微软认证高级技术培训教材系列

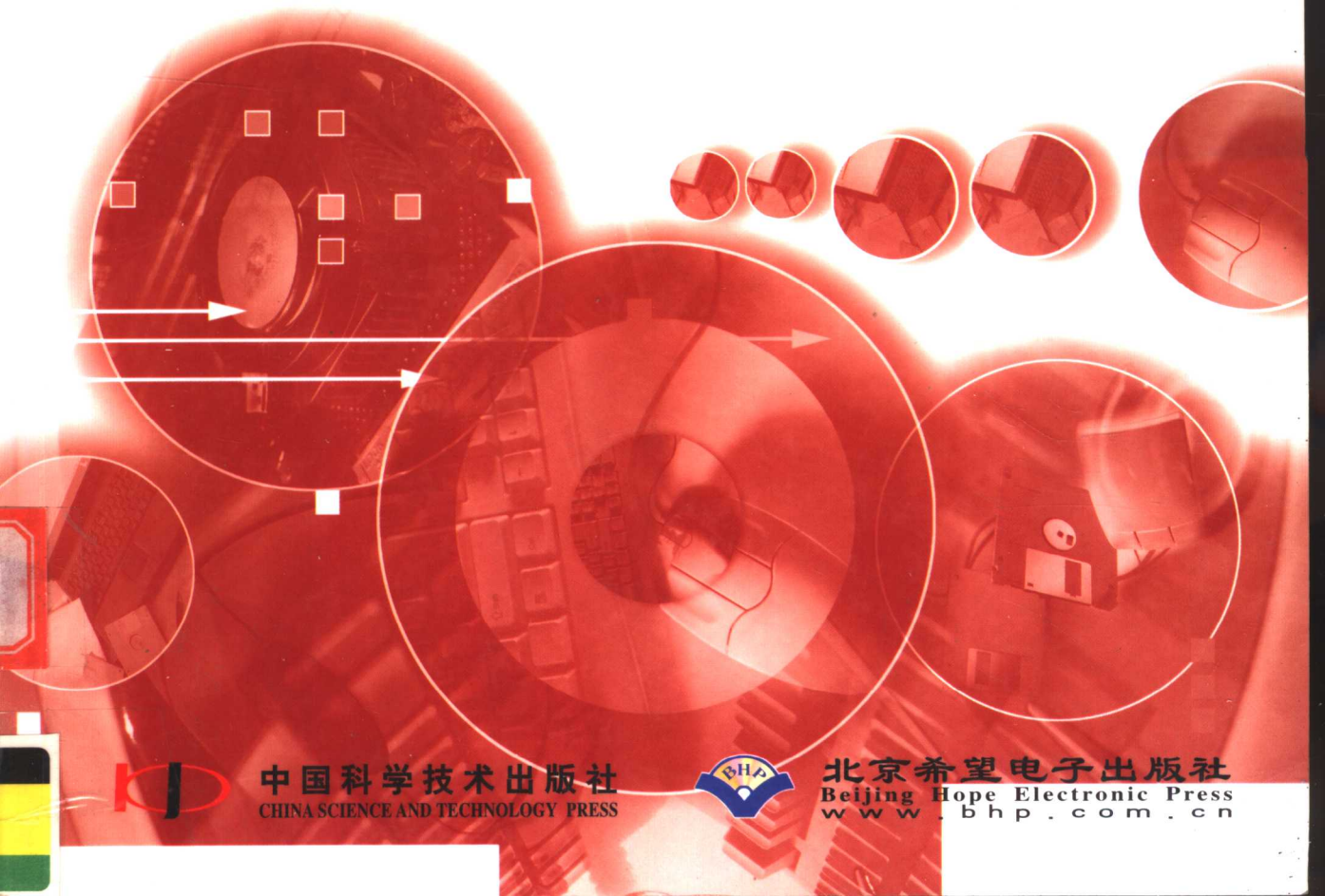
考试号：70-320

MCSD/MCAD .net 认证

使用 Microsoft Visual C# 开发 XML Web Services 和 Server Components

北京希望电子出版社　总策划

王瑄　李燕　编写



中国科学技术出版社
CHINA SCIENCE AND TECHNOLOGY PRESS

北京希望电子出版社
Beijing Hope Electronic Press
www.bhp.com.cn

微软认证高级技术培训教材系列

考试号：70—320

MCSD/MCAD .net 认证

使用 Microsoft Visual C# 开发 XML Web Services 和 Server Components

北京希望电子出版社　总策划

王瑄　李燕　编写

中国科学技术出版社
CHINA SCIENCE AND TECHNOLOGY PRESS

北京希望电子出版社
Beijing Hope Electronic Press
www.bhp.com.cn

图书在版编目（CIP）数据

使用 Microsoft Visual C#开发 XML Web Services 和 Server Components / 王瑄、李燕编写. —北京：中国科学技术出版社，2003.10
（微软认证高级技术培训教材系列）
ISBN 7-5046-3659-2

Ⅰ.使… Ⅱ. ①王… ②李…Ⅲ.C 语言—程序设计—技术培训—教材 Ⅳ.TP312

中国版本图书馆 CIP 数据核字（2003）第 077896 号

系　列　书：微软认证高级技术培训教材系列
书　　名：使用Microsoft Visual C#开发XML Web Services和Server Components
总　策　划：北京希望电子出版社
文本著作者：王瑄 李燕
责任编辑：栾大成 周晓慧
出版、发行者：中国科学技术出版社 北京希望电子出版社
地　　址：北京市海淀区中关村南大街 16 号 100081
北京市海淀区知春路甲 63 号卫星大厦 三层 100080
网址: www.bhp.com.cn E-mail: kobe@bhp.com.cn yb@bhp.com.cn
电话: 010-62520290, 62521724, 62528991, 62630301, 62524940, 62521921, 62521724（发行） 010-82675588-318,62532258,62562329（门市）
010-82675588-501,82675588-201（编辑部）
经　　销：各地新华书店、软件连锁店
排　　版：希望图书输出中心 孙红
印　刷　者：北京双青印刷厂
开本 / 规格：787 毫米×1092 毫米 1/16 24.75 印张 581 千字
版次 / 印次：2003 年 12 月第 1 版 2003 年 12 月第 1 次印刷
印　　数：1～5000 册
书　　号：ISBN 7-5046-3659-2/**TP•204**
定　　价：40.00 元

内容简介

本书是微软认证高级技术培训教材系列之一。对应考试号：70-320。

由于本书的主要目的是辅助读者通过微软认证的70-320考试，利用VS.NET开发.NET中各种令人激动的应用程序将是本书的主要内容。也可以作为XML Web Services和.NET Framework的快速入门教材。

本书共有9章，分别讲述：.NET概述，XML基础，读写XML，开发Windows Services，Serviced Components，XML Web Services，.NET Remoting，ADO.NET，调试、部署和安全性。另外为了便于读者理解，读者可以在附录的词汇表中查到。书中我们介绍了一些使用经验和心得，难免有不当之处，或者还有更好的方法，欢迎赐教。如果有需要交流的地方，请发email或访问我们的网站www.xmlchina.net。

需要本书或需要得到技术支持的读者，请与北京中关村083信箱北京希望电子出版社（邮编100080）联系，电话：010-62630301，82675588(总机)，传真：010-62520573，E-mail：kobe@bhp.com.cn

出版说明

为了配合微软高级技术培训的教学与考试，进一步推广微软认证应用程序开发专家（MCAD）、微软认证解决方案开发专家（MCSD）的培训和考试，特组织优秀的微软认证讲师（MCT）和前沿程序员编写了本套微软认证高级技术培训教材。

全套教材共 6 本：

序号	书　名	对应考试号
1	使用 Visual Basic.NET 开发 Web 应用	70-305
2	使用 Visual Basic.NET 开发 Windows 应用	70-306
3	使用 VB.NET 开发 XML Web Services 和 Server Components	70-310
4	使用 Visual C#.NET 开发 Windows 应用	70-315
5	使用 Visual C#.NET 开发 Web 应用	70-316
6	使用 Visual C#开发 XML Web Services 和 Server Components	70-320

本套培训教材都由第一线的微软认证高级技术培训中心讲师（MCT）和走在技术最前沿的程序员编写，凝聚了多年教学和开发经验，符合中国人阅读习惯，教材的每章都有学习重点，在必要章节附有实验，供学员练习。本套教材还包括大量的模拟试题，所有模拟试题都加入了试题分析和知识点解析以适应不同考生考证使用。力求通过学习本套教材，即可通过 MCSD，MCAD 考试。

有两种 MCSD/MCAD 认证，分别对应不同的开发语言：VB 或 C#

- VB

学员通过学习课程 1，2，3 并通过对应考试，可以获得 MCAD 证书（除此以外，通过 70-300 考试即可获得 MCSD 证书）；

- C#

学员通过学习课程 4，5，6，通过对应考试，可获得 MCAD 证书（除此以外，通过 70-300 考试即可获得 MCSD 证书）。

注：70-300:《为 Microsoft .NET 分析需求和定义解决方案架构》目前不在本系列教材之中，请参考相关教材。

本套教材既可作为微软认证高级技术培训教材，微软高级技术认证的自学教材，也可供广大网络、数据库技术人员和爱好者学习、参考使用。

编　者

前 言

伴随着 Web 的出现，Internet 一直在高速地发展着，我们也习惯了利用 Internet 获得或者传递各种信息。每个人都相信 Web 的发展空间仍旧广阔，但是随着.com 神话的泡沫破灭，人们认识到应该将主导权回归到信息需求的实质：企业、消费者及应用开发者。而 Microsoft 的目标就是提供能够实现 Web 新梦想的远景及技术。

Microsoft 的野心和决心在正在进行中的.NET 之役中体现得淋漓尽致。.NET 是 Microsoft 最新的软件战略，三年前，当盖茨首次公开阐述.NET 平台的设想时，立即引起了整个业界的高度重视。三年来，随着一个又一个重量级.NET 产品的投入使用以及铺天盖地的宣传攻势，人们不得不相信：Microsoft 再一次抓住了问题的实质。

Microsoft 认为在 Internet 高度发达的将来，服务而非软件将成为 IT 公司向客户提供的产品。可以想到的可能的服务包括：评价某公司的信用级别、提供某个地理坐标的精确卫星图像、复杂数学公式的计算等等。通过向目标客户提供自身业务逻辑的编程接口，IT 公司可以实现从产品供应商到服务供应商的质的飞跃。

软件服务的概念由来已久，当我们调用一个远程服务器上的 COM+组件所提供的功能时，其实就已经享受了某种软件服务。但是，复杂性和通信上的诸多限制使 COM+或者与之抗衡的 CORBA 和 RMI 均无法成为 Internet 上通用的解决方案。因此，XML Web Services 架构成了 IT 界统一的希望和目标。一个 Web Srevice 是运用新的技术在逻辑上为其他应用程序独立提供数据和服务的应用程序。在本书的正文中，你将会深入的了解：为什么 XML Web Services 具有足够的优势能够成为 Internet 上统一的解决方案。

.NET 就是 Microsoft 的 XML Web Services 平台。以 XML 为基础的.NET 平台整合了计算机与通讯的双重优势，它将衍生出新一代的网络服务，使人们可随时、随地利用任何设备调用相关的 Web 服务以获取他们需要的信息。

同时，.NET 也将 Microsoft 提升至另一新的层次，使 Microsoft 具有了更强的竞争力。.NET 平台可以看作由以下几部分组成：

- 基础操作平台

  比如操作系统 Windows、Web 服务器 IIS、Exchange Server 等，这些都是在计算机上实现 Web 服务所不可或缺的必要部分；

- 面向个人或企业的 Web 服务

  除了提供基础技术，Microsoft 也致力于构建自己的 Web 服务，比如目前正在使用的.NET Transport。你可以选择使用业界公开发布的 Web 服务（有些免费，有些则是要付费的），也可以自己构建一个。在第六章，我们将学习如何创建自己的 Web Service；

- .NET 开发环境或工具

  .NET Framework 是 Microsoft 的.NET 计划的根本体现，主要包括两部分：Common Language Runtime（CLR，公共语言运行时）和强大的类库（Class Library），我们

将在第一章详细介绍.NET Framework 的方方面面。

Visual Studio.NET 则是 Microsoft 为开发人员提供的开发工具，简化了大量的编码工作。利用 VS.NET，你可以高效的开发出各种.NET Framework 下的应用程序：不止是 Web Services，还包括 Windows Form、Windows Service、控制台应用、Web 应用、控件等。

Microsoft 还提供了免费的.NET Framework SDK，包括所有的语言编辑器、部分工具和文档。虽然缺乏可视化操作，使用 SDK 也可以在没有 VS.NET 的情况下开发出.NET Framework 的应用程序。

由于本书的主要目的是辅助读者通过微软认证的 70-320 考试，利用 VS.NET 开发.NET 中各种令人激动的应用程序将是本书的主要内容，由于涉及内容比较多，因此无法针对每一部分做深入的探讨。当然，你也可以把本书作为 XML Web Services 和.NET Framework 的快速入门教材。

在本书中，为了便于读者理解，对于可能产生歧义或有多种中文翻译版本的关键词汇，在其第一次出现的地方，我们一律标出了英文原词，读者也可以在附录的词汇表中查到。

书中我们介绍了一些使用经验和心得，难免有不当之处，或者还有更好的方法，欢迎赐教。如果有需要交流的地方，请发 email 或访问我们的网站 www.xmlchina.net。

感谢希望出版社和中国科学技术出版社感谢编辑栾大成先生，没有他的认真和宽容的态度，本书无法完成；同时还要感谢在编写本书过程中给予我们帮助的同事和朋友：倪凯、金殷勇、柴晓路、孙涛、梅雪峰、韦超、关键等。

王瑄、李燕 2003 年于上海

目 录

第 1 章　.NET 概述

- .NET Framework 简介
- C#语法简介
- .NET 程序的基础——Assembly（程序集）
- .NET 中的四种组件模型简介

在本章中，我们将介绍 Microsoft 的.NET 战略并学习.NET Framework 作为.NET 的运行环境是如何工作的。

注意： 本章所学习的内容将在接下来的章节中一一展开，希望通过本章的学习，你能够对.NET 有一个整体认识。

如果你不是一个使用 Microsoft 开发技术的新手，那么对 Visual Basic 和 Visual C++这样的开发语言不会感到陌生，并会津津乐道于在开发桌面级应用程序时这些语言带给我们的便捷和好处。配合以 COM 和 DCOM 技术，我们甚至能用它们开发分布式的、可重用的企业级的应用程序。虽然存在诸如 DLL Hell 这样的令人头痛的问题，编写的代码也不能方便地在不同种类的平台上进行移植。

我们本可以继续满足于现有的编程体系，而无需把有限的精力投入到新技术的研究和学习中。但是，一切都因为 Internet 的前进而改变了。

每个人都期望通过访问 Internet 得到和发布某些信息，Internet 改变了世界运行的方式，它的前进不可阻挡。随着人们的生产、生活方式与 Internet 连接得越来越紧密，将原来局限于 intranet 内部的分布式应用移植到 Internet 上的需求越来越迫切，应用程序通过 Internet 访问远程服务器上的组件将成为必然的需求，但是，这种实现显得困难重重：

- Internet 上运行组件的服务器可谓千差万别，各种硬件和操作系统的组合构成了无数种组件的运行环境，加上编写组件的编程语言也不下几十种，我们如何保证应用程序可以顺利的调用对方的组件？
- DCOM 和类似技术，如 CORBA 和 Java RMI，原本都仅限于 Intranet 内部所安装的程序和组件，它们都使用自身默认的专有协议进行通信，一旦移植到 Internet，且不说这些专有协议与 Internet 协议的兼容性，单单设想，假如我们编写的应用程序需要调用使用不同技术构建的组件，就不得不为之编写不同的专有协议，这就够让人为难了；
- 出于安全原因的考虑，绝大多数的网管采用路由器和防火墙禁止了除 80 端口通信以外的一切类型的 Internet 通信，这样做就意味着对 DCOM 及其类似技术关上了大门。
- DCOM 及其类似技术本质上都是面向连接的，不能很好地处理网络中断，在便于控制的 Intranet 环境中，我们当然可以通过保持网络的良好运行解决这一问题，但是，Internet 并不在任何组织和个人的掌控之下，所以不能对连接的可靠性和质量

做任何假设。如果发生网络中断，很可能导致采用以上技术的系统的崩溃。

Microsoft 显然意识到了以上问题，秉承其向下兼容的一贯作风，Microsoft 对现有技术进行了改进和扩展，从而得到了易学易用的 Active Server Pages（ASP），也得到了 COM 技术的扩展——COM+，我们甚至得到了可以通过 80 端口在应用程序和远程组件间建立 DCOM 连接的 COM Internet Services（CIS）技术……；但是，在“喘息”着接受新鲜事物的同时，我们不仅感受到了 ASP 的非面向对象编码带来的可怜的维护性和效率低下，感受到了 CIS 的非标准化，同时，还切身感受到了 Microsoft 演进其技术所带来的最大缺点：整个体系日益的复杂和凌乱。

当面对似乎是一夜之间扑面而来的 Microsoft 有关.NET 的宣传时，我们意识到：我们终于有了一个新的开端！.NET 的最终目标是让应用程序可以通过 Web Services 架构从 Internet 上的任何设备存取和访问信息，由于现存应用的多样性和复杂性，实现这一目标还有待时日，但是.NET 的确为我们提供了一系列令人激动的东西来解决程序开发中那些令人烦恼的问题。

为了实现.NET 的宏伟蓝图，Microsoft 开发了.NET Framework。

学习目标

- 了解.NET Framework 诞生的原因
- 了解.NET Framework 的基本组成部分和它们之间的关系
- 掌握 C#的基本语法
- 了解 Assembly（程序集）的作用和结构
- 了解.NET 中有哪些组件模型

1.1 .NET Framework

写好一份程序，到处可以执行（Code Once，Run Anywhere），是程序员梦寐以求的愿望，.NET Framework 提供了实现这一愿望的可能性。

如图 1.1 所示，.NET Framework 由通用语言运行时环境（Common Language Runtime）、基础类库（Base Class Library）以及各种帮助我们创建 Web 应用程序（ASP.NET）和 Windows 应用程序（Windows Forms）的语言和工具组成。

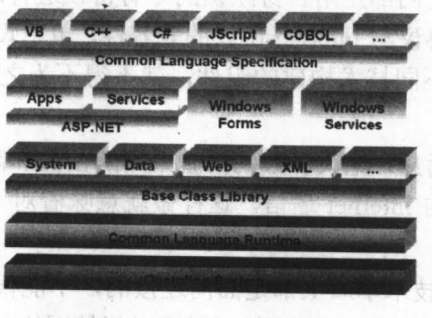


图 1-1 .NET Framework

图中提到的编程语言，除了 C#以外，其余都是大家耳熟能详的老朋友。所有这些语言都能够独立地创建 Web Services、Web 应用程序和 Windows 应用程序。它们都可以利用丰富而强大的基础类库。基础类库包括了从输入输出到数据访问等各方面的内容，提供了一个统一的面向对象的、层次化的、可扩展的编程接口。编制完成的代码将运行于 CLR（Common Language Runtime，通用语言运行时）环境。这样，使用任何符合 CLS（Common Language Specifications，通用语言规范）的语言开发的程序，都可以在任何支持 CLR 的系统下运行，而无需考虑该系统的硬件和操作系统。

1.1.1　Common Language Runtime

基于 Common Language Runtime 开发的代码称为受控代码，它的运行步骤大体如下：

首先使用一种 Common Language Runtime 支持的编程语言编写源代码→然后使用针对 Common Language Runtime 的编译器生成独立于机器的中间语言（Microsoft Intermediate Language，MSIL），同时产生运行所需的元数据→在代码运行时再使用即时编译器（Just In Time Compiler）生成相应的机器代码来执行（如图 1-2）。

说明： 摒弃了传统开发环境中由程序开发语言的编译器直接将程序编译为机器代码的做法，转而采用由 CLR 在运行时即时产生机器可执行代码的方式是为了避免传统的开发环境中对硬件和操作系统的依赖。

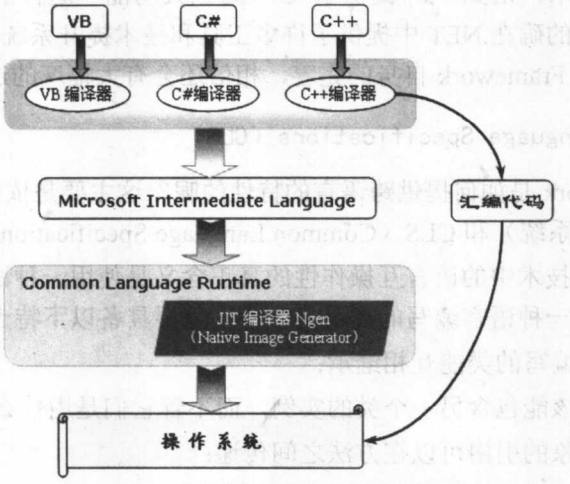


图 1-2　基于 CLR 的代码的运行步骤

Common Language Runtime 的优点在于：

- 简化开发并可综合不同的开发语言　不再需要使用 IDL 来描述组件和接口，无需理会 IUnknow。在 CLR 下，不同语言编写的应用程序可以方便地互相调用。
- 安全地部署和执行　无论程序使用哪种.NET 开发语言开发，无论程序在哪种硬件和操作系统下运行，CLR 都可以通过 IL（中间语言）了解程序所要执行的功能，同时，CLR 会自动检查程序是否有执行权限，这样，就确保了这个程序无法对系统或其他应用程序造成伤害。

- 自动对象管理　CLR 的 Garbage Collector（垃圾回收）功能负责内存资源的自动回收，这意味着，所有.NET 对象所占用的内存都由 CLR 统一处理，无需开发人员操心，这就避免了由于程序员的疏漏而造成的内存泄漏（Memory Leak）和乱指的指针。
- 使用程序集（Assembly）　对 DLLs 技术进行了提升。

说明： Microsoft Visual Studio.NET 中文版联机文档中将 Assembly 翻译为“程序集”，虽然这个翻译和 Assembly 一词的英文意思相去甚远，但是较为符合其实际角色，因此本书在必须使用其中文名称的地方将沿用该翻译。

简单来说，Assembly 是编译好的、面向 MSIL 代码的程序集合，类似于以前的 DLL 和可执行文件。但与它们不同的是，Assembly 不是一个物理单元，而是一个逻辑单元，换句话说，一个 Assembly 可以分布在多个文件中。

Assembly 是自我描述单元，它不仅封装了程序的功能代码，也封装了有关此应用程序的所有信息，包括标识、版本等内容。很多人都遇到过称为 DLL Hell 的麻烦，引起这种麻烦的原因是程序因为某个新安装的程序更新了它所使用的组件而不能正常运行。Assembly 的自描述特性和其强大的命名机制可以完美地解决这一问题。

- 提高系统的效率和可维护性　效率和可维护性是程序界永恒的话题，虽然，就目前情况看来，诸如“C#提供了 C++的强大功能”这样的宣传稍嫌夸张，但是，Microsoft 的确在.NET 中提供了许多工具和技术提升系统的效率和可维护性，并且随着.NET Framework 自身的完善，相信还会有大幅度的进步。

1.1.2 Common Language Specifications（CLS）

.NET Framework 是如何提供跨语言的特性的呢？这主要是依赖于 CTS（Common Type System，公共类型系统）和 CLS（Common Language Specifications，公共语言规范）。

面向对象编程技术中的语言互操作性的真正含义是使用一种语言编写的 class（类）应该能直接与使用另一种语言编写的类通信。特别是要具备以下特点：

- 不同语言编写的类能互相继承；
- 一个类应该能包含另一个类的实例，而不管它们是用什么语言编写的；
- 对象或对象的引用可以在方法之间传递；
- 可以在调试环境中调试不同语言编写的源代码。

这是一个雄心勃勃的目标，但令人惊奇的是，Microsoft 在.NET 中已经实现了这一目标！Visual Studio.NET 为所有的.NET 语言提供了统一的开发和调试环境，而.NET 语言所遵循的 Common Language Specifications 则提供了语言互操作的特性。

Common Type System（CTS）的真正含义是用 IL（中间语言）定义的一组.NET 运行库所能理解、并且随后.NET 应用程序可以使用的一系列数据类型的集合。

注意： 并不是所有.NET 语言都支持 CTS 中定义的所有类型。

Common Language Specifications（CLS）是所有.NET 语言都支持的 CTS 的一个子集。数据类型的统一保证了使用不同.NET 语言编写的应用程序之间的互操作性。

任何语言只要符合 Microsoft 制定的 CLS 标准，就可以轻易整合进.NET Framework，享用.NET Framework 的资源。

CLS 已经被提交给 ECMA 进行标准化，这显示了 Microsoft 在自己的.NET Framework 中统一语言规范的决心和努力。除了 Visual Studio.NET 提供的几种.NET 语言外，已经有包括 Delphi 在内的多种编程语言拥有了兼容 CLS 的版本，从而加入了.NET 语言的行列。

1.1.3 Base Class Library（BCL）

.NET 的 Base Class Library（基础类库）包括了从输入输出到数据访问等各方面的服务，提供了一个统一的面向对象的、层次化的、可扩展的编程接口。在.NET 中，属性相同的基础类被包含到同一个 namespace（名称空间）中。在名称空间中，最底层也是最常用的名称空间是系统名称空间（System），在系统名称空间中包含的类均为基类，作用上类似于 Cobject 类在 MFC 中的作用。另外，系统名称空间还包括了异常处理、垃圾回收、控制台输入输出等一系列的重要功能特性。总之，想要利用.NET Framework 的任何特性，都需要和名称空间及其所拥有的类进行交互。开发人员还可以在现有名称空间的基础上派生自己的类和名称空间。

有了.NET 强大的基础类库的支持，使用.NET 语言的程序员可以以更高的效率、更快的速度、更好的模式开发应用程序。

1.1.4 ADO.NET

几乎所有的应用程序都需要访问从简单的文本文件到大型的关系型数据库等各种类型的数据源。在.NET Framework 中，实现数据访问功能的是 ADO.NET 技术。ADO.NET 提供了一系列连接到数据库、运行 SQL 命令、返回记录集的类库供程序员使用。虽然名为 ADO.NET，但其与前任 ADO 的关系似乎并不大。与 ADO 相比，ADO.NET 的革命性变化主要体现在：

- 对 XML 的强大支持　这本身就是 ADO.NET 最主要的设计目标。在 ADO.NET 中通过使用 XMLReader、XMLWriter、XMLNavigator 等可以方便地创建和使用 XML 数据，ADO.NET 支持 W3C 的 XSLT、XSD、DTD、XDR 等标准。其对 XML 的支持也为 XML 成为.NET 中数据交换的统一格式提供了基础。
- 离线的数据操作技术　ADO.NET 引入了数据集（DataSet）的概念，这是一个存在于本地内存中的数据的关系型视图。应用程序先对本机上的数据集进行数据的添加、删除或修改，然后将其更新回真正的数据来源。同时，无论数据来源于一个关系型数据库，还是来源于一个 XML 文档，我们都可以用一个统一的编程模型来创建和使用它，这样就提高了程序的交互性和可扩展性，尤其适合于分布式系统应用。
- 多种数据操作方式　ADO.NET 中引入了一些新的数据操作方式，如 DataReader，使用它可以产生一个只读的记录集，从而高效读取数据。针对不同场合，我们可以选择不同的数据操作方式，以期达到系统的最优化。

总之，ADO.NET 与 XML 紧密结合，提供了一系列新对象和编程模型，使得在.NET

Framework 中的数据操作十分简便而高效。

1.1.5 ASP.NET

ASP.NET 是 Microsoft.NET 中的网络编程结构，它使得建造、运行和发布网络应用非常方便和高效。

ASP.NET 的设计目的就是使得开发者能够非常容易地创建网络表单，它把 Visual Basic 中的快速开发模型引入到网络开发中来，从而大大简化了网络应用的开发。具体地说：

- 在 ASP.NET 中可以支持多种语言，不仅仅支持脚本语言，通用语言运行时支持的所有语言在 ASP.NET 中都可以使用；
- 代码和内容分开，在现在的 ASP(Active Server Pages)开发中，内容和脚本交错，维护和升级很困难，将他们分开可以使得开发人员和设计人员能够更好地分工合作，提高开发效率；
- 在 ASP.NET 中通过引入服务器端控件，将类似 Visual Basic 的快速开发应用到了网络开发中来，这样大大提高了构建网络表单效率，并且服务器端控件是可扩展的，开发者可以建造自己需要的服务器端控件。

ASP.NET 应用不再是解释脚本，而是编译运行，再加上灵活的缓冲技术，从根本上提高了性能；由于 ASP.NET 的应用框架基于通用语言运行时，发布一个网络应用，仅仅是一个拷贝文件的过程，即使是组件的发布也是如此，更新和删除网络应用，可以直接替换/删除文件；开发者可以将应用的配置信息存放在 XML 格式的文件中，管理员和开发者对应用程序的管理可以分开进行；ASP.NET 还提供了更多样的认证和安全管理方式；在可靠性等多方面都有很大提高。

1.1.6 Web Services 与 ASP.NET

Web Services（Web 服务）是下一代可编程网络的核心，它实际上就是一个可命名的网络资源，可用来在 Internet 范围内方便地表现和使用对象，就像使用今天的 COM 对象一样，不同的是使用和表现 Web Services 是通过 SOAP（简单对象访问协议）甚至 HTTP 来实现的。在 ASP.NET 中，建造和使用 Web Services 都非常方便。

在 ASP.NET 中构造 Web Services 就是编写一个后缀为.ASMX 的文件，在这个文件中加入想要发布出来的方法就可以了，Web Services 的建造者不需要了解 SOAP 和 XML 的细节，只需要把精力集中在自己的服务本身，这也为独立软件服务开发商提供了很好的机会；使用 Web Services 最简单方式就是使用 HTTP 协议（HTTP GET 或 HTTP POST），用户只需要直接访问 Web Services（ASMX 文件）的 URL 即可；当然用户还可以通过 SOAP 在自己的应用中更灵活地使用 Web Services。

1.2 Microsoft Visual C#

C#是 Microsoft 为.NET Framework 量身定做的新程序语言。它与 C++和 Java 是如此的相象，以至于原先使用这两种语言的程序员可以很容易地使用它。同时，原先 C++中难以学习或容易引发错误的功能，如指针（Pointer）、宏（Macro）、Template 及多重继承等，在

C#中已不复存在。

C#和.NET Framework 是密不可分的关系。C#的数据类型就是.NET Framework 所提供的数据类型。C#语言本身并无类库，而是直接使用.NET Framework 提供的类库。此外，诸如自动资源回收（Automatic Garbage Collection）、类型安全检查（Type-safety Verify）、结构化错误处理（Structured Exception Handling）等操作，C#都是交给 Common Language Runtime 处理。

Microsoft 对 C#寄予厚望，似乎有意将其作为构建.NET Framework 上的应用程序的首选语言。

1.2.1 第一个 C#应用程序

依照程序语言学习的传统，我们编写第一个 C#程序——Hello World：

使用任何文本编辑器，填入以下代码并将其保存为文件 Hello.cs（.cs 是 C#代码文件的默认后缀）：

```
using System;
class Hello
{
    public static int Main()
    {
        Console.WriteLine("Hello World!");
        return 0;
    }
}
```

为了运行以上代码，需要将它们编译成 Microsoft Intermediate Language（IL，中间语言），这部分工作由 C#编译器完成。其实，在 Visual Studio.NET 提供的 IDE 界面中，编写、编译和调试代码是非常轻松的事情，但是，本章为了尽量清晰地讲解 C#语言的概念，将坚持使用记事本这样的文本编辑器和命令行界面逐步完成以上工作。

1. 如果你已经安装了 Microsoft Visual Studio.NET，那么请在菜单 Start→Program→Microsoft Visual Studio.NET→Visual Studio .NET Tools 中选择 Visual Studio .NET Command Prompt（命令行窗口），打开命令行。
2. 在命令行中键入：csc Hello.cs，编译器开始工作，并在显示了一系列编译器版本信息后结束编译。
3. 这时，我们可以在代码文件 Hello.cs 所在目录下找到一个名为 Hello.exe 的可执行文件，运行它，我们将看到熟悉的 Dos 界面下的 Hello World 问候。

如果你是一个 C++或 Java 程序员，对以上代码不会感到陌生，惟一需要注意的地方是代码的第一行（using System;）；这行代码的作用实际上是向 CLR 系统报告所要使用类库的资源，在这里，using 类似于 Java 中的 import 关键字。

如果你是一个 Visual Basic 或 ASP 程序员，很可能会对以上代码感到有些手足无措，这是由于对面向对象编程方法的生疏造成的，不用着急，只要稍微了解面向对象的特性之后，一切疑惑也就迎刃而解了。那么让我们花一点时间熟悉一下面向对象的程序开发。

1.2.2 面向对象的程序开发

面向对象＝对象＋类＋继承＋通信

如果一个应用程序是使用这样四个概念来设计和实现的，我们就认为这个应用程序是面向对象的。

- 对象（Object） 对象是面向对象开发方法的其本元素。每个对象都可以用它本身的一系列属性和其上的一系列操作来定义。对象可以是现实生活中的一个物理对象，也可以是某一类概念实体的实例（Instance）。比如，一辆汽车，一个人，一台电脑，乃至一种语言，一个图形都可以作为一个对象。
  从分析和设计的角度来看，对象表示了一种概念，它把现实世界的实体模型化。实体的有关声明有：描述实体，包括实体的属性和可执行的操作。例如对于电脑这个对象，它的型号、重量等是其属性，而开机、关机、待机则是它可以执行的操作。
- 类（Class） 类是一组具有相同属性（也可以理解为数据结构）和相同操作的对象的集合。类是对一系列具有相同性质的对象的抽象，是对对象共同特征的描述。比如：电脑可以被定义为一个类，具体的某一台电脑就是一个对象。
  在一个类中，每个对象都是类的实例，可以使用类中提供的方法。要想从类定义中产生对象，必须有创建实例的操作。
- 继承（Inheritance） 继承是使用已存在的定义作为基础建立新定义的技术。新类的定义可以是已经存在的类所声明的数据结构和新类所增加的声明的组合。新类复用已经存在的定义，而不要修改已经存在的类。已经存在的类可以作为基类来引用，而新类可以作为派生类来引用。这种复用技术大大提高了软件开发的效率。例如，电脑作为一个类已经存在，作为具有自身特征的笔记本电脑就可以从电脑中继承。它同电脑一样，具有型号、重量这些特征，可以开机、关机和待机。它还具备一般电脑所不具备的特征，如可随身携带等。

面向对象的应用程序实际上是阶层式的对象所组成的集合。在 C#中，这种阶层式关系以 Namespace（名称空间）来表示，如 System.Data。对象是某一类型的实体。类型是用来描述变量在内存中的样子。如 Int 类型代表长度为 4 字节（Byte），大小为正负 2^{31} 范围内的整数。在 C#中，有许多类型，而类是其中最重要的类型之一。

类之间可以通过继承来使用、扩展类的功能。在 C#中，所有类都是继承自 Sysetem.Object 类。因此，class Hello 实际上将被编译为 Class Hello：System.Object（“：”表示“继承自”）。

类由成员（Member）组成，类的成员可分为数据（Data）成员和函数成员。数据成员中包括常量（Constant）、原始类型（Primitive Type）和参考类型（Reference Type）。函数成员包括 Constructor，Destructor、Method、Property、Event、Indexer、Operator 等。

类可以通过其成员的访问限制符（Member Access modifier）来控制外部对类成员的访问。C#中类成员的访问限制符包括 public、private、internal、protected 等。在 Hello World 范例中，成员函数 Main() 的访问限制符为 public，表示任何外部类都可以不受限制地调用

这个函数。

C#中，Main()函数是特殊的函数名称，它是 C#程序的入口方法，所有的 C#程序都是由此开始执行的。

类成员可以是静态成员（Static Member）。静态成员表示不需要产生类的实体便可以被访问。在编译器的实际操作中。通常将静态成员当作是一个前置名称空间的全局变量（Global Variable）或全局函数（Global Function）。举例来说，调用 DateTime.Now 取得系统日期，其中 DateTime 为类名称，Now 为静态成员。在 C#中，规定必须将 Main()函数声明为 static 的，所以编译器可以在不产生对象的情况下调用此函数。

1.2.3 C#语法简介

在了解了 C#语言的面向对象的层次结构以后，我们来看一下 C#的基本语法。由于本书目的是介绍.NET Framework 中的基本内容而不是单单讲述 C#语言，所以不要指望在本节中获得所有的详细的语法介绍，更为细致的讲解请参考有关专业书籍。

变量声明

在 C#中声明变量时，要注意以下几点：

- 变量命名不能使用保留字
- 变量名中不能使用空格
- 变量名应以文字或下划线“_”开始

声明变量的语法为：

```
变量类型 变量名称 [= 初始值];
```

例如声明一个字符串变量可以：

```
string myString;
```

也可以在声明变量的同时初始化这个变量：

```
string myString = "Hello World";
```

类型（Type）

对象本身是类型（Type）的实体（Instance），而类型（Type）是描述对象在内存中的样子，因此要理解 C#，必须了解其类型系统。

在 C#中，类型可以分为两大类：实值类型（Value Type）与引用类型（Reference Type），两者的不同之处在于实值类型变量直接存放此类型的数据，而引用类型变量则存放指向实际对象的指针。

例如代码 int i = 5；所声明的变量 i 是实值类型，其值 5 直接存放在 Stack 中；而代码 string s = “Hello World”所声明的变量 s 是引用类型，其在 Stack 中保存的是指向 Managed Heap 中实际对象的引用指针。

C#将类型分为两大类主要是出于对代码执行的效率和安全性的考虑。由于实值类型变量直接将该类型的值保存在 Stack 中，而无需内存引用的操作和开辟新的内存空间，因此使用实值类型在代码执行的效率上比使用引用类型高。但是使用引用类型的优点是可以灵活的满足需求，而且.NET Framework 的资源回收器（Garbage Collector）会自动配置其在