

<<深入理解C# (第2版)>>

图书基本信息

书名：<<深入理解C# (第2版)>>

13位ISBN编号：9787115269249

10位ISBN编号：7115269246

出版时间：2012-1

出版时间：人民邮电出版社

作者：Jon Skeet

页数：446

译者：周靖,朱永光,姚琪琳

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>

## <<深入理解C# (第2版)>>

### 内容概要

本书是C#领域不可多得的经典著作。

作者Jon

Skeet在详尽地展示C#各个知识点的同时，更注重从现象中挖掘本质。

本书深入探索了C#的核心概念和经典特性，并将这些特性融入到代码中，让读者能够真正领会到C#之“深入”与“精妙”。

在第1版的基础上，书中新增了C#

4的新特性，如动态类型、命名实参和可选参数等，这些特性将C#语言提升到了一个新的层次。

## <<深入理解C# (第2版)>>

### 作者简介

Jon Skeet Google软件工程师，微软资深C# MVP，拥有近10年的C#项目开发经验。他是C#社区和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。他还在个人网站上撰写文章，阐述C#和.NET最难理解的问题。他还著有另一本畅销书Groovy in Action。

## &lt;&lt;深入理解C# (第2版)&gt;&gt;

## 书籍目录

## 第一部分 基础知识

## 第1章 C#开发的进化史

## 1.1 从简单的数据类型开始

## 1.1.1 C# 1 中定义的产品类型

## 1.1.2 C# 2 中的强类型集合

## 1.1.3 C# 3 中自动实现的属性

## 1.1.4 C# 4 中的命名实参

## 1.2 排序和过滤

## 1.2.1 按名称对产品进行排序

## 1.2.2 查询集合

## 1.3 处理未知数据

## 1.3.1 表示未知的价格

## 1.3.2 可选参数和默认值

## 1.4 LINQ 简介

## 1.4.1 查询表达式和进程内查询

## 1.4.2 查询XML

## 1.4.3 LINQ to SQL

## 1.5 COM 和动态类型

## 1.5.1 简化COM 互操作

## 1.5.2 与动态语言互操作

## 1.6 剖析.NET 平台

## 1.6.1 C#语言

## 1.6.2 运行时

## 1.6.3 框架库

## 1.7 怎样写出超炫的代码

## 1.7.1 采用代码段形式的全能代码

## 1.7.2 教学代码不是产品代码

## 1.7.3 你的新朋友：语言规范

## 1.8 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 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

## &lt;&lt;深入理解C# (第2版)&gt;&gt;

## 2.4 C# 1 之外：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 泛型在C#和其他语言中的限制

## 3.5.1 泛型可变性的缺乏

## 3.5.2 缺乏操作符约束或者“数值”约束

## 3.5.3 缺乏泛型属性、索引器和其他成员类型

## 3.5.4 同C++模板的对比

## 3.5.5 和Java 泛型的对比

## 3.6 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和System.Nullable

## 4.2.1 Nullable简介

## 4.2.2 Nullable装箱和拆箱

## 4.2.3 Nullable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 对可空类型使用as 操作符

## 4.3.6 空合并操作符

## 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 空合并操作符让比较不再痛苦

## &lt;&lt;深入理解C# (第2版)&gt;&gt;

## 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

## 5.3.1 委托参数的逆变性

## 5.3.2 委托返回类型的协变性

## 5.3.3 不兼容的风险

## 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

## 6.2.4 具体实现中的奇特之处

## 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 迭代文件中的行

## 6.3.3 使用迭代器块和谓词对项进行延迟筛选

## 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

## &lt;&lt;深入理解C# (第2版)&gt;&gt;

- 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<T> 委托类型简介
    - 9.1.2 第一次转换成Lambda表达式
    - 9.1.3 用单一表达式作为主体
    - 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
    - 9.3.5 LINQ 之外的表达式树
  - 9.4 类型推断和重载决策发生的改变
    - 9.4.1 改变的起因：精简泛型方法调用
    - 9.4.2 推断匿名函数的返回类型

## &lt;&lt;深入理解C# (第2版)&gt;&gt;

- 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 插曲：似曾相识的Where方法
    - 10.3.4 用Select 方法和匿名类型进行投影
    - 10.3.5 用OrderBy 方法进行排序
    - 10.3.6 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 Objects
  - 11.1 LINQ 介绍
    - 11.1.1 LINQ 中的基础概念
    - 11.1.2 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
    - 11.4.2 透明标识符
  - 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 子句进行交叉联接和合并序列
  - 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 在查询表达式和点标记之间作出选择

## &lt;&lt;深入理解C# (第2版)&gt;&gt;

- 11.7.1 需要使用点标记的操作
- 11.7.2 选择点标记
- 11.7.3 选择查询表达式
- 11.8 小结
- 第12章 超越集合的LINQ
- 12.1 使用LINQ to SQL 查询数据库
  - 12.1.1 数据库和模型
  - 12.1.2 用查询表达式访问数据库
  - 12.1.3 包含联接的查询
- 12.2 用IQueryable 和IQueryProvider 进行转换
  - 12.2.1 IQueryable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 友好的API 和LINQ to XML
  - 12.3.1 LINQ to XML 中的核心类型
  - 12.3.2 声明式构造
  - 12.3.3 查询单个节点
  - 12.3.4 合并查询操作符
  - 12.3.5 与LINQ 和谐共处
- 12.4 用并行LINQ 代替LINQ to Objects
  - 12.4.1 在单线程中绘制曼德博罗特集
  - 12.4.2 ParallelEnumerable、Parallel-Query 和AsParallel
  - 12.4.3 调整并行查询
- 12.5 使用LINQ to Rx 反转查询模型
  - 12.5.1 IObservable和IObserver
  - 12.5.2 简单地开始
  - 12.5.3 查询可观察对象
  - 12.5.4 意义何在
- 12.6 扩展LINQ to Objects
  - 12.6.1 设计和实现指南
  - 12.6.2 示例扩展：选择随机元素
- 12.7 小结
- 第四部分 C# 4：良好的交互性
- 第13章 简化代码的微小修改
- 13.1 可选参数和命名实参
  - 13.1.1 可选参数
  - 13.1.2 命名实参
  - 13.1.3 两者相结合
- 13.2 改善COM 互操作性
  - 13.2.1 在C# 4 之前操纵Word是十分恐怖的
  - 13.2.2 可选参数和命名实参的复仇
  - 13.2.3 按值传递ref 参数
  - 13.2.4 调用命名索引器
  - 13.2.5 链接主互操作程序集
- 13.3 接口和委托的泛型可变性

## &lt;&lt;深入理解C# (第2版)&gt;&gt;

- 13.3.1 可变性的种类：协变性和逆变性
- 13.3.2 在接口中使用可变性
- 13.3.3 在委托中使用可变性
- 13.3.4 复杂情况
- 13.3.5 限制和说明
- 13.4 对锁和字段风格的事件的微小改变
  - 13.4.1 健壮的锁
  - 13.4.2 字段风格的事件
- 13.5 小结
- 第14章 静态语言中的动态绑定
  - 14.1 何谓，何时，为何，如何
    - 14.1.1 何谓动态类型
    - 14.1.2 动态类型什么时候有用，为什么
    - 14.1.3 C# 4 如何提供动态类型
  - 14.2 关于动态的快速指南
  - 14.3 动态类型示例
    - 14.3.1 COM 和Office
    - 14.3.2 动态语言
    - 14.3.3 纯托管代码中的动态类型
  - 14.4 幕后原理
    - 14.4.1 DLR 简介
    - 14.4.2 DLR 核心概念
    - 14.4.3 C#编译器如何处理动态
    - 14.4.4 更加智能的C#编译器
    - 14.4.5 动态代码的约束
  - 14.5 实现动态行为
    - 14.5.1 使用ExpandableObject
    - 14.5.2 使用DynamicObject
    - 14.5.3 实现IDynamicMetaObject-Provider
  - 14.6 小结
- 第15章 使用契约让代码更加清晰
  - 15.1 未引入代码契约之前的状态
  - 15.2 代码契约
    - 15.2.1 前置条件
    - 15.2.2 后置条件
    - 15.2.3 固定条件
    - 15.2.4 断言和假设
    - 15.2.5 旧式契约
  - 15.3 使用ccrewrite 和ccrefgen 重写二进制
    - 15.3.1 简单重写
    - 15.3.2 契约继承
    - 15.3.3 契约引用程序集
    - 15.3.4 失败行为
  - 15.4 静态检查
    - 15.4.1 开始静态检查
    - 15.4.2 隐式职责
    - 15.4.3 有选择性的检查

<<深入理解C# (第2版)>>

15.5 使用ccdocgen 将契约文档化

15.6 契约实战

15.6.1 契约中有什么

15.6.2 如何开始

15.6.3 无处不在的选项

15.7 小结

第16章 何去何从

16.1 C#——传统与现代的结合

16.2 计算机科学和.NET

16.3 计算机世界

16.4 再会

附录A LINQ 标准查询操作符

附录B .NET 中的泛型集合

附录C 版本总结

<<深入理解C# (第2版)>>

编辑推荐

从现象中挖掘本质 全面剖析C# 4新特性 深入理解，探求本源 提升C#编程功力的首选。

<<深入理解C# (第2版)>>

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>