**JAVASCRIPT**

***closest();***

**closest();** Метод **Element.closest()** повертає найближчий батьківський елемент (або сам елемент), який відповідає заданому **CSS**-селектор або null, якщо таких елементів взагалі немає.

*Функція у функції*

У функції можна додати функції у необмеженій кількості.

Наприклад:

**function firstFunc(){**

**function secondFunc(){**

**console.log(“SecondFunction”);**

**}**

**console.log(“FirstFunction”);**

**secondFunc()**

**}**

**firstFunc()**

**CSS**

***z-index***

**CSS**-властивість **z-index** визначає положення позиціонованого елемента та його дочірніх елементів або флекс-елементів осі **z**. Перекриваючі елементи з великим значенням **z-index** накладатимуться поверх елементів з меншим **z-index**.

Для позиціонованого елемента (тобто якщо у нього задано властивість position зі значенням, що відрізняється від **static**), властивість **z-index** відповідає за:

1. Порядок накладання у контексті накладання.

2. Можливість створення локального контексту. наложения.

Наприклад:

**z-index: 5;**

***backdrop-filter;***

Властивість **backdrop-filter CSS** дозволяє застосовувати графічні ефекти, такі як розмиття або зміщення кольору, до області за елементом. Оскільки це стосується всього, що знаходиться за елементом, щоб побачити ефект, ви повинні зробити елемент або його фон принаймні частково прозорими.

**CSS** фільтри такі як **blur()**, **contrast()** або **grayscale()** — чудовий додаток до **CSS**, незважаючи на те, що **SVG** може дозволити зробити чудові речі. Більше дізнатися про них можна на безлічі ресурсів.

Наприклад:

**backdrop-filter: blur(5px);**

***Modal***

Модальні створюються за допомогою **HTML**, **CSS** і **JavaScript**.

Вони розташовані над усім іншим у документі та видаляють прокручування з **<body>**, щоб натомість прокручувався модальний вміст.

Натискання модального «задника» автоматично закриє модаль.

**Bootstrap** підтримує лише одне модальне вікно за раз. Вкладені режими не підтримуються, оскільки ми вважаємо, що вони погано сприймаються користувачами.

У модалах використовується **position: fixed**, що іноді може бути дещо специфічним щодо його рендерингу.

За можливості розміщуйте свій модальний **HTML** у позиції верхнього рівня, щоб уникнути потенційних перешкод з боку інших елементів.

Ймовірно, ви зіткнетеся з проблемами під час вкладення **.modal** в інший фіксований елемент.

Знову ж таки, через позицію:

виправлено, є деякі застереження щодо використання модалів на мобільних пристроях.

Щоб отримати докладніші відомості, перегляньте наші документи підтримки браузера.

Через те, як **HTML5** визначає свою семантику, атрибут **HTML** автофокусу не впливає на моди **Bootstrap**. Щоб досягти такого ж ефекту, використовуйте власний **JavaScript**:

$('#myModal').on('shown.bs.modal', function () {

$('#myInput').trigger('focus')

})

*Оператор інкременту*

Оператор інкременту (**++**) збільшує (додає до) свій операнд і повертає значення до або після інкременту, залежно від того, де розміщено оператор.

Наприклад:

let count = +count.innerHTML ++

***opacity;***

**CSS**-властивість **opacity** встановлює непрозорість елемента. Непрозорість - це ступінь, у якому вміст ховається за елементом, є протилежністю прозорості.

Наприклад:

**opacity: 0;** – Прозорій.

**opacity: 1;** — Не прозорій.

**setInterval & setTimeout**

Ми можемо викликати функцію не в даний момент, а пізніше через заданий інтервал часу. Це називається "планування виклику".

Для цього існують два методи:

**setTimeout** дозволяє викликати функцію один раз через певний проміжок часу.

**setInterval** дозволяє викликати функцію регулярно, повторюючи виклик через певний проміжок часу.

Ці методи є частиною специфікації **JavaScript**. Але більшість середовищ виконання **JS**-коду мають внутрішній планувальник та надають доступ до цих методів. Зокрема, вони підтримуються у всіх браузерах та **Node.js**.

Наприклад:

**setTimeout**

**function sayHi() {**

**alert('Привіт');**

**}**

**setTimeout(sayHi, 1000);**

**setInterval**

**let timerId = setInterval(() => alert('tick'), 2000);**

**cursor;**

Властивість **cursor** **CSS** встановлює курсор миші, якщо такий є, щоб показувати, коли вказівник миші знаходиться над елементом.

Налаштування курсору має інформувати користувачів про операції миші, які можна виконувати в поточному місці, зокрема: виділення тексту, активація довідки чи контекстного меню, копіювання вмісту, зміна розміру таблиці тощо. Ви можете вказати тип курсору за допомогою ключового слова або завантажити певну піктограму для використання.