***Конспект***

**Події клавіатури**

**Події keydown та keyup**

Подія **keydown** відбувається при натисканні кнопки, а **keyup** – при відпусканні.

**event.code** уточнює, яка саме з них була натиснута.

**event.key** повідомляє про “сенс” клавіші: що взагалі було натиснуто.

|  |  |  |
| --- | --- | --- |
| ***event.code*** | ***Клавіші*** | ***event.key*** |
| “KeyA” | a | ’a’ |
| “KeyA” | Shift+a | ’A’ |
| ‘Digit1’ | 1 | ‘1’ |
| ‘Enter’ | Enter | ‘Enter’ |

**Події фокісування. Зміни input. Буферу обміну.**

**Фокусування**

Фокусування(**focus**) зазвичай означає: “приготуйся до введення даних на цьому елементі”, це гарний момент, щоб ініціалізувати або завантажити що-небудь. Момент втрати фокусу (“**blur**”) може бути важливішим.

Подія focus викликається на момент фокусування, а **blur** – коли елемент втрачає фокус.

Вони не спливають. Але можна використовувати події **focusin**/**focusout**.

Більшість елементів не підтримують стандартне фокусування. Використовуйте **tabindex** атрибут **HTML**, щоб зробити фокусування на будь-який елемент.

Подія: **change**

Подія **change** спрацьовує після зміни елемента.

Поки ми друкуємо у текстовому полі, подія не відбувається. Але коли ми переміщуємо фокус в інше місце, наприклад, натискаючи кнопку, то відбудеться подія **change**.

Подія: **input**

Подія input спрацьовує щоразу при зміні значення.

Воно працює за будь-яких змін значень, навість якщо вони не пов’язані з клавіатурними діями: вставка за допомогою миші або розпізнавання мови при диктуванні тексту.

Подія: **cut**, **copy**, **paste**

Ці події відбуваються при вирізанні/копіюванні/вставці даних.

Ми також можемо використовувати **event.preventDefault()** для запобігання дії за промовчанням, і в результаті нічого не скопіюється/не вставиться.

Властивість **event.clipboardData** надає доступ на читання/запис у буфер обміну.

**Валідація форми**

**Що таке валідація форми?**

Валідація форми — це перевірка даних, суть якої переконатися, що обов’язкові поля форми заповнені даними у коректному форматі. Валідація форми робиться на боці клієнта і допомагає переконатися, що дані, введені у кожен елемент форми, відповідають вимогам.

Приклад валідації

«Обов’язкове поле» (Ви не можете залишити поле порожнім).

«Будь ласка, введіть корректну **email**-адресу»

«Довжина пароля повинна бути від 8 до 30 символів і включати одну велику літеру, один символ, і одну цифру.»

При відправленні форми спрацьовує подія **submit**, вона зазвичай використовується для перевірки (валідації) форми перед її відправкою на сервер або запобігання відправці та обробці її за допомогою **JavaScript**.

Метод **form.submit()** дозволяє ініціювати відправлення форми **JavaScript**. Ми можемо використовувати його для динамічного створення та надсилання наших власних форм на сервер.

Обидві дії згенерують подію **submit** на формі. Оброблювач може перевірити дані, і якщо є помилки, показати їх та викликати **event.preventDefault()**, тоді форма не буде відправлена на сервер.

Якщо коротко, то:  
**submit** - подія відправлення форми;  
**event.preventDefault()** - скасовуємо відправку форми;  
**form.submit()** - надсилає форму.

**Практика**

***HTML***

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

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

    <title>Document</title>

</head>

<body>

    <div class="content">

        <form class="reg" action="#">

            <div class="block-input">

                <p class="p-input-reg">Enter your user name</p>

                <input class="input" type="text" id="username" placeholder="username">

                <p class="p-input-reg">Enter your email</p>

                <input class="input" type="text" id="email" placeholder="email">

                <p class="p-input-reg">Enter your password(not 1234567890)</p>

                <input class="input" type="text" id="password" placeholder="password">

            </div>

            <br>

            <div class="block-button-reg">

                <input type="submit" class="button-reg" value="Registration">

             </div>

        </form>

    </div>

    <script src="script.js"></script>

</body>

</html>

***JAVASCRIPT***

const form = document.querySelector(".reg");

const bannedWords = [

    '%',

    '^',

    '&',

]

form.addEventListener("submit", (event)=>{

    event.preventDefault();

    let inputs = document.querySelectorAll('.input');

    let divErrors = document.querySelectorAll('.error')

    for (let error of divErrors){

        error.remove();

    }

    let result = true;

    for (let input of inputs){

        if (validation(input) == false){

            result = false;

        }

    }

    if (result == true ){

    form.submit()

}

})

function createError(text, elem){

    let divError = document.createElement('div');

    let pError = document.createElement('p');

    let imgError = document.createElement('img');

    divError.append(pError);

    divError.append(imgError);

    elem.after(divError);

    imgError.src = "error.svg";

    pError.innerHTML = text;

    divError.classList.add('error');

}

function validation(input) {

    if (input.value == '') {

        createError(`Поле пустое`, input);

        return false;

    }

    if (input.id == 'username') {

        for (let symbol of input.value) {

            if (bannedWords.includes(symbol)) {

                createError(`Введен недопустимый символ: ${symbol}`, input);

                return false;

            }

        }

    }

    else if (input.id == "email"){

        if (!input.value.includes("@")) {

            createError(`Нету @`, input);

            return false;

        }

    }

    return true

}

**Матрица**