**Лабораторна робота №8**

**Тема**: **Функції setInterval та setTimeout.**

**Мета роботи:** вивчити особливості відкладеного виконання програмного коду.

**Хід роботи:**

**Завдання 1.**

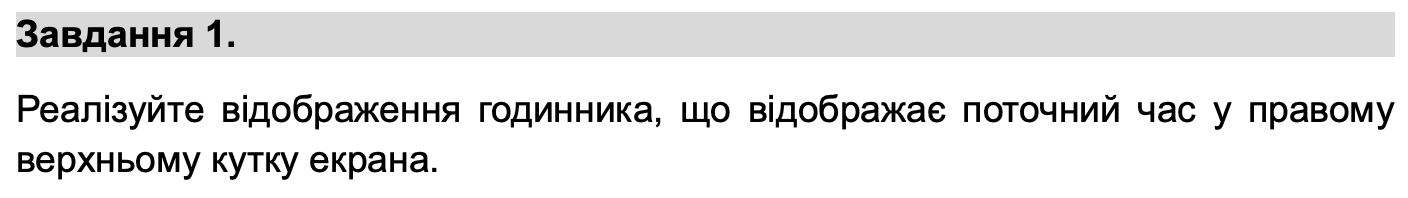


Рис. 1. Завдання 1

Виконання завдання:

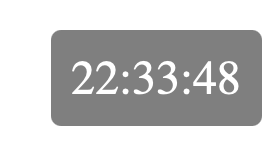


Рис. 2. Виконання завдання 1

Програма до виконаного завдання:

Html:

<!doctype html>  
<html lang="en">  
 <head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
 <link rel="stylesheet" href="style.css">  
 <script defer src="scrip.js"></script>  
 <title>Годинник</title>  
 </head>  
 <body>  
 <div id="clock"></div>  
 </body>  
</html>

Css:

#clock {  
 position: fixed;  
 top: 10px;  
 right: 10px;  
 font-size: 24px;  
 background-color: rgba(0, 0, 0, 0.5);  
 color: white;  
 padding: 10px;  
 border-radius: 5px;  
}

Js:

// Функція для оновлення часу  
function updateClock() {  
 const now = new *Date*();  
 const hours = now.getHours().toString().padStart(2, '0');  
 const minutes = now.getMinutes().toString().padStart(2, '0');  
 const seconds = now.getSeconds().toString().padStart(2, '0');  
  
 const timeString = `${hours}:${minutes}:${seconds}`;  
 *document*.getElementById('clock').textContent = timeString;  
}  
setInterval(updateClock, 1000);  
updateClock();

**Завдання 2.**

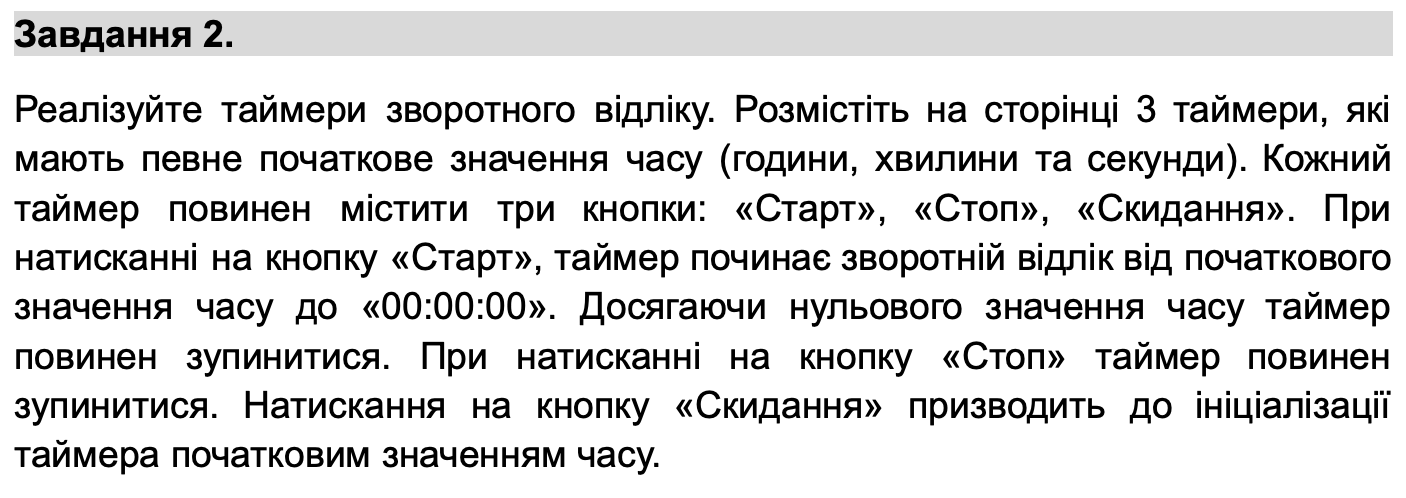


Рис. 3. Завдання 2

Виконання завдання:

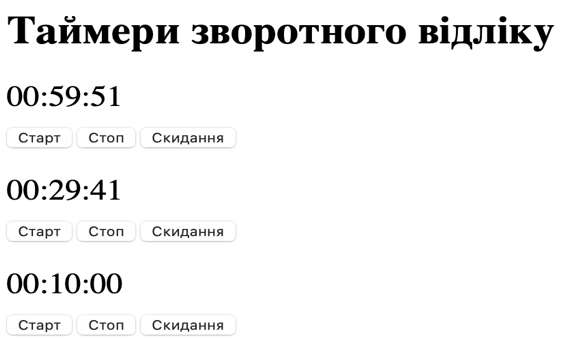


Рис. 4. Виконання завдання 2

Програма до виконаного завдання:

Html:

<!doctype html>  
<html lang="en">  
 <head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
 <link rel="stylesheet" href="style.css">  
 <script defer src="script.js"></script>  
 <title>Таймери зворотного відліку</title>  
 </head>  
 <body>  
 <h1>Таймери зворотного відліку</h1>  
 <div class="timer">  
 <div class="timer-display" id="timer1-display">01:00:00</div>  
 <button onclick="startTimer(0)">Старт</button>  
 <button onclick="stopTimer(0)">Стоп</button>  
 <button onclick="resetTimer(0, 1, 0, 0)">Скидання</button>  
 </div>  
  
 <div class="timer">  
 <div class="timer-display" id="timer2-display">00:30:00</div>  
 <button onclick="startTimer(1)">Старт</button>  
 <button onclick="stopTimer(1)">Стоп</button>  
 <button onclick="resetTimer(1, 0, 30, 0)">Скидання</button>  
 </div>  
  
 <div class="timer">  
 <div class="timer-display" id="timer3-display">00:10:00</div>  
 <button onclick="startTimer(2)">Старт</button>  
 <button onclick="stopTimer(2)">Стоп</button>  
 <button onclick="resetTimer(2, 0, 10, 0)">Скидання</button>  
 </div>  
 </body>  
</html>

Css:

.timer {  
 margin-bottom: 20px;  
}  
.timer-display {  
 font-size: 24px;  
 margin: 10px 0;  
}

Js:

const *timers* = [  
 { hours: 1, minutes: 0, seconds: 0, intervalId: null, displayId: 'timer1-display' },  
 { hours: 0, minutes: 30, seconds: 0, intervalId: null, displayId: 'timer2-display' },  
 { hours: 0, minutes: 10, seconds: 0, intervalId: null, displayId: 'timer3-display' }  
];  
  
function formatTime(hours, minutes, seconds) {  
 return `${hours.toString().padStart(2, '0')}:${minutes.toString().padStart(2, '0')}:${seconds.toString().padStart(2, '0')}`;  
}  
  
function updateDisplay(timerIndex) {  
 const timer = *timers*[timerIndex];  
 *document*.getElementById(timer.displayId).textContent = formatTime(timer.hours, timer.minutes, timer.seconds);  
}  
  
function startTimer(timerIndex) {  
 const timer = *timers*[timerIndex];  
 if (timer.intervalId) return; // Запобігає повторному запуску  
  
 timer.intervalId = setInterval(() => {  
 if (timer.hours === 0 && timer.minutes === 0 && timer.seconds === 0) {  
 stopTimer(timerIndex); // зупиняємо таймер при досягненні 00:00:00  
 return;  
 }  
 if (timer.seconds === 0) {  
 if (timer.minutes === 0) {  
 if (timer.hours > 0) {  
 timer.hours--;  
 timer.minutes = 59;  
 }  
 } else {  
 timer.minutes--;  
 }  
 timer.seconds = 59;  
 } else {  
 timer.seconds--;  
 }  
 updateDisplay(timerIndex);  
 }, 1000);  
}  
  
function stopTimer(timerIndex) {  
 const timer = *timers*[timerIndex];  
 clearInterval(timer.intervalId);  
 timer.intervalId = null;  
}

function resetTimer(timerIndex, hours, minutes, seconds) {  
 stopTimer(timerIndex);  
 const timer = *timers*[timerIndex];  
 timer.hours = hours;  
 timer.minutes = minutes;  
 timer.seconds = seconds;  
 updateDisplay(timerIndex);  
}

**Завдання 3.**

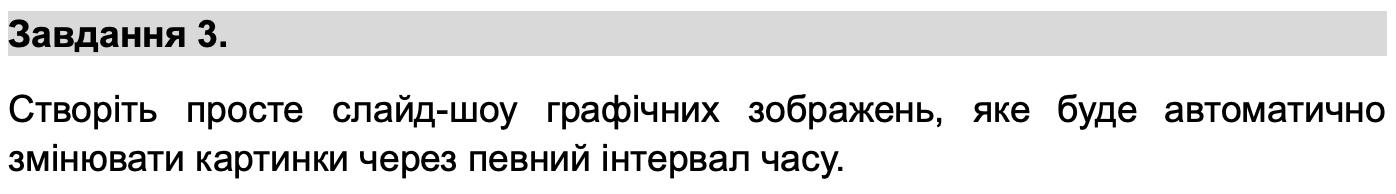


Рис. 5. Завдання 3

Виконання завдання:

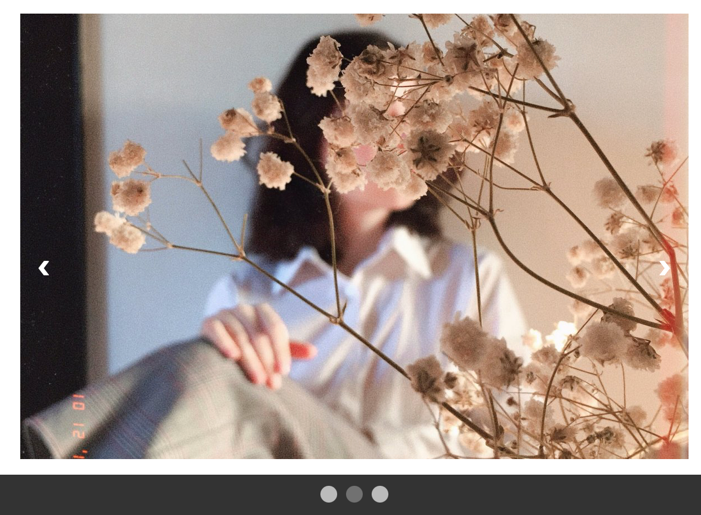


Рис. 6. Виконання завдання 3

Програма до виконаного завдання:

Html:

<!doctype html>  
<html lang="en">  
 <head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
 <link rel="stylesheet" href="style.css">  
 <script defer src="script.js"></script>  
 <title>Слайд-шоу</title>  
 </head>  
 <body>

<div class="slideshow-container">  
 <div class="slide">  
 <img src="image1.jpeg" style="width:100%">  
 </div>  
  
 <div class="slide">  
 <img src="image2.jpeg" style="width:100%">  
 </div>  
  
 <div class="slide">  
 <img src="image3.jpeg" style="width:100%">  
 </div>

<a class="prev" onclick="changeSlide(-1)">&#10094;</a>  
 <a class="next" onclick="changeSlide(1)">&#10095;</a>  
 </div>  
  
 <div class="dots">  
 <span class="dot" onclick="currentSlide(1)"></span>  
 <span class="dot" onclick="currentSlide(2)"></span>  
 <span class="dot" onclick="currentSlide(3)"></span>  
 </div>  
 </body>  
</html>

Css:

.slideshow-container {  
 width: 100%;  
 max-width: 600px;  
 position: relative;  
 margin: auto;  
 overflow: hidden;  
}  
  
.slide {  
 display: none;  
 width: 100%;  
}  
  
.slide img {  
 width: 100%;  
 height: 400px;  
 object-fit: cover;  
}  
  
.prev, .next {  
 cursor: pointer;  
 position: absolute;  
 top: 50%;  
 width: auto;  
 padding: 16px;  
 color: white;  
 font-weight: bold;

font-size: 18px;  
 transition: 0.3s;  
 user-select: none;  
 border-radius: 5px;  
}  
  
.next {  
 right: 0;  
 border-radius: 0 3px 3px 0;  
}  
.prev {  
 left: 0;  
 border-radius: 3px 0 0 3px;  
}  
  
.dots {  
 text-align: center;  
 padding: 10px;  
 background: #333;  
 margin-top: 10px;  
}  
  
.dot {  
 cursor: pointer;  
 height: 15px;  
 width: 15px;  
 margin: 0 2px;  
 background-color: #bbb;  
 border-radius: 50%;  
 display: inline-block;  
 transition: background-color 0.3s ease;  
}  
  
.active, .dot:hover {  
 background-color: #717171;  
}

Js:

let *slideIndex* = 0;  
let *slideInterval*;  
  
function showSlide(n) {  
 let slides = *document*.getElementsByClassName("slide");  
 let dots = *document*.getElementsByClassName("dot");  
 if (n >= slides.length) { *slideIndex* = 0; }  
 if (n < 0) { *slideIndex* = slides.length - 1; }  
  
 for (let slide of slides) {  
 slide.style.display = "none";  
 }  
 for (let dot of dots) {  
 dot.className = dot.className.replace(" active", "");  
 }

slides[*slideIndex*].style.display = "block";  
 dots[*slideIndex*].className += " active";  
}  
function changeSlide(n) {  
 clearInterval(*slideInterval*);  
 *slideIndex* += n;  
 showSlide(*slideIndex*);  
 startSlideshow();  
}  
  
function currentSlide(n) {  
 clearInterval(*slideInterval*);  
 *slideIndex* = n - 1;  
 showSlide(*slideIndex*);  
 startSlideshow();  
}  
  
function startSlideshow() {  
 *slideInterval* = setInterval(() => {  
 *slideIndex*++;  
 showSlide(*slideIndex*);  
 }, 3000); // Інтервал зміни слайдів – 3 секунди  
}  
  
// Ініціалізація слайд-шоу  
showSlide(*slideIndex*);  
startSlideshow();

**Завдання 4.**

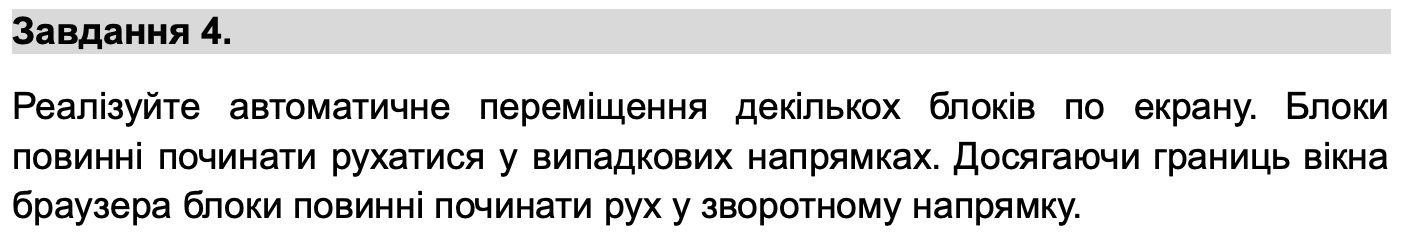


Рис. 7. Завдання 4

Виконання завдання:

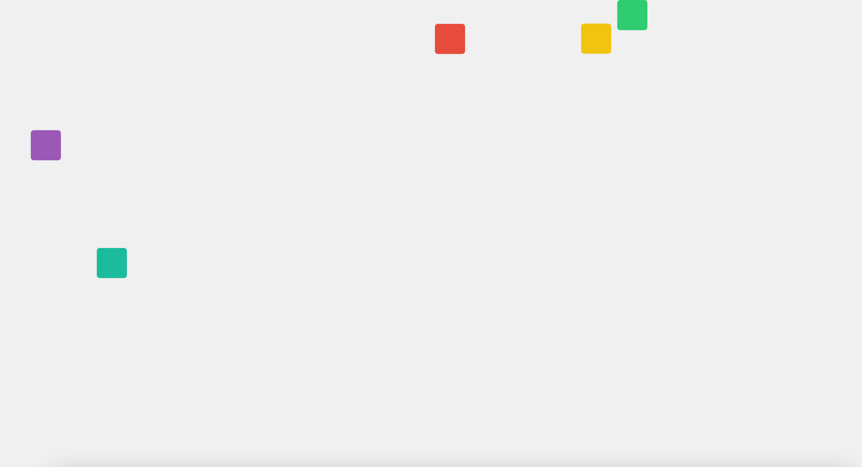
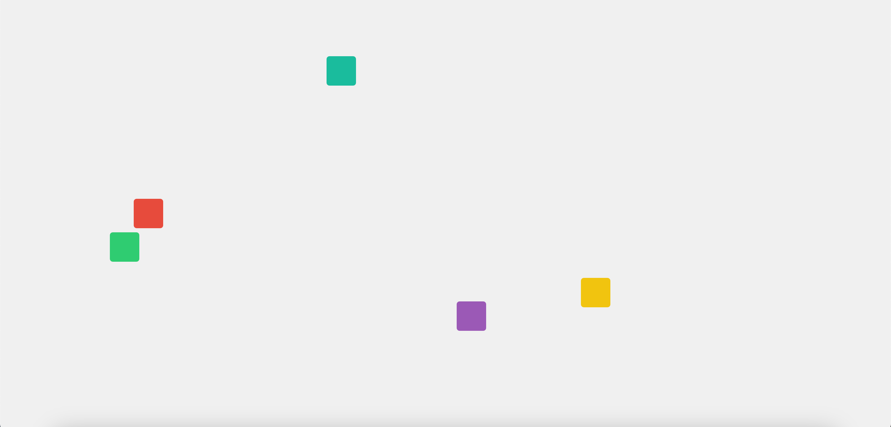
 

Рис. 8. Виконання завдання 4

Програма до виконаного завдання:

Html:

<!doctype html>  
<html lang="en">  
 <head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1">  
 <link rel="stylesheet" href="style.css">  
 <script defer src="script.js"></script>  
 <title>Автоматичне переміщення блоків</title>  
 </head>  
 <body>  
 <div class="block"></div>  
 <div class="block"></div>  
 <div class="block"></div>  
 <div class="block"></div>  
 <div class="block"></div>  
 </body>  
</html>

Css:

body {  
 overflow: hidden;  
 margin: 0;  
 display: flex;  
 justify-content: center;  
 align-items: center;  
 height: 100vh;  
 background-color: #f0f0f0;  
}  
  
.block {  
 position: absolute;  
 width: 50px;  
 height: 50px;  
 background-color: #3498db;  
 border-radius: 5px;  
}  
.block:nth-child(1) { background-color: #e74c3c; }  
.block:nth-child(2) { background-color: #2ecc71; }  
.block:nth-child(3) { background-color: #9b59b6; }  
.block:nth-child(4) { background-color: #f1c40f; }  
.block:nth-child(5) { background-color: #1abc9c; }

Js:

const *blocks* = *document*.querySelectorAll('.block');  
const *windowWidth* = *window*.innerWidth;  
const *windowHeight* = *window*.innerHeight;

// Створення об'єктів для кожного блоку з їхніми позиціями і швидкістю  
const *blockData* = *Array*.from(*blocks*).map(block => ({  
 element: block,  
 x: *Math*.random() \* (*windowWidth* - 50),  
 y: *Math*.random() \* (*windowHeight* - 50),  
 dx: (*Math*.random() \* 2 + 1) \* (*Math*.random() > 0.5 ? 1 : -1),  
 dy: (*Math*.random() \* 2 + 1) \* (*Math*.random() > 0.5 ? 1 : -1)  
}));  
  
function moveBlocks() {  
 *blockData*.forEach(data => {  
 data.x += data.dx;  
 data.y += data.dy;  
 // Перевірка на досягнення меж екрану  
 if (data.x <= 0 || data.x >= *windowWidth* - 50) {  
 data.dx \*= -1; // Зміна напрямку руху по x  
 }  
 if (data.y <= 0 || data.y >= *windowHeight* - 50) {  
 data.dy \*= -1; // Зміна напрямку руху по y  
 }  
 // Застосування нових позицій  
 data.element.style.left = `${data.x}px`;  
 data.element.style.top = `${data.y}px`;  
 });  
}  
// Оновлення позицій кожні 20 мілісекунд  
setInterval(moveBlocks, 20);

**Висновок:** У цій роботі було вивчено функції setTimeout та setInterval, які дозволяють виконувати код із затримкою або повторно через певні інтервали часу. Навчилась використовувати clearTimeout і clearInterval для зупинки таймерів. Ці знання корисні для створення таймерів, анімацій та асинхронного виконання завдань.