

《C#本质论》

图书基本信息

书名：《C#本质论》

13位ISBN编号：9787115181879

10位ISBN编号：711518187X

出版时间：2008-7

出版社：人民邮电出版社

作者：米凯利斯

页数：498

译者：周靖

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：www.tushu000.com

《C#本质论》

内容概要

《C#本质论》是一本清晰、简明的C#教程，涵盖了C#2.0。书中对C#语言的每个重要结构都用简短的示例代码进行说明，并且和其他语言进行了全方位比较。每章开头的“思维导图”指明了本章要讨论的主题以及每个主题同整体的关系。全书由18章和3个附录组成。在简单介绍了C#之后，重点讨论了C#的数据类型、运算符、方法、类等基本概念，随后还对泛型、迭代器、反射、线程、互操作性等高级主题进行了深入而透彻的讨论。

点击链接进入新版：[C#本质论\(第3版\)](#)

《C#本质论》

作者简介

Mark Michaelis，微软Visual Studio MVP，现任Itron公司企业架构师，并在著名顾问公司Pluralsight担任导师。他是广受尊敬的资深C#专家，曾经受邀担任C#和VSTS等多个微软产品的软件设计审查顾问。除本书外，他还与Herbert Schildt合写过COM+方面的著作。

书籍目录

第1章 C#概述1.1 Hello, World1.1.1 应用程序的编译和运行1.1.2 托管执行和公共语言基础结构1.2 C#语法基础1.2.1 C#关键字1.2.2 类型定义1.2.3 Main1.2.4 语句和语句分隔符1.2.5 空白1.3 使用变量1.3.1 数据类型1.3.2 变量的声明1.3.3 变量的赋值1.3.4 变量的使用1.4 控制台输入和输出1.4.1 从控制台获取输入1.4.2 将输出写入控制台1.5 注释1.6 CIL和ILDASM1.7 小结第2章 数据类型2.1 基本数值类型2.1.1 整数类型2.1.2 浮点类型2.1.3 decimal类型2.1.4 字面值2.2 更多基本类型2.2.1 布尔类型2.2.2 字符类型2.2.3 字符串2.3 null和void2.3.1 null2.3.2 void2.4 类型的分类2.4.1 值类型2.4.2 引用类型2.5 可空修饰符2.6 数据类型之间的转换2.6.1 显式转型2.6.2 隐式转型2.6.3 不进行转型的类型转换2.7 数组2.7.1 数组的声明2.7.2 数组的实例化和赋值2.7.3 数组的使用2.7.4 字符串作为数组使用2.7.5 常见错误2.8 小结第3章 运算符和控制流3.1 运算符3.1.1 一元运算符正和负3.1.2 二元算术运算符3.1.3 圆括号运算符3.1.4 赋值运算符3.1.5 递增和递减运算符3.1.6 常量表达式3.2 流控制概述3.2.1 if语句3.2.2 嵌套if3.3 代码块3.4 作用域3.5 布尔表达式3.5.1 关系运算符和相等性运算符3.5.2 逻辑布尔运算符3.5.3 逻辑求反运算符3.5.4 条件运算符3.6 按位运算符3.6.1 移位运算符3.6.2 按位运算符3.6.3 按位赋值运算符3.6.4 按位取反运算符3.7 控制流语句3.7.1 while和do/while循环3.7.2 for循环3.7.3 foreach循环3.7.4 switch语句3.8 跳转语句3.8.1 break语句3.8.2 continue语句3.8.3 goto语句3.9 C#预处理器指令3.9.1 排除和包含代码3.9.2 定义预处理器符号3.9.3 生成错误和警告3.9.4 关闭警告消息3.9.5 nowarn:选项3.9.6 指定行号3.9.7 可视编辑器提示3.10 小结第4章 方法和参数4.1 方法的调用4.1.1 命名空间4.1.2 类型名称4.1.3 作用域4.1.4 方法名称4.1.5 参数4.1.6 方法返回值4.1.7 语句与方法调用的比较4.2 方法的声明4.2.1 参数声明4.2.2 方法返回值声明4.3 using指令4.4 Main()的返回值和参数4.5 参数4.5.1 值参数4.5.2 引用参数(ref)4.5.3 输出参数(out)4.5.4 参数数组(params)4.6 递归4.7 方法重载4.8 用异常实现基本错误处理4.8.1 捕捉错误4.8.2 使用throw语句报告错误4.9 小结第5章 类5.1 类的定义和实例化5.2 实例字段5.2.1 实例字段的声明5.2.2 实例字段的访问5.3 实例方法5.4 使用this关键字5.5 访问修饰符5.6 构造器5.6.1 构造器的声明5.6.2 默认构造器5.6.3 构造器的重载5.6.4 使用this调用另一个构造器5.7 静态5.7.1 静态字段5.7.2 静态方法5.7.3 静态构造器5.7.4 静态类5.8 const和readonly修饰符5.8.1 const5.8.2 readonly5.9 属性5.9.1 属性的声明5.9.2 命名规范5.9.3 静态属性5.9.4 提供属性验证5.9.5 只读和只写属性5.9.6 为getter和setter指定访问修饰符5.9.7 属性作为虚字段使用5.9.8 属性和方法调用不允许作为ref或out参数值使用5.10 嵌套类5.11 分部类5.12 小结第6章 继承6.1 派生6.1.1 基类型和派生类型之间的转型6.1.2 对参数“协变”和“逆变”的支持6.1.3 private访问修饰符6.1.4 protected访问修饰符6.1.5 单一继承6.1.6 密封类6.2 基类的重写6.2.1 virtual修饰符6.2.2 new修饰符6.2.3 sealed修饰符6.2.4 base成员6.2.5 构造器6.3 抽象类6.4 一切最终都从System.Object派生6.5 使用is运算符验证基础类型6.6 使用as运算符进行转换6.7 小结第7章 接口7.1 接口概述7.2 通过接口来实现多态性7.3 接口实现7.3.1 显式成员实现7.3.2 隐式成员实现7.3.3 显式接口实现与隐式接口实现的比较7.4 “实现类”与其接口之间的转型7.5 接口继承7.6 多接口继承7.7 通过接口来实现多重继承7.8 版本控制7.9 接口与类的比较7.10 小结第8章 值类型8.1 结构8.1.1 struct的初始化8.1.2 default运算符的使用8.1.3 值类型的继承和接口8.2 装箱8.3 枚举8.3.1 枚举之间的类型兼容性8.3.2 枚举和字符串之间的转换8.3.3 枚举作为标志使用8.4 小结第9章 合式类型9.1 重写object的成员9.1.1 重写ToString()9.1.2 重写GetHashCode()9.1.3 重写Equals()9.1.4 相等性实现的指导原则9.2 运算符重载9.2.1 比较运算符9.2.2 二元运算符9.2.3 赋值与二元运算符的结合9.2.4 条件逻辑运算符9.2.5 一元运算符9.2.6 转换运算符9.2.7 转换运算符的指导原则9.3 引用其他程序集9.3.1 更改程序集目标9.3.2 类型封装9.3.3 引用程序集9.4 定义命名空间9.5 XML注释9.5.1 将XML注释与代码构造关联到一起9.5.2 生成XML文档文件9.6 垃圾回收9.7 资源清理9.7.1 终结器9.7.2 使用using语句进行确定性终结9.7.3 垃圾回收和终结9.7.4 资源利用和终结的指导原则9.8 小结第10章 异常处理10.1 多异常类型10.2 捕捉异常10.3 常规catch块10.4 异常处理的指导原则10.5 定义自定义异常10.6 小结第11章 泛型11.1 如果C#没有泛型11.2 泛型类型概述11.2.1 泛型类的使用11.2.2 简单泛型类的定义11.2.3 泛型的优点11.2.4 类型参数命名的指导原则11.2.5 泛型接口和struct11.2.6 构造器和终结器的定义11.2.7 默认值的指定11.2.8 多个类型参数11.2.9 嵌套泛型类型11.2.10 “类型参数”兼容的泛型类

之间的类型兼容性11.3 约束11.3.1 接口约束11.3.2 基类约束11.3.3 struct/class约束11.3.4 多个约束11.3.5 构造器约束11.3.6 约束继承11.4 泛型方法11.4.1 类型推断11.4.2 约束的指定11.5 泛型的内部机制11.5.1 基于值类型的泛型的实例化11.5.2 基于引用类型的泛型的实例化11.6 小结第12章 集合12.1 主要集合类12.1.1 列表集合：List和ArrayList12.1.2 字典集合：Dictionary和Hashtable12.1.3 已排序集合：SortedDictionary和SortedList12.1.4 栈集合：Stack和Stack12.1.5 队列集合：Queue和Queue12.1.6 链表：LinkedList12.2 集合类接口概述12.2.1 IList和IDictionary12.2.2 IComparable12.2.3 ICollection12.2.4 使用foreach循环来迭代12.3 提供一个索引运算符12.4 返回Null或者空集合12.5 迭代器12.5.1 迭代器的定义12.5.2 迭代器语法12.5.3 从迭代器yield值12.5.4 迭代器和状态12.5.5 更多的迭代器例子12.5.6 将yield return语句放到循环中12.5.7 取消更多的迭代：yield break12.5.8 在单个类中创建多个迭代器12.5.9 yield语句的特征12.6 小结第13章 委托和事件13.1 方法指针13.1.1 定义场景13.1.2 委托数据类型13.1.3 委托的内部机制13.1.4 委托类型的定义13.1.5 委托的实例化13.1.6 匿名方法13.1.7 外部变量13.2 multicast委托和Observer模式13.2.1 使用委托来编写Observer模式13.2.2 顺序调用13.3 事件13.3.1 事件的作用13.3.2 事件的声明13.3.3 编程规范13.3.4 泛型和委托13.3.5 自定义事件的实现13.4 小结第14章 反射和attribute14.1 反射14.1.1 使用System.Type访问元数据14.1.2 成员调用14.1.3 泛型类型上的反射14.2 attribute14.2.1 自定义attribute14.2.2 查找attribute14.2.3 使用构造器来初始化attribute14.2.4 System.AttributeUsage-Attribute14.2.5 具名参数14.3 小结第15章 多线程处理15.1 独立线程的运行和控制15.1.1 线程的启动15.1.2 线程管理15.2 向线程传递参数15.3 线程池处理15.4 未处理的异常15.5 同步15.5.1 使用Monitor来同步15.5.2 使用lock关键字15.5.3 lock对象的选择15.5.4 为什么要避免在this和typeof(type)上锁定15.5.5 将字段声明为volatile15.5.6 使用System.Threading.Interlocked类15.5.7 多个线程时的事件通知15.5.8 同步设计最佳实践15.5.9 更多的同步类型15.6 计时器15.7 小结第16章 多线程处理模式16.1 Asynchronous Results模式16.1.1 Asynchronous Results模式概述16.1.2 向轮换线程传入数据以及从轮换线程传出数据16.1.3 接收线程完成通知16.1.4 传递任意状态16.1.5 Asynchronous Results小结16.2 Background Worker模式16.2.1 模式的建立16.2.2 异常处理16.3 Windows窗体16.4 小结第17章 平台互操作性和不安全的代码17.1 平台调用17.1.1 外部函数的声明17.1.2 参数的数据类型17.1.3 使用ref而不是指针17.1.4 为顺序布局使用Struct-LayoutAttribute17.1.5 错误处理17.1.6 使用SafeHandle17.1.7 外部函数的调用17.1.8 用包装简化API调用17.1.9 函数指针映射到委托17.1.10 指导原则17.2 指针和地址17.2.1 不安全的代码17.2.2 指针的声明17.2.3 指针的赋值17.2.4 指针的解引用17.2.5 引用类型的成员的访问17.3 小结第18章 CLI18.1 CLI的定义18.2 CLI实现18.3 C#编译成机器码18.4 运行时18.4.1 垃圾回收18.4.2 .NET的垃圾回收18.4.3 类型安全18.4.4 代码访问安全性18.4.5 平台可移植性18.4.6 性能18.5 应用程序域18.6 程序集、清单和模块18.7 公共中间语言18.8 公共类型系统18.9 公共语言规范18.10 基类库18.11 元数据18.12 小结附录A 下载和安装C#编译器与CLI平台A.1 Microsoft.NET A.2 Mono附录B 完整源代码清单附录C C# 2.0主题

章节摘录

第1章 C#概述 C#是一种相当新的语言，它基于之前的C风格语言（C、C++和Java）的特性而设计，所以许多有经验的程序员很快就能熟悉它。C#作为用于构建软件组件和应用程序的一种编程语言，是一个更人、更复杂的执行平台——“公共语言基础结构”（Common Language Infrastructure, CLI）——的一部分。本章使用传统的HelloWorld程序来介绍C#。我将重点放在C#语法基础上，其中包括在C#程序的可执行文件中定义一个入口。通过本章的学习，你将熟悉C#的语法风格和结构，并能够开始写最简单的C#程序。在讨论C#语法基础之前，我简单介绍了托管执行环境，解释了C#程序在运行时是如何执行的。本章最后讨论了变量声明、如何在控制台上写入和检索数据以及为C#代码添加注释的基础知识。

《C#本质论》

媒体关注与评论

我坚信这本参考书兼教程会成为你的良师益友。现在就开始阅读吧！我完全相信，无论是C#语言的新手还是有经验的开发者，都会在阅读中体验到“灵机一动”的感觉。——Prashant Sridharan 微软C#产品经理

令人耳目一新！本书对新手和专家都极有价值。 Jon Skeet 微软C#MVP

《C#本质论》

编辑推荐

《C#本质论》适用于对C#感兴趣的各种层次的读者，无论是初学者还是高级编程人员，都能从《C#本质论》中获益。Amazon全五星图书在与其他语言的比较中轻松学习微软C#产品经理强烈推荐

精彩短评

- 1、差一点就读完了。
- 2、C#必读之一
- 3、很不错的一本入门书
- 4、相当精辟 的确本质 私以为学C#必读
- 5、觉得基础烂的不行，就又把图解教程翻了一遍
- 6、该去再买本4.0
- 7、可以作为C#的入门书，讲的不罗嗦，但是例子不多。
- 8、初学C#的第一本书

《C#本质论》

精彩书评

- 1、本书实在是鄙人学习计算机语言以来见过的最好的书。无论是行文以及例子，都是那么得无懈可击。真的很好，非常典型。无论读者是什么水平，都能从中获益！
- 2、优点:作者对C#的基础知识讲的很清楚,同时例子也很有代表性,对初学者来说有点难度但不是很大,读起来很顺畅,相比其他的C#书籍厚度也适中,作为一本C#入门书籍非常理想.不足:对.NET框架的内部原理和应用介绍偏少.总结:该书是一本优秀的C#入门书籍,但若是搞平台开发,书中讲述的知识略显不足.
- 3、真的很实在，虽然我不是学c#的，但也能从中学到点东西。尤其是对内部细节的一些深入探讨，在很多入门书中几乎是不可能看到的。很高兴在国内还能看到这样的一本书，总体来说，翻译的质量并不能说非常好，但是对于很多原先就是用中文写的书来说还是要好得多，感谢作者，感谢译者。

章节试读

1、《C#本质论》的笔记-第229页

```
public readonly Longitude Longitude;
public readonly Latitude Latitude;

public override int GetHashCode()
{
    int hashCode = Longitude.GetHashCode();
    // As long as the hash codes are not equal
    if(Longitude != Latitude)
    {
        hashCode ^= Latitude.GetHashCode(); // eXclusive OR
    }

    return hashCode;
}
```

一个struct的GetHashCode()方法重写的范例，我没看懂 ==

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:www.tushu000.com