**Лабораторна робота № 6**

Тема: Робота з графікою

## Мета: Навчитися реалізувати необхідні класи графічних об'єктів

## **Теоретичні відомості**

Графіка Windows Forms

У платформі .NET Framework реалізована вдосконалена система графічного програмування - GDI+. Вона розташовується в бібліотеці System.Drawing.dll.

Малювати на контексті пристрою дозволяє клас System.Drawing.Graphics.

Поняття контекст пристрою в Windows є концептуальним. Як графічний контекст можуть виступати: вікна, принтер, плоттер, графічний файл і т.д.

Узагальнене малювання GDI+ також включає в себе наступні основні поняття:

* колір (Color),
* перо (Pen) для малювання ліній,
* пензель (Brush) для заливання областей,
* шрифт (Font) для відображення тексту,
* точка (Point),
* розмір (Size),
* прямокутник (Rectangle) для задавання прямокутних областей відсікань,
* контур (GraphicsPath).

Наприклад, наступний код намалює лінію червоним пером шириною в один піксель:

Graphics g = ...//Одержання об'єкта Graphics.

g.DrawLine(Pens.Red, 10, 10, 200, 100);

Цей код буде однаково виконуватися для будь-якого графічного контексту, будь то принтер, форма або графічний файл. Наступні приклади дозволяють переконатися в незалежності цієї системи від пристроїв.

**Приклад простої програми для малювання на pictureBox:**

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 WindowsFormsApplication1

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

// кнопка «Намалювати»

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

{

Graphics g = pictureBox1.CreateGraphics();

g.DrawLine(Pens.Red, 200, 200, 100, 20);

g.DrawEllipse(Pens.Blue, 50, 50, 100, 150);

g.DrawRectangle(Pens.Green, 200, 100, 102, 100);

}

}

}

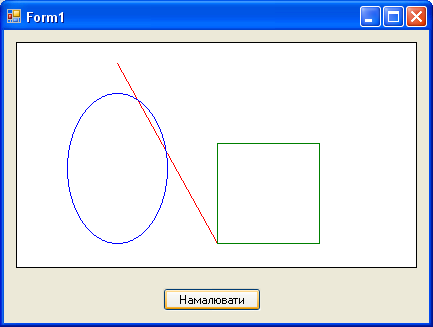


Рисунок 6.1 – Приклад скриншоту простої програми для виведення графіки

Колір

Кольори в Windows Forms засновані на моделі ARGB ( red-green-blue). Кольори визначаються однобайтовими значеннями червоного, зеленого й синього. Альфа-канал визначає прозорість кольору. Значення alpha лежать у діапазоні від 0 для прозорого й до 255 для непрозорого. Для задання кольору необхідно вказати його складові або скористатися визначеними кольорами.

Таблиця 6.1. Приклади задання кольору

|  |  |
| --- | --- |
| **Приклад** | **Пояснення** |
| Color c = Color.FromArgb(127, 0, 255, 0); | Зелений напівпрозорий колір |
| Color c1 = Color.Green;  Color c2 = Color.Aqua;  ... | Визначені кольори.  Усього 141 |
| Color c1 = SystemColors.ActiveBorder;  Color c2 = SystemColors.MenuText;  ... | Системні кольори.  Усього 33 |

Системні кольори можна змінити в панелі керування Windows.

Пензель

Таблиця 6.2. Приклади заливання областей пензлем

|  |  |
| --- | --- |
| **Приклад** | **Пояснення** |
| Brush brush = new SolidBrush(Color.Red); | Суцільний червоний пензель |
| Brush brush1 = new LinearGradientBrush(point1, point2, c1, c2); | Градієнтний пензель на основі двох точок і двох кольорів |
| Brush brush2 = new HatchBrush(hs, c1, c2); | Виконує штрихування заданими двома кольорами. Змінна hs має тип перерахування HatchStyle:  Image24 |

Перо

Таблиця 6.3. Приклади малювання ліній або кривих пером

|  |  |
| --- | --- |
| **Приклад** | **Пояснення** |
| Pen pen = new Pen(c1); | Перо зеленого кольору товщиною 1 піксель |
| Pen pen1 = new Pen(c1, 5); | Перо зеленого кольору товщиною 5 пікселів |
| Pen pen2 = new Pen(brush1); | Перо на основі градієнтного пензля |

Шрифт

Таблиця 6.4. Приклади виведення тексту (основні властивості Name і Size)

|  |  |
| --- | --- |
| **Приклад** | **Пояснення** |
| Font font = new Font("Arial", 10); | Шрифт Arial розміром 10 |

Точка, Розмір, Прямокутник

Точку на площині характеризує структура Point. Її основні властивості X і Y.

Структура Size призначена для завдання розміру. Вона містить властивості для висоти (Height) і ширини (Width).

Структуру Rectangle можна розглядати як об'єднання точки й розміру.

Методи класу Graphics для малювання

Функції малювання починаються із префікса Draw або Fill. Методи Draw малюють прямі й криві; методи Fill служать для заливання областей.

Нехай задані опорні точки, пензель, перо, шрифт:

Pen pen = new Pen(Color.Red, 2);

int startAngle = 0; //Початковий кут у градусах для малювання дуг.

int sweepAngle = -135; //Кінцевий кут для малювання дуг.

int x = 5, y = 5, width = 100, height = 50;

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

Point point = new Point(x, y);

Point pt1 = new Point(7, 50), pt2 = new Point(35, 7);

Point pt3 = new Point(100, 7), pt4 = new Point(100, 50);

Point[] points = new Point[] { pt1, pt2, pt3, pt4 };

Image image = Image.FromFile("H:\\PrintHS.png");//Завантаження зображення з файлу.

string s = "Текст";

Brush brush = new SolidBrush(Color.DarkGray);

Font font = new Font("Times New Roman", 14);

Таблиця 6.5. Приклади малювання фігур та деяких інших об’єктів

|  |  |
| --- | --- |
| **Приклад** | **Результат** |
| Дуга  g.DrawArc(pen, rect, startAngle, sweepAngle);  або  g.DrawArc(pen, x, y, width, height, startAngle, sweepAngle);  Позитивний напрямок виміру дуги - за годинниковою стрілкою | Image5 |
| Сплайн Безье  g.DrawBezier(pen, pt1, pt2, pt3, pt4); | Image6 |
| Канонічний сплайн  g.DrawCurve(pen, points); | Image7 |
| Замкнутий канонічний сплайн  g.DrawClosedCurve(pen, points); | Image8 |
| Еліпс  g.DrawEllipse(pen, rect);  або  g.DrawEllipse(pen, x, y, width, height); | Image9 |
| Зображення  g.DrawImage(image, point);  або  g.DrawImage(image, x, y); | Image10 |
| Лінія  g.DrawLine(pen, pt1, pt3);  або  g.DrawLine(pen, pt1.X, pt1.Y, pt3.X, pt3.Y); | Image11 |
| Зчленовані лінії  g.DrawLines(pen, points); | Image12 |
| Сектор  g.DrawPie(pen, rect, startAngle, sweepAngle);  або  g.DrawPie(pen, x, y, width, height, startAngle, sweepAngle); | Image13 |
| Полігон  g.DrawPolygon(pen, points); | Image14 |
| Прямокутник  g.DrawRectangle(pen, rect);  або  g.DrawRectangle(pen, x, y, width, height); | Image15 |
| Текст  g.DrawString(s, font, brush, x, y); | Image16 |
| Заливання замкнутого канонічного сплайна  g.FillClosedCurve(brush, points); | Image17 |
| Заливання еліпса  g.FillEllipse(brush, rect);  або  g.FillEllipse(brush, x, y, width, height); | Image18 |
| Заливання сектора  g.FillPie(brush, rect, startAngle, sweepAngle);  або  g.FillPie(brush, x, y, width, height, startAngle, sweepAngle); | Image19 |
| Заливання полігона  g.FillPolygon(brush, points); | Image20 |
| Заливання прямокутника  g.FillRectangle(brush, rect);  або  g.FillRectangle(brush, x, y, width, height); | Image21 |

Контур

Контур - набір точок координат, що описує сукупність прямих і кривих. У контур можна додавати прямі й криві, використовуючи методи із префіксом Add. Ці методи аналогічні методам із префіксом Draw класу Graphics.

Таблиця 6.6. Приклади застосування контуру

|  |  |
| --- | --- |
| **Приклад** | **Результат** |
| Відображення контуру  GraphicsPath path = new GraphicsPath();  path.AddLine(pt1, pt2);  path.AddLine(pt3, pt4);  g.DrawPath(pen, path); | Image22 |
| Заливання контуру  g.FillPath(brush, path); | Image23 |

## **Завдання до лабораторної роботи:**

Намалювати у вікні Windows-додатку, методами бібліотеки System.Drawing.dll, наступні об’єкти:

- прямокутник, залитий суцільним пензлем;

- еліпс, залитий суцільним пензлем із границею;

- коло, залите суцільним пензлем із границею;

- будиночок, залитий суцільною кистю із границею (висота даху не менше 50 пікселів);

- заштрихований будиночок;

- коло, зафарбоване чорно-білими квадратами;

- квітку, що складається з жовтого кола та блакитних еліпсів;

- конверт, залитий суцільною кистю із границею (висота кришки конверта не менше 30 пікселів, відношення ширини до висоти 5 до 3, ширина кратна 5);

- заштрихований конверт.

***Контрольні питання:***

1. В який бібліотеці С# розташовані засоби роботи з графікою?

2. Які основні поняття включає в себе малювання в С#?

3. Для чого служить інструмент Пензель?

4. Яким чином можна намалювати прямокутник?

5. Яким чином можна намалювати еліпс?