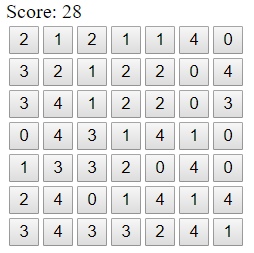
Гра «Кульки»



<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Document</title>

<script>

class Bublle{

constructor(rowCount, colCount,colorsCount){

this.rowCount=rowCount

this.colCount=colCount

this.colorsCount=colorsCount

this.score=0

this.gameTable=this.createStartTable()

this.isFirstClick=true

}

getRandomColor(){

return Math.floor(Math.random()\*this.colorsCount)

}

createStartTable(){

let gameTable=[]

for (let i = 0; i < this.rowCount; i++) {

let row=[]

for (let j = 0; j < this.colCount; j++) {

row.push(this.getRandomColor())

}

gameTable.push(row)

}

return gameTable

}

getVerticalChain(i,j){

let start=i

while(start>0 && this.gameTable[start-1][j]==this.gameTable[start][j])

start--

//-------------

let stop=i

while(stop<this.rowCount-1 && this.gameTable[stop+1][j]==this.gameTable[stop][j])

stop++

return [start, stop]

}

getHorizontalChain(i,j){

let start=j

while(start>0 && this.gameTable[i][start-1]==this.gameTable[i][start])

start--

//-------------

let stop=j

while(stop<this.colCount-1 && this.gameTable[i][stop+1]==this.gameTable[i][stop])

stop++

return [start, stop]

}

burnVerticalChain(start,stop,j){

let currentInsPos=stop

let currentReadPos=start-1

while(currentReadPos>=0)

{

this.gameTable[currentInsPos][j]=this.gameTable[currentReadPos][j]

currentInsPos--

currentReadPos--

}

for (let i = currentInsPos; i >=0; i--) {

this.gameTable[currentInsPos][j]=this.getRandomColor()

}

this.score+=stop-start+1

this.scoreSpan.innerText=this.score

this.showTable()

}

showTable=()=>{

this.divTable.innerHTML=''

this.divTable.appendChild(this.getHTMLTable())

}

burnHorizontalChain(i,start,stop){

for (let j = start; j <=stop; j++){

this.burnVerticalChain(i,i,j)

}

}

burn=()=>{

for (let i = this.rowCount-1; i >=0; i--) {

for (let j = 0; j < this.colCount; j++) {

const [startChain,stopChain]=this.getHorizontalChain(i,j)

if(stopChain-startChain>=2)

{

this.burnHorizontalChain(i,startChain,stopChain)

i++

break

}

const [startChainv,stopChainv]=this.getVerticalChain(i,j)

if(stopChainv-startChainv>=2)

{

this.burnVerticalChain(startChainv,stopChainv,j)

i=stopChainv+1

break

}

}

}

}

onClick=(e)=>{

let btn=e.target

if(this.isFirstClick)

{

this.firstRow=btn.getAttribute('row')

this.firstCol=btn.getAttribute('col')

this.isFirstClick=false

}

else{

let secondRow=btn.getAttribute('row')

let secondCol=btn.getAttribute('col')

if(this.firstRow==secondRow && Math.abs(this.firstCol-secondCol)==1 ||

this.firstCol==secondCol && Math.abs(this.firstRow-secondRow)==1)

{

let temp =this.gameTable[this.firstRow][this.firstCol]

this.gameTable[this.firstRow][this.firstCol]=this.gameTable[secondRow][secondCol]

this.gameTable[secondRow][secondCol]=temp

this.burn()

this.showTable()

}

this.isFirstClick=true

}

}

getHTMLTable(){

let table=document.createElement('table')

for (let i = 0; i < this.rowCount; i++) {

let tr= document.createElement('tr')

for (let j = 0; j < this.colCount; j++) {

let td=document.createElement('td')

let btn=document.createElement('button')

btn.onclick=this.onClick

btn.innerText=this.gameTable[i][j]

btn.setAttribute('row',i)

btn.setAttribute('col',j)

td.appendChild(btn)

tr.appendChild(td)

}

table.appendChild(tr)

}

return table

}

render(containerID){

let container = document.getElementById(containerID)

let div=document.createElement('div')

div.innerText='Score: '

this.scoreSpan=document.createElement('span')

this.scoreSpan.innerText=this.score

div.appendChild(this.scoreSpan)

container.appendChild(div)

this.divTable=document.createElement('div')

this.divTable.appendChild(this.getHTMLTable())

container.appendChild(this.divTable)

setTimeout(this.burn,1000)

}

}

//------------------

window.onload=function(){

let game= new Bublle(7,7,5);

game.render('game')

}

</script>

</head>

<body>

<div id="game"></div>

</body>

</html>