

BÀI 12: GDI+

Thời gian: 120 phút

Giảng viên: PHẠM PHÚ KHƯƠNG

Email: phamphukhuong@dtu.edu.vn

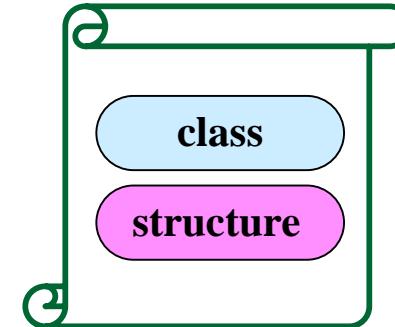
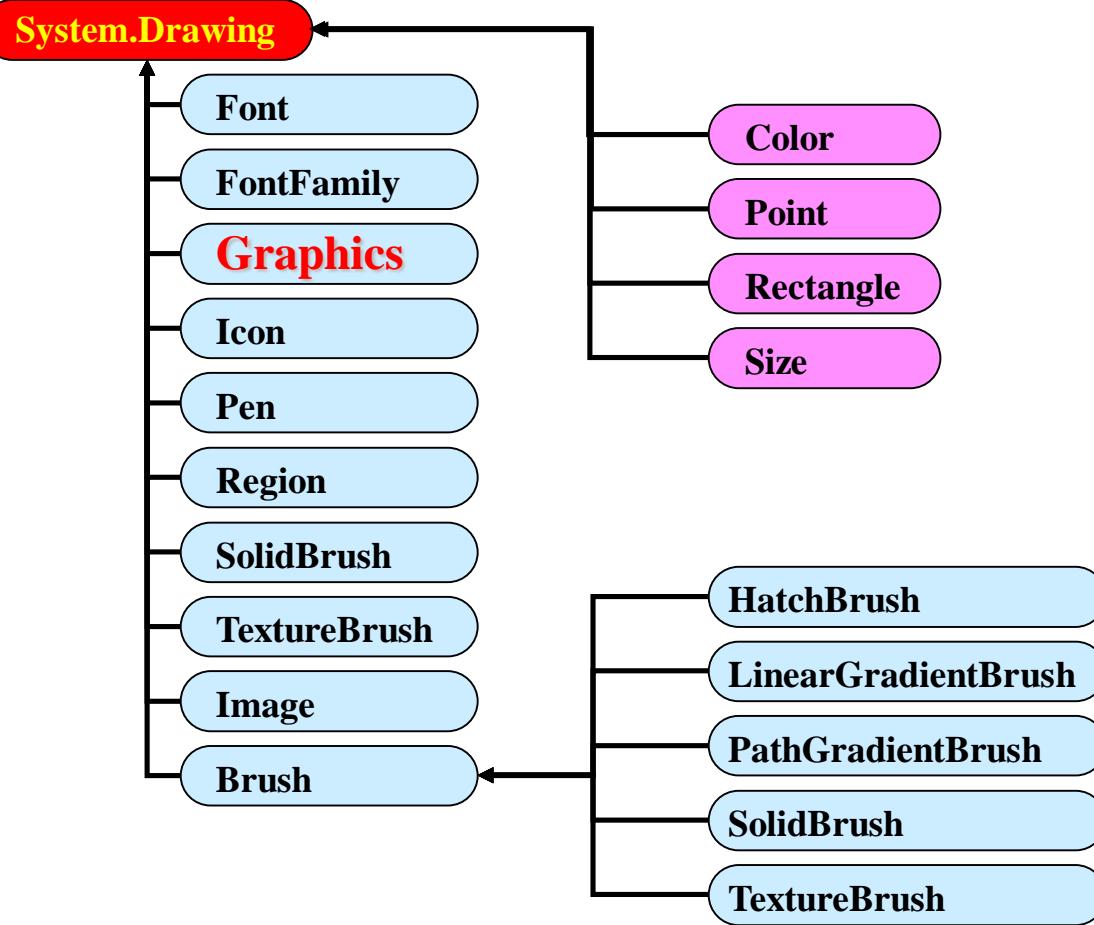
Điện thoại: 0905635421

Nội Dung

- GDI+
- Chương trình vẽ cơ bản trên Form
- Lớp Graphics và hàm OnPaint()
- Lớp Color và Font
- Lớp Pen và Brush
- Các hàm vẽ đường thẳng, hình chữ nhật, ellipse
- Các hàm vẽ cung, đa giác
- Hiển thị ảnh
- Minh họa Multimedia

- GDI: **Graphical Device Interface**
- **GDI+ là API (Application programming interface)** cung cấp các lớp cho phép
 - Tạo những đồ họa 2D vector
 - Thao tác trên font, chuỗi ký tự
 - Hiển thị các đường, hình và ảnh...
- Thư viện FCL chứa các lớp thao tác vẽ trong namespace
 - **System.Drawing**
- Tất cả các thao tác tô vẽ trên GUI đều thực hiện bởi chức năng GDI+

System.Drawing



Vẽ trên Form

```
public partial class Form1 : Form
{
    public Form1()
    {
        InitializeComponent();
    }
    protected override void OnPaint(PaintEventArgs e)
    {
        Graphics g = e.Graphics;
        g.DrawString("Hello GDI!", Font, Brushes.Red, 20, 20);
    }
}
```

Graphics

- Lớp Graphics thể hiện
 - “*Abstract*” drawing surface
 - Tập hợp những “*tool*” cho phép thao tác trên surface đó
- Để lấy đối tượng Graphics
 - Sử dụng thuộc tính Graphics được truyền cho OnPaint()
 - Sử dụng phương thức CreateGraphics() của control
 - Lấy từ đối tượng dẫn xuất từ Bitmap
- Gọi hàm Invalidate() thay vì OnPaint()

Graphics

Lấy đối tượng Graphics

```
protected override void OnPaint(PaintEventArgs paintevent)  
{  
    Graphics graf=paintevent.Graphics;  
}
```

Tùy tham số PaintEventArgs

```
private void mainForm_Paint(object sender, PaintEventArgs  
    paintevent)  
{  
    Graphics graf=paintevent.Graphics;  
}
```

Lấy đối tượng Graphics

```
private void PaintMe(Control testcontrol)
{
    Graphics graf=testcontrol.CreateGraphics();
    ...
}
```

Lấy từ control

```
protected override void OnPaint(PaintEventArgs
    paintevent)
{
    Bitmap bmpimage=new Bitmap("hutech.jpg");
    Graphics graf = Graphics.FromImage (bmpimage);
    ...
}
```

Lấy từ ảnh

DrawString() method

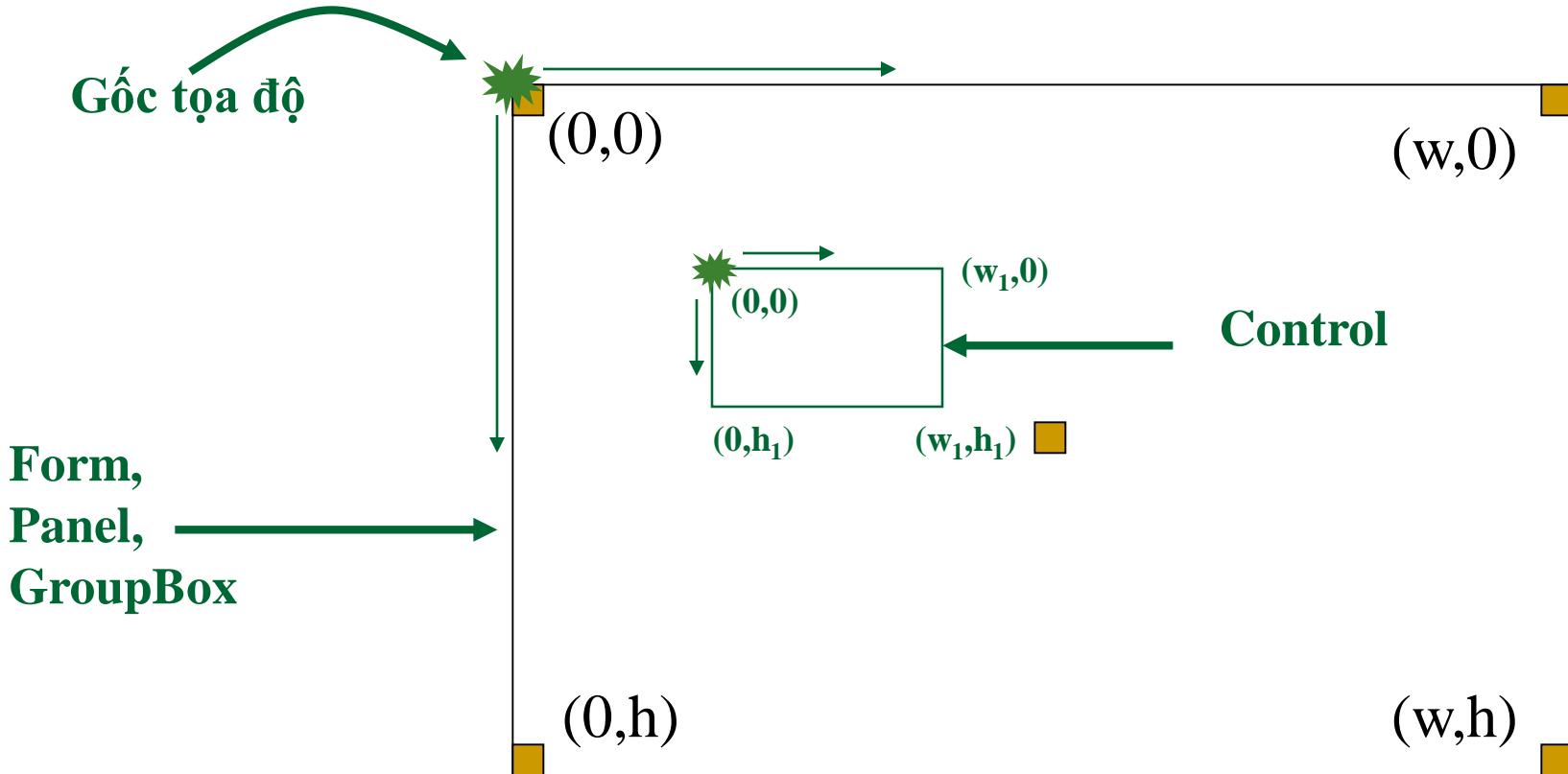
- Hiển thị text trong Graphics cụ thể

- Có nhiều phiên bản

- `DrawString(String text, // Text thể hiện
Font f, // Font
Brush b, // Color & texture
float x, float y); // vị trí góc trái trên`

- Tham số Font và Brush không có mặc định nên phải truyền vào.

Tọa độ hệ thống



Tọa độ hệ thống

- **Graphics.GraphicsUnit**: xác định đơn vị của bề mặt
 - **GraphicsUnit.Pixel** (default)
 - **GraphicsUnit.Inch**
 - **GraphicsUnit.Milimeter**
 - **GraphicsUnit.Point**
- **Graphics.PageScale**: tỷ lệ output
 - **g.PageScale = 1f** (default)

Color

- Sử dụng màu được định nghĩa trong Color
 - Color.Blue, Color.Red, Color.White...
- Sử dụng màu định nghĩa cho hệ thống
 - SystemColors.Control, SystemColors.ControlText...
- Sử dụng màu ARGB
 - 32 bit để thể hiện màu
 - A (alpha) thể hiện mức độ trong suốt (255 opaque)
 - RGB là **Red**, **Green** và **Blue**
 - Tạo màu sử dụng hàm FromArgb()
 - Color red = Color.FromArgb(255,0,0);
 - Color blue = Color.FromArgb(128, 0, 255, 0);



- Cách tạo đối tượng Font: new Font(...)
 - Có 13 phiên bản của constructor

```
□ Font fa = new Font("Times New Roman", 8);  
Font fb = new Font("Arial", 36, FontStyle.Bold);  
Font fc = new Font(fb, FontStyle.Bold | FontStyle.Italic);  
Font fd = new Font("Arial", 1, GraphicsUnit.Inch);
```

Size = 8 pixel

Size = 1 inch

- Nếu tên font không tìm thấy thì font mặc định được sử dụng.

Font

```
protected void DemoFont(Graphics g)
{
    Font fa = new Font("Times New Roman", 14);
    g.DrawString("Hutech", fa, Brushes.Salmon, 10, 10);

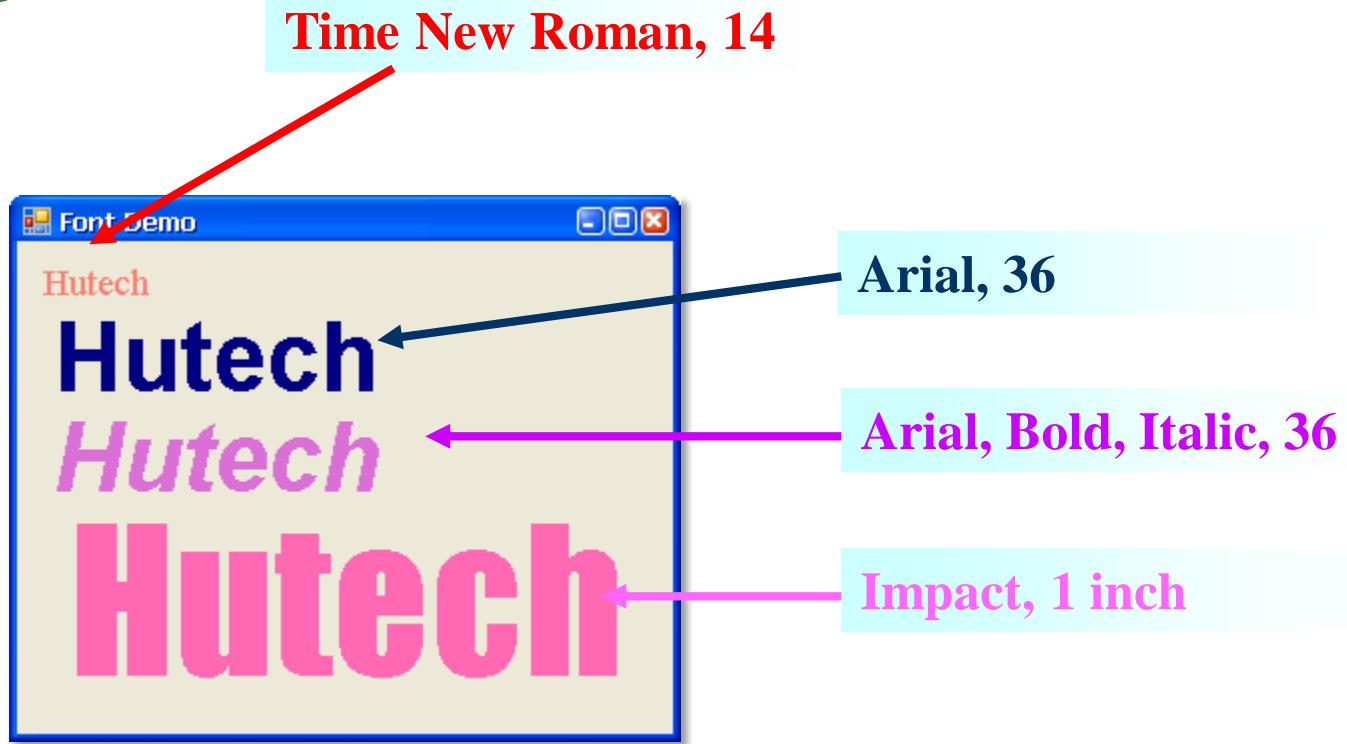
    Font fb = new Font("Arial", 36, FontStyle.Bold);
    g.DrawString("Hutech", fb, Brushes.Navy, 10, 30);

    Font fc = new Font(fb, FontStyle.Bold | FontStyle.Italic);
    g.DrawString("Hutech", fc, Brushes.Orchid, 10, 80);

    Font fd = new Font("Impact", 1, GraphicsUnit.Inch);
    g.DrawString("Hutech", fd, Brushes.HotPink, 10, 120);
}
```

Font

■ Demo



Pen



- Xác định **width**, **style**, **fill style**
- Không cho kế thừa, nhưng tạo thể hiện được
- Trong namespace **System.Drawing**
 - Pen p1 = new Pen(Color.Green);
Pen p2 = new Pen(Color.blue, 10);
 - Sử dụng lớp **Pens** có 141 pen được định nghĩa trước.
 - Pen p3 = Pens.Violet;

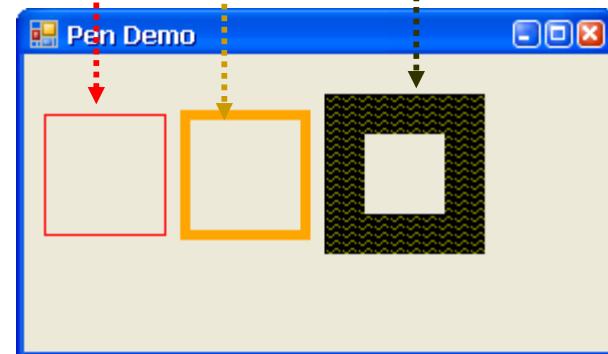
Pen

```
protected override void OnPaint(PaintEventArgs e)
{
    Graphics g = e.Graphics;

    Pen p1 = new Pen(Color.Red); .....
    g.DrawRectangle(p1, new Rectangle(10, 30, 60, 60));

    Pen p2 = new Pen(Color.Orange,5); .....
    g.DrawRectangle(p2,new Rectangle(80,30,60,60));

    HatchBrush hb = new HatchBrush( HatchStyle.Wave,Color.Olive);
    Pen p3 = new Pen(hb, 20); .....
    g.DrawRectangle(p3, new Rectangle(160, 30, 60, 60));
}
```



Brush

- Dùng để tô vùng bên trong của hình
- Lớp Brush là lớp abstract nên không tạo thể hiện
- Sử dụng các lớp kế thừa sau để tạo brush
 - **SolidBrush**
 - **LinearGradientBrush**
 - **TextureBrush**
 - **HatchBrush**
- Sử dụng lớp Brushes định nghĩa trước các brush.



Brush

```
protected override void OnPaint(PaintEventArgs e)
{
    Graphics g = e.Graphics;

    SolidBrush b1 = new SolidBrush(Color.Aquamarine); Solid
    g.FillEllipse(b1, 40, 40, 80, 80);

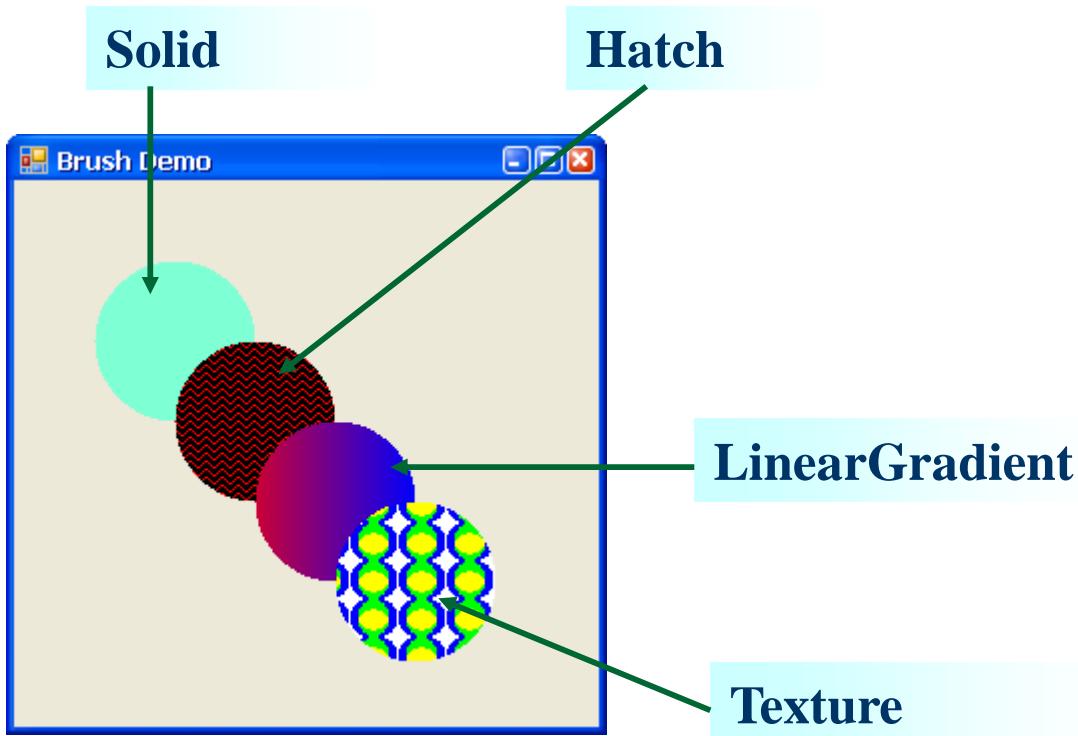
    HatchBrush b2 = new HatchBrush(HatchStyle.ZigZag, Color.Red);
    g.FillEllipse(b2, 80, 80, 80, 80); Hatch

    Rectangle rect = new Rectangle(0,0,100,100);
    LinearGradientBrush b3 = new LinearGradientBrush(rect,
        Color.Red, Color.Blue, LinearGradientMode.Horizontal);
    g.FillEllipse(b3, 120, 120, 80, 80); LinearGradient

    Image img = Image.FromFile("bitmap1.bmp");
    TextureBrush b4 = new TextureBrush(img);
    g.FillEllipse(b4, 160, 160, 80, 80); Texture
}
```

Bursh

■ Demo



Line, Rectangle, Ellipse

■ DrawLine

- *(Pen p, int x1, int y1, int x2, int y2)*

■ DrawRectangle

- *(Pen p, int x, int y, int width, int height)*

■ DrawEllipse

- *(Pen p, int x, int y, int width, int height)*

■ FillRectangle

- *(Brush b, int x, int y, int width, int height)*

■ FillEllipse

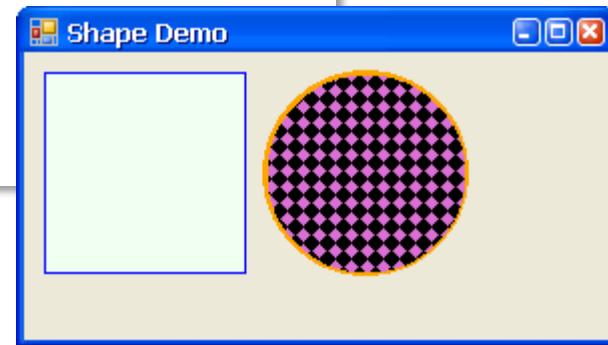
- *(Brush b, int x, int y, int width, int height)*

Line, Rectangle, Ellipse

```
protected void DemoShape(Graphics g)
{
    Rectangle r1 = new Rectangle(10, 10, 100, 100);
    Rectangle r2 = new Rectangle(11, 11, 99, 99);
    Rectangle r3 = new Rectangle(120, 10, 100, 100);
    Rectangle r4 = new Rectangle(121, 11, 99, 99);
    Pen p = new Pen(Color.Orange, 3);
    HatchBrush hb = new
        HatchBrush(HatchStyle.SolidDiamond, Color.Orchid);

    g.DrawRectangle(Pens.Blue, r1);
    g.FillRectangle(Brushes.Honeydew, r2);

    g.DrawEllipse(p, r3);
    g.FillEllipse(hb, r4);
}
```

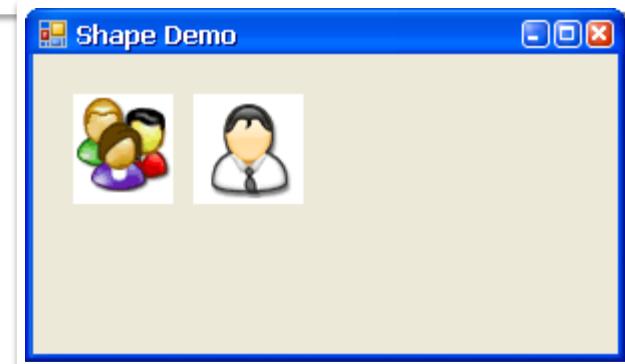


Image

- Lớp Image hiển thị các ảnh bitmap
 - Các dạng ảnh: *.bmp, *.gif, *.jpg, *.ico...
- Phương thức static **FromFile** tạo ảnh từ file
 - `Image img = Image.FromFile("hutech.bmp");`
 - `Image img2 = Image.FromFile("hutech.gif");`
- Phương thức **DrawImage** xuất ảnh lên Graphics
 - `g.DrawImage(img, 10, 10);`
 - `G.DrawImage(img2, 10, 10, 100,100); // scale trong
hình chữ nhật kích thước 100x100`

Image

```
protected void DemoImage(Graphics g)
{
    //đọc ảnh từ đĩa
    Image img = Image.FromFile("people.bmp");
    g.DrawImage(img, 20, 20);
    //đọc ảnh từ embedded resource
    Image img2 = new Bitmap(GetType(), "people3.bmp");
    g.DrawImage(img2, 80, 20);
}
```



Image

```
protected void DemoImage2(Graphics g)
{
    Image img = new Bitmap(GetType(), "bluekimono.bmp");
    // lấy đối tượng Graphics của img
    Graphics GImg = Graphics.FromImage(img);
    // tô hình ellipse đặc
    GImg.FillEllipse(Brushes.Gold, 10, 10, 60, 60);

    // hiển thị ra Graphics của form
    g.DrawImage(img, 0, 0, this.Width, Height);
}
```

Ellipse được vẽ lên ảnh, rồi sau đó với vẽ ảnh lên Form



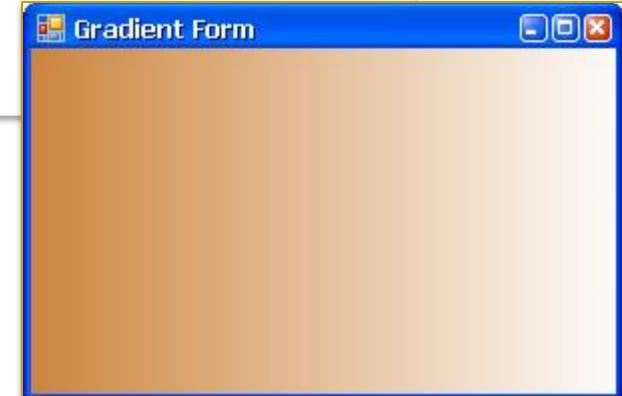
Minh họa 1

■ Custom lại nền của Form

```
protected override void OnPaintBackground(PaintEventArgs e)
{
    Graphics g = e.Graphics;
    Rectangle rect = new Rectangle(0, 0, Width, Height);
    LinearGradientBrush b = new
        LinearGradientBrush(rect, Color.Peru, Color.White,
        LinearGradientMode.Horizontal);

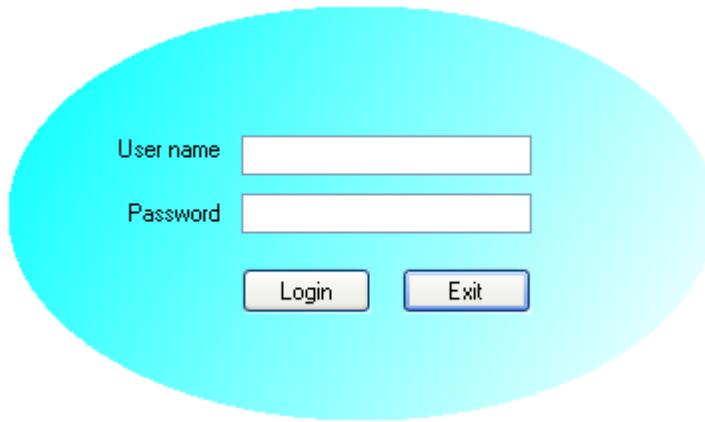
    g.FillRectangle(b, 0, 0, Width, Height);
}
```

Override phương thức
OnPaintBackground của Form



Form có dạng NonRectangle

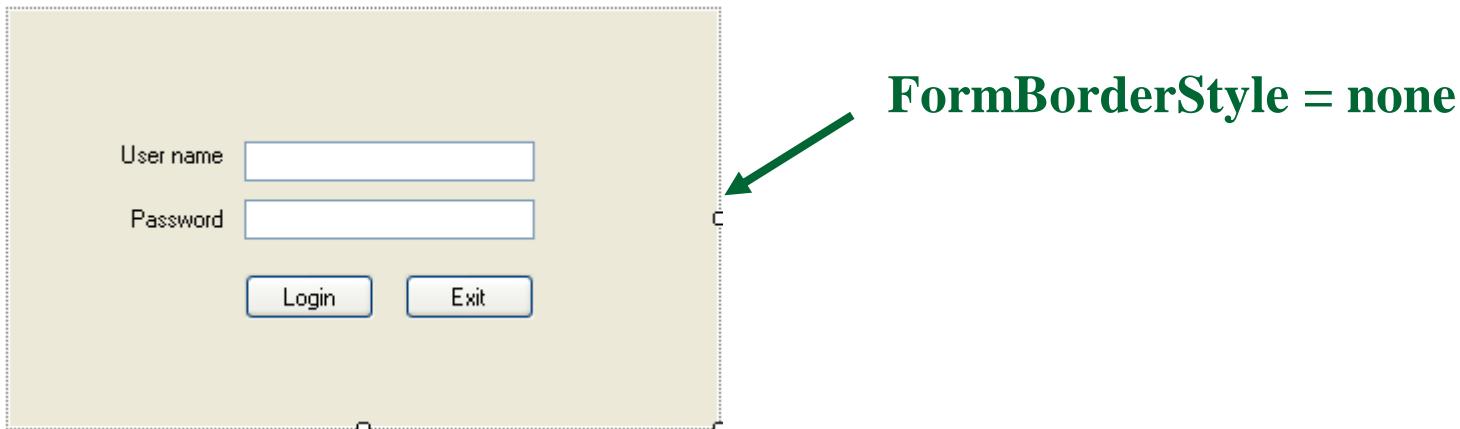
- Tạo form có hình dạng khác hình chữ nhật
 - Sử dụng thuộc tính **TransparencyKey** của Form
 - Sử dụng các hiệu ứng màu được tô



Form có dạng NonRectangle

■ Bước 1:

- ❑ Tạo ứng dụng Windows Application
- ❑ Thiết kế Form có dạng như sau



Form có dạng NonRectangle

- **Bước 2:**
 - Thiết lập các thuộc tính cho Form như sau:
 - **TransparencyKey = Control:** màu sẽ trong suốt khi vẽ trên Form
 - **FormBorderStyle = None:** Form không có đường biên
 - Thiết lập màu nền cho 2 Label là Transparent
 - Phần background của 2 label sẽ tiệp với nền bên dưới

Form có dạng NonRectangle

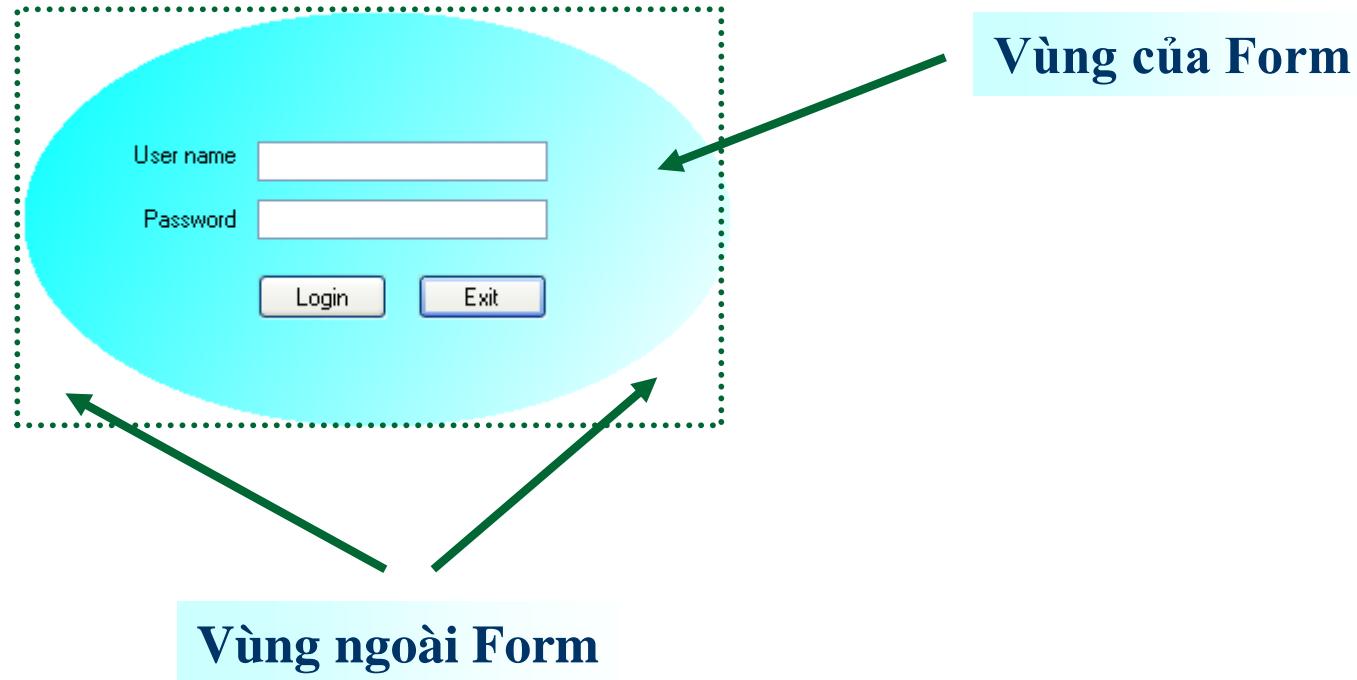
■ Bước 3: Tạo trình xử lý cho sự kiện Paint

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
    Graphics myGraphic = e.Graphics;

    // khai báo các tọa độ & kích thước
    Point p = new Point(0, 0);
    Point p2 = new Point(Width, Height);
    Size size = new Size(Width, Height); // kích thước của form
    // tạo brush gradient
    LinearGradientBrush myBruch =
        new LinearGradientBrush(p, p2, Color.Cyan, Color.White);
    // khai báo hình ellipse làm hình dạng của form
    Rectangle r1 = new Rectangle(p, size);
    // tô hình ellipse với brush vừa định nghĩa
    myGraphic.FillEllipse(myBruch, r1);
}
```

Form có dạng NonRectangle

■ Demo



Form có dạng NonRectangle

- **Bổ sung di chuyển form**
 - Thêm namespace:
 - **System.Runtime.InteropServices;**
 - **Load các hàm từ DLL vào project**

```
public const int WM_NCLBUTTONDOWN = 0xA1;  
public const int HT_CAPTION = 0x2;
```

```
[DllImportAttribute("user32.dll")]  
public static extern int SendMessage(IntPtr hWnd,  
        int Msg, int wParam, int lParam);  
  
[DllImportAttribute("user32.dll")]  
public static extern bool ReleaseCapture();
```

Form có dạng NonRectangle

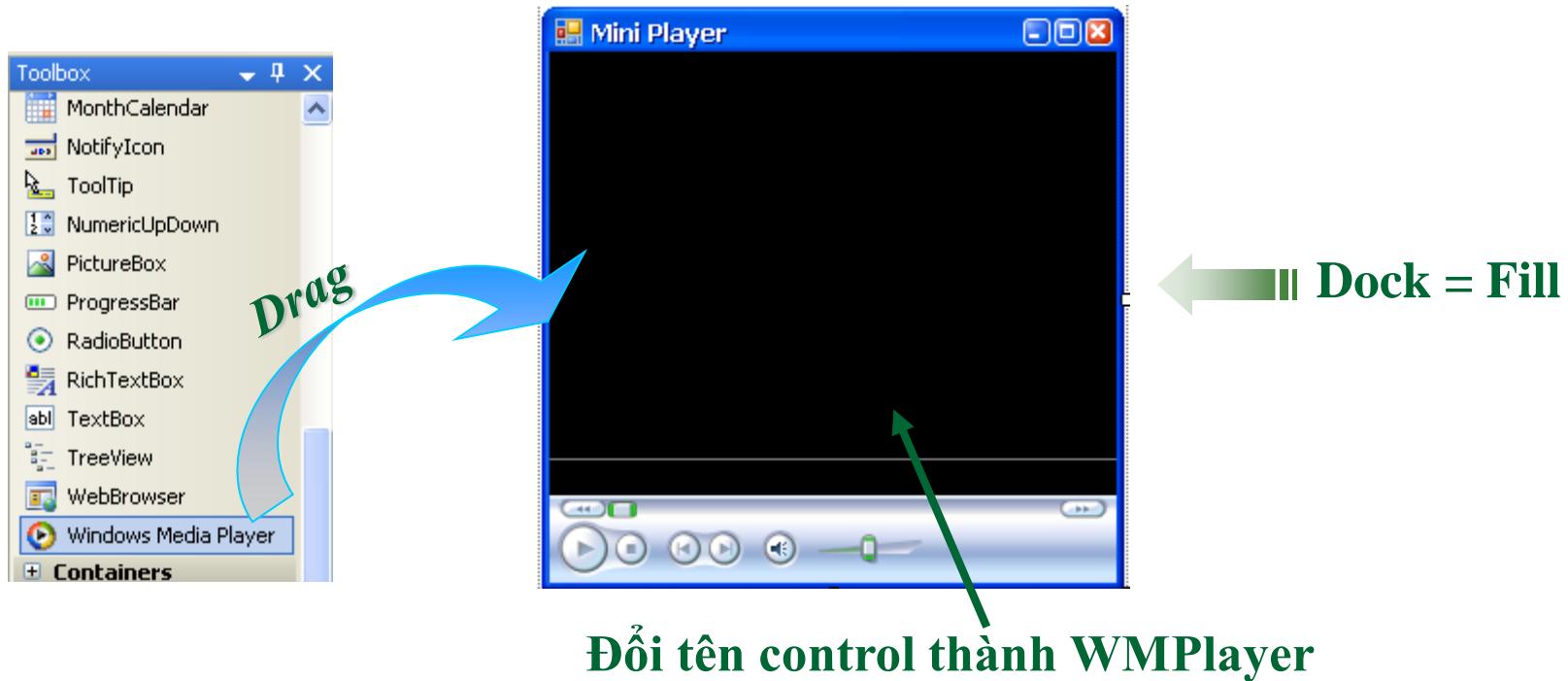
- **Bổ sung code vào trình xử lý sự kiện MouseDown**

```
private void Form1_MouseDown(object sender, MouseEventArgs e)
{
    ReleaseCapture();
    SendMessage(Handle, WM_NCLBUTTONDOWN, HT_CAPTION, 0);
}
```

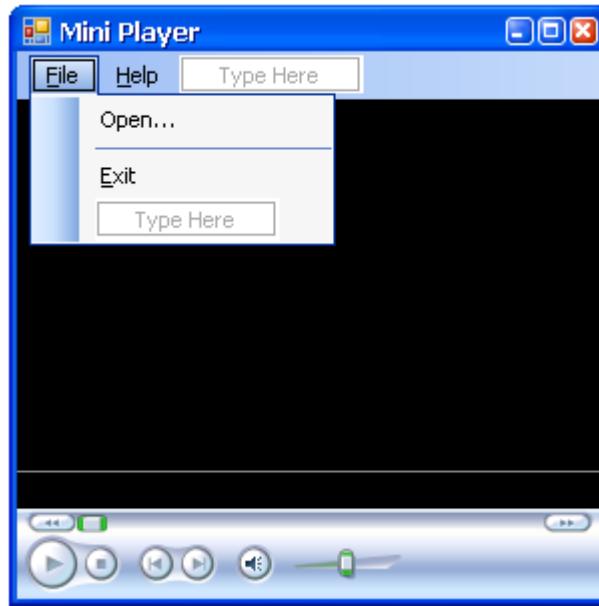
- **Tạo ứng dụng chứa Windows Media Player control cho phép**
 - Play các file video và sound theo nhiều dạng format
 - **MPEG (Motion Pictures Expert Group): video**
 - **AVI (Audio-video Interleave): video**
 - **WAV (Windows Wave-file Format): audio**
 - **MIDI (Musical Instrument Digital Interface): audio**

- **Bước 1: bổ sung Windows Media Player vào ToolBox**
 - Kích chuột phải vào **ToolBox** → chọn **Choose Items...**
 - Trong Dialog Choose Toolbox Items chọn **COM Components**
 - Chọn **Windows Media Player**
 - Khi đó control WMP sẽ hiện ở dưới cùng của ToolBox

- **Bước 2: kéo Windows Media Player thả vào Form**
 - Thiết lập Dock = Fill



- **Bước 3: Tạo MenuStrip để bổ sung chức năng Open File media**



menuStrip1

■ Bước 4: viết trình xử lý cho MenuItem Open

```
private void openMenuItem_Click(object sender, EventArgs e)
{
    // tạo hộp thoại mở file
    OpenFileDialog dlg = new OpenFileDialog();
    // lọc hiển thị các loại file
    dlg.Filter = "AVI file|*.avi|MPEG File|*.mpeg|"+
                  "WAV File|*wav|MIDI File|*.midi";
    // Hiển thị Open Dialog
    if (dlg.ShowDialog() == DialogResult.OK)
    {
        // lấy tên file cần mở cho WMPPlayer
        WMPPlayer.URL = dlg.FileName;
    }
}
```

■ Demo



Q & A

CÂU HỎI

1. **GDI LÀ GÌ**
2. **Để đồ họa trên form ta dùng lớp nào?**
3. **Phương thức Onpaint() để làm gì**

YOUTUBE

- <https://www.youtube.com/watch?v=AAaZBiYR4PM>
- <https://www.youtube.com/watch?v=8kr-V4EQ7gw>