Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра  автоматики та управління в технічних системах

**Лабораторна робота № 14**

по дисципліні «Основи клієнтської розробки»

Тема: JavaScript. Події. Обробники подій.

|  |  |
| --- | --- |
| Виконав:  студент групи ІА-94  Чумак Володимир Володимирович  Дата здачі \_\_\_\_  Захищено з балом \_\_\_ |  |

Київ 2020

**1. Тема**

JavaScript. Події. Обробники подій.

**2. Завдання**

Використовувати тему, обрану в 4 роботі.

Використати одну з подій миші. Написати фукцію-обробник. Призначити

функцію-обробник події через атрибут і через властивість.

Використати метод **addEventListener**, приначити одній події різні

обробники (написати функції-обробники).

Призначити обробником події об’єкт за допомогою **addEventListener**,

застосувати метод **handleEvent**, вивести елемент, на якому спрацював обробник, використовуючи **event.currentTarget**

Видалити об’єкт, використовуючи **removeEventListener**

**3. Теоретичні відомості**

Подія - це сигнал від браузера про те, що щось сталося. Все DOM-вузли

подають такі сигнали (хоча події бувають і не тільки в DOM).

Ось список найбільш часто використовуваних DOM-подій, поки просто

для ознайомлення:

Події миші:

click - відбувається, коли клікнули на елемент лівою кнопкою миші (на

пристроях з сенсорними екранами воно відбувається при торканні).

contextmenu - відбувається, коли клікнули на елемент правою кнопкою

миші.

mouseover / mouseout - коли миша наводиться на / покидає елемент.

mousedown / mouseup - коли натиснули / віджали кнопку миші на

елементі.

mousemove - при русі миші.

Події на елементах управління:

submit - користувач відправив форму <form>.

focus - користувач фокусується на елементі, наприклад натискає на

<input>.

Клавіатурні події:

keydown і keyup - коли користувач натискає / відпускає клавішу.

Обробники подій

Події можна призначити обробник, тобто функцію, яка спрацює, як тільки

подія відбулася.

Саме завдяки обробникам JavaScript-код може реагувати на дії

користувача.

Є кілька способів призначити події обробник. Зараз ми їх розглянемо,

починаючи з найпростішого.

Використання атрибута HTML

Обробник може бути призначений прямо в розмітці, в атрибуті, який

називається on <подія>.

Наприклад, щоб призначити обробник події click на елементі input, можна

використовувати атрибут onclick, ось так:

При натисканні мишкою на кнопці виконається код, вказаний в атрибуті

onclick.

Зверніть увагу, для вмісту атрибута onclick використовуються одинарні

лапки, так як сам атрибут знаходиться в подвійних. Якщо ми забудемо про це і

поставимо подвійні лапки всередині атрибута, ось так: onclick = "alert (" Click!

")", Код не буде працювати.

Атрибут HTML-тега - не найзручніше місце для написання великої

кількості коду, тому краще створити окрему JavaScript-функцію і викликати її

там.

Як ми пам'ятаємо, атрибут HTML-тега не чутливий до регістру, тому

ONCLICK буде працювати так само, як onClick і onCLICK ... Але, як правило,

атрибути пишуть в нижньому регістрі: onclick.

Використання властивості DOM-об'єкта

Можна призначати обробник, використовуючи властивість DOM-елемента

on <подія>.

Якщо обробник заданий через атрибут, то браузер читає HTML-розмітку,

створює нову функцію з вмісту атрибута і записує в властивість.

Обробник завжди зберігається у властивості DOM-об'єкта, а атрибут -

лише один із способів його ініціалізації.

Так як у елемента DOM може бути тільки одна властивість з ім'ям onclick,

то призначити більше одного обробника так не можна.

Прибрати обробник можна призначенням elem.onclick = null.

Доступ до елементу через this

Усередині обробника події this посилається на поточний елемент, тобто на

той, на якому, як кажуть, «висить» (тобто призначений) обробник.

У коді нижче button виводить свій вміст, використовуючи this.innerHTML:

<button onclick="alert(this.innerHTML)">Нажми меня</button>

**4. Порядок виконання лабораторної роботи**

1) ознайомитися з теоретичними відомостями;

2) виконати завдання

3) оформити звіт, який включатиме: титульний аркуш, завдання,

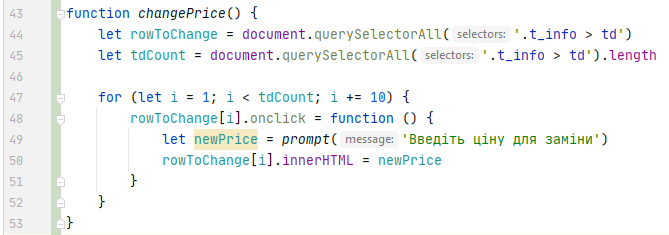
теоретичні відомості, результати і висновки по роботі;

4) продемонструвати результат на комп'ютері і захистити лабораторну

роботу.

**Результати виконання роботи**

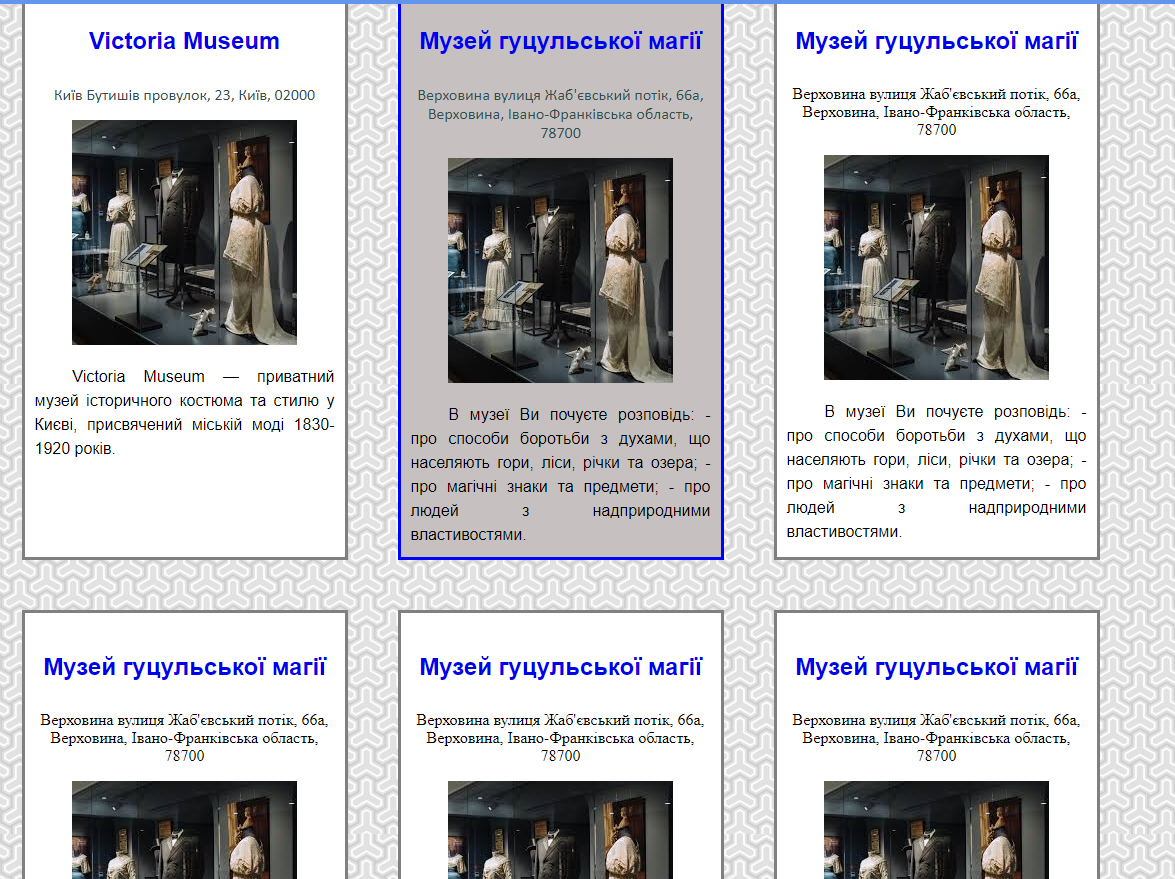
1. Призначення функції-обробника події через **властивість**:



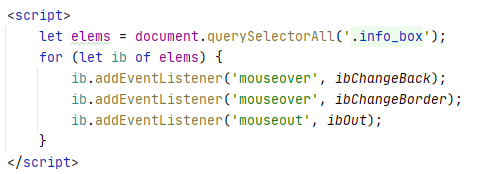
1. Призначення функції-обробника події через **атрибут**:



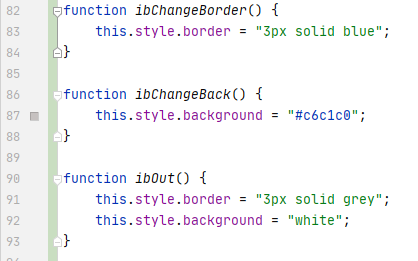
Використання функції **addEventListener** для призначення одній події різних обробників:



Код з файлу index.html:



Код з файлу scripts.js:

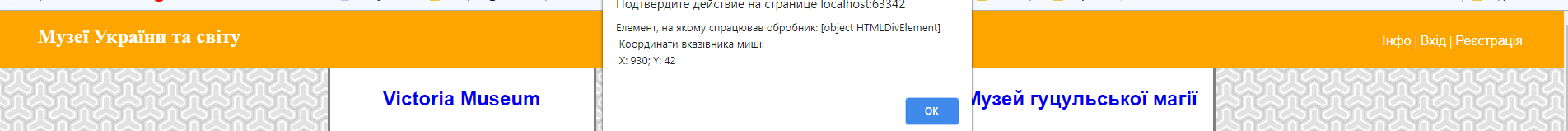


1. Призначення обробником **об’єкт** за допомогою addEventListener:

При натисканні ЛКМ:

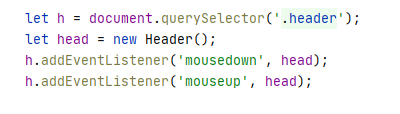


Після відтискання ЛКМ:





Код з файлу index.html:



Код з файлу scripts.js:



1. Видалення об’єкту, використовуючи removeEventListener:



Код з файлу index.html:



**Висновки**

У цій лабораторній роботі я навчився використовувати обробники подій. Починаючи від призначення обробника подій за допомогою властивості та атрибуту, і до використання функції addEventListener() та інших функцій для обробки подій, та роботи з ними.