|  |
| --- |
| **МИНОБРНАУКИ РОССИИ**  федеральное государственное бюджетное образовательное  учреждение высшего образования  **«Национальный исследовательский университет «МЭИ»** |

|  |  |
| --- | --- |
| **Институт** | ИВТИ |
| **Кафедра** | ПМИИ |

|  |  |
| --- | --- |
|  |  |
| **Дисциплина: «Системное программирование»**  **Отчет по лабораторной работе №6**  **«Основы GDI+»**  **Выполнил: студент группы А-13а-19**  **Рамазанов Н. М.**  **Преподаватель: Меньшикова К. Г.**  **Москва, 2021** | |

**1. Краткое описание функций, параметров и используемых событий.**

1) Дескриптор контекста устройства и функции для его получения.

* **e.Graphics**

Класс предоставляет методы для рисования объектов на устройстве отображения. Объект Graphics связан с конкретным контекстом устройства.

* **this.CreateGraphics()**

Создает объект Graphics для элемента управления.

* **Graphics.FromHwnd**()

Создает новый объект Graphics из указанного дескриптора окна.

2) Выбор и создание кисти.

* **System.Drawing.Brushes**

Определяет объекты, которые используются для заливки внутри графических фигур, таких как прямоугольники, эллипсы, круги, многоугольники и дорожки.

* **SolidBrush**

Определяет кисть одного цвета. Кисти используются для заливки графических фигур, таких как прямоугольники, эллипсы, круги, многоугольники и контуры.

* **System.Drawing.Drawing2D.HatchBrush**

Задает прямоугольную кисть со стилем штриховки, основным цветом и цветом фона.

* **System.Drawing.TextureBrush**

Каждое свойство класса TextureBrush является объектом Brush, использующим изображение для заливки внутренней части формы.

* **LinearGradientBrush**

Инкапсулирует объект Brush с линейным градиентом.

3) Выбор и создание карандаша.

Стандартные карандаши: **System.Drawing.Pens**

Создание карандаша.

Например: p = **new Pen**(Color.Aquamarine, 3);

Ширина: p.**Width** = 5;

Цвет: p.**Color** = Color.Coral;

Стиль: p.**DashStyle** = System.Drawing.Drawing2D.DashStyle.DashDot;

Определяет объект, используемый для рисования прямых линий и кривых. Pen рисует линию указанной ширины и стиля. DashStyle - свойство позволяет нарисовать несколько штрихов пунктирных линий. Линия, нарисованная Pen, может быть заполнена различными стилями заливки, включая сплошные цвета и текстуры. Стиль заливки зависит от кисти или текстуры, используемой в качестве объекта заливки.

4) Функции рисования.

|  |  |
| --- | --- |
| **Функция** | **Смысл** |
| DrawArc | Рисует дугу по заданным координатам |
| DrawLine | Рисует линию от текущей позиции до места, указанного в аргументах |
| DrawEllipse | Рисует эллипс по заданным координатам |
| DrawRectangle | Рисует прямоугольник по заданным координатам |
| DrawPolygon | Рисует многоугольник по массиву точек |
| DrawImage | Рисует картинку |
| DrawString | Отображает строку текста |
| FillEllipse | Заливка эллипса выбранным цветом |
| FillRectangle | Заливка прямоугольника выбранным цветом |
| FillPolygon | Заливка многоугольника выбранным цветом |
| FillRegion | Заливка области выбранным цветом |

5) Прямоугольники, регионы, область рисования.

**Rectangle** r=new Rectangle(100,100,50,150);

Содержит набор из четырех целых чисел, определяющих расположение и размер прямоугольника.

**Region** reg = new Region(r);

Описывает внутреннюю часть графической формы, состоящей из

прямоугольников и контуров.

Свойство **Clip** контекста устройства

Возвращает или задает объект Region, ограничивающий область рисования данного объекта Graphics.

6) Область отсечения и обновление окна.

* Button.**Refresh**()

Принудительно создает условия, при которых элемент управления делает недоступной свою клиентскую область и немедленно перерисовывает себя и все дочерние элементы.

* **Invalidate()**

Делает недействительной указанную область элемента управления (добавляет ее к области обновления элемента, которая будет перерисована при следующей операции рисования) и вызывает отправку сообщения рисования элементу управления.

7) Цвет.

Color col=**Color.FromArgb**(255,255,100);

Создает структуру Color из указанных 8-разрядных значений цветов (красный, зеленый, синий). Значение альфа неявно определено как 255 (полностью непрозрачно). Хотя и этот метод позволяет передать 32-разрядное значение для каждого компонента цвета, значение каждого из них ограничено 8 разрядами.

Color col=**Color.FromArgb**(100,255,255,100);

Создает структуру Color из четырех значений компонентов ARGB (альфа, красный, зеленый и синий). Хотя и этот метод позволяет передать 32-разрядное значение для каждого компонента, значение каждого из них ограничено 8 разрядами.

Метод **Clear** – обновление области рисования и заливка заданным цветом. Удаляет из буфера консоли и ее окна отображаемую информацию.

**2. Выполнение работы.**

Для выполнения работы создал приложение на языке C#.

1. Придумал маленькую эмблему своей программы и нарисовал ее на всех кнопках. Картинка не исчезает после сворачивания или перекрытия окна.

События Paint кнопок:

private void button1\_Paint(object sender, PaintEventArgs e) { DrawImgOnBtn(e); }

private void button2\_Paint(object sender, PaintEventArgs e) { DrawImgOnBtn(e); }

Функция рисования эмблемы:

private void DrawImgOnBtn(PaintEventArgs e)

{

Pen rPen = new Pen(Color.Red, 4);

SolidBrush rBrush = new SolidBrush(Color.Tomato);

Point[] points = new Point[8];

points[0] = new Point(25, 50);

points[1] = new Point(10, 30);

points[2] = new Point(15, 20);

points[3] = new Point(20, 20);

points[4] = new Point(25, 30);

points[5] = new Point(30, 20);

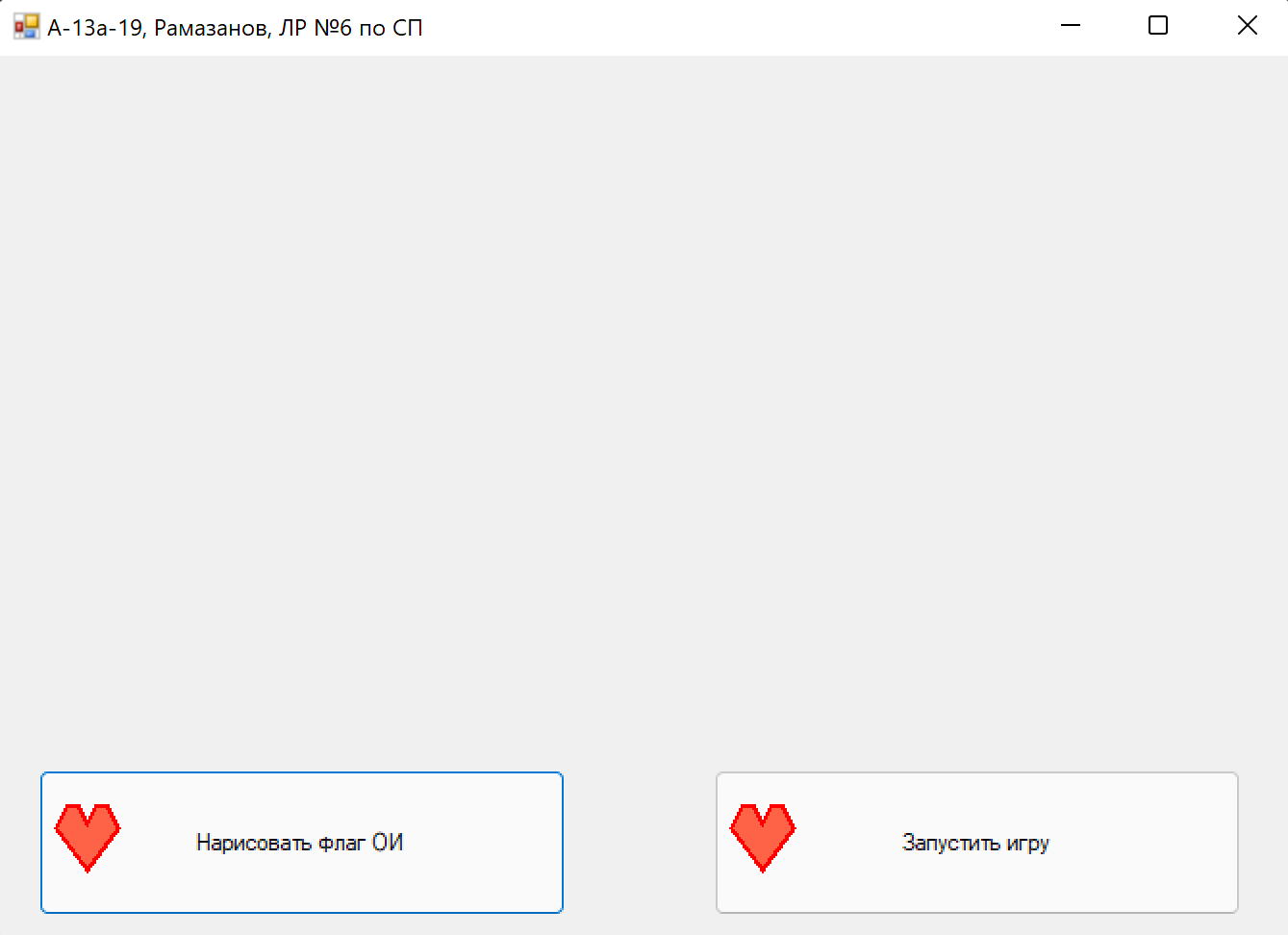
points[6] = new Point(35, 20);

points[7] = new Point(40, 30);

e.Graphics.DrawPolygon(rPen, points);

e.Graphics.FillPolygon(rBrush, points);

}



1. Нарисовал флаг Олимпийских игр.

Определил логическую переменную: public bool is\_draw\_olympics = false;

Реакция на нажатие кнопки:

private void button1\_Click(object sender, EventArgs e)

{

is\_draw\_olympics = !is\_draw\_olympics;

Refresh();

}

Добавил в событие Paint окна Form1 отрисовку флага:

private void Form1\_Paint(object sender, PaintEventArgs e)

{

if (is\_draw\_olympics)

{

Pen rPen = new Pen(Color.Red, 6);

Pen bPen = new Pen(Color.Black, 6);

Pen bluePen = new Pen(Color.Blue, 6);

Pen gPen = new Pen(Color.Green, 6);

Pen yPen = new Pen(Color.Yellow, 6);

SolidBrush wBrush = new SolidBrush(Color.White);

e.Graphics.DrawRectangle(bPen, 0, 0, 325, 175);

e.Graphics.FillRectangle(wBrush, 0, 0, 325, 175);

e.Graphics.DrawEllipse(bluePen, 10, 10, 90, 90);

e.Graphics.DrawEllipse(bPen, 110, 10, 90, 90);

e.Graphics.DrawEllipse(yPen, 60, 60, 90, 90);

Rectangle r1 = new Rectangle(110, 10, 90, 90);

Rectangle r = new Rectangle(10, 10, 90, 90);

e.Graphics.DrawArc(bluePen, r, -30, 90);

e.Graphics.DrawArc(bPen, r1, 90, 20);

e.Graphics.DrawEllipse(rPen, 210, 10, 90, 90);

e.Graphics.DrawEllipse(gPen, 160, 60, 90, 90);

Rectangle r2 = new Rectangle(110, 10, 90, 90);

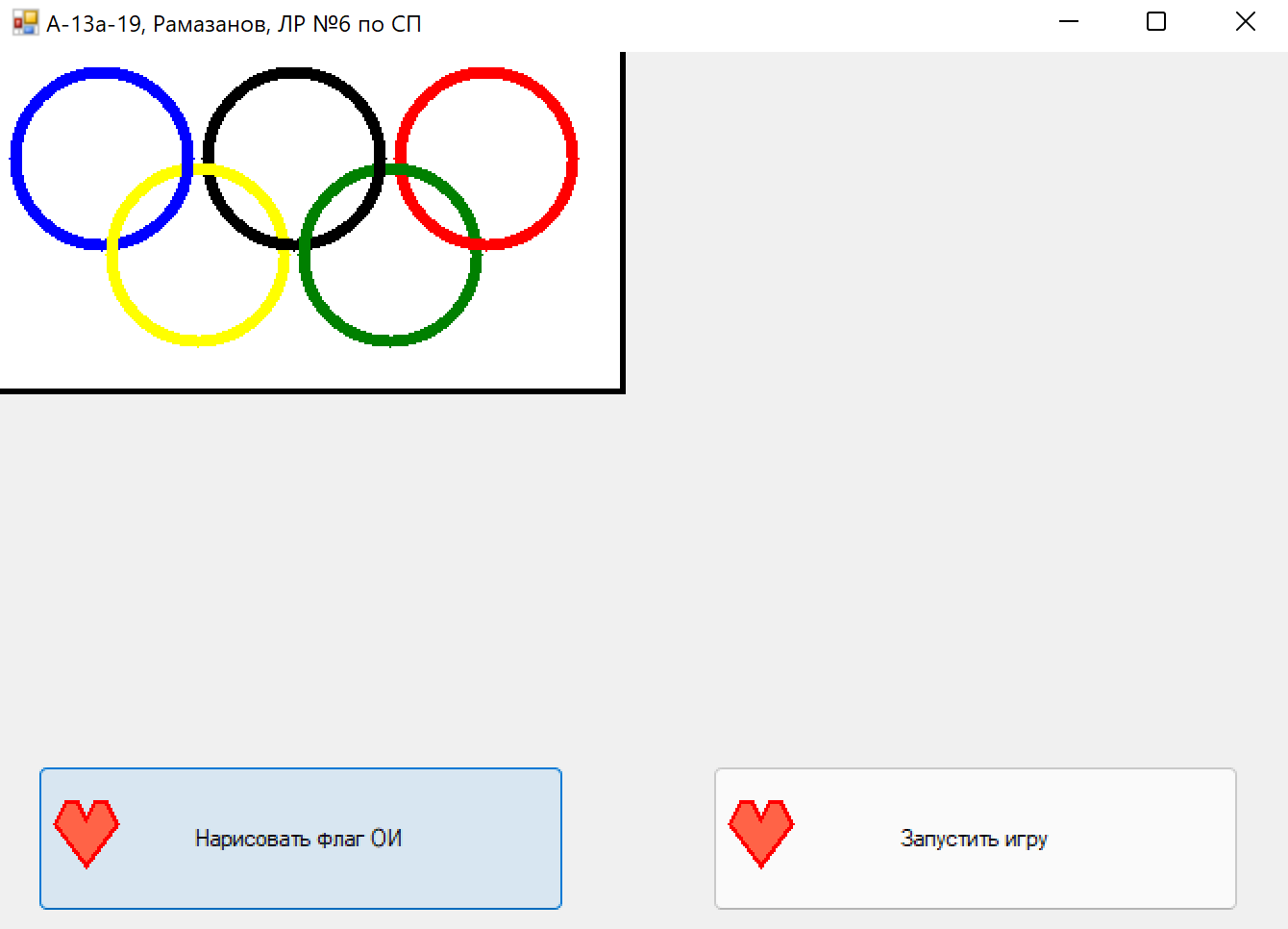
e.Graphics.DrawArc(bPen, r2, -30, 90);

Rectangle r3 = new Rectangle(210, 10, 90, 90);

e.Graphics.DrawArc(rPen, r3, 90, 20);

}

}



1. Создал программу «Игра», похожую на крестики-нолики. Игровое поле – клетки-прямоугольники. При щелчке на клетке визуализируется ход игрока номер 1, следующий щелчок совершает игрок номер 2. Выигрывает тот, кто выбрал 4 клетки подряд. При создании интерфейса для игры использовать текстурные и градиентные кисти, а также режим прозрачности.

Добавим кнопку для начала игры:

private void button2\_Click(object sender, EventArgs e)

{

is\_play = !is\_play;

label1.Visible = !label1.Visible;

label1.Text = "Ход игрока №1";

Refresh();

}

Добавим в Form1\_Paint отрисовку игрового поля и рисование крестиков-ноликов:

if (is\_play)

{

Image background = Image.FromFile("D:\\учеба\\5th\_semester\\СистПрог\\lab6\\imglab6.jpg");

Pen bPen = new Pen(Color.Black, 4);

Rectangle myR = new Rectangle(350, 100, 250, 250);

LinearGradientBrush myLGB = new LinearGradientBrush(myR, Color.Aqua, Color.LightGreen, LinearGradientMode.Vertical);

e.Graphics.FillRectangle(myLGB, myR);

TextureBrush myTB = new TextureBrush(background);

e.Graphics.FillRectangle(myTB, 400, 150, 150, 150);

HatchBrush myHB = new HatchBrush(HatchStyle.Percent90, Color.LightGray);

e.Graphics.FillRectangle(myHB, 450, 200, 50, 50);

for (int i = 100; i <= 350; i += 50)

{

PointF point1 = new PointF(350, i);

PointF point2 = new PointF(600, i);

e.Graphics.DrawLine(bPen, point1, point2);

}

for (int i = 350; i <= 600; i += 50)

{

PointF point1 = new PointF(i, 100);

PointF point2 = new PointF(i, 350);

e.Graphics.DrawLine(bPen, point1, point2);

}

DrawCrossOrZero(e);

}

Реакция на нажатие мыши в области окна во время игры:

private void Form1\_MouseClick(object sender, MouseEventArgs e)

{

if (is\_play)

{

mouse\_coords = new PointF(e.X, e.Y);

Tictactoe();

Refresh();

}

}

Для игры создал заполненный нулями двумерный массив 5х5, олицетворяющий игровое поле. Если игрок №1 выбирает пустую ячейку поля, соответствующий элемент массива меняется на единичку, если игрок №2 – на двойку.

public int[,] mas = new int[5, 5];

public void FillMasZeros(int[,] mas)

{

for (int i = 0; i < 5; i++)

for (int j = 0; j < 5; j++)

mas[i, j] = 0;

}

Функции для игры: определение нажатой клетки с переходом хода другому игроку, если выбрана пустая клетка, и повторной попыткой хода, если выбрана непустая клетка или клик совершен не в игровом поле; рисование крестика и нолика; функция проверки выигрыша (выигрывает игрок, который собрал 4 в ряд ячейки по горизонтали или вертикали); функция вывода результата игры.

public void Tictactoe()

{

if (player == 1)

{

label1.Text = "Ход игрока №2";

int indi = 0;

int indj = 0;

bool br = false;

for (int i = 100; i < 350; i += 50)

{

indj = 0;

for (int j = 350; j < 600; j += 50)

{

if ((mouse\_coords.X >= j && mouse\_coords.X <= j + 50) &&

((mouse\_coords.Y >= i && mouse\_coords.Y <= i + 50)))

br = true;

if (br)

break;

indj++;

}

if (br)

break;

indi++;

}

if (indi > 4 || indj > 4 || mas[indi, indj] != 0)

{

label1.Text = "Выберите пустую клетку";

return;

}

mas[indi, indj] = 1;

player = 2;

}

else

{

label1.Text = "Ход игрока №1";

int indi = 0;

int indj = 0;

bool br = false;

for (int i = 100; i < 350; i += 50)

{

indj = 0;

for (int j = 350; j < 600; j += 50)

{

if ((mouse\_coords.X >= j && mouse\_coords.X <= j + 50) &&

((mouse\_coords.Y >= i && mouse\_coords.Y <= i + 50)))

br = true;

if (br)

break;

indj++;

}

if (br)

break;

indi++;

}

if (indi > 4 || indj > 4 || mas[indi, indj] != 0)

{

label1.Text = "Выберите пустую клетку";

return;

}

mas[indi, indj] = 2;

player = 1;

}

}

public void DrawCrossOrZero(PaintEventArgs e)

{

Pen p1Pen = new Pen(Color.Red, 4);

Pen p2Pen = new Pen(Color.Blue, 4);

for (int i = 2; i < 7; i++)

for (int j = 7; j < 12; j++)

{

if (mas[i - 2, j - 7] == 1)

{

PointF point1 = new PointF(j \* 50, i \* 50);

PointF point2 = new PointF(j \* 50 + 50, i \* 50 + 50);

e.Graphics.DrawLine(p1Pen, point1, point2);

point1 = new PointF(j \* 50, i \* 50 + 50);

point2 = new PointF(j \* 50 + 50, i \* 50);

e.Graphics.DrawLine(p1Pen, point1, point2);

}

if (mas[i - 2, j - 7] == 2)

e.Graphics.DrawEllipse(p2Pen, j \* 50, i \* 50, 50, 50);

}

CheckWinner(e);

}

public void CheckWinner(PaintEventArgs e)

{

Pen gPen = new Pen(Color.Green, 4);

for (int i = 0; i < 5; i++)

for (int j = 0; j < 2; j++)

{

if (mas[i, j] == 1 && mas[i, j + 1] == 1 && mas[i, j + 2] == 1 && mas[i, j + 3] == 1)

{

PointF point1 = new PointF((j + 7) \* 50, (2 + i) \* 50 + 25);

PointF point2 = new PointF((11 + j) \* 50, (2 + i) \* 50 + 25);

e.Graphics.DrawLine(gPen, point1, point2);

win1 = true;

WhoWin();

return;

}

else if (mas[i, j] == 2 && mas[i, j + 1] == 2 && mas[i, j + 2] == 2 && mas[i, j + 3] == 2)

{

PointF point1 = new PointF((j + 7) \* 50, (2 + i) \* 50 + 25);

PointF point2 = new PointF((11 + j) \* 50, (2 + i) \* 50 + 25);

e.Graphics.DrawLine(gPen, point1, point2);

win2 = true;

WhoWin();

return;

}

}

for (int i = 0; i < 5; i++)

for (int j = 0; j < 2; j++)

{

if (mas[j, i] == 1 && mas[j + 1, i] == 1 && mas[j + 2, i] == 1 && mas[j + 3, i] == 1)

{

PointF point1 = new PointF((i + 7) \* 50 + 25, (2 + j) \* 50);

PointF point2 = new PointF((7 + i) \* 50 + 25, (6 + j) \* 50);

e.Graphics.DrawLine(gPen, point1, point2);

win1 = true;

WhoWin();

return;

}

else if (mas[j, i] == 2 && mas[j + 1, i] == 2 && mas[j + 2, i] == 2 && mas[j + 3, i] == 2)

{

PointF point1 = new PointF((i + 7) \* 50 + 25, (2 + j) \* 50);

PointF point2 = new PointF((7 + i) \* 50 + 25, (6 + j) \* 50);

e.Graphics.DrawLine(gPen, point1, point2);

win2 = true;

WhoWin();

return;

}

}

WhoWin();

}

public void WhoWin() {

if (win1)

{

label1.Text = "Победитель: игрок №1";

win1 = false;

player = 1;

FillMasZeros(mas);

}

else if (win2)

{

label1.Text = "Победитель: игрок №2";

win2 = false;

player = 1;

FillMasZeros(mas);

}

else

{

for (int i = 0; i < 5; i++)

for (int j = 0; j < 5; j++)

if (mas[i, j] == 0)

return;

label1.Text = "Игра закончилась вничью";

}

}

Полный код программы доступен по ссылке: <https://github.com/programzan/SystemProgramming/blob/lab6/Form1.cs>

Здесь же размещены все сопутствующие файлы: <https://github.com/programzan/SystemProgramming/tree/lab6>