

<<Scala编程>>

图书基本信息

书名：<<Scala编程>>

13位ISBN编号：9787121121197

10位ISBN编号：7121121190

出版时间：2010-12

出版时间：电子工业出版社

作者：Martin Odersky、Lex Spoon、Bill Venners 著

页数：492

字数：782000

译者：黄海旭,高宇翔

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>

<<Scala编程>>

内容概要

本书介绍了一种新的编程语言，它把面向对象和函数式编程概念有机地结合为整体，从而形成一种完整统一、语义丰富的新思维体系。

本书循序渐进，由浅入深，经作者精心组织、仔细编排，将语言中的各种概念自然地铺陈在字里行间。

除此之外，本书还包含了大量富有针对性和趣味性的示例，它们除了提供对语言各个方面的具体演示之外，还从侧面说明了如何将函数式编程的理念切实并广泛地应用到面向对象编程中。

本书面向的读者是有一定编程经验的开发人员，他们希望能够开拓眼界，并致力于提高在软件开发各方面的技能。

<<Scala编程>>

作者简介

关于作者

Martin Odersky是Scala语言的创造者。

作为瑞士洛桑联邦理工学院（EPFL）的教授，他主要从事编程语言领域的工作。

更具体地说，是面向对象和函数式编程的语言。

他研究的论题是，这两种编程模式是硬币的两面，应该被尽可能地统一在一起。

为了证明这点，他已试验性地设计了大量的语言，从Pizza到GJ到Functional Nets语言。

他还作为Java泛型的联合设计师及当前javac参考编译器的原作者影响了Java的发展。

从2001年起，他主要从事Scala编程语言的设计、实现及改进工作。

Lex Spoon是Google的软件工程师。

他以EPFL博士后身份在Scala方面工作了两年时间，从佐治亚理工学院（Georgia Tech）获得计算机科学的博士学位。

那时他的主要工作是动态语言的静态分析。

除了Scala之外，他还从事大量其他的编程语言工作，范围从动态语言Smalltalk到科学性语言X10。

他和他的妻子，两只猫、一条吉娃娃还有一只乌龟，现生活于亚特兰大。

Bill Venners是Artima的总裁，兼Artima开发者网站（www.artima.com）的发行人。

他是《深入Java虚拟机》（“Inside the Java Virtual Machine”）的作者，该书是定向为程序员的Java平台架构和内部组织的总体研究。

他在JavaWorld杂志上有很受欢迎的专栏，内容涵盖Java内部机制，面向对象设计，还有Jini。

Bill从Jini诞生伊始就活跃于Jini社区，他曾领导Jini社区的ServiceUI项目，而其中的ServiceUI API已经变成了联系用户界面和Jini服务之间的事实标准。

Bill还是ScalaTest（Scala和Java开发的开源测试工具）的首席开发者（lead developer）和设计者。

书籍目录

图表清单前言致谢导读第1章 可伸展的语言第2章 Scala入门初探第3章 Scala入门再探第4章 类和对象第5章 基本类型和操作第6章 函数式对象第7章 内建控制结构第8章 函数和闭包第9章 控制抽象第10章 组合与继承第11章 Scala的层级第12章 特质第13章 包和引用第14章 断言和单元测试第15章 样本类和模式匹配第16章 使用列表第17章 集合类型第18章 有状态的对象第19章 类型参数化第20章 Abstract Members抽象成员第21章 隐式转换和参数第22章 实现列表第23章 重访for表达式第24章 Extractors抽取器第25章 注解第26章 使用XML第27章 使用对象的模块化编程第28章 对象相等性第29章 结合Scala和Java第30章 Actor和并发第31章 连结符解析第32章 GUI编程第33章 试算表附录A Unix和Windows的Scala脚本术语表参考文献关于作者索引

<<Scala编程>>

章节摘录

当然也可以直接使用Java的类库。
但结果却不容乐观，因为尽管Java允许创建新的类，但这些类总感觉不像原生的语言支持那么方便。
前面的例子演示了如何在Scala中增加新的类型，使得它们用起来方便得像内建类型一样。
同样的扩展理念也应用在了控制结构上。
这种扩展可以由Scala的“基于actor”的并发编程API阐明。
随着近年来多核处理器的激增，为了获取可接受的性能，应用中必须运用更多的并行机制。
这常常就意味着须重写代码以使计算分布到若干并发线程上。
不幸的是，创建可依赖的多线程程序经实践证明非常具有挑战性。
Java的线程模型是围绕着共享内存和锁建立的，当系统在规模和复杂度都不断变大的时候，这种模型会越发变得难以理解。
很难说程序里面没有资源竞争或潜藏的死锁--有些东西不是能经测试检验出的，或许只在投入生产后才会表现出来。
目前可以认为比较安全的可选方案是消息传递架构，例如在Erlang编程语言中应用的“actor”方案。
Java带了一个丰富的、基于线程的并发库。
Scala也可以像使用其他Java API那样用它编程。
不过，Scala还提供了一个实质上实现了Erlang的actor模型的附加库。

.....

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>