

Illustrated C# 2012 **Fourth Edition**

C#图解教程

(第4版)

迄今为止最容易看懂的一本C#入门图书

[美] Daniel M. Solis 著
姚琪琳 苏林 朱晔 等 译



人民邮电出版社
POSTS & TELECOM PRESS

TURING

图灵程序设计丛书

Illustrated C# 2012 Fourth Edition

C#图解教程

(第4版)

[美] Daniel M. Solis 著
姚琪琳 苏林 朱晔 等 译

人民邮电出版社
北 京

图书在版编目 (C I P) 数据

C#图解教程 : 第4版 / (美) 索利斯 (Solis, D. M.)
著 ; 姚琪琳等译. — 北京 : 人民邮电出版社, 2013.7
(图灵程序设计丛书)
书名原文: Illustrated C# 2012, Fourth Edition
ISBN 978-7-115-32090-2

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

中国版本图书馆CIP数据核字 (2013) 第115305号

内 容 提 要

本书是广受赞誉的 C# 图解教程的最新版本。作者在本书中创造了一种全新的可视化叙述方式, 以图文并茂的形式、朴实简洁的文字, 并辅以大量表格和代码示例, 全面、直观地阐述了 C# 语言的各种特性。新版本除了精心修订旧版内容外, 还全面涵盖了 C# 5.0 的新增特性, 比如异步编程、调用者信息、case 表达式、带参数的泛型构造函数、支持 null 类型运算等。通过本书, 读者能够快速、深入地理解 C#, 为自己的编程生涯打下良好的基础。

本书是 C# 入门的经典好书, 适合对 C# 感兴趣的所有读者。

-
- ◆ 著 [美] Daniel M. Solis
 - 译 姚琪琳 苏林 朱晔等
 - 责任编辑 刘美英
 - 责任印制 焦志炜
 - ◆ 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 电子邮件 315@ptpress.com.cn
 - 网址 <http://www.ptpress.com.cn>
 - 北京艺辉印刷有限公司印刷
 - ◆ 开本: 800×1000 1/16
 - 印张: 33.75
 - 字数: 798千字 2013年7月第1版
 - 印数: 1-3 000册 2013年7月北京第1次印刷
 - 著作权合同登记号 图字: 01-2013-3659号
-

定价: 89.00元

读者服务热线: (010)51095186转604 印装质量热线: (010)67129223

反盗版热线: (010)67171154

广告经营许可证: 京崇工商广字第 0021 号

第 3 版译者序

C#是一门基于.NET的高级语言，正是因为C#处于.NET温暖的怀抱，所以许多C#程序员，甚至许多C#高级程序员对.NET在内存和指令等本质问题上的认识不够。况且有许多使用C#的程序员在使用ASP.NET技术进行网站开发，他们有的从脚本语言转型而来，有的在充分学习C#的情况下就投入了开发工作，那么他们可能对本质问题的认识就更差一点。但是笔者认为，不管怎么样，都非常有必要更深入理解语言背后的机制，而不仅仅停留在掌握API使用的层次上。只有这样，你才能意识到很多bug的关键点和性能问题的关键点，并且理解那些高级的特性。

从目录上来看本书就像其他C#入门书籍一样，介绍了一个又一个语言特性，但是如果你翻阅一下正文就会发现它的不同。可能因为作者有C/C++的背景的关系，对于每一个语言特性，作者对其使用方式只是轻描淡写，而对特性背后的机制做了浓墨重彩的介绍，并且在文字介绍中穿插大量图示来展现内存对象的面貌。其实，市面上很多所谓的进阶书籍都只是介绍如何使用那些高级API、高级特性，而忽略了语言本质，但这一块恰巧是最重要的。因此，对于那些用了几年C#的程序员来说，本书具有非常大的价值。

不管怎样，一句话，本书值得一读。但是由于时间仓促，本书在翻译过程中难免出现失误。如果有任何问题，欢迎来信交流，笔者邮箱为yzhu@live.com。

朱 晔

2011年3月

第2版译者序

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注新技术和新平台的参考资料。一本描述清晰、内容详实的书能使我们快速掌握这些技术。

笔者不才，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

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。C#是一门非常棒的编程语言，我喜欢用它编写代码！这些年来，我自己都不记得学过多少门编程语言了，但C#一直是我的最爱。我希望购买本书的读者能从书中读到C#的美和优雅。

大多数编程图书以文字为主要载体。对于小说而言，文字形式当然是最恰当不过了，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。

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工作中我发现，我在白板上画的图能帮助学生最快地理解我要传达的概念。然而，单靠图表并不足以解释一种编程语言和平台。本书的目标是以最佳方式结合文字和图表，使你对这种语言有透彻的理解，并且让本书能用作参考工具。

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明晰，无需费力卒读就能直接获得想要的信息。无论哪种程序员，内容本身的图形化呈现方式都能帮助你更容易地学习本书。

祝学习愉快！

目标读者、源代码和联系信息

本书针对编程新手和中级水平的程序员，当然还有对C#感兴趣的其他语言编程人员（如Visual Basic和Java）。我尽力专注C#语言本身，详尽深入地描述语言及各部分，少涉及.NET和相关编程实践。本书写作过程中，笔者始终坚持确保内容简洁性的同时又能透彻地讲解这门语言。如果读者对其他主题感兴趣，有大量好书值得推荐。

你可以从Apress网站本书页面（www.illustratedcsharp.com）下载书中所有示例程序的源代码。尽管我不能回答有关代码的一些细节问题，但是你可以通过dansolis@sbcglobal.net和我取得联系，提出建议或反馈。

我希望本书可以让你享受学习C#的过程！祝你好运！

致 谢

感谢Sian每天支持并鼓励我，感谢我的父母、兄弟姐妹，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，是他们与我携手完成了本书。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	2.8.3 注释类型总结.....22
1.1 在.NET 之前.....1	第 3 章 类型、存储和变量.....23
1.1.1 20 世纪 90 年代末的 Windows 编程.....1	3.1 C#程序是一组类型声明.....23
1.1.2 下一代平台服务的目标.....2	3.2 类型是一种模板.....24
1.2 .NET 时代.....2	3.3 实例化类型.....24
1.2.1 .NET 框架的组成.....2	3.4 数据成员和函数成员.....25
1.2.2 大大改进的编程环境.....3	3.5 预定义类型.....26
1.3 编译成 CIL.....5	3.6 用户定义类型.....27
1.4 编译成本机代码并执行.....6	3.7 栈和堆.....28
1.5 CLR.....7	3.7.1 栈.....28
1.6 CLI.....8	3.7.2 堆.....29
1.7 各种缩写.....9	3.8 值类型和引用类型.....30
1.8 C#的演化.....9	3.8.1 存储引用类型对象的成员.....31
第 2 章 C#编程概述.....10	3.8.2 C#类型的分类.....31
2.1 一个简单的 C#程序.....10	3.9 变量.....32
2.2 标识符.....12	3.9.1 变量声明.....32
2.3 关键字.....12	3.9.2 多变量声明.....34
2.4 Main: 程序的起始点.....13	3.9.3 使用变量的值.....34
2.5 空白.....13	3.10 静态类型和 dynamic 关键字.....34
2.6 语句.....14	3.11 可空类型.....35
2.7 从程序中输出文本.....15	第 4 章 类的基本概念.....36
2.7.1 Write.....15	4.1 类的概述.....36
2.7.2 WriteLine.....15	4.2 程序和类: 一个快速示例.....37
2.7.3 格式字符串.....16	4.3 声明类.....38
2.7.4 多重标记和值.....16	4.4 类成员.....38
2.7.5 格式化数字字符串.....17	4.4.1 字段.....38
2.8 注释: 为代码添加注解.....20	4.4.2 方法.....40
2.8.1 关于注释的补充.....21	4.5 创建变量和类的实例.....41
2.8.2 文档注释.....21	4.6 为数据分配内存.....41

4.7 实例成员	42	6.5.1 静态字段示例	87
4.8 访问修饰符	43	6.5.2 静态成员的生存期	87
4.9 从类的内部访问成员	45	6.6 静态函数成员	88
4.10 从类的外部访问成员	46	6.7 其他静态类成员类型	89
4.11 综合应用	47	6.8 成员常量	90
第 5 章 方法	49	6.9 常量与静态量	90
5.1 方法的结构	49	6.10 属性	91
5.2 方法体内部的代码执行	50	6.10.1 属性声明和访问器	92
5.3 本地变量	51	6.10.2 属性示例	93
5.3.1 类型推断和 var 关键字	52	6.10.3 使用属性	94
5.3.2 嵌套块中的本地变量	52	6.10.4 属性和关联字段	94
5.4 本地常量	53	6.10.5 执行其他计算	96
5.5 控制流	54	6.10.6 只读和只写属性	96
5.6 方法调用	55	6.10.7 属性与公共字段	97
5.7 返回值	56	6.10.8 计算只读属性示例	97
5.8 返回语句和 void 方法	57	6.10.9 自动实现属性	98
5.9 参数	59	6.10.10 静态属性	99
5.9.1 形参	59	6.11 实例构造函数	100
5.9.2 实参	59	6.11.1 带参数的构造函数	101
5.10 值参数	61	6.11.2 默认构造函数	102
5.11 引用参数	63	6.12 静态构造函数	102
5.12 引用类型作为值参数和引用参数	65	6.13 对象初始化语句	104
5.13 输出参数	68	6.14 析构函数	105
5.14 参数数组	70	6.15 readonly 修饰符	105
5.14.1 方法调用	71	6.16 this 关键字	106
5.14.2 用数组作为实参	73	6.17 索引器	107
5.15 参数类型总结	74	6.17.1 什么是索引器	108
5.16 方法重载	74	6.17.2 索引器和属性	108
5.17 命名参数	75	6.17.3 声明索引器	109
5.18 可选参数	76	6.17.4 索引器的 set 访问器	110
5.19 栈帧	79	6.17.5 索引器的 get 访问器	110
5.20 递归	81	6.17.6 关于索引器的补充	111
第 6 章 深入理解类	83	6.17.7 为 Employee 示例声明 索引器	111
6.1 类成员	83	6.17.8 另一个索引器的示例	112
6.2 成员修饰符的顺序	84	6.17.9 索引器重载	113
6.3 实例类成员	85	6.18 访问器的访问修饰符	114
6.4 静态字段	86	6.19 分部类和分部类型	115
6.5 从类的外部访问静态成员	86	6.20 分部方法	116

第 7 章 类和继承	118	8.2.4 字符串字面量	154
7.1 类继承	118	8.3 求值顺序	156
7.2 访问继承的成员	119	8.3.1 优先级	156
7.3 所有类都派生自 object 类	120	8.3.2 结合性	157
7.4 屏蔽基类的成员	121	8.4 简单算术运算符	157
7.5 基类访问	123	8.5 求余运算符	158
7.6 使用基类的引用	124	8.6 关系比较运算符和相等比较运 算符	159
7.6.1 虚方法和覆写方法	125	8.7 递增运算符和递减运算符	160
7.6.2 覆写标记为 override 的 方法	127	8.8 条件逻辑运算符	162
7.6.3 覆盖其他成员类型	130	8.9 逻辑运算符	163
7.7 构造函数的执行	130	8.10 移位运算符	164
7.7.1 构造函数初始化语句	132	8.11 赋值运算符	165
7.7.2 类访问修饰符	134	8.12 条件运算符	167
7.8 程序集间的继承	134	8.13 一元算术运算符	168
7.9 成员访问修饰符	136	8.14 用户定义的类型转换	169
7.9.1 访问成员的区域	137	8.15 运算符重载	172
7.9.2 公有成员的可访问性	138	8.15.1 运算符重载的限制	172
7.9.3 私有成员的可访问性	138	8.15.2 运算符重载的示例	173
7.9.4 受保护成员的可访问性	138	8.16 typeof 运算符	174
7.9.5 内部成员的可访问性	139	8.17 其他运算符	176
7.9.6 受保护内部成员的可访 问性	139		
7.9.7 成员访问修饰符小结	140	第 9 章 语句	177
7.10 抽象成员	141	9.1 什么是语句	177
7.11 抽象类	142	9.2 表达式语句	178
7.11.1 抽象类和抽象方法的 示例	142	9.3 控制流语句	179
7.11.2 抽象类的另一个例子	143	9.4 if 语句	179
7.12 密封类	144	9.5 if...else 语句	180
7.13 静态类	144	9.6 while 循环	181
7.14 扩展方法	145	9.7 do 循环	182
7.15 命名约定	148	9.8 for 循环	183
第 8 章 表达式和运算符	150	9.8.1 for 语句中变量的作用域	185
8.1 表达式	150	9.8.2 初始化和迭代表达式中的 多表达式	185
8.2 字面量	151	9.9 switch 语句	186
8.2.1 整数字面量	152	9.9.1 分支示例	187
8.2.2 实数字面量	153	9.9.2 switch 语句的补充	188
8.2.3 字符串字面量	153	9.9.3 分支标签	189
		9.10 跳转语句	189
		9.11 break 语句	190

9.12	continue 语句	190	12.2	数组的类型	217
9.13	标签语句	191	12.3	数组是对象	218
9.13.1	标签	191	12.4	一维数组和矩形数组	219
9.13.2	标签语句的作用域	192	12.5	实例化一维数组或矩形数组	220
9.14	goto 语句	192	12.6	访问数组元素	220
9.15	using 语句	193	12.7	初始化数组	221
9.15.1	资源的包装使用	194	12.7.1	显式初始化一维数组	222
9.15.2	using 语句的示例	195	12.7.2	显式初始化矩形数组	222
9.15.3	多个资源和嵌套	196	12.7.3	初始化矩形数组的语法点	223
9.15.4	using 语句的另一种形式	197	12.7.4	快捷语法	223
9.16	其他语句	197	12.7.5	隐式类型数组	223
第 10 章	结构	198	12.7.6	综合内容	224
10.1	什么是结构	198	12.8	交错数组	224
10.2	结构是值类型	199	12.8.1	声明交错数组	225
10.3	对结构赋值	200	12.8.2	快捷实例化	225
10.4	构造函数和析构函数	201	12.8.3	实例化交错数组	226
10.4.1	实例构造函数	201	12.8.4	交错数组中的子数组	227
10.4.2	静态构造函数	202	12.9	比较矩形数组和交错数组	228
10.4.3	构造函数和析构函数小结	203	12.10	foreach 语句	229
10.5	字段初始化语句是不允许的	203	12.10.1	迭代变量是只读的	230
10.6	结构是密封的	203	12.10.2	foreach 语句和多维数组	231
10.7	装箱和拆箱	203	12.11	数组协变	232
10.8	结构作为返回值和参数	204	12.12	数组继承的有用成员	233
10.9	关于结构的其他信息	204	12.13	比较数组类型	236
第 11 章	枚举	205	第 13 章	委托	237
11.1	枚举	205	13.1	什么是委托	237
11.1.1	设置底层类型和显式值	206	13.2	委托概述	239
11.1.2	隐式成员编号	207	13.3	声明委托类型	240
11.2	位标志	208	13.4	创建委托对象	241
11.2.1	Flags 特性	210	13.5	给委托赋值	242
11.2.2	使用位标志的示例	212	13.6	组合委托	243
11.3	关于枚举的补充	213	13.7	为委托添加方法	243
第 12 章	数组	216	13.8	从委托移除方法	244
12.1	数组	216	13.9	调用委托	244
12.1.1	定义	216	13.10	委托的示例	245
12.1.2	重要细节	217	13.11	调用带返回值的委托	246
			13.12	调用带引用参数的委托	247

13.13 匿名方法	248	16.5.2 溢出检测上下文	291
13.13.1 使用匿名方法	249	16.5.3 显式数字转换	292
13.13.2 匿名方法的语法	249	16.6 引用转换	295
13.13.3 变量和参数的作用域	250	16.6.1 隐式引用转换	296
13.14 Lambda 表达式	252	16.6.2 显式引用转换	297
第 14 章 事件	255	16.6.3 有效显式引用转换	298
14.1 发布者和订阅者	255	16.7 装箱转换	299
14.2 源代码组件概览	257	16.7.1 装箱是创建副本	300
14.3 声明事件	257	16.7.2 装箱转换	300
14.4 订阅事件	258	16.8 拆箱转换	301
14.5 触发事件	259	16.9 用户自定义转换	302
14.6 标准事件的使用	261	16.9.1 用户自定义转换的约束	302
14.6.1 通过扩展 EventArgs		16.9.2 用户自定义转换的示例	302
来传递数据	262	16.9.3 评估用户自定义转换	304
14.6.2 移除事件处理程序	264	16.9.4 多步用户自定义转换的	
14.7 事件访问器	265	示例	304
第 15 章 接口	267	16.10 is 运算符	305
15.1 什么是接口	267	16.11 as 运算符	306
15.2 声明接口	272	第 17 章 泛型	308
15.3 实现接口	273	17.1 什么是泛型	308
15.4 接口是引用类型	275	17.2 C#中的泛型	310
15.5 接口和 as 运算符	276	17.3 泛型类	311
15.6 实现多个接口	276	17.4 声明泛型类	312
15.7 实现具有重复成员的接口	277	17.5 创建构造类型	312
15.8 多个接口的引用	279	17.6 创建变量和实例	313
15.9 派生成员作为实现	280	17.6.1 使用泛型的栈的示例	315
15.10 显式接口成员实现	281	17.6.2 比较泛型和非泛型栈	316
15.11 接口可以继承接口	283	17.7 类型参数的约束	317
15.12 不同类实现一个接口的		17.7.1 Where 子句	317
示例	284	17.7.2 约束类型和次序	318
第 16 章 转换	286	17.8 泛型方法	319
16.1 什么是转换	286	17.8.1 声明泛型方法	319
16.2 隐式转换	287	17.8.2 调用泛型方法	320
16.3 显式转换和强制转换	288	17.8.3 泛型方法的示例	321
16.4 转换的类型	289	17.9 扩展方法和泛型类	322
16.5 数字的转换	290	17.10 泛型结构	323
16.5.1 隐式数字转换	290	17.11 泛型委托	323
		17.12 泛型接口	325
		17.12.1 使用泛型接口的示例	326

17.12.2 泛型接口的实现必须 唯一.....	327
17.13 协变	328
17.14 逆变	330
17.14.1 接口的协变和逆变	332
17.14.2 有关可变性的更多内容	333
第 18 章 枚举器和迭代器	335
18.1 枚举器和可枚举类型	335
18.2 IEnumerator 接口	337
18.3 泛型枚举接口	341
18.4 迭代器	343
18.4.1 迭代器块	343
18.4.2 使用迭代器来创建枚举器	344
18.4.3 使用迭代器来创建可枚举 类型	346
18.5 常见迭代器模式	347
18.6 产生多个可枚举类型	348
18.7 将迭代器作为属性	349
18.8 迭代器实质	350
第 19 章 LINQ	352
19.1 什么是 LINQ	352
19.2 LINQ 提供程序	353
19.3 方法语法和查询语法	355
19.4 查询变量	356
19.5 查询表达式的结构	357
19.5.1 from 子句	358
19.5.2 join 子句	359
19.5.3 什么是联结	360
19.5.4 查询主体中的 from...let ...where 片段	362
19.5.5 orderby 子句	365
19.5.6 select...group 子句	366
19.5.7 查询中的匿名类型	367
19.5.8 group 子句	368
19.5.9 查询延续: into 子句	369
19.6 标准查询运算符	370
19.6.1 标准查询运算符的签名	373
19.6.2 查询表达式和标准查询运 算符	374
19.6.3 将委托作为参数	375
19.6.4 LINQ 预定义的委托 类型	376
19.6.5 使用委托参数的示例	377
19.6.6 使用 Lambda 表达式参数的 示例	378
19.7 LINQ to XML	379
19.7.1 标记语言	379
19.7.2 XML 基础	379
19.7.3 XML 类	381
19.7.4 使用 XML 特性	387
19.7.5 节点的其他类型	389
19.7.6 使用 LINQ to XML 的 LINQ 查询	390
第 20 章 异步编程	393
20.1 什么是异步	393
20.2 async/await 特性的结构	398
20.3 什么是异步方法	399
20.3.1 异步方法的控制流	402
20.3.2 await 表达式	404
20.3.3 取消一个异步操作	407
20.3.4 异常处理和 await 表达式	409
20.3.5 在调用方法中同步地等待 任务	410
20.3.6 在异步方法中异步地等待 任务	413
20.3.7 Task.Delay 方法	415
20.4 在 GUI 程序中执行异步操作	416
20.5 使用异步 Lambda 表达式	420
20.6 完整的 GUI 程序	421
20.7 BackgroundWorker 类	423
20.8 并行循环	428
20.9 其他异步编程模式	430
20.10 BeginInvoke 和 EndInvoke	431
20.10.1 等待—直到结束模式	432
20.10.2 AsyncResult 类	433
20.10.3 轮询模式	434
20.10.4 回调模式	435
20.11 计时器	438

第 21 章 命名空间和程序集.....	440	23.5 条件编译结构.....	475
21.1 引用其他程序集.....	440	23.6 诊断指令.....	476
21.2 命名空间.....	444	23.7 行号指令.....	477
21.2.1 命名空间名称.....	447	23.8 区域指令.....	478
21.2.2 命名空间的补充.....	447	23.9 #pragma warning 指令.....	479
21.2.3 命名空间跨文件伸展.....	448		
21.2.4 嵌套命名空间.....	449	第 24 章 反射和特性.....	480
21.3 using 指令.....	450	24.1 元数据和反射.....	480
21.3.1 using 命名空间指令.....	450	24.2 Type 类.....	480
21.3.2 using 别名指令.....	451	24.3 获取 Type 对象.....	482
21.4 程序集的结构.....	451	24.4 什么是特性.....	484
21.5 程序集标识符.....	453	24.5 应用特性.....	485
21.6 强命名程序集.....	453	24.6 预定义的保留的特性.....	485
21.7 程序集的私有方式部署.....	455	24.6.1 Obsolete 特性.....	485
21.8 共享程序集和 GAC.....	455	24.6.2 Conditional 特性.....	486
21.8.1 把程序集安装到 GAC.....	455	24.6.3 调用者信息特性.....	488
21.8.2 GAC 内的并肩执行.....	456	24.6.4 DebuggerStepThrough 特性.....	488
21.9 配置文件.....	457	24.6.5 其他预定义特性.....	489
21.10 延迟签名.....	457	24.7 有关应用特性的更多内容.....	490
第 22 章 异常.....	459	24.7.1 多个特性.....	490
22.1 什么是异常.....	459	24.7.2 其他类型的目标.....	490
22.2 try 语句.....	460	24.7.3 全局特性.....	491
22.3 异常类.....	461	24.8 自定义特性.....	491
22.4 catch 子句.....	462	24.8.1 声明自定义特性.....	492
22.5 使用特定 catch 子句的示例.....	462	24.8.2 使用特性的构造函数.....	492
22.6 catch 子句段.....	463	24.8.3 指定构造函数.....	492
22.7 finally 块.....	464	24.8.4 使用构造函数.....	493
22.8 为异常寻找处理程序.....	465	24.8.5 构造函数中的位置参数和 命名参数.....	493
22.9 更进一步搜索.....	466	24.8.6 限制特性的使用.....	494
22.9.1 一般法则.....	466	24.8.7 自定义特性的最佳实践.....	495
22.9.2 搜索调用栈的示例.....	467	24.9 访问特性.....	496
22.10 抛出异常.....	469	24.9.1 使用 IsDefined 方法.....	496
22.11 不带异常对象的抛出.....	470	24.9.2 使用 GetCustomAttributes 方法.....	497
第 23 章 预处理指令.....	472	第 25 章 其他主题.....	499
23.1 什么是预处理指令.....	472	25.1 概述.....	499
23.2 基本规则.....	472	25.2 字符串.....	499
23.3 #define 和#undef 指令.....	473		
23.4 条件编译.....	474		

25.3	使用 <code>StringBuilder</code> 类	501
25.4	把字符串解析为数据值	502
25.5	关于可空类型的更多内容	503
25.5.1	为可空类型赋值	505
25.5.2	使用空接合运算符	505
25.5.3	使用可空用户自定义 类型	506
25.6	Main 方法	508
25.7	文档注释	509
25.7.1	插入文档注释	510

25.7.2	使用其他 XML 标签	510
25.8	嵌套类型	511
25.8.1	嵌套类的示例	512
25.8.2	可见性和嵌套类型	513
25.9	析构函数和 <code>dispose</code> 模式	514
25.9.1	标准 <code>dispose</code> 模式	515
25.9.2	比较构造函数和析构函数	517
25.10	和 COM 的互操作	518
索引		521

第 1 章

C#和.NET框架

1

本章内容

- ☐ 在.NET之前
- ☐ .NET时代
- ☐ 编译成CIL
- ☐ 编译成本机代码并执行
- ☐ CLR
- ☐ CLI
- ☐ 缩写回顾
- ☐ C#的演化

1.1 在.NET 之前

C#编程语言是为在微软公司的.NET框架^①上开发程序而设计的。本章将简要介绍.NET从何而来,以及它的基本架构。在开始之前,我要指出C#的正确发音: see sharp^②。

1.1.1 20 世纪 90 年代末的Windows编程

20世纪90年代末,使用微软平台的Windows编程分化成许多分支。大多数程序员使用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 Class, 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虽然概念简单,但

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!我过了一会儿才弄清楚他在说什么。

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。

所有这些编程技术还有一个缺点是它们主要针对桌面程序而不是Internet进行开发。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。

1.1.2 下一代平台服务的目标

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，微软打算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。

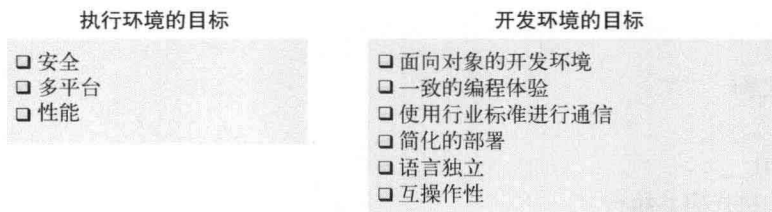


图1-1 下一代平台的目标

1.2 .NET 时代

2002年，微软发布了.NET框架的第一个版本，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和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 该系统可以在各种计算机上运行，从服务器、桌面机到PDA，还能在移动电话上运行。
- ☐ 行业标准 该系统使用行业标准的通信协议，比如XML、HTTP、SOAP、JSON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 执行环境称为CLR（Common Language Runtime，公共语言运行库）。CLR在运行时管理程序的执行，包括以下内容。

- ☐ 内存管理和垃圾收集。
- ☐ 代码安全验证。
- ☐ 代码执行、线程管理及异常处理。

^① 严格地说，.NET框架由CLR和FCL（框架类库）两部分组成，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