



PH
PTR

C# Windows 编程

- ▶ 功能强大，快速开发应用程序
- ▶ 专家手把手教你 C#
- ▶ 完整的代码实例让你学习更轻松
- ▶ 两个案例分析，展示完整的应用程序

[美] Chris H. Pappas 著
William H. Murray

李海龙 英宇 译

人民邮电出版社

POSTS & TELECOMMUNICATIONS PRESS

C# Windows 编程

[美] Chris H. Pappas William H. Murray 著

李海龙 英宇 译

人 民 邮 电 出 版 社

图书在版编目 (CIP) 数据

C# Windows 编程/(美) 帕帕斯 (Pappas, C. H.), (美) 默里 (Murray, W. H.) 著;
李海龙, 英字译. —北京: 人民邮电出版社, 2002.10

ISBN 7-115-10581-2

I. C... II. ①帕... ②默... ③李... ④英... III. C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字 (2002) 第 067360 号

版 权 声 明

Simplified Chinese Edition Copyright © 2002 by PEARSON EDUCATION NORTH ASIA LIMITED and POSTS & TELECOMMUNICATIONS PRESS.

C# for Windows Programming

By Chris H. Pappas and William H. Murray

Copyright © 2002

All Rights Reserved.

Published by arrangement with the original publisher, Pearson Education, Inc.,

Publishing as Prentice Hall PTR.

This edition is authorized for sale only in People's Republic of China (excluding the Special Administrative Region of Hong Kong and Macao).

本书封面贴有 Pearson Education (培生教育出版集团) 激光防伪标签, 无标签者不得销售。

C# Windows 编程

◆ 著 [美] Chris H. Pappas William H. Murray

译 李海龙 英 宇

责任编辑 李 际

◆ 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

北京鸿佳印刷厂印刷

新华书店总店北京发行所经销

◆ 开本: 787×1092 1/16

印张: 28.75

字数: 696 千字

2002 年 10 月第 1 版

印数: 1-4 000 册

2002 年 10 月北京第 1 次印刷

著作权合同登记 图字: 01-2002-3744 号

ISBN 7-115-10581-2/TP·3050

定价: 50.00 元

本书如有印装质量问题, 请与本社联系 电话: (010) 67129223

内 容 提 要

C#是 Visual Studio .NET 环境中一种全新的面向对象的编程语言，本书作者以亲身体验的技术以及长期的 C#教学经验，带你跃入 Windows 应用编程的下一高潮。

本书首先简要介绍了 C#语言及其编程元素，然后针对 Windows 面向对象的编程特性，详细讲述了用户界面的设计、控件及其属性、事件及事件处理程序、输入输出、图形处理等等，最后两章是完整的财务和图表应用程序实例，即使你是编程新手，也能从完整的实例中体会到编程成功的喜悦感。通过本书的学习，你能学到丰富的 Windows 编程策略，快速地开发出功能强大的 Windows 应用程序。

本书适合于对 C/C++/VB 有一定了解的读者，尤其适合于那些已经具备了 Windows 编程经验的读者。

前 言

这本书是为那些具有良好的 C/C++ 语言知识的程序员编写的。如果你已经具有开发 Windows 应用程序的经验，那么你将领先于其他人一步。

本书旨在向你介绍 C# 语言的关键元素，并结合利用 C# 语言开发 Windows 应用程序的过程进行讲述。本书首先快速分析了 C# 语言的关键特征，然后将你带入 Windows 应用程序开发的世界。你将会学习到丰富的 Windows 编程策略，这些策略可以指导你进行 C# 编程，使得程序更容易编写。

本书采用实例的方式来教学。在书中我们力求涵盖大多数 Windows 应用程序的重要特征。每个例子都给出了所有代码，而不仅仅是部分代码。对于有经验的程序员来说这可能显得冗余和累赘，但是生活的经验告诉我们当需要它们的时候，完整的代码会帮助解决最困难的程序设计思想问题。

C# 语言看起来与传统的 C++ MFC 编程背道而驰，Windows 程序员对 C++ MFC 已经很熟悉了。虽然这种语言很像 C 语言，但是它的接口的确不是传统的 SDK。相反，它实际上与 Visual Basic 语言非常相似，因为在这里也使用了窗体、控件等等，你肯定会喜欢这个环境的。

本书也强调了 C# Windows 应用程序开发与传统的 Windows 应用程序开发的根本区别。程序员仍可以在 C# 中使用 C/C++ 元素，但是 C# 中使用了早期的拖放设计模式。这个修正可以大大改进 Windows 应用程序设计。

准备好了吗？那么加油吧——Windows 应用程序开发从来都不会太容易的！

目 录

第 1 章 C#与 Windows 简介	1
1.1 Visual Studio.NET 和 C#	1
1.2 创建 C#应用程序	1
1.3 第一个 C#控制台应用程序	2
1.4 C#编程元素	5
1.4.1 数组	6
1.4.2 属性 (attribute)、事件、索引器、属性以及版本转换	7
1.4.3 装箱、拆箱以及统一类型系统	9
1.4.4 类、结构和枚举	9
1.4.5 命名空间	11
1.4.6 预定义类型	12
1.4.7 语句	13
1.4.8 数值类型和引用类型	17
1.5 第一个 C# Windows 应用程序	17
1.5.1 其他程序细节	24
1.5.2 命名空间	24
1.5.3 窗体	25
1.5.4 设计者变量	26
1.5.5 初始化组件	26
1.5.6 事件处理程序	27
1.5.7 结束	27
1.6 小结	28
第 2 章 用户界面设计基础	29
2.1 回顾应用程序设计	29
2.2 为什么在 Windows 项目中使用 C#	30
2.3 C#语言和 Windows 应用程序	30
2.4 基于事件的编程	31
2.5 C#中的 Visual Studio 工具	31
2.6 标准控件	32
2.7 控件属性	34

2.7.1	改变默认的控制属性	36
2.7.2	改变几个控制属性的快捷方法	37
2.7.3	对象名和标签	37
2.7.4	事件处理程序	37
2.7.5	用代码改变属性	38
2.8	创建优秀的用户界面	39
2.8.1	优秀设计的组成	39
2.8.2	使用栅格	40
2.8.3	控件基础知识	40
2.8.4	营业税计算器	42
2.8.5	设计其他的控件	45
2.9	优秀的设计技术	50
第 3 章	设计时控件属性	52
3.1	属性	52
3.2	属性详解	52
3.3	修改属性	55
3.3.1	按钮属性	55
3.3.2	复选框控件属性	57
3.3.3	颜色对话框控件属性	58
3.3.4	组合框控件属性	59
3.3.5	日期时间采集器控件属性	59
3.3.6	字体对话框属性	61
3.3.7	窗体颜色属性	62
3.3.8	分组框控件属性	63
3.3.9	水平滚动条和垂直滚动条控件属性	64
3.3.10	标签控件属性	65
3.3.11	列表框控件属性	66
3.3.12	主菜单属性	67
3.3.13	月份日历属性	68
3.3.14	图形框控件属性	69
3.3.15	进度条属性	70
3.3.16	单选按钮控件属性	71
3.3.17	文本框控件属性	73
3.3.18	工具栏属性	74
3.3.19	轨道条属性	75
3.4	编写代码控制属性	77

第4章 运行时控件属性	78
4.1 为控件编写代码	78
4.2 加上代码的属性	79
4.3 事件处理程序	79
4.4 动态修改控件属性	81
4.4.1 按钮属性	81
4.4.2 复选框控件属性	82
4.4.3 日期时间捕获器属性	84
4.4.4 窗体颜色属性	86
4.4.5 垂直滚动条和水平滚动条控件属性	87
4.4.6 标签控件属性	89
4.4.7 列表框控件属性	89
4.4.8 月份日历控件属性	91
4.4.9 图形框控件属性	92
4.4.10 进度条控件属性	93
4.4.11 单选按钮控件属性	95
4.4.12 文本框控件属性	97
4.4.13 轨道条属性	99
4.5 再次研究营业税计算器	100
4.5.1 合理使用控件的功能	102
4.5.2 项目代码	104
4.6 小结	109
第5章 事件	110
5.1 事件处理程序的快速回顾	110
5.2 事件	111
5.2.1 Activate	113
5.2.2 ButtonClick	114
5.2.3 ButtonDropDown	114
5.2.4 CheckStateChanged	115
5.2.5 CheckedChanged	115
5.2.6 Click	115
5.2.7 Closed	116
5.2.8 CloseUp	116
5.2.9 Closing	117
5.2.10 DateChanged	117
5.2.11 DateSelected	118

5.2.12	Deactivate	118
5.2.13	DoubleClick	118
5.2.14	DragDrop	119
5.2.15	DragEnter	119
5.2.16	DragLeave	120
5.2.17	DragOver	120
5.2.18	DrawItem	121
5.2.19	Enter	121
5.2.20	Format	122
5.2.21	FormatQuery	122
5.2.22	GiveFeedback	122
5.2.23	Help	123
5.2.24	InputLangChange	123
5.2.25	InputLangChangeRequest	124
5.2.26	KeyDown	124
5.2.27	KeyPress	125
5.2.28	KeyUp	125
5.2.29	Layout	126
5.2.30	Leave	126
5.2.31	MDIChildActivate	127
5.2.32	MenuComplete	127
5.2.33	MenuStart	127
5.2.34	MouseDown	128
5.2.35	MouseEnter	128
5.2.36	MouseHover	128
5.2.37	MouseLeave	129
5.2.38	MouseMove	130
5.2.39	MouseUp	130
5.2.40	Move	131
5.2.41	PanelClick	131
5.2.42	QueryContinueDrag	131
5.2.43	Resize	132
5.2.44	Scroll	132
5.2.45	TextChanged	133
5.2.46	UserString	133
5.2.47	Validated	134
5.2.48	Validating	134

5.2.49 ValueChanged	134
5.3 小结	135
第6章 输入	136
6.1 控件和窗体	136
6.1.1 使用文本框控件	137
6.1.2 使用滚动条控件	158
6.2 鼠标	168
6.3 更多的输入	176
第7章 多窗体、菜单与通用对话框	177
7.1 多窗体	177
7.2 菜单	184
7.2.1 为菜单项编写代码	184
7.2.2 在菜单项上放置选择标记	192
7.3 通用对话框	193
7.3.1 添加一个颜色对话框	193
7.3.2 添加一个字体对话框	199
7.4 独特的用户输入	205
第8章 输出	206
8.1 将它发送出去	206
8.2 消息框输出	206
8.3 用文本框或标签控件输出	211
8.4 用多行文本框控件输出	217
8.5 使用文本框控件实现表格式输出	221
8.6 使用窗体进行表格式输出	227
8.7 输出到打印机	232
8.8 其他的输出技术和格式	240
第9章 图形基础	241
9.1 System.Drawing 命名空间	241
9.2 System.Drawing.Drawing2D 命名空间	244
9.3 C#中的图形类	246
9.4 坐标系统	250
9.5 绘图面	254
9.6 图形属性	255

9.6.1	颜色	255
9.6.2	线条绘制样式	257
9.6.3	刷子填充样式	258
9.6.4	DrawAndFill 项目	259
9.7	图形绘制元素	263
9.7.1	DrawArc	263
9.7.2	DrawEllipse	264
9.7.3	DrawLine	264
9.7.4	DrawPie	265
9.7.5	DrawPolygon	265
9.7.6	DrawRectangle	266
9.7.7	DrawString	266
9.7.8	FillEllipse	267
9.7.9	FillPie	267
9.7.10	FillPolygon	268
9.7.11	FillRectangle	269
9.7.12	DrawingPrimitives 项目	269
9.8	其他图形技术	274
第 10 章	图像和图形	275
10.1	展示图像或图片	275
10.2	一个简单的动画例子	279
10.3	一个图表项目	287
10.4	下一步该做什么	304
第 11 章	数值示例	305
11.1	基数变换计算器	305
11.2	素数计数器	312
11.3	三角函数表	319
11.3.1	Form2	319
11.3.2	Form1	324
11.4	在一个月中寻找天数	332
11.5	一天中的时间	339
11.6	统计	349
11.7	排序	356
11.8	接下来做什么	362

第 12 章 财务应用程序	363
12.1 在账户中定期存款	363
12.1.1 编写项目代码	364
12.1.2 定期投资与收益	370
12.1.3 从账户中定期取款	372
12.1.4 为定期取款算法编写项目代码	373
12.1.5 享用你的退休金	379
12.2 资产贬值	380
12.2.1 为窗体附上代码	380
12.2.2 资产贬值和数据共享	387
12.3 偿还贷款	388
12.3.1 为贷款偿还项目编写代码	389
12.3.2 查看贷款偿还选项	395
12.4 抵押分期付款表	397
12.4.1 为 Mortgage 项目编码	398
12.4.2 可能的收入税扣除	410
12.5 接下来做什么	411
第 13 章 专业质量的条形图表和饼图图表	412
13.1 条形图表	412
13.1.1 编写条形图表项目代码	413
13.1.2 绘制独特的条形图表	429
13.2 饼图图表	431
13.2.1 为饼图图表项目编写代码	432
13.2.2 绘制独特的饼图图表	444
13.3 下一步该做什么	445

第 1 章 C#与 Windows 简介

微软声称：“C#是一个从 C 和 C++演化过来的简单的、现代的、面向对象的和类型安全的编程语言”。在使用 C#（C sharp）的时候你会注意的第一件事是这种语言的结构是多么地熟悉了。通过面向对象的设计，C#提供了访问 Visual Basic 和 Visual C++类库的能力，然而，C#并没有它自己的类库。

C#是微软在 Microsoft Visual Studio 的最新版本中实现的，它提供了对下一代 Windows 服务（NGWS）的访问能力。这些服务包含了一个专门为代码开发而设计的通用执行引擎。

1.1 Visual Studio.NET 和 C#

最新版本的 Visual Studio.NET 为开发不同的应用程序提供了语言丰富的环境。程序可以由不同的语言开发，例如 C、C++、C#、Visual Basic 等等。应用程序也可以既包括标准的控制台（命令行或者是 DOS 模式）应用，又包括 Microsoft Windows 应用程序。

C#虽然是冲击这个市场的 C 的最新版本，但它只是这个更大的开发包中的一个组件。在发布 Visual Studio.NET 的时候，微软的目标之一就是能够为项目需求提供无缝解决方案。针对一个任务的解决方案可能需要将 C++、Visual Basic 和 C#中的不同组件集成到一个无缝可执行文件中。

本书着重于在创建 Windows 应用程序的时候对 C#的运用。如果你是那种想研究一种语言的所有细枝末节的程序员，这正是一本适合的书。

1.2 创建 C#应用程序

C#应用程序分为两类：命令行或控制台应用程序和 Windows 应用程序。使用 AppWizards，你会发现依据项目中的一些必需的模板代码，这两种应用和程序都很容易创建。

在我们的第一个项目中，我们创建一个大家熟悉的 Hello World 控制台应用程序。我们将这个项目命名为 Hello World。位于本章结束处的第二个应用程序叫做 CircleArea。这是一个完整的、面向对象的 Windows 应用程序。

在每一个项目中都尽量向你介绍 Visual Studio AppWizards 的使用方法，教你如何创建本书中开发的每一个项目的基本模板代码。这些代码都是你做标记的好地方，还可以在空白的地方做一些笔记。

1.3 第一个 C#控制台应用程序

要想用 C#创建一个控制台应用程序，首先启动 Visual Studio。选择 File | New | Project Sequence 来打开“New Project”对话框，如图 1-1 所示。

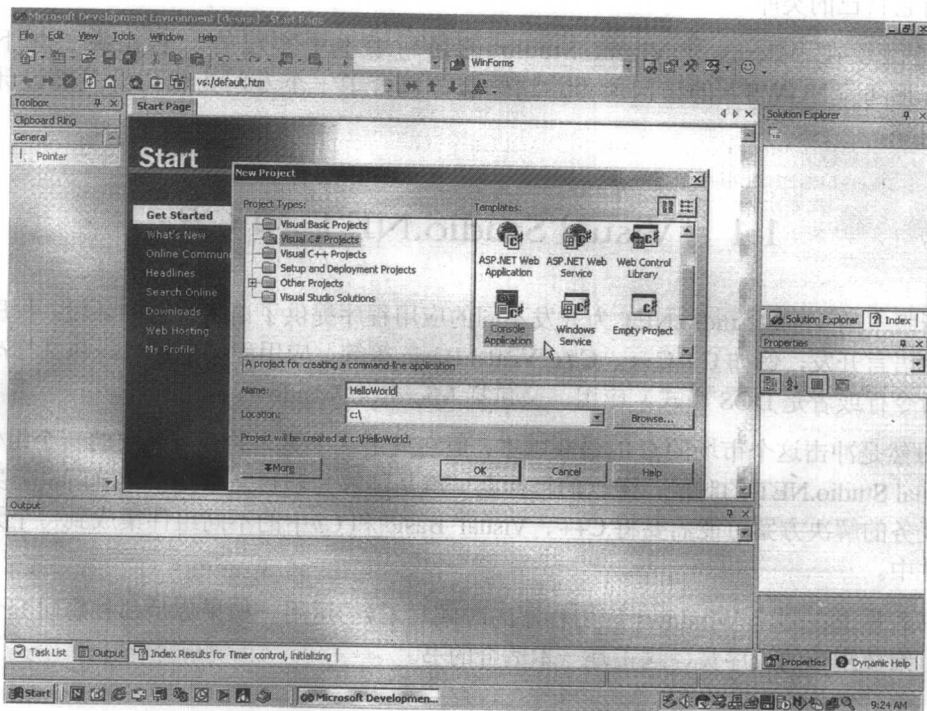


图 1-1 New Project 对话框使我们可以指定一个 C#控制台应用程序

将这个项目名称为 HelloWorld，并在根目录下面指定一个子目录，如图 1-1 所示。

当单击 OK 按钮的时候，C# AppWizard 就会创建如图 1-2 所示的模板代码。

现在可以修改这个模板代码来实现你的目标了。图 1-3 显示了我们是如何修改模板代码来完成 HelloWorld 项目的。

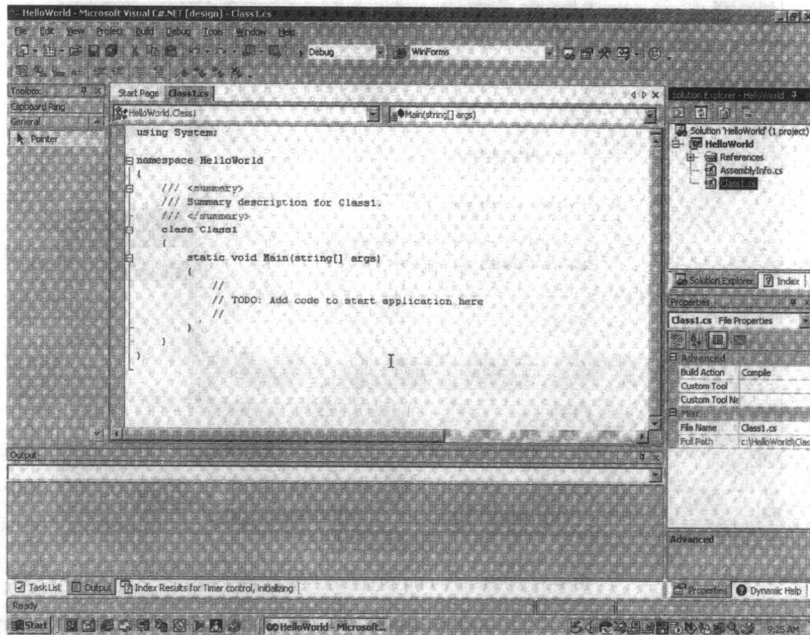


图 1-2 AppWizard 的控制台应用程序的 C#模板代码

观察图 1-3 并将其与下面的代码相比较。注意加黑了一行代码：

`using System;`

```
namespace HelloWorld
{
    /// <summary>
    /// Summary description for Class1.
    /// </summary>
    class Class1
    {
        static void Main(string[] args)
        {
            //
            // TODO: Add code to start application here
            //
            Console.WriteLine("Hello C# World!");
        }
    }
}
```

再次观察图 1-3，注意 Build 菜单已经被打开，而且正要选择 Rebuild 子菜单项。单击这个菜单项将创建这个项目。

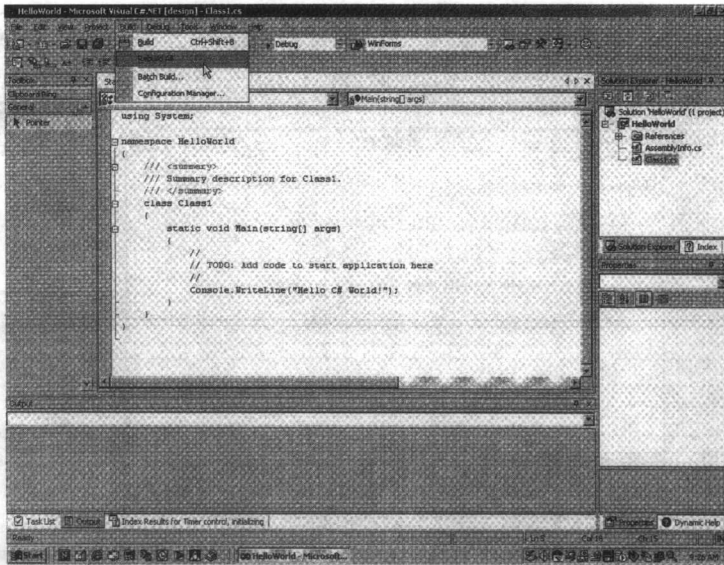


图 1-3 为 HelloWorld 项目修改模板代码

当观察这段例子代码的时候，你会发现这里的很多元素都是在 C 和 C++ 控制台应用程序中非常熟悉的了。图 1-4 显示了 Debug 菜单已经被打开，而且正要选择 Start Without Debugging 子菜单项时的情况。单击这个菜单项，就可以在 Visual Studio 集成开发环境中运行这个应用程序了。

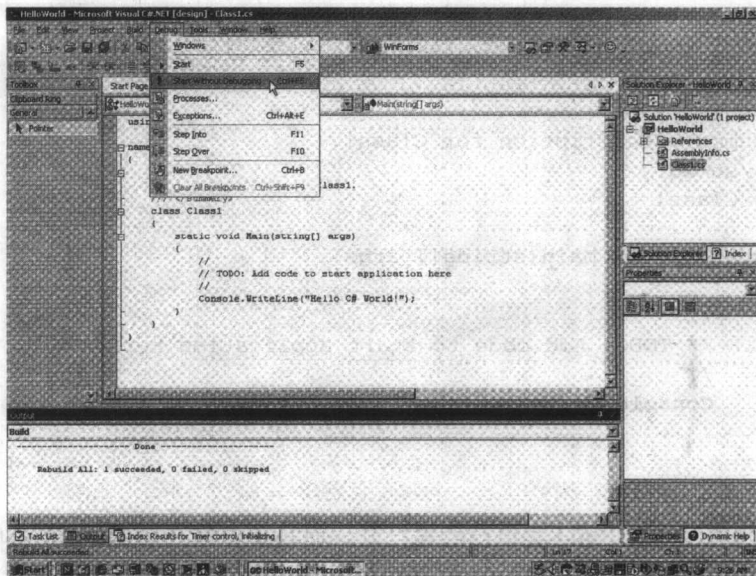


图 1-4 在 Visual Studio 集成开发环境中运行这个应用程序

当执行这个程序的时候，一个控制台（命令行或者是 DOS）窗口将会出现，并展示程序的输出。图 1-5 展示了这个应用程序的输出结果。

现在让我们简要浏览一下这个程序中那些熟悉的元素以及一些新的特性。首先，这个应用程序使用了系统指令。在运行时由 NGWS 提供的系统命名空间允许在 Main 方法中访问 Console 类。代码中所使用的 Console.WriteLine（）实际上是 System.Console.WriteLine（）的缩写形式，在这里 System 代表一个命名空间，Console 代表在这个命名空间中定义的一个类，WriteLine（）是在这个 Console 类中定义的一个静态方法。

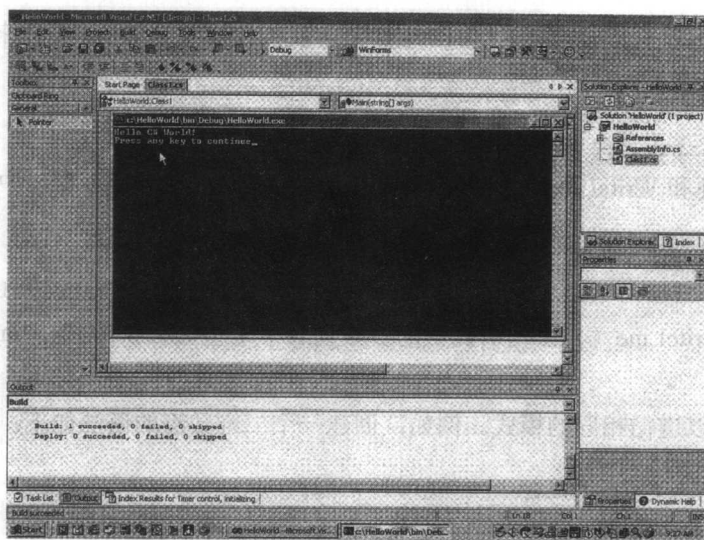


图 1-5 控制台窗口展示了项目的输出

其他程序细节

在 C#应用程序中，函数和变量都是包含在类和结构的定义中的，从来都不是全局的。

你可能会注意到在复合名之间使用“.”分隔符来加以分隔。C#使用这个分隔符来代替“::”和“->”。同时在 C#中也不需要前向声明，因为在这里顺序是不重要的。没有#include 语句表明 C#语言只是象征性地处理依赖关系。C#的另外一个特性是自动内存管理，这使得程序员可以从处理复杂的内存问题中解脱出来。

1.4 C#编程元素

本节将学习在本书中我们所要用到的关键的 C#语言编程元素。有时，我们会介绍一些额外的 C#知识，但是下面小节中的内容在今后将会多次用到。