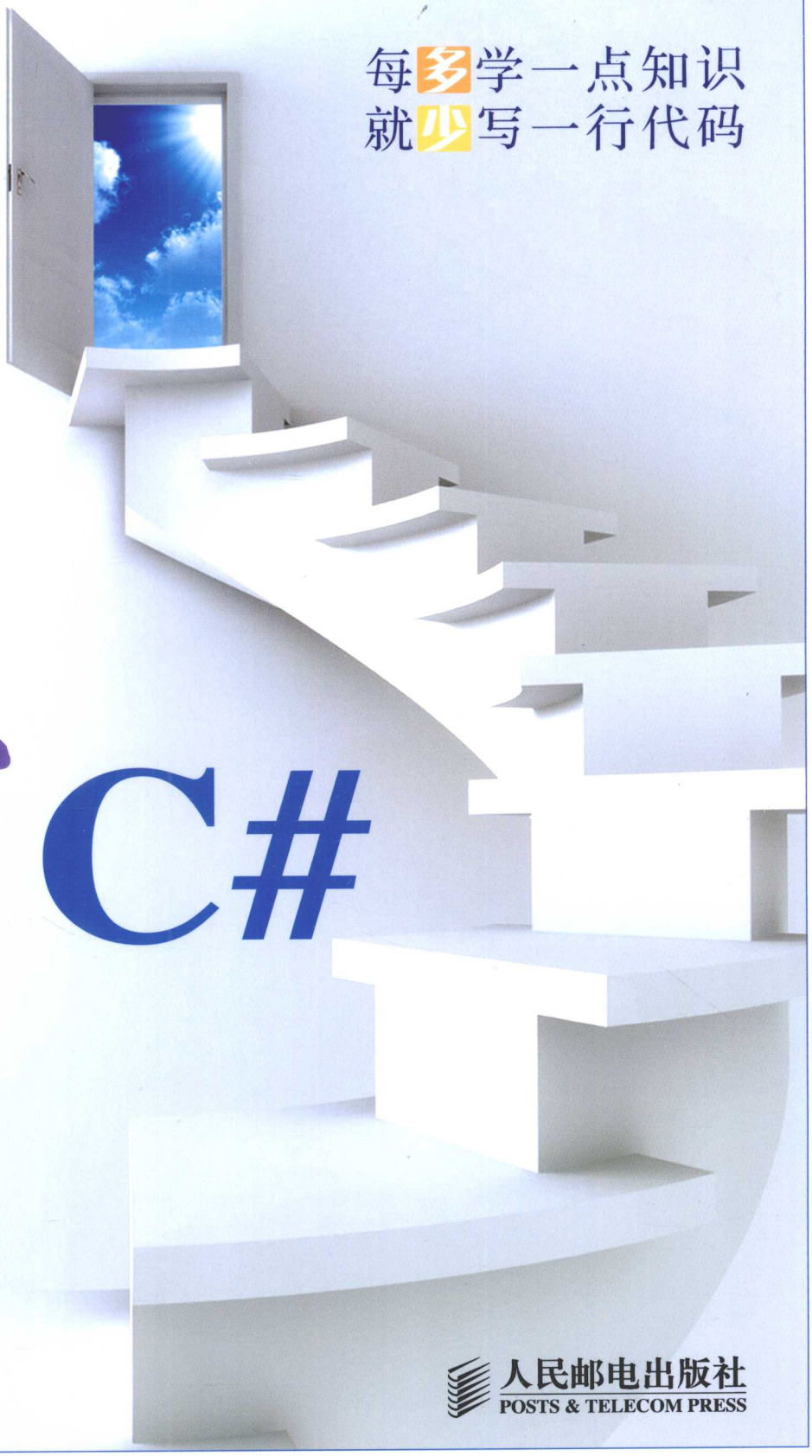
提供全书
示例代码下载

每多学一点知识
就少写一行代码



易学C#

马伟◎编著

人民邮电出版社
POSTS & TELECOM PRESS

易学C#

马伟◎编著

人民邮电出版社
北京

图书在版编目（CIP）数据

易学C#／马伟编著. —北京：人民邮电出版社，2009.10
ISBN 978-7-115-21198-9

Ⅰ. 易… Ⅱ. 马… Ⅲ. C语言－程序设计 Ⅳ. TP312

中国版本图书馆CIP数据核字（2009）第159355号

易学 C#

◆ 编　　著　马　伟
责任编辑　屈艳莲
执行编辑　蒋　佳

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京铭成印刷有限公司印刷

◆ 开本：800×1000　1/16
印张：26.5
字数：634 千字　　2009 年 10 月第 1 版
印数：1－4 000 册　　2009 年 10 月北京第 1 次印刷

ISBN 978-7-115-21198-9

定价：45.00 元

读者服务热线：(010)67132692　印装质量热线：(010)67129223
反盗版热线：(010)67171154

内容提要

C#语言是微软公司近几年推出的一种新型的完全面向对象的程序设计语言，到目前为止，它已经成为了应用软件开发的主流语言，尤其是在 Web 开发方面更是无与伦比。UML 则是面向对象软件的标准化建模语言，无论是企业信息系统、基于 Web 的分布式系统还是实时系统等都适合于使用 UML 来进行建模分析。本书正是 C#与 UML 融合的产物，书中不仅向读者阐述了 C#语言的编程基础知识与高级特性，而且还阐述了如何利用 UML 图形来进行面向对象的分析与设计。本书旨在帮助读者在较短的时间里对 C#语言与 UML 得到全面深刻的理解与认识，从而使读者将 C#与 UML 融合到一起，为读者以后的软件设计生涯打下坚实的基础。另外，本书还配备了许多经典的习题，这些习题全部是从各大软件公司的面试题中所提取的。认真地练习这些习题，能够让你在以后的职业面试中信心与成功率倍增。

本书文字简洁生动，并辅之以大量的图表和代码示例，对于希望学习 C#语言的学员具有自学指导的作用。本书既可作为大专、本科院校相关专业的教材，又可作为软件开发人员的技术参考手册。

作者简介

马伟：微软公司最有影响力的开发者之一，软件研发工程师与系统分析师。在其以往的程序岁月中，曾经负责过远程视频监控系统、视频营销平台、企业管理应用系统、南方电网四分统计平台等项目的架构设计与开发。擅长系统分析、架构设计、数据库设计与实现、设计模式、SOA 等技术，尤其对 C#与 UML、C++与 UML 的架构设计与编程有独到的见解。

前　言

本书的编写目的

众所周知，C#语言是微软公司在新一代开发平台.NET上推出的一种完全面向对象的新型语言。虽然只有短短几年时间的发展与推广，但C#语言凭借其自身的特性，使得它像程序设计语言中的一件艺术品一样，吸引着越来越多的开发人员转向它。

在这期间，市面上关于C#语言的辅导书非常多，使读者应接不暇、无所适从。但在仔细观察这些书后，我们不难发现普遍存在着这样一个问题：几乎市面上所有的C#书籍都向读者讲解的是纯粹的C#语言基础知识。大家知道，C#语言是一种完全面向对象的语言，在我们的日常程序设计中，只有真正地掌握好面向对象的思想才能够设计出好的面向对象程序。如果没有成熟的面向对象的思想作为指导，即便把语言学得再精通也是徒劳无功。

那么，为什么我们就不能够从一开始学习程序设计的时候就把C#语言和面向对象设计思想结合起来学习呢？为什么我们就不能够在写C#的相关书籍中融入软件工程与面向对象的思想呢？

基于上面的这些讨论和问题，使笔者产生了编写本书的源动力。可以这么说，本书不仅仅是教会读者如何使用C#语言进行程序设计，更重要的是教会读者如何用软件工程与面向对象的思想去分析设计软件以及如何使用C#语言进行面向对象程序设计，让读者从一开始学习编程就养成良好的程序设计习惯并打下坚实的基础。

在接下来的写作过程中，笔者遇到了这样一个问题，那就是C#语言和.NET Framework的发展。在写这本书时，笔者是以C# 3.0语言和.NET Framework 3.5版本来写的。但是，到目前为止，C# 4.0已经开始蠢蠢欲动、整装待发了。这使得笔者感到微软技术的发展之快，发出了“学习微软的东西在某种程度上是一种痛苦”之叹！但是，笔者始终认为，如果过分地去追求其形，而不修其心，那么将是花把势。学习一门语言如果不懂得该语言最本质、最基础的东西，而是一味地追求语言的新特性对于程序员自身的修炼来说是有害无益的。

本书的内容结构

基于上面的描述，在本书中分别使用了4部分来阐述如何使用C#语言与UML进行面向对象的程序设计与编码。

- 第1部分是C#语言与UML基础。本部分将重点向读者阐述C#语言的发展史与基础语法知识，最后介绍相关UML图形的画法。
- 第2部分是C#语言与面向对象程序设计。本部分是本书的重点，全面地阐述了如何使用C#语言和UML进行面向对象程序设计以及面向对象程序设计的相关原则。
- 第3部分是C#语言高级特性。本部分描述了C#语言的相关高级话题，例如进程与线程、反射、委托、序列化、泛型、异常处理与C# 3.0的新特性等。

· 第 4 部分是基于 UML 的面向对象分析与设计过程。本部分是对前面的一个总结，通过一个项目实例“基于 Web 的插件式框架系统的分析与设计”来详细阐述如何使用 C#与 UML 对实际项目进行面向对象的分析、设计与编码。

Web 站点和联机学习中心

为了能够和广大读者更好地沟通与交流，本书特别提供了如下 Web 站点供广大读者学习与交流：www.comesns.com/csharp。

在此 Web 站点里，您不仅可以直接和笔者与广大读者进行交流，而且还可以下载到本书的所有源代码与相关的电子教学文档，同时还有大量的学习资料与读者共享。

特别鸣谢

本书由马伟负责编写并统编全书稿，同时参与本书编写工作的还有梁建全、孟志勇、田利军、周力、卞征辉、邹小梅、张亚东、张庆丰、袁翠、杨蕙如、王紫、谭翔天、孙蓓蓓、芮巍、刘玖增、郭光、段世林、薛立滨、郑展鸿、许丽艳、王常友、何雪飞、方基成、董丽丽等，没有他们的帮助与付出，很难如期完成本书的写作。尤其要感谢下面这些人。

· 首先，出版社的编辑蒋佳老师为本书的修订和出版做了大量的工作，与他的合作是非常愉快的。同时，也正因为他对本书不断地提出要求，才使得本书的内容更加深刻化、语言更加幽默易懂化。

· 其次，笔者要感谢自己的家人，尤其是妻子吴亚峰女士。为了完成这本书的写作，笔者投入了大量的时间和精力，牺牲了许多可以陪家人的周末和节假日。

· 最后，笔者要感谢那些曾经为本书的编排提过意见的朋友与同事，感谢他们对本书的支持。

尽管笔者在本书的写作过程中非常认真与努力，但由于水平有限，书中难免存在错误和不足之处，恳请广大读者批评指正。如果您对本书有任何意见、问题或想法，欢迎通过笔者的论坛留言，或者用下面的邮件通知笔者，笔者将不胜感激。

论坛： www.comesns.com/bbs

E-mail： madengwei@hotmail.com

马 伟

2009 年 4 月

目 录

预备课：学习从这里开始……1

1. 软件=程序+文档……1
2. 程序起什么作用……2
3. 为何要面向对象……2
4. 什么是 UML……3
5. .NET 与 C#……4
   谈微软公司的.NET 战略……4
   C#的产生与特点……5
   C#与.NET 的关系……6
6. 开启 C#的钥匙——兴趣与正确的学习方法……7
7. 习题练习……8

第 1 部分 C#与 UML 基础

第 1 章 开篇经典——"Hello，World"……11

1.1 Hello，World……11
1.2 程序的结构分析……13
   1.2.1 命名空间……13
   1.2.2 类和类的方法……13
   1.2.3 程序的输入输出……14
1.3 程序的执行起点——Main 函数……14
1.4 控制台的输入和输出……14
   1.4.1 控制台的输入：Read 和 ReadLine……15
   1.4.2 控制台的输出：Write 和 WriteLine……15
1.5 程序的注释……15
   1.5.1 模块（类）注释规范……16
   1.5.2 类属性注释规范……16
   1.5.3 方法注释规范……16
   1.5.4 代码间注释规范……16
1.6 习题练习……17

第 2 章 数据类型和表达式……18

2.1 数据类型简介……18
2.2 值类型……19
   2.2.1 简单类型……19
   2.2.2 结构类型……22
   2.2.3 枚举类型……23
2.3 引用类型……24
2.4 null 和 void……24
   2.4.1 null……24
   2.4.2 void……24
2.5 变量和常量……25
   2.5.1 变量……25
   2.5.2 常量……29
2.6 运算处理……29
   2.6.1 算术运算……30
   2.6.2 赋值运算……32
   2.6.3 关系运算……32
   2.6.4 逻辑运算……33
   2.6.5 位运算……33
   2.6.6 条件运算……35
   2.6.7 其他运算符……36
   2.6.8 运算符的优先级别……36
2.7 指针类型……37
   2.7.1 指针的定义与声明……37
   2.7.2 指针的内容……37

2.7.3 指针的运算……38
2.7.4 指针的使用……40
2.8 习题练习……41

第 3 章 数据类型转换……43

3.1 装箱和拆箱……43
3.1.1 装箱（Boxing）……43
3.1.2 拆箱（Unboxing）……44
3.2 隐式数据类型转换……45
3.2.1 隐式数值类型转换……45
3.2.2 隐式枚举类型转换……46
3.2.3 隐式引用类型转换……47
3.3 显式数据类型转换……48
3.3.1 显式数值类型转换……48
3.3.2 显式枚举类型转换……50
3.3.3 显式引用类型转换……50
3.4 习题练习……51

第 4 章 计算控制——结构化程序设计……53

4.1 如果——条件语句……53
4.1.1 如果，那么——if……53
4.1.2 如果，那么；否则——if,else……55
4.1.3 如果中的如果——嵌套 if,else……56
4.1.4 如果，那么；否则的另一种表达方式——？：……57
4.1.5 多条件选择——switch……58
4.2 会转圈的语句——循环语句……60
4.2.1 for 循环……60
4.2.2 while 循环……63
4.2.3 do/while 循环……65
4.2.4 foreach 循环……66
4.3 跳出循环——跳转语句……67
4.3.1 使用 break 和 continue 语句的区别……67
4.3.2 饱受争议的 goto 语句……68
4.3.3 return 语句的使用……70
4.4 C#预处理器指令……70
4.4.1 使用预处理指令——#define 和#undef……70
4.4.2 条件编译——#if、#elif、#else 和#endif……71
4.4.3 警告与错误信息——#warning 和# error……72
4.4.4 可视编辑器提示——#region 和#endregion……72
4.4.5 指定行号——#line……72
4.4.6 关闭警告消息——#pragma……72
4.5 程序员加油站——递归法……73
4.5.1 递归法概述……73
4.5.2 递归的精髓——求解汉诺塔问题……73
4.6 习题练习……75

第 5 章 字符串、数组与集合……77

5.1 理解字符串……77
5.1.1 字符串类型定义……77
5.1.2 字符串类型应用实例……79
5.1.3 string 与 String 的区别……82
5.2 存储盒子的“仓库”——数组……82
5.2.1 数组的定义……82
5.2.2 二维数组……85
5.2.3 多维数组和交错数组……87
5.3 程序员加油站——数组的应用……88
5.3.1 把数组作为参数传递给方法……88
5.3.2 使用数组模拟栈操作……89
5.3.3 数据的排序与冒泡排序算法……91
5.4 一个与数组相似的新类型——集合……92

5.4.1 集合的概述……93
5.4.2 集合类的使用实例……93
5.4.3 集合与数组的区别……97
5.5 习题练习……97

第6章 C#程序员UML建模基础……100

6.1 为什么要建模……100
6.2 UML概述……101
6.3 用例建模……101
6.3.1 用例图……101
6.3.2 用例描述……102
6.3.3 用例建模实例……102
6.4 类图……103
6.4.1 绘制类图……104
6.4.2 类图中的各种关系……105
6.5 活动图……107
6.6 状态图……107
6.7 序列图……108
6.8 习题练习……109

第2部分 C#与面向对象程序设计

第7章 初识面向对象程序设计……113

7.1 万物皆为对象……113
7.2 初识封装……114
7.3 类和对象……114
7.3.1 类的声明……114
7.3.2 类的成员……115
7.3.3 类的成员的可见性……116
7.3.4 对象的创建与使用……117
7.3.5 静态成员与非静态成员……118
7.3.6 this关键字……118
7.4 对象的创建与销毁……119
7.4.1 构造函数……120
7.4.2 静态构造函数……122
7.4.3 析构函数……124
7.5 类的执行工具——方法……124
7.5.1 方法的声明……125
7.5.2 方法的返回值……125
7.5.3 方法的参数……126
7.5.4 方法的重载……127
7.6 域、属性和索引指示器……128
7.6.1 域……128
7.6.2 属性……131
7.6.3 索引指示器……133
7.7 再论封装……134
7.7.1 用传统的读、写方法封装……134
7.7.2 用属性来实现封装……135
7.8 命名空间……135
7.8.1 命名空间的声明……136
7.8.2 成员与类型声明……137
7.8.3 使用指示符……137
7.9 习题练习……140

第8章 复用现有的代码——继承与多重继承……142

8.1 继承的意义……142
8.1.1 继承的引入……142
8.1.2 为何要使用继承……143
8.2 用UML图描述继承……144
8.3 父与子——类的继承……145
8.3.1 基类与派生类……145
8.3.2 继承中的构造函数……151
8.3.3 访问和隐藏基类方法……152
8.3.4 虚方法与重写方法……153
8.3.5 抽象类和抽象方法……154
8.3.6 密封类和密封方法……155
8.4 继承与访问修饰符……156
8.4.1 可见性访问修饰符……156
8.4.2 其他访问修饰符……157
8.5 双亲与子——多重继承……157

8.5.1 水陆两用汽车的类图设计 ···157
8.5.2 用接口实现多重继承 ···158
8.5.3 用扩展方法实现多重继承 ···160
8.6 C#的继承规则 ···161
8.7 再论继承与封装的关系 ···161
8.8 程序员加油站——又论冒泡排序算法 ···162
8.9 习题练习 ···166

第 9 章 改写对象的行为——多态 ···169

9.1 笔与其派生类的写方法 ···169
9.2 多态的类型 ···170
9.2.1 编译时的多态性 ···170
9.2.2 运行时的多态性 ···170
9.3 多态的实现 ···172
9.3.1 通过接口实现的多态性 ···172
9.3.2 通过继承实现的多态性 ···174
9.3.3 通过抽象类实现的多态性 ···174
9.4 程序员加油站——经典的图形游戏 ···176
9.4.1 类图分析 ···176
9.4.2 代码实现与分析 ···177
9.4.3 游戏的客户端 ···181
9.5 习题练习 ···183

第 10 章 多功能的运算符——运算符重载 ···184

10.1 引入运算符重载 ···184
10.2 运算符重载的规则 ···185
10.3 一元运算符的重载 ···186
10.4 二元运算符的重载 ···188
10.5 比较运算符的重载 ···190
10.6 习题练习 ···191

第 11 章 软件模块之间的协定——接口 ···192

11.1 打印机程序的困惑 ···192
11.2 接口的意义 ···193
11.2.1 软件的多层设计思考 ···193
11.2.2 组件与接口 ···193
11.2.3 组件化程序设计思考 ···194
11.3 接口的定义 ···195
11.4 接口的成员 ···195
11.5 在 UML 中对接口的描述 ···197
11.6 接口的继承与多重继承 ···198
11.7 接口成员访问 ···199
11.8 接口的实现 ···200
11.8.1 在类中实现接口 ···201
11.8.2 显式实现接口成员 ···203
11.8.3 显式与隐式实现接口成员的区别 ···205
11.8.4 接口映射 ···205
11.8.5 接口的重实现 ···209
11.9 抽象类与接口 ···210
11.9.1 在抽象类中实现接口 ···210
11.9.2 抽象与接口的区别 ···210
11.9.3 抽象与接口的使用 ···211
11.10 接口作为返回值与参数 ···211
11.11 程序员加油站——打印机程序的设计 ···214
11.12 习题练习 ···216

第 12 章 面向对象设计原则与 UML 描述 ···218

12.1 腐化的软件设计 ···218
12.1.1 设计的臭味 ···218
12.1.2 软件为何会腐化 ···220
12.2 简单就是美——单一职责原则（SRP） ···220
12.2.1 从 Communication 类的设计来看待单一职责原则 ···221
12.2.2 分离耦合的职责 ···222
12.3 修改封闭扩展开放——开放—封闭原则（OCP） ···222

12.3.1 开封—封闭原则概述……222
12.3.2 银行储蓄业务的分析与设计……223
12.4 子类型替换基类型——Liskov替换原则（LSP）……226
12.4.1 Liskov 替换原则概述……226
12.4.2 违反 Liskov 替换原则的场景……226
12.4.3 用提取公共部分的方法来代替继承……228
12.5 依赖于抽象——依赖倒置原则（DIP）……229
12.5.1 依赖倒置原则概述……229
12.5.2 再论银行储蓄业务的设计……230
12.6 分离接口——接口隔离原则（ISP）……232
12.6.1 接口污染……232
12.6.2 分离客户就是分离接口……234
12.6.3 使用委托分离接口……235
12.6.4 使用多重继承分离接口……235
12.7 习题练习……236

第 3 部分 C#语言高级特性

第 13 章 程序的体检医生——异常处理……239
13.1 异常处理概述……239
13.2 C#异常处理类……240
13.3 try、catch 和 finally……241
13.3.1 try 和 catch 的用法……241
13.3.2 finally 的用法……244
13.4 把异常传给调用者……245
13.4.1 调用者处理……245
13.4.2 抛出异常……245
13.4.3 重发异常……246
13.4.4 添加异常信息……247
13.5 用户自定义异常……248
13.5.1 定义自己的异常类……248
13.5.2 从自己的代码中抛出异常……248
13.6 细说异常使用……249
13.6.1 何时考虑抛出异常……249
13.6.2 在异常处理程序中做什么……249
13.6.3 在何处放置异常处理程序……250
13.6.4 异常与返回错误代码的对比……250
13.6.5 永远不要预测 bug 造成的后果能够被异常处理程序所捕获……250
13.7 习题练习……250

第 14 章 房屋中介与租房——委托与事件……252
14.1 从房屋中介与租房看待委托与事件……252
14.2 又论“Hello，World”……253
14.3 委托的概述……254
14.3.1 委托的声明……254
14.3.2 将方法绑定到委托……256
14.4 事件……257
14.4.1 引入事件……258
14.4.2 声明事件……260
14.4.3 使用事件……260
14.4.4 访问器形式的事件……261
14.4.5 在接口中声明事件……263
14.5 多播委托……264
14.6 委托中的协变和逆变……266
14.7 匿名方法……268
14.7.1 什么是匿名方法……268

14.7.2 匿名方法的参数与返回值……268
14.7.3 匿名方法块规则……269
14.7.4 匿名方法的外部变量……269
14.8 习题练习……272

第 15 章 程序集与反射……274

15.1 再论“Hello, World”……274
15.1.1 创建“Hello, World”的业务逻辑类库……274
15.1.2 详解 C#程序集版本控制文件……275
15.1.3 创建“Hello, World”应用层……277
15.2 反射的概述……278
15.2.1 什么是反射……278
15.2.2 反射的作用……279
15.3 反射的实际使用……280
15.3.1 反射 AppDomain 的程序集……280
15.3.2 利用反射获取类型信息……281
15.3.3 通过反射创建类型的实例……282
15.3.4 利用反射动态调用类成员……284
15.4 反射的性能……286
15.5 习题练习……286

第 16 章 序列化与反序列化……287

16.1 序列化的概述……287
16.1.1 从权限系统来看待序列化和反序列化……287
16.1.2 序列化的作用……288
16.1.3 为序列化配置对象……289
16.2 序列化和反序列化方式……290
16.3 使用 BinaryFormatter 进行序列化……291
16.4 使用 SoapFormatter 进行序列化……293
16.5 使用 XmlSerializer 进行序列化……295
16.6 自定义序列化……297
16.6.1 使用 ISerializable 自定义序列化……298
16.6.2 使用特性自定义序列化……301
16.7 序列化过程中的步骤……302
16.8 序列化准则……303
16.9 习题练习……303

第 17 章 项目小组和程序员——进程和线程……304

17.1 进程与线程概述……304
17.1.1 从打印工资报表看待进程与线程……304
17.1.2 进程与线程的关系……306
17.2 .NET 平台下与进程进行交互……306
17.2.1 Process 类简介……306
17.2.2 简单的进程操作实例……307
17.3 初识多线程操作……310
17.3.1 System.Threading 概述……310
17.3.2 一个简单的线程例子……312
17.3.3 ThreadStart 的使用……313
17.3.4 线程的并行操作……314
17.3.5 终止线程……316
17.3.6 前台线程与后台线程……318
17.3.7 线程的优先级……319
17.4 线程中存在的问题……321
17.4.1 竞争条件……321
17.4.2 锁定……323
17.5 线程同步……325
17.5.1 没有同步的情况……325
17.5.2 lock 语句……326
17.5.3 Monitor 类……328

17.5.4 使用 lock 与 Monitor 来解决生产者与消费者的问题……330
17.5.5 Events 类……333
17.5.6 互斥对象……336
17.6 多线程的自动管理……336
17.6.1 线程池……337
17.6.2 定时器……340
17.7 习题练习……343

第 18 章 万能模板——泛型……345

18.1 泛型的引入……345
18.1.1 为什么要使用泛型……345
18.1.2 创建和使用泛型……347
18.1.3 命名规则……348
18.2 泛型类……348
18.2.1 声明泛型类……348
18.2.2 类型参数……349
18.2.3 泛型类的继承……350
18.2.4 泛型类的成员……350
18.2.5 泛型类中的静态变量……351
18.2.6 泛型类中的静态构造函数……351
18.2.7 访问受保护的成员……352
18.2.8 泛型类中的方法重载……352
18.2.9 创建泛型类的注意事项……353
18.3 泛型接口……353
18.3.1 实现接口的惟一性……353
18.3.2 显式接口成员实现……354
18.4 泛型委托……355
18.5 泛型方法……355
18.5.1 创建和使用泛型方法……355
18.5.2 abstract、virtual 和 override 泛型方法……356
18.5.3 泛型静态方法……357
18.5.4 类型实参推断……358
18.6 参数约束……358
18.7 习题练习……360

第 19 章 C# 3.0 语言新特性……364

19.1 隐式类型……364
19.1.1 隐式类型局部变量的声明与使用……364
19.1.2 隐式类型数组……365
19.1.3 隐式类型局部变量的使用限制……366
19.1.4 隐式类型局部变量的使用注意事项……366
19.2 匿名类型……367
19.2.1 引入匿名类型……367
19.2.2 创建和使用匿名类型……368
19.2.3 匿名类型与隐式类型变量的区别……369
19.3 扩展方法……369
19.3.1 定义扩展方法……369
19.3.2 导入扩展方法……370
19.3.3 调用扩展方法……370
19.3.4 扩展方法使用注意事项……372
19.4 对象和集合初始化器……372
19.4.1 引入对象初始化器……372
19.4.2 在初始化语法中调用自定义构造函数……374
19.4.3 初始化内部类型……375
19.4.4 集合初始化器……376
19.5 Lambda 表达式……377
19.5.1 创建 Lambda 表达式……377
19.5.2 Lambda 表达式转换……378
19.5.3 类型推断……379
19.5.4 重载抉择……381
19.5.5 表达式树……381
19.6 习题练习……382

第 4 部分　基于 UML 的面向对象分析与设计过程

第 20 章　基于 UML 的面向对象分析与设计过程……384
20.1　细说需求分析……384
20.1.1　需求分析流程……385
20.1.2　基于 Web 通用框架的需求描述……387
20.2　构造业务用例图……387
20.2.1　确定系统的参与者……387
20.2.2　确定系统的业务用例……388
20.2.3　绘制业务用例图……389
20.3　使用活动图来描述业务用例……389
20.3.1　绘制操作子系统插件菜单的活动图……389
20.3.2　绘制管理用户权限的活动图……390
20.4　构造系统用例图……390
20.5　用例规约……391
20.5.1　基本流……392
20.5.2　备选流……393
20.5.3　特殊需求……393
20.5.4　前置条件与后置条件……393
20.5.5　描述用例规约实例……393
20.6　绘制业务领域类图……396
20.7　系统架构设计……397
20.8　数据库设计……398
20.8.1　数据库表设计……398
20.8.2　表关系设计……399
20.9　总体类图设计……400
20.10　模块详细设计与编码……401
20.10.1　数据库通用操作组件：IBHDataAccess……401
20.10.2　用户权限管理接口：IBHPowerMangageInterface……402
20.10.3　用户权限管理业务逻辑操作层：IBHPowerManageClass……404
20.10.4　用户权限管理页面表示层：IBHPowerManage……405
20.10.5　Web 通用框架：IBHWebFramework……406
20.11　Web 通用框架的部署与测试……408
20.12　习题练习……410

预备课：学习从这里开始

俗话说："万事开头难"，学习 C#语言也一样。如果有一个好的开始，会让我们提高学习的兴趣，增加学习的信心，当然也增加一点学习的成就感。

下面，我们就来阐述学习程序设计前需要了解的一些概念性知识和学习程序设计的方法，并且对 UML 和 C#作简要的介绍，使读者对 C#与 UML 有个初步的认识，从而有个良好的学习开端。主要内容有如下几个方面。

- 软件与程序的关系。
- 面向对象与 UML 的初步认识。
- .NET 与 C#的概述。
- 学习程序设计的方法。

1. 软件=程序+文档

不可否认，我们的日常工作和生活越来越离不开计算机。正是有了计算机的帮助，我们的工作效率才能得到明显的提升，社会的发展才会如此神速。然而，透过显示屏，我们究竟是通过怎样的方式和计算机交互的呢?

答案是计算机软件（Software）。

Windows 操作系统、Word、Excell、QQ 等都是我们常用的软件。软件提供了平台和界面，让人们把想要做的事情通过各种直接的方式与处理器进行交互，然后由软件指挥计算机硬件去完成。

更加科幻一些，我们创造出各种机器人，软件就是机器人的大脑，硬件则像机器人的躯体，大脑指挥躯体去完成各项动作。

那么，软件与程序（Program）又有什么关系呢? 我们先来看看下面的公式：

软件 = 程序 + 文档

从上述公式可以看出，软件是由若干个相关的程序、运行这些程序所需要的数据和相关文档（例如帮助文档）等多个文件组成的。而程序则是计算机执行一系列有序的动作的指令集合。通过一个程序，可以使计算机完成某一类有着共同特点的工作。通常，计算机程序要经过编译和链接而成为一种

人们不易理解而计算机理解的格式，然后运行。未经编译就可以运行的程序通常称之为脚本程序。

软件是包含程序的有机集合体，程序是软件的必要元素。任何软件都有可运行的程序，至少一个。

2. 程序起什么作用

程序起什么作用？答案很简单：那就是使计算机能够帮助我们解决日常的各种计算处理、办公管理等工作。说到这里，又一个问题出现了：如何让程序帮助我们解决这些问题呢？

其实，答案很简单，那就是“算法（Algorithm）”。我们可以把算法理解为有基本运算及规定的运算顺序所构成的完整的解题步骤。或者看成按照要求设计好的有限的确切的计算序列，并且这样的步骤和序列可以解决一类问题。下面，我们以五子棋游戏的设计为例，用传统的面向过程的设计思路来描述其算法。

（1）开始游戏，初始化各种参数；

（2）黑子先走棋；

（3）绘制棋局画面；

（4）判断输赢；

（5）轮到白子走棋；

（6）绘制棋局画面；

（7）判断输赢；

（8）返回步骤 2；

（9）输出最后下棋结果。

以上便是用自然语言描述的五子棋游戏的算法。程序设计所要做的便是探求这种能解决一类问题的算法，并且要将这种算法用计算机能够“看懂”的语言表达出来，从而来指挥计算机帮助我们工作。

3. 为何要面向对象

对于上面的五子棋游戏的算法，用心的读者会发现：同样是绘制棋局画面，这样的行为在面向过程的设计中分散在了众多设计步骤中。这样就很可能出现不同的绘制版本，因为通常设计人员会考虑到实际情况进行各种各样的简化。还有一点值得注意的是，当我们以后要修改某一步骤的算法时，就必须要相应地修改它所有的相关步骤，从而增加了工作量和程序的维护难度。那么我们要如何来解决这些问题呢？

要解决这些问题，就要使用面向对象的方法了。在介绍面向对象方法的理论之前，我们先看看如何用面向对象的设计思路来解决五子棋的设计问题。在这里，我们可以简单地把五子棋分为以下三类对象：

- 黑白双方，这两方的行为是一模一样的；
- 棋盘系统，负责绘制画面；
- 规则系统，负责判定诸如犯规、输赢等。