



Developing C# Windows Software
A Windows Forms Tutorial

Windows

应用高级编程

—— C# 编程篇

Benny Johansen
Matthew Reynolds
张哲峰

等
译



清华大学出版社
<http://www.tup.com.cn>



Windows 应用高级编程

—— C#编程篇

Benny Johansen

Matthew Reynolds

等著

张哲峰

译

清华大学出版社

北京市版权局著作权合同登记号：01-2002-3182

内 容 简 介

本书全面介绍创建功能强大的 Windows 应用程序的各种高级编程技术。具体内容包括：利用各种控件创建图形用户界面，在 Windows 应用程序中使用图形和 GDI+，在标准 Windows 文件中加载和保存数据，通过应用程序打印信息，开发能够处理多文档的应用程序，链接数据库，提供帮助，部署应用程序等；同时，通过创建实用的应用程序示例来展示这些编程技术的用法和技巧。

本书适合希望掌握如何使用 Windows Forms 创建功能强大、稳健且界面友好的 Windows 应用程序的开发人员。另外，本书读者还要求具备一定的 C# 编程基本技能并熟悉数据库的概念。

Benny Johansen, Matthew Reynolds: Developing C# Windows Software

EISBN: 1-86100-737-X

Copyright© 2002 by Wrox Press Ltd.

Authorized translation from the English language edition published by Wrox Press Ltd.

All rights reserved.

Chinese simplified language edition published by Tsinghua University Press.

本书中文简体字版由英国乐思出版公司授权清华大学出版社出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，翻印必究。

本书封面贴有清华大学出版社激光防伪标签，无标签者不得销售。

图书在版编目(CIP)数据

Windows 应用高级编程/(美)杰汉森, (美)雷诺德著; 张哲峰译. —北京: 清华大学出版社, 2002

书名原文: Developing C# Windows Software

ISBN 7-302-06199-8

I. 无... II. ①杰...②雷...③张... III. 窗口软件, Windows—C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(2002) 第 106239 号

出 版 者: 清华大学出版社(北京清华大学学研大厦, 邮编 100084)

<http://www.tup.com.cn>

责任编辑: 夏兆彦

印 刷 者: 北京鑫丰华彩印有限公司

发 行 者: 新华书店总店北京发行所

开 本: 787×1092 1/16 印张: 21.75 字数: 556 千字

版 次: 2003 年 1 月第 1 版 2003 年 1 月第 1 次印刷

书 号: ISBN 7-302-06199-8/TP·3706

印 数: 0001~4000

定 价: 45.00 元

前 言

C#的强大功能配合 Visual Studio .NET 开发 Windows Forms 的简捷性,使 Windows 应用程序的开发更加迅速、便捷。

本书讲述如何使用 Windows Forms 和 C#设计、实现和部署功能强大的 Windows 应用程序。本书将阐述如何在应用程序中最有效地利用 Microsoft .NET Framework 类,并对 Visual Studio .NET IDE 的用法进行指导。其中, Visual Studio .NET IDE 是开发 Windows 应用程序的工具之一。各章将对 Window 应用程序设计的重要方面进行探讨,并通过创建实用的示例应用程序来具体说明它们的用法。通过学习本书,您将很快成为技术熟练的 Windows 应用程序开发人员。

0.1 本书读者对象

本书的读者对象是那些希望学习如何使用 Windows Forms 创建功能强大、稳健且用户友好的 Windows 应用程序的开发人员。学习该书之前,需要具备一定的 C#编程基本技能并熟悉数据库的概念。如果具有使用 Visual Studio .NET IDE 和 Windows Forms 的经验将会大有帮助,不过这并不是必要的条件。

0.2 本书主要内容

第 1 章——Windows 编程基础。首先使用 C#以及开发 Windows 应用程序时所使用的主要用户界面组件来创建第一个 Windows 应用程序。

第 2、3 和 4 章。这 3 章介绍如何使用 Visual Studio .NET 工具箱中的控件创建用户界面 (User Interface)。

第 5 章——使用图形。本章简要介绍如何在 Windows 应用程序中使用图形和 GDI+。

第 6 章——打印。本章介绍如何才能使应用程序具有打印功能,以便将结果输出到屏幕上,这样用户就可以将输出结果发送到与本地计算机相连或与网络相连的打印机进行打印。

第 7 章——链接数据库。本章介绍如何将 .NET 应用程序连接到 DBMS 数据库和如何通过应用程序代码对数据库数据进行操控,以及如何检索并显示数据库数据。



第 8 章——文件和注册表操作。本章将介绍两个可以相互替代的存储数据的位置：文件系统和注册表（Registry）。我们将剖析那些用于将数据（尤其是文本）读写到文件系统的类。讲述如何导航文件系统，并演示如何将对象状态串行化为 XML 格式，然后探讨如何组织注册表以及那些用于访问它的类。

第 9 章——使用文档/视图模型。本章将介绍如何同时处理多个文档，并展示如何设计和实现可同时处理多个文档的 Windows 应用程序。

第 10 章——提供帮助。首先讲述创建帮助系统的设计指导原则，然后分析可以使用的帮助创建工具。另外，本章还要介绍将外部帮助系统链接到 Windows 应用程序的支持，该支持由 .NET Framework 提供。

第 11 章——部署 Windows 应用程序。本章将介绍 .NET Framework 提供的一个支持，该支持可将 Windows 应用程序部署到终端用户计算机上。另外，还要介绍 .NET Framework 提供的不同部署选项类型。

0.3 使用本书的前提条件

要想很好地学习本书，应该具备一定的知识及必要的软件：

软件

- Microsoft Windows 2000 或 XP
- Visual Studio .NET 完全版或 Visual C#标准版

知识

- 已掌握一些 C#语言方面的知识

0.4 客户支持

我们一贯重视读者的意见，并想知道每位读者对本书的看法，包括读者喜欢和不喜欢的内容，以及读者希望我们下一次完善的地方。您可以通过发送电子邮件(地址为 feedback@wrox.com)向我们反馈意见。请确保在反馈信息中提到本书的书名。

0.4.1 如何下载本书的示例代码

当您访问 Wrox 公司站点(地址为 <http://www.wrox.com/>)时，通过 Search 工具或书名列表，可以方便地定位需要的书目。然后，单击 Code 列中的 Download 超链接，或

者单击本书的详细页面中的 **Download Code** 超链接，就可以下载相应的示例代码。

当您单击下载本书中的代码时，将会看的带有以下三个选项的 Web 页面：

- 如果您是 **Wrox Developer Community** 的成员(即如果您已经在 **ASPToday**、**C#Today** 或者 **Wroxbase** 上注册过)，就可以使用一贯的用户名和密码进行登录，下载代码。
- 如果不是其成员，则会询问是否愿意注册为会员，以便可以免费下载代码。此外，也可以从 **Wrox Press** 下载免费的文章。注册为会员后，可以得到本书升级版本和新版本的有关信息。
- 第三个选项是完全绕过注册过程，直接下载代码。

对于本书而言，不注册也能下载代码，但是，如果您愿意注册后下载代码，则您的注册信息不会泄漏给第三方。关于这方面的详细条款和条件，可以通过点击下载页面上的相关链接来察看。

从我们的站点下载的文件都是使用 **WinZip** 压缩过的文档。保存文件到本地磁盘上的文件夹中后，需要使用一个解压缩程序(例如 **WinZip** 或 **PKUnzip**)来解压缩文件。在解压缩文件时，通常将代码解压缩到每一章所在的文件夹中。在解压缩的过程中，应确保解压缩程序(**WinZip**、**PKUnzip**，其他)被设置为使用原有文件夹名。

0.4.2 勘误表

我们已经尽最大努力确保本书中的文本和代码没有错误，但是错误仍然在所难免。如果您发现本书存在错误，例如拼写错误或不正确的代码段，请给我们反馈信息，我们将不胜感激。勘误表的发送可以节约其他读者学习本书的时间，而且能够帮助我们提供更高质量的信息。您的反馈信息将被检查，如果正确，将被粘贴到本书的勘误页面上，或者在本书的后续版本中使用。

要在我们的站点上找到勘误表，请访问 <http://www.wrox.com/>，并通过 **Advanced Search** 或者书名列表轻松定位本书页面。然后，单击 **Book Errata** 超链接即可，该链接位于本书的详细信息页面中的封面图解下面。

0.4.3 E-Mail 支持

如果您希望直接向详细了解本书的专家咨询本书中问题，可以发送电子邮件到 support@wrox.com，要求在邮件的主题栏中带上本书的书名和 **ISBN**(国际标准图书编号)的后 4 位数字。一封典型的电子邮件应包括下面的内容：

- 在主题栏中必须有本书的书名、**ISBN** 的后 4 位数字和问题所在的页码。



- 正文应包括读者的名字、联系信息和问题。

我们将不向您发送无用邮件，因为我们仅仅需要有用的详细资料，以便可节约您和我们的时间。当您发送一条电子邮件信息时，它将经过下面一系列的支持：

- 客户支持：首先，您的信息将被递送到我们的客户支持人员手中，并由他们阅读。对于一些被频繁提到的问题将被归档，并将立即回答有关本书或者 Web 站点的任何常见问题。
- 编辑支持：接着，一些有深度的问题将被送到对本书负责的技术编辑手中，他们在程序设计语言或者特定的产品上有着丰富的经验，能够回答相关主题的详细技术问题。
- 作者支持：最后，如果编辑不能回答您的问题(这种情况很少发生)，他们将请求本书的作者。我们将尽量保护作者免受干扰，以便不影响其写作。然而，我们也非常高兴转寄给他们一些特殊的问题。所有 Wrox 公司的作者都为他们的图书提供技术支持。作为回应，他们将发送电子邮件给用户和编辑，进而使所有的读者受益。

Wrox 公司的支持过程仅仅对那些与我们出版的书目内容直接相关的问题提供支持，对于超出常规书目支持的问题，您可以从 <http://p2p.wrox.com/forum> 中的公共列表中获得支持信息。

0.4.4 p2p.wrox.com 站点

为了便于作者和其他人讨论，特将讨论内容加入到 P2P 站点的邮件列表中，而且我们惟一的系统将 **programmer to programmer™**(由程序员为程序员而作)的编程理念与邮件列表、论坛、新闻组以及所有其他服务内容(一对一的邮件支持系统除外)相联系。如果您向 P2P 发送一个问题，应该相信它一定会被登录邮件列表的 Wrox 公司作者和其他相关专家查看到。无论您是在阅读本书，还是在开发自己的应用程序，都可以在 p2p.wrox.com 站点中找到许多对自己有所帮助的邮件列表。

遵循下面的步骤可以订阅一个邮件列表：

- (1) 登录 <http://p2p.wrox.com>/站点。
- (2) 从左边的主菜单栏选择一个适当的类别。
- (3) 单击希望加入的邮件列表。
- (4) 按照说明订阅并填写自己的邮件地址和密码。
- (5) 回复您收到的确认邮件。
- (6) 使用订阅管理程序加入更多的邮件列表并设置自己的邮件首选参数。

本系统提供最佳支持的原因

您可以加入到整个邮件列表，也可以只接收每周的邮件摘要。如果您没有时间和工具来接收邮件列表，可以直接查找我们的在线文档。独特的 Lyris 系统可以将一些没有用的垃圾邮件删除，并保护您的电子邮件地址不被侵扰。当存在加入和离开列表、以及任何有关列表的其他常见问题时，请发送邮件到 listsupport@p2p.wrox.com。

目 录

第 1 章	Windows 编程基础	1
1.1	Windows 和窗体	1
1.1.1	Windows 窗体的概念	1
1.1.2	System.Windows.Forms 命名空间	2
1.2	Visual Studio .NET IDE 简介	3
1.2.1	解决方案和项目	3
1.2.2	Toolbox 和 Properties 窗口	5
1.2.3	动态帮助窗口	9
1.2.4	类视图	9
1.2.5	对象浏览器	10
1.2.6	代码编辑器	11
1.3	事件处理	13
1.3.1	使用消息框	14
1.3.2	使用 IntelliSense	17
1.4	小结	18
第 2 章	简单用户界面控件	19
2.1	用户界面设计原则	19
2.2	漫游简单用户控件	20
2.2.1	PictureBox 控件	20
2.2.2	Label 控件	21
2.2.3	Textbox 控件	22
2.2.4	List 控件	24
2.2.5	基于按钮的控件	29
2.2.6	Bar 控件	39
2.3	小结	40
第 3 章	高级用户界面控件	41
3.1	高级控件简介	41
3.2	概述	41
3.2.1	TabControl 控件	42



3.2.2	TreeView 控件和 ListView 控件	51
3.2.3	Splitter 控件	61
3.2.4	DateTimePicker 控件	62
3.2.5	MonthCalendar 控件	66
3.3	控制光标外观	69
3.4	小结	71
第 4 章	菜单和工具栏	73
4.1	菜单控件和类	73
4.1.1	创建主菜单	74
4.1.2	创建上下文菜单	83
4.2	工具栏控件和类	85
4.2.1	创建工具栏	85
4.2.2	显示/隐藏工具栏	90
4.3	小结	94
第 5 章	使用图形	96
5.1	设备语境	96
5.2	Graphics 类	97
5.2.1	使用 Graphics 类	97
5.2.2	Graphics 类的方法	98
5.3	指定位置、大小和颜色	99
5.3.1	位置和大小	99
5.3.2	颜色	101
5.4	绘制线条和外形	103
5.4.1	Pen 类	103
5.4.2	画图方法	104
5.4.3	按要求画图	115
5.5	颜色填充	120
5.5.1	刷子工具	120
5.5.2	填充方法	122
5.6	使用文本	127
5.7	图像	134
5.8	小结	138
第 6 章	打印	139
6.1	System.Drawing.Printing 命名空间	139

6.2	打印图像	141
6.2.1	更改页面设置	148
6.2.2	打印对话框	151
6.2.3	保持高宽比	153
6.3	打印多页	158
6.3.1	分页	158
6.3.2	添加页眉	168
6.4	小结	171
第 7 章	链接数据库	172
7.1	.NET 数据库连接性	172
7.2	连接数据库	174
7.3	加载和显示数据	181
7.4	修改数据库	187
7.5	表的关系	192
7.6	利用自定义查询	199
7.7	小结	202
第 8 章	文件和注册表操作	203
8.1	读写文件	203
8.1.1	流的概念	203
8.1.2	Stream 类	204
8.1.3	读写文本	208
8.2	访问目录信息	220
8.2.1	DirectoryInfo 类	221
8.2.2	FileInfo 类	222
8.3	串行化对象	228
8.4	操作注册表	238
8.4.1	注册表的结构	239
8.4.2	用于注册表操作的类	240
8.5	小结	248
第 9 章	使用文档/视图模式	249
9.1	什么是文档/视图模式	249
9.2	创建文档/视图应用程序	251
9.2.1	管理多文档窗口	252
9.2.2	添加文件视图功能	257



9.2.3	添加文件保存功能	263
9.2.4	在窗口间切换	275
9.3	小结	281
第 10 章	提供帮助	282
10.1	用户需要帮助系统的原因	282
10.2	用于创建帮助系统的设计原则	283
10.2.1	呈现	283
10.2.2	导航	284
10.2.3	内容	284
10.3	帮助创建系统	285
10.4	HTML Help 详述	286
10.5	.NET 提供的帮助支持	293
10.5.1	Help 类	294
10.5.2	HelpProvider 类	294
10.5.3	HelpNavigator 枚举	295
10.5.4	通过 Help 按钮显示弹出式帮助	302
10.5.5	将弹出式帮助显示为字符串	304
10.5.6	使用工具提示提供控件帮助	305
10.6	小结	307
第 11 章	部署 Windows 应用程序	308
11.1	部署和安装	308
11.2	在 .NET 环境下部署	308
11.2.1	通过 XCOPY 部署进行安装	309
11.2.2	使用 Visual Studio .NET Installer 进行部署	310
11.2.3	使用 XCOPY 还是 Windows Installer	311
11.2.4	Visual Studio .NET 中的部署项目模板	312
11.2.5	部署项目属性	315
11.2.6	使用安装编辑器	316
11.2.7	创建安装程序程序包	326
11.2.8	安装应用程序	327
11.2.9	卸载或准备应用程序	331
11.3	小结	332

第1章 Windows 编程基础

随着 .NET 和 C#语言的发布, Microsoft 已经为我们建立多样化的应用程序提供了一套非常有效而且功能强大的工具, 这些应用程序包括交互式的 Web 应用程序和基于 XML 的 Web 服务。然而, 本书的重点在于开发桌面 Windows 应用程序。尽管它不可能取得和同 Web 相关的技术那样快速的发展, 但是随着 .NET 的到来, 桌面应用程序的开发已经发生了重大的变革。阅读完本书后, 您应该可以愉快地使用 Visual Studio .NET 和 C# 语言开发 Windows 应用程序。

本章将开始使用 C#建立第一个 Windows 应用程序。然后, 我们将介绍各种主题, 为以后的章节打下基础, 这些章节都将向您的应用程序添加更加复杂的功能。学完本章后, 您将可以:

- 熟悉在开发 Windows 应用程序时使用的主要用户界面(User Interface)组件。
- 熟悉 Visual Studio .NET IDE 以及它为 Windows 开发人员提供的工具。
- 掌握如何开始一个新的基于 Windows 窗体的应用程序项目。
- 掌握如何像单击一个按钮一样响应用户事件。
- 掌握如何向用户显示一个消息框。

1.1 Windows 和窗体

在 .NET 中, 经常会看到可交换使用的两个术语 `window`(窗口)和 `form`(窗体)。这是因为在 .NET 的术语范畴中, 它们是可以互换的。到目前为止, 您可以在屏幕上的标题栏中指出一个对象作为窗口。当然, 现在这些被指定为窗体更加合适。无论如何, 我们现在要学习的是将要创建的每一个可视化 Windows 应用程序的主要元素——Windows 窗体(Windows Form)。

1.1.1 Windows 窗体的概念

Windows 窗体是一个可以用来为用户提供信息以及接收其输入的窗口——换句话说, 可以用窗体创建一个用户界面。窗体可以是标准的窗口、多文档界面(MDI)窗口、对话框、或者图形例程的显示界面。您通常会使用 Visual Studio .NET IDE 创建一个基于 Windows 窗体的应用程序, 我们会在下 1.2 节中讨论它。



正如 .NET Framework 中的所有对象一样，窗体是类的实例。在 Visual Studio .NET Designer(设计器)中创建的窗体是一个类，并且当您在运行阶段显示这个窗体的实例时，这个类是用于创建窗体的一个模板。窗体是对象，提供定义外观的属性，定义行为的方法，以及定义与用户交互的事件。通过设置窗体的属性以及编程来响应事件，这样就可以定制对象，以满足应用程序的要求。

Framework(框架)也允许继承现有的窗体，然后添加功能或者修改现有的行为。在项目中添加一个窗体时，您可以选择是直接从 Framework 所提供的 `System.Windows.Forms.Form` 类中继承，还是从您已经创建的窗体中继承。要获取更多关于窗体类的信息(或者框架中的其他任何类)，请参阅 MSDN 资料库，网址是 <http://msdn.microsoft.com/library/>。

1. 控件

控件是对象，它可以向用户显示信息，或者响应用户的输入。它们包括在 Form 对象中。每一种控件具有它本身的一组属性、方法和事件，这使得它可以适用于特定的目的。结合使用不同的控件和自己的代码，您可以获取用户信息并响应它，还可以操作现有的数据存储，查询并回复在用户的本地计算机上的文件系统和注册表。在 Visual Studio .NET Designer 中，可以在窗体中操作控件，甚至在运行时通过编程动态地添加控件。在接下来的几章中，我们将更进一步地学习更多的通用控件。

最后，应该注意窗体也是控件的一种，因为它们是从 `System.Windows.Forms.Control` 类中继承而来的。

2. 菜单和工具栏

菜单是用户获取应用程序中主要功能和实用程序的简单途径，例如，打开文件、创建新文件、保存文件。工具栏是另一种获取应用程序主要功能的常用方法：例如工具栏里的“save”按钮。在第 4 章里我们将深入学习菜单和工具栏的内容。

1.1.2 System.Windows.Forms 命名空间

您可能从一个开发人员的角度认识到 .NET 的核心是 .NET Framework 类库。它提供了获取 .NET 中所有公共类和接口的渠道，它也是所有应用程序、组件以及控件的创建基础。相关的类和接口都被分组存放在命名空间里。

可以看出，当我们开发基于 Windows 窗体的 Windows 应用程序时，最应该关心的命名空间是 `System.Windows.Forms`。这个命名空间包含了所有创建基于 Windows 应用程序的类，这些应用程序包括窗体、控件、组件(像菜单)和通用对话框(打开、保存、字体和打印)。这个命名空间包含了大约 200 个类。不过，其中大多数的类被其他类后

台调用, 或者在 Visual Studio IDE 中被可视化操作。

关于 System.Windows.Forms 命名空间, 一般会误解为它取代了 Win32 API (Application Programming Interface)。事实上, Windows 窗体仅仅可以被看作一个基本级别的包装器, 或者, 用开发人员友好的方式来看, 它使用了 .NET 之前开发工具用过的同一系列的 OS 功能。也就是说, 以下列方式包装所有功能和操作: 开发人员能够更少地关注对 Windows 机制的理解, 而把更多的精力放在创建功能强大的应用程序上面, 那正是我们要做的。如果您对于有关 .NET 内核方面的知识很感兴趣, 请浏览目前可以参考的大量其他书目, 像清华大学出版社引进出版的《.NET Framework 高级编程》等。

1.2 Visual Studio .NET IDE 简介

现在 Visual Studio IDE 家族已经以多种形式活跃一段时间了。尽管其中的大部分看上去与以前使用的版本非常相似, 但 Visual Studio 的 .NET 版本的确有了重要的改进和补充。许多单独的应用程序已经被统一到一个开发环境之中。例如, Visual Studio 6 中的 InterDev 就不再是一个单独的应用程序, 它集成于 IDE 中; SQL Server 也通过 Server Explorer 集成到 IDE 中。我们可以为解决方案添加不同种类的项目, 例如, VB.NET 和 C#(Windows、ASP.NET、或者 Class Library 项目等等)。Microsoft 已经完成了一项很好的工作, 使得 Visual Studio .NET 变得尽可能直观, 所以我们首先回顾一个基础知识, 然后深入学习后续内容。

说明:

有些用户可能没有使用 Visual Studio .NET 的完整 Professional 版本, 而使用标准 Visual C# IDE(它实际上是 Visual Studio .NET 的一个精简版本, 它只可以用 C#开发应用程序)。因此要确保这本书中涉及的示例也可以在这两个 IDE 上运行。

1.2.1 解决方案和项目

在 Visual Studio .NET 中, 解决方案和项目包罗了所有的工作。一个项目可以是一个单独的 Windows 应用程序(.exe)、一个类库(.dll)、或者甚至是一个网站。一个解决方案仅仅是项目的集合, 它是 Windows 应用程序、Web 应用程序和 Web 服务的集合。例如, 可以在解决方案中放置应用程序的可执行文件以及所有与它相关的.dll 程序。应该注意到当您创建一个新的项目而没有特别说明将它添加到当前解决方案中时, Visual Studio .NET 会为您创建一个新的解决方案文件。由于解决方案和项目在目录结构中创建的方式, 所以在此建议创建一个解决方案的文件夹和包含单个项目的子文件夹(解决



方案文件和项目文件存在于同一个目录中时，很容易混淆)。

虽然项目文件的扩展名由所包含项目的类型来决定，解决方案文件的扩展名总是.sln。C#项目的扩展名是.csproj。

显而易见，学习 Visual Studio .NET 的最好方法是打开并实际使用它。下面让我们一起来学习。

试一试：创建一个新的 Windows 应用程序项目

在第一个示例中，我们从简单处着手，在 Visual Studio .NET 中打开一个新的 Windows 应用程序项目，以便仔细研究所使用的开发环境。

(1) 启动 Visual Studio .NET。可以看到 Start 页面，使用 New Project 按钮创建新项目，或者也可以使用 File | New Project... 菜单命令弹出如图 1-1 所示的对话框。

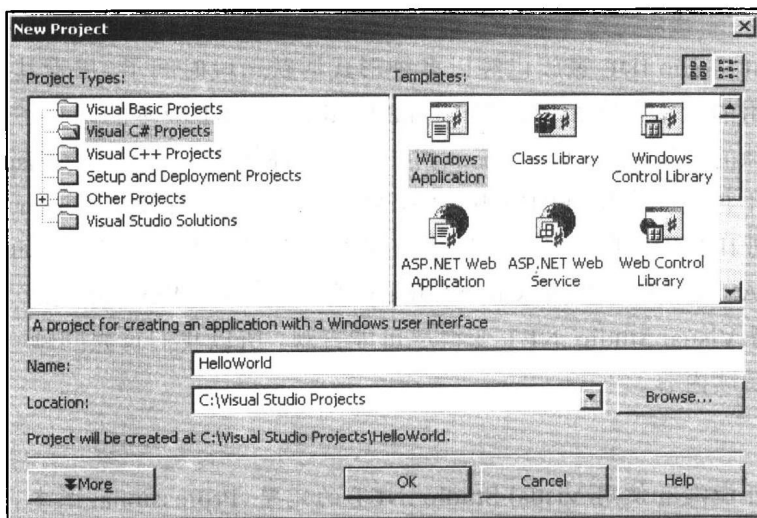


图 1-1

(2) 选择 Visual C# Projects 文件夹，再选择 Windows Application 模板。可以将项目命名为任何您喜欢的名字，这里将所创建的项目命名为 HelloWorld。您也可以选择保存解决方案和项目的位置。

要说明的一点是，Visual Studio .NET 会以您所提供的位置作为基本目录，并且在那个位置下以相同的名字创建一个子文件夹作为项目。所有的项目文件都会存放于此目录下，包括项目(.csproj) 和解决方案(.sln) 文件本身。如果单击 More 按钮，您可以创建一个初始解决方案目录，然后在其自身的子文件夹中包含项目文件。后者为包含多重项目的解决方案所普遍使用。

在本书中，我们主要研究 Windows 应用程序类型的项目，但是在第 11 章将会介绍安装和部署项目。

(3) 一旦创建了新项目，就可以看到 Visual Studio .NET IDE 的全貌(见图 1-2)。

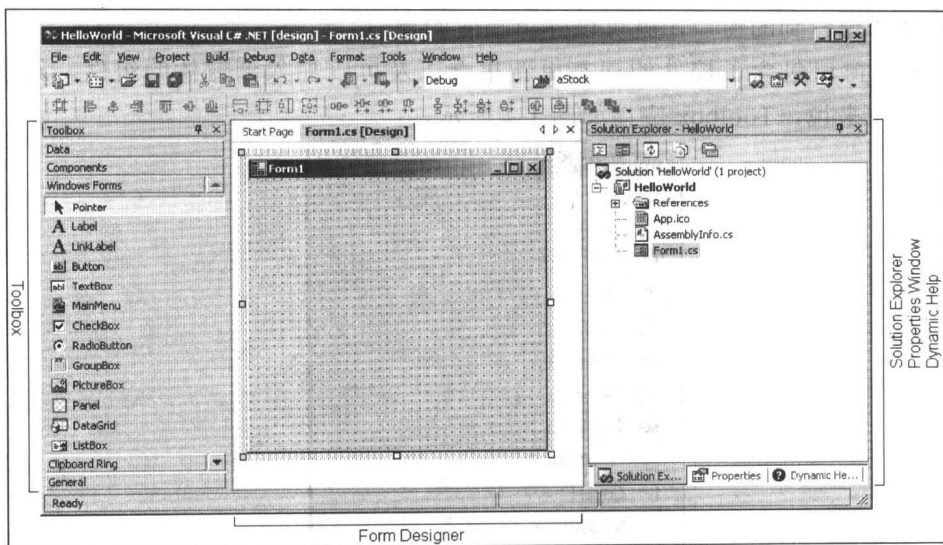


图 1-2

应该注意到, IDE 被分为几个面板, 主要有: **Toolbox**(工具箱)、**Form Designer**(窗体设计器)和一个包含 **Solution Explorer**(解决方案资源管理器)、**Class View**(类视图)、**Properties**(属性)窗口和 **Dynamic Help**(动态帮助)窗口的面板。设计器允许您把存储在 **Toolbox** 中的控件拖放到窗体中。在设计器中的窗体中双击会弹出 **Code Editor**(代码编辑器)窗口; IDE 会自动产生窗体类框架。**Solution Explorer** 窗口允许您快速获取解决方案的内容(见图 1-3)。



图 1-3

让我们进一步讨论 IDE 中使用最频繁的面板和窗体。

1.2.2 Toolbox 和 Properties 窗口

开发 Windows 应用程序常用的两个窗口是 **Toolbox** 和 **Properties** 窗口。进入 **View**