



普通高等教育“十一五”国家级规划教材

新世纪高职高专实用规划教材

● 计算机系列

C# 面向对象程序设计

C# MIANXIANG DUIXIANG CHENGXU SHEJI

邵鹏鸣 编 著

赠 送
电子课件



清华大学出版社

普通高等教育“十一五”国家级规划教材
新世纪高职高专实用规划教材 计算机系列

C#面向对象程序设计

邵鹏鸣 编著

清华大学出版社

北京

内 容 简 介

本书详细介绍了如何使用 C#语言进行面向对象编程的知识。书中贯穿了面向对象的思想,使初学程序开发的读者能够逐步体会并深刻理解“对象”技术的强大功能。同时,还以先入为主的方式引入了事件的处理与 GUI 的使用等知识。书中通过具体的实际应用程序示例讲述使用 ADO.NET 和 SQL 访问数据库的编程技术。并通过多个详尽的案例分析,使读者能够尽快地掌握面向对象的编程技巧。此外,书中提供了大量与开发有关的技术要点,并配合实用、有效的 GUI 应用程序,使读者能够迅速掌握并巩固所学到的知识。本书汲取了很多来自实际编程中的经验和做法;这将为读者培养良好的编程习惯打下一个坚实的基础。

本书可作为高职高专计算机专业和高等技术型院校进行编程语言教学和 C#教学的教材,也可供程序设计 with 开发人员参考。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

C#面向对象程序设计/邵鹏鸣编著. —北京:清华大学出版社,2008.4

(新世纪高职高专实用规划教材 计算机系列)

ISBN 978-7-302-17167-6

I. C… II. 邵… III. C 语言—程序设计—高等学校:技术学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2008)第 028817 号

责任编辑:黄 飞 宋延清

封面设计:杨玉兰

版式设计:北京东方人华科技有限公司

责任印制:何 芊

出版发行:清华大学出版社

地 址:北京清华大学学研大厦 A 座

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

邮 编:100084

社 总 机:010-62770175

邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

印 刷 者:北京市清华园胶印厂

装 订 者:北京市密云县京文制本装订厂

经 销:全国新华书店

开 本:185×260 印 张:30.25 字 数:725 千字

版 次:2008 年 4 月第 1 版 印 次:2008 年 4 月第 1 次印刷

印 数:1~4000

定 价:39.50 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。联系电话:(010)62770177 转 3103 产品编号:027953-01

前言

随着 21 世纪的到来, 计算机技术的发展更加迅猛, 在各行各业的应用更加广泛, 面对日新月异的新技术、新方法, 我们必须对现有计算机课程的设置和教学内容进行调整, 以适应技术进步和市场变化的需要, 使讲授的内容是市场上最需要的。

今天, 应用程序已由驻留在用户硬盘上的独立可执行文件发展为由 Web 服务器在 Internet 上传送的分布式应用程序; 相应地, 任何一种开发平台及程序设计语言都必须适应这种变化。微软的 .NET 是一种开发平台, C#(读作 C sharp)是微软特别为 .NET 平台设计的一种现代编程语言, .NET 有着广阔的应用前景, .NET 的应用必将对整个计算机产业产生重要而深刻的影响。NET 不但改变了开发人员开发应用程序的方式及思维方式, 而且使开发人员能创建出各种全新的应用程序, 大幅度提高软件生产率。未来 .NET 将无处不在。

微软以 .NET 平台和 C#、Visual Basic .NET 编程语言的形式推出了与 Java 全面竞争的对手。这是微软在编程语言和编程环境这块竞技场上所取得的最出色的成果。当然, 微软具备很有利的条件: 可以看到 Java 在什么方面做得好, 在什么方面做得还不够好, 然后, 基于此去构建 C#, 使其具备 Java 所不具备的优点。

目前, 与 Java 相比, 对 .NET 平台人们所关心的最重要的问题是, 微软是否会将它完全移植到其他平台上。微软宣称这没有问题, 而且 Mono 项目(www.go-mono.com)已经有了一个在 Linux 上运行的 .NET 的部分实现。

C#是一种简单的、现代的、面向对象的且类型安全的编程语言, C#语言从 C 和 C++语言演化而来, C#同时具备“快速应用程序开发(RAD)”语言的高效率和 C++固有的强大能力。同时它吸收了 Java 语言的特点和精华, 熟悉 Java 的人会觉得它很像 Java。比尔·盖茨曾说过“Java 是最卓越的程序设计语言”, 不过从 C#诞生的那一时刻起, 这已成为过去。C#是微软将 Java 集成到 .NET 中的产物, 它是整个 .NET 平台的基础, 是未来主流的编程语言。

我们知道 C 语言是一种代码效率很高但不易于进行快速开发的程序设计语言。面向对象程序设计(Object-Oriented Programming, OOP)出现后, OOP 与 C 语言融合, 产生了 C++, 继而有了 C++集成开发环境。但 C++的出现, 并没有使 C 语言家族在应用开发方面获得突飞猛进的发展, 基本上仍然只占据着 UNIX、Linux 以及底层应用开发的天地, 而在 Windows 大型应用软件特别是数据库和 Web 开发上, 由于 C 语言编程固有的复杂性和缺乏针对性, 就不如 Visual Basic(一般可简称为 VB)等具有很强针对性的开发工具。那些使用过包括 C 和 C++在内的多种程序设计语言的人, 相信会深深地体会到它们之间的区别。比如与 Visual Basic 相比, Visual C++(一般可简称为 VC++)程序员为实现同样的功能就要花费更长的开发周期。

另一方面虽然 C 和 C++为我们带来高度的灵活性, 但必须忍受学习的艰苦和开发的长期性, 特别对 VC++来说, 大部分的程序结构都被封装在 MFC(微软基本类库)中。所以对于初学者来说, 程序结构显得十分混乱, 学习将变得十分艰苦。而且从 VC++ 2.0 起,

为了适应不断更新的技术(例如 COM、ATL 等),以及为了与前一个版本兼容,VC++的每一次升级都给程序员带来一份痛苦——程序结构变得越来越复杂,而且出现了越来越多的变量类型,从而带来了更多的问题。

C#的出现弥补了C语言家族的上述不足,它借鉴了Java、C++、C甚至VB的优点,既具有C、C++那样的强大功能,又具有Java那样的面向对象机制,同时还具有类似于VB开发的高效率和方便性。

学习一种先进的语言和一种先进的编程方法将激起学生更大的兴趣。这些知识在他们离开学校,进入一个信息占据重要地位的世界时可以立即发挥作用。当今的学生必须同时掌握基础语言、面向对象编程和类库知识,而C#课程适合这类知识的学习。

很多院校都设立了软件专业,有的学校的学生,在几年的学习时间里,不仅学习了C、C++,而且也学习了Java、VB或VB.NET甚至C#,同时还学习了Web应用程序设计及其他有关软件开发的课程。

但据作者对一些院校的调查,却发现有些院校软件专业的学生在毕业找工作时,竟然很难找到一份对口的工作,所找到的工作与软件无关、甚至与计算机无关——不是企业不需要软件专业人才,而是没有物色到所需要的人才。

我认识一位学生,学习成绩不错,学习也很勤奋,在毕业前夕还通过了全国计算机技术与软件专业技术资格(水平)考试,获得了软件设计师资格。但当我交给他一份实际任务时,他却根本不知从何下手,尽管该任务很简单。这些都说明,我们的教学体系、教学内容、教学模式和方法都存在着很多需要改进的地方。

多年来,作者一直都对计算机软件专业课程的课程体系、结构、课程内容和教学模式、方法进行研究,从宏观和微观的角度把握每门课程,从市场和可持续性的角度设计每门课程。将基本技能培养与主流技术相结合,把软件工程的思想融入到教学体系中,在职业能力分析的基础上,以工作任务为中心来整合相应的知识、技能,从应用与工程实践的角度出发来设计课程体系、课程内容和教学模式。

检验成功与否的标准是“不管白猫黑猫,抓到老鼠就是好猫”,也就是说,不管你怎么说,怎么做,学生学完后,能够将知识转化为能力、能够将知识转化为应用,并具有可持续性,就是成功的。否则,难免是纸上谈兵、言之无物、空中楼阁。根据这种指导思想,作者努力将教学理念和思想转化为实际应用,对计算机软件专业主干课程的课程体系、结构、课程内容和教学模式、方法进行了改革和实践。2006年作者在清华大学出版社出版了两部教材。在机械工业出版社出版了一部教材,2007年在清华大学出版社又出版了作者项目研究中的另外两部教材,它们是作者的教学理念和思想的结晶,是作者多年来教学研究与实践的结晶。

作者极力反对用“玩具例程”说明概念、原理的教学方法。那些例子完全脱离实际,看起来很幼稚,尽管它能够说明一个概念,学生也容易看懂,上课也容易听懂,但是检查学习好坏的标准,不是“知道还是不知道,听懂还是没有听懂,看懂还是没有看懂”,而是“会不会应用”。真实例程同样能够说明概念、原理和知识点,只不过是面对复杂的技术实践体系,需要花大量的时间和精力精心地去设计这样的例程,只要设计得好,真实例程同样能够说明概念、原理和知识点,学生同样容易看懂、听懂,并且懂得在实际应用中如何运用这些概念、原理和知识点,获得事半功倍的效果。

作者的目标很明确：写一本实用的、有一定深度且易读的程序设计语言教材。无论读者有无编程经验，要做到开卷有益，重要的是能够激发他们的学习兴趣，能够将他们所学的知识应用于实际。

本书将基本技能培养与主流技术相结合，以市场对人才的需求为依据，从应用与工程实践的角度出发进行组织和撰写，其主要特点如下。

1. 新体系、新内容、新手段、新思路。无论是内容体系、编写的思路、教学的模式还是理念等都具有一定的新意。

2. 是基于工作过程的教学过程的设计。在职业能力分析的基础上，以工作任务为中心来整合相应的知识、技能，从应用与工程实践的角度出发来设计课程体系、课程内容和教学模式。教学项目不是对企业项目简单地再现与复制，而是源于企业并高于企业，是在真实性基础上构建学习性的环境。

3. 具有先进性及实用性。本书的内容反映出最新的实用程序设计方法及技术，顺应并符合新世纪教学发展的规律，书中讲授的程序设计方法与现代编程方法步调一致，具有很强的实用性。

4. 采用以问题、模型为中心的教学方法和面向实际的应用程序驱动的教学方式，并追求理论和实践一体化。本书完全通过示例(任务)来学习。书中通过大量的完整示例来介绍相关的主题。本书不是采用传统的“提出概念→解释概念→举例说明”的方法，而是以实践为主线，以应用为目标，通过完成实际应用程序的方式来学习程序设计知识。整个学习是由许多小的教学模块和任务组成。每个教学模块和任务首先提出一个来自实际的问题，然后学生可“尝试”解决该问题的应用程序并了解其工作过程，接着教师介绍该问题的解决方案及方法和步骤，然后归纳出一般的规律、概念及知识点。随后学生参照教师的解决方案及方法和步骤来解决该问题，最后教师对学生提出一个新的实际问题，学生应用所学的方法和步骤来解决该问题。

学生通过这样一种“实践→学习→实践→提高”的过程，不仅可以更快、更深入地理解和掌握课程的内容，而且可以提高自己的实践能力和独立解决实际问题的能力。很好地贯彻了在实践中学习、在学习中实践、理论和实践教学一体化的思想。

针对每一个新概念、知识点的提出，都伴随着一个完整的、可实际运行的实用程序及其输入、输出。学生通过完成实际应用程序，来掌握概念和知识点。

5. 以独特的体系结构、设计编排和内容组织模式，将复杂的技术体系以简洁易懂的方式呈现给学生。教学内容在设计时贯穿了一个思想：面向实际；同时使学生容易理解、明白所包含的全部细节。

因此本书在安排时，循序渐进、由浅入深、综合应用。后续章节在讲授新知识点的同时综合应用了前面章节所学的知识。这样做有利于温故知新，既巩固了前面所学的知识，又懂得了如何综合应用已学的知识来解决问题。书中很多实际应用程序示例不只是讲授一个新的知识点，而是在讲授新知识点的同时，应用了前面所学过的知识。

6. 使学生从一开始就编写有用的程序，这有助于保持学生的学习兴趣及动力。书中的示例不是只有几行代码的小程序，这些示例来源于实际应用程序或其中的一部分，它们与我们的生活相关，或是大家都感兴趣的内容。本书包含大量这样的示例，这些示例及其代码分析与讨论是本书的精髓。学生可先按照所描述的方法和步骤编写代码，并运行自己

所创建的应用程序,“尝试”该应用程序并了解其工作过程。然后通过其代码分析和讨论来掌握其包含的概念、原理和方法。

7. 逐步展现主题鲜明的各章。本书将集中重要的主题并进行充分的论述,而不是肤浅地涉及许多问题,罗列许多概念、术语、语法。本书的内容及体系结构使学生感到它既具有易读性又能增进知识,具有很强的实用性。

8. 讲述面向实际的技术。学习程序设计语言的目的是为了开发程序,本书不只是讲授 C#的语言要素、语法,而是教会读者如何用 C#开发程序,书中的每一个面向实际的应用程序示例都说明了开发过程及包含的知识点。通过学习如何开发程序来掌握语言要素和语法。在第 8 章,通过一个实际应用程序案例,讲授三层架构的应用程序设计方法和开发过程,三层架构应用程序设计是企业应用程序常用的程序设计方法。后续的许多示例都采用三层架构设计方式,通过这种方式,使学生能够熟练地掌握企业常用的程序设计方法和开发过程。

9. 传授面向对象的程序设计技能。本书由始至终贯穿着面向对象编程的思想,目标之一是使学生习惯于现实世界的程序设计,仅仅知道面向对象的概念是不够的,学生们必须能够运用这些知识开发现实世界的程序。教学的核心概念之一是在成为一名对象设计者之前,必须首先成为一名对象用户。换句话说,在有效地设计自己的类之前,必须首先学会使用预定义的类。书中从第一个程序开始就接触如何使用类。学习使用类库中的类与编写自己的类二者相辅相成、互相促进。学习使用类库中的类有助于学习编写自己定义的类,学习自定义类又有助于学会使用类库中的一些类,并加深对类库中的一些类的理解。当今学生必须同时掌握基础语言、面向对象编程和类库,我们力争将这三者结合在一起。

任何一本真正讲授面向对象方法的教材都必须从对象出发,即所有处理过程都需要依照面向对象的术语来进行讨论。然而,这并不意味着学生看到的第一个程序就涉及多个类及方法。在掌握类和方法的编写之前,应教会学生如何使用它们。本书采用的是一个自然推进的教学方式,目的是最终能够让学生在实践中设计出面向对象的问题解决方案。

10. 重视 GUI 及控件。带有图形用户界面的程序总能激发学生的学习欲望,以这类程序作为讲授面向对象概念的示例会取得很不错的效果。因此,在本书中,示例采用图形用户界面,并介绍这方面的相关知识,包括事件的处理与 GUI 的使用等。通过对这些内容的不断学习,学生们将逐步掌握 GUI 的创建技能。

11. 重视结合数据库。数据库应用程序如今对所有企业来说是至关重要的,因此本书用两章的篇幅通过具体的实际应用程序案例来讨论使用 ADO.NET 和 SQL 访问数据库的编程技术,并应用三层架构的应用程序设计方法和开发过程。

针对 GUI 及控件概念的提出,我们采用的是一个循序渐进的过程。在讨论 GUI 及控件之前,学生已对事件的处理与 GUI 的使用有了一个基本的认识。因此在学习 GUI 及控件一章时,我们引入了访问数据库的一些关键性要素。通过创建数据库应用程序来学习 GUI 及控件。这里采用的是自然推进的教学方式,学生在学习这部分内容时没有感觉困难,而是充满了成就感,激发了学生的学习欲望。

12. 带给学生真实的编程体验。本书汲取了很多来自实际编程中的经验,这将为培养学生良好的编程习惯打下一个坚实的基础。这些经验会贯穿于所有示例中,并在探讨中得到进一步强调,使学生最终学会如何解决实际问题。在本书中,还将引入并吸收许多来自

软件工程方面的基础知识。

我们建议读者遵循书中介绍的方法和步骤实际建立案例程序，然后对案例程序进行修改或扩展，并通过对案例代码进行分析和讨论以掌握案例背后所包含的概念、原理、知识点和方法等，这是学习程序设计最稳妥、最有效、最快捷的途径。

通过本书的学习，读者应达到下列目标：

- 掌握面向对象的程序设计。
- 掌握 Windows 应用程序设计。
- 掌握 ADO.NET 及数据库应用程序设计。
- 学会文件的输入输出。
- 熟练掌握以上内容的综合应用。

尽管作者以细致入微的创作态度来努力完善本书，但错误之处仍在所难免，欢迎读者提出宝贵意见(pmsbao@163.com)。本书的源代码可在 <http://61.144.43.233/CSharp> 站点下载。如果我们发现书中的内容和代码有不当之处，我们也会在该站点发布勘误信息。要想与本书的作者进行交流，可加入论坛 <http://61.144.43.233/CSharp/form/default.aspx>。

邵鹏鸣

2008 年 3 月 广州

第 1 章 认识 C#	1
1.1 什么是.NET	1
1.1.1 从用户代码到机器代码	2
1.1.2 中间语言介绍	3
1.2 第一个简单的控制台应用程序	4
任务 1.1: 打印一行文字	4
1.3 简单的 Windows 应用程序	6
任务 1.2: 在对话框中显示一行文字	6
任务 1.3: 在文本框中显示一行文字	7
1.4 习题	12
第 2 章 C#编程基础	13
2.1 变量与常数	13
2.1.1 变量的含意	13
2.1.2 变量声明	15
任务 2.1 计算路程	15
2.1.3 常数	18
2.2 基本数据类型	19
2.2.1 整型	20
任务 2.2: 整数相乘	20
2.2.2 字符数据类型	24
任务 2.3: 字符检查	24
2.2.3 非整型	26
任务 2.4: 贷款计算器	27
2.2.4 格式化输出	30
2.2.5 算术运算	31
2.2.6 基本数据类型的相互转换	32
任务 2.5: 隐式和强制数值转换	34
2.2.7 算术溢出及显式转换溢出	35
任务 2.6: 算术溢出及显式转换溢出	35
2.2.8 布尔类型	37
任务 2.7: 数值比较	38
2.3 面向对象程序设计初步	39

2.3.1 类和对象	39
2.3.2 消息和方法	39
2.3.3 使用现有的类	40
任务 2.8: 文字游戏	40
2.3.4 创建自己的类	44
任务 2.9: 使用可实例化类的 贷款计算器	44
2.4 习题	53
第 3 章 程序流控制	55
3.1 选择语句	55
3.1.1 if 语句	55
任务 3.1: 考试结果分析	56
3.1.2 if ... else 语句	58
任务 3.2: 猜数游戏	59
3.1.3 条件运算符	60
任务 3.3: 显示时间	61
3.1.4 if ... else if ... else 语句	63
任务 3.4: 工资发放	63
3.1.5 if 语句的嵌套	65
任务 3.5: 求数的绝对值	65
3.1.6 switch 语句	67
任务 3.6: 计算器	68
3.1.7 复合赋值运算符	71
3.1.8 条件逻辑运算符 和逻辑运算符	71
3.2 循环语句	72
3.2.1 while 语句	72
任务 3.7: 计算复利存款(一)	72
3.2.2 do ... while 语句	75
任务 3.8: 计算复利存款(二)	75
3.2.3 for 语句	76
任务 3.9: 打印字母表及对应的 ASCII 码(一)	77
3.2.4 嵌套循环	79

任务 3.10: 打印字母表及对应的 ASCII 码(二).....	79	任务 4.8: 调用方法获得多个值 (使用 ref 参数).....	117
3.3 跳转语句	80	任务 4.9: 调用方法获得多个值 (使用 out 参数).....	119
3.3.1 goto 语句	80	任务 4.10: 改变矩形位置	120
任务 3.11: 找数	81	4.4.6 以传引用方式传递引用 类型参数.....	122
任务 3.12: 自动售货机.....	82	任务 4.11: 调用方法为实参创建 新的对象.....	122
3.3.2 break 语句.....	83	4.5 习题	125
任务 3.13: 打印字母表及对应的 ASCII 码(三).....	83	第 5 章 基于对象的程序设计	127
3.3.3 continue 语句.....	84	5.1 类、对象和封装.....	127
任务 3.14: 找数	84	5.2 字段	128
3.3.4 运算符的优先级.....	85	任务 5.1: 改写贷款计算器.....	129
3.4 习题	86	5.2.1 常数和只读字段.....	131
第 4 章 数组与方法	88	5.2.2 成员访问控制.....	132
4.1 数组	88	5.3 属性	133
任务 4.1: 计算月平均降雨量.....	89	任务 5.2: 声明和使用属性.....	133
4.1.1 数组初始化	94	5.3.1 类作用域.....	138
任务 4.2: 显示月名称.....	94	5.3.2 索引器.....	138
4.1.2 可变长度的数组的声明.....	95	任务 5.3: 贷款分析.....	139
任务 4.3: 创建数组.....	96	5.3.3 属性与字段、属性 和方法的比较.....	144
4.2 多维数组	98	5.3.4 使用 this 关键字	145
4.2.1 多维数组的声明和创建.....	98	任务 5.4: 雇员税金计算.....	145
4.2.2 多维数组初始化.....	99	5.4 实例构造函数.....	147
4.2.3 二维数组应用举例.....	99	5.4.1 默认实例构造函数.....	147
任务 4.4: 二维数组.....	99	任务 5.5: 定义 Person 类.....	147
任务 4.5: 学生考试成绩统计.....	101	5.4.2 默认初始化字段.....	149
4.3 值类型与引用类型.....	102	5.4.3 显式初始化字段.....	149
4.4 方法	105	5.4.4 实例构造函数声明.....	150
4.4.1 传值方式	106	任务 5.6: 定义矩形(一).....	150
4.4.2 以传值方式传递值 类型参数	106	5.5 实例构造函数重载.....	154
任务 4.6: 移动矩形.....	106	5.5.1 使用重载的实例 构造函数.....	154
4.4.3 以传值方式传递引用 类型参数	111	任务 5.7: 定义矩形(二).....	154
任务 4.7: 以传值方式传递数组.....	112	5.5.2 调用同类中的 其他构造函数.....	156
4.4.4 传引用方式	116		
4.4.5 以传引用方式传递值 类型参数	116		

任务 5.8: 调用其他构造函数.....	156	任务 6.7: 定义派生类 Student(二) ...	190
5.6 静态成员与实例成员.....	157	6.5 虚拟方法与重写方法.....	191
5.6.1 静态字段和实例字段.....	157	任务 6.8: 多级继承层次结构.....	192
任务 5.9: 自动编号.....	157	6.5.1 继承中构造函数的 执行过程.....	201
5.6.2 静态构造函数.....	159	6.5.2 重载、重写和隐藏的比较.....	201
任务 5.10: 自动编号从随机整数 开始.....	159	6.5.3 垃圾回收和析构函数.....	202
5.6.3 静态方法.....	160	任务 6.9: 保存状态信息.....	203
5.6.4 静态成员和实例 成员的特征.....	160	任务 6.10: 保存雇员状态信息.....	204
5.7 以对象为参数与以返回值为对象.....	162	6.6 习题.....	205
5.7.1 以对象为参数.....	162	第 7 章 多态性.....	206
任务 5.11: 放大矩形.....	162	7.1 抽象方法与抽象类.....	206
任务 5.12: 以传引用方式传递 对象参数.....	164	7.1.1 抽象方法和抽象属性.....	207
5.7.2 以返回值为对象.....	165	任务 7.1: 多态性及实现(一).....	207
任务 5.13: 放大矩形的副本.....	165	7.1.2 抽象类继承.....	212
5.8 方法的重载.....	166	任务 7.2: 多态性及实现(二).....	213
任务 5.14: 定义矩形(三).....	167	7.2 接口.....	218
5.9 使用 ArrayList 类.....	169	7.2.1 声明和实现接口.....	218
任务 5.15: 地址簿.....	169	任务 7.3: 创建和使用接口.....	218
5.9.1 ArrayList 类的常用属性.....	175	7.2.2 接口成员的显式实现.....	222
5.9.2 ArrayList 类的常用方法.....	175	任务 7.4: 用接口实现不同的 度量衡系统.....	223
5.10 习题.....	176	7.2.3 接口和抽象类.....	224
第 6 章 继承.....	178	任务 7.5: 薪水发放系统.....	224
6.1 直接基类与派生类.....	179	7.2.4 接口与抽象类的比较.....	230
任务 6.1: 定义基类 Person.....	179	7.3 代理.....	231
任务 6.2: 定义派生类 Student(一).....	180	7.3.1 使用代理.....	231
6.2 派生类实例构造函数声明.....	182	任务 7.6: 使用代理实现运算.....	231
任务 6.3: 复数加法.....	182	7.3.2 组合代理.....	234
6.2.1 运算符重载.....	183	任务 7.7: 使用组合代理实现运算... ..	234
任务 6.4: 复数减法.....	184	7.3.3 代理应用举例.....	236
6.2.2 显式调用直接基类实例 构造函数.....	188	任务 7.8: 用代理排序数组.....	236
任务 6.5: 调用基类实例构造函数.....	188	7.4 事件.....	240
6.3 隐藏从基类继承的成员.....	189	7.4.1 自定义事件.....	240
任务 6.6: 隐藏继承字段.....	189	任务 7.9: 密码监听器.....	240
6.4 含直接基类构造函数的 构造函数声明.....	190	7.4.2 声明持有事件数据的类.....	244
		任务 7.10: 能够获取新旧密码的 密码监听器.....	244

7.5 习题	245	8.8.2 列表框控件的常用事件	297
第8章 控件及 GUI 程序设计	247	8.8.3 TextBox 的常用属性	297
8.1 滚动条	247	8.8.4 TextBox 的常用事件	299
任务 8.1: 调色板	248	8.9 列表框	300
8.1.1 滚动条常用属性	250	任务 8.10: 产品信息管理	300
8.1.2 滚动条常用事件	251	8.9.1 列表框控件的常用属性	315
8.1.3 用户定义的颜色	252	8.9.2 列表框控件的常用方法	316
8.2 Windows 窗体事件及事件		8.9.3 列表框控件的常用事件	318
处理程序	252	8.10 带复选框的列表框	318
任务 8.2: 计算器	253	任务 8.11: CheckedListBoxTest	
8.2.1 Button 按钮的常用属性	257	程序	318
8.2.2 Button 按钮的常用事件	258	8.10.1 复选列表框控件的	
8.3 复选框和单选按钮	258	常用属性	321
任务 8.3: Font 程序	258	8.10.2 复选列表框控件的	
8.3.1 如何设置字体	260	常用方法和事件	321
8.3.2 复选框的常用属性	260	8.11 习题	321
8.3.3 复选框的常用事件	261	第9章 使用 ADO.NET 进行	
8.3.4 单选按钮的常用属性	261	数据库编程	324
8.3.5 单选按钮的常用事件	262	9.1 数据表 DataTable	325
8.4 Connection 对象和 Command 对象	262	任务 9.1: 客户信息管理	326
8.4.1 Connection 对象	262	9.2 数据集和数据适配器	336
任务 8.4: 创建并打开一个到		9.2.1 通过数据集更新数据源	336
SQL Server 的连接	262	任务 9.2: 客户信息管理	337
8.4.2 Command 对象	264	9.2.2 将数据集绑定到	
任务 8.5: 操作数据库	264	DataGridView 控件	351
8.5 使用 SqlDataReader	266	任务 9.3: 以浏览方式管理	
任务 8.6: 雇员信息管理	266	客户信息	351
8.6 创建三层应用程序	278	9.2.3 行状态与行版本	353
任务 8.7: 增强雇员信息管理程序	279	9.3 Windows 窗体中的数据绑定	354
8.7 PictureBox 图片框控件	285	9.3.1 简单绑定控件属性	355
任务 8.8: 雇员相片管理	285	任务 9.4: 类别信息管理(一)	355
8.7.1 PictureBox 的常用属性	288	9.3.2 使用 CurrencyManager	361
8.7.2 PictureBox 的常用事件	289	9.4 创建和使用数据视图	362
8.7.3 Image.FromFile 方法	289	9.4.1 使用数据视图查询数据	362
8.8 组合框控件和文本框控件	289	任务 9.5: 类别信息管理(二)	362
任务 8.9: 登录程序	290	9.4.2 数据视图的常用属性	
8.8.1 ComboBox 的常用属性		及方法	366
和方法	294		

9.5 DataRelation 对象	368	10.6 习题	407
9.5.1 创建 DataRelation 对象	369	第 11 章 用流进行文件输入和输出	408
任务 9.6: 类别信息管理(三)	369	11.1 文件与流	408
9.5.2 导航表间的关系	373	11.1.1 FileStream	409
9.6 习题	373	任务 11.1: 使用 FileStream	409
第 10 章 与用户交互	375	任务 11.2: 将图像文件存入 数据库	411
10.1 菜单	375	任务 11.3: 从数据库读取 Image 类型数据并保存为 图像文件	413
10.1.1 创建菜单	375	11.1.2 随机访问文件	414
任务 10.1: 随机画矩形	376	任务 11.4: 定位操作	415
10.1.2 Timer 控件	379	11.1.3 向文件追加数据	416
10.1.3 MainMenu 控件的 常用属性	379	任务 11.5: 向文件追加数据	416
10.1.4 快捷菜单	380	11.2 内存和缓冲流	417
任务 10.2: 实现快捷菜单	380	任务 11.6: 显示图像	417
10.2 鼠标事件	381	11.3 StreamReader 和 StreamWriter	419
任务 10.3: 用鼠标画直线 和画矩形	381	任务 11.7: 电话号码簿	419
10.2.1 鼠标事件	385	11.4 二进制读取器和写出器	422
10.2.2 鼠标事件发生顺序	386	任务 11.8: 学生名册	423
10.3 键盘事件处理	386	11.5 序列化对象	426
任务 10.4: 键盘事件程序	387	任务 11.9: 序列化对象	426
10.4 通用对话框	390	11.6 可视化继承	428
10.4.1 “打开文件”对话框	390	任务 11.10: 创建基窗体	429
任务 10.5: 打开文件	391	任务 11.11: 创建可视继承窗体	430
10.4.2 “保存文件”对话框	394	11.7 顺序访问文件	431
任务 10.6: 保存文件	394	任务 11.12: 产品类别管理程序	432
10.4.3 “字体”对话框	395	11.8 随机存取文件	444
任务 10.7: 改变文本的字体	395	任务 11.13: 产品供应商管理程序	444
10.4.4 “颜色”对话框	397	11.9 使用序列化对象存储数据	453
任务 10.8: 改变文本颜色	397	任务 11.14: 产品管理程序	454
10.5 编写多文档界面应用程序	398	11.10 习题	465
任务 10.9: 字处理器	398		

第1章 认识 C#

主要内容:

- (1) 什么是.NET。
- (2) 第一个简单的控制台应用程序。
- (3) 创建简单的 Windows 应用程序。
- (4) 对象、属性和方法, 给对象的属性赋值。
- (5) Click 事件及 Click 事件处理程序。

学习目标:

- (1) 熟悉 Visual Studio.NET 集成开发环境(IDE)。
- (2) 学会创建、编译和执行简单的.NET 应用程序。
- (3) 使用输入和输出。
- (4) 初步认识和了解窗体、控件、事件和方法。

1.1 什么是.NET

.NET 是微软公司的新战略, 它包含微软公司对未来的核心战略、规划和洞察力。所有微软公司的产品都将围绕这个战略开发。此战略的核心就是 .NET Framework, 该框架提供了全面支持 .NET 的核心技术。 .NET Framework 是一种新的计算平台, 它简化了在高度分布式 Internet 环境中的应用程序开发。 .NET Framework 具有两个主要组件: 公共语言运行库和 .NET Framework 类库。

.NET 用来解决编程人员面临的许多问题:

- 它负责处理在创建大型、可靠的应用程序时的大量艰辛工作。
- 它允许程序员统一两种架构——在本地机器上运行的应用程序和通过 Web 访问的应用程序。
- 它减少了与编程框架相关的传统开销——不再需要高性能编程语言来编写复杂的代码以获取高速的 .NET 程序。
- 它允许不同语言的程序员在同一个应用程序中协同工作。
- 它开始兼容各种最终用户工具。包括桌面、PDA 和手机。最终将实现使开发人员能够创建出摆脱设备硬件束缚的、能够在各种操作系统上运行的应用程序, 能够轻松地实现互联网的连接。

总之, .NET 提供了一种更简单、更快捷、更廉价的方式来获得高效的程序。

在某些方面, .NET 很像 Java。实际上, Java 的口号“一旦编写出来, 就能在任何地方运行”也很符合 .NET 的原则。不过 .NET 并不是 Java 的克隆, 微软的方法与之并不相同。

Java 是“一种语言, 多个平台”, 而 .NET 是“多种语言, 一个平台……(就目前而言)”, 由于微软想扫除进入 .NET 的障碍, 因此计划让使用任何语言的人都可对它进行访问。目前

Visual Studio.NET 提供了 C#和 Visual Basic.NET 这两种.NET 下的编程语言。

微软的.NET 策略更像是一场军事战役。它运用对 Windows 平台的理解,首先攻占 Windows 平台。当它从 Windows 平台中脱颖而出后,就可以“侵袭”别的平台。例如 Linux,证明.NET 应用程序可以从一个平台移植到另一个平台上。“侵袭并征服了 Linux”后,它又可转向另外的平台,例如 Solaris。

简而言之,.NET 将操作系统平台划分为层次结构。无无论是哪种平台,例如 Windows、Linux 等,都可以分成两个层次:程序设计层和执行层,.NET 开发人员是为程序设计层而不是执行层编写软件,将来不管占统治地位的是 Windows 平台还是 Linux 平台,或者是其他出乎意料的系统,都将遵循这种统一一致的模式。

可以将.NET Framework 的讨论分解成以下几部分。

- **MS 中间语言(MS Intermediate Language, MSIL):** 在执行用户编写的所有程序代码前,应将其编译成更抽象、精简的形式。程序员可以使用任何.NET 语言编写代码,包括 VB、C#、JScript 和其他大约 20 种语言。这些代码都会编译为 MSIL 这种.NET 的通用语言。.NET 在这个层次的操作无须用户干涉,因此本书不讨论它。
- **公共语言运行库(Common Language Runtime, CLR):** 这是一个复杂的系统,用于在计算机上执行 MSIL 代码。公共语言运行库负责执行与 Windows 和 IIS 交互时所涉及的全部实质性任务。这个层次也是在后台进行的,本书也不涉及。
- **.NET Framework 类库(.NET Framework Class Libraries):** 实现大量重要功能的代码库。用户可以非常方便地在应用程序中调用库函数,使复杂任务的程序代码更为简洁。本书将介绍这些功能。
- **.NET 语言(.NET Language):** .NET 语言是符合一些特殊结构要求(由公共语言规范定义)的编程语言,能够编译成 MSIL。可以用任一种语言进行开发,例如 C#或 VB.NET,这没有任何限制,也可以用多种语言来开发程序。
- **ASP.NET:** 这个代码模块扩展了 Internet Information Server(IIS),使之能为 Web 页面执行.NET Framework。
- **Web 服务:** 虽然 Web 服务不是严格意义上的.NET 的一部分,但.NET 明确地支持它。它们是可以通过 Web 访问的组件,可以是任何主题,例如新闻标题、天气预报、股票走势、病毒预防和操作系统更新等。

1.1.1 从用户代码到机器代码

读者也许已经知道,计算机能够识别以二进制形式表示的任何事情(二进制位可以构成表示指令和数据的 1 与 0 的序列)。因此,人们热衷于用“数字”一词来描述任何事情,即使它们与计算机的关系很小。通常把这种二进制指令称为机器码。很明显,对大多数人来说,记住用于打印“Good Morning”时的 0、1 代码序列是不可能的,更不用说要记住定义复杂 Web 应用程序的程序代码了。因此,人们使用用户级编程语言(相对来说比较高级的语言),通过类似英语的单词来编写代码。

一旦用高级语言编写了程序代码,就需要将其转换为机器码,这一转换过程称为编译,编译器软件可以将人类可读指令编译成机器可读指令。编译过程包括将本机环境信息写入经编译的程序代码这一过程,因此编译后的程序代码可以最高效地利用计算机的所有可用资源。

多年来,有以下两种编译代码类型,它们的编译过程完全不同。

- 预编译型代码:是在编码完成后并在运行前进行编译的代码。对这种代码,由于编译器会花时间分析全部的代码和要运行它的机器,因此代码能以很快的速度执行。不过,由于预编译的代码是针对某一计算机的,因此,除非仍在该计算机上使用编译过的代码,否则需要将另一台计算机设成与执行编译的计算机相同的系统和资源。
- 解释型代码:是边执行(在用户请求页面时)、边编译的代码。这种代码执行较慢,因为需要为每个请求编译代码,且系统没有机会全面优化所编写的代码。不过,其优点是解释过程可以调整,以适应运行代码的机器。

所以,开发人员在选择编程语言时要做出取舍。可以选择较慢的解释型代码,以适应运行代码的机器;也可以选择运行较快速的预编译代码,但不能充分利用机器的优势。

1.1.2 中间语言介绍

.NET 在编译时采用两步来解决代码与机器的交互问题。编写在 .NET Framework 上运行的程序后(通常用 C# 或 VB.NET 编写),需要在使用这些程序之前编译这些可读代码。.NET 编译器的设计方式意味着它并没有把我们带入会引起可移植性问题的二进制码。事实上,.NET 编译器将程序代码编译成称为 MS 中间语言(MSIL)的特殊格式。由于 MSIL 结构不需要像源代码那样易读,因此编译过程包括了一些优化操作。但是,并没有针对某台机器进行优化。因此,MSIL 具有一般性的优化性能,并可以移植到任何 .NET 服务器上。

当用户执行 MSIL 代码时(例如请求 ASP.NET 页面),将其传给 CLR(公共语言运行库),CLR 是 .NET Framework 的另一个核心。CLR 使用 JIT(Just-In-Time)编译器将代码编译成真正的机器码,并对程序进行最后的且与机器相匹配的优化,以使程序能在其所在的计算机上以尽可能快的速度运行。

通过 MSIL 和 CLR 组合使用,带来了前述两种编译代码的综合性优点,即获得了预编译码的结构优化特性和解释码的可移植性。

更为重要的是,MSIL 本身是与机器无关的。因此,可以在装有 CLR 的任何一台计算机上运行。实际上,一旦编写出 .NET 程序代码并将其编译,就可以将它复制到装有 CLR 的任何计算机,并在该计算机上执行。虽然目前的 CLR 只有与 Windows(9x、NT、2000、XP 版)兼容的版本,但现在已着手建立基于其他操作系统的版本。在 Web 上搜索与 Mono Project 有关的信息,就可以找到更多的 CLR 版本。

MSIL 可以由任何遵循 CLS(公共语言规范)的可读语言生成。C#、VB.NET 和 JScript.NET 是“与 .NET 兼容的”三大语言。另外 MSIL 编译器还支持其他 20 多种语言。因此,可以

在应用程序内部交替地使用这些兼容语言。一旦将一套文件编译成 MSIL，它们都将统一为一种语言。这种灵活性允许不同的小组在同一个 Web 站点上用不同的语言协同工作。

1.2 第一个简单的控制台应用程序

任务 1.1: 打印一行文字

主要知识点 ①C#程序的基本结构。②控制台的输入和输出。

问题描述 在命令窗口中输出一行文字“Hello World”，如图 1.1 所示。

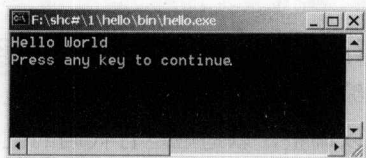


图 1.1 打印一行文字

解决方案

(1) 启动 Visual Studio。

(2) 从【文件】菜单中选择【新建】|【项目】命令，打开【新建项目】对话框。

(3) 在【项目类型】窗格中选择【Visual C#项目】，然后在【模板】窗格中选择【空项目】(Visual Studio 2005 中可选择【控制台应用程序】以直接获得代码文件)。

(4) 在【名称】文本框中，键入“Hello”作为该项目的名称。在【位置】文本框中，输入要将项目保存到的目录，或单击【浏览】按钮以选择目录。

(5) 单击【确定】按钮。Visual Studio 将创建一个新项目“Hello”，并显示解决方案资源管理器。若没有显示解决方案资源管理器，请选择【视图】|【解决方案资源管理器】，或按 Ctrl+Alt+L 键，解决方案资源管理器即可出现。

(6) 在解决方案资源管理器中，右击解决方案下的 Hello 项目，在弹出的快捷菜单中选择【添加】|【添加新项】命令。

(7) 在【类别】窗格中选择【本地项目】，然后在【模板】窗格中选择【代码文件】。在【名称】文本框中，键入“Hello”。

(8) 单击【打开】按钮。然后在空的 Hello 代码文件中输入如下代码：

```
//第一个简单的 C#控制台应用程序
class Hello
{
    static void Main()
    {
        System.Console.WriteLine("Hello World"); //主要代码
    }
}
```

(9) 在解决方案资源管理器中，用鼠标选择解决方案下的 Hello 项目，单击鼠标右键，然后在弹出的快捷菜单中选择【属性】命令。

(10) 在输出类型下，单击向下箭头，选择【控制台应用程序】，单击【确定】按钮。