**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №1**

**по дисциплине «WEB-технологии»**

Тема: Тетрис на JavaScript

|  |  |  |
| --- | --- | --- |
| Студент гр. 0304 |  | Жиглов Д.С. |
| Преподаватель |  | Беляев С.А. |

Санкт-Петербург

2022

## Цель работы.

Целью работы является изучение работы web-сервера nginx со статическими файлами и создание клиентских JavaScript web-приложений.

## Задание.

Необходимо создать web-приложение – игру в тетрис. Основные требования:

– сервер – nginx, протокол взаимодействия – HTTPS версии не ниже 2.0;

– отображается страница для ввода имени пользователя с использованием HTML-элементов <input>;

– статическая страница отображает «стакан» для тетриса с использованием HTML-элемента <canvas>, элемент <div> используется для отображения следующей фигуры, отображается имя пользователя;

– фигуры в игре – классические фигуры тетриса (7 шт. тетрамино);

– случайным образом генерируется фигура и начинает падать в «стакан»;

– пользователь имеет возможность двигать фигуру влево и вправо, повернуть на 90º и «уронить»;

– если собралась целая «строка», она должна исчезнуть;

– при наборе некоторого заданного числа очков увеличивается уровень,

что заключается в увеличении скорости игры;

– пользователь проигрывает, когда стакан «заполняется», после чего ему

отображается локальная таблица рекордов;

– вся логика приложения написана на JavaScript.

Необязательно: оформление с использованием CSS.

## Основные теоретические положения.

Асимметричные ключи используются в асимметричных алгоритмах шифрования и являются ключевой парой. Закрытый ключ известен только владельцу. Открытый ключ может быть опубликован и используется для проверки подлинности подписанного документа (сообщения). Открытый ключ вычисляется, как значение некоторой функции от закрытого ключа, но знание открытого ключа не дает возможности определить закрытый ключ. По секретному ключу можно вычислить открытый ключ, но по открытому ключу практически невозможно вычислить закрытый ключ.

nginx — веб-сервер, работающий на Unix-подобных операционных системах и в операционной системе Windows.

JavaScript — язык программирования, он поддерживает объектно-ориентированный и функциональный стили программирования. Является реализацией стандарта ECMAScript.

## Выполнение работы.

1. Создание части страницы ввода имени игрока

Для получения имени игрока был создан div на начальной странице index.html (рис. 1), которая просит игрока ввести имя, под которым в таблицу рекордов будет занесён его лучший результат. Для корректного отображения имени, максимальная длина была ограничена 16 символами.

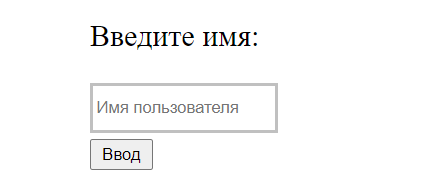


Рисунок 1. Страница index.html

1. Создание html страницы игры

Для создания игры была разработана разметка html страницы main.html, которая включает блок <table> с пользовательской информацией и блок <div> класса “main”, который создаёт стакан для тетрамино. Для удобной настройки стилей был создан style.css, в котором задаются стили для приведённых блоков.

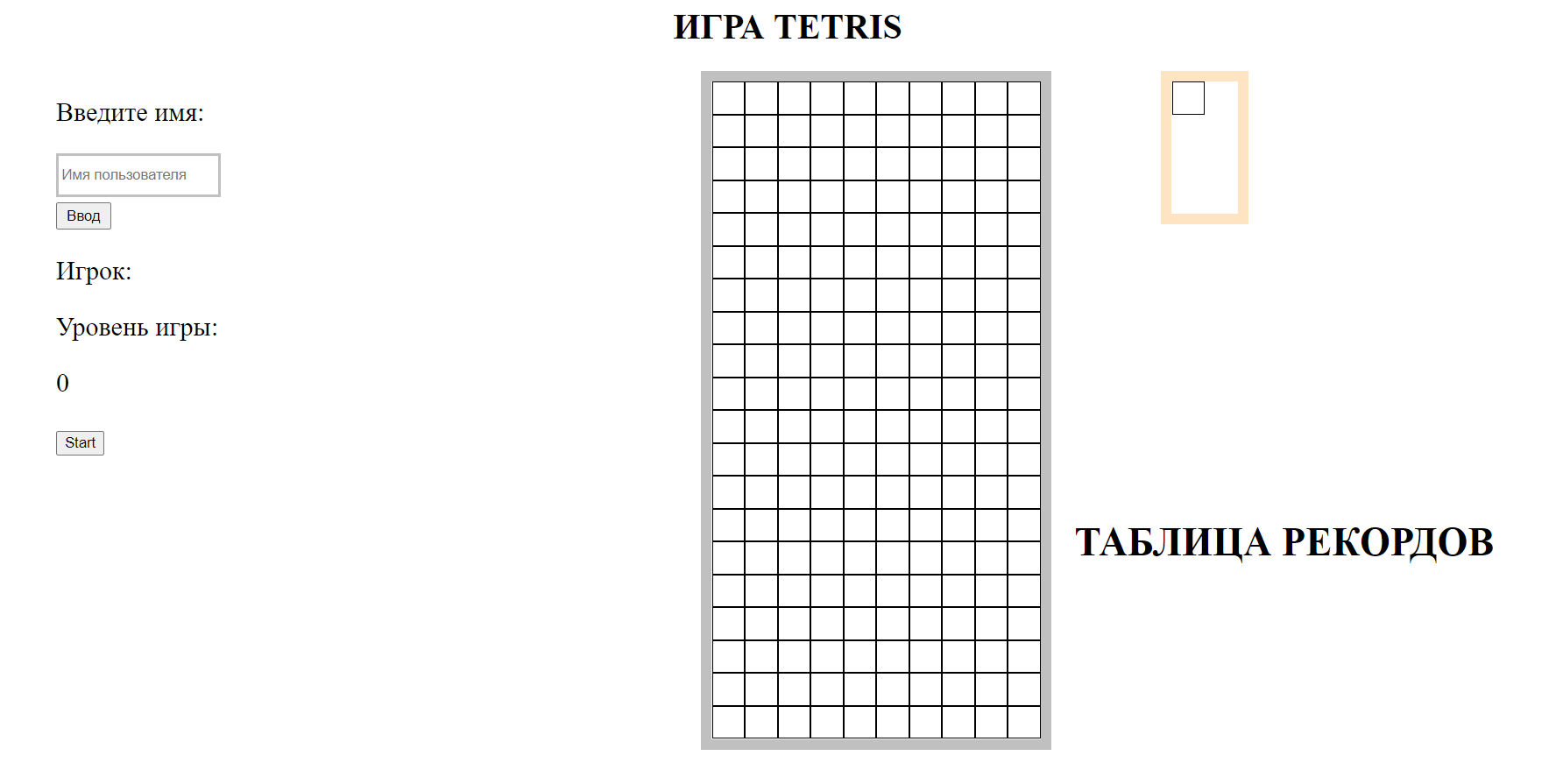


Рисунок 2. Страница main.html

1. Создание новых тетрамино.

Для создания новых фигурок на поле, была прописана функция draw\_next\_figure(), которая пробегается по верху стакана и записывает в него новое тетрамино, если же места под новую фигурку нет, значит башня достигла верха и игра проиграна.

Были прописаны функции для движения тетрамино :

* move\_down— проверяет возможность объекта переместиться вниз при условии что под фигурой нет другой фигуры, так же внутри данной функции вызывается функция check\_bottom, которая проверяет, не достигла ли фигурка дна.
* check\_rotation(delta\_x, delta\_y) — проверка на возможность поворота и установка новых координат после поворота.
* Move\_left(),move\_right()— установка новых координат после движения влево и вправо.
* Check\_line() – падение всех фигурок что были выше линии, которая полностью заполнилась и удалилась.

4. Создание внутренней логики main.js

В main.js создаются функции по созданию тетрамино, проверки линий в стакане, обработки окончания игры.

* Draw\_next\_figure() - создание новой тетрамино.
* clean\_line() - отчистка линии.
* check\_line() - проверка линий на заполненность.
* game\_over() - обработка конца игры и сохранение результатов.
* set\_next\_level() - переход на следующий уровень.

5. Считывание ввода.

Для реализации считывания клавиш был создан файл input.js, который функцией addEventListener() слушает нажатые клавиши и с помощью switch() вызывает функцию move() с необходимыми параметрами.

**Выводы.**

В ходе лабораторной работы были изучены основы JavaScript, HTML и CSS. Применены знания по созданию nginx сервера на системе Windows. Для обеспечения HTTPS протокола был сгенерирован самоподписной сертификат.

# Приложение А Исходный код программы

Файл: main.js

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

let next\_figure = document.querySelector("#next\_figure")

let records = document.querySelector("#record")

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

difficulty.innerHTML = 0;

let difficulty\_number = 0;

let player = document.querySelector("#player");

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

let resultElement = document.querySelector(".result")

button.addEventListener("click", function (){

resultElement.innerHTML = player.value;

});

let i = 0;

let Game = true;

let gameSpeed = 400;

let timerID;

class Point {

constructor(x\_coords, y\_coords,coords) {

this.x\_coords = y\_coords

this.y\_coords = x\_coords

this.coords = coords

}

}

let figure1 = new Point;

let figure2 = new Point;

let figures = [

[1, 3, 5, 7],

[2, 4, 5, 7],

[3, 5, 4, 6],

[3, 5, 4, 7],

[2, 3, 5, 7],

[3, 5, 7, 6],

[2, 3, 4, 5],

[1, 2, 3, 4],

]

let playfield = [

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],

]

function generate\_new\_figure(){

let random = getRandomInt(7);

let x\_coords = measure\_coords\_x(figures[random]);

let y\_coords = measure\_coords\_y(figures[random]);

figure1 = new Point(x\_coords, y\_coords,figures[random]);

random = getRandomInt(7);

x\_coords = measure\_coords\_x(figures[random]);

y\_coords = measure\_coords\_y(figures[random]);

figure2 = new Point(x\_coords,y\_coords,figures[random])

}

function getRandomInt(max) {

return Math.floor(Math.random() \* max);

}

function measure\_coords\_x(arr){

let x\_coords = []

for (let i = 0; i < 4; i++) {

x\_coords[i] = (arr[i] % 2) + 3;

}

return x\_coords

}

function measure\_coords\_y(arr){

let y\_coords = []

if(arr[0] !== 1) {

for (let i = 0; i < 4; i++) {

y\_coords[i] = (Math.floor(arr[i] / 2)) - 1;

}

}

else{

for (let i = 0; i < 4; i++) {

y\_coords[i] = (Math.floor(arr[i] / 2));

}

}

return y\_coords

}

function draw\_next\_figure() {

let next\_figureIneerHTML = "";

for(let x = 0; x <4;x++){

for (let y = 0; y < 2; y++) {

let count = 0;

for(let i = 0;i < 4;i++){

if (figure2.y\_coords[i]-3 === y && figure2.x\_coords[i] === x) {

next\_figureIneerHTML += '<div class = "cell fixedCell"></div>';

count = 1;

}

}

if(count !== 1){

next\_figureIneerHTML += '<div class = "cell "></div>';}

}

}

next\_figure.innerHTML = next\_figureIneerHTML;

}

function draw(playfield){

let mainInnerHTML = "";

for(let y = 0; y < playfield.length;y++){

for (let x = 0; x < playfield[y].length; x++) {

if(playfield[y][x] === 1){

mainInnerHTML += '<div class = "cell movingCell"></div>';

}

if(playfield[y][x] === 0){

mainInnerHTML += '<div class = "cell "></div>';

}

if(playfield[y][x] === 2) {

mainInnerHTML += '<div class = "cell fixedCell "></div>';

}

}

}

main.innerHTML = mainInnerHTML;

}

function check\_line() {

for (let y = 0; y < playfield.length; y++) {

let count\_line = 0

let y\_axis = 0

for (let x = 0; x < playfield[y].length; x++) {

if(playfield[y][x] === 2){

count\_line += 1

}

if(count\_line === 10){

y\_axis = y

clearInterval(timerID);

gameSpeed -= 20;

difficulty\_number += 1;

difficulty.innerHTML = difficulty\_number;

timerID = setInterval(UpdateGame, gameSpeed);

for(let x = 0; x < playfield[y].length;x++){

playfield[y\_axis][x] = 0;

}

for (let y = y\_axis; y >= 0; y--) {

for (let x = 0; x < playfield[y].length; x++) {

if(playfield[y][x] === 2){

playfield[y][x] = 0;

playfield[y+1][x] = 2;

}

}

}

}

}

}

}

function check\_bottom(){

let biggest = Math.max.apply(null, figure1.x\_coords);

let index\_last = (figure1.x\_coords.indexOf(biggest,0))

if(figure1.x\_coords[index\_last]+1 >= 20){

return true;

}

for (let i = 0; i < 4; i++) {

if( playfield[figure1.x\_coords[i]+1][figure1.y\_coords[i]]=== 2){

return true;

}

}

return false;

}

function check\_left\_borders() {

for (let i = 0; i < 4; i++) {

if (figure1.y\_coords[i] === 0) {

return true;

}

}

for (let i = 0; i < 4; i++) {

if (playfield[figure1.x\_coords[i]][figure1.y\_coords[i] - 1] === 2) {

return true;

}

}

return false;

}

function check\_rotation(x\_coords,y\_coords) {

for (let i = 0; i < 4; i++) {

if( playfield[x\_coords[i]][y\_coords[i]]=== 2 || y\_coords[i] === -1 || y\_coords[i] === 10){

return true

}

}

return false;

}

function check\_right\_borders() {

for (let i = 0; i < 4; i++) {

if (figure1.y\_coords[i] >= 9) {

return true;

}

}

for (let i = 0; i < 4; i++) {

if (playfield[figure1.x\_coords[i]][figure1.y\_coords[i] + 1] === 2) {

return true;

}

}

return false;

}

draw(playfield)

function move\_down() {

draw(playfield)

if (check\_bottom(figure1.x\_coords) === false) {

for (let i = 0; i < 4; i++) {

playfield[figure1.x\_coords[i]][figure1.y\_coords[i]] = 0

}

for (let i = 0; i < 4; i++) {

figure1.x\_coords[i] += 1;

}

for (let i = 0; i < 4; i++) {

playfield[figure1.x\_coords[i]][figure1.y\_coords[i]] = 1

}

} else {

for (let i = 0; i < 4; i++) {

playfield[figure1.x\_coords[i]][figure1.y\_coords[i]] = 2

}

check\_line()

figure1 = figure2;

let random = getRandomInt(7);

let x\_coords = measure\_coords\_x(figures[random]);

let y\_coords = measure\_coords\_y(figures[random]);

figure2 = new Point(x\_coords, y\_coords,figures[random]);

console.log(figure2)

draw\_next\_figure();

for (let i = 0; i < 4; i++) {

playfield[figure1.x\_coords[i]][figure1.y\_coords[i]] = 1

}

if(check\_bottom() === true){

for (let i = 0; i < 4; i++) {

playfield[figure1.x\_coords[i]][figure1.y\_coords[i]] = 2

}

draw(playfield)

clearInterval(timerID)

console.log("end")

Game = false;

localStorage.setItem(player.value.toString(),difficulty\_number.toString())

//localStorage.setItem('person',JSON.stringify(gamer));

// let raw = localStorage.getItem(player.value.toString());

//let name = JSON.stringify(gamer.name).replace(/"/g,'');

//console.log(raw)

// for(let key in localStorage)

// {

// if (!localStorage.hasOwnProperty(key))

// { continue; }

// else{

// records.append(`${key}: ${localStorage.getItem(key)}`)

//

// }

// }

records.append(`${localStorage.key(i)}: ${localStorage.getItem(localStorage.key(i))}`)

records.innerHTML += '<br>'

i++;

//localStorage.clear()

// records.append(localStorage.key(0))

// records.append(":")

// records.append(JSON.stringify(gamer.level))

//alert("pizdec")

}

}

draw(playfield);

}

function move\_left() {

if (check\_left\_borders() === false) {

for (let i = 0; i < 4; i++) {

playfield[figure1.x\_coords[i]][figure1.y\_coords[i]] = 0

}

for (let i = 0; i < 4; i++) {

figure1.y\_coords[i] -= 1;

}

for (let i = 0; i < 4; i++) {

playfield[figure1.x\_coords[i]][figure1.y\_coords[i]] = 1

}

}

}

function move\_right() {

if (check\_right\_borders() === false) {

for (let i = 0; i < 4; i++) {

playfield[figure1.x\_coords[i]][figure1.y\_coords[i]] = 0

}

for (let i = 0; i < 4; i++) {

figure1.y\_coords[i] += 1;

}

for (let i = 0; i < 4; i++) {

playfield[figure1.x\_coords[i]][figure1.y\_coords[i]] = 1

}

}

}

function rotate(){

for (let i = 0; i < 4; i++) {

playfield[figure1.x\_coords[i]][figure1.y\_coords[i]] = 0

}

let central\_point\_x = figure1.x\_coords[1];

let central\_point\_y = figure1.y\_coords[1];

let x\_coords = []

let y\_coords = []

for (let i = 0; i < 4; i++) {

let x = figure1.y\_coords[i] - central\_point\_y;

let y = figure1.x\_coords[i] - central\_point\_x;

x\_coords[i] = central\_point\_x + x;

y\_coords[i] = central\_point\_y - y;

}

if(check\_rotation(x\_coords,y\_coords) === false ){

for (let i = 0; i < 4; i++) {

figure1.y\_coords[i] = y\_coords[i];

figure1.x\_coords[i] = x\_coords[i];

playfield[figure1.x\_coords[i]][figure1.y\_coords[i]] = 1

}

}

else{

for (let i = 0; i < 4; i++) {

playfield[figure1.x\_coords[i]][figure1.y\_coords[i]] = 1

}

}

draw(playfield);

}

function UpdateGame() {

//move\_down();

//timerID = setTimeout(UpdateGame, gameSpeed);

draw(playfield);

//sleep(2000);

move\_down();

}

document.getElementById("button").addEventListener("click", StartGame)

function StartGame(){

// let random = getRandomInt(7);

// let x\_coords = measure\_coords\_x(figures[random]);

// let y\_coords = measure\_coords\_y(figures[random]);

// figure1 = new Point(x\_coords, y\_coords);

generate\_new\_figure();

gameSpeed = 400;

clearInterval(timerID);

difficulty\_number = 0;

difficulty.innerHTML = 0;

Game = true;

for(let y = 0; y < playfield.length;y++){

for (let x = 0; x < playfield[y].length; x++) {

playfield[y][x] = 0;

}

}

draw\_next\_figure();

for (let i = 0; i < 4; i++) {

playfield[figure1.x\_coords[i]][figure1.y\_coords[i]] = 1

}

draw(playfield);

//UpdateGame();

timerID = setInterval(UpdateGame, gameSpeed);

}

document.onkeydown = function (event) {

if(Game !== false) {

switch (event.code) {

case "ArrowDown":

move\_down();

break;

case "ArrowLeft":

move\_left();

break;

case "ArrowRight":

move\_right();

break;

case "ArrowUp":

rotate();

break;

}

}

draw(playfield)

}

файл: index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Title</title>

</head>

<body>

<h1>ИГРА TETRIS</h1>

<div class = "header">

<div id="table">

<p>Введите имя:</p>

<input id="player" type="text" placeholder="Имя пользователя" value=""><br>

<button id="ipt" >Ввод </button>

<p>Игрок:</p>

<p class = "result"></p>

<p>Уровень игры: </p>

<p class = "difficulty"></p>

<button id = "button">Start</button>

</div>

<div class="main">

<div class = "cell"></div>

<div class = "cell movingCell"></div>

<div class = "cell fixedCell"></div>

</div>

<div id = "next\_figure">

<div class = "cell"></div>

</div>

<div id = "record">

<h2>ТАБЛИЦА РЕКОРДОВ</h2>

</div>

</div>

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

</body>

</html>