

特色教材·精选系列

FEATURE SELECTION
MATERIALS SERIES

C# 程序设计基础

杨棉绒 李栋 张鑫 主编

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

013031488

TP312C-43

801

普通高等学校计算机类专业特色教材·精选系列

C#程序设计基础

杨棉绒 李 栋 张 鑫 主 编

胡鹏飞 冯庆峰 副主编



TP312C-43
801

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE



北航

C1640008

884180810

内 容 简 介

本书详细介绍了 C#程序设计的环境、概念和基本方法。全书分为 10 章,由浅入深介绍了 C#基本语法与面向对象程序设计的知识,C#数据库程序设计方法和编程思想,利用 C#开发图形用户界面的编程方法以及文件和线程的基本概念。

本书最大的特点是,在不过分依靠集成开发环境的前提下,引导学生从本质上认识 C#语言与.NET 框架,比如本书在第 7 章之前的章节中基本不使用集成开发环境进行程序设计,就是为了加强读者对.NET 框架的 SDK 环境的认知和理解。

本书适合作为高等学校计算机及相关专业面向对象程序设计课程的教材,也可作为软件开发人员学习 C#程序设计的参考书。

图书在版编目(CIP)数据

C#程序设计基础 / 杨棉绒, 李栋, 张鑫主编. — 北京: 中国铁道出版社, 2013. 2

普通高等学校计算机类专业特色教材. 精选系列

ISBN 978-7-113-16004-3

I. ①C… II. ①杨… ②李… ③张… III. ①

C 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 (2013) 第 016989 号

书 名: C#程序设计基础

作 者: 杨棉绒 李栋 张鑫 主编

策 划: 巨 凤

读者热线: 400-668-0820

责任编辑: 姚文娟

封面设计: 一克米工作室

责任印制: 李 佳

出版发行: 中国铁道出版社 (100054, 北京市西城区右安门西街 8 号)

网 址: <http://www.51eds.com>

印 刷: 北京新魏印刷厂

版 次: 2013 年 2 月第 1 版 2013 年 2 月第 1 次印刷

开 本: 787mm×1092mm 1/16 印张: 18 字数: 423 千

书 号: ISBN 978-7-113-16004-3

定 价: 37.00 元

版权所有 侵权必究

凡购买铁道版图书, 如有印制质量问题, 请与本社教材图书营销部联系调换。电话: (010) 63550836

打击盗版举报电话: (010) 63549504

C#语言是.NET平台上的核心开发语言。随着.NET平台在软件开发领域取得成功,C#程序设计语言越来越受到开发者的青睐,成为广大程序员喜爱的开发工具,也成为众多软件公司竞相选用的开发环境之一。

目前许多高等院校都开设了“C#语言程序设计”这门课程,关于C#的教材,市面上已经有不少,其中有一些图书是编写人员根据相关书籍翻译或者编译而成,有些则是简单的功能罗列。而初学者接触C#语言,其实仅仅是想掌握一种实用编程工具,他们可能具有其他编程语言基础,也可能是零基础。如何学习C#语言,如何通过一本教材循序渐进地让读者了解和掌握C#及其支撑环境,继而达到能够实际编程的目的,本书做了较好的尝试。

编者根据多年实际编程经验和在高等院校任教的教学思路,制定本书的写作思想。具体实现为,不过分依靠集成开发环境,引导学生从本质上认识C#语言与.NET框架,尽量不使用或少用集成开发环境进行程序设计,使用最原始的记事本加SDK编译器的开发方式进行程序开发,从而加强读者对.NET框架的SDK环境的认知和理解。

为了方便读者学习,本书在讲解基本知识的同时,结合大量典型范例以加深读者对知识的理解与掌握,逐步培养读者分析问题、解决问题的能力。本书每章末附有适量的课后题,其中,编程题及上机实训题可以帮助读者加深对该章内容的理解,提高编程能力,同时可检验对知识的掌握程度。

本书对C#程序设计以及面向对象程序设计的基本思想进行了全面系统的介绍。全书共10章,第1章对C#和.NET框架进行了概述;第2章对C#基本语法和数据类型进行了简要介绍;第3章介绍了C#程序的面向对象程序设计;第4、5章对异常、委托、事件、集合等语法和编程规则进行了详细的分析说明,并通过实例介绍了相关的编程技巧;第6章详细介绍了ADO.NET和数据库编程的思想;第7、8章系统讲解了使用C#如何开发具有图形化用户界面的WinForm程序;第9、10章分别介绍了程序设计中常用的文件和线程的基础概念以提高学生对计算机体系结构的认知。

课时安排建议:非统考科目为18个教学周,除去假期和考试周,有效教学周一般为16周,每周4课时,共计64课时。以下是本书课程教学课时安排,部分内容可根据实际情况灵活调整。

学 习 内 容	课 时	课 时 分 配
第1章 C#概述	2	C#语言及其开发环境介绍,.NETFramework 的获取方法与安装配置:1课时 创建第一个C#应用程序并调试运行:1课时
第2章 C#基本语法	8	变量与常量、C#的基本数据类型:1课时 不同数据类型之间的转换、运算符与表达式:2课时 控制台的输入、输出与输出格式化:1课时 C#中的流程控制语句:2课时 数组:2课时

续表

学 习 内 容	课 时	课 时 分 配
第 3 章 在 C#中实现 OOP	10	面向对象的基本概念、C#中的类、对象：2 课时 构造函数和析构函数：2 课时 静态成员和实例成员、C#中的继承：2 课时 属性与索引器、抽象类、抽象方法与抽象属性：2 课时 接口、多态性的实现：2 课时
第 4 章 C#中的高级概念	8	异常：2 课时 委托、事件：2 课时 泛型、几种特殊类：2 课时 字符串和正则表达式：2 课时
第 5 章 C#中的集合	4	集合接口：1 课时 常用集合类型：3 课时
第 6 章 C#中的数据库操作基础知识	10	数据库基础、SQL 语言基础：4 课时 数据库访问过程、ADO.NET：4 课时 用 ADO.NET 编写一个小型图书管理系统：2 课时
第 7 章 窗体	4	Windows 下程序的运行机制：1 课时 SharpDevelop 集成开发环境：1 课时 窗体的属性、事件与方法：2 课时
第 8 章 C#的控件	8	Button 控件、标签控件、选择按钮、文本框控件：2 课时 图片框、图片列表、工具栏：2 课时 主菜单、列表框、组合框、定时器：2 课时 通用对话框：2 课时
第 9 章 文件操作	4	基本概念、文件和目录：2 课时 文件的读写：2 课时
第 10 章 线程	4	基本概念、线程处理、线程的同步：2 课时 线程池的处理、计时器：2 课时
总复习	2	
小计	64	

本书由杨棉绒、李栋、张鑫担任主编，胡鹏飞、冯庆峰担任副主编。其中杨棉绒编写第 1 章、第 2 章、第 5 章，李栋编写第 3 章、第 4 章，张鑫编写第 6 章、第 7 章、第 8 章、第 9 章，冯庆峰编写第 10 章，所有章节课后习题由胡鹏飞编写。全书由杨棉绒统稿、审核。同时还要感谢张东亮副教授在本书编写过程中给予的指导和帮助。

由于时间仓促，加之编者水平有限，书中的不当和疏漏之处在所难免，恳请读者批评指正。

编 者

2012 年 10 月

第 1 章 C#概述	1
1.1 C#与其集成开发环境	1
1.1.1 C#简介	1
1.1.2 Visual Studio.NET 简介	2
1.1.3 SharpDevelop	2
1.1.4 .NET 框架、集成开发环境、操作系统之间的关系	3
1.2 .NET Framework 的获取与安装	4
1.2.1 .NET Framework 的获取	4
1.2.2 .NET Framework 的安装与环境配置	5
1.3 创建 C#应用程序	6
1.4 程序集	10
1.5 命名空间	10
1.6 本章小结	11
1.7 课后习题	11
第 2 章 C#基本语法	12
2.1 变量与常量	12
2.1.1 变量	12
2.1.2 常量	14
2.2 C#的基本数据类型	14
2.2.1 C#数据类型的分类	15
2.2.2 整数类型	16
2.2.3 字符类型与字符串类型	17
2.2.4 实数类型与十进制类型	18
2.2.5 布尔类型	19
2.2.6 枚举类型	19
2.2.7 结构类型	20
2.3 不同数据类型之间的转换	22
2.3.1 隐式转换与显式转换	22
2.3.2 装箱和拆箱	23
2.4 运算符与表达式	24
2.4.1 运算符分类及使用说明	24
2.4.2 运算符的优先级	27
2.4.3 表达式	28



2.5 控制台的输入、输出与输出格式化.....	28
2.5.1 控制台的输入方法.....	28
2.5.2 控制台的输出方法.....	29
2.6 C#中的流程控制语句.....	32
2.6.1 选择语句.....	32
2.6.2 循环语句.....	34
2.6.3 转移语句.....	38
2.7 数组.....	40
2.8 本章小结.....	42
2.9 课后习题.....	42
第3章 在C#中实现 OOP.....	46
3.1 面向对象的基本概念.....	46
3.1.1 类和对象.....	46
3.1.2 封装、继承和多态.....	47
3.2 C#中的类.....	47
3.2.1 类的声明.....	48
3.2.2 类成员.....	48
3.2.3 类成员的访问控制.....	49
3.3 对象.....	49
3.4 构造函数和析构函数.....	53
3.4.1 构造函数.....	53
3.4.2 析构函数.....	56
3.5 静态成员和实例成员.....	58
3.6 方法.....	60
3.6.1 声明方法.....	61
3.6.2 方法调用.....	62
3.6.3 静态方法.....	71
3.7 C#中的继承.....	72
3.7.1 继承与 base 关键字.....	73
3.7.2 C#方法的重写.....	76
3.7.3 重载、重写和隐藏的比较.....	77
3.8 属性与索引器.....	79
3.8.1 属性.....	79
3.8.2 索引器.....	82
3.9 抽象类、抽象方法与抽象属性.....	85
3.9.1 抽象类.....	86
3.9.2 抽象方法.....	86
3.9.3 抽象属性.....	87



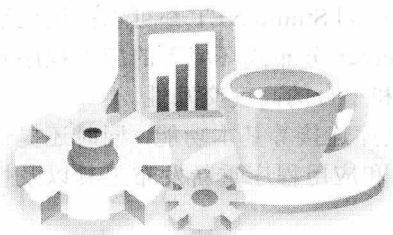
3.10 接口	88
3.10.1 接口的声明	89
3.10.2 接口的实现	89
3.10.3 抽象类与接口的区别	94
3.11 多态性的实现	96
3.11.1 类继承实现的多态性	96
3.11.2 接口实现的多态性	98
3.12 本章小结	100
3.13 课后习题	100
第 4 章 C#中的高级概念	105
4.1 异常	105
4.1.1 异常类	105
4.1.2 try 和 catch 块	107
4.1.3 使用 throw 引发异常	109
4.1.4 finally	111
4.2 委托	112
4.2.1 委托的定义	113
4.2.2 委托的实例化和使用	113
4.2.3 匿名方法	114
4.2.4 多路广播委托	115
4.2.5 多重委托	116
4.3 事件	117
4.3.1 定义事件	118
4.3.2 预定事件	119
4.3.3 引发事件	119
4.4 泛型	122
4.4.1 泛型方法	123
4.4.2 泛型类创建与使用	124
4.4.3 类型参数的约束	125
4.5 几种特殊类	126
4.5.1 分部类	126
4.5.2 密封类	127
4.5.3 内部类	128
4.5.4 静态类	130
4.6 字符串和正则表达式	132
4.6.1 System.String 类	132
4.6.2 System.StringBuilder 类	136
4.6.3 正则表达式	139



4.7 本章小结	146
4.8 课后习题	146
第 5 章 C#中的集合	150
5.1 集合接口	150
5.2 常用集合类型	155
5.2.1 System.Collections.ArrayList 类	155
5.2.2 System.Collections.Stack 类	158
5.2.3 System.Collections.Queue 类	159
5.2.4 System.Collections.SortedList 类	160
5.3 本章小结	162
5.4 课后习题	162
第 6 章 C#中的数据库操作基础知识	165
6.1 数据库基础	165
6.1.1 数据库的基本概念	165
6.1.2 数据库管理系统 (DBMS)	167
6.2 SQL 语言基础	168
6.2.1 SQL 语言简介	168
6.2.2 SELECT 语句的使用	168
6.2.3 INSERT 语句的使用	169
6.2.4 DELETE 语句的使用	169
6.2.5 UPDATE 语句的使用	169
6.3 数据库访问过程	170
6.4 ADO.NET	171
6.4.1 ADO.NET 概述	171
6.4.2 ADO.NET 访问数据库的两种方式	172
6.4.3 保持连接状态的数据操作	173
6.4.4 无连接状态的数据操作	178
6.5 用 ADO.NET 编写一个小型图书管理系统	182
6.5.1 需求分析	182
6.5.2 程序的实现	183
6.6 本章小结	186
6.7 课后习题	186
第 7 章 窗体	189
7.1 Windows 下程序的运行机制	189
7.1.1 系统控制台模式下的过程驱动	189
7.1.2 Windows 环境下程序的运行机制	190



7.1.3 C#中消息的封装	192
7.2 SharpDevelop 集成开发环境	194
7.2.1 SharpDevelop 界面简介	194
7.2.2 定制 SharpDevelop 界面	194
7.2.3 定制编码	196
7.2.4 创建 Windows Forms	197
7.3 窗体的属性、事件与方法	198
7.3.1 窗体的属性	198
7.3.2 窗体的事件	200
7.3.3 窗体的方法	202
7.4 本章小结	206
7.5 课后习题	206
第 8 章 C#的控制	208
8.1 Button 控件	208
8.1.1 集成开发环境中的创建及使用方法	209
8.1.2 使用记事本加 SDK 创建有控件窗体程序的方法	212
8.2 标签 (Label) 控件	216
8.3 链接标签 (LinkLabel) 控件	217
8.4 文本框 (TextBox) 控件	218
8.5 单选按钮 (RadioButton) 控件	220
8.6 复选框 (CheckBox) 控件	222
8.7 图片框 (PictureBox) 控件	225
8.8 图片列表 (ImageList) 控件	226
8.9 工具栏 (ToolBar)	226
8.10 主菜单 (MainMenu)	229
8.11 列表框 (ListBox)	231
8.12 组合框 (ComboBox)	233
8.13 定时器 (Timer)	235
8.14 通用对话框	236
8.14.1 通用对话框概述	236
8.14.2 打开文件对话框 (OpenFileDialog)	237
8.14.3 保存文件对话框 (SaveFileDialog)	239
8.14.4 字体对话框 (FontDialog)	241
8.14.5 颜色对话框 (ColorDialog)	242
8.15 本章小结	243
8.16 课后习题	243
第 9 章 文件操作	247
9.1 基本概念	247



第 1 章

C# 概述

C#是美国微软公司推出的一种面向对象的编程语言。它功能强大、编程简捷明快，是.NET Framework 开发平台的重要组成部分。所以，C#语言的应用飞速发展，使用十分广泛。

1.1 C#与其集成开发环境

1.1.1 C#简介

C#语言是从 C 语言和 C++语言演化而来的，是 Microsoft 专门为使用.NET 平台而创建的，并且结合了其他语言的许多优点。

开发相同功能的应用程序时，使用 C#开发比 C++要相对简单，这是因为 C#的语法较为简单，但 C#也是一种功能强大的语言，在 C++中能完成的任务在 C#中同样也能完成，只不过 C#的代码要比 C++略长一些，这是因为 C#是一种类型安全的语言（与 C++不同），虽然 C#的代码略长但其代码更健壮，调试也比较简单。

C#是唯一为.NET Framework 而设计的语言，是移植到其他操作系统上的.NET 版本中使用的主要语言，能使用.NET Framework 代码库提供的各种功能。如果要使用 VB.NET 等其他语言，由于其为了保持与之前老版本语言的类似性并且仍然需要遵循 CLR，所以这些语言就不能完全支持.NET 代码库的某些功能。

使用 C#可以创建下面列出的几种常见应用程序：

1. Windows 应用程序

如 Microsoft Office 应用程序，其外观与操作方式均与 Windows 类似，使用.NET Framework 的 Windows Form 模块的控件，可以建立 Windows 用户界面。

2. Web 应用程序

.Net Framework 包括一个动态生成 Web 内容的强大系统，即通常所说的 ASP.NET，可以使用 C#通过 Web Form 创建 ASP.NET 应用程序。

3. Web 服务

这是创建各种分布式应用程序的新方式，使用 Web 服务可以通过 Internet 虚拟交换数据，无论使用什么语言创建 Web，也无论 Web 服务驻留在什么系统上，都使用一样简单的语法。

1.1.2 Visual Studio.NET 简介

Visual Studio.NET 是一套完整的开发工具，是.NET 平台下最为强大的软件开发系统，无论是软件服务商，还是企业应用程序的部署与发布，Visual Studio.NET 都可以提供近乎完美的解决方案。.NET 开发平台可以简化在第三代 Internet 分布式环境下的应用程序开发，它基于开发互联网标准和协议之上，实现异构语言和平台的高度交互性。

Visual Studio.NET 提供了编码设计、编译调试、数据库连接等基本功能，同时还提供了基于开放架构的服务器组件开发平台、企业开发工具和应用程序重新发布工具以及性能测评报告等高级功能。其开发平台如图 1-1 所示。

其中用于软件开发的的部分就是 Visual Studio.NET，这部分的重要作用是提供了 4 种开发语言，并提供了第三方语言的接口支持，这就是图 1-1 中所示的最上层，也是用户经常使用的部分。Visual Studio.NET 是微软公司.NET 程序设计方案的重要组成部分，是微软公司提供的用户使用.NET Framework 的桥梁与纽带。

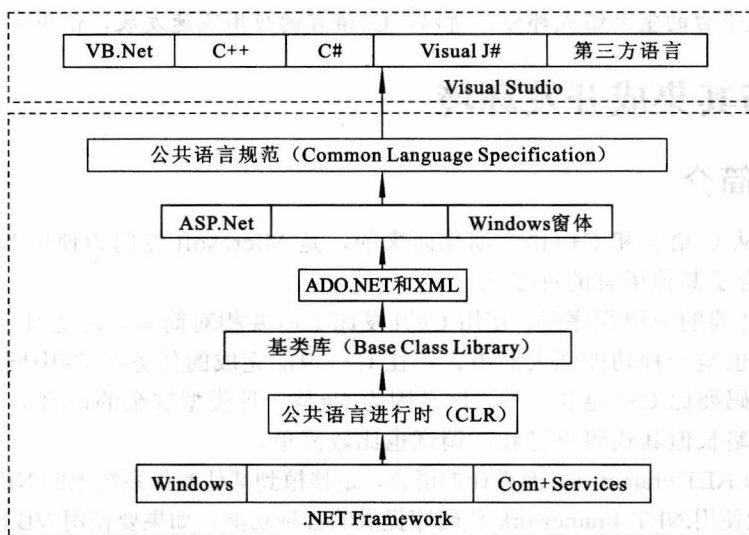


图 1-1 用 Visual Studio.Net 开发 C#时平台的架构图

从图 1-1 可以看出，Visual Studio.Net 提供了 4 种语言来操作.NET Framework，无论选择何种语言来创建、测试和部署应用程序，它的底层环境都是相同的。因此也可以把 Visual Studio.NET 理解为部署于.NET Framework 之上用来调用.NET Framework 框架中的各种类库进行.NET 程序设计的一个 IDE（集成开发环境）。但这里需要指出的是，Visual Studio.Net 虽然很方便也很强大，但是它并不是一个免费的集成开发环境。所以如果想使用 Visual Studio.Net 需要付出比较高昂的注册费用。

1.1.3 SharpDevelop

SharpDevelop 是一个为制作 C#或者 VB.NET 的项目而设计的免费 IDE。同时，这个 IDE 编辑器本身就是使用 C#开发的，而且公开了全部源代码。SharpDevelop 这个轻型的开发工具支持多种程序语言，包括 C#以及 VB.NET，同时还支持多种语言界面。



这个 IDE 编辑器的界面风格类似于 Office 以及 VS.NET, 这个编辑器可以支持书写 C#, ASP.NET, VB.NET, XML, HTML 等多种代码, 支持基于项目或者是文件的开发, 可以对 C#, HTML, ASP, ASP.NET, VBscript, VB.NET, XML 等提供彩色语法显示支持, 同时还可以把彩色的代码输出为 HTML 格式文件。支持丰富的代码模板以及外接插件。

因此对于初学者或者是不想为 IDE 交注册费的用户来说, SharpDevelop 也是一个不错的选择。用 SharpDevelop 开发 C#程序时平台的架构图如图 1-2 所示。

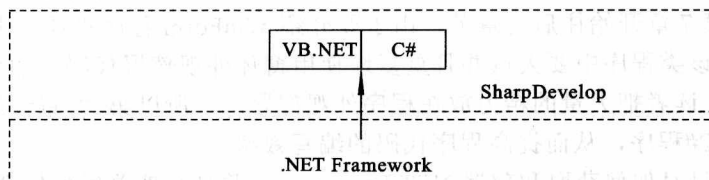


图 1-2 用 SharpDevelop 开发 C#程序时平台的架构图 (.NET Framework 结构省略)

1.1.4 .NET 框架、集成开发环境、操作系统之间的关系

从图 1-1 和图 1-2 我们可以看到无论是 Visual Studio.Net 还是 SharpDevelop, 它们都仅仅是一个部署在 .NET 框架 (.Net Framework) 之上用来方便用户使用 .NET 框架的桥梁与纽带。

而 .NET 框架则是一种采用系统虚拟机运行机制, 运行在操作系统层之上的编程平台, 以通用语言运行库 (Common Language Runtime) 为基础, 支持多种语言 (C#、VB、C++ 等) 的开发。 .NET 也为应用程序接口 (API) 提供了新功能和开发工具。这些革新使得程序设计员可以同时进行 Windows 应用软件和网络应用软件以及组件的开发。

因此, 综合来看, 集成开发环境、.NET 框架和操作系统三者之间的关系如图 1-3 所示。

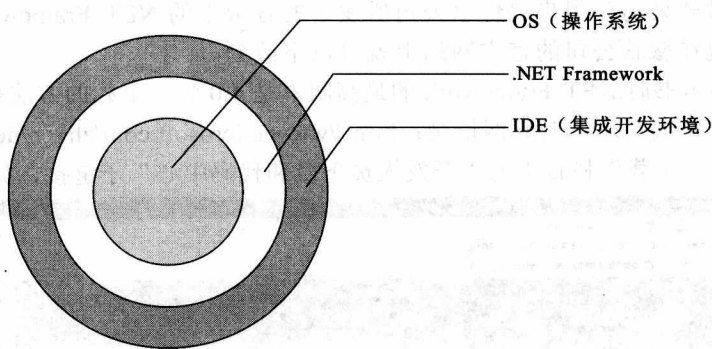


图 1-3 操作系统、.NET Framework、集成开发环境三者之间的关系

从图 1-3 可以看出, 如果想使用 C#编写程序或者是开发 C#程序, 除了要在计算机上安装操作系统之外还必须安装 .NET Framework, 因为 C#语言所编制的任何应用程序都需要 .NET Framework 的支持才可以正常运行。并且 C#程序的 SDK (Software Development Kit, 软件开发工具包) 也包含在 .NET Framework 中, 所以要想开发 C#程序也需要 .NET Framework 的支持才能够进行。

学习过 Java 的读者会感觉到, C#的这一特点和 Java 非常相似。同时要说明的是 C#也像 Java 程序一样只要安装了 SDK 不安装任何集成开发环境就可以开发 C#程序, 只是



这样就会对程序开发者的要求稍微高一点,需要更清晰地记忆 C#的基本指令;并且在编写 C#代码后,需要开发者手动使用.NET 的 SDK 对 C#源代码进行编译操作。如果在做工程时这样开发 C#程序的话,会使程序开发的时间变长。

但是,对于学习阶段来说,如果使用这样的方法来开发,能使学习者更深层地接触到.NET 的本质,更透彻地理解 C#的精髓。所以本书第 1~6 章中所有的例子都是采用记事本程序直接编写 C#源代码,然后手动调用.NET 的 SDK 对源代码进行编译形成可执行程序。从本书第 7 章开始往后的章节,由于要介绍 WinForm 程序设计(带 Windows 界面的程序设计),该类程序中要大量并且重复地使用窗体外观部署代码,如果还使用记事本去编写代码会使读者把大量的精力放在程序外观部署上,所以第 7 章详细介绍了如何使用 IDE 去开发 C#程序,从而提高程序代码的编写效率。

因此,本章只对如何获取和部署.NET Framework,并且在部署后如何调用.NET 的 SDK 对 C#源代码进行编译,从而得到可执行程序的过程进行详细的介绍。

1.2 .NET Framework 的获取与安装

从图 1-3 可以看出,想执行或者是开发 C#程序都必须有 .Net Framework 的支持,那么该怎么去获取 .Net Framework 并将其安装到自己的计算机上呢?本节就对这一问题进行详细介绍。

1.2.1 .NET Framework 的获取

由于 C#、VB.NET 等语言所设计的应用程序都需要 .NET Framework 的支持才能够正常运行,所以微软公司为用户提供该公司所发布的各版本的 .NET Framework 的免费下载包。这些下载包在微软公司的官方网站上就可以下载到。

在笔者编写本书时.NET Framework 的最高版本是 4.0 版,下载的方法是首先打开微软中国官方网站。在编写本书时其网址是: <http://www.microsoft.com/zh-cn/default.aspx>, 打开网页后选择其“下载”栏目下的“开发人员下载和代码中心”子链接,如图 1-4 所示。



图 1-4 微软中国官方网站



在该链接关联的子页面中就可以找到关于 .NET Framework 的下载链接，由于篇幅限制就不逐级在书中展示图片了。读者可以到该网站中自行查找下载，该过程并不复杂。

到目前为止常用的 .NET Framework 版本有 2.0 版、3.0 版、3.5 版和最新的 4.0 版本，书中所有实例都在以上各个版本中调试通过。

1.2.2 .NET Framework 的安装与环境配置

在微软网站中下载 .NET Framework 完整安装包后会得到一个用于安装 .NET 框架可执行程序（对于 .NET Framework 4.0 来说该程序的名字为 dotNetFx40_Full_x86_x64.exe），用鼠标双击执行该程序，会看到如图 1-5 所示安装向导界面。

选择“我已阅读并接受许可条款(A)”复选项，然后单击“安装”按钮就可以由该向导程序自动为计算机安装 .NET Framework 框架了。正确安装后会看到如图 1-6 所示的运行界面。并且安装 .NET Framework 后最好是按照安装向导的要求重启一次计算机。

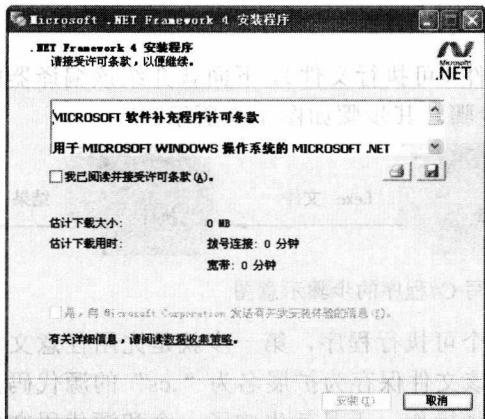


图 1-5 .NET Framework 4.0 安装向导

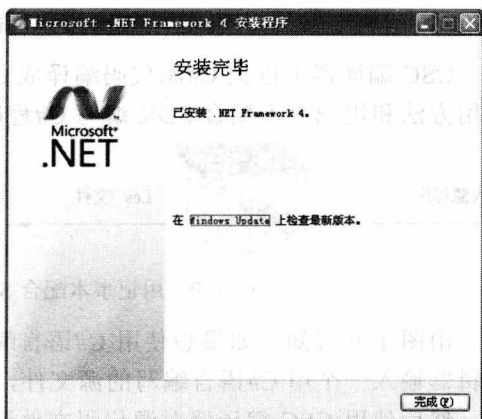


图 1-6 .NET Framework 4.0 安装完成

计算机重启后，打开计算机操作系统所在文件夹，由于笔者的操作系统是安装在计算机硬盘“C:\Windows”位置，所以对于笔者的计算机来说系统文件夹就是指“C:\Windows”文件夹。安装 .NET 框架后系统文件夹中会有一个“Microsoft.NET”子文件夹，在该文件夹中还有一个名叫“Framework”的二级文件夹，该文件夹中有计算机中已经安装的所有版本对应的 .NET Framework 子目录。笔者计算机中该目录的内容如图 1-7 所示。

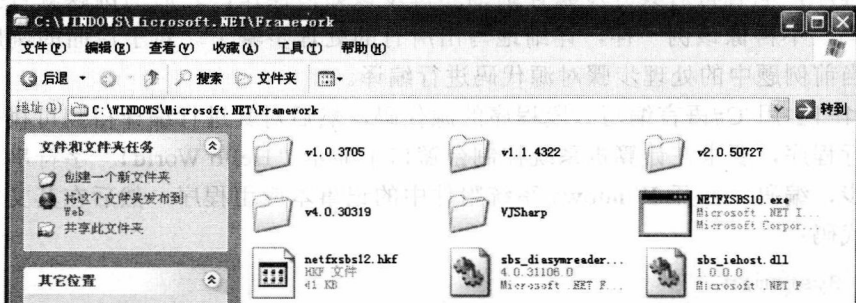


图 1-7 笔者计算机“Framework”文件夹中的内容



前面安装的.NET Framework 4.0 所对应的文件夹名称是“v4.0.30319”，当然这个名称也可能会随着.NET Framework 4.0 的子版本的不同而发生变化。在这个文件夹中能找到一个名为 `csc.exe` 的可执行程序（见图 1-8），该程序就是.NET Framework 4.0 中关于 C#语言的 SDK 编译器。

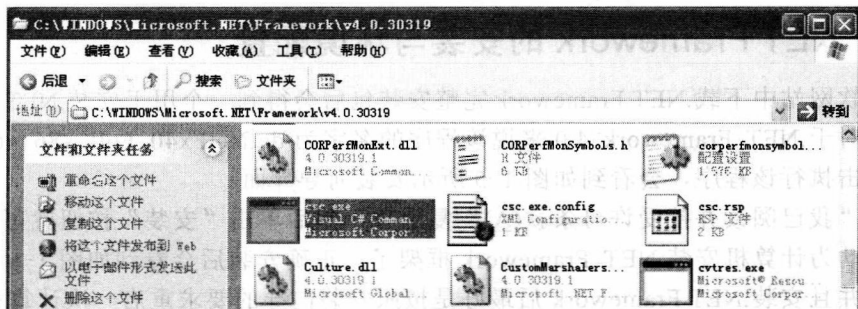


图 1-8 C#语言的 SDK 编译器

CSC 编译器可以把 C#源代码编译成.exe 文件（可执行文件），下面就介绍该编译器的使用方法和用记事本配合 SDK 编写 C#程序的步骤。其步骤如图 1-9 所示。

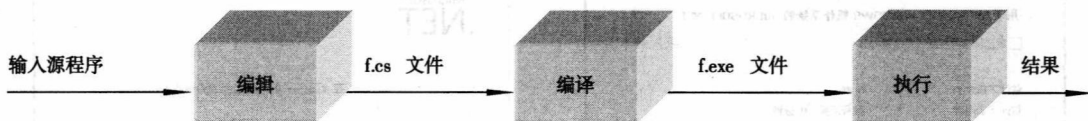


图 1-9 用记事本配合 SDK 编写 C#程序的步骤示意图

由图 1-9 可知，如果想使用 C#语言编写一个可执行程序，第一步就是先用任意文本编辑器输入一个用 C#语言编写的源文件，再将该文件保存为扩展名为“.cs”的源代码文件；然后使用 CSC 编译器对源代码文件进行编译操作，于是就生成了一个和源代码文件的主文件名相同，扩展名称为“.exe”的可执行程序；最后运行该可执行文件就能够得到所编程序的运行结果。

1.3 创建 C#应用程序

下面就通过编写一个简单的例题程序来演示图 1-9 所示的编译步骤。由于这是本书的第一个程序，笔者将对其分步骤详细描述每步该如何操作，之后的例题就直接给出程序的源代码，不再像该例一样，详细地写出所有的处理步骤了。对于后面的例题，读者可以仿照当前例题中的处理步骤对源代码进行编译。

【例 1-1】用 C#语言编写一段程序的源代码，然后对其进行编译得到可执行程序，运行可执行程序，要求在计算机系统控制台窗口中显示“Hello World!”字符串。

第一步，编码。打开 Windows 系统附件中的记事本应用程序，然后在其文本编辑区输入如下代码：

```
using System;
namespace My_first
{
```



```
class first_Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Hello World!");
        Console.ReadKey();
    }
}
```

第二步，保存。打开记事本工具中的“文件(F)”菜单，选择“另存为(A)”菜单项。这时记事本程序会打开一个标题为“另存为”的通用对话框，在该对话框中选择源代码文件的保存位置，这里选择的保存位置是 C:\ 盘根目录；然后在该对话框的“保存类型”下拉列表框中选择“所有文件”选项；接着在对话框的“文件名(N)”文本框中输入源代码文件的保存名称，并且该文件名的扩展名称必须是“.cs”，笔者保存该文件时所制定的文件名是“first.cs”；最后单击“保存(S)”按钮将该文件保存，这样就得到了一个 C#源代码文件，该源代码文件的全路径名称为“C:\first.cs”。本步骤操作结果如图 1-10 所示。

第三步，编译。由本章前面的内容可知，要想最终得到可执行程序必须对 C#源代码文件进行编译操作，而进行编译操作就必须明确自己的编译器是什么、放在哪里、该如何调用。而这三个疑问中的前两个已经在 1.2 节中给予了详细的说明，所以我们知道对于前面所假设的系统环境中，编译器放在 C:\WINDOWS\Microsoft.NET\Framework\v4.0.30319 目录中，文件名叫 csc.exe。明确这一概念后接下来介绍该如何调用 csc 编译器对源代码进行编译。

首先，单击 Windows 系统的“开始”菜单，执行“运行(R)”命令打开运行对话框，输入 cmd 命令，单击“确定”按钮，如图 1-11 所示。



图 1-10 记事本程序的“另存为”对话框

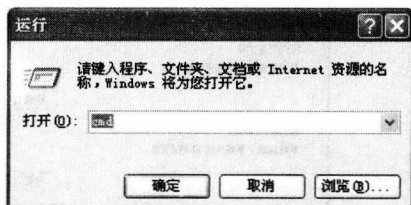


图 1-11 “运行”对话框

单击“确定”按钮后会打开 Windows 的系统控制台窗口，在控制台窗口中可以输入 DOS 命令来控制计算机执行各种操作。接着在控制台中输入如下两条命令就可以对源代