В данной лабораторной работе у вас есть 3 файла. Файлы .html .css изменять нельзя, все задания выполняете при помощи .js файла.

Что необходимо сделать.

Прилагаемый HTML-файл содержит 3 div, с которыми должен работать ваш код, однако ваш код должен работать независимо от того, сколько имеется div (он не должен быть привязан к этим конкретным трём div), но вы можете предположить, что они все будут одного и того же класса target. Различные события мыши должны вызывать разные отклики элементов div: становиться выбранными или отменяться при щелчке, перемещаться, если пользователь нажимает и перетаскивает (или если пользователь дважды щелкает по ним).

При помощи обработчиков событий (EventListener) необходимо сделать следующее:

1. Возможность перетаскивать элементы (т.е. нажатие на элемент курсором мыши и дальнейшее перемещение без отпускания лкм должно увлекать элемент за мышью). При отпускании лкм элемент должен отвязаться от мыши, но остаться на том же месте.

2. Двойной клик по элементу должен «приклеить» элемент к мыши и перемещаться вслед за ней, даже если лкм мыши была отжата, при этом перемещаемый элемент меняет цвет. Во время такого типа перетаскивание открепление элемента от курсора происходит по клику мыши.

3. Нажатие клавиши esc во время выполнения действий выше должно прервать перетаскивание и вернуть элемент на исходную позицию.

// Получаем все элементы с классом 'target'

const targets = document.querySelectorAll('.target');

// Переменные для отслеживания состояния

let isDragging = false;

let isPinned = false;

let currentElement = null;

let offsetX, offsetY;

let originalPosition = {};

// Функция для начала перетаскивания

function startDrag(event) {

if (!isPinned) {

isDragging = true;

currentElement = event.target;

offsetX = event.clientX - currentElement.getBoundingClientRect().left;

offsetY = event.clientY - currentElement.getBoundingClientRect().top;

}

}

// Функция для выполнения перетаскивания

function drag(event) {

if (isDragging && currentElement) {

currentElement.style.position = 'absolute';

currentElement.style.left = `${event.clientX - offsetX}px`;

currentElement.style.top = `${event.clientY - offsetY}px`;

}

}

// Функция для завершения перетаскивания

function endDrag() {

isDragging = false;

currentElement = null;

}

// Функция для двойного клика

function doubleClick(event) {

isPinned = true;

currentElement = event.target;

originalPosition = {

left: currentElement.style.left || '0px',

top: currentElement.style.top || '0px',

};

currentElement.style.backgroundColor = 'lightblue'; // меняем цвет

}

// Функция для отключения пиннинга

function unpin() {

if (currentElement) {

isPinned = false;

currentElement.style.backgroundColor = ''; // сбрасываем цвет

currentElement = null;

}

}

// Функция для обработки нажатия клавиши Esc

function handleKeyDown(event) {

if (event.key === 'Escape' && currentElement) {

currentElement.style.left = originalPosition.left;

currentElement.style.top = originalPosition.top;

unpin();

}

}

// Добавляем обработчики событий к каждому элементу

targets.forEach(target => {

target.addEventListener('mousedown', startDrag);

target.addEventListener('dblclick', doubleClick);

});

document.addEventListener('mousemove', drag);

document.addEventListener('mouseup', endDrag);

document.addEventListener('click', unpin);

document.addEventListener('keydown', handleKeyDown);