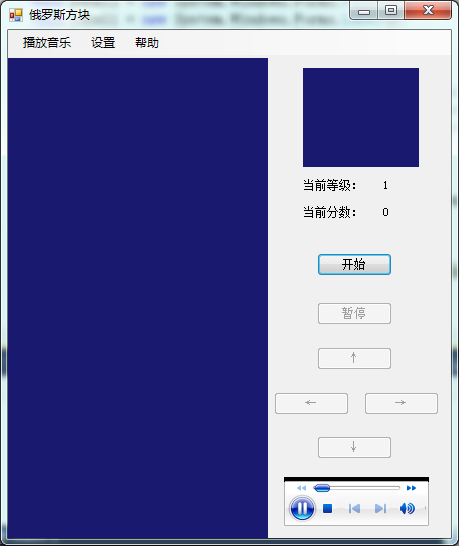
**俄罗斯方块**

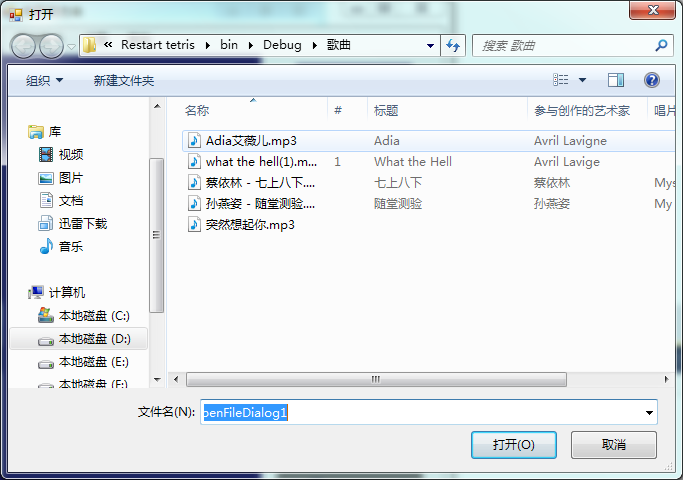
**一：介绍**

借鉴了以前前辈的一些做法，写了一个简单的俄罗斯方块的游戏，初始化界面如下：

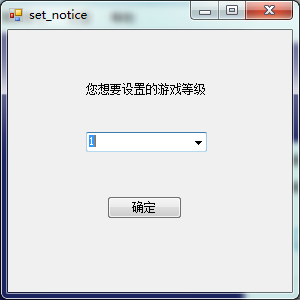


主界面的右下角是一个WindowsMediaPlayer控件，用于边游戏边听歌。

“播放音乐”按钮按下的界面：

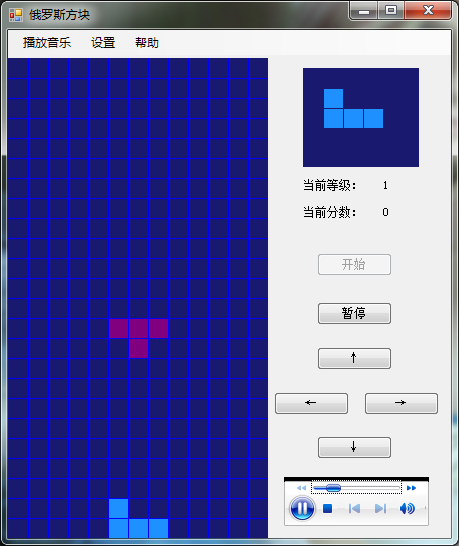


“设置”按钮按下的界面：

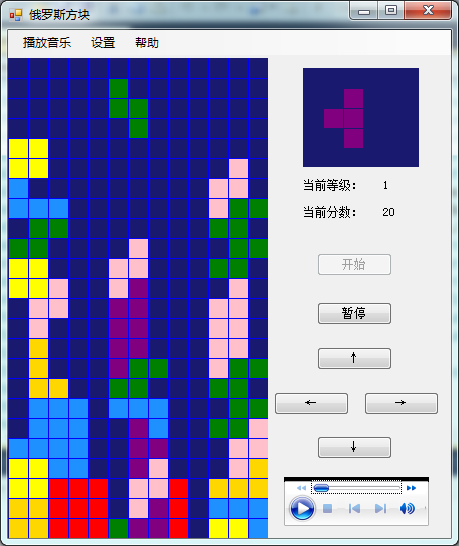


“帮助”按钮按下没有界面，直接弹出帮助文档。

点击开始后的运行界面：



可以通过向上键来变换砖块，向下键快速降落，左键向左移，右键向右移（也可以用W、S、A、D分别代替上下左右键），来调整砖块的位置及形状，从而获得分数。



游戏结束的界面：



**二：代码**

我用的是VS2010编的，但是10版有个bug，就是把含有颜色的汉字贴到word上时会出现乱码，每個中文字后会连着一到两个多余的符号，不过仔细看能看个大概明白，大家就将就下，不好意思。

下面我直接贴上源代码。

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Collections;

using System.Runtime.InteropServices;

namespace Restart\_tetris

{

public partial class MainForm : Form

{

//存ä?储ä¡é所¨´有®D的Ì?砖Á?块¨¦的Ì?样¨´式º?及¡ã颜?色¦?

ArrayList bricksarraylist=new ArrayList();

Random rd = new Random();

int index;

//存ä?储ä¡é当Ì¡À前¡ã的Ì?panel1需¨¨要°a画-的Ì?砖Á?块¨¦

standardbrick currentstandardbrick;

//存ä?储ä¡é当Ì¡À前¡ã的Ì?panel2需¨¨要°a画-的Ì?砖Á?块¨¦

standardbrick nextstandardbrick;

Graphics panel1g ;

Graphics panel2g;

//存ä?储ä¡é新?的Ì?砖Á?块¨¦在¨²panel1上¦?画-出?来¤¡ä之?前¡ã，ê?panel1上¦?已°?有®D的Ì?图ª?案ã?

Color[,] preservecolor = new Color[13, 24];

//xx，ê?yy调Ì¡Â整?砖Á?块¨¦在¨²panel1上¦?的Ì?位?置?

int xx;

int yy;

//存ä?储ä¡é玩ª?家¨°所¨´得Ì?分¤?数ºy

int score;

//需¨¨要°a检¨¬查¨¦的Ì?位?置?的Ì?坐Á?标À¨º（ê¡§因°¨°为a在¨²准Á?备À?执¡ä行D旋y转Áa或¨°向¨°下?等Ì¨¨操¨´作Á¡Â时º¡À要°a检¨¬查¨¦所¨´需¨¨的Ì?位?置?是º?否¤?为a空?）ê?

Point[] pout;

//画-布?上¦?需¨¨要°a检¨¬查¨¦的Ì?位?置?的Ì?实º¦Ì际¨º的Ì?颜?色¦?

ArrayList nextcolor=new ArrayList();

IntPtr hdc;

[DllImport("gdi32.dll")]//windows API

private static extern int GetPixel(IntPtr hDc, int x, int y);

public MainForm()

{

InitializeComponent();

bricksarraylist.Add(new concretebrick("0000001100001000010000000", Color.Gold));

bricksarraylist.Add(new concretebrick("0010000100001000010000000", Color.Red));

bricksarraylist.Add(new concretebrick("0000000100011000100000000", Color.Pink));

bricksarraylist.Add(new concretebrick("0000000100011000010000000", Color.Purple));

bricksarraylist.Add(new concretebrick("0000000110011000000000000", Color.Green));

bricksarraylist.Add(new concretebrick("0000000110001000010000000", Color.DodgerBlue));

bricksarraylist.Add(new concretebrick("0000000110001100000000000", Color.Yellow));

panel1g = panel1.CreateGraphics();

panel2g = panel2.CreateGraphics();

//hdc = panel1g.GetHdc();

score=0;

xx = 6;

yy = 2;

for (int i = 0; i < 13; i++)

for (int j = 0; j < 24; j++)

//初?始º?化¡¥为apanel1的Ì?backcolor

preservecolor[i, j] = Color.MidnightBlue;

}

private void 播£¤放¤?音°?乐¤?ToolStripMenuItem\_Click(object sender, EventArgs e)

{

//添¬¨ª加¨®了¢?一°?个?openfiledialog控?件t，ê?名?字Á?叫Dopenmusic

openmusic.CheckFileExists = true;

openmusic.CheckPathExists = true;

openmusic.InitialDirectory=Application.StartupPath+"\\歌¨¨曲¨²";

if (openmusic.ShowDialog() == DialogResult.OK)

this.axWindowsMediaPlayer1.URL = openmusic.FileName;

}

private void 设¦¨¨置?ToolStripMenuItem\_Click(object sender, EventArgs e)

{

//setnotice是º?另¢¨ª一°?个?form窗ä¡ã体¬?，ê?用®?来¤¡ä设¦¨¨置?玩ª?家¨°需¨¨要°a的Ì?等Ì¨¨级?

set\_notice setnotice = new set\_notice();

setnotice.Show();

setnotice.Owner = this;

}

private void 帮ã?助¨²ToolStripMenuItem\_Click(object sender, EventArgs e)

{

//helpdocument helpd = new helpdocument();

//helpd.Show();

//打ä¨°开a帮ã?助¨²文?档Ì¦Ì

System.Diagnostics.Process.Start(Application.StartupPath + "\\GPIB连¢?接¨®仪°?器¡Â.pdf");

}

//“¡ã暂Y停ª¡ê”¡À或¨°“¡ã继¨¬续?”¡À按ã¡ä钮£¤的Ì?触ä£¤发¤¡é事º?件t

private void pause\_Click(object sender, EventArgs e)

{

if (((Button)sender).Text == "暂Y停ª¡ê")

{

this.pause.Text = "继¨¬续?";

this.timer1.Stop();

}

else

{

this.pause.Text = "暂Y停ª¡ê";

this.timer1.Start();

}

}

//“¡ã开a始º?”¡À按ã¡ä钮£¤的Ì?触ä£¤发¤¡é事º?件t

private void start\_Click(object sender, EventArgs e)

{

//label2.text是º?setnotice中D表À¨ª示º?等Ì¨¨级?的Ì?

switch (this.label2.Text)

{

case "1":

this.timer1.Interval = 1000;

break;

case "2":

this.timer1.Interval = 800;

break;

case "3":

this.timer1.Interval = 600;

break;

case "4":

this.timer1.Interval = 400;

break;

case "5":

this.timer1.Interval = 200;

break;

}

this.transformation.Enabled = true;

this.left.Enabled = true;

this.right.Enabled = true;

this.down.Enabled = true;

this.start.Enabled = false;

this.pause.Enabled = true;

//产¨²生¦¨²一°?个?砖Á?块¨¦，ê?用®?于®¨²在¨²主¡Â画-布?上¦?显?示º?

currentstandardbrick = createbrick();

//产¨²生¦¨²下?一°?个?砖Á?块¨¦，ê?显?示º?在¨²小?画-布?上¦?，ê?用®?于®¨²提¬¨¢示º?

nextstandardbrick = createbrick();

//在¨²主¡Â画-布?上¦?显?示º?currentstandardbrick

paint(currentstandardbrick.brickpointsduplicate, currentstandardbrick.brickcolorduplicate,panel1g,preservecolor);

//在¨²小?上¦?画-布?上¦?显?示º?下?一°?个?砖Á?块¨¦

paint2(nextstandardbrick.brickpointsduplicate, nextstandardbrick.brickcolorduplicate, panel2g);

this.timer1.Start();

}

private standardbrick createbrick()

{

//每?当Ì¡À产¨²生¦¨²新?砖Á?块¨¦时º¡À，ê?都?说¦Ì明¡Âcurrentstandardbrick已°?经-更¨¹新?，ê?所¨´以°?画-到Ì?画-布?上¦?应®|时º¡À应®|调Ì¡Â整?位?置?

xx = 6;

yy = 2;

//随?机¨²从ä¨®bricksarraylist中D取¨?出?一°?个?砖Á?块¨¦

index = rd.Next(bricksarraylist.Count);

concretebrick tempconcretebrick=(concretebrick)bricksarraylist[index];

string tempcode = tempconcretebrick.cbcodeduplicate;

Color tempcolor = tempconcretebrick.cbcolorduplicate;

//下?面?主¡Â要°a是º?将?那?些?01的Ì?代ä¨²码?转Áa换?成¨¦实º¦Ì际¨º的Ì?坐Á?标À¨º，ê?只?有®D有®D坐Á?标À¨º才?可¨¦以°?在¨²panel1上¦?绘?出?

List<Point> list=new List<Point>();

for (int i = 0; i < tempcode.Length;i++ )

{

if (tempcode[i] == '1')

{

Point p = new Point(i % 5, i / 5);

//使º1砖Á?块¨¦的Ì?坐Á?标À¨º以°?坐Á?标À¨º原-点Ì?为a中D心?，ê?便À?于®¨²后¨®面?旋y转Áa等Ì¨¨操¨´作Á¡Â的Ì?坐Á?标À¨º变À?换?

p.Offset(-2, -2);

list.Add(p);

}

}

standardbrick tempstandardbrick=new standardbrick(list.ToArray(),tempcolor);

//随?机¨²选?择?是º?否¤?旋y转Áa一°?下?

if (rd.Next(2) == 1)

clockwise(tempstandardbrick);

return tempstandardbrick;

}

//在¨²panel1上¦?画-出?格?子Á¨®，ê?便À?于®¨²游®?戏¡¤时º¡À调Ì¡Â整?砖Á?块¨¦的Ì?位?置?

private void paintgrid(Graphics g)

{

try

{

lock (g)

{

using (Pen p = new Pen(Color.Blue, 1))

{

for (int i = 1; i < 13; i++)

g.DrawLine(p, i \* 20, 0, i \* 20, 480);

for (int j=1; j< 24;j++ )

g.DrawLine(p, 0, j \* 20, 260, j \* 20);

}

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

//在¨²panel1上¦?画-出?currentstandardbrick、¡é

private void paint(Point[] ps,Color c,Graphics g,Color[,] preservec)

{

foreach (Point p in ps)

{

lock (g)

{

try

{

g.FillRectangle(new SolidBrush(c), pointstorects(p));

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

}

//画-出?网ª?格?

paintgrid(g);

//画-出?以°?前¡ã的Ì?砖Á?块¨¦

for (int i = 0; i < 13; i++)

for (int j = 0; j < 24;j++ )

if(preservecolor[i,j]!=Color.MidnightBlue)

panel1g.FillRectangle(new SolidBrush(preservec[i, j]), i \* 20 + 1, j \* 20 + 1, 20 - 1, 20 - 1);

}

//将?坐Á?标À¨º画-成¨¦格?子Á¨®

private Rectangle pointstorects(Point p)

{

Rectangle rect = new Rectangle((p.X+xx) \* 20+1, (p.Y+yy) \* 20+1, 20 - 1, 20 - 1);

return rect;

}

//在¨²panel2上¦?画-出?nextstandardbrick

private void paint2(Point[] ps, Color c, Graphics g)

{

g.Clear(Color.MidnightBlue);

foreach (Point p in ps)

{

lock (g)

{

try

{

g.FillRectangle(new SolidBrush(c), pointstorects2(p));

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

}

}

//将?坐Á?标À¨º画-成¨¦格?子Á¨®

private Rectangle pointstorects2(Point p)

{

Rectangle rect = new Rectangle((p.X +2) \* 20 + 1, (p.Y + yy) \* 20 + 1, 20 - 1, 20 - 1);

return rect;

}

//panel1的Ì?重?绘?函¡¥数ºy

private void panel1\_Paint(object sender, PaintEventArgs e)

{

this.SuspendLayout();

if (currentstandardbrick != null)

paint(currentstandardbrick.brickpointsduplicate, currentstandardbrick.brickcolorduplicate, e.Graphics, preservecolor);

this.ResumeLayout();

}

//控?制?砖Á?块¨¦的Ì?下?降¦Ì

private void timer1\_Tick(object sender, EventArgs e)

{

//判D断?是º?否¤?能¨¹下?降¦Ì

if (moveorstop(currentstandardbrick, 3))

{

yy++;

this.panel1.Refresh();

}

//不?能¨¹

else

{

//保À¡ê存ä?的Ì?panel上¦?已°?画-出?的Ì?砖Á?块¨¦，ê?即¡ä更¨¹新?preservecolor

preservebricks();

//更¨¹新?是º?否¤?有®D满¨²格?，ê?若¨?有®D则¨°需¨¨要°a更¨¹新?preservecolor

checkandout();

//更¨¹新?当Ì¡À前¡ã砖Á?块¨¦

currentstandardbrick = nextstandardbrick;

//检¨¬测a是º?否¤?可¨¦以°?显?示º?当Ì¡À前¡ã砖Á?块¨¦，ê?若¨?显?示º?当Ì¡À前¡ã砖Á?块¨¦需¨¨要°a的Ì?位?置?都?已°?填¬?满¨²则¨°不?能¨¹显?示º?

int showornot=0;

for (int i = 0; i < currentstandardbrick.brickpointsduplicate.Length;i++ )

if (preservecolor[currentstandardbrick.brickpointsduplicate[i].X + 6, currentstandardbrick.brickpointsduplicate[i].Y+2]==Color.MidnightBlue)

showornot++;

if (showornot==currentstandardbrick.brickpointsduplicate.Length)

{

//若¨?能¨¹显?示º?

nextstandardbrick = createbrick();

paint(currentstandardbrick.brickpointsduplicate, currentstandardbrick.brickcolorduplicate, panel1g,preservecolor);

paint2(nextstandardbrick.brickpointsduplicate, nextstandardbrick.brickcolorduplicate, panel2g);

}

else

{

//若¨?不?能¨¹，ê?游®?戏¡¤结¨¢束º?

this.timer1.Stop();

panel2g.Clear(Color.MidnightBlue);

xx = 6; yy = 2;

paint(currentstandardbrick.brickpointsduplicate, currentstandardbrick.brickcolorduplicate, panel1g,preservecolor);

MessageBox.Show("game over!");

panel1g.DrawString("GAME OVER!", new Font("Arial BLACK",20f), new SolidBrush(Color.DarkTurquoise), new RectangleF(30, 140, 300, 100));

//游®?戏¡¤结¨¢束º?时º¡À各¡Â个?控?件t都?不?能¨¹用®?

stopfunction();

currentstandardbrick = null;

nextstandardbrick = null;

//panel1.Refresh();//这a里¤?刷¡é新?会¨¢把ã?刚?写¡ä上¦?去¨£¤的Ì?game over也°2刷¡é掉Ì?（ê¡§可¨¦能¨¹refresh是º?要°a再¨´重?绘?一°?次ä?控?件t表À¨ª面?，ê?即¡ä调Ì¡Â用®?paint函¡¥数ºy但Ì?这a后¨®一°?次ä?的Ì?重?绘?没?有®D包ã¨¹含?game over重?绘?在¨²内¨²）ê?

//panel2.Refresh();

}

}

}

//顺3时º¡À针?旋y转Áa改?变À?currentstandardbrick的Ì?坐Á?标À¨º

private void clockwise(standardbrick sb)

{

if(sb!=null)

{

int temp2;

for (int i = 0; i <sb.brickpointsduplicate.Length; i++)

{

temp2 = sb.brickpointsduplicate[i].Y;

sb.brickpointsduplicate[i].Y = -sb.brickpointsduplicate[i].X;

sb.brickpointsduplicate[i].X = temp2;

}

}

else MessageBox.Show("程¨¬序¨°出?现?异°¨¬常¡ê");

}

//左Á¨®移°?函¡¥数ºy

private void leftfunction(standardbrick sb)

{

if (sb != null)

{

int temp2 = sb.brickpointsduplicate[0].X;

for (int i = 1; i < sb.brickpointsduplicate.Length; i++)

{

if (temp2> sb.brickpointsduplicate[i].X)

temp2 = sb.brickpointsduplicate[i].X;

}

if (temp2 +xx>0)

xx--;

}

else MessageBox.Show("程¨¬序¨°出?现?异°¨¬常¡ê");

}

//右®¨°移°?函¡¥数ºy

private void rightfunction(standardbrick sb)

{

if (sb != null)

{

int temp2 = sb.brickpointsduplicate[0].X;

for (int i = 1; i < sb.brickpointsduplicate.Length; i++)

{

if (temp2<sb.brickpointsduplicate[i].X)

temp2 = sb.brickpointsduplicate[i].X;

}

if (temp2 + xx<12)

xx++;

}

else MessageBox.Show("程¨¬序¨°出?现?异°¨¬常¡ê");

}

//检¨¬查¨¦是º?否¤?有®D满¨²格?，ê?若¨?有®D则¨°需¨¨要°a更¨¹新?preservecolor

private Color[,] checkandout()

{

int position=0;

int length=0;

for (int j=23; j>=0; j--)

{

int sign = 0;

for (int i=12; i>=0; i--)

{

if (preservecolor[i, j] != Color.MidnightBlue)

sign++;

}

if (sign == 13)

{

length++;

if (position == 0)

position = j;

}

}

for (int j = position; j >=length; j--)

for (int i = 12; i >= 0; i--)

preservecolor[i, j] = preservecolor[i, j - length];

for (int j = length - 1; j >= 0; j--)

for (int i = 12; i >= 0; i--)

preservecolor[i, j] = Color.MidnightBlue;

//有®D满¨²格?时º¡À，ê?增?加¨®分¤?数ºy

score+=10\*length;

this.label4.Text = score.ToString();

//this.Invoke(new setdelegate(setfunction), new object[] { score.ToString() });

return preservecolor;

}

//记?录?当Ì¡À前¡ãpanel1上¦?的Ì?每?个?格?子Á¨®的Ì?颜?色¦?，ê?保À¡ê存ä?在¨²preservecolor内¨²

private Color[,] preservebricks()

{

try

{

hdc = panel1g.GetHdc();

for (int i = 0; i < 13; i++)

for (int j = 0; j < 24; j++)

{

//getpixel返¤¦Ì回?的Ì?是º?颜?色¦?的Ì?B、¡éG、¡éR的Ì?值¦Ì

switch (GetPixel(hdc, i \* 20 + 10, j \* 20 + 10))

{

case 0x00d7ff:

preservecolor[i, j] = Color.Gold;

break;

case 0xff:

preservecolor[i, j] = Color.Red;

break;

case 0xcbc0ff:

preservecolor[i, j] = Color.Pink;

break;

case 0x800080:

preservecolor[i, j] = Color.Purple;

break;

case 0x08000:

preservecolor[i, j] = Color.Green;

break;

case 0xff901e:

preservecolor[i, j] = Color.DodgerBlue;

break;

case 0xffff:

preservecolor[i, j] = Color.Yellow;

break;

//case 0xd1ce00:

// preservecolor[i, j] = Color.DarkTurquoise;

// break;

case 0x701919:

preservecolor[i, j] = Color.MidnightBlue;

break;

}

}

//每?次ä?用®?完ª¨º需¨¨要°a释º¨ª放¤?

panel1g.ReleaseHdc();

return preservecolor;

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

return null;

}

}

//快¨¬速¨´下?降¦Ì的Ì?函¡¥数ºy

private void downfunction(standardbrick sb)

{

if (sb != null)

{

//若¨?检¨¬测a能¨¹下?降¦Ì则¨°yy一°?直¡À加¨®

while (moveorstop(currentstandardbrick,3))

yy++;

//若¨?不?能¨¹下?降¦Ì

this.panel1.Refresh();

//保À¡ê存ä?当Ì¡À前¡ãpanel1上¦?的Ì?图ª?案ã?,保À¡ê存ä?在¨²preservecolor内¨²

preservebricks();

//检¨¬查¨¦是º?否¤?有®D满¨²格?，ê?若¨?有®D需¨¨要°a更¨¹新?preservecolor

checkandout();

//更¨¹新?当Ì¡À前¡ã砖Á?块¨¦

currentstandardbrick = nextstandardbrick;

//检¨¬测a是º?否¤?可¨¦以°?显?示º?当Ì¡À前¡ã砖Á?块¨¦，ê?若¨?显?示º?当Ì¡À前¡ã砖Á?块¨¦需¨¨要°a的Ì?位?置?都?已°?填¬?满¨²则¨°不?能¨¹显?示º?

int showornot = 0;

for (int i = 0; i < currentstandardbrick.brickpointsduplicate.Length; i++)

if (preservecolor[currentstandardbrick.brickpointsduplicate[i].X + 6, currentstandardbrick.brickpointsduplicate[i].Y + 2] == Color.MidnightBlue)

showornot++;

if (showornot == currentstandardbrick.brickpointsduplicate.Length)

{

//若¨?可¨¦以°?显?示º?

nextstandardbrick = createbrick();

paint(currentstandardbrick.brickpointsduplicate, currentstandardbrick.brickcolorduplicate, panel1g,preservecolor);

paint2(nextstandardbrick.brickpointsduplicate, nextstandardbrick.brickcolorduplicate, panel2g);

}

else

{

//若¨?不?能¨¹，ê?游®?戏¡¤结¨¢束º?

this.timer1.Stop();

panel2g.Clear(Color.MidnightBlue);

xx = 6; yy = 2;

paint(currentstandardbrick.brickpointsduplicate, currentstandardbrick.brickcolorduplicate, panel1g,preservecolor);

MessageBox.Show("game over!");

panel1g.DrawString("GAME OVER!", new Font("Arial BLACK", 20f), new SolidBrush(Color.DarkTurquoise), new RectangleF(30,140, 300, 100));

//游®?戏¡¤结¨¢束º?时º¡À，ê?所¨´有®D控?件t都?不?能¨¹用®?

stopfunction();

currentstandardbrick = null;

nextstandardbrick = null;

}

}

else MessageBox.Show("game over!");

}

//游®?戏¡¤结¨¢束º?时º¡À，ê?设¦¨¨置?各¡Â个?控?件t不?能¨¹用®?

private void stopfunction()

{

this.start.Enabled=false;

this.pause.Enabled=false;

this.transformation.Enabled=false;

this.left.Enabled=false;

this.right.Enabled=false;

this.down.Enabled=false;

}

//这a里¤?是º?检¨¬测a是º?否¤?可¨¦以°?移°?动¡¥

private bool moveorstop(standardbrick sb,int flag)

{

if (sb != null)

{

//这a里¤?返¤¦Ì回?需¨¨要°a检¨¬查¨¦的Ì?位?置?的Ì?坐Á?标À¨º

Point[] checkpoint = inandout(sb.brickpointsduplicate,flag);

//检¨¬查¨¦是º?否¤?在¨²panel1的Ì?区?域®¨°里¤?

int numberin=0;

for(int i=0;i<checkpoint.Length;i++)

{

if((checkpoint[i].X+xx)\*20>=0&&(checkpoint[i].X+xx)\*20<260&&(checkpoint[i].Y+yy)\*20>=0&&(checkpoint[i].Y+yy)\*20<480)

numberin++;

}

if(numberin==checkpoint.Length)

{

//若¨?都?在¨²panel的Ì?区?域®¨°里¤?，ê?且¨°都?未¡ä覆2盖?，ê?返¤¦Ì回?true

hdc = panel1g.GetHdc();

//nextcolor存ä?储ä¡é这a些?坐Á?标À¨º的Ì?颜?色¦?

nextcolor.Clear();

for (int i = 0; i < checkpoint.Length; i++)

{

nextcolor.Add(GetPixel(hdc, (checkpoint[i].X + xx) \* 20 + 5, (checkpoint[i].Y + yy) \* 20 + 5));

}

panel1g.ReleaseHdc();

int nextcolorcount = 0;

for (int i = 0; i < nextcolor.Count; i++)

{

if ((int)nextcolor[i] ==0x701919)

nextcolorcount++;

}

if (nextcolorcount == nextcolor.Count)

return true;

else return false;

}

else return false;

}

else

{

MessageBox.Show("程¨¬序¨°出?现?异°¨¬常¡ê");

return false;

}

}

//这a里¤?的Ì?功|能¨¹是º?输º?出?需¨¨要°a检¨¬查¨¦的Ì?位?置?的Ì?坐Á?标À¨º

private Point[] inandout(Point[] pin,int tag)

{

List<Point> plisttemp=new List<Point>();

if (tag == 3)//若¨?下?移°?，ê?则¨°需¨¨要°a检¨¬查¨¦的Ì?坐Á?标À¨º

{

for (int i = 0; i < pin.Length; i++)

{

plisttemp.Add(new Point(pin[i].X, pin[i].Y + 1));

}

List<Point> plisttemp1 =new List<Point>(plisttemp);

for (int i = 0; i <pin.Length; i++)

{

for (int j = 0; j < pin.Length; j++)

if (plisttemp[i] == pin[j])

plisttemp1.Remove(plisttemp[i]);

}

pout=plisttemp1.ToArray();

}

if (tag == 2)//若¨?变À?换?，ê?则¨°需¨¨要°a检¨¬查¨¦的Ì?坐Á?标À¨º

{

for (int i = 0; i < pin.Length; i++)

{

plisttemp.Add(new Point(pin[i].Y,-pin[i].X));

}

List<Point> plisttemp1 = new List<Point>(plisttemp);

for (int i = 0; i < pin.Length; i++)

{

for (int j = 0; j < pin.Length; j++)

if (plisttemp[i] == pin[j])

plisttemp1.Remove(plisttemp[i]);

}

pout = plisttemp1.ToArray();

}

if (tag == 1)//若¨?左Á¨®移°?，ê?则¨°需¨¨要°a检¨¬查¨¦的Ì?坐Á?标À¨º

{

for (int i = 0; i < pin.Length; i++)

{

plisttemp.Add(new Point(pin[i].X-1, pin[i].Y ));

}

List<Point> plisttemp1 = new List<Point>(plisttemp);

for (int i = 0; i < pin.Length; i++)

{

for (int j = 0; j < pin.Length; j++)

if (plisttemp[i] == pin[j])

plisttemp1.Remove(plisttemp[i]);

}

pout = plisttemp1.ToArray();

}

if (tag == 0)//若¨?右®¨°移°?，ê?则¨°需¨¨要°a检¨¬查¨¦的Ì?坐Á?标À¨º

{

for (int i = 0; i < pin.Length; i++)

{

plisttemp.Add(new Point(pin[i].X+1, pin[i].Y));

}

List<Point> plisttemp1 = new List<Point>(plisttemp);

for (int i = 0; i < pin.Length; i++)

{

for (int j = 0; j < pin.Length; j++)

if (plisttemp[i] == pin[j])

plisttemp1.Remove(plisttemp[i]);

}

pout = plisttemp1.ToArray();

}

return pout;

}

//↑¨¹键¨¹的Ì?触ä£¤发¤¡é函¡¥数ºy

private void transformation\_Click(object sender, EventArgs e)

{

if (moveorstop(currentstandardbrick, 2))

clockwise(currentstandardbrick);

this.panel1.Refresh();

}

//←?键¨¹的Ì?触ä£¤发¤¡é函¡¥数ºy

private void left\_Click(object sender, EventArgs e)

{

if (moveorstop(currentstandardbrick, 1))

leftfunction(currentstandardbrick);

this.panel1.Refresh();

}

//→¨²键¨¹的Ì?触ä£¤发¤¡é函¡¥数ºy

private void right\_Click(object sender, EventArgs e)

{

if (moveorstop(currentstandardbrick,0))

rightfunction(currentstandardbrick);

this.panel1.Refresh();

}

//↓y键¨¹的Ì?触ä£¤发¤¡é函¡¥数ºy

private void down\_Click(object sender, EventArgs e)

{

downfunction(currentstandardbrick);

}

//用®?W、¡éA、¡éD、¡éS四?个?键¨¹分¤?与®?上¦?、¡é左Á¨®、¡é右®¨°、¡é下?四?个?按ã¡ä钮£¤绑ã¨®定¡§

private void MainForm\_KeyDown(object sender, KeyEventArgs e)

{

switch (e.KeyCode)

{

case Keys.W:

if (moveorstop(currentstandardbrick, 2))

clockwise(currentstandardbrick);

this.panel1.Refresh();

break;

case Keys.A:

if (moveorstop(currentstandardbrick, 1))

leftfunction(currentstandardbrick);

this.panel1.Refresh();

break;

case Keys.D:

if (moveorstop(currentstandardbrick,0))

rightfunction(currentstandardbrick);

this.panel1.Refresh();

break;

case Keys.S:

downfunction(currentstandardbrick);

break;

}

}

}

//创ä¡ä建¡§这a个?类¤¨¤是º?用®?于®¨²描¨¨述º?一°?个?标À¨º准Á?样¨´式º?的Ì?砖Á?块¨¦（ê¡§包ã¨¹含?坐Á?标À¨º和¨ª颜?色¦?）ê?

class standardbrick

{

Point[] brickpoints;

Color brickcolor;

public standardbrick(Point[] p, Color c)

{

brickpoints = p;

brickcolor = c;

}

public Point[] brickpointsduplicate

{

get { return brickpoints; }

}

public Color brickcolorduplicate

{

get { return brickcolor; }

}

}

//创ä¡ä建¡§这a个?类¤¨¤也°2是º?用®?于®¨²描¨¨述º?一°?个?具?体¬?的Ì?砖Á?块¨¦（ê¡§包ã¨¹含?样¨´式º?和¨ª颜?色¦?）ê?

class concretebrick

{

private string cbcode;

private Color cbcolor;

public concretebrick(string str, Color c)

{

cbcode = str;

cbcolor = c;

}

public string cbcodeduplicate

{

get { return cbcode; }

}

public Color cbcolorduplicate

{

get { return cbcolor; }

}

}

}