

 万水计算机技术实用大全系列

Visual C++ .NET

精彩编程实例集锦

罗斌 等编著



包含所有实例的源代码



中国水利水电出版社
www.waterpub.com.cn

万水计算机技术实用大全系列

Visual C++.NET 精彩编程实例集锦

罗 斌 等编著

中国水利水电出版社

内 容 提 要

本书用 150 个独立的实例程序介绍了 Microsoft Visual C++.NET 2003 托管编程技术。书中实例程序基于 Microsoft .NET Framework, 书中所有内容摒弃语法说教, 侧重实际应用, 主要内容包括: 使用 GDI+ 技术开发图形特效程序技巧、使用 ADO.NET 开发数据库程序技巧、使用 Crystal Report 水晶报表工具开发报表程序技巧及网络程序设计技巧等。本书还介绍了托管状态下的文件操作、加密解密、系统管理、注册表应用、界面设计等难得一见的精彩编程技术。本书所有实例程序在配套光盘中均有源代码, 让您所见即所得、所学即所用。

本书适合于使用 Microsoft Visual C++.NET 2003 进行 .NET Framework 程序开发的初、中级用户参考, 也可作为高等院校及计算机培训学校在校学生的学习案例。

图书在版编目 (CIP) 数据

Visual C++.NET 精彩编程实例集锦 / 罗斌等编著. —北京: 中国水利水电出版社, 2005

(万水计算机技术实用大全系列)

ISBN 7-5084-3343-2

I. V... II. 罗... III. C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2005) 第 119315 号

书 名	Visual C++.NET 精彩编程实例集锦
作 者	罗 斌 等编著
出版 发行	中国水利水电出版社 (北京市三里河路 6 号 100044) 网址: www.waterpub.com.cn E-mail: mchannel@263.net (万水) sales@waterpub.com.cn
经 售	电话: (010) 63202266 (总机)、68331835 (营销中心)、82562819 (万水) 全国各地新华书店和相关出版物销售网点
排 版	北京万水电子信息有限公司
印 刷	北京市天竺颖华印刷厂
规 格	787mm×1092mm 16 开本 28.25 印张 696 千字
版 次	2006 年 1 月第 1 版 2006 年 1 月第 1 次印刷
印 数	0001—5000 册
定 价	48.00 元 (含 1CD)

凡购买我社图书, 如有缺页、倒页、脱页的, 本社营销中心负责调换
版权所有·侵权必究

前 言

随着互联网的不断发展和深入，基于.NET 平台的应用软件开发必将渗透到每一个角落。学习.NET、应用.NET 技术将成为微软大旗下的程序员们的必然选择。而对于 Visual C++ 程序员来说，这或许是一次难得的机遇。MFC 在 Visual C++ 这辆战车上奋战十几年后，几乎成了 Visual C++ 的代名词；然而，从最近几次的版本更新可以看出，MFC 进步甚少，没有与 Borland C++ 同在当时那种快速发展的感觉，因此有的程序员惊呼：Visual C++ 在打败 Borland C++ 以后，就开始垄断经营而不思进取了吗？事实上，当我们把目光转向 Visual C++ .NET 的另一个主力技术，一个可以完全脱离于 MFC 的全新技术——托管 C++ 时，就会发现一个令人惊奇的世界，这就是.NET。

.NET 是微软在 2000 年推出的下一代互联网软件和服务战略，它旨在实现下列目标：

(1) 提供一个一致的面向对象的编程环境，而无论对象代码是在本地存储和执行，还是在本地执行但在 Internet 上分布，或者是在远程执行的。

(2) 提供一个将软件部署和版本控制冲突最小化的代码执行环境。

(3) 提供一个保证代码安全执行的代码执行环境。

(4) 提供一个可消除脚本环境或解释环境的性能问题的代码执行环境。

(5) 使开发人员的经验在面对类型大不相同的应用程序时保持一致。

(6) 按照工业标准生成所有通信，以确保基于.NET Framework 的代码可与任何其他代码集成。很明显，使用 MFC 技术实现.NET 目标可谓鞭长莫及。为了将.NET 的最新技术引入 Visual C++，在 Visual C++ .NET 中，微软推出了托管 C++ 语言，该语言建立在 C++ 语言基础上，但它与 C++ 有着本质的区别，它能够在 Visual C++ .NET 环境中开发出具有如垃圾自动收集、程序间相互访问等高性能的真正的.NET Framework 托管应用程序。

本书不是一本详细讲解托管 C++ 编程语言语法和概念的编程指导书，使用托管 C++ 语言、用鲜活的实例诠释.NET Framework 的精华、释放.NET 的威力是本书编写时追求的目标。全书由简到繁、从易到难共分为程序界面编程实例、程序控制编程实例、系统管理编程实例、GDI+ 图像编程实例和数据库网络编程实例共五大部分 150 个编程实例，每一个编程实例都给出了简要的编写说明、具体的实现步骤和完整的源代码（见光盘），每一个编程实例都至少介绍一种方法、解决一个问题，这些方法和问题可能与语言有关、可能与系统有关、可能与应用有关，总之，它们都可能是您在使用.NET 解决问题时可能遇到的问题。

本书适合于使用 Microsoft Visual C++ .NET 2003 进行.NET Framework 程序开发的初、中级用户参考，也可作为高等院校及计算机培训学校在校学生的学习案例。书中所有内容和思想并非一人之力所能及，而是凝聚了众多编程高手多年的心得并经过充分的提炼和总结而成，在此对他们的智慧表示崇高的敬意和衷心的感谢！本书编写人员包括罗兴禄、雷

治英、罗斌、罗帅、雷玲、曹勇、雷丽、李平、雷平、邓承会、雷国忠、王年素、雷浩、李佳星、范刚强、杨琴、罗顺文、吴多、王彬、张志红、罗兰、刘显国和刘婷等，终稿由罗斌统筹完成。书中的所有实例程序均在 Microsoft Visual Studio.NET 2003/Microsoft Office 2003 /Microsoft Windows XP 环境中编译执行成功，因此，请在以上环境中进行实例程序的学习，由于作者水平有限，其中也可能存在一些疏漏和不当之处，敬请读者批评指正。联系邮箱 binluobin@163.com。

罗斌

2005年10月于重庆长寿

目 录

前言

第 1 章 程序界面编程实例	1	实例 29 如何操作输入法	57
实例 1 如何创建不规则窗体	1	实例 30 如何操作定时器	60
实例 2 如何创建图形化窗体	2	实例 31 如何操作帮助文件	63
实例 3 如何创建文字形窗体	4	实例 32 如何操作文件属性	66
实例 4 如何创建半透明窗体	5	实例 33 如何操作文件夹时间属性	70
实例 5 如何创建全屏幕窗体	7	实例 34 如何使用字体颜色对话框	74
实例 6 如何创建标签页窗体	8	实例 35 如何在程序中启动其他程序	77
实例 7 如何创建托盘窗体	11	实例 36 如何在程序中关闭其他程序	79
实例 8 如何创建分隔窗体	13	实例 37 如何控制字符串输出方向	83
实例 9 如何创建闪屏窗体	16	实例 38 如何分栏显示字符串	86
实例 10 如何创建精灵窗体	18	实例 39 如何省略显示字符串	89
实例 11 如何拖动无标题栏窗体	21	实例 40 如何定位字符串	92
实例 12 如何隐藏标题栏按钮	24	实例 41 如何替换字符串	95
实例 13 如何设置标题栏图标	25	实例 42 如何监视系统文件变化	98
实例 14 如何设置标题栏文字	26	实例 43 如何获取程序版本信息	101
实例 15 如何禁止最大化按钮	27	实例 44 如何获取程序路径信息	104
实例 16 如何禁止最小化按钮	29	实例 45 如何给文本添加行号	106
实例 17 如何定制工具栏	30	实例 46 如何读写 INI 文件	110
实例 18 如何定制状态栏	34	实例 47 如何读写文本文件	113
实例 19 如何定制消息提示框	37	实例 48 如何打印文本文件	118
实例 20 如何定制打开对话框	40	实例 49 如何实现打印预览	121
实例 21 如何定制保存对话框	42	实例 50 如何切分组合文件	127
实例 22 如何设置程序光标	45	实例 51 如何加密解密文件	132
实例 23 如何设置程序背景图像	46	实例 52 如何递归搜索文件	139
实例 24 如何设置程序启动位置	48	实例 53 如何选择多个文件	143
实例 25 如何实现中英文菜单切换	49	实例 54 如何转换长文件名	144
实例 26 如何显示简单关于对话框	52	实例 55 如何设置共享文件夹	146
实例 27 如何禁止程序在任务栏上出现	53	实例 56 如何设置功能键	149
实例 28 如何使程序窗体始终显示 在最前面	55	实例 57 如何添加快捷菜单	153
第 2 章 程序控制编程实例	57	实例 58 如何模拟键盘输入	155
		实例 59 如何遍历窗体所有控件	158

实例 60	如何创建字体对话框	159	实例 96	如何操作系统注册表	263
实例 61	如何创建多文档程序	165	第 4 章 GDI+ 图像编程实例	270	
实例 62	如何创建多线程程序	171	实例 97	如何变换颜色渐变图形	270
实例 63	如何创建进程跟踪器	175	实例 98	如何设置色彩输出通道	272
实例 64	如何创建安装程序	178	实例 99	如何控制图片缩放质量	275
实例 65	如何创建临时文件	182	实例 100	如何使用色彩变换矩阵	278
实例 66	如何制作 MP3 播放器	184	实例 101	如何使用调色板校正图片	284
实例 67	如何制作 RealPlayer 播放器	190	实例 102	如何使用阈值显示图片	286
第 3 章 系统管理编程实例	194		实例 103	如何淡入淡出显示图片	288
实例 68	如何获取 CPU 序列号	194	实例 104	如何以底片效果显示图片	291
实例 69	如何获取硬盘 ID 号	195	实例 105	如何以浮雕效果显示图片	293
实例 70	如何获取驱动器卷标	197	实例 106	如何以光照效果显示图片	296
实例 71	如何获取驱动器类型	199	实例 107	如何以灰度效果显示图片	298
实例 72	如何获取驱动器容量	202	实例 108	如何以木刻效果显示图片	301
实例 73	如何获取系统主板信息	204	实例 109	如何以柔化效果显示图片	303
实例 74	如何获取系统基本信息	206	实例 110	如何以锐化效果显示图片	305
实例 75	如何获取系统服务信息	210	实例 111	如何以油画效果显示图片	307
实例 76	如何获取系统日志信息	215	实例 112	如何旋转显示图片	309
实例 77	如何获取系统共享资源	221	实例 113	如何分块显示图片	311
实例 78	如何获取系统计数器信息	222	实例 114	如何缩放显示图片	314
实例 79	如何获取系统已安装字体	226	实例 115	如何投射显示图片	316
实例 80	如何获取系统特殊文件夹信息	227	实例 116	如何镜像显示图片	318
实例 81	如何获取系统文化和国家信息	230	实例 117	如何缩放显示文字	321
实例 82	如何启动系统日期设置程序	233	实例 118	如何绘制阴影字	322
实例 83	如何启动系统声音设置程序	235	实例 119	如何绘制投射字	324
实例 84	如何启动系统鼠标设置程序	237	实例 120	如何绘制渐变色文字	326
实例 85	如何启动系统添加删除程序	240	实例 121	如何绘制大小渐变文字	328
实例 86	如何启动系统显示设置程序	242	实例 122	如何绘制快捷键下划线	330
实例 87	如何启动系统语言设置程序	244	实例 123	如何绘制渐变透明图形	332
实例 88	如何启动系统属性设置程序	246	实例 124	如何绘制颜色渐变图形	334
实例 89	如何启动系统电话拨号程序	249	实例 125	如何绘制汽车速度计	336
实例 90	如何启动系统屏幕保护程序	250	实例 126	如何旋转图片颜色	338
实例 91	如何启动系统查找对话框	252	实例 127	如何获取图像缩略图	344
实例 92	如何启动文件关联程序	253	实例 128	如何在图像上添加标记	346
实例 93	如何隐藏显示桌面图标	256	实例 129	如何在桌面上显示图像	349
实例 94	如何设置系统桌面墙纸	258	实例 130	如何实现复制粘贴图像	351
实例 95	如何操作系统 API 函数	260	实例 131	如何转换图像文件格式	355

第 5 章 数据库网络编程实例.....	359	实例 141 如何使用水晶报表创建	
实例 132 如何获取 ODBC 驱动信息	359	打印报表	408
实例 133 如何创建 ODBC 数据源	360	实例 142 如何读写 XML 文件	414
实例 134 如何使用数据表统计信息	363	实例 143 如何发送电子邮件	421
实例 135 如何获取 Access 数据表信息	367	实例 144 如何启动邮件发送程序	424
实例 136 如何读写 Access 数据库信息	370	实例 145 如何自动生成网页	426
实例 137 如何创建 SQL Server 数据库	383	实例 146 如何下载网页内容	429
实例 138 如何读写 SQL Server		实例 147 如何创建网页浏览器	434
数据库信息	388	实例 148 如何启动网页浏览器	436
实例 139 如何读写 SQL Server		实例 149 如何启动系统 Internet	
数据库图像信息	400	设置程序	438
实例 140 如何操作数据库主从关系表	405	实例 150 如何获取主机 IP 信息	441

第 1 章 程序界面编程实例

实例 1 如何创建不规则窗体

实例目的

此实例主要介绍了如何通过设置程序窗体属性创建不规则应用程序窗体的编程技术。实例程序执行后，将显示一个不规则的应用程序窗体，效果如图 1-1 所示。

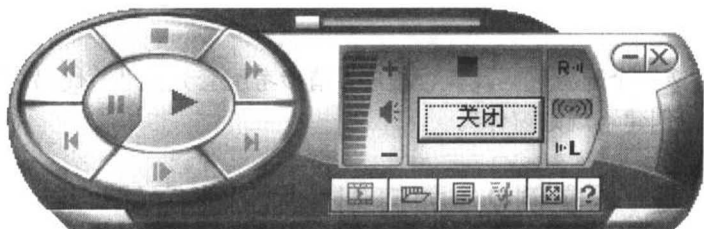


图 1-1

实现方法

在使用 MFC 或 API 函数根据图像创建不规则的应用程序窗体时，通常是通过读取图像颜色值，然后再将这些颜色值与需要掩盖的颜色值进行运算，才能得到这种效果。然而，在 Visual C++ .NET 中，当使用 .NET 技术创建一个 Form 时，只需要设置这个 Form 的 TransparencyKey 属性即可创建一个不规则的应用程序窗体。当设置 Form（窗体）的 TransparencyKey 属性的颜色值后，窗体背景具有相同 TransparencyKey 属性所指定颜色的窗体区域将透明显示；在窗体的透明区域执行的任何鼠标操作（如鼠标单击）都将传递到该透明区域下的窗口。例如，如果使窗体工作区的某部分透明，则在该区域上单击鼠标会将单击的事件通知发送到该区域下的任何窗口。下面是实例程序据此创建不规则的应用程序窗体的具体操作步骤：

(1) 在画图或其他图像处理软件中制作一幅不规则的图像并保存为一个图像文件，注意其透明色应为一个确定的颜色，该颜色必须与 Form 的 TransparencyKey 属性指定的颜色值一致，实例程序选择的是白色，如图 1-1 所示。

(2) 启动 Visual C++ .NET，选择“文件\新建\项目\Visual C++项目\ .NET\Windows 窗体应用程序 (.NET)”，创建一个 Visual C++ Windows (.NET) 应用程序框架。

(3) 在 Form1 窗体的设计视图中（如果 Form1.h 设计视图没有被打开，在“解决方案资源管理器”中，双击“头文件”节点下面的 Form1.h 文件名即可打开），选择该设计窗体，在属性对话框中设置其 TransparencyKey 属性为 White，设置其 FormBorderStyle 属性为 None；单击 BackgroundImage 属性右边的“...”按钮，在弹出的文件“打开”对话框中选择步骤一制

作的图像文件后，单击“打开”按钮返回。此时编译执行程序就可以看见创建的不规则应用程序窗体。

(4) 选择“视图\工具箱”菜单，指向“工具箱”的“Windows 窗体”页，在程序设计窗体上拖放一个 Button 控件，在属性对话框中设置其 Text 属性为“关闭”。然后为这个按钮的鼠标单击事件 (Click) 加入如下所示的实现代码 (在 Form1.h 设计视图中双击该按钮)：

```
private: System::Void button1_Click(System::Object * sender,
    System::EventArgs * e)
{ //关闭程序窗体
    this->Close();
}
```

最后，编译执行程序即可。

注意：如果你的操作系统设置的颜色质量为“最高 32 位”，可能按照此种方法创建的不规则应用程序窗体无效，在此情况下，在系统显示属性中将颜色质量设置为较低的值即可。

实例源代码

有关程序项目文件全部源代码及执行程序，请参考配套光盘“源代码\第 1 章 程序界面编程实例\实例 1 如何创建不规则窗体”。

实例 2 如何创建图形化窗体

实例目的

此实例主要介绍了如何创建图形化的应用程序窗体的编程技术。实例程序执行后，将显示一个用基数样条曲线绘制的程序窗体，效果如图 2-1 所示。



图 2-1

实现方法

在 Visual C++.NET 中，当使用 .NET 技术创建一个 Form 时，默认情况下，这个 Form 就是一个标准的矩形窗口。但是，使用 .NET 技术创建的 Form 有一个 Region 属性，该属性表示应用程序的窗体区域。通常，Region 对象除了以常规的矩形表示外，它还可以有多种形式，如圆形、椭圆形以及其他任何封闭曲线所围成的区域等，这取决于创建 Region 对象时所使用

的 `GraphicsPath` 对象中的图形。因此，通过设置 `GraphicsPath` 对象中的图形风格就可以创建多种图形化的程序窗体。`GraphicsPath` 类中用于添加图形的常用公共方法包括：

- (1) `AddArc()`方法，该方法向当前图形追加一段椭圆弧。
- (2) `AddBezier()`方法，该方法在当前图形中添加一段立方贝塞尔曲线。
- (3) `AddBeziers()`方法，该方法在当前图形中添加一系列相互连接的立方贝塞尔曲线。
- (4) `AddClosedCurve()`方法，该方法向此路径添加一个闭合曲线，由于曲线经过数组中的每个点，因此使用基数样条曲线。
- (5) `AddCurve()`方法，该方法向当前图形添加一段样条曲线，由于曲线经过数组中的每个点，因此使用基数样条曲线。
- (6) `AddEllipse()`方法，该方法向当前路径添加一个椭圆。
- (7) `AddLine()`方法，该方法向此 `GraphicsPath` 对象追加一条线段。
- (8) `AddLines()`方法，该方法向此 `GraphicsPath` 对象末尾追加一系列相互连接的线段。
- (9) `AddPath()`方法，该方法将指定的 `GraphicsPath` 对象追加到该路径。
- (10) `AddPie()`方法，该方法向此路径添加一个扇形轮廓。
- (11) `AddPolygon()`方法，该方法向此路径添加多边形。
- (12) `AddRectangle()`方法，该方法向此路径添加一个矩形。
- (13) `AddRectangles()`方法，该方法向此路径添加一系列矩形。

以下是实例程序通过在路径对象 `GraphicsPath` 中添加基数样条曲线实现创建图形化的程序窗体的具体操作步骤：

(1) 启动 Visual C++ .NET，选择“文件\新建\项目\Visual C++项目\ .NET\Windows 窗体应用程序 (.NET)”，创建一个 Visual C++ Windows (.NET) 应用程序框架。

(2) 由于创建基数样条曲线的 `GraphicsPath` 类位于 `System::Drawing::Drawing2D` 命名空间下，因此请在 `Form1.h` 文件的命名空间声明位置处添加如下所示的代码：

```
using namespace System::Drawing::Drawing2D;
```

(3) 为了在程序窗口绘制时创建路径对象，请为 `Form1` 窗体的窗口重绘事件 (`Paint`) 添加如下所示的实现代码：

```
private: System::Void Form1_Paint(System::Object * sender,
                                System::Windows::Forms::PaintEventArgs * e)
{ //创建图形化的程序窗体
    Point MyPoints[]={Point(10,50),Point(150,60),Point(200,80),
                      Point(200,100),Point(320,140),Point(80,200)};
    //创建一个路径对象
    GraphicsPath *MyPath;
    MyPath=new GraphicsPath();
    //将绘制的基数样条图形加入路径
    MyPath->AddClosedCurve(MyPoints);
    //使用基数样条图形构造一个区域，并将此区域作为程序窗体区域
    System::Drawing::Region *MyRegion=new System::Drawing::Region(MyPath);
    this->Region=MyRegion;
}
```

(4) 选择“视图\工具箱”菜单，指向“工具箱”的“Windows 窗体”页，在程序设计窗体上拖放一个 `Button` 控件，在属性对话框中设置其 `Text` 属性为“关闭”。然后为这个按钮的鼠

标单击事件 (Click) 加入如下所示的实现代码:

```
private: System::Void button1_Click(System::Object * sender,
                                   System::EventArgs * e)
{
    //关闭程序窗体
    this->Close();
}
```

(5) 在 Form1.h 文件的命名空间声明处添加如下所示的声明代码:

```
using namespace System::Drawing::Drawing2D;
最后, 编译执行程序即可。
```

实例源代码

有关程序项目文件全部源代码及执行程序, 请参考配套光盘“源代码\第 1 章 程序界面编程实例\实例 2 如何创建图形化窗体”。

实例 3 如何创建文字形窗体

实例目的

此实例主要介绍了如何使用 GDI+ 技术创建文字形窗体的编程技术。实例程序执行后, 程序窗体将以“精彩”两个文字形式显示, 效果如图 3-1 所示。



图 3-1

实现方法

在 Visual C++.NET 中, 当使用 .NET 向导创建一个 .NET 应用程序时, 默认生成的窗体 Form1 有一个 Region 属性, 当设置窗体 Form1 的 Region 属性为一个具体的 Region 对象时, 如文字, 应用程序窗体就将以文字的形式显示。实例程序将把文字串对象“精彩”添加到 GraphicsPath 对象中, 然后再使用 GraphicsPath 对象构造一个 Region 对象, 从而实现创建文字形窗体。GraphicsPath 类添加文字串的公共方法 AddString() 有多种重载形式, 实例程序使用的重载形式的声明如下:

```
public: void AddString(
    String* s,                //要添加的 String 对象
    FontFamily* family,      //FontFamily 对象, 表示绘制文本所用字体的名称
    int style,               //FontStyle 枚举, 表示有关文本的样式信息
```

```

float emSize,           //限定字符的 Em (字体大小) 方框的高度
PointF origin,         //PointF 对象, 表示文本从其起始的点
StringFormat* format   //指定文本格式设置信息的 StringFormat 对象
);

```

下面是实例程序通过设置应用程序窗体的 **Region** 属性从而创建文字形窗体的具体操作步骤:

(1) 启动 Visual C++.NET, 选择“文件\新建\项目\Visual C++项目\NET\Windows 窗体应用程序 (.NET)”, 创建一个 Visual C++ Windows (.NET) 应用程序框架。

(2) 为应用程序窗体 Form1 的 Load 事件添加如下所示的实现代码:

```

private: System::Void Form1_Load(System::Object *sender, System::EventArgs *e)
{ //创建文字形窗体
    Graphics* g=this->CreateGraphics();
    g->Clear(Color::White);
    //设定文本输出质量
    g->set_TextRenderingHint
        (System::Drawing::Text::TextRenderingHint::ClearTypeGridFit);
    g->set_SmoothingMode(System::Drawing::Drawing2D::SmoothingMode::AntiAlias);
    FontFamily* MyFamily=new FontFamily("隶书");
    //创建路径区域
    System::Drawing::Drawing2D::GraphicsPath* MyPath=
new System::Drawing::Drawing2D::GraphicsPath();
    //向区域中追加文本
    StringFormat* MyFormat=new StringFormat();
    MyPath->AddString("精彩",MyFamily,(int)FontStyle::Regular,200.0f,Point
(0,30),MyFormat);
    Pen* MyPen=new Pen(Color::Red);
    //绘制路径
    g->DrawPath(MyPen,MyPath);
    //从路径中构造区域
    System::Drawing::Region* MyRegion=new System::Drawing::Region(MyPath);
    this->Region=MyRegion;
}

```

最后, 编译执行程序即可。

实例源代码

有关程序项目文件全部源代码及执行程序, 请参考配套光盘“源代码\第1章 程序界面编程实例\实例3 如何创建文字形窗体”。

实例4 如何创建半透明窗体

实例目的

此实例主要介绍了如何以半透明方式显示程序窗体的编程技术。实例程序执行后, 拖动刻度条的滑块控件, 即可改变程序窗体的透明度, 效果如图 4-1 所示。



图 4-1

实现方法

在 Visual C++.NET 中, 当使用 .NET 技术创建一个 Windows 窗体应用程序时, 通过设置应用程序窗体 (Form) 的 `Opacity` 属性可以改变应用程序窗体的透明度。Opacity 属性是一个百分比值, 100% 表示完全不透明 (这也是程序窗体的默认值), 0% 表示完全透明。在完全透明的情况下, 程序窗体基本看不见; 可以使用此属性提供不同级别的透明度, 据此实现窗体逐渐进入或逐渐退出用户视野的效果, 例如, 通过将 `Opacity` 属性设置为值 0% (0.00), 并逐渐增加该值直到它到达 100% (1.00), 可以使程序窗体实现逐渐进入用户视野的效果。此属性与应用程序窗体的 `TransparencyKey` 属性提供的透明度不同, 后者只能使窗体及其控件完全透明, 条件是窗体及其控件与 `TransparencyKey` 属性中指定的值所表示的颜色相同。实例程序将使用一个刻度条控件, 在 0~100% 范围内任意改变刻度条的值, 达到任意改变程序窗体透明度的目的。以下是实例程序据此实现半透明显示程序窗体的具体操作步骤:

(1) 启动 Visual C++.NET, 选择“文件\新建\项目\Visual C++项目\ .NET\Windows 窗体应用程序 (.NET)”, 创建一个 Visual C++ Windows (.NET) 应用程序框架。

(2) 选择“视图\工具箱”菜单, 指向“工具箱”的“Windows 窗体”页, 在程序设计窗体上拖放一个 `TrackBar` 控件 (`trackBar1`), 在属性对话框中设置其 `TickStyle` 属性为 `BottomRight`, 设置其 `Value` 属性为 25, 设置其 `SmallChange` 属性为 2, 设置其 `Maximum` 属性为 50, 设置其 `Minimum` 属性为 0。然后为 `TrackBar` 控件 (`trackBar1`) 的刻度值改变事件 (`ValueChanged`) 添加如下所示的实现代码:

```
private: System::Void trackBar1_ValueChanged(System::Object * sender,
                                           System::EventArgs * e)
{
    //改变程序窗体透明度
    this->Opacity=(float)(100-this->trackBar1->Value*2)/100;
}
```

最后, 编译执行程序即可。

实例源代码

有关程序项目文件全部源代码及执行程序, 请参考配套光盘“源代码\第 1 章 程序界面编程实例\实例 4 如何创建半透明窗体”。

实例 5 如何创建全屏幕窗体

实例目的

此实例主要介绍了如何以全屏幕方式、最大化方式、最小化方式和正常方式显示程序窗体的编程技术。实例程序执行后，单击“最大化显示”按钮，则程序窗体将以最大化方式显示；单击“最小化显示”按钮，则程序窗体将以最小化方式显示；单击“正常显示”按钮，则程序窗体将以正常方式显示，效果如图 5-1 所示；单击“全屏幕显示”按钮，则程序窗体界面将充满整个屏幕。

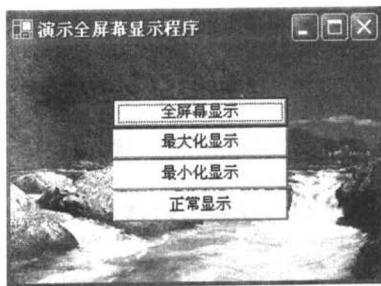


图 5-1

实现方法

在 Visual C++.NET 中，当使用 .NET 技术创建一个 Windows 窗体应用程序时，通过设置应用程序窗体 (Form) 的 WindowState 属性可以设置应用程序窗体的显示方式。WindowState 属性是一个 FormWindowState 枚举类型，该枚举类型的成员如表 5-1 所示。

表 5-1

成员名称	说明
Maximized	最大化的窗口
Minimized	最小化的窗口
Normal	默认大小的窗口

此外，应用程序窗体 (Form) 还有一个 FormBorderStyle 属性 (关于该属性的说明请参考本书的其他部分)，如果将 FormBorderStyle 属性设置为 FormBorderStyle::None，并且设置窗体 (Form) 的 WindowState 属性为 FormWindowState::Maximized，则应用程序窗体就会以全屏幕方式显示。以下是实例程序据此实现以全屏幕方式、最大化方式、最小化方式和正常方式显示程序窗体的具体操作步骤：

(1) 启动 Visual C++.NET，选择“文件\新建\项目\Visual C++项目\ .NET\Windows 窗体应用程序 (.NET)”，创建一个 Visual C++ Windows (.NET) 应用程序框架。

(2) 选择“视图\工具箱”菜单，指向“工具箱”的“Windows 窗体”页，在程序设计窗体上拖放四个 Button 控件，在属性对话框中设置四个 Button 控件的 Text 属性分别为“全屏幕

显示”、“最大化显示”、“最小化显示”和“正常显示”，并为这四个按钮控件的鼠标单击事件（Click）加入如下所示的实现代码：

```
private: System::Void button1_Click(System::Object * sender, System::EventArgs *e)
{
    //全屏显示程序窗体
    this->FormBorderStyle=FormBorderStyle::None;
    this->WindowState=FormWindowState::Maximized;
}
private: System::Void button2_Click(System::Object * sender, System::EventArgs *e)
{
    //最小化显示程序窗体
    this->FormBorderStyle=FormBorderStyle::Sizable;
    this->WindowState=FormWindowState::Minimized;
}
private: System::Void button4_Click(System::Object * sender, System::EventArgs *e)
{
    //最大化显示程序窗体
    this->FormBorderStyle=FormBorderStyle::Sizable;
    this->WindowState=FormWindowState::Maximized;
}
private: System::Void button3_Click(System::Object * sender, System::EventArgs *e)
{
    //正常显示程序窗体
    this->FormBorderStyle=FormBorderStyle::Sizable;
    this->WindowState=FormWindowState::Normal;
}
}
```

最后，编译执行程序即可。

实例源代码

有关程序项目文件全部源代码及执行程序，请参考配套光盘“源代码第1章 程序界面编程实例\实例5 如何创建全屏幕窗体”。

实例6 如何创建标签页窗体

实例目的

此实例主要介绍了如何使用.NET 工具箱中的标签页控件（TabControl）创建标签页窗体界面的编程技术。实例程序执行后，单击不同标题的标签会显示不同的图片，效果如图6-1所示。



图 6-1

实现方法

.NET 工具箱中的标签页控件 (TabControl) 是一个包括 TabPage 对象的容器。在这个容器中, 用户可以方便地增加 (Add()方法) 和移除 (Remove()方法) 标签页 TabPage 对象, 也可以设置标签的图标, 甚至多行显示标签页。该控件的标签页 TabPage 对象存放在其 TabPages 属性中, 标签页图标存放在其 ImageList/Images 属性中。TabControl 控件常用的公共属性包括:

- (1) DrawMode 属性, 该属性用于获取或设置绘制控件的标签页方式。
- (2) ImageList 属性, 该属性用于获取或设置在控件的标签页上显示的图像。
- (3) ItemSize 属性, 该属性用于获取或设置控件的标签页大小。
- (4) Multiline 属性, 该属性用于获取或设置一个值, 该值指示是否可以显示一行以上的标签页。
- (5) RowCount 属性, 该属性用于获取控件的标签页条中当前正显示的行数。
- (6) SelectedIndex 属性, 该属性用于获取或设置当前选定的标签页索引。
- (7) SelectedTab 属性, 该属性用于获取或设置当前选定的标签页。
- (8) ShowToolTips 属性, 该属性用于获取或设置一个值, 该值指示当鼠标移到标签页上时是否显示该标签页的“工具提示”。
- (9) SizeMode 属性, 该属性用于获取或设置调整控件的标签页大小的方式。
- (10) TabCount 属性, 该属性用于获取标签页条中标签页的数目。
- (11) TabPages 属性, 该属性用于获取该标签页控件中的标签页集合。

下面是实例程序使用 TabControl 控件创建标签页程序窗体的具体步骤:

- (1) 启动 Visual C++.NET, 选择“文件\新建\项目\Visual C++项目\ .NET\Windows 窗体应用程序 (.NET)”, 创建一个 Visual C++ Windows (.NET) 应用程序框架。
- (2) 选择“视图\工具箱”菜单, 指向“工具箱”的“Windows 窗体”页, 在程序设计窗体上拖放一个 ImageList 控件 (imageList1), 在属性对话框中单击 Images 属性右边的“...”按钮, 弹出“Image 集合编辑器”。
- (3) 单击“Image 集合编辑器”对话框中的“添加”按钮, 弹出文件“打开”对话框, 选择一个图标文件并打开, 则该图标文件自动添加到 ImageList 控件中, 按照同样的方法添加其余两个图标, 最后效果如图 6-2 所示。

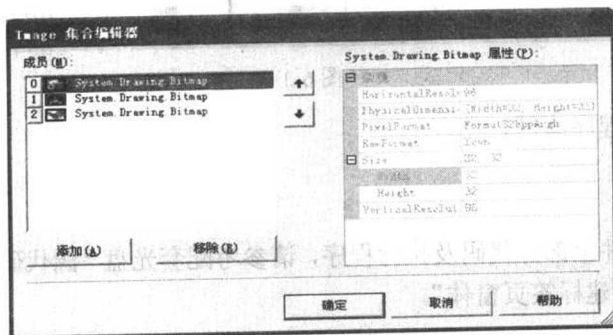


图 6-2