

<<你必须知道的.NET>>

图书基本信息

书名：<<你必须知道的.NET>>

13位ISBN编号：9787121058912

10位ISBN编号：712105891X

出版时间：2008-4

出版时间：电子工业出版社

作者：王涛

页数：497

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>

<<你必须知道的.NET>>

内容概要

本书来自于微软MVP的最新技术心得和感悟，将技术问题以生动易懂的语言展开，层层深入，以例说理。

全书主要包括了.NET基础知识及其深度分析，以.NET Framework和CLR研究为核心展开.NET本质论述，涵盖了.NET基本知识几乎所有的重点内容。

全书分为5个部分，第1部分讲述.NET与面向对象，从底层实现角度分析了.NET如何实现面向对象机制，进一步分析了面向对象设计原则；第2部分论述了.NET类型系统和CLR的内存管理机制，并对IL语言进行了相应介绍；第3部分论述.NET Framework框架的方方面面，详细分析了.NET框架的所有重点、难点和疑点内容，对框架类库的全貌进行了必要的专题性探讨；第4部分重点介绍了.NET泛型和安全性的相关知识和本质解密；第5部分对.NET 3.0/3.5新特性进行了详细的介绍和引导，对于快速入门.NET新特性提供了方便之门。

本书适于对.NET有一定了解的技术学习者、软件工程师和系统架构师阅读，同时也有助于.NET初学者进行快速提高，可作为大中专院校和.NET技术培训机构的参考教材。

<<你必须知道的.NET>>

作者简介

王涛，微软C# MVP，高级软件工程师，机械工程硕士，主要研究方向为.NET底层架构和企业级系统应用。

近年来一直致力于.NET技术与开发，现就职于一家提供系统集成和电子政务解决方案的知名软件公司，负责架构设计、软件开发和项目管理方面的工作。

作者对.NET基础架构和CLR底层运行机制有浓厚的研究兴趣和造诣，熟悉ASP.NET、XML、SQL Server、ADO.NET相关技术，精通数据库应用系统和大型Web系统的开发流程、技术体系和架构设计，对面向对象、设计模式和软件架构有一定的研究与实践经验。

<<你必须知道的.NET>>

书籍目录

第1部分 渊源——.NET与面向对象第1章 OO大智慧1.1 对象的旅行1.1.1 引言1.1.2 出生1.1.3 旅程1.1.4 插曲1.1.5 消亡1.1.6 结论1.2 什么是继承1.2.1 引言1.2.2 基础为上1.2.3 继承本质论1.2.4 秘境追踪1.2.5 规则制胜1.2.6 结论1.3 封装的秘密1.3.1 引言1.3.2 让ATM告诉你,什么是封装1.3.3 秘密何处:字段、属性和方法1.3.4 封装的意义1.3.5 封装规则1.3.6 结论1.4 多态的艺术1.4.1 引言1.4.2 问题的抛出1.4.3 最初的实现1.4.4 多态,救命的稻草1.4.5 随需而变的业务1.4.6 多态的类型、本质和规则1.4.7 结论1.5 玩转接口1.5.1 引言1.5.2 什么是接口1.5.3 .NET中的接口1.5.4 面向接口的编程1.5.5 接口之规则1.5.6 结论参考文献第2章 OO大原则2.1 OO原则综述2.1.1 引言2.1.2 讲述之前2.1.3 原则综述2.1.4 学习建议2.1.5 结论2.2 单一职责原则2.2.1 引言2.2.2 引经据典2.2.3 应用反思2.2.4 规则建议2.2.5 结论2.3 开放封闭原则2.3.1 引言2.3.2 引经据典2.3.3 应用反思2.3.4 规则建议2.3.5 结论2.4 依赖倒置原则2.4.1 引言2.4.2 引经据典2.4.3 应用反思2.4.4 规则建议2.4.5 结论2.5 接口隔离原则2.5.1 引言2.5.2 引经据典2.5.3 应用反思2.5.4 规则建议2.5.5 结论2.6 Liskov替换原则2.6.1 引言2.6.2 引经据典2.6.3 应用反思2.6.4 规则建议2.6.5 结论参考文献第2部分 本质——.NET深入浅出第3章 一切从IL开始3.1 从Hello, world开始认识IL3.1.1 引言3.1.2 从Hello, world开始3.1.3 IL体验中心3.1.4 结论3.2 教你认识IL代码——从基础到工具3.2.1 引言3.2.2 使用工具3.2.3 为何而探索3.2.4 结论3.3 教你认识IL代码——IL语言基础3.3.1 引言3.3.2 变量的声明3.3.3 基本类型3.3.4 基本运算3.3.5 数据加载与保存3.3.6 流程控制3.3.7 结论3.4 经典指令解析之实例创建3.4.1 引言3.4.2 newobj和initobj3.4.3 ldstr3.4.4 newarr3.4.5 结论3.5 经典指令解析之方法调度3.5.1 引言3.5.2 方法调度简论:call、callvirt3.5.2 和calli3.5.3 直接调度3.5.4 间接调度3.5.5 动态调度3.5.6 结论参考文献第4章 品味类型4.1 品味类型——从通用类型系统开始4.1.1 引言4.1.2 基本概念4.1.3 位置与关系4.1.4 通用规则4.1.5 结论4.2 品味类型——值类型与引用类型4.2.1 引言4.2.2 内存有理4.2.3 通用规则与比较4.2.4 对症下药——应用场合与4.2.4 注意事项4.2.5 再论类型判等4.2.6 再论类型转换4.2.7 以代码剖析4.2.8 结论4.3 参数之惑——传递的艺术4.3.1 引言4.3.2 参数基础论4.3.3 传递的基础4.3.4 深入讨论,传递的艺术4.3.5 结论4.4 皆有可能——装箱与拆箱4.4.1 引言4.4.2 品读概念4.4.3 原理分拆4.4.4 还是性能4.4.5 重在应用4.4.6 结论参考文献第5章 内存天下5.1 内存管理概要5.1.1 引言5.1.2 内存管理概观要论5.1.3 结论5.2 对象创建始末5.2.1 引言5.2.2 内存分配5.2.3 结论5.3 垃圾回收5.3.1 引言5.3.2 垃圾回收5.3.3 非托管资源清理5.3.4 结论5.4 性能优化的多方探讨5.4.1 引言5.4.2 性能条款5.4.3 结论参考文献第3部分 格局——.NET面面俱到第6章 深入浅出——关键字的秘密6.1 把new说透6.1.1 引言6.1.2 基本概念6.1.3 深入浅出6.1.4 结论6.2 base和this6.2.1 引言6.2.2 基本概念6.2.3 深入浅出6.2.4 通用规则6.2.5 结论6.3 using的多重身份6.3.1 引言6.3.2 引入命名空间6.3.3 创建别名6.3.4 强制资源清理6.3.5 结论6.4 转换关键字6.4.1 引言6.4.2 自定义类型转换探讨6.4.3 本质分析6.4.4 结论6.5 预处理指令关键字6.5.1 引言6.5.2 预处理指令简述6.5.3 #if、#else、#elif、#endif6.5.4 #define、#undef6.5.5 #warning、#error6.5.6 #line6.5.7 结论6.6 非主流关键字6.6.1 引言6.6.2 checked/unchecked6.6.3 yield6.6.4 lock6.6.5 unsafe6.6.6 sealed6.6.7 结论参考文献第7章 巅峰对决——走出误区7.1 什么才是不变:const和readonly7.1.1 引言7.1.2 从基础到本质7.1.3 比较,还是规则7.1.4 进一步的探讨7.1.5 结论7.2 后来居上:class和struct7.2.1 引言7.2.2 基本概念7.2.3 相同点,不同点7.2.4 经典示例7.2.5 结论7.3 历史纠葛:特性和属性7.3.1 引言7.3.2 概念引入7.3.3 通用规则7.3.4 特性的应用7.3.5 应用示例7.3.6 结论7.4 面向抽象编程:接口和抽象类7.4.1 引言7.4.2 概念引入7.4.3 相同点,不同点7.4.4 经典示例7.4.5 他山之石7.4.6 结论7.5 恩怨情仇:is和as7.5.1 引言7.5.2 概念引入7.5.3 原理与示例说明7.5.4 结论7.6 貌合神离:覆写和重载7.6.1 引言7.6.2 认识覆写和重载7.6.3 在多态中的应用7.6.4 比较,还是规则7.6.5 进一步的探讨7.6.6 结论7.7 有深有浅的克隆:浅拷贝和深拷贝7.7.1 引言7.7.2 从对象克隆说起7.7.3 浅拷贝和深拷贝的实现7.7.4 结论7.8 动静之间:静态和非静态7.8.1 引言7.8.2 一言蔽之7.8.3 分而致之7.8.4 结论7.9 集合通论7.9.1 引言7.9.2 中心思想——纵论集合7.9.3 各分秋色——.NET集合类大观7.9.4 自我成全——实现自定义集合7.9.5 结论参考文献第8章 本来面目——

<<你必须知道的.NET>>

框架诠释8.1 万物归宗：System.Object8.1.1 引言8.1.2 初识8.1.3 分解8.1.4 意义8.1.5 结论8.2 规则而定：对象判等8.2.1 引言8.2.2 本质分析8.2.3 覆写Equals方法8.2.4 与GetHashCode方法同步8.2.5 规则8.2.6 结论8.3 如此特殊：大话String8.3.1 引言8.3.2 字符串创建8.3.3 字符串恒定性8.3.4 字符串驻留8.3.5 字符串操作典籍8.3.6 补充的礼物：StringBuilder8.3.7 结论8.4 简易不简单：认识枚举8.4.1 引言8.4.2 枚举类型解析8.4.3 枚举种种8.4.4 位枚举8.4.5 规则与意义8.4.6 结论8.5 一脉相承：委托、匿名方法和Lambda表达式8.5.1 引言8.5.2 解密委托8.5.3 委托和事件8.5.4 匿名方法8.5.5 Lambda表达式8.5.6 规则8.5.7 结论8.6 直面异常8.6.1 引言8.6.2 为何而抛？8.6.3 从try/catch/finally说起：解析8.6.3 异常机制8.6.4 .NET系统异常类8.6.5 定义自己的异常类8.6.6 异常法则8.6.7 结论参考文献第9章 格局之选——命名空间剖析9.1 基础——.NET框架概览9.1.1 引言9.1.2 框架概览9.1.3 历史变迁9.1.4 结论9.2 布局——框架类库研究9.2.1 引言9.2.2 为什么了解？9.2.3 框架类库的格局9.2.4 一点补充9.2.5 结论9.3 根基——System命名空间9.3.1 引言9.3.2 从基础类型说起9.3.3 基本服务9.3.4 结论9.4 核心——System次级命名空间9.4.1 引言9.4.2 System.IO9.4.3 System.Diagnostics9.4.4 System.Runtime.Serialization和9.4.4 System.Xml.Serialization9.4.5 结论参考文献第4部分 拾遗——.NET也有春天第10章 接触泛型10.1 追溯泛型10.1.1 引言10.1.2 推进思维，为什么泛型？10.1.3 解析泛型——运行时本质10.1.4 结论10.2 了解泛型10.2.1 引言10.2.2 领略泛型——基础概要10.2.3 典型.NET泛型类10.2.4 基础规则10.2.5 结论10.3 深入泛型10.3.1 引言10.3.2 泛型方法10.3.3 泛型接口10.3.4 泛型委托10.3.5 结论参考文献第11章 如此安全性11.1 怎么样才算是安全11.1.1 引言11.1.2 怎么样才算是安全11.1.3 .NET安全模型11.1.4 结论11.2 代码访问安全11.2.1 引言11.2.2 证据（Evidence）11.2.3 权限（Permission）和权限集11.2.4 代码组（Code Group）11.2.5 安全策略（Security Policy）11.2.6 规则总结11.2.7 结论11.3 基于角色的安全11.3.1 引言11.3.2 Principal（主体）11.3.3 Identity（标识）11.3.4 PrincipalPermission11.3.5 应用示例11.3.6 结论参考文献第5部分 未来——.NET技术展望第12章 .NET 3.0/3.5新革命12.1 品读新特性12.1.1 引言12.1.2 .NET新纪元12.1.3 程序语言新特性12.1.4 WPF、WCF、WF12.1.5 Visual Studio 2008新体验12.1.6 其他12.1.7 结论12.2 赏析C# 3.012.2.1 引言12.2.2 对象初始化器（Object Initializers）12.2.3 集合初始化器（Collection Initializers）12.2.4 自动属性（Automatic Properties）12.2.5 隐式类型变量（Implicitly Typed Local Variables）和隐式类型数组（Implicitly Typed Array）12.2.6 匿名类型（Anonymous Type）12.2.7 扩展方法（ExtensionMethods）12.2.8 查询表达式（Query Expressions）12.2.9 结论12.3 体验LINQ12.3.1 引言12.3.2 LINQ概览12.3.3 查询操作符12.3.4 LINQ to XML示例12.3.5 规则12.3.6 结论12.4 抢鲜Visual Studio 200812.4.1 引言12.4.2 Visual Studio 2008概览12.4.3 新特性简介12.4.4 开发示例12.4.5 结论12.5 江湖一统：WPF、WCF、WF12.5.1 引言12.5.2 WPF12.5.3 WCF12.5.4 WF12.5.5 结论参考文献编后记：遇见幸福

<<你必须知道的.NET>>

编辑推荐

适于对.NET有一定了解的技术学习者、软件工程师和系统架构师阅读，同时也有助于.NET初学者进行快速提高，可作为大中专院校和.NET技术培训机构的参考教材。

<<你必须知道的.NET>>

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>