

**计算机与信息工程学院**

**软**

**件**

**工**

**程**

**与**

**计**

**算**

**二**

**论**

**文**

**网络编程**

**都鑫**

**20151104678**

**软件工程与计算二**

**都鑫**

**摘要：**通过这学期软件工程二这门课的学习，我了解了c#语言的基本使用过程，使用方法和winform的用法，掌握了picturebox工具的用法。掌握了vs的截图方法和文件存储功能的用法。

**关键字：**c#，vs，winfrom，picturebox。

**Understanding of computer systems**

Duxin

**Abstract**: Through this semester's software engineering two, I learned about the basic use of the c# language, the use of the method and the usage of WinForm, and the use of the PictureBox tools. Mastered the screenshot method of VS and the usage of file storage function.

**Key words：**c#，vs，winfrom，picturebox。

**引言：**

1. **程序主要原理，功能**

**1.原理：进入visual studio创建项目，在winfrom中建立picturebox。在picturebox中写出五子棋棋板，判断X轴，Y轴，（X,Y）轴，（Y，X）轴五个棋子连在一起的胜负。实现截图的功能，实现时间的显示，棋子重置。**

**2.功能：判断胜负，五子棋的基本功能。实现截图功能，实现时间的显示重置，棋子的重置。**

1. **主要代码**

**1.WinFrom界面主要代码**

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;

namespace WindowsFormsApplicationgobang

{

public partial class Form1 : Form

{

private static bool[,] isDraw = new bool[20, 20];//二t维?数簓组哩?用?于?检ì验é当獭?前°坐?标括?是?否?放?置?了?棋?子哩?

private MouseEventArgs e;

private static int ncount=1;

//private static bool isBlackStoneWin=false;

//private static int N = 1;

private static bool isBlackStoneWin = false;

private static bool isWhiteStoneWin = false;

//黑ú棋?

private const int n = 32;

private bool isPlayBlackStone;

private static int nblackstone = 0;

private static int bcount = 0;

private BlackStone[] black=new BlackStone[n];

private static bool[,] isBlackStone = new bool[20, 20];

//白恪?棋?

private bool isPlayWhiteStone;

private static int nwhitestone = 0;

private static int wcount = 0;

private WhiteStone[] white = new WhiteStone[n];

private static bool[,] isWhiteStone = new bool[20, 20];

public Form1()

{

InitializeComponent();

InitBlackStone();

InitWhiteStone();

labelBlack.BackColor = Color.Red;

labelBlack.Text = "黑ú方?下?";

}

public void InitBlackStone()

{

isPlayBlackStone = false;

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

{

black[i] = new BlackStone();

black[i].X = -10;

black[i].Y = -10;

}

for (int i = 0; i < 20; i++)

{

for (int j = 0; j < 20; j++)

{

isDraw[i, j] = false;

isBlackStone[i, j] = false;

}

}

}

public void InitWhiteStone()

{

isPlayWhiteStone = false;

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

{

white[i] = new WhiteStone();

white[i].X = -10;

white[i].Y = -10;

}

for (int i = 0; i < 20; i++)

{

for (int j = 0; j < 20; j++)

{

isWhiteStone[i, j] = false;

}

}

}

public void PlayBlackStone(MouseEventArgs e)

{

int x = panelGobang.ClientRectangle.X;

int y = panelGobang.ClientRectangle.Y;

int[] gobangX = new int[20];

int[] gobangY = new int[20];

for (int i = 0; i < 20; i++)

{

gobangX[i] = x + (i + 1) \* 20;

}

for (int i = 0; i < 20; i++)

{

gobangY[i] = y + (i + 1) \* 20;

}

for (int i = 0; i < 20; i++)

{

if ((e.X >= gobangX[i] - 10) && (e.X <= gobangX[i] + 10))

{

for (int j = 0; j < 20; j++)

{

if ((e.Y >= gobangY[j] - 10) && (e.Y <= gobangY[j] + 10))

{

black[bcount].X = gobangX[i];

black[bcount].Y = gobangY[j];

isPlayBlackStone = true;

break;

}

}

break;

}

}

}

public void PlayWhiteStone(MouseEventArgs e)

{

int x = panelGobang.ClientRectangle.X;

int y = panelGobang.ClientRectangle.Y;

int[] gobangX = new int[20];

int[] gobangY = new int[20];

for (int i = 0; i < 20; i++)

{

gobangX[i] = x + (i + 1) \* 20;

}

for (int i = 0; i < 20; i++)

{

gobangY[i] = y + (i + 1) \* 20;

}

for (int i = 0; i < 20; i++)

{

if ((e.X >= gobangX[i] - 10) && (e.X <= gobangX[i] + 10))

{

for (int j = 0; j < 20; j++)

{

if ((e.Y >= gobangY[j] - 10) && (e.Y <= gobangY[j] + 10))

{

white[wcount].X = gobangX[i];

white[wcount].Y = gobangY[j];

isPlayWhiteStone = true;

break;

}

}

break;

}

}

}

public bool IsBlackStoneWinSR()

{

int N = 1;

int x1 = black[bcount].X - 20;

int x2 = black[bcount].X + 20;

int y1 = black[bcount].Y + 20;

int y2 = black[bcount].Y - 20;

while (x1 >= 20 && y1 <= 360)

{

if (N == 5)

{

break;

}

if (isBlackStone[x1 / 20 - 1, y1 / 20 - 1] != false)

{

N++;

x1 = x1 - 20;

y1 = y1 + 20;

}

else

{

break;

}

}

if (N < 5)

{

while (x2 <= 360 && y2 >= 20)

{

if (N == 5)

{

break;

}

if (isBlackStone[x2 / 20 - 1, y2 / 20 - 1])

{

N++;

x2 = x2 + 20;

y2 = y2 - 20;

}

else

{

break;

}

}

}

if (N == 5)

{

return true;

}

else

{

return false;

}

}

public bool IsBlackStoneWinSL()

{

int N = 1;

int x1 = black[bcount].X - 20;

int x2 = black[bcount].X + 20;

int y1 = black[bcount].Y - 20;

int y2 = black[bcount].Y + 20;

while (x1>=20&&y1 >= 20)

{

if (N == 5)

{

break;

}

if (isBlackStone[x1 / 20 - 1, y1 / 20 - 1] != false)

{

N++;

x1 = x1 - 20;

y1 = y1 - 20;

}

else

{

break;

}

}

if (N < 5)

{

while (x2<=360&&y2 <= 360)

{

if (N == 5)

{

break;

}

if (isBlackStone[x2 / 20 - 1, y2 / 20 - 1])

{

N++;

x2 = x2 + 20;

y2 = y2 + 20;

}

else

{

break;

}

}

}

if (N == 5)

{

return true;

}

else

{

return IsBlackStoneWinSR();

}

}

public bool IsBlackStoneWinY()

{

int N = 1;

int y1 = black[bcount].Y - 20;

int y2 = black[bcount].Y + 20;

while (y1 >= 20)

{

if (N == 5)

{

break;

}

if (isBlackStone[black[bcount].X / 20 - 1, y1 / 20 - 1] != false)

{

N++;

y1 = y1 - 20;

}

else

{

break;

}

}

if (N < 5)

{

while (y2 <= 360)

{

if (N == 5)

{

break;

}

if (isBlackStone[black[bcount].X / 20 - 1, y2 / 20 - 1])

{

N++;

y2 = y2 + 20;

}

else

{

break;

}

}

}

if (N == 5)

{

return true;

}

else

{

return IsBlackStoneWinSL();

}

}

public bool IsBlackStoneWinX()

{

int N = 1;

int x1 = black[bcount].X - 20;

int x2 = black[bcount].X + 20;

while (x1 >= 20)

{

if (N == 5)

{

break;

}

if (isBlackStone[x1 / 20 - 1, black[bcount].Y / 20 - 1] != false)

{

N++;

x1 = x1 - 20;

}

else

{

break;

}

}

if (N < 5)

{

while (x2 <= 360)

{

if (N == 5)

{

break;

}

if (isBlackStone[x2 / 20 - 1, black[bcount].Y / 20 - 1])

{

N++;

x2 = x2 + 20;

}

else

{

break;

}

}

}

if (N == 5)

{

return true;

}

else

{

return IsBlackStoneWinY();

}

}

public bool IsWhiteStoneWinSR()

{

int N = 1;

int x1 =white[wcount].X - 20;

int x2 = white[wcount].X + 20;

int y1 = white[wcount].Y + 20;

int y2 = white[wcount].Y - 20;

while (x1 >= 20 && y1 <= 360)

{

if (N == 5)

{

break;

}

if (isWhiteStone[x1 / 20 - 1, y1 / 20 - 1] != false)

{

N++;

x1 = x1 - 20;

y1 = y1 + 20;

}

else

{

break;

}

}

if (N < 5)

{

while (x2 <= 360 && y2 >= 20)

{

if (N == 5)

{

break;

}

if (isWhiteStone[x2 / 20 - 1, y2 / 20 - 1])

{

N++;

x2 = x2 + 20;

y2 = y2 - 20;

}

else

{

break;

}

}

}

if (N == 5)

{

return true;

}

else

{

return false;

}

}

public bool IsWhiteStoneWinSL()

{

int N = 1;

int x1 = white[wcount].X - 20;

int x2 = white[wcount].X + 20;

int y1 = white[wcount].Y - 20;

int y2 = white[wcount].Y + 20;

while (x1 >= 20 && y1 >= 20)

{

if (N == 5)

{

break;

}

if (isWhiteStone[x1 / 20 - 1, y1 / 20 - 1] != false)

{

N++;

x1 = x1 - 20;

y1 = y1 - 20;

}

else

{

break;

}

}

if (N < 5)

{

while (x2 <= 360 && y2 <= 360)

{

if (N == 5)

{

break;

}

if (isWhiteStone[x2 / 20 - 1, y2 / 20 - 1])

{

N++;

x2 = x2 + 20;

y2 = y2 + 20;

}

else

{

break;

}

}

}

if (N == 5)

{

return true;

}

else

{

return IsWhiteStoneWinSR();

}

}

public bool IsWhiteStoneWinY()

{

int N = 1;

int y1 = white[wcount].Y - 20;

int y2 = white[wcount].Y + 20;

while (y1 >= 20)

{

if (N == 5)

{

break;

}

if (isWhiteStone[white[wcount].X / 20 - 1, y1 / 20 - 1] != false)

{

N++;

y1 = y1 - 20;

}

else

{

break;

}

}

if (N < 5)

{

while (y2 <= 360)

{

if (N == 5)

{

break;

}

if (isWhiteStone[white[wcount].X / 20 - 1, y2 / 20 - 1])

{

N++;

y2 = y2 + 20;

}

else

{

break;

}

}

}

if (N == 5)

{

return true;

}

else

{

return IsWhiteStoneWinSL();

}

}

public bool IsWhiteStoneWinX()

{

int N = 1;

int x1 = white[wcount].X - 20;

int x2 = white[wcount].X + 20;

while (x1 >= 20)

{

if (N == 5)

{

break;

}

if (isWhiteStone[x1 / 20 - 1, white[wcount].Y / 20 - 1] != false)

{

N++;

x1 = x1 - 20;

}

else

{

break;

}

}

if (N < 5)

{

while (x2 <= 360)

{

if (N == 5)

{

break;

}

if (isWhiteStone[x2 / 20 - 1, white[wcount].Y / 20 - 1])

{

N++;

x2 = x2 + 20;

}

else

{

break;

}

}

}

if (N == 5)

{

return true;

}

else

{

return IsWhiteStoneWinY();

}

}

public void RenewStone()

{

bcount = 0;

nblackstone = 0;

isPlayBlackStone = false;

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

{

//black[i] = new BlackStone();

black[i].X = -10;

black[i].Y = -10;

//white[i] = new WhiteStone();

white[i].X = -10;

white[i].Y = -10;

}

for (int i = 0; i < 20; i++)

{

for (int j = 0; j < 20; j++)

{

isDraw[i, j] = false;

isBlackStone[i, j] = false;

}

}

ncount = 1;

wcount = 0;

nwhitestone = 0;

isPlayWhiteStone = false;

for (int i = 0; i < 20; i++)

{

for (int j = 0; j < 20; j++)

{

isWhiteStone[i, j] = false;

}

}

}

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

{

int x = panelGobang.ClientRectangle.X;

int y = panelGobang.ClientRectangle.Y;

for (int i = 0; i < 22; i++)

{

e.Graphics.DrawLine(new Pen(new SolidBrush(Color.Black)), new Point(x, y + 20 \* i), new Point(x + 400, y + 20 \* i));

}

for (int i = 0; i < 22; i++)

{

e.Graphics.DrawLine(new Pen(new SolidBrush(Color.Black)), new Point(x + 20 \* i, y), new Point(x + 20 \* i, y + 400));

}

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

{

black[i].DrawBlackStone(e.Graphics);

}

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

{

white[i].DrawWhiteStone(e.Graphics);

}

if (ncount % 2 != 0)

{

if (isBlackStoneWin == true)

{

isBlackStoneWin = false;

labelBlack.Text = "黑ú方?下?";

}

if (isWhiteStoneWin == true)

{

isWhiteStoneWin = false;

labelBlack.Text="黑ú方?下?";

labelWhite.Text = "";

}

if (isPlayBlackStone)

{

if (!isDraw[(black[bcount].X) / 20 - 1, (black[bcount].Y) / 20 - 1])

{

black[bcount].DrawBlackStone(e.Graphics);

nblackstone = bcount + 1;

isPlayBlackStone = false;

isDraw[(black[bcount].X) / 20 - 1, (black[bcount].Y) / 20 - 1] = true;

isBlackStone[(black[bcount].X) / 20 - 1, (black[bcount].Y) / 20 - 1] = true;

if (IsBlackStoneWinX() == true)

{

MessageBox.Show("黑ú方?获?胜骸?);

label2.Text = DateTime.Now.ToString();

// RenewStone();

isBlackStoneWin = true;

//labelBlack.Text = "";

//labelWhite.Text = "白恪?方?下?";

panelGobang.Invalidate();

return;

}

bcount++;

ncount++;

labelWhite.BackColor = Color.RoyalBlue;

labelWhite.Text = "白恪?方?下?";

labelBlack.Text = "";

}

}

}

else

{

if (isPlayWhiteStone)

{

if (!isDraw[(white[wcount].X) / 20 - 1, (white[wcount].Y) / 20 - 1])

{

white[wcount].DrawWhiteStone(e.Graphics);

nwhitestone = wcount + 1;

isPlayWhiteStone = false;

isDraw[(white[wcount].X) / 20 - 1, (white[wcount].Y) / 20 - 1] = true;

isWhiteStone[(white[wcount].X) / 20 - 1, (white[wcount].Y) / 20 - 1] = true;

if (IsWhiteStoneWinX() == true)

{

MessageBox.Show("白恪?方?获?胜骸?);

label2.Text = DateTime.Now.ToString();

// RenewStone();

isWhiteStoneWin = true;

//labelBlack.Text = "";

//labelWhite.Text = "白恪?方?下?";

panelGobang.Invalidate();

labelWhite.Text = "白恪?方?下?";

return;

}

wcount++;

ncount++;

labelBlack.BackColor = Color.Red;

labelBlack.Text = "黑ú方?下?";

labelWhite.Text = "";

}

}

}

}

private void panelGobang\_MouseClick(object sender, MouseEventArgs e)

{

if (ncount % 2 != 0)

{

if (bcount >= n)

{

if (wcount == n)

{

MessageBox.Show("您ú的?棋?子哩?己o用?完?");

isPlayBlackStone = false;

return;

}

}

this.e = e;

PlayBlackStone(e);

panelGobang.Invalidate();

}

else

{

if (wcount >= n)

{

MessageBox.Show("您ú的?棋?子哩?己o用?完?");

isPlayWhiteStone = false;

return;

}

this.e = e;

PlayWhiteStone(e);

panelGobang.Invalidate();

}

}

int i = 0;

int w = Screen.PrimaryScreen.Bounds.Width;

int h = Screen.PrimaryScreen.Bounds.Height;

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

{

label2.Text = DateTime.Now.ToString();

i += 1;

Bitmap bit1 = new Bitmap(this.Width, this.Height);

this.DrawToBitmap(bit1, new Rectangle(0, 0, this.Width, this.Height));

int border = (this.Width - this.ClientSize.Width) / 2;//边?框ò宽í度è

int caption = (this.Height - this.ClientSize.Height) - border;//标括?题琣栏?高?度è

Bitmap bit2 = bit1.Clone(new Rectangle(border, caption, this.ClientSize.Width, this.ClientSize.Height), System.Drawing.Imaging.PixelFormat.Format24bppRgb);

bit1.Save("C:\\Users\\q\\Desktop\\截?图?小?王?子哩甛\比括?赛è?结á果?" + i + ".jpg", System.Drawing.Imaging.ImageFormat.Jpeg);//包悒?括ぁ?标括?题琣栏?和í边?框ò

// bit2.Save("D:\\BBB.jpg", System.Drawing.Imaging.ImageFormat.Jpeg);//不?包悒?括ぁ?标括?题琣栏?和í边?框ò

bit1.Dispose();

// bit2.Dispose();

}

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

{

System.Diagnostics.Process.Start("C:\\Users\\q\\Desktop\\截?图?小?王?子哩?);

TopMost = false;

this.Cursor = Cursors.Default;

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void Form1\_Load(object sender, EventArgs e)

{

label1.Text = DateTime.Now.ToString();

}

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

{

label1.Text = DateTime.Now.ToString();

RenewStone();

}

private void label2\_Click(object sender, EventArgs e)

{

}

}

}

**2.白棋定义类**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Drawing;

namespace WindowsFormsApplicationgobang

{

class WhiteStone

{

public int X

{

set;

get;

}

public int Y

{

set;

get;

}

public void DrawWhiteStone(Graphics g)

{

using (SolidBrush brush = new SolidBrush(Color.Silver))

{

g.FillEllipse(brush, X - 10, Y - 10, 20, 20);

}

}

}

}

**3.黑棋定义类**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Drawing;

namespace WindowsFormsApplicationgobang

{

class BlackStone

{

public int X

{

set;

get;

}

public int Y

{

set;

get;

}

public void DrawBlackStone(Graphics g)

{

using (SolidBrush brush = new SolidBrush(Color.Black))

{

g.FillEllipse(brush, X - 10, Y - 10, 20, 20);

}

}

}

}

**4.panel1.cs定义**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace WindowsFormsApplicationgobang

{

class panel1:Panel

{

public panel1()

{

SetStyle(ControlStyles.UserPaint |

ControlStyles.AllPaintingInWmPaint |

ControlStyles.OptimizedDoubleBuffer |

ControlStyles.ResizeRedraw |

ControlStyles.SupportsTransparentBackColor, true);

}

}

}

**四．实验小结**

在本次实验中，经过认真的分析题目和熟悉使用 vs 2010，我决定开发一个五子棋小游戏程序，在本次实验中对picturebox绘制图片技术有了新的掌握，还了解了如何将程序运行界面保存成一张图片存储在一个文件夹里，保证了下棋的公正公平，也保证了棋局的回放。

**参考文献**

1. **Github**

**[2]www.cnblogs.com**