计 算 机 基 础 课 程 系 列 教 材

C#程序设计教程

为教师配有
教学课件等

郑阿奇 梁敬东 主编
朱毅华 时跃华 赵青松 编著

机械工业出版社
China Machine Press

计算机基础课程系列教材

C#程序设计教程

郑阿奇 梁敬东 主编
朱毅华 时跃华 赵青松 编著

机械工业出版社
China Machine Press

本书以微软Visual Studio .NET 2003/2005作为工作平台，系统介绍C#语言基础知识、面向对象编程及应用等相关内容。主要内容包括：C#的编程基础、面向对象编程、Windows应用程序开发、GDI+编程、文件操作、数据库应用开发、多线程技术和Web应用程序开发。书中包含大量习题，可帮助读者进一步掌握基本编程和基本概念。书后还附有10个实验，可锻炼编程和应用的实践能力。

本书可作为高等学校相关专业C#程序设计课程的教材，也可供编程人员参考。

本书配有教学课件和应用实例的源文件，需要者可在华章网站www.hzbook.com下载。

图书在版编目（CIP）数据

C#程序设计教程/郑阿奇，梁敬东主编. -北京：机械工业出版社，2007.3
（计算机基础课程系列教材）
ISBN 978-7-111-20684-2

Ⅰ.C…　Ⅱ.①郑…　②梁…　Ⅲ.C语言-程序设计-高等学校-教材　Ⅳ.TP312

中国版本图书馆CIP数据核字（2007）第002710号

机械工业出版社（北京市西城区百万庄大街22号　邮政编码　100037）
责任编辑：李东震
北京瑞德印刷有限公司印刷 · 新华书店北京发行所发行
2007年3月第1版第1次印刷
184mm × 260mm · 23印张
定价：33.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换
本社购书热线：（010）68326294

前　言

C#在继承C++和Java等语言优点的基础上，不仅具有封装、继承和多态等特性，而且还增加了不少创新元素，是程序设计语言研究的重大成果。它能较好地适应软件工程的需要，是目前主流的程序设计语言之一。

C#以微软.NET作为工作平台，除了Windows基本功能外，在用户交互界面、Web应用、数据库应用等方面得到了广泛应用。所以，C# 深受用户欢迎。

本书包含教程、习题和实验。教程理论联系实际，少理论多应用。本书教程部分在介绍微软.NET与C#的基础上，通过一个简单实例了解操作过程；再比较系统地介绍C#的编程基础，在此基础上分两章由浅入深介绍面向对象编程；然后系统介绍 Windows应用程序开发、GDI+编程、文件操作、数据库应用开发、多线程技术和Web应用程序开发。习题部分主要突出基本编程和基本概念。实验部分主要锻炼编程和应用，读者先跟着做，然后自己练习。一般来说，读者通过教程学习、习题练习、特别是认真的上机操作，都能够在较短的时间内学会在Windows环境下用C#解决一些基础的应用问题。

本书配有教学课件和应用实例的源文件，教师可在华章网站www.hzbook.com下载。

本书操作的环境为微软Visual Studio .NET 2003/2005。

本书由南京农业大学梁敬东编写第5、6、7章，朱毅华编写第1、8、9章，时跃华编写第2、3章，赵青松编写第4、10章。全书由郑阿奇（南京师范大学）和梁敬东（南京农业大学）统编和定稿，另外，南京农业大学孙剑清、谢晋志也参与了部分工作。本书封面署名不分先后。

由于编者水平有限，疏漏之处在所难免，敬请读者批评指正。

编　者

目　录

第1章 .NET与C#

C#是为.NET平台应用开发而全新设计的一种现代编程语言，随着微软的.NET战略进入开发人员的视野，C#很快成为Windows应用开发语言中的宠儿。为什么要设计一门新的编程语言？C#在微软的.NET平台中占据什么样的地位？C#与已有的C++、Java语言有什么样的关系？本章试图给读者一个简要的解答，帮助读者对C#有一个准确的定位，并能对自己的学习做好安排。

1.1 初识C#

1.1.1 Microsoft .NET与C#

学习C#离不开对.NET的理解，但是由于商业宣传的超前定位与产品线的混乱，以及目前对.NET的各种解释已构成了一个庞大杂乱的概念体系，使得人们对.NET战略的理解也存在着很多疑惑，为了学习C#而去死抠概念与定义只会让读者陷入迷惑。

为了帮助读者理解，下面从方便学习的角度介绍一些与C#学习有关的必要的.NET基础知识：

1. .NET战略定位于Web Services平台

2000年6月22日，微软公司正式推出了其下一个十年的战略核心产品：Microsoft .NET（以下简称.NET）。在之后两年的时间里，.NET从计划逐步变成现实，给软件开发商和软件开发人员提供了支持未来计算的高效Web Service开发工具。在第三代互联网中，各个网站已经不再是一个一个的信息孤岛，而是相互联结、相互调用，共同为用户甚至是智能设备提供服务，它们之间的联结就是依靠Web Service建立起来的。举个简单的例子，未来某个家庭里的智能冰箱检测到食物不足，就会自动通过Web Service联结指定超市服务来订购食品，并通过银行提供的Web Service来付账。虽然现在看起来这些还只是梦想，但它至少指明了未来互联网的发展方向。

表面看Web Service就是一个应用程序，它向外界暴露出一个能够通过标准的互联网协议（如超文本传输协议（HTTP）和XML）进行调用的应用程序接口（Application Service Interface, API）。一旦部署一个Web Service以后，其他Web Service应用程序可以发现并调用它部署的服务。如一个集成的天气预报服务可以通过Web调用所有城市的天气预报网站实现其功能，只要这些网站能够实现Web Service接口。Web Service使得网站的服务不仅能被用户访问，也能被其他应用程序访问，从而集成更强大的功能。

要做到这些，.NET提供了一套含数据库服务器和Web服务器在内的运行平台、一套新的编程模型、一套可编程的Web Service和一种简单一致的访问应用程序、服务和设备的方法。微软试图通过这些为Internet网络和分布式应用程序的开发提供一个新的开发平台，简化应用程序开发和部署，从而为构建Web Service提供一个标准框架，改善系统和应用程序之间的交互性和集成性，使应用程序能够访问任何设备。.NET致力将手机、PDA、浏览器和门户应用程序集成到一起，形成一个统一的开发和运行环境。目前已经有了适用于PC、智能设备、嵌入式设备的.NET运行环境，而开发工具Visual Studio .NET则提供了开发Web Service和这些设

备应用程序的统一环境。

2. .NET是一个运行时平台

为了实现微软的战略目标，.NET的编程模型将开发语言与运行平台分离，实现了独立于语言的组件技术，通过不同的运行平台，.NET 应用可以被扩展到PC、PDA、手机和嵌入式设备上。.NET运行平台称为.NET 框架（.NET Framework），是.NET平台的基础架构，它创造了一个完全可操控的安全的和特性丰富的应用执行环境，这不但使得应用程序的开发与发布更加简单，并且实现了众多种类语言间的无缝集成。

.NET框架的体系结构如图1-1所示。

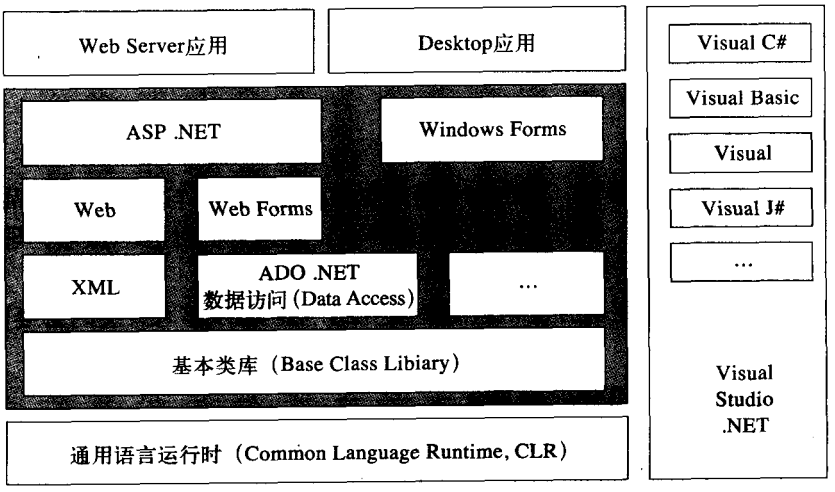

图1-1 .NET框架的组成

从图中可以看出，CLR是.NET框架的底层。通过实现不同版本的.NET框架，.NET应用可以扩展到不同的平台和设备上，因此在某种程度上CLR和Java的虚拟机有异曲同工之妙。CLR的意义在于：从运行时的角度来看，.NET是基于Windows系列操作系统（包括服务器、工作站、移动设备（如智能手机、PDA））的跨设备的统一运行平台；而从开发者的角度来看，凡是符合公共语言规范（Common Language Specification, CLS）的程序语言（如C#和Visual Basic .NET等）所开发的程序都可以在任何有CLR的操作系统上执行和互操作，具有与平台无关的特性，同时可以相互调用。例如用C#开发的组件可以被Visual Basic .NET无缝集成。所以可以说 .NET是跨语言的集成开发平台。

CLR的另一个重要性是基于该平台的所有语言的特性（如数据类型、垃圾收集、异常处理等）都是在CLR层面实现的，而不是由语言本身实现的，C#也不例外。因此学习.NET编程语言不仅需要对语言本身的理解，同时需要对.NET的CLR技术架构有一定的理解。

CLR之上是.NET框架提供的一整套从基本输入输出到数据访问以及用于构建Web应用和Windows桌面应用的组件类库。其中ADO .NET提供了对数据库访问的支持，ASP .NET提供Web应用开发的支持，Windows Form组件库则提供标准的Windows GUI组件用于构建桌面应用程序。这些类库独立于编程语言而存在，无论是C#还是Visual Basic .NET应用程序都可以调用。事实上，C#并没有属于自己的类库，它所使用的编程接口就是 .NET提供的类库。要使用一门.NET程序设计语言进行开发，除了要学习语言本身的内容外，还需要掌握 .NET框架的类库。

使用C#进行应用开发真正需要费功夫学习的其实是 .NET的类库。ADO .NET、Windows Forms和ASP .NET等组件库不是C#或Visual Basic .NET语言的一部分，但却是进行相应的应用软件开发所必须掌握的。因为C#本身只有77个关键词，其语法风格对于了解C++和Java的程序员来讲非常熟悉，而 .NET的类库则非常庞大，包含了超过4500个类，在C#开发中要用到这些类。

3. .NET是一个开发平台

为了开发.NET的应用，微软推出了强大的 .NET开发平台Visual Studio .NET（以下简称VS .NET）作为微软 .NET战略的重要组成部分。VS .NET包含C#、Visual Basic .NET、Visual C++ .NET等多个开发语言，同时还为第三方语言工具厂商提供了接口，只要支持CLS公共语言规范的语言都可以集成到VS .NET环境中。

VS .NET充分发挥CLR的潜力，为开发者提供了一个统一的集成开发环境和调试器。由于采用统一类型定义和共享类库，各语言不仅在运行时，而且在设计时就可以实现对象级的交互。VS .NET不仅提供Web Services开发工具，而且提供从Web Services开发到发布、注册、整合的全过程支持。为创建和部署Web Services，.NET平台采用一系列标准的互联网协议，如XML、SOAP、WSDL、UDDI等用于Web Service的布署、请求和响应。

组件尤其是中间件的市场，长期以来一直是Java的天下，而 .NET则试图杀入面向组件的开发。无论是面向组件的语言（C#, VB .NET, C++等），还是传统的语言（COBOL），在 .NET平台中，都可以产生基于微软中间语言的组件，而这些组件在运行时具有对象级的交互能力。

软件人员最关心的还是开发效率的提高，VS .NET人性化界面和众多工具将成倍提高开发效率。无论针对传统的Windows桌面开发，还是Web Services，开发人员都不必再为每种不同应用重写全部代码，不必为不同的客户设备定制不同的界面。

4. C#是 .NET的核心开发语言

C#语言是随 .NET一起设计出来的全新开发语言，其设计目的就是作为VS .NET的核心语言，它是为了那些愿意牺牲C++某些原有能力换得更多便利和效率的企业应用开发者而创造的。经过短短几年的发展，C#已经成为Windows平台上软件开发的主流语言之一。学习和掌握C#语言进行.NET应用开发，可以说是相当一部分软件开发人员面临的挑战和机遇。C#受到如此的欢迎，主要是因为两个方面，首先，C#是专门为 .NET开发而量身定制的，因而是 .NET开发的最佳语言，.NET框架提供的由一百多万行代码构建的类库也基本上由C#实现。其次，C#是一种基于现代面向对象设计方法的语言，如果抛开一切非技术方面的因素，C#无疑是有史以来最好的编程语言和企业级开发语言的集大成者。面向对象、类型安全、组件技术、自动内存管理、跨平台异常处理、版本控制、代码安全管理……现在不可能在其他语言中找到所有这些特性。对于C#这些优点，我们可以从C#的设计思路中得到更深层次的认识。

1.1.2 C#的优势

.NET是一种新的跨语言的组件式开发模型，传统的语言如果不作适当的修改就难以适应，Visual Basic .NET为了适应 .NET而被改得像一门全新的语言，对C++的修改也导致了很多批评。对微软来说，充分吸收所有关于软件开发和软件工程研究的最新成果，为 .NET全新打造一门新的语言是最佳选择，这就是C#。在设计C#时，微软吸取了其他类似语言如C/C++和Java的经验，这些语言是近20年来面向对象规则得到广泛应用后开发出来的，然而C#已经走得更远。

1. C#在设计上的优势

作为编程语言，C#是现代的、简单的、完全面向对象的，而且是类型安全的。如果你是一个C/C++或Java程序员，那么你的学习曲线将会很平坦。许多C#语句是直接从你所喜爱的语言里借来的，包括表达式和操作符，乍看上去，一个C#程序很像一个C++或Java程序。重要的是，C#是一种现代编程语言。在类、名字空间、方法重载和异常处理等方面，C#去掉了C++中的许多复杂性，借鉴了和修改了Java的许多特性，使其变得更加易于使用，不易出错。对易于使用的贡献是去掉了C++的一些特性，它没有宏，没有模板，没有了多重继承。尤其是对于企业开发者来说这些特性所造成的问题要比其带来的益处还要多。新增特点有：严格的类型安全性、版本处理技术、垃圾收集等，所有这些特性都是为了方便开发面向组件的软件。下面列举了一些C#在设计上的优点。

• 简单性

C#在设计上的首要目标就是简单性。

没有指针是C#的一个显著特性。在默认情况下，使用一种可操控的（Managed）代码进行工作，此时一些不安全的操作如直接的内存存取将是不允许的。

在C#中不再需要记住那些源于不同处理器结构的数据类型，例如可变长的整数类型，C#在CLR层面统一了数据类型，使得 .NET上的不同语言具有相同的类型系统。可以将每种类型看作一个对象，不管它是初始数据类型还是完全的类。

在这里，整型和布尔数据类型是完全不同的类型。这意味着if判别式的结果只能是布尔数据类型，如果是别的类型则编译器会报错。因此那种搞混了比较和赋值运算的错误不会再发生了。

C#去掉了多年来在C++中语法和功能的冗余问题，常用的形式在C#中被保留下来，而其他冗余形式在C#的设计中被清除掉了。

• 现代性

许多在传统语言中必须由你自己来实现或者根本不存在的特征，都成为基础C#实现的一个部分。金融类型对于企业级编程语言来说是很受欢迎的一个附加类型。你可以使用一个新的decimal数据类型进行货币计算。

与Java一样，以往烦琐而容易出错的指针操作和内存管理不需要程序员来处理。CLR平台运行环境提供了一个垃圾收集器负责C#程序的内存管理工作。对于C++程序员来说，异常处理是C#的一个主要特性，然而和C++中不同，.NET的异常处理是跨语言的（CLR运行环境的另一特点）。

安全性是现代应用的头等要求，C#通过代码访问安全机制来保证安全性，根据代码的身份来源，可以分为不同的安全级别，不同级别的代码在被调用时会受到不同的限制。

• 面向对象

C#支持面向对象的所有关键概念：封装、继承和多态性。整个C#的类模型是建立在.NET虚拟对象系统（VOS, Virtual Object System）之上的，而这个对象模型是基础架构的一部分而不再是编程语言的一部分——它们是跨语言的。

C#中没有全局函数、变量或常数。每样东西必须被封装在一个类中，或者作为一个实例成员（通过类的一个实例对象来访问），或者作为一个静态成员（通过类型来访问），这会使你的C#代码具有更好的可读性，并且减少了发生命名冲突的可能性。

在类中定义的方法默认情况下不是虚拟的（不能被派生类所覆盖），这一做法的优点是可

以消除另一个产生错误的来源——对方法的误覆盖。如果一个方法可以被覆盖，它必须有一个显式Virtual修饰符，C#也提供了新的new修饰符对方法进行重定义。

在用访问修饰符为类的成员指定不同的访问级别上，C#不仅支持private，protected和public访问修饰符，而且增加了第4个internal修饰符。关于这些访问修饰符的细节将在后面的章节中详细介绍。

多重继承的优劣一直是面向对象领域争论的话题之一，然而在实际的开发中很少用到，在多数情况下从多个基类派生所带来的问题比这种做法所能解决的问题要更多，因此C#的继承机制只允许一个基类。如果需要多重继承，你可以使用接口。

一个可能产生的问题是：既然在C#中没有指针，那么怎样去模拟指针功能呢？这个问题的答案是delegates（委托），它提供了.NET事件模型的基础架构。至于具体如何实现，我们将在后面的章节中对其进行详细介绍。

• 类型安全性

当你在C/C++中定义了一个指针后，你可以自由地把它指向任意一个类型，包括做一些相当危险的事，比如将一个整型指针指向双精度型数据。只要内存支持这一操作，它就会凑合着工作，这当然不是你所设想的企业级编程语言类型安全性。与此相反，C#实施了最严格的类型安全来保护它自身及其垃圾收集器。因此，程序员必须遵守关于变量的一些规定，如：不能使用未初始化变量。对于对象的成员变量，编译器负责将它们置零。局部变量你应自己负责。如果使用了未经初始化的变量，编译器会提醒你。这样做的好处是：你可以摆脱因使用未初始化变量得到一个可笑结果的错误。

边界检查。当数组实际上只有n−1个元素时，不可能访问到它“额外”的数组元素n，这使重写未经分配的内存成为不可能。

算术运算溢出检查。C#允许在应用级或者语句级检查这类操作中的溢出，使用溢出检查，当溢出发生时会出现一个异常。

C#中传递的引用参数是类型安全的。

• 版本处理技术

在过去的几年里，几乎所有的程序员都和所谓的“DLL地狱”打过交道，产生这个问题是因为许多计算机上安装了同一DLL的不同版本。DLL是（Dynamic Link Library，动态链接库）的缩写，是一种编译为二进制机器代码的函数库。DLL在调用程序运行时才被调入内存执行，而不是在编译时链接到可执行程序内部，这样可以使用程序代码在二进制级别实现共享，而不必在每个应用程序中编译进一个副本，如果DLL中的代码更新了，只需要替换DLL文件即可更新所有使用该DLL的程序。然而这同时会带来的DLL文件版本的问题，不同版本的DLL可能与不同调用程序之间并不兼容，同一版本DLL也不能同时为不同的调用程序服务，结果是造成应用程序出现无法预料的错误，或者是在用户的计算机中不得不更换文件名保存同一DLL的多个版本。这种混乱的状态被称为“DLL地狱”。有时，旧的应用与较新版的DLL一起工作有时还可以运行，但大多数情况下，它们崩溃了。版本处理技术是当今一个真正令人很头疼的问题。与Java的设计不同，CLR的设计时就考虑到了版本处理技术支持，C#则尽其所能支持这种版本处理功能，虽然C#自己并不能保证提供正确的版本处理结果，但它为程序员提供了这种版本处理的可能性。有了这个适当的支持，开发者可以确保当他开发的类库升级时，会与已有的客户应用保持二进制兼容。

• 兼容性

C#不是存在于一个封闭的世界里。它允许你通过遵守重要的 .NET公共语言规范访问不同的API。公共语言规范定义了遵守这一标准规范的语言间相互操作标准。为了加强公共语言规范的兼容性，C#编译器检查所有公开输出项所遵守的条件，发现不符合规范的情况就会报错。

如果希望能够访问的旧式COM对象。.NET平台提供了对COM的透明访问，同样C#允许与C风格的API进行相互操作。动态链接库（DLL）的任何入口点——它们是以C风格给出的——可以在C#应用程序中进行访问。

• 灵活性

在访问旧式DLL API函数时很可能遇到指针的问题，需要传递指针参数的API有很多。为了解决这个问题，虽然C#代码默认是安全模式，但它也可以声明某些类或者仅仅是类的某些方法为非安全的，这一声明使你能够使用指针、结构和静态分配的数组了。安全代码和非安全代码两者都在可操控空间中运行。

2. C#与Java

C#与Java有些非常大的相似度，与Java相比，C#又有什么优势呢？

相对于其他传统编程语言，Java有一个无庸置疑的优点：用户以及编译器第一次不必了解生成可执行代码的特定CPU细节。Java引入了一个称为字节代码的编译代码中间层，使用一个虚拟抽象的机器而不是一个真实的机器。当Java编译器结束了一个源文件的编译后，你所得到的不是可以立即在一个给定平台上运行的代码，而是可以在任何真实的平台上运行的字节代码，唯一的条件就是这个平台具有Java虚拟机（JVM：Java Virtual Machine）。

Java模型的好处在于它是一种先进的、面向对象的语言，包含了预防常见错误的内置功能，并在仅仅一两个对象中携带了许多经常需要用到的功能。与C++相比，Java更易于读写，不容易出错，而且更加美观，但是它速度较慢也不太灵活。为了实现在任何软件和硬件平台上都可虚拟移植，Java放弃了将每个平台开发到极限的能力。其次，虚拟机的概念本身就是可移植和可共用的，因此对于分布式环境来说是理想的。Java对于为非Windows平台开发代码是最好的语言。

对于Windows平台来说，让Java适应Windows是不可能的，这是由于Sun的许可约束问题。但是Java实在是太吸引人了，Microsoft比谁都能更清楚这一点。因此，Microsoft又一次采取了“拿来主义”的手法，很好地利用了Java 的众多特性，设计出相当简单但十分强大的面向对象的C#编程语言。可以说，Java具备的优点，C#都可以或者都将具备，如果选择Windows平台应用的开发，C#语言无疑是未来之星。

1.1.3 C#学习环境

用户可以从微软的网站上得到免费的C#编译器，.NET框架的运行时也可以免费获得。为了有效地进行Windows Form、ADO .NET和ASP .NET的应用开发，读者应该安装和配置Visual C# .NET开发环境，本书后续章节的很多范例也将在Visual C# .NET环境下运行。

为了完整地使用本书内容，操作系统可以选择Windows 2000 Professional、Windows 2000 /2003 Server或Windows XP Professional。Windows XP Home不能安装IIS，无法执行ASP .NET，因此不能作为学习环境，而Windows 95/98/ME和Windows NT都无法安装Visual C# .NET。要进行本书的学习，需要一套Visual C# .NET或Visual Studio .NET，可以是Standard版本，也可以是Professional或Enterprise版本。

要顺利地安装与运行Visual C# .NET，机器主频应该在450MHz以上，内存256MB以上，否则即使安装成功运行起来也十分吃力。本节以Visual Studio .NET 2003版为例介绍本书学习环境的配置。

1. 准备安装

Visual Studio .NET和Visual Studio 6.0可装到同一台计算机上并可同时运行。但是在安装前如果计算机上曾安装过 .NET框架或Visual Studio .NET的早期版本，需要将其卸载。

如果已使用Visual Studio .NET的早期版本创建了应用程序，并使用 .NET框架合并模块对这些应用程序进行了部署，必须在安装 Visual Studio .NET 之前卸载这些应用程序，使用“控制面板”中的“添加/删除程序”功能即可完成卸载。此外Visual Studio .NET企业版附带的Visual Studio Analyzer版本与 Visual Studio Analyzer 6.0 不兼容。如在已安装Visual Studio Analyzer 6.0 的计算机上安装Visual Studio .NET企业版所提供的Visual Studio Analyzer，安装前要先卸载Visual Studio Analyzer 6.0。

在运行安装程序之前，用户应该在“控制面板”中的“添加/删除程序”功能中通过“添加删除Windows组件”安装好IIS服务器，否则将不能使用ASP .NET环境。为了保证性能与安全性，硬盘最好为NTFS分区。

2. 安装

安装过程如下：

（1）关闭所有打开的应用程序，以防止在安装过程中需要进行系统的重新启动。

（2）插入标为Visual Studio .NET CD1或DVD光盘，运行Setup.exe，进入“安装”对话框，如图1-2所示（由于发行版本的不同，读者的界面可能与此不同，但基本内容是一致的）。

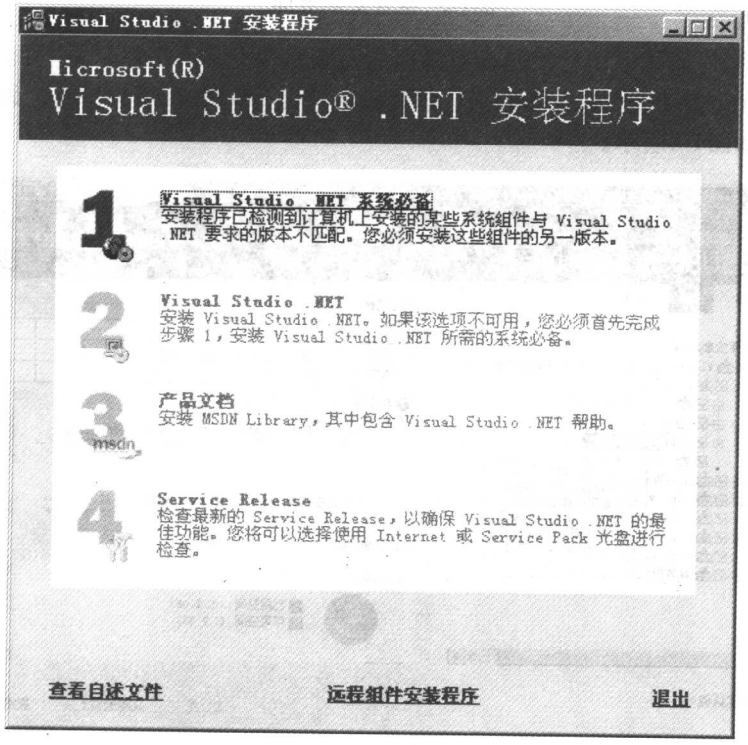

图1-2　安装程序主界面

（3）安装程序对已安装的组件进行扫描。如果确定系统需要进行组件更新，则在“安装”对话框中会出现“Visual Studio .NET系统必备”链接，单击该链接更新系统组件。此时需要插入安装光盘中的组件更新光盘完成组件的更新。安装程序检查系统后会显示图1-3所示界面，列出所需要安装或更新的系统组件，点击“立即安装！”完成组件更新。

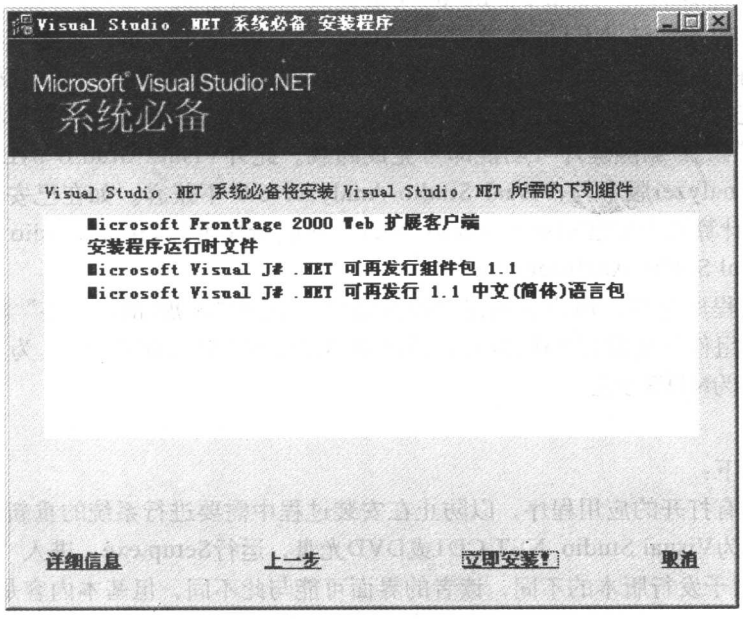

图1-3 组件更新界面

（4）在安装程序确认系统已经包含有最新的系统组件之后，“安装”主界面将会激活链接2——Visual Studio .NET，单击该链接后需要输入产品序列号和用户信息，输入正确之后则“安装”对话框如图1-4所示。

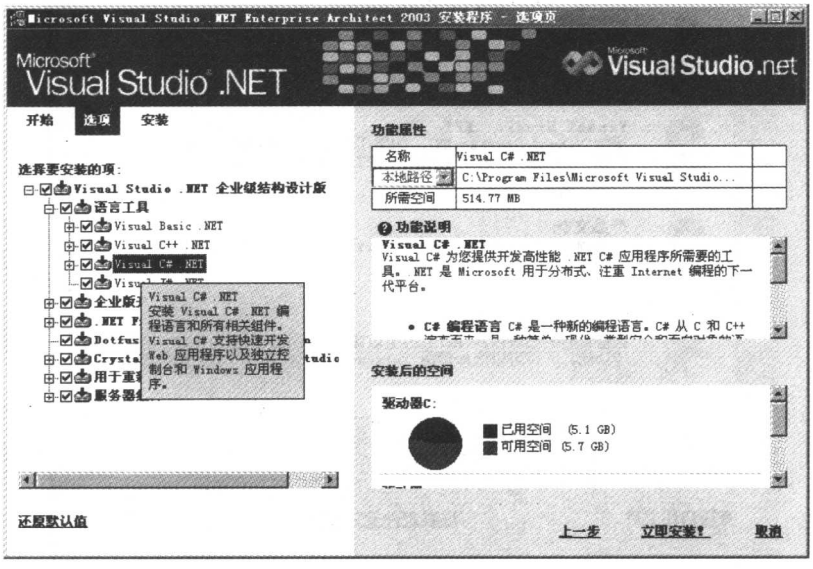

图1-4 安装选项

在左侧窗格中选择要安装的各项，右侧设置安装路径，请确认选中了Visual C# .NET选项，然后单击“立即安装！”按钮，其他选项一般采用默认值即可。配置完成后点击“立即安装！”即进入安装进程，需要等待数十分钟或更长时间，如果一切正常会显示如图1-5所示提示。

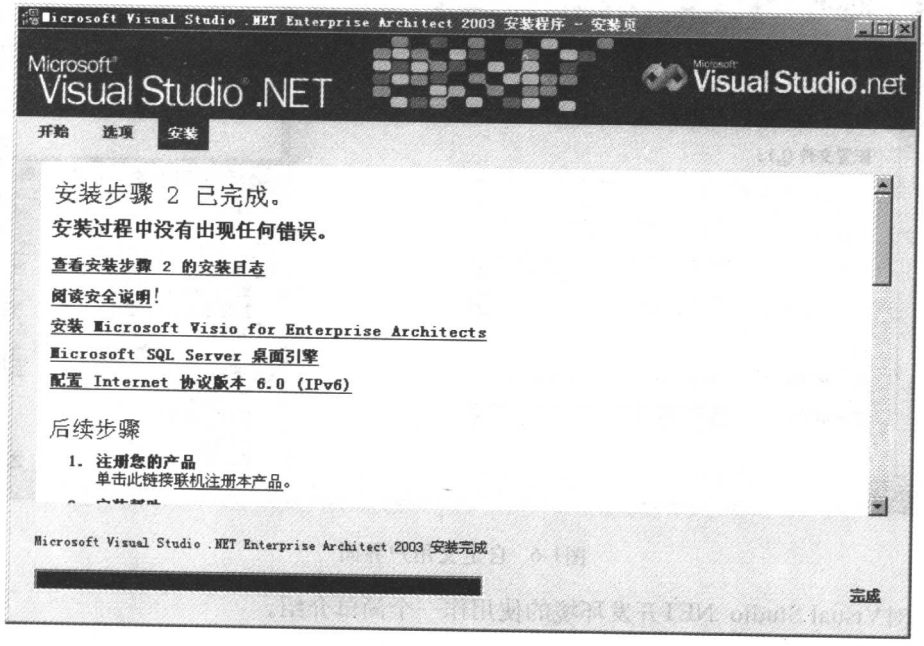

图1-5 安装成功

在此用户可以选择安装Visio和SQL Server桌面版，前者是一个可用于建模的绘图CASE工具；后者是微软SQL Server数据库服务器的简化版，可用于数据库应用的学习与开发。

3. 安装文档与更新

安装过程完成后安装主界面将激活链接3——“产品文档”和链接4——“Service Release”（由于版本不同，有的安装界面没有产品文档链接），请依次进入并根据提示完成安装。

4. 熟悉和定制Visual Studio .NET开发环境

一切安装完成之后，在“开始”菜单中可以找到“程序”→“Microsoft Visual Stuido .NET 2003”→“Microsoft Visual Stuido .NET 2003”快捷方式，点击后即可启动 .NET的IDE开发环境。与早期Visual Studio不同，Visual Studio .NET将所有开发语言都集成在同一个IDE开发环境之中，因此不会再有“Visual C++”、“Visual Basic”这样独立的程序项了。

启动IDE环境后首先进入的是起始页，页面中提供了“项目”、“联机资源”和“我的配置文件”选项卡，其中“项目”选项卡提供打开已有项目文件和新建项目的快捷入口，“联机资源”选项卡提供了网上开发资源的链接，“我的配置文件”选项卡中可以根据用户操作习惯配置界面布局方式和操作方式。要修改用户操作习惯，点击“我的配置文件”选项卡，如图1-6所示。

在这个页面中，可以设置键盘方案、窗口的布局、帮助的显示方式、在 .NET启动时显示哪些内容。

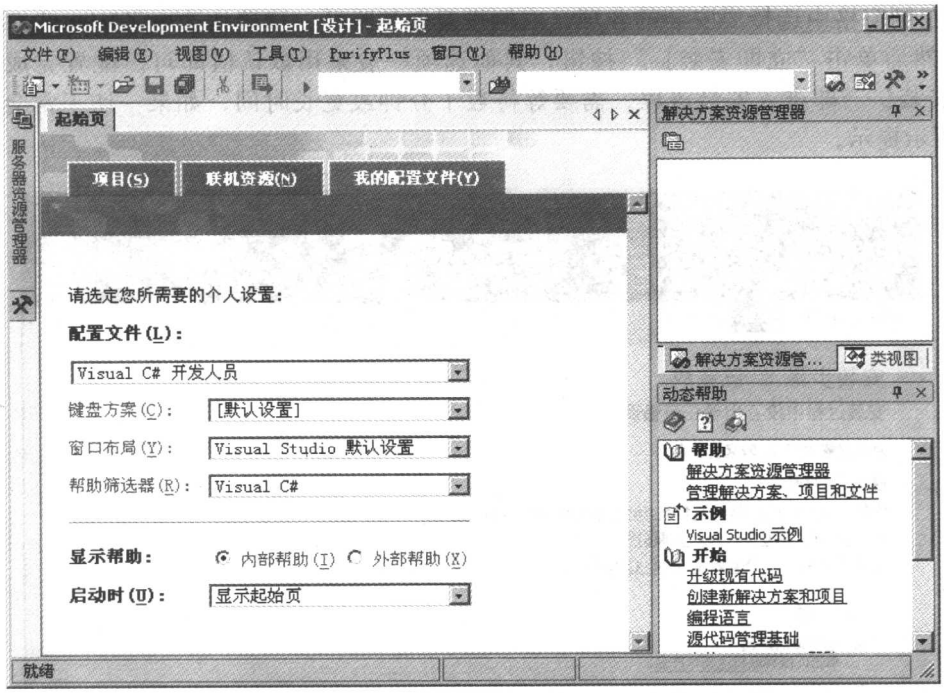

图1-6 自定义用户界面

下面对Visual Studio .NET开发环境的使用作一个简单介绍。

• 命令行编译

对于本书中的一些简单示例程序，如果只有一个C#源文件，可以直接调用C#编译器编译为CLR的可执行程序，方法是先用任意的文本编辑软件输入源代码并保存，再运行“开始”菜单中的“程序”→“Microsoft Visual Stuido .NET 2003”→“Microsoft Visual Stuido .NET工具”→“Microsoft Visual Stuido .NET 2003命令提示”打开命令行窗口，在命令窗口中输入：

```
csc 源文件驱动器:源文件路径\源文件名
```

csc是 .NET框架提供的C#编译器，将C#源文件的完整路径输入后编译器将对其进行编译并在当前目录下生成与源文件主文件名相同的.EXE可执行文件，运行该文件即可看到结果。csc编译器有一些开关参数，常用的如下：

/out:目标文件路径	指定输出的文件路径
/target:winexe	指定生成windows窗口应用程序
/target:exe	指定生成控制台应用程序
/target:library	指定生成库文件
/target:module	指定生成模块文件
/reference:引用文件名	指定源文件中using关键字指定的参考文件

其他的参数解释可以通过运行“csc /help”来查看。

• 利用Visual C# .NET开发环境

大多数情况下，要开始编写一个C#应用，首先应该建立一个项目。在起始页面的“项目”选项卡中点击“新建项目”按钮或者通过主菜单“文件”→“新建”→“项目...”打开“新建项目”对话框，如图1-7所示。