高职高专计算机任务驱动模式教材

C#语言程序设计

李继武　编著

清华大学出版社

高职高专计算机任务驱动模式教材

C#语言程序设计

李继武　编著

清华大学出版社
北京

内容简介

C#语言是近年来非常流行的一种编程语言，它依托于强大的.NET平台，可方便地开发各种应用程序，它像C++一样功能强大，却像VB一样使用方便，具有很好的发展前景。

本书共8章，第1章简要介绍C#语言诞生的背景、特点以及.NET框架的组成等内容；第2章讲解如何通过C#语言进行结构化程序设计，主要包括数据类型、变量、操作符与表达式、语句和数组等内容；第3章详细讲解如何通过C#语言进行面向对象程序设计，主要包括各种类成员设计、面向对象程序设计的三大支柱以及接口和委托等内容；第4章讲解如何通过C#语言进行I/O程序设计，主要包括目录与文件操作、字符流读写文本文件和字节流读写文件等内容；第5章详细讲解如何通过C#语言进行Windows Forms程序设计，主要包括公共控件、容器控件、工具控件和对话框等内容，最后讲解一个类似Notepad的示例程序；第6章讲解如何通过C#语言进行ADO.NET程序设计，主要包括SQL Server 2005基础、ADO.NET基础、数据源、数据集和数据绑定等内容；第7章讲解如何通过C#语言进行ASP.NET程序设计，主要包括ASP.NET Web窗体和ASP.NET服务器控件等内容；第8章详细讲解一个实战案例(上市公司财务分析软件)的设计与实现过程。

本书结构编排巧妙，内容详略得当，案例设计合理，讲解深入浅出。

本书可作为高职高专院校开设C#语言程序设计课程的教材，也可作为社会上各种计算机培训班学习C#语言的教材或读者自学C#语言的参考书。

图书在版编目(CIP)数据

C#语言程序设计 / 李继武编著. —北京：清华大学出版社，2011.4
(高职高专计算机任务驱动模式教材)
ISBN 978-7-302-24439-4

Ⅰ. ①C… Ⅱ. ①李… Ⅲ. ①C语言－程序设计－高等学校：•技术学术－教材
Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2010)第260794号

责任编辑：张　景
责任校对：李　梅
责任印制：李红英

出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦A座
http://www.tup.com.cn　　**邮　　编**：100084
社　总　机：010-62770175　　**邮　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：北京富博印刷有限公司
装 订 者：北京市密云县京文制本装订厂
经　　销：全国新华书店
开　　本：185×260　　**印　　张**：17　　**字　　数**：386千字
版　　次：2011年4月第1版　　**印　　次**：2011年4月第1次印刷
印　　数：1～3000
定　　价：33.00元

产品编号：038187-01

前　言

开发 Windows 软件的程序员都希望又快又好地开发出满足用户需求的软件产品，当然这除了要依靠程序员的能力和勤奋以外，还要有好用的软件开发平台，正所谓“工欲善其事，必先利其器”。自 2002 年微软推出 C＃语言和. NET 平台以来，经过 8 年多的发展，现在已经有越来越多的程序员开始使用 C＃语言和. NET 平台来开发各种应用软件。

作为一个软件开发平台，. NET 框架提供了一个庞大的类库，该类库以面向对象的方式全新封装了 Windows 底层的各种 API 函数，通过它程序员可以高效地开发各种应用软件，从而摆脱“编程语言＋WIN32 API 函数”的低效率软件开发模式。在. NET 框架类库中有两个非常重要的技术，那就是 ADO. NET 和 ASP. NET，前者是数据访问平台，后者是 Web 开发平台，它们为开发目前热门的数据库程序和 Web 应用程序提供了强有力的支持。另外，利用. NET 类库开发的程序将编译成 MSIL（微软中间语言）代码，并需要在. NET 框架中的托管平台 CLR（公共语言运行时）上运行，CLR 将为其提供安全保证和垃圾回收等功能。

C＃语言是一种优雅的编程语言，它汲取了目前几种主流编程语言，诸如 C++ 、Java 和 Visual Basic 的精华，拥有语法简洁、面向对象、类型安全和垃圾回收等现代语言的诸多特征，从而成为利用. NET 平台开发应用程序的最佳编程利器。

为了更好地利用 C＃语言（当然还包括其他支持. NET 平台开发的语言）和. NET 框架类库开发应用程序，微软开发了当今最优秀的集成开发环境之一——Visual Studio. NET，这是一个多语言统一的、多组件集成的、可视化的编程环境，它可以有效地加速应用软件的开发过程，快速构建商业中间层组件，并有助于开发人员构建可靠的、可伸缩的企业级解决方案。

本书是一本详细讲解 C＃语言程序设计的教材，全书共 8 章。第 1 章简要介绍 C＃语言、. NET 框架和 Visual Studio. NET 开发工具；第 2 章讲解 C＃语言结构化程序设计；第 3 章讲解 C＃语言面向对象程序设计；第 4 章讲解如何通过 C＃语言进行 I/O 程序设计；第 5 章讲解如何通过 C＃语言进行 Windows Forms 程序设计；第 6 章讲解如何通过 C＃语

言进行 ADO. NET 程序设计；第 7 章讲解如何通过 C# 语言进行 ASP. NET 程序设计；第 8 章讲解实战案例（上市公司财务分析软件）的开发过程。

本书精心设计 80 个示例程序，每个程序都对关键的知识点做了透彻的演示，本书还精心设计一个综合性的实战项目，该项目将贯穿本书的重点与难点都巧妙地融合起来，起到了实战提高的效果。

本书作者长期在一线从事各类软件项目的开发工作，拥有丰富的软件设计与开发经验，并担任过多个知名软件培训中心的软件开发培训讲师，拥有丰富的教学经验，这些经验为本书的写作提供了很好的保证。

写作本书的过程就像披荆斩棘一样从 C# 语言和. NET 框架的路途上走过，我希望凭着自己的经验和努力，为读者开辟一条学习这些知识的坦途，如果能得到读者的认可，那么一切辛劳都是值得的。

最后请读者不吝指教书中的不妥之处，我愿与您共同提高。

李继武

2011 年 2 月

目 录

第1章 概 述

1.1 C#语言简介

1999年，微软公司秘密开发了一个名叫COOL的新语言，并于2000年6月26日在美国奥兰多（美国佛罗里达州中部城市）举行的“专业开发者大会”（Professional Developer Conference，PDC）上推出了这个新语言，并改名为C#（读作C Sharp）。

1.1.1 C#语言诞生的背景

早在1995年，Sun公司的James Gosling（詹姆斯·格斯林）就开发出了Java语言，这门语言简单、面向对象、功能强大，并且由于JVM（Java Virtual Machine，Java虚拟机）的缘故，它还可以跨平台运行。这些特性使Java语言逐渐成为企业级应用系统开发的首选工具，越来越多使用C/C++开发软件的人员开始转向使用Java进行应用系统开发。

微软公司感觉到了这种压力，于是在Anders Hejlsberg（安德斯·海尔斯伯格）的领导下，迅速开发出了Java语言的微软版——Visual J++，这个产品很快成为强大的Windows应用开发平台，并成为业界公认的优秀Java编译器。

Sun公司以Visual J++主要用在Windows平台系统开发为由，起诉微软公司违反了Java开发平台的中立性，并终止了对微软公司的Java授权，使微软公司陷入了被动局面。

为了彻底摆脱这种局面，微软公司于1998年12月启动了一个全新的语言项目——COOL，它是C#语言的前身，其首席开发者仍然是Anders Hejlsberg。Anders Hejlsberg是一位杰出的软件天才，是Borland公司的创始人之一，也是Delphi之父，由于后期在Borland公司不受重用，被比尔·盖茨慧眼识才、三顾茅庐请到微软公司主持Visual J++的开发工作。

由于Visual J++语言陷入僵局，Anders Hejlsberg干脆另起炉灶，于1999年开始了C#语言的开发历程。同年7月，COOL语言完成了一个内部版本，2000年2月正式更名为C#。2000年7月，微软公司发布了C#语言的第一个预览版（又名Preview版，软件开发商为了满足那些对新版本很关注的人而发布的可以看到大部分功能的测试版）。在此后一年多的时间里，微软公司不断地修补各个测试版中的Bug（缺陷或漏洞），直到2002年2月，微软公司终于推出了C#语言的第一个正式版——C#1.0，关于C#语言的版本变化将在1.3节介绍。

1.1.2 C#语言的特点

C#语言是一门简单、现代、面向对象和类型安全的编程语言。

1. 简单

C#语言是一门简单的编程语言。当然，简单是相对的概念。比如C/C++这类语言，它们的表达能力很强，但是比较琐碎。换句话说，程序员需要关注的细节特别多。而C#语言借鉴了C/C++以及Java语言的优点，避免了它们的不足，语法上变得简洁而优雅。

2. 现代

C#语言是一门现代的编程语言。说到其现代就要谈一谈编程语言的历史。

自计算机诞生以来，最初的机器语言只能由当时的科学家来使用，之后汇编语言开始流行，但学起来相当难，鉴于此，高级语言诞生了，同机器语言和汇编语言相比，高级语言不依赖于计算机硬件，而且学习难度显著降低了，并且这期间计算机开始逐渐普及，这使得通过高级语言进行编程成为一种社会上的职业需求。

由于计算机硬件的发展，软件规模越来越大、需求越来越多，这种日益暴露的软件危机促使高级语言由早期传统的编程模式向现代编程模式转变，这其中就包括由结构化编程向面向对象编程过渡，以及由只支持单一平台开发向跨平台开发演变等。具体来说，C#语言支持面向对象编程、支持基于CLR(Common Language Runtime，公共语言运行时)虚拟机模式的跨平台开发、支持内存的自动管理等现代语言编程机制。

3. 面向对象

C#语言是一门面向对象的编程语言。目前，主流的编程语言几乎都支持面向对象编程，比如Java、VB、C++等，同它们相比，C#语言在支持面向对象编程方面做得更纯粹、更彻底。比如，通过C++可以面向对象编程，也可以不面向对象编程。从这个角度看，C++是通用的编程语言，而不是纯正的面向对象编程语言。当然，这不是C++语言的缺点，而是C++语言犀利的表现，但在面向对象理论大行其道的今天，C++对不面向对象编程的支持使得其语法更复杂、学习难度更大。

4. 类型安全

C#语言是一门类型安全(强类型)的语言。所谓类型安全就是指不可以将A类型强制转换成B类型从而对转换后的A类型进行B类型上定义的操作。换句话说，变量类型定义后不能将其再转换成其他类型(非本类型或非本类型的子类型)。

由于类型安全直接涉及内存安全，所以保证类型安全是CLR的使命之一，C#可以直接享受类型安全所带来的好处。

总而言之，C＃是一门潜力很大的编程语言，读者可以通过本书学到它的经典部分。

1.2　.NET 框架简介

没有.NET 框架而单纯说 C＃语言是没有意义的，因为 C＃语言编程离不开.NET 框架的支持，如果非要比较二者的重要性，那显然.NET 框架更重要。因为没有 C＃语言，还有其他语言（比如 VB.NET）可以使用.NET 框架，而没有.NET 框架，C＃语言将无法生存。那么，.NET 框架到底是什么？

1.2.1　.NET 框架诞生的背景

回顾 Windows 软件开发的历史可以知道，语言从来没有离开过框架。

1. C/API 开发模式

早期的 C 语言开发需要程序员花大气力掌握数千 Windows API（Application Programming Interface，应用程序编程接口）函数，然后以一种很费时的方式开发出成功的应用。

2. C++ /MFC 开发模式

C++ 给程序员带来了面向对象的编程理念，摆脱了过程化编程的冗长与乏味，而且有了以 C++ 类的形式封装了 Windows API 的 MFC（Microsoft Foundation Classes，微软基础类库）框架，这些都大大减少了应用程序开发人员的工作量。

不过，使用 C++ 与 MFC 开发程序依然是个艰难且易犯错误的过程。

3. Java/J2EE 开发模式

Java 语言在保留了 C++ 强大的同时剔除了 C++ 中令人生厌的语法。伴随着网络的兴起，Java 及 J2EE（Java 2 Enterprise Edition）框架赢得了越来越多程序员的青睐，它们踩准了软件开发趋势的节奏，获得了市场。

通过 Java 语言和 J2EE 框架进行企业级应用系统开发事实证明是明智的选择。

4. C＃/.NET 开发模式

C＃语言与.NET 框架的出现彻底颠覆了 Windows 系统软件开发的传统模式，使程序员可以从繁杂冗长的编程细节中解脱出来，把更多的注意力投向满足用户的需求以及获得问题的真正解决方案。在需要编程时，由简洁明快的 C＃语言和强大的.NET 框架为实现解决方案提供有力的支持。

1.2.2 .NET 框架的组成

.NET 框架主要由 CLR 和.NET 类库这两部分组成。

1. CLR

CLR(Common Language Runtime,公共语言运行时)是.NET 程序的虚拟机平台,此处重点讲解它的 3 个特性:平台无关性、内存的自动管理和代码验证功能。

(1) 平台无关性

CLR 在整个.NET 平台中是个什么角色?说清楚这个问题前先看看图 1-1。

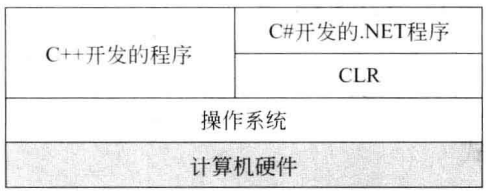

图 1-1 CLR 工作原理

从图 1-1 中可以看出,C#开发的.NET 程序是以 CLR 为运行平台的,这与 C++不同。

实际上,用 C++开发的程序会直接编译成本机的二进制代码,这种代码 CPU 可以直接识别,而用 C#开发的.NET 程序并不直接编译成本机代码,而是编译成一种称为 MSIL(Microsoft Intermediate Language)的中间语言,这种中间语言 CPU 无法直接识别。所以当运行用 C#开发的.NET 程序时,需要先由 CLR 将这种中间语言即时编译成 CPU 可直接识别的本机代码然后再运行。由此看来,CLR 就是.NET 程序运行的虚拟机平台。

.NET 程序为什么要采用这种中间语言加虚拟机运行的模式?这当然是有原因的,这其中获得的最大好处就是.NET 程序可以跨平台了。

试想一下,C++程序编译以后可以由 CPU 直接识别,而 CPU 是有多种架构的,这就说明 C++程序是依赖于计算机硬件的,这是其一;其二,C++程序直接同操作系统打交道,在某种操作系统下开发的 C++程序将无法保证它在其他操作系统下正常运行,说明 C++程序是依赖于操作系统的。有了这两个限制,C++程序是无法跨平台的(这里说的平台指计算机硬件平台和操作系统平台)。

现在看看用 C#语言开发的.NET 程序,由于它并不直接编译成本机代码,不需要 CPU 直接识别,所以它就没有了对 CPU 架构的依赖。又由于它不直接同操作系统打交道,而是同 CLR 打交道,这就没有了对操作系统的依赖。没有了这两个依赖,它就实现了平台的无关性。

.NET 程序虽然不依赖于 CPU 和操作系统,可是它依赖于 CLR。而 CLR 由微软公

司根据不同平台来提供不同版本，这使得. NET 程序员只需专心开发. NET 程序即可。

(2) 内存的自动管理

在讲 CLR 内存管理之前，先讲一下 C++ 是如何处理的。

C++ 程序员使用内存要向操作系统申请，获得内存并在使用完后还要写代码释放。谁申请谁释放，应该说这个过程顺理成章，但是大量的编程实践表明，很多 C++ 程序员在使用完内存后会忘记释放，结果导致操作系统无法回收已经不用的内存，造成内存的不必要紧张，这种现象叫“内存泄漏”，要避免只能靠 C++ 程序员不忘记释放。

上述问题在. NET 编程时就不会存在，因为. NET 程序需要内存时是向 CLR 申请，而 CLR 将它所掌控的内存划分成栈内存和堆内存。其中栈内存用后自动释放无须管理；堆内存在分配给. NET 程序后，也不需要. NET 程序员写代码释放，而是由 CLR 来管理。CLR 有一个单独的线程专门用来管理它分配出去的堆内存，当通过线程发现某块堆内存处于无主的废弃状态时，CLR 就会主动将其回收。这种机制有个专门术语叫“垃圾回收”，有了这种机制，. NET 程序员不用再担心出现“内存泄漏”了。

(3) 代码验证功能

由于. NET 程序首先被编译成中间代码，因此在运行前需要先由 CLR 即时编译成本机代码。事实上，CLR 在编译前有一个验证过程，该过程检查中间代码是否安全，也就是要确保中间代码不会访问不应该访问的内存。这一点 C++ 程序无法做到，因为 C++ 程序事先已经全部编译完成，它无法预料运行时内存的具体使用情况。

CLR 还有一些诸如跨语言的互操作等功能，随着学习的深入会逐步了解。

2. . NET 类库

. NET 类库同 C# 程序员的关系是最紧密的。CLR 是个平台，不理解它也不影响 C# 程序的运行，可. NET 类库不同，所有的. NET 程序都或多或少用到其中的类，并且. NET 程序员的编程工作就是基于. NET 类库展开的。所以，. NET 类库是. NET 程序员学习的重点。

. NET 类库以命名空间的形式组织类，此处仅介绍常用的几个命名空间。

(1) System. Data 命名空间

该命名空间提供了对 ADO. NET 组件类的访问，通过 ADO. NET 组件，. NET 程序可以访问并管理多个数据源的数据。

(2) System. Drawing 命名空间

该命名空间提供了对 GDI+ 基本图形功能的访问，通过 GDI+ 组件，. NET 程序可以开发一些图形输出的功能。

(3) System. IO 命名空间

该命名空间包含允许读写文件和数据流的类，以及提供基本文件和目录支持的类，有了这些类，. NET 程序就可以实现一些文件的 I/O 功能。

(4) System. Net 命名空间

该命名空间为当前网络上使用的多种协议提供了简单的编程接口，. NET 程序可以借此开发出使用 Internet 资源的应用程序，而不必考虑各种不同协议的具体细节。

(5) System. Web 命名空间

该命名空间提供了可以进行浏览器与服务器通信的类和接口。

(6) System. Web. UI 命名空间

该命名空间提供的类和接口可用于创建使用 ASP. NET 服务器控件的 ASP. NET 网页。

(7) System. Windows. Forms 命名空间

该命名空间包含用于创建基于 Windows 应用程序的类,可以充分利用 Windows 操作系统中提供的丰富的用户界面功能。

(8) System. Xml 命名空间

该命名空间为处理 XML 文件提供了基于标准的支持。

(9) System. Linq 命名空间

该命名空间提供支持使用语言集成查询(LINQ)进行查询的类和接口。

1.3 Visual Studio.NET 简介

微软提供的 Visual Studio. NET(VS. NET)集成开发平台无疑是业界最好用的开发平台之一,程序员通过它可以快捷高效地进行软件开发,事实上,很多程序员喜欢微软的技术就是因为喜欢 VS. NET。有过 Java 语言编程经历的人都知道,Java 语言和 J2EE 平台没有一个像 VS. NET 那样好用的集成开发平台,这不能不说是 Java 程序员的痛苦。初学者可能不了解 C# 语言、. NET 框架和 Visual Studio. NET 这三者之间的关系,下面简单地解释一下。

C# 语言是程序员手中的编程工具,. NET 框架为程序员提供了编程时要使用的各种功能各异的类库,而 VS. NET 集成开发平台则为程序员使用 C# 语言操作. NET 类库提供了方便,所以对于 C# 程序员来说,这三者往往是分不开的,微软也经常将三者的版本更新一同发布。下面通过表 1-1 来了解一下这些产品的版本变化历程。

表 1-1 C# 语言、. NET 框架及 VS. NET 开发工具的版本变化历程

正式版发布时间	VS. NET 版本	. NET 框架版本	C# 版本
2002-02-13	2002	1.0	1.0
2003-04-01	2003	1.1	1.1
2005-11-07	2005	2.0	2.0
2006-11-30	—	3.0*	—
2007-11-06	2008	3.5	3.0
2010-04-12	2010	4.0	4.0

* 注:. NET 框架 3.0 版随同 Windows Vista 操作系统一同发布。

第 2 章　C# 语言结构化程序设计

2.1　概　　述

C＃语言是一门面向对象的编程语言，而面向对象是一种编程思想，它指导人们如何更好地编写程序，它的实现离不开结构化程序设计，就好像一座设计精巧的大楼离不开砖头瓦块一样。

结构化程序设计最终要落实到语句上，而构建一个语句则需要数据类型、变量、操作符和表达式。其中，数据类型规定了内存的申请方式和合适的值范围；变量则代表了内存的位置；操作符用于构建表达式；而表达式又代表了程序的应用逻辑。语句自顶向下按照顺序编写，又可能会出现选择和循环。总之，语句的结构要体现解决问题的思路。

本章讲解 C＃语言的结构化程序设计部分，内容包括数据类型、变量、操作符、表达式、语句和数组，这些内容将为进一步学习 C＃语言的面向对象程序设计奠定良好的基础。

2.2　数 据 类 型

一个程序的执行需要 CPU 和内存，而程序员在写这个程序时对内存的申请主要就是通过数据类型声明变量来完成的。数据类型不仅指明了要申请多少字节的内存，而且规定了什么样的数据可以存放在该内存中。由此看来，要编写好 C＃程序就一定要掌握好 C＃语言的类型系统。

2.2.1　数据类型的种类

C＃语言有一个统一的类型系统，也就是说，所有的数据类型均来自一个称为 Object 的根类（注意，Object 类来自. NET 框架类库，而 object 是 Object 类的别名，它是 C＃语言的关键字，编程时二者通用）。由于对内存的使用方式不同，数据类型分成两类：值类型和引用类型。通过图 2-1 可以先初步了解 C＃语言中数据类型的概貌。

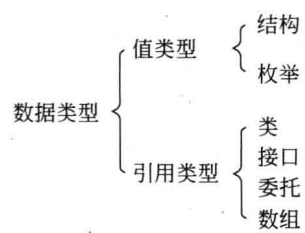

图 2-1　C＃语言中的数据类型

值类型和引用类型的区别主要是使用内存的方式不同。要知道，在 C＃语言中，内存有栈内存和堆内存之分，如果声明一个值类型的变量，那么系统将会在栈内存中分配空间来存储变量值；如果声明一个引用类型的变量，那