Санкт-Петербургский политехнический университет Петра Великого

Институт прикладно математики и механики

**Кафедра «Теоретическая механика»**

**КУРСОВОЙ ПРОЕКТ**

**Реализация и визуализация приложения «2048»**

по дисциплине «Основы программирования и алгоритмизации»

Выполнил   
студент гр.3630103/00001 Ананьчев Егор

Руководитель

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Санкт-Петербург

2021

**СОДЕРЖАНИЕ**

Введение………………………………………………………………3

Постановка задачи……………………………………………………3

Описание кода HTML………………………………………………...4

Описание кода CSS…………………………………………………...4

Описание кода JavaScript…………………………………………….5

Результаты…………………………………………………………...14

Вывод………………………………………………………………...15

Введение

# Актуальность темы.

Актуальность программы обусловлена потребностью человека в логическом развитии и заинтересованности к простым действиям, которые приводят к выигрышныму результату. Данная тема позволяет нам понять основные принципы создания простейших программ на языке программирования JavaScript.

# Объект исследования. Процесс использования языка программирования JavaScript для создания простого приложения.

# Предмет исследования. Разнообразный инструментарий языков JavaScript, HTML и CSS для создания простого приложения.

Постановка задачи

Главной задачей было создание игры «2048»

Основными пунктами реализации приложения стали:

* Поле плиток;
* Плитки с определенными цветами, в зависимотсти от их значения
* Подсчет очков
* Реальзация выигрыша и проигрыша

# Технологии, использованные для написания приложения:

При написании «2048» были использованы стандартные библиотеки и возможности языков программирования JavaScript, HTML.

Описание кода HTML

<!DOCTYPE html>

<html>

<head>

<meta charset='utf8'>

<title>2048.html</title>

<link rel="stylesheet" href="2048.css">

<script src="2048.js"></script>

</head>

<body>

<div class="grid"></div>

<span id="score" class="score">0</span>

</body>

</html>

Описание кода CSS

.grid{

display: flex;

flex-wrap: wrap;

width: 400px;

height: 400px;

background-color: black;

}

.grid div{

width: 100px;

height: 100px;

background-color: #d6d5d4;

font-size: 50px;

text-align: center;

color: #d6d5d4;

}

Описание кода JavaScript

window.addEventListener('load',main,false)

function main(){

const gridDisplay = document.querySelector('.grid')

var squares = []

var dscore = document.getElementById('score');

var score = 0

function createPlace(){

for (var i=0;i<16;i++){

square = document.createElement('div')

square.innerHTML = 0

gridDisplay.appendChild(square)

squares.push(square)

}

create();

create();

}

createPlace();

function color(){

for(var i = 0; i<16;i++){

if(squares[i].innerHTML == 4){

squares[i].style.backgroundColor = '#8442f5'

}

else if(squares[i].innerHTML == 2){

squares[i].style.backgroundColor = '#4287f5'

}

else if(squares[i].innerHTML == 8){

squares[i].style.backgroundColor = '#f542f5'

}else if(squares[i].innerHTML == 16){

squares[i].style.backgroundColor = '#f5425d'

}else if(squares[i].innerHTML == 32){

squares[i].style.backgroundColor = '#42bcf5'

}else if(squares[i].innerHTML == 64){

squares[i].style.backgroundColor = '#42f5c8'

}else if(squares[i].innerHTML == 128){

squares[i].style.backgroundColor = '#42f569'

}else if(squares[i].innerHTML == 256){

squares[i].style.backgroundColor = '#84f542'

}else if(squares[i].innerHTML == 512){

squares[i].style.backgroundColor = '#ddf542'

}else if(squares[i].innerHTML == 1024){

squares[i].style.backgroundColor = '#f5b642'

}else if(squares[i].innerHTML == 2048){

squares[i].style.backgroundColor = '#f58742'

}else if(squares[i].innerHTML == 4096){

squares[i].style.backgroundColor = '#fa05e5'

}else if(squares[i].innerHTML == 0){

squares[i].style.backgroundColor = '#d6d5d4'

}

}

}

function create(){

var r = Math.round(Math.random()\*(15-0)+0)

var ch = Math.round(Math.random()\*(100-0)+0)

if(squares[r].innerHTML == 0){

if(ch<90){

squares[r].innerHTML = 2;

}

else{

squares[r].innerHTML = 4;

}

}

else create();

color();

}

function down(){

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

var one = squares[i].innerHTML

var two = squares[i+4].innerHTML

var three = squares[i+8].innerHTML

var four = squares[i+12].innerHTML

var cloumns = [parseInt(one),parseInt(two),parseInt(three),parseInt(four)]

var fCol = cloumns.filter(num => num)

var mis = 4-fCol.length;

var zeros = Array(mis).fill(0)

var newCol = zeros.concat(fCol)

squares[i].innerHTML = newCol[0]

squares[i+4].innerHTML = newCol[1]

squares[i+8].innerHTML = newCol[2]

squares[i+12].innerHTML = newCol[3]

}

}

function up(){

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

var one = squares[i].innerHTML

var two = squares[i+4].innerHTML

var three = squares[i+8].innerHTML

var four = squares[i+12].innerHTML

var cloumns = [parseInt(one),parseInt(two),parseInt(three),parseInt(four)]

var fCol = cloumns.filter(num => num)

var mis = 4-fCol.length;

var zeros = Array(mis).fill(0)

var newCol = fCol.concat(zeros)

squares[i].innerHTML = newCol[0]

squares[i+4].innerHTML = newCol[1]

squares[i+8].innerHTML = newCol[2]

squares[i+12].innerHTML = newCol[3]

}

}

function right(){

for(var i =0;i<16;i++){

if(i%4 == 0){

var f = squares[i].innerHTML

var s = squares[i+1].innerHTML

var t = squares[i+2].innerHTML

var ff = squares[i+3].innerHTML

var mas = [parseInt(f),parseInt(s),parseInt(t),parseInt(ff)]

var fm = mas.filter(num => num)

var mis = 4-fm.length

var z = Array(mis).fill(0)

var nz = z.concat(fm)

squares[i].innerHTML = nz[0]

squares[i+1].innerHTML = nz[1]

squares[i+2].innerHTML = nz[2]

squares[i+3].innerHTML = nz[3]

}

}

}

function left(){

for(var i =0;i<16;i++){

if(i%4 == 0){

var f = squares[i].innerHTML

var s = squares[i+1].innerHTML

var t = squares[i+2].innerHTML

var ff = squares[i+3].innerHTML

var mas = [parseInt(f),parseInt(s),parseInt(t),parseInt(ff)]

var fm = mas.filter(num => num)

var mis = 4-fm.length

var z = Array(mis).fill(0)

var nz = fm.concat(z)

squares[i].innerHTML = nz[0]

squares[i+1].innerHTML = nz[1]

squares[i+2].innerHTML = nz[2]

squares[i+3].innerHTML = nz[3]

}

}

}

function comb(){

for(var i =0;i<15;i++){

if(squares[i].innerHTML === squares[i+1].innerHTML){

var combined = parseInt(squares[i].innerHTML)+parseInt(squares[i+1].innerHTML)

squares[i].innerHTML = combined

squares[i+1].innerHTML = 0

score+=parseInt(squares[i].innerHTML)

}

if(squares[i].innerHTML == 2048){

alert("Win")

}

}

dscore.innerHTML = score

}

function combCol(){

for(var i =0;i<12;i++){

if(squares[i].innerHTML === squares[i+4].innerHTML){

var combined = parseInt(squares[i].innerHTML)+parseInt(squares[i+4].innerHTML)

squares[i].innerHTML = combined

squares[i+4].innerHTML = 0

score+=parseInt(squares[i].innerHTML)

color();

}

if(squares[i].innerHTML == 2048){

alert("You win")

}

}

dscore.innerHTML = score

}

window.addEventListener('keydown',chekKeyPress,false);

function chekKeyPress(key){

if(key.keyCode === 39){

keyRight();

color();

lose();

}else if(key.keyCode === 37){

keyLeft();

color();

lose();

}else if(key.keyCode === 38){

keyUp();

color();

lose();

}else if(key.keyCode === 40){

keyDown();

color();

lose();

}

}

function keyRight(){

right();

comb();

right();

create();

}

function keyLeft(){

left();

comb();

left();

create();

}

function keyUp(){

up();

combCol();

up();

create();

}

function keyDown(){

down();

combCol();

down();

create();

}

function lose(){

var k =0;

for(var i = 0; i<16;i++){

if(squares[i].innerHTML == 0){

k++;

}

}

for(var i =0;i<15;i++){

if(squares[i].innerHTML == squares[i+1].innerHTML){

//console.log('k')

k++

}

}

for(var j = 0;j<12;j++){

if(squares[j].innerHTML == squares[j+4].innerHTML){

k++

}

}

if(k===0){

//chekOver();

alert("you lose")

location.reload();

}

}

}

Результаты

В результате получаем рабочую версию игры 2048, которая удовлетворяет поставленным задачам.

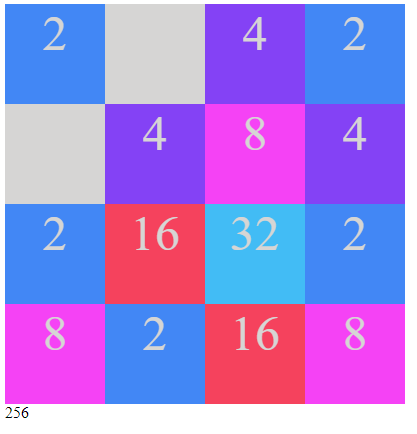


Рис.1 Внешний вид «2048»

Вывод

Подводя итоги курсовой работы, мы можем сказать, что поставленная задача выполнена, а также достигнуты сопутствующие ей цели. В ходе выполнения данного проекта был получен ценный опыт работы с необходимым инструментарием языка JavaScript.