МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Российский государственный университет им. А.Н. Косыгина   
(Технологии. Дизайн. Искусство)»

Кафедра автоматизированных систем обработки информации и управления

Отчет по лабораторной работе № 9

по дисциплине «Технологии программирования»

Тема: «Делегаты»

Выполнил: Стычинский С.В., МВИ-121

Проверили: Кузьмина Т. М. и Адаев Р. Б.

Москва 2023

Цель работы

Научиться работать с делегатами.

Задание

Этапы выполнения работы.

1. Описать делегат, который будет работать с методами, выводящими фрагменты рисунка. Можно описать делегат, который принимает объект класса Graphics, как параметр. Можно описать делегат без параметров, рассчитывая, что будет доступен «глобальный» объект класса Graphics. Считается, что рисунок будет неподвижным, имеющим фиксированные координаты, поэтому координаты передавать как параметры не надо.

2. Создать «глобальный» для методов формы экземпляр делегата.

3. С помощью методов класса Graphics вывести на первую панель формы некоторый рисунок. Примеры рисунков приведены ниже. Можно создать свой рисунок, а не повторять рисунки из приведенных примеров. Рисунок каждого студента должен быть уникальным. Если рисунок одного студента повторит работу другого студента, то преподаватель укажет, какие изменения придется внести в копию.

4. Программу, которая выводит рисунок надо разбить на 6 фрагментов, которые нужно оформить, как отдельные методы. Фрагменты оформить в виде методов, имеющих ту же структуру, что и определенный ранее делегат. Эти методы должны быть следующих видов: 1) Статический метод класса Form1. 2) Нестатический метод класса Form1. 3) Статический метод пользовательского класса (для этого создать класс, имя которого должно совпадать с именем или фамилией разработчика). 4) Нестатический метод пользовательского класса (использовать для этого класс, определенный для выполнения пункта 3). 5) Нестатический метод, определяемый лямбда выражением. 6) Лямбда выражение, связанное с делегатом напрямую, без использования вспомогательного метода.

5. На форму поместить 5 компонента СheckBox, для подключения и отключения всех методов, кроме лямбда-выражения (пункт 6 предыдущего задания). Каждый СheckBox назвать по имени фрагмента рисунка, выводом которого он будет заведовать (например, «крыша», «окно», «дверь», «ручка на двери» для домика). Если флажок компонента СheckBox поднят, то соответствующий фрагмент рисунка должен появиться на экране, в противном случае его быть на экране не должно.

6. Фрагмент рисунка, заданный лямбда выражением, связанным с делегатом напрямую (пункт 6 предыдущего задания), должен выводится на экран во всех случаях. Для включения и исключения остальных фрагментов в рисунок нужно использовать операторы += и -=.

7. На форме должна быть кнопка с названием «Рисовать», обработчик события клик этой кнопки должен содержать команду очистки панели для рисования и вызов делегата, который обеспечивает рисование выбранных фрагментов.

Распечатка программы

Form1

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace lab9

{

public partial class Form1 : Form

{

private Stychinskiy ob = new Stychinskiy();

public static Bitmap bmp;

public static Graphics g;

public static Graphics gg;

delegate void Draw(); //Делегат

private Draw draw;

public Form1()

{

InitializeComponent();

bmp = new Bitmap(pictureBox1.Width, pictureBox1.Height);

g = Graphics.FromImage(bmp);

gg = Graphics.FromImage(bmp);

pictureBox1.BackColor = Color.Transparent;

draw += DrawBody;

}

// статический метод класса Form1

public static void DrawHead()

{

//ГОЛОВА НАЧАЛО ---

Point[] points1 =

{

new Point(345,50),

new Point(395,60),

new Point(470,98),

new Point(495,153),

new Point(445,160),

new Point(395,175),

new Point(370,180),

new Point(345,185),

new Point(320,180),

new Point(295, 175),

new Point(245,160),

new Point(195,153),

new Point(220,98),

new Point(295,60),

new Point(345,50)

};

//g.DrawLines(new Pen(Color.Black, 3), points1);

g.FillPolygon(new SolidBrush(Color.LightGray), points1);

//левое ушко

Point[] points2 =

{

new Point(224,97),

new Point(227,28),

new Point(263,78),

new Point(228,94),

};

g.DrawLines(new Pen(Color.LightGray, 10), points2);

g.FillPolygon(new SolidBrush(Color.Pink), points2);

//правое ушко

Point[] points3 =

{

new Point(430,78),

new Point(460,28),

new Point(468,98),

new Point(430,78),

};

g.DrawLines(new Pen(Color.LightGray, 10), points3);

g.FillPolygon(new SolidBrush(Color.Pink), points3);

//рот

g.FillEllipse(Brushes.White, 315, 140, 60, 40); //белая часть у рта

g.FillEllipse(Brushes.Pink, 337, 150, 20, 10); //розовая часть рта

Point[] pointsMouth =

{

new Point(347,160),

new Point(341,172),

new Point(335,165),

};

g.DrawLines(new Pen(Color.Black, 2), pointsMouth);

Point[] pointsMouth2 =

{

new Point(347,160),

new Point(353,172),

new Point(360,165)

};

g.DrawLines(new Pen(Color.Black, 2), pointsMouth2);

//глаза

Point[] PointsEye1 =

{

new Point(339,103),

new Point(300,95),

new Point(261,115),

new Point(229,158),

new Point(280,170),

new Point(311,140),

new Point(339,123),

new Point(339,103),

};

Point[] PointsEye2 =

{

new Point(356,103),

new Point(387,95),

new Point(430,115),

new Point(461,158),

new Point(410,170),

new Point(374,140),

new Point(356,123),

new Point(356,103),

};

g.FillPolygon(new SolidBrush(Color.Gray), PointsEye1);

g.FillPolygon(new SolidBrush(Color.Gray), PointsEye2);

//левый глаз

g.FillEllipse(Brushes.White, 295, 102, 30, 30);

g.FillEllipse(Brushes.Black, 303, 110, 15, 15);

//правый глаз

g.FillEllipse(Brushes.White, 365, 102, 30, 30);

g.FillEllipse(Brushes.Black, 373, 110, 15, 15);

}

// нестатический метод класса Form1

public void DrawTail()

{

//хвост

Point[] xvost =

{

new Point(275, 375),

new Point(259,382),

new Point(248, 385),

new Point(237, 387),

new Point(221, 380),

new Point(212, 368),

new Point(204,350),

new Point(202,349),

new Point(196, 351),

new Point(176, 356),

new Point(162, 358),

new Point(155, 355),

new Point(156, 348),

new Point(165, 341),

new Point(171, 336),

new Point(179, 330),

new Point(189, 325),

new Point(203, 322),

new Point(220,322),

new Point(235, 327),

new Point(246, 333),

new Point(261, 350),

new Point(263, 356),

new Point(267, 363),

new Point(275, 375),

};

g.FillPolygon(new SolidBrush(Color.LightGray), xvost);

//4 цветная часть хвоста

Point[] xxx4 =

{

new Point(164, 342),

new Point(171, 337),

new Point(176, 342),

new Point(180, 355),

new Point(169, 356),

new Point(164, 342),

};

g.FillPolygon(new SolidBrush(Color.Gray), xxx4);

//3 цветная часть хвоста

Point[] xxx3 =

{

new Point(201, 348),

new Point(200, 322),

new Point(187, 327),

new Point(191, 352),

new Point(201, 348),

};

g.FillPolygon(new SolidBrush(Color.Gray), xxx3);

//2 цветная часть хвоста

Point[] xxx2 =

{

new Point(219,323),

new Point(221,331),

new Point(221,337),

new Point(221,347),

new Point(219,353),

new Point(215,360),

new Point(210,364),

new Point(218,375),

new Point(223,371),

new Point(227,368),

new Point(231,362),

new Point(233,355),

new Point(233,348),

new Point(233,341),

new Point(233,335),

new Point(232,332),

new Point(232,329),

new Point(230,326),

new Point(230,326),

new Point(226,324),

new Point(219,323),

};

g.FillPolygon(new SolidBrush(Color.Gray), xxx2);

//1 цветная часть хвоста

Point[] xxx1 =

{

new Point(251, 340),

new Point(260, 349),

new Point(258, 362),

new Point(256, 370),

new Point(252, 376),

new Point(248 , 386),

new Point(234, 386),

new Point(242, 370),

new Point(245, 364),

new Point(249, 346),

new Point(251, 340),

};

g.FillPolygon(new SolidBrush(Color.Gray), xxx1);

}

// хвост

private static void SDrawBody()

{

g.FillEllipse(Brushes.LightGray, 250, 170, 200, 250);

g.FillEllipse(Brushes.White, 275, 172, 150, 190);

}

// лямба-выражение -> хвост

public void DrawBody() => SDrawBody();

private void head\_CheckedChanged(object sender, EventArgs e)

{

if (head.Checked) draw += DrawHead;

else draw -= DrawHead;

}

private void Paws1\_CheckedChanged(object sender, EventArgs e)

{

if (Paws1.Checked) draw += ob.DrawPaws1; //обращение к нестатическому методу

else draw -= ob.DrawPaws1;

}

private void Paws2\_CheckedChanged(object sender, EventArgs e)

{

if (Paws2.Checked) draw += Stychinskiy.DrawPaws2; //обращение к статическому методу

else draw -= Stychinskiy.DrawPaws2;

}

private void tail\_CheckedChanged(object sender, EventArgs e)

{

if (tail.Checked) draw += DrawTail;

else draw -= DrawTail;

}

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

{

g.Clear(Color.Transparent);

gg.Clear(Color.Transparent);

pictureBox1.Image = null;

if (draw != null) draw();

pictureBox1.Image = bmp;

}

}

}

Класс Stychinskiy

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab9

{

public class Stychinskiy

{

private Graphics g = Form1.g;

private static Graphics gg = Form1.gg;

private static Stychinskiy ttt = new Stychinskiy();

//передние лапки

public void DrawPaws1() //нестатический метод

{

ttt = this;

this.g = Form1.g;

//верхняя лапка 1

Point[] Lapca1 =

{

new Point(255, 253),

new Point(217, 230),

new Point(184,218),

new Point(184,209),

new Point(188,204),

new Point(208,209),

new Point(217,220),

new Point(262,239),

new Point(255, 253),

};

g.FillPolygon(new SolidBrush(Color.Gray), Lapca1);

//верхняя лапка 2

Point[] Lapca2 =

{

new Point(439,239),

new Point(483,223),

new Point(493,214),

new Point(494,210),

new Point(498,205),

new Point(516,200),

new Point(520, 210),

new Point(516, 219),

new Point(484, 232),

new Point(443, 253),

new Point(439,239),

};

g.FillPolygon(new SolidBrush(Color.Gray), Lapca2);

}

//задние лаки

public static void DrawPaws2() //статический метод

{

gg = Form1.gg;

//нижняя лапка 3

gg.FillEllipse(Brushes.Gray, 300, 405, 50, 30);

//нижняя лапка 4

gg.FillEllipse(Brushes.Gray, 355, 405, 50, 30);

}

}

}

Результат работы программы

