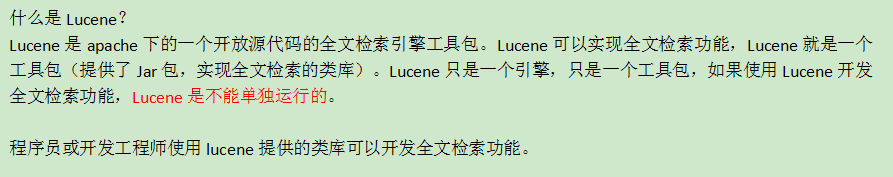
# Lucene

## Lucene介绍



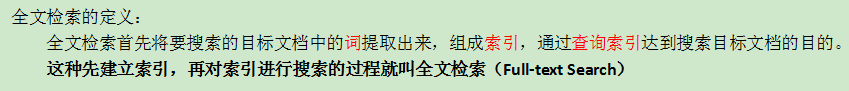
## 全文检索技术的应用场景

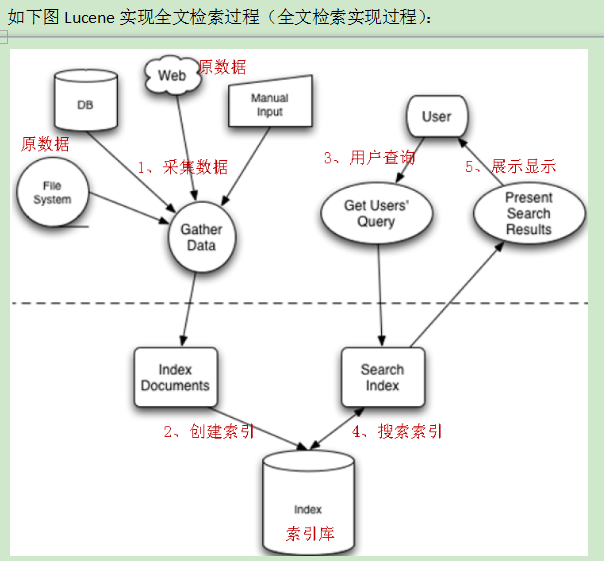
使用全文检索可以实现搜索引擎（百度、Google）

Lucene和搜索引擎的区别：搜索引擎是对外提供了全文检索服务，是可以单独运行的。Lucene只是工具包不能单独运行，需要在项目中加入Lucene的jar包，最终在JVM上运行。

使用全文检索技术可以实现站内搜索，站内搜索只能搜索本网站的信息（网页、pdf电子书、视频、音乐、关系数据库中的信息等等），比如：电商网站搜索商品信息，论坛网站搜索网内帖子。。。

## Lucene全文检索过程概括





全文检索过程分为 索引、搜索两个过程。

索引：

1、从关系数据库中、互联网上、文件系统采集源数据(要搜索的目标信息)

源数据的来源是很广泛的。

2、将源数据采集到一个统一的地方，要创建索引，将索引创建到一个索引库（文件系统）中

从源数据库中提取关键信息，从关键信息中抽取一个一个词，词和源数据是有关联。

索引库中就是一个一个的库。

搜索：

3、用户执行搜索（全文检索）编写查询关键字（全文检索查询语法，类似关系数据库中的sql）

4、从索引库中搜索索引，根据查询关键字搜索索引库中的一个一个词

5、展示搜索的结果。

创建索引时词和源数据有关联，索引库中记录了这个关联，如果找到了词就说明找到了源数据（http的网页、pdf电子书）。

## 环境搭建

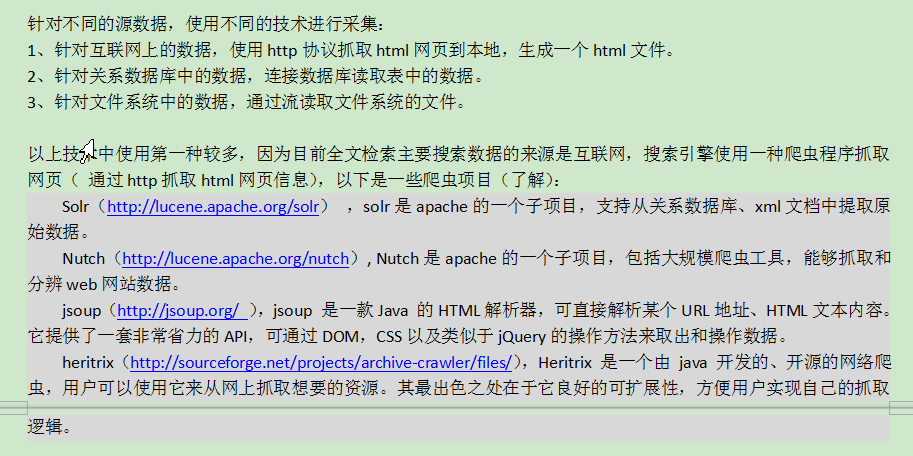
Lucene导入的包：

core核心包

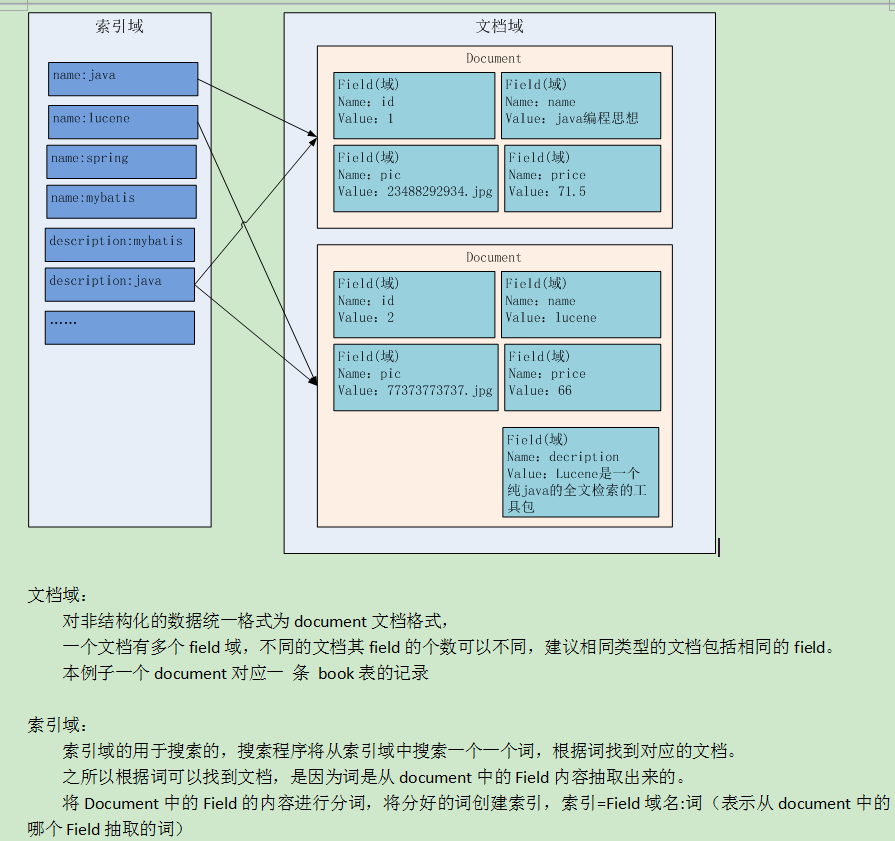
lucene-analyzers-common-4.10.3.jar 分析器通用包

lucene-queryparser-4.10.3.jar 查询解析器包

数据库驱动



## 入门

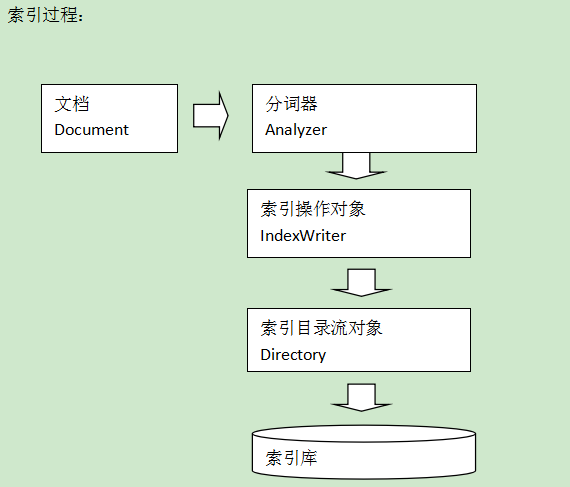


### 创建Document





### 创建索引



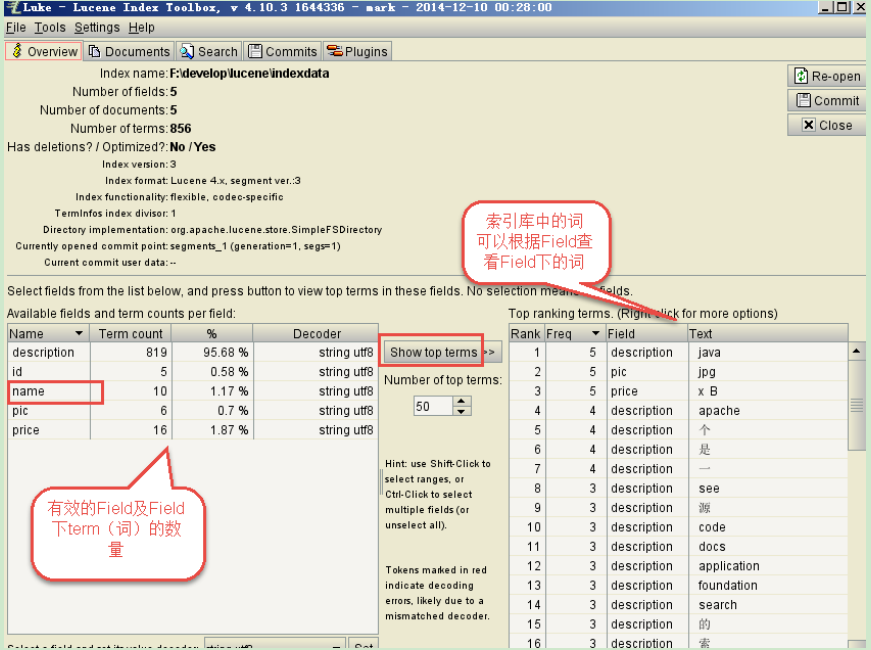


### luke查看分词

luke是一个小工具

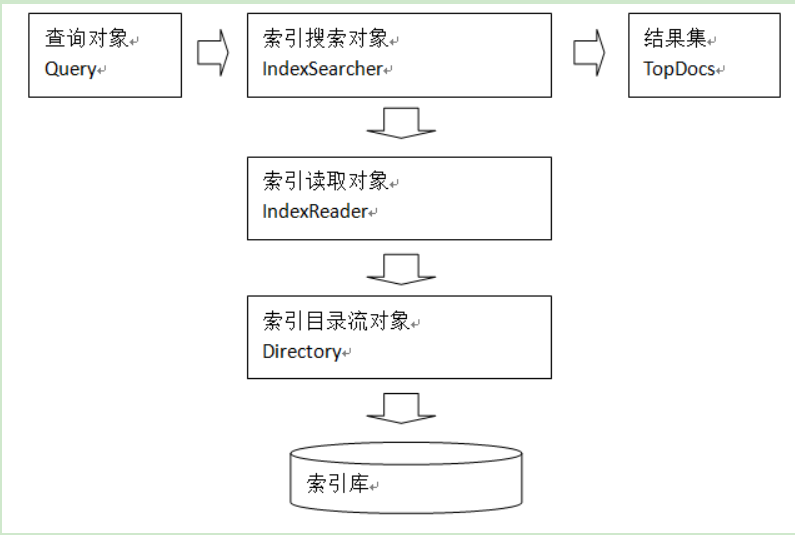
打开cmd 执行lukeall-4.10.3.jar，指定索引文件夹就可以了

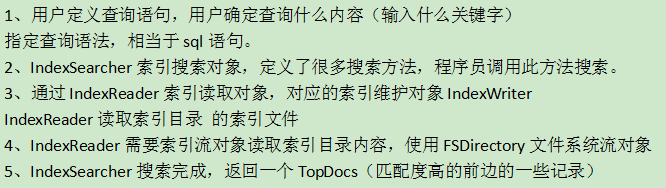
java -jar lukeall-4.10.3.jar

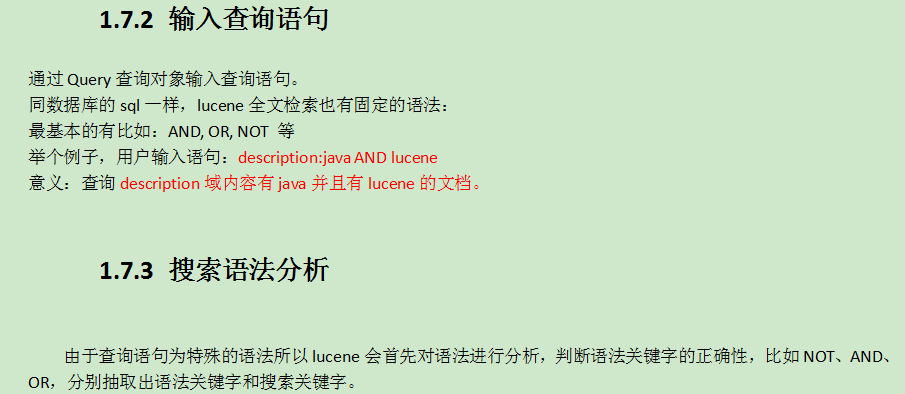


### 搜索部分

#### 搜索过程

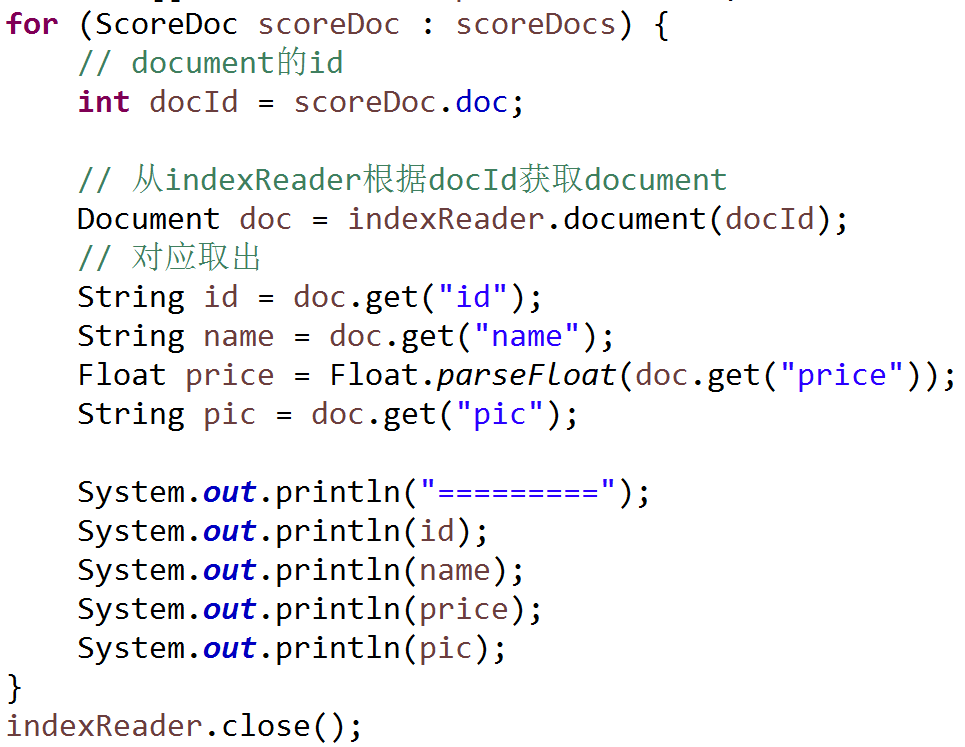






#### 搜索代码





## 索引深入

### 索引优化

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field类 | 数据类型 | Tokenized是否分词 | Indexed  是否索引 | Stored  是否存储 | 说明 |
| StringField(FieldName, FieldValue,Store.YES)) | 字符串 | N | Y | Y或N | 这个Field用来构建一个字符串Field，但是不会进行分析，会将整个串存储在索引中，比如(订单号,姓名等)  是否存储在文档中用Store.YES或Store.NO决定 |
| LongField(FieldName, FieldValue,Store.YES) | Long型 | Y | Y | Y或N | 这个Field用来构建一个Long数字型Field，进行分析和索引，比如(价格)  是否存储在文档中用Store.YES或Store.NO决定 |
| StoredField(FieldName, FieldValue) | 重载方法，支持多种类型 | N | N | Y | 这个Field用来构建不同类型Field  不分析，不索引，但要Field存储在文档中 |
| TextField(FieldName, FieldValue, Store.NO)  或  TextField(FieldName, reader) | 字符串  或  流 | Y | Y | Y或N | 如果是一个Reader, lucene猜测内容比较多,会采用Unstored的策略. |

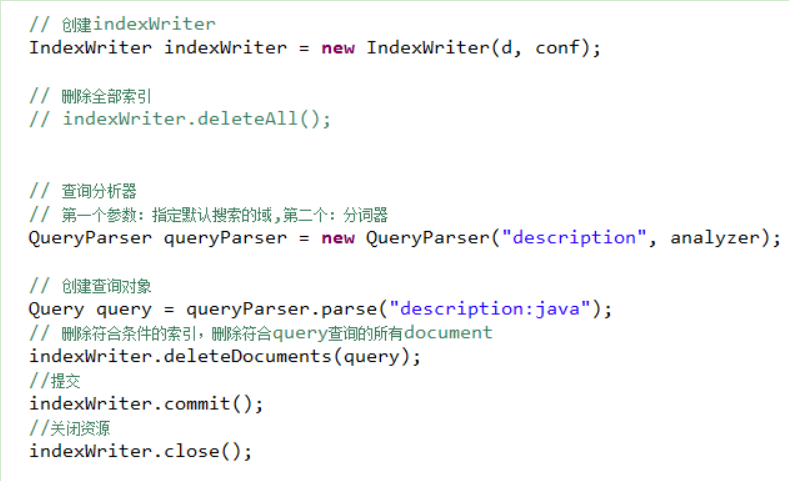
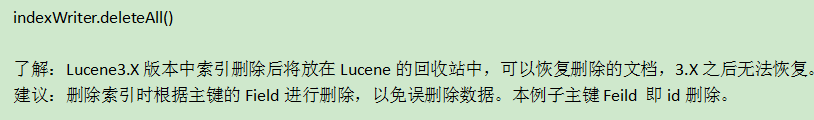
### 索引维护

数据库信息发生改变，需要对索引及时更新

#### 添加索引

调用 indexWriter.addDocument（doc）添加索引。

#### 删除索引

1. 按条件删除  
   
2. 删除全部  
   

#### 更新索引

更新思路：先查询、再删除、再添加。

建议根据主键field查询document，避免误更新。



### 查询

核心对象：indexSearcher

#### Query

子类

TermQuery根据指定域中的词查询

NumericRangeQuery 根据数字范围查询

BooleanQuery 组合查询

#### 基于QueryParser

QueryParser qp = new QueryParser(“域”,分词器);

Query query = qp.parse(“java”);

+ 并且 AND

- 非 NOT

空格 或 OR

Query query = qp.parse(“java spring”);

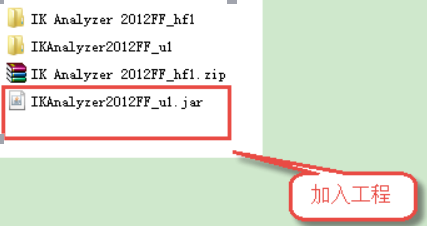
MultiFieldQueryParser(常用) 多个域组合查询

还有具体看课堂笔记

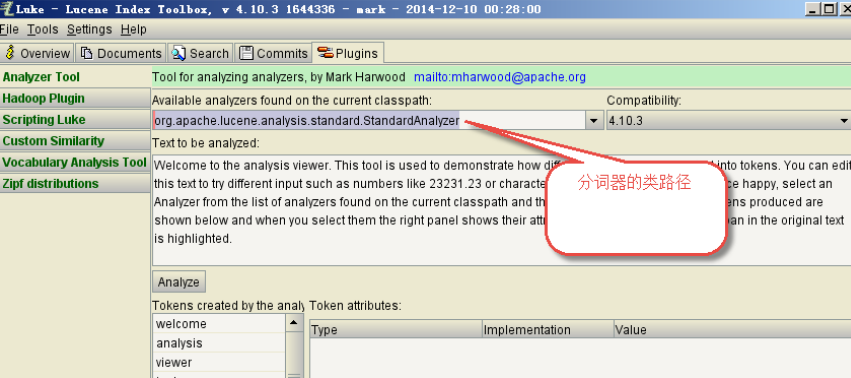
## 中文分词器

### ik分词器

IK Analyzer是一个开源的，基于java语言开发的轻量级的中文分词工具包。从2006年12月推出1.0版开始， IKAnalyzer已经推出了4个大版本。最初，它是以开源项目Luence为应用主体的，结合词典分词和文法分析算法的中文分词组件。从3.0版本开始，IK发展为面向Java的公用分词组件，独立于Lucene项目，同时提供了对Lucene的默认优化实现。在2012版本中，IK实现了简单的分词歧义排除算法，标志着IK分词器从单纯的词典分词向模拟语义分词衍化。



### 使用luke测试中文分词



使用Luke测试第三方分词器分词效果，需通过java.ext.dirs加载jar包:

可简单的将第三方分词器和lukeall放在一块儿，cmd下运行：

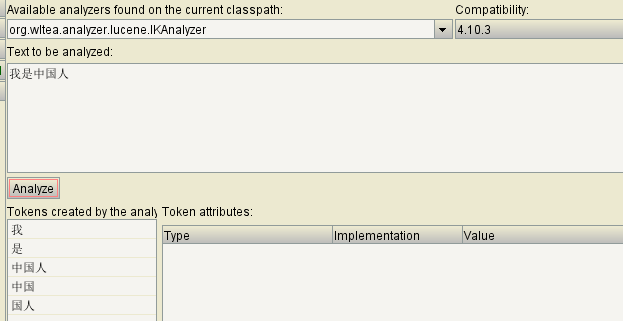
**java -Djava.ext.dirs=. -jar lukeall-4.10.3.jar**



### 如何使用中文分词器

如果使用中文分词器ik-analyzer，就在索引和搜索程序中使用一致的分词器ik-analyzer。

创建ik的分词器创建索引



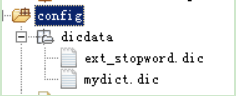
搜索使用ik也要创建ik分词器

### 扩展中文分词器词库

在classpath下定义IKAnalyzer.cfg.xml文件，如下：



在classpath下的编辑dicdata/mydict.dic文件，此文件中存储扩展词库，在dicdata/ext\_stopword.dic文件中存放停用词。



注意：mydict.dic文件的编码格式 为utf-8 而不是 utf-8 + bom格式！！！！