

# Lucene实战



[Lucene实战\\_下载链接1](#)

著者:Michael McCandless

出版者:人民邮电出版社

出版时间:2011-6-1

装帧:平装

isbn:9787115251770

Michael McCandless的《Lucene实战(第2版)》基于Apache的Lucene 3.0，从Lucene核心、Lucene应用、案例分析3个方面详细系统地介绍了Lucene，包括认识Lucene、建立索引、为应用程序添加搜索功能、高级搜索技术、扩展搜索、使用tika提取文本、Lucene的高级扩展、使用其他编程语言访问Lucene、Lucene管理和性能

调优等内容，最后还提供了三大经典成功案例，为读者展示了一个奇妙的搜索世界。  
《Lucene实战(第2版)》适合于已具有一定Java编程基本的读者，以及希望能够把强大的搜索功能添加到自己的应用程序中的开发人员。本书对于从事搜索引擎工作的工程技术人员，以及在Java平台上进行各类软件开发的人员和编程爱好者，具有很好的学习参考价值。

作者介绍:

Michael McCandless是Lucene PMC的成员和负责人。他有10年以上有关构建搜索引擎的相关经验。

Erik Hatcher和Otis Gospodretic是本书第1版的作者，长期以来，为Lucene、Solr、Mahout和其他基于Lucene的项目做出了贡献。

Erik Hatcher和Otis Gospodretic是本书第1版的作者，长期以来，为Lucene、Solr、Mahout和其他基于Lucene的项目做出了贡献。

目录: 第1部分 lucene核心

第1章 初识lucene

1.1 应对信息爆炸

1.2 lucene是什么

1.2.1 lucene能做些什么

1.2.2 lucene的历史

1.3 lucene和搜索程序组件

1.3.1 索引组件

1.3.2 搜索组件

1.3.3 搜索程序的其他模块

1.3.4 lucene与应用程序的整合点

1.4 lucene实战：程序示例

1.4.1 建立索引

1.4.2 搜索索引

1.5 理解索引过程的核心类

1.5.1 indexwriter

1.5.2 directory

1.5.3 analyzer

1.5.4 document

1.5.5 field

1.6 理解搜索过程的核心类

1.6.1 indexsearcher

1.6.2 term

1.6.3 query

1.6.4 termquery

1.6.5 topdocs

1.7 小结

第2章 构建索引

2.1 lucene如何对搜索内容进行建模

2.1.1 文档和域

2.1.2 灵活的架构

2.1.3 反向规格化 (denormalization)

2.2 理解索引过程

- 2.2.1 提取文本和创建文档
- 2.2.2 分析文档
- 2.2.3 向索引添加文档
- 2.3 基本索引操作
  - 2.3.1 向索引添加文档
  - 2.3.2 删除索引中的文档
  - 2.3.3 更新索引中的文档
- 2.4 域选项
  - 2.4.1 域索引选项
  - 2.4.2 域存储选项
  - 2.4.3 域的项向量选项
  - 2.4.4 reader、tokenstream和byte[]域值
  - 2.4.5 域选项组合
  - 2.4.6 域排序选项
  - 2.4.7 多值域
- 2.5 对文档和域进行加权操作
  - 2.5.1 文档加权操作
  - 2.5.2 域加权操作
  - 2.5.3 加权基准 (norms)
- 2.6 索引数字、日期和时间
  - 2.6.1 索引数字
  - 2.6.2 索引日期和时间
- 2.7 域截取 (field truncation)
- 2.8 近实时搜索 (near-real-time search)
- 2.9 优化索引
- 2.10 其他directory子类
- 2.11 并发、线程安全及锁机制
  - 2.11.1 线程安全和多虚拟机安全
  - 2.11.2 通过远程文件系统访问索引
  - 2.11.3 索引锁机制
- 2.12 调试索引
- 2.13 高级索引概念
  - 2.13.1 用indexreader删除文档
  - 2.13.2 回收被删除文档所使用过的磁盘空间
  - 2.13.3 缓冲和刷新
  - 2.13.4 索引提交
  - 2.13.5 acid事务和索引连续性
  - 2.13.6 合并段
- 2.14 小结
- 第3章 为应用程序添加搜索功能
  - 3.1 实现简单的搜索功能
    - 3.1.1 对特定项的搜索
    - 3.1.2 解析用户输入的查询表达式：queryparser
  - 3.2 使用indexsearcher类
    - 3.2.1 创建indexsearcher类
    - 3.2.2 实现搜索功能
    - 3.2.3 使用topdocs类
    - 3.2.4 搜索结果分页
    - 3.2.5 近实时搜索
  - 3.3 理解lucene的评分机制
    - 3.3.1 lucene如何评分
    - 3.3.2 使用explain () 理解搜索结果评分
  - 3.4 lucene的多样化查询
    - 3.4.1 通过项进行搜索：termquery类

- 3.4.2 在指定的项范围内搜索：termrangequery类
- 3.4.3 在指定的数字范围内搜索：numericrangequery类
- 3.4.4 通过字符串搜索：prefixquery类
- 3.4.5 组合查询：booleanquery类
- 3.4.6 通过短语搜索：phrasequery类
- 3.4.7 通配符查询：wildcardquery类
- 3.4.8 搜索类似项：fuzzyquery类
- 3.4.9 匹配所有文档：matchalldocsquery类
- 3.5 解析查询表达式：queryparser
  - 3.5.1 query.toString方法
  - 3.5.2 termquery
  - 3.5.3 项范围查询
  - 3.5.4 数值范围搜索和日期范围搜索
  - 3.5.5 前缀查询和通配符查询
  - 3.5.6 布尔操作符
  - 3.5.7 短语查询
  - 3.5.8 模糊查询
  - 3.5.9 matchalldocsquery
  - 3.5.10 分组查询
  - 3.5.11 域选择
  - 3.5.12 为子查询设置加权
  - 3.5.13 是否一定要使用queryparse
- 3.6 小结

第4章 lucene的分析过程

- 4.1 使用分析器
  - 4.1.1 索引过程中的分析
  - 4.1.2 queryparser分析
  - 4.1.3 解析vs分析：分析器何时不再适用
- 4.2 剖析分析器
  - 4.2.1 语汇单元的组成
  - 4.2.2 语汇单元流揭秘
  - 4.2.3 观察分析器
  - 4.2.4 语汇单元过滤器：过滤顺序的重要性
- 4.3 使用内置分析器
  - 4.3.1 stopanalyzer
  - 4.3.2 standardanalyzer
  - 4.3.3 应当采用哪种核心分析器
- 4.4 近音词查询
- 4.5 同义词、别名和其他表示相同意义的词
  - 4.5.1 创建synonymanalyzer
  - 4.5.2 显示语汇单元的位置
- 4.6 词干分析
  - 4.6.1 stopfilter保留空位
  - 4.6.2 合并词干操作和停用词移除操作
- 4.7 域分析
  - 4.7.1 多值域分析
  - 4.7.2 特定域分析
  - 4.7.3 搜索未被分析的域
- 4.8 语言分析
  - 4.8.1 unicode与字符编码
  - 4.8.2 非英语语种分析
  - 4.8.3 字符规范化处理
  - 4.8.4 亚洲语种分析
  - 4.8.5 有关非英语语种分析的其他问题

#### 4.9 nutch分析

#### 4.10 小结

### 第5章 高级搜索技术

#### 5.1 lucene域缓存

##### 5.1.1 为所有文档加载域值

##### 5.1.2 段对应的reader

#### 5.2 对搜索结果进行排序

##### 5.2.1 根据域值进行排序

##### 5.2.2 按照相关性进行排序

##### 5.2.3 按照索引顺序进行排序

##### 5.2.4 通过域进行排序

##### 5.2.5 倒排序

##### 5.2.6 通过多个域进行排序

##### 5.2.7 为排序域选择类型

##### 5.2.8 使用非默认的locale方式进行排序

#### 5.3 使用multiphrasequery

#### 5.4 针对多个域的一次性查询

#### 5.5 跨度查询

##### 5.5.1 跨度查询的构建模块：spantermquery

##### 5.5.2 在域的起点查找跨度

##### 5.5.3 彼此相邻的跨度

##### 5.5.4 在匹配结果中排除重叠的跨度

##### 5.5.5 spanorquery类

##### 5.5.6 spanquery类和queryparser类

#### 5.6 搜索过滤

##### 5.6.1 termrangefilter

##### 5.6.2 numericrangefilter

##### 5.6.3 fieldcachefilter

##### 5.6.4 特定项过滤

##### 5.6.5 使用querywrapperfilter类

##### 5.6.6 使用spanqueryfilter类

##### 5.6.7 安全过滤器

##### 5.6.8 使用booleanquery类进行过滤

##### 5.6.9 prefixfilter

##### 5.6.10 缓存过滤结果

##### 5.6.11 将filter封装成query

##### 5.6.12 对过滤器进行过滤

##### 5.6.13 非lucene内置的过滤器

#### 5.7 使用功能查询实现自定义评分

##### 5.7.1 功能查询的相关类

##### 5.7.2 使用功能查询对最近修改过的文档进行加权

#### 5.8 针对多索引的搜索

##### 5.8.1 使用multisearch类

##### 5.8.2 使用parallelmultisearcher进行多线程搜索

#### 5.9 使用项向量

##### 5.9.1 查找相似书籍

##### 5.9.2 它属于哪个类别

##### 5.9.3 termvectormapper类

##### 5.10 使用fieldselector加载域

##### 5.11 停止较慢的搜索

#### 5.12 小结

### 第6章 扩展搜索

#### 6.1 使用自定义排序方法

##### 6.1.1 针对地理位置排序方式进行文档索引

- 6.1.2 实现自定义的地理位置排序方式
- 6.1.3 访问自定义排序中的值
- 6.2 开发自定义的collector
  - 6.2.1 collector基类
  - 6.2.2 自定义collector: booklinkcollector
  - 6.2.3 alldoccollector类
- 6.3 扩展queryparser类
  - 6.3.1 自定义queryparser的行为
  - 6.3.2 禁用模糊查询和通配符查询
  - 6.3.3 处理数值域的范围查询
  - 6.3.4 处理日期范围
  - 6.3.5 对已排序短语进行查询
- 6.4 自定义过滤器
  - 6.4.1 实现自定义过滤器
  - 6.4.2 搜索期间使用自定义过滤器
  - 6.4.3 另一种选择: filterquery类
- 6.5 有效载荷 (payloads)
  - 6.5.1 分析期间生成有效载荷
  - 6.5.2 搜索期间使用有效载荷
  - 6.5.3 有效载荷和跨度查询
  - 6.5.4 通过termpositions来检索有效载荷
- 6.6 小结

## 第2部分 lucene应用

## 第7章 使用tika提取文本

- 7.1 tika是什么
- 7.2 tika的逻辑设计和api
- 7.3 安装tika
- 7.4 tika的内置文本提取工具
- 7.5 编程实现文本提取
  - 7.5.1 索引lucene文档
  - 7.5.2 tika工具类
  - 7.5.3 选择自定义分析器
- 7.6 tika的局限
- 7.7 索引自定义的xml文件
  - 7.7.1 使用sax进行解析
  - 7.7.2 使用apache commons digester进行解析和索引
- 7.8 其他选择
- 7.9 小结

## 第8章 lucene基本扩展

- 8.1 luke: lucene的索引工具箱
  - 8.1.1 overview标签页: 索引的全局视图
  - 8.1.2 浏览文档
  - 8.1.3 使用queryparser进行搜索
  - 8.1.4 files and plugins标签页
- 8.2 分析器、语汇单元器和语汇单元过滤器
  - 8.2.1 snowballanalyzer
  - 8.2.2 ngram过滤器
  - 8.2.3 shingle过滤器
  - 8.2.4 获取捐赠分析器
- 8.3 高亮显示查询项
  - 8.3.1 高亮显示模块
  - 8.3.2 独立的高亮显示示例
  - 8.3.3 使用css进行高亮显示处理
  - 8.3.4 高亮显示搜索结果

- 8.4 fastvector highlighter类
- 8.5 拼写检查
  - 8.5.1 生成提示列表
  - 8.5.2 选择最佳提示
  - 8.5.3 向用户展示搜索结果
  - 8.5.4 一些加强拼写检查的考虑
- 8.6 引人注目的查询扩展功能
  - 8.6.1 morelikethis
  - 8.6.2 fuzzylikethisquery
  - 8.6.3 boostingquery
  - 8.6.4 termsfilter
  - 8.6.5 duplicatefilter
  - 8.6.6 regexquery
- 8.7 构建软件捐赠模块 (contrib module)
  - 8.7.1 源代码获取方式
  - 8.7.2 contrib目录的ant插件
- 8.8 小结
- 第9章 lucene高级扩展
  - 9.1 链式过滤器
  - 9.2 使用berkeley db存储索引
  - 9.3 wordnet同义词
    - 9.3.1 建立同义词索引
    - 9.3.2 将wordnet同义词链接到分析器中
  - 9.4 基于内存的快速索引
  - 9.5 xml queryparser: 超出 “one box” 的搜索接口
    - 9.5.1 使用xmlqueryparser
    - 9.5.2 扩展xml查询语法
  - 9.6 外围查询语言
  - 9.7 spatial lucene
    - 9.7.1 索引空间数据
    - 9.7.2 搜索空间数据
    - 9.7.3 spatial lucene的性能特点
  - 9.8 远程进行多索引搜索
  - 9.9 灵活的queryparser
  - 9.10 其他内容
  - 9.11 小结
- 第10章 其他编程语言使用lucene
  - 10.1 移植入门
    - 10.1.1 移植取舍
    - 10.1.2 选择合适的移植版本
  - 10.2 clucene (c++)
    - 10.2.1 移植目的
    - 10.2.2 api和索引兼容
    - 10.2.3 支持的平台
    - 10.2.4 当前情况以及未来展望
  - 10.3 lucene.net (c#和其他.net编程语言)
    - 10.3.1 api兼容
    - 10.3.2 索引兼容
  - 10.4 kinosearch和lucy (perl)
    - 10.4.1 kinosearch
    - 10.4.2 lucy
    - 10.4.3 其他perl选项
  - 10.5 ferret (ruby)
  - 10.6 php

- 10.6.1 zend framework
- 10.6.2 php bridge
- 10.7 pylucene (python)
  - 10.7.1 api兼容
  - 10.7.2 其他python选项
- 10.8 solr (包含多种编程语言)
- 10.9 小结
- 第11章 lucene管理和性能调优
  - 11.1 性能调优
    - 11.1.1 简单的性能调优步骤
    - 11.1.2 测试方法
    - 11.1.3 索引-搜索时延调优
    - 11.1.4 索引操作吞吐量调优
    - 11.1.5 搜索时延和搜索吞吐量调优
  - 11.2 多线程和并行处理
    - 11.2.1 使用多线程进行索引操作
    - 11.2.2 使用多线程进行搜索操作
  - 11.3 资源消耗管理
    - 11.3.1 磁盘空间管理
    - 11.3.2 文件描述符管理
    - 11.3.3 内存管理
  - 11.4 热备份索引
    - 11.4.1 创建索引备份
    - 11.4.2 恢复索引
  - 11.5 常见错误
    - 11.5.1 索引损坏
    - 11.5.2 修复索引
  - 11.6 小结
- 第3部分 案例分析
- 第12章 案例分析1: krugle
  - 12.1 krugle介绍
  - 12.2 应用架构
  - 12.3 搜索性能
  - 12.4 源代码解析
  - 12.5 子串搜索
  - 12.6 查询vs搜索
  - 12.7 改进空间
    - 12.7.1 fieldcache内存使用
    - 12.7.2 合并索引
  - 12.8 小结
- 第13章 案例分析2: siren
  - 13.1 siren介绍
  - 13.2 siren优势
    - 13.2.1 通过所有域进行搜索
    - 13.2.2 一种高效词典
    - 13.2.3 可变域
    - 13.2.4 对多值域的高效处理
  - 13.3 使用siren索引实体
    - 13.3.1 数据模型
    - 13.3.2 实现问题
    - 13.3.3 索引概要
    - 13.3.4 索引前的数据准备
  - 13.4 使用siren搜索实体
    - 13.4.1 搜索内容



- 13.4.2 根据单元限制搜索范围
- 13.4.3 将单元合并成元组
- 13.4.4 针对实体描述进行查询
- 13.5 在solr中集成siren
- 13.6 benchmark
- 13.7 小结
- 第14章 案例分析3: linkedin
- 14.1 使用bobo browse进行分组搜索
  - 14.1.1 bobo browse的设计
  - 14.1.2 深层次分组搜索
- 14.2 使用zoie进行实时搜索
  - 14.2.1 zoie架构
  - 14.2.2 实时vs近实时
  - 14.2.3 文档与索引请求
  - 14.2.4 自定义indexreaders
  - 14.2.5 与lucene的近实时搜索进行比较
  - 14.2.6 分布式搜索
- 14.3 小结
- 附录a 安装lucene
  - a.1 二进制文件安装
  - a.2 运行命令行演示程序
  - a.3 运行web应用演示程序
  - a.4 编译源代码
  - a.5 排错
- 附录b lucene索引格式
  - b.1 逻辑索引视图
  - b.2 关于索引结构
    - b.2.1 理解多文件索引结构
    - b.2.2 理解复合索引结构
    - b.2.3 转换索引结构
  - b.3 倒排索引
  - b.4 小结
- 附录c lucene/contrib benchmark
  - c.1 运行测试脚本
  - c.2 测试脚本的组成部分
    - c.2.1 内容源和文档生成器
    - c.2.2 查询生成器
  - c.3 控制结构
  - c.4 内置任务
    - c.4.1 建立和使用行文件
    - c.4.2 内置报表任务
  - c.5 评估搜索质量
  - c.6 出错处理
  - c.7 小结
- 附录d 资源
  - d.1 lucene知识库
  - d.2 国际化
  - d.3 语言探测
  - d.4 项向量
  - d.5 lucene移植版本
  - d.6 案例分析
  - d.7 其他
  - d.8 信息检索软件
  - d.9 doug cutting的著作

d.9.1 会议论文  
d.9.2 美国专利  
• • • • • ([收起](#))

[Lucene实战\\_下载链接1\\_](#)

## 标签

lucene

搜索

搜索引擎

Java

全文检索

全文搜索

数据挖掘

编程

## 评论

这翻译要是能把初学者教会那就真是见鬼了

-----  
比较全面，也比较旧……好多示例都是过期代码了  
-----

书还不错，一个快速而简单的使用和理解Lucene工作的手册，就是翻译糙了点。

我才知道Lucene是其创建者Doug Cutting的妻子的中间名。翻译扣一星

又是一本翻译的很烂的书，逼得我又得慢慢看英文版的，公式抄错，很多重要的句子被忽略不翻译，专业术语你不懂就直接用英文得了，误人子弟！建议还是慢慢看英文版

对lucene的原理和使用的介绍还是比较详细的。lucene提供的索引和搜索接口很简单，使用lucene构建搜索引擎的难点是如何实现分布式、如何提高索引-搜索实时性、如何确定更好评分rank结果，基本各家都在lucene基础上进行了各种定制开发。

翻译真是简直了，还好之前用Lucene的时间不短，靠耐力读完了。

翻译不大好

读了一小部分， 暂时够用

简单过了下，翻译较差，很多翻译不太清楚的地方

入门绝对够用了。以前我一直以为lucene只是简单的索引关键字，然后进行搜索的东西，看了以后才发现有搜索引擎的机制。非常不错，考虑一下结合自己的项目，加入搜索引擎的机制。

翻译校对工作做得不好，减一星

-----  
翻译惨不忍睹 本书的重点还是 Lucene 的逻辑模型 (IndexWriter、Analyzer、Document、Field) 和 (IndexReader、Searcher、Query、Term、QueryAnalyser) ;  
难能可贵的, 是性能调优部分 (如: 利用多线程)、一致性、Field Cache 和排序相关部分 ... 后面研究下索引的结构、solr & es 来加深理解 ... Lucene 其实很像基于 SSTable 的存储引擎 ... 20190614 记录: 百度 PS  
的思路很棒, 通过生成与文档静态打分 (机器学习打分) 顺序一致的文档 ID 来排序, 不仅能够实现布尔检索, 还能更高效地排序

-----  
有java基础, 想学习全文搜索, 此书是最好的选择, 很快就让你进去全文搜索领域。不过中文版的翻译有和英文版的有些出入, 建议中文版和英文版的对照着学习。

-----  
java写的搜索引擎, 从未接触过搜索引擎方面的知识, 看了一边lucene全掌握, 你说他是不是好书?

-----  
公司在用ElasticSearch, 但是没有人精通。公司给我这个刚毕业的学生三个月时间, 能够为公司的ElasticSearch提出优化建议, 很慌啊, 虽然以前稍微学过Lucene和Solr, 但是想要达到精通的地步还是必须得看看这本书啊。 ---  
Lucene版本更新太快了, 这本书的有些内容明显过时, 自己结合书看了些源码, 里面很多内容是这本书没有讲到的。期待Lucene in Action 第三版

-----  
中文翻译的不好, 读起来不顺畅。还有错别字, 重复字等问题。总体是讲lucene api的使用, 基本没有讲解原理性内容。随便翻翻看, 工具书吧。

-----  
Lucene的入门, 做毕设才看的, 看了最简单的一部分, 不过这本书也没讲特别深入...

-----  
搜索引擎必备!! , 翻译上有些错误, 不过不影响

-----  
基本看过一遍, 很实用

## 书评

抛去翻译的问题，还是一本不错的lucene入门读物。最少可以让读者知道怎么简单的使用Lucene，进行简单的性能调整。不过现在lucuen已经扩展出太多的应用，无论是中文分词，文件系统调整或者动态的及时索引更新等问题都是没有讨论。当然作者是老外人家不分词，这个我忘记了。有兴趣...

-----

很久以前见百度的人用过这个，感觉是一本圣书。但是，初次看的时候，很失望。书中就是对lucene的几个基本接口作了介绍，举了一些例子。但是对实现的细节没有做说明。要彻底认识lucene还得从阅读源代码入手，结合lucene in action中介绍的API，沿着数据处理流...

-----

不错的一本书，对Lucene，或者说，Search中的一些关键点都有详细的讲述。看完后再去看源代码，可以做到事半功倍。

-----

我们team一直用lucene，不过把lucene用的跟关系表似的 汗一个  
搜索引擎三大块，索引查找和打分  
这本书索引讲的不够深入，其实lucene索引的内部的数据结构还是很经典的  
打分写的太浅，应该找个例子更深入一些 查找部分我个人认为是写的可以的，  
可作为入门书，一定要记得学习下...

-----

开源的IR系统中lucene是做得最好最有名，本书详细介绍了重要的模块。但是我最喜欢的是最后的例子：LinkedIn，SIREn他们所使用的技术和实现方法。在一个更高层次的观览全局，真的让我学到了很多。

-----

昨天去图书城，在最显眼的位置就是一堆Lucene实战！花了点时间翻了翻，个人感觉翻译得一般，很多翻译的都很直白，在因为中很多有前后语义逻辑关系的，翻译过后就看不出有这层关系了。不过可以理解的是，原版是09年6月左右出的，然后联系出版社

，翻译，校对等等都是很需...

-----  
书写得挺好，全面介绍了Lucene这个非常流行的java全文搜索引擎的框架。英文不难，条理清晰，读起来挺有味道。遗憾的是示例的API过时了。例如现在Lucene3.0 中的Field的创建方式与本书中所说的相差很大；IndexWriter的构造函数也有变化。相信还有其他deprecated 的地方...

-----  
做Lucene也只有这本书能参考了，没啥选择。还不错，全面，重要的细节也讲了，做Lucene必备参考书。

-----  
[Lucene实战\\_下载链接1\\_](#)