**Лабораторна робота №9**

**Завдання 0:**

Ознайомитись з теоретичними відомостями за посиланням:

[Прототипи, наслідування](https://uk.javascript.info/prototypes)

[Класи](https://uk.javascript.info/classes)

**Завдання 1**

Створіть клас "Модальне вікно" (Modal), який створює модальне вікно на сторінці. Клас повинен мати методи для відкриття, закриття та встановлення вмісту модального вікна. Зробіть можливість переміщення вікна по екрану за допомогою мишки (щоб можна було перетягувати за заголовок вікна).

Лістининг коду

HTML

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Tusk 1</title>

    <link rel="stylesheet" href="style.css">

</head>

  <body>

  <button onclick="modalInstance.open()">Відкрити вікно</button>

  <div id="myModal" class="modal">

      <div class="modal-header" onmousedown="modalInstance.dragStart(event)">

          <h2>Модальне вікно</h2>

      </div>

      <div class="modal-content">

          <label for="login">Логін:</label>

          <input type="text" id="login" name="login"><br><br>

          <label for="password">Пароль:</label>

          <input type="password" id="password" name="password"><br><br>

          <button onclick="modalInstance.login()">Вхід</button>

          <button onclick="modalInstance.close()">Закрити</button>

      </div>

  </div>

  <script src="main.js"></script>

  </body>

</html>

CSS

.modal {

    display: none;

    position: fixed;

    top: 50%;

    left: 50%;

    width: 300px; /\* фіксована ширина \*/

    height: 240px; /\* фіксована висота \*/

    margin-left: -150px; /\* половина ширини \*/

    margin-top: -100px; /\* половина висоти \*/

    border: 1px solid #ccc;

    background-color: #fff;

    z-index: 1;

}

label {

    margin-left: 5%;

}

button {

    width: 30%;

}

.modal-header {

    cursor: move;

    padding: 10px;

    background-color: #f1f1f1;

}

JS

class Modal {

    constructor() {

        this.modal = document.getElementById('myModal');

        this.isDragging = false;

    }

    open() {

        this.modal.style.display = 'block';

    }

    close() {

        this.modal.style.display = 'none';

    }

    dragStart(e) {

        this.isDragging = true;

        this.offsetX = e.clientX - this.modal.offsetLeft;

        this.offsetY = e.clientY - this.modal.offsetTop;

        const dragMove = (e) => {

            if (this.isDragging) {

                this.modal.style.left = e.clientX - this.offsetX + 'px';

                this.modal.style.top = e.clientY - this.offsetY + 'px';

            }

        };

        const dragEnd = () => {

            this.isDragging = false;

            document.removeEventListener('mousemove', dragMove);

            document.removeEventListener('mouseup', dragEnd);

        };

        document.addEventListener('mousemove', dragMove);

        document.addEventListener('mouseup', dragEnd);

    }

    login() {

        const loginValue = document.getElementById('login').value;

        const passwordValue = document.getElementById('password').value;

        console.log('Логін:', loginValue);

        console.log('Пароль:', passwordValue);

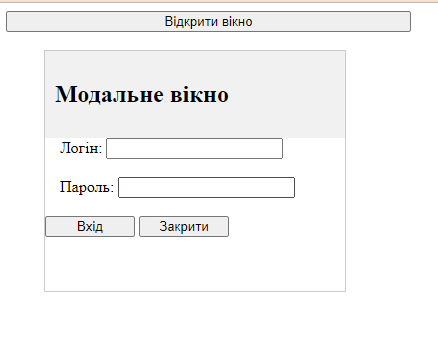
        this.close();

    }

}

const modalInstance = new Modal();

Результат



**Завдання 2**

Створіть клас "Слайдер" (Slider), який дозволяє створювати горизонтальний або вертикальний слайдер з можливістю перегляду зображень або вмісту. Клас повинен мати методи для додавання слайдів та переміщення між ними.

Лістининг коду

HTML

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Tusk 2</title>

    <link rel="stylesheet" href="style.css">

</head>

    <body>

        <h2>OOP-Slider</h2>

        <div id="mySlider" class="slider">

            <div class="slides"></div>

            <div class="slider-btn slider-btn-prev" onclick="mySlider.movePrev()">←</div>

            <div class="slider-btn slider-btn-next" onclick="mySlider.moveNext()">→</div>

        </div>

        <script src="main.js"></script>

    </body>

</html>

JS

class Slider {

    constructor(selector, orientation = 'horizontal') {

        this.slider = document.querySelector(selector);

        this.slidesContainer = this.slider.querySelector('.slides');

        this.slides = [];

        this.currentIndex = 0;

        this.orientation = orientation;

        this.initStyles();

    }

    initStyles() {

        this.slider.style.overflow = 'hidden';

        this.slidesContainer.style.display = 'flex';

        this.slidesContainer.style.transition = 'transform 0.5s ease-in-out';

        if (this.orientation === 'vertical') {

            this.slider.style.height = '200px'; // Set the height for vertical slider

            this.slidesContainer.style.flexDirection = 'column';

        } else {

            this.slider.style.width = '200px'; // Set the width for horizontal slider

            this.slidesContainer.style.flexDirection = 'row';

        }

    }

    addSlide(content) {

        const slide = document.createElement('div');

        slide.classList.add('slide');

        slide.innerHTML = content;

        this.slides.push(slide);

        this.slidesContainer.appendChild(slide);

        this.updateSlidePosition();

    }

    moveNext() {

        if (this.currentIndex < this.slides.length - 1) {

            this.currentIndex++;

            this.updateSlidePosition();

        }

    }

    movePrev() {

        if (this.currentIndex > 0) {

            this.currentIndex--;

            this.updateSlidePosition();

        }

    }

    updateSlidePosition() {

        const moveAmount = -this.currentIndex \* 200;

        this.slidesContainer.style.transform =

            this.orientation === 'vertical' ? `translateY(${moveAmount}px)` : `translateX(${moveAmount}px)`;

    }

}

// Example Usage

const mySlider = new Slider('#mySlider', 'horizontal');

mySlider.addSlide('<img src="galery/image1.jpg" alt="Slide 1">');

mySlider.addSlide('<img src="galery/image2.jpg" alt="Slide 2">');

mySlider.addSlide('<img src="galery/image4.jpg" alt="Slide 4">');

mySlider.addSlide('<img src="galery/image5.jpg" alt="Slide 5">');

mySlider.addSlide('<img src="galery/image6.jpg" alt="Slide 6">');

mySlider.addSlide('<img src="galery/image8.jpg" alt="Slide 8">');

CSS

.slider {

    overflow: hidden;

    position: relative;

    width: 200px;

    height: 200px;

    border: 1px solid #ccc;

}

.slides {

    display: flex;

    transition: transform 0.5s ease-in-out;

}

.slide {

    flex: 0 0 200px;

    width: 200px;

    height: 200px;

}

img {

    max-width: 100%;

    max-height: 100%;

    object-fit: cover;

}

.slider-btn {

    position: absolute;

    top: 50%;

    transform: translateY(-50%);

    cursor: pointer;

    background-color: #f1f1f1;

    border: 1px solid #ccc;

    padding: 5px;

}

.slider-btn-prev {

    left: 5px;

}

.slider-btn-next {

    right: 5px;

}

Результат



**Завдання 3**

Створіть клас "Таблиця" (Table), який генерує HTML-таблицю на основі заданих даних. Клас повинен мати методи для додавання рядків та стовпців до таблиці.

Лістининг коду

HTML

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Tusk 3</title>

    <link rel="stylesheet" href="style.css">

</head>

    <body>

        <div id="myTable"></div>

        <script src="main.js"></script>

    </body>

</html>

CSS

table {

    border-collapse: collapse;

    width: 100%;

    margin-top: 20px;

}

table, th, td {

    border: 1px solid #ccc;

    margin-bottom: 50px;

}

th, td {

    padding: 10px;

    text-align: left;

}

input {

    margin-right: 10px;

    margin-bottom: 10px;

}

button {

    margin-bottom: 20px;

}

JS

class Table {

    constructor(data, selector) {

        this.data = data;

        this.tableContainer = document.querySelector(selector);

        this.table = document.createElement('table');

        this.tableContainer.appendChild(this.table);

        this.renderTable();

        this.setupManualEntryForm();

    }

    renderTable() {

        this.data.forEach(rowData => this.addRow(rowData));

    }

    addRow(rowData) {

        const row = this.table.insertRow();

        rowData.forEach(cellData => {

            const cell = row.insertCell();

            cell.textContent = cellData;

        });

    }

    addColumn(columnData) {

        this.table.tHead.insertRow().insertCell().textContent = columnData[0];

        columnData.slice(1).forEach((cellData, rowIndex) => {

            const cell = this.table.rows[rowIndex + 1].insertCell();

            cell.textContent = cellData;

        });

    }

    setupManualEntryForm() {

        const inputRow = document.createElement('div');

        inputRow.classList.add('input-row');

        this.data[0].forEach(header => {

            const input = document.createElement('input');

            input.setAttribute('placeholder', header);

            inputRow.appendChild(input);

        });

        const addButton = document.createElement('button');

        addButton.textContent = 'Add';

        addButton.addEventListener('click', () => this.handleAddManualEntry());

        inputRow.appendChild(addButton);

        this.tableContainer.appendChild(inputRow);

    }

    handleAddManualEntry() {

        const inputRow = this.tableContainer.querySelector('.input-row');

        const inputElements = inputRow.querySelectorAll('input');

        const rowData = Array.from(inputElements).map(input => input.value);

        this.addRow(rowData);

        inputElements.forEach(input => (input.value = '')); // Clear input fields

    }

}

// Example Usage

const initialData = [

    ['Name', 'Age', 'Country'],

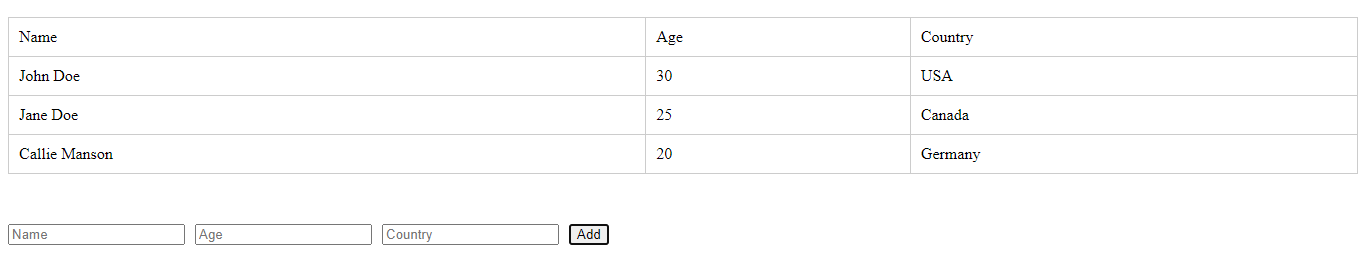
    ['John Doe', '30', 'USA'],

    ['Jane Doe', '25', 'Canada']

];

const myTable = new Table(initialData, '#myTable');

Результат

****

**Завдання 4**

Створіть клас "Форма" (Form), який дозволяє створювати HTML-форми з заданими полями вводу, кнопками та обробниками подій для них. Клас повинен мати методи для обробки відправки форми та валідації введених даних.

Лістининг коду

HTML

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Tusk 4</title>

    <link rel="stylesheet" href="style.css">

</head>

    <body>

        <div id="myForm"></div>

        <script src="main.js"></script>

    </body>

</html>

CSS

body {

    font-family: 'Arial', sans-serif;

    background-color: #f4f4f4;

    display: flex;

    align-items: center;

    justify-content: center;

    height: 100vh;

    margin: 0;

}

#myForm {

    background-color: #fff;

    padding: 20px;

    border-radius: 8px;

    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

    width: 300px;

}

input {

    width: 100%;

    padding: 10px;

    margin-bottom: 10px;

    box-sizing: border-box;

    border: 1px solid #ccc;

    border-radius: 4px;

    font-size: 14px;

}

button {

    background-color: #4caf50;

    color: #fff;

    padding: 10px;

    border: none;

    border-radius: 4px;

    cursor: pointer;

    font-size: 16px;

}

button:hover {

    background-color: #45a049;

}

JS

class Form {

    constructor(selector) {

        this.formContainer = document.querySelector(selector);

        this.form = document.createElement('form');

        this.fields = [];

        this.formContainer.appendChild(this.form);

    }

    addInput(type, name, placeholder) {

        const input = document.createElement('input');

        input.type = type;

        input.name = name;

        input.placeholder = placeholder;

        this.fields.push(input);

        this.form.appendChild(input);

    }

    addButton(label, callback) {

        const button = document.createElement('button');

        button.textContent = label;

        button.addEventListener('click', event => {

            event.preventDefault();

            this.validateAndSubmit(callback);

        });

        this.form.appendChild(button);

    }

    validateAndSubmit(callback) {

        const formData = {};

        for (const field of this.fields) {

            const value = field.value.trim();

            const name = field.name;

            if (!value) {

                alert(`Field "${name}" cannot be empty.`);

                return;

            }

            formData[name] = value;

        }

        if (callback)

            callback(formData);

    }

}

const myForm = new Form('#myForm');

myForm.addInput('text', 'name', 'Enter your name');

myForm.addInput('email', 'email', 'Enter your email');

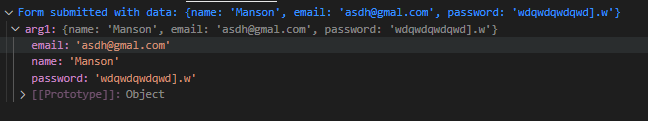
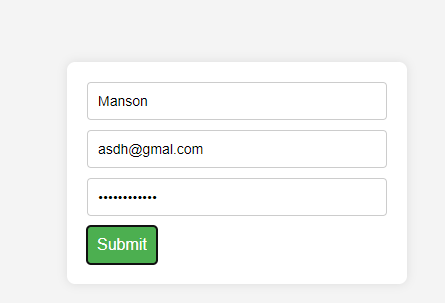
myForm.addInput('password', 'password', 'Enter your password');

myForm.addButton('Submit', data => {

    console.log('Form submitted with data:', data);

});

Результат



**Завдання 5**

Створіть клас "Вкладка" (Tab), який створює вкладки для переключення між різними блоками вмісту. Клас повинен мати методи для додавання вкладок та вмісту для них, а також для переключення між вкладками.

Лістининг коду

HTML

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Tusk 5</title>

    <link rel="stylesheet" href="style.css">

</head>

    <body>

        <div id="myTabs"></div>

        <script src="main.js"></script>

    </body>

</html>

CSS

.tabs {

    display: flex;

    margin-bottom: 10px;

}

.tab {

    padding: 10px;

    cursor: pointer;

    border: 1px solid #ccc;

    border-radius: 4px 4px 0 0;

    margin-right: -1px;

}

.tab-content {

    border: 1px solid #ccc;

    padding: 10px;

    border-radius: 0 0 4px 4px;

}

.hidden {

    display: none;

}

JS

class Tab {

    constructor(selector) {

        this.tabsContainer = document.querySelector(selector);

        this.tabs = [];

        this.contents = [];

    }

    addTab(tabName, content) {

        const tab = document.createElement('div');

        tab.classList.add('tab');

        tab.textContent = tabName;

        tab.addEventListener('click', () => this.switchTab(tab));

        this.tabs.push(tab);

        this.tabsContainer.appendChild(tab);

        const contentContainer = document.createElement('div');

        contentContainer.classList.add('tab-content', 'hidden');

        contentContainer.innerHTML = content;

        this.contents.push(contentContainer);

        this.tabsContainer.appendChild(contentContainer);

        if (this.tabs.length === 1) {

            this.switchTab(tab);

        }

    }

    switchTab(selectedTab) {

        this.tabs.forEach((tab, index) => {

            const content = this.contents[index];

            if (tab === selectedTab) {

                tab.classList.add('active');

                content.classList.remove('hidden');

            } else {

                tab.classList.remove('active');

                content.classList.add('hidden');

            }

        });

    }

}

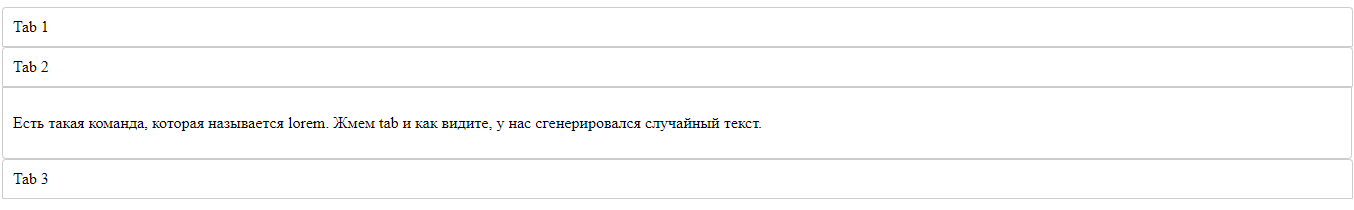
const myTabs = new Tab('#myTabs');

myTabs.addTab('Tab 1', '<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Totam perferendis, voluptas non debitis ipsa ut unde? Illum nobis, harum, tempora perspiciatis aspernatur iure consequatur, quis at vero fuga architecto praesentium.</p>');

myTabs.addTab('Tab 2', '<p>Есть такая команда, которая называется lorem. Жмем tab и как видите, у нас сгенерировался случайный текст.</p>');

myTabs.addTab('Tab 3', '<p>Если нам нужно разместить его внутри элемента p, пишем p>lorem\*3</p>');

Результат



**Завдання 6**

Створіть клас "Сповіщення" (Notification), який дозволяє створювати спливаючі повідомлення на сторінці. Клас повинен мати методи для створення сповіщень з різними стилями та для їх закриття.

Лістининг коду

HTML

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Tusk 6</title>

    <link rel="stylesheet" href="style.css">

</head>

    <body>

        <button onclick="showSuccess()">Показати успішне повідомлення</button>

        <button onclick="showError()">Показати помилкове повідомлення</button>

        <button onclick="showInfo()">Показати інформаційне повідомлення</button>

        <div id="myNotification" class="notification"></div>

        <script src="main.js"></script>

    </body>

</html>

JS

class Notification {

    constructor() {

        this.container = document.getElementById('myNotification');

    }

    show(message, type) {

        this.container.textContent = message;

        this.container.className = `notification ${type}`;

        this.container.style.display = 'block';

        const closeBtn = document.createElement('span');

        closeBtn.innerHTML = '&times;';

        closeBtn.className = 'close-btn';

        closeBtn.addEventListener('click', () => this.hide());

        this.container.appendChild(closeBtn);

    }

    hide() {

        this.container.style.display = 'none';

    }

}

const notification = new Notification();

function showSuccess() {

    notification.show('Успішне повідомлення!', 'success');

}

function showError() {

    notification.show('Помилкове повідомлення!', 'error');

}

function showInfo() {

    notification.show('Інформаційне повідомлення.', 'info');

}

CSS

body {

    font-family: 'Helvetica Neue', sans-serif;

    background-color: #f4f4f4;

    display: flex;

    align-items: center;

    justify-content: center;

    height: 100vh;

    margin: 0;

}

.notification {

    position: fixed;

    top: 10px;

    right: 10px;

    padding: 15px;

    border-radius: 12px;

    color: #fff;

    font-size: 16px;

    display: none;

}

.success {

    background-color: #4CAF50;

}

.error {

    background-color: #f44336;

}

.info {

    background-color: #2196F3;

}

.close-btn {

    cursor: pointer;

    margin-left: 10px;

    font-size: 20px;

    font-weight: bold;

}

button {

    background-color: #000;

    color: #fff;

    padding: 15px;

    border: none;

    border-radius: 12px;

    cursor: pointer;

    font-size: 16px;

    margin-right: 10px;

}

button:hover {

    background-color: #333;

}

Результат

