Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем и программной инженерии

Лабораторная работа № 17-18

по дисциплине "Программирование графических приложений"

ТЕМА РАБОТЫ:

Разработка игровой анимированной сцены

Выполнил:

студент гр. ПРИм-124

Парахин К.В.

Принял:

Жигалов И.Е.

Владимир 2024 г.

Цель работы:

Изучение методов создания игровой анимированной 3D-сцены в WebGL на JavaScript с использованием библиотеки Three.js.

Выполнение работы:

Ознакомиться по методическим указаниям и литературе с теоретическим материалом тем №17 и №18. Изучить программные коды разделов. 2. На основе приведенных примеров программ дополнить сцены, аналогичные построенным в результате выполнения лабораторных работ по темам 5 и 9, игровой логикой, добавив необходимое управление объектами сцены пользователем и игровым роботом, взаимодействие объектов, формирование и вывод на экран игровой информации.

Сделаю игру, в которой осуществляется перемещение рыбки по аквариуму.  
  
Рыбка импортируется как .png изображение без фона, во время игры она все время плывет вправо – по мере ее перемещения справа от нее генерируется уровень (а именно вертикальные препятствия между которыми рыбка может проплыть) – пользователь имеет возможность перемещать рыбку, двигая мышь вверх/вниз.   
По мере наплыва на препятствия накапливается счетчик столкновений, и когда он становится равен пределу (100) – то игра заканчивается  
Смысл игры – просто плыть как можно большее время, оставаясь в игре

Листинг кода:

// html файл

<!DOCTYPE html>

<html lang="ru">

<head>

    <meta charset="UTF-8">

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

    <title>Игра</title>

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

</head>

<body>

    <div id="game-container">

        <canvas id="gameCanvas"></canvas>

        <div id="info-panel">

            <span id="score">Столкновений: 0</span>

            <span id="score">Максимум: 100</span>

            <span id="time">Время: 0 секунд</span>

        </div>

    </div>

    <script src="game.js"></script>

</body>

</html>

// css файл

\* {

    margin: 0;

    padding: 0;

    box-sizing: border-box;

}

body, html {

    height: 100%;

    background-color: #aee2e0; /\* Цвет фона воды \*/

    display: flex;

    justify-content: center;

    align-items: center;

}

#game-container {

    position: relative;

}

#gameCanvas {

    background: linear-gradient(to top, #4ba3c7, #aee2e0); /\* Аквариум \*/

    border: 2px solid #0077b6;

    box-shadow: 0 0 15px rgba(0, 0, 0, 0.5);

}

#info-panel {

    position: absolute;

    top: 10px;

    left: 10px;

    color: #fff;

    font-family: Arial, sans-serif;

    font-size: 18px;

}

#score, #time {

    margin-right: 20px;

}

// js файл

const canvas = document.getElementById("gameCanvas");

const ctx = canvas.getContext("2d");

canvas.width = 800;

canvas.height = 600;

let fishY = canvas.height / 2;

let fishSpeed = 2;

let fishHeight = 50;

let fishWidth = 50;

let obstacles = [];

let obstacleWidth = 60;

let obstacleGap = 200;

let obstacleSpeed = 2;

let score = 0;

let hitCount = 0;

let startTime = Date.now();

let gameOver = false;

const fishImage = new Image();

fishImage.src = "fish.png";

fishImage.onload = () => {

    console.log('Image loaded successfully!');

    gameLoop();

};

fishImage.onerror = () => {

    console.log('Error loading fish image');

};

canvas.addEventListener('mousemove', (e) => {

    if (!gameOver) {

        fishY = e.clientY - canvas.getBoundingClientRect().top;

    }

});

function createObstacle() {

    let obstacleHeight, lowerObstacleHeight;

    do {

        obstacleHeight = Math.floor(Math.random() \* (canvas.height - obstacleGap));

        lowerObstacleHeight = obstacleHeight + obstacleGap;

    } while (lowerObstacleHeight + fishHeight > canvas.height || lowerObstacleHeight - obstacleHeight < fishHeight);

    let lastObstacle = obstacles[obstacles.length - 1];

    let xPos = canvas.width;

    if (lastObstacle) {

        const minGap = fishWidth \* 2;

        xPos = lastObstacle.x + obstacleWidth + minGap;

    }

    obstacles.push({ x: xPos, y: obstacleHeight, lowerY: lowerObstacleHeight });

}

hit\_count\_max = 100;

function updateObstacles() {

    for (let i = obstacles.length - 1; i >= 0; i--) {

        obstacles[i].x -= obstacleSpeed;

        if (

            obstacles[i].x < fishWidth &&

            obstacles[i].x + obstacleWidth > 0 &&

            (fishY < obstacles[i].y || fishY + fishHeight > obstacles[i].lowerY)

        ) {

            hitCount++;

            if (hitCount > hit\_count\_max) {

                gameOver = true;

            }

        }

        if (obstacles[i].x + obstacleWidth < 0) {

            obstacles.splice(i, 1);

        }

    }

}

function drawObstacles() {

    ctx.fillStyle = "#d9534f";

    obstacles.forEach((obstacle) => {

        ctx.fillRect(obstacle.x, 0, obstacleWidth, obstacle.y);

        ctx.fillRect(obstacle.x, obstacle.lowerY, obstacleWidth, canvas.height - obstacle.lowerY);

    });

}

function drawFish() {

    ctx.drawImage(fishImage, 0, fishY - fishHeight / 2, fishWidth, fishHeight);

}

function drawInfo() {

    const elapsedTime = Math.floor((Date.now() - startTime) / 1000);

    document.getElementById("score").textContent = `Счёт: ${hitCount}`;

    document.getElementById("time").textContent = `Время: ${elapsedTime} секунд`;

    if (gameOver) {

        ctx.fillStyle = "rgba(0, 0, 0, 0.5)";

        ctx.fillRect(0, canvas.height / 3, canvas.width, 100);

        ctx.fillStyle = "#fff";

        ctx.font = "30px Arial";

        ctx.textAlign = "center";

        ctx.fillText("Игра окончена! Превышено количество столкновений.", canvas.width / 2, canvas.height / 2);

    }

}

function gameLoop() {

    if (gameOver) return;

    ctx.clearRect(0, 0, canvas.width, canvas.height);

    drawFish();

    updateObstacles();

    drawObstacles();

    drawInfo();

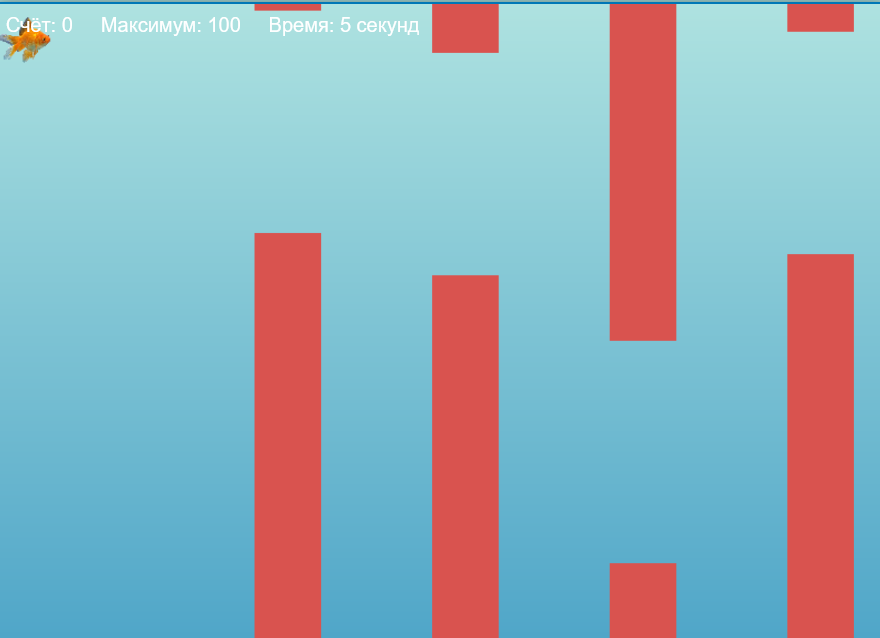
    if (Math.random() < 0.01) {

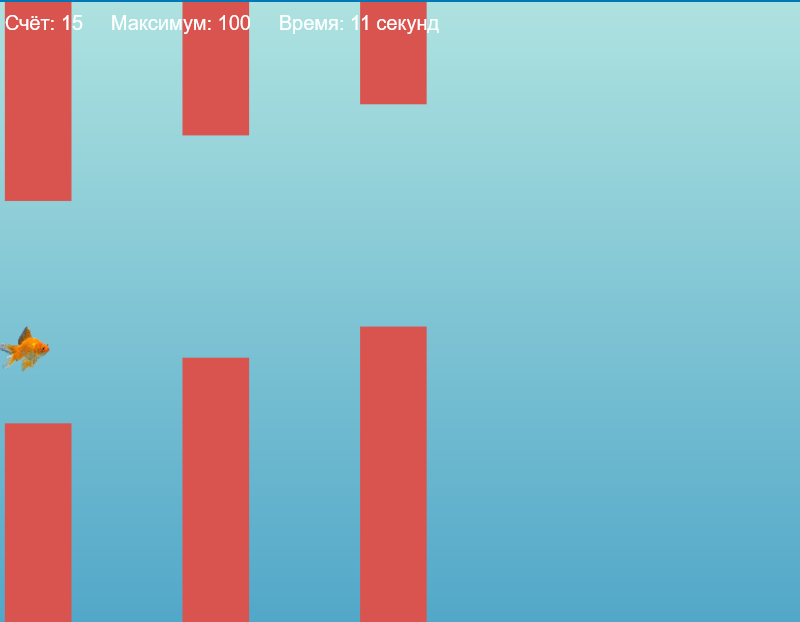
        createObstacle();

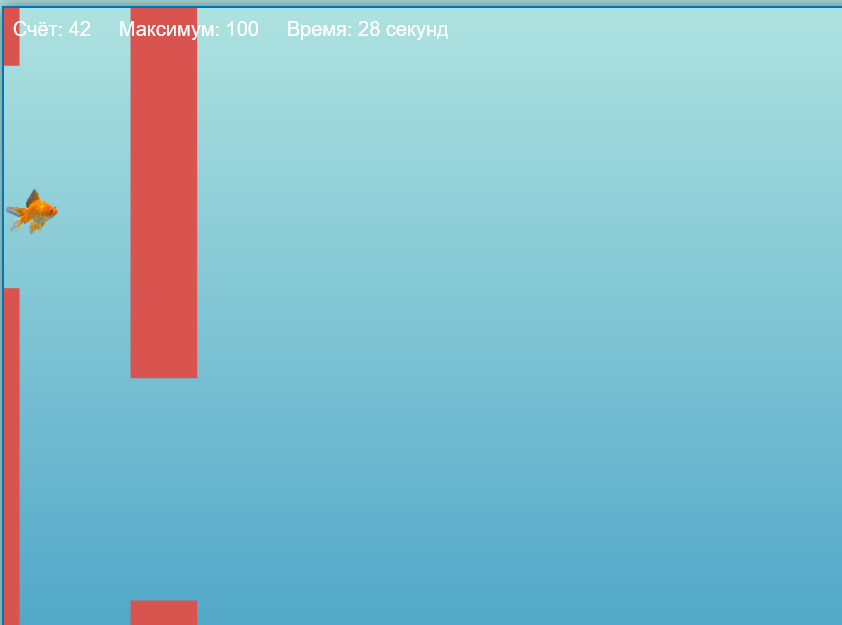
    }

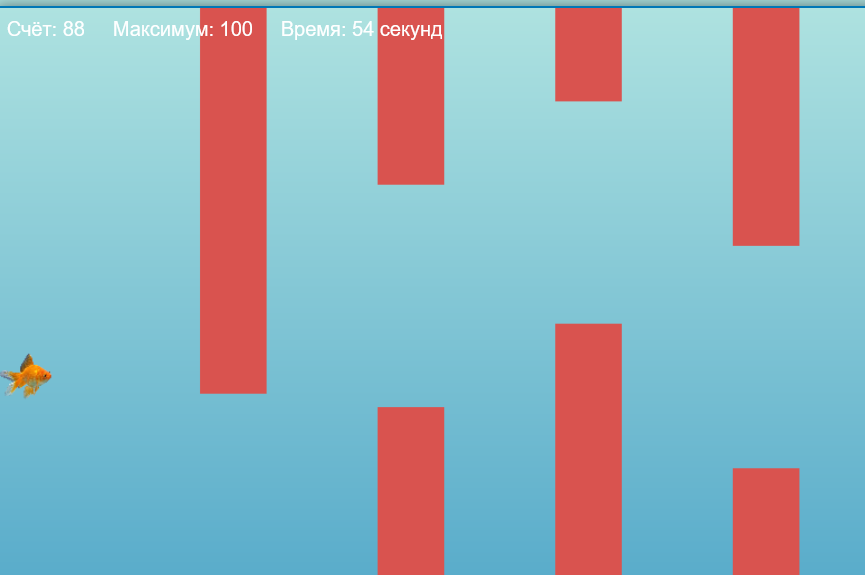
    requestAnimationFrame(gameLoop);

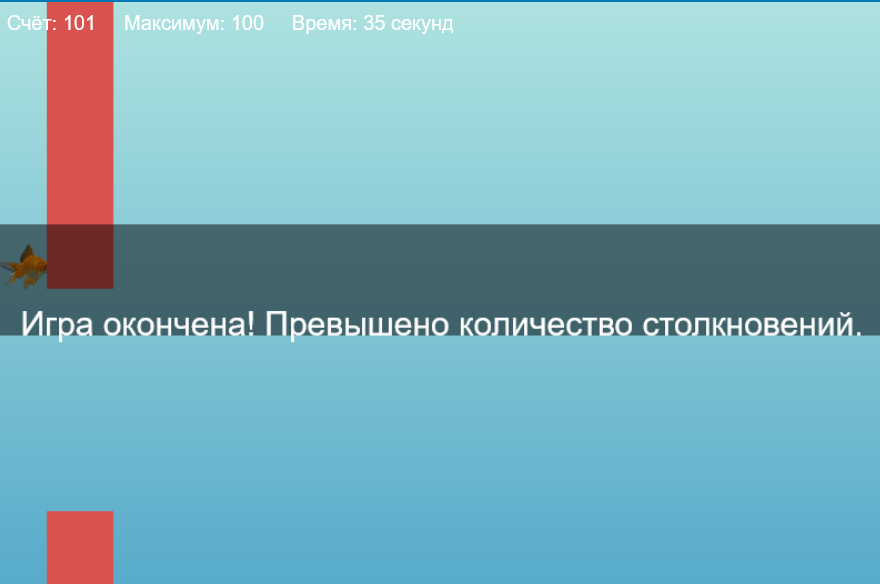
}

Скриншоты игры:  
  










Вывод

В результате выполнения работы я провел изучение методов создания игровой анимированной 3D-сцены в WebGL на JavaScript с использованием библиотеки Three.js.