

LẬP TRÌNH GIAO DIỆN

Nguyễn Thị Mai Trang

1

1

Chương 8

Đồ họa

2

2

MỤC TIÊU

- Trình bày được các thành phần quan trọng trong thư viện GDI+ được sử dụng để vẽ
- Sử dụng các đối tượng thuộc lớp Graphics để vẽ đường, vẽ và tô màu các đối tượng hình học, hiển thị văn bản và hình ảnh.
- Xây dựng được ứng dụng vẽ bằng chuột

3

3

NỘI DUNG

1. Giới thiệu về GDI+
2. Một số đối tượng GDI+
3. Lớp Graphics
4. Làm việc với các đối tượng đồ họa
5. Vẽ, tô màu các đối tượng cơ bản

4

4

8.1 Giới thiệu về GDI+

- Được phát triển từ GDI (**G**raphics **D**evice **I**nteface), chỉ có trong .NET Framework.
- GDI+ là một thư viện cung cấp các lớp cho phép làm việc với các đối tượng đồ họa 2D như:
 - Vẽ, tô màu đối tượng
 - Hiển thị văn bản (vẽ chữ)
 - Vẽ hình ảnh, biến đổi hình ảnh.
 - Thư viện .NET Framework chứa các lớp liên quan đến thao tác vẽ trong không gian tên System.Drawing.

5

5

Giới thiệu về GDI+ (tt)

- GDI+ cung cấp các lớp thư viện cho phép thực hiện tiến trình vẽ, bao gồm các bước:
 - Xác định phạm vi (bề mặt) để vẽ
 - Hệ thống tọa độ
 - Các cấu trúc dữ liệu như Rectangle, Point, Size
 - Tạo các công cụ để vẽ
 - Cọ tô (Brush)
 - Bút vẽ (Pen)
 - Phong chữ (Font)
 - Thực hiện thao tác vẽ, tô màu
 - Lớp Graphics

6

6

Xác định phạm vi để vẽ

- Chiều rộng, chiều cao: xác định vị trí, kích thước bề mặt để vẽ
- Độ phân giải: số điểm ảnh theo chiều ngang và chiều dọc của màn hình
- Độ sâu màu: số lượng màu sắc được sử dụng cho mỗi điểm ảnh
- Điểm ảnh: pixel, là đơn vị nhỏ nhất tham gia vào quá trình hiển thị đối tượng, gồm 3 thành phần đỏ, xanh lá, xanh dương (RGB)

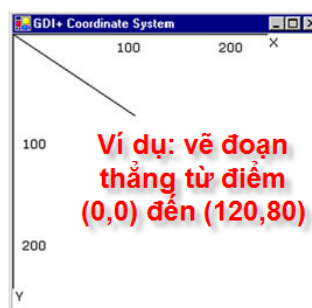
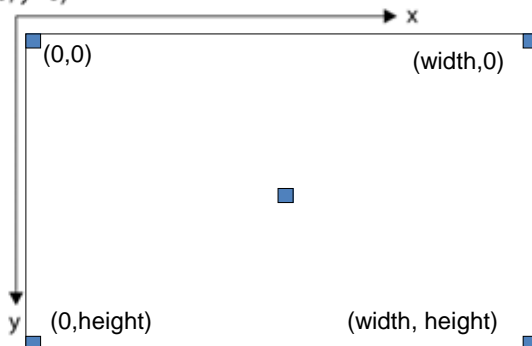
7

7

Xác định phạm vi để vẽ

- Hệ thống tọa độ (Coordinate System)

($x=0$, $y=0$)

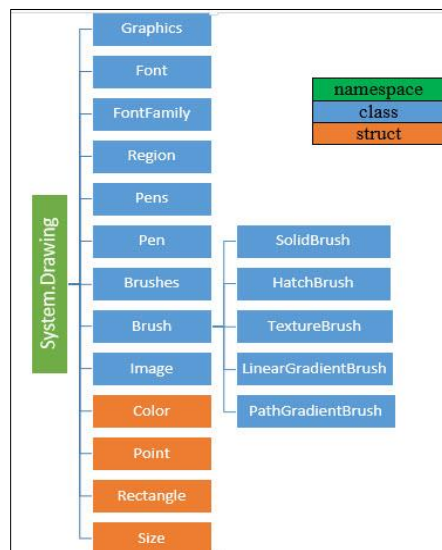


8

8

Tạo các công cụ để vẽ

- Cọ tô (Brush)
- Bút vẽ (Pen)
- Phong chữ (Font)
- Màu sắc (cấu trúc Color)



9

Thực hiện các thao tác vẽ và tô màu

- Sử dụng các phương thức thuộc lớp Graphics

– Draw...

FontConverter,FontUnitConverter
Class

– Fill...

FontFamily Class

FontStyle Enumeration

Graphics Class

Graphics Methods

AddMetafileComment Method

BeginContainer Method

Clear Method

CopyFromScreen Method

Dispose Method

DrawArc Method

DrawBezier Method

DrawBeziers Method

DrawClosedCurve Method

DrawCurve Method

DrawEllipse Method

Graphics Class

.NET Framework (current version) | Other Versions

Encapsulates a GDI+ drawing surface. This class cannot be inherited.

Namespace: System.Drawing

Assembly: System.Drawing (in System.Drawing.dll)

Inheritance Hierarchy

System.Object

System.MarshalByRefObject

System.Drawing.Graphics

Syntax

```
C# C++ F# VB
public sealed class Graphics : MarshalByRefObject
    IDeviceContext
```

Properties

10

10

8.2 Một số đối tượng GDI

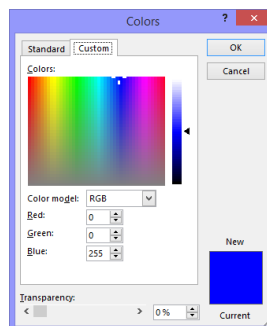
- Color
- Point
- Rectangle
- Size

11

11

Color



- Là một cấu trúc dữ liệu thể hiện màu sắc, là sự kết hợp giữa 4 giá trị:
 - R: Red
 - G: Green
 - B: Blue
 - A: Alpha: độ trong suốt của màu.
- Giá trị mỗi thành phần từ 0-255
 - $2^8 2^8 2^8 2^8 = 2^{32} \rightarrow 32$ bit



12

12

Color

- Tạo đối tượng Color từ các giá trị ARGB: sử dụng phương thức `Color.FromArgb`
 - Ví dụ: tạo đối tượng Color màu đỏ
`Color red = Color.FromArgb(255, 0, 0);` 
 - Lưu ý: nếu ta bỏ qua giá trị A, mặc định A = 255
- Tạo đối tượng Color từ một chuỗi tên màu xác định: sử dụng phương thức `Color.FromName`
 - Ví dụ: tạo đối tượng Color màu xanh dương
`Color blue = Color.FromName ("Blue");` 

13

13

Point

- Là cấu trúc dữ liệu xác định một điểm trong mặt phẳng với hai thuộc tính x, y
- Có thể tạo một đối tượng Point thông qua các phương thức khởi tạo sau:
 - `public Point(int);`
 - ví dụ: `Point pt1 = new Point(10);`
 - `public Point(Size);`
 - ví dụ: `Point pt2 = new Point(new Size(20, 20));`
 - `public Point(int, int);`
 - ví dụ: `Point pt3 = new Point(30, 30);`

14

14

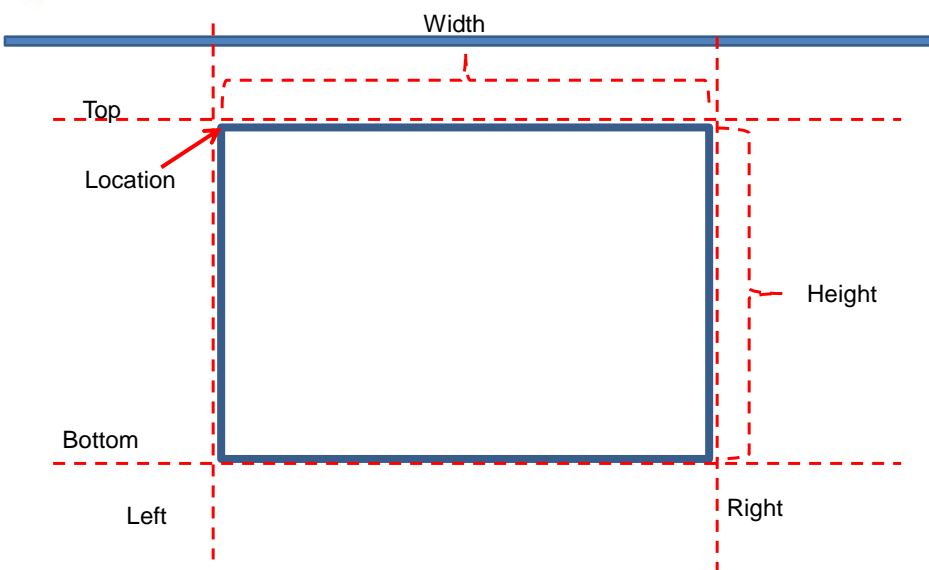
Rectangle

- Là một cấu trúc dữ liệu xác định vị trí, kích thước một vùng hình chữ nhật trong mặt phẳng
- Các thuộc tính của Rectangle:
 - **Left, Right**: giá trị x của tọa độ điểm bên trái và phải của HCN
 - **Top, Bottom**: giá trị y của tọa độ điểm bên trên và dưới của HCN
 - **Width, Height**: chiều rộng, cao của HCN
 - **X, Y**: giá trị x, y (int) của tọa độ điểm góc trên bên trái của HCN
 - **Location**: tọa độ điểm (Point) góc trên bên trái của HCN
 - **IsEmpty**: kiểm tra HCN rỗng (tọa độ 4 điểm đều bằng 0: True)
 - **Size**: trả về đối tượng Size cho biết kích thước của HCN

15

15

Rectangle



16

16

Rectangle (tt)

- Các phương thức
 - Contains: kiểm tra một điểm có nằm trong HCN?
 - Inflate: thay đổi kích thước HCN theo các chiều
 - Offset: thay đổi vị trí HCN
 - Ceiling: chuyển RectangleF → Rectangle với các giá trị tọa độ được làm tròn lên
 - Round: chuyển RectangleF → Rectangle với các giá trị tọa độ được làm tròn
 - Truncate: chuyển RectangleF → Rectangle với các giá trị tọa độ được làm tròn xuống

17

17

Size

- Là một cấu trúc dữ liệu xác định kích thước một vùng hình chữ nhật, với hai thuộc tính Width, Height
- Ví dụ:
 - Size sz = new SizeF(100,80);

18

18

8.3 Lớp Graphics

- Là thành phần chính của GDI+, cung cấp các tài nguyên và phương pháp thao tác với các đối tượng đồ họa
- **Một số thuộc tính lớp Graphics**
 - **Clip**: get/set phạm vi bản vẽ
 - **ClipBounds**: trả về cấu trúc Rectangle là phạm vi bản vẽ
 - **DpiX, DpiY**: trả về độ phân giải của đối tượng đồ họa tính theo inch
 - **PageUnit**: get/set đơn vị hệ thống tọa độ
 - **SmoothingMode**: get/set chế độ làm mịn của đối tượng đồ họa:
 - AntiAlias: chống răng cưa
 - HighQuality: chất lượng cao
 - HighSpeed: tốc độ cao

19

19

Lớp Graphics (tt)

- **Tạo đối tượng Graphics**
 - Sử dụng thuộc tính Graphics được truyền cho **Paint ()**
Ví dụ: vẽ hình chữ nhật màu đỏ

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    e.Graphics.DrawRectangle (Pens.Red,10,10,100,60);
}
```

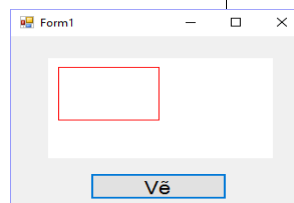
20

20

Lớp Graphics (tt)

- Sử dụng phương thức **CreateGraphics()** của form, control
- Ví dụ: click vào button Vẽ để vẽ hình chữ nhật trên panel

```
private void btDraw_Click(object sender, EventArgs e)
{
    DrawInControl(panel1);
}
1 reference
private void DrawInControl(Control c)
{
    Graphics g = c.CreateGraphics();
    g.DrawRectangle(Pens.Red, 10, 10, 100, 60);
}
```



21

21

Lớp Graphics (tt)

• Tạo đối tượng Graphics

- Lấy từ đối tượng dẫn xuất từ Bitmap để vẽ trên hình.
- Ví dụ: vẽ một hình chữ nhật lên đối tượng Image load từ tập tin hình.jpg

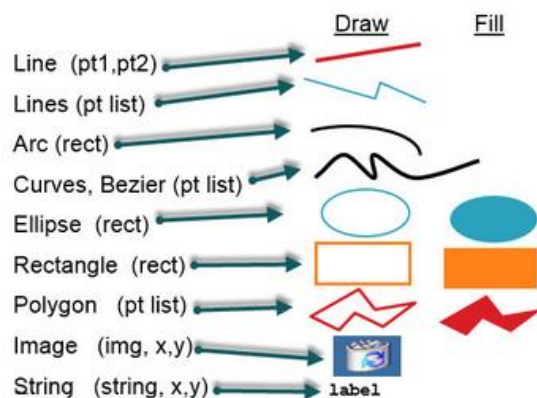
```
private void DrawInImage()
{
    Image img = Image.FromFile("hinh.jpg");
    Graphics g = Graphics.FromImage(img);
    g.DrawRectangle(Pens.Red, 10, 10, 100, 60);
}
```

22

22

Lớp Graphics (tt)

- Một số phương thức vẽ và tô màu của lớp Graphics



23

23

8.4 Làm việc với các đối tượng đồ họa

- Brushes
- Pens
- Font

24

24

8.4.1 Brushes

- Brushes là đối tượng được sử dụng để tô màu vùng bên trong của hình, tô màu văn bản, thường được kết hợp trong các phương thức **Fill...** của lớp Graphics
- Được định nghĩa trong không gian tên
 - System.Drawing: Brushes, SolidBrush
 - System.Drawing.Drawing2D:
 - HatchBrush
 - TextureBrush
 - LinearGradientBrush
 - PathGradientBrush

25

25

Brushes (tt)

- Bao gồm các lớp:
 - Brushes
 - Brush
 - SolidBrush
 - HatchBrush
 - TextureBrush
 - LinearGradientBrush
 - PathGradientBrush

26

26

class Brushes

- Là một lớp không được kế thừa (sealed class)
- Cung cấp hơn 140 thuộc tính tô màu chuẩn (standard colors)

```
namespace System.Drawing
{
    public sealed class Brushes
    {
        public static Brush AliceBlue { get; }
        public static Brush AntiqueWhite { get; }
        public static Brush Aqua { get; }
        public static Brush Aquamarine { get; }
        public static Brush Azure { get; }
        public static Brush Beige { get; }
        public static Brush Bisque { get; }
        public static Brush Black { get; }
```

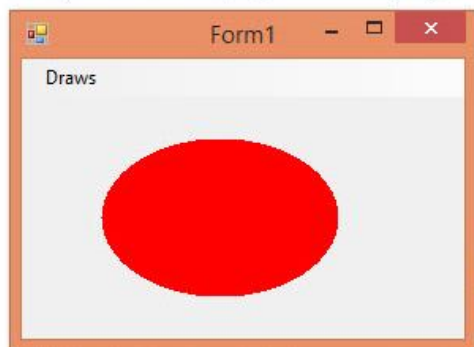
27

27

Brushes (tt)

- Ví dụ sử dụng class Brushes: tô màu đỏ cho một ellipse

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    e.Graphics.FillEllipse(Brushes.Red, 50, 50, 150, 100);
```



ng mode

```
s;
) to (100, 100
00, 100);
```

28

28

Solid Brushes

- Là loại cọ tô một màu đồng nhất
- Tạo SolidBrush: sử dụng phương thức khởi tạo
- Ví dụ:

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    Color color = Color.FromArgb(255, 100, 50);
    SolidBrush brush = new SolidBrush(color);
    e.Graphics.FillRectangle(brush, 50, 40, 150, 80);
}
```



29

29

Hatch Brushes

- Là loại cọ tô có hiệu ứng nền
- Hai phương thức khởi tạo:
 - public HatchBrush(HatchStyle, Color);
 - public HatchBrush(HatchStyle, Color, Color);
- Trong đó, HatchStyle là các kiểu nền

30

30

Hatch Brushes (tt)

- HatchStyle:



31

31

Hatch Brushes (tt)

- Ví dụ

```
private void FrmHatchBrush_Paint(object sender, PaintEventArgs e)
{
    //khởi tạo vị trí bắt đầu vẽ
    int x = 20;
    int y = 20;
    HatchBrush myBrush;
    foreach (HatchStyle brushStyle in//duyet qua các HatchStle
        System.Enum.GetValues(typeof(HatchStyle)))
    {
        //với mỗi HatchStyle tạo một HatchBrush
        myBrush = new HatchBrush(brushStyle, Color.Blue, Color.Red );
        e.Graphics.FillRectangle(myBrush, x, y, 40, 20);
        e.Graphics.DrawString(brushStyle.ToString(),
            new Font("Tahoma", 8), Brushes.Black, 50 + x, y + 5);
        y += 30;//tăng y để vẽ xuống
        if ((y + 30) > this.ClientSize.Height)
        {
            //nếu vượt quá chiều cao vùng Client thì vẽ sang cột khác
            y = 20;
            x += 180;
        }
    }
}
```

32

32

Texture Brushes

- Là loại cọ tô có nền là một đối tượng Image
- Lớp này có các thuộc tính:
 - Image: ảnh nền
 - WrapMode: Clamp, Tile, TileFlipX, TileFlipY, TileFlipXY
- Ví dụ:

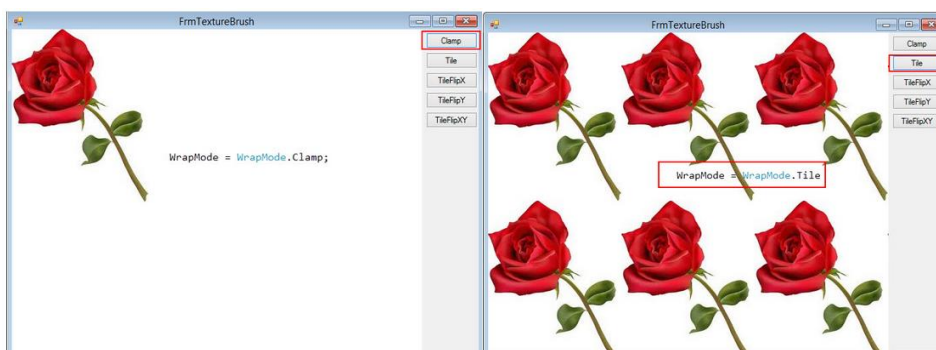
```
private void FrmTextureBrush_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;
    Image img = Image.FromFile("rose.jpg");
    TextureBrush txtrBrush = new TextureBrush(img);
    g.FillRectangle(txtrBrush, ClientRectangle);
}
```

33

33

Texture Brushes (tt)

- Các kiểu WrapMode

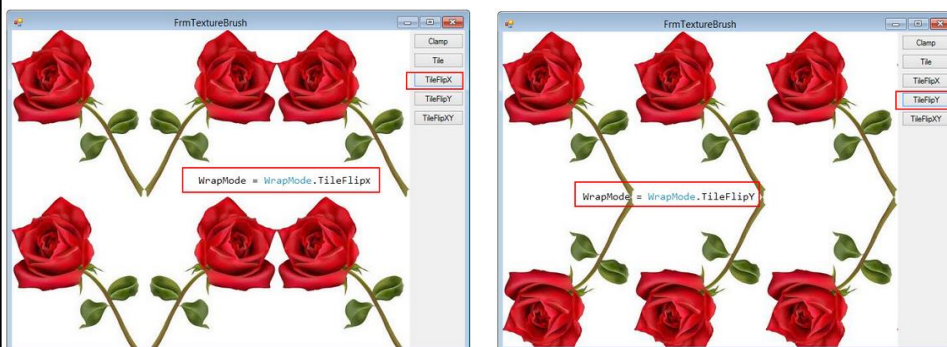


34

34

Texture Brushes (tt)

- Các kiểu WrapMode

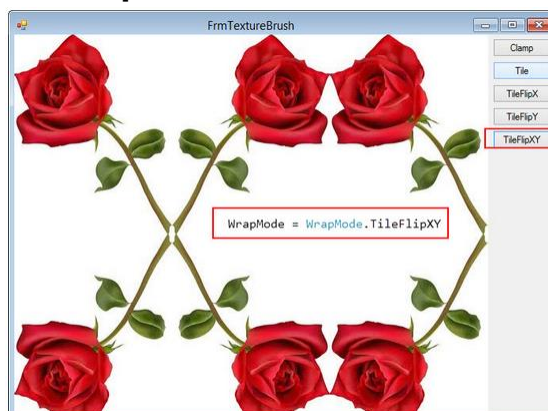


35

35

Texture Brushes (tt)

- Các kiểu WrapMode



36

36

Linear Gradient Brushes

- Là loại cọ tô pha trộn giữa hai màu
- Các phương thức khởi tạo:
 - LinearGradientBrush (Point, Point, Color, Color)
 - LinearGradientBrush (PointF, PointF, Color, Color)
 - LinearGradientBrush (Rectangle, Color, Color, LinearGradientMode)
 - LinearGradientBrush (Rectangle, Color, Color, Single)
 - LinearGradientBrush (Rectangle, Color, Color, Single, Boolean)
 - LinearGradientBrush (RectangleF, Color, Color, LinearGradientMode)
 - LinearGradientBrush (RectangleF, Color, Color, Single)
 - LinearGradientBrush (RectangleF, Color, Color, Single, Boolean)

37

37

Linear Gradient Brushes

- Trong đó:
 - LinearGradientMode là hướng pha trộn, gồm các giá trị sau:
 - Horizontal
 - Vertical
 - BackwardDiagonal
 - ForwardDiagonal
 - angle: góc pha trộn.
 - Point, PointF, Rectangle, RectangleF: kích thước của brush

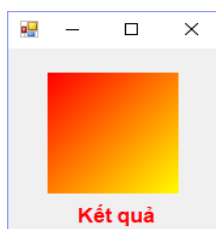
38

38

Linear Gradient Brushes (tt)

- Ví dụ:

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    Rectangle rect = new Rectangle(30,20, 100, 100);
    LinearGradientBrush brush =
        new LinearGradientBrush(rect, Color.Red, Color.Yellow, 45);
    e.Graphics.FillRectangle(brush, rect);
}
```



39

39

Path Gradient Brushes

- Là loại cọ tô được tạo từ một đối tượng GraphicsPath, có thể pha trộn nhiều màu

```
private void FrmPathGradient_Paint(object sender, PaintEventArgs e)
{
    Graphics g = e.Graphics;
    GraphicsPath gp = new GraphicsPath();
    gp.AddLine(10, 10, 110, 15);
    gp.AddLine(110, 15, 100, 95);
    gp.AddLine(100, 95, 15, 110);
    gp.CloseFigure();
    g.FillRectangle(Brushes.White, this.ClientRectangle);
    g.SmoothingMode = SmoothingMode.AntiAlias;
    PathGradientBrush pgb = new PathGradientBrush(gp);
    pgb.CenterColor = Color.White;
    pgb.SurroundColors = new Color[] { Color.Red, Color.Yellow, Color.Orange };
    g.FillPath(pgb, gp);
    g.DrawPath(Pens.Black, gp);
    pgb.Dispose();
    gp.Dispose();
}
```



40

40

8.4.2 Pens

- Pen - còn gọi là bút vẽ, là đối tượng được dùng để vẽ đường thẳng, đường cong, đường viền cho các đối tượng,
- Pen thường được kết hợp trong các phương thức **Draw...** của lớp Graphics
- Có hai lớp bút vẽ: Pens và Pen.

41

41

Pens (tt)

- Class Pens: là một lớp không cho phép kế thừa, cung cấp loại bút vẽ có độ dày bằng 1 pixel với các màu chuẩn

```
namespace System.Drawing
{
    public sealed class Pens
    {
        public static Pen AliceBlue { get; }
        public static Pen AntiqueWhite { get; }
        public static Pen Aqua { get; }
        public static Pen Aquamarine { get; }
        public static Pen Azure { get; }
        public static Pen Beige { get; }
        public static Pen Bisque { get; }
        public static Pen Black { get; }
        public static Pen BlanchedAlmond { get; }
```

42

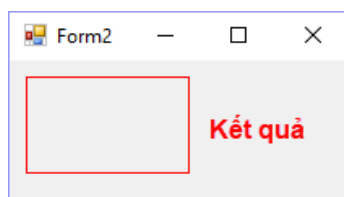
42

Pens (tt)

- Class Pens:

- Ví dụ sử dụng class Pens để vẽ một khung hình chữ nhật màu đỏ:

```
private void Form2_Paint(object sender, PaintEventArgs e)
{
    e.Graphics.DrawRectangle(Pens.Red, 10, 10, 100, 60);
}
```



43

43

Pen (tt)

- class Pen: cho phép tạo đối tượng Pen dựa vào các phương thức khởi tạo sau:
 - public Pen ([Brush](#) brush) : tạo đối tượng Pen từ đối tượng Brush, với độ dày nét vẽ là 1 pixel.
 - public Pen ([Color](#) color): tạo đối tượng Pen từ đối tượng Color, với độ dày nét vẽ là 1 pixel.
 - public Pen([Brush](#) brush, float width) : tạo đối tượng Pen từ đối tượng Brush, với độ dày nét vẽ là width.
 - public Pen([Color](#) color, float width) : tạo đối tượng Pen từ đối tượng Color, với độ dày nét vẽ là width.

44

44

Pen (tt)

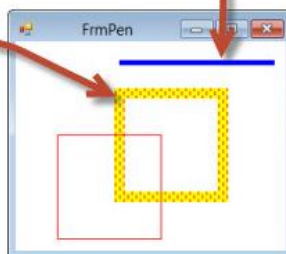
```
private void FrmPen_Paint(object sender, PaintEventArgs e)
{
    Graphics g = this.CreateGraphics();
    g.Clear(this.BackColor);

    SolidBrush blueBrush = new SolidBrush(Color.Blue);
    Pen pn1 = new Pen(blueBrush, 5);
    g.DrawLine(pn1, new Point(100, 20), new Point(250, 20));

    HatchBrush hatchBrush = new HatchBrush(HatchStyle.DashedVertical,
        Color.Red, Color.Yellow);
    Pen pn2 = new Pen(hatchBrush, 10);
    g.DrawRectangle(pn2, 100, 50, 100, 100);

    Pen pn3 = new Pen(Color.Red);
    g.DrawRectangle(pn3, 40, 90, 100, 100);

    pn1.Dispose();
    pn2.Dispose();
    pn3.Dispose();
    blueBrush.Dispose();
    hatchBrush.Dispose();
    g.Dispose();
}
```



45

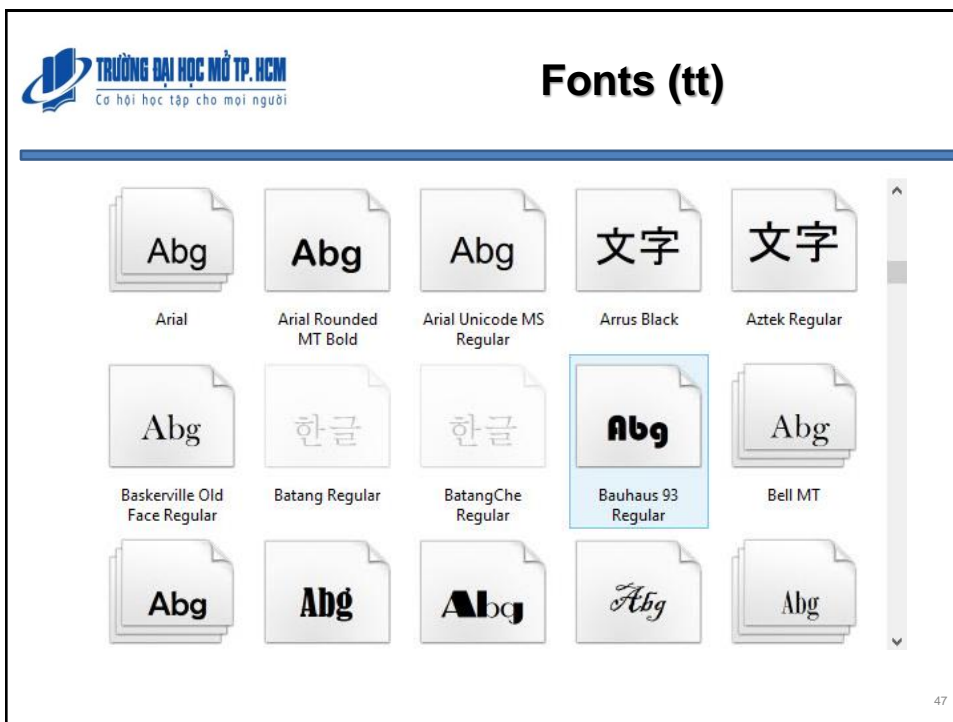
45

8.4.3 Fonts


- Windows hỗ trợ hai loại phông chữ: GDI fonts và device fonts.
 - Device fonts** có nguồn gốc từ các thiết bị đầu ra như màn hình, máy in,...
 - GDI fonts** được lưu trữ trong các tập tin trên hệ thống, thường chứa trong thư mục Windows\Fonts.
- Mỗi font có tập tin riêng, ví dụ, Arial, Arial Black, Arial Bold, Arial Italic, Italic Arial Black, Arial Bold Italic, Arial Narrow, Italic Arial Narrow Bold, và Arial Narrow Italic,...

46

46



47



Fonts (tt)

- Thư viện GDI+ cung cấp các lớp font cho phép thiết lập font chữ khi xuất văn bản là **Font** và **FontFamily**
- **class FontFamily**: cung cấp các loại font chữ được định nghĩa sẵn để sử dụng trong kết xuất văn bản, lớp này chứa các thuộc tính như
 - **Name**: tên font chữ
 - **Families**: mảng chứa tất cả các font families trong thiết bị ngữ cảnh

48

Fonts (tt)

- **class Font:** là lớp cho phép tạo đối tượng font chữ với một số thuộc tính như kiểu chữ (đậm, nghiêng, gạch dưới,...), kích thước,...
 - Một số phương thức khởi tạo thường được sử dụng:
 - [Font \(FontFamily, Single\)](#)
 - [Font \(String, Single\)](#)
 - [Font \(FontFamily, Single, FontStyle\)](#)
 - [Font \(String, Single, FontStyle\)](#)

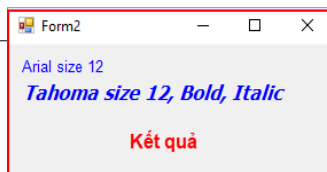
49

49

Font (tt)

– Ví dụ sử dụng Font

```
private void Form2_Paint(object sender, PaintEventArgs e)
{
    Font arial = new Font("Arial", 10);
    e.Graphics.DrawString("Arial size 12", arial,
                          Brushes.Blue, 10, 10);
    FontStyle style = FontStyle.Bold | FontStyle.Italic;
    FontFamily family = new FontFamily("Tahoma");
    Font tahoma = new Font(family, 12, style);
    e.Graphics.DrawString("Tahoma size 12, Bold, Italic",
                          tahoma, Brushes.Blue, 10, 30);
}
```



50

50

8.5 Vẽ, tô màu các đối tượng

- Vẽ đường thẳng
 - Vẽ, tô màu các đối tượng hình học
 - Hình chữ nhật
 - Ellipse
 - Đa giác
 - Vẽ Image
 - Tô màu các đối tượng
 - Vẽ chữ (văn bản)
 - Vẽ bằng chuột
- ☞ Sử dụng các phương thức thuộc lớp Graphics

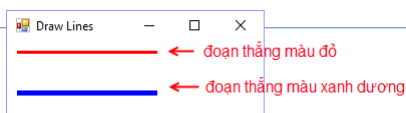
51

51

Vẽ đường thẳng

- public static DrawLine (Pen pen, Point pStart, Point pEnd)
- public static DrawLine (Pen pen, int x1, int y1, int x2, int y2)

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    //tạo bút vẽ màu đỏ, độ dày nét 3 pixel
    Pen RedPen = new Pen(Color.Red, 3);
    e.Graphics.DrawLine(RedPen, 10, 10, 150, 10);
    Point p1 = new Point(10, 50);
    Point p2 = new Point(150, 50);
    //tạo bút vẽ màu xanh dương, độ dày nét 5 pixel
    Pen BluePen = new Pen(Color.Blue, 5);
    e.Graphics.DrawLine(BluePen, p1, p2);
}
```



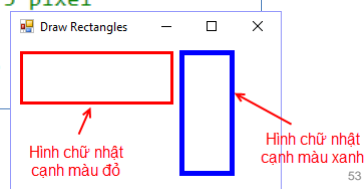
52

52

Vẽ hình chữ nhật

- `public static DrawRectangle (Pen pen, int x, int y, int width, int height)`
- `public static DrawRectangle (Pen pen, Rectangle rect)`

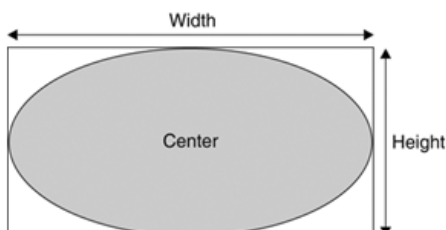
```
private void FrmRectangle_Paint(object sender, PaintEventArgs e)
{
    //tạo bút vẽ màu đỏ, độ dày nét 3 pixel
    Pen RedPen = new Pen(Color.Red, 3);
    e.Graphics.DrawRectangle(RedPen, 10, 10, 150, 50);
    Rectangle rect = new Rectangle(170, 10, 50, 120);
    //tạo bút vẽ màu xanh dương, độ dày nét 5 pixel
    Pen BluePen = new Pen(Color.Blue, 5);
    e.Graphics.DrawRectangle(BluePen, rect);
}
```



53

Vẽ Ellipses và hình tròn

- Ellipse là một hình nội tiếp trong một vùng hình chữ nhật
- Để vẽ một Ellipse, cần xác định hình chữ nhật ngoại tiếp nó
- Ellipse nội tiếp trong một hình vuông là một hình tròn



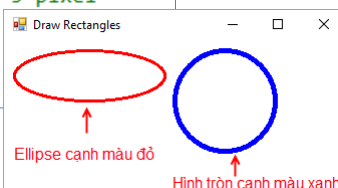
54

54

Vẽ Ellipses và hình tròn

- `public static DrawEllipse (Pen pen, int x, int y, int width, int height)`
- `public static DrawEllipse (Pen pen, Rectangle rect)`

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    //tạo bút vẽ màu đỏ, độ dày nét 3 pixel
    Pen RedPen = new Pen(Color.Red, 3);
    e.Graphics.DrawEllipse(RedPen, 10, 10, 150, 50);
    Rectangle rect = new Rectangle(170, 10, 100, 100);
    //tạo bút vẽ màu xanh dương, độ dày nét 5 pixel
    Pen BluePen = new Pen(Color.Blue, 5);
    e.Graphics.DrawEllipse(BluePen, rect);
}
```

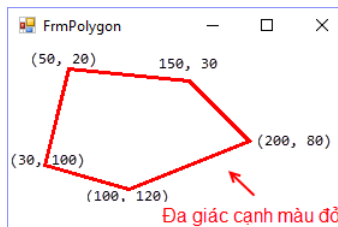


55

Vẽ đa giác

- `public static DrawPolygon(Pen pen, Point [] points)`

```
private void FrmPolygon_Paint(object sender, PaintEventArgs e)
{
    Point p1 = new Point(50, 20);
    Point p2 = new Point(150, 30);
    Point p3 = new Point(200, 80);
    Point p4 = new Point(100, 120);
    Point p5 = new Point(30, 100);
    Point[] arrPoint = { p1, p2, p3, p4, p5 };
    Pen RedPen = new Pen(Color.Red, 3);
    e.Graphics.DrawPolygon(RedPen, arrPoint);
}
```



56

Vẽ Image

- Net Framework cung cấp lớp Image cho phép trình bày hình ảnh.
- Một số phương thức thông dụng của lớp Image:
 - **FromFile (string)**: phương thức static cho phép tạo ảnh từ đường dẫn đến một file (bmp, jpeg, jpg, gif, png, ico,...)
 - **Save (string, ImageFormat)**: lưu ảnh thành file
 - **RotateFlip (RotateFlipType)**: xoay, lật ảnh.
 - Lưu ý: để sử dụng **ImageFormat**, ta cần phải khai báo namespace **System.Drawing.Imaging**
- Để vẽ Image, sử dụng Phương thức **DrawImage** của lớp Graphics

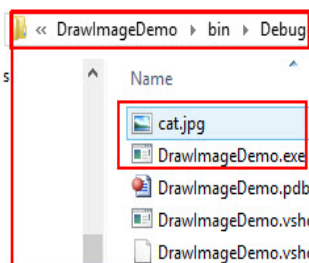
57

57

Vẽ Image

- Ví dụ:

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    string pathImage = Application.StartupPath + @"\cat.jpg";
    Image img1 = Image.FromFile(pathImage);
    Image img2 = Image.FromFile(pathImage);
    Image img3 = Image.FromFile(pathImage);
    Image img4 = Image.FromFile(pathImage);
    int imgWidth = ClientRectangle.Width / 2;
    int imgHeight = ClientRectangle.Height / 2;
    Rectangle rc1 = new Rectangle(0, 0, imgWidth, imgHeight);
    Rectangle rc2 = new Rectangle(ClientRectangle.Width / 2, 0, imgWidth, imgHeight);
    Rectangle rc3 = new Rectangle(0, ClientRectangle.Height/2, imgWidth, imgHeight);
    Rectangle rc4 = new Rectangle(ClientRectangle.Width / 2, ClientRectangle.Height / 2,
                                imgWidth, imgHeight);
}
```



58

58

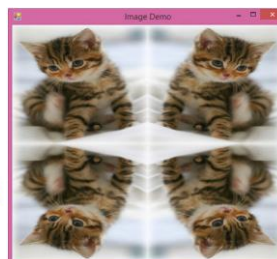
Vẽ Image

```
Graphics g = e.Graphics;
g.DrawImage(img1, rc1);

img2.RotateFlip(RotateFlipType.RotateNoneFlipX);
g.DrawImage(img2, rc2);

img3.RotateFlip(RotateFlipType.RotateNoneFlipY);
g.DrawImage(img3, rc3);

img4.RotateFlip(RotateFlipType.RotateNoneFlipXY);
g.DrawImage(img4, rc4);
img4.Save(@"D:\cat4.jpg", ImageFormat.Png );
```



59

59

Tô màu các đối tượng

- Sử dụng các phương thức **Fill...** của lớp Graphics.
- Các phương thức Fill có cú pháp tương tự như các phương thức **Draw...** tương ứng, chỉ khác ở tham số đầu tiên.
 - Với phương thức **Draw** (ngoại trừ phương thức DrawString), tham số đầu tiên là đối tượng **Pen**
 - Với phương thức **Fill**, tham số đầu tiên là đối tượng **Brush**.

60

60

Tô màu các đối tượng (tt)

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    //tạo cọ tô màu đỏ
    SolidBrush RedBr = new SolidBrush(Color.Red);
    e.Graphics.FillEllipse(RedBr, 10, 10, 150, 50);
    Rectangle rect = new Rectangle(170, 10, 100, 100);
    //tạo cọ tô loại HatchBrush
    HatchBrush HatchBr = new HatchBrush(HatchStyle.Cross,
                                         Color.White, Color.Blue);
    e.Graphics.FillEllipse(HatchBr, rect);
}
```



61

61

Vẽ chữ

- Sử dụng phương thức DrawString

– Vẽ chữ xuất phát tại một điểm:

- DrawString (string, Brush, Font, Point): vẽ chuỗi string, với Font, tại vị trí Point, màu Brush
- DrawString (string, Brush, Font, Point, StringFormat): vẽ chuỗi string, với Font, tại vị trí Point, màu Brush và thiết lập định dạng văn bản xuất, chuyển hướng văn bản phụ thuộc vào **StringFormat**,...

Trong đó, **StringFormat** là đối tượng cho phép thiết lập các định dạng xuất của văn bản.

62

62

Vẽ chữ (tt)

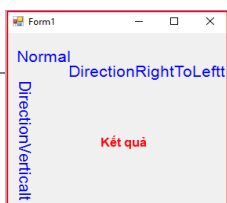
- FormatFlags : được dùng để canh chỉnh, chuyển hướng văn bản xuất.
- Giá trị của FormatFlags là một giá trị thuộc kiểu dữ liệu liệt kê **StringFormatFlags**
 - DirectionRightToLeft
 - DirectionVertical
 - NoClip
 - NoWrap
 - ...

63

63

Vẽ chữ (tt)

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    Font font = new Font("Arial", 16);
    e.Graphics.DrawString("Normal", font, Brushes.Blue, 10, 20);
    StringFormat format = new StringFormat();
    format.FormatFlags = StringFormatFlags.DirectionRightToLeft;
    e.Graphics.DrawString("DirectionRightToLeft", font, Brushes.Blue,
        ClientRectangle.Width, 40, format);
    format.FormatFlags = StringFormatFlags.DirectionVertical;
    e.Graphics.DrawString("DirectionVertical", font, Brushes.Blue,
        10, 60, format);
}
```



64

64

Vẽ chữ (tt)

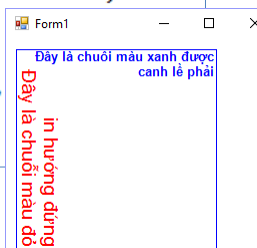
- Vẽ chữ trong khung chữ nhật:
 - DrawString (string, Brush, Font, Rectangle): vẽ chuỗi string, với Font, màu Brush, tại góc trên trái hình chữ nhật Rectangle
 - DrawString (string, Brush, Font, Rectangle, StringFormat): vẽ chuỗi string, với Font, màu Brush, có thiết lập định dạng văn bản xuất, canh lề, chuyển hướng văn bản phụ thuộc vào StringFormat

65

65

Vẽ chữ (tt)

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    Rectangle rect = new Rectangle(10, 10, 200, 200);
    e.Graphics.DrawRectangle(new Pen(Color.Blue), rect);
    StringFormat Format1 = new StringFormat();
    Format1.FormatFlags = StringFormatFlags.DirectionRightToLeft;
    string str = "Đây là chuỗi màu xanh được canh lề phải";
    e.Graphics.DrawString(str, new Font("Arial", 10, FontStyle.Bold),
        new SolidBrush(Color.Blue), rect, Format1);
    StringFormat Format2 = new StringFormat();
    Format2.FormatFlags = StringFormatFlags.DirectionVertical;
    Format2.Alignment = StringAlignment.Far;
    str = "Đây là chuỗi màu đỏ in hướng đứng";
    e.Graphics.DrawString(str, new Font("Arial", 14),
        new SolidBrush(Color.Red), rect, Format2);
}
```



66

66

Vẽ bằng chuột

- Nắm bắt các sự kiện MouseDown, MouseUp, MouseMove để thực hiện thao tác vẽ
 - Trong sự kiện **MouseDown**, lưu giữ vị trí điểm nhấn chuột
 - Trong sự kiện **MouseMove**, kiểm tra nếu như trong quá trình di chuyển chuột, người dùng có giữ nút trái chuột, mới thực hiện thao tác vẽ, (có thể kết hợp xử lý trong sự kiện MouseUp), sau đó, cập nhật lại vị trí điểm vẽ hiện tại
 - **Lưu ý**: để tránh tình trạng giật màn hình, khi vẽ, chúng ta vẽ trên một bitmap tạm, sau đó mới sao chép ảnh tạm này lên form (cần khai báo thuộc tính DoubleBuffered của form là **True**)

67