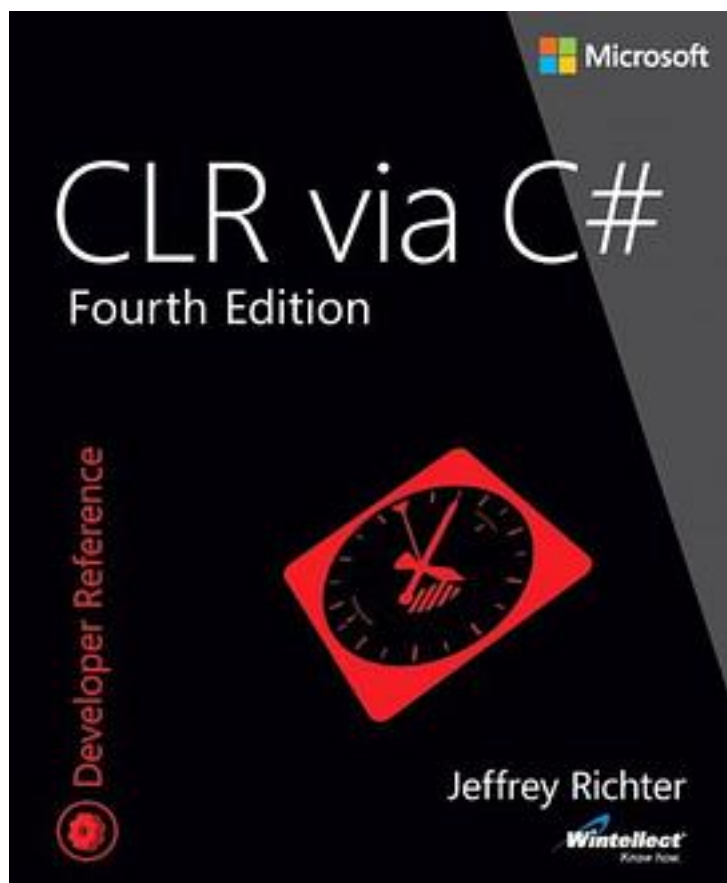


CLR via C#



[CLR via C# 下载链接1](#)

著者:(美)Jeffrey Richter

出版者:清华大学出版社

出版时间:2010-9

装帧:

isbn:9787302232599

本书针对CLR和.NET Framework 4.0进行深入、全面的探讨，并结合实例介绍了如何利用它们进行设计、开发和调试。全书5部分29章。第Ⅰ部分介绍CLR基础，第Ⅱ部分解释如何设计类型，第Ⅲ部分介绍基本类型，第Ⅳ部分以实用特性为主题，第Ⅴ部分花大量篇幅重点介绍线程处理。

通过本书的阅读，读者可以掌握CLR和.NET Framework的精髓，轻松、高效地创建高性能应用程序。

作者介绍:

Jeffrey

Richter是一位在全球享有盛誉的技术作家，尤其在Windows/.NET领域有着杰出的贡献。他的第一本Windows著作Windows 3: A Developer's Guide大获好评，从而声名远扬。之后，他又推出了经典著作《Windows高级编程指南》和《Windows核心编程》。如今这两本书早已成为Windows程序设计领域的颠峰之作，培育了几代软件开发设计人员。他的每一本新作问世，我们都有理由相信这是一本巨著，我们想要的一切尽在其中。Jeffery是Wintellect公司的创始人之一，也是MSDN杂志.NET专栏的特邀编辑。

周靖 从1983年接触苹果机开始，

“浸染”IT业数十年，涉猎广泛，具有深厚的技术功底和良好的文学素养。早期痴迷于硬件，曾担任《微型计算机》杂志特约作者多年。继而潜心钻研编程，并乐在其中积累了丰富的经验。其翻译风格严谨、准确、朴实、流畅，深受读者欢迎。代表译著有《c++面向对象程序设计》(Walter Savitch著)、《Windows核心编程》(Jeffrey Richter著)和《C++图形与游戏编程基础》(Tony Gaddis著)等。业余时间，听音乐和看电影是他的爱好。

目录: 第I部分 CLR基础

第1章 CLR的执行模型 3

1.1 将源代码编译成托管模块 3

1.2 将托管模块合并成程序集 6

1.3 加载公共语言运行时 8

1.4 执行程序集的代码 10

1.4.1 IL和验证 15

1.4.2 不安全的代码 16

1.5 本地代码生成器: NGen.exe 18

1.6 Framework类库 20

1.7 通用类型系统 22

1.8 公共语言规范 24

1.9 与非托管代码的互操作性 28

第2章 生成、打包、部署和管理应用程序及类型 29

2.1 .NET Framework部署目标 29

2.2 将类型生成到模块中 31 响应文件 32

2.3 元数据概述 34

2.4 将模块合并成程序集 39

2.4.1 使用Visual Studio IDE将程序集添加到项目中 45

2.4.2 使用程序集链接器 46

2.4.3 为程序集添加资源文件 48

2.5 程序集版本资源信息 49

2.6 语言文化 53

2.7 简单应用程序部署(私有部署的程序集) 54

2.8 简单管理控制(配置) 55

第3章 共享程序集和强命名程序集 59

3.1 两种程序集，两种部署 60

3.2 为程序集分配强名称 61

3.3 全局程序集缓存 65

3.4 在生成的程序集中引用一个强命名程序集 67

- 3.5 强命名程序集能防范篡改 69
- 3.6 延迟签名 70
- 3.7 私有部署强命名程序集 72
- 3.8 “运行时”如何解析类型引用 73
- 3.9 高级管理控制(配置) 76发布者策略控制 78
- 第II部分 设计类型
- 第4章 类型基础 83
 - 4.1 所有类型都从System.Object派生 83
 - 4.2 类型转换 85
 - 4.3 命名空间和程序集 89
 - 4.4 运行时的相互联系 92
- 第5章 基元类型、引用类型和值类型 101
 - 5.1 编程语言的基元类型 101
 - 5.2 引用类型和值类型 108
 - 5.3 值类型的装箱和拆箱 113
 - 5.3.1 使用接口更改已装箱值类型中的字段(以及为什么不应该这样做) 124
 - 5.3.2 对象相等性和同一性 127
 - 5.4 对象哈希码 129
 - 5.5 dynamic基元类型 131
- 第6章 类型和成员基础 137
 - 6.1 类型的各种成员 137
 - 6.2 类型的可见性 140友元程序集 140
 - 6.3 成员的可访问性 142
 - 6.4 静态类 143
 - 6.5 分部类、结构和接口 145
 - 6.6 组件、多态和版本控制 146
 - 6.6.1 CLR如何调用虚方法、属性和事件 148
 - 6.6.2 合理使用类型的可见性和成员的可访问性 151
 - 6.6.3 对类型进行版本控制时的虚方法的处理 154
- 第7章 常量和字段 159
 - 7.1 常量 159
 - 7.2 字段 160
- 第8章 方法 165
 - 8.1 实例构造器和类(引用类型) 165
 - 8.2 实例构造器和结构(值类型) 168
 - 8.3 类型构造器 171
 - 8.4 操作符重载方法 176
 - 8.5 转换操作符方法 179
 - 8.6 扩展方法 182
 - 8.6.1 规则和原则 184
 - 8.6.2 用扩展方法扩展各种类型 185
 - 8.6.3 ExtensionAttribute类 187
 - 8.7 分部方法 188
- 第9章 参数 191
 - 9.1 可选参数和命名参数 191
 - 9.1.1 规则和原则 192
 - 9.1.2 DefaultParameterValueAttribute和OptionalAttribute 194
 - 9.2 隐式类型的局部变量 194
 - 9.3 以传引用的方式向方法传递参数 196
 - 9.4 向方法传递可变数量的参数 201
 - 9.5 参数和返回类型的指导原则 203
 - 9.6 常量性 205
- 第10章 属性 207
 - 10.1 无参属性 207

- 10.1.1 自动实现的属性 210
- 10.1.2 合理定义属性 211
- 10.1.3 对象和集合初始化器 214
- 10.1.4 匿名类型 215
- 10.1.5 System.Tuple类型 218
- 10.2 有参属性 220
- 10.3 调用属性访问器方法时的性能 225
- 10.4 属性访问器的可访问性 225
- 10.5 泛型属性访问器方法 225
- 第11章 事件 227
 - 11.1 设计要公开事件的类型 228
 - 11.1.1 第一步：定义类型来容纳所有需要发送给事件通知接收者的附加信息 229
 - 11.1.2 第二步：定义事件成员 229
 - 11.1.3 第三步：定义负责引发事件的方法来通知事件的登记对象 231
 - 11.1.4 第四步：定义方法将输入转化为期望事件 233
 - 11.2 编译器如何实现事件 233
 - 11.3 设计侦听事件的类型 235
 - 11.4 显式实现事件 237
- 第12章 泛型 241
 - 12.1 Framework类库中的泛型 245
 - 12.2 Wintellect的Power Collections库 246
 - 12.3 泛型基础结构 247
 - 12.3.1 开放类型和封闭类型 247
 - 12.3.2 泛型类型和继承 249
 - 12.3.3 泛型类型同一性 251
 - 12.3.4 代码爆炸 252
 - 12.4 泛型接口 252
 - 12.5 泛型委托 253
 - 12.6 委托和接口的逆变和协变泛型类型实参 254
 - 12.7 泛型方法 256
 - 12.8 泛型和其他成员 258
 - 12.9 可验证性和约束 259
 - 12.9.1 主要约束 261
 - 12.9.2 次要约束 262
 - 12.9.3 构造器约束 263
 - 12.9.4 其他可验证性问题 264
- 第III部分 基本类型
- 第13章 接口 267
 - 13.1 类和接口继承 267
 - 13.2 定义接口 268
 - 13.3 继承接口 269
 - 13.4 关于调用接口方法的更多探讨 271
 - 13.5 隐式和显式接口方法实现(幕后发生的事情) 272
 - 13.6 泛型接口 274
 - 13.7 泛型和接口约束 276
 - 13.8 实现多个具有相同方法名和签名的接口 277
 - 13.9 用显式接口方法实现来增强编译时类型安全性 278
 - 13.10 谨慎使用显式接口方法实现 280
 - 13.11 设计：基类还是接口 282
- 第14章 字符、字符串和文本处理 287
 - 14.1 字符 287
 - 14.2 System.String类型 290
 - 14.2.1 构造字符串 290
 - 14.2.2 字符串是不可变的 292

- 14.2.3 比较字符串 293
- 14.2.4 字符串留用 298
- 14.2.5 字符串池 301
- 14.2.6 检查字符串中的字符和文本元素 301
- 14.2.7 其他字符串操作 303
- 14.3 高效率构造字符串 304
 - 14.3.1 构造StringBuilder对象 304
 - 14.3.2 StringBuilder的成员 305
- 14.4 获取对象的字符串表示：ToString 307
 - 14.4.1 指定具体的格式和语言文化 308
 - 14.4.2 将多个对象格式成一个字符串 311
 - 14.4.3 提供定制格式化器 313
- 14.5 解析字符串来获取对象：Parse 315
- 14.6 编码：字符和字节的相互转换 317
 - 14.6.1 字符和字节流的编码和解码 322
 - 14.6.2 Base-64字符串编码和解码 323
- 14.7 安全字符串 324
- 第15章 枚举类型和位标志 327
 - 15.1 枚举类型 327
 - 15.2 位标志 332
 - 15.3 向枚举类型添加方法 335
- 第16章 数组 337
 - 16.1 初始化数组元素 339
 - 16.2 数组转型 341
 - 16.3 所有数组都隐式派生自System.Array 343
 - 16.4 所有数组都隐式实现IEnumerable，ICollection和IList 344
 - 16.5 数组的传递和返回 345
 - 16.6 创建下限非零的数组 346
 - 16.7 数组的访问性能 347
 - 16.8 不安全的数组访问和固定大小的数组 351
- 第17章 委托 353
 - 17.1 初识委托 353
 - 17.2 用委托回调静态方法 355
 - 17.3 用委托回调实例方法 357
 - 17.4 委托揭秘 357
 - 17.5 用委托回调许多方法(委托链) 361
 - 17.5.1 C#对委托链的支持 365
 - 17.5.2 取得对委托链调用的更多控制 365
 - 17.6 委托定义太多(泛型委托) 368
 - 17.7 C#为委托提供的简化语法 369
 - 17.7.1 简化语法1：不需要构造委托对象 369
 - 17.7.2 简化语法2：不需要定义回调方法 370
 - 17.7.3 简化语法3：局部变量不需要手动包装到类中即可传给回调方法 373
 - 17.8 委托和反射 375
- 第18章 定制attribute 379
 - 18.1 使用定制attribute 379
 - 18.2 定义自己的attribute类 382
 - 18.3 attribute的构造器和字段/属性的数据类型 386
 - 18.4 检测定制attribute 387
 - 18.5 两个attribute实例的相互匹配 391
 - 18.6 检测定制attribute时不创建从Attribute派生的对象 393
 - 18.7 条件attribute类 396
- 第19章 可空值类型 399
 - 19.1 C#对可空值类型的支持 401

- 19.2 C#的空接合操作符 403
- 19.3 CLR对可空值类型的特殊支持 404
 - 19.3.1 可空值类型的装箱 404
 - 19.3.2 可空值类型的拆箱 405
 - 19.3.3 通过可空值类型调用GetType 405
 - 19.3.4 通过可空值类型调用接口方法 405
- 第IV部分 核心机制
- 第20章 异常和状态管理 409
 - 20.1 定义“异常” 409
 - 20.2 异常处理机制 411
 - 20.2.1 try块 412
 - 20.2.2 catch块 412
 - 20.2.3 finally块 414
 - 20.3 System.Exception类 417
 - 20.4 FCL定义的异常类 420
 - 20.5 抛出异常 422
 - 20.6 定义自己的异常类 423
 - 20.7 用可靠性换取开发效率 425
 - 20.8 指导原则和最佳实践 433
 - 20.8.1 善用finally块 433
 - 20.8.2 不要什么都捕捉 435
 - 20.8.3 得体地从异常中恢复 436
 - 20.8.4 发生不可恢复的异常时回滚部分完成的操作——维持状态 436
 - 20.8.5 隐藏实现细节来维系契约 437
 - 20.9 未处理的异常 440
 - 20.10 对异常进行调试 444
 - 20.11 异常处理的性能问题 446
 - 20.12 约束执行区域(CER) 448
 - 20.13 代码契约 451
- 第21章 自动内存管理(垃圾回收) 459
 - 21.1 理解垃圾回收平台的基本工作原理 459
 - 21.2 垃圾回收算法 463
 - 21.3 垃圾回收与调试 466
 - 21.4 使用终结操作来释放本地资源 469
 - 21.4.1 使用CriticalFinalizerObject类型确保终结 470
 - 21.4.2 SafeHandle类型及其派生类型 471
 - 21.4.3 使用SafeHandle类型与非托管代码进行互操作 473
 - 21.5 对托管资源使用终结操作 475
 - 21.6 什么会导致Finalize方法被调用 477
 - 21.7 终结操作揭秘 478
 - 21.8 Dispose模式：强制对象清理资源 481
 - 21.9 使用实现了Dispose模式的类型 485
 - 21.10 C#的using语句 488
 - 21.11 一个有趣的依赖性问题 490
 - 21.12 手动监视和控制对象的生存期 491
 - 21.13 对象复活 501
 - 21.14 代 503
 - 21.15 用于本地资源的其他垃圾回收功能 508
 - 21.16 预测需求大量内存的操作能否成功 512
 - 21.17 编程控制垃圾回收器 513
 - 21.18 线程劫持 516
 - 21.19 垃圾回收模式 517
 - 21.20 大对象 520
 - 21.21 监视垃圾回收 520

- 第22章 CLR寄宿和AppDomain 523
 - 22.1 CLR寄宿 523
 - 22.2 AppDomain 526
 - 22.3 卸载AppDomain 538
 - 22.4 监视AppDomain 540
 - 22.5 AppDomain FirstChance异常通知 541
 - 22.6 宿主如何使用AppDomain 541
 - 22.6.1 可执行应用程序 542
 - 22.6.2 Microsoft Silverlight富Internet应用程序 542
 - 22.6.3 Microsoft ASP.NET Web窗体和XML Web服务应用程序 542
 - 22.6.4 Microsoft SQL Server 543
 - 22.6.5 更多的用法只局限于你自己的想象力 543
 - 22.7 高级宿主控制 544
 - 22.7.1 使用托管代码管理CLR 544
 - 22.7.2 编写健壮的宿主应用程序 544
 - 22.7.3 宿主如何拿回它的线程 546
- 第23章 程序集加载和反射 549
 - 23.1 程序集加载 549
 - 23.2 使用反射构建动态可扩展应用程序 554
 - 23.3 反射的性能 555
 - 23.3.1 发现程序集中定义的类型 556
 - 23.3.2 类型对象的准确含义 556
 - 23.3.3 构建Exception派生类型的一个层次结构 558
 - 23.3.4 构造类型的实例 560
 - 23.4 设计支持加载项的应用程序 562
 - 23.5 使用反射发现类型的成员 564
 - 23.5.1 发现类型成员 565
 - 23.5.2 BindingFlags: 筛选返回的成员种类 569
 - 23.5.3 发现类型的接口 570
 - 23.5.4 调用类型的成员 571
 - 23.5.5 一次绑定, 多次调用 575
 - 23.5.6 使用绑定句柄来减少进程的内存耗用 581
- 第24章 运行时序列化 585
 - 24.1 序列化/反序列化快速入门 586
 - 24.2 使类型可序列化 590
 - 24.3 控制序列化和反序列化 592
 - 24.4 格式化器如何序列化类型实例 595
 - 24.5 控制序列化/反序列化的数据 597
 - 24.6 流上下文 603
 - 24.7 将类型序列化为不同的类型以及将对象反序列化为不同的对象 604
 - 24.8 序列化代理 606
 - 代理选择器链 609
 - 24.9 反序列化对象时重写程序集和/或类型 610
- 第 V 部分 线程处理
- 第25章 线程基础 615
 - 25.1 Windows为什么要支持线程 615
 - 25.2 线程开销 616
 - 25.3 停止疯狂 620
 - 25.4 CPU发展趋势 622
 - 25.5 NUMA架构的机器 623
 - 25.6 CLR线程和Windows线程 625
 - 25.7 使用专用线程执行异步的计算限制操作 625
 - 25.8 使用线程的理由 627
 - 25.9 线程调度和优先级 629

- 25.10 前台线程和后台线程 634
- 25.11 继续学习 635
- 第26章 计算限制的异步操作 637
 - 26.1 CLR线程池基础 638
 - 26.2 执行简单的计算限制操作 639
 - 26.3 执行上下文 640
 - 26.4 协作式取消 642
 - 26.5 任务 645
 - 26.5.1 等待任务完成并获取它的结果 646
 - 26.5.2 取消任务 648
 - 26.5.3 一个任务完成时自动启动一个新任务 649
 - 26.5.4 任务可以启动子任务 651
 - 26.5.5 任务内部揭秘 652
 - 26.5.6 任务工厂 653
 - 26.5.7 任务调度器 655
 - 26.6 Parallel的静态For, ForEach和Invoke方法 657
 - 26.7 并行语言集成查询(PLINQ) 660
 - 26.8 执行定时计算限制操作 663
 - 26.9 线程池如何管理线程 665
 - 26.9.1 设置线程池限制 665
 - 26.9.2 如何管理工作线程 666
 - 26.10 缓存线和伪共享 667
- 第27章 I/O限制的异步操作 671
 - 27.1 Windows如何执行I/O操作 671
 - 27.2 CLR的异步编程模型(APM) 675
 - 27.3 AsyncEnumerator类 679
 - 27.4 APM和异常 682
 - 27.5 应用程序及其线程处理模型 683
 - 27.6 异步实现服务器 687
 - 27.7 APM和计算限制的操作 687
 - 27.8 APM的注意事项 689
 - 27.8.1 在没有线程池的前提下使用APM 689
 - 27.8.2 总是调用EndXxx方法, 而且只调用一次 690
 - 27.8.3 调用EndXxx方法时总是使用相同的对象 690
 - 27.8.4 为BeginXxx和EndXxx方法使用ref, out和params实参 691
 - 27.8.5 不能取消异步I/O限制操作 691
 - 27.8.6 内存消耗 691
 - 27.8.7 有的I/O操作必须同步完成 691
 - 27.8.8 FileStream特有的问题 692
 - 27.9 I/O请求优先级 693
 - 27.10 将IAsyncResult APM转换为Task 695
 - 27.11 基于事件的异步模式 696
 - 27.11.1 将EAP转换为Task 698
 - 27.11.2 APM和EAP的对比 699
 - 27.12 编程模型的泥沼 700
 - 第28章 基元线程同步构造 703
 - 28.1 类库和线程安全 705
 - 28.2 基元用户模式和内核模式构造 706
 - 28.3 用户模式构造 707
 - 28.3.1 易失构造 708
 - 28.3.2 互锁构造 713
 - 28.3.3 实现简单的Spin Lock 717
 - 28.3.4 Interlocked Anything模式 720
 - 28.4 内核模式构造 722

28.4.1 Event构造 725
28.4.2 Semaphore构造 727
28.4.3 Mutex构造 728
28.4.4 在一个内核构造可用时调用一个方法 730
第29章 混合线程同步构造 733
29.1 一个简单的混合锁 733
29.2 自旋、线程所有权和递归 735
29.3 混合构造的大杂烩 737
29.3.1 ManualResetEventSlim类和SemaphoreSlim类 737
29.3.2 Monitor类和同步块 738
29.3.3 ReaderWriterLockSlim类 743
29.3.4 OneManyLock类 745
29.3.5 CountdownEvent类 747
29.3.6 Barrier类 747
29.3.7 线程同步构造小结 748
29.4 著名的双检锁技术 750
29.5 条件变量模式 754
29.6 用集合防止占有锁太长的时间 756
29.7 并发集合类 760
• • • • • ([收起](#))

[CLR via C# 下载链接1](#)

标签

C

#CLR

.Net

编程

软件开发

.NET

计算机

微软

评论

啥也不说了。老婆的序写的很亮。。

20170314~0808 粗读完第一遍。主要讲了 C# 基础语法、类型系统、GC、异步编程（Task、APM、计算限制、I/O 限制）、线程同步构造（用户模式、内核模式）。周老师翻译得也好！PS：还是觉得并发编程比较难。啥时候再细读一遍呢？

已略读。

弃

这是一本好书,所有想进阶的C#程序员必看的书.

net圣经第3版

刚发现买错版本了，已经有了第三版，内容没的说，作者就是一块金字招牌

C# .net的经典中级书

CLR & C# 全面深入剖析，C#深入必备利器。。。

C#的葵花宝典，没啥好说的。

翻译的不咋地

花了我一个月的时间看完，不过现在看起来都没啥激情了。想当年看第二版的时候，哎
~~~

-----  
c# clr 方面经典书籍

-----  
先有语言基础，才能谈得上设计模式等。用这本书去夯实语言基础！

-----  
见到了Jeffrey，无憾。

-----  
好书啊，讲的深入全面，翻译的也好。

-----  
经常YY，觉得自己如果认真、深入的学习，也能写成这样就好了。看技术还是不能从头看到尾，累，也把兴趣看没了，呵呵，学习自学中。

-----  
讲得很完备，使用C#的必读

-----  
算是C#必读书目。搁了好几年，趁着做Unity3D再精进一下。

-----  
开始看

-----  
[CLR via C# 下载链接1](#)

## 书评

.NET技术领域有两位世界级专家。一位是Don Box。他以《Essential COM》确立了自己COM专家的地位，在.NET时代，Don Box又以《Essential .NET》（Volume I The Common Language Runtime）确立了自己.NET专家的地位。2002年，Microsoft将其招致麾下，成为.NET Architect，与Ande...

-----  
最近工作比较闲，我把这本书看了两遍。  
第一遍是从17年3月份开始，断断续续直到上周读完，翻到最后一页的时候，心里如释重负，终于看完了，虽然这本书真的很厉害，但是看这么厚的书，真的很煎熬啊！然而，过了半天，我就忘记讲了什么，只记得委托是一个类（因为我会拿来跟同事...

-----  
看点学业相关大体很有必要...虽然是多线程逼得不行了才下狠心看了看，结果到最后一章混合锁实在翻不下去，草草了事了。  
作者不愧是搞内核的大牛，讲起原理来头头是道，作为.net必读书目，恰如其分，不过如果不是专业程序员，大约看起来颇为鸡肋。要看这本书，大约还是需要点基...

-----  
第一次看感觉好难，但硬着头皮看下去后，感觉又不是很难了，从这本书可以学习到更多的知识，用Keni的话，这是一本入门的书，因为读完后，你会有很多的疑问，带着这些疑问，你会不断的去研究

-----  
本书的作者对C#的设计理念和设计细节了解透彻，他不仅教会你C#语言的各种细节，同时还把发生在背后的各种原理娓娓道来，细读本书之后，你学到的不仅仅是C#语言，同时还能够洞悉C#的本质，让你不仅仅是写出可以执行的代码，而是写出可以高效运行的代码。本书的译者不仅是一位...

-----  
哦读过第二版的，刚刚从图书馆借出了第三版。力荐～～！！  
没对过这本书很难对C#有深入的了解的。 力荐～～

-----  
在我的博客更新了本书的勘误：  
<http://transbot.blog.163.com/blog/static/5421396120086107913329/>

-----  
What's new in CLR via C#, 4th Edition as compared to the 3rd Edition  
与第三版相比，《CLR via C#》第四版有什么新内容？ The 4th edition of my CLR via C# book is coming out next month and you can order it here. The new edition is updated for Visual Studio 20...

-----  
书很好，很多书中一两句带过的地方，其实能引出很多东西来，所以还是有些底子再看的好。感觉最好是做工具书用，当研究某个问题时沿着问题一路追过来，这样比较有效率。可惜翻译有点差，我在看appdomain的相关内容的时候差点被误导。。。所以，建议对照着英文看比较好。。。...

-----  
以前一直以为是给初学者入门看的书，一直也就没买。直到最近一个朋友推荐，于是去买了本仔细的读了一下，确实是好书，很多以前不是很清晰的地方突然就开朗了。

-----  
不明白为什么书商不将翻译版和影印版一起推出？  
在电子版和影印版久等不到的情况下，只能拿翻译版读了，翻译质量不能说太差（我见过比这更差的！），聊胜于无罢了。等到后来有了电子版，翻译版的作用就小多了。再后来，又在书店里看到了影印版，但太迟了……如果《CLR via...

-----  
了解了很多CLR的内部机制，Boxing,GC,Generic等等  
看了以后觉得自己可以写出更优美，效率更高的代码来：)

-----  
读这本书重温java。  
1，event和method并列，解决了初学者容易写出紧耦合代码的问题，减轻了编写各种listener的痛苦。  
2，对const关键字的理解和java一致，不支持，它带来的麻烦超过了哪一点约束附加的

好处。 3，相对java增加了ref， struct等， property， operator overload， d...

-----  
我先看了下电子版的.. 后来我毅然决定在amazon.com上面买了这本..原版. 加邮费  
一共大概340多RMB啊.....但是我都觉得值得

-----  
[CLR via C#\\_下载链接1](#)