

TURING

图灵程序设计丛书 微软技术系列

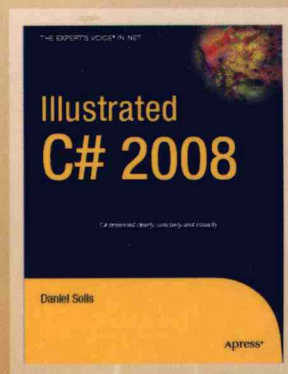
Apress®

Illustrated C# 2008

C#图解教程

[美] Daniel Solis 著
苏林 朱晔 等译

- 用图说话，最易学的C#教程
- Amazon全五星盛誉
- 涵盖Visual C# 2008和.NET 3.5最新特性



人民邮电出版社
POSTS & TELECOM PRESS

TURING

图灵程序设计丛书

微软技术系列

Illustrated C# 2008

C#图解教程

[美] Daniel Solis 著

苏林 朱晔 等译

人民邮电出版社
北京

图书在版编目 (CIP) 数据

C# 图解教程 / (美) 索利斯 (Solis, D.) 著; 苏林等译.
—北京: 人民邮电出版社, 2009.1
(图灵程序设计丛书)
ISBN 978-7-115-18773-4

I. C… II. ①索…②苏… III. C 语言-程序设计-教材 IV.
TP312

中国版本图书馆CIP数据核字 (2008) 第136010号

内 容 提 要

本书是一本广受赞誉的 C# 教程。它以图文并茂的形式, 用朴实简洁的文字, 并辅之以大量表格和代码示例, 精炼而全面地阐述了最新版 C# 语言的各种特性, 使读者能够快速理解、学习和使用 C#。同时, 本书还讲解了 C# 与 VB、C++ 等主流语言的不同点和相似之处。

本书是一本经典的 C# 入门书, 不仅适合没有任何编程语言基础的初级读者, 而且还是 VB、C++ 等语言基础的 C# 初学者的最佳选择。

图灵程序设计丛书

C# 图解教程

- ◆ 著 [美] Daniel Solis
- 译 苏 林 朱 晔 等
- 责任编辑 傅志红
- 执行编辑 杨福川
- ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
三河市海波印务有限公司印刷
- ◆ 开本: 800×1000 1/16
印张: 29
字数: 685千字 2009年1月第1版
印数: 1-4 000册 2009年1月河北第1次印刷

著作权合同登记号 图字: 01-2008-3294号

ISBN 978-7-115-18773-4/TP

定价: 65.00 元

读者服务热线: (010)88593802 印装质量热线: (010)67129223

反盗版热线: (010)67171154

版 权 声 明

Original English language edition, entitled *Illustrated C# 2008* by Daniel Solis published by Apress, 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705.

Copyright © 2008 by Daniel Solis. Simplified Chinese-language edition copyright © 2008 by Posts & Telecom Press. All rights reserved.

本书中文简体字版由 Apress L. P. 授权人民邮电出版社独家出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

版权所有，侵权必究。

谨将此书献给我的父母——Salt和Amy，并献给Sian和Sue。

译者序（一）

书是知识的载体，是智慧的传播者。技术图书在技术的普及、发展过程中的作用是毋庸置疑的。在这个知识爆炸、信息技术迅猛发展的时代，技术图书的作用更加突出。我们比以往任何时候都需要关于新技术和新平台的参考资料。一本描述清晰、内容详细的书能使我们快速掌握这些技术。

译者不才，自己无力写出这样的书，愿意以虫蚁之能，行搬运之事，将优秀外文书籍译成中文，以利于国人参考和学习，从而为技术传播尽自己的绵薄之力。

C#和.NET平台近年来迅速普及，已经成为很多公司使用的主要技术之一。有很多出色的应用都是使用C#开发的，包括很多Web 2.0时代的网络应用。虽然.NET平台目前还只能在Windows操作系统下工作，但是这并没有妨碍它发展壮大。一方面是因为Windows操作系统的普及程度已经给.NET提供了巨大的发展空间；另一方面是因为.NET确实是个优秀的平台，而且C#也确实算得上是新一代的优秀的面向对象编程语言。作为一个与时俱进的软件工程师，忽视C#和.NET是很不明智的。

本书是一部极为出色的C#著作。正如本书作者所说，它不仅包含了入门的基础知识，而且同时还能作为开发过程中的参考书使用。书中使用了大量的示例和图表，使内容一目了然。即便是有经验的C#程序员，阅读这本书也会受益匪浅。

在本书的翻译过程中，我尽量保持原书清晰明了的风格，并努力保证术语及用词的准确。由于能力有限，我虽已尽所能，但仍难免有不妥之处，望读者朋友海涵。

感谢我的妻子毛毛！在我翻译本书的过程中，她承担了大部分的家务，并给予了我很多支持和鼓励。没有她的爱和付出，本书的翻译工作肯定不会进展得如此顺利。

相信这本书一定对你有帮助！

苏 林

2008年5月于上海

译者序（二）

.NET经过近8年的发展后已经变得非常庞大，也非常成熟了，而且发展的速度越来越快。

学好C#是在.NET平台上构建一切应用的前提，因此，我强烈建议欲涉足.NET的初学者无论如何都应该学习C#，并且要学好。

说点题外话，很多人没有将.NET框架、CLR和C#语言这三者之间的关系区分清楚，认为其版本号是一一对应的。其实，.NET框架是一个独立发布的软件包，包括了CLR、类库以及相关的语言编译器等工具。C#代码经过编译之后在CLR环境中运行。由于.NET 3.0/3.5其实是.NET 2.0的扩展（只是增加了一些新的程序集），所以.NET 3.0/3.5的CLR版本还是2.0。而且，.NET 3.0其实只扩展了WF、WPF、WCF、WCS等组件，并没有提供新的C#编译器，直到.NET 3.5中才打包了C#3.0的编译器。所以，.NET框架、CLR和C#的版本之间的对应关系如下表所示：

.NET版本	1.0	1.1	2.0	3.0	3.5
CLR版本	1.0	1.1	2.0	2.0	2.0
C#版本	1.0	1.1	2.0	2.0	3.0

也就是说，对于那些不涉及新程序集的C# 3.0新特性（比如自动属性、匿名类型等）在.NET 2.0的环境中也可以运行，CLR对这些特性是一无所知的。

言归正传，拿到本书的英文版后，我粗略地看了一下目录，认为此书是一本彻底面向初学者的基础书籍。在翻译了几章之后，才发现先前的认识不完全正确。此书和一般的面向初学者的C#书籍不太一样：

- 可能和作者的C++背景有关，作者喜欢从底层（比如内存布局）的角度来剖析一些知识点。这有助于读者在知其然的同时还能知其所以然，从而打下扎实的基础。
- 书如其名，本书的特点就是有大量示意性的表格和插图，简洁明了，非常易于读者对知识点的理解。书中还有大量的范例代码，代码中也添加了很多注解，可以帮助读者理解代码的要点。
- 另外，本书绝对不是老版的旧瓶装新酒。C# 3.0的所有新特性都完全地融合在其中，而不是在老版本基础上加一些关于新特性的章节。

因此，如果你确实已经使用C#构建了很多应用或已经对C# 2.0有所掌握，那么本书或许会对你非常有用，很多关于CLR本质的内容将能帮助你更深入地理解C#。

由于时间关系，译者在翻译的过程中难免有疏漏。本书的第1~13章由苏林先生翻译，第14~25章由我翻译。欢迎对C#或.NET感兴趣的朋友与我交流，我的邮箱是yzzhu@live.com，个人BLOG是 <http://lovecherry.cnblogs.com>。

最后，预祝你在阅读本书之后能有所收获，编程快乐。

朱 晔

2008年5月于上海

前 言

本书的目的是讲授C#编程语言的基础知识和工作原理。大多数图书主要使用文字讲授编程。文字对于小说来说足够了，但对于编程语言中的很多重要概念，综合运用文字、图形和表格会更容易理解。

我们中许多人都习惯于形象思维，而图形和表格有助于我们更清晰地理解概念。在几年的编程语言教学过程中，我发现是我在白板上画的图帮助学生最快地理解了我要传达的概念。

然而，单是图表并不足以解释一种编程语言和平台。本书的目标是找到文字和图表的最佳结合，以使你对这种语言有透彻的理解，并且也让本书能当作参考工具使用。

本书写给所有想要学习C#的人——从初学者到有经验的程序员。刚开始学编程的人会发现，书中全面讲述了基础知识；有经验的程序员会觉得，内容的叙述非常简洁，无需苦苦寻觅就能直接获得想要的信息。对于这两类程序员，内容本身都用图形化方式呈现，这种方式使这种语言更容易学习。

请享受本书吧！

致谢

我想感谢Sian每天支持并鼓励我，我还想感谢我的父母、兄弟和姐妹，他们一直爱我并支持我。

我还想对Apress的朋友表达诚挚的感谢，他们与我一起工作并完成这本书。我真心感激他们理解并赏识我努力做的事情，并和我一起完成它。感谢你们所有人。

目 录

第 1 章 C#和.NET 框架.....1	2.7.3 注释类型总结..... 18
1.1 在.NET 之前..... 1	第 3 章 类型、存储和变量..... 19
1.1.1 20 世纪 90 年代后期的 Windows 编程..... 1	3.1 C#程序是一组类型声明..... 19
1.1.2 下一代平台的目标..... 2	3.2 类型是一种模板..... 20
1.2 进入 Microsoft .NET..... 2	3.3 实例化类型..... 20
1.2.1 .NET 框架的组成..... 2	3.4 数据成员和函数成员..... 21
1.2.2 大大改进的编程环境..... 3	3.5 预定义类型..... 21
1.3 编译成 CIL..... 4	3.6 用户定义类型..... 23
1.4 编译成本机代码并执行..... 5	3.7 栈和堆..... 24
1.5 CLR..... 6	3.7.1 栈..... 24
1.6 CLI..... 7	3.7.2 堆..... 24
1.7 缩写回顾..... 8	3.8 值类型和引用类型..... 25
第 2 章 C#编程概述.....9	3.8.1 存储引用类型对象的成员..... 26
2.1 一个简单的 C#程序..... 9	3.8.2 C#类型的分类..... 27
2.2 标识符和关键字..... 10	3.9 变量..... 27
2.2.1 命名约定..... 11	3.9.1 变量声明..... 27
2.2.2 关键字..... 12	3.9.2 多重变量声明..... 29
2.3 Main: 程序的起始点..... 12	3.9.3 使用变量的值..... 29
2.4 空白..... 13	第 4 章 类: 基础..... 30
2.5 语句..... 13	4.1 类的概述..... 30
2.5.1 简单语句..... 13	4.2 程序和类: 一个快速的示例..... 31
2.5.2 块..... 13	4.3 声明类..... 31
2.6 从程序中输出文本..... 14	4.4 类成员..... 32
2.6.1 Write..... 14	4.4.1 字段..... 32
2.6.2 WriteLine..... 15	4.4.2 显式和隐式字段初始化..... 33
2.6.3 格式字符串..... 15	4.4.3 声明多个字段..... 34
2.6.4 多重标记和值..... 16	4.4.4 方法..... 34
2.7 注释..... 16	4.5 创建变量和类的实例..... 35
2.7.1 关于注释的补充..... 17	4.6 为数据分配内存..... 35
2.7.2 文档注释..... 17	4.7 实例成员..... 36

4.8 访问修饰符	37	6.8.2 属性示例	78
4.9 从类的内部访问成员	39	6.8.3 使用属性	79
4.10 从类的外部访问成员	40	6.8.4 属性和关联字段	80
4.11 综合应用	41	6.8.5 执行其他计算	81
第5章 方法	43	6.8.6 只读和只写属性	82
5.1 方法的结构	43	6.8.7 计算只读属性示例	82
5.2 本地变量	45	6.8.8 属性和数据库示例	83
5.2.1 类型推断和 var 关键字	45	6.8.9 自动实现属性	83
5.2.2 嵌套块中的本地变量	46	6.8.10 静态属性	84
5.3 本地常量	47	6.9 实例构造函数	85
5.4 方法调用	49	6.9.1 带参数的构造函数	86
5.5 返回值	50	6.9.2 默认构造函数	87
5.6 参数	52	6.10 静态构造函数	88
5.6.1 形参	53	6.10.1 静态构造函数示例	88
5.6.2 实参	53	6.10.2 构造函数的可访问性	89
5.6.3 带输入参数的方法示例	54	6.11 对象初始化列表	89
5.7 值参数	54	6.12 析构函数	90
5.8 引用参数	56	6.12.1 调用析构函数	91
5.9 输出参数	58	6.12.2 标准清理模式	93
5.10 参数数组	60	6.13 比较构造函数和析构函数	94
5.10.1 方法调用	61	6.14 readonly 修饰符	94
5.10.2 数组作实参	63	6.15 this 关键字	95
5.11 参数类型总结	64	6.16 索引	96
5.12 栈帧	64	6.16.1 什么是索引	97
5.13 递归	65	6.16.2 索引和属性	98
5.14 方法重载	66	6.16.3 声明索引	98
第6章 类进阶	69	6.16.4 set 访问器	99
6.1 类成员	69	6.16.5 get 访问器	100
6.2 实例类成员	71	6.16.6 关于索引的补充	100
6.3 静态字段	71	6.16.7 为 Employee 示例声明索引	100
6.4 从类的外部访问静态成员	72	6.16.8 另一个索引示例	101
6.4.1 静态字段示例	73	6.16.9 索引重载	102
6.4.2 静态成员的生存期	73	6.17 访问器的访问修饰符	103
6.5 静态函数成员	74	6.18 分部类和分部类型	104
6.6 其他静态类成员类型	75	第7章 类和继承	107
6.7 成员常量	75	7.1 类继承	107
6.8 属性	77	7.2 访问继承的成员	108
6.8.1 属性声明和访问器	78	7.3 隐藏基类的成员	110
		7.4 基类访问	111

7.5 使用基类的引用	112	8.12 条件运算符	151
7.5.1 虚方法和覆写方法	114	8.13 一元算术运算符	152
7.5.2 覆写标记为 override 的方法	115	8.14 用户定义类型转换	153
7.6 构造函数的执行	118	8.15 运算符重载	155
7.6.1 构造函数初始化语句	119	8.15.1 运算符重载的限制	156
7.6.2 类访问修饰符	120	8.15.2 运算符重载的示例	157
7.7 程序集间的继承	121	8.16 typeof 运算符	158
7.8 成员访问修饰符	123	第9章 语句	161
7.8.1 访问成员的区域	123	9.1 什么是语句	161
7.8.2 公有成员的可访问性	124	9.2 表达式语句	162
7.8.3 私有成员的可访问性	125	9.3 控制流语句	163
7.8.4 受保护成员的可访问性	125	9.4 if 语句	163
7.8.5 内部成员的可访问性	126	9.5 if..else 语句	164
7.8.6 受保护内部成员的可访问性	126	9.6 switch 语句	165
7.8.7 成员访问修饰符的总结	126	9.6.1 分支示例	166
7.9 抽象成员	127	9.6.2 switch 语句的补充	167
7.10 抽象类	128	9.6.3 分支标签	168
7.11 密封类	129	9.7 while 循环	169
7.12 静态类	130	9.8 do 循环	169
7.13 扩展方法	130	9.9 for 循环	170
7.14 外部方法	133	9.9.1 for 语句中变量的有效范围	172
第8章 表达式和运算符	135	9.9.2 初始化语句和迭代表达式中 的多表达式	172
8.1 表达式	135	9.10 跳转语句	172
8.2 字面量	136	9.10.1 break 语句	173
8.2.1 整数字面量	137	9.10.2 continue 语句	173
8.2.2 实数字面量	137	9.11 标签语句	174
8.2.3 字符字面量	138	9.11.1 标签	174
8.2.4 字符串字面量	139	9.11.2 标签语句的范围	175
8.3 求值顺序	140	9.12 goto 语句	175
8.3.1 优先级	141	9.13 using 语句	176
8.3.2 结合性	141	9.13.1 资源的包装使用	177
8.4 简单算术运算符	142	9.13.2 using 语句的示例	178
8.5 求余运算符	142	9.13.3 多个资源和嵌套	179
8.6 关系比较运算符和相等比较运算符	143	9.13.4 using 语句的另一种形式	180
8.7 递增运算符和递减运算符	145	9.14 其他语句	180
8.8 条件逻辑运算符	146	第10章 命名空间和程序集	181
8.9 逻辑运算符	147	10.1 引用其他程序集	181
8.10 移位运算符	148		
8.11 赋值运算符	150		

10.2	命名空间	184	12.4.3	构造函数和析构函数的总结	214
10.2.1	命名空间名称	186	12.5	字段初始化是不允许的	215
10.2.2	命名空间的补充	187	12.6	结构是密封的	215
10.2.3	命名空间跨文件伸展	188	12.7	装箱和取消装箱	215
10.2.4	嵌套命名空间	188	12.8	结构作为返回类型和参数	215
10.3	using 指令	189	12.9	关于结构的附加信息	215
10.3.1	using 命名空间指令	189	第 13 章 枚举		217
10.3.2	using 别名指令	190	13.1	枚举	217
10.4	程序集的结构	191	13.1.1	设置底层类型和显式值	218
10.5	程序集标识符	192	13.1.2	隐式成员编号	219
10.6	强命名程序集	193	13.2	位标志	220
10.7	程序集的私有方式部署	194	13.2.1	Flags 特性	222
10.8	共享程序集和 GAC	194	13.2.2	使用位标志的示例	223
10.8.1	把程序集安装到 GAC	194	13.3	关于枚举的补充	224
10.8.2	GAC 内的并行执行	195	第 14 章 数组		226
10.9	配置文件	196	14.1	数组	226
10.10	延迟签名	196	14.1.1	定义	226
第 11 章 异常		198	14.1.2	重要细节	227
11.1	什么是异常	198	14.2	数组的类型	227
11.2	try 语句	199	14.3	数组是对象	228
11.3	异常类	200	14.4	一维数组和矩形数组	229
11.4	catch 子句	200	14.5	实例化一维数组或矩形数组	229
11.4.1	使用特定 catch 子句的示例	201	14.6	访问数组元素	230
11.4.2	catch 子句段	202	14.7	初始化数组	231
11.5	finally 块	203	14.7.1	显式初始化一维数组	231
11.6	为异常寻找处理代码	204	14.7.2	显式初始化矩形数组	232
11.7	更进一步搜索	204	14.7.3	初始化矩形数组的语法点	232
11.7.1	一般法则	205	14.7.4	快捷语法	233
11.7.2	搜索调用栈的示例	206	14.7.5	隐式类型数组	233
11.8	抛出异常	207	14.7.6	综合内容	234
11.9	不带异常对象的抛出	208	14.8	交错数组	234
第 12 章 结构		210	14.8.1	声明交错数组	235
12.1	什么是结构	210	14.8.2	快捷实例化	235
12.2	结构是值类型	211	14.8.3	实例化交错数组	235
12.3	对结构赋值	212	14.8.4	交错数组中的子数组	236
12.4	构造函数和析构函数	213	14.9	比较矩形数组和交错数组	237
12.4.1	实例构造函数	213	14.10	foreach 语句	238
12.4.2	静态构造函数	214	14.10.1	迭代变量是只读的	239

14.10.2 foreach 语句和多维数组	240	17.2 声明接口	276
14.11 数组协变	241	17.3 实现接口	277
14.12 数组继承的有用成员	242	17.4 接口是引用类型	278
14.13 比较数组类型	245	17.5 接口和 as 运算符	280
第 15 章 委托	246	17.6 实现多个接口	280
15.1 什么是委托	246	17.7 实现具有重复成员的接口	281
15.2 声明委托类型	247	17.8 多个接口的引用	282
15.3 创建委托对象	248	17.9 派生成员作为实现	284
15.4 赋值委托	249	17.10 显式接口成员实现	284
15.5 组合委托	250	17.11 接口可以继承接口	287
15.6 为委托增加方法	250	第 18 章 转换	290
15.7 从委托移除方法	251	18.1 什么是转换	290
15.8 调用委托	251	18.2 隐式转换	291
15.9 委托的示例	252	18.3 显式转换和强制转换	292
15.10 调用带返回值的委托	253	18.4 转换的类型	293
15.11 调用带引用参数的委托	254	18.5 数字的转换	294
15.12 匿名方法	255	18.5.1 隐式数字转换	294
15.12.1 使用匿名方法	255	18.5.2 溢出检测上下文	295
15.12.2 匿名方法的语法	256	18.5.3 显式数字转换	296
15.12.3 变量和参数的作用域	257	18.6 引用转换	299
15.13 Lambda 表达式	259	18.6.1 隐式引用转换	300
第 16 章 事件	261	18.6.2 显式引用转换	301
16.1 事件和委托相似	261	18.6.3 有效显式引用转换	301
16.2 源代码组件概览	262	18.7 装箱转换	302
16.3 声明事件	263	18.8 拆箱转换	304
16.3.1 事件是成员	263	18.9 用户自定义转换	305
16.3.2 委托类型和 EventHandler	264	18.9.1 用户自定义转换的约束	305
16.4 触发事件	264	18.9.2 用户自定义转换的示例	306
16.5 订阅事件	265	18.9.3 计算用户自定义转换	307
16.6 标准事件的用法	267	18.9.4 多步用户自定义转换的示例	308
16.6.1 使用 EventArgs 类	267	18.10 is 运算符	309
16.6.2 通过扩展 EventArgs 来传递 数据	268	18.11 as 运算符	310
16.6.3 使用自定义委托	268	第 19 章 泛型	311
16.7 MyTimerClass 代码	270	19.1 什么是泛型	311
16.8 事件访问器	271	19.2 C#中的泛型	313
第 17 章 接口	273	19.3 泛型类	314
17.1 什么是接口	273	19.4 声明泛型类	314
		19.5 创建构造类型	315

19.6	创建变量和实例	316	21.2	LINQ 提供程序	353
19.6.1	使用泛型的栈的示例	317	21.3	查询语法和方法语法	355
19.6.2	比较泛型和非泛型栈	318	21.4	查询变量	356
19.7	类型参数的约束	319	21.5	查询表达式的结构	357
19.7.1	Where 子句	320	21.5.1	from 子句	358
19.7.2	约束类型和次序	320	21.5.2	join 子句	359
19.8	泛型结构	321	21.5.3	什么是联结	360
19.9	泛型接口	322	21.5.4	查询主体中的 from...let... where 片段	362
19.9.1	使用泛型接口的示例	323	21.5.5	orderby 子句	365
19.9.2	泛型接口的实现必须唯一	323	21.5.6	select...group 子句	366
19.10	泛型委托	324	21.5.7	查询中的匿名类型	367
19.11	泛型方法	326	21.5.8	group 子句	368
19.11.1	声明泛型方法	326	21.5.9	查询延续	369
19.11.2	调用泛型方法	327	21.6	标准查询运算符	370
19.11.3	泛型方法的示例	328	21.6.1	查询表达式和标准查询 运算符	371
19.12	扩展方法和泛型类	329	21.6.2	标准查询运算符的签名	372
第 20 章	枚举数和迭代器	331	21.6.3	委托作为参数	373
20.1	枚举数和可枚举类型	331	21.6.4	LINQ 预定义的委托类型	374
20.1.1	使用 foreach 语句	331	21.6.5	使用委托参数的示例	375
20.1.2	枚举数类型	332	21.6.6	使用 Lambda 表达式参数的 示例	375
20.2	使用 IEnumerator 接口	333	21.7	LINQ to XML	376
20.3	IEnumerable 接口	336	21.7.1	标记语言	377
20.4	不实现接口的枚举数	338	21.7.2	XML 基础	377
20.5	泛型枚举接口	339	21.7.3	XML 类	378
20.6	IEnumerator<T>接口	339	21.7.4	使用 XML 树的值	381
20.7	IEnumerable<T>接口	341	21.7.5	使用 XML 属性	384
20.8	迭代器	342	21.7.6	节点的其他类型	387
20.8.1	迭代器块	343	21.7.7	使用 LINQ to XML 的 LINQ 查询	388
20.8.2	使用迭代器来创建枚举数	344	第 22 章	异步编程简介	391
20.8.3	使用迭代器来创建可枚 举类型	345	22.1	进程、线程以及异步编程	391
20.9	常见迭代器模式	346	22.1.1	多线程处理带来的问题	391
20.10	产生可枚举类型和枚举数	347	22.1.2	多线程处理的复杂度	392
20.11	产生多个可枚举类型	348	22.2	异步编程模式	392
20.12	产生多个枚举数	349	22.3	BeginInvoke 和 EndInvoke	393
20.13	迭代器实质	350	22.3.1	等待-直到结束模式	395
第 21 章	介绍 LINQ	352			
21.1	什么是 LINQ	352			

22.3.2 AsyncResult 类	396	24.8.1 声明自定义特性	421
22.3.3 轮询模式	396	24.8.2 使用特性构造函数	421
22.3.4 回调模式	397	24.8.3 指定构造函数	421
22.4 计时器	400	24.8.4 使用构造函数	422
第 23 章 预处理指令	403	24.8.5 构造函数中的位置参数和 命名参数	422
23.1 什么是预处理指令	403	24.8.6 限制特性的使用	423
23.2 基本规则	403	24.8.7 自定义特性的最佳实践	424
23.3 #define 和#undef 指令	404	24.9 访问特性	425
23.4 条件编译	405	24.9.1 使用 IsDefined 方法	425
23.5 条件编译结构	406	24.9.2 使用 GetCustomAttribute 方法	426
23.6 诊断指令	408	第 25 章 其他主题	428
23.7 行号指令	408	25.1 概述	428
23.8 区域指令	409	25.2 字符串	428
23.9 #pragma warning 指令	410	25.2.1 使用 StringBuilder 类	429
第 24 章 反射和特性	411	25.2.2 格式化数字字符串	430
24.1 元数据和反射	411	25.3 把字符串解析为数据值	433
24.2 Type 类	411	25.4 可空类型	434
24.3 获取 Type 对象	413	25.4.1 创建可空类型	434
24.4 什么是特性	415	25.4.2 为可空类型赋值	436
24.5 应用特性	415	25.4.3 使用可空用户自定义类型	437
24.6 预定义的保留的特性	416	25.5 Main 方法	438
24.6.1 Obsolete 特性	416	25.6 文档注释	440
24.6.2 Conditional 特性	417	25.6.1 插入文档注释	440
24.6.3 预定义的特性	418	25.6.2 使用其他 XML 标签	441
24.7 有关应用特性的更多内容	419	25.7 嵌套类型	441
24.7.1 多个特性	419	25.7.1 嵌套类的示例	442
24.7.2 其他类型的目标	419	25.7.2 可见性和嵌套类型	443
24.7.3 全局特性	420		
24.8 自定义特性	420		

第 1 章

C#和.NET框架

1

本章内容

- 在.NET之前
- 进入Microsoft .NET
- 编译成CIL
- 编译成本机代码并执行
- CLR
- CLI
- 缩写回顾

1.1 在.NET 之前

C#编程语言是为开发微软公司的.NET框架^①上的程序而设计的。本章将简要介绍.NET从何而来，以及它的基本架构。这只是为了确保你从正确的一步开始，让我借此机会提醒你一件可能显而易见的事情：C#的发音为*see sharp*^②。

1.1.1 20世纪90年代后期的Windows编程

在20世纪90年代后期，使用微软平台的Windows编程分化成许多分支。大多数程序员在使用Visual Basic (VB)、C或C++。一些C和C++程序员在使用纯Win32 API，但大多数人在使用MFC (Microsoft Foundation Classes，微软基础类库)。其他人已经转向了COM (Component Object Model，组件对象模型)。

所有这些技术都有自己的问题。纯Win32 API不是面向对象的，而且使用它的工作量比使用MFC的更大。MFC是面向对象的，但是它却不一致，并逐渐变得陈旧。COM虽然概念上简单，但它的实际代码复杂，并且需要很多丑陋的、不雅的底层基础代码。

所有这些编程技术的另外一个缺点是它们主要针对桌面程序而不是Internet的开发。那时，Web编程还是以后的事情，而且看起来和桌面编程非常不同。

① 微软正式中文文献中一般称.NET Framework，本书考虑了国内读者习惯，统一译为.NET框架。——编者注

② 有一次我去应聘一个C#编程的职位，当时人力资源面试官问我从事“see pound”（应为see sharp）的经验有多少！我过了一會兒才弄清楚他在说什么。

1.1.2 下一代平台的目标

我们真正需要的是一个新的开始——一个集成的、面向对象的开发框架，它可以把一致和优雅带回编程。为满足这个需求，微软宣布开发一个代码执行环境和一个可以实现这些目标的代码开发环境。这些目标列在图1-1中。



图1-1 下一代平台的目标

1.2 进入 Microsoft .NET

在2002年，微软发布了.NET框架，声称其解决了旧问题并实现了下一代系统的目标。.NET框架是一种比MFC或COM编程技术更一致并面向对象的环境。它的特点包括以下几点。

- 多平台：该系统可以在广泛的计算机上运行，包括从服务器、桌面机到PDA和移动电话。
- 行业标准：该系统使用行业标准的通信协议，比如XML、HTTP、SOAP和WSDL。
- 安全性：该系统能提供更加安全的执行环境，即使有来源可疑的代码存在。

1.2.1 .NET 框架的组成

.NET框架由三部分组成，如图1-2所示。^①

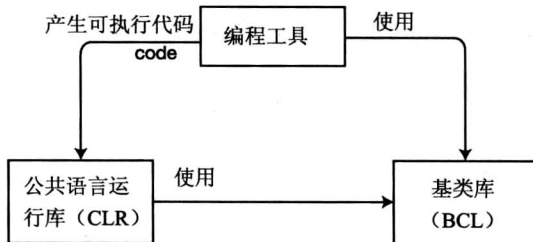


图1-2 .NET框架的组成

执行环境称为CLR（Common Language Runtime，公共语言运行库）。CLR在运行期管理程序的执行，包括以下内容。

- 内存管理

^① 严格地说，.NET框架由两部分组成：CLR和FCL（框架类库），不包括工具。FCL是BCL的超集，还包括Windows Forms、ASP.NET LINQ以及更多命名空间。——编者注