



高等学校计算机科学与技术教材

C# 程序设计教程

(第2版)

COMPUTER Science and Technology

□ 唐大仕 编著

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

C#程序设计教程

(第2版)

唐大仕 编著

清华大学出版社
北京交通大学出版社

· 北京 ·

内 容 简 介

本书详细介绍了C#程序设计的基本环境、概念、方法和应用。内容分为C#语言及C#应用两方面：C#语言方面，包括C#基础语法、面向对象的C#语言、C#语言高级特性等；C#应用方面，包括工具类、集合、常用算法、流式文件、文本应用、Windows窗体和控件、图形用户界面、多线程、ADO.NET数据库应用、网络通信编程等。书中也详细讲解了C#的一些新特性，如Lambda表达式、Linq、异步编程以及深入理解C#语言。

本书内容详尽、循序渐进，既介绍语法，又讲解语言机制，还注重C#的应用。本书提供了大量典型实例，并配有相关的电子资源、课件、视频。

本书内容和组织方式立足高等学校的教学教材，也可作为计算机技术的培训教材，还可作为Coursera、中国大学慕课上的“C#程序设计”课程配套教材。

本书封面贴有清华大学出版社防伪标签，无标签者不得销售。

版权所有，侵权必究。侵权举报电话：010-62782989 13501256678 13801310933

图书在版编目(CIP)数据

C#程序设计教程 / 唐大仕编著. —2版. —北京：北京交通大学出版社：清华大学出版社，2018.3

(高等学校计算机科学与技术教材)

ISBN 978-7-5121-3396-9

I. ①C… II. ①唐… III. ①C语言-程序设计-高等学校-教材 IV. ①TP312.8

中国版本图书馆CIP数据核字(2017)第265688号

C#程序设计教程

C# CHENGXU SHEJI JIAOCHENG

责任编辑：谭文芳

出版发行：清华大学出版社

邮编：100084

电话：010-62776969

<http://www.tup.com.cn>

北京交通大学出版社

邮编：100044

电话：010-51686414

<http://www.bjtup.com.cn>

印刷者：北京时代华都印刷有限公司

经 销：全国新华书店

开 本：185 mm × 260 mm 印张：36.75 字数：1168千字

版 次：2018年3月第2版 2018年3月第1次印刷

书 号：ISBN 978-7-5121-3396-9/TP·853

印 数：1~3000册 定价：69.00元

本书如有质量问题，请向北京交通大学出版社质监组反映。对您的意见和批评，我们表示欢迎和感谢。

投诉电话：010-51686043, 51686008；传真：010-62225406；E-mail: press@bjtu.edu.cn。

前言

作为一种优秀的面向对象语言，C#不仅具有封装、继承与多态等特性，而且还增加了索引器、委托、事件、Attribute、Linq 等创新性元素。在继承了 C++ 和 Java 等语言的优点的基础上，C#代表了程序设计语言演变的一个新阶段，这是与现代软件工程相适应的。

C#语言还利用 .NET Framework 作为其强大的平台，使得它在 Windows 图形用户界面、ASP. Net Web 应用，以及 ADO. NET 数据库等方面有广泛的应用，并且已经可以运行在 Windows、Linux、Mac OS 等平台上，甚至可以用于开发跨平台的手机应用。正因为这样，C#是目前主流的程序设计语言之一。

从学习的角度，C#语言的基本语法与传统的 C、C++、Java 语言有不少的相似性，学习者易于入门，而且使用功能强大的 Visual Studio 集成开发工具可以进行快速应用开发，因此将 C#作为程序设计的教学和开发语言不失为一种好的选择。

对于学习者而言，选择一本好的教材至关重要。笔者基于多年程序设计语言的教学经验，结合个人的软件开发实践，力图使本书突出以下特色：

1. 在详细介绍 C#语法的同时，还着重讲解 C#的语言机制，如类的封装与继承、类型转换、参数传递、虚方法调用、构造与析构、异常处理、迭代器等，让学习者知其然，并知其所以然。

2. 对于 C#的一些新特性，如 Lambda、Linq 及异步编程等，也进行了介绍，使学习者能简化代码的书写、提高开发效率。

3. 对 C#用到的基本类库和基本应用，包括集合、文件、文本界面、图形界面等进行详细讲解，精选大量典型而实用的例子，力图使学习者触类旁通，举一反三。

4. 对一些高级应用，如正则表达式、XML、网络信息获取、数据库访问、网络通信编程等内容也介绍其概念和编程方法，以利于学习者能做出具有实际应用价值的程序。

本书在内容安排上，大致可以划分为四个部分：第一部分介绍 C#语言基础，如数据、控制结构、数组、类、接口、结构等；第二部分介绍 C#高级特性，如事件、委托、泛型、Linq、运算符重载等；第三部分介绍 C#的基本类及其在 .NET 应用，如工具类、集合、常用算法、流式文件、文本应用、Windows 窗体和控件、图形用户界面；第四部分介绍 C#的高级应用，如多线程、正则表达式、XML、网络信息获取、数据库访问、网络通信编程等，还专门用一章来讲解深入理解 C#语言。

本书提供了配套的电子资源，包括源代码、课件、视频，一些应用实例由于源代码较

长，书中只列出了关键性代码，完整的源代码可以在配套的电子资源中找到。电子资源的下载地址是 <https://pan.baidu.com/s/1gfpNfyj>，提取密码是：x8cv。

本书内容和组织方式适合作为高等学校的教学教材，也可作为计算机技术的培训教材或自学用书。笔者在 Coursera、中国大学慕课等平台上开设了“C#程序设计”课程，本书也适合作为慕课的配套教材。在中国大学慕课上的 C#程序设计的地址是：<http://www.icourse163.org/course/PKU-1001663016>。

本书从第 1 版以来，得到全国不少师生及慕课学员的意见和建议，在此表示感谢。对于书中仍然存在的不足，也恳请读者批评指正。欢迎与笔者联系，电子邮件地址是：dstang2000@263.net。

下面是电子资源地址、慕课网址、笔者邮件地址的二维码。



唐大仕

于北京大学信息科学技术学院

2018 年 1 月

目 录

第1章 C#程序设计简介	1
1.1 C#语言及其环境	1
1.1.1 C#的产生与发展	1
1.1.2 C#的环境——Microsoft .NET	2
1.1.3 C#的特点	5
1.1.4 C#和C、C++ 的比较	6
1.1.5 C#与Java 的比较	8
1.2 简单的C#程序	8
1.2.1 HelloWorld	9
1.2.2 C#程序设计快速入门	12
1.2.3 对象的三个基本要素	16
1.2.4 C#程序的基本构成	17
1.3 程序中的输入输出及运算	20
1.3.1 控制台应用程序的输入输出	20
1.3.2 Windows 应用程序输入输出	22
1.3.3 常用的运算	23
1.4 开发工具	24
1.4.1 .NET Framework SDK 及 Visual Studio	24
1.4.2 使用命令行编译及运行程序	24
1.4.3 辅助工具 EditPlus	27
1.4.4 辅助工具 Visual Studio Code	29
1.5 应用程序类型	30
1.5.1 Visual Studio 建立不同类型的应用程序	30
1.5.2 WPF 应用程序	31
1.5.3 Web 应用程序	32
1.6 面向对象程序设计的基本概念	33
1.6.1 面向对象概述	33
1.6.2 对象、类与实体	34
1.6.3 封装、继承、多态	36
1.6.4 面向对象的软件开发过程	37
习题1	38
第2章 C#语言基础	40
2.1 数据类型、变量与常量	40
2.1.1 数据类型	40

2.1.2	标识符和关键字	42
2.1.3	字面常量	44
2.1.4	变量	45
2.1.5	C#编码惯例与注释	46
2.2	运算符与表达式	50
2.2.1	算术运算符	50
2.2.2	关系运算符	52
2.2.3	逻辑运算符	52
2.2.4	位运算符	53
2.2.5	赋值与强制类型转换	54
2.2.6	条件运算符	55
2.2.7	运算的优先级、结合性	55
2.3	流程控制语句	56
2.3.1	结构化程序设计的三种基本流程	56
2.3.2	简单语句	57
2.3.3	分支语句	57
2.3.4	循环语句	61
2.3.5	跳转语句	66
2.4	数组	68
2.4.1	数组的声明	68
2.4.2	数组的初始化	70
2.4.3	数组元素的使用	71
2.4.4	数组与 System.Array	72
2.4.5	使用 foreach 语句访问数组	73
2.4.6	数组应用举例	74
	习题 2	76
第 3 章	类、接口与结构	79
3.1	类、字段、方法	79
3.1.1	定义类中的字段和方法	79
3.1.2	构造方法与析构方法	80
3.1.3	对象的创建与使用	82
3.1.4	方法的重载	83
3.1.5	使用 this	85
3.2	属性、索引器	86
3.2.1	属性	86
3.2.2	索引器	91
3.3	类的继承	95
3.3.1	派生子类	95
3.3.2	字段的继承、添加与隐藏	96

3.3.3	方法的继承、添加与覆盖	97
3.3.4	使用 base	99
3.3.5	父类与子类的转换以及 as 运算符	100
3.3.6	属性、索引器的继承	102
3.4	修饰符	102
3.4.1	访问控制符	102
3.4.2	static	108
3.4.3	const 及 readonly	110
3.4.4	sealed 及 abstract	112
3.4.5	new、virtual、override	113
3.4.6	一个应用模型——单例	117
3.5	接口	118
3.5.1	接口的概念	118
3.5.2	定义接口	119
3.5.3	实现接口	120
3.5.4	对接口的引用	122
3.5.5	显式接口成员实现	123
3.6	结构、枚举	125
3.6.1	结构	125
3.6.2	枚举	127
	习题 3	129
第 4 章	C#高级特性	132
4.1	泛型	132
4.1.1	泛型的基本使用	132
4.1.2	自定义泛型	133
4.2	委托及 Lambda 表达式	137
4.2.1	委托类型与赋值	137
4.2.2	Lambda 表达式	145
4.2.3	使用系统定义的 Action 及 Func	146
4.3	事件	147
4.3.1	事件的应用	147
4.3.2	自定义事件	148
4.3.3	事件的语法细节	151
4.4	异常处理	152
4.4.1	异常的概念	152
4.4.2	捕获和处理异常	154
4.4.3	创建用户自定义异常类	158
4.4.4	重抛异常及异常链接	159
4.4.5	算术溢出与 checked	161

4.5 命名空间、嵌套类型、程序集	162
4.5.1 命名空间	163
4.5.2 嵌套类型	166
4.5.3 程序集	170
4.6 C#语言中的其他成分	174
4.6.1 运算符重载	175
4.6.2 使用 Attribute	175
4.6.3 编译预处理	176
4.6.4 unsafe 及指针	178
4.6.5 C#几个语法的小结	180
习题4	181
第5章 基础类及常用算法	183
5.1 C#语言基础类	183
5.1.1 .NET Framework 基础类库	183
5.1.2 Object 类	184
5.1.3 简单数据类型及转换	188
5.1.4 Math 类及 Random 类	190
5.1.5 DateTime 类及 TimeSpan 类	191
5.1.6 Console 类	194
5.2 字符串	194
5.2.1 String 类	194
5.2.2 StringBuilder 类	196
5.2.3 数据的格式化	198
5.3 集合类	199
5.3.1 集合的遍历	199
5.3.2 List、Stack 及 Queue 类	201
5.3.3 Dictionary 及 Hashtable 类	204
5.3.4 其他集合类	208
5.4 排序与查找	209
5.4.1 IComparable 接口和 IComparer 接口	209
5.4.2 使用 Array 类进行排序与查找	210
5.4.3 集合类中的排序与查找	212
5.4.4 自己编写排序程序	214
5.5 Linq	217
5.5.1 Linq 的基本用法	217
5.5.2 Linq 的查询方法	219
5.6 遍历、迭代、递归	222
5.6.1 遍历	222
5.6.2 迭代	223

5.6.3 递归	225
习题5	229
第6章 流、文件IO	233
6.1 流及二进制输入/输出	233
6.1.1 流	233
6.1.2 使用流进行二进制输入/输出	237
6.1.3 使用 File 的二进制功能	239
6.1.4 序列化及反序列化	240
6.2 文本输入/输出	242
6.2.1 使用 Reader 和 Writer 的文本 I/O	243
6.2.2 使用 File 的文本文件功能	247
6.2.3 标准输入/输出	249
6.2.4 应用示例: 背单词	249
6.3 文件、目录、注册表	251
6.3.1 文件与目录管理	251
6.3.2 监控文件和目录的改动	256
6.3.3 注册表	259
6.4 环境参数及事件日志	261
6.4.1 命令行参数	261
6.4.2 获得环境参数	263
6.4.3 使用事件日志	264
6.5 程序的调试、追踪与测试	267
6.5.1 程序的调试	267
6.5.2 程序的追踪	269
6.5.3 程序的单元测试	272
习题6	273
第7章 Windows 窗体及控件	275
7.1 Windows 窗体应用程序概述	275
7.1.1 Windows 图形用户界面	275
7.1.2 创建 Windows 窗体	276
7.1.3 添加控件	279
7.1.4 设定布局	282
7.1.5 事件处理	286
7.2 常用控件	290
7.2.1 Control 类	290
7.2.2 标签与按钮	292
7.2.3 文本框	296
7.2.4 列表框、UpDown 控件	300
7.2.5 滚动条、进度条	304

7.2.6	定时器、时间、日历类	306
7.2.7	图片框	308
7.2.8	其他几个控件	309
7.3	一些容器类控件	310
7.3.1	Panel 控件	310
7.3.2	ImageList 控件	311
7.3.3	TreeView 控件	312
7.3.4	ListView 控件	314
7.3.5	TabControl 控件	318
7.3.6	使用 Splitter 控件	318
7.4	窗体及对话框	319
7.4.1	Form 类	319
7.4.2	窗体的创建	321
7.4.3	使用 Form 作对话框	322
7.4.4	通用对话框	324
7.4.5	显示消息框	326
7.5	MDI 窗体、菜单、工具栏	327
7.5.1	MDI 窗体	327
7.5.2	菜单	328
7.5.3	使用主菜单及上下文菜单	329
7.5.4	工具栏	330
7.5.5	状态栏	331
7.5.6	一个综合的例子	332
	习题 7	336
第 8 章	绘图及图像	339
8.1	绘图基础支持类	339
8.1.1	位置及大小	339
8.1.2	颜色	342
8.1.3	画笔	344
8.1.4	刷子	347
8.2	绘图	350
8.2.1	Graphics 类	350
8.2.2	获得 Graphics 对象	352
8.2.3	进行绘图的一般步骤	353
8.2.4	坐标变换	355
8.2.5	处理重绘和无效操作	358
8.2.6	绘图示例	359
8.3	字体	366
8.3.1	Font 类	366

8.3.2	使用字体来绘制文本	367
8.4	图像	370
8.4.1	与图像相关的类	370
8.4.2	在窗体上显示图像	372
8.4.3	窗体、图片框上的图标及图像	374
8.4.4	图像处理	375
8.5	在自定义控件中使用绘图	381
8.5.1	自定义控件	381
8.5.2	在自定义控件中绘图	384
习题 8		387
第 9 章	文本、XML 及网络信息获取	388
9.1	文本及正则表达式	388
9.1.1	文本命名空间	388
9.1.2	正则表达式	389
9.1.3	应用示例: 播放歌词	394
9.2	XML 编程	398
9.2.1	XML 概念	399
9.2.2	XML 基本编程	401
9.2.3	Linq to XML	406
9.3	网络信息获取及编程	408
9.3.1	网络信息获取	408
9.3.2	WebRequest 及 WebClient	410
9.4	几类不同网络信息的处理	412
9.4.1	使用正则表达式处理网络文本	412
9.4.2	从网络上获取 XML 并进行处理	417
9.4.3	从网络上获取 Json 并进行处理	418
9.4.4	从网络上获取二进制信息并进行处理	421
习题 9		422
第 10 章	多线程及异步编程	424
10.1	线程基础	424
10.1.1	多线程的相关概念	424
10.1.2	线程的创建与控制	425
10.1.3	线程的同步	430
10.2	线程池与计时器	435
10.2.1	线程池	435
10.2.2	线程计时器	437
10.2.3	窗体计时器	438
10.3	集合与 Windows 程序中的线程	438
10.3.1	集合的线程安全性	438

10.3.2	窗体应用程序中的线程	440
10.4	并行编程	445
10.4.1	并行程序的相关概念	445
10.4.2	并行 Linq	451
10.5	异步编程	453
10.5.1	async 及 await	453
10.5.2	异步 I/O	455
10.5.3	其他实现异步的方法	458
习题 10		460
第 11 章	数据库、网络、多媒体编程	462
11.1	ADO. NET 数据库编程	462
11.1.1	ADO. NET 简介	462
11.1.2	数据集	465
11.1.3	连接到数据源	468
11.1.4	使用 DataAdapter 和 DataSet	469
11.1.5	使用 Command 和 DataReader	471
11.1.6	使用数据绑定控件	472
11.2	使用高级数据工具	475
11.2.1	使用 Visual Studio 的数据工具	475
11.2.2	使用 Entity Framework	475
11.2.3	使用 Linq 访问数据库	476
11.3	网络通信编程	477
11.3.1	使用 System. Net	478
11.3.2	TcpClient 及 TcpListener	478
11.3.3	E-mail 编程	484
11.4	互操作与多媒体编程	485
11.4.1	C#、VB. NET、JScript 的互操作	485
11.4.2	使用 Win32 API 进行声音播放	487
11.4.3	使用 COM 组件操作 Office 文档	488
11.4.4	使用 ActiveX 控件进行多媒体播放	490
习题 11		491
第 12 章	深入理解 C#语言	493
12.1	类型及转换	493
12.1.1	值类型及引用类型	493
12.1.2	值类型的转换	496
12.1.3	引用类型转换	497
12.1.4	装箱与拆箱	499
12.2	变量及其传递	502
12.2.1	字段与局部变量	502

12.2.2	按值传递的参数	503
12.2.3	ref 参数及 out 参数	505
12.2.4	params 参数	509
12.2.5	变量的返回	510
12.3	多态与虚方法调用	511
12.3.1	上溯造型	511
12.3.2	虚方法调用	512
12.4	类型与反射	516
12.4.1	typeof 及 GetType	516
12.4.2	is 运算符	519
12.4.3	反射及动态类型创建	520
12.5	对象构造与析构	521
12.5.1	调用本类或父类的构造方法	521
12.5.2	构造方法的执行过程	524
12.5.3	静态构造方法	526
12.5.4	析构方法与垃圾回收	528
12.5.5	显式资源管理与 IDisposable	529
12.6	运算符重载	531
12.6.1	运算符重载的概念	532
12.6.2	一元运算符	533
12.6.3	二元运算符	535
12.6.4	转换运算符	536
12.6.5	== 及 != 运算符	538
12.7	特性	539
12.7.1	使用系统定义的 Attribute	540
12.7.2	自定义 Attribute	542
12.8	枚举器与迭代器	546
12.8.1	枚举器	546
12.8.2	迭代器	549
习题 12		553
附录 A	C#语言各个版本的新特性	555
附录 B	C#语言相关网络资源	569
参考文献		571

第 1 章 C#程序设计简介

本章介绍 C#语言的特点、开发 C#程序的基本步骤、C#程序的构成、基本输入输出以及 C#的开发工具等。通过本章的学习，可以对 C#程序设计有一个初步的认识。

1.1 C#语言及其环境

C#（发音为“C Sharp”）是由 Microsoft 开发的面向对象的编程语言。它继承了 C 和 C++、Java 等语言的优点并且有了较大的发展，是迄今为止最为优秀、最为通用的程序设计语言之一。

1.1.1 C#的产生与发展

C#是直接从世界上最成功的计算机语言 C 和 C++ 继承而来的，又与 Java 紧密相关。理解 C#的产生与发展有助于 C#的学习。

1. 结构化编程与 C 语言

C 语言的产生标志着现代编程时代的开始。C 语言是 20 世纪 70 年代由 Dennis Ritchie 在基于 UNIX 操作系统上创建的。在一定意义上，20 世纪 60 年代的结构化编程造就了 C 语言。在结构化编程语言产生之前，大型的程序是很难编写的。因为往往在编写大型程序的时候，会由于存在大量的跳转、调用和返回而很难进行跟踪调试。结构化的编程语言加入了优化定义的控制语句，子程序中采用了局部变量和其他的改进，使得这种问题得到了解决。C 语言是结构化编程语言中最为成功的一种，C 语言至今仍是最常用的语言之一。

2. 面向对象编程与 C++ 语言

C 语言有它自身的局限性。20 世纪 70 年代末期，很多项目的代码长度都接近或者到达了结构化编程方法和 C 语言能够处理的极限。为了解决这个问题，出现了新的编程方法，即面向对象编程（object-oriented programming, OOP），程序员使用 OOP 可以编写出更大型的程序。1979 年初，Bjarne Stroustrup 在贝尔实验室创造了 C++。

C++ 是 C 语言的面向对象的版本。对于 C 程序员，可以方便地过渡到 C++，从而进行面向对象的编程。20 世纪 90 年代中期，C++ 成为广泛使用的编程语言。

3. 网络的发展与 Java 语言

随着网络的发展，编程语言进入到下一个主流，即 Java。Java 的创造工作于 1991 年在 Sun 公司开始，其主要发明者是 James Gosling。

Java 是一种面向对象的语言，它的语法和思想起源于 C++。Java 最重要的一个特点是具有编写跨平台、可移植代码的能力，Java 能够将一个程序的源代码转换到被称为字节码的中间语言，实现了程序的可移植性。该字节码在 Java 虚拟机上被执行。因此，Java 程序可移植到有 Java 虚拟机的任何环境中。由于 Java 虚拟机相对比较容易实现，所以适用于大部分的环境。

在 Java 中采用中间语言是很重要的，在其后的 C#中采用了类似的方案。

4. C#语言的产生

Microsoft 公司在 20 世纪 90 年代末开发了 C#，其首席设计师是 Anders Heilsberg。

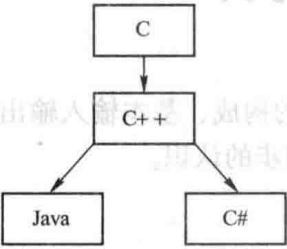


图 1-1 C#的产生历史

图 1-1 展示了 C#的家族史。C#的祖父是 C，C#继承了 C 的语法、关键词和运算符。C#建立在 C++ 定义的对象模型基础上，并加以改进。

C#起源于 C 和 C++，并且与 Java 有许多相似之处，同时 C#包含了许多创新的特性，这些特性将会在本书中进行详细的讲解。

5. C#语言的发展

C#语言从 2002 年正式发布以来，经历了多个版本，主要的版本如表 1-1 所示。C#的发展是与 C#的运行环境（Microsoft. NET）以及集成开发工具（Visual Studio）同步发展的。

表 1-1 C#语言的各个版本

年 月	C#版本	. NET 版本	Visual Studio 版本
2002 年 1 月	1.0	1.0.3705.0	Visual Studio. NET 2002
2003 年 4 月	1.1, 1.2	1.1.4322.573	Visual Studio. NET 2003
2005 年 11 月	2.0	2.0.50727.42	Visual Studio 2005
2006 年 11 月	3.0	3.0.4506.30	Visual Studio 2005
2007 年 11 月	3.0	3.5.21022.8	Visual Studio 2008
2010 年 4 月	4.0	4.0.330319	Visual Studio 2010
2012 年 9 月	5.0	4.5	Visual Studio 2012
2013 年 11 月	5.0	4.5.1	Visual Studio 2013
2014 年 11 月	6.0	4.6	Visual Studio 2015
2017 年 3 月	7.0	4.7	Visual Studio 2017

C#语言仍在迅速发展之中，除了主要用在 Windows 平台外，现在还可以用于 Linux、Mac OS 平台。

对于学习者而言，C#可以很快被 C 和 C++、Java 程序员所熟悉，而且 C#避免了其他语言中一些容易出错、难以使用的成分，并且可以利用集成开发环境 Visual Studio 大大提高程序开发的效率。可见，学习 C#语言不仅必要，而且是可以学得好的。

1.1.2 C#的环境——Microsoft. NET

尽管 C#是一种可以单独学习的计算机编程语言，但是它与其运行期环境——Microsoft. NET 框架仍然有密切联系。究其原因有二：其一，微软最初设计 C#语言是为了编写 .NET 框架；其二，C#使用的函数库是 .NET 框架定义的函数库中的一部分。因此，虽然可以把 C#与 .NET 环境分开，但是它们实际上是密切联系的。由于以上原因，对 .NET 框架有一个基本了解是非常必要的。事实上，本书不仅介绍 C#语言本身，还介绍 C#语言在 .NET 环境中的应用。

1. 什么是 Microsoft.NET

Microsoft.NET 是一个综合性的术语，它描述了微软公司发布的许多技术。总的来说，Microsoft.NET 包括以下技术领域：

- ① .NET 框架 (.NET Framework)；
- ② .NET 语言，包括 C#，F#，C++，Visual Basic 等；
- ③ 开发工具，主要是 Visual Studio。

这几个部分之间的关系如图 1-2 所示。

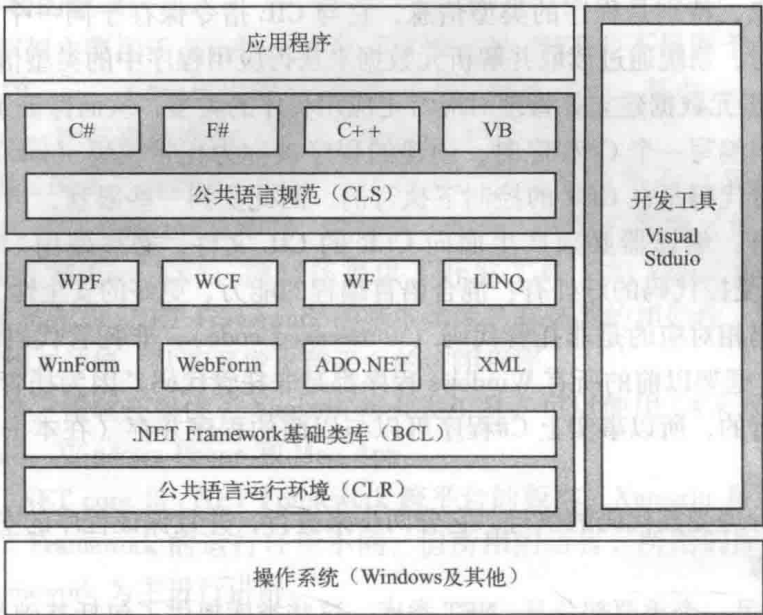


图 1-2 Microsoft.NET 体系

其中，.NET 框架位于操作系统之上，它定义了一种支持开发和执行应用程序的环境。

.NET 框架定义包含了两个重要的部分：一个是公共语言运行环境（Common Language Runtime, CLR），它管理程序的执行；另一个是 .NET 类库，使得我们在进行 C# 编程时可以调用这些库。

2. 公共语言运行环境

公共语言运行环境（CLR）管理 .NET 代码的执行，可以认为它是一个执行 C# 程序的虚拟运行环境（虚拟机）。CLR 负责为应用程序提供内存分配、线程管理、安全以及垃圾回收等服务，负责对代码进行严格的类型安全检查，以保证代码安全、正确地运行。

C# 源程序首先编译成中间代码，在执行时，再即时转换成机器指令。如图 1-3 所示。

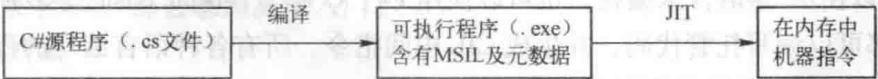


图 1-3 C# 程序的编译与执行过程

当编译 C# 程序时，编译器并不输出 CPU 指令，而是一种特殊的中间代码。这些中间代码被称为通用中间语言（common intermediate language, CIL）（又称为 microsoft intermediate language, MSIL 或 IL）。CIL 定义了一系列与 CPU 类型无关的可移植指令集，可以认为是一