

<<深入解析C#>>

图书基本信息

书名：<<深入解析C#>>

13位ISBN编号：9787115226488

10位ISBN编号：7115226482

出版时间：201005

出版单位：人民邮电出版社

作者：Jon Skeet

页数：333

译者：周靖,朱永光等

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<深入解析C#>>

前言

世上有两类钢琴家。

一类钢琴家弹琴并不是因为他们喜欢，而是因为他们的父母强迫他们上钢琴课。

另一类钢琴家弹琴是因为他们喜欢音乐，想创作音乐。

他们不需要被强迫，相反，他们时常忘记什么时候要停下来。

后一类人中，有人是把弹钢琴当作一种爱好。

而另一些人则是为了生活，因此更需要投入、技巧和天赋。

他们有一定的灵活性来选择弹奏时的音乐流派和风格，不过这些选择主要还是由雇主的需要或者听众的口味来决定的。

后一类人中，有人主要就是为了钱，但也有一些专业人士即便没有报酬，也愿意在公共场合弹奏钢琴。

他们喜欢运用自己的技巧和天赋为别人演奏音乐。

在这个过程中，他们能找到许多乐趣。

如果同时还有报酬，当然更是锦上添花。

后一类人中，有人是自学成材的，他们演奏乐曲是不看谱的。

这些人有极高的天赋和能力，但除非通过音乐本身，否则无法向别人传递那种直观的感受。

还有一些人无论在理论还是实践上都经过了正统的训练，他们能清楚地理解作曲家是用什么手法得到预期的情绪效果，并相应地改进自己的演绎手法。

后一类人中，有人从来没有打开钢琴看它的内部构造。

还有一些人则对钢琴的发声原理感到好奇不已，最后发现是由于杠杆装置和绞盘在音锤敲击琴弦前的瞬间，牵引制音器的擒纵器，他们为弄明白由5000~10000个运动机件组成的这个乐器装置而感到高兴和自豪。

后一类人中，有人会对自己的手艺和成就心满意足，对它们带来的心灵上的愉悦和经济上的收入感到非常满意。

但是，还有一些人不仅仅是艺术家、理论家和技师，他们会抽时间以导师的身份，将那些知识传授给其他人。

我不知道Jon Skeet是哪一类钢琴家。

但是，我与这位微软C# MVP有多年的电子邮件交流，并经常看他的博客。

我本人至少3遍逐字读完他的这本书，我清楚地知道Jon是后一种软件开发者：热情、博学、天资极高、有好奇心以及善于分析——是其他人的好老师。

C#是一种极为实用和快速发展的语言。

通过添加查询能力、更丰富的类型推断、精简的匿名函数语法，等等，一种全新风格的编程语言已出现在我们的面前。

与此同时，它代表的仍然是一种静态类型的、面向组件的开发方式，C#取得成功的立足之本没有变。

。

<<深入解析C#>>

内容概要

《深入解析C#》是针对那些想提高编程技能的C#开发人员而作。

《深入解析C#》的一个基本原则是侧重于从现象中挖掘本质，而不是简单罗列C#的每个知识点。

如果你就是喜欢探索本源的人，那么《深入解析C#》正适合你。

作者深入探索了崭新的C#2和3特性及其核心概念，并将新特性融入C#代码中，这使《深入解析C#》更独具匠心。

《深入解析C#》适合各层次.NET开发人员阅读。

<<深入解析C#>>

作者简介

作者：（美国）斯基特（Jon Skeet）译者：周靖 朱永光

书籍目录

第一部分 基础知识第1章 C#开发的进化史 1.1 实战演变：变化的代码 1.1.1 定义产品类型 1.1.2 按名称对产品进行排序 1.1.3 查询集合 1.1.4 表示未知的价格 1.1.5 LINQ和查询表达式 1.2 C#（及相关技术）简史 1.2.1 C#问世前的世界 1.2.2 C#和.NET降生 1.2.3 .NET 1.1的小幅更新和第一次重要跨越：.NET 2.0 1.2.4 “下一代”产品 1.2.5 历史回顾和争夺开发者之战 1.3 .NET平台 221.3.1 区分语言、运行时和库 1.3.2 解开版本号的谜团 1.4 采用代码段形式的全能代码 1.4.1 代码段及其扩展形式 1.4.2 Snippy介绍 1.5 小结 第2章 C# 1所搭建的核心基础 2.1 委托 2.1.1 简单委托的构成 2.1.2 合并和删除委托 2.1.3 对事件的简单讨论 2.1.4 委托小结 2.2 类型系统的特征 2.2.1 C#在类型系统世界中的位置 2.2.2 C# 1的类型系统在什么时候不够用 2.2.3 C# 1的类型系统在什么时候会碍事 2.2.4 类型系统特征总结 2.3 值类型和引用类型 2.3.1 现实世界中的值和引用 2.3.2 值类型和引用类型基础知识 2.3.3 走出误区 2.3.4 装箱和拆箱 2.3.5 值类型和引用类型小结 2.4 C# 2和3：构建于坚实基础之上的新特性 2.4.1 与委托有关的特性 2.4.2 与类型系统有关的特性 2.4.3 与值类型有关的特性 2.5 小结 第二部分 C# 2：解决C# 1的问题第3章 用泛型实现参数化类型 3.1 为什么需要泛型 3.2 日常使用的简单泛型 3.2.1 通过例子来学习：泛型字典 3.2.2 泛型类型和类型参数 3.2.3 泛型方法和判读泛型声明 3.3 深化与提高 3.3.1 类型约束 3.3.2 泛型方法类型实参的类型推断 3.3.3 实现泛型 3.4 高级泛型 3.4.1 静态字段和静态构造函数 3.4.2 JIT编译器如何处理泛型 3.4.3 泛型迭代 3.4.4 反射和泛型 3.5 .NET 2.0中的泛型集合类 3.5.1 List[T] 3.5.2 Dictionary[TKey, TValue] 3.5.3 Queue[T]和Stack[T] 3.5.4 SortedList[TKey, TValue]和SortedDictionary[TKey, TValue] 3.5.5 LinkedList[T] 3.6 泛型在C#和其他语言中的限制 3.6.1 协变性和逆变性的缺乏 3.6.2 缺乏操作符约束或者“数值”约束 3.6.3 缺乏泛型属性、索引器和其他成员类型 3.6.4 同C++模板的对比 3.6.5 和Java泛型的对比 3.7 小结 第4章 可空类型 4.1 没有值时怎么办 4.1.1 为什么值类型的变量不能是null 4.1.2 在C# 1中表示空值的模式 4.2 System.Nullable[T]和System.Nullable 4.2.1 Nullable[T]简介 4.2.2 装箱和拆箱 4.2.3 Nullable[T]实例的相等性 4.2.4 来自非泛型Nullable类的支持 4.3 C# 2为可空类型提供的语法糖 4.3.1 ?修饰符 4.3.2 使用null进行赋值和比较 4.3.3 可空转换和操作符 4.3.4 可空逻辑 4.3.5 空接合操作符 4.4 可空类型的新奇用法 4.4.1 尝试一个不使用输出参数的操作 4.4.2 空接合操作符让比较不再痛苦 4.5 小结 第5章 进入快速通道的委托 5.1 向笨拙的委托语法说拜拜 5.2 方法组转换 5.3 协变性和逆变性 5.4 使用匿名方法的内联委托操作 5.4.1 从简单的开始：处理一个参数 5.4.2 匿名方法的返回值 5.4.3 忽略委托参数 5.5 在匿名方法中捕捉变量 5.5.1 定义闭包和不同的变量类型 5.5.2 测试被捕获的变量的行为 5.5.3 捕获变量到底有什么用处 5.5.4 捕获变量的延长生存期 5.5.5 局部变量实例化 5.5.6 共享和非共享的变量混合使用 5.5.7 捕获变量的使用规则和小结 5.6 小结 第6章 实现迭代器的捷径 6.1 C# 1：手写迭代器的痛苦 6.2 C# 2：利用yield语句简化迭代器 6.2.1 迭代器块和yield return简介 6.2.2 观察迭代器的工作流程 6.2.3 进一步了解迭代器执行流程 6.3 真实的例子：迭代范围值 6.3.1 迭代时刻表中的日期 6.3.2 定义Range类的作用域 6.3.3 使用迭代器块的实现代码 6.4 使用CCR实现伪同步代码 6.5 小结 第7章 结束C# 2的讲解：最后的一些特性 7.1 分部类型 7.1.1 在多个文件中创建一个类型 7.1.2 分部类型的使用 7.1.3 C# 3独有的分部方法 7.2 静态类型 7.3 独立的取值方法/赋值方法属性访问器 7.4 命名空间别名 7.4.1 限定的命名空间别名 7.4.2 全局命名空间别名 7.4.3 外部别名 7.5 Pragma指令 7.5.1 警告pragma 7.5.2 校验和pragma 7.6 非安全代码中的固定大小的缓冲区 7.7 把内部成员暴露给选定的程序集 7.7.1 在简单情况下的友元程序集 7.7.2 为什么使用InternalsVisibleTo 7.7.3 InternalsVisibleTo和签名程序集 7.8 小结 第三部分 C# 3——革新写代码的方式第8章 用智能的编译器来防错 8.1 自动实现的属性 8.2 隐式类型的局部变量 8.2.1 用var声明局部变量 8.2.2 隐式类型的限制 8.2.3 隐式类型的优缺点 8.2.4 建议 8.3 简化的初始化 8.3.1 定义示例类型 8.3.2 设置简单属性 8.3.3 为嵌入对象设置属性 8.3.4 集合初始化列表 8.3.5 初始化特性的应用 8.4 隐式类型的数组 8.5 匿名类型 8.5.1 第一次邂逅匿名类型 8.5.2 匿名类型的成员 8.5.3 投影初始化列表 8.5.4 重点何在 8.6 小结 第9章 Lambda表达式和表达式树 9.1 作为委托的Lambda表达式 9.1.1 准备工作：Func[...]委托类型简介 9.1.2 第一次转换成Lambda表达式 9.1.3 用一个简单表达式作为主体 9.1.4 隐式类型的参数列表 9.1.5 单一参数的快捷语法 9.2 使用List[T]和事件的简单例子 9.2.1 对列表进行筛选、排序并设置其他操作 9.2.2 在事件处理程序中进行记录 9.3 表达式树 9.3.1 在程序中构建表达式树 9.3.2 将表达式树编译成委托 9.3.3 将C# Lambda表达式转换成表达式树 9.3.4 位于LINQ核心的表达式树 9.4 类型推断和重载决策发生的改变 9.4.1 改变的起因：精

<<深入解析C#>>

简泛型方法调用 9.4.2 推断匿名函数的返回类型 9.4.3 分两个阶段进行的类型推断 9.4.4 选择正确的被重载的方法 9.4.5 类型推断和重载决策 9.5 小结 第10章 扩展方法 10.1 未引入扩展方法之前的状态 10.2 扩展方法的语法 10.2.1 声明扩展方法 10.2.2 调用扩展方法 10.2.3 扩展方法是怎样被发现的 10.2.4 在空引用上调用方法 10.3 .NET 3.5中的扩展方法 10.3.1 从Enumerable开始起步 10.3.2 用Where筛选并将方法调用链接到一起 10.3.3 用Select方法和匿名类型进行投影 10.3.4 用OrderBy方法进行排序 10.3.5 涉及链接的实际例子 10.4 使用思路和原则 10.4.1 “扩展世界”和使接口更丰富 10.4.2 流畅接口 10.4.3 理智使用扩展方法 10.5 小结 第11章 查询表达式和LINQ to Object 11.1 LINQ介绍 11.1.1 这个名称中有什么 11.1.2 LINQ中的基础概念 11.1.3 定义示例数据模型 11.2 简单的开始：选择元素 11.2.1 以数据源作为开始，以选择作为结束 11.2.2 作为查询表达式基础的编译器转换 11.2.3 范围变量和重要的投影 11.2.4 Cast、OfType和显式类型的范围变量 11.3 对序列进行过滤和排序 11.3.1 使用where子句进行过滤 11.3.2 退化的查询表达式 11.3.3 使用orderby子句进行排序 11.4 let子句和透明标识符 11.4.1 用let来进行中间计算 11.5 联接 11.5.1 使用join子句的内联接 11.5.2 使用join...into子句进行分组联接 11.5.3 使用多个from子句进行交叉联接 11.6 分组和延续 11.6.1 使用group...by子句进行分组 11.6.2 查询延续 11.7 小结 第12章 超越集合的LINQ 12.1 LINQ to SQL 12.1.1 创建缺陷数据库和实体 12.1.2 用示例数据填充数据库 12.1.3 用查询表达式访问数据库 12.1.4 更新数据库 12.1.5 LINQ to SQL小结 12.2 用IQueryable和IQueryProvider进行转换 12.2.1 IQueryable[T]和相关接口的介绍 12.2.2 模拟接口实现来记录调用 12.2.3 把表达式粘合在一起：Queryable的扩展方法 12.2.4 模拟实际运行的查询提供器 12.2.5 包装IQueryable 12.3 LINQ to DataSet 12.3.1 处理非类型化数据集 12.3.2 处理类型化数据集 12.4 LINQ to XML 12.4.1 XElement和XAttribute 12.4.2 把示例缺陷数据转换为XML 12.4.3 在LINQ to XML中进行查询 12.4.4 LINQ to XML小结 12.5 超越.NET 3.5的LINQ 12.5.1 第三方LINQ 12.5.2 未来的微软LINQ技术 12.6 小结 第13章 新时代的优雅代码 13.1 语言特性 13.1.1 更重视函数化 13.1.2 静态、动态、隐式、显式或混合 13.2 把委托作为实现继承的新方式 13.3 易读的结果高于实现 13.4 并行宇宙中的生活 13.5 再见 附录A LINQ标准查询操作符

<<深入解析C#>>

章节摘录

C# 2于2005年11月作为 .NET 2.0的一部分发布,同时发布的还有Visual Studio 2005和VB8。Visual Studio作为一个IDE,变得更富有成效——特别是它终于包含了“重构”功能。

除此之外,语言和平台的大幅改进得到了大多数开发者的热烈欢迎。

为了说明世界的变化有多快以及产品实际上市要多长的时间——请注意第一次宣布C# 3是在C# 2发布之前的两个月,也就是在2005年9月的微软PDC上。

但悲哀的是,虽然貌似只需2年,一个产品就能完成“从宣布到上市”的过程,但整个行业至少要花费额外的1年或2年的时间才能广泛地采纳它。

前面已经说过,许多公司只是从 .NET 1.1迁移到了2.0。

在此,我们只能祝愿 .NET 3.0和3.5能在更短的时间里被广泛采纳(C# 3与 .NET 3.5是一起出现的,但在面向 .NET 2.0平台开发软件时,C# 3的许多特性都是可以使用的。

稍后就会讲到版本号的问题)。

.NET 2.0这么晚问世是由于它要嵌入到SQL Server 2005中。

所以,必须在健壮性和可靠性上大费一番周折,否则无法与这样的一个系统完美地配合。

这样一来,.NET代码就能直接在数据库中执行,并可以使用更丰富的、与数据更“贴近”的逻辑。从事数据库方面工作的人对此可能会持相当谨慎的态度,也只有时间才能告诉我们这个功能会被多少人接受。

但是,如果你真的需要它时,会发现它确实是一个很强大的工具。

1.2.4 “下一代”产品2006年11月(.NET 2.0发布后一年),微软公司发布了Windows Vista、Office 2007和Exchange Server 2007。

同时发布的还有 .NET 3.0,并预装到Vista上。

基于两方面的原因,这为 .NET客户端应用程序的推广提供了帮助。

首先,“不是所有计算机都安装了 .NET”这个老生常谈的反对理由可以不必理会了,可以放心地认为假如用户运行的是Vista,就能运行 .NET应用程序。

其次,WPF(Windows Presentation Foundation)是微软公司为开发者选择的富客户端平台,它现在只能在 .NET上使用。

同样,在微软公司忙活Vista和其他产品时,世界上的其他地方也在发生着革新。

轻量级框架(lightweight framework)开始获得发展的动力,而ORM(Object Relational Mapping)受到了开发者的广泛关注,其部分原因是由于高质量的免费框架(如Hibernate)。

Linq to SQL的功能远比我们目前看到的多(目前只接触了它的数据查询能力),这标志着微软公司在这个领域的动作不会再像以前那样“蜻蜓点水”了(如当年ObjectSpaces)。

相反,这标志着微软公司在这个领域迈出了非常坚定的一步。

只有时间才能告诉我们,Linq to SQL(或者它的表亲ADO .NET Entity Framework)是不是真的能使数据库访问变得简单——虽然他们肯定会这样承诺。

<<深入解析C#>>

媒体关注与评论

“如果你想学好C#。
那么一定要把本书买回家。

——Amazon.com “伪行家只能用令人费解的专业术语和抽象的比喻来解释复杂的主题。
而真正的行家能够用通俗易懂的语言和形象生动的比喻来讲解复杂的概念，本书的作者是真正的行家

。”
——Amazon.com

<<深入解析C#>>

编辑推荐

《深入解析C#》图灵程序设计丛书。

《深入解析C#》是世界级顶尖技术专家“十年磨一剑”的经典之作。

在C#和.NET业界享有极高的声誉。

与其他泛泛介绍C#的书籍不同。

《深入解析C#》深度探究C#2和3的新增特性。

并结合技术发展，引领读者深入C#的时空。

作者从语言设计的动机出发。

介绍支持这些特性的语言的核心概念。

书中将新的语言特性放在C#语言发展的背景之上。

用极富实际意义的示例，向读者展示写代码和设计解决方案的最佳方式。

同时作者将多年的C#开发经验与读者分享，读者可咀其精华、免走弯路。

使程序设计水平更上一层楼。

Jon Skeet资深C#MVP。

经验丰富的C#项目开发人员，有近10年的C#项目开发经验。

他是C#社区以及新闻组中非常活跃的技术专家。

回答了数以万计的C#和.NET相关问题。

同时他还在其个人网站上写文章来阐述C#和.NET最难理解的方面。

除《深入解析C#》外。

他还是畅销书GroovyinAction的作者。

资深C# MVP扛鼎之作、深入解析，探求本源、亚马逊网上书店全五星评价。

<<深入解析C#>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>