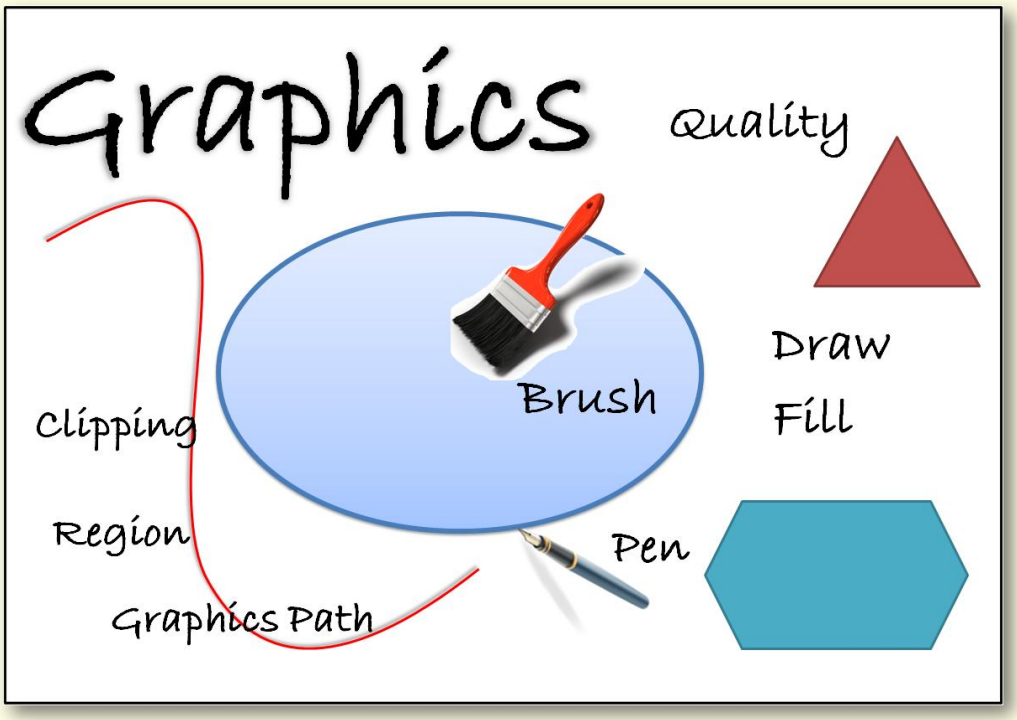
C# & VB

Control/Image

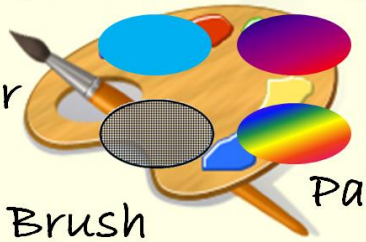


+

Color Blend

Texture Brush

Linear Gradation Brush



Color

Hatch Brush

Path Gradation Brush

GDI

.NET Framework

Training Version

グラフィック入門

グラフィック トレーニング

概要

.NET のグラフィック描画は、どんなことができるのでしょうか？

グラフィックオブジェクトやグラフィック環境、概念を理解するためには、クラスを使って馴れることが近道です。本書に記載されているコードをカット アンド ペーストして、一つ一つの機能を体験してください。

前提

グラフィックを行うためには、Visual Studio の基本操作や C#または VB の言語およびプログラミングの経験が必要です。本書では、上記条件を満たしているとして記述しています。

Form 継承クラスは、画面を作成する際の基本中の基本ですから、本書では Form 継承クラス上で「グラフィック」のトレーニングを行います。

本来 Form クラスでは特別な理由がない限り、描画を行うことは推奨されません。一般的には、Panel などのコンテナや PictureBox コントロール、または、コントロールのオーナ描画やカスタム コントロール内で描画をします。本書では、「グラフィック入門」ということでこれらの環境には触れません。

```
このブロックは、C#言語のサンプルコードを示します。
```

```
このブロックは、VB のサンプルコードを示します。
```

トレーニング環境用のプロジェクトを作成

1. Visual Studio 2008 を実行して、「新しいプロジェクト」を実行します。
2. 「Windows フォーム アプリケーション」を選択して、適当なプロジェクト名で作成します。
3. ウィザードで「Form1」という Form クラスを継承したクラスが自動生成されます。
4. 本書では、この Form1 クラスを使って描画のトレーニングを行います。

グラフィック描画のためのイベントハンドラを作成

1. Form1 のデザインを表示します。
2. プロパティ ウィンドウを開いて、イベントのプロパティを選択します。
3. カテゴリ[表示]の Paint を選択して、ダブルクリックすると以下のコードが生成されます。

```
private void Form1_Paint(object sender, PaintEventArgs e)
{
}
```

```
Private Sub Form1_Paint(ByVal sender As System.Object, ByVal e As System.Windows.Forms.PaintEv
entArgs) Handles MyBase.Paint

End Sub
```

描画のためのグラフィック　オブジェクトを取得

Form1_Paint イベントハンドラに以下のコードを追加してグラフィック オブジェクト変数を作成します。

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

```
Private Sub Form1_Paint(ByVal sender As System.Object, ByVal e As System.Windows.Forms.PaintEv
entArgs) Handles MyBase.Paint
    Dim g As Graphics = e.Graphics
End Sub
```

イベントハンドラの引数 e からグラフィック オブジェクトを取り出し、グラフィックオブジェクト変数 g を作成します。以降のトレーニングでは、グラフィックオブジェクト変数 g を使用します。

サンプル コードは、それぞれが独立した（ユニークな名称）メソッドになっていますので、Form1_Paint イベントハンドラよりグラフィックオブジェクト g を引数にして実行してください。

例えば、「直線を描画する(DrawLine)」の場合、メソッドは paint_DrawLine ですので以下の様に記述します。

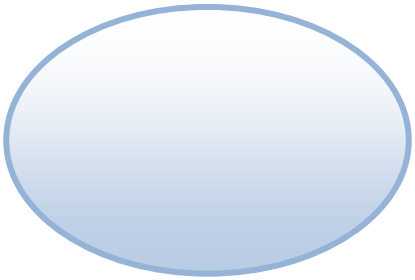
```
Imports System.Drawing.Drawing2D                ' 追記

Public Class Form1
    Private Sub Form1_Paint(ByVal sender As System.Object, ByVal e As System.Windows.Forms.Pain
tEventArgs) Handles MyBase.Paint
        Dim g As Graphics = e.Graphics          ' 追記
    End Sub

    Private Sub paint_DrawLine(ByVal g As Graphics)
        ＜省略＞
    End Sub
End Class
```

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Drawing.Drawing2D;      // 追記
using System.Text;
using System.Windows.Forms;

namespace GDICS
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Paint(object sender, PaintEventArgs e)
        {
            Graphics g = e.Graphics;
            paint_DrawLine(g);          // 追記
        }

        // 直線を描画する
        private void paint_DrawLine(Graphics g)
        {
            ＜省略＞
        }
    }
}
```

トレーニング内容

1. GDI+ 輪郭線（アウトライン）のグラフィック描画
2. GDI+ 塗りつぶし（ペイント）のグラフィック描画
3. GDI+ パスの作成とグラフィック描画

図形の描画は、線を含む輪郭線と塗りつぶしに分かれています。それぞれは独立していて同時に行うことはできません。



上の画像でもわかるとおり、輪郭線には、形状、色、幅、線種などがあり、塗りつぶしの場合、形状、べた塗り、網掛け、グラデーションなどがあります。これらを柔軟に描画するために、それぞれを分けて行います。

また、基本の描画機能だけでは複雑な図形には対応できないため、パス（GraphicPath）というオブジェクトが用意されています。

例えば、基本図形に角丸の矩形は存在していません。パスにコーナー用円弧を 4 箇所とそれぞれの円弧を結ぶ直線のパスを作成して描画します。

グラフィック オブジェクトに作成したパスを渡して描画します。パス自身に描画機能はありません。

グラフィックス描画の注意点

1. Form の Load イベントで描画処理を行ってはいけません。Form が完全に生成しきっていない状態ですから、動作保証はされません。
2. Paint イベントハンドラ以外のイベントでの描画は、結果を理解した上で行うようにしてください。
   Windows の仕組みでもあるのですが、描画はアプリケーション側の責任で行う必要があります。Windows は、描画タイミングを Paint イベントで知らせてくれるだけですから、Paint イベントハンドラで必要な描画をおこないます。ここで描画しなければ、他のイベントで描画した場合、その内容は消えてしまうことになります。
3. サンプルコードでは省略していますが、実運用の際は Pen や Brush などのオブジェクト使用後は、Dispose メソッドを呼び出しシステムリソースを解放するようにしてください。

GDI+ 輪郭線（アウトライン）のグラフィック描画

直線を描画する(DrawLine)

折れ線を描画する(DrawLines)

矩形を描画する(DrawRectangle)

複数矩形を描画する(DrawRectangles)

多角形を描画する(DrawPolygon)

楕円を描画する(DrawEllipse)

円弧（楕円の一部）を描画する(DrawArc)

扇形を描画する(DrawPie)

曲線（ベジエスプライン）を描画する(DrawBezier)

複数曲線（ベジエスプライン）を描画する(DrawBeziers)

曲線（カーディナル・スプライン）を描画する(DrawCurve)

閉じた曲線（カーディナル・スプライン）を描画する(DrawClosedCurve)

直線を描画する(DrawLine)

開始点(X,Y)と終了点(X,Y)の間を結ぶ直線を Pen で描画する。事前に Pen の色と線幅を指定。

①	**DrawLine(Pen, Point, Point)** **DrawLine(Pen, PointF, PointF)**	2 つの Point 構造体を接続する直線を描画します。
②	**DrawLine(Pen, Integer, Integer, Integer, Integer)** **DrawLine(Pen, Single, Single, Single, Single)**	座標ペアで指定された 2 つの点を結ぶ直線を描画します。

```
// 直線を描画する
private void paint_DrawLine(Graphics g)
{
    g.DrawLine(new Pen(Color.Black, 1), new Point(10, 10), new Point(500, 10));
    g.DrawLine(new Pen(Color.Blue, 2), new PointF(10.0F, 30.0F), new PointF(500.0F, 30.0F));
    g.DrawLine(new Pen(Color.Red, 3), 10, 50, 500, 50);
    // g.DrawLine(new Pen(Color.Yellow, 4), 10.0F, 70.0F, 500.0F, 70.0F);

    g.DrawLine(new Pen(new SolidBrush(Color.Pink), 4), 10.0F, 70.0F, 500.0F, 70.0F);

    Brush br = new LinearGradientBrush(new Rectangle(10, 70, 500, 70), Color.Pink,
        Color.Blue, LinearGradientMode.BackwardDiagonal);
    g.DrawLine(new Pen(br, 4), 10.0F, 90.0F, 500.0F, 90.0F);

    // blackPen = new Pen(Color.Black, 3)
    // x1 = 100.0F
    // y1 = 100.0F
    // x2 = 500.0F
    // y2 = 100.0F

    // g.DrawLine(blackPen, x1, y1, x2, y2)
}
```

```
Private Sub paint_DrawLine(ByVal g As Graphics)

    g.DrawLine(New Pen(Color.Black, 1), New Point(10, 10), New Point(500, 10))
    g.DrawLine(New Pen(Color.Blue, 2), New PointF(10.0F, 30.0F), New PointF(500.0F, 30.0F))
    g.DrawLine(New Pen(Color.Red, 3), 10, 50, 500, 50)
    'g.DrawLine(New Pen(Color.Yellow, 4), 10.0F, 70.0F, 500.0F, 70.0F)

    g.DrawLine(New Pen(New SolidBrush(Color.Pink), 4), 10.0F, 70.0F, 500.0F, 70.0F)

    Dim br As Brush = New LinearGradientBrush(New Rectangle(10, 70, 500, 70), Color.Pink, _
    Color.Blue, LinearGradientMode.BackwardDiagonal)
    g.DrawLine(New Pen(br, 4), 10.0F, 90.0F, 500.0F, 90.0F)

    'Dim blackPen As New Pen(Color.Black, 3)
    'Dim x1 As Single = 100.0F
    'Dim y1 As Single = 100.0F
    'Dim x2 As Single = 500.0F
    'Dim y2 As Single = 100.0F

    'g.DrawLine(blackPen, x1, y1, x2, y2)

End Sub
```

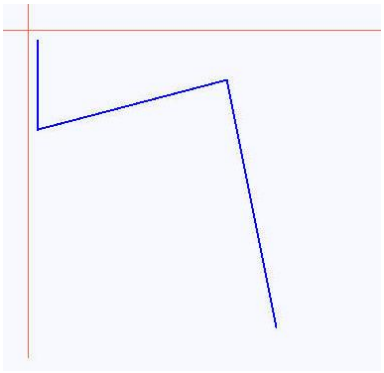
折れ線を描画する(DrawLines)

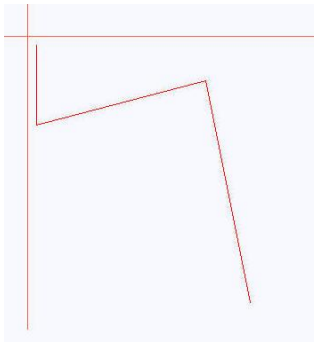
複数の指定された点を順に直線で連結しながら Pen で描画する

DrawLines(Pen, Point())

① Point 構造体の配列を接続する一連の線分を描画します。

DrawLines(Pen, PointF())





```
// 折れ線1
private void paint_DrawLinesPoint(Graphics g)
{
    Pen blackPen = new Pen(Color.Blue, 2);
    Point[] points =
    {
        new Point(10, 10),
        new Point(10, 100),
        new Point(200, 50),
        new Point(250, 300)
    };
    g.DrawLines(blackPen, points);
}

// 折れ線2
private void paint_DrawLinesPointF(Graphics g)
{

    Pen blackPen = new Pen(Color.Red, 1);
    PointF[] points =
    {
        new PointF(10.0F, 10.0F),
        new PointF(10.0F, 100.0F),
        new PointF(200.0F, 50.0F),
        new PointF(250.0F, 300.0F)
    };

    g.DrawLines(blackPen, points);
}
```

```
'折れ線1
Private Sub paint_DrawLinesPoint(ByVal g As Graphics)

    Dim blackPen As New Pen(Color.Blue, 2)
    Dim points As Point() = { _
        New Point(10, 10), _
        New Point(10, 100), _
        New Point(200, 50), _
        New Point(250, 300) _
    }

    g.DrawLines(blackPen, points)

End Sub

'折れ線2
Private Sub paint_DrawLinesPointF(ByVal g As Graphics)

    Dim blackPen As New Pen(Color.Red, 1)
    Dim points As PointF() = _
    { _
        New PointF(10.0F, 10.0F), _
        New PointF(10.0F, 100.0F), _
        New PointF(200.0F, 50.0F), _
        New PointF(250.0F, 300.0F) _
    }

    g.DrawLines(blackPen, points)

End Sub
```