

IT培训专家王寅永、李降宇新作，  
完美展现C#技术！

- **百问**：所有问题来自于作者多年的软件开发和实训经验，将C#开发的所有问题都糅进标题，从最基本的概念知识到复杂的技术内容，针对性更强。
- **百答**：针对标题提出的知识点一针见血地找到问题→分析问题→解决问题。
- **百例**：针对问题和分析给出300多个独家精彩实例，2个大型项目示例，帮助开发人员从实践中成长。

王寅永 李降宇 李广歌 编著

# C# 深入详解

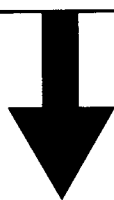
章.NET基础体系结构1.1什么是.NET1.2我们为什么需要.NET技术1.3什么是.NET Framework1.4.NET和J2EE的相同点1.4和不同点1.5.NET和C#之间的关系1.6C#的特点1.7C#程序的编译运行1.7.1什么是CLR1.7.2什么是CLR1.7.3什么是IL1.7.4C#程序是如何编译运行的第2章C#开发环境2.1C#语言需要什么开发工具2.2软件安装、卸载2.2.1安装Visual Studio 20052.2.2需要的软、硬件配置2.2.2安装Visual Studio 20052.2.2需要的权限设置2.2.3并行安装Visual Studio版本2.2.4进行.NET Framework版本2.2.4和并行安装2.2.5安装VS2.2.6安装Visual Studio 20052.2.7卸载12.3Visual Studio 2005开发2.3环境实际应用2.3.1管理解决方案、项目和文件32.3.2属性2.3.3编辑代码和资源文件2.3.4生成、调试和测试第3章编写第一个C#程序3.1C#结构3.2Main()和命令行参数3.2.1运用命令行参数3.2.2使用foreach访问命令行参3.2.3Main()返回值标识3.3学习第一个C#程序3.3.1编写第一个C#代码3.3.2程序添注释3.3.3编译程序第4章C#程序设计之基础知识4.1数据类型4.1.1C#的数据类型4.1.2C#数据类型4.1.3引用类型包括的内容4.2语句C#语言的语句类型4.3运算符C#运算符定义4.4数组4.4.1数组的定义和标识4.4.2一维数组的定义标识4.4.3数组初始化4.4.4多维数组的定义和标识4.4.5多维数组的初始化4.4.6交叉数组的定义和标识4.4.7在数组使用foreach4.4.8将一维数组作为参数传递4.4.9将多维数组作为参数传递4.4.10使用ref和out传递数组4.5字符串4.5.1字符串的定义和标识4.5.2访问字符串的字符4.5.3连接字符串4.5.4字符串进行比较4.5.5使用Split方法分析字符串4.6用字符串方法搜索字符串4.5.7修改字符串内容4.6命名空间4.6.1命名空间的定义和标识4.6.2访问命名空间4.6.3使用命名空间别名4.6.4使用命名空间来控制范围C#处理器指令4.7.1C#预处理的指令种类4.7.2预定义指令——#if的定义4.7.2和应用4.7.3预定义指令——#else的定义4.7.3和应用4.7.4预定义指令——#elif的定义4.7.4和应用4.7.5预定义指令——#endif的定义4.7.5和应用4.7.6预定义指令——define的定义4.7.6和应用4.7.7预定义指令——#undef的定义4.7.7和应用4.7.8预定义指令——#warning的定义4.7.8定义和应用4.7.9预定义指令——#error的定义4.7.9和应用4.7.10预定义指令——#line的定义4.7.10和应用4.7.11预定义指令——#region的定义4.7.11定义和应用4.7.12预定义指令——#endregion4.7.12的定义和应用4.7.13预定义指令——#pragma4.7.13的定义和应用4.7.14预定义指令——#pragmawarning4.7.14定义和应用4.7.15预定义指令——#pragma4.7.15checksum的定义和应用第5章面向对象的程序第5章设计思想5.1面向对象的基本概念5.1.1面向对象的概念5.1.2面向对象术的由来5.2面向对象的模型技术5.2.1对象模型技术5.3面向对象的分析5.3.1面向对象分析的概念5.3.2面向对象分析的任务5.3.3面向对象分析层次5.3.4面向对象分析步骤5.4面向对象的设计5.4.1面向对象设计的概念5.4.2面向对象设计阶段5.4.3面向对象设计的几个步骤第6章面向对象的C#语言6.1类class6.1.1类的概念6.1.5类的继承6.1.6类的修饰符6.1.7静态6.1.8类的成员6.1.9构造函数6.1.10析构函数6.2方法6.2.1静态方法和非静态的方法6.3结构6.3.1结构的标识6.3.2结构的特点6.3.3使用结构6.3.4传递结构与传递类实例6.4枚举6.4.1枚举的概念6.4.2枚举的使用6.4.3枚举的密封6.4.4抽象类和密封类6.4.5多态6.5多态6.5.1属性的概念6.5.2接口属性的使用6.5.3非对称访问器的使用6.6事件

TP312/2842

2008

# C#深入详解

王寅永 李降宇 李广歌 编著



電子工業出版社

Publishing House of Electronics Industry

北京•BEIJING

## 内 容 简 介

本书全面系统地介绍了C#这门编程语言,所涉及的内容涵盖了C#语言的各个领域。第1章,介绍.NET的基础体系结构,让读者了解相关的基础知识。第2章,介绍C#语言开发环境的搭建和使用。第3章,教读者开发第一个C#程序,亲身体验C#语言的开发、运行过程。第4章,介绍C#语言的基本知识,包括数据结构、运算符、基本语法等内容。第5章,介绍面向对象的基本理论和思想。第6章,介绍C#语言的类、方法、属性、事件等相关内容。第7章,深入学习C#语言,掌握接口、委托、索引器、线程等内容。第8章,学习C#语言对于文件的操作、数据库的相关操作、XML文件的操作。第9章,通过2个大型实例项目中数据结构的设计、系统结构的设计,以及源码内容的介绍,让读者全面深入了解C#语言。

本书结合了作者多年的开发和教学经验,从最基本的理论概念到实践样例,从最简单的C#编程知识到最复杂的技术领域,都对C#编程语言进行了介绍和分析。

本书内容丰富,结构清晰,通过300多个独家精彩实例和2个大型真实项目示例,帮助开发人员从实践中成长。本书是C#初学者的入门指导书,同样适合具备一定编程经验的开发人员。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

### 图书在版编目(CIP)数据

C#深入详解 / 王寅永, 李降宇, 李广歌编著. —北京: 电子工业出版社, 2008.5  
ISBN 978-7-121-06319-0

I. C… II. ①王… ②李… ③李… III. C语言—程序设计 IV. TP312

中国版本图书馆CIP数据核字(2008)第043796号

责任编辑: 李 冰

印 刷: 北京东光印刷厂

装 订: 三河市鹏成印业有限公司

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: 787×1092 1/16 印张: 33 字数: 659千字

印 次: 2008年5月第1次印刷

印 数: 5000册 定价: 55.00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 zltz@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线:(010) 88258888。

如果你正准备学习 C# 语言,或者你已经使用 C# 语言开发过一些项目,但总觉得缺少实例帮你领会 C# 技术的核心思想,那么这本书将是你最好的选择。作为进入编程行业的入门指导书之一,无论是买来细读,还是实例模拟训练,这本书都将帮助更多的初学者跨过 C# 程序员的门槛。

## 本书的创作契机

近两年,我在进行实际项目开发的同时,担任了兼职.NET 实训老师,为已经毕业的大学生或企业新入职员工讲解.NET 开发。讲课期间,看到了很多同学学习很刻苦,但是感觉入门很难。虽然他们已经很认真地在听课,很努力地动手做实验,也花了很多时间阅读编程类图书,但在面对一些很简单的问题时,却依然不知道如何入手。当看到他们愁眉紧锁的时候,我不禁自问,他们还缺什么?在后来的教学中,和他们接触多了,逐渐知道他们欠缺的是什么——适合他们的、易于理解的、简单而又全面的讲解项目开发实战的图书。

浏览目前市面上的那些编程专著,个个数百上千页,而且讲的内容比较深奥,初学者很难读懂。细读这些书,中、高级的开发人员会受益无穷,而作为初学者却会越看越迷糊。作为从事了十多年软件开发的老工程师,同时通过对学生的学习情况的了解掌握,我们深深知道,这些初学者和有经验的工程师他们都需要什么样的书——适合他们的、易于理解的、一针见血的、简单而又全面的图书。

很幸运的是,在我们萌生写一本适合初学者的入门书和有着开发经验工程师的工具书的想法时候,遇到了电子工业出版社的李冰编辑,在李冰编辑积极帮助下,我们开始了本书的编写。

## 本书的章节安排

本书结合了作者多年的开发和教学经验,对 C# 编程语言从最基本的理论概念到实践样例,从最简单的 C# 编程知识到最复杂技术领域,都进行了介绍和分析。

◎百问:所有问题来自于作者多年的软件开发和实训经验,将 C# 开发的所有问题都揉进标题,从最基本的概念知识到复杂的技术内容,针对性更强。

◎百答:针对标题提出的知识点一针见血地找到问题→分析问题→解决问题。

◎百例:针对问题和分析给出 300 多个独家精彩实例,2 个大型项目示例,帮助开发人员从实践中成长。

通过理论学习与动手实践紧密结合的方式,更能准确地找到这门语言的关键

知识点，加强了对这些关键知识点的记忆和理解。

本书共 9 章，从 C# 语言最基础的内容到最复杂的内容，都有相关的介绍，几乎涵盖了 C# 语言的各个领域。

|              |                             |                                                              |
|--------------|-----------------------------|--------------------------------------------------------------|
| 循序<br>渐<br>进 | 第 1 章 .NET 的基础体系结构          | .NET 的概念、.NET 和 C# 之间的关系、.NET 和 Java 之间的区别、C# 程序的编译运行过程等内容。  |
|              | 第 2 章 C# 开发环境               | Visual Studio 2005 软件的安装和卸载，Visual Studio 2005 开发环境的实际应用等内容。 |
|              | 第 3 章 编写第一个 C# 程序           | C# 程序的常规结构、Main ( ) 和命令行参数、学习编写第一个 C# 程序等内容。                 |
|              | 第 4 章 C# 程序设计基础知识           | C# 的数据类型、C# 语句、C# 的运算符、C# 的数组、C# 的字符串、C# 的命名空间、C# 的预处理指令等内容。 |
|              | 第 5 章 面向对象的程序设计思想           | 面向对象的基本概念、面向对象的模型技术、面向对象的分析、面向对象的设计等内容。                      |
|              | 第 6 章 面向对象的 C# 语言           | C# 语言的类、方法、结构、继承、属性、事件等内容。                                   |
|              | 第 7 章 深入了解 C#               | 接口、委托、索引器、迭代器、线程、反射、DLLs、程序集和全局程序集缓存，互操作性、泛型等内容。             |
|              | 第 8 章 C# 对于文件、数据库、XML 的基本应用 | 文件的操作，数据库的相关操作，XML 文件的相关应用等内容。                               |
|              | 第 9 章 项目示例                  | 中小型企业税收申报管理系统和企业进销存管理系统两个应用系统的介绍、数据设计、功能说明、源码解释等内容。          |
|              | 附录                          | C# 的关键字，C# 的常见错误码，参考资料等内容。                                   |

本书所有源代码资源读者可以登录 [www.broadview.com.cn](http://www.broadview.com.cn) 进行下载，并且我们提供技术博客回答读者问题 ([http://hi.csdn.net/wang\\_yy](http://hi.csdn.net/wang_yy))。

## 感言

本书的出版，家人们给了很大的支持！另外老师、同学、同事，以及一些网友朋友提供了很多的资料，给予了很大的帮助，在此表示深深的感谢！

编程是一个创造性的劳动，无论你是初学者还是编程高手，都会遇到问题，只要在遇到问题的时候，多问自己几个为什么，然后自己动手、动脑去解决这些问题，并且很好地积累这些经验教训，你的技术能力就会越来越强。

最后感谢电子工业出版社的编辑们不厌其烦地改正了书稿中的许多错误。

因时间仓促，编者水平有限，错误之处或者不准确的地方在所难免，敬请读者给予批评指正！

王寅永

# 目 录

## PREFACE

### 第 1 章 .NET 基础体系结构.....1

- 1.1 什么是.NET.....1
- 1.2 我们为什么需要.NET 技术.....2
- 1.3 什么是.NET Framework.....2
- 1.4 .NET 和 J2EE 的相同点和  
不同点.....2
- 1.5 .NET 和 C# 之间的关系.....3
- 1.6 C#语言的特点.....4
- 1.7 C#程序的编译运行.....5
  - 1.7.1 什么是 CLI.....5
  - 1.7.2 什么是 CLR.....5
  - 1.7.3 什么是 IL.....5
  - 1.7.4 C#程序是如何编译运行的.....5

### 第 2 章 C#开发环境.....7

- 2.1 C#语言需要的开发工具.....7
- 2.2 软件的安装、卸载.....7
  - 2.2.1 安装 Visual Studio 2005 需要的  
软、硬件配置.....7
  - 2.2.2 安装 Visual Studio 2005 需要的  
权限设置.....8
  - 2.2.3 并行安装 Visual Studio 版本.....8
  - 2.2.4 进行.NET Framework 版本的  
并行安装.....8
  - 2.2.5 安装 IIS.....9
  - 2.2.6 安装 Visual Studio 2005.....9
  - 2.2.7 卸载.....11
- 2.3 Visual Studio 2005 开发环境  
实际应用.....13
  - 2.3.1 管理解决方案、项目和文件.....13
    - 2.3.1.1 创建解决方案.....13
    - 2.3.1.2 创建解决方案的目录.....13

- 2.3.1.3 更改或添加默认编辑器..... 14
- 2.3.1.4 升级使用 Visual Studio .NET  
2002 或 Visual Studio .NET  
2003 创建的项目..... 16
- 2.3.1.5 设置启动项目..... 17
- 2.3.1.6 修改项目属性和配置设置..... 18
- 2.3.1.7 添加新项目项..... 18
- 2.3.1.8 复制项目..... 19
- 2.3.1.9 删除或移除项目..... 19
- 2.3.1.10 卸载和重新加载项目..... 20
- 2.3.1.11 移动项..... 21
- 2.3.1.12 刷新解决方案资源管理器  
中的项..... 21
- 2.3.1.13 如何重命名解决方案、  
项目和项..... 22
- 2.3.2 项目属性..... 22
  - 2.3.2.1 指定程序集信息..... 23
  - 2.3.2.2 更改程序集名称..... 24
  - 2.3.2.3 更改应用程序的命名空间..... 24
  - 2.3.2.4 启用或禁用编译器警告..... 24
  - 2.3.2.5 更改应用程序的生成位置..... 25
  - 2.3.2.6 为项目生成 XML 文档..... 26
  - 2.3.2.7 设置编译常量..... 27
  - 2.3.2.8 针对特定的 CPU 类型优化  
应用程序..... 28
  - 2.3.2.9 在 Visual Studio 中添加或  
移除引用..... 28
  - 2.3.2.10 设置引用的 Copy Local  
属性..... 30
  - 2.3.2.11 添加或移除字符串资源..... 30
  - 2.3.2.12 访问设置事件..... 32
  - 2.3.2.13 对应用程序和部署清单  
进行签名..... 33
  - 2.3.2.14 对程序集进行签名..... 35

|                          |                       |    |          |                      |    |
|--------------------------|-----------------------|----|----------|----------------------|----|
| 2.3.2.15                 | 设置引用路径                | 36 | 4.1.2.3  | 值类型介绍——char 类型       | 60 |
| 2.3.2.16                 | 设置生成属性                | 36 | 4.1.2.4  | 值类型介绍——decimal 类型    | 60 |
| 2.3.2.17                 | 指定生成事件                | 37 | 4.1.2.5  | 值类型介绍——double 类型     | 62 |
| 2.3.3                    | 编辑代码和资源文件             | 37 | 4.1.2.6  | 值类型介绍——enum 类型       | 64 |
| 2.3.3.1                  | 选择和更改文本               | 38 | 4.1.2.7  | 值类型介绍——float 类型      | 65 |
| 2.3.3.2                  | 显示代码大纲和隐藏代码           | 39 | 4.1.2.8  | 值类型介绍——int 类型        | 66 |
| 2.3.3.3                  | 在编辑器中管理自动换行           | 39 | 4.1.2.9  | 值类型介绍——long 类型       | 67 |
| 2.3.3.4                  | 在编辑器中显示行号             | 40 | 4.1.2.10 | 值类型介绍——sbyte 类型      | 68 |
| 2.3.3.5                  | 在编辑器中将 URL 显示为链接      | 40 | 4.1.2.11 | 值类型介绍——short 类型      | 69 |
| 2.3.3.6                  | 以递增方式搜索文档             | 41 | 4.1.2.12 | 值类型介绍——struct 类型     | 70 |
| 2.3.3.7                  | 交互式搜索                 | 41 | 4.1.2.13 | 值类型介绍——uint 类型       | 70 |
| 2.3.4                    | 生成、调试和测试              | 42 | 4.1.2.14 | 值类型介绍——ulong 类型      | 71 |
| 2.3.4.1                  | 启用/禁用实时调试             | 42 | 4.1.2.15 | 值类型介绍——ushort 类型     | 72 |
| 2.3.4.2                  | 设置简单断点                | 43 | 4.1.3    | 引用类型包括的内容            | 73 |
| 2.3.4.3                  | 启用 SQL Server 2005 调试 | 44 | 4.2      | 语句                   | 77 |
| 2.3.4.4                  | 更改应用程序调试的启动操作         | 44 | 4.2.1    | C#语言的语句类型            | 77 |
| 2.3.4.5                  | 设置应用程序调试的启动选项         | 45 | 4.2.1.1  | 选择语句——if-else 定义及使用  | 78 |
| 2.3.4.6                  | 启用远程调试                | 47 | 4.2.1.2  | 选择语句——switch 定义及使用   | 80 |
| <b>第 3 章 编写第一个 C#程序</b>  | <b>48</b>             |    | 4.2.1.3  | 迭代语句——do 定义及使用       | 82 |
| 3.1                      | C#程序的常规结构             | 48 | 4.2.1.4  | 迭代语句——for 定义及使用      | 83 |
| 3.2                      | Main()和命令行参数          | 49 | 4.2.1.5  | 迭代语句——foreach 定义及使用  | 84 |
| 3.2.1                    | 运用命令行参数               | 49 | 4.2.1.6  | 迭代语句——while 定义及使用    | 84 |
| 3.2.2                    | 使用 foreach 访问命令行参数    | 50 | 4.2.1.7  | 跳转语句——break 定义及使用    | 86 |
| 3.2.3                    | Main()返回值标识           | 51 | 4.2.1.8  | 跳转语句——continue 定义及使用 | 87 |
| 3.3                      | 学习第一个 C#程序            | 51 | 4.2.1.9  | 跳转语句——goto 定义及使用     | 88 |
| 3.3.1                    | 编写第一个 C#代码            | 51 | 4.2.1.10 | 跳转语句——return 定义及使用   | 89 |
| 3.3.2                    | 程序添加注释                | 52 |          |                      |    |
| 3.3.3                    | 编译程序                  | 53 |          |                      |    |
| <b>第 4 章 C#程序设计之基础知识</b> | <b>57</b>             |    |          |                      |    |
| 4.1                      | 数据类型                  | 57 |          |                      |    |
| 4.1.1                    | C#的数据类型               | 57 |          |                      |    |
| 4.1.2                    | C#值类型                 | 57 |          |                      |    |
| 4.1.2.1                  | C#值类型介绍——bool 类型      | 57 |          |                      |    |
| 4.1.2.2                  | 值类型介绍——byte 类型        | 59 |          |                      |    |

|          |                                           |     |          |                         |     |
|----------|-------------------------------------------|-----|----------|-------------------------|-----|
| 4.2.1.11 | 异常处理语句——throw<br>定义及使用 .....              | 89  | 4.3.1.26 | !=运算符定义和应用 .....        | 117 |
| 4.2.1.12 | 异常处理语句——try-catch<br>定义及使用 .....          | 90  | 4.3.1.27 | <=运算符定义和应用 .....        | 118 |
| 4.2.1.13 | 异常处理语句——try-finally<br>定义及使用 .....        | 93  | 4.3.1.28 | >=运算符定义和应用 .....        | 118 |
| 4.2.1.14 | 异常处理语句——try-catch<br>-finally 定义及使用 ..... | 94  | 4.3.1.29 | +=运算符定义和应用 .....        | 119 |
| 4.2.1.15 | 检查处理语句——Checked<br>定义及使用 .....            | 94  | 4.3.1.30 | -=运算符定义和应用 .....        | 120 |
| 4.2.1.16 | 检查处理语句——unchecked<br>定义及使用 .....          | 96  | 4.3.1.31 | *=运算符定义和应用 .....        | 121 |
| 4.2.1.17 | Fixed 语句定义及使用 .....                       | 97  | 4.3.1.32 | /=运算符定义和应用 .....        | 122 |
| 4.2.1.18 | lock 语句定义及使用 .....                        | 97  | 4.3.1.33 | %=运算符定义和应用 .....        | 123 |
| 4.3      | 运算符 .....                                 | 98  | 4.3.1.34 | &=运算符定义和应用 .....        | 124 |
| 4.3.1    | C#运算符定义 .....                             | 98  | 4.3.1.35 | =运算符定义和应用 .....         | 124 |
| 4.3.1.1  | []运算符定义和应用 .....                          | 99  | 4.3.1.36 | ^=运算符定义和应用 .....        | 125 |
| 4.3.1.2  | ()运算符定义和应用 .....                          | 100 | 4.3.1.37 | <<=运算符定义和应用 .....       | 126 |
| 4.3.1.3  | ()运算符定义和应用 .....                          | 100 | 4.3.1.38 | >>=运算符定义和应用 .....       | 127 |
| 4.3.1.4  | (::)运算符定义和应用 .....                        | 101 | 4.3.1.39 | ->运算符定义和应用 .....        | 128 |
| 4.3.1.5  | +运算符定义和应用 .....                           | 101 | 4.3.1.40 | ??运算符定义和应用 .....        | 129 |
| 4.3.1.6  | -运算符定义和应用 .....                           | 102 | 4.4      | 数组 .....                | 130 |
| 4.3.1.7  | *运算符定义和应用 .....                           | 103 | 4.4.1    | 数组的定义和标识 .....          | 130 |
| 4.3.1.8  | /运算符定义和应用 .....                           | 103 | 4.4.2    | 一维数组的定义标识 .....         | 130 |
| 4.3.1.9  | %运算符定义和应用 .....                           | 104 | 4.4.3    | 一维数组初始化 .....           | 130 |
| 4.3.1.10 | &运算符定义和应用 .....                           | 104 | 4.4.4    | 多维数组的定义和标识 .....        | 131 |
| 4.3.1.11 | 运算符定义和应用 .....                            | 105 | 4.4.5    | 多维数组的初始化 .....          | 131 |
| 4.3.1.12 | ^运算符定义和应用 .....                           | 106 | 4.4.6    | 交叉数组的定义和标识 .....        | 132 |
| 4.3.1.13 | !运算符定义和应用 .....                           | 107 | 4.4.7    | 在数组使用 foreach .....     | 133 |
| 4.3.1.14 | ~运算符定义和应用 .....                           | 107 | 4.4.8    | 将一维数组作为参数传递 .....       | 134 |
| 4.3.1.15 | =运算符定义和应用 .....                           | 108 | 4.4.9    | 将多维数组作为参数传递 .....       | 135 |
| 4.3.1.16 | <运算符定义和应用 .....                           | 109 | 4.4.10   | 使用 ref 和 out 传递数组 ..... | 136 |
| 4.3.1.17 | >运算符定义和应用 .....                           | 110 | 4.5      | 字符串 .....               | 138 |
| 4.3.1.18 | ?运算符定义和应用 .....                           | 110 | 4.5.1    | 字符串的定义和标识 .....         | 138 |
| 4.3.1.19 | ++运算符定义和应用 .....                          | 111 | 4.5.2    | 访问字符串的字符 .....          | 139 |
| 4.3.1.20 | --运算符定义和应用 .....                          | 112 | 4.5.3    | 连接字符串 .....             | 139 |
| 4.3.1.21 | &&运算符定义和应用 .....                          | 113 | 4.5.4    | 字符串进行比较 .....           | 140 |
| 4.3.1.22 | 运算符定义和应用 .....                            | 114 | 4.5.5    | 使用 Split 方法分析字符串 .....  | 140 |
| 4.3.1.23 | <<运算符定义和应用 .....                          | 115 | 4.5.6    | 使用字符串方法搜索字符串 .....      | 141 |
| 4.3.1.24 | >>运算符定义和应用 .....                          | 116 | 4.5.7    | 修改字符串内容 .....           | 143 |
| 4.3.1.25 | ==运算符定义和应用 .....                          | 116 | 4.6      | 命名空间 .....              | 144 |
|          |                                           |     | 4.6.1    | 命名空间的定义和标识 .....        | 144 |
|          |                                           |     | 4.6.2    | 访问命名空间 .....            | 145 |
|          |                                           |     | 4.6.3    | 使用命名空间别名 .....          | 145 |

|                                                |            |                                        |            |
|------------------------------------------------|------------|----------------------------------------|------------|
| 4.6.4 使用命名空间来控制范围 .....                        | 146        | 5.2 面向对象的模型技术 .....                    | 157        |
| 4.7 C#预处理器指令 .....                             | 147        | 5.2.1 对象模型技术 .....                     | 157        |
| 4.7.1 C#预处理的指令种类 .....                         | 147        | 5.2.1.1 对象模型的概念及建立<br>步骤 .....         | 157        |
| 4.7.2 预定义指令——#if 的定义和<br>应用 .....              | 147        | 5.2.1.2 功能模型的概念及建立<br>步骤 .....         | 158        |
| 4.7.3 预定义指令——#else 的定义和<br>应用 .....            | 148        | 5.2.1.3 动态模型的概念及建立<br>步骤 .....         | 158        |
| 4.7.4 预定义指令——#elif 的定义和<br>应用 .....            | 148        | 5.3 面向对象的分析 .....                      | 159        |
| 4.7.5 预定义指令——#endif 的<br>定义和应用 .....           | 149        | 5.3.1 面向对象分析的概念 .....                  | 159        |
| 4.7.6 预定义指令——#define 的定义<br>和应用 .....          | 149        | 5.3.2 面向对象分析的任务 .....                  | 159        |
| 4.7.7 预定义指令——#undef 的定义<br>和应用 .....           | 150        | 5.3.3 面向对象分析的层次 .....                  | 159        |
| 4.7.8 预定义指令——#warning 的<br>定义和应用 .....         | 150        | 5.3.4 面向对象分析的步骤 .....                  | 159        |
| 4.7.9 预定义指令——#error 的定义<br>和应用 .....           | 151        | 5.3.4.1 面向对象分析的阶段——<br>论域分析阶段的定义 ..... | 160        |
| 4.7.10 预定义指令——#line 的定义和<br>应用 .....           | 151        | 5.3.4.2 面向对象分析的阶段——<br>应用分析阶段的定义 ..... | 160        |
| 4.7.11 预定义指令——#region 的<br>定义和应用 .....         | 153        | 5.4 面向对象的设计 .....                      | 161        |
| 4.7.12 预定义指令——#endregion 的<br>定义和应用 .....      | 153        | 5.4.1 面向对象设计的概念 .....                  | 161        |
| 4.7.13 预定义指令——#pragma 的<br>定义和应用 .....         | 153        | 5.4.2 面向对象设计阶段 .....                   | 161        |
| 4.7.14 预定义指令——#pragmewarning<br>的定义和应用 .....   | 154        | 5.4.2.1 面向对象设计——高层<br>设计的概念 .....      | 161        |
| 4.7.15 预定义指令——#pragma<br>checksum 的定义和应用 ..... | 154        | 5.4.2.2 面向对象设计——低层<br>设计的概念 .....      | 161        |
| <b>第5章 面向对象的程序设计</b>                           |            | 5.4.3 面向对象设计的几个步骤 .....                | 162        |
| <b>思想 .....</b>                                | <b>156</b> | 5.4.3.1 面向对象设计——问题<br>论域设计的内容 .....    | 162        |
| 5.1 面向对象的基本概念 .....                            | 156        | 5.4.3.2 面向对象设计——用户<br>界面设计的内容 .....    | 162        |
| 5.1.1 对象的概念 .....                              | 156        | 5.4.3.3 面向对象设计——任务<br>管理设计的内容 .....    | 163        |
| 5.1.2 面向对象技术的由来 .....                          | 156        | 5.4.3.4 面向对象设计——数据<br>管理设计的内容 .....    | 163        |
|                                                |            | <b>第6章 面向对象的C#语言 .....</b>             | <b>164</b> |
|                                                |            | 6.1 类(class) .....                     | 164        |
|                                                |            | 6.1.1 类的概念 .....                       | 164        |
|                                                |            | 6.1.2 类的标识 .....                       | 164        |
|                                                |            | 6.1.3 类的特点 .....                       | 165        |
|                                                |            | 6.1.4 使用类创建对象 .....                    | 165        |

|                             |     |                                  |            |
|-----------------------------|-----|----------------------------------|------------|
| 6.1.5 类的继承                  | 165 | 6.4.3 base 保留字的使用                | 198        |
| 6.1.6 类的修饰符                 | 167 | 6.4.4 抽象类和密封类                    | 201        |
| 6.1.7 静态类                   | 169 | 6.4.4.1 抽象类的标识                   | 201        |
| 6.1.7.1 静态类的概念              | 169 | 6.4.4.2 密封类的使用                   | 201        |
| 6.1.7.2 静态类的特点              | 169 | 6.4.5 多态                         | 202        |
| 6.1.7.3 使用静态类               | 169 | 6.4.5.1 多态的概念                    | 202        |
| 6.1.8 类的成员                  | 171 | 6.4.5.2 多态的应用                    | 202        |
| 6.1.8.1 类成员的概念              | 171 | 6.4.5.3 重写 OnPaint 和 ToString    | 208        |
| 6.1.8.2 类的成员——this 保留字的运用   | 171 | 6.5 属性                           | 209        |
| 6.1.8.3 类的成员——静态成员的标识       | 173 | 6.5.1 属性的概念                      | 209        |
| 6.1.8.4 类的成员——静态成员和非静态成员的区分 | 175 | 6.5.1.1 get 访问器的使用               | 210        |
| 6.1.8.5 类的成员——成员常量          | 176 | 6.5.1.2 set 访问器的使用               | 211        |
| 6.1.9 构造函数                  | 177 | 6.5.2 接口属性的使用                    | 212        |
| 6.1.9.1 构造函数的使用             | 177 | 6.5.3 非对称访问器的使用                  | 214        |
| 6.1.9.2 实例构造函数              | 179 | 6.6 事件                           | 217        |
| 6.1.9.3 私有构造函数              | 181 | 6.6.1 事件的概念                      | 217        |
| 6.1.9.4 静态构造函数              | 182 | 6.6.2 使用事件                       | 217        |
| 6.1.9.5 复制构造函数              | 183 | 6.6.2.1 引发事件的使用                  | 218        |
| 6.1.10 析构函数                 | 184 | 6.6.2.2 订阅事件的使用                  | 218        |
| 6.2 方法                      | 186 | 6.6.3 创建响应事件的控件                  | 219        |
| 6.2.1 方法的标识                 | 187 | 6.6.4 接口中声明一个事件并类中实现该事件          | 222        |
| 6.2.2 方法返回值                 | 187 | 6.6.5 在 Visual C# 代码编辑器中创建事件处理程序 | 223        |
| 6.2.3 方法中的参数类型              | 188 | <b>第7章 深入了解 C#</b>               | <b>224</b> |
| 6.2.3.1 值参数                 | 189 | 7.1 接口                           | 224        |
| 6.2.3.2 引用参数                | 189 | 7.1.1 接口的概念                      | 224        |
| 6.2.3.3 输出参数                | 191 | 7.1.2 接口的标识                      | 224        |
| 6.2.3.4 数组型参数               | 192 | 7.1.3 显式接口的实现                    | 225        |
| 6.2.4 静态和非静态的方法             | 193 | 7.1.4 显式实现接口成员                   | 226        |
| 6.3 结构                      | 194 | 7.2 委托                           | 227        |
| 6.3.1 结构的标识                 | 194 | 7.2.1 委托的概念                      | 227        |
| 6.3.2 结构的特点                 | 195 | 7.2.2 委托的应用                      | 228        |
| 6.3.3 使用结构                  | 195 | 7.2.3 委托中命名方法的应用                 | 230        |
| 6.3.4 传递结构与传递类实例            | 196 | 7.2.4 委托中匿名方法的应用                 | 232        |
| 6.4 继承                      | 198 | 7.2.5 使用委托而不使用接口                 | 233        |
| 6.4.1 继承的概念                 | 198 | 7.2.6 合并委托                       | 234        |
| 6.4.2 继承的标识                 | 198 | 7.3 索引器                          | 235        |



|          |                                        |     |          |                               |     |
|----------|----------------------------------------|-----|----------|-------------------------------|-----|
| 8.1.1.11 | 用于文件 I/O 的类——<br>SerialPort 的应用        | 315 | 8.1.1.28 | 小结——从文件写入<br>文本               | 342 |
| 8.1.1.12 | 用于从流读取和写入流<br>的类——BinaryReader 的<br>应用 | 315 | 8.1.1.29 | 小结——从字符串中读取<br>字符             | 344 |
| 8.1.1.13 | 用于从流读取和写入流<br>的类——BinaryWriter 的<br>应用 | 317 | 8.1.1.30 | 小结——向字符串写入<br>字符              | 344 |
| 8.1.1.14 | 用于从流读取和写入流<br>的类——StreamReader 的<br>应用 | 320 | 8.1.2    | 构成流                           | 345 |
| 8.1.1.15 | 用于从流读取和写入流<br>的类——StreamWriter 的<br>应用 | 321 | 8.1.2.1  | 构成流的概念                        | 345 |
| 8.1.1.16 | 用于从流读取和写入流<br>的类——StringReader<br>如何应用 | 322 | 8.1.2.2  | 构成流的应用                        | 345 |
| 8.1.1.17 | 用于从流读取和写入流<br>的类——StringWriter 的<br>应用 | 324 | 8.1.3    | 异步文件 I/O                      | 347 |
| 8.1.1.18 | 用于从流读取和写入流<br>的类——TextReader 的<br>应用   | 326 | 8.1.4    | 独立存储                          | 348 |
| 8.1.1.19 | 用于从流读取和写入流<br>的类——TextWriter 的<br>应用   | 327 | 8.1.4.1  | 独立存储的概念                       | 348 |
| 8.1.1.20 | 通用 I/O 流类——<br>BufferedStream 的应用      | 329 | 8.1.4.2  | 独立存储的使用                       | 348 |
| 8.1.1.21 | 通用 I/O 流类——<br>CryptoStream 应用         | 332 | 8.1.4.3  | 隔离的类型                         | 349 |
| 8.1.1.22 | 通用 I/O 流类——<br>MemoryStream 的应用        | 335 | 8.2      | 数据库                           | 350 |
| 8.1.1.23 | 通用 I/O 流类——<br>NetworkStream 的应用       | 337 | 8.2.1    | 数据访问入门                        | 350 |
| 8.1.1.24 | 小结——创建目录清单                             | 338 | 8.2.1.1  | 数据源的概念                        | 350 |
| 8.1.1.25 | 小结——对新建的数据<br>文件进行读取和写入                | 339 | 8.2.1.2  | 本地数据的概念                       | 351 |
| 8.1.1.26 | 小结——打开并追加到<br>日志文件                     | 340 | 8.2.1.3  | 数据访问策略的策略                     | 351 |
| 8.1.1.27 | 小结——从文件读取<br>文本                        | 341 | 8.2.1.4  | ADO.NET 的好处                   | 354 |
|          |                                        |     | 8.2.1.5  | ADO.NET 和 ADO 的相同点<br>和不同点    | 356 |
|          |                                        |     | 8.2.2    | TableAdapter                  | 358 |
|          |                                        |     | 8.2.2.1  | TableAdapter 的概念              | 358 |
|          |                                        |     | 8.2.2.2  | 启动 TableAdapter 配置<br>向导      | 360 |
|          |                                        |     | 8.2.2.3  | 创建 TableAdapter               | 361 |
|          |                                        |     | 8.2.2.4  | 创建 TableAdapter 查询            | 362 |
|          |                                        |     | 8.2.2.5  | 扩展 TableAdapter 的功能           | 363 |
|          |                                        |     | 8.2.2.6  | 执行 TableAdapter 查询            | 363 |
|          |                                        |     | 8.2.2.7  | 使用 TableAdapter 直接<br>访问数据库   | 365 |
|          |                                        |     | 8.2.3    | 连接到 Visual Studio 中的数据        | 365 |
|          |                                        |     | 8.2.3.1  | 连接到 Visual Studio 的<br>数据源的配置 | 366 |
|          |                                        |     | 8.2.3.2  | 保存连接字符串                       | 367 |
|          |                                        |     | 8.2.3.3  | 如何连接到对象中的数据                   | 368 |
|          |                                        |     | 8.2.3.4  | 创建与 Access 数据库的<br>连接         | 368 |
|          |                                        |     | 8.2.3.5  | 创建与 SQL Server 数据<br>库的连接     | 369 |

|          |                                     |     |                        |                                      |     |
|----------|-------------------------------------|-----|------------------------|--------------------------------------|-----|
| 8.2.3.6  | 创建与 Oracle 数据库的连接.....              | 369 | 8.2.5.21               | 在 ADO.NET 中修改数据 ...                  | 395 |
| 8.2.3.7  | 连接到 Web 服务中的数据.....                 | 370 | 8.2.5.22               | 执行编录操作 .....                         | 397 |
| 8.2.3.8  | 连接到 SQL Server Express 数据库中的数据..... | 370 | 8.2.5.23               | 如何将 BLOB 值写入数据源.....                 | 397 |
| 8.2.4    | 将数据获取到应用程序 .....                    | 371 | 8.2.6                  | 事务处理 .....                           | 399 |
| 8.2.4.1  | 使用数据填充数据集 .....                     | 371 | 8.2.6.1                | 使用事务范围实现隐式事务 .....                   | 399 |
| 8.2.4.2  | 创建和执行返回行的 SQL 语句.....               | 372 | 8.2.6.2                | 使用可提交事务实现显式事务.....                   | 400 |
| 8.2.4.3  | 创建和执行返回单个值的 SQL 语句.....             | 373 | 8.2.6.3                | 在单阶段和多阶段中提交事务 .....                  | 401 |
| 8.2.4.4  | 创建和执行不返回值的 SQL 语句.....              | 374 | 8.3                    | XML 技术 .....                         | 404 |
| 8.2.4.5  | 执行返回行的存储过程.....                     | 375 | 8.3.1                  | XML 技术概述 .....                       | 404 |
| 8.2.4.6  | 客户端和中间层编程中的数据访问 .....               | 376 | 8.3.2                  | XML 技术产生的背景 .....                    | 405 |
| 8.2.5    | ADO.NET .....                       | 377 | 8.3.3                  | XML 的优缺点 .....                       | 406 |
| 8.2.5.1  | ADO.NET 概述 .....                    | 377 | 8.3.4                  | XML 的作用及应用前景.....                    | 407 |
| 8.2.5.2  | ADO.NET 设计目标 .....                  | 378 | 8.3.5                  | XML 的语法结构 .....                      | 407 |
| 8.2.5.3  | ADO.NET 结构 .....                    | 379 | 8.3.6                  | 设计“格式良好的”XML 文档.....                 | 409 |
| 8.2.5.4  | ADO.NET DataSet 的应用.....            | 380 | 8.3.7                  | 设计“有效的”XML 文档——DTD 和 XML Schema..... | 409 |
| 8.2.5.5  | 向 DataSet 添加 DataTable.....         | 381 | 8.3.8                  | 进行 XML 数据的显示控制.....                  | 411 |
| 8.2.5.6  | 向 DataSet 添加表间关系 .....              | 381 | 8.3.9                  | 进行 XML 与 HTML 的绑定与操作.....            | 413 |
| 8.2.5.7  | 创建和使用 DataTables .....              | 382 | 8.3.10                 | 进行基于 DOM 的数据操作 .....                 | 414 |
| 8.2.5.8  | 将数据添加到表中 .....                      | 383 | <b>第 9 章 项目实践.....</b> | <b>418</b>                           |     |
| 8.2.5.9  | 访问表中的数据 .....                       | 383 | 9.1                    | 中小型企业税收申报管理系统 .....                  | 418 |
| 8.2.5.10 | 编辑表中的数据 .....                       | 384 | 9.1.1                  | 系统平台 .....                           | 418 |
| 8.2.5.11 | 从表中删除行.....                         | 385 | 9.1.2                  | 系统分析 .....                           | 419 |
| 8.2.5.12 | 添加和读取行错误信息.....                     | 386 | 9.1.3                  | 项目规划 .....                           | 419 |
| 8.2.5.13 | 使用 DataTable 事件.....                | 387 | 9.1.4                  | 总体规则 .....                           | 420 |
| 8.2.5.14 | 创建和使用 DataView .....                | 387 | 9.1.5                  | 数据库设计 .....                          | 420 |
| 8.2.5.15 | 使用 DataView 对数据排序和筛选.....           | 388 | 9.1.6                  | 配置文件代码分析 .....                       | 435 |
| 8.2.5.16 | 查看 DataView 的内容.....                | 388 | 9.1.7                  | 数据库操作类代码分析 .....                     | 435 |
| 8.2.5.17 | 使用 DataView 修改数据 .....              | 389 | 9.1.7.1                | 实现数据库连接 .....                        | 435 |
| 8.2.5.18 | 使用 DataView 事件 .....                | 390 | 9.1.7.2                | 实现关闭、释放一个数据库连接 .....                 | 436 |
| 8.2.5.19 | ADO.NET 使用连接字符串 .....               | 391 | 9.1.7.3                | 增加 sql 命令对应的参数.....                  | 436 |
| 8.2.5.20 | 在 ADO.NET 中连接和检索数据 .....            | 394 | 9.1.7.4                | 用 DataSet 对象, 更新数据库 .....            | 438 |

|                                               |     |                         |     |
|-----------------------------------------------|-----|-------------------------|-----|
| 9.1.7.5 更新 BLOB 类型的<br>字段值 .....              | 439 | 9.2.5 编码规则 .....        | 476 |
| 9.1.7.6 读取 BLOB 类型的字段值 .....                  | 440 | 9.2.6 数据库设计 .....       | 476 |
| 9.1.7.7 执行 SQL 查询命令, 并<br>返回 DataSet 对象 ..... | 440 | 9.2.7 数据库操作类代码分析 .....  | 479 |
| 9.1.8 业务功能模块代码分析 .....                        | 440 | 9.2.7.1 实现数据库连接 .....   | 479 |
| 9.1.8.1 登录界面代码分析 .....                        | 441 | 9.2.7.2 实现数据库关闭 .....   | 480 |
| 9.1.8.2 主界面模块 .....                           | 442 | 9.2.7.3 释放数据库连接资源 ..... | 480 |
| 9.1.8.3 用户管理模块 .....                          | 447 | 9.2.7.4 执行参数命令文本 .....  | 480 |
| 9.1.8.4 组管理模块 .....                           | 452 | 9.2.8 业务功能模块代码分析 .....  | 482 |
| 9.1.8.5 业务信息维护模块 .....                        | 456 | 9.2.8.1 登录界面代码分析 .....  | 482 |
| 9.1.8.6 税收申报模块 .....                          | 460 | 9.2.8.2 主界面代码分析 .....   | 483 |
| 9.1.8.7 统计管理模块 .....                          | 467 | 9.2.8.3 基本信息管理模块 .....  | 484 |
| 9.1.8.8 票证管理模块 .....                          | 469 | 9.2.8.4 系统维护模块 .....    | 487 |
| 9.1.9 系统安装 .....                              | 472 | 9.2.8.5 进货模块 .....      | 489 |
| 9.2 企业进销存管理系统 .....                           | 474 | 9.2.8.6 出货模块 .....      | 492 |
| 9.2.1 系统平台 .....                              | 474 | 9.2.8.7 库存模块 .....      | 494 |
| 9.2.2 系统分析 .....                              | 475 | 9.2.8.8 辅助工具模块 .....    | 495 |
| 9.2.3 项目规划 .....                              | 475 | 附录 A C#关键字 .....        | 497 |
| 9.2.4 总体规则 .....                              | 475 | 附录 B C#的错误码 .....       | 498 |
|                                               |     | 参考文献 .....              | 509 |

# 第 1 章

## .NET 基础体系结构

### 1.1 什么是.NET

2000 年 6 月 22 日, 微软公司推出了其下一代计算计划——Microsoft.NET (以下简称.NET), 这项计划将使微软现有的软件在 Web 时代不仅适用于传统 PC 设备, 而且也能够满足强劲增长势头的新设备、新业务的要求。

首先说 .NET 是一个开发平台。.NET 定义了一种公用语言子集 (Common Language Subset, CLS), 这是一种为符合其规范的语言与类库之间提供无缝集成的混合语。.NET 统一了编程类库, 提供了网络通信标准可扩展标记语言 (Extensible Markup Language, XML) 的完全支持, .NET 的推出使开发人员开发应用程序变得更容易、更简单。

.NET 是一个开发的平台, 它实现了人和计算机的更好的交流, 最为重要的是.NET 将改变人们对计算机软件技术的一贯看法, .NET 与 Windows 平台紧密集成, 并和计算机的操作系统融合在一起, 成为一个无缝的平台。

再说 .NET 是一场软件革命。

- .NET 对最终用户来说非常重要, 因为计算机的功能将会得到大幅度提升, 同时计算机操作也会变得非常简单。用户将完全摆脱人为的硬件束缚, 更好地体会在自由冲浪于因特网的快乐, 自由访问、自由查看、自由使用共享的数据, 而不是束缚在便携式电脑的方寸空间。
- .NET 对开发人员来说也十分重要, 因为它改变了开发人员以往开发应用程序的方式, 使得开发人员能快速地创建出全新的应用程序, 极大地提高软件生产率。对于公司而言, 软件开发周期的缩短将能使它们更好地应付网络经济的竞争。

简言之, Microsoft.NET 是 Microsoft XML Web Services 平台。XML Web

Services 允许应用程序通过 Internet 进行通信和共享数据，而不管所采用的是哪种操作系统、设备或编程语言。Microsoft .NET 平台提供创建 XML Web services 并将这些服务集成在一起之所需。对个人用户的好处是无缝的、吸引人的体验。

## 1.2 我们为什么需要.NET 技术

旧的大型计算机的工作模式大多是这样的：信息被储存在中央服务器内，而用户的所有操作都要依靠它们。但是让这些不同的网址之间相互传递有意义的信息，或者让它们相互合作来提供更广泛和更深层次的服务，还是一件十分困难的事。

.NET 的出现意味着人们可以只用一种简单的界面就可以编写、浏览、编辑和分享信息，而且还可以得到功能强大的信息管理工具。由于使用的所有的文件都以符合网络协议的格式存在，所以，所有的商业用户和个人用户都可以方便地查找和使用其中的信息，用户可以使用相同的工具与他们的供应商、商业伙伴和客户高效地沟通和分享信息，这样就创造出一种全新的协同工作模式。

## 1.3 什么是.NET Framework

简单地说，.NET Framework 其实就是.NET 平台的一个运行、执行环境。

.NET Framework 是 Windows 的一个内部组件，可以支持生成和运行下一代应用程序和 XML Web services。

- .NET Framework 提供一个一致的面向对象的编程环境，而无论代码是在本地还是在远程分布和执行。
- .NET Framework 提供一个软件部署的执行环境。
- .NET Framework 提供一个提高代码安全性能的执行环境。
- .NET Framework 提供了一个标准，使其基于 .NET Framework 的代码可与其他代码很好地集成。

.NET Framework 具有两个主要组件：公共语言运行库（CLR）和 .NET Framework 类库。

（1）公共语言运行库是.NET Framework 的基础内容。它提供内存管理、线程管理和远程处理等核心服务，并且还强制实施代码的安全性和可靠性管理。

（2）.NET Framework 的另一个主要组件是类库，它是一个综合性的类型集合，开发人员可以使用它开发多种模式的应用程序，无论是命令行形式或者图形界面形式的应用。

## 1.4 .NET 和 J2EE 的相同点和不同点

J2EE 平台提供了一个基于组件的方法，用来设计、开发、装配及部署 Java

类型的应用程序。另外，它还提供了多层的分布式应用模型、组件重用、一致化的安全模型，以及灵活的事务控制等功能，并保证代码和应用平台无关性。而.NET平台是按照微软公司的设计思想，在任何一个操作平台上只要安装了公共语言运行库（CLR）就可以运行.NET程序。所以，在设计新技术的出发点上应该说.NET和J2EE是十分相似的。但是这两种技术在实现方法和具体的实现技术上面有很大的区别。

- 区别 1：支持的开发语言种类不同

.NET 所支持的开发语言比较广泛，比如 C++、VB、C#和 J#等，因而开发人员可以很容易地找到适合自己的开发语言。而 J2EE 只支持 Java 语言，因此 J2EE 对语言的选择是比较狭窄的。

- 区别 2：支持的开发语言的语言标准不同

J2EE 支持 Java 语言，.NET 支持 XML/SOAP。就标准的开放性来说，XML/SOAP 好于前者。XML 由 W3C（全球广域网协会）组织提出，得到众多厂家支持，是下一代互联网上内容的表示的标准，它能够有效地表达网络的各种信息。

- 区别 3：跨平台

在.NET 平台上开发程序真正实现了“代码重用”，即运行时和具体的语言分开，所有的资源管理、内存分配和变量类型等都是由公共语言运行库处理。例如，用 C#写的类就可以直接用在 C/C++程序中。而在 J2EE 平台上只能用 Java 来开发程序，运行时和具体的语言混在一起。

## 1.5 .NET 和 C# 之间的关系

简单地说，C#其实就是一种基于.NET 平台的一种编程开发语言。

.NET 的作用不仅仅是将开发人员从必须掌握多种框架的束缚中解脱出来，通过创建跨编程语言的公共 API 集，.NET 框架可提供强大的跨语言继承性、错误处理和调试功能。现在开发人员可以自由地选择他们喜欢的编程语言。.NET 将使编程人员梦想的语言互用性变成为近在眼前的现实。想想看，一个在 Visual Basic (VB) 中定义类能够在另一种与它完全不同的语言比如 C#语言的环境中使用、调试，甚至继承，这是多么令人兴奋的事情！

我们不能孤立地使用 C#语言，而必须和.NET Framework 一起考虑。C#编写的所有代码总是在.NET Framework 中运行。

C#能够在新的微软.NET 平台上快速开发种类丰富的应用程序。.NET 平台提供了大量的工具和服务，能够最大限度地发掘和使用计算及通信能力。由于其一流的面向对象的设计，从构建组件形式的高层商业对象到构造系统级应用程序，你都会发现 C#将是最合适的选择。

不但如此，C#还能为 C++程序员提供快捷的开发方式，又没有丢掉 C 和 C++的基本特征——强大的控制能力。C#与 C、C++有着很大程度上的相似性，熟