = (index) => {

            tasks = tasks.filter((\_, i) => i !== index);

            saveTasks();

            renderTasks();

        };

        const filterTasks = (newFilter) => {

            filter = newFilter;

            renderTasks();

        };

        renderTasks();

    </script>

</body>

</html>

**Заключение**

В данной лабораторной работе было реализовано веб-приложение "Список задач" с применением принципов функционального программирования.

Используется иммутабельность данных – массив tasks не изменяется напрямую, а создается новая версия через map или filter.

Все функции (renderTasks, taskToHTML, toggleTask, deleteTask, filterTasks) являются чистыми – они не изменяют глобальное состояние напрямую, а оперируют переданными данными.

Применяются функции высшего порядка (map для обновления списка, filter для фильтрации).



Рис. 1 Демонстрация веб-приложения

Ссылка на гитхаб:

<https://github.com/XuMuK57/laba2>