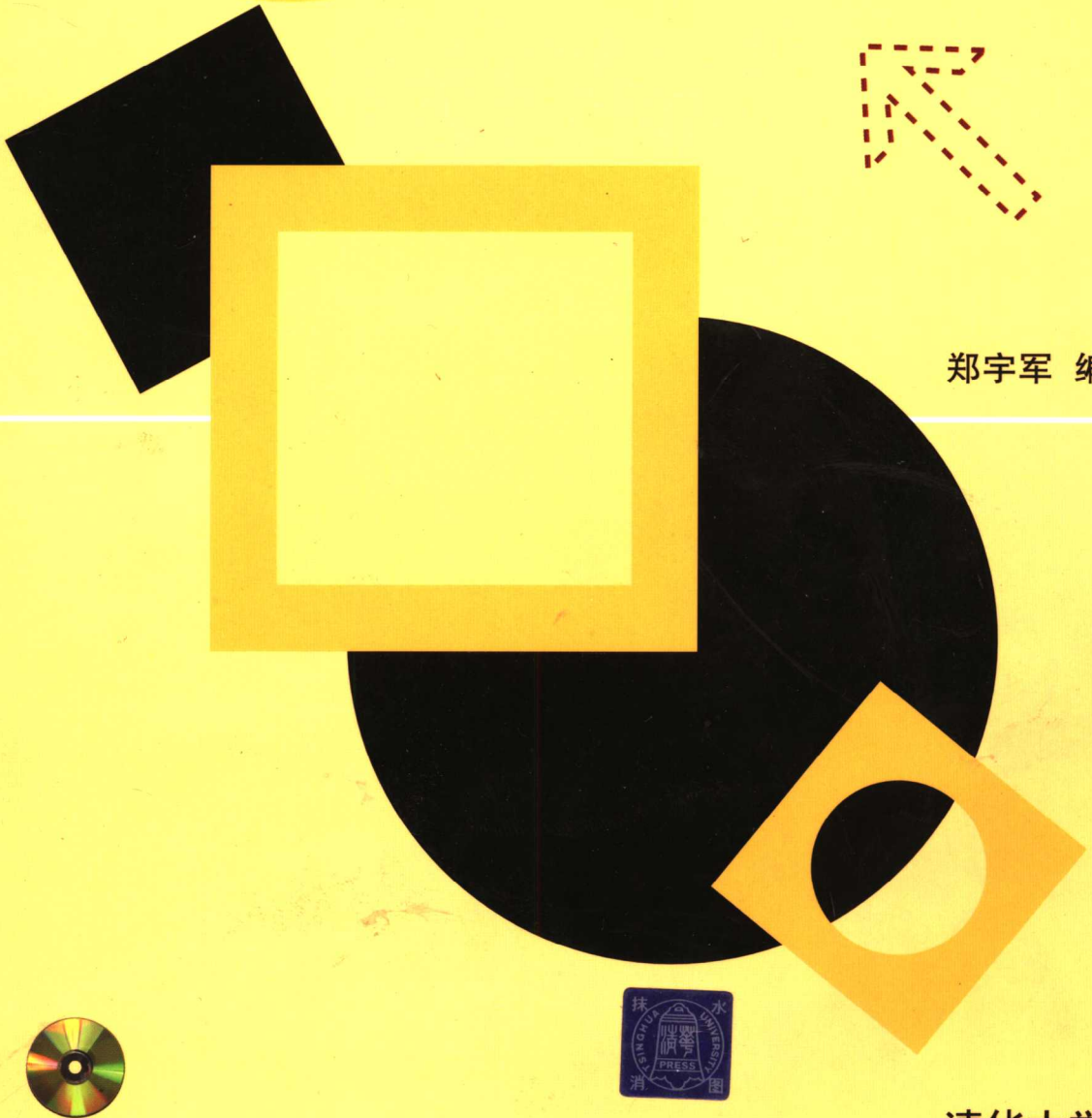
计算机与信息技术专业应用教材

# C# 2.0 程序设计教程



郑宇军 编著

清华大学出版社

# C# 2.0 程序设计教程

郑宇军　编著

清 华 大 学 出 版 社

北　京

## 内 容 简 介

C# 2.0 是第一门真正将泛型思想和对象技术进行完美融合的高级语言。

本书通过丰富的范例全面系统地讲解了C#语言的编程技术，并重点介绍了C# 2.0的新增特性，其中包括.NET框架、C#语法基础、类型系统、对象程序设计、泛型编程、匿名方法及商业开发等多个层面的内容。每章之后给出了丰富的练习题，为读者进一步巩固和拓展所学知识提供了广阔空间。

本书面向C#语言的初中级读者，可作为大中专院校及各类培训机构的程序设计语言教材，对于专业开发人员而言也不失为一本理想的参考书。



**图书在版编目（CIP）数据**

C# 2.0 程序设计教程/郑宇军编著.—北京：清华大学出版社，2004.11

ISBN 7-302-10153-1

Ⅰ.C…　Ⅱ.郑…　Ⅲ.C语言—程序设计

Ⅳ.TP312

中国版本图书馆CIP数据核字（2004）第135503号

**出 版 者**：清华大学出版社　　**地　　址**：北京清华大学学研大厦

http://www.tup.com.cn　　**邮　　编**：100084

社总机：010-62770175　　**客户服务**：010-62776969

**组稿编辑**：科海

**文稿编辑**：何武

**封面设计**：付剑飞

**版式设计**：科海

**印 刷 者**：北京市艺辉印刷有限公司

**发 行 者**：新华书店总店北京发行所

**开　　本**：787×1092　1/16　　**印张**：26.25　　**字数**：639千字

**版　　次**：2005年1月第1版　　2005年1月第1次印刷

**书　　号**：ISBN 7-302-10153-1/TP・1066

**印　　数**：1~4000

**定　　价**：39.00元（1CD）

# 前　言

易则易知，简则易从，易知则有亲，易从则有功。

——《易经·系辞》

软件工程的研究目标，就是如何提高软件系统的质量和开发效率。作为软件工程技术中的重要一环，程序设计语言的每一次进步，无不是大师们才华和创意的结晶。他们当中有不少人一直坚信：终有一天，程序员这个职业将不复存在，工程师、科学家以及商业管理者可以轻松地构建自己所需的软件系统。

如果这一天真正到来的话，人们回顾程序设计语言的发展史，C#必将被留下浓墨重彩的一笔。样板式的设计、简洁的语法、完整的结构、标准化的接口，这一切使得C#更像是程序设计语言中的一件艺术品。无数非专业人员迅速地学习并掌握了这门语言，编程的门槛又降低了。

然而，作为一件新生事物，C#语言不可避免地受到了诸多怀疑和批评。寄生在.NET框架之上，而且相比起C++来底层控制能力大幅降低，这使得许多有经验的开发人员不愿意去接受它。不少专家早早地下了定论：C#不可能像汇编、Fortran、Pascal或C和C++那样成为程序设计语言发展历程上的一个里程碑。

但C#从正式推出到现在不过两年多的时间，而C++从面世到成为面向对象技术的集大成者走过了十余年。C#自我完善的脚步之快出乎所有人的意料：随着C# 2.0的出现，C# 语言已经完成了一次重大升级。然而更出乎人们意料的是：面对质疑，C# 2.0并没有去增加底层控制能力，也没有消除对.NET框架的依赖；所改进的重点几乎都集中在C# 语言最初的强项上，带来的则是更灵活、更简便的开发方式。

这种看似孤军冒进的做法仔细想来却在情理之中。C++现在几乎已经成了面向对象技术的代名词，初出茅庐的C#，如果要在方方面面都去和它争一日之长短，结果恐怕是自取其辱。既然定位为商业应用开发语言，不入硬件开发人员以及黑客们的法眼也就罢了。C#语言中的对象框架原本就是一个简洁而完整的精心设计，而此次C# 2.0在对象和泛型这两个编程概念的完美结合上已经超越了C++。也许用不了多久，C# 就将在许多大学计算机专业的高级语言课程上取代C++。而要深究C++为何在STL技术上停滞不前，原因可能就是因为注重的方面太多、实现的功能太细太杂，以至于泥塘深陷。有了前车之鉴，加之关于多继承、指针等的优劣性尚在争论之中，C# 当然不会去舍长取短了。

至于对.NET框架的依赖，这对C# 来说不得不说是一种冒险。和成熟的对象技术相比，组件技术特别是分布式组件技术正在快速发展当中。在今后相当长的一段时间内，从COM泥塘中爬出来的.NET技术与CORBA、EJB必将会有一场激烈的比拼，也不排除会有新的组件技术和标准参与到角逐中来。C# 把组件设计与交互的重担卸给.NET，无疑是为了轻装前进，在语言本身上精益求精，吸引更多追求简洁和高效的开发人员，特别是非专业的开

发人员。即便有一天.NET技术不幸寿终正寝，C# 未必就会一同殉葬，只不过难免要经历一场脱胎换骨的阵痛；考虑到微软及其Windows平台的统治力，也许有朝一日.NET将成为组件技术的集大成者，那么作为.NET母语的C#，必然也会成为网络服务开发技术的代名词。

## 本书的读者对象

本书可作为计算机专业师生学习C# 语言的教材或辅导书，也可供软件从业人员以及程序设计爱好者自学参考。已有一定经验的C# 程序员则可以利用本书来学习C# 2.0中的新增特性。

## 本书的内容组织

全书采用循序渐进的方式，逐步引导读者全面而系统地掌握C# 语言的语法规则和基本特性。书中前两章简要地介绍了.NET技术和C# 语言的概貌，帮助读者尽快入门。第3章到第6章的内容是C# 的语法基础。第7章到第9章介绍了如何使用C# 进行面向对象的程序设计。第10章到第15章讲解的是C# 2.0的主要新增特性，包括泛型、可空类型、遍历器和匿名方法，有经验的C# 程序员可以只阅读这一部分。第16章到第18章涉及的是商业应用开发的基础知识，包括异常处理、文件操作、代码组织、帮助文档等内容。将本书作为教程使用时，可以将第16章以后的章节作为选修内容。

书中的每一章在结束时都进行了小结，对该章的关键概念进行了回顾，并指出了学习过程中的难点和重点。此外，各重点章节都配备了一定数量的习题，通过这些练习能帮助读者有的放矢地温习所学的内容；配套光盘中还提供了大多数习题的参考答案和一套完整的程序设计模拟试题。

对于面向对象技术的初学者来说，最困难的部分可以说是类的继承和多态性。而对于有一定基础的开发者来说，C# 语言中最不易掌握的可能当属代表（delegate）这个概念。而C# 2.0所新增的两个最主要的特性——泛型和匿名方法，又恰恰增加了这两方面内容的学习难度。因此第9章、第10章和第15章是整本书的难点所在。

## 源代码使用

本书通过大量精心设计、极具代表性的源程序来讲解C# 的各种特性，其中可实际运行的完整程序近150个，源代码总数超过18 000行。它们都可以在本书的配套光盘中找到。

编译这些源代码的一个必备工具是Microsoft .NET Framework 2.0，它可以从Microsoft的网站上免费获取。该工具针对32位处理器的链接地址是：

http://www.microsoft.com/downloads/details.aspx?familyid=b7adc595-717c-4ef7-817b-bde

fd6947019&displaylang=en，下载文件的大小约为24MB；

针对64位处理器的链接地址是：

http://www.microsoft.com/downloads/details.aspx?familyid=1061fd6f-86d6-4eb3-b70f-c42ed6f49ae8&displaylang=en，下载文件的大小约为39MB。

## 致　谢

清华大学的陈立航和郑艳华参与了部分章节的编写和程序调试工作。北京科海电子出版社陈跃琴老师对本书的策划和出版给予了热情支持，责任编辑何武付出了辛勤劳动，在此一并表示衷心的感谢。

## 意 见 反 馈

本书力求在第一时间将C# 2.0语言的全貌奉献给读者。由于时间仓促，加上水平所限，尽管我们作了最大努力，书中一定还存在许多不周到和不准确的地方，恳切希望读者提出批评和建议；也欢迎读者能够在.NET和C# 技术方面与我们做进一步的交流。我们的E-mail地址是：uchengz@yahoo.com.cn。

编　者

2004年10月

# 目　录

# 第 1 章　.NET 和 C#概述

本章将介绍.NET技术的由来和关键组成部分，并提供.NET的核心编程语言C#及其升级版本的基本信息。

## 1.1　Microsoft .NET技术的由来

Internet技术在20世纪60年代末期才开始起步，却成为人类历史上意义最为深刻的变革之一。随着网络应用需求的飞速增长，以资源共享和协同工作为主要特征的网络分布计算已经成为新一代计算和应用的主流。随着主机计算向网络分布计算的过渡，软件系统的规模和复杂度呈几何级数增加，传统的软件开发方法面临前所未有的挑战。

为突破时间、空间及不同平台的限制，提高软件系统的重用和集成能力，组件化软件开发技术应运而生。它将对象技术和组件技术有机地结合在一起，强调组件的可重用性和互操作性，力图通过组件集成来构建高效的分布式软件系统。20世纪90年代以来出现了3种典型的组件技术标准，它们分别是OMG（Object Management Group，对象管理组织）的CORBA、Microsoft的COM/DCOM以及Sun公司的JavaBeans。这些技术各有优劣，都面临着不断改进和发展的要求，谁也没有一统天下的实力。

2000年6月22日，Microsoft正式宣布了其新一代计算计划，即Microsoft .NET（读作“dot-net”），这是Microsoft针对第三代Internet推出的全新设计思想，其目标之一就是希望.NET能够取代COM，进而成为Windows应用和Web应用的主流开发模型。这可以说是Microsoft继使用Windows取代DOS操作系统之后又一项战略性的举措。从技术角度理解，.NET是一个全新的计算平台，它的主要特点有：

- 面向异构网络、硬件平台和操作系统，为软件提供最大限度的开放性、可扩展性、互操作性及可重用性。例如，它能够将个人电脑上运行的软件方便地移植到手机、掌上电脑等各种终端之上。
- 实现软件系统之间的智能交互和协同工作，提高整个网络的效率和利用率，特别是实现企业级的系统集成和资源优化，给开放型企业的生产力水平带来质的飞跃。其中的一个关键组件就是以XML和SOAP协议为基础的Web服务。
- 提供一个标准化的、安全的、一致的模型和环境，简化分布式应用程序的开发难度，从而大幅提高软件系统的质量和生产率。例如，.NET支持以不同编程语言开发的组件之间进行无缝的交互和集成。

.NET在理念中包含了对操作系统和计算机网络设计思想的延伸，即将Internet本身作为构建新一代操作系统的基础。Microsoft的宏伟目标是让.NET彻底改变软件的开发方式、发

行方式和使用方式，构建第三代Internet平台，解决各种协同合作问题，实现信息的高效沟通和分享，让整个Internet为人们提供最全面的服务。

Microsoft .NET技术可以分为两大部分：规范和实现。

## 1.2 公共语言架构（CLI）

Microsoft为.NET技术制定了一套完整的规范，并将它提交到ECMA等标准化组织，以促进.NET的广泛应用。这套规范就是公共语言架构（CLI，Common Language Infrastructure）。它的组成部分包括：

- 通用类型系统（CTS，Common Type System）。定义了一套类型系统的框架，规定了数据类型的声明、使用和管理方法。.NET中的任何数据类型都必须遵守该框架中的约定。
- 公共语言规范（CLS，Common Language Specification）。一组语言规则的集合。如果某种编程语言符合其中的所有规则，它就是标准的.NET编程语言，可以实现和其他.NET编程语言的跨语言集成；如果某个组件中的代码使用了其中规定的功能，它就是标准的.NET组件，可以实现和其他.NET组件的交互。
- 通用中间语言（CIL，Common Intermediate Language）。一种中性语言，更准确地说是一套处理器无关的指令集合。可以针对不同的平台将通用中间语言翻译为可执行的机器指令（二进制代码），而任何.NET编程语言所编写的代码可以被编译成通用中间语言指令集。
- 其他相关的标准化文件、格式、规定等。

## 1.3 .NET Framework

公共语言架构是.NET的技术规范，Microsoft依据该规范在Windows平台上开发了一个完整的实现，这就是.NET Framework，它包括.NET类库和公共语言运行时（CLR，Common Language Runtime）两大部分，其整体环境结构如图1.1所示。

其中，.NET类库定义了大量的可重用对象和组件，开发人员可以利用这些现成的对象和组件快速开发各种服务和应用。不论是.NET类库中预定义的对象和组件，还是开发人员自行创建的对象和组件，它们都运行在公共语言运行时这个平台上，而底层平台的操作细节由公共语言运行时负责实现。

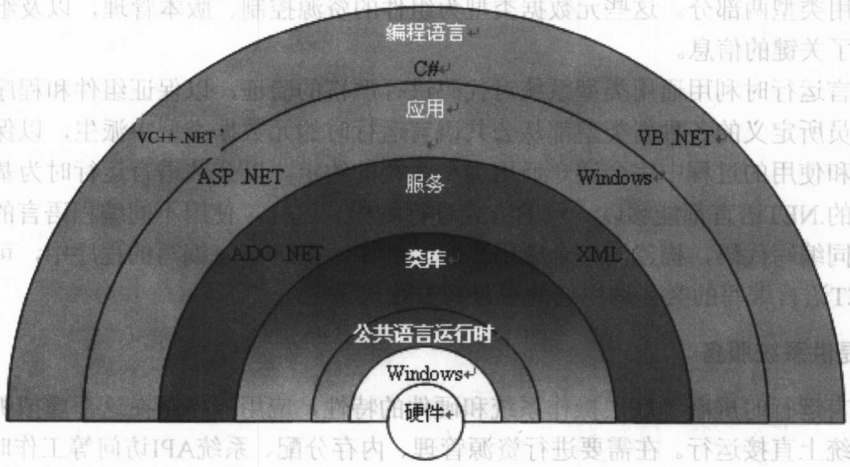



图 1.1 .NET Framework 环境结构

### 1.3.1 公共语言运行时（CLR）

准确地说，公共语言运行时就是一个虚拟机，它为各种.NET应用提供了一个高性能的、抽象于底层操作系统和硬件的运行时环境。公共语言运行时的主要功能体现在3个方面：

**（1）管理代码的执行**

各类.NET应用程序的代码被编译为通用中间语言。在程序执行时，公共语言运行时将通用中间语言翻译为具体的机器指令，负责加载所需的元数据类型、组件及其他各种资源，并在执行过程中提供安全性控制、错误处理、垃圾回收等服务；在必要时，公共语言运行时还可以进行程序的版本处理、代码调试等工作。

由于公共语言运行时对程序执行细节进行了封装，开发人员就可以专注于程序的业务和功能流程。此外，公共语言运行时提供了一致的实现模型，各类组件和程序能够以统一的方式进行交互。因此，公共语言运行时是在.NET Framework上实现程序高度的互操作性、集成性以及可重用性的关键所在。

为了充分利用公共语言运行时所提供的强大功能，Microsoft推荐使用托管代码（Managed Code）来开发各种.NET应用。托管代码是指满足公共语言规范要求、所使用元数据类型均属于通用类型系统、并使用针对公共语言运行时的语言编译器所开发的代码。使用托管代码开发的对象，其整个生命周期都在公共语言运行时的管理范围当中，因而能够享受跨语言集成、跨语言异常处理、增强的安全性、版本控制和部署支持、简化的组件交互模型、调试和分析等多种服务。

然而，.NET也允许开发人员使用非托管代码来绕过公共语言运行时的某些服务，或自行实现公共语言运行时的某些功能。托管代码和非托管代码之间可以相互调用。

**（2）提供通用类型系统**

公共语言运行时定义了一套完整的、符合通用类型系统约定的元数据类型集合，包括

值类型和引用类型两部分。这些元数据类型为组件的资源控制、版本管理，以及组件之间的交互提供了关键的信息。

公共语言运行时利用通用类型系统对代码进行严格的验证，以保证组件和程序的可靠性。开发人员所定义的各种新类型都从公共语言运行时的元数据类型中派生，以保证这些类型在生成和使用的过程中完全遵守通用类型系统的约定。以公共语言运行时为基础，现有的及新增的.NET语言都能够以一致的方式进行编程。这样，使用不同编程语言的开发人员就可以共同编写代码，构建同一个应用程序。例如，在C#语言编写的程序中，可以继承使用VB .NET语言编写的类，调用它的对象和方法。

**（3）提供系统服务**

公共语言运行时屏蔽了底层操作系统和硬件的特性，应用程序是在这个虚拟机上而不是在操作系统上直接运行。在需要进行资源管理、内存分配、系统API访问等工作时，由公共语言运行时与操作系统打交道，.NET组件和应用程序则使用公共语言运行时提供的统一接口，这不仅简化了开发难度，而且还使得程序能够在不同的平台上方便地移植。

### 1.3.2 .NET类库

.NET类库是Microsoft开发的一个面向对象的可重用类型集合，旨在帮助开发人员进行各种Windows应用和Web应用开发。这些预定义类型实现了丰富的功能，并提供了良好的可重用性。类库中的主要内容有：

- 基础类型定义，包括各种数学类型及其运算、字符串类型、对象类型等；
- 数据结构封装，包括集合、链表、队列、堆栈等；
- Windows界面要素，包括标签、按钮、菜单、工具栏、文本框、列表框、数据绑定和导航控件等；
- Web界面要素，能够实现Windows界面所具有的大部分功能，还包括Web客户端所特有的要素，如验证控件、客户端缓存等；
- Web Service，包括服务描述、消息与响应、协议等；
- XML文档处理，包括XML文件、属性、元素、节点、读写器、解析器等；
- 文件输入输出，包括驱动器、目录、文件、流、读写器等；
- 数据访问，包括数据连接、适配器、数据集、表格、记录、主键、关联等；
- 类型反射，包括程序集、类和对象、方法、属性、字段等的元数据类型信息获取；
- 异常处理，包括系统、应用程序、数学运算、安全性限制等引发的异常。

.NET类库中的类型主要包括类、接口和值类型，它通过命名空间对这些类型进行层次化的组织和管理。

## 1.4　C#语言简介

20世纪80年代以来，C和C++一直是使用最为广泛的商业应用开发语言。这两种语言在带来强大控制能力和高度灵活性的同时，代价是相对较长的学习周期和较低的开发效率，同时对控制能力和灵活性的滥用也给程序的安全性带来了潜在的威胁。C++语言过度的功能扩张也破坏了面向对象的设计理念。软件行业迫切地需要一种全新的现代程序设计语言，它能够在控制能力与生产效率之间求得良好的平衡，特别是将快速应用开发与对底层平台所有功能的访问紧密结合在一起，并与Web标准保持同步，C#（读作“C-Sharp”）语言就是这一历史使命的承担者。

C#语言从C和C++发展而来，它汲取了包括C、C++、Java在内的多种语言的精华，是一种简单、完备、类型安全和完全面向对象的高级程序设计语言。它的设计目标就是在继承C和C++强大功能的同时，兼有RAD（Rapid Application Development，快速应用程序开发）语言的简易和高效。作为.NET的核心编程语言，C#充分享受了公共语言运行时所提供的优势，能够与其他应用程序方便地集成和交互。下面对它的几个突出特点进行了描述：

- 简洁的语法。C#取消了指针，也不定义烦乱的伪关键字，它使用有限的指令、修饰符和操作符，语法上几乎不存在任何冗余，整个程序结构十分清晰。初学者可以轻松快速地掌握C#的基本特性，而C和C++程序员转入C#则几乎不会有任何障碍。
- 精心的面向对象设计。C#具有面向对象的语言所应有的基本特性：封装、继承和多态性。它禁止多继承，禁止各种全局方法、全局变量和常量。C#以类为基础来构建所有的类型，并通过命名空间对代码进行层次化的组织和管理，减少了发生命名冲突的可能性。
- 与Web的紧密结合。借助Web服务框架，C#使得网络开发和本地开发几乎一样简单。开发人员无需了解网络的细节，可以用统一的方式来处理本地的和远程的C#对象，而C#组件能够方便地转变为Web服务，并被其他平台上的各种编程语言调用。
- 完整的安全性与错误处理。C#符合通用类型系统的类型安全性要求，并用公共语言运行时所提供的代码访问安全特性，从而能够在程序中方便地配置安全等级和用户权限。此外，垃圾收集机制自动管理对象的生命周期，开发人员无须再负担内存管理的任务。应用程序的可靠性进一步得到了提高。
- 版本管理技术。C#在语言中内置了版本控制功能，并通过接口和继承来实现应用的可扩展性。应用程序的维护和升级更加易于管理。
- 灵活性与兼容性。C#中允许使用非托管代码，能够与各种现有的组件和程序（包括COM组件、WIN32 API等）进行集成和交互。它还可以通过代表（delegates）来模拟指针的功能，通过接口来模拟多继承的实现。

为了吸引软件开发人员和合作伙伴对.NET的认同，Microsoft推出了新一代的集成开发环境——Microsoft Visual Studio .NET。该环境提供了对C#语言编程的可视化支持，使得开发人员能够方便地创建、运行、调试和发布C#程序，从而针对.NET平台快速地构建广泛的应用。

## 1.5 C# 2.0的新增特性

作为一个新生事物，.NET技术很快取得了巨大的成功。Internet上仿佛一夜之间出现了不计其数的Web服务组件，大量的网站（特别是电子商务网站）升级到了.NET平台，包括Borland在内的许多开发工具厂商纷纷推出了自己的.NET编程语言。据统计，在美国有一半以上的开发人员在使用Visual Studio .NET进行开发，而C#则成为其中最受欢迎的开发语言。然而，.NET前进的脚步并没有因此而停止。Microsoft在2003年初将.NET Framework从1.0版本升级到了1.1版本，主要内容是对.NET类库的修改和完善，并通过.NET Framework精简版强化了对各种智能设备编程的支持。Microsoft还在新一代操作系统Windows 2003中内置了.NET Framework 1.1。

作为Microsoft .NET战略计划的关键一步，.NET Framework 2.0版本的推出则标志着.NET技术一次全新的重大升级。.NET Framework 2.0中新增的主要功能有：

- 对64位处理器和操作系统的全面支持；
- 对IPv6网络协议的支持；
- 可轻松实现权限管理的访问控制列表编程方法；
- 全新的数据访问策略ADO .NET；
- 新增的数据保护编程接口；
- 全新的Web应用开发平台ASP .NET 2.0；
- 高安全性的加密数据流；
- 增强的COM互操作性；
- 增强的证书管理功能；
- 更为强大的调试功能；
- 网络连接中的变化检测；
- 包括FTP客户端请求、HTTP资源缓存等在内的分布式计算新功能的实现；
- 在通用类型系统中对泛型的支持；
- 事务管理和线程管理。

与此同时，Microsoft也将C#语言升级到了C# 2.0，用以全面支持在.NET Framework 2.0上的编程。在新版本的Visual Studio 2005中也包含了对C# 2.0可视化编程的支持。C# 2.0中的新增特性主要有：

- 泛型（Generic Type）。这是从C++的模板（Templates）发展而来的一个特性，它允许在类和方法的定义中使用参数来指代抽象的数据类型，需要时可以用不同具体的数据类型来取代该参数，从而在更高的层次上实现代码的可重用性。C#将类的继承机制和泛型有机地结合在一起，把面向对象的方法推向了一个新的高度。泛型的概念适用于类（请参看第10章）、结构和接口（请参看第11章），也适用于单独的方法（请参看第13章）。
- 可空类型（Nullable Type）。这是泛型的一个具体应用，它允许在变量中包含未确

定的值，例如一个布尔变量的值可以是真或假，也可以未确定。这就为值类型的处理带来了更大的方便（请参看第12章）。

- 遍历器（Iterator）。允许使用foreach循环结构来遍历集合中的每个元素。遍历器所实现的功能并不复杂，但它是面向对象设计模式的一个典型应用。.NET类库采用了完美的面向对象设计，其中用到了多种设计模式，遍历器算是其中一个杰出的代表（请参看第14章）。
- 匿名方法（Anonymous Method）。可以把一段代码直接作为参数使用，而无需显式地定义一个方法，这样不仅减少了编程的工作量，提高了代码的可维护性，更极大地方便了程序中的各种运算方式（请参看第15章）。
- 分布类型（Partial Type）。允许将一个类型的定义分布到多个代码段乃至多个源文件中，从而提高代码的可读性和可维护性（请参看第18章）。

## 1.6 小　结

Microsoft .NET是面向Internet的计算平台，它将软件开发和应用的理念、方式和模型都推向了一个全新的境界。C#则是Microsoft专门针对.NET而设计的高级编程语言，它在保持强大功能的同时，能够显著地提高现代软件的开发效率。C# 2.0是对C#语言的一次全面升级，使用它编写的程序具有更为优越的可重用性、可维护性和灵活性。本书随后的各章节将对C#语言的主要功能特性逐一进行深入的讲解。