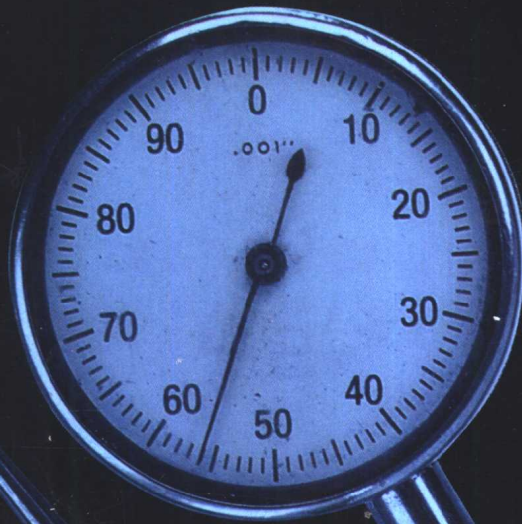


核心参考

Microsoft
程序设计系列



Microsoft®

C# Windows®

程序设计

(上册)



[美] Charles Petzold 著

北京大学出版社

Microsoft C# Windows 程序设计

(上册)

[美] Charles Petzold 著

天宏工作室 译

北京大学出版社

·北京·

Microsoft C# Windows 程序设计

(下 册)

[美] Charles Petzold 著

天宏工作室 译

北京 大学 出版 社

·北 京·

著作权合同登记 图字：01-2001-3136 号

Programming Microsoft Windows with C#

Charles Petzold

本书版权为 Charles Petzold 所有,2002。(Copyright © 2002 by Charles Petzold. All rights reserved.)

本书中文版由美国 Microsoft 出版社授权北京大学出版社独家出版,2002。

本书封面贴有北京大学出版社的激光防伪标签,无标签者不得销售。

版权所有,翻印必究。

图书在版编目(CIP)数据

Microsoft C# Windows 程序设计/(美)佩特佐德(Petzold, C.)著;天宏工作室译. —北京:北京大学出版社,2002.9

ISBN 7-301-05140-9

I . M… II . ①佩… ②天… III . C 语言-程序设计 IV . TP312

中国版本图书馆 CIP 数据核字(2002)第 027469 号

书 名: Microsoft C# Windows 程序设计

著作责任者:[美] Charles Petzold 著 天宏工作室 译

责任编辑:邱淑清 范彦 郭晓华 徐涛

标准书号:ISBN 7-301-05140-9/TP·0572

出版者:北京大学出版社

地 址:北京市海淀区中关村北京大学校内 100871

网 址:<http://cbs.pku.edu.cn>

电 话:出版部 62754962 发行部 62754140 编辑室 62757065

电子信箱:zpup@pup.pku.edu.cn

排 印 者:北京大学印刷厂

发 行 者:北京大学出版社

经 销 者:新华书店

787 毫米×1092 毫米 16 开本 77 印张 1600 千字

2002 年 9 月第 1 版 2002 年 9 月第 1 次印刷

定 价:160.00 元(上下两册,含光盘)

北京大学出版社
Peking University Press

**权威的 Windows 专家 Charles Petzold
告诉您**

如何使用 C# 进行 Windows API 编程。

**本书是全球最畅销的
C# 编程图书之一 !!!**

简介

本书介绍了如何编写在 Microsoft Windows 环境下运行的程序。存在许多方式编写这样的程序。在这本书中,我使用了新的面向对象的程序设计语言 C# (发音为“C sharp”)和名为“Windows Forms”的新式类库。Windows Forms 类库是 Microsoft .NET (“dot.net”) Framework 的一部分,后者在 2000 年夏季出现,大约一年半之后被引入。

Microsoft .NET Framework 是一个广泛的类集合,它为程序员提供了大量编写 Internet、Web 和 Windows 应用程序所需要的类。许多 .NET 的媒体报道都集中在 Web 程序设计上。本书讨论 .NET 的其他部分。您可以使用 Windows Forms 编写传统的独立 Windows 应用程序(现在有时被称为“客户”应用程序)或者分布式应用程序的前端。

Windows Forms 几乎提供了在编写正式的 Windows 应用程序时所需要的每一个组件。重要的遗漏是多媒体支持。甚至没有哪一个 Windows Forms 函数能够使计算机的扬声器发出蜂鸣声!我总想编写自己的多媒体类,但是总是假设下一个 Windows Forms 版本将会包括灵活、强大和使用简单的多媒体支持,并将自己限制在这个假设下(我希望是合理的)。

在 .NET Framework 中定义的类是语言中性的。Microsoft 已经发布了可以使用这些类的 C++ 和 Visual Basic 新版本,以及新的程序设计语言 C#。其他语言开发商正在使他们自己的语言适合于使用 .NET 类。这些新的编译器(可选的或默认的)在一个 .exe 文件中将源代码转换为一种中间语言。在运行时,这种中间语言被编译为适当的微处理器机器码。这样,.NET Framework 实际是与平台无关的。

我为本书选择使用 C# 是因为 C# 和 .NET 是——在某种真正意义上——为彼此而创建的。因为 .NET Framework 具有语言中性这个特征,所以您或许可以使用

本书学习如何与其他 .NET 语言一起编写 Windows Forms 应用程序。

Windows 程序设计概述

Microsoft 在 1985 年秋季发布了第一个 Windows 版本。从那时起,Windows 逐步更新和增强,特别引人注目的是在 Windows 从 16 位转向 32 位体系结构时出现的 Windows NT (1993)和 Windows 95 (1995)。

在第一次发布 Windows 时,实际上只有一种编写 Windows 应用程序的方法,那就是使用 C 程序设计语言访问 Windows 应用程序编程接口 (Application Programming Interface, API)。虽然您也可以使用 Microsoft Pascal 访问 Windows API,但是这种方法已经很少使用了。

这些年来,其他许多语言已经适合进行 Windows 程序设计了,包括 Visual Basic 和 C++。C++ 和 C# 都是面向对象的语言,它们支持 C 程序设计语言的大部分类型、操作符、表达式和语句。出于这个原因,有时将 C++ 和 C# (以及 Java)称为基于 C 的语言或者 C 系列语言。

随着 .NET 的引入,Microsoft 目前提供了三种使用基于 C 的语言编写 Windows 应用程序的方法,如下表所示。

如何使用基于 C 的语言编写 Windows 应用程序

引入年份	语言	接口
1985	C	Windows 应用程序编程接口 (Application Programming Interface, API)
1992	C++	Microsoft 基础类库 (Microsoft Foundation Class, MFC)
2001	C# 或 C++	Windows Forms (.NET Framework 的一部分)

告诉您应该使用何种语言或接口编写 Windows 应用程序不是本书的任务。那是您只能根据特定的程序设计任务和可用的资源才可以做出的决定。

为了了解有关 Windows API 的更多信息,许多人都发现我著的《Programming Windows》(第 5 版, Microsoft 出版社, 1998)(该书中文版《Windows 程序设计》已由北京大学出版社于 1999 年出版, 2002 年 9 月第 5 次印刷)非常有价值。

我从来不喜欢 MFC。自从它出现以来,我就认为它设计拙劣,并且勉强接近面向对象风格。因此,我从来不写有关 MFC 的任何书籍。但是这只是个人观点。其他许多程序员已经成功地使用了 MFC,并且 MFC 是目前最流行的 Windows 程序设计方法之一。了解 MFC 的一本好书是 Jeff Prosise 编写的《Programming Windows

with MFC》(第2版, Microsoft 出版社, 1999)。对于更高级的 Windows 程序员, 我还想推荐 Jeffrey Richter 编写的《Programming Applications for Microsoft Windows》(Microsoft 出版社, 1999)。

从我的观点来看, Windows Forms 比 MFC 设计得更好, 更接近我所想象的理想的 Windows 面向对象接口。在我编写这本书的过去 14 个月中, Windows Forms 变成了我首选的 Windows 程序设计方法。

从程序设计的角度来看, MFC 和 Windows Forms 接口都通过调用 Windows API 工作。在体系结构上, 可以说它们位于 API 的顶部。这些更高级的接口的目的是使 Windows 程序设计变得更简单。在 MFC 或 Windows Forms 中执行特定的任务与使用 API 时相比, 通常您可以使用更少的语句。

虽然像 MFC 或 Windows Forms 这样的高级接口通常会提高程序员的生产率, 但是任何利用一个接口的其他接口明显不如基础接口通用。使用 Windows API 可以完成许多使用 Windows Forms 类无法完成的事情。

幸运的是, 只要一点额外的工作, 就可以从一个 Windows Forms 程序中调用 Windows API。我只是在本书中偶尔遇到 .NET Framework 中缺少的东西很深奥时才需要利用这个功能。我的总体原则是注意 Windows Forms 提供的与 Windows 本身的内部工件的隔离。

用户要求

为了更有效地使用本书, 您需要能够编译和运行 C# 程序。要想编译这些程序, 您需要一个 C# 编译器。要想运行这些程序, 您需要 .NET 运行时, 它被称为“公共语言运行时”(common language runtime, CLR), 它是一个动态链接库的集合。

这两项都包括在 Microsoft Visual C# 中, 它是现代的集成开发环境。或者, 您可以购买更广泛和更昂贵的 Microsoft Visual Studio .NET, 它也允许您在 C# 以外的 C++ 和 Visual Basic 中进行程序设计。

如果您喜欢更复杂的方法, 则可以下载免费的 .NET Framework 软件开发工具包 (Software Development Kit, SDK)。这个下载包括一个命令行 C# 编译器和 .NET 运行时。首先进入到 <http://msdn.microsoft.com/downloads>。在左侧, 选定 Software Development Kit, 然后查找 .NET Framework。(记住, 与本书中提到的所有 Web 站点一样, 这个 Web 站点在任何时候都可能会改变、移动, 在某些情况下也可能完全消失。)

我编写这本书的假设是您至少知道如何在 C 中进行程序设计。熟悉 C++ 或 Java 是有帮助的,但不是必需的。因为 C# 是一种新型语言,所以本书的第一章简要介绍 C# 和面向对象程序设计的重要概念。在本书的其余部分,我经常在遇到各种 C# 概念时花些时间讨论它们。

但是,本书并不提供 C# 的详尽教程。如果想要在使用这种语言时拥有更多背景知识和技巧,则可以参考有关 C# 的其他书籍。随着这种语言变得更加流行,无疑可以获得其他许多书籍。Tom Archer 编写的《Inside C#》(Microsoft 出版社,2001)提供了关于编写 C# 代码及其内部技术内容的信息。John Sharp 和 Jon Jagger 编写的《Microsoft Visual C# Step by Step》(Microsoft 出版社,2001)采用了一种更加指导性的方法。

在本书中,我有时引用 Windows API。像我在前面所说的一样,您可以查阅我的著作《Windows 程序设计》,以了解关于 API 的更多信息。

系统要求

正如在前面小节中提到的一样,为了有效地使用本书,您需要能够编译和运行 C# 程序。系统要求如下:

- Microsoft .NET Framework SDK(至少)、Microsoft Visual C# 或 Microsoft Visual Studio .NET(更好);
- Microsoft Windows NT、Windows 2000 或 Windows XP。

要想在其他计算机上运行自己的 C# 程序,需要在那些计算机上安装 .NET 运行时(也被称为 .NET Framework“可再分发的软件包”)。该软件包是随 .NET Framework SDK、Visual C# 和 Visual Studio .NET 一起附带的。您可以将这个可再分发的软件包安装在前面已经提到过的 Windows 版本以及 Windows 98 或 Windows Millennium Edition (Me)上。

如果想要将配套 CD 上的示例文件安装到自己的硬盘驱动器上,则需要大约 2.1 MB 的额外硬盘空间。(在完全编译之后,这些示例仅仅占用 20 MB 以上的空间。)

本书的结构

在首次发布 Windows 1.0 时,完整的 API 是在三个名为 KERNEL、USER 和 GDI 的动

态链接库中实现的。虽然这些与 Windows 有关的 DLL 已经变得更加庞大,但是将 Windows 函数调用(或架构类)分为这三个种类仍然有用:核心调用是在操作系统的体系结构内部实现的调用,它们通常与任务分派、内存管理和文件 I/O 有关。术语 user 是指用户界面。这些是创建窗口、使用菜单和对话框、显示按钮和滚动条这种控件的函数。GDI 是图形设备接口(Graphics Device Interface, GDI),它是负责在屏幕和打印机上显示图形输出(包括文本)的 Windows 部分。

本书以 4 个介绍性章节开始。从第五章开始(介绍如何绘制直线和曲线),直到第二十四章(关于 Windows 剪贴板),这些章都在图形主题(奇数章)和用户界面主题(偶数章)之间交替变换。

像本书这样的书籍一般不会在非 Windows 主题(如文件 I/O、浮点运算和字符串操作)上花费太多时间。不过,因为 .NET Framework 和 C# 非常新,所以我发现自己希望有一个贯穿这些类的连贯指南。因此我自己写下这样的指南。这些作为关于文件、数学运算和字符串的三个附录被包括在本书中。您可以在阅读第一章之后的任何时候查阅这些附录。

我尽力排列了各章——以及各章的主题——以便每个主题都建立在成功的主题上,使“向前引用”的数量最少。我写这本书是为了让您可以直接阅读,就像阅读《The Stand》或《The Decline and Fall of the Roman Empire》的未删节版本一样。

当然,如果像本书一样长的书籍可以作为参考和叙述文,那么这是一件好事。出于这个原因,在 Windows Forms 程序设计中的许多重要方法、属性和枚举都列在了讨论它们的各章节中的表中。不过,即使一本这样篇幅的书也不能够希望它覆盖 Windows Forms 中的每一项内容。它不是正式类文档的替代物。

Windows Forms 程序要求很少的系统开销,因此本书以完整的程序形式包括了大量示例代码。您可以随意地从这些程序中将代码片段剪切和粘贴到自己的程序中。(这正是提供这些程序的目的。)但是不要原封不动地分发书中的代码或程序。这是本书的目的。

C# 编译器具有一个非常强大的特性,它允许您编写带有 XML 标记的注释。不过,我没有选择使用这个特性。本书中的程序通常只有很少的注释,因为代码是在围绕程序的文本中描述的。

正如您可能知道的一样,Visual C# 允许您交互地设计应用程序的外观。您将各种控件(按钮和滚动条等)放在窗口的表面,由 Visual C# 生成代码。虽然这样的技术对于快速设计对话框和前端面板类型的应用程序非常有用,但是我在本书中忽略了 Visual C# 的这个特性。

在本书中,我们打算让 Visual C# 为我们生成代码。我们将学习如何编写自己的代码。

CD-ROM

配套 CD-ROM 包含了本书中的所有示例程序。您可以将解决方案文件(.sln)或项目文件(.csproj)载入 Visual C# 并重新编译这些程序。

坦白地说,我从来没有太多地使用本书的 CD-ROM。在学习一种新的语言时,我喜欢自己键入源代码——即使它最初是其他人的源代码。我发现我使用这种方式学习语言的速度更快。但是这只是我个人的学习习惯。

如果 CD-ROM 丢失或损坏,则不要向我发送电子邮件请求更换。相反,我不会给您邮寄一张新的 CD-ROM。Microsoft 出版社是本书和 CD-ROM 的独家发行商。要想获得替换的 CD-ROM 或关于本书的其他支持信息,请与 Microsoft 出版社联系。

支持

每一项努力都是为了确保本书和配套 CD-ROM 的内容准确。Microsoft 出版社通过 Web 站点提供书籍的更正信息,其地址如下:

<http://www.microsoft.com/mspress/support>

要想直接进入 Microsoft 出版社的知识库(Knowledge Base),并输入一个与您的问题有关的查询,可以进入:

<http://www.microsoft.com/mspress/support/search.asp>

如果有关于本书或配套 CD-ROM 的评论、问题或想法,请使用下面任何一种方法将其发送到 Microsoft 出版社:

邮政地址:

Microsoft Press

Attn: Programming Microsoft Windows with C# Editor

One Microsoft Way

Redmond, WA 98052-6399

电子邮件:

MSPINPUT@MICROSOFT.COM

请注意,Microsoft 并不通过以上邮政地址提供产品支持。有关 C#、Visual Studio 或 .NET Framework 的支持信息,请访问 Microsoft 产品支持 Web 站点,其地址为:
<http://support.microsoft.com>

特别感谢

写作通常是一项非常枯燥的工作,但幸运的是,总有几个人使工作变得更简单。

我想要感谢我的代理人——Moore Literary Agency 的 Claudette Moore,他使这个项目得以进行并处理所有各种法律材料。

与平常一样,Microsoft 出版社的朋友们是绝对令人愉快的合作伙伴,他们再次帮助我避免遇到麻烦。如果没有我的项目编辑 Sally Stickney 和技术编辑 Jean Ross,那么本书中将出现大量的混乱信息和错误代码。虽然编辑们可能有时看上去像超人,但是遗憾的是他们并不是。留在本书中的任何错误或无法理解的句子都是我的错,而不是其他任何人的错。

我不会忘记感谢 Johannes Brahms 在我工作时提供的音乐伴奏,以及 Anthony Trollope 在夜晚提供的逃避现实的文学作品。

在星期日、星期二和星期四的朋友聚会上,朋友们以有时明显、有时微妙,但通常是非常宝贵的方式继续帮助和支持我。

我尤其想要感谢我的未婚妻 Deirdre,她为我提供了一个完全不同的(非.NET)框架,我们在其中生活、工作和恋爱。

Charles Petzold
2001 年 11 月于纽约

上册目录

简介	1
第一章 控制台本身	1
1.1 C# 版本	2
1.2 程序分析	4
1.3 C# 名称空间	6
1.4 控制台 I/O	8
1.5 C# 数据类型	10
1.6 表达式和操作符	14
1.7 条件和循环	17
1.8 对象的跳跃	21
1.9 使用 C# 的关键特性进行程序设计	25
1.10 静态方法	27
1.11 异常处理	28
1.12 抛出异常	31
1.13 获取和设置属性	32
1.14 构造函数	36
1.15 实例和继承	39
1.16 更宏伟的蓝图	43
1.17 命名约定	45
1.18 几句题外话	46
第二章 欢迎使用 Windows Forms	47
2.1 消息框	49

2.2	窗体	55
2.3	显示窗体	57
2.4	它是一个应用程序,我们想要运行它	59
2.5	主题的多种形式	61
2.6	窗体的属性	62
2.7	事件驱动的输出	64
2.8	处理 <i>Paint</i> 事件	66
2.9	显示文本	69
2.9.1	字体	70
2.9.2	画刷	71
2.9.3	坐标点	72
2.10	<i>Paint</i> 事件很特殊!	74
2.11	多个窗体,多个处理程序	75
2.12	继承窗体	77
2.13	<i>OnPaint</i> 方法	79
2.14	<i>Main</i> 属于这里吗?	81
2.15	事件和 <i>On</i> 方法	82
第三章	基本结构	87
3.1	类和结构	87
3.2	二维坐标点	89
3.3	<i>Point</i> 数组	91
3.4	<i>Size</i> 结构	92
3.5	浮点版本	93
3.6	矩形是一个 <i>Point</i> 和一个 <i>Size</i>	96
3.7	<i>Rectangle</i> 属性和方法	97
3.8	大小合适的窗体	100
3.9	窗体和客户区	101
3.10	坐标变换	105
3.11	颜色结构	106
3.12	141 种颜色名称	107
3.13	画笔和画刷	108
3.14	系统颜色	110

3.15	已知的颜色	113
3.16	使用什么?	114
3.17	感受重画	115
3.18	使 Hello World 居中	117
3.19	测量字符串	121
3.20	矩形中的文本	123
第四章	文本输出中的练习	127
4.1	系统信息	127
4.2	为文本之间添加间距	128
4.3	属性值	129
4.4	格式化为列	131
4.5	每样东西都是一个对象	133
4.6	列出系统信息	137
4.7	Windows 窗体和滚动条	140
4.8	滚动面板控件	141
4.9	<i>ScrollableControl</i> 的继承关系	145
4.10	没有控件的滚动	146
4.11	实际数值	148
4.12	保持最新	149
4.13	不要成为一头猪	152
4.14	展望未来	154
第五章	直线、曲线和区域填充	159
5.1	如何获得一个图形对象	160
5.2	画笔简介	161
5.3	直线	162
5.4	打印	164
5.5	属性和状态	170
5.6	图形保真	171
5.7	多重连接的直线	174
5.8	曲线和参数方程	178
5.9	普遍存在的矩形	181

5.10	一般化的多边形	183
5.11	更简单的椭圆	184
5.12	弧线和饼图	185
5.13	填充矩形、椭圆和饼图	192
5.14	偏离 1 个像素	194
5.15	多边形和填充模式	195
第六章	接近键盘	199
6.1	忽略键盘	199
6.2	谁获得了焦点?	200
6.3	键和字符	201
6.4	键的按下和弹起	202
6.5	Keys 枚举	204
6.6	测试修饰键	213
6.7	实际检验	214
6.8	SysInfo 的键盘接口	215
6.9	字符的击键	217
6.10	控制字符	217
6.11	查看键码	218
6.12	调用 Win32 API	223
6.13	处理来自外语键盘的输入	225
6.14	输入焦点	228
6.15	丢失的插入符号	229
6.16	回显键字符	234
6.17	从右到左的问题	237
第七章	页面和变换	241
7.1	文本的设备无关性	241
7.2	实际数值是多少?	242
7.3	每英寸的点数	245
7.4	打印机的情况怎样?	246
7.5	手工转换	247
7.6	页单位和页缩放	249