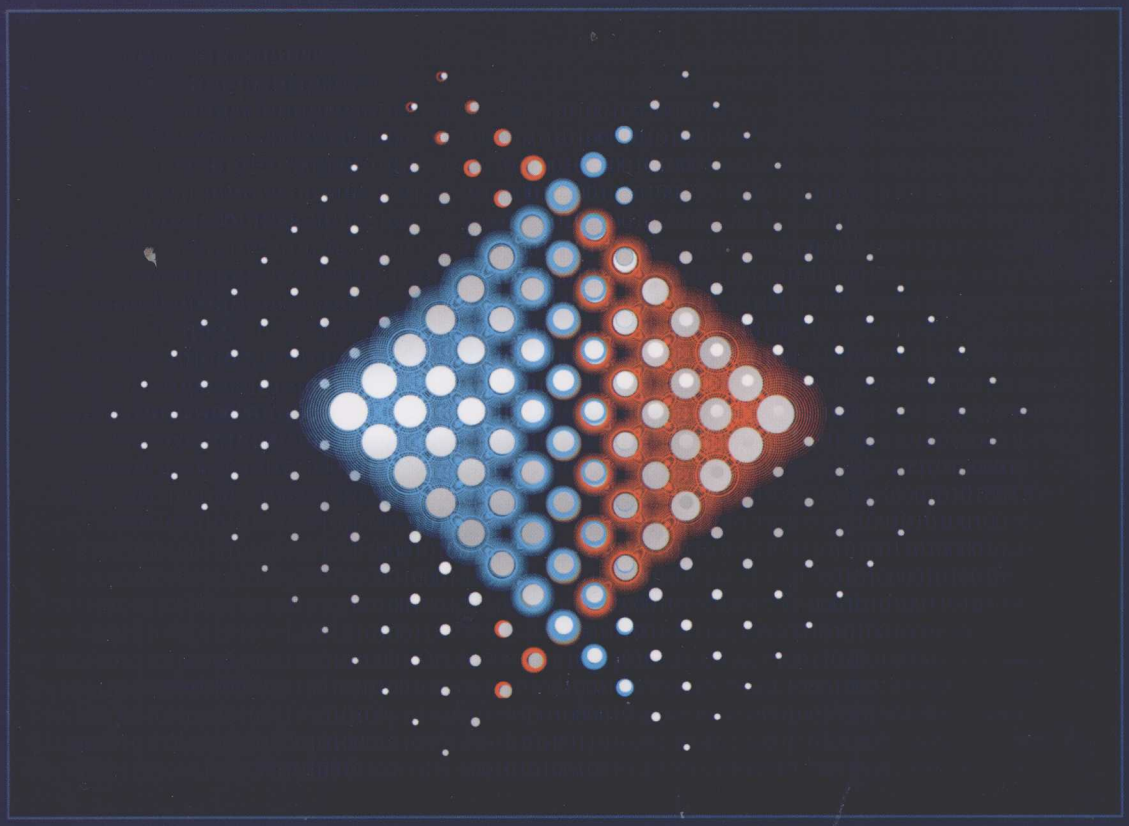
新编计算机类本科规划教材

ASP.NET 2.0 (C#) 大学实用教程

刘丹妮 主编

郭洪涛 陈明华 贾 跃 副主编



電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

新编计算机类本科规划教材

ASP.NET 2.0（C#）
大学实用教程

刘丹妮　主编

郭洪涛　陈明华　贾　跃　副主编

電子工業出版社
Publishing House of Electronics Industry
北京・BEIJING

内 容 简 介

本书围绕 ASP.NET 2.0 技术展开系统讲解，示例程序使用 C# 语言编写。全书共 14 章，内容包括：.NET 概述、C# 语法基础、ASP.NET 基础、服务器控件、页面设计、ASP.NET 内置对象、SQL Server 2005 与 T-SQL 语言、数据库基本操作、ADO.NET 数据库高级操作、XML 语言和使用、Web 服务、配置 ASP.NET 应用程序、综合示例和拓展实验等。

本书注重实用性和可操作性，内容循序渐进，示例面向应用，各章配有精心设计的习题，并为任课教师提供免费的电子课件和源代码。

本书适合作为高等院校相关专业及各类动态网站编程培训机构的教材，也可作为.NET 开发人员的参考用书。

图书在版编目（CIP）数据

ASP.NET 2.0（C#）大学实用教程 / 刘丹妮主编 .—北京：电子工业出版社，2009.1
新编计算机类本科规划教材
ISBN 978-7-121-07983-2

I. A… Ⅱ. 刘… Ⅲ. 主页制作—程序设计—高等学校—教材 Ⅳ. TP393.092

中国版本图书馆 CIP 数据核字（2008）第 197049 号

策划编辑：张 濮
责任编辑：李秦华
印 刷：北京季蜂印刷有限公司
装 订：三河市万和装订厂
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编 100036
开 本：787×1 092 1/16 印张：21.75 字数：557 千字
印 次：2009 年 1 月第 1 次印刷
印 数：4 000 册 定价：32.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

PREFACE 前言

自 2000 年 6 月微软公司推出 Microsoft .NET 以来，.NET 技术和相应产品已被广泛应用在信息技术领域的各个方面。在刚刚推出的 ASP.NET 2.0 中，ASP.NET 已经发展成一种非常成熟的产品。为了使 Web 开发方便快捷、代码量减少，它新增了许多控件、服务和功能。

本书讲解 ASP.NET 2.0 的基础知识，并以典型实例巩固读者学到的知识。经过最后一章综合实例的学习，读者将具备 Web 开发的基本技能，为 Ajax 等知识的进一步学习奠定良好的基础。

全书按照教材体例编写，共 14 章，各章章末配有丰富的习题以帮助读者巩固所学的内容。第 1 章至第 3 章介绍了 ASP.NET 的基础知识，包括 .NET 技术的概念、C#语言的语法基础和 ASP.NET 应用程序基础，使读者能够创建简单的 ASP.NET 应用程序。第 4 章至第 6 章分别介绍了服务器控件（HTML 服务器控件、Web 服务器控件、验证服务器控件和用户控件等）的特点和应用、页面设计技巧以及 7 种常用的 ASP.NET 内置对象，这些都是 ASP.NET 高级应用的基础。第 7 章至第 9 章循序渐进地介绍了对数据库的操作，包括 SQL 语言简介、数据库基本操作和 ADO.NET 数据库高级操作。第 10 章至第 12 章分别介绍了 XML 语言和使用、Web 服务的创建和调用、ASP.NET 应用程序的配置，帮助读者创建更加完善的 Web 应用程序。第 13 章是一个论文比赛支持网站的综合示例，帮助读者掌握 ASP.NET 开发的三层体系结构，并加深对本书内容的理解。第 14 章是拓展实验，帮助读者解决开发中的常见问题。

本书最大的特点是包含了大量示例，对于每个重要的知识点都有相应的示例。这些示例简单明了、针对性强，可以帮助读者快速理解所学的内容。特别是本书最后的综合示例和拓展实验，帮助读者提高编写 Web 应用程序的综合能力。

本书的读者分为两类。对于 ASP.NET 的初学者，可以按照目录安排，循序渐进地阅读本书，学习理解相应的示例，从而掌握基本的 Web 应用程序的开发技能；对于有一定编程基础和 Web 开发经验的读者，可以仔细研读示例，尤其是最后的综合示例，从而提高独立编写中小型 Web 应用程序的能力。本书适合作为高等院校相关专业及各类动态网站编程培训机构的教材，也可作为 .NET 开发人员的参考书。

本书所有示例都采用 C# 语言编写，以 Visual Studio 2005 作为开发环境，读者可根据附录中的介绍安装相应的开发环境。本书配有配套的电子课件、程序源代码和习题解答，读者可以登录电子工业出版社华信教育资源网（www.hxedu.com.cn）下载。

本书由刘丹妮担任主编、郭洪涛、陈明华、贾跃担任副主编，由刘丹妮统稿。其中，第3章、第7章、第8章、第10章、第9.1节、第9.3节、第9.4节、拓展实验和附录由刘丹妮编写，第1章、第2章、第4章和第11章由郭洪涛编写，第5.1节、第5.3节、第6章和第12章由陈明华编写，第5.2节、第9.2节和第13章由贾跃编写。

在本书的编写过程中，得到了魏鸿磊、高雪、刘忠萃、韩锐等同志的大力帮助，在此谨表示深深的谢意。

由于作者水平有限，书中疏漏和错误之处在所难免，恳请广大读者批评指正。如有问题，请发送邮件至 liudanni@neusoft.edu.cn 联系我们。

作　者

2008年7月

CONTENTS 目录

第 1 章 .NET 概述

1.1 .NET 的基本概念

.NET 平台将会对任何一种编程方式产生影响，它会使用户界面有根本性的变革，如同从 MS-DOS 到 Windows 的转变一样。它使用户能够在任何时间、任何地点通过一种自然化的界面来获取信息。

——比尔·盖茨

2000 年 6 月，微软公司向世界公布了 Microsoft .NET。它是微软公司倾注了所有的物力、人力和财力打造的一种战略，是一系列产品和技术的总称，是各种设备得以互通互连的平台。

微软公司表示，“Microsoft .NET 将会使计算和通信工作变得容易到前所未有的程度”，“它将产生出新一代因特网服务，使成千上万的软件开发商有机会创造出革命性的在线服务和新业务”。

如今，微软的.NET 技术和相应产品已经被广泛使用，并在软件领域获得高速发展，得到了越来越多开发人员的喜爱。从中小型的应用程序到大型的企业级应用程序，从桌面应用程序、Web 应用程序、Mobile 应用程序、Web 服务，到操作系统开发、嵌入式设备开发等，都能看到.NET 的身影。

.NET 2.0 版的推出使.NET 技术得到了更为广泛的应用，以.NET 2.0 为平台的应用软件如雨后春笋，.NET 已经成为了开发人员最重要的开发利器之一。

在学习.NET 的相关内容时，首先要了解微软公司.NET 战略的目标。它的目标是要构建一种新一代的互连网络环境，使得任何人在任何时间、任何地点、使用任何设备，都能够做到互通互连。围绕着这样一种充满想象力的构想，微软开发出了相应的软件产品和技术，而这些产品和技术就组成了一个面向网络、支持各种用户终端的开发平台环境——Microsoft .NET 平台。

1.2 .NET 平台的组成

Microsoft .NET 平台就像是微软的灵魂产品 Windows 一样。Windows 是一种硬件设备和软件运行环境的平台，在这样一个操作系统中，所有的外部设备（键盘、鼠标、显示器、打印机等）都变成了可以自由使用、无缝集成的一个整体，Windows 消除了不同硬件设备之间的差别。类似地，.NET 平台所要做的就是，消除互连环境中不同硬件、软件、服务的差别，使不同的设备、不同的系统都可以相互通信，使不同的程序和服务之间都可以相互调用。图 1-1 所示为 Microsoft .NET 平台的组成。之所以称为 Microsoft .NET 平台，是因为它是由微软官方

推出的.NET 平台组成标准。不过随后也会看到，这些标准的组成部分是可以替代的。

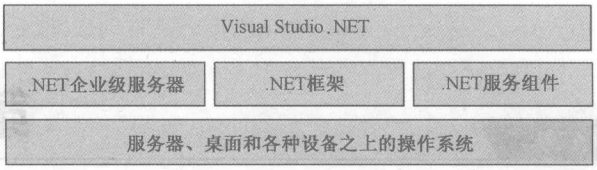


图 1-1 Microsoft .NET 平台的组成

在.NET 平台中，需要操作系统的支持。这些操作系统可以包括任何一种 Windows 平台，如 Windows 2000，Windows XP，Windows CE，Windows 2003 和 Windows Vista 等。操作系统是各种设备得以运行的基础，所以，操作系统也成为了.NET 平台的基本组成部分。微软公司推出的.NET 平台推荐使用的是 Windows 操作系统，但是后面会看到，.NET 技术并不局限于微软的 Windows 操作系统，它在其他的操作系统中也可以使用。只不过出于商业的考虑，Microsoft .NET 并没有跨越其他操作系统，但是从原理上来看，.NET 跨越平台是完全可行的。

.NET 平台的另一个组成部分是.NET 企业级服务器，包括用于简化大型服务器系统的开发和管理的产品工具，如 Application Center，BizTalk Server，Commerce Server，Exchange Server，Host Integration Server，Internet Security and Acceleration Server 和 SQL Server 等，而这些产品也不断升级来支持.NET 技术的开发。在编写.NET 应用程序的时候，还会需要数据库服务器、Web 服务器和邮件服务器等。在一个互连的、分布式的环境中，这样的企业级服务器产品是必不可少的。在使用.NET 技术来开发应用程序的时候，也完全可以采用其他厂商的服务器产品，如 Oracle，MySQL 等数据库，而不仅仅是由微软公司所提供的产品。

.NET 服务组件是指在.NET 平台中的关键性技术——Web 服务，全称为 XML Web Service。Web 服务作为一种全新的开发模式，是构建下一代互连网络的关键技术，也是.NET 技术中重要的组成部分。Microsoft .NET 平台提供了一些可供开发人员使用的 Web 服务组件。例如，用于用户验证的网络通行证 Passport 服务，开发人员只需要在自己的应用程序中使用这种服务，该程序就具有了身份验证等功能。另外，还有微软的日历服务、目录服务、搜索服务等 Web 服务组件，它们都可作为 Microsoft .NET 平台的组成部分。.NET 所构建的互连互通的远景必须借助于 Web 服务来实现，因为 Web 服务可以在不同的设备和平台上使用，真正实现无缝连通。同样地，第三方组件开发商也可以生产自己的 Web 服务组件来提供对.NET 平台的支持。

.NET 框架（.NET Framework）是.NET 平台的核心，是开发.NET 应用程序、运行.NET 应用程序的前提条件。.NET 框架提供了构建和执行应用程序及 XML Web Service 的基础，其统一的特性意味着所有的应用程序，无论 Windows 应用程序、Web 应用程序，还是 XML Web Service，都使用一套通用的工具和代码来开发，并且易于互相集成。.NET 框架由两部分组成：公共语言运行库（Common Language Runtime，CLR）和.NET 框架类库（Framework Class Library，FCL）。CLR 提供了类似于 Java 虚拟机（JVM）的功能，为.NET 下的编程语言提供可靠的、安全的编译环境，具有跨语言集成、内存管理、及时编译等功能。FCL 为.NET 平台下的开发提供了统一的编程模型，类库中有 7 000 多种类型，包括类、结构、接口、枚举和委托等，一些 FCL 的类包含了 100 多个方法、属性和其他成员。这么多的类和方法是通过命名空间来组织分类的。

Microsoft .NET Framework 2.0 版是 Microsoft .NET Framework 的更新，其中包括了运行使用.NET Framework 开发的应用程序时所需的全部内容。

Microsoft .NET Framework 2.0 版提供了改进的缓存，使用 ClickOnce 改进了应用程序部署和更新，通过 ASP.NET 2.0 控件和服务对最广泛的浏览器和设备提供更强大的支持，从而提高了可扩展性和其他性能。

Visual Studio .NET 是一个优秀的、全新的.NET 开发工具，它与以往的 Visual Studio 系列工具相比，有了质的飞跃。它内置支持 Visual Basic .NET、Visual C# .NET、Visual C++ .NET 和 Visual J# .NET 等多种语言，并且统一了所有的开发环境，拥有跨语言调试、XML Schema 编辑器等功能，第三方工具也可以作为插件集成在其中，Visual Studio .NET 现在比较流行的版本是 Visual Studio 2005。同样地，Visual Studio .NET 也不是开发人员唯一的选择，还有其他的开发环境可供选择，如 C# Builder，Web Matrix，Sharp Developer 等。

以上 5 个部分组成了 Microsoft .NET 平台，包括了底层操作系统、辅助产品、Web 服务、开发平台和集成开发工具。

1.3 .NET 框架的组成

图 1-2 所示为.NET 框架的组成。.NET 框架是.NET 平台的核心，主要由两部分组成：公共语言运行库（CLR）和.NET 框架类库（FCL），分别介绍如下。

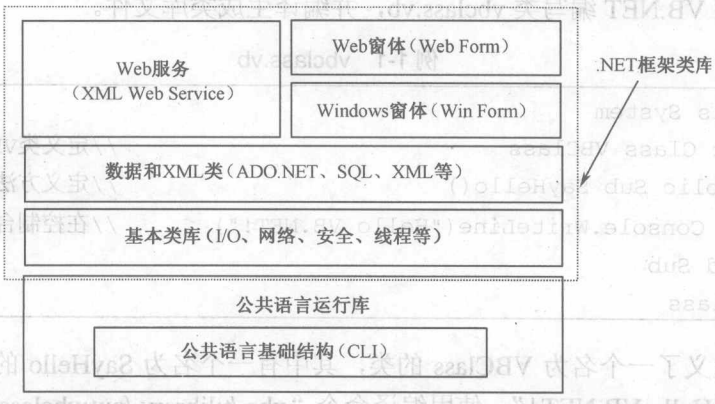


图 1-2 .NET 框架的组成

1.3.1 公共语言运行库

公共语言运行库（CLR）是.NET 平台下各种编程语言使用的运行时机制，是.NET 应用程序的执行引擎。在.NET 环境中，应用程序在编译的时候并不是编译成本地的机器代码，而是编译成中间语言（Intermediate Language，IL）代码，如同 Java 应用程序编译生成的字节码一样。但与 Java 应用程序不同的是，中间语言是一种实实在在的基于堆栈的编程语言，而 Java 的字节码不是。中间语言代码和元数据（Metadata）组成了托管模块，也就是程序集（Assembly）。这些托管模块必须借助于 CLR 中的即时编译器（Just-In-Time，JIT），并根据本地操作系统和硬件平台，来生成本地机器代码，该过程如图 1-3 所示。

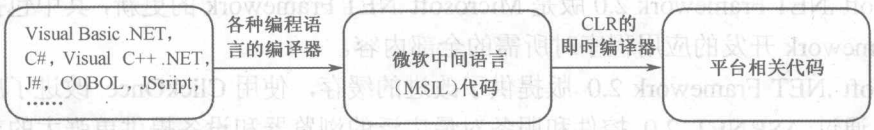


图 1-3 .NET 编程语言编译过程示意图

微软公司提供了公共语言规范（Common Language Specification，CLS）和通用类型系统（Common Type System，CTS），使得任何希望编写与 CLR 兼容的语言的公司都可以实现其愿望。只要遵循 CLS 和 CTS，将其原来的语言修改一番，就可成为支持.NET 开发的新的编程语言，这就大大增强了.NET 技术的生命力。现在可以支持.NET 开发的编程语言包括 APL，COBOL，Perl，Delphi 8 等，共有 25 种以上，而且其数量还将不断增加。

无论源代码用什么语言编写，最终都会被相应的编译器编译成为中间语言，所以在进入公共语言运行库时，其差异都不复存在了。以一个字符串为例，无论它在原来的编程语言中是以什么样的源代码表达的，它在程序内部的实现和在其他编程语言中的内部实现都是相同的，原因就在于，它们最终使用的都是公共语言运行库中的 String 对象。这样，不同编程语言就可以集成在一起，使用不同编程语言的开发人员就可以共同编写代码来开发同一个应用程序了。例如，在使用 C#语言编写的程序中可以继承使用 Visual Basic .NET（简称 VB.NET）语言编写的类，并调用其对象和方法。

【例 1-1】 同时使用 VB.NET 和 C#语言编写程序，体会.NET 跨语言集成的特点。

首先，使用 VB.NET 编写类 vbclass.vb，并编译生成类库文件。

例 1-1 vbclass.vb

```
Imports System
Public Class VBClass                              //定义类VBClass
   Public Sub SayHello()                          //定义方法SayHello()
      Console.WriteLine("Hello VB.NET!")          //在控制台中输出字符串
   End Sub
End Class
```

上述代码定义了一个名为 VBClass 的类，其中有一个名为 SayHello 的方法，它在控制台中输出字符串“Hello VB.NET!”。使用编译命令“vbc /t:library /out:vbclass.dll vbclass.vb”，将程序编译成名称为 vbclass.dll 的类库文件。

说明：编译 VB.NET 代码的命令为 vbc。其中，参数/t 表示将要编译成的程序类型，包括：/t:exe 创建控制台应用程序，/t:winexe 创建 Windows 应用程序，/t:library 创建类库文件等。参数/out 表示要指定的输出文件名称。

然后，使用 C#编写程序 csclass.cs 来继承类 VBClass。

例 1-1 csclass.cs

```
using System;                                     //引入命名空间System
using VBTest;                                     //引入命名空间VBTest
namespace CSTest                                  //定义命名空间为CSTest
```

```
{
    public class CSClass:VBClass          //定义类CSClass继承自VBClass
    {
        public void SayAgain()
        {
            Console.WriteLine("Hello C#!");
        }
    }
    public class MainClass
    {
        public static void Main()
        {
            CSClass obj=new CSClass();  //声明CSClass对象
            obj.SayHello();             //调用CSClass对象继承自基类的方法
            obj.SayAgain();             //调用CSClass对象自定义的方法
        }
    }
}
```

上述代码定义了一个名为 CSClass 的类，该类由 VBClass 继承而来，因此具有名为 SayHello 的方法。另外，该类中还有另一个方法 SayAgain，该方法输出字符串“Hello C#!”。代码中还定义了类 MainClass，这是程序的主类，具有静态（static）的 Main 方法。在 Main 方法中，声明 CSClass 类的对象 obj，并分别调用 obj 的 SayHello 方法和 SayAgain 方法。使用编译命令“csc /reference: vbclass.dll csclass.cs”生成名为 csclass.exe 的可执行文件，运行结果如图 1-4 所示。

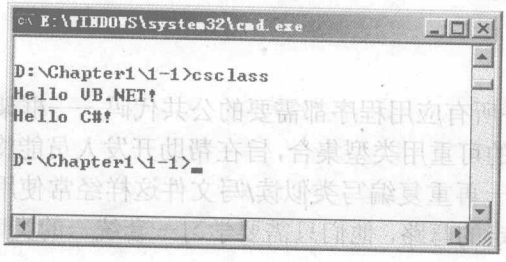


图 1-4 例 1-1 的运行结果

说明：编译 C#代码的命令为 csc。其中，参数/reference 表示从指定的程序集引用元数据。在本例中，“/reference: vbclass.dll”将由 vbclass.vb 生成的类库文件引用到本程序中，从而使用类 VBTest.VBClass。

.NET 在原理上具有跨越平台的特性。所有的编程语言首先要编译成为 IL，然后 IL 通过 CLR 的 JIT 即时编译成为本地代码。因此，IL 本身是与平台无关的，可以在装有 CLR 的任何一台计算机上运行。实际上，在编写出.NET 程序代码并编译完成后，就可以将它复制到装有 CLR 的计算机上运行了。目前的 CLR 只能运行在 Windows 系列操作系统下，但也有一些软件

组织将 CLR 移植到了其他操作系统中，其中比较知名的是 Mono 项目。

CLR 本身还有很多新的设计特性，如自动的内存管理——无用单元收集器（Garbage Collection, GC）。像 C++这样的编程语言，要求开发人员手工释放不再被使用的对象的内存，但是这样会导致下列两个问题。

（1）如果开发人员创建了某个对象并使用，但在使用完毕之后忘记了释放该对象，那么将造成内存泄漏，消耗掉进程的整个内存空间，最后导致进程崩溃。

（2）开发人员释放掉了某个对象，但是事后企图再次访问该对象的内存空间。有时候，组成该对象内存的物理介质中仍然包含着其他似是而非的数据，于是程序就会使用这些被破坏的数据继续运行，从而造成了程序的破坏。

在.NET 平台下，编程语言可以避免上述问题的出现。.NET 利用托管堆来对代码进行托管执行。托管堆是指被 CLR 保留的进程内存中的一部分，CLR 之所以保留一部分进程内存，其目的是为了进行自动内存管理。系统的线程对托管堆中的所有对象进行检测，如果对于某个对象的所有引用全部消失，则该对象成为无用单元（垃圾），同时该对象被移出托管堆。然后，托管堆中剩下的对象被压缩到一起，程序中的引用都指向新的位置。这样就解决了手动释放内存的问题，而且不需要开发人员编写任何代码。这个过程是 CLR 自动进行的，但也可以通过 System.GC.Collect 方法选择手工强制进行无用单元收集，增加了内存管理的灵活性。

CLR 还提供了安全性检测、异常处理、版本控制、代码调试等功能，因此，依靠 CLR 环境来运行的代码称为托管代码（Managed Code，MC）。托管代码具体是指，满足公共语言规范要求，所使用的元数据类型属于通用类型系统（CTS），并使用针对 CLR 的编译器开发的程序代码。但是，.NET 技术也允许开发人员使用非托管代码，来绕过 CLR 的某些服务或自行实现 CLR 的某些功能。托管代码和非托管代码之间可以相互调用。

另外值得一提的是，CLR 中的核心公共语言基础结构（Common Language Infrastructure，CLI）已经提交给国际标准组织 ECMA（European Computer Manufacturers Association），这意味着.NET 具备了开源的特征。

1.3.2 框架类库

.NET 框架提供了几乎所有应用程序都需要的公共代码——框架类库（FCL）。FCL 是微软公司开发的一个面向对象的可重用类型集合，旨在帮助开发人员能将精力集中在编写应用程序所独有的代码上，而不必一再重复编写类似读/写文件这样经常使用的功能代码。这就大大减少了开发人员在学习上所走的弯路，他们只需要学习一套统一的类库，而不是大量不同的应用程序接口（API）。

FCL 中有 7000 多种类型，提供了如下的内容：

- 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 界面具有的大部分功能，还包括验证控件、客户端缓存等 Web 独有的要素；
- Web 服务，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、数据命令、读取器、适配器、数据集、表格、记录、关系等；
- 类型反射，包括程序集、类和对象、方法、特性、字段等元数据类型信息的获取；
- 异常处理，包括系统、应用程序、数字运算、安全性限制等引发的异常。

上述这些类型依靠命名空间（Namespace）来管理。将多个提供相似功能的类，以分层的命名空间结构，组织在相关的单元中。例如，System.Data 命名空间包含了构成 ADO.NET 体系结构的类；System.Xml 命名空间是所有 XML 类的总体命名空间，这些 XML 类为处理 XML 提供了基于标准的支持；System 命名空间是.NET 框架中所有类型的根命名空间，System 命名空间还包含异常处理、无用单元回收、控制台 I/O、多种工具类型、格式数据类型、随机数生成器及数学函数等。表 1-1 所示为 FCL 中的常用命名空间。

表 1-1 FCL 中的常用命名空间

命 名 空 间	说 明
System	基础数据类型和辅助类
System.Collections	包含定义各种对象集合（如列表、队列、位数组、哈希表和字典）的接口和类
System.Data	ADO.NET 数据访问类
System.Drawing	生成图形输出（GDI+）的类
System.IO	包含允许对数据流和文件进行同步和异步读/写的类型
System.Net	封装网络协议（如 HTTP）的类
System.Reflection	读/写元数据的类
System.Runtime.Remoting	编写分布式应用程序的类
System.ServiceProcess	编写 Windows 服务的类
System.Threading	创建和管理线程的类
System.Web	支持 HTTP 的类
System.Web.Services	编写 Web 服务的类
System.Web.Services.Protocols	编写 Web 服务客户端的类
System.Web.UI	ASP.NET 使用的基础类
System.Web.UI.WebControls	ASP.NET 服务控件
System.Windows.Forms	GUI 应用程序的类
System.Xml	读/写 XML 数据的类

开发人员也可以通过扩展类库中的类来定义自己的类，还可以定义唯一的专有命名空间来区别于微软或其他开发人员开发的类。例如，在例 1-1 中引入命名空间 System，其目的是引入.NET 框架类库中的基础类型和辅助类，这样才可以使用 Console 类的 Write 方法在控制台中输出字符串。在 vbclass.vb 中，使用关键字 Namespace 来定义自己的命名空间 VBTest，这样类 VBClass 就存在于命名空间 VBTest 下面了，完整的写法是 VBTest.VBClass。所以，在 csclass.cs 中要使用类 VBClass，就需要引入其命名空间 VBTest。在 C#中，引用命名空间的语法是 using VBTest。

在.NET 程序开发中，可以使用多种遵循公共语言规范（CLS）的编程语言，其中 C#，Visual Basic .NET，Visual C++ .NET 和 Visual J# .NET 是使用较多的几种语言。C# 是微软公司唯一一种从一开始就专门针对 CLR 进行设计的语言，微软公司本身已经使用 C# 来创建诸如类库和

ASP.NET 等子系统中的托管代码。C# 语言是从 C 和 C++发展而来的，它吸取了 C，C++，Java 等多种语言的精华，是一种简洁、完备、类型安全和完全面向对象的高级程序设计语言，本书将主要以 C# 语言为例进行讲解。Visual Basic .NET 与以前的 Visual Basic 版本相比有了很大的改变，使用 CLR 和类库来取代类似的 Visual Basic 组件和插件。Visual Basic .NET 还有高级的新功能，如对多线程和结构化异常处理的支持。Visual C++ .NET 是 Visual C++经过修改和更新之后的版本，可以支持.NET 开发平台。但 Visual C++ .NET 仍然具有本机代码编译器，可以开发托管代码和非托管代码，如开发底层设备驱动程序这样的应用。Visual J#使开发人员可以使用 Java 语言的语法编写应用程序，但最终使用 CLR 而不是 JVM 来运行。对于那些使用 Visual J++工作并且熟悉 Java 语言的程序员，如果要转换到.NET 开发平台，那么 Visual J#. NET 是非常有吸引力的编程语言。

1.4 .NET 的特点

.NET 技术作为微软公司最新的技术，融合了目前计算机行业中最流行的各种开发技术的优点，弥补了它们的不足，具有鲜明的特点。

1. 基于组件的技术

组件技术始于微软公司的组件对象模型（Component Object Model，COM）。COM 是在系统内创建的工作良好的高聚合模块，并且开发人员可以通过分离不同的任务模块降低组件的耦合度。由于 COM 具有强大的功能和跨语言集成等优点，因而它受到了广大程序员的青睐。但是，COM 的开发是极其复杂的，开发人员需要了解系统接口和注册表等。.NET 也是一种基于组件的技术，但是相对于 COM 而言，编写.NET 组件要简单得多，开发门槛也低得多。

2. 跨语言集成

目前支持.NET 平台开发的编程语言超过了 25 种，并且这个数量还在不断地增加。通过例 1-1 可以看到，多种语言是可以集成在一起使用的。与 COM 不同，这种跨语言的集成是在 IL 层次上面的集成，可以调试抛出的异常，可以扩展相应的功能，而这些是在二进制基础上进行跨语言集成的 COM 所不能做到的。

3. 简化开发过程

虽然在.NET 平台上有多种语言可以用来开发.NET 应用程序，但是这些语言用到的类库只有一个，即框架类库（FCL）。不同的语言在一些语法细节上面有差异，但在调用类库的方法时都是相同的，而且在 Visual Studio .NET 这样的开发环境中，不同语言所对应的开发环境也是一模一样的。这样就减少了在不同开发语言之间转换时所需要的学习适应时间，提高了开发效率。

4. 简化部署

在 Windows 平台上，曾经困扰大多数程序员的一个问题就是著名的“DLL Hell”问题，即在部署应用程序的时候可能造成的重名 DLL 文件的覆盖，这会给系统或应用程序带来不可预测的危害。但是在.NET 中，采用了版本协调等手段，避免了重名 DLL 文件的覆盖问题，也就避免了“DLL Hell”。而且，由于.NET 应用程序在部署的时候不依赖于操作系统的注册表，因此可以做到“安装、卸载零影响”。只要将应用程序所在文件夹复制到目的文件夹中就可以使用，将其删除就做到了卸载。