Лабораторна робота №2

Створення клієнтських Web-додатків засобами мови Javascript

Метою лабораторної роботи є оволодіння практичними навичками створення динамічних Web-сайтів у відповідності до загальноприйнятих стандартів.

Завдання на лабораторну роботу:

1. Вивчити основи мови програмування Javascript зокрема, особливості стандарту ECMAScript 2015 (ES6).
2. Набути навичок створення програм, побудованих за патерном MVC (Model-View-Controller).
3. Ознайомитись із реалізацією фонових потоків (WebWorkers) у Web-браузері.
4. Реалізувати функціональні вимоги, визначені у варіанті, відповідно до наступних вимог:

   4.1. Виконати кодування додатку згідно із загальноприйнятими вимогами (див. https://github.com/airbnb/javascript).

   4.2. Організувати код у вигляді модулів шаблону MVC (див. приклад додатку ToDo: https://github.com/Aroxed/js-todo-mvc).

   4.3. Реалізувати виконання фонової задачі за допомогою WebWorker (https://www.w3schools.com/html/html5_webworkers.asp та https://developer.mozilla.org/ru/docs/DOM/Using_web_workers).

   4.4. Як сховище даних використовувати структури даних Javascript: масиви, об’єкти, рядки тощо.

Методичні рекомендації

Запуск додатку “Список справ”

Вихідний код додатку доступний за посиланням: https://github.com/Aroxed/js-todo-mvc.

Для програмування мовою Javascript рекомендовано використовувати середовище розробки Visual Studio Code (VS Code, https://code.visualstudio.com/), яке включає інтеграцію із системами контролю версій (зокрема, Git), засоби відлагодження коду (debugger), підтримку автозавершення (autocomplete) та форматування коду, а також інші функції.

Після встановлення VS Code необхідно завантажити додаток, виконавши у командному рядку команду:

```
git clone https://github.com/Aroxed/js-todo-mvc
```

Далі у VS Code необхідно вибрати та відкрити каталог `js-todo-mvc`.

Для підтримки відлагодження коду та уникнення порушення політики спільного використання ресурсів з різних джерел CORS (https://uk.wikipedia.org/wiki/Cross-Origin_Resource_Sharing) запуск додатку слід виконувати не як файлу операційної системи (наприклад, як на рис. 1), а за допомогою Web-сервера.

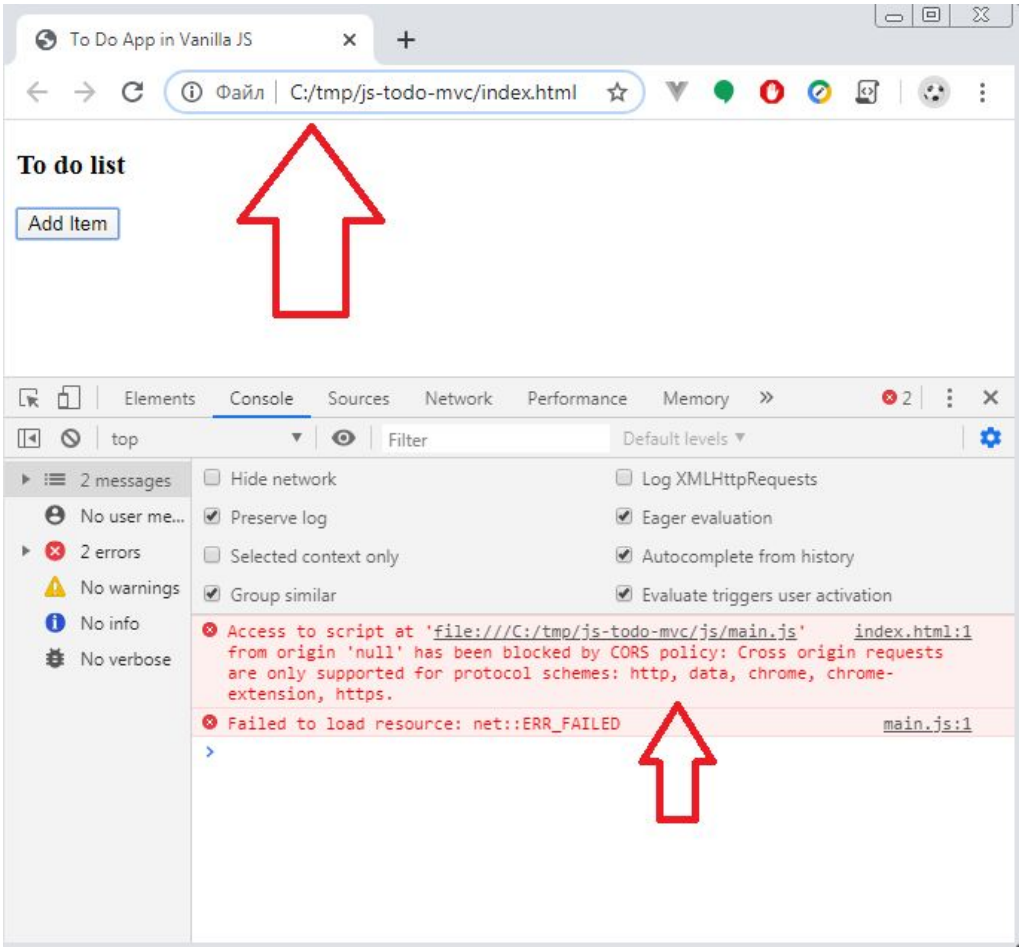

Рис.1. Некоректний запуск додатку

Для цього у VS Code необхідно встановити розширення Live Server (див. рис. 2).

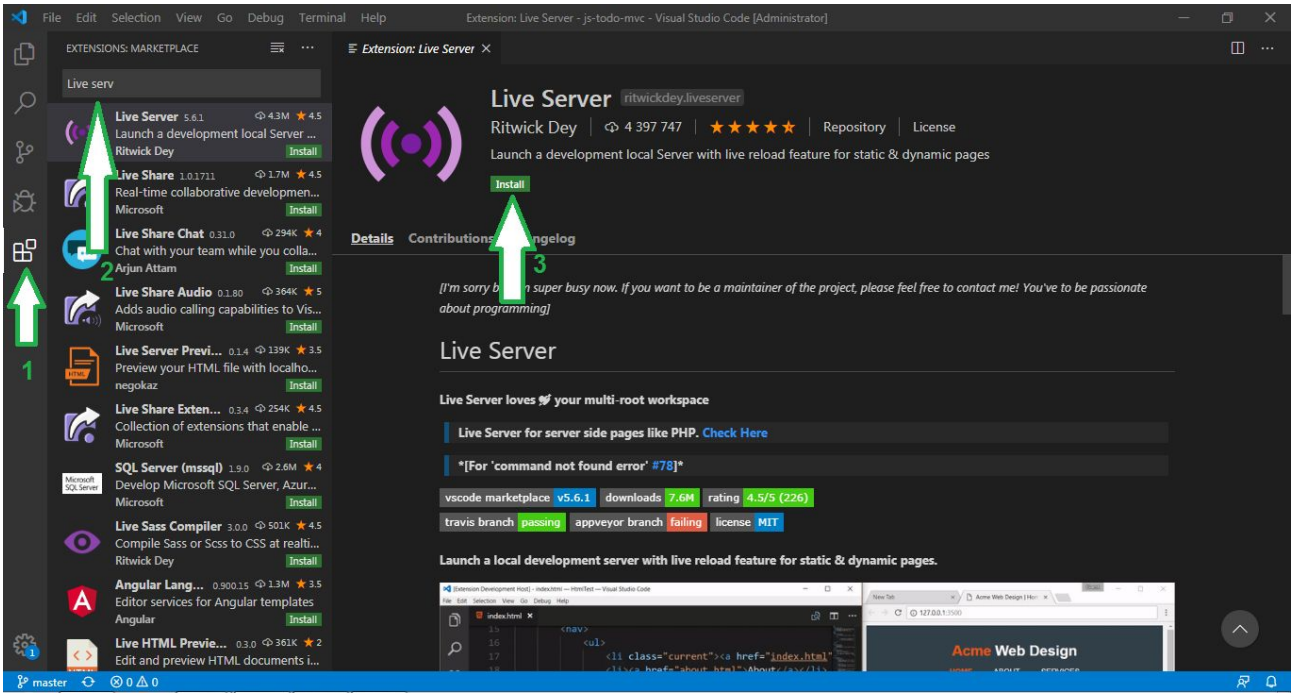

Рис. 2. Встановлення тестового Web-сервера Live Server.

Після встановлення тестового Web-сервера Live Server його необхідно запустити (див. рис.3):

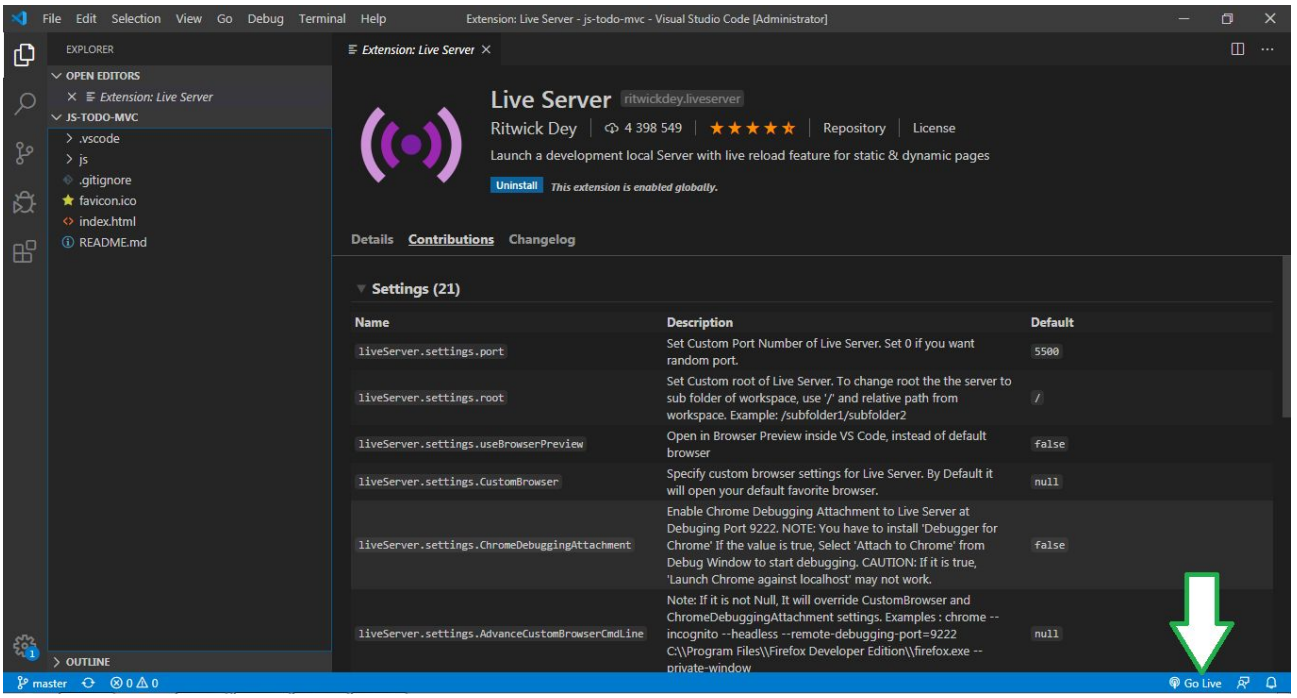

Рис. 3. Запуск Web-сервера Live Server.

Після запуску Веб-сервера автоматично відкриється сторінка Web-браузера як на рис.4.

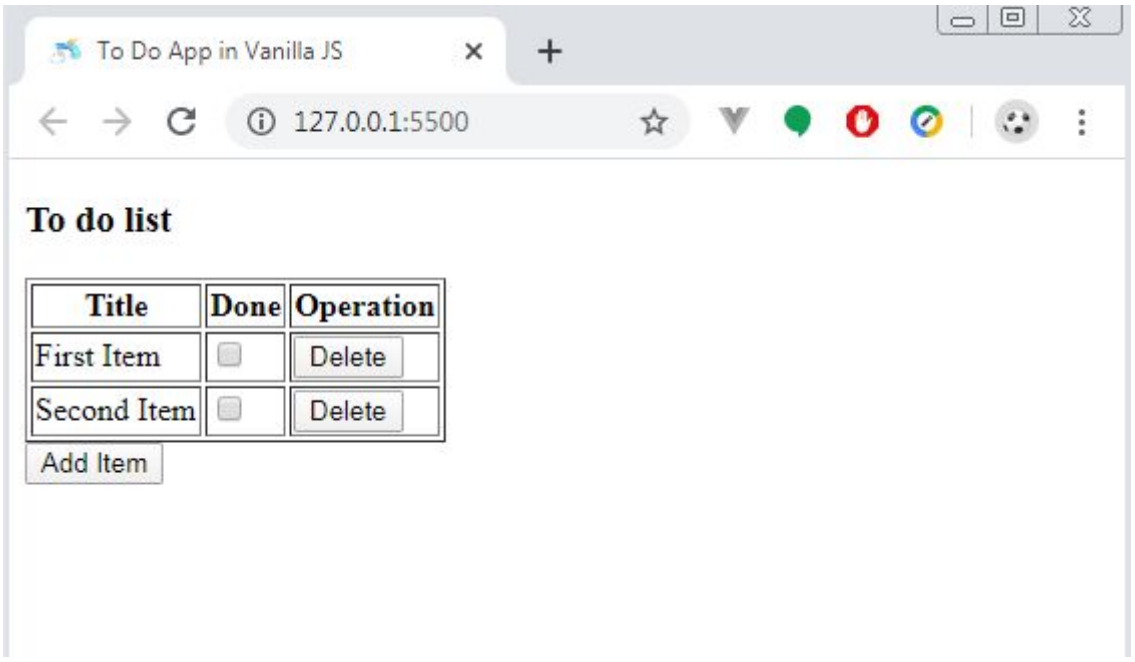

Рис. 4. Запуск Web-браузера зі сторінкою додатка To Do.

Після запуску будуть доступні елементи форми для додавання, перегляду, відмічання події як виконаної, а також вилучення елемента списку.

Особливості реалізації додатку

Виконання програми починається з файлу index.html, де описано базову структуру сторінки та вказано, що необхідно запустити файл main.js як стартовий модуль (рядок 15):

```
<script type="module" src="js/main.js"></script>
```

Варто звернути увагу на атрибут type=”module”, який вказує що main.js є модулем, який може імпортувати інші модулі.

Програмний код додатку розділено на три каталоги (model, view, controller) та один головний файл main.js. Останній призначений для ініціалізації класів моделей, подання та контролера, а також створення двох елементів списка справ (див. https://github.com/Aroxed/js-todo-mvc/blob/master/js/main.js).

За традиційним визначенням *класи моделей* описують поведінку створення та зміну об’єкта. *Класи подання* - спосіб відображення моделей, а *контролер* - управління виведенням та обробку подій від компонентів форми (кнопок, перемикачів тощо).

Каталог model складається з двох файлів:

- *Item.js* описує модель елемента списку. Цікавим методом цього класу є initOnModelChange, який “перехоплює” операцію зміни об’єкта класу Item, завдяки використанню вбудованого об’єкту Proxy (https://developer.mozilla.org/ru/docs/Web/JavaScript/Reference/Global_Objects/Proxy/handler/set).

```
initOnModelChange() {

    let handler = {

        set: (obj, prop, val) => {

            obj[prop] = val;

            if (this.onChangeCallback) this.onChangeCallback(this);

            return true;

        }

    }

    return new Proxy(this, handler);

}
```

Семантикою цієї функції є передача на обробку функції onChangeCallback() під час зміни властивостей даного об'єкта. Ідея полягає у тому, щоб делегувати обробку оновлення моделі його контейнеру - списку елементів, який, в свою чергу, делегуватиме обробку контролеру.

- *ItemListModel.js* описує поведінку контейнера елементів списку - власне список елементів. Його задачею є створення елемента, його вилучення, зміна статусу виконання та виклик редагування (див. https://github.com/Aroxed/js-todo-mvc/blob/master/js/model/ItemListModel.js)

Каталог view складається з двох файлів:

- ItemView.js відповідає за відображення елемента список (метод toHtml() файлу https://github.com/Aroxed/js-todo-mvc/blob/master/js/view/ItemView.js).

- ItemListView.js призначений для організації “транзитного” виклику оброблювачів подій з елементів списку до контролера, а також для виведення вікна створення елемента та формування html-вмісту всього списку справ.

Каталог controller містить єдиний файл Controller.js, який призначено для управління виведенням та обробкою подій від візуальних компонентів. Крім того, контролер призначено для перехоплення змін у контейнері елементів за допомогою стандартного об’єкта Proxy (функція initOnModelChange()).

Варіанти завдання

Роботу виконувати самостійно. Номер варіанта завдання визначити, взявши останні дві цифри номера залікової книжки студента. Для номерів більше 12, взяти залишок від ділення номера на 12.

1. **Скорочення URL-посилань**

Додаток має надавати можливість ведення списку URL користувача (додавання, перегляду, вилучення та редагування) із можливістю генерації скороченого вигляду URL у певному інтернет-домені. Наприклад, посилання https://docs.google.com/document/d/11a7OcjCWY8RBjoJiY4yN5NwZk-Uq9_FIcFB2iX9ZJ_A/edit?usp=sharing може бути замінено як https://cutt.ly/AtpHHPt . Забезпечити можливість генерації скорочених посилань у пакетному режимі за допомогою WebWorker.

2. **Калькулятор**

Забезпечити реалізацію базових арифметичних операцій +, -, *, /, x^y, а також перетворення числа з десяткової системи у двійкову та навпаки. Кнопки з цифрами та знаками операцій мають бути доступні у інтерфейсі вікна. Виконання операцій перетворення здійснювати у фоновому потоці за

допомогою WebWorker. Забезпечити обробку виключних ситуацій (наприклад, ділення на нуль). Вести історію виконаних операцій з можливістю їх перегляду.

3. **Будильник**

Основні функції: увімкнення-вимкнення, введення дати та часу. Врахувати можливість включення довільної кількості будильників, а також перегляд «заведених будильників». При досягненні часу спрацьовування, заданого для будильника, видавати повідомлення. Операції з таймером здійснювати у фоновому потоці за допомогою WebWorker.

4. **Облік робочого часу**

Реалізувати можливість обліку часу (запуск таймера, призупинення/продовження, зупинка, збереження назви, часу початку і завершення сеансу роботи). Передбачити можливість генерації звітів у табличному вигляді щодо роботи за день, тиждень та місяць у розрізі за днями, а також сумарного часу за період. Операції генерування звітів здійснювати у фоновому потоці за допомогою WebWorker.

5. **Телефонний довідник**

Реалізувати базу даних телефонного довідника (Прізвище абонента, список телефонних номерів абонента) з можливістю додавання, вилучення та редагування. Передбачити можливість генерації виведення списку абонентів та їх номерів, згрупованих та відсортованих за першою літерою. Операції генерування виведення здійснювати у фоновому потоці за допомогою WebWorker.

6. **Текстовий редактор**

Реалізувати редактор, який дозволяє виконувати форматування тексту, який уводиться. Обов‘язкові елементи форматування: напівжирність, нахил, підкреслювання, колір тексту, регістр символів. Елементи форматування мають бути доступними через відповідні кнопки, форматований текст – у окремому

вікні на сторінці. Виконувати розрахунок кількості слів у тексті за допомогою фонового потоку за допомогою WebWorker.

7. Опитування

Реалізувати можливість створення опитування у вигляді запитання та варіантів відповідей на нього. Передбачити введення відповідей на запитання. У фоновому потоці WebWorker згенерувати звіт що поточного стану опитування (кількості відповідей з кожного питання). Виводити звіт у окремому місці.

8. Блог з коментарями

Реалізувати можливість уведення (перегляду списку, додавання, вилучення) повідомлень блогу (назва, текстовий вміст), а також коментарів до кожного повідомлення. У фоновому потоці WebWorker реалізувати підрахунок кількості коментарів до кожного запису блогу з можливістю виведення статистики у окремому місці вікна.

9. Вивчення іноземних слів

Надати можливість створення (редагування, вилучення, перегляду) словника слів з перекладом, наприклад українсько-англійського. У окремому вікні виводити слово та варіанти перекладу. Користувач має обрати варіант. Додаток видає коректний варіант перекладу. У фоновому потоці WebWorker розраховувати кількість коректних відповідей користувача.

10. Перевірка орфографії

Надати можливість ведення словника слів (додавання, вилучення, редагування, перегляду) із можливістю його застосування для перевірки орфографії. Перевірка орфографії має відбуватись у фоновому потоці WebWorker. У випадку, коли слово не знайдено у словнику, виконати підкреслення хвилястою лінією цього слова як у звичайному текстовому редакторі.

11. Валідація форми

Надати можливість створення правил валідації за допомогою регулярних виразів (редагування та вилучення). Передбачити можливість створення текстових елементів, вказавши їх кількість. Кожному полю автоматично (рандомізовано) призначити правило валідації. При введенні даних у поле користувачем у фоновому потоці WebWorker виконувати валідацію. У випадку уведення некоректних даних видавати про це повідомлення поруч із полем.

12. Управління користувачами

Реалізувати можливість створення облікового запису користувача з даними (логін, прізвище, email). У фоновому потоці WebWorker виконувати перевірку унікальності логіну, у випадку знаходження дублікату вивести повідомлення і заборонити додавання. Надати можливість перегляду списку облікових записів, їх редагування та вилучення.

Вимоги до оформлення лабораторної роботи у електронному вигляді

Звіт про лабораторну роботу в репозиторії GitHub включає: назву лабораторної роботи, варіант студента, 2-3 копії екранних форм (screenshots).

Контрольні запитання

1. Пояснити сутність шаблону MVC.
2. Перерахувати особливості стандарту мови Javascript ECMAScript 2015 (ES6).
3. Пояснити призначення WebWorker у браузері.