var tetris={

OFFSET:15,//格子区域相对于背景图片的偏移

CSIZE:26,//每个格子的大小

shape:null,//保存主角图形对象

nextShape:null,//保存备胎图形

pg:null,//保存游戏主界面div

interval:800,//下落速度（时间间隔）

timer:null,//保存定时器序号

RN:20,//总行数

CN:10,//总列数

wall:[],//方块墙: 保存所有停止下落的方块的数组

score:0,//当前游戏得分

lines:0,//删除的总行数

SCORES:[0,10,30,70,150],

//0 1 2 3 4

state:1, //当前游戏的状态

GAMEOVER:0, //结束

RUNNING:1, //运行中

PAUSE:2, //暂停

level:1, //保存游戏的等级

start:function(){

this.state=this.RUNNING; //重置游戏状态为RUNNING

this.score=0;

this.lines=0;

this.wall=[];//wall置为[]

//r从0开始，到<RN结束

for(var r=0;r<this.RN;r++){

//将wall中r行赋值为一个CN个空元素的新数组

this.wall[r]=new Array(this.CN);

}

//找到class为playground的div，保存在pg

this.pg=document.querySelector(".playground");

//实例化一个T图形，保存在shape中

this.shape=this.randomShape();

this.nextShape=this.randomShape();

this.paint();//重绘一切

//启动周期性定时器,序号保存在timer中

this.timer=setInterval(

//任务: moveDown, 间隔:interval

this.moveDown.bind(this),

this.interval

);

//为document绑定按键事件

document.onkeydown=function(e){

switch(e.keyCode){//判断键盘号

//如果是37: 就调用游戏对象的moveLeft

case 37: this.state==this.RUNNING && this.moveLeft(); break;

//如果是39: 就调用游戏对象的moveRight

case 39: this.state==this.RUNNING && this.moveRight(); break;

//如果是40: 就调用游戏对象的moveDown

case 40: this.state==this.RUNNING && this.moveDown(); break;

//如果是32: 就一落到底

case 32: this.state==this.RUNNING && this.hardDrop(); break;

//如果是38: 就顺时针转一次

case 38: this.state==this.RUNNING && this.rotateR(); break;

//如果是90: 就逆时针转一次

case 90: this.state==this.RUNNING && this.rotateL(); break;

//如果是83->S，重启游戏

case 83: this.state==this.GAMEOVER && this.start(); break;

//如果是80->P，暂停游戏

case 80: this.state==this.RUNNING && this.pause(); break;

//如果是67->C，继续游戏

case 67: this.state==this.PAUSE && this.myContinue(); break;

}

}.bind(this);

},

myContinue:function(){

this.state=this.RUNNING;

this.paint(); //重新绘制一切

},

pause:function(){

this.state=this.PAUSE;

this.paint();

},

canRotate:function(){

//遍历shape中每个cell

for(var i=0;i<this.shape.cells.length;i++){

var cell=this.shape.cells[i];

//如果cell的r<0或>=RN

//或cell的c<0或>=CN

//或wall中和cell相同位置有格

if(cell.r<0||cell.r>=this.RN

||cell.c<0||cell.c>=this.CN

||this.wall[cell.r][cell.c]){

return false;//就返回false

}

}//(遍历结束)

return true;//返回true

},

rotateR:function(){//顺时针转一次

//调用shape的rotateR

this.shape.rotateR();

//如果不能旋转，再左转回来

if(!this.canRotate()){

this.shape.rotateL();

}

this.paint();//重绘一切

},

rotateL:function(){//逆时针转一次

//调用shape的rotateL

this.shape.rotateL();

//如果不能旋转，再右转回来

if(!this.canRotate()){

this.shape.rotateR();

}

this.paint();//重绘一切

},

randomShape:function(){

//在0~2之间生成一个随机整数r

var r=parseInt(Math.random()\*3);

switch(r){//判断r:

//如果是0: 返回新的O图形

case 0: return new O();

//如果是1: 返回新的I图形

case 1: return new I();

//如果是2: 返回新的T图形

case 2: return new T();

}

},

hardDrop:function(){

//循环: 只要可以下落

while(this.canDown()){

this.moveDown();//调用moveDown

}

},

canLeft:function(){

//遍历shape中每个cell

for(var i=0;i<this.shape.cells.length;i++){

var cell=this.shape.cells[i];

//如果当前cell的c是0

if(cell.c==0

//或wall中cell左侧有格

||this.wall[cell.r][cell.c-1]){

return false;//就返回false

}

}//(遍历结束)

return true;//返回true

},

moveLeft:function(){

if(this.canLeft()){//如果可以左移

//就调用shape的moveLeft

this.shape.moveLeft();

}

this.paint();

},

canRight:function(){

//遍历shape中每个cell

for(var i=0;i<this.shape.cells.length;i++){

var cell=this.shape.cells[i];

//如果当前cell的c是CN-1,

if(cell.c==this.CN-1

//或wall中cell右侧有格

||this.wall[cell.r][cell.c+1]){

return false;//就返回false

}

}//(遍历结束)

return true;//返回true

},

moveRight:function(){

if(this.canRight()){//如果可以右移

//就调用shape的moveRight

this.shape.moveRight();

}

this.paint();

},

canDown:function(){//判断是否可下落

//遍历shape中每个cell

for(var i=0;i<this.shape.cells.length;i++){

var cell=this.shape.cells[i];

//如果当前cell的r等于RN-1

if(cell.r==this.RN-1

||this.wall[cell.r+1][cell.c]){

return false;//返回false

}

}//(遍历结束)

return true;//返回true

},

moveDown:function(){//让主角图形下落一步

//如果游戏状态为RUNNING：执行以下操作：

if(this.state==this.RUNNING){

if(this.canDown()){//如果可以下落

//调用shape的moveDown方法

this.shape.moveDown();

}else{//否则

//让shape中的格子落到墙里

this.landIntoWall();

//判断并删除满格行

var ln=this.deleteRows();

//将ln累加到lines上

this.lines+=ln;

//从SCORES中ln位置获取相应的得分累加到score上

this.score+=this.SCORES[ln];

/\*\*\*\*\*\*\*\*\*\*\*升级部分\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//假设规定每消除10行升一级

//每升一级时间间隔减100ms

var l=parseInt(this.lines/2)+1; //每10行升一级

if(l>this.level){

this.level=l;

if(this.interval>100){

this.interval-=(this.level-1)\*100;

alert("恭喜您升一级！当前等级为："+this.level);

clearInterval(this.timer);

this.timer=setInterval(

this.moveDown.bind(this),

this.interval

);

}

}

//如果游戏没有结束

if(!this.isGameOver()){

//重新实例化一个主角图形，保存在shape中

this.shape=this.nextShape;

this.nextShape=this.randomShape();

}else{//否则：

//停止计时器：

clearInterval(this.timer);

//清空timer:

this.timer=null;

//修改游戏的状态为GAMEOVER

this.state=this.GAMEOVER;

}

}

this.paint();//重绘一切

}

},

paintState:function(){

//创建img

var img=new Image();

//如果游戏的状态为GAMEOVER

if(this.state==this.GAMEOVER){

//设置img的src为‘img/game-over.png’

img.src='img/game-over.png';

}else if(this.state==this.PAUSE){//否则，如果游戏的状态为PAUSE

//设置img的src为'img/pause.png'

img.src='img/pause.png';

}

//将img追加到pg下

this.pg.appendChild(img);

},

isGameOver:function(){

//遍历nextShape中每个cell

for(var i=0;i<this.nextShape.cells.length;i++){

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

var cell=this.nextShape.cells[i];

//如果wall中cell相同位置有格，就返回true

if(this.wall[cell.r][cell.c]){

return true;

}

}//遍历结束，返回false

return false;

},

//遍历并删除所有满格行

deleteRows:function(){

//自底向上遍历wall中每一行，同时声明ln=0

for(var r=this.RN-1,ln=0;r>=0;r--){

//如果wall中r行为空，就直接退出循环

if(this.wall[r].join("")==""){break;}

if(this.isFull(r)){//如果r是满格

this.deleteRow(r);//删除第r行

r++;//r留在原地

ln++;//ln+1

//如果ln是4，就直接退出循环

if(ln==4){break;}

}

}//(遍历结束)

return ln;

},

deleteRow:function(r){//删除第r行

//i从r开始，自底向上遍历wall中每一行

for(var i=r;i>=0;i--){

//将wall中i-1行赋值给wall中第i行

this.wall[i]=this.wall[i-1];

//遍历wall中第i行的每个cell

for(c=0;c<this.CN;c++){

//如果wall中i行c列有格

if(this.wall[i][c]){

//才将当前cell的r+1

this.wall[i][c].r++;

}

}//(遍历结束)

//创建CN个空元素的新数组赋值给wall中i-1行

this.wall[i-1]=new Array(this.CN);

//如果wall中i-2行为空

if(this.wall[i-2].join("")==""){

break;//退出循环

}

}

},

isFull:function(r){//判断r行是否满格

//将wall中r行拍照后，验证是否包含^,或,,或,$，转为!，返回结果

return !/^,|,,|,$/.test(String(this.wall[r]));

},

//将主角图形的格子落到墙里

landIntoWall:function(){

//遍历shape中每个cell

for(var i=0;i<this.shape.cells.length;i++){

var cell=this.shape.cells[i];

//将wall中和当前cell相同r,c位置的元素赋值为cell

this.wall[cell.r][cell.c]=cell;

}

},

paint:function(){//重绘一切！

//删除pg下所有img

this.pg.innerHTML=this.pg.innerHTML.replace(/<img\s+[^>]\*>/g,"");

this.paintShape();//重绘主角图形

this.paintWall();

this.paintScore();

this.paintNext();

this.paintState();

},

paintNext:function(){//重绘备胎图形

//创建frag

var frag= document.createDocumentFragment();

//遍历nextShape中每个cell

for(var i=0;i<this.nextShape.cells.length;i++){

//将当前cell保存在变量cell中

var cell=this.nextShape.cells[i];

var img=new Image();//创建新img

//设置img的src为cell的src

img.src=cell.src;

//设置img的top为(cell的r+1)\*CSIZE+OFFSET

img.style.top=(cell.r+1)\*this.CSIZE+this.OFFSET+"px";

//设置img的left为(cell的c+11)\*CSIZE+OFFSET

img.style.left=(cell.c+11)\*this.CSIZE+this.OFFSET+"px";

//将img追加到frag中

frag.appendChild(img);

}//(遍历结束)

//将frag追加到pg中

this.pg.appendChild(frag);

},

paintScore:function(){

//在pg下找第1个p元素下的span，设置其内容为score

this.pg.querySelector("p:first-child>span").innerHTML=this.score;

//在pg下找第2个p元素下的span，设置其内容为lines

this.pg.querySelector("p:nth-child(2)>span").innerHTML=this.lines;

//在pg下找第3个p元素下的span，设置其内容为level

this.pg.querySelector("p:nth-child(3)>span").innerHTML=this.level;

},

paintWall:function(){

//创建文档片段frag:

var frag=document.createDocumentFragment();

//r从RN-1开始，到>=0结束,每次-1

for(var r=this.RN-1;r>=0;r--){

//如果wall中r行无缝拼接后等于""

if(this.wall[r].join("")==""){

break;//退出循环

}else{//否则

//c从0开始，到<CN结束

for(var c=0;c<this.CN;c++){

//如果当前格有效,才绘制当前格

this.wall[r][c]&&

this.paintCell(this.wall[r][c],frag);

}

}

}//(遍历结束)

//将frag追加到pg中

this.pg.appendChild(frag);

},

paintCell:function(cell,frag){

//创建一个img元素

var img=new Image();

//设置img的src为cell的src

img.src=cell.src;

//设置img的top: r\*CSIZE+OFFSET

img.style.top=cell.r\*this.CSIZE+this.OFFSET+"px";

//设置img的left: c\*CSIZE+OFFSET

img.style.left=cell.c\*this.CSIZE+this.OFFSET+"px";

//将img追加到frag中

frag.appendChild(img);

},

paintShape:function(){//负责绘制主角图形

//创建文档片段frag

var frag=document.createDocumentFragment();

//遍历shape中cells中的每个图形

for(var i=0;i<this.shape.cells.length;i++){

//将当前图形临时存储在变量cell中

var cell=this.shape.cells[i];

this.paintCell(cell,frag);//绘制当前格

}//(遍历结束)

//将frag追加到pg中

this.pg.appendChild(frag);

},

}

tetris.start();