<<编写高质量代码>>

图书基本信息

书名:<<编写高质量代码>>

13位ISBN编号: 9787111356493

10位ISBN编号:7111356497

出版时间:2011-10

出版时间:机械工业出版社

作者:陆敏技

页数:347

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

<<编写高质量代码>>

内容概要

本书是C#程序员进阶修炼的必读之作,包含的全部都是C#编码的最佳实践,从语言本身、程序的设计和架构、编码规范和编程习惯等三大方面对C#程序员遇到的经典问题给出了经验性的解决方案,为C#程序员如何编写更高质量的C#代码提供了157条极为宝贵的建议。

对于每一个问题,不仅以建议的方式给出了被实践证明为十分优秀的解决方案,而且还给出了经常被 误用或被错误理解的不好的解决方案,从正反两个方面进行了分析和对比。

全书一共三个部分,第一部分专注于C#语言本身,一共89条建议,涵盖了C#语言基本要素、集合、LINQ、泛型、委托、事件、资源管理、序列化、异常处理、异步、多线程、任务和并行编程等与C#语法相关的核心内容;第二部分重点讲解了C#程序的设计和架构,一共32条建议,涉及成员设计、面向对象的类型设计、安全性设计等重要方面的内容;第三部分探讨了C#的编码规范及编程习惯,一共36条建议,包含C#命名规范、如何使代码更整洁以及如何规范开发行为等方面的内容。

本书是一本关于如何编写高质量C#代码的工具书,列举的问题非常典型,给出的建议也非常实用,其中的每一条建议都有可能在我们编写下一行代码的时候被用到。 你可以将此书搁置在案头,以便有需要的时候随时查阅。

<<编写高质量代码>>

作者简介

陆敏技,资深软件工程师、项目经理和架构师,从事软件开发工作近10年。 尤其精通微软技术,对C#、WPF、WCF、ASP.NET和.NET技术有十分深入的研究,曾参与和主导了大量的相关项目的架构和开发工作,积累了丰富的经验。 此外,他还非常擅长于分布式开发技术,而且有丰富的培训和授课经验。 活跃于博客园等技术社区,乐于分享,有较高的知名度和社区影响力。

书籍目录

•	_
-	=
ни	

第一部分 语言篇

第1章 基本语言要素

建议1:正确操作字符串建议2:使用默认转型方法

建议3:区别对待强制转型与as和is

建议4:TryParse比Parse好

建议5:使用int?来确保值类型也可以为null 建议6:区别readonly和const的使用方法

建议7:将0值作为枚举的默认值

建议8:避免给枚举类型的元素提供显式的值

建议9:习惯重载运算符

建议10:创建对象时需要考虑是否实现比较器

建议11:区别对待==和Equals

建议12:重写Equals时也要重写GetHashCode

建议13:为类型输出格式化字符串 建议14:正确实现浅拷贝和深拷贝 建议15:使用dynamic来简化反射实现

第2章 集合和LINQ

建议16:元素数量可变的情况下不应使用数组建议17:多数情况下使用foreach进行循环遍历

建议18: foreach不能代替for

建议19:使用更有效的对象和集合初始化 建议20:使用泛型集合代替非泛型集合

建议21:选择正确的集合建议22:确保集合的线程安全

建议23:避免将List作为自定义集合类的基类

建议24: 迭代器应该是只读的 建议25: 谨慎集合属性的可写操作

建议26:使用匿名类型存储LINQ查询结果

建议27:在查询中使用Lambda表达式

建议28:理解延迟求值和主动求值之间的区别

建议29:区别LINQ查询中的IEnumerable和IQueryable 建议30:使用LINQ取代集合中的比较器和迭代器

建议31:在LINQ查询中避免不必要的迭代

第3章 泛型、委托和事件

建议32:总是优先考虑泛型

建议33:避免在泛型类型中声明静态成员

建议34:为泛型参数设定约束

建议35:使用default为泛型类型变量指定初始值

建议36:使用FCL中的委托声明

建议37:使用Lambda表达式代替方法和匿名方法

建议38:小心闭包中的陷阱建议39:了解委托的实质

建议40:使用event关键字为委托施加保护

建议41:实现标准的事件模型

建议42:使用泛型参数兼容泛型接口的不可变性

建议43:让接口中的泛型参数支持协变

建议44:理解委托中的协变

建议45:为泛型类型参数指定逆变

第4章 资源管理和序列化

建议46:显式释放资源需继承接口IDisposable

建议47:即使提供了显式释放方法,也应该在终结器中提供隐式清理

建议48: Dispose方法应允许被多次调用

建议49:在Dispose模式中应提取一个受保护的虚方法

建议50:在Dispose模式中应区别对待托管资源和非托管资源

建议51:具有可释放字段的类型或拥有本机资源的类型应该是可释放的

建议52:及时释放资源

建议53:必要时应将不再使用的对象引用赋值为null

建议54:为无用字段标注不可序列化

建议55:利用定制特性减少可序列化的字段

建议56:使用继承ISerializable接口更灵活地控制序列化过程

建议57:实现ISerializable的子类型应负责父类的序列化

第5章 异常与自定义异常

建议58:用抛出异常代替返回错误代码

建议59:不要在不恰当的场合下引发异常

建议60:重新引发异常时使用Inner Exception

建议61:避免在finally内撰写无效代码

建议62:避免嵌套异常

建议63:避免"吃掉"异常

建议64:为循环增加Tester-Doer模式而不是将try-catch置于循环内

建议65:总是处理未捕获的异常

建议66:正确捕获多线程中的异常

建议67: 慎用自定义异常

建议68:从System.Exception或其他常见的基本异常中派生异常

建议69:应使用finally避免资源泄漏

建议70:避免在调用栈较低的位置记录异常

第6章 异步、多线程、任务和并行

建议71:区分异步和多线程应用场景

建议72:在线程同步中使用信号量

建议73:避免锁定不恰当的同步对象

建议74:警惕线程的IsBackground

建议75:警惕线程不会立即启动

建议76:警惕线程的优先级

建议77:正确停止线程

建议78:应避免线程数量过多

建议79:使用ThreadPool或BackgroundWorker代替Thread

建议80:用Task代替ThreadPool

建议81:使用Parallel简化同步状态下Task的使用

建议82:Parallel简化但不等同于Task默认行为

建议83:小心Parallel中的陷阱

建议84:使用PLINQ

建议85: Task中的异常处理 建议86: Parallel中的异常处理

建议87:区分WPF和WinForm的线程模型

建议88:并行并不总是速度更快 建议89:在并行方法体中谨慎使用锁

第二部分 架构篇 第7章 成员设计

建议90:不要为抽象类提供公开的构造方法

建议91:可见字段应该重构为属性建议92:谨慎将数组或集合作为属性

建议93:构造方法应初始化主要属性和字段

建议94:区别对待override和new

建议95:避免在构造方法中调用虚成员

建议96:成员应优先考虑公开基类型或接口

建议97:优先考虑将基类型或接口作为参数传递

建议98:用params减少重复参数 建议99:重写时不应使用子类参数

建议100:静态方法和实例方法没有区别

建议101:使用扩展方法,向现有类型"添加"方法

第8章 类型设计

建议102:区分接口和抽象类的应用场合建议103:区分组合和继承的应用场合

建议104:用多态代替条件语句

建议105:使用私有构造函数强化单例 建议106:为静态类添加静态构造函数

建议107:区分静态类和单例建议108:将类型标识为sealed建议109:谨慎使用嵌套类建议110:用类来代替enum建议111:避免双向耦合

建议112:将现实世界中的对象抽象为类,将可复用对象圈起来就是命名空间

第9章 安全性设计

建议113:声明变量前考虑最大值

建议114:MD5不再安全

建议115:通过HASH来验证文件是否被篡改

建议116:避免用非对称算法加密文件 建议117:使用SSL确保通信中的数据安全

建议118:使用SecureString保存密钥等机密字符串

建议119:不要使用自己的加密算法建议120:为程序集指定强名称建议121:为应用程序设定运行权限

第三部分 编码规范及习惯

第10章 命名规范

建议122:以.为命名空间命名

建议123:程序集不必与命名空间同名建议124:考虑在命名空间中使用复数

建议125:避免用FCL的类型名称命名自己的类型

建议126:用名词和名词组给类型命名建议127:用形容词组给接口命名

建议128:考虑让派生类的名字以基类名字作为后缀

建议129:泛型类型参数要以T作为前缀

建议130:以复数命名枚举类型,以单数命名枚举元素

建议131:用PascalCasing命名公开元素 建议132:考虑用类名作为属性名

建议133:用camelCasing命名私有字段和局部变量

建议134:有条件地使用前缀

建议135: 考虑使用肯定性的短语命名布尔属性建议136:优先使用后缀表示已有类型的新版本

建议137:委托和事件类型应添加上级后缀

建议138:事件和委托变量使用动词或形容词短语命名

建议139:事件处理器命名采用组合方式

第11章 代码整洁

建议140:使用默认的访问修饰符

建议141:不知道该不该用大括号时,就用

建议142: 总是提供有意义的命名

建议143:方法抽象级别应在同一层次

建议144:一个方法只做一件事

建议145:避免过长的方法和过长的类

建议146:只对外公布必要的操作

建议147: 重构多个相关属性为一个类

建议148: 不重复代码

建议149:使用表驱动法避免过长的if和switch分支 建议150:使用匿名方法、Lambda表达式代替方法 建议151:使用事件访问器替换公开的事件成员变量

建议152:最少,甚至是不要注释

建议153:若抛出异常,则必须要注释

第12章 规范开发行为

建议154:不要过度设计,在敏捷中体会重构的乐趣

建议155:随生产代码一起提交单元测试代码建议156:利用特性为应用程序提供多个版本

建议157:从写第一个界面开始,就进行自动化测试

<<编写高质量代码>>

媒体关注与评论

"这是一本关于C#编码最佳实践的集大成之作,也是一本能指导C#程序员编写出高质量代码的指点迷津之作。

全书从C#语法、程序的架构和设计、编码规范和编程习惯3大方面为广大的C#程序员们总结出了157条极富借鉴意义的建议,这些建议都是在实践中被证明是解决C#编码中疑难问题的最佳实践。

如果能掌握本书中的内容,不仅能加深对C#语言的理解和提升程序架构和设计方面的能力,而且还能 规范我们的开发行为和习惯,让我们成为一名优秀的程序员,让我们能编写出更高质量的代码。

"——51CTO "作为一名程序员,没有人愿意留下一堆糟糕的代码。

如果我们编写的代码运行速度慢、消耗的内存多,而且还时不时地抛出一些莫名其妙的错误,我们一 定会十分疑惑:我们的代码到底怎么了?

问题很明显:我们编写的代码质量不高。

本书从C#语言本身、程序的架构和设计、编码规范和编程习惯等3大方面就如何编写高质量的C#代码给出了大量的宝贵建议。

如果能理解并熟练使用本书中的知识,我们不仅能在一定程度上避免糟糕的代码,而且还能大幅度提升编码水平。

- "——马伟(资深微软技术专家/ASP.NET MVP/畅销书《ASP.NET 4权威指南》作者)
- "学习任何一门编程语言,入门一般都不难,难的是进阶和提高;让程序跑起来不难,难的是如何让程序跑得又快又好。

作为一个程序员,我们在进阶的路上总会遇到各种各样的问题,有时候需要为这些问题付出代价,需要在不断试错和纠错中积累经验。

很幸运的是,本书针对C#语法、程序的架构和设计、编码规范和编程习惯等3大方面给出了157条宝贵的建议,每一条建议都来自于实践和经验的总结,每一条建议都能帮你避免在编码时可能会犯下的错误,实用性极强。

强烈推荐!

" ——姜晓东(资深微软技术专家/畅销书《C# 4.0权威指南》作者)

<<编写高质量代码>>

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