Министерство образования и науки РФ

ФГБОУ ВПО «Тамбовский государственный технический университет»

Кафедра «»

Лабораторная работа №2

по дисциплине «Объектно-ориентированное программирование»

«Графический редактор»

Выполнил: студент гр..

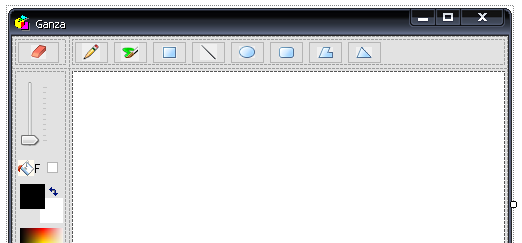
Проверил:.

Тамбов, 20

***Цели и задачи****.*

Разработать простейший графический редактор, содержащий шесть элементов управления для рисования геометрических фигур, а так же ластик.

***Решение задачи****.*

Форму делим на три области, верхняя для кнопок выбора фигур, левая для выбора режимов рисования и сам холст – в оставшейся по центру области.

Так как у фигур имеются общие черты, то их сводим в один общий класс Shape.cs, но по факту – без наполнения( абстрактный ). Общим делаем цвет линий, состояние для сигнализации о необходимости заливки и один метод для отрисовки, он так же абстрактный. Далее, путем наследования , имея один общий класс, можно приступить к созданию разных вариаций фигур, например построение линии будет следующим:

class Line:Shape

{

public Point Begin { get; set; }

public Point End { get; set; }

public Line()

{

Begin = new Point();

End = new Point();

if (\_Pen==null)

{

\_Pen = new Pen(Color.Black);

}

}

public override void Draw(System.Drawing.Graphics graphics)

{

graphics.DrawLine(\_Pen, Begin, End);

}

}

Здесь в объявлении класса через двоеточие идет унаследование класса Shape, далее два свойства для хранения точек начала и конца будущей линии, ниже описывается конструктор класса, в котором происходит инициализация начальных значений для будущего использования при создании обьекта класса. В заключении объявлен метод отрисовки с атрибутом override, тоесть данный вариант метода «перекроет» базовую реализацию, а раз базой тут является Shape, то метод будет простой реализацией абстрактного метода из базы.

***Код***

Класс Shape.cs:

using System.Drawing;

namespace Ganza

{

public abstract class Shape

{//отрисовка

public abstract void Draw(Graphics graphics);

public Pen \_Pen { get; set; }

public bool State { get; set; }//заливка контроль

}

}

Класс Triangle.cs:

using System.Collections.Generic;

using System.Drawing;

using System.Drawing.Drawing2D;

namespace Ganza

{

class Triangle:Shape

{

public List<GraphicsPath> Triangles { get; set; }

public Triangle()

{

Triangles=new List<GraphicsPath>();

if (\_Pen==null)

{

\_Pen = new Pen(Color.Black);

}

}

public override void Draw(Graphics graphics)

{//Рисуем треугольники из списка;

foreach (GraphicsPath gp in Triangles)

{

if (State) graphics.FillPath(new SolidBrush(\_Pen.Color), gp);

graphics.DrawPath(\_Pen, gp);

}

}

}

}

Класс Rect.cs:

using System.Drawing;

namespace Ganza

{

class Rect:Shape

{

public Rectangle Rectangle { get; set; }

public Rect( )

{

Rectangle = new Rectangle();

\_Pen = new Pen(Color.Black);

}

public override void Draw(System.Drawing.Graphics graphics)

{

if (State) graphics.FillRectangle(new SolidBrush(\_Pen.Color),Rectangle);

graphics.DrawRectangle(\_Pen, Rectangle);

}

}

}

Класс Elipsise.cs:

using System.Drawing;

namespace Ganza

{

class Elipsise : Shape

{

public Point Begin { get; set; }

public Point End { get; set; }

public Elipsise()

{

Begin = End = new Point();

if (\_Pen == null)

\_Pen = new Pen(Color.Black);

}

public override void Draw(System.Drawing.Graphics graphics)

{

if (State) graphics.FillEllipse(new SolidBrush(\_Pen.Color), Begin.X, Begin.Y, End.X - Begin.X, End.Y - Begin.Y);

graphics.DrawEllipse(\_Pen, Begin.X, Begin.Y, End.X - Begin.X, End.Y - Begin.Y);

}

}

}

Класс RoundRectangle.cs:

using System.Drawing;

using System.Drawing.Drawing2D;

namespace Ganza

{

class RoundRectangle:Shape

{

public Rectangle Rectangle { get; set; }

public RoundRectangle()

{

Rectangle = new Rectangle();

\_Pen = new Pen(Color.Black);

}

private void DrawRoundRect(Graphics g, Pen p, float x, float y, float width, float height, float radius)

{

GraphicsPath gp = new GraphicsPath();

gp.AddLine(x + radius, y, x + width - (radius \* 2), y); // Line

gp.AddArc(x + width - (radius \* 2), y, radius \* 2, radius \* 2, 270, 90); // Дуга

gp.AddLine(x + width, y + radius, x + width, y + height - (radius \* 2)); // Line

gp.AddArc(x + width - (radius \* 2), y + height - (radius \* 2), radius \* 2, radius \* 2, 0, 90); // Corner

gp.AddLine(x + width - (radius \* 2), y + height, x + radius, y + height); // Line

gp.AddArc(x, y + height - (radius \* 2), radius \* 2, radius \* 2, 90, 90); // Дуга

gp.AddLine(x, y + height - (radius \* 2), x, y + radius); // Line

gp.AddArc(x, y, radius \* 2, radius \* 2, 180, 90); // Дуга

gp.CloseFigure();

if(State) g.FillPath(new SolidBrush(\_Pen.Color),gp);

g.DrawPath(p, gp);

gp.Dispose();

}

public override void Draw(System.Drawing.Graphics graphics)

{

DrawRoundRect(graphics,\_Pen,Rectangle.X,Rectangle.Y,Rectangle.Width,Rectangle.Height,15);

}

}

}

Класс Karandash.cs:

using System.Drawing;

namespace Ganza

{

class Karandash:Shape

{

public Point Begin { get; set; }

public Point End { get; set; }

public Karandash()

{

Begin = new Point();

End = new Point();

if (\_Pen==null)

{

\_Pen = new Pen(Color.Black);

}

}

public override void Draw(System.Drawing.Graphics graphics)

{

graphics.DrawLine(\_Pen, Begin.X, Begin.Y, End.X, End.Y);

}

}

}

Класс Line.cs:

using System.Drawing;

namespace Ganza

{

class Line:Shape

{

public Point Begin { get; set; }

public Point End { get; set; }

public Line()

{

Begin = new Point();

End = new Point();

if (\_Pen==null)

{

\_Pen = new Pen(Color.Black);

}

}

public override void Draw(System.Drawing.Graphics graphics)

{

graphics.DrawLine(\_Pen, Begin, End);

}

}

}

Класс PolyGon.cs:

using System.Drawing;

namespace Ganza

{

class PolyGon:Shape

{

public Point Begin { get; set; }

public Point End { get; set; }

public PolyGon()

{

Begin = new Point();

End = new Point();

if (\_Pen==null)

{

\_Pen = new Pen(Color.Black);

}

}

/\* ПКМ - бросаем начальную точку

\* ЛКМ - конечная точка линии

\*/

public override void Draw(System.Drawing.Graphics graphics)

{

graphics.DrawLine(\_Pen, Begin, End);

}

}

}

Класс Form1.cs:

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Drawing.Drawing2D;

using System.Windows.Forms;

//ООП Лаб-2 Paint 6 фигур и ластик

namespace Ganza

{

public partial class FormGanza : Form

{

public FormGanza()

{

InitializeComponent();

toolTipForBtn.SetToolTip(btnPen,"Карандаш");

toolTipForBtn.SetToolTip(btnBrush, "Кисть");

toolTipForBtn.SetToolTip(btnRectangle, "Прямоугольник");

toolTipForBtn.SetToolTip(btnLine, "Линия");

toolTipForBtn.SetToolTip(btnElipsise, "Эллипсис");

toolTipForBtn.SetToolTip(btnRoundRectangle, "Скругленный прямоугольник");

toolTipForBtn.SetToolTip(btnPolyGon, "Многоугольник");

toolTipForBtn.SetToolTip(btnSterka, "Ластик");

toolTipForBtn.SetToolTip(btnTriangle, "Триугольник");

\_drawableObject = new Karandash();

StartPosition = FormStartPosition.CenterScreen;

\_bufferedGraphicsContext = BufferedGraphicsManager.Current;

InitializeGraphics();

}

private int \_key;

private Color \_colorColorSelector = Color.Black;

private Point \_movePoint, \_downPoint;//конечные координаты X-Y//начальные координаты Х-Y

private void pictureBox1\_MouseMove(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Left)

{

\_movePoint = e.Location;

if (\_drawableObject is Line)

{//отрисовка линии

Line line = (Line)\_drawableObject;

line.Begin = \_downPoint;

line.End = \_movePoint;

line.\_Pen = new Pen(\_colorColorSelector, trackBarForPenSize.Value);

}

else if (\_drawableObject is Rect)

{//отрисовка прямоугольника

Rect rect = (Rect)\_drawableObject;

rect.\_Pen = new Pen(\_colorColorSelector, trackBarForPenSize.Value);

rect.State = \_state;//true на заливку фигуры

int width, height, x, y;

//КООРДИНАТНЫЕ ОБЛАСТИ:

if (\_movePoint.X > \_downPoint.X)

{

width = \_movePoint.X - \_downPoint.X;

x = \_downPoint.X;

}

else

{

width = \_downPoint.X - \_movePoint.X;

x = \_movePoint.X;

}

if (\_movePoint.Y > \_downPoint.Y)

{

height = \_movePoint.Y - \_downPoint.Y;

y = \_downPoint.Y;

}

else

{

height = \_downPoint.Y - \_movePoint.Y;

y = \_movePoint.Y;

}

//отрисовка

rect.Rectangle = new Rectangle(x, y, width, height);

}

else if (\_drawableObject is Karandash)

{//отрисовка карандаш\кисть\ластик

Karandash karandash = (Karandash)\_drawableObject;

if (\_key != 7) karandash.\_Pen = new Pen(\_colorColorSelector, trackBarForPenSize.Value);

else karandash.\_Pen = new Pen(Color.White, trackBarForPenSize.Value);//ластик

karandash.Begin = \_downPoint;

karandash.End = \_movePoint;

\_drawableObject.Draw(\_bmpGraphics);//Запись в буфер перед обновлением холста

\_downPoint = \_movePoint; //начало=конец

}

else if (\_drawableObject is Elipsise)

{//отрисовка элипсиса

Elipsise elipsise = (Elipsise) \_drawableObject;

elipsise.\_Pen = new Pen(\_colorColorSelector, trackBarForPenSize.Value);

elipsise.State = \_state;

elipsise.Begin = \_movePoint;

elipsise.End = \_downPoint;

}

else if (\_drawableObject is RoundRectangle)

{//отрисовка прямоугольника со скруглеными углами

RoundRectangle rect = (RoundRectangle)\_drawableObject;

rect.\_Pen = new Pen(\_colorColorSelector, trackBarForPenSize.Value);

rect.State = \_state;

int width, height, x, y;

//КООРДИНАТНЫЕ ОБЛАСТИ:

if (\_movePoint.X > \_downPoint.X)

{

width = \_movePoint.X - \_downPoint.X;

x = \_downPoint.X;

}

else

{

width = \_downPoint.X - \_movePoint.X;

x = \_movePoint.X;

}

if (\_movePoint.Y > \_downPoint.Y)

{

height = \_movePoint.Y - \_downPoint.Y;

y = \_downPoint.Y;

}

else

{

height = \_downPoint.Y - \_movePoint.Y;

y = \_movePoint.Y;

}

//отрисовка

rect.Rectangle = new Rectangle(x, y, width, height);

}

if (\_drawableObject is PolyGon)

{//Многоугольник

PolyGon polyGon = (PolyGon)\_drawableObject;

polyGon.\_Pen = new Pen(\_colorColorSelector, trackBarForPenSize.Value);

polyGon.Begin = \_pointMouseUp;

/\*ПКМ - бросаем начальную точку

ЛКМ - конечная точка линии\*/

polyGon.End = \_movePoint;

}

if (\_drawableObject is Triangle)

{//Триугольник

Triangle triangle = (Triangle)\_drawableObject;

triangle.Triangles.Clear();

GraphicsPath gp = new GraphicsPath();

gp.AddPolygon(Vertexes);

triangle.\_Pen = new Pen(\_colorColorSelector, trackBarForPenSize.Value);

triangle.State = \_state;

Vertexes[0] = \_downPoint;//Запоминаем первую точку

Vertexes[1] = e.Location;//Запоминаем вторую вершину

Vertexes[2] = GetThirdVertex(\_downPoint, e.Location);//Считаем третью вершину

triangle.Triangles.Add(gp);

}

DrawToBuffer();//На обновление холста

}//end if(btn.Left)

}//end void

DrawMode \_drawMode;

private Point \_pointMouseUp;

private void pictureBox1\_MouseUp(object sender, MouseEventArgs e)

{

\_pointMouseUp = e.Location;//для полигона

switch (\_drawMode)

{//запись изменений в буфер холста

case DrawMode.Normal:

if (\_drawableObject != null)

\_drawableObject.Draw(\_bmpGraphics);

break;

}

}

private void pictureBox1\_MouseDown(object sender, MouseEventArgs e)

{//начальные координаты

//Включаем режим рисования

\_downPoint = e.Location;

}

private void colorSelect\_ForegroundColorChanged(object sender, EventArgs e)

{//Верхний цвет выбран

\_colorColorSelector = colorSelect.ForegroundColor;

}

private void colorSelect\_BackgroundColorChanged(object sender, EventArgs e)

{//Нижний цвет выбран

\_colorColorSelector = colorSelect.BackgroundColor;

}

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

{//Выбран один из элементов рисования

\_key = 0;

int k = Convert.ToInt32((sender as Button).TabIndex);

switch (k)

{

case 0://btnPen

\_drawableObject = new Karandash();

break;

case 1://btnBrush

\_drawableObject = new Karandash();

break;

case 2://btnRectangle

\_drawableObject = new Rect();

break;

case 3://btnLine

\_drawableObject = new Line();

break;

case 4://btnElipsise

\_drawableObject = new Elipsise();

break;

case 5://btnRoundRectangle

\_drawableObject = new RoundRectangle();

break;

case 6://btnPolyGon

\_key = 1;

\_drawableObject = new PolyGon();

break;

case 7://btnSterka

\_key = 7;

\_drawableObject = new Karandash();

break;

case 8://btnTriangle

\_key = 8;

\_drawableObject = new Triangle();

break;

}

}

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

{

DrawToBuffer();

}

private BufferedGraphics \_bufferedGraphics;

private Shape \_drawableObject;

Bitmap \_bitmap;

private void DrawToBuffer()

{

\_bufferedGraphics.Graphics.Clear(Color.White);//фон

if (\_drawableObject != null)

\_drawableObject.Draw(\_bufferedGraphics.Graphics);

\_bufferedGraphics.Graphics.DrawImageUnscaled(\_bitmap, 0, 0);//\_canvasRect.X, \_canvasRect.Y, \_canvasRect.Width, \_canvasRect.Height);

\_bufferedGraphics.Render();//Обновление холста

}

private Rectangle \_canvasRect;//Область для рисования

private Graphics \_gr, \_bmpGraphics;

private readonly BufferedGraphicsContext \_bufferedGraphicsContext;

private void InitializeGraphics()

{

\_canvasRect = new Rectangle(0, 0, pictureBoxHolst.Width, pictureBoxHolst.Height);//холст размер квадрата

\_gr = pictureBoxHolst.CreateGraphics();

\_bufferedGraphics = \_bufferedGraphicsContext.Allocate(\_gr, \_canvasRect);

Bitmap prevBmp = null;

if (\_bitmap != null)

prevBmp = \_bitmap;

\_bitmap = new Bitmap(ClientSize.Width+1, ClientSize.Height+1);

\_bmpGraphics = Graphics.FromImage(\_bitmap);

if (prevBmp != null)

\_bmpGraphics.DrawImage(prevBmp, 0, 0, prevBmp.Width, prevBmp.Height);

}

private void FormGanza\_Resize(object sender, EventArgs e)

{

pictureBoxHolst.Update();

InitializeGraphics();

}

#region -----Triangle-------Vertex---------

PointF[] Vertexes = new PointF[3];//Координаты вершин треугольника

private PointF GetThirdVertex(PointF FirstVertex, PointF SecondVertex)

{

//Угол наклона прямой, образованной известными вершинами

double alpha = Math.Atan((FirstVertex.Y - SecondVertex.Y) / (FirstVertex.X - SecondVertex.X));

/\* k13 - коэффициент наклона прямой, проходящей через первую и третью вершину;

\* k23 - коэффициент наклона прямой, проходящей через вторую и третью вершину;\*/

double k13 = 0, k23 = 0;

float x, y;

//Вычисление угловых коэффициентов в зависимости от угла наклона первой стороны

if (FirstVertex.Y > SecondVertex.Y)

{

k13 = Math.Tan(alpha - 2 \* Math.PI / 3);

k23 = Math.Tan(alpha - Math.PI / 3);

}

else if (FirstVertex.Y <= SecondVertex.Y)

{

k13 = Math.Tan(alpha + Math.PI / 3);

k23 = Math.Tan(alpha + 2 \* Math.PI / 3);

}

//Координата X третьей вершины

x = (float)((k13 \* FirstVertex.X - k23 \* SecondVertex.X + SecondVertex.Y - FirstVertex.Y) / (k13 - k23));

//Координата Y третьей вершины

y = (float)(k13 \* (x - FirstVertex.X) + FirstVertex.Y);

return new PointF(x, y);

}

#endregion

private void trackBarForPenSize\_MouseUp(object sender, MouseEventArgs e)

{//Изменение размера кисти выводим на btnSterka

btnSterka.Text = trackBarForPenSize.Value.ToString();

}

private bool \_state;

private void checkBoxForFill\_CheckStateChanged(object sender, EventArgs e)

{//на заливку фигуры

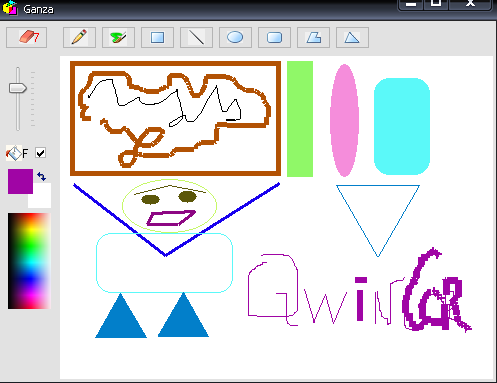
\_state = checkBoxForFill.Checked;

}

}

}

***Результат***



ОМТ

