**Лабораторна робота № 7**

**Події миші та клавіатури***.*

**Хід роботи:**

**Завдання 1:**

**Надати квадратикам можливість перетягування (draggable) в межах контейнера*.*** При переміщенні курсора, наведеному на даний блок-квадратик, та утриманні кнопки миші даний елемент повинен перетягуватись. Квадратики не повинні заходити навіть частково за межі контейнера.

**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 id="area">

    </div>

</body>

</html>

Java Script file:

let area = document.getElementById("area");

let blocks\_counter = 4;

let indexZ = 0;

let activeBlock = null;

let isPressed = false;

let mouseCoord = {

    x: 0,

    y: 0

}

let mousePosChange = function (x, y) {

    mouseCoord.x = x;

    mouseCoord.y = y;

}

let mouseMove = function (event) {

    if (activeBlock !== null && isPressed === true) {

        let collisionX = parseInt(activeBlock.style.left) + (event.clientX - mouseCoord.x);

        let collisionY = parseInt(activeBlock.style.top) + (event.clientY - mouseCoord.y);

        if (collisionX >= 0 && collisionX + parseInt(activeBlock.style.width) <= area.clientWidth) {

            activeBlock.style.left = `${collisionX}px`;

        }

        if (collisionY >= 0 && collisionY + parseInt(activeBlock.style.height) <= area.clientHeight) {

            activeBlock.style.top = `${collisionY}px`;

        }

        mousePosChange(event.clientX, event.clientY);

    }

}

let mouseDown = function (event) {

    if (event.target.classList.contains("block") && activeBlock === null) {

        console.log("+")

        activeBlock = event.target;

        mousePosChange(event.clientX, event.clientY);

        isPressed = true;

    }

}

let mouseUp = function (event) {

    activeBlock = null;

    isPressed = false;

}

document.addEventListener("mousemove", mouseMove);

document.documentElement.addEventListener("mousedown", mouseDown);

document.documentElement.addEventListener("mouseup", mouseUp);

let createBlock = function (width, height) {

    let newDiv = document.createElement("div");

    let divSize = Math.floor(20 + Math.random() \* 20);

    let x = Math.floor(Math.random() \* (200 - divSize));

    let y = Math.floor(Math.random() \* (200 - divSize));

    let r = Math.floor(1 + Math.random() \* 255);

    let g = Math.floor(1 + Math.random() \* 255);

    let b = Math.floor(1 + Math.random() \* 255);

    newDiv.style.left = `${x}px`;

    newDiv.style.top = `${y}px`;

    newDiv.style.width = `${divSize}px`;

    newDiv.style.height = `${divSize}px`;

    newDiv.style.backgroundColor = `rgb(${r},${g},${b})`;

    newDiv.style.outline = `3px rgb(${r \* 0.8},${g \* 0.8},${b \* 0.8}) solid`;

    newDiv.classList.add("block");

    return newDiv;

}

for (let i = 0; i < blocks\_counter; ++i) {

    area.appendChild(createBlock(area.clientWidth, area.clientHeight));

}

CSS file:

#area {

    position: relative;

    width: 200px;

    height: 200px;

    border: 5px solid yellow;

    outline: 1px solid black;

    display: flex;

    top: 50%;

    left: 50%;

    transform: translate(-50%);

}

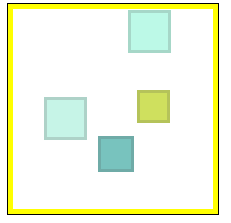
.block {

    position: absolute;

    opacity: 0.7;

}

**Результати виконання:**

****

**Завдання 2:**

**Розмістіть на сторінці випадковим чином 20 кольорових кружечків з радіусами від 10 до 30**. Перший з них є активним (реалізуйте візуальне виділення активного кружечка).  Передбачте такі можливості:

* при натисканні на клавішу «Tab» повинен ставати активним наступний кружечок.
* при натисканні комбінації клавіш “Shift-Tab”, активним повинен ставати попередній кружечок.
* клавіші стрілок повинні зміщувати активний кружечок у відповідну сторону на 10 пікселів.

**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>

</body>

</html>

Java Script file:

let keyEvent = function (event) {

    switch (event.key) {

        case "Tab":

            if (event.shiftKey) {

                previousBlock();

            } else {

                nextBlock();

            }

            event.preventDefault(); break;

        case "ArrowUp": moveActiveBlock("top", -10); break;

        case "ArrowDown": moveActiveBlock("top", 10); break;

        case "ArrowLeft": moveActiveBlock("left", -10); break;

        case "ArrowRight": moveActiveBlock("left", 10); break;

    }

}

let moveActiveBlock = function (direction, value) {

    if (direction === "top") {

        activeBlock.style.top = `${parseInt(activeBlock.style.top) + value}px`;

    } else if (direction === "left") {

        activeBlock.style.left = `${parseInt(activeBlock.style.left) + value}px`;

    }

}

let setActiveBlock = function (block) {

    block.classList.add("selected")

    activeBlock.classList.remove("selected")

    activeBlock = block

}

let previousBlock = function () {

    let preBlock = activeBlock.previousElementSibling;

    console.log(activeBlock.previousElementSibling.tagName)

    if (preBlock != null && preBlock.classList.contains("circle")) {

        setActiveBlock(preBlock)

    } else if (preBlock.tagName != "DIV") {

        setActiveBlock(document.querySelector(".circle:last-of-type"))

    }

}

let nextBlock = function () {

    let nextBlock = activeBlock.nextElementSibling;

    if (nextBlock != null && nextBlock.classList.contains("circle")) {

        setActiveBlock(nextBlock)

    } else if (nextBlock == null) {

        setActiveBlock(document.querySelector(".circle"))

    }

}

document.addEventListener("keydown", keyEvent)

let blocks\_counter = 4

let createBlock = function (width, height) {

    let newDiv = document.createElement("div");

    let divSize = Math.floor(10 + Math.random() \* 30);

    let x = Math.floor(Math.random() \* (width - divSize));

    let y = Math.floor(Math.random() \* (height - divSize));

    let r = Math.floor(1 + Math.random() \* 255);

    let g = Math.floor(1 + Math.random() \* 255);

    let b = Math.floor(1 + Math.random() \* 255);

    newDiv.style.left = `${x}px`;

    newDiv.style.top = `${y}px`;

    newDiv.style.width = `${divSize}px`;

    newDiv.style.height = `${divSize}px`;

    newDiv.style.backgroundColor = `rgb(${r},${g},${b})`;

    newDiv.style.borderRadius = "50%"

    newDiv.style.border = `3px rgb(${r \* 0.8},${g \* 0.8},${b \* 0.8}) solid`;

    newDiv.classList.add("circle");

    return newDiv;

}

for (let i = 0; i < blocks\_counter; ++i) {

    document.body.appendChild(createBlock(window.innerWidth, window.innerHeight));

}

document.querySelector(".circle").classList.add("selected")

let activeBlock = document.querySelector(".selected")

CSS file:

.circle {

    position: absolute;

}

.selected {

    outline: 5px solid red;

}

**Результати виконання:**

****

**Завдання 3:**

Реалізуйте «утікаючу» кнопку. В **div**-елементі знаходиться кнопка. Проте при наведенні на неї курсора миші, вона змінює своє положення так, що унеможливлює натиснення на неї. Кнопка не повинна виходити за межі  **div**-елемента.

**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><button class="runwaway">Click me!</button></div>

</body>

</html>

JavaScript file:

let blockPosChange = function (target, block, button) {

    let newTop;

    let newLeft;

    do {

        newTop = Math.floor(Math.random() \* block.offsetHeight);

        newLeft = Math.floor(Math.random() \* block.offsetWidth);

    } while ((newTop - button.offsetHeight / 2) <= 1 || (newTop + button.offsetHeight / 2) >= block.offsetHeight ||

    (newLeft - button.offsetWidth / 2) <= 1 || (newLeft + button.offsetWidth / 2) >= block.offsetWidth);

    target.style.top = `${newTop}px`;

    target.style.left = `${newLeft}px`;

};

let mouseMooving = function (event) {

    let divBlock = document.querySelector("div");

    let button = document.querySelector("button");

    if (event.target == button) {

        blockPosChange(event.target, divBlock, button);

    }

}

document.addEventListener("mousemove", mouseMooving);

CSS file:

button {

    width: 100px;

    height: 40px;

    position: absolute;

    top: 50%;

    left: 50%;

    transform: translate(-50%, -50%);

    background-color: aquamarine;

    transition: background-color 0.6s ease;

}

button:hover {

    background-color: violet;

}

div {

    width: 600px;

    height: 600px;

    position: absolute;

    top: 50%;

    left: 50%;

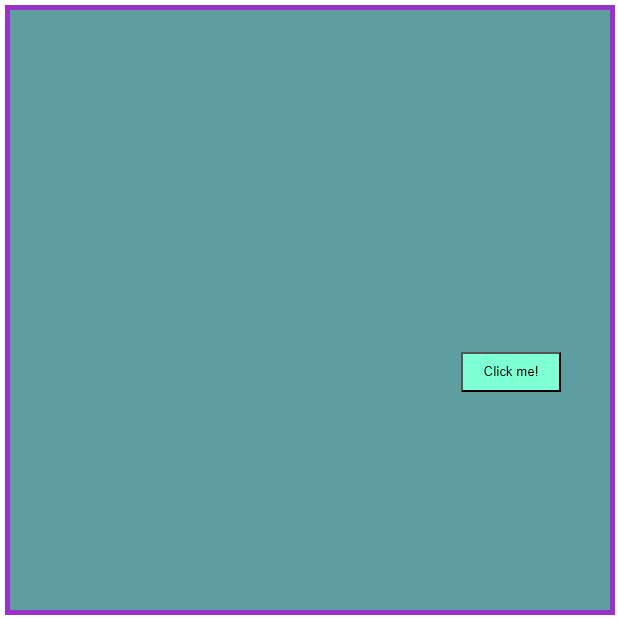
    transform: translate(-50%, -50%);

    background-color: cadetblue;

    outline: 5px solid darkorchid;

}

**Результати виконання:**



**Завдання 4:**

Розмістіть список з елементів, які можна буде виділяти, як у файлових менеджерах. Передбачте такі можливості:

* клік на елементі списку виділяє лише цей елемент (додає клас **.selected**), та знімає виділення з усіх інших;

клік за допомогою **Ctrl** (**Cmd** для **Mac**) виділяє елемент (або якщо елемент виділений, то знімає виділення з нього), виділення інших елементів при цьому не змінюються..

**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">

<body>

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

    <ul>

        <li>Some text</li>

        <li>Something</li>

        <li>Something else</li>

        <li>Also something</li>

        <li>And also something else</li>

    </ul>

</body>

</html>

JavaScript file:

let list = document.querySelectorAll("li")

list.forEach((target) => {

    target.addEventListener("click", (event) => {

        if (event.ctrlKey) {

            event.target.classList.toggle("selected")

        } else {

            list.forEach((li) => {

                if (li !== event.target) {

                    li.classList.remove("selected")

                }

            })

            event.target.classList.add("selected")

        }

    })

})

CSS file:

.selected {

    background-color: cadetblue;

    color: bisque;

}

li {

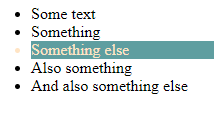
    cursor: pointer;

}

#del {

    background-image: url("del.png");

**Результати виконання:**

****

**Завдання 5:**

Реалізуйте елемент “слайдер”, який має такий вигляд:



Передбачте такі особливості його роботи:

* при наведенні курсору миші на «повзунок», користувач затискає кнопку миші і рухає «повзунок» переміщаючи курсор миші;
* при натиснутій кнопці миші, курсор може виходити за межі слайдера, але слайдер все одно має працювати (це зручно для користувача);
* слайдер повинен нормально працювати при різкому русі миші ліворуч або праворуч за межі слайдера. При цьому «повзунок» повинен зупинятися чітко біля його краю.

**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="slider">

        <div class="line"></div>

        <div class="control"></div>

    </div>

</body>

</html>

Java Script file:

let isPressed = false

let mousePosX = null

let control = document.querySelector(".control")

document.addEventListener("mousemove", (event) => {

    let newPosX = control.offsetLeft + (event.clientX - mousePosX)

    let line = document.querySelector(".line")

    if (isPressed && newPosX >= 0 && newPosX + control.offsetWidth <= line.offsetWidth) {

        control.style.left = `${newPosX}px`

        mousePosX = event.clientX

    }

})

document.addEventListener("mousedown", (event) => {

    isPressed = event.target === control ? true : false

    mousePosX = event.clientX

})

document.addEventListener("mouseup", (event) => {

    isPressed = false

})

CSS file:

.line {

    position: absolute;

    top: 50%;

    left: 50%;

    transform: translate(-50%, -50%);

    width: 100%;

    height: 70%;

    background-color: lightgray;

    border-radius: 5px;

}

.slider {

    display: collapse;

    position: absolute;

    top: 50%;

    left: 50%;

    transform: translate(-50%, -50%);

    width: 400px;

    height: 30px;

}

.control {

    position: absolute;

    top: 50%;

    transform: translate(0, -50%);

    width: 10px;

    height: 100%;

    background-color: darkorange;

    border-radius: 5px;

    transition: 0.6s background-color;

    cursor: pointer;

}

.control:hover {

    background-color: rgb(168, 103, 24);

}

**Виконання програми:**

****

GitHub:

https://github.com/djproblematic/frontend