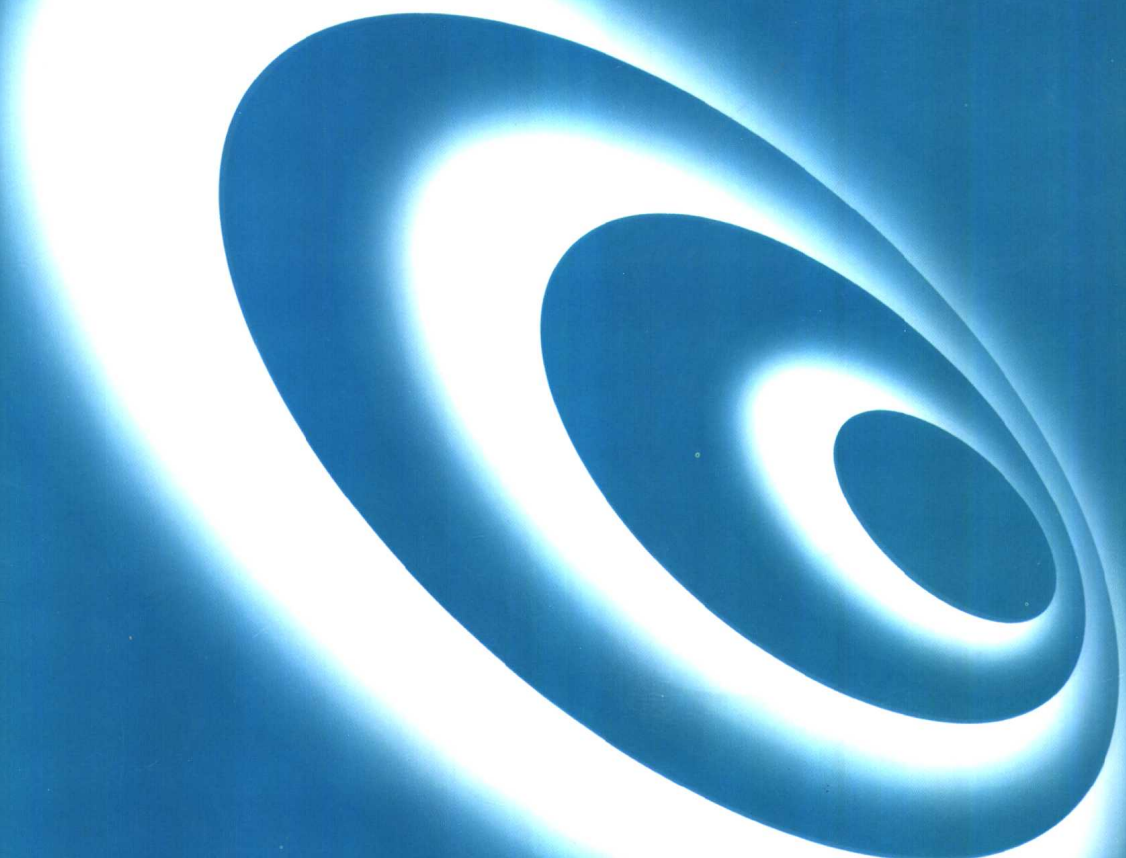
计算机技术培训系列教程

C#程序设计

培训教程

刘甲耀　严桂兰　等编著



机械工业出版社
CHINA MACHINE PRESS

计算机技术培训系列教程

C#程序设计培训教程

刘甲耀　严桂兰　等编著

机械工业出版社

C# 是汇集了 C、C++、VB、Delphi 以及 Java 的优点，加上自身的许多特色，形成的新一代面向组件、面向对象的程序设计语言。本书阐述了 C# 程序设计的方法与技巧，取材广泛，概念清晰，由浅入深。内容包括：简单的 C# 程序设计；类型系统与数据类型；控制台输入/输出；表达式与运算符；程序流控制；类；方法；属性、数组与索引器；属性信息；接口；异常处理；代表与事件处理以及结构与枚举；运算符重载和用户定义的转换；多线程程序设计；元数据查询与文件操作。书中共有 200 多个范例，且均在 Microsoft.NET 平台上通过，读者通过这些范例可领悟 C# 程序设计的技巧与内涵。各章均附有小结与习题，并给出了参考答案。

本书可作为大专院校计算机及相关专业的教材，也可作为 C# 培训班的教材，并可供各行各业从事计算机技术的人员编程使用，从事电子商务系统工程和企业管理的人员也可以本书作为参考。

图书在版编目（CIP）数据

C# 程序设计培训教程/刘甲耀等编著. —北京：机械工业出版社，2004.1
（计算机技术培训系列教程）
ISBN 7-111-13681-0

Ⅰ. C… Ⅱ. 刘… 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 118753 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）
策　　划：胡毓坚
责任编辑：孙　业
责任印制：闫　焱

北京中加印刷有限公司印刷·新华书店北京发行所发行

2004 年 1 月第 1 版·第 1 次印刷
787mm×1092mm　1/16·27.25 印张·677 千字
0001—5000 册
定价：40.00 元

前　言

C#（读为“C Sharp”）是C和C++语言家族中最新的面向组件语言，它是从C和C++派生的一种简单的、现代的、类型安全的面向对象程序设计语言。用户能在Microsoft.NET平台上快速地构建各式各样的应用程序，包括通用网络协议的Internet服务软件、Windows图形用户界面程序、各种数据库、网络服务应用程序，特别适用于构建各式各样的、从高级商务对象到系统的应用程序。使用简单的C#语言结构，可把这些组件转换为Web服务，因为C#的组建和运行依赖于.NET平台，可通过Internet调用任何操作系统上运行的任何语言，因此，C#的各种特性与.NET平台密切联系。.NET的主要组件和一些相关的技术有：公共语言运行库（CLR）、公共语言规范（CLS）、以CLR为目标的丰富语言集、Web服务、Visual Studio.NET。

C#与其他语言相比作了如下重大的改进：

- C#的类型系统使用自动的存储管理。
- C#类型系统是统一的，其任何事物都是一个对象。
- C#支持属性信息，能定义和使用有关组件的声明信息。属性、方法和事件是基础。

C#有以下五大特点：

1. 能提高软件生产率与安全性。
2. 包含了Web程序设计标准。
3. 能消除程序设计错误。
4. 有很强的功能、表达程度与灵活性。
5. 有广泛的相互操作性。

C#用户能在已有的C和C++，乃至Java的基础上，快速地构建C#程序以及Web开发，这就是C#生命力所在。

为适应当前Internet和电子商务系统工程的迅猛发展，我们特编写《C#语言程序设计培训教程》一书，以满足计算机技术人员学习C#语言、大专院校的本科生和研究生开设新一代面向对象程序设计课程以及C#语言培训班教材的需要。

本书有以下三个特点：

1. 适应当前Internet和电子商务迅猛发展，提供最新教材。
2. 取材广泛，由浅入深，重点难点分明，易学、易掌握。
3. 编程方法与实例并举，提供的大量实例均在.NET环境上机通过。

本书主要由广州市私立华联学院刘甲耀、严桂兰教授编写，美国某公司管理总监刘涌博士提供了大量资料，参加编写工作的还有陈妙燕、陈小玲、郑福珠、李燕、高培根。

在计算机程序设计语言快速发展的今天，加上C#的新颖风格，编写中难免在理解与掌握上有定位不准或考虑不周之处，敬请读者谅解与指正。

书中例题、习题代码下载网址：http://www.cmpbook.com（具体链接见本书相关网页）。

编　者

目　录

第 1 章　简单的 C#程序设计

C#是一种简单的、现代的完全面向对象的编程语言，因此，在 C#编程之前，首先介绍面向对象编程的一些基本概念以及 C#的运行环境 Microsoft.NET，然后再介绍简单的 C#程序设计。内容涉及：面向对象编程的基本概念；C#的运行环境 Microsoft.NET；简单的 C#程序的编写与运行；遍历简单 C#代码；C#应用程序设计准则。

1.1　面向对象编程的基本概念

1.1.1　任何事物都是一个对象

1．对象的基本概念

在真正的面向对象语言中，所有问题论域实体都是通过对象表示的（问题论域实体是用户根据具体事物的要求及其复杂程度等，试图求解的问题），对象是面向对象程序设计的主要概念，它把有关的现实世界的实体模型化。

对象是面向对象编程方法的基本成分，每个对象都可用它本身的一组属性和在其上的一组操作来定义。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。任何事物都是一个对象，例如，一个人、一辆汽车、一本书，乃至一种语言、一个图形、一种管理方式，都可作为一个对象。声明实体的方法有：描述实体，包括实体的属性和可执行的操作。例如，对于汽车对象，它的颜色、重量都可以作为对象的属性，它可以执行的操作是行驶、鸣笛等。而面向对象方法，就是发送信息给对象，通过发送信息实现对它的请求，而不管信息在其内部如何存储。面向对象编程的主要原则之一就是封装，封装就是隐藏对象的内部数据和方法，而只提供一个给外界访问的接口。一个对象的内部如何实现它的工作并不重要，只要该对象能进行工作即可。给对象提供一个接口，使用该接口使对象执行一个给定的任务。因此，模拟问题论域的现实世界的对象更易设计和编写程序。根据定义，一个对象不仅包含数据，而且还包含在该数据上工作的方法。这意味着，当我们用一个问题论域工作时，可以更好地设计所需的数据结构。

2．对象与类

类是一种数据类型，它有与之相关的方法。一个对象就是一个类型或类的实例。最合适的定义：一个类就是一个对象的蓝本。作为一个开发者来说，如同一个建筑工程师，创建一个蓝本就是要创建一个房子的蓝本，一旦蓝本完成，则只要具有给定任一房子形式的蓝本即可。如此的表示，一个类就是一个给定功能集的蓝本，而创建一个特定类的一个对象，它就具有所构造类的所有功能。

3．实例化

实例化是面向对象编程的一个独特的术语。实例化是创建一个类的实例的行为，该实例

是一个对象。形式为：

```
类名 对象名=new 类名();
```

或

```
类名 对象名=new 类名(参数表);
```

在实例化之后，可通过它的公有（public）成员与这个对象通信，如果没有一个实际的对象，则不能做这一工作。但作为静态成员时例外。

当每个对象都有同样的能力时，每个实例将包含它自己的实例数据，并能分别进行处理。

1.1.2 面向对象程序设计语言的三大原则

所有面向对象的语言都必须支持三个概念：对象、类和继承，即面向对象语言都依赖于封装、继承和多态性三个原则，而封装和多态性则像类和继承一样，可完整地构建面向对象的系统。

1．封装

封装常常称为信息隐藏。封装能隐藏一个对象的本质（内部细节），使它与用户分离开来，并只对用户能直接操作的那些成员提供一个接口。封装对一个类的外部接口（即对类的用户，公有成员是可见的）和它的内部实现细节之间提供边界，封装能使面临一个类的成员仍然是静态的或者不变的成员，而可隐藏更多的动态的和非永久性的类内部细节。在 C#中，封装是依靠对每个类成员指定一个访问修饰符（public，private 或 protected）来达到的。

（1）抽象的设计。抽象表示程序中一个给定空间的问题，程序设计语言本身提供了抽象。面向对象语言允许我们声明名字和接口更精确地模拟现实世界问题论域实体的类，以便使用对象能实现它们。

应该能为类的用户提供最好的抽象设计，以使用户不陷入到类工作的细节中。

类的接口是抽象的实现，例如自动售货机，它内部的情况是完全被封装的，它必须有接受现金及用户的选择项，以实行交换，因此，自动售货机必须具有一个向它的用户表达功能的机构——接口，即通过一个货币投切口及选择所需项目的按钮（是机器接口的一部分）。虽然自动售货机有许多种类，但主要是它的内部结构不同，基本接口则不需要有很多改变。为此，创建一个接口，由它提供给用户访问，并提供与类的内部工作分离的信息与方法。

（2）充分抽象的好处。类的抽象设计在开发可重用软件方面是很重要的，如果开发一个不变的、持续交叉实现改变的静态接口，则只需要比较少的时间来修改应用程序。

用户与实现细节分离，能使得一个完整的系统较容易理解，因而较容易维护。

2．继承

（1）继承的定义。继承是指定一个类与另一个类具有一种关系。通过继承，我们可以创建（或派生）一个基于已有类的新类。因而，可以修改所需的类的状况和创建新的派生类的对象。这种能力是创建一个类层次的精华。在抽象的外部，继承是一个系统的全部设计的最主要部分。派生类是被创建的新类，而基类是派生新类的类。最近派生的类继承了基类的所有成员，因而使得我们能够重用以前的工作。

（2）固有继承的定义。固有继承就是替换性，是指派生类通知的行为对基类来说是可

替换的。它的最重要的法则是掌握构建层次。当创建类层次时，一个派生类在任何继承的接口上，不应再要求并允许和它的基类一样。当拥有一个派生类的引用时，总是要把该类作为它的基类来处理，一个派生类可以添加更为限制有关它的要求的行为，并允许作为它需要的一小部分。

3. 多态性

多态性具有老代码调用新代码的功能，它允许我们扩展或增强系统而又不修改或破坏已有的代码。

多态性有两个好处：第一，它能聚集公用基类的对象，并始终如一地处理它们。由于多态性，当我们调用这些对象的方法之一时，运行时能确保调用正确的派生对象的方法。第二，老代码可以使用新代码。多态性运行时能确保调用正确的派生类的成员，也可添加其他派生的类型到系统，所有代码都继续工作，而无需改变初始的代码。

1.2 C#运行环境 Microsoft.NET 简介

1.2.1 Microsoft .NET 平台

1. 使用.NET 平台的好处

Microsoft.NET 是美国微软开发的平台，是一种面向网络、支持各种用户终端的开发平台环境。它不仅引进了如 C#之类功能强大的语言，而且还对图像、多媒体、Web 开发以及语言开发提供了极大的便利。使用.NET 平台能缩短产品开发时间，简化发行与管理，提高运行效率，其主要好处是：

- 可使用任何编程语言：该平台允许开发人员以任何语言进行开发，使得不同语言开发的程序结合得更紧，并使现有的开发技巧得以继续使用。
- 减少编写代码量：该平台使用了高度模块化设计，使得开发人员将更多的精力集中到处理逻辑事务上，而不必再把时间花在写各种代码上。
- 以 XML/SOAP 为核心：该平台的目标是将软件转化为服务，因此，基于 XML 和 SOAP 系列的集成标准，只需注出所需的方法调用，该平台就能将它们转化为完整的 XML Web 服务。
- 提高了应用程序的可靠性：该平台引入了新的技术，使程序运行更可靠。比如，以该平台来管理内存、线程及进程，能减轻程序员的负担。此外，ASP.NET 还监视 Web 程序的运行，并根据管理员设定的时间间隔，每过一段时间，自动地将此程序重新执行一次。
- 性能更加优化：该平台优化了传统的 Web 程序。ASP.NET 引入了高级的编译技术和缓存特性，获得了比现在的 ASP 程序高 1～2 倍的性能。

2. Microsoft .NET 平台的核心组件

在.NET 平台上，不同的网络之间通过相关的协定联系在一起，网站之间形成自动交流，协同工作，提供最全面的服务。为此，.NET 包含四个核心组件：

- .NET 为构建组件服务，或进行某些服务的访问。例如，文件存储，日历管理（calendar）以及 Passport.NET（身份核实服务）。

- 提供.NET 设备软件，使新设备在 Internet 运行。
- .NET 用户信息，包括如固有的界面，信息媒界和标签的特性，用户创建的文档的自动超链接技术。
- .NET 基础结构，包括.NET 框架，Microsoft Visual Studio.NET，.NET 企业服务器以及 Microsoft Windows.NET。

3. .NET 基础结构

- 大多数开发人员引用 .NET 时都引用 .NET 基础结构。.NET 基础结构涉及组成创建和运行分布式应用程序新环境的所有技术，能使我们开发应用程序的.NET 部分就是.NET 框架。
- .NET 框架包括公共语言的运行期环境（CLR）和.NET 框架类库，称之为基类库（BCL），CLR 可看作.NET 应用程序的虚拟机。所有 .NET 语言的配置中都有 .NET 框架类库。如果掌握 Microsoft 基础类（MFC）或 Borland 的对象视窗库（OWL），就掌握了类库。.NET 框架类库支持从文件 I/O 和数据库 I/O 到 XML 和 SOAP 的一切东西。

应当指出，前面所讲的术语“虚拟机”，并不是指 Java 虚拟机。虚拟机是在一个完全封装环境中，其他操作系统能够工作的一种高级操作系统的抽象。当我们把 CLR 看作虚拟机时，实际上是指在 CLR 的一个封装和管理环境中运行代码，使代码在机器上与其他处理分离。

1.2.2 .NET 框架

1. Windows DNA 与 .NET

.NET 是创建和运行分布式应用程序的新环境，该平台称之为 Windows DNA，它是通过使用 Microsoft 的服务产品解决商务问题的一种平台。

2. 公用语言运行期

CLR 是.NET 的核心。顾名思义，它是一个运行期环境，在该环境中，能使用不同语言编写的应用程序——称之为交叉语言相互操作性。公用语言规范（CLS）是语言编译器必须依附的一组规则，以便在 CLR 运行时创建.NET 应用程序。

管理代码只不过是在 CLR 保护下运行的代码，因而它是由 CLR 管理的代码。

3. .NET 框架类库

.NET 框架类库对提供语言相互操作性是非常重要的，因为它们允许开发人员使用单个程序设计接口。如果我们在 Windows 开发中，常常使用多于一个以上的不同的语言，这特别有用。

公用的类库集在技术上意指所有语言都具有同样的能力，因为它们都必须使用这些类库来完成说明变量之外的任何事情。

4. Microsoft 中间语言与中间编译器

为使不同语言编写的程序容易在.NET 的语言端口实现，微软开发了一种相近于汇编语言的语言，称为微软中间语言（MSIL）。为编译.NET 的应用程序，编译器对输入和输出采取产生一样的 MSIL 源代码。MSIL 是一种可以写应用程序的完整语言。

当编译一个 C#应用程序或用 CLS 依从的语言编写的任何应用程序时，则应用程序被

编译成 MSIL，然后，当应用程序由 CLR 第一次执行时，这个 MSIL 进一步被编译成本地的 CPU 指令（实际上，只有第一次调用它们时才编译被调用的函数）。

5. 统一的类型系统

任何开发环境的主要特点是它的类型系统。.NET 运行库不仅仅是给单个开发人员使用，统一的类型系统是用于交叉的所有 CLS 依从的语言。它也让语言编写人员通过增加新的类型（其作用就像加入的类型）来扩展类型系统。这意味着，作为一个开发人员，可以以一样的方式使用所有的类型。而不管它们是否是.NET 预定义的类型或用户创建的类型。

6. 元数据与映射

CLS 依从的编译器接受源代码作为输入并产生运行期编译（通过 JIT 编译器）和执行的 MSIL 代码。除了映射源代码到 MSIL 指令系列之外，CLS 依从的编译器还嵌入元数据到.EXE 文件。

元数据是描述数据的数据。从这个意义讲，元数据是组成.EXE 文件（例如类型说明和方法实现）的计划性项目的集合。这种元数据类似于用构件对象模型（COM）构件产生的类型库。从.NET 编译器输出的元数据，实际上不仅比我们惯用的 COM 类型库更有意义、更完整，而且元数据也总是嵌入在.EXE 中。

实际上，.NET 构架类库提供了完整的映射方法集，允许任何应用程序（不仅仅是 CLR）询问其他应用程序的元数据。

7. 安全性

任何分布式应用程序开发环境的最重要方面是如何处理安全性，实际上，通过 CLR 装入类，类装载器是.NET 安全系统的一部分。例如，当一个类在.NET 运行期被装入时，就验证安全有关的因素，如可达性规则和自一致性要求。此外，安全性检查确保一部分代码具有资格来访问某些源代码，安全的代码确保决定和识别信息。

8. 调度

.NET 应用程序调度的关键是程序集概念。程序集只不过是作为单个文件或多个文件实体构造的相关的行为包。如何运用应用程序的细节将根据开发一个 Web 服务器应用程序或者 Windows 的应用程序来确定。然而，用引入程序集作为完全封装的功能集，调度可以像复制程序集到目标单元一样简单。

9. 具有非管理代码的相互操作性

非管理代码并不具有管理代码所具有的优点。例如，垃圾收集，统一的类型系统以及元数据。然而，任何人都需要在.NET 环境中运行非管理代码，而非管理代码又不是由.NET 运行期间控制的代码，为此，微软必须放置下列特性到.NET 中，使在.NET 环境中能运行非管理代码：

- 管理代码调用非管理的 DLL 函数。
- 管理代码使用 COM 构件。
- 非管代码使用.NET 服务。

1.3　简单 C#程序的编写与运行

在 Visual Studio.NET 环境下，开发 C#控制台应用程序，可使用.NET 编辑器编写或使用控制台应用程序框架来进行。

1.3.1 使用.NET 编辑器构建与运行 C#程序

1．操作步骤

使用普通的编程方式，就是在 C#.NET 代码编辑器中输入源程序，然后运行。具体步骤如下：

（1）打开代码编辑器（即进入编辑状态）。

① 在 Winwods 桌面上单击“开始”中的“程序”→单击“Microsoft Visual Studio.NET”下的“Microsoft Visual Studio.NET”→进入“Microsoft Development Environment [设计]-起始页”画面。

② 单击“文件（F）”→选择“关闭（C）”。

③ 单击“文件（F）”→选择“新建（N）”下的“项目（P）”→选择项目类型为“Visual C#”→单击模板中的“空项目”→“确定”。

④ 单击“文件（F）”→选择“添加新项目”→选择类别中的“本地项目”项→单击模板中的“代码文件”→“打开（O）”→进入 C#编辑状态（Project1-Microsoft Visual C#.NET[]-CodeFile1.cs 画面）。

（2）在代码编辑器中输入 C#源程序。

（3）存盘。单击“文件（F）”→选择“CoreFile1.cs 另存为”→在“保存位置”中选定保存文件的路径→在“文件名框”中输入用户文件名.cs→单击“保存（S）”。

（4）运行程序。

方法一：单击菜单中的“调试（D）”→选择“开始执行（不调试）（G）”→输出运行结果或等待用户输入数据后产生运行结果。

方法二：单击菜单中的“生成（B）”→选择“生成 Project1”→单击“调试（D）”→选择“开始执行（不调试）（G）”→输出运行结果或等待用户输入数据后产生运行结果。

注意：

① 若输出窗口一闪而过，则应在用户程序最后加入一行。

```
string str=System.Console.ReadLine();
```

或 Console.Read();或 Console.ReadLine();

其意表示程序暂停，使用户看清运行结果后，按〈Enter〉键结束。

② 当程序在“生成”中顺利完成时，则会在输出框中显示生成情况。

③ 当程序在“生成”中遇到编译错误时，则会在输出框中提示错误的类型，双击提示语句，光标会在源程序的某错误行上闪烁，用户可在该行上进行修改，重新编译生成，产生运行结果。

2．编写简单的 C#应用程序

例 1-1 用 C#.NET 编程，在屏幕上输出“Welcome to Programming Using C#.NET”。

方案一：

```
//CSharpApp.cs
class CSharpApp
{
```