<<C#程序设计语言>>

图书基本信息

书名：<<C#程序设计语言>>

13位ISBN编号：9787111282617

10位ISBN编号：7111282612

出版时间：2010-1

出版时间：机械工业出版社

作者：Anders Hejlsberg,Mads Torgersen,Scott Wiltamuth

页数：548

译者：顾雁宏,徐旭铭

前言

自2000年夏，.NET发布以来已经有8个年头了。
对我来说,当时，.NET最重要的两点就是结合了托管代码的本地执行和用于程序之间通信的XML消息机制。
不过那个时候我还没有意识到C#变得那么重要。
C#从一开始就是程序员们理解和使用。
.NET的主要手段。
如果你问一个普通的。
.NET程序员一个值类型和一个引用类型的区别是什么,通常的回答都是“结构和类的区别”,而非“是否是从System。
Value Type继承而来的类型”。
为什么?因为我们都是用语言,而不是通过API来和运行时(更重要的是,其他人)交流想法和意图的。
如果没有一门出色的语言,一个平台要想成功是不可想象的。
C#最初就为人们如何看待。
.NET打下了坚实的基础。
随着。
.NET的不断发展,C#的重要性也与日俱增,诸如迭代器和真正的闭包(也叫匿名方法)都是用C#译器实现的纯语言特性,而不是平台特性。
C#3.0的发布更意味着C#为了。
.NET不断创新的主角,它引入了标准化的查询操作符,简洁的lambda表达式。
扩展方法,以及在运行时访问表达式树的能力——这些都是通过语言和编译器实现的。
说到C#就一定会提到它的缔造者Anders Hejlsberg。
我非常荣幸地在C#3.0设计阶段连续好几个月参与了C#的设计会议,Anders的工作让我大开眼界。
他那种深谙程序员喜欢什么和不喜欢什么的天赋实在是一流——同时他又能和设计团队紧密合作,并最终获得最佳的设计方案。
特别是在C#3.0上,Anders在从函数式语言社区获取灵感并将它们带给广大群众的过程中层现出无与伦比的能力。
要知道这绝对不是一件容易的事情。
Guy Steele曾经在谈论Java时说到：“我们没打算要吸引Lisp程序员,我们的目标是C++程序员。
我们成功地把他们从转向Lisp的路上吸引了过来。
”当我看到C#3.0的时候,我就知道C#已经至少获得了一名C++程序员(就是我自己)的青睐。
即使C#很出色,但是人们还是需要一份用自然语言(这里是英文)和一些范式(BNF)写成的文档来帮助他们抓住要点,以及任何消除晦涩的地方。
而你手中的这本书正是这样的一份文档。
据我的经验,我敢说每个。
.NET程序员在读这本书的时候都至少会有一次“啊,原来如此”的感叹,它能让你的水平更上一层楼。
享受它吧。

<<C#程序设计语言>>

内容概要

C#语言结合了快速应用开发语言的高效和C/C++语言的强大。
现在C# 3.0又加入了函数式编程技术和语言集成查询（LINQ，Language INtegrated Query）。
《C#程序设计语言 (原书第3版)》正是C# 3.0的权威技术指南。

这一版由C#的缔造者Anders Hejlsberg和他的同事们合著，全部内容都更新到了C# 3.0版。
《C#程序设计语言 (原书第3版)》提供了C# 3.0语言完整的规格说明、参考资料、范例代码和来自九位卓越的C#大师的详细注解。

这些注解所达到的深度和广度是很难在其他书中找到的。
《C#程序设计语言 (原书第3版)》的正文介绍了C#的概念，而这些恰到好处的注解则解释了为什么这些特性是重要的，应该怎么使用它们，它们和其他语言的关系是什么，甚至它们是如何进化而来的。

对任何希望深入理解C#的程序员来说，这本书都是不容错过的参考经典。

作者简介

作者：(美国)海杰尔斯伯格(Anders Hejlsberg) (美国)Mads Torgersen (美国)Scott Wiltamuth 等 译者：顾雁宏 徐旭铭Ander-Hejlsberg是编程界的传奇人物。
他是C#语言的架构师，同时也是微软技术专家。
他在1996年加入微软，之前13年的职业生涯则是在Borland度过，他曾经是Delphi和Turbo Pascal的首席架构师。
Mads Torgersen是微软的资深程序经理。
作为C#程序经理，他负责召开C#语言的设计会议及维护C#语言的规范。
在2005年加入微软之前，Mads是奥尔胡斯大学的副教授，主要教授和研究面向对象编程语言。
在那里，他领导的小组设计实现了Java的泛型通配符。
Scott Wiltamuth是Visual Studio的合作程序经理。
他在微软做过很多面向程序员的项目，包括Visual Basic、VBScript、Jscript、Visual J++和Visual C#。
Scott是C#语言的设计师之一，他拥有斯坦福大学计算机科学硕士学位。
Peter Golde在离开微软之前是微软C#编译器的首席程序员。
他作为微软在ECMA委员会（这个委员会负责了c#的标准化工作）的主要代表，领导实现了编译器并参与了语言的设计。

<<C#程序设计语言>>

书籍目录

章节摘录

插图：5.3.2 未赋初值的变量下列的变量类别都被归类为还未赋初值的：·结构变量里还未赋初值的实例变量。
·输出参数，包括结构实例构造函数里的this变量。
·局部变量，除了那些在catch子句或for each语句里声明的以外。
5.3.3 确定明确赋值的精确规则编译器必须采用一套和在这一节里的描述等价的流程来确定每个使用的变量都是明确赋值的。
编译器会处理每一个拥有一到多个还未赋初值变量的函数成员的主体。
对于每一个还未赋初值的变量v，编译器会在函数成员里以下的每一个地方确认v的明确赋值状态：·在每个语句的开始。
·在每个语句的终点（8.1节）。
·在每个arc上控制转移到另一个语句或者到语句的终点。
·在每个表达式的开始。
·在每个表达式的终点。
v的明确赋值状态可以是：·明确赋值。
这表示在通向这一点的所有可能的控制流上，v都已经被赋值。
·未明确赋值。
对于在bool类型的表达式结尾的变量状态，还未明确赋值的变量状态可以（但不一定）属于以下的子状态之一：在true表达式之后明确赋值。
这个状态表示：如果布尔表达式的求值结果为true，那么v就是明确赋值的；但要是它的求值结果为false，那么就不一定会被赋值。
在false表达式之后明确赋值。
这个状态表示：如果布尔表达式的求值结果为false，那么v就是明确赋值的；但要是它的求值结果为true，那么就不一定会被赋值。
下面的规则控制了变量v的状态是如何在每一个位置上被决定的。
5.3.3.1 语句的一般规则·v在函数成员主体的开始是还未明确赋值的。
·v在任何不可及的语句开始都是明确赋值的。

媒体关注与评论

“据我的经验，我敢说每个程序员在读这本书的时候都至少会有一次‘啊，原来如此’的感叹，它能让你的编程水平更上一层楼。
”　　——选自Don Box的序

<<C#程序设计语言>>

编辑推荐

《C#程序设计语言 (原书第3版)》：开发人员专业技术丛书

版权说明