



大数据科学丛书
BIG DATA SCIENCE



Scala

语言基础与开发实战

王家林 等 编著

大数据科学丛书

Scala 语言基础与开发实战

王家林 管祥青 等编著



机械工业出版社

本书分为基础篇、中级篇、高级篇及分布式框架四大部分，从 Scala 零基础入门，步步深入，引导读者由浅入深地学习 Scala 及其应用。本书从手把手指引读者搭建 Scala 语言开发环境开始，详细介绍了 Scala 的语法基础，以代码实例形式分别讲解了 Scala 面向对象开发及函数式编程；在此基础上进一步深入讲解了 Scala 的中高级语法特性，包括模式匹配、集合、类型参数、高级类型、隐式转化及各语法特性在 Spark 源码中的应用解析，并引出 Scala 的 Actor 模型及其应用详解。本书还详细介绍了以 Scala 为基础的两大框架——Akka 和 Kafka。

本书每章开始均有重点介绍，以引导读者有目的、有重点地阅读或查阅。另外，针对不同语法特性的源码及应用解析是本书的另一大特点。

本书适合具备一定编程语言基础、对大数据开发有兴趣的在校学生，同时，对有面向对象编程或函数式编程经验的人员，本书也可以作为开发实例的参考书籍。

图书在版编目 (CIP) 数据

Scala 语言基础与开发实战/王家林等编著. —北京: 机械工业出版社, 2016. 6
(大数据科学丛书)

ISBN 978-7-111-54169-1

I. ①S… II. ①王… III. ①JAVA 语言 - 程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字 (2016) 第 151317 号

机械工业出版社 (北京市百万庄大街 22 号 邮政编码 100037)

策划编辑: 王 斌 责任编辑: 王 斌

责任校对: 张艳霞

印刷 (装订)

2016 年 7 月第 1 版 · 第 1 次印刷

184mm × 260mm · 26.5 印张 · 640 千字

0001 - 3000 册

标准书号: ISBN 978-7-111-54169-1

定价: 69.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

电话服务

网络服务

社服务中心: (010) 88361066

门户网: <http://www.cmpbook.com>

销 售 一 部: (010) 68326294

教材网: <http://www.cmpedu.com>

销 售 二 部: (010) 88379649

读者购书热线: (010) 88379203

封面无防伪标均为盗版

前 言



采用 Scala 语言编写实现的，大数据领域最火爆的计算框架 Spark（其实 Spark 在 Apache 下的数据处理领域也是最火爆的计算框架），正在以迅雷不及掩耳之势快速发展。很少有一门语言能够像 Scala 这样，因其作为大数据框架 Spark 的核心和首选开发语言而爆发式地普及起来。Spark 本身起源于 2009 年，是美国加州大学伯克利分校 AMP 实验室的一个研究性项目，于 2010 年开源，在 2014、2015 年大数据领域软件排名中，Spark 都以绝对优势遥遥领先！虽然基于 Spark 平台可以采用 Scala、Java、Python、R 等 4 种语言开发，但据 Spark 官方统计，2014 年和 2015 年全世界范围内基于 Spark 开发采用最多的语言一直都是 Scala。另外，在大数据领域越来越多的其他技术框架，例如 Kafka 等也都把 Scala 作为实现和开发语言。因此，为了打好大数据领域学习的基础，本书面向广大 Scala 爱好者和大数据开发者，以实战为主导，并用实战与理论相结合的方式来帮助读者学习 Scala 语言。

从 2012 年美国政府的“大数据研发计划”，到 2015 年我国国务院发布的《促进大数据发展行动纲要》，可以说，大数据已经迎来了它的黄金时代。本书紧跟时代潮流，除了讲解 Scala 语言之外，还额外挑选了当前在大数据领域中应用非常广泛 Akka 和 Kafka 两大框架进行讲解，并且详细讲解了 Scala 语言在其中的应用。Akka 是一个在 JVM 上构建高并发、分布式和可快速恢复的消息驱动应用的工具包；Kafka 是高产出的分布式消息系统，它实现了生产者和消费者之间的无缝连接，实现了处理速度快、高可扩展性的分布式实时系统。

本书编写的主线是以 Scala 实战实例为主导，由浅入深，从 Scala 的基础篇、中级篇直至高级篇，对 Scala 各个知识点加以详细分析并给出相应的实例及解析。然后更进一步地引入分布式框架篇，针对当前大数据领域使用非常广泛的大分布式框架 Akka 和 Kafka，通过介绍 Scala 语言在开发分布式框架时的实战案例，为读者进一步学习大数据领域各个框架打好基础。

参与本书编写的有王家林、段智华、管祥青、徐奔、张敏、徐香玉等。

本书能顺利出版，离不开出版社编辑们的大力支持与帮助，在此表示诚挚的感谢。

非常感谢本书的技术审核徐香玉为审核本书技术相关内容所做出的努力。

在阅读本书的过程中，若发现任何问题或有任何疑问，可以加入本书的阅读群（QQ：418110145）提出讨论，会有专人帮忙答疑。同时，该群中也会提供本书所用实例代码。

如果读者想要了解或者学习更多大数据相关技术，可以通过以下方式参与互动交流：

关注 DT 大数据梦工厂微信公众号：DT_Spark 及 QQ 群：163728659，或者通过扫描下方二维码咨询，也可以通过 YY 客户端登录 68917580 永久频道直接体验。

我的新浪微博是 <http://weibo.com/ilovepains/>，也欢迎大家在微博上进行互动。
由于时间仓促，书中难免存在不妥之处，请读者谅解，并提出宝贵意见。



王家林
2016. 1. 18 日于北京

目 录

前言

基 础 篇

第 1 章 Scala 零基础入门	3
1.1 Scala 概述	3
1.2 Windows 及 Linux 下 Scala 运行环境安装配置	4
1.2.1 软件工具准备	4
1.2.2 Windows 环境下的 Scala 安装	6
1.2.3 Linux 环境下的 Scala 安装	10
1.2.4 Linux 环境下的 Hadoop 安装与配置	13
1.2.5 Linux 环境下的 Spark 安装与配置	23
1.3 Scala 开发环境搭建和 HelloWorld 实例	28
1.3.1 Scala 集成开发工具的安装	28
1.3.2 HelloWorld 编程实例	30
1.3.3 Worksheet 的使用	36
1.4 变量的使用	37
1.4.1 Scala 解释器中的变量示例	37
1.4.2 val 变量的定义	38
1.4.3 var 变量的定义	39
1.4.4 var 变量与 val 变量的使用比较	39
1.5 函数的定义、流程控制、异常处理	41
1.5.1 函数的定义	41
1.5.2 流程控制 (if、while、for)	43
1.5.3 异常处理	52
1.6 Tuple、Array、Map 与文件操作	54
1.6.1 Tuple 元组	54
1.6.2 Array 数组	56
1.6.3 文件操作	59
1.6.4 Map 映射	62
1.7 Scala 中的 apply 方法	63
1.7.1 Object 中的 apply	63
1.7.2 Class 中的 apply	64
1.7.3 Array 数组的 apply 实现	65



1.8	小结	66
第2章	Scala 面向对象编程开发	67
2.1	类的定义及属性	67
2.1.1	类定义	67
2.1.2	带有 getter 和 setter 的属性	68
2.2	主构造器、私有构造器、构造器重载	70
2.2.1	构造器重载之辅助构造器	70
2.2.2	主构造器	71
2.2.3	不同访问权限的构造器	72
2.3	内部类和外部类	73
2.4	单例对象、伴生对象	77
2.5	继承：超类的构造、重写字段、重写方法	78
2.5.1	超类的构造	79
2.5.2	重写字段	80
2.5.3	重写方法	80
2.6	抽象类、抽象字段、抽象方法	82
2.6.1	抽象类	82
2.6.2	抽象字段	82
2.6.3	抽象方法	82
2.7	trait 特质	83
2.7.1	作为接口使用的 trait	84
2.7.2	在对象中混入 trait	85
2.7.3	trait 深入解析	86
2.8	多重继承、多重继承构造器执行顺序及 AOP 实现	88
2.8.1	多重继承	88
2.8.2	多重继承构造器执行顺序	89
2.8.3	AOP 实现	89
2.9	包的定义、包对象、包的引用、包的隐式引用	91
2.9.1	包的定义	91
2.9.2	包对象	91
2.9.3	包的引用	92
2.9.4	包的隐式引用	92
2.10	包、类、对象、成员、伴生类、伴生对象访问权限	92
2.10.1	包、类、对象、成员访问权限	92
2.10.2	伴生类、伴生对象访问权限	93
2.11	小结	94
第3章	Scala 高阶函数	95
3.1	匿名函数	95
3.2	偏应用函数	96

3.3	闭包	98
3.4	SAM 转换	100
3.5	Curring 函数	102
3.6	高阶函数	103
3.7	高阶函数在 Spark 中的应用	107
3.8	小结	109

中 级 篇

第 4 章	Scala 模式匹配	113
4.1	模式匹配简介	113
4.2	模式匹配类型	115
4.2.1	常量模式	116
4.2.2	变量模式	116
4.2.3	构造器模式	117
4.2.4	序列 (Sequence) 模式	118
4.2.5	元组 (Tuple) 模式	119
4.2.6	类型模式	120
4.2.7	变量绑定模式	121
4.3	模式匹配与 Case Class	122
4.3.1	构造器模式匹配原理	122
4.3.2	序列模式匹配原理	125
4.3.3	Sealed Class 在模式匹配中的应用	126
4.4	模式匹配应用实例	127
4.4.1	for 循环控制结构中的模式匹配	127
4.4.2	正则表达式中的模式匹配	128
4.4.3	异常处理中的模式匹配	132
4.4.4	Spark 源码中的模式匹配使用	133
4.5	小结	136
第 5 章	Scala 集合	137
5.1	可变集合与不可变集合 (Collection)	137
5.1.1	集合的概述	137
5.1.2	集合的相关操作	141
5.1.3	集合的操作示例	145
5.2	序列 (Seq)	151
5.2.1	序列的概述	151
5.2.2	序列的相关操作	152
5.2.3	序列的操作示例	154
5.3	列表 (List)	158



5.3.1	列表的概述	158
5.3.2	列表的相关操作	158
5.3.3	列表的操作示例	159
5.4	集 (Set)	161
5.4.1	集的概念	161
5.4.2	集的相关操作	162
5.4.3	集的操作示例	164
5.5	映射 (Map)	165
5.5.1	映射的概念	165
5.5.2	映射的相关操作	166
5.5.3	映射的操作示例	168
5.6	迭代器 (Iterator)	172
5.6.1	迭代器的概念	172
5.6.2	迭代器的相关操作	173
5.6.3	迭代器的操作示例	176
5.7	集合的架构	185
5.8	小结	189

高级篇

第6章	Scala 类型参数	193
6.1	泛型	193
6.1.1	泛型的概述	193
6.1.2	泛型的操作示例	194
6.2	界定	195
6.2.1	上下界定	196
6.2.2	视图界定	196
6.2.3	上下文界定	196
6.2.4	多重界定	196
6.2.5	界定的操作示例	197
6.3	类型约束	204
6.3.1	类型约束的概述	204
6.3.2	类型约束的操作示例	205
6.4	类型系统	205
6.4.1	类型系统的概述	205
6.4.2	类型系统的操作示例	206
6.5	型变 Variance	207
6.5.1	协变	208
6.5.2	逆变	208

6.5.3 协变与逆变的操作示例	208
6.6 结合 Spark 源码说明 Scala 类型参数的使用	210
6.7 小结	212
第7章 Scala 高级类型	213
7.1 单例类型	213
7.1.1 单例类型概述	213
7.1.2 单例类型示例	214
7.2 类型别名	217
7.2.1 类型别名概述	217
7.2.2 类型别名示例	217
7.3 自身类型	218
7.3.1 自身类型概述	218
7.3.2 自身类型示例	219
7.4 中置类型	219
7.4.1 中置类型概述	219
7.4.2 中置类型示例	219
7.5 类型投影	221
7.5.1 类型投影概述	221
7.5.2 类型投影实例	221
7.6 结构类型	223
7.6.1 结构类型概述	223
7.6.2 结构类型示例	224
7.7 复合类型	226
7.7.1 复合类型概述	226
7.7.2 复合类型示例	226
7.8 存在类型	227
7.8.1 存在类型概述	227
7.8.2 存在类型示例	227
7.9 函数类型	229
7.9.1 函数类型概述	229
7.9.2 函数类型示例	229
7.10 抽象类型	230
7.10.1 抽象类型概述	230
7.10.2 抽象类型实例	230
7.11 Spark 源码中的高级类型使用	231
7.12 小结	233
第8章 Scala 隐式转换	234
8.1 隐式转换函数	234
8.1.1 隐式转换函数的定义	234



8.1.2	隐式转换函数的功能	235
8.2	隐式类与隐式对象	236
8.2.1	隐式类	236
8.2.2	隐式参数与隐式值	237
8.3	类型证明中的隐式转换	239
8.3.1	类型证明的定义	239
8.3.2	类型证明使用实例	239
8.4	上下文界定、视图界定中的隐式转换	241
8.4.1	Ordering 与 Ordered 特质	241
8.4.2	视图界定中的隐式转换	245
8.4.3	上下文界定中的隐式转换	246
8.5	隐式转换规则	248
8.5.1	发生隐式转换的条件	248
8.5.2	不会发生隐式转换的条件	249
8.6	Spark 源码中的隐式转换使用	252
8.6.1	隐式转换函数	252
8.6.2	隐式类	253
8.6.3	隐式参数	253
8.7	小结	253
第9章	Scala 并发编程	255
9.1	Scala 的 Actor 模型简介	256
9.2	Scala Actor 的构建方式	256
9.2.1	继承 Actor 类	256
9.2.2	Actor 工具方法	257
9.3	Actor 的生命周期	258
9.3.1	start 方法的幂等性	258
9.3.2	Actor 的不同状态	259
9.4	Actor 之间的通信	260
9.4.1	Actor 之间发送消息	260
9.4.2	Actor 接收消息	260
9.5	使用 react 重用线程提升性能	262
9.6	Channel 通道	263
9.6.1	OutputChannel	264
9.6.2	InputChannel	264
9.6.3	创建和共享 channel	264
9.7	同步和 Future	266
9.8	Scala 并发编程实例	266
9.8.1	Scala Actor 并发编程	267
9.8.2	ExecutorService 并发编程	268

9.9 小结	269
--------------	-----

分布式框架篇

第 10 章 Akka 的设计理念	273
10.1 Akka 框架模型	274
10.2 创建 Actor	275
10.2.1 通过实现 akka.actor.Actor 来创建 Actor 类	275
10.2.2 使用非缺省构造方法创建 Actor	277
10.2.3 创建匿名 Actor	278
10.3 Actor API	280
10.3.1 Actor trait 基本接口	280
10.3.2 使用 DeathWatch 进行生命周期监控	281
10.3.3 Hook 函数的调用	282
10.3.4 查找 Actor	283
10.3.5 消息的不可变性	283
10.3.6 发送消息	283
10.3.7 转发消息	287
10.3.8 接收消息	287
10.3.9 回应消息	287
10.3.10 终止 Actor	288
10.3.11 Become/Unbecome	289
10.3.12 杀死 Actor	290
10.4 不同类型的 Actor	290
10.4.1 方法派发语义	294
10.4.2 终止有类型 Actor	295
10.5 小结	295
第 11 章 Akka 核心组件及核心特性剖析	296
11.1 Dispatchers 和 Routers	296
11.1.1 为 Actor 指定派发器	297
11.1.2 派发器的类型	298
11.1.3 邮箱	300
11.1.4 Routers	300
11.1.5 路由的使用	301
11.1.6 远程部署 router	302
11.2 Supervision 和 Monitoring	302
11.2.1 Supervision	302
11.2.2 Monitoring	305
11.3 Akka 中的事务	306



11.3.1	STM	306
11.3.2	使用STM事务	308
11.3.3	读取Agent事务中的数据	309
11.3.4	更新Agent事务中的数据	311
11.3.5	Actor中的事务	313
11.3.6	创建Transactor	316
11.4	小结	318
第12章	Akka 程序设计实践	319
12.1	Akka 的配置、日志及部署	319
12.1.1	Akka 中配置文件的读写	319
12.1.2	Akka 中日志配置	323
12.1.3	Akka 部署及应用场景	324
12.2	使用Akka框架实现单词统计	324
12.3	分布式Akka环境搭建	329
12.4	使用Akka微内核部署应用	333
12.5	Akka框架在Spark中的运用	334
12.6	小结	338
第13章	Kafka 设计理念与基本架构	339
13.1	Kafka 产生的背景	339
13.2	消息队列系统	340
13.2.1	概述	340
13.2.2	常用的消息队列系统对比	341
13.2.3	Kafka 特点及特性	342
13.2.4	Kafka 系统应用场景	342
13.3	Kafka 设计理念	343
13.3.1	专业术语解析	343
13.3.2	消息存储与缓存设计	344
13.3.3	消费者与生产者模型	344
13.3.4	Push 与 Pull 机制	345
13.3.5	镜像机制	346
13.4	Kafka 整体架构	346
13.4.1	Kafka 基本组成结构	346
13.4.2	Kafka 工作流程	347
13.5	Kafka 性能分析及优化	348
13.6	Kafka 未来研究方向	350
13.7	小结	352
第14章	Kafka 核心组件及核心特性剖析	353
14.1	Kafka 核心组件剖析	353
14.1.1	Producers	353

14.1.2	Consumers	354
14.1.3	Low Level Consumer	355
14.1.4	High Level Consumer	356
14.2	Kafka 核心特性剖析	357
14.2.1	Topic、Partitions	357
14.2.2	Replication 和 Leader Election	359
14.2.3	Consumer Rebalance	361
14.2.4	消息传送机制	363
14.2.5	Kafka 的可靠性	364
14.2.6	Kafka 的高效性	364
14.3	Kafka 即将发布版本核心组件及特性剖析	365
14.3.1	重新设计的 Consumer	365
14.3.2	Coordinator Rebalance	366
14.4	小结	370
第 15 章 Kafka 应用实践		371
15.1	Kafka 开发环境搭建及运行环境部署	371
15.1.1	Kafka 开发环境配置	371
15.1.2	Kafka 运行环境安装与部署	374
15.2	基于 Kafka 客户端开发	381
15.2.1	消息生产者 (Producer) 设计	382
15.2.2	消息消费者 (Consumer) 设计	384
15.2.3	Kafka 消费者与生产者配置	390
15.3	Spark Streaming 整合 Kafka	392
15.3.1	基本架构设计流程	392
15.3.2	消息消费者 (Consumer) 设计——基于 Receiver 方法	393
15.3.3	消息消费者 (Consumer) 设计——基于 No Receiver 方法	398
15.3.4	消息生产者 (Producer) 设计	401
15.4	小结	403
附录 Kafka 集群 server.properties 配置文档		404
参考文献		407



基础篇

作为本书的开篇，本篇将从零开始引导读者由浅入深，循序渐进，依照图例的指引一步一步地真正动手搭建 Scala 编程环境及大数据实验环境。本篇由最基本的 Scala 语法开始讲起，让读者在理解语法的基础上动手编程实验，逐步学会 Scala 基本编程，为后续学习 Scala 中高级篇的高级语法，以及基于 Scala 的三大主流分布式大数据框架打下坚实的基础。

本篇共 3 章，第 1 章介绍了 Scala 零基础入门的环境搭建及 Scala 的基础语法知识；第 2 章基于基础语法，以实例分析的形式详解 Scala 面向对象的编程开发；第 3 章讲解了 Scala 的函数式编程，并以实例分析的形式详解其中的高阶函数应用。



第 1 章 Scala 零基础入门



本章将通过详细的图例，清晰地介绍了 Scala 实验环境的搭建（包括 Scala 安装、Hadoop 安装及 MapReduce 词频统计、Spark 安装、Scala IDE 安装），Scala 的基础知识（变量、函数、流程控制、异常处理），Scala Tuple、Array、Map 与文件操作，以及 Scala apply 应用等内容。

Section

1.1

Scala 概述

曾经有人问 Java 的创始人高斯林（James Gosling）这样一个问题：“除了 Java 语言以外，您现在还使用 JVM（Java Virtual Machine，Java 虚拟机）上的哪种编程语言？”他毫不犹豫地说是 Scala。Scala 是一门集成面向对象和函数式特性的语言，它的创始人是 Martin Odersky。Martin Odersky 也是 Javac 编译器的作者，他曾将泛型引入到 Java 语言中。因为 Java 是强约束的语言，出于 Java 本身设计上的局限性，诸如函数式编程等特性在 Java 8 之前都没有实现，因此 Martin Odersky 设想基于 JVM 和 Java 类库，创建一个比 Java 更高级的语言。他于 2001 年开始基于 Funnel 语言设计 Scala，在 2003 年年底 Scala 发布正式（基于 Java 平台）版本。

截至 2016 年 1 月，Scala 2.12 系列的最新版本是 Scala 2.12.0 - M3，Scala 2.11 系列的最新版本是 Scala 2.11.7。2016 年，Scala 2.11 系列预计于一季度实现 Scala 2.11.8 版本，在三季度实现 Scala 2.11.9 版本；Scala 2.12 系列预计在二季度重点实现 Scala 2.12.0 - RC1 版本，在 2.12 系列的两个里程碑 Scala 2.12.0 - M4 及 Scala 2.12.0 - M5 Scala 版本中，Scala 编译器将会有很大的变化，包括 Lambda 表达式和使用新的编码方式，充分利用 Java 8 的新特性，以及 Scala 的优化等。

Scala 是一个运行在 Java 虚拟机环境（JVM）中将面向对象编程和函数式编程的最佳特性完美结合起来具有强大类型系统的，优雅、简洁的语言。

Scala 语言的优势如下：

1) Scala 是面向对象编程语言：在 Scala 中，所有预先定义的类型是对象，所有用户定义的类型也是对象，每个操作都是方法调用。

2) Scala 是函数式编程语言：在 Scala 中，函数是“头等公民”，函数可以被当成参数传递给其他函数，也可以当成结果返回或保存为变量。Scala 编写的代码简洁、优雅，同时又具备丰富的表达能力，在实际开发过程中推荐使用大量不可变的变量及无副作用的函数进行代码编写；Scala 支持匿名函数、高阶函数、柯里化函数、函数嵌套的应用；相对于指令式编程语言而言，Scala 可以使用更少的代码量来实现相同的功能，充分体现 Scala 语言简洁、精炼、优雅的特点。