Web 信息处理与应用实验一程序说明——金泽文（PB15111604）

简单的搜索引擎实现

【实验内容】

通过 Java 语言,使用 Lucene 制作一个简单的搜索引擎,
其中创建索引以及查询的分词工具,使用中科院中文分词工具 ICTCLAS,
再编写简单的 jsp 查询界面.
搜索引擎的数据库使用新浪网 5 个季度的新闻网页.

【实验环境】

编程语言: Java SE8, jsp

编程环境: Win10 Pro 1703, IntelliJ IDEA Ultimate 2017.3

运行环境: JRE 1.8.0_152, JVM OpenJDK 64-Bit Server VM.

使用工具: Apache Lucene7.1.0,
NLPIR ICTCLAS 6.6.0,
Apache Tomcat 9.0.1
IntelliJ IDEA Ultimate 2017.3
Chrome 62.0

【实验步骤及方法】

主要步骤:

1. 调研与学习:

   学习 java.(这部分主要是通过 <Core Java>这两本书, 以及算法课的实验(故意用 java 写实验来学习并熟练))

   调研当下业界认可的 java IDE,以及业界认可的分词器.

   调研并学习 Lucene 框架和 ICTCLAS 工具的配置,使用方法.(这部分主要是通过官方的 github 上的 demo 以及 lucene 官方文档进行学习)

lucene-analyzers-nlpir-ictclas-6.6.0

NLPIR/ICTCLAS for Lucene/Solr 6.6.0 analyzer plugin. Support: MacOS,Linux x86/64, Windows x86/64

The project resources folder is a source folder, which contains all platform's dynamic libraries and push them to the classpath.//Source Folder 保证所有平台下的动态库自动部署到classpath环境下，以便JNA加载动态库。

- NLPIRTokenizerAnalyzer DEMO

```
String text="我是中国人";
NLPIRTokenizerAnalyzer nta = new NLPIRTokenizerAnalyzer("", 1, "", "", false);
TokenStream  ts  = nta.tokenStream("word", text);
ts.reset();
CharTermAttribute  term = ts.getAttribute(CharTermAttribute.class);
while(ts.incrementToken()){
    System.out.println(term.toString());
}
ts.end();
ts.close();
nta.close();
```

- Lucene DEMO

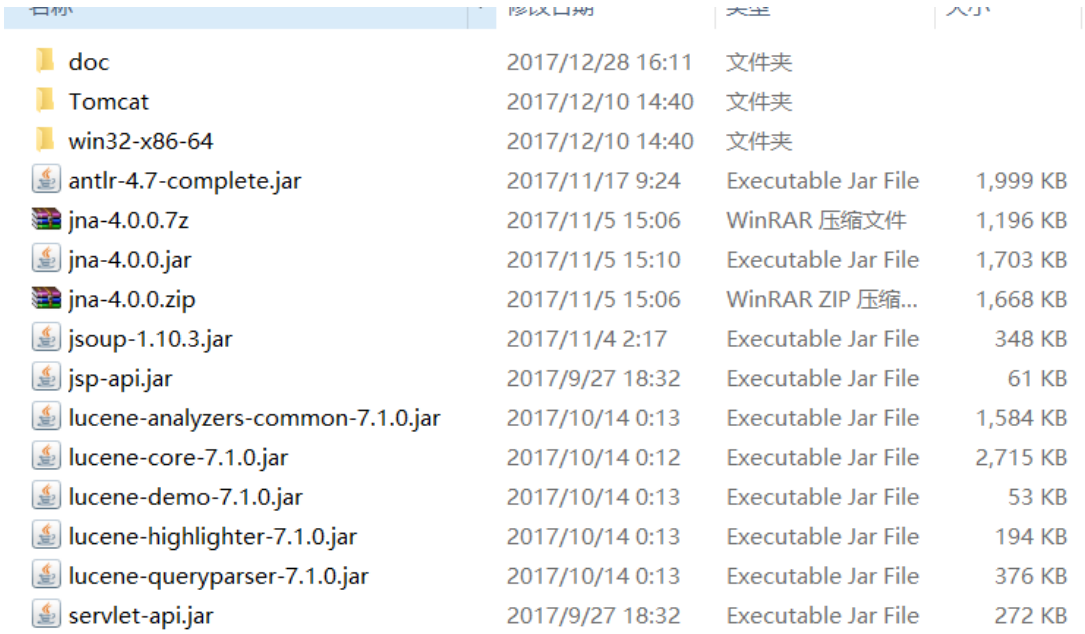
```
//For indexing
NLPIRTokenizerAnalyzer nta = new NLPIRTokenizerAnalyzer("", 1, "", "", false);
IndexWriterConfig inconf=new IndexWriterConfig(nta);
inconf.setOpenMode(OpenMode.CREATE_OR_APPEND);
IndexWriter index=new IndexWriter(FSDirectory.open(Paths.get("index/")),inconf);
Document doc = new Document();
doc.add(new TextField("contents", "特朗普表示，很高兴汉堡会晤后再次同习近平主席通话。我同习主席就重大问题保持沟通和
index.addDocument(doc);
index.flush();
index.close();
//for searching
String field = "contents";
IndexReader reader = DirectoryReader.open(FSDirectory.open(Paths.get("index/")));
IndexSearcher searcher = new IndexSearcher(reader);
```

   调研并学习网页 jsp 的语法规则,以及 Tomcat 的配置与使用方法

2. 配置环境:

   下载 IntelliJ IDEA, 进行配置.

下载 Lucene 和 ICTCLAS,以及 tomcat,添加到环境变量中,并且配置项目的路径.



名称	修改日期	类型	大小
doc	2017/12/28 16:11	文件夹	
Tomcat	2017/12/10 14:40	文件夹	
win32-x86-64	2017/12/10 14:40	文件夹	
antlr-4.7-complete.jar	2017/11/17 9:24	Executable Jar File	1,999 KB
jna-4.0.0.7z	2017/11/5 15:06	WinRAR 压缩文件	1,196 KB
jna-4.0.0.jar	2017/11/5 15:10	Executable Jar File	1,703 KB
jna-4.0.0.zip	2017/11/5 15:06	WinRAR ZIP 压缩...	1,668 KB
jsoup-1.10.3.jar	2017/11/4 2:17	Executable Jar File	348 KB
jsp-api.jar	2017/9/27 18:32	Executable Jar File	61 KB
lucene-analyzers-common-7.1.0.jar	2017/10/14 0:13	Executable Jar File	1,584 KB
lucene-core-7.1.0.jar	2017/10/14 0:12	Executable Jar File	2,715 KB
lucene-demo-7.1.0.jar	2017/10/14 0:13	Executable Jar File	53 KB
lucene-highlighter-7.1.0.jar	2017/10/14 0:13	Executable Jar File	194 KB
lucene-queryparser-7.1.0.jar	2017/10/14 0:13	Executable Jar File	376 KB
servlet-api.jar	2017/9/27 18:32	Executable Jar File	272 KB

值得一提的是,中间因为 IntelliJ 和 Tomcat 的包路径等问题,调试了好几天.

3. Coding 和 debugging

1. ICTCLAS 的封装

接口类 CNLPIRLibrary 的封装

```
public interface CNLPIRLibrary extends Library {

    CNLPIRLibrary Instance = (CNLPIRLibrary) Native.loadLibrary
 ( name: "NLPIR", CNLPIRLibrary.class);

    public boolean NLPIR_Init(String sDataPath, int encoding, String
 sLicenceCode);

    public String NLPIR_ParagraphProcess(String sParagraph, int bPOSTagged);

    public int NLPIR_ImportUserDict(String dictFileName, boolean bOverwrite);

    public String NLPIR_GetLastErrorMsg();
}
```

NLPIRTokenizerAnalyzer 的封装

```
package ICTCLAS_Analyzer;

import org.apache.lucene.analysis.Analyzer;
import org.apache.lucene.analysis.Tokenizer;

public class NLPIRTokenizerAnalyzer extends Analyzer{

    String data=null;
    int encoding=1;
    String sLicenceCode=null;
    String userDict=null;
    boolean bOverwrite=false;

    public NLPIRTokenizerAnalyzer(String data,int encoding,String
 sLicenceCode,String userDict,boolean bOverwrite) {
        this.data=data;
        this.encoding=encoding;
        this.sLicenceCode=sLicenceCode;
        this.userDict=userDict;
        this.bOverwrite=bOverwrite;
    }

    @Override
    protected TokenStreamComponents createComponents(String fieldName) {
        final Tokenizer tokenizer = new NLPIRTokenizer(this.data,this
 .encoding,this.sLicenceCode,this.userDict,this.bOverwrite);
        return new TokenStreamComponents(tokenizer);
    }

}
```

NLPIRTokenizer 的封装与实现

```
11  public class NLPIRTokenizer extends Tokenizer {
12
13      private final CharTermAttribute termAtt = addAttribute(CharTermAttribute.class);
14      private final OffsetAttribute offsetAtt = addAttribute(OffsetAttribute.class);
15      private final TypeAttribute typeAtt = addAttribute(TypeAttribute.class);
16
17      private String[] buffer = null;
18      private StringBuffer cbuffer = null;
19      int start = 0;
20      int end = 0;
21      int current = 0;
22
23      String data=null;
24      int encoding=1;
25      String sLicenceCode=null;
26      String userDict=null;
27      boolean bOverwrite=false;
28
29      public void defaultInit() {}
31
32      public NLPIRTokenizer(AttributeFactory factory) {...}
37
38      public NLPIRTokenizer(String data, int encoding, String sLicenceCode, String
        userDict, boolean bOverwrite) {...}
41
42      public NLPIRTokenizer(AttributeFactory factory, String data, int encoding, String
        sLicenceCode, String userDict, boolean bOverwrite) {...}
46
47      private void init(String data, int encoding, String sLicenceCode, String userDict,
        boolean bOverwrite) {...}
65
66      @Override
67      public boolean incrementToken() throws IOException {...}
94
95  }
```

2. 网页预处理的实现

·过滤无效标签:

对于每个文件的每个<doc>...</doc>部分进行处理.通过filterTags 方法,过滤掉无效标签.

```
59
60      // 过滤无效标签
61  @   public static String filterTags(String s){
62          StringBuilder sb = new StringBuilder(s);
63          // uselessTags中的前五对一起处理
64          for(int i = 0; i < 10; i+=2){
65              int j = sb.indexOf(uselessTags[i]);
66              // 一直删除,直到sb里找不到.
67              while(j > -1) {
68                  int k = sb.indexOf(uselessTags[i + 1]);
69                  if(k < 0) {
70                      sb.delete(j, j + uselessTags[i].length());
71                      j = sb.indexOf(uselessTags[i]);
72                      continue;
73                  }
74                  if (k <= j) {
75                      sb.delete(k, k + uselessTags[i + 1].length());
76                      continue;
77                  }
78                  sb.delete(j, k + uselessTags[i + 1].length());
79                  j = sb.indexOf(uselessTags[i]);
80              }
81          }
82          //" " 单独处理
83          String nbsp = " ";
84          int i = sb.indexOf(nbsp);
85          while(i > -1) {
86              sb.delete(i, i + nbsp.length());
87              i = sb.indexOf(nbsp);
88          }
89          return sb.toString();
90      }
91
```

删除的标签存在数组中,其中包括 5 对标签和一个” ”,另外还有防止遗漏的正则表达式.

```
final private static String[] uselessTags = {"<a ","</a>","<em>",
"</em>","<span ","</span>","<strong>","</strong>","<iframe ",
"</iframe>"," "};
```

·提取有用标签

通过正则匹配与 html 解析工具 Jsoup 进行有用标签的提取,加入索引文档中.

```
130                 if(content.charAt(1) == 'm'){
131                     // <meta attr>
132                     org.jsoup.nodes.Document doc = Jsoup.parse(content);
133                     org.jsoup.select.Elements attribute = doc.select( cssQuery: "meta");
134                     if(attribute.size() == 0)
135                         continue;
136                     if(attribute.attr( attributeKey: "name").equals("keywords"))
137                         docLucene.add(new TextField( name: "keywords", attribute.attr
( attributeKey: "content"), Store.YES));
138                     else if(attribute.attr( attributeKey: "name").equals("description") && (docLucene
.get("description")==null || docLucene.get("description").length() == 0))
139                         docLucene.add(new TextField( name: "description", attribute.attr
( attributeKey: "content"), Store.YES));
140                     else if(attribute.attr( attributeKey: "name").equals("publishid")&& (docLucene.get
("publishid")==null || docLucene.get("publishid").length() == 0))
141                         docLucene.add(new StringField( name: "publishid", attribute.attr
( attributeKey: "content"), Store.YES));
142                     else if(attribute.attr( attributeKey: "name").equals("subjectid")&& (docLucene.get
("subjectid")==null || docLucene.get("subjectid").length() == 0))
143                         docLucene.add(new StringField( name: "subjectid", attribute.attr
( attributeKey: "content"), Store.YES));
144                 }
145                 else{
146                     Matcher m = ptTitle.matcher(content);
147                     String tmp;
148                     if(m.find()){
149                         tmp = m.group(2).trim();
150                         if(docLucene.get("title") == null || docLucene.get("title").length() == 0)
151                         docLucene.add(new TextField( name: "title", tmp, Store.YES));
152                     }
153                     m = ptUrl.matcher(content);
154                     if(m.find()){
155                         tmp = m.group(2).trim();
156                         if(docLucene.get("url") == null || docLucene.get("url").length() == 0)
157                         docLucene.add(new StringField( name: "url", tmp, Store.YES));
158                     }
159                 }
160             }
```

3. 创建索引的实现

仿照官方 demo,得到:

索引创建的中间过程在 RAM 中进行,最后再输出到磁盘系统中,提高性能.使用的 analyzer 为上面实现的 NLPIRTokenizerAnalyzer.用 analyzer 设置 IndexWriter,之后 writer.add()的过程入上面选取有效标签之后加入的代码片段,不再赘述.

```
41      private static Analyzer analyzer;
42      private static Directory ramDirectory;              //
    在RAM中进行操作,加快速度,最后将创建的索引放在磁盘上.
43      private static Directory fsDirectory;               // 最后存在这里
44      private static IndexWriterConfig iwc;
45      private static IndexWriter writer;
46      private static File[] files;
47
48      // 初始化
49      private static void init()throws IOException {
50          ramDirectory = new RAMDirectory();
51          analyzer = new NLPIRTokenizerAnalyzer( data: "",
    encoding: 1,  sLicenceCode: "",  userDict: "",  bOverwrite: false);
52          iwc = new IndexWriterConfig(analyzer);
53          iwc.setOpenMode(OpenMode.CREATE_OR_APPEND);
54          iwc.setRAMBufferSizeMB(4096.0);
55          writer = new IndexWriter(ramDirectory, iwc);
56          files = new File(originDataPath).listFiles();
57          fsDirectory = FSDirectory.open(Paths.get(indexPath));
58      }
```

下面是 main 函数部分

迭代为每个文件创建索引之后,从内存中将索引结果转入磁盘中,以便后面搜索.

```
246         if(fsDirectory.listAll().length == 0) {
247             // 迭代处理每个.txt文件
248             for (File file : files) {
249                 System.out.print("creating index for " + file.getName()
    );
250                 createIndexPerFile(file);
251                 System.out.println("\n      " + file.getName() + "
    done!");
252             }
253             writer.forceMerge( maxNumSegments: 1);
254             writer.close();
255         }
256         for (String file : ramDirectory.listAll()) {
257             fsDirectory.copyFrom(ramDirectory, file, file, IOContext
    .DEFAULT);
258         }
```

4. 查询的实现

·高亮的实现：我的高亮没有使用现成的 highlight 类，而是自己利用 NLPIR 的接口先进行分词，然后进行匹配.

```
81      private static String hightlight(String text, String data)throws
    IOException{
82          // 防止token生成错误,设立flag
83          // flag = true - 已找到匹配
84          //      = false - 尚未找到匹配
85          boolean flag = false;
86          if(data == null)
87              return data;
88          NLPIRTokenizerAnalyzer nta = new NLPIRTokenizerAnalyzer( data: "",
     encoding: 1,  sLicenceCode: "",  userDict: "",  bOverwrite: false);
89          TokenStream ts = nta.tokenStream( fieldName: "word", text);
90          ts.reset();
91          CharTermAttribute term = ts.getAttribute(CharTermAttribute.class);
92          while(ts.incrementToken()){
93              String pattern = term.toString();
94
95              if(pattern.equals(""))
96                  break;
97              if(data.indexOf(pattern) < 0)
98                  break;
99              flag = true;
100             data = data.replaceAll(pattern,  replacement: "<font color=\"red\">"
    + pattern + "</font>");
101         }
102         ts.end();
103         ts.close();
104         nta.close();
105         // 万一生成token的时候出错,导致没有匹配,那么就再试一下text本身.
106         if(flag == false && data.indexOf(text) > -1)
107             data = data.replaceAll(text,  replacement: "<font color=\"red\">" +
    text + "</font>");
108         return data;
109     }
110
```

·查询模块 - search 函数的实现:

search 函数返回 ArrayList.下面是初始化:

```
30      public static ArrayList<Map<String,String>> search(String text) throws
    IOException{
31
32          analyzer = new NLPIRTokenizerAnalyzer( data: "",
     encoding: 1,  sLicenceCode: "",  userDict: "",  bOverwrite: false);
33          directory = FSDirectory.open(Paths.get(indexPath));
34          DirectoryReader reader = DirectoryReader.open(directory);
35          IndexSearcher searcher = new IndexSearcher(reader);
36          QueryParser parser = new QueryParser( f: "contents", analyzer);
37          ArrayList<Map<String, String>> result = new ArrayList<>();
38          String contents;
```

然后是根据 index 得到的标签通过 hitDoc.get(“title”)等方式得到对应字符串,并且通过 map.put(”title”,str)的方式加入 HashMap 中,最后再加入到 ArrayList 中,并且返回.

5. 搜索界面的实现

由于我不是对前端设计很感兴趣,所以这部分没有花费太多精力设计.

代码如下:通过 post 方法,传递搜索串到 result.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java"
 pageEncoding="UTF-8"%>
<html>
<head>
    <title>泽文的搜索引擎</title>
</head>
<body>
<br><br>
<form method = "POST" action = "result.jsp">
    <p align = "center"><font size = "12" face="Microsoft YaHei" color
 = "#f4a460">泽文的搜索引擎</font></p><br><br>
    <p align = "center">
        <font size = "12">
            <input type = "text" name = "query" style = "..." id="kw
 "><input type = "submit" value = "搜索" style = "..." id="su">
        </font>
    </p>
</form>

<style>
    body
    {
        background:url(./green.png);
        background-size:100% 100%;
        background-repeat:no-repeat;
        padding-top:80px;
    }
</style>
</body>
</html>
```

6. 搜索结果界面的实现

这部分主要是嵌入的 java 代码.

首先读取几个参数.

这部分还要考虑到搜索结果页面还可以继续搜索,

并且还要分页,跳转,储存搜索记录.

所以相比于初始界面要多一些东西.

这是开始部分的处理参数部分:

需要处理搜索串,页数,以及搜索记录.

```
<%
    String queryText, queryPage, queryNext, queryLast;
    String strHistory;
    request.setCharacterEncoding("UTF-8");
    queryText = request.getParameter("query");
    queryPage = request.getParameter("page");
    queryLast = request.getParameter("last");
    strHistory = request.getParameter("history");
    if(strHistory == null)
        strHistory = "    ";
    if(strHistory != null && !strHistory.equals("null") && queryText != null && !strHistory.contains(queryText))
        strHistory = "    "+queryText + "<br>" +strHistory ;
    if(queryPage == null || queryPage.equals(""))
        queryPage = "1";
    queryNext = Integer.toString(Integer.parseInt(queryPage)+1);
    if(queryLast != null && !queryLast.equals(queryText)){
        queryPage = "1";
        queryNext = "2";
    }
%>
```

这是本页面的 post 部分,包括查询字符串和跳转

```
<form method = "POST" action = "result.jsp">
    <input type = "text" name = "query" style = "..." value="<%= queryText %>">
    <input type = "submit" value = "搜索" style = "...">
    <input type = "submit" value = "跳转到第" style = "...">
    <input type = "text" name = "page" style = "..." value="<%= queryNext %>" >
    <input type = "hidden" name = "history" value="<%= strHistory %>" >

    <input type = "hidden" name = "last" value="<%= queryText %>" >
    页
</form>
<br><br>
```

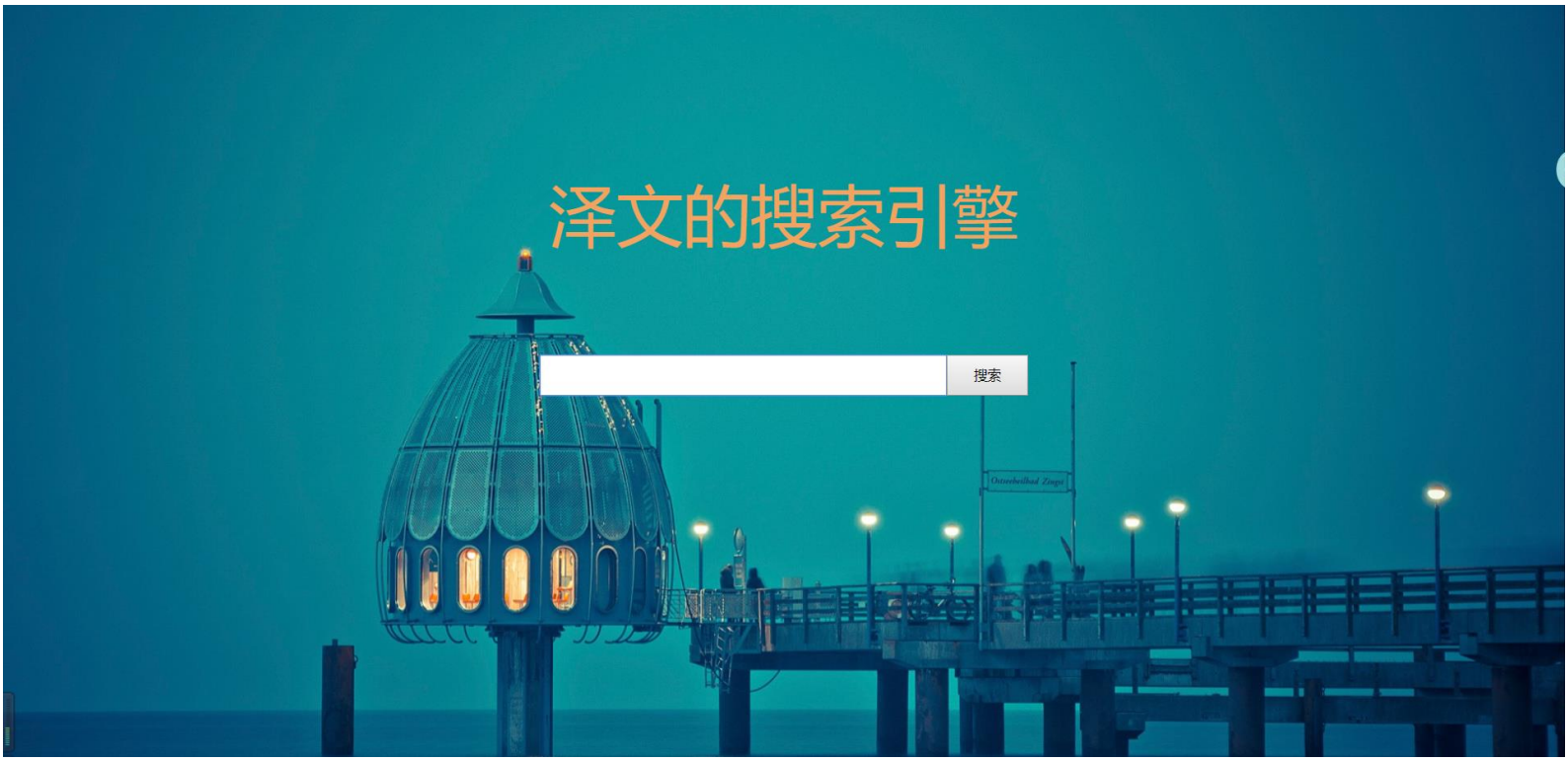
这是后面输出的部分,包括调用 Search 类的查询函数,以及界面结果的格式化输出.

```
<%
    System.out.println(queryText);
    boolean flag = false;
    Search searcher = new Search();
    ArrayList<Map<String,String>> totalResults, results=new ArrayList<>();
    int Page = 1, size = 0;
    if( queryPage != null && queryPage.length()!=0)
        Page = Integer.parseInt(queryPage);
    if(queryText != null){

        totalResults = searcher.search(queryText);
        size = totalResults.size();
        out.println("<font color = \"green\" size = \"1\">" + "找到了 " +
size + " 个结果");
        out.println("<font size = \"1\">" + "每页最多显示15条,当前是第 " +
Page + " 页, 一共" + (size / 15 + 1) + "页");
        out.println("</font>" + "<br><br>");
        if(size > (Page-1)*15)
            for (int i = 0; i < 15; i++) {
                if((Page-1)*15+i < size)
                    results.add(totalResults.get((Page-1)*15+i));
            }
        else {
            out.println("<font color = \"red\" size = \"4\">");
            out.print("抱歉,超出了页数范围: 1 - " + (size/15+1));
            out.println("</font>" + "<br>");
            flag = true;
        }
        if(results != null && results.size() != 0 ){
            String strBody, strTitle, strUrl, strScore, strKeywords;
            for(int i = 0 ; i < results.size() ; i ++){
                Map<String,String> map = results.get(i);

                strTitle = map.get("title");
                strBody = map.get("contents");
                strUrl = map.get("url");
                strScore = map.get("score");
                strKeywords = map.get("keywords");

                out.println("<font color = \"blue\" size = \"4\">");
                out.print("<a href=\"" + strUrl + "\">" + strTitle + "</a>");
                out.println("</font>" + "<br>");
                out.println("<font color = \"black\" size = \"1\">" +
"<strong>score</strong>: " + strScore + "
<strong>关键字</strong>:" + (strKeywords==null?"":strKeywords));
                out.println("</font>" + "<br>");
                out.println("<font color = \"black\" size = \"3\">" +
strBody + "<br>" + "<br>");
            }
            out.println("<br>");
        }
        else if(flag == false){
            out.println("<font color = \"red\" size = \"4\">");
            out.print("抱歉,没有找到" + queryText);
            out.println("</font>" + "<br>");
        }
        out.println("搜索记录:<br>");
        out.println("<font color = \"purple\" size = \"3\">");
        out.println(strHistory);
        out.println("</font>" + "<br>");
    }
```

【实验结果说明及演示】

实现的功能: 高亮显示, 搜索记录, 结果分页.

初始页面:



搜索”篮球”的结果:

篮球 搜索 跳转到第 2 页

找到了 1829 个结果 每页最多显示15条,当前是第 1 页, 一共122页

第二届薪火阵营训练营开营_阿联现身用篮球传递爱_篮球-CBA_新浪竞技风暴_新浪网
score: 9.722 关键字:第二届薪火阵营训练营开营 阿联现身用篮球传递爱,易建联
新浪体育讯 北京时间8月21日，第二届易建联“传递爱-薪火阵营”训练营开营仪式在新疆乌鲁木齐新疆大学本部篮球馆举行。易建联出席了开营仪式，在开营仪式上，易建联表示可以通过“薪火阵营”传递爱， 让更多喜欢篮球的孩子实现梦想。“传递爱-薪火阵营”篮球训练营由易建联发起，由新疆维吾尔自治区人民政府以及中国宋庆龄基金会主办，中国宋庆龄基金会文化艺术中心、“传递爱”公益活动、百麟投资管理有限公司承办的面 向...

闵鹿蕾造访欧洲篮球总部 合影登欧冠官网首页(图)_篮球-CBA_新浪竞技风暴_新浪网
score: 9.714 关键字:闵鹿蕾造访欧洲篮球总部 合影登欧冠官网首页(图),闵鹿蕾
新浪体育讯 北京时间6月7日消息，北京队主教练闵鹿蕾继续着自己的欧洲学习之旅，近日他造访了位于巴塞罗那的欧洲篮球总部，并现场观看了巴塞罗那与皇家马德里的西甲决赛。欧洲篮球网的记者对闵鹿蕾进行了专访。欧洲篮球网认为，闵鹿蕾作为北京队的主教练，在上赛季给自己的球队带来了很大的变化，他们打进了总决赛并击败了卫冕冠军广东队，夺得了北京队历史上第一个CBA冠军奖杯。以下是欧洲篮球网对于闵鹿蕾的专访：我想学习...

网络新闻首次荣获中国篮球新闻奖 新浪体育开先河_篮球-CBA_新浪竞技风暴_新浪网
score: 9.663 关键字:网络新闻首次荣获中国篮球新闻奖 新浪体育开先河
新浪体育讯 北京时间4月16日消息，“2011年度乔丹体育中国篮球新闻奖颁奖典礼”在北京天坛饭店举行。这次颁奖典礼一共颁出了25份新闻作品奖，其中包括电视类(7份)、广播类(2份)、文字类(10份)、摄影类(8份)作品获奖。其中新浪体育的一篇作品获得文字类奖，这是互联网中国篮球报道文字作品第一次获得该奖项。文字类作品中，新浪体育伏明的《谁有勇气创造中国篮球历史》首次代表互联网获此殊荣，互联网作品能...

广东广宁一村篮球协会成化解村民纠纷平台|广东|篮球协会|纠纷_新浪新闻
score: 9.652 关键字:广东,篮球协会,纠纷
南方农村报讯(记者黄进) 8月22日，广东肇庆市广宁县横山镇曾宽村旁横跨绥江的一座桥上，一名10来岁的小男孩骑着小轮自行车急速驶过。他身上的红色球衣上，“HEAT6”(美职篮迈阿密热火队球员詹姆斯的球衣号码)图标掉得只剩下一个“T”字。他告诉南方农村报记者，其爱打篮球，喜欢詹姆斯，也是“广宁横山曾宽村委会篮球协会”的会员。曾宽村篮球协会是该村村民自发成立的一个体育组织，但在实际运行过程中，该协会在

可以看出,有高亮,有分页,跳转.

再搜索”习近平”:

习近平　搜索　跳转到第 2 页

找到了 1981 个结果 每页最多显示15条,当前是第 1 页, 一共133页

俄罗斯共产党主席:习近平懂得培养和用人方法|俄罗斯|习近平|人才_新浪新闻
score: 9.598 关键字:俄罗斯,习近平,人才,国家主席习近平首次外访
新华网莫斯科3月18日电通讯：难忘的会面——三位俄罗斯友人谈与习近平的交往故事新华社记者刘恺　娄琛　吕国栋国家主席习近平对俄罗斯进行国事访问前夕，三位俄罗斯友人回忆起三年前与习近平的会面情形，从他们的讲述中可以看出，习近平的友好和真诚给他们留下了极为深刻的印象。今年69岁的俄罗斯共产党主席久加诺夫是一位博学而健谈的学者型政治家，他颇有“中国缘”，曾7次到访中国。执掌俄共20年，久加诺夫与许多中国领...

美国媒体热议习近平访美_新闻中心_新浪网
score: 9.563 关键字:美国媒体热议习近平访美,习近平
中新社纽约2月15日电　(记者　李洋)美国主流报章15日纷纷在显要位置刊登中国国家副主席习近平访问美国的报道，均认为此访对未来的中美关系具有重要意义，冀望未来的两国关系稳定发展。美国《纽约时报》15日头版全面报道了习近平在华盛顿的访问行程。报道说，习近平和美国副总统拜登就一系列相互关心的问题进行了开诚布公的讨论，虽然议题严肃，但会谈中也不时传出笑声。该报说，在习近平与美国总统奥巴马的会晤中，两人也...

外媒关注习近平当选国家主席　期盼其推动改革|习近平|国家主席|改革_新浪新闻
score: 9.543 关键字:习近平,国家主席,改革,2013年全国两会
中新网3月14日电　十二届全国人大一次会议14日举行第四次全体会议，习近平当选中国国家主席，李源潮当选国家副主席。外媒对此纷纷予以热议，期待他以更大的魄力，推进中国的各项改革。去年11月当选中共中央总书记以来，习近平提出“中国梦”口号，显示出提高党内凝聚力的态度。履新后不久，习近平在国家博物馆发表讲话，首次提及“中国梦”的概念。他指出，实现中华民族伟大复兴，就是中华民族近代以来最伟大的梦想。美国《...

习近平将逗留美国小镇1小时　与17位老友叙旧　新闻中心　新浪网

并且末尾有搜索记录:

[illegible]断，其“官话民说”、“有话直说”的风格凸显了他极富故事性和人情味的独特话语方式。从华盛顿到艾奥瓦州再到洛杉矶州，习近平几乎每到一地，都会用最朴实的民间语言讲述一个故事或阐述一个复杂的道理。这次访问，让中美两国原本并不为人熟知的两个小镇一下名扬四海，成为媒体关注...

习近平:中坦两国堪称全天候朋友|习近平|出访|坦桑尼亚_新浪新闻
score: 9.419 关键字:习近平,出访,坦桑尼亚,国家主席习近平首次外访
新华网达累斯萨拉姆3月24日电(记者　张平　程志良　李拯宇)国家主席习近平24日抵达达累斯萨拉姆，开始对坦桑尼亚进行国事访问。达累斯萨拉姆风和日丽，阳光灿烂。尼雷尔国际机场洋溢着一派友好热烈气氛，中坦两国国旗迎风飘扬，当地民众身着节日盛装，挥动着五彩缤纷的旗帜，兴高采烈地迎候中国贵宾的到来。当地时间16时25分许，习近平乘坐的专机抵达机场。习近平和夫人彭丽媛走下舷梯，受到坦桑尼亚总统基奎特和夫人热...

搜索记录:
习近平
篮球

跳转到第 20 页:

并且如果跳转页面非法,则会提示(容错性):

习近平 搜索 跳转到第 135 页

找到了 1981 个结果 每页最多显示15条,当前是第 134 页, 一共133页

抱歉,超出了页数范围: 1 - 133

搜索记录:

习近平

篮球

习近平

找到了 1981 个结果 每页最多显示15条,当前是第 20 页, 一共133页

法国总统致电祝贺习近平当选中国国家主席|习近平|中法关系|奥朗德_新浪新闻

score: 8.724 关键字:习近平,中法关系,奥朗德,2013年全国两会

中新网3月15日电 据外交部网站消息，3月15日晚，中国国家主席习近平应约同法国总统奥朗德通电话。奥朗德代表法国人民热烈祝贺习主席。奥朗德表示，法中两国都是联合国安理会常任理事国，都主张世界多极化，在许多重大问题上有相同或相近的主张。中国对促进世界展具有举足轻重的作用。法方同中方加强全面战略伙伴关系的决心是坚定的。明年是两国建交50周年，法方愿以此为契机，同中方一道，

习近平:新疆工作在国家工作中具有特殊战略地位_新闻中心_新浪网

score: 8.714 关键字:习近平:新疆工作在国家工作中具有特殊战略地位,习近平,新疆

新华网北京3月9日电 中共中央政治局常委、国家副主席习近平参加了新疆代表团审议。代表们围绕会议议题畅所欲言，会场气氛活跃。在的发言后，习近平指出，新疆工作在党和国家工作全局中具有特殊重要的战略地位。他希望自治区和生产建设兵团按照中央新疆工作座谈会动摇地坚持发展和稳定两手抓、两手都要硬，突出抓好科学发展，切实保障和改善民生，始终高举团结稳定旗帜，谱写科学发展、稳疆兴疆

习近平会见上合组织秘书长:望发挥维护稳定作用|习近平|上合组织|维护地区稳定_新浪新闻

score: 8.714 关键字:习近平,上合组织,维护地区稳定

原标题：习近平：希望上合组织为维护地区安全稳定发挥重要作用国际在线消息 （记者 丁宁）：中共中央总书记习近平22日北京会见上海书长梅津采夫时表示，希望上海合作组织落实好成员国领导人达成的重要共识，为维护地区安全稳定、促进成员国共同发展繁荣发挥重要作

【实验总结】

亮点:

1. 使用的工具:
   为了效果的极致,也为了让自己更贴近工业界,我使用的都是目前业界最广泛好评的工具,以及最新的版本.
   比如 IntelliJ IDEA 这款比较新的 IDE,个人认为大部分场合都完胜 Eclipse.
   比如 Lucene 和 ICTCLAS,没有用 FTP 中的老版本,而是自己找的官方最新版.
2. 高性能:
   创建索引的过程是在内存中全部完成之后再转移到磁盘中,这样大大加快了创建索引的过程.
3. 高亮:
   高亮部分没有使用现成轮子,而是通过 ICTCLAS 的 API,得到 token 流直接进行字符串匹配来替换.并且考虑到 ICTCLAS 分词工具偶尔出现的编码转换错误,进行了额外处理.
4. 分页:
   没有用数据库进行分页的管理,而是通过简单的 post 传递,以及页数的计算进行分页显示.
   并且考虑到了超出页面限制的情况.
5. 搜索记录
   同样没有使用数据库.而是使用一个数组来存储搜索记录,每次调用 result 时通过 post 传递,来进行显示.
6. 容错性
   为了使最后的作品更加鲁棒,所以考虑了很多细节.
   比如非法页数,非法字符等.

找到了 0 个结果 每页最多显示15条,当前是第 1 页, 一共1页

抱歉,没有找到#$%^&*
搜索记录:
#$%^&*
이번생은처음이라

找到了 1981 个结果 每页最多显示15条,当前是第 200 页, 一共133页

抱歉,超出了页数范围: 1 - 133
搜索记录:
习近平

找到了 1981 个结果 每页最多显示15条,当前是第 -1 页, 一共133页

抱歉,超出了页数范围: 1 - 133
搜索记录:
习近平

不足:

1. 由于时间精力不足,没来得及实现推荐系统,高级搜索,模糊搜索等,这些如果加进来,那么会棒很多!
2. 性能还是有些不足,比如有个同学使用了多线程搜索,这一点可以改进.
3. 前端界面不是很美观,虽然也能看得过去.
4. 没有使用 cookie,数据库,如果想要做大这个作品,就必须加这些东西.

以上就是不足和可以改进的地方!

总结:

这次实验的收获还是非常大的!借这个机会学习了 java,
了解并学会掌握了 lucene,ictclas,IntelliJ 的使用,还学到了 jsp.
自学能力再次提升!
虽然三个实验只占 20 分,并且没有多少区分度,但是本次试验的收获让我非常开心!
感谢!