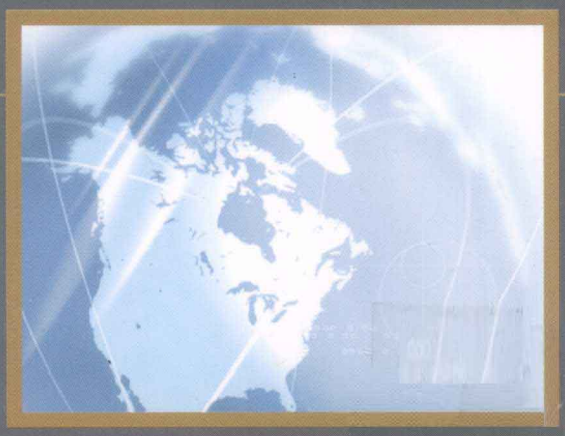
高等学校计算机科学与技术教材

面向对象程序设计
（C#实现）

□ 杨晓光　编著



- 原理与技术的完美结合
- 教学与科研的最新成果
- 语言精炼，实例丰富
- 操作性强，实用性突出

清华大学出版社　●　北京交通大学出版社

高等学校计算机科学与技术教材

面向对象程序设计

（C#实现）

杨晓光　编著

清华大学出版社
北京交通大学出版社
·北京·

内 容 简 介

本书以面向对象程序设计思想与C#相结合为切入点，着力培养学生以面向对象思想分析、设计软件，以及使用C#进行面向对象程序设计。内容涵盖面向对象基础知识、C#语言基础、C#面向对象特性、泛型与集合、Windows程序设计、C#图形图像处理、媒体播放、数据库应用、LINQ、WPF基本知识及应用和综合实例讲解。书中精选大量实用例程和适量习题，并以一个“毕业设计管理系统”综合案例贯穿全书，帮助学生进一步理解如何使用C#语言及面向对象思想进行程序设计。

本书可作为高等院校计算机、软件及信息技术等相关专业的教科书，也适用于各类工程技术人员和程序设计人员参考使用。

图书在版编目(CIP)数据

面向对象程序设计：C#实现/杨晓光编著. —北京：清华大学出版社；北京交通大学出版社，2010.12
（高等学校计算机科学与技术教材）
ISBN 978－7－5121－0419－8

Ⅰ. ①面…　Ⅱ. ①杨…　Ⅲ. ①C语言－程序设计－高等学校－教材　Ⅳ. ①TP312

中国版本图书馆CIP数据核字（2010）第242292号

责任编辑：谭文芳　　特邀编辑：李晓敏
出版发行：清华大学出版社　　邮编：100084　　电话：010－62776969
　　北京交通大学出版社　　邮编：100044　　电话：010－51686414
印 刷 者：北京市德美印刷厂
经　　销：全国新华书店
开　　本：185×260　　印张：25.25　　字数：643千字
版　　次：2011年1月第1版　　2011年1月第1次印刷
书　　号：ISBN 978－7－5121－0419－8/TP·626
印　　数：1～4 000册　　定价：41.00元

前　言

20 世纪 80 年代中期以来，面向对象技术逐渐成为主流的软件开发技术。它以现实世界中的客观事物为中心，以接近人类自然思维方式来认识问题和解决问题。这使得开发的软件更能适应需求变化、更易于复用、有利于维护和修改。

随着面向对象技术水平的提高，面向对象程序设计语言也得到了飞速发展，出现了许多面向对象程序设计语言，如 C ++ 、Java、C#、Eiffel、Smalltalk 等。在众多的面向对象程序设计语言中，C#语言脱颖而出。它囊括了软件开发和软件工程研究的最新成果，如面向对象、类型安全、跨语言、跨平台、自动内存管理、版本控制、组件技术等。它与 C ++ 不同，是一种纯粹的面向对象语言，这一点从其语言特性中展露无遗。

本书力求改变那种只讲 C#语法的传统模式，尝试将面向对象程序设计思想与 C#结合起来，培养学生用面向对象思想分析、设计软件，以及使用 C#进行面向对象程序设计。本书以通俗易懂的语言结合日常生活中的常见事例讲述面向对象理论与 C#基础知识，避免长篇大论，深奥的理论，一切以实用为原则。选材本着够用、实用为原则，对一些不常用部分（如线程、网络编程部分）予以舍弃。考虑到ASP，NET 和 Web Service 部分是 Web 应用程序设计课程的内容，为避免与之重复，也一并予以舍弃。对于表达式、程序流程控制等基础语法予以略讲，因为在其前序课程 C 和C++语言中已经讲得比较详细。本书重点是 C#面向对象部分，将面向对象思想与 C#程序设计融为一体。同时，为了激发学生学习 C#的兴趣，以及展示如何运用 C#，引入 Windows 程序设计和数据库应用部分。

本书主要由两大部分组成。第一部分为基础理论篇，由 6 章组成。第 1 章初步介绍面向对象程序设计的基本知识，以及与 C#有关的 . NET Framework 和 Visual Studio. NET 集成开发环境；第 2 章和第 3 章介绍 C#的基本语法，为使用 C#语言打下基础；第 4 章和第 5 章结合 C#语言讲解抽象、封装、继承、多态等面向对象的基本理论，带领学生进入面向对象领域；第 6 章介绍泛型、集合等 C#高级特性。

第二部分为应用篇，由 9 章组成。第 7 章和第 8 章讲解如何使用 C#进行 Windows 应用程序设计；第 9 章介绍 Windows 程序的事件处理机制；第 10 章和第 11 章介绍 C#的图形图像处理，以及媒体播放的基础知识；第 12 章介绍 C#数据库应用；第 13 章介绍 LINQ 数据源访问技术；第 14 章介绍 WPF 基本知识和基本应用；第 15 章给出一个综合实例，使学生领会如何使用 C#语言以面向对象思想进行程序设计。

书中每章均给出大量代码，以及适量的课后习题，帮助学生领会和应用本章的知识点。特别是将第 15 章的综合实例拆分，分布到书中各个章节中，以降低学生的学习难度，使学生在学完本课程时，能够轻松地完成这个案例，为学生今后开发软件项目打下基础。

本书所提供的程序代码和综合实例均在 Visual Studio. NET 2008 和 Windows XP 环境下调试通过。有需要的读者可通过“http://press. bjtu. edu. cn”网站下载或与出版社联系。

本书可作为高等院校计算机专业、软件专业和信息类专业本科教材，亦适合工程技术人

员参考。

本书由杨晓光编著。在本书编写和教学实践过程中，郑志荣、郭文平、马延宏、傅岚岚、杨晓君、杨亚红等参与了文字整理和程序调试工作，在此向他们表示衷心感谢。由于作者水平有限，加之.NET技术博大精深，书中纰漏和考虑不周之处在所难免，敬请专家和读者批评指正。

编 者

2010年12月

目　录

第 1 章　C#概述

对于程序员来说，选择一门好的程序设计语言，以及先进的软件开发方法，是走向成功的一把金钥匙。C#语言以其简洁、高效、功能强大成为我们的首选语言，而面向对象程序设计经过了近 30 年的发展已经成为软件开发方法的主流。因此，本章围绕 C#语言和面向对象程序设计方法，为大家介绍两个主要的程序设计方法，以及 Microsoft. NET 平台。针对 Microsoft. NET 平台，为大家介绍 . NET 应用程序的运行环境 . NET Framework，以及 . NET 开发环境 Visual Studio. NET。

内容提要：

- ☑ 面向对象程序设计
- ☑ C#语言简介
- ☑ . NET Framework 基础知识
- ☑ Visual Studio. NET 集成开发环境
- ☑ 案例简介

1.1　面向对象程序设计

随着计算机技术的不断发展，人们对软件的需求越来越多，软件的复杂性也越来越高，因此要求在软件开发方法上不断创新。

1. 结构化程序设计

在 20 世纪 60 年代末至 70 年代中期，出现了面向过程程序设计方法，其中较有代表性的是结构化程序设计方法。结构化程序设计的基本思想是采用“自顶向下，逐步求精”的程序设计方法和“单入口单出口”的控制结构。

在程序设计时，结构化程序设计方法强调应先考虑总体，再考虑细节。经过对问题的逐步细化，将问题分解为基本程序模块，从而将原来较为复杂的问题化简为一系列简单的模块。

另外，结构化程序设计认为任何复杂的程序都由顺序、选择和循环三种基本程序结构通过组合、嵌套构成，从而形成一个单入口单出口的程序。

下面以一个例子为大家展示一下结构化程序设计。假设要开发一个银行系统。如果你马上想到的是该银行系统能够对账户进行存款、取款和转账操作。而且进行存款操作时，需要提供“账号”、“存款日期”、“存入金额”等参数信息。进行取款操作时，需要……。恭喜你，你已经用了结构化程序设计方法了。这段程序可能是：

```
long accountID = Convert. ToInt64( Console. ReadLine( ) ) ;
double amount = Convert. ToDouble( Console. ReadLine( ) ) ;
DateTime date = DateTime. Today;
double balance = Acount. MakeDeposit( accountID, date, amount) ;
Console. WriteLine( "你存入了{0}元,账户上现有余额{1}元", amount, balance) ;
```

如果银行说:“我不但有现金账户,还要有支票账户。”显然支票账户的存款、取款和转账操作与现金账户不同。这时,就要加一个变量 accountType,用于表示账户类型。并且在存款、取款和转账操作中,添加以下代码:

```
if( acountType == "现金" )
    …
else
    …
```

用于区分不同账户的操作。如果银行说:“我还要有基金账户。”这时,程序中的许多操作又要修改,加入大量的 if…else,这种修改又会引入代码冗余,给后期维护带来极大麻烦。

虽然结构化程序设计与非结构化程序相比,具有容易调试、可读性好、可维护性好的优点,但其缺点是显而易见的。其缺点为:

- 数据与过程分离;
- 代码重用性差;
- 不能很好地适应需求变化;
- 后期维护困难。

2. 面向对象程序设计

在 20 世纪 80 年代中期至 90 年代,出现了面向对象程序设计方法。它解决了结构化程序设计存在的缺点,从而逐渐成为主流的软件开发方法。

面向对象程序设计以人类的自然思维方式建立问题域模型。以抽象、封装、继承、多态等方式认识问题和解决问题。强调以现实世界中的客观事物为中心,而不是以功能为中心。用对象来描述现实世界中的客观事物,使得解空间与问题空间具有自然的对应关系,有利于对复杂问题给出解决方案。

对于银行系统来说,如果首先想到的是账户,每个账户具有账号、账户类型、明细等,可以对账户进行存款、取款和转账操作,那么用的是面向对象程序设计方法。针对银行系统,可以使用一个抽象基类 Acount,表示账户。它的代码可能是:

```
abstract class Account
{
    protected long acountID;
    protected double balance;
    protected List < Item > items = null;
    public double MakeDeposit( ) ;
    public double WithDraw( ) ;
    public double Transform( ) ;
}
```

然后再派生出 CashAcount、ChequeAccount 与 FundAccount 类,分别表示现金账户、支

票账户和基金账户。每一个类维护与这类账户有关的信息，这些信息不必出现在其他账户类中，从而使得对这类账户的修改不会影响到其他账户。另外，添加新的账户类型时，只需要派生出新的类即可，也不会影响到其他类，这种方式会给后期修改及维护带来极大便利。

由此可以看到，面向对象程序设计具有以下优点：

- 程序设计过程更自然、更易于理解；
- 容易实现软件复用；
- 能够较好地适应需求变化；
- 有利于后期的维护。

1.2　C#及 . NET Framework

1995 年，SUN 公司正式推出了面向对象的开发语言 Java，并提出了跨平台的概念，使得 Java 成为企业级应用开发的首选工具，越来越多的开发人员转向了 Java，使得 Java 如日中天。为了与 Java 相抗衡，2000 年 6 月 22 日，比尔 · 盖茨向全球宣布其下一代软件和服务，即 Microsoft . NET 平台。其战略构想就是帮助用户能够在任何时候、任何地方、利用任何工具（包括计算机、手持设备、智能终端等）都可以获得网络上的信息和服务。

Microsoft . NET 平台主要包括 . NET Framework、. NET 企业级服务器，以及各种 . NET 开发语言和语言工具。

1.2.1　C#简介

C#（读做“C sharp”）是微软公司发布的、简洁的、功能强大的、类型安全的，由 C 和 C ++ 衍生出来的面向对象的编程语言，开发人员可以使用它构建在 . NET Framework 上运行的各种安全、可靠的应用程序。它综合了 C ++ 的强大灵活、Java 的简洁、VB 和 Delphi 的简单易用及可视化操作的特性，从而成为 . NET 开发的首选语言。

说起 C#，它还有一段非常有趣的历史。1995 年 Java 语言一经出现，就得到了迅猛发展。微软也不甘落后，很快也推出了自己基于 Java 语言的编译器 Visual J ++，Visual J ++ 在最短的时间里由 1.1 版本升级到了 6.0 版本，成为业界公认的优秀 Java 编译器。但其编写的应用程序主要运行于 Windows 平台上，SUN 公司认为 Visual J ++ 违反了 Java 的许可协议，即 Java 开发平台的中立性，因而对微软提起了诉讼。微软为了应对，在 1998 年 12 月，启动了一个全新的语言项目——COOL，它是一个专门为 . NET Framework 量身定做的纯面向对象语言。在 1999 年 7 月，完成了 COOL 语言的一个内部版本。在 2000 年 2 月，微软将 COOL 语言正式更名为 C#。

在 2000 年 7 月，微软发布了 C#的第一个预览版本。在 2002 年 2 月，随着 Visual Studio . NET 2002 的发布，C# 1.0 诞生了。在 2003 年 5 月，发布了 C# 1.1，它提出了纯粹的面向对象概念。在随后的 C# 1. x 中，实现了面向对象的所有概念。

2004 年 6 月，微软推出了 C# 2.0。C# 2.0 为我们带来了泛型编程这一新概念，以及匿名方法、Lambda 表达式等内容。

2005 年 9 月，C# 3.0 为我们带来了 LINQ（Language Integrated Querys 语言集成查询）。

LINQ 以一种面向对象的语法实现对各种数据源的查询工作,使得我们可以用类似于 SQL 语句的语法从数据源中获得我们需要的数据。

综上所述,可以看出 C#为我们带来了许多新特性。其主要特色如下。

(1) 完全面向对象

C#具有面向对象语言的一切特性,例如,封装、继承、多态等。并且所有的变量和方法,包括 Main 方法(主函数),都封装在类定义中。甚至连 int、double 等基础类型都是类。

(2) 简单易学

任何熟悉 C、C ++ 或 Java 的程序设计人员,都可以迅速掌握 C#语言。而且 C#去除了 C ++ 中诸多复杂的语法, 如 C#只支持单一继承,避免了多重继承带来的复杂性。

(3) 安全

C#虽然支持指针,但不建议使用指针,以避免不安全的操作。C#还具有自动内存管理和垃圾回收的特点,既减轻了程序员的负担,又避免了内存泄漏问题。使用委托取代函数指针,从而增强了类型安全和安全性。

(4) 跨平台

用 C#编写的应用程序,可以运行在具有不同操作系统的计算机上,以及手机、PDA 等智能设备上。

(5) 跨语言

用 C#编写的程序能最大限度地和任何支持 . NET 的语言互相交换信息。如用 C#编写的类能够在 VB 语言中被继承,从而派生出新的类。

(6) 强大的 Web 编程能力

C#与 ASP. NET 紧密结合,能够开发出强大的 Web 应用程序。

1.2.2 . NET Framework 基础知识

C#应用程序必须在 . NET Framework(也称为 . NET 框架)上运行,因此有必要了解一些有关 . NET Framework 的基础知识。

1. . NET 框架

. NET 框架是支持构建、部署和运行下一代应用程序(Windows 应用程序和 Web 应用程序)和 Web 服务(XML Web Services)的一个 Windows 组件。它提供创建、部署和运行 . NET 应用程序和 Web 服务的一个环境。

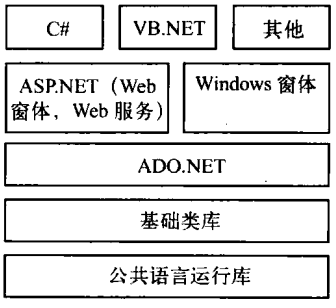


图 1-1 . NET 框架构成

. NET 框架主要由一个称为公共语言运行库(CLR)的虚拟执行系统和一组统一的类库组成,如图 1-1 所示。

. NET Framework 旨在实现下列目标:

- 提供一个一致的面向对象的编程环境,而无论对象代码是在本地存储和执行,还是在本地执行但在 Internet 上分布,或者是在远程执行的;
- 提供一个将软件部署和版本控制冲突最小化的代码执行环境;
- 提供一个可提高代码(包括由未知的或不完全受信任的