PEARSON
Addison Wesley

Windows Forms 程序设计

〔美〕Chris Sells 著
荣 耀 蒋贤哲 译

Windows Forms Programming in C#

人民邮电出版社
POSTS & TELECOM PRESS

Windows Forms 程序设计

■ [美] Chris Sells 著

■ 荣 耀 蒋贤哲 译

人 民 邮 电 出 版 社

Windows Forms 程序设计

◆ 著　　[美] Chris Sells
译　　荣　耀　蒋贤哲
责任编辑　陈冀康

◆ 人民邮电出版社出版发行　北京市崇文区夕照寺街 14 号
邮编　100061　电子函件　315@ptpress.com.cn
网址　http://www. ptpress.com.cn
读者热线　010-67132705
北京汉魂图文设计有限公司制作
北京顺义振华印刷厂印刷
新华书店总店北京发行所经销

◆ 开本：800×1000　1/16
印张：34.75　彩插：4
字数：836 千字　2004 年 9 月第 1 版
印数：1-4 000 册　2004 年 9 月北京第 1 次印刷

著作权合同登记　图字：01 - 2003 - 6943 号
ISBN 7-115-12489-2/TP・4115

定价：65.00 元

内 容 提 要

本书是 Microsoft.NET Forms 程序库的最佳使用指南。

全书包括 15 章和 4 个附录，对 WinForms 进行了全面而深入的讲解。前半部分讨论了窗体、对话框、GDI+以及打印等相对常见的技术；后半部分则专注于高级主题，内容涵盖设计期集成、资源、应用程序和设置、多线程用户界面以及 Web 部署等。其中第 12、13 章专门讲述了 ADO.NET 和 WinForms 的结合使用。附录分别介绍了从 MFC 转移到 WinForms、委托和事件、序列化和标准 WinForms 组件和控件。

本书适合有.NET 背景知识的 WinForms 学习者阅读，有经验的 WinForms 程序员则可从中学到大量的高级技巧。

作译者简介

Chris Sells 是微软 MSDN 内容小组策略制定者之一，原为 DevelopMentor 软件工程指导者。他是 *Windows Telephony Programming*（Addison-Wesley，1998）的作者，并且是 *Effective COM*（Addison-Wesley，1999）、*ATL Internals*（Addison-Wesley，1999）、*Essential .NET Volume I*（Addison-Wesley，2003）以及 *Windows Forms Programming in Visual Basic .NET*（Addison-Wesley，2004）的合著者。他的个人网站是 www.sellsbrothers.com。

荣耀是南京师范大学教师，原为电力自动化研究院工程师和项目经理，是《.NET 大局观》和《C++ Templates 全览》的合译者。他的技术研究领域包括 C++、C#、Java、OOP、Generic Programming 和 Design Patterns。他的个人网站是 www.royaloo.com。

蒋贤哲是南京航空航天大学自动化专业研究生，爱好程序设计，主要技术兴趣包括 C++ 和.NET。

译者序

今天，尽管基于 Web 的应用程序开发占据着主导地位，然而在许多场合我们仍然需要编写传统的独立 Windows 应用程序，比如从日常使用的单机程序到范围广泛的网络应用客户端。Windows Forms（简称 WinForms）是微软.NET 框架类库中专门用于编写传统 Windows 应用程序的重要组成部分。

在这本书中，Chris Sells 对 WinForms 提供了全面而深入的讲解。前半部分讨论了窗体、对话框、GDI+以及打印等相对常见的技术，后半部分则专注于高级主题，内容涵盖设计期集成、资源、应用程序和设置、多线程用户界面以及 Web 部署等。其中第 12、13 章专门讲述了 ADO.NET 和 WinForms 的结合使用。鉴于需要访问数据库的企业应用开发是如此普遍，这两章的重要性不言而喻。另外，附录部分丝毫不比正文内容逊色，附录 A“从 MFC 转移到 WinForms”和附录 B“委托和事件”尤其值得一读。

开发实际 WinForms 应用的程序员几乎总是工作于 Visual C# .NET 这样的集成开发环境中，而不是完全手工编写代码。作为一名项目经验丰富的开发者，Sells 很清楚这一点。本书专注于使用 Visual C# .NET 开发 WinForms 应用程序，具有很强的实践性，阅读之前，建议您首先下载本书中代码的完整版本（www.sellsbrothers.com）并安装好 Visual C# .NET 开发环境。

一本技术书籍的价值往往并非限于初次阅读带来的愉悦，更重要的是它能否成为日后编程过程中的左右手。除了是很好的自学读本外，本书还是一本非常好的参考手册，倒不是因为它面面俱到——那是 MSDN 的责任，而是因为它讲解了许多别处很难见到但实际项目开发中又非常需要的高级技巧。如果您希望成为一名 WinForms 程序设计高手，请不要错过这本书。

Sells 属于那种善于将复杂的技术以简单的方式表达出来（而不是相反）的技术作家。 他的文风简练、清晰、流畅。当然，除了高超的写作技巧外，Sells 拥有的深厚的 Windows 平台技术背景则更是让他得以挥洒自如。这一点还体现在他的网站上。那儿除了维护有与本书直接相关的内容外，还有许多您会感兴趣的.NET 文章、代码、工具和评论。如果您对基于文档的应用程序开发特别感兴趣，请留意 Sells 牵头开发的开放源码项目 Genghis。

本书由我和蒋贤哲先生合译，贤哲完成了大部分章节的初译工作，在定稿期间亦有很好的交流。我在翻译过程中纠正了英文原书存在的大大小小数十个错误，并和 Sells 进行了必要的沟通。不过，限于种种主客观因素，在纠正错误的同时亦可能会引入新的错误，欢迎各位读者批评指正。

感谢编辑陈冀康先生为本书付出的所有台前幕后工作。感谢朱艳和荣珅，生活因为你们而精彩。

祝各位阅读快乐！

荣　耀
2004 年 7 月
南京师范大学
www.royaloo.com

序　言

自从 1983 年 Windows 1.0 被引入以来，Windows 应用程序开发已经发生了重大变化。Windows 程序员编写软件的方式以及所编写的软件的架构都已经发生了显著的变化。这个持续不断的演化进程的最近一步涉及到微软.NET 框架。这一新平台既影响了开发者使用的工具也影响了他们对“应用程序”的定义。

.NET 框架、与之适应的语言以及支持它们的工具，使得开发者能够拿一点点性能和控制权与开发生产力、代码安全性以及整个应用程序的全面强健性的改善作“交易”。对许多开发者而言这个交易极其令人兴奋，因为他们可以在无损于软件质量的前提下更快地完成工作。实际上，一些开发者发现利用 C#或 VB.NET 这样的语言可以在很大程度上提高代码的质量，因为语言自身天生就是对较早提供的语言的改进。

紧跟这些新的托管工具集之后的是一个流行于软件开发社群中的普遍误解：.NET 框架被设计成只能用于编写“Web 应用程序”，即应用程序位于一台中央 Web 服务器上并通过 Internet Explorer 这样的 Web 浏览器来展示其用户界面。对于某些解决方案来说，基于 Web 的应用程序非常具有吸引力。因为可能世界上每一台机器都已经安装某种 Web 浏览器了，所以我们无需分发 Web 应用程序。只要用户知道如何访问应用程序所在的服务器并与之交互就可以顺利地使用它们。对这种应用程序的更新是在它实际运行的服务器上进行的，并不需要考虑更新每一个具有本机软件拷贝的客户端的问题，因为根本就不存在本机拷贝！

然而，并非所有使用.NET 框架编写的应用程序都必须是 Web 应用程序。.NET 框架提供了一套被设计为实现“智能客户”应用程序的类（统称为“Windows Forms”）。智能客户是“提供直接使用 Windows 用户界面特性的本机用户界面而非使用 HTML 作为表示层”的应用程序。实际上，使用 Windows Forms 编写具有丰富的用户界面和离线工作能力并且仍然能够让开发者从.NET 框架中受益的独立客户应用程序，随处可见。

智能客户应用程序具有一些超过面向 Web 的应用程序的优点。在离线状态下可以完成某些事情是它的一个重要特性，并且能够保持如此直到人们希望完成工作的任何地方可以获得高速宽带连接——从飞机到起居室皆然。因为代码在本机执行，所以在智能客户应用程序对用户做出响应之前无需通过网络到服务器来回往返。不需要来回往返还使得智能客户应用程序免受网络响应时间的影响。

因为和它所运行的机器的亲密无间的关系，智能客户应用程序通常能够提供丰富得多的用户

体验。自定义绘制、有趣的字体设置以及方便的控件都是一些看得见的特性，它们有助于将智能客户应用程序同基于 Web 的应用程序区分开。智能客户应用程序还能够利用客户机器上的所有可用资源，并且很容易存取本机磁盘和内存，天性使然。

可以从这些力量中受益的问题是采用智能客户解决方案的优秀候选者。这些解决方案的机遇一直都存在。最近十年来，MFC 使得 C++程序员编写具有丰富用户界面的客户端应用程序成为可能。MFC 的目标是提供一个面向对象的包装层，它为 C++程序员带来了很大的价值。有时候，此价值只在于使得特定 API 在 C++中访问更加便利，但 MFC 的主体在于提供一个框架，通过集成智能客户应用程序中最常见的特性从而使得此类应用程序易于编写。

Windows Forms 开发人员比 MFC 开发人员享受一套更加完备的针对系统 API 的轻量级包装。和 MFC 开发人员不同的是，范围广泛的.NET 框架类库生来使得“托管程序员”在直接调用系统原生 API 时面临一些挑战。

在这本书里，Chris Sells 讨论了 Windows Forms 类以及它们的支撑基础设施是如何用于编写强健且丰富的智能客户应用程序的。如果你是一名经验丰富的 Windows 程序员，以前使用 MFC 或直接利用 Win32 API 来编写此类应用程序，你将会发现 Chris 直截了当的陈述非常适合将你的知识转移到托管类。如果你的 Windows 开发经验寥寥无几，你将发现这本书对于运用 UI 编程中的核心概念的描述是不可或缺的。

一门新语言以及一个用于编写智能客户应用程序的新框架为软件工程师提供了一套新的折衷方案。软件工程师除了选择一个为其用户带来可接受的折衷方案外还选择什么？工具。开发人员拥有的工具越多、将其利用得越好就能够解决越多的问题（最大的好处在于，一套用于构建客户应用程序的新工具将会为 Chris 带来新的注目焦点，从而停止向我发泄对 MFC 的抱怨）。请阅读这本书并将 Windows Forms 加入你的工具箱。

Mike Blaszczak
MFC 主创人员
微软公司
mikeblas@msn.com

前　言

作为一个在 Windows 开发社群中有那么一些名气的人物，时常有人问我.NET 是否会腾飞，我总是报以同样的回答：这不是“是否”的问题，而是“何时”的问题。

微软.NET 框架好处如此之多，以至于连我这样的鬓发斑白的 C++/Win32 老人也抵制不了托管开发环境之呼声的诱惑。经济的暂时不景气导致人们不愿再尝试任何新东西，然而.NET 却可以显著地缩短产品的上市时间和成本并可以提高代码质量，这真具有讽刺意味。业已采用了.NET 的组织意识到自己将会拥有长久而快乐的生活，这反过来又进一步推动了微软关于 Windows 平台（包括服务端和客户端）的未来计划。

.NET 中首要的服务端技术非 ASP.NET 莫属，它提供了构建 Web 站点和 Web services 所需的基础设施。ASP.NET 通过瞄准中间一代 Web 浏览器提供的特性基线，为开发者提供一种向任何人部署 Web 站点的途径。为了提供最高级别的功能，ASP.NET 在服务端做了大部分工作，使得客户端 HTML 变成了一个“触发对服务端新数据网页请求”的瘦包装器。服务端几乎包办了一切，包括从数据操纵到呈现菜单和工具栏这样的简单的东西。总而言之，这个模型提供了跨越操作系统和浏览器的最大的可用性。

另一方面，如果你的目标客户是 Windows 用户，基于 HTML 的体验将用户局限于“最小公分母”的方式就变得不再必要。实际上，为了力图提供更丰富的客户端体验，许多组织都很清楚其目标 Windows 用户需要特定版本的微软 Internet Explorer（IE）Web 浏览器。一旦如此，IE 将变得不再像浏览器，更像是基于 HTML 的应用程序运行时（runtime）。对于这个用途而言，HTML 对象模型相当原始，完成很简单的事情（比如跟踪用户会话状态）通常都需要你付出大量的工作。如果你打算以 Windows 用户为目标，.NET 框架为你提供了一套更丰富的对象，用于构建交互式用户界面。

这为我带来了本书的主题：Windows Forms（WinForms）。WinForms 是.NET 在客户端的脸面，它提供了基于窗体的开发环境，意欲提供最佳 UI 对象模型。此外，它还提供了一个迄今为止没有哪种基于 Windows 的开发框架提供过的特性：基于 HTML 的 Web 应用程序的部署特性。通过将 Windows 应用程序的丰富性和 Web 应用程序的部署特性联合起来，WinForms 向 Windows 开发者展示了一个全新的世界。这个世界使我感到更加愉快，从而乐于放弃非托管代码所招致的混乱。

谁应该阅读这本书

在写作本书过程中，我的脑海中浮现两类目标观众。我希望同时为已经在.NET 环境中编程的程序员和尚未在.NET 环境中编程的程序员提供真实世界的 WinForms 知识。为此，必要时我会简介核心.NET 主题。然而，.NET 框架的领地实在是太大了，本书绝不会佯称已经涵盖了一切。相反，只要我能够想到对读者有用的更多的信息，我都会引用另外提供了完整细节的作品。特别是，我反复提及了 *Essential .NET*（由 Don Box 和 Chris Sells 合著），从而使其成为本书很好的搭档。在同样的范畴，我还推荐 *Pragmatic ADO.NET*（Shawn Wildermuth 著）、*Advanced .NET Remoting*（Ingo Rammer 著）、*.NET Web Services*（Keith Ballinger 著）以及 *Applied Microsoft .NET Framework Programming*（Jeffrey Richter 著）（如欲了解关于这些书籍更多的细节，请查阅“参考文献”）。

有两个核心.NET 主题对于 WinForms 程序员来说特别重要，我在附录 B“委托和事件”和附录 C“序列化基础”中更详细地描述了它们。如果你是一名.NET 新手，“委托和事件”的内容尤为重要，尽管在你获得特定于 WinForms 的框架参考（从第 1 章“Hello, Windows Forms”开始大约占全书 1/3 内容）之前我并不推荐你深入钻研这个主题。

还有一点需要说明一下。许多年前，我写了我的第一本“5 天培训课程”，主题是 Windows 95，并包括若干课时关于新控件的讨论：它们看起来如何，它们的属性、方法和事件是什么，如何对它们进行编程等。这些课时看起来占用了我和学生们不少时间。实际上，只有当你使用该控件时，特定控件的细节才有意思。这些特定控件的文档和“智能感知”技术做了非凡的工作，它们为你提供了所需的信息。鉴于此，本书没有彻底涵盖任何一个标准控件，相反，每一个控件都处于当前讨论主题的有趣的上下文之中。比如在第 13 章“数据绑定和数据网格”中，DataGrid 控件得到了适当的讨论。还有，第 8 章“控件”和第 9 章“设计期集成”中介绍了 WinForms 提供的广泛的控件，包括.NET 中称为非可视控件的组件。

最后，为了给你一个关于全部控件和组件的形象的认识，并向你逐一介绍它们的主要功能，附录 D“标准 WinForms 组件和控件”提供了一个关于标准控件和组件的清单。我不想因企图比.NET 框架 SDK 和 Visual Studio .NET 携带的参考文档更加详尽而浪费你的时间，相反，这本书专注于别处未详尽讨论的真实世界的场景。

约定

如果你决定尝试这本书，我首先要感谢你的信任并表达我希望达到的愿望。为了帮助你阅读接下来的文字，我希望你熟悉我的作品中使用的一些约定。

首先也是最重要的是，WinForms 最奇妙之处在于它的高度可视化，这也是我为什么大量地使用插图来阐明其特性的原因。其中一些插图确实需要做成彩插，这样有助于更好地说明问题。请检查这本书中间的彩页以便查看彩色插图。

同时，我认为代码和插图一样重要。代码采用等宽字体：

```
System.Console.WriteLine("Hello, WinForms.");
```

控制台应用程序的启动命令也显示为等宽字体：

```
C:\> csc.exe hello.cs
```

如果一部分代码片段或启动命令行特别重要，我会将其标为粗体并且通常提供一个注释：

```
// 注意对.NET System 命名空间的使用
System.Console.WriteLine("Hello, WinForms.");
```

当我希望将你的注意力指引向更完整的代码时，我会将多余的代码用省略号代替：

```
class MyForm : System.Windows.Forms.Form {
  ... // 字段
  private void MyForm_Load(object sender, System.ComponentModel.EventArgs e) {
    MessageBox.Show("Hello from MyForm");
  }
}
```

进一步而言，为了使打印出来的代码可读性更好，我通常会删除掉命名空间和保护性关键字——假如它们没有提供额外信息的话：

```
// 缩短了的“System.Windows.Forms.Form”基类
class MyForm : Form {
  ... // 字段
  // 移去“private”修饰符和“System.ComponentModel”命名空间
  void MyForm_Load(object sender, EventArgs e) {
    MessageBox.Show("Hello from MyForm");
  }
}
```

而当展示.NET 特性时，我会使用其全名：

```
[SerializableAttribute]
class MyCustomType {...}
```

有些语言，比如 C#，允许为了方便而将“Attribute”后缀省略掉，但这也使得难于在联机文档中查找特性类的细节。

有时为了清晰，我在打印的代码中略去了错误检查，但我会尽量将其保留于配书示例代码中。

在正文中，如果我打算定义一个新的术语，我常常将一个单词或短语设置为黑体。“**霸权**是指压倒性的势力或权力（**hegemony** is a preponderant influence or authority）”就是这类术语及其定义的一个例子，这也是一个行之有效的惯用做法。

最后，我常常提到键盘快捷键因为我发现它们用起来很方便。我提到的是默认 Visual Studio 开发者键盘绑定设置，如果你使用的不是这种键盘绑定，请将键盘快捷键映射到你自己的设置。

联系

这本书的最新信息，包括源代码和勘误表，均在 http://www.sellsbrothers.com/上给出。该站点还为你提供了向我发送本书反馈意见的途径，赞美的和批评的，我都欢迎。

致谢

在此，我要向我的家人致以谢意。尽管我在家中工作，但在本书完成的最后阶段我通常必须花费大量的额外时间。在临近最后期限之际，我的爱妻 Melissa 给予我极大的理解，并提供按时完成本书所需的宽松环境。我习惯将办公室的门开着，因为我的儿子 John 和 Tom 经常会进来和我讨论他们的事情。尽管他们分别只有 9 岁和 7 岁，但当我需要集中精力“再工作 5 分钟”时，他们完全能够理解我（当我对他们两个的轻易允诺没有兑现时，痛苦的总是我。以后我再说给你们听）。在我的亲朋好友中，我要感谢我的妹妹 Beth 和妹夫 Joel Howie，他们给予我更充分的空间。在他们的外甥听我说“再给我 5 分钟”已经感到厌烦时，他们会和外甥们交流。当然，我要感谢我的父母，是他们使我成为一个热情的读者并传给我直到最近我才意识到的写作技巧。

尽管我的家人给予我写作本书的宽松环境，但倘若缺乏一些极有帮助的人的努力本书也不会有现在的模样。Michael Weinhardt 是我在 *MSDN Magazine* 上发表的 *Building Windows Forms Controls and Components with Rich Design-Time Features*（1，2）系列文章的合作者，这两篇文章是第 9 章“设计期集成”的前身。另外，他还贡献了本书中最好的一些插图，其中包括第 10 章“资源”中的资源解析插图。ADO 专家 Shawn Wildermuth 不仅在关于数据库的那两章给予我极大的帮助，而且对第 8 章“控件”和附录 D“标准 WinForms 组件和控件”也提供了莫大的帮助。Mike，Shawn，如果没有你们二位，这本书不会有今天这么好的面貌。

另外，Mike Woodring 在本书的线程部分给予我大量的反馈。Keith Brown 一直是我“安全”翱翔的双翼下的劲风，在.NET 安全如何工作（以及为何）方面他给予我大量的帮助。当本书作为一个为期 5 天培训的教材时，Fritz Onion 和 Ian Griffiths 给了我大量的反馈。我非常幸运，当 Allen Cooper 重新对编程感兴趣时就被我碰上了。他阅读了每一章。他给予我难以置信的真正留下了烙印的反馈（尤其是，第 10 章“资源”）。

另外，我还要感谢那些他们自己并不知道对我和本书起到帮助作用的人们，包括在 dotnetjunkies.com 上发表 *Help Authoring in .NET* 的 Ljubomir Spasovski，在 dotnet247.com 上发表 DrawRoundRect 方法的 Bob Powell，在 flounder.com 上发表关于多实例文章的 Joseph Newcomer。在这方面，我还要感谢 Jeff Prosise 的 *Programming Microsoft .NET*，该书中的 WinForms 一章鼓舞我写这本书，因为我无法接受将 WinForms 概述于一章之中的主意。我也要感谢 Don Box 允许我一直参考他的 *Essential .NET* 第一卷，尽管我没有为那本书写一个字，但与它的亲密接触教会我.NET 是如何工作的。这种教诲一直延伸到本书。

这本书的审稿人应该受到特别的感谢，他们是 Bill Woodruff、Brian Graf、Christophe Nasarre、Cristof Falk、Doug Reilly、Fumiaki Yoshimatsu、Johan Ericsson、Mark Rideout、Martin Heller、Pierre Nallet、Richard Blewett、Scott Densmore、Serge Shimanovsky、Suresh Jambu、Tim Tabor 以及 Zeeshan Amjad。审稿人给出的意见对于本书影响之大，以至于我无法表达诸位对我而言是何等重要。我尤其要感谢 Bill、Christophe、Mark 和 Martin 所给予的特别详尽的建议。Christophe（再次感谢）则使我在吸取审稿人的意见之后确保一切东西仍有意义，我无法说出你为我挣脱了多少困窘，Christophe，对你惟有感谢！

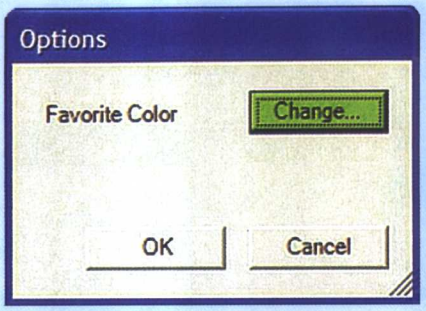
当然，我还必须感谢微软那些发明了这项技术并帮助社群（尤其是我自己）理解它的专家们，包括 Mark Boulter、Chris Anderson 和 Jamie Cool。

我要感谢 *MSDN Magazine*、*MSDN Online* 和 *Windows Developer* 杂志允许我重用最先发表于它们那儿的文章（如“参考文献”所列）。我还要感谢我的读者，他们对最初的文章所给予的反馈有助于形成这些内容的最终版本，并激励我比当初更加深入地钻研下去。

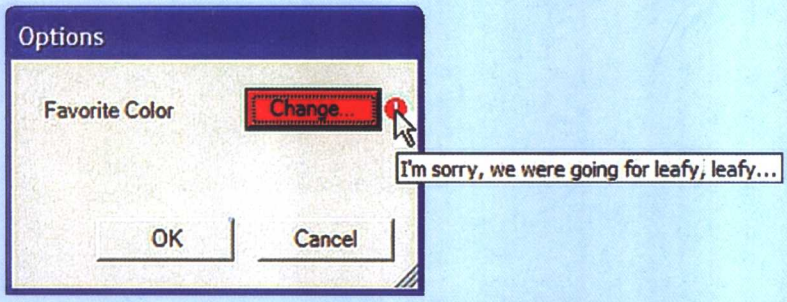
最后（但并非最不重要），我要感谢 Addison-Wesley 的各位名人。在这个节奏日趋加快的时代，他们设法为我提供了一个宽松的环境，从而使我可以写出我认为最好的东西。我要特别感谢文字编辑 Betsy Hardinger，她是一位受挫的小说作者和志趣相投的益友。除了将我的文章更加英语化以外，她还捕捉到了那些为纯粹的开发人员所遗漏的技术上的不一致。谢谢你，Betsy！

以上人士，以及肯定还有被我遗漏的其他很多人，均助我使本书一切东西面貌良好，书中残存的任何错误，责任全在我自己。

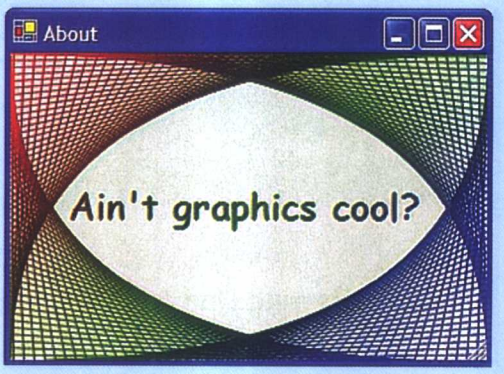
Chris Sells
2003 年 4 月
www.sellsbrothers.com



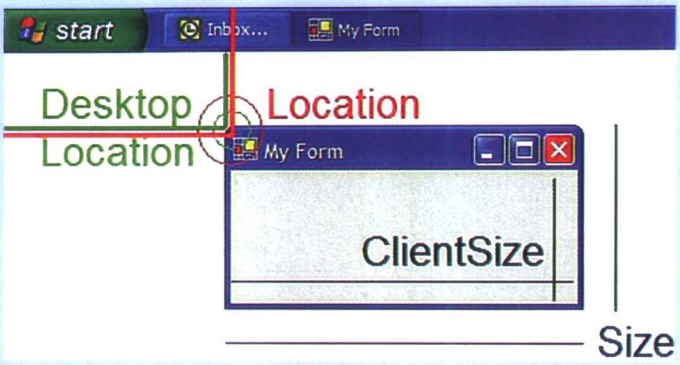
彩图 1（图 1.15）：一个对话框



彩图 2（图 1.16）：使用 ErrorProvider 提供一个错误信息

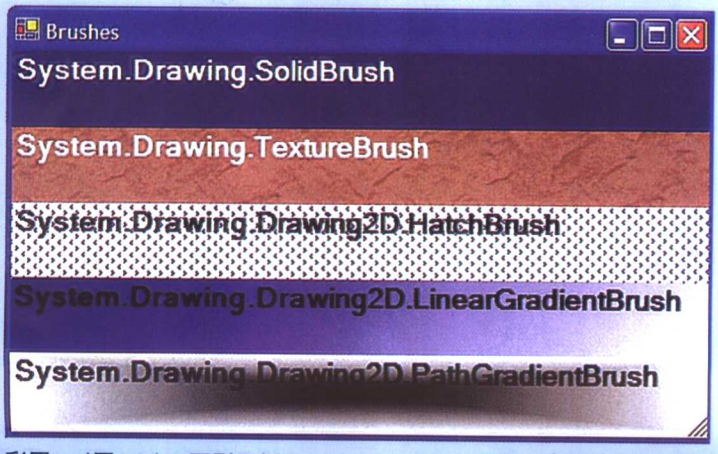


彩图 3（图 1.17）：自定义绘图

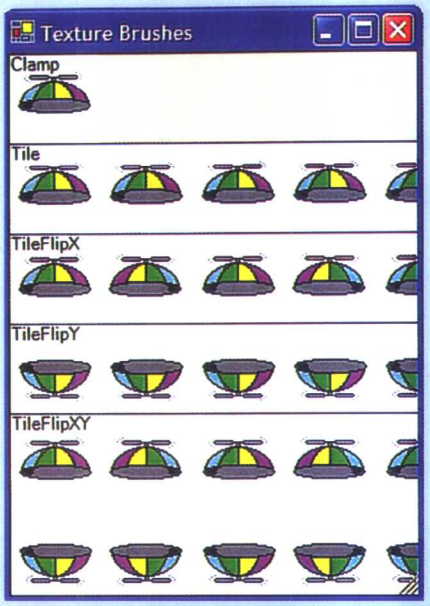


彩图 4（图 2.3）：窗体的 DesktopLocation、Location、ClinentSize 和 Size 属性

彩图 5（图 2.4）：不透明度

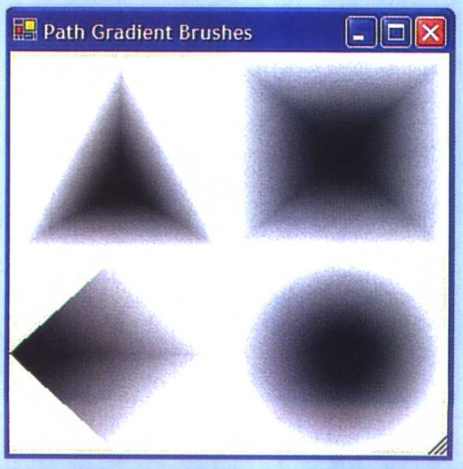


彩图 6（图 4.3）：画刷示例



彩图 7（图 4.4）：不同的 TextureBrush WrapMode 值的效果

彩图 8（图 4.6）：常规、三角和贝尔线性渐变画刷



彩图 9（图 4.7）：PathGradientBrush 类的 4 个示例用法