

张子阳 著



# .NET之美

## .NET关键技术深入解析

Beautiful .NET: In-depth Analysis of Key Points of .NET

资深.NET技术专家多年开发经验结晶，悉数对C#和.NET中实用的、关键的和难以理解的知识点进行了深入解析，尽显.NET之美

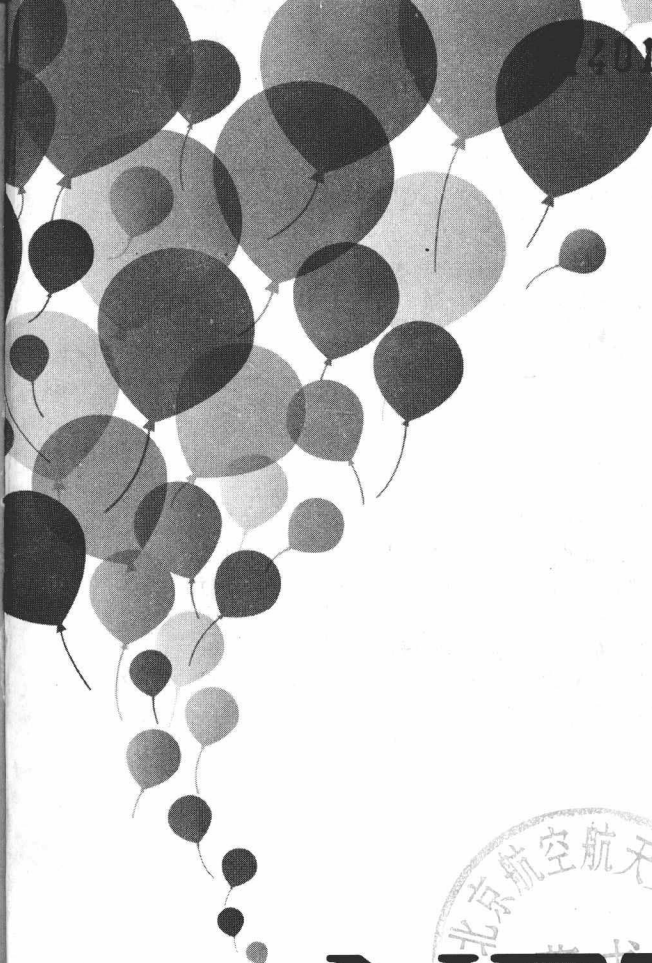
每个知识点都辅之以精心设计的案例，理论与实践并重，是修炼为高级.NET程序员必读的著作之一



机械工业出版社  
China Machine Press

4018956

TP312C  
2253



张子阳 著

# .NET之美

## .NET关键技术深入解析

Beautiful .NET: In-depth Analysis of Key Points of .NET



TP312C  
2253

★ 机械工业出版社



北航

C1705081

## 图书在版编目 (CIP) 数据

---

.NET 之美: .NET 关键技术深入解析 / 张子阳著. —北京: 机械工业出版社, 2014.1

ISBN 978-7-111-44532-6

I. N… II. 张… III. ① C 语言—程序设计 ② 计算机网络—程序设计 IV. ① TP312 ② TP393

中国版本图书馆 CIP 数据核字 (2013) 第 252910 号

---

### 版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问 北京市展达律师事务所

本书是 .NET 程序员进阶修炼的必读之作, 由拥有多年开发经验的资深 .NET 技术专家对 C# 和 .NET 中实用的、关键的和难以理解的知识点进行了深入解析, 旨在帮助读者在尽可能短的时间内以尽可能低的学习成本去掌握那些最应该被掌握的知识。书中的每个知识点都辅之以精心设计的案例, 易于理解, 实践性强。

全书共 17 章, 分为两个部分: 第一部分 (1~5 章) 主要讲解了 C# 语言中的一些关键知识点, 如类型、泛型、委托、事件、对象、LINQ 等; 第二部分 (6~17 章) 则对 .NET 中的关键知识点进行了深入剖析, 如程序集、流和序列化、加密与解密、网络编程、.NET Remoting、在 .NET 中操作 XML、.NET 应用程序配置、基于角色的安全性、反射、多线程、对象生存期与垃圾回收等。

机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码 100037)

责任编辑: 姜 影

藁城市京瑞印刷有限公司印刷

2014 年 1 月第 1 版第 1 次印刷

186mm × 240mm · 27.75 印张

标准书号: ISBN 978-7-111-44532-6

定 价: 79.00 元

---

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010) 88378991 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzjsj@hzbook.com

# 前 言

## 为什么要写这本书

我一直认为写总结是学习的最好方式之一，因为总结的过程中也是一个不断思考的过程，可以让你的领悟更加深刻。作为一名 .NET 开发人员，有很多关键的知识点是必须掌握的，否则就会遇到很多障碍。不善于总结的人很难成为一名合格的 .NET 工程师。我就习惯将所遇到的各种知识点尽可能地研究透彻，然后总结成文章。起初，我将一些文章发表在了我的博客上，出乎意料的是大部分的文章都很受欢迎。有不少朋友留言或者发邮件给我，鼓励我出一本关于 .NET 的书。我想如果我的文章能帮助到更多的朋友，那么出一本又何妨？于是，就有了现在这本书。

写书需要很高的技巧，对于一个技术或者知识点，自己理解它或许还算是容易，但把自己理解的内容写出来让别人也能理解却不是一件容易的事情。书中的每个章节，我都尽量采用循序渐进的方式进行讲解，由一个看似简单、微不足道的知识点进行切入，随后不断延伸，从而展开整个主题。这种方式对于一些朋友来说可能稍嫌累赘，但对于大多数朋友来说会更容易接受。

微软推出 .NET 已经有很多年了，现在市场已经不乏 .NET 的书籍，很多 .NET 开发人员的枕边案头已经堆放了不少一本的 .NET 书籍。因此，在本书中，我不想去重复一些简单基础的内容，因为很多书中都会涉及这方面的知识，例如类型声明、语法、循环语句等；也不想去写一些高深莫测几乎永远也用不到的特性和功能，例如代码访问安全性和一些很底层的东西。本书选择的主题，大多是关键、重要且不是很好理解的，相信认真学完本书，应该就可以帮你打通“.NET 任督二脉”了。

.NET 在过去十年当中的发展可谓是日新月异，经历了多次重要版本更新。2002 推出了 .NET 的第一个版本 1.0；2005 年推出了 .NET 2.0，2.0 的最大变化就是引入了泛型，同时新增了大量的类型；2006 年推出了 .NET 3.0，并预装在 Vista 操作系统中，3.0 主要引入了 WF、WCF、WPF 几项技术；2007 年推出了 .NET 3.5，3.5 中最激动人心的变化就是引入了 LINQ，LINQ 的推出在很大程度上改变了以前程序员编程的习惯和方式；2010 年 .NET 迎来了 4.0 版本，其中的主要革新是加入了动态编程、并行计算、默认参数、协变和逆变；目前，最先的版本是 .NET 4.5，于 2012 年发

布, 4.5 是对 4.0 的一个就地更新。展望未来, .NET 还会不断地变化和发展, 函数式编程、并行计算、动态特性都有可能成为 .NET 继续延伸的方向。

显然, 本书不可能涉及所有这些版本中引入的新特性, 很多的单一主题就足够写一整本书了。但是, 这本书将会帮助你奠定一个良好的 .NET 功底, 有了这个功底以后, 再去学习这些琳琅满目的新特性, 就会变得得心应手了。

## 读者对象

本书的读者对象是已经入门且正在向中高级进阶的 .NET 开发人员, 包括:

- .NET 工程师
- 由其他技术转向学习 .NET 的技术人员
- 学习 .NET 的高校学生
- 开设相关课程的大专院校的师生

## 如何阅读本书

本书分为两大部分:

第一部分为 C# 语言, C# 是为 .NET 而生的语言, 该部分重点讲解了类型基础、泛型、委托、LINQ 几个部分。

第二部分为 .NET 平台, 共挑选了 11 个 .NET 中的重点主题进行讲解。

本书各章的知识点是相对独立的, 因此学习其中一个章并不需要你通读前面的所有章节, 可以自由地选择感兴趣和薄弱的环节进行阅读。

## 勘误和支持

由于我的水平有限, 加之编写时间仓促, 书中难免会出现一些错误或者不准确的地方, 恳请读者批评指正。如果在书中发现错误或疏漏, 可以发送邮件至 [JimmyZhang@SoftComz.com](mailto:JimmyZhang@SoftComz.com) 与我联系, 我会将勘误发表在我的博客上 <http://jimmyzhang.cnblogs.com>。同时, 如果在阅读中遇到任何问题, 也可以给我发送邮件与我进行讨论。

## 致谢

首先要感谢那些鼓励我出一本书的朋友们, 如果不是一次次看到你们的留言、电子邮件、手机

短信催促我出一本书，我是很难下定决心的。

感谢机械工业出版社华章公司的编辑杨福川老师，当我提出出版一本书的时候，你给了我最大的支持。感谢华章公司的姜影编辑，在写作的过程中不断修缮我的稿件，显著提升了我的文字水平和稿件的质量。

感谢 .NET 社区——博客园 `cnblogs`，通过这个平台我认识了非常多的 .NET 行业的朋友，与你们交流是一件很幸福的事。

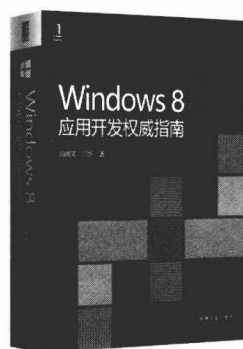
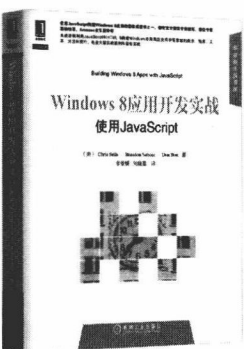
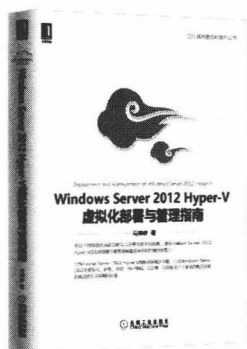
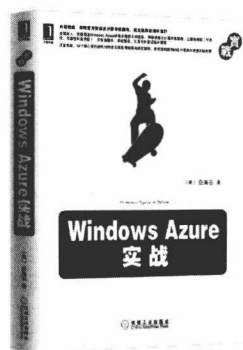
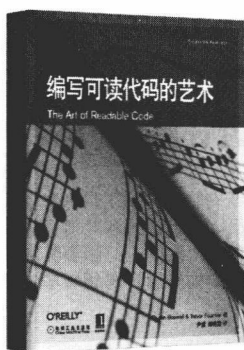
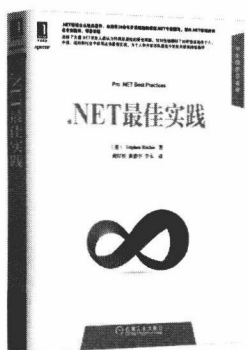
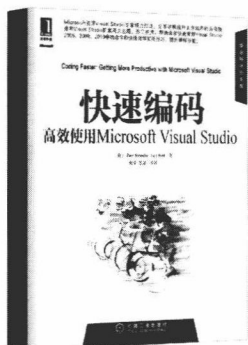
最后感谢我的家人，尤其是我的妻子，在编写本书的过程中，牺牲了很多照顾和陪伴你们的时间。

谨以此书献给众多热爱 .NET 的朋友们！

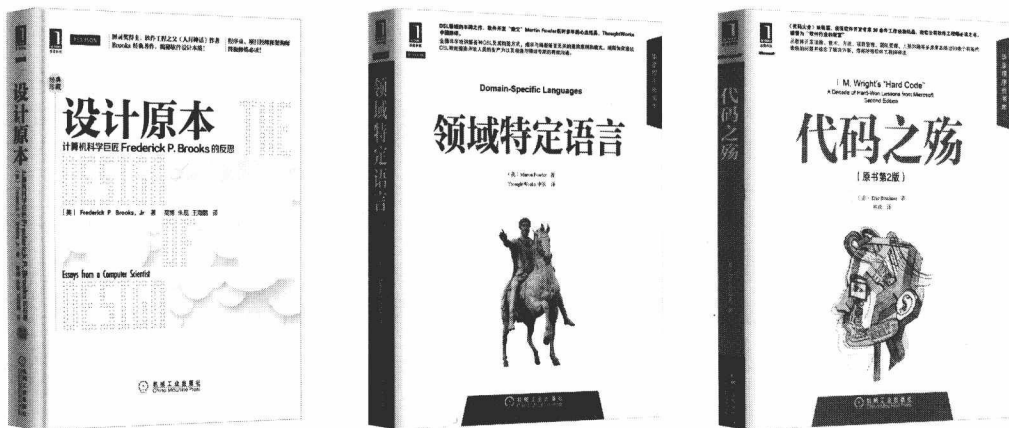
张子阳 (JimmyZhang)

于中国深圳

# 推荐阅读



# 推荐阅读



## 设计原本（精装本）

如果说《人月神话》是近40年来所有软件开发工程师和项目经理们必读的一本书，那么本书将会是未来数十年内从事软件行业的程序员、项目经理和架构师必读的一本书。它是《人月神话》作者、著名计算机科学家、软件工程教父、美国两院院士、图灵奖和IEEE计算机先驱奖得主Brooks在计算机软硬件架构与设计、建筑和组织机构的架构与设计等领域毕生经验的结晶，是计算机图书领域的又一史诗级著作。

## 领域特定语言

本书是DSL领域的丰碑之作，由世界级软件开发大师和软件开发“教父”Martin Fowler历时多年写作而成。全面详尽地讲解了各种DSL及其构造方式，揭示了与编程语言无关的通用原则和模式，阐释了如何通过DSL有效提高开发人员的生产力以及增进与领域专家的有效沟通，能为开发人员选择和使用DSL提供有效的决策依据和指导方法。

## 代码之殇

本书是《代码大全》的姊妹篇，被誉为“软件行业的财富”，资深软件开发专家30余年工作经验的结晶，是微软公司软件工程师必读之书。它从软件开发流程、技术、方法、项目管理、团队管理、人际沟通等多角度总结出90余个具有代表性的问题（大多数问题都可能会给公司或软件项目带来毁灭性灾难），并给出了问题的解决方案和最佳实践，值得所有软件开发工程师和项目管理者研读。



北航

C1705081



# 目 录

## 第一部分 C# 语言基础

### 第 1 章 C# 类型基础 ..... 2

#### 1.1 值类型和引用类型 ..... 2

##### 1.1.1 值类型 ..... 2

##### 1.1.2 引用类型 ..... 5

##### 1.1.3 简单类型 ..... 6

##### 1.1.4 装箱和拆箱 ..... 7

#### 1.2 对象判等 ..... 8

##### 1.2.1 引用类型判等 ..... 9

##### 1.2.2 简单值类型判等 ..... 10

##### 1.2.3 复杂值类型判等 ..... 12

#### 1.3 对象复制 ..... 14

##### 1.3.1 浅度复制 ..... 14

##### 1.3.2 深度复制 ..... 17

#### 1.4 不可变类型 ..... 19

##### 1.4.1 从类型设计谈起, Class 还是 Struct ..... 19

##### 1.4.2 数据不一致的问题 ..... 20

##### 1.4.3 常量性和原子性 ..... 20

##### 1.4.4 避免外部类型对类型 内部的访问 ..... 21

#### 1.5 本章小结 ..... 23

### 第 2 章 C# 中的泛型 ..... 24

#### 2.1 理解泛型 ..... 24

##### 2.1.1 为什么要有泛型 ..... 24

##### 2.1.2 类型参数约束 ..... 27

##### 2.1.3 泛型方法 ..... 31

#### 2.2 泛型与集合类型 ..... 32

##### 2.2.1 避免隐式的装箱和拆箱 ..... 32

##### 2.2.2 编译时的类型安全 ..... 34

##### 2.2.3 使用泛型的一个小技巧 ..... 34

#### 2.3 本章小结 ..... 35

### 第 3 章 C# 中的委托和事件 ..... 36

#### 3.1 理解委托 ..... 36

##### 3.1.1 将方法作为方法的参数 ..... 36

##### 3.1.2 将方法绑定到委托 ..... 39

##### 3.1.3 委托与接口 ..... 41

#### 3.2 事件的由来 ..... 42

##### 3.2.1 更好的封装性 ..... 42

##### 3.2.2 限制类型能力 ..... 45

#### 3.3 委托的编译代码 ..... 47

#### 3.4 .NET 框架中的委托和事件 ..... 48

##### 3.4.1 示例说明 ..... 48

##### 3.4.2 Observer 设计模式简介 ..... 49

3.4.3 实现示例的 Observer 设计模式 .....	50	4.3 本章小结 .....	96
3.4.4 .NET 框架中的委托与事件 ...	51	<b>第 5 章 LINQ</b> .....	97
3.5 委托进阶 .....	53	5.1 LINQ 预备知识 .....	98
3.5.1 为什么委托定义的返回值 通常都为 void .....	53	5.1.1 泛型和委托 .....	98
3.5.2 如何让事件只允许一个 客户订阅 .....	54	5.1.2 隐式类型的局部变量 ...	100
3.5.3 获得多个返回值与异常 处理 .....	56	5.1.3 匿名类型 .....	101
3.6 订阅者方法超时的处理 .....	60	5.1.4 扩展方法 .....	102
3.7 委托和方法的异步调用 .....	64	5.1.5 匿名方法和 Lambda 表达式 .....	104
3.8 不使用委托实现 Observer 模式 ...	69	5.2 集合 .....	107
3.8.1 设计思想概述 .....	69	5.2.1 理解集合 .....	107
3.8.2 Observer 模式的接口定义 ...	70	5.2.2 创建集合类 .....	108
3.8.3 Observer 模式的实现 .....	72	5.2.3 实现 IEnumerable<T> 接口 .....	113
3.8.4 推模式和拉模式 .....	74	5.3 LINQ 查询 .....	115
3.8.5 推模式和拉模式的区别 ...	79	5.3.1 LINQ to Objects .....	115
3.9 本章小结 .....	79	5.3.2 查询表达式 .....	116
<b>第 4 章 对象的筛选和排序</b> .....	80	5.3.3 延迟加载 .....	117
4.1 对象的筛选 .....	80	5.3.4 混合使用 LINQ to Objects .....	120
4.1.1 基于拼装 SQL 的筛选 ...	80	5.4 LINQ 查询运算符 .....	121
4.1.2 基于对象的筛选 .....	84	5.4.1 返回 IEnumerable<T> ...	121
4.1.3 事件探查器 .....	89	5.4.2 返回其他序列类型 ...	127
4.2 对象的排序 .....	89	5.4.3 返回序列中元素 .....	127
4.2.1 简单排序——对固定 属性的默认排序 .....	90	5.4.4 返回标量值 .....	129
4.2.2 高级排序——多个属性 组合排序 .....	92	5.4.5 其他方法 .....	130
4.2.3 页面调用 .....	96	5.5 本章小结 .....	130
		<b>第二部分 .NET 框架</b>	
		<b>第 6 章 认识 .NET 平台</b> .....	132
		6.1 引子 .....	132

6.2	CIL——公共中间语言	133	8.2.1	关于流的类比	171
6.3	BCL 和 FCL	136	8.2.2	使用流进行文件复制	172
6.3.1	BCL——基类库	136	8.2.3	流的类型体系	175
6.3.2	FCL——框架类库	140	8.3	序列化	180
6.4	CTS——公共类型系统	140	8.3.1	基本操作	180
6.5	CLS——公共语言规范	141	8.3.2	事件响应	183
6.6	CLR——公共语言运行时	143	8.3.3	自定义序列化过程	186
6.6.1	程序集概述	143	8.4	本章小结	188
6.6.2	运行程序集	145			
6.7	CLI——公共语言基础	147			
6.8	本章小结	147			
<b>第 7 章</b>	<b>程序集</b>	<b>148</b>	<b>第 9 章</b>	<b>.NET 中的加密和解密</b>	<b>189</b>
7.1	程序集详探	148	9.1	加密和解密的相关概念	189
7.1.1	程序集模块	148	9.1.1	散列运算	190
7.1.2	清单和元数据	151	9.1.2	对称加密	191
7.1.3	程序集资源	153	9.1.3	非对称加密	192
7.2	强名称程序集	159	9.1.4	数字签名	194
7.2.1	非强名称程序集的问题	159	9.1.5	综合实现	195
7.2.2	强名称的定义	161	9.1.6	证书机制	195
7.2.3	为程序集赋予强名称	162	9.2	.NET 对加密和解密的支持	196
7.2.4	防篡改和数字签名	163	9.2.1	散列运算	196
7.2.5	全局程序集缓存	165	9.2.2	对称加密和解密	198
7.2.6	延迟签名	167	9.2.3	非对称加密	201
7.3	本章小结	168	9.3	本章小结	206
<b>第 8 章</b>	<b>流和序列化</b>	<b>169</b>	<b>第 10 章</b>	<b>网络编程</b>	<b>207</b>
8.1	文件	169	10.1	网络编程基本概念	207
8.1.1	不同视角下的文件	169	10.1.1	面向连接的传输 协议——TCP	207
8.1.2	位、字节和字节数组	170	10.1.2	即时通信程序的三种 模式	209
8.2	流	171	10.2	基本操作	211
			10.2.1	服务端对端口进行侦听	211



11.6.3	服务端、客户端会话模型 .....	286	13.2.1	使用 .NET 内置节点和 .NET 内置处理程序 ...	316
11.6.4	宿主应用程序 .....	286	13.2.2	使用自定义节点和 .NET 内置处理程序 ...	318
11.6.5	程序运行测试 .....	288	13.2.3	使用自定义节点和自定义处理程序 .....	321
11.7	本章小结 .....	289	13.2.4	“存储”类型实例 .....	329
<b>第 12 章 在 .NET 中操作 XML .....</b>		<b>290</b>	13.2.5	统一节点配置管理 .....	335
12.1	XML 概述 .....	290	13.3	本章小结 .....	338
12.1.1	为什么要有 XML .....	290	<b>第 14 章 基于角色的安全性 .....</b>		
12.1.2	XML 文档结构 .....	292	14.1	概述 .....	339
12.1.3	XML 的处理模型 .....	296	14.2	在 ASP.NET 中使用基于角色的安全性 .....	341
12.1.4	XML 验证——XSD、DTD 和 XDR .....	297	14.3	开始前的准备 .....	341
12.1.5	XML 格式转换——XSLT .....	299	14.3.1	创建页面，配置 Web.config .....	341
12.1.6	XML 选择器——XPath .....	299	14.3.2	创建用户数据表和数据访问 .....	342
12.2	操作 XML .....	300	14.4	用户登录——为 Identity 添加用户数据 .....	344
12.2.1	节点类型 .....	300	14.4.1	Login.aspx 页面实现 ...	344
12.2.2	使用 XmlReader 和 XmlWriter .....	302	14.4.2	Default.aspx 页面预览 ...	346
12.2.3	使用 XmlDocument 和 XPath .....	306	14.5	自定义 IPincipal 和 Identity ...	347
12.2.4	使用 XSD 验证 XML ...	309	14.6	自定义类型携带用户数据 .....	350
12.2.5	使用 XSLT 对 XML 进行转换 .....	311	14.7	本章小结 .....	353
12.3	本章小结 .....	312	<b>第 15 章 .NET 中的反射 .....</b>		
<b>第 13 章 .NET 应用程序配置 .....</b>		<b>314</b>	15.1	反射初步 .....	354
13.1	.NET 中的程序配置介绍 .....	314	15.1.1	建表及其问题 .....	354
13.2	.NET 应用程序配置方法 .....	316	15.1.2	数组及其问题 .....	355

15.1.3	枚举及其问题	356	15.6	动态创建对象	381
15.1.4	使用反射遍历枚举字段	357	15.6.1	使用无参数构造函数 创建对象	381
15.1.5	使用泛型来达到代码 重用	359	15.6.2	使用有参数构造函数 创建对象	382
15.1.6	.NET 中反射的一个 示例	360	15.7	动态调用方法	382
15.2	Type 类	362	15.7.1	使用 InvokeMember 调用方法	383
15.2.1	反射的作用	362	15.7.2	使用 MethodInfo.Invoke 调用方法	384
15.2.2	获取 Type 对象实例	362	15.8	示例: 遍历 System.Drawing. Color 结构	385
15.2.3	Type 类型及 System. Reflection 命名空间的 组织结构	362	15.9	本章小结	388
15.3	反射程序集	365	<b>第 16 章 多线程</b>		389
15.4	反射基本类型	367	16.1	线程的概念	389
15.4.1	获取基本信息	367	16.2	线程的基本操作	392
15.4.2	成员信息与 MemberInfo 类型	369	16.2.1	创建新线程	392
15.4.3	字段信息与 FieldInfo 类型	371	16.2.2	查看当前线程	394
15.4.4	属性信息与 PropertyInfo 类型	372	16.2.3	Sleep() 方法	394
15.4.5	方法信息与 MethodInfo 类型	373	16.2.4	Interrupt() 方法	395
15.4.6	ConstructorInfo 类型 和 EventInfo 类型	373	16.2.5	前台线程和后台线程	396
15.5	反射特性	373	16.2.6	Join() 方法	397
15.5.1	.NET 内置特性介绍	374	16.2.7	Suspend() 和 Resume() 方法	398
15.5.2	自定义特性 (Custom Attributes)	375	16.2.8	线程异常	400
15.5.3	使用反射查看自定义 特性	380	16.2.9	Abort() 方法	401
			16.3	线程同步	402
			16.3.1	使用 Monitor	403
			16.3.2	使用 WaitHandle	413
			16.4	本章小结	421

<b>第 17 章 对象生存期与垃圾收集</b> ...	422	17.3 对象析构	428
17.1 基础概念回顾	422	17.3.1 Finalizer 析构器	428
17.2 垃圾回收机制	424	17.3.2 Dispose() 和 Finalize()	429
17.2.1 判断哪些对象需要进行回收	424	17.3.3 结合析构器函数和 Dispose()	431
17.2.2 对象如何分配在堆上	425	17.4 本章小结	432
17.2.3 垃圾回收的执行过程	426		

## 第一部分

# C# 语言基础

- 第 1 章 C# 类型基础
- 第 2 章 C# 中的泛型
- 第 3 章 C# 中的委托和事件
- 第 4 章 对象的筛选和排序
- 第 5 章 LINQ



## 第 1 章

# C# 类型基础

学习任何一门语言，都需要搞清楚语言的类型系统，C# 也是一样，本章是全书的第一章，将详细介绍 C# 中的两种类型：值类型和引用类型，以及和它们所有关的一些其他概念。

### 1.1 值类型和引用类型

C# 中的类型一共分为两类，一类是值类型（Value Type），一类是引用类型（Reference Type）。值类型和引用类型是以它们在计算机内存中是如何被分配的来划分的。值类型包括了结构和枚举，引用类型则包括了类、接口、委托等。还有一种特殊的值类型，称为简单类型（Simple Type），比如 byte, int 等，这些简单类型实际上是 BCL 基类库类型的别名。比如，声明一个 int 类型，实际上是声明一个 System.Int32 结构类型。因此，在 Int32 类型中定义的方法或属性，都可以在 int 类型上调用，比如“123.Equals(2)”。

所有的值类型都隐式地继承自 System.ValueType 类型（注意 System.ValueType 本身是一个类类型）。之所以说是“隐式地”，是因为在 C# 代码中，是看不到这个继承关系的，这个关系只有通过 MSIL 代码才可以看到。System.ValueType 类型和所有的引用类型都继承自 System.Object 基类。

C# 不支持多重继承，因为结构已经隐式地继承自 ValueType，所以结构不支持继承。

#### 说明

栈（stack）是一种后进先出的数据结构，在内存中，变量会被分配在栈上来进行操作。堆（heap）是用于为引用类型的实例（对象）分配空间的内存区域，在堆上创建一个对象，会将对象的地址传给栈上的变量（反过来叫变量指向此对象，或者变量引用此对象）。

#### 1.1.1 值类型

现在我们更详细地看一下值类型。当声明一个值类型的变量（Variable）的时候，变量本