**Лабораторна робота № 8**

**Функції setInterval та setTimeout***.*

**Хід роботи:**

**Завдання 1 + 2:**

1. Реалізуйте відображення годинника, що відображає поточний час у правому верхньому кутку екрана.
2. Реалізуйте таймери зворотного відліку. Розмістіть на сторінці 3 таймери, які мають певне початкове значення часу (години, хвилини та секунди). Кожний таймер повинен містити три кнопки: «Старт», «Стоп», «Скидання». При натисканні на кнопку «Старт», таймер починає зворотній відлік від початкового значення часу до «00:00:00». Досягаючи нульового значення часу таймер повинен зупинитися. При натисканні на кнопку «Стоп» таймер повинен зупинитися. Натискання на кнопку «Скидання» призводить до ініціалізації таймера початковим значенням часу.

**Лістинг програми:**

HTML file:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Lab 8</title>

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

</head>

<body>

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

    <div id="clock">00:00:00</div>

    <div class="timer">

        <div class="tablo">00:00:00</div>

        <div class="button">

            <button type="button" class="startButton">Start</button>

            <button disabled="disabled" type="button" class="stopButton">Stop</button>

            <button type="button" class="resetButton">Reset</button>

        </div>

    </div>

    <div class="timer">

        <div class="tablo">00:00:00</div>

        <div class="button">

            <button type="button" class="startButton">Start</button>

            <button disabled="disabled" type="button" class="stopButton">Stop</button>

            <button type="button" class="resetButton">Reset</button>

        </div>

    </div>

    <div class="timer">

        <div class="tablo">00:00:00</div>

        <div class="button">

            <button type="button" class="startButton">Start</button>

            <button disabled="disabled" type="button" class="stopButton">Stop</button>

            <button type="button" class="resetButton">Reset</button>

        </div>

    </div>

</body>

</html>

Java Script file:

document.addEventListener('DOMContentLoaded', function () {

    initializeTimers(); // Викликаю функцію ініціалізації під час завантаження сторінки

});

function initializeTimers() {

    for (let i = 0; i < timerCount; i++) {

        timerValues[i] = { h: 0, m: 55, s: 0, timerId: null }; // Встановив початкове значення для кожного таймера

        timerUpdate(i); // Оновлює відображення

    }

}

function clockTick() {

    let date = new Date();

    let hours = date.getHours();

    let minutes = date.getMinutes();

    if (minutes < 10) {

        minutes = '0' + minutes;

    };

    let seconds = date.getSeconds();

    if (seconds < 10) {

        seconds = '0' + seconds;

    };

    let timeString = `${hours}:${minutes}:${seconds}`;

    clock.innerText = timeString;

}

clockTick()

setInterval(clockTick, 1000);

let timerValues = [];

let timerElements = document.querySelectorAll('.timer');

let timerElementsArray = Array.from(timerElements);

let timerCount = timerElements.length;

for (i = 0; i < timerCount; i++) {

    timerValues.push({

        h: 0, m: 0, s: 0, timerId: null

    });

}

console.log(timerValues);

let timerId;

function timerTick(index) {

    if (timerValues[index].s > 0) {

        timerValues[index].s--;

    } else if (timerValues[index].m > 0) {

        timerValues[index].s = 59;

        timerValues[index].m--;

    } else if (timerValues[index].h > 0) {

        timerValues[index].s = 59;

        timerValues[index].m = 59;

        timerValues[index].h--;

    }

    timerUpdate(index);

    if (timerValues[index].h === 0 && timerValues[index].m === 0 && timerValues[index].s === 0) {

        clearInterval(timerValues[index].timerId);

    }

}

function addZero(value) {

    if (value < 10) {

        return '0' + value;

    }

    return value;

}

function timerUpdate(index) {

    let timerNewValue = {

        h: addZero(timerValues[index].h),

        m: addZero(timerValues[index].m),

        s: addZero(timerValues[index].s),

    }

    document.querySelectorAll(`.timer`)[index].querySelector('.tablo').innerHTML = `${timerNewValue.h}:${timerNewValue.m}:${timerNewValue.s}`;

}

document.documentElement.addEventListener('click', function (event) {

    let parentTimerDiv = event.target.closest('.timer');

    if (parentTimerDiv !== null) {

        let tablo = parentTimerDiv.querySelector('.tablo');

        let startButton = parentTimerDiv.querySelector('.startButton');

        let stopButton = parentTimerDiv.querySelector('.stopButton');

        let resetButton = parentTimerDiv.querySelector('.resetButton');

        let currentTimerIndex = timerElementsArray.indexOf(parentTimerDiv);

        if (event.target.classList.contains('startButton')) {

            // parentTimerDiv.style.borderColor = 'red';

            clearInterval(timerValues[currentTimerIndex].timerId); // Скидання попереднього таймера

            timerValues[currentTimerIndex].timerId = setInterval(timerTick, 1000, currentTimerIndex);

            startButton.setAttribute('disabled', 'disabled');

            stopButton.removeAttribute('disabled');

            resetButton.setAttribute('disabled', 'disabled');

        }

        if (event.target.classList.contains('stopButton')) {

            clearInterval(timerValues[currentTimerIndex].timerId);

            startButton.removeAttribute('disabled');

            stopButton.setAttribute('disabled', 'disabled');

            resetButton.removeAttribute('disabled');

        }

        if (event.target.classList.contains('resetButton')) {

            clearInterval(timerValues[currentTimerIndex].timerId);

            timerValues[currentTimerIndex] = { h: 0, m: 55, s: 0, timerId: null };

            timerUpdate(currentTimerIndex);  // Время для обратного отсчёта

            startButton.removeAttribute('disabled');

            stopButton.setAttribute('disabled', 'disabled');

            resetButton.setAttribute('disabled', 'disabled');

        }

    }

});

CSS file:

#clock {

    position: absolute;

    right: 0;

    top: 0;

    padding: 5px;

    background-color: rgba(72, 255, 0, 0.932);

}

.tablo {

    font-size: 1.5em;

    padding: 5px;

}

.timer {

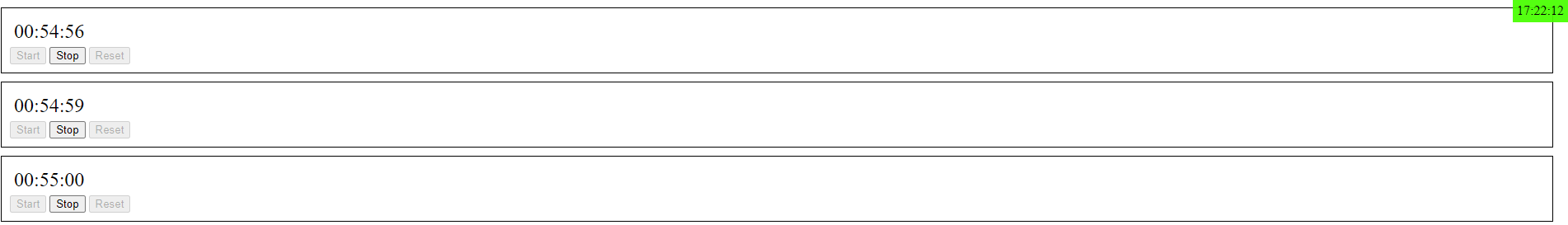
    border: 1px solid black;

    margin: 10px;

    padding: 10px;

}

**Результати виконання:**

****

**Завдання 3:**

Створіть просте слайд-шоу графічних зображень, яке буде автоматично змінювати картинки через певний інтервал часу

**Лістинг програми:**

HTML file:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Document</title>

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

</head>

<body>

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

    <div class="container">

        <img src="1.jpg" class="slide">

        <img src="2.jpg">

        <img src="3.jpg">

        <img src="4.jpg">

    </div>

</body>

</html>

Java Script file:

let slider = document.querySelector(".slide");

let sliderStyle = getComputedStyle(slider);

setInterval(() => {

    let marginLeft = parseInt(sliderStyle.marginLeft.replace("px", ""));

    if (marginLeft > -(parseInt(sliderStyle.width.replace("px", "")) \* 3)) {

        slider.style.marginLeft = `${marginLeft - parseInt(sliderStyle.width.replace("px", ""))}px`;

    } else {

        slider.style.marginLeft = "0px"

    }

}, 1000)

CSS file:

img {

    height: 100%;

    width: 100%;

}

.slide {

    transition: 0.7s ease;

}

.container {

    border-radius: 10px;

    border: 10px solid darkslateblue;

    width: 500px;

    height: 500px;

    position: absolute;

    top: 50%;

    left: 50%;

    transform: translate(-50%, -50%);

    display: flex;

    overflow: hidden;

}

body {

    background-color: darkslategrey;

}

**Результати виконання:**

****

**Завдання 4:**

Реалізуйте автоматичне переміщення декількох блоків по екрану. Блоки повинні починати рухатися у випадкових напрямках. Досягаючи границь вікна браузера блоки повинні починати рух у зворотному напрямкута.

**Лістинг програми:**

HTML File:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Document</title>

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

</head>

<body>

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

    <div class="block"></div>

    <div class="block"></div>

    <div class="block"></div>

    <div class="block"></div>

</body>

</html>

JavaScript file:

let movement = function (object, direction) {

    let windowWidth = window.innerWidth

    let windowHeight = window.innerHeight

    switch (direction) {

        case "right":

            if (object.offsetLeft + directionArray[direction] + object.offsetWidth / 2 >= windowWidth) {

                direction = "left"

            }

            object.style.left = `${object.offsetLeft + directionArray[direction]}px`;

            break;

        case "left":

            if (object.offsetLeft + directionArray[direction] - object.offsetWidth / 2 <= 0) {

                direction = "right"

            }

            object.style.left = `${object.offsetLeft + directionArray[direction]}px`;

            break;

        case "top":

            if (object.offsetTop + directionArray[direction] - object.offsetHeight / 2 <= 0) {

                direction = "bottom"

            }

            object.style.top = `${object.offsetTop + directionArray[direction]}px`;

            break;

        case "bottom":

            if (object.offsetTop + directionArray[direction] + object.offsetHeight / 2 >= windowHeight) {

                direction = "top"

            }

            object.style.top = `${object.offsetTop + directionArray[direction]}px`;

            break;

    }

    return direction

}

let directionArray = {

    "right": 5,

    "left": -5,

    "top": -5,

    "bottom": 5,

}

document.querySelectorAll(".block").forEach((block) => {

    block.style.backgroundColor = `rgb(${Math.floor(Math.random() \* 255)}, ${Math.floor(Math.random() \* 255)}, ${Math.floor(Math.random() \* 255)})`

    let directions = Object.keys(directionArray)

    let randomDirectionKey = directions[Math.floor(Math.random() \* directions.length)]

    setInterval(() => {

        randomDirectionKey = movement(block, randomDirectionKey)

    }, 20)

})

CSS file:

.block {

    width: 50px;

    height: 50px;

    opacity: 0.7;

    outline: 3px solid black;

    position: absolute;

    top: 50%;

    left: 50%;

    transform: translate(-50%, -50%);

}

**Результати виконання:**



GitHub:

https://github.com/djproblematic/frontend