ЧЕРНІВЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ЮРІЯ ФЕДЬКОВИЧА

Навчально-науковий інститут фізико-технічних та комп'ютерних наук

Кафедра комп'ютерних наук

ЗВІТ

з лабораторної роботи № 1

на тему: «**Застосування мови JavaScript для керування динамічною поведінкою Web -сайтів.**»

Виконав(ла):
студент(ка) 2 курсу, групи №244а
Ільницька Марія Романівна
(ПІБ)

Прийняв:
асистент, канд. комп. наук
Олар О. В.

Чернівці — 2025

Хід роботи

Я проаналізувала перший скрипт і можу зробити такі висновки:

- За допомогою *getElementById("blinking")* знаходиться абзац з *id="blinking"*.
- Використовується *setInterval*, який кожні 2 секунди виконує функцію.
- Функція перевіряє поточний розмір шрифту абзацу:
  - Якщо *13px* — змінює на *18px* та колір на *#D25424*.
  - Інакше встановлює *13px* та колір *#9E1AD2*.
- В результаті абзац постійно змінює розмір і колір, створюючи ефект миготіння.

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="utf-8">
   <title>Практична робота_01-JS01</title>
   <style>
      h1 {
         font: Monaco, Courier, monospace 28px;
         color: #618ad2;
      }
   </style>
</head>
<body>
   <h1>Підключення внутрішніх скриптів JavaScript</h1>
   <p id="blinking">Підключення скриптів можна виконувати різними способами. <br />
Один із способів полягає в описі
      скриптів прямо всередині HTML-сторінки.</p>
   <script>
      // ---Застосування методу DOM getElementById для пошуку елементу з id
="blinking"---
      var p = document.getElementById("blinking");
      // --- метод setInterval - пповторює певні дії через вказаний проміжок ---
часу:setInterval (function(){}, час);
      setInterval(function () {
         if (p.style.fontSize == "13px") {
            // ---Задаємо всім абзацам розмір шрифту та колір---
            p.style.fontSize = "18px";
            p.style.color = '#D25424';
         } else {
            p.style.fontSize = "13px";
            p.style.color = '#9E1AD2';
         }
      }, 2000);
   </script>
</body>
</html>
```

Я видалила минулий скрипт і вставила наступний, і побачила такі зміни:

- За допомогою *querySelector('h1')* знаходиться заголовок *<h1>*.
- Використовується *setInterval*, який кожну секунду виконує функцію.
- Функція генерує випадковий колір у форматі RGB і встановлює його для тексту заголовку.
- В результаті заголовок постійно змінює свій колір, створюючи ефект динамічного мерехтіння.

Я створила JavaScript, який реалізує динамічне привітання та повернення сторінки до початкового стану:

- **Перша кнопка**
  - Знаходиться на сторінці за допомогою *querySelector('button').*
  - При натисканні виконує функцію *myFunction*.
  - Функція ховає кнопку, створює заголовок *<h1>* з текстом "Я ВІТАЮ ТЕБЕ СВІТ!!!" і застосовує до нього стилі (червоний колір, великий розмір шрифту, вирівнювання по центру).
- **Друга кнопка**
  - Створюється динамічно всередині *myFunction* після появи заголовку.
  - Текст кнопки: "Твоя кнопка".
  - Прив’язаний обробник події *onclick* до функції *removeHeading.*
- **Функція *removeHeading***
  - Видаляє заголовок *<h1>* та другу кнопку зі сторінки.
  - Повертає першу кнопку в початковий вигляд, щоб її можна було натиснути знову.

Результат:

- Натискання першої кнопки створює великий червоний заголовок і другу кнопку.
- Натискання другої кнопки видаляє заголовок та саму кнопку, повертаючи сторінку до початкового стану.

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name='viewport' content='width=device-width, initial-scale=1.0'>
```

```
    <meta name="apple-mobile-web-app-capable" content="yes" />
    <title>Практична робота_01-JS02</title>
</head>
<body>
    Сторінка привітання <br>
    "Hello WORLD"<br>
    <button value="Кнопка вітання">Привітати СВІТ</button>
    <script>
        document.querySelector('button').onclick = myFunction;
        function myFunction() {
            "use strict";
            var myHeading = document.querySelector('button');
            myHeading.style.display = "none";
            var h1 = document.createElement('h1');
            h1.className = "big";
            h1.id = "big";
            document.body.appendChild(h1);
            h1.appendChild(document.createTextNode("Я ВІТАЮ ТЕБЕ СВІТ!!!"));
            document.getElementsByTagName("h1")[0].style.color = "red";
            document.getElementById("big").style.fontSize = "4em";
            document.getElementById("big").style.textAlign = "center";

            var button = document.createElement('button');
            button.className = "removeH1";
            button.id = "removeH1";
            document.body.appendChild(button);
            button.appendChild(document.createTextNode("Твоя кнопка"));
            button.onclick = removeHeading;
        }
        function removeHeading() {
            var findElem = document.getElementById("big");
            findElem.remove();
            var removeBtn = document.getElementById("removeH1");
            removeBtn.remove();
            document.querySelector('button').style.display = "inline-block";
        }
    </script>
</body>
</html>
```

Відповіді до контрольних запитань

1. Що таке *window.document*?

   window.document — це об’єкт, що представляє HTML-документ і дозволяє працювати з його елементами через DOM.

2. Яке призначення методу *querySelector*?

   querySelector — це метод для пошуку першого елемента на сторінці за CSS-селектором.

3. Пояснити призначення та відмінності методів *textContent* та *innerHTML*.

textContent — вставляє або отримує лише текст всередині елемента; *innerHTML* — вставляє або отримує HTML-код елемента разом з тегами.

4. Яке призначення методу *getElementById*?

   getElementById — знаходить елемент за його унікальним атрибутом id.

5. Поясніть синтаксис запису *getElementsByTagName("h1")[0]* ?

   getElementsByTagName("h1")[0] — повертає перший елемент *<h1>* на сторінці зі списку всіх таких тегів.

6. Які інші події «кліка», які викликаються браузером, коли ми клацаємо по об’єкту HTML-сторінки мишею Ви знаєте?

   ondblclick (подвійний клік), *onmousedown* (натискання кнопки миші), *onmouseup* (відпускання кнопки миші).

7. Яке призначення методу *.remove()* та *element.removeChild(child)*?

   .remove() — видаляє елемент зі сторінки; *element.removeChild(child)* — видаляє конкретний дочірній елемент із батьківського елемента.