C#程序设计及应用

唐大仕

dstang2000@263.net

北京大学

第12章 绘图及图像处理

唐大仕
dstang2000@263.net
http://www.dstang.com

内容提要

- GDI+及其基本类
- Graphics对象及绘图方法
- 控件与绘图
- 图像处理
- 应用示例

GDI+
及其基本类

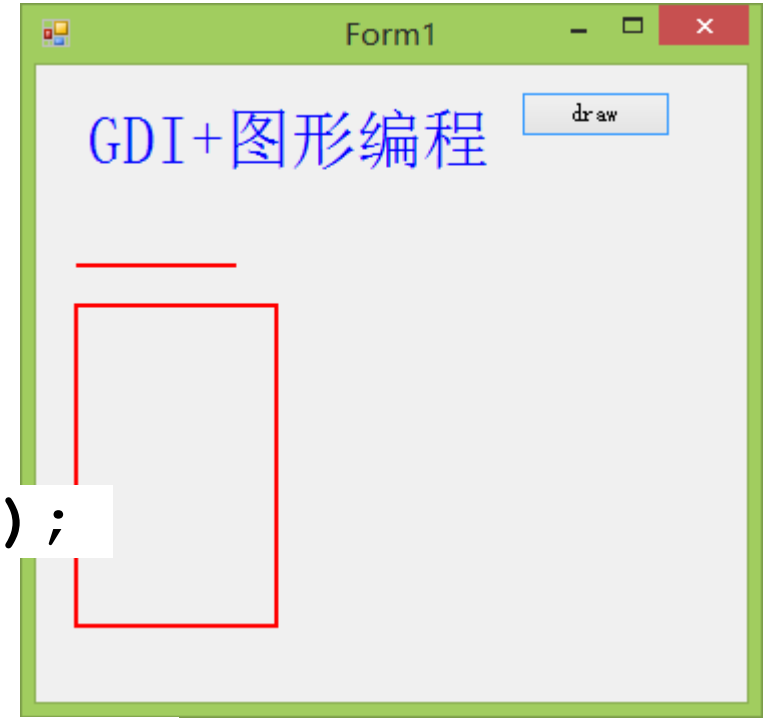
绘图　图像

GDI+技术简介

- GDI: Graphics Device Interface.
- GDI+：GDI的改进，同时也是.NET框架结构的重要组成部分
- 和GDI一样它提供对二维图形图像和文字排版处理的支持 .

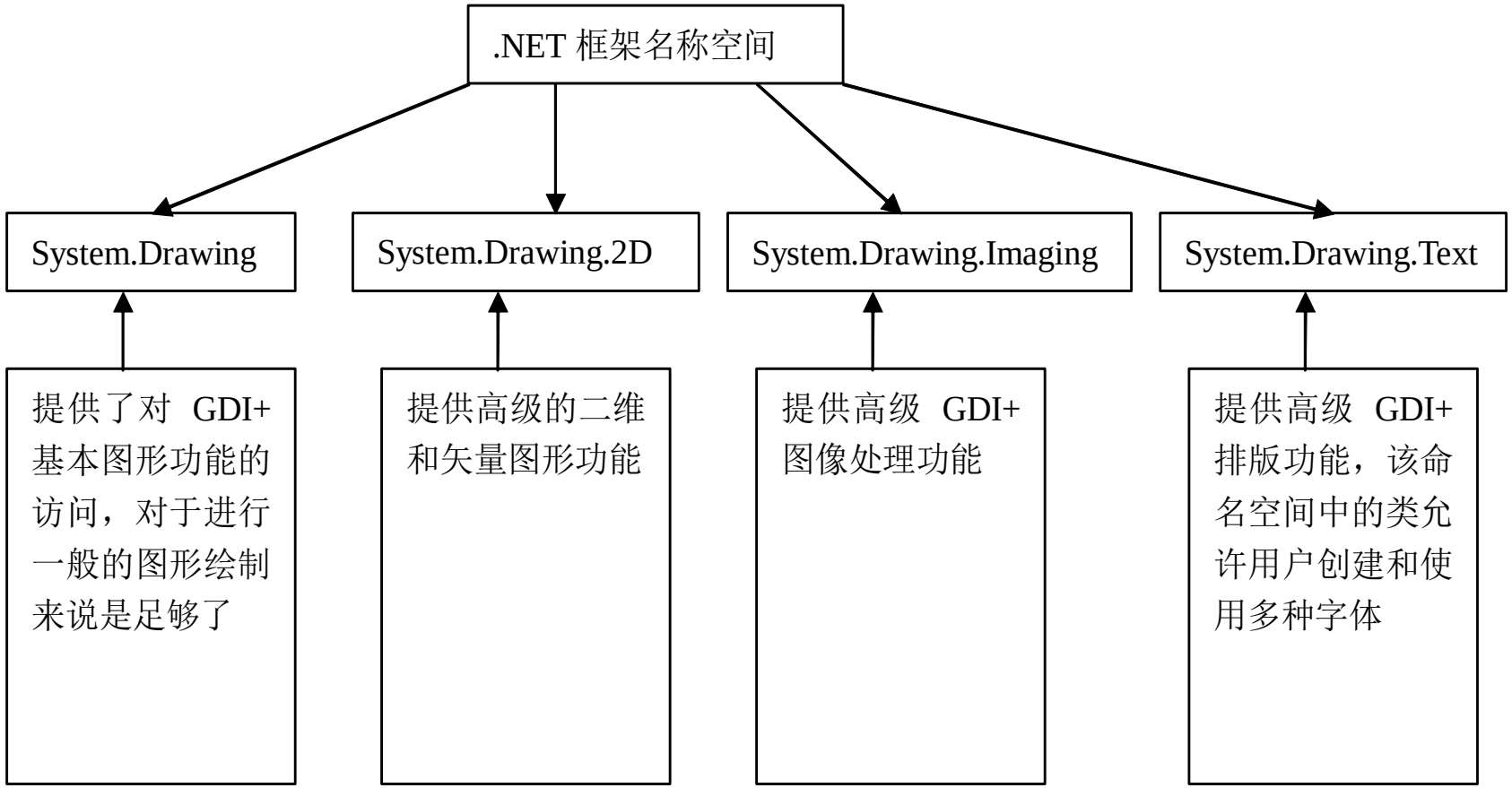
简单的示例

- 新建一个C#WindowsForm程序
- 为Form添加按钮，在Click事件中添加如下代码：




```
Graphics g = this.CreateGraphics();
Pen pen = new Pen(Color.Red, 2);
Brush brush = new SolidBrush(Color.Blue);
Font font = new Font("宋体", 24);
Rectangle rect = new Rectangle(20,120,100,160);
g.DrawLine(pen, 20, 100, 100, 100);
g.DrawRectangle(pen, rect);
g.DrawString("GDI+图形编程", font, brush, 20, 20);
brush.Dispose(); font.Dispose(); pen.Dispose();
g.Dispose();
```

.NET对GDI的封装


.NET 框架名称空间
System.Drawing
System.Drawing.2D
System.Drawing.Imaging
System.Drawing.Text
提供了对 GDI+ 基本图形功能的访问，对于进行一般的图形绘制来说是足够了
提供高级的二维和矢量图形功能
提供高级 GDI+ 图像处理功能
提供高级 GDI+ 排版功能，该命名空间中的类允许用户创建和使用多种字体

坐标系统

- GDI+的坐标系统

System.Drawing中常用的结构

- Color
  - Color封装了对颜色的定义。该结构中封装了 数百个分别对应与标准调色板色彩的静态成员。如Color.Red代表红色，Color.Purple代表紫色
  - 有用的静态方法:
    - FromArgb:通过三原色构建Color对象
    - FromKnownColor:通过已知颜色构建Color对象
    - FromName:通过颜色名称来构建Color对象
  - 例如：
    - Color temp1 = Color.Black;
    - Color temp2 = Color.FromArgb(0,0,0)
    - Color temp3 = Color.FromName(“Black”);

Size和SizeF

- 表示绘制平面上的一个尺寸，一个为整数，一个为浮点数
- 构造函数
  - Size sz1 = new Size(10,10)
- 属性：
  - Width: 表示宽度值
  - Height:表示高度值
- 重载了加、减、比较、赋值操作
  - Size sz2 = sz1;
  - Size sz3 = sz1 -sz2;
  - Size sz4 = sz1+sz2;

Point和PointF

- 表示绘制平面上点的坐标，一个为整数，另外一个为浮点数
- 构造方法：
  - Point pt = new Point(20,20);
  - Point pt = new Point(new Size(10,10))
- 重载了加、减、比较、赋值操作

Rectangle和RectangleF

- 表示绘制平面上的一个矩形区域
- 属性
  - Bottom:矩形底部的纵座标
  - Top:矩形顶部的纵座标
  - Left:矩形坐部的横座标
  - Right:矩形右部的横座标
  - Height:矩形的高度
  - Width:矩形的宽度
  - Size:矩形的尺寸
  - IsEmpty:矩形是否为空（高度和宽度是否都是0）
  - X：矩形左上角的横座标
  - Y：矩形左上角的纵座标

2. Graphics对象及绘图方法

2

绘图 Graphics对象及绘图方法 图像

得到Graphics对象

- OnPaint 事件中使用

```
Protected override void OnPaint(PaintEventArgs e){
        Graphics g = e.Graphics;
        ……
}
```

- 在其他情况使用
  - Graphics g = this.CreateGraphics();

关于Graphics的释放

- 对于CreateGraphics()得到的Graphics对象，
- 系统会自动释放，也可以显式地释放：
  - g.Dispose();
- 用这样也可以 using(Graphics g = this.CreateGraphics()) { .... }
  - 这种写法，相当于 try{.....} finally{ g.Dispose(); }

Graphics对象绘图方法

- DrawArc:绘制圆弧
- DrawBezier:绘制贝塞尔曲线
- DrawBeziers:绘制贝塞尔曲线组
- DrawClosedCurve:绘制封闭曲线
- DrawCurve:绘制曲线
- DrawEllipse:绘制椭圆
- DrawIcon:绘制图标
- DrawIconUnstretched:无缩放绘制图标
- DrawImage:绘制图像
- DrawImageUnscaled:无缩放绘制图像
- DrawLine:绘制直线
- DrawLines:绘制直线组
- DrawPath:绘制GraphicsPath对象
- DrawPie:绘制圆饼
- DrawPolygon:绘制多边形
- DrawRectangle:绘制矩形
- DrawRectangles:绘制矩形组
- DrawString:绘制文本

Graphics对象绘图方法

- Graphics对象绘制实心图形方法
  - FillClosedCurve:绘制实心封闭曲线
  - FillEllipe:封闭实心椭圆
  - FillPath:GraphicsPath对象
  - FillPie:绘制实心圆饼
  - FillPolygon:绘制实心多边形
  - FillRectangle:绘制实心矩形
  - FillRectangles:绘制实心矩形组
  - FillRegion:绘制实心Region对象

Pen 画笔

- Pen
  - 在System.Drawing名称空间中
  - 用来指定图形的轮廓，如颜色和宽度等
  - 画笔创建
    - Pen pen = new Pen(Color.Blue,5)
  - 使用Pens类，直接用系统定义好的Pen
    - 如 Pens.Red Pens.Blue

• 画笔的属性：

属性	描述	取值
Alignment	指定相对于理论上、零宽度的线条的 Pen 对象的对齐方式	PenAlignment.Center:位于所绘制线条的中央 PenAlignment:Insert:位于所绘制线条的嵌入内部 PenAlignment.Left:位于所绘制线条的左侧 PenAlignment.OutSet:位于所绘制线条的嵌入外部 PenAlignment.Right:位于所绘制线条的右侧
DashStyle	绘制线条的虚线类型	DashStyle.Custom:用户自定义 DashStyle.Dash:线条由线段组成 DashStyle.DashDot:线条由线段和点组成 DashStyle.DashDotDot:线条由线段、点和点组成 DashStyle.Dot:线条由点组成 DashStyle.Solid:线条由实线组成
StartCap EndCap	绘制线条的起点和终点类型 LineCap	**LAnchorMask**指定用于检查线帽是否为锚头帽的掩码。**ArrowAnchor**指定箭头状锚头帽。Custom指定自定义线帽。**DiamondAnchor**指定菱形锚头帽。Flat指定平线帽。**NoAnchor**指定没有锚。Round指定圆线帽。**RoundAnchor**指定圆锚头帽。Square指定方线帽。**SquareAnchor**指定方锚头帽。Triangle指定三角线帽。

Brush 画刷

- Brush是一个抽象类，不能被直接new实例化。
- 它有 5 个派生类，分别实行不同类型的画刷
  - SolidBrush:实心画刷（最简单）
  - HatchBrush:带阴影线的画刷
  - LinearGradientBrush:填充颜色线性渐变的画刷
  - PathGradientBrush:填充颜色沿路径渐变的画刷
  - TextureBrush:使用图像进行填充的画刷
- 使用Brushes类
  - 如 Brushes.Red，　Brushes.Yellow

绘制文本

1. Font类

FontFamiliy:字体家族，如Times New Roman、宋体等

- 字体大小:float类型
- 字体风格
  - Bold:粗体
  - Italic：斜体
  - Regular:正规
  - Strikeout：加删除线
  - Underline：加下划线
- 例如：

Font myFont = new Font(“宋体” ,16, FontStyle.Bold | FontStyle.Italic);

以上代码创建了宋体家族的字体对象，字体大小为１６，样式为粗斜体。

这里用|（按拉或）运算来组合字体的风格

DrawString方法

- DrawString(string,Font,Brush,PointF);
- DrawString(string,Font,Brush,RectangleF);
- DrawString(string,Font,Brush,PointF,StringFormat);
- DrawString(string,Font,Brush,RectangleF,StringFormat);
- DrawString(string,Font,Brush,float,float);
- DrawString(string,Font,Brush,float,float,StringFormat);

示例：绘制函数图

示例：万花筒

- 参见WinGDI目录

示例：分形图

更多示例

- 王宝强贱图生成器

3. 控件重绘与DoubleBuffer

3

绘图与DoubleBuffer 控件重绘 图像

处理重绘和无效操作

- 调用以下几个方法
  - void Invalidate();
  - void Invalidate(Rectangle) ;
    - 使控件的特定区域无效并向控件发送绘制消息
  - void Update();
    - 使控件重绘其工作区内的无效区域
  - void Refresh();
    - 相当于 this.Invalidate(true); this.Update();

双缓冲技术

- 1、在内存中建立一块“虚拟画布”：
  - Bitmap bmp = new Bitmap(600, 600);
- 2、获取这块内存画布的Graphics引用：
  - Graphics g = Graphics.FromImage(bmp);
- 3、在这块内存画布上绘图：
  - g.FillEllipse(brush, i * 10, j * 10, 10, 10);
  - g.DrawLine()　g.DrawString() 等等
- 4、将内存画布画到窗口中
  - this.CreateGraphics().DrawImage(bmp, 0, 0);

控件的DoubleBuffered属性

- 在framework3以上版本中有

4. 图像处理

4

Bitmap类与图像处理

绘图 图像

GDI+中的图像处理

- GDI+中对图像处理提供了以下支持：
  - 支持BMP、GIF、JPEG、PNG、TIFF、ICON等等广泛格式的图像文件
  - 提供了用于多种光栅图像格式进行编码和解码的公共接口
  - 支持为图像格式添加动态格式
  - 支持对图像的像素进行多种处理，包括亮度、对比度、颜色平衡、模糊、消弱等
  - 支持对图像进行旋转、剪切等操作
- 主要通过Image实现

Bitmap类

- Image是抽象类，Bitmap从Image派生
- 可以处理BMP、Jpeg、GIF、PNG等格式
- 构建
  - Bitmap bt1 = new Bitmap(“c:\\1.bmp”);
  - Bitmap bt2 = new Bitmap(bt1,200,300);
  - Bitmap bt3;

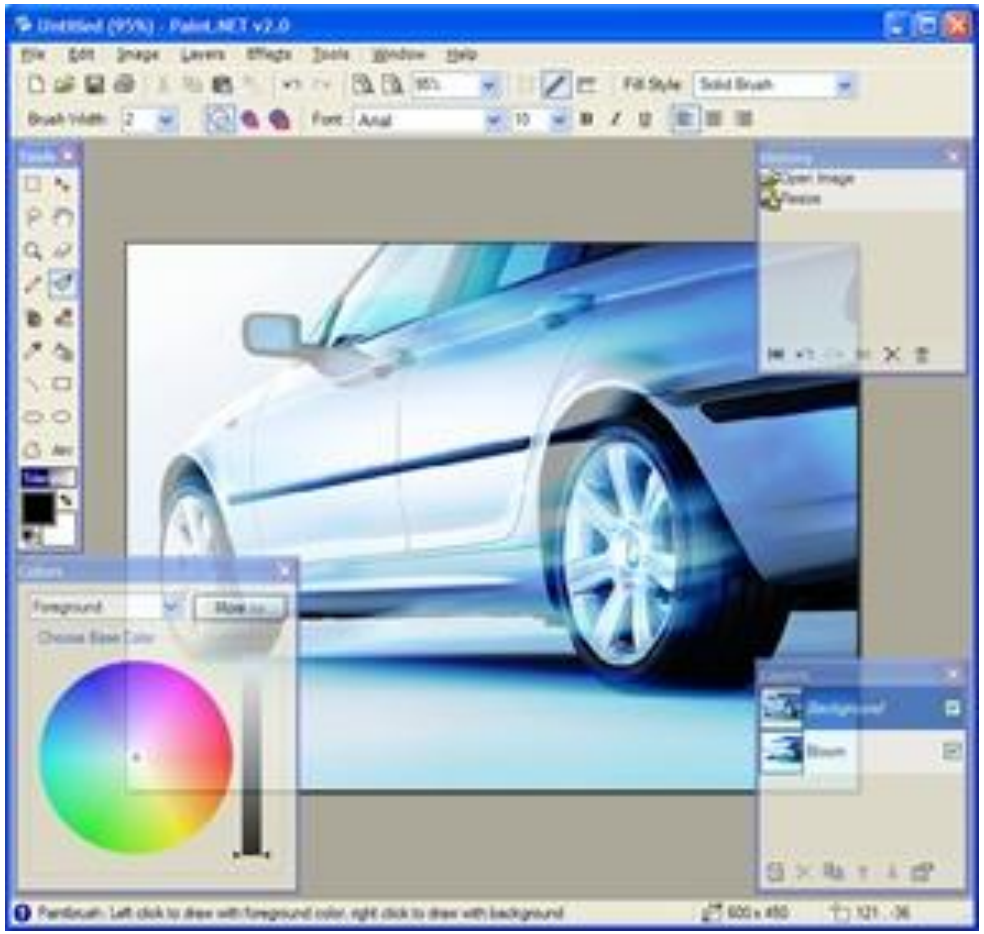
    bt3.FromFile(“文件名称”);

常见的处理方法

- 整个图像的处理
  - DrawImage
  - 示例 ThumbnailTest.cs
- 针对图像像素的处理

Paint.NET

- http://www.eecs.wsu.edu/paint.net/



5. 应用示例

- Splatter
- DrawRunningCurve
- ScreenSaver
- GravityBall

关于Transform

- myGraphics.DrawEllipse(myPen, 0, 0, 100, 50);
- myGraphics.ScaleTransform(1, 0.5f);
  myGraphics.TranslateTransform(50, 0, atrixOrder.Append);
- myGraphics.RotateTransform(30, MatrixOrder.Append);
- myGraphics.DrawEllipse(myPen, 0, 0, 100, 50);
- myGraphics.Transfrom 是一个Matrix

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Scale}} \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 50 & 0 & 1 \end{bmatrix}}_{\text{Translate}} \underbrace{\begin{bmatrix} \cos 30^\circ & \sin 30^\circ & 0 \\ -\sin 30^\circ & \cos 30^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Rotate}} = \begin{bmatrix} \cos 30^\circ & \sin 30^\circ & 0 \\ -0.5\sin 30^\circ & 0.5\cos 30^\circ & 0 \\ 50\cos 30^\circ & 50\sin 30^\circ & 1 \end{bmatrix}$$

关于Path、关于渐变Brush

图像的处理

- 图像文件bmp的格式
- 使用Lock及指针
- 几个例子

- http://code.google.com/p/aforge

问题与讨论

dstang2000@263.net