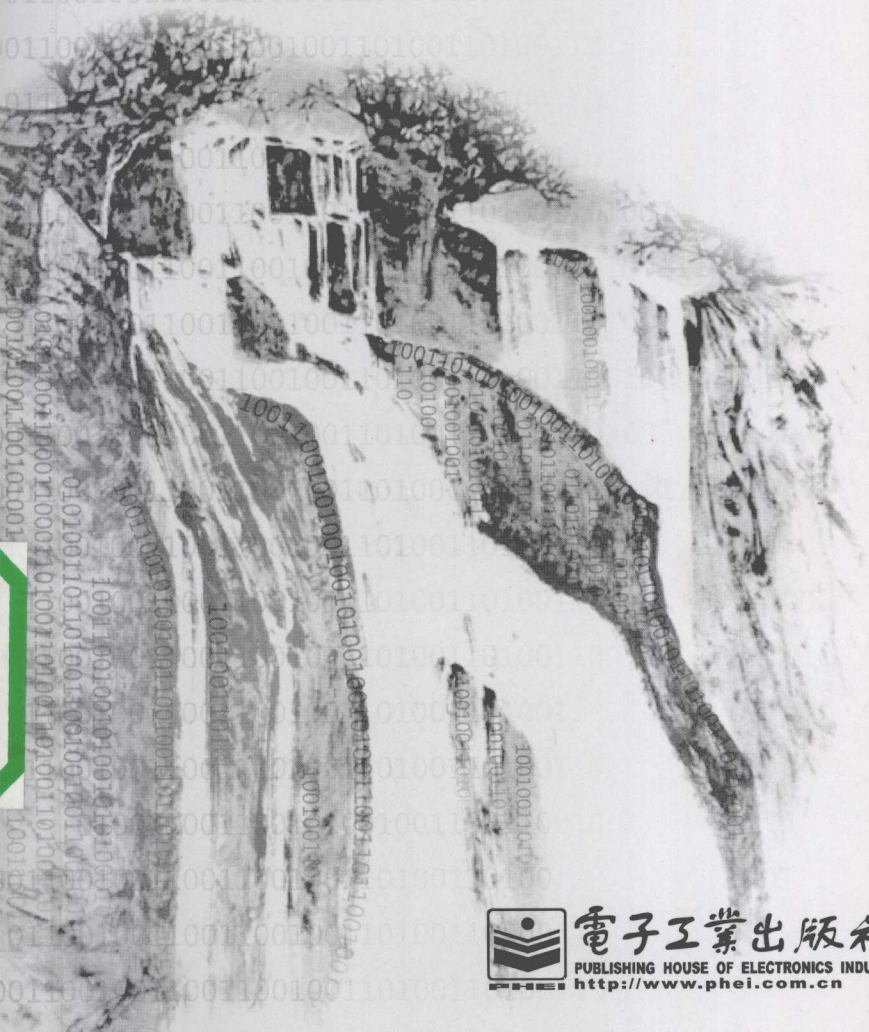
Broadview
www.broadview.com.cn

◎ 首次揭示商业级中文搜索实现秘密
◎ 业内知名开发团队倾情奉献
◎ 引领Lucene开发技术升级

自己动手写搜索引擎

罗 刚 编著



電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

自己动手写搜索引擎

罗刚 编著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内 容 简 介

本书是猎兔企业搜索开发团队的软件产品研发和项目实践的经验汇总。本书全方位展现出一个商用级别的 Lucene 搜索解决方案，主要包括爬虫、自然语言处理和搜索实现部分。

爬虫部分介绍了网页遍历方法和从网页提取主要内容的方法。

自然语言处理部分包括了中文分词从理论到实现以及在搜索引擎中的实用等细节。

其他自然语言处理的经典问题与实现包括：文档排重、文本分类、自动聚类、语法解析树、拼写检查、拼音转换等理论与实现方法。

在实现搜索方面，本书用简单的例子介绍了完整的搜索实现过程，覆盖了从索引库的设计和索引库与数据库的同步到搜索用户界面设计与实现。搜索用户界面包括实现布尔逻辑查询、按区间范围查询、搜索结果按日期排序等。本书还进一步介绍了搜索排序的优化方法。

最后以基于 Lucene 的搜索服务器 Solr 为例，展示了 Lucene 的最新应用方法。

图书在版编目（CIP）数据

自己动手写搜索引擎 / 罗刚编著. —北京：电子工业出版社，2009.12
ISBN 978-7-121-09640-2

I. 自… II. 罗… III. 互联网络－情报检索 IV. G354.4

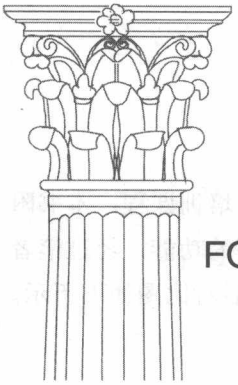
中国版本图书馆 CIP 数据核字（2009）第 178728 号

责任编辑：江 立
印 刷：北京智力达印刷有限公司
装 订：北京中新伟业印刷有限公司
出版发行：电子工业出版社
北京市海淀区万寿路 173 信箱 邮编 100036
开 本：787×980 1/16 印张：23 字数：390 千字
印 次：2009 年 12 月第 1 次印刷
印 数：4000 册 定价：55.00 元（含 DVD 光盘 1 张）

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。



FOREWORD 前 言

15 在中国，随着互联网从城市到农村的普及，搜索引擎对日常生活产生越来越大的影响。例如，笔者自己一般每天就有 15 个左右的问题需要求助于搜索引擎。从 04 年开始笔者也从数据库相关软件开发转入搜索引擎相关开发工作。

Google 20 世纪末，在美国国家科学基金会的支持下，斯坦福大学的两个学生在他们的教授指导下开始了一个数字图书馆项目。后来，他们创建了 Google 公司，开创了通过互联网搜索技术共享人类信息的新纪元。Google 通过网络广告取得了巨大的商业回报，到现在仍然是世界 500 强企业中赢利能力最强的公司之一。NASDAQ 证券交易市场的最高股价是 Google 公司的股票。搜索引擎开发成为一项极有含金量的技术。

Web 开始写作《自己动手写搜索引擎》这本书以前，已经有一些介绍搜索理论或者搜索开发工具的图书，但是往往表现出来的是纯粹的理论推导和公式定理，或者仅仅是现成开源软件的介绍、分析和使用，并没有介绍其理论依据。有的读者是数学专业的博士，对于相关的数学模型一看就明白，但对于算法实现可能仍然缺少经验。有的读者是培训学校毕业的学生，可能对 Web 开发框架和软件工具的使用很熟悉，但缺少理论基础和深入创新的能力。本书的一个特点在于前面是原理介绍，接着是具体的代码实现。不仅讲解抽象的知识，更重要的是把知识转化成具体软件应用的过程也展示出来。

Lucene《自己动手写搜索引擎》是猎兔企业搜索开发团队的软件产品研发和项目实践的经验汇总。感谢 Lucene，它把搜索引擎开发工作变成了广大程序员都能够参与的游戏。所以本书选用 Lucene 来全方位展现一个商用级别的搜索解决方案。中文分词当前仍然是实现中文搜索的热门话题之一。本书重点介绍了中文分词的相关理论和代码实现，以及在搜索引擎中实用中文分词等细节。本书用简单的例子介绍了搜索引擎完整的实现过程，同时也没有忽略一些经典的算法实现。

该书适合需要具体实现搜索引擎的程序员使用，对于自然语言处理等相关研究人员也

有一定参考价值，同时猎兔搜索团队也已经开发出以本书为基础的专门培训课程。本书附带光盘中的代码经过了详细的注释。为了帮助初学者更容易地了解程序的功能，经过笔者的精心整理后，每个主要变量和每行主要的执行程序都加上了注释，前后对比图如下所示。

```
public int wordSegment(String Sentence)
{
    int senLen = Sentence.length();
    int i = 0, j = 0;
    int M = 12;
    String word;
    boolean bFind = false;
    while (i < senLen)
    {
        int N = i + M < senLen ? i + M : senLen;
        bFind = false;
        for (j = N; j > i; j--)
        {
            word = Sentence.substring(i, j);
            if (dic.Find(word)) {
                System.out.print(word + " ");
                bFind = true;
                i = j;
                break;
            }
        }
        if (bFind == false)
        {
            word = Sentence.substring(i, i + 1);
            System.out.print(word + " ");
            ++i;
        }
    }
    System.out.println();
    return 1;
}
```

```
public int wordSegment(String Sentence)// 传入一个字符串作为要处理的对象。
{
    int senLen = Sentence.length();// 首先计算出传入的这句话的字符长度
    int i = 0, j = 0;// i是用来控制截取的起始位置的变量，j是用来控制截取的结束位置的变量
    int M = 12;// 所取得的词组的最大值不超过12。
    String word;// 我们需要与词库中做对比的词。
    boolean bFind = false;// 用来判断是否是词库中的词。
    while (i < senLen)// 如果i的大小小于此句话的长度就进入循环，如果i的大小等于了这句话的长度也就证明了匹配过程已经结束了
    {
        int N = i + M < senLen ? i + M : senLen;// 如果i+M<senLen为真就把i+M的值赋给N，如果其为假就把senLen的结果赋值给N
        // 我们首先坐的事就是从一句话中取出一个处理范围
        // 以上这句话就是为了界定我们所要处理的字或者是词组的大小。
        bFind = false;// 假设这个我们取出来的词组不是词库中的词
        for (j = N; j > i; j--)// 正向最大匹配
        {
            word = Sentence.substring(i, j);// 截取我们所需要的字符串。
            // 如果没有匹陪到合适的词的话就会一直的在这里循环直到出循环。
            if (dic.Find(word)) {
                System.out.print(word + " ");// 如果这个词是词库中的那么就打印出来
                bFind = true;
                i = j;// 如果你在词库中找到了这个词那么就截取的末尾的位置j赋给i
                // System.out.println(i);
                break;// 跳出for循环
            }
        }
        if (bFind == false)// 如果在我们所处理的范围内一直都没有找到匹配上的词，就一个字一个字的打印出来。
        {
            word = Sentence.substring(i, i + 1);
            System.out.print(word + " ");
            ++i;// 这句话其实就是控制读的顺序
        }
    }
    System.out.println();
    return 1;
}
```

3年 笔者花了3年时间编写本书，但限于时间和水平，其中很多内容笔者仍然正在改写中，希望读者不吝赐教，使本书在重印时内容能更全面。如果条件允许，笔者也很愿意探索一些非比寻常的方式来介绍技术。

API 笔者把写书的过程看成是笔者自身和作者所组建的技术团队修炼和提高的过程。笔者相信技术高手并非天生，而是来源于对卓越品质的追求和自我的不断提升。起点低不是问题，笔者所在的搜索开发团队就包括一些高中学历的程序员，不愿意学习和深入钻研问题才是最大的问题。从企业知识管理的角度，这本书也是有益的。团队里每次有新人进来，我们都先把相关文档发给他们，这样内部交流技术就多了一些基础，不用每次都从 Lucene 最基本的 API 等这些基本知识开始介绍。

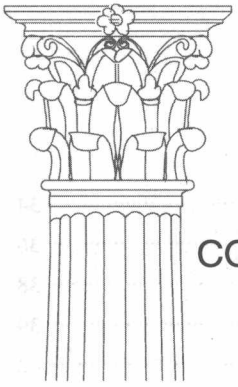
搜索 从学习的过程来看，有人会问："你没有去过**公司，怎么知道开发搜索引擎？"笔者没有去过大的搜索公司，所以只能从公开的论文、资料和代码等开始学习。感谢搜索引擎技术本身的发展，这方面的学习材料越来越容易通过搜索获得。笔者没有在课堂上学习过自然语言处理等方面的课程，刚开始做中文分词的时候，吴春尧博士问："你知不知道语言模型？"笔者当时对这些概念一无所知。但是通过了解中文分词程序每一行代码的作用，阅读相关的论文，终于了解了其中一些常用的数学方法。

Solr 从开发过程来看，首先由客户或者其他开发人员提出问题，解决问题，重新审查整个过程并寻找更好的解决方法。比如实现搜索结果中分类统计，最开始用 QueryFilter 方法可以实现基本的功能，后来 Solr 的作者实现了一个更快的 DocSetHitCollector 方法，如果用查表法来改进这个 DocSetHitCollector 方法可以达到比 Solr 官方版本更快的计算速度。搜索软件开发的过程借用日本产品研发大师稻盛和夫的话与大家共勉——"对产品的现场，重新进行审视、体察、贴心、倾听。这样的话，就可以听见神灵的声音。现场产品传来喃喃细语告知我们解决问题的诀窍——这样试试如何？"

准确 从成书过程来看，这本书也找过一些内部的开发人员请他们挑问题使得本书的表达更准确。最后，电子工业出版社的胡辛征老师对本书提出了很多具体而必要的建议，

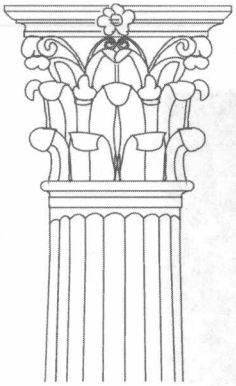
江立老师反复做了细致的修订工作，他们丰富的经验和认真负责的态度使得本书从粗糙的讲稿变成了书本的形式。

感谢最后笔者衷心感谢家人、关心我的老师和朋友们、创业伙伴、搜索培训班的学员以及选择猎兔搜索软件的客户多年来的支持。第一次开办搜索培训班时，由于讲稿本身准备得不够充分，学员有诸多的反馈意见。经过不断的修改后，新学员的负面反馈越来越少。他们的意见也在时刻提醒笔者要保持谦虚的态度。猎兔搜索的客户对于完善搜索技术提供了重要的支持，客户公司包括：天下互联公司、北京电视台、港澳资讯公司、三星鹏泰公司、中国自动化网、中国工控网、中国机电贸易网、易车网、敦煌网、G 宝盘、消息树网站等。



CONTENTS

目　录



第 1 章　遍历搜索引擎技术

搜索引擎是我们每天上网经常使用的功能，本书介绍的搜索技术需要 Java 编程语言基础。本章从快速实现基本的搜索入手，然后深入展开分析搜索的基本技术。

1.1　30 分钟实现的搜索引擎

首先从一个简单的搜索引擎入手，实现一个简单的指定目录文件的搜索引擎。实现之前需要读者具有 Java 开发方面的基础知识。

1.1.1　准备工作环境（10 分钟）

首先要准备一个 Java 的开发环境。当前可以使用 JDK 1.6。JDK 1.6 可以从 Sun 的官方网站 java.sun.com 下载得到。使用默认方式安装即可。

然后要使用的是一个用来管理搜索引擎索引库的 jar 包，叫做 Lucene。目前可以从 http://lucene.apache.org/java/docs/index.html 下载到最新的 Lucene，当前的版本是 2.3。另外，使用集成开发环境 Eclipse，其开发界面如图 1-1 所示。

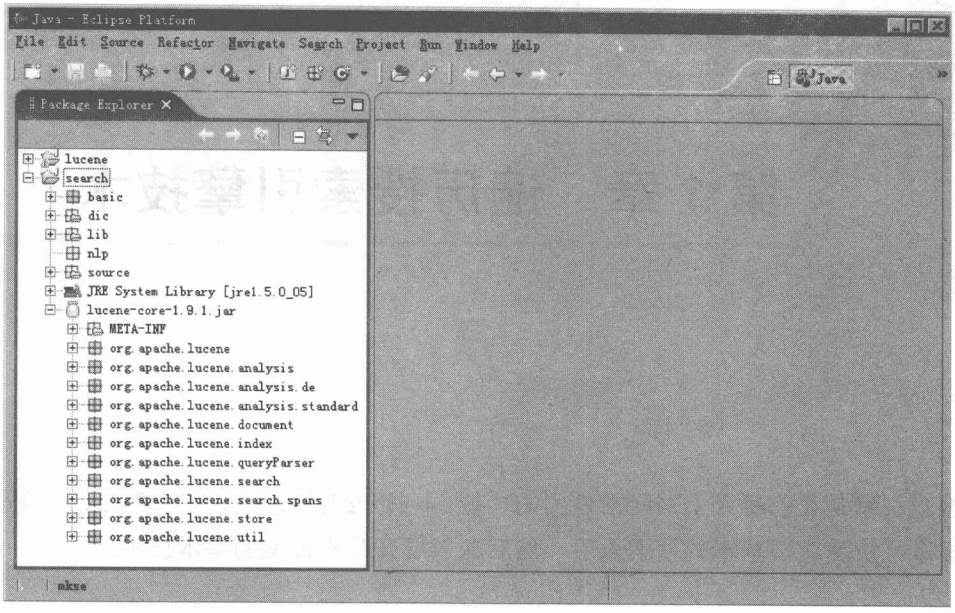


图 1-1　Java 开发界面

如果需要用 Web 界面搜索，还要下载 Tomcat，当前可以从 http://tomcat.apache.org/ 下载到，推荐使用 Tomcat 5.5 以上的版本。

然后增加 Tomcat 的内存使用量，防止内存溢出。

如果是在 Linux 下，可以用 vi 编辑器修改./catalina.sh 文件，增加行：

```
JAVA_OPTS=-Xmx600m
```

如果是在 Windows 下，可以用文本编辑器“记事本”修改./catalina.bat 文件，增加行：

```
set JAVA_OPTS=-Xmx600m
```

【作者提示】如果不增加内存使用量，运行时可能会出现 java.lang.OutOfMemoryError 异常。

1.1.2　编写代码（15 分钟）

搜索引擎的基础在于对全文索引库的管理，在 Lucene 中，通过 IndexWriter 来写入索引库。伪代码如下：

1. 创建 IndexWriter，准备写索引；

2. 遍历要索引的路径；

3. 优化索引。

下面是主要的实现代码：

```
    public void go() throws Exception {
            long start = System.currentTimeMillis();
            if (verbose) {
                System.out.println("Creating index in: " + indexDir);
                //创建索引目录或者建立增量索引
                if (incremental) System.out.println("- using incremental mode");
            }
            Index = new IndexWriter(new File(indexDir), new StandardAnalyzer(),
                !incremental);//打开或创建索引库，indexDir 是索引存放的路径

            File dir = new File(sSourceDir);//待索引的文件存放的路径
            indexDir(dir);//索引路径
            index.optimize();//索引优化
            index.close();//关闭索引库
            if(verbose)
            System.out.println("index complete in :"+(System. currentTime
Millis() - start)/1000);
    }
```

下面这段代码把文件内容加到索引库：

```
    private void indexFile(File item) {
```

```
            if (verbose) System.out.println("Adding FILE: " + item);
            News news = loadFile(item);//把文件中的内容加载到 news 对象
            if ( news!= null && news.body != null) {
                Document doc = new Document();
                //创建网址列
                Field f = new Field("url", news.URL ,
                        Field.Store.YES, Field.Index.UN_TOKENIZED,
                        Field.TermVector.NO);
                doc.add(f);
                //创建标题列
                f = new Field("title", news.title ,
                        Field.Store.YES, Field.Index.TOKENIZED,
                        Field.TermVector.WITH_POSITIONS_OFFSETS);
                doc.add(f);
                //创建内容列
                f = new Field("body", news.body.toString() ,
                        Field.Store.YES, Field.Index.TOKENIZED,
                        Field.TermVector.WITH_POSITIONS_OFFSETS);
                doc.add(f);
                try{
                    //文档增加到索引库
                    index.addDocument(doc);
                }
                catch(Exception e)
                {
                    e.printStackTrace();
                    System.exit(-1);
                }
            }
    }
```

完整的代码可以在本书附带的光盘中找到。

运行以后，一般会生成以下三个索引文件：