Web 信息处理与应用实验一程序说明——张劲暾（PB16111485）

简单的搜索引擎实现

【实验内容】

本实验是开发出面向 DBWorld 实时信息的灵活、多功能的搜索引擎，本次实验的内容主要包括以下几部分:

1. 针对研究人员对 DBWorld 的信息搜索需求,设计了一个 DBWorld 搜索引擎,给出总体设计
2. 针对 DBWorld 原始数据的特点,对 DBWolrd 进行信息抽取,包括 DBWorld 网页中的地点、时间、主题等信息
3. 实现一个基于 WEB 的 DBWolrd 搜索引擎

功能需求:

1. 提供定时抓取,定时更新 DBworld 信息的功能,以确保数据的及时、准确、有效性。
2. 提供多种搜索选项,包括提交截止日期,会议起止日期,地点,主题等。
3. 提供一个搜索和显示界面。

性能需求:

1. 需要保证 DBWorld 信息的查准率。
2. 需要保证搜索引擎的快速响应。

【实验环境】

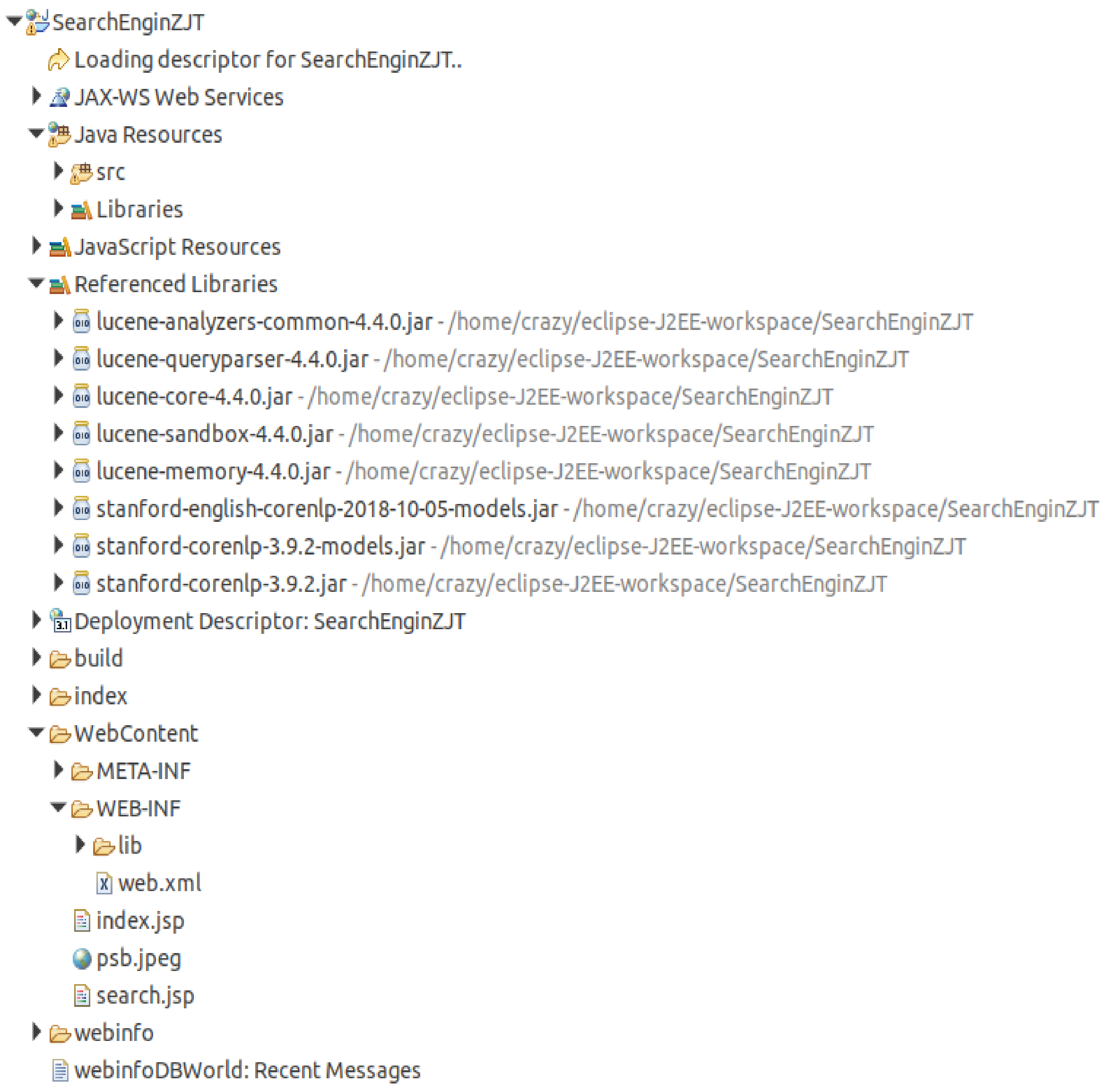
编程语言：Java 1.8.0

编程环境：Ubuntu 17.10

运行环境：Tomcat v8.5 + jdk-8

使用工具：eclipse-jee

工程结构如下:

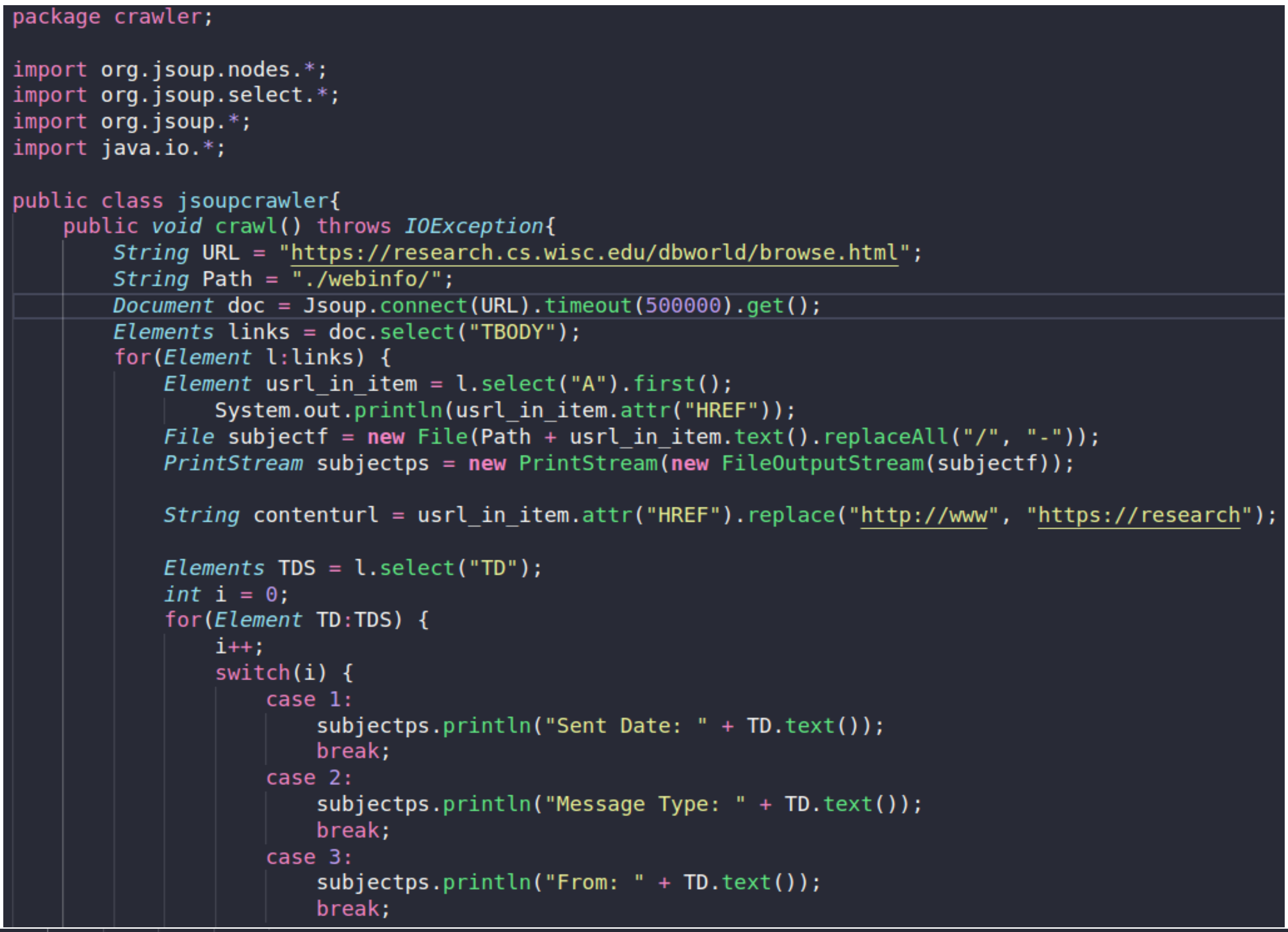


```
SearchEnginZJT
    Loading descriptor for SearchEnginZJT..
    JAX-WS Web Services
    Java Resources
        src
        Libraries
    JavaScript Resources
    Referenced Libraries
        lucene-analyzers-common-4.4.0.jar - /home/crazy/eclipse-J2EE-workspace/SearchEnginZJT
        lucene-queryparser-4.4.0.jar - /home/crazy/eclipse-J2EE-workspace/SearchEnginZJT
        lucene-core-4.4.0.jar - /home/crazy/eclipse-J2EE-workspace/SearchEnginZJT
        lucene-sandbox-4.4.0.jar - /home/crazy/eclipse-J2EE-workspace/SearchEnginZJT
        lucene-memory-4.4.0.jar - /home/crazy/eclipse-J2EE-workspace/SearchEnginZJT
        stanford-english-corenlp-2018-10-05-models.jar - /home/crazy/eclipse-J2EE-workspace/SearchEnginZJT
        stanford-corenlp-3.9.2-models.jar - /home/crazy/eclipse-J2EE-workspace/SearchEnginZJT
        stanford-corenlp-3.9.2.jar - /home/crazy/eclipse-J2EE-workspace/SearchEnginZJT
    Deployment Descriptor: SearchEnginZJT
    build
    index
    WebContent
        META-INF
        WEB-INF
            lib
            web.xml
        index.jsp
        psb.jpeg
        search.jsp
    webinfo
    webinfoDBWorld: Recent Messages
```

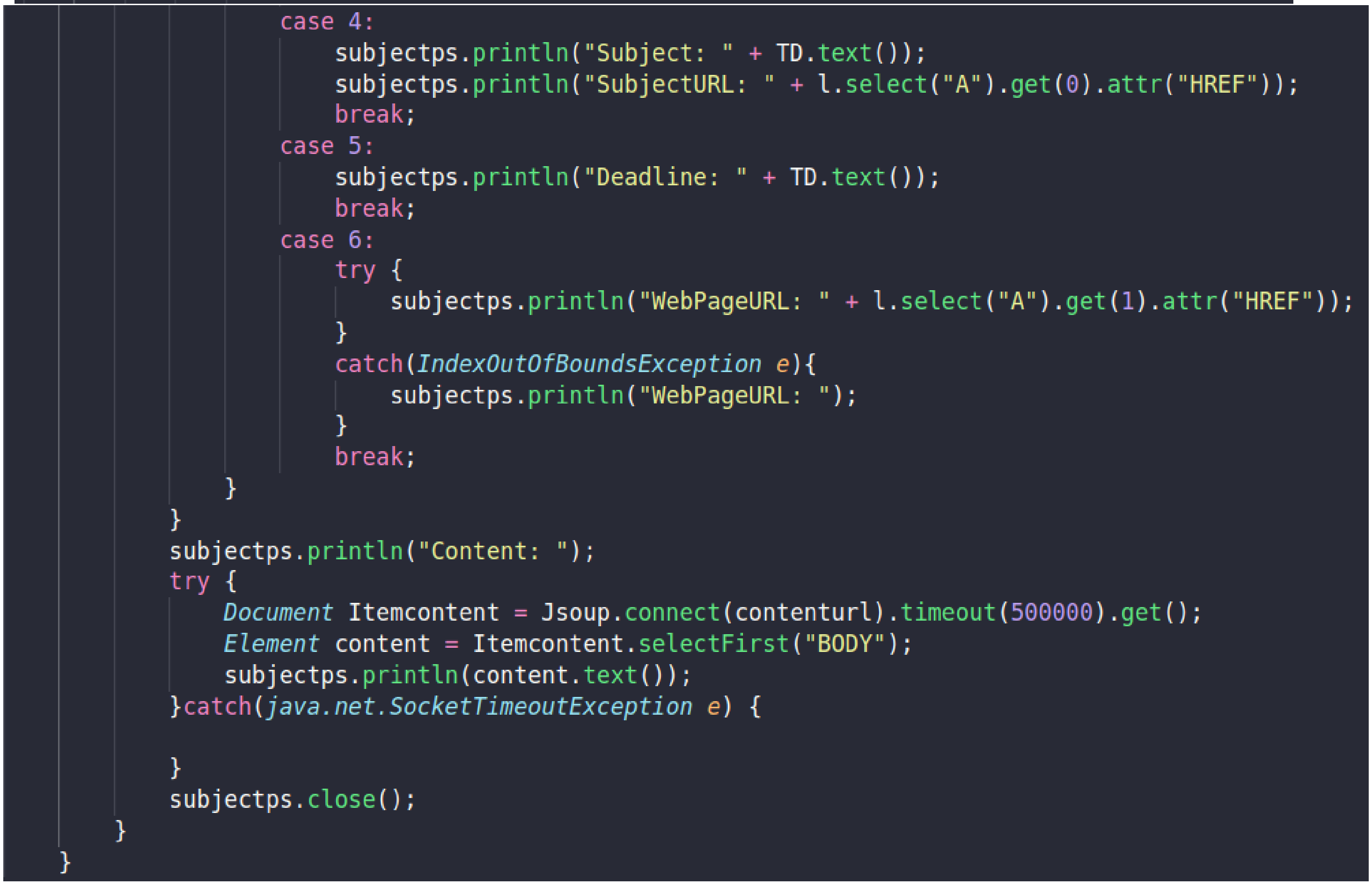
【实验步骤及方法】【总体设计见实验总结（第 13 页）】

一、 数据爬取

由于本实验需要爬取的网站网页结构相对简单，即只包含初始的目录页面和二级的简介文本页面，对于关于主题的外部链接，由于不是本网站的内容，所以并没有考虑，那么网站的结构就是一个简单的二层结构，不需要套用复杂的爬虫框架，所以本实验中只使用 jsoup 工具对网页进行简单的解析，并且根据目录页的信息保存文件：

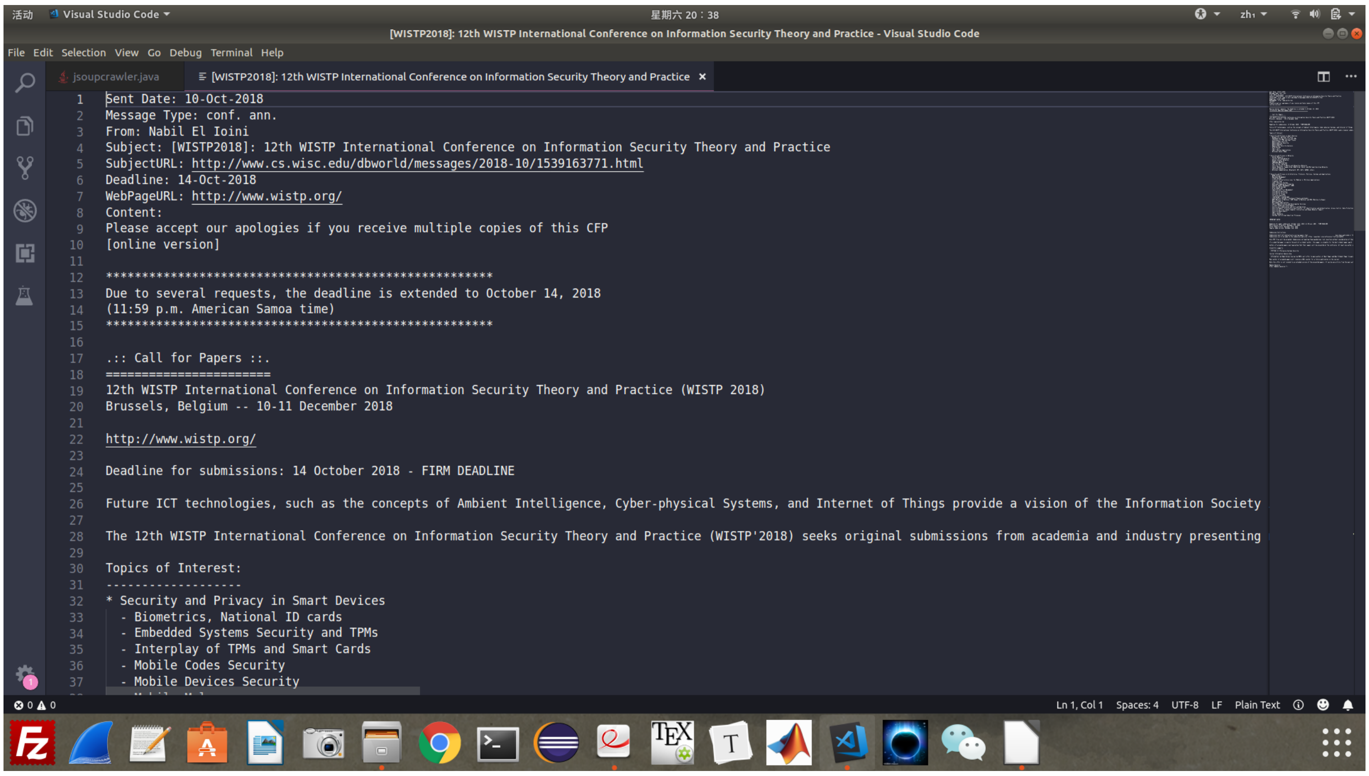


```java
package crawler;

import org.jsoup.nodes.*;
import org.jsoup.select.*;
import org.jsoup.*;
import java.io.*;

public class jsoupcrawler{
    public void crawl() throws IOException{
        String URL = "https://research.cs.wisc.edu/dbworld/browse.html";
        String Path = "./webinfo/";
        Document doc = Jsoup.connect(URL).timeout(500000).get();
        Elements links = doc.select("TBODY");
        for(Element l:links) {
            Element usrl_in_item = l.select("A").first();
                System.out.println(usrl_in_item.attr("HREF"));
            File subjectf = new File(Path + usrl_in_item.text().replaceAll("/", "-"));
            PrintStream subjectps = new PrintStream(new FileOutputStream(subjectf));

            String contenturl = usrl_in_item.attr("HREF").replace("http://www", "https://research");

            Elements TDS = l.select("TD");
            int i = 0;
            for(Element TD:TDS) {
                i++;
                switch(i) {
                    case 1:
                        subjectps.println("Sent Date: " + TD.text());
                        break;
                    case 2:
                        subjectps.println("Message Type: " + TD.text());
                        break;
                    case 3:
                        subjectps.println("From: " + TD.text());
                        break;
```



```java
                    case 4:
                        subjectps.println("Subject: " + TD.text());
                        subjectps.println("SubjectURL: " + l.select("A").get(0).attr("HREF"));
                        break;
                    case 5:
                        subjectps.println("Deadline: " + TD.text());
                        break;
                    case 6:
                        try {
                            subjectps.println("WebPageURL: " + l.select("A").get(1).attr("HREF"));
                        }
                        catch(IndexOutOfBoundsException e){
                            subjectps.println("WebPageURL: ");
                        }
                        break;
                }
            }
            subjectps.println("Content: ");
            try {
                Document Itemcontent = Jsoup.connect(contenturl).timeout(500000).get();
                Element content = Itemcontent.selectFirst("BODY");
                subjectps.println(content.text());
            }catch(java.net.SocketTimeoutException e) {

            }
            subjectps.close();
        }
    }
```

对于目录页的每一个 TBODY 表格项，依次抽取发布日期，消息类型，发布者，主题，截止日期和内容并保存在相关文件中。
爬取文件效果如下:

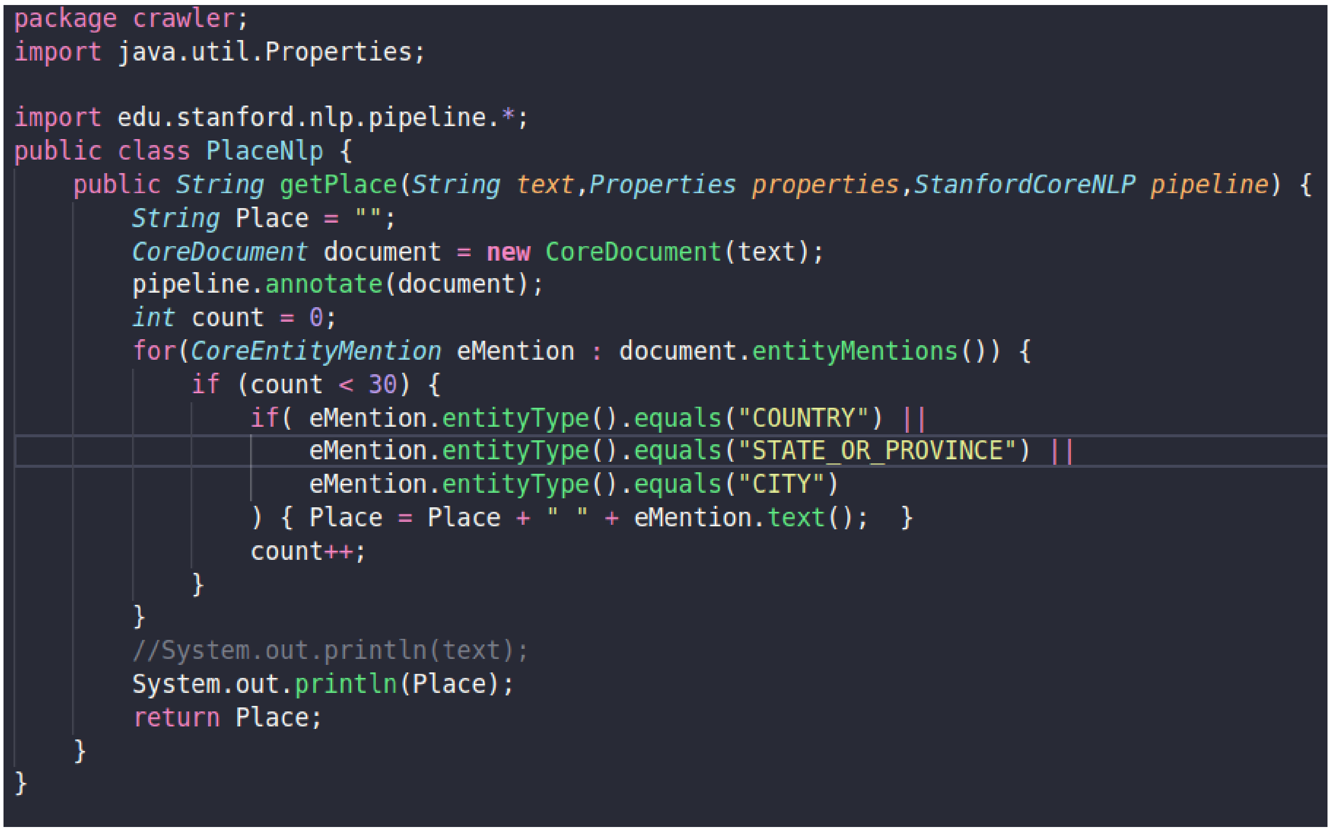

活动
Visual Studio Code
星期六 20：38
[WISTP2018]: 12th WISTP International Conference on Information Security Theory and Practice - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
jsoupcrawler.java
[WISTP2018]: 12th WISTP International Conference on Information Security Theory and Practice
Ln 1, Col 1　Spaces: 4　UTF-8　LF　Plain Text

```
Sent Date: 10-Oct-2018
Message Type: conf. ann.
From: Nabil El Ioini
Subject: [WISTP2018]: 12th WISTP International Conference on Information Security Theory and Practice
SubjectURL: http://www.cs.wisc.edu/dbworld/messages/2018-10/1539163771.html
Deadline: 14-Oct-2018
WebPageURL: http://www.wistp.org/
Content:
Please accept our apologies if you receive multiple copies of this CFP
[online version]

******************************************************
Due to several requests, the deadline is extended to October 14, 2018
(11:59 p.m. American Samoa time)
******************************************************

.:: Call for Papers ::.
=======================
12th WISTP International Conference on Information Security Theory and Practice (WISTP 2018)
Brussels, Belgium -- 10-11 December 2018

http://www.wistp.org/

Deadline for submissions: 14 October 2018 - FIRM DEADLINE

Future ICT technologies, such as the concepts of Ambient Intelligence, Cyber-physical Systems, and Internet of Things provide a vision of the Information Society

The 12th WISTP International Conference on Information Security Theory and Practice (WISTP'2018) seeks original submissions from academia and industry presenting

Topics of Interest:
------------------
* Security and Privacy in Smart Devices
  - Biometrics, National ID cards
  - Embedded Systems Security and TPMs
  - Interplay of TPMs and Smart Cards
  - Mobile Codes Security
  - Mobile Devices Security
```

二、 实体识别抽取与索引建立

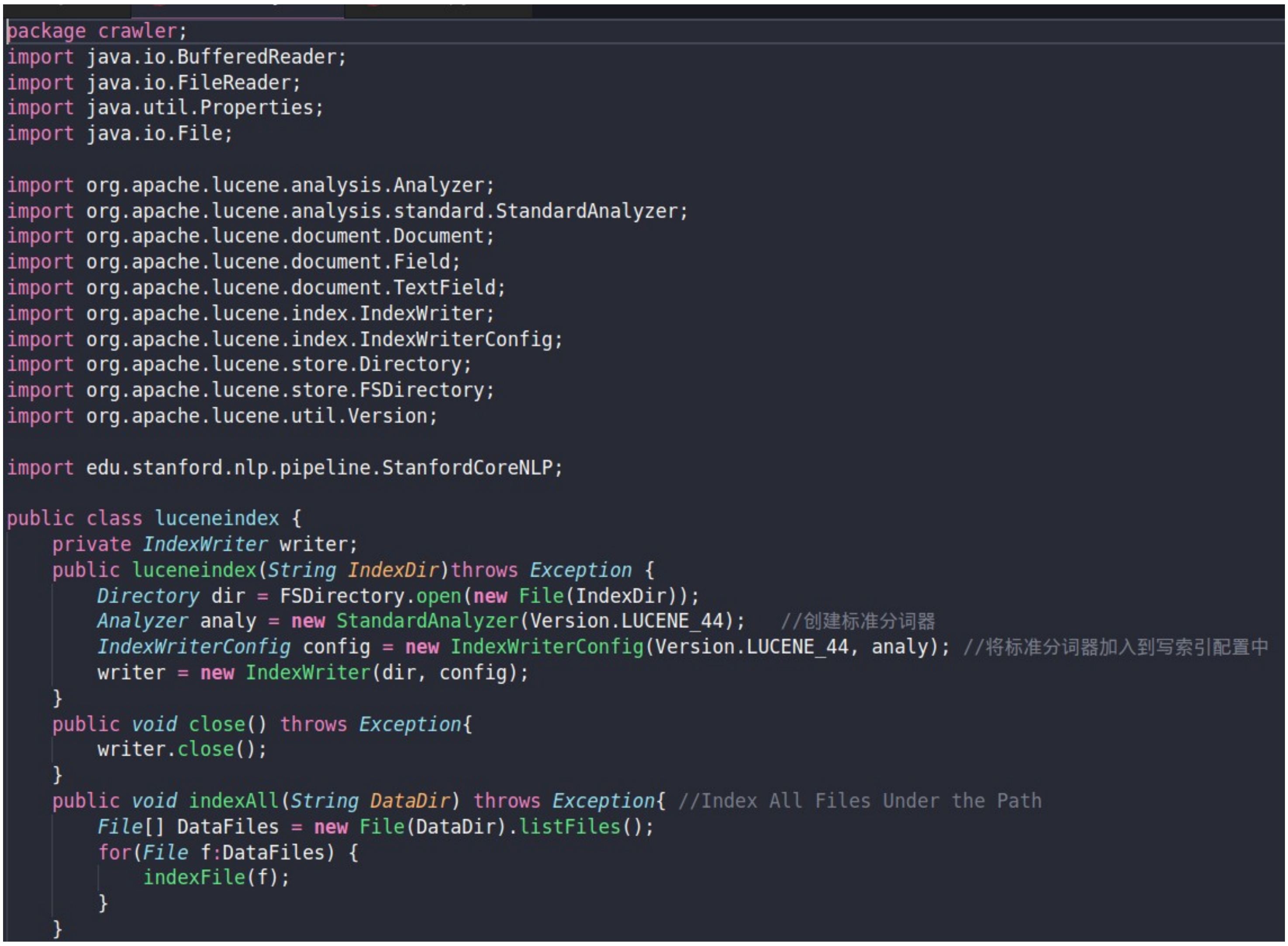
利用 lucene 建立索引，lucene 对于文件的管理是基于文本域 TextField 的，我们把文本中的各种信息和由 Stanford CoreNLP 识别得到的地点名称实体分别保存在不同的文本域 TextField 中，然后建立索引并基于文本域 TextField 查询。

Stanford CoreNLP 地点名称实体识别，对于内容中的国名，省/州名，城市名作为名称实体保存在 place 域中:

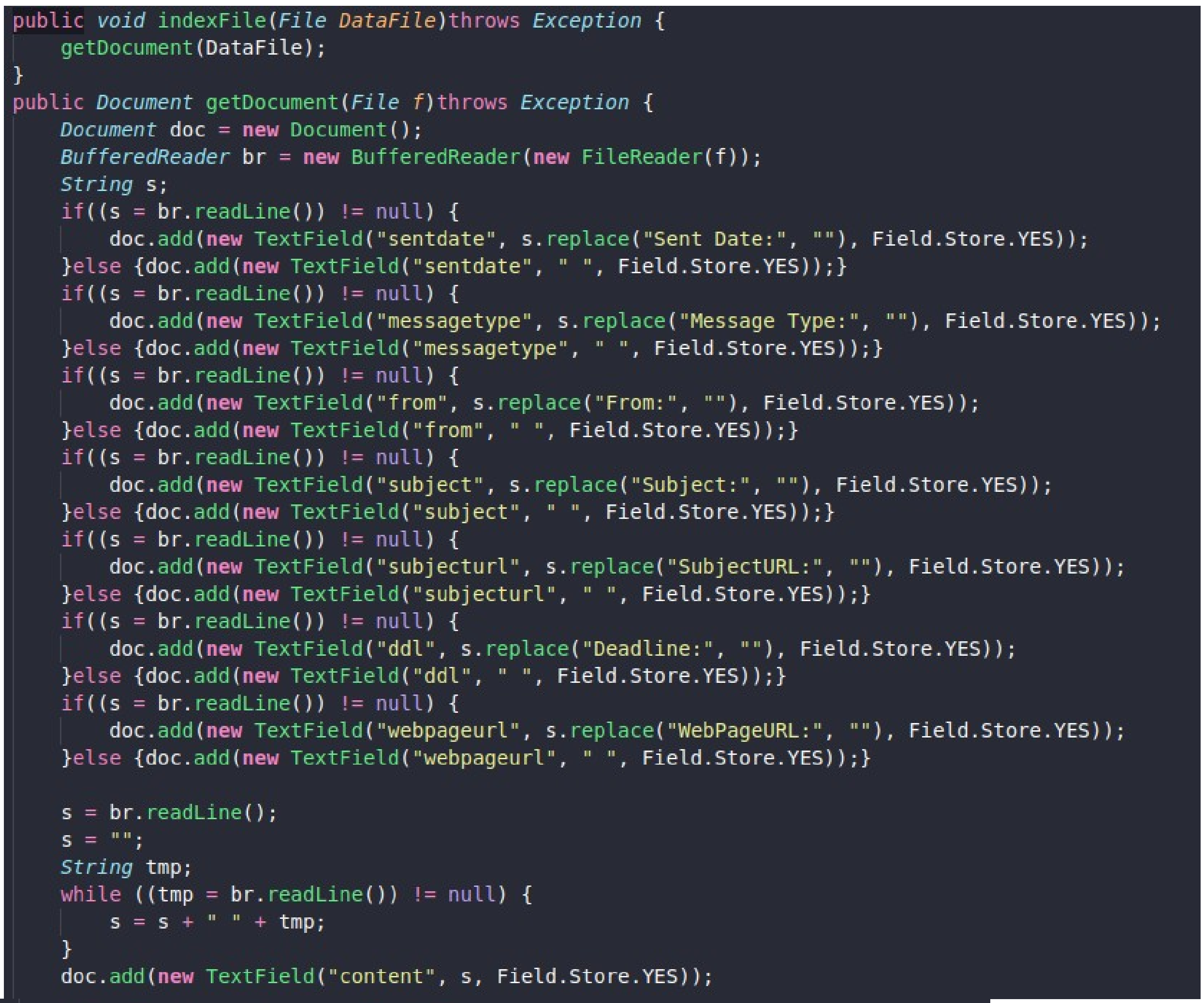


```java
package crawler;
import java.util.Properties;

import edu.stanford.nlp.pipeline.*;
public class PlaceNlp {
    public String getPlace(String text,Properties properties,StanfordCoreNLP pipeline) {
        String Place = "";
        CoreDocument document = new CoreDocument(text);
        pipeline.annotate(document);
        int count = 0;
        for(CoreEntityMention eMention : document.entityMentions()) {
            if (count < 30) {
                if( eMention.entityType().equals("COUNTRY") ||
                    eMention.entityType().equals("STATE_OR_PROVINCE") ||
                    eMention.entityType().equals("CITY")
                ) { Place = Place + " " + eMention.text();  }
                count++;
            }
        }
        //System.out.println(text);
        System.out.println(Place);
        return Place;
    }
}
```

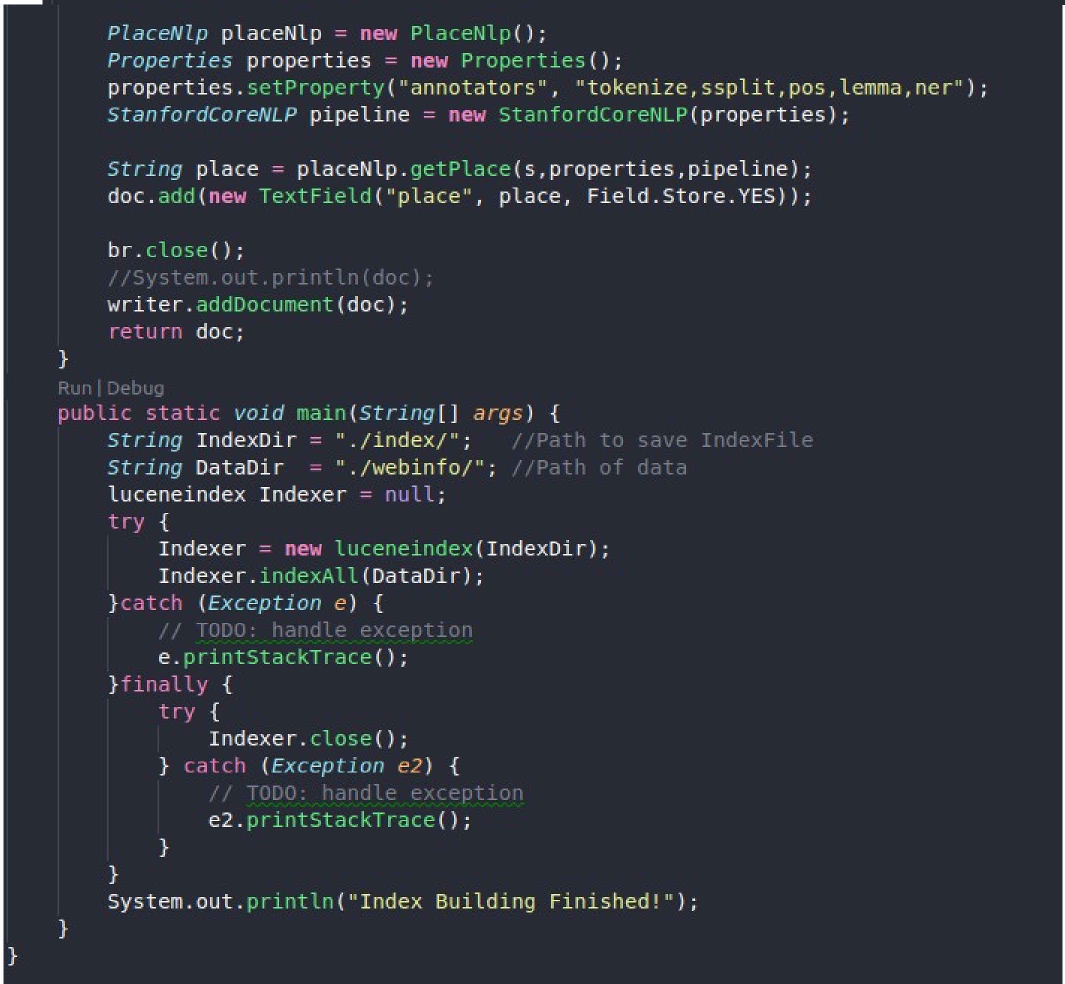
lucene 索引建立：



```java
package crawler;
import java.io.BufferedReader;
import java.io.FileReader;
import java.util.Properties;
import java.io.File;

import org.apache.lucene.analysis.Analyzer;
import org.apache.lucene.analysis.standard.StandardAnalyzer;
import org.apache.lucene.document.Document;
import org.apache.lucene.document.Field;
import org.apache.lucene.document.TextField;
import org.apache.lucene.index.IndexWriter;
import org.apache.lucene.index.IndexWriterConfig;
import org.apache.lucene.store.Directory;
import org.apache.lucene.store.FSDirectory;
import org.apache.lucene.util.Version;

import edu.stanford.nlp.pipeline.StanfordCoreNLP;

public class luceneindex {
    private IndexWriter writer;
    public luceneindex(String IndexDir)throws Exception {
        Directory dir = FSDirectory.open(new File(IndexDir));
        Analyzer analy = new StandardAnalyzer(Version.LUCENE_44);   //创建标准分词器
        IndexWriterConfig config = new IndexWriterConfig(Version.LUCENE_44, analy); //将标准分词器加入到写索引配置中
        writer = new IndexWriter(dir, config);
    }
    public void close() throws Exception{
        writer.close();
    }
    public void indexAll(String DataDir) throws Exception{ //Index All Files Under the Path
        File[] DataFiles = new File(DataDir).listFiles();
        for(File f:DataFiles) {
            indexFile(f);
        }
    }
```



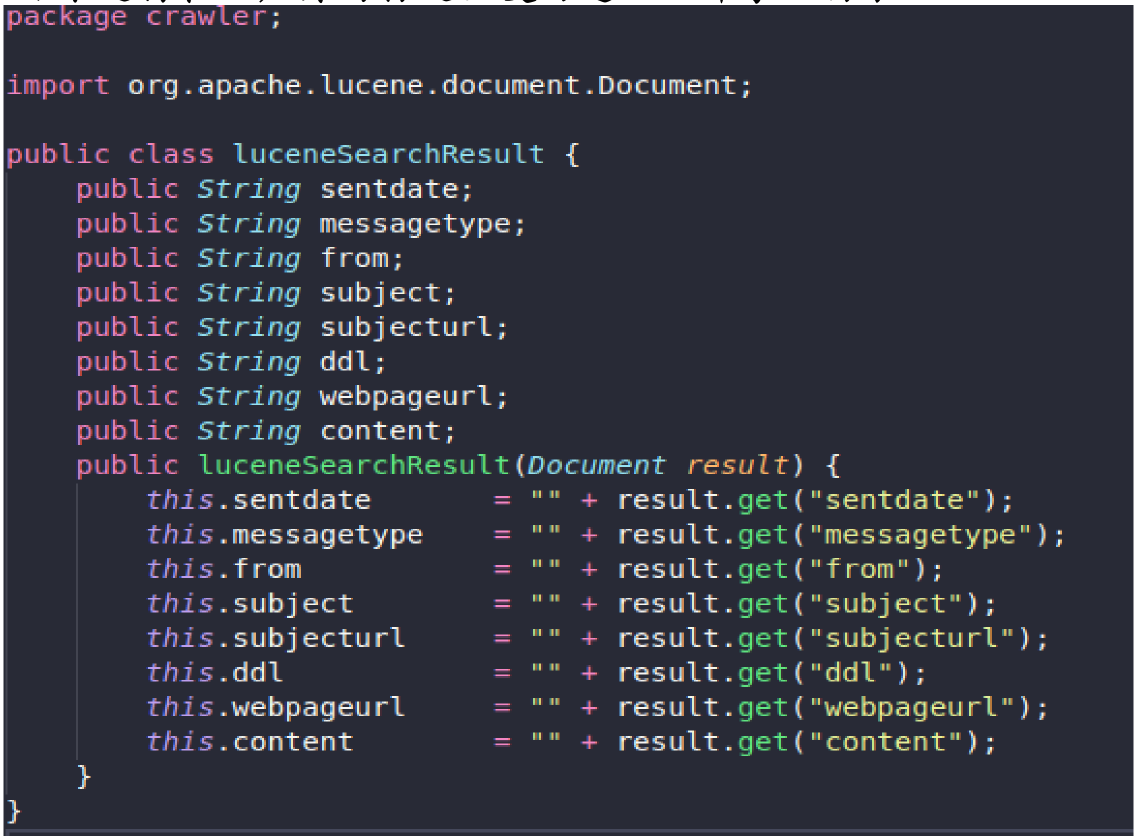
```java
public void indexFile(File DataFile)throws Exception {
    getDocument(DataFile);
}
public Document getDocument(File f)throws Exception {
    Document doc = new Document();
    BufferedReader br = new BufferedReader(new FileReader(f));
    String s;
    if((s = br.readLine()) != null) {
        doc.add(new TextField("sentdate", s.replace("Sent Date:", ""), Field.Store.YES));
    }else {doc.add(new TextField("sentdate", " ", Field.Store.YES));}
    if((s = br.readLine()) != null) {
        doc.add(new TextField("messagetype", s.replace("Message Type:", ""), Field.Store.YES));
    }else {doc.add(new TextField("messagetype", " ", Field.Store.YES));}
    if((s = br.readLine()) != null) {
        doc.add(new TextField("from", s.replace("From:", ""), Field.Store.YES));
    }else {doc.add(new TextField("from", " ", Field.Store.YES));}
    if((s = br.readLine()) != null) {
        doc.add(new TextField("subject", s.replace("Subject:", ""), Field.Store.YES));
    }else {doc.add(new TextField("subject", " ", Field.Store.YES));}
    if((s = br.readLine()) != null) {
        doc.add(new TextField("subjecturl", s.replace("SubjectURL:", ""), Field.Store.YES));
    }else {doc.add(new TextField("subjecturl", " ", Field.Store.YES));}
    if((s = br.readLine()) != null) {
        doc.add(new TextField("ddl", s.replace("Deadline:", ""), Field.Store.YES));
    }else {doc.add(new TextField("ddl", " ", Field.Store.YES));}
    if((s = br.readLine()) != null) {
        doc.add(new TextField("webpageurl", s.replace("WebPageURL:", ""), Field.Store.YES));
    }else {doc.add(new TextField("webpageurl", " ", Field.Store.YES));}

    s = br.readLine();
    s = "";
    String tmp;
    while ((tmp = br.readLine()) != null) {
        s = s + " " + tmp;
    }
    doc.add(new TextField("content", s, Field.Store.YES));
```



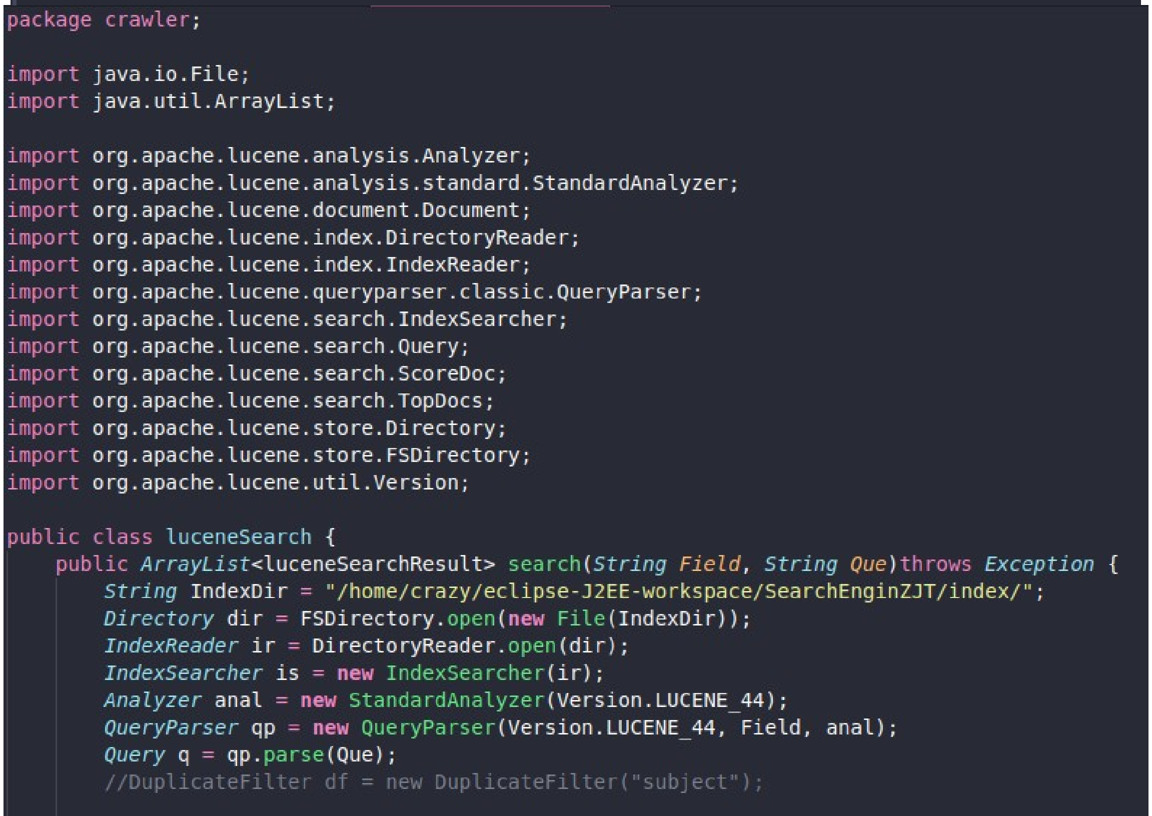
```java
        PlaceNlp placeNlp = new PlaceNlp();
        Properties properties = new Properties();
        properties.setProperty("annotators", "tokenize,ssplit,pos,lemma,ner");
        StanfordCoreNLP pipeline = new StanfordCoreNLP(properties);

        String place = placeNlp.getPlace(s,properties,pipeline);
        doc.add(new TextField("place", place, Field.Store.YES));

        br.close();
        //System.out.println(doc);
        writer.addDocument(doc);
        return doc;
    }
    Run | Debug
    public static void main(String[] args) {
        String IndexDir = "./index/";   //Path to save IndexFile
        String DataDir  = "./webinfo/"; //Path of data
        luceneindex Indexer = null;
        try {
            Indexer = new luceneindex(IndexDir);
            Indexer.indexAll(DataDir);
        }catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
        }finally {
            try {
                Indexer.close();
            } catch (Exception e2) {
                // TODO: handle exception
                e2.printStackTrace();
            }
        }
        System.out.println("Index Building Finished!");
    }
}
```

三、 给予索引的查询

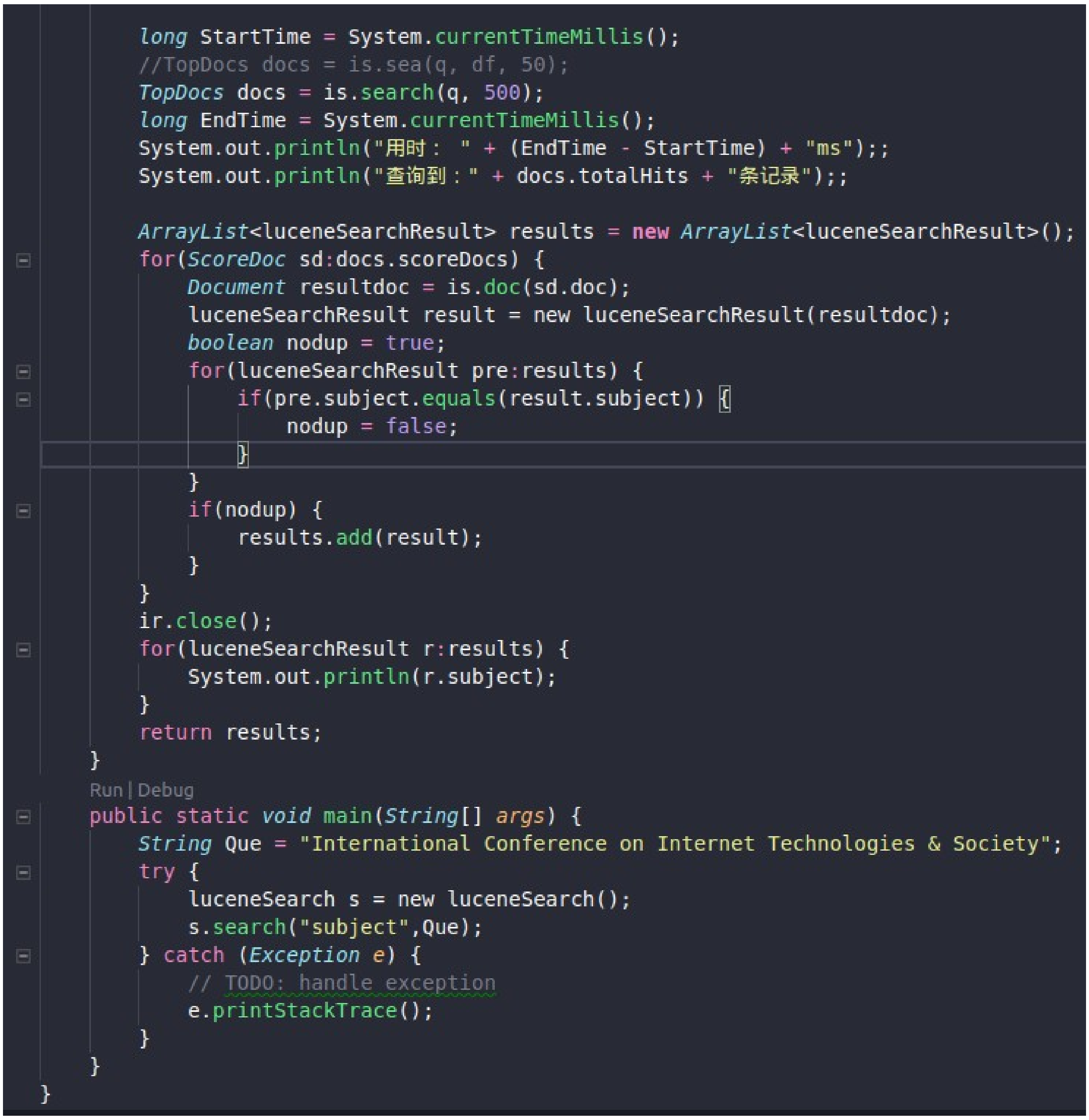
利用 lucene 提供的 QueryParser 类对于查询文本生成 Query 实例，对于指定的文本域进行查询，因为 lucene 可能将一个文本多次返回，所以利用 luceneSearchResult 类对于查询结果进行管理，并对标题重复的返回文件予以剔除。



```java
package crawler;

import org.apache.lucene.document.Document;

public class luceneSearchResult {
    public String sentdate;
    public String messagetype;
    public String from;
    public String subject;
    public String subjecturl;
    public String ddl;
    public String webpageurl;
    public String content;
    public luceneSearchResult(Document result) {
        this.sentdate       = "" + result.get("sentdate");
        this.messagetype    = "" + result.get("messagetype");
        this.from           = "" + result.get("from");
        this.subject        = "" + result.get("subject");
        this.subjecturl     = "" + result.get("subjecturl");
        this.ddl            = "" + result.get("ddl");
        this.webpageurl     = "" + result.get("webpageurl");
        this.content        = "" + result.get("content");
    }
}
```



```java
package crawler;

import java.io.File;
import java.util.ArrayList;

import org.apache.lucene.analysis.Analyzer;
import org.apache.lucene.analysis.standard.StandardAnalyzer;
import org.apache.lucene.document.Document;
import org.apache.lucene.index.DirectoryReader;
import org.apache.lucene.index.IndexReader;
import org.apache.lucene.queryparser.classic.QueryParser;
import org.apache.lucene.search.IndexSearcher;
import org.apache.lucene.search.Query;
import org.apache.lucene.search.ScoreDoc;
import org.apache.lucene.search.TopDocs;
import org.apache.lucene.store.Directory;
import org.apache.lucene.store.FSDirectory;
import org.apache.lucene.util.Version;

public class luceneSearch {
    public ArrayList<luceneSearchResult> search(String Field, String Que)throws Exception {
        String IndexDir = "/home/crazy/eclipse-J2EE-workspace/SearchEnginZJT/index/";
        Directory dir = FSDirectory.open(new File(IndexDir));
        IndexReader ir = DirectoryReader.open(dir);
        IndexSearcher is = new IndexSearcher(ir);
        Analyzer anal = new StandardAnalyzer(Version.LUCENE_44);
        QueryParser qp = new QueryParser(Version.LUCENE_44, Field, anal);
        Query q = qp.parse(Que);
        //DuplicateFilter df = new DuplicateFilter("subject");
```

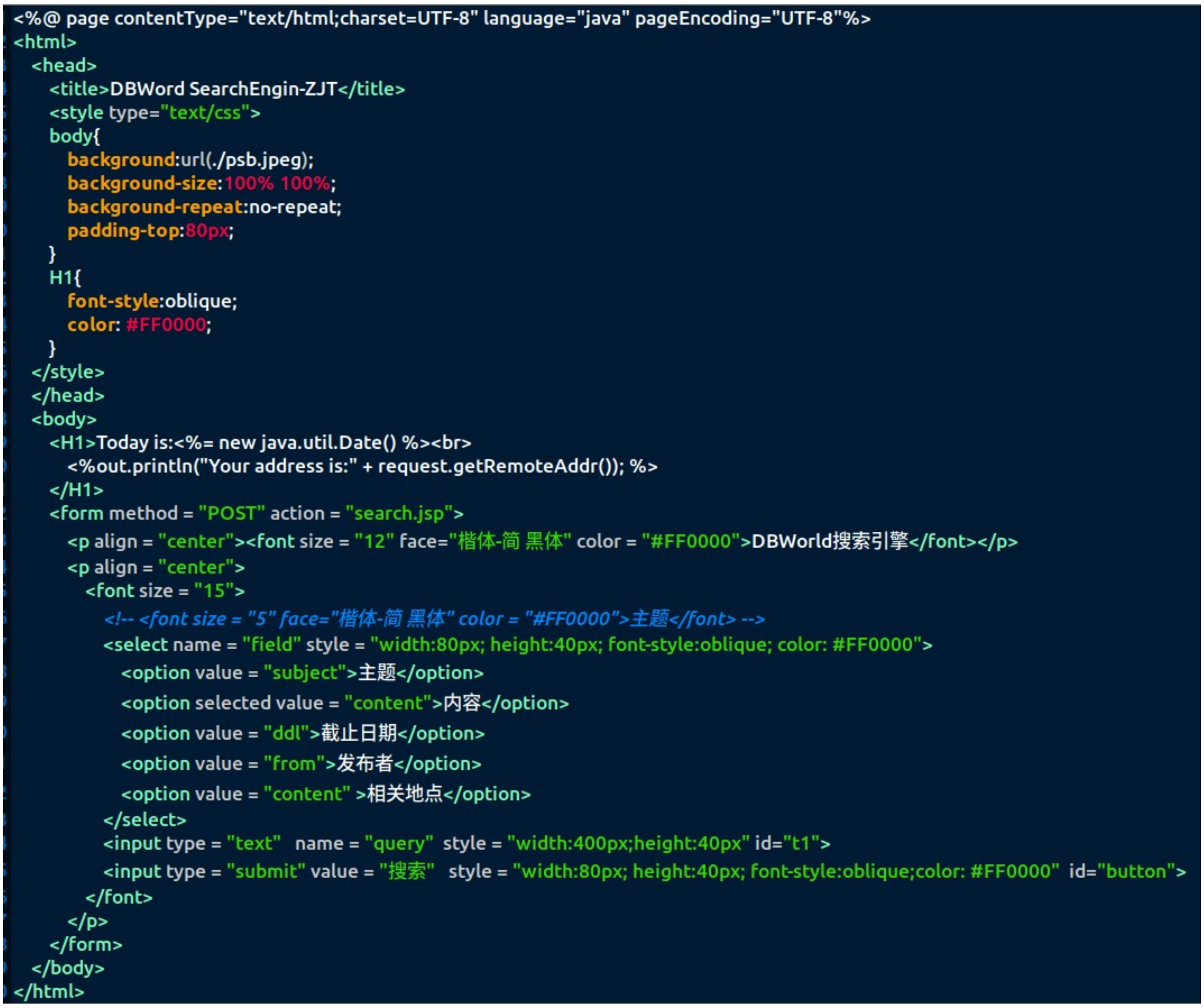


```java
        long StartTime = System.currentTimeMillis();
        //TopDocs docs = is.sea(q, df, 50);
        TopDocs docs = is.search(q, 500);
        long EndTime = System.currentTimeMillis();
        System.out.println("用时： " + (EndTime - StartTime) + "ms");;
        System.out.println("查询到：" + docs.totalHits + "条记录");;

        ArrayList<luceneSearchResult> results = new ArrayList<luceneSearchResult>();
        for(ScoreDoc sd:docs.scoreDocs) {
            Document resultdoc = is.doc(sd.doc);
            luceneSearchResult result = new luceneSearchResult(resultdoc);
            boolean nodup = true;
            for(luceneSearchResult pre:results) {
                if(pre.subject.equals(result.subject)) {
                    nodup = false;
                }
            }
            if(nodup) {
                results.add(result);
            }
        }
        ir.close();
        for(luceneSearchResult r:results) {
            System.out.println(r.subject);
        }
        return results;
    }
    Run | Debug
    public static void main(String[] args) {
        String Que = "International Conference on Internet Technologies & Society";
        try {
            luceneSearch s = new luceneSearch();
            s.search("subject",Que);
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
        }
    }
}
```

四、 简单的搜索界面

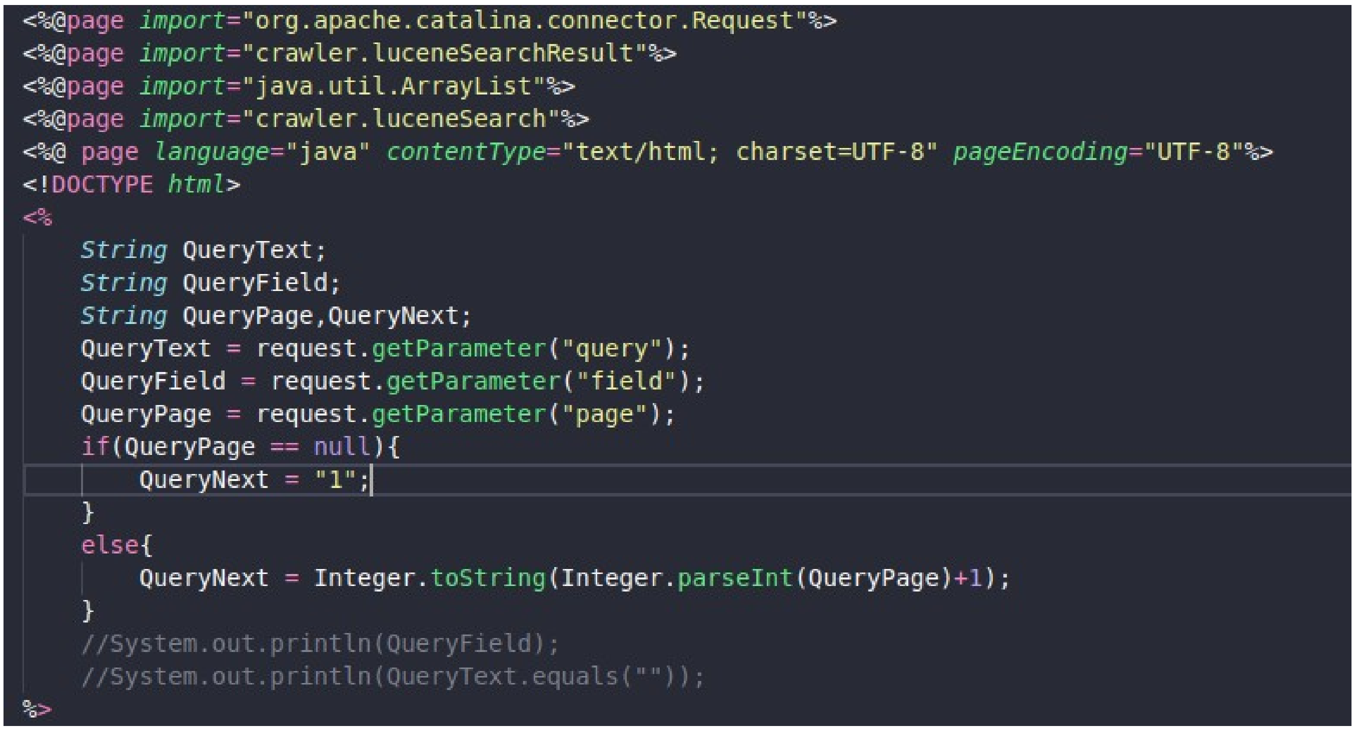
搜索页面由两部分组成：首先由 index.jsp 获取查询域和查询文本，然后在 search.jsp 中创建 luceneSearch 对象进行查询并对查询结果进行分页管理:

index.jsp:

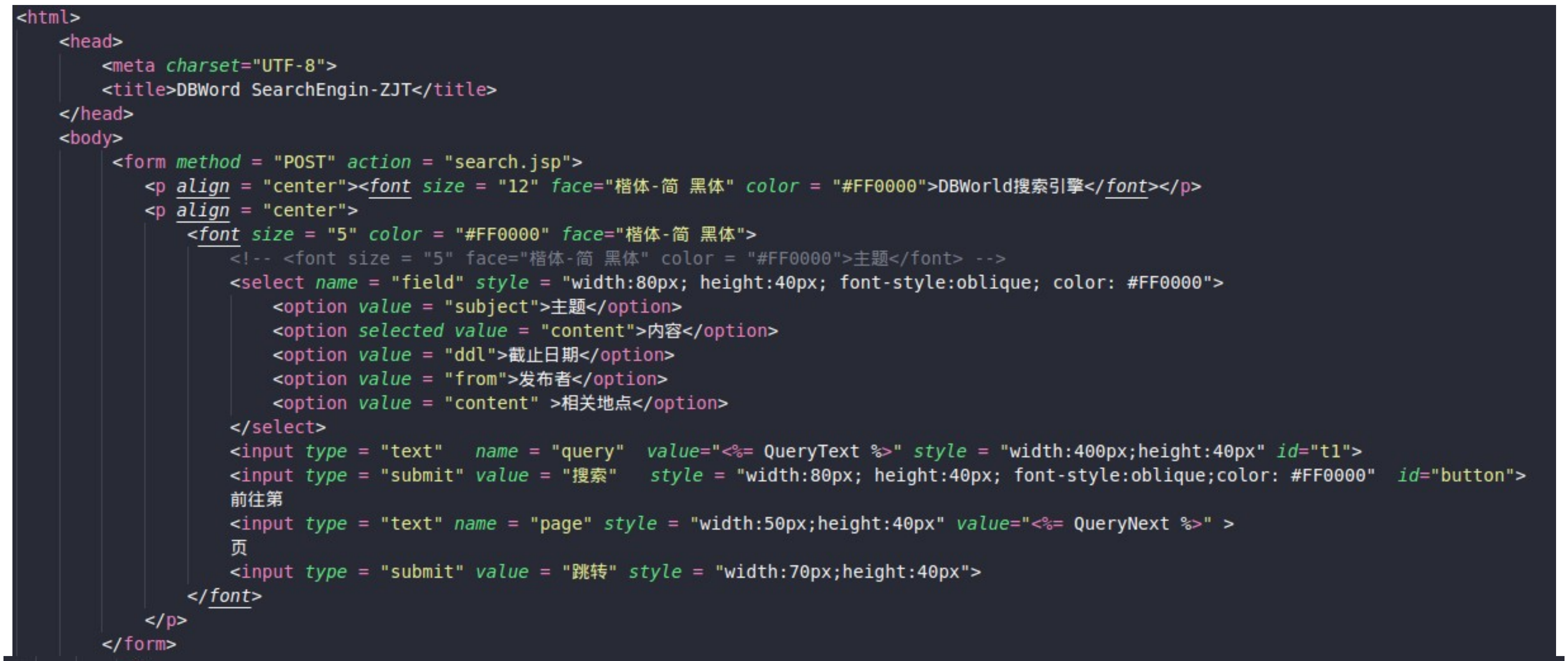


```
<%@ page contentType="text/html;charset=UTF-8" language="java" pageEncoding="UTF-8"%>
<html>
  <head>
    <title>DBWord SearchEngin-ZJT</title>
    <style type="text/css">
    body{
      background:url(./psb.jpeg);
      background-size:100% 100%;
      background-repeat:no-repeat;
      padding-top:80px;
    }
    H1{
      font-style:oblique;
      color: #FF0000;
    }
  </style>
  </head>
  <body>
    <H1>Today is:<%= new java.util.Date() %><br>
      <%out.println("Your address is:" + request.getRemoteAddr()); %>
    </H1>
    <form method = "POST" action = "search.jsp">
      <p align = "center"><font size = "12" face="楷体-简 黑体" color = "#FF0000">DBWorld搜索引擎</font></p>
      <p align = "center">
        <font size = "15">
          <!-- <font size = "5" face="楷体-简 黑体" color = "#FF0000">主题</font> -->
          <select name = "field" style = "width:80px; height:40px; font-style:oblique; color: #FF0000">
            <option value = "subject">主题</option>
            <option selected value = "content">内容</option>
            <option value = "ddl">截止日期</option>
            <option value = "from">发布者</option>
            <option value = "content" >相关地点</option>
          </select>
          <input type = "text"   name = "query"  style = "width:400px;height:40px" id="t1">
          <input type = "submit" value = "搜索"   style = "width:80px; height:40px; font-style:oblique;color: #FF0000"  id="button">
        </font>
      </p>
    </form>
  </body>
</html>
```

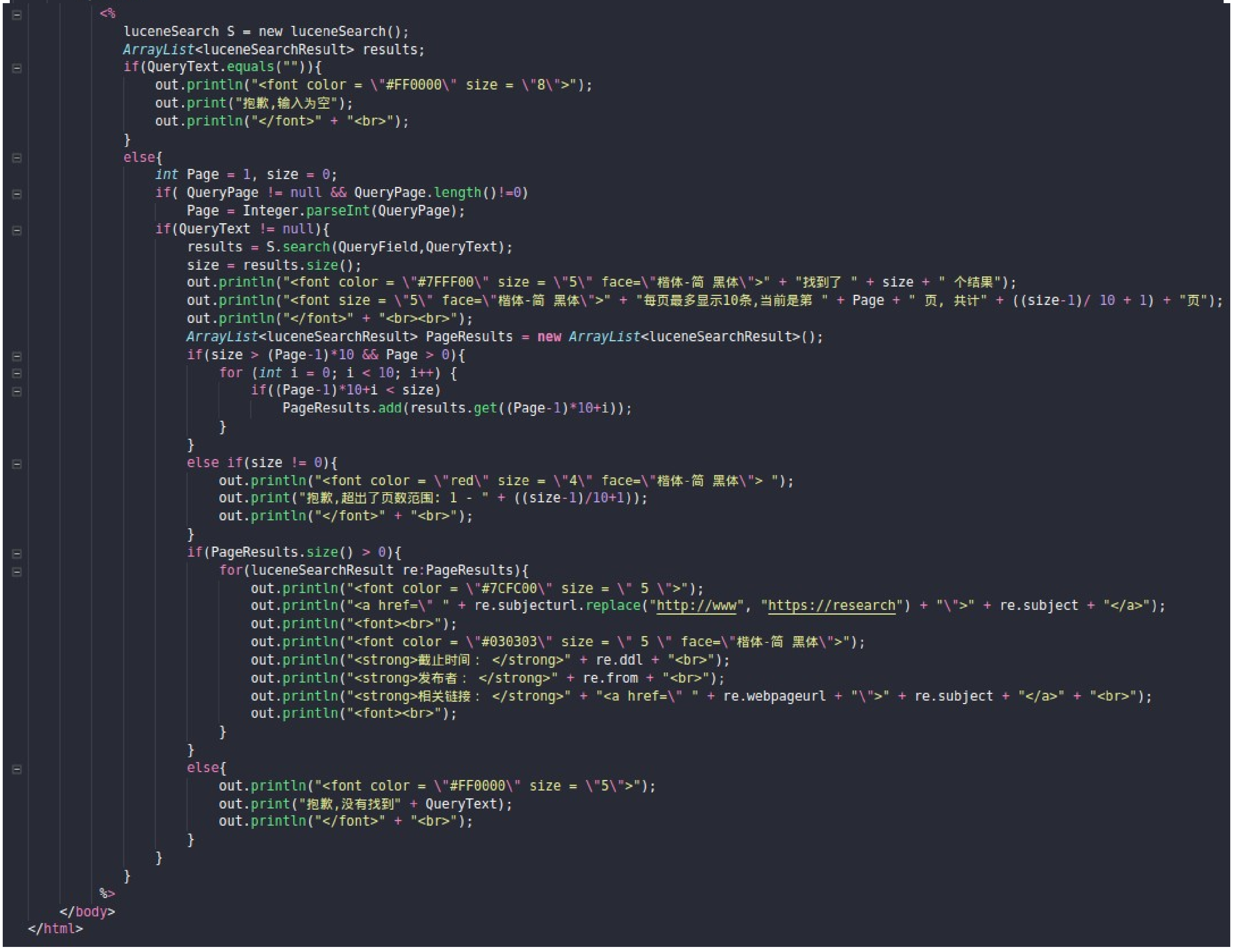
search.jsp:



```
<%@page import="org.apache.catalina.connector.Request"%>
<%@page import="crawler.luceneSearchResult"%>
<%@page import="java.util.ArrayList"%>
<%@page import="crawler.luceneSearch"%>
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<%
    String QueryText;
    String QueryField;
    String QueryPage,QueryNext;
    QueryText = request.getParameter("query");
    QueryField = request.getParameter("field");
    QueryPage = request.getParameter("page");
    if(QueryPage == null){
        QueryNext = "1";
    }
    else{
        QueryNext = Integer.toString(Integer.parseInt(QueryPage)+1);
    }
    //System.out.println(QueryField);
    //System.out.println(QueryText.equals(""));
%>
```



```
<html>
    <head>
        <meta charset="UTF-8">
        <title>DBWord SearchEngin-ZJT</title>
    </head>
    <body>
         <form method = "POST" action = "search.jsp">
            <p align = "center"><font size = "12" face="楷体-简 黑体" color = "#FF0000">DBWorld搜索引擎</font></p>
            <p align = "center">
                <font size = "5" color = "#FF0000" face="楷体-简 黑体">
                    <!-- <font size = "5" face="楷体-简 黑体" color = "#FF0000">主题</font> -->
                    <select name = "field" style = "width:80px; height:40px; font-style:oblique; color: #FF0000">
                        <option value = "subject">主题</option>
                        <option selected value = "content">内容</option>
                        <option value = "ddl">截止日期</option>
                        <option value = "from">发布者</option>
                        <option value = "content" >相关地点</option>
                    </select>
                    <input type = "text"   name = "query"  value="<%= QueryText %>" style = "width:400px;height:40px" id="t1">
                    <input type = "submit" value = "搜索"   style = "width:80px; height:40px; font-style:oblique;color: #FF0000"  id="button">
                    前往第
                    <input type = "text" name = "page" style = "width:50px;height:40px" value="<%= QueryNext %>" >
                    页
                    <input type = "submit" value = "跳转" style = "width:70px;height:40px">
                </font>
            </p>
        </form>
```

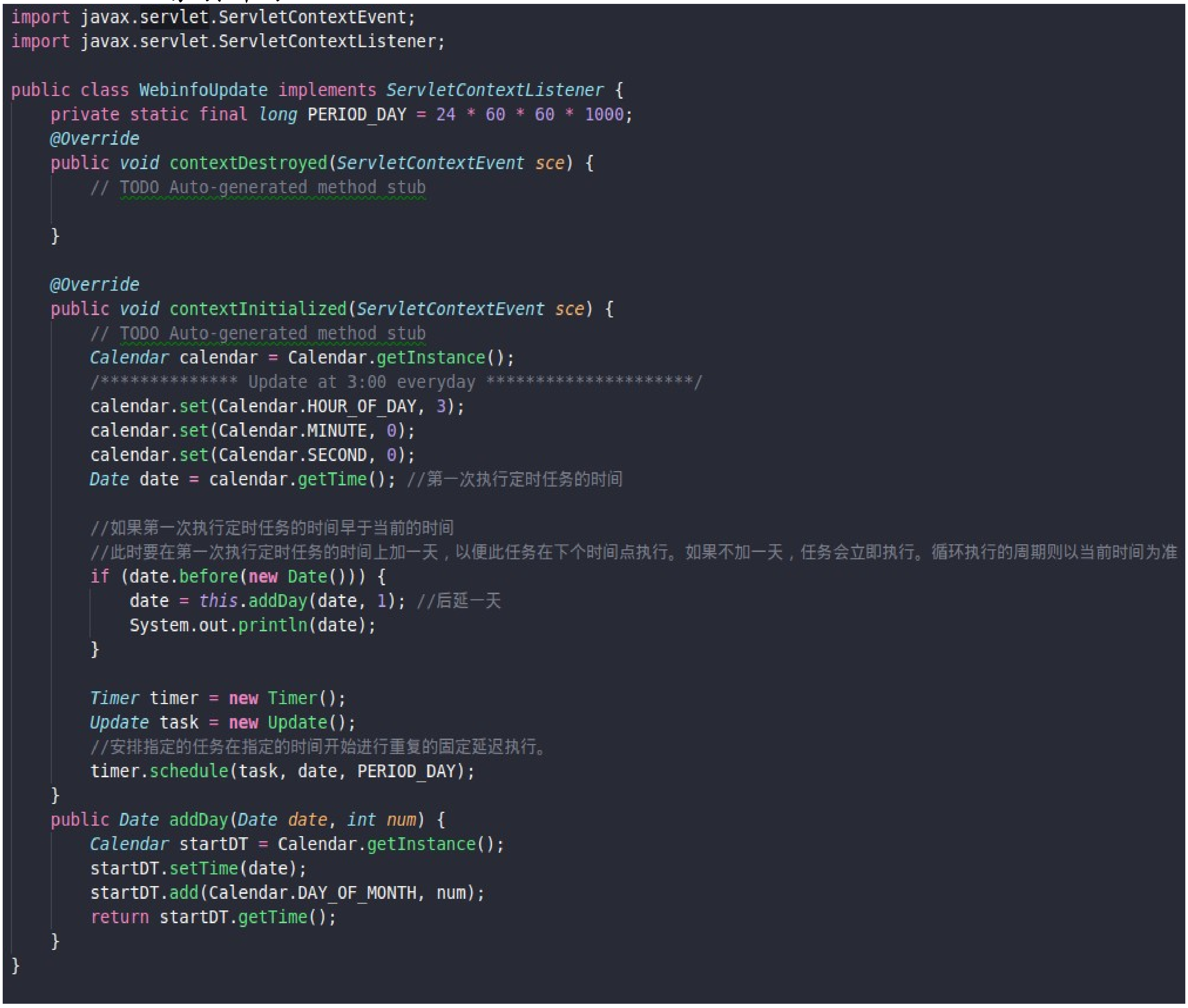


```
            <%
                luceneSearch S = new luceneSearch();
                ArrayList<luceneSearchResult> results;
                if(QueryText.equals("")){
                    out.println("<font color = \"#FF0000\" size = \"8\">");
                    out.print("抱歉,输入为空");
                    out.println("</font>" + "<br>");
                }
                else{
                    int Page = 1, size = 0;
                    if( QueryPage != null && QueryPage.length()!=0)
                        Page = Integer.parseInt(QueryPage);
                    if(QueryText != null){
                        results = S.search(QueryField,QueryText);
                        size = results.size();
                        out.println("<font color = \"#7FFF00\" size = \"5\" face=\"楷体-简 黑体\">" + "找到了 " + size + " 个结果");
                        out.println("<font size = \"5\" face=\"楷体-简 黑体\">" + "每页最多显示10条,当前是第 " + Page + " 页, 共计" + ((size-1)/ 10 + 1) + "页");
                        out.println("</font>" + "<br><br>");
                        ArrayList<luceneSearchResult> PageResults = new ArrayList<luceneSearchResult>();
                        if(size > (Page-1)*10 && Page > 0){
                            for (int i = 0; i < 10; i++) {
                                if((Page-1)*10+i < size)
                                    PageResults.add(results.get((Page-1)*10+i));
                            }
                        }
                        else if(size != 0){
                            out.println("<font color = \"red\" size = \"4\" face=\"楷体-简 黑体\"> ");
                            out.print("抱歉,超出了页数范围: 1 - " + ((size-1)/10+1));
                            out.println("</font>" + "<br>");
                        }
                        if(PageResults.size() > 0){
                            for(luceneSearchResult re:PageResults){
                                out.println("<font color = \"#7CFC00\" size = \" 5 \">");
                                out.println("<a href=\" " + re.subjecturl.replace("http://www", "https://research") + "\">" + re.subject + "</a>");
                                out.println("<font><br>");
                                out.println("<font color = \"#030303\" size = \" 5 \" face=\"楷体-简 黑体\">");
                                out.println("<strong>截止时间： </strong>" + re.ddl + "<br>");
                                out.println("<strong>发布者： </strong>" + re.from + "<br>");
                                out.println("<strong>相关链接： </strong>" + "<a href=\" " + re.webpageurl + "\">" + re.subject + "</a>" + "<br>");
                                out.println("<font><br>");
                            }
                        }
                        else{
                            out.println("<font color = \"#FF0000\" size = \"5\">");
                            out.print("抱歉,没有找到" + QueryText);
                            out.println("</font>" + "<br>");
                        }
                    }
                }
            %>
    </body>
</html>
```

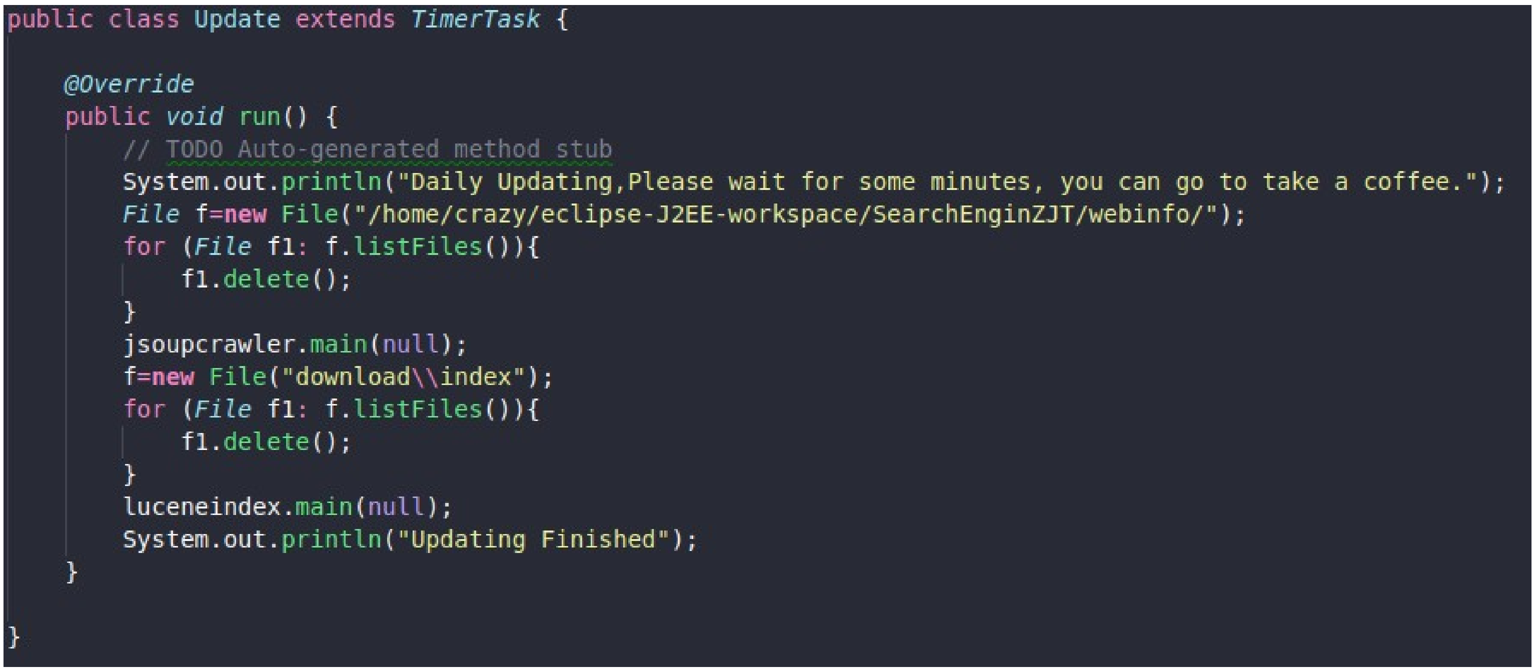
五、 定时更新

tomcat 作为一个 Servlet，在 Servlet API 中有一个 ServletContextListener 接口，它能够监听 ServletContext 对象的生命周期，实际上就是监听 Web 应用的生命周期。

当 Servlet 容器启动或终止 Web 应用时，会触发 ServletContextEvent 事件，该事件由 ServletContextListener 来处理。 我们只要实现 ServletContextListener 接口，产生计时线程，在每天的固定时间执行定时任务，重新爬取数据、构建索引即可：



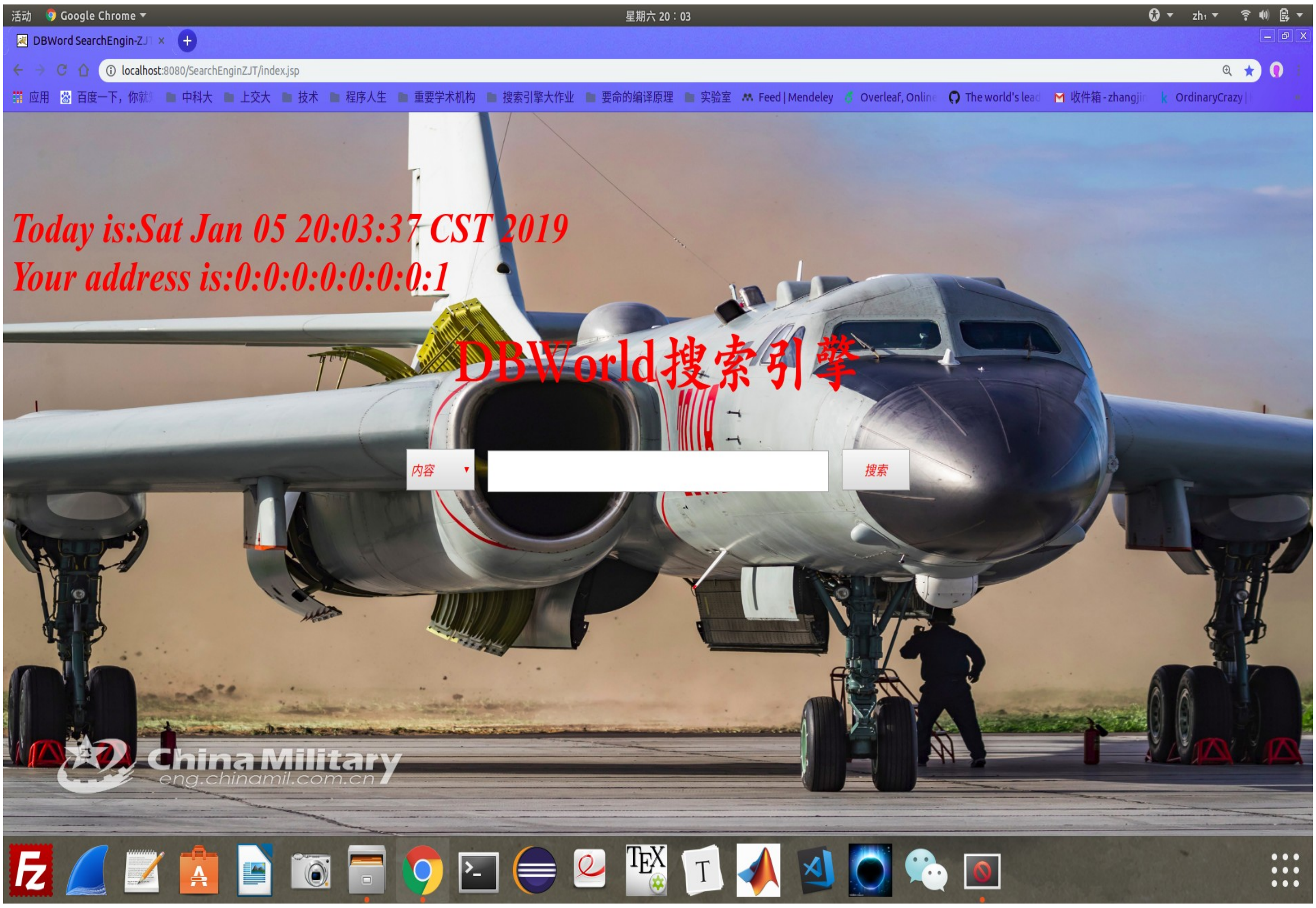
```
import javax.servlet.ServletContextEvent;
import javax.servlet.ServletContextListener;

public class WebinfoUpdate implements ServletContextListener {
    private static final long PERIOD_DAY = 24 * 60 * 60 * 1000;
    @Override
    public void contextDestroyed(ServletContextEvent sce) {
        // TODO Auto-generated method stub

    }

    @Override
    public void contextInitialized(ServletContextEvent sce) {
        // TODO Auto-generated method stub
        Calendar calendar = Calendar.getInstance();
        /************** Update at 3:00 everyday *********************/
        calendar.set(Calendar.HOUR_OF_DAY, 3);
        calendar.set(Calendar.MINUTE, 0);
        calendar.set(Calendar.SECOND, 0);
        Date date = calendar.getTime(); //第一次执行定时任务的时间

        //如果第一次执行定时任务的时间早于当前的时间
        //此时要在第一次执行定时任务的时间上加一天，以便此任务在下个时间点执行。如果不加一天，任务会立即执行。循环执行的周期则以当前时间为准
        if (date.before(new Date())) {
            date = this.addDay(date, 1); //后延一天
            System.out.println(date);
        }

        Timer timer = new Timer();
        Update task = new Update();
        //安排指定的任务在指定的时间开始进行重复的固定延迟执行。
        timer.schedule(task, date, PERIOD_DAY);
    }
    public Date addDay(Date date, int num) {
        Calendar startDT = Calendar.getInstance();
        startDT.setTime(date);
        startDT.add(Calendar.DAY_OF_MONTH, num);
        return startDT.getTime();
    }
}
```



```java
public class Update extends TimerTask {

    @Override
    public void run() {
        // TODO Auto-generated method stub
        System.out.println("Daily Updating,Please wait for some minutes, you can go to take a coffee.");
        File f=new File("/home/crazy/eclipse-J2EE-workspace/SearchEnginZJT/webinfo/");
        for (File f1: f.listFiles()){
            f1.delete();
        }
        jsoupcrawler.main(null);
        f=new File("download\\index");
        for (File f1: f.listFiles()){
            f1.delete();
        }
        luceneindex.main(null);
        System.out.println("Updating Finished");
    }

}
```

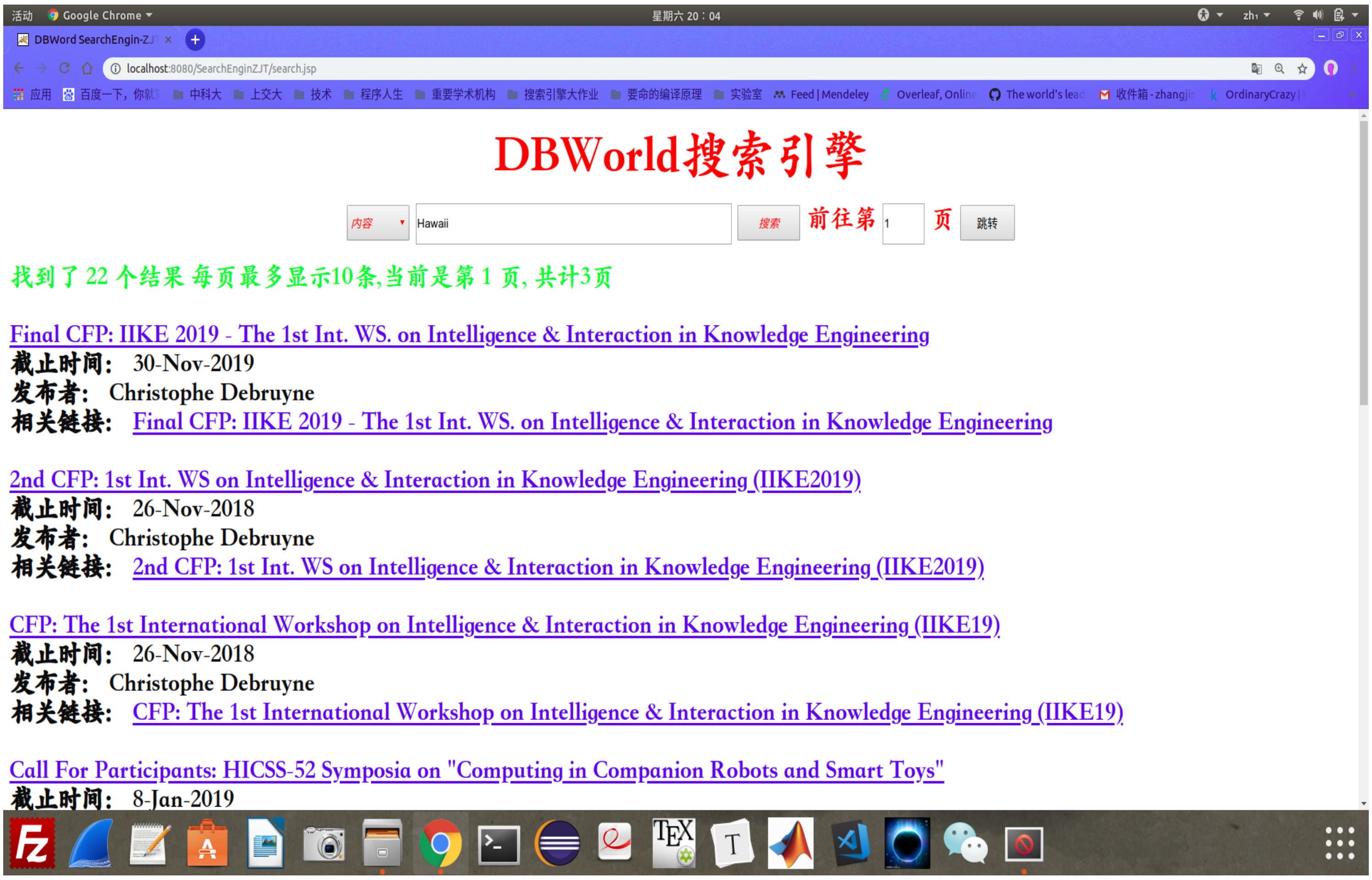
【实验结果说明及演示】

1.　界面实现:



Today is:Sat Jan 05 20:03:37 CST 2019
Your address is:0:0:0:0:0:0:0:1
DBWorld搜索引擎

2. 搜索测试:


活动
Google Chrome
星期六 20：04
DBWord SearchEngin-ZJT
localhost:8080/SearchEnginZJT/search.jsp
应用　百度一下，你就　中科大　上交大　技术　程序人生　重要学术机构　搜索引擎大作业　要命的编译原理　实验室　Feed | Mendeley　Overleaf, Online　The world's lead　收件箱 - zhangjin　OrdinaryCrazy |
内容
Hawaii
搜索
前往第
1
页
跳转

DBWorld搜索引擎

找到了 22 个结果 每页最多显示10条,当前是第 1 页, 共计3页

Final CFP: IIKE 2019 - The 1st Int. WS. on Intelligence & Interaction in Knowledge Engineering
截止时间: 30-Nov-2019
发布者: Christophe Debruyne
相关链接: Final CFP: IIKE 2019 - The 1st Int. WS. on Intelligence & Interaction in Knowledge Engineering

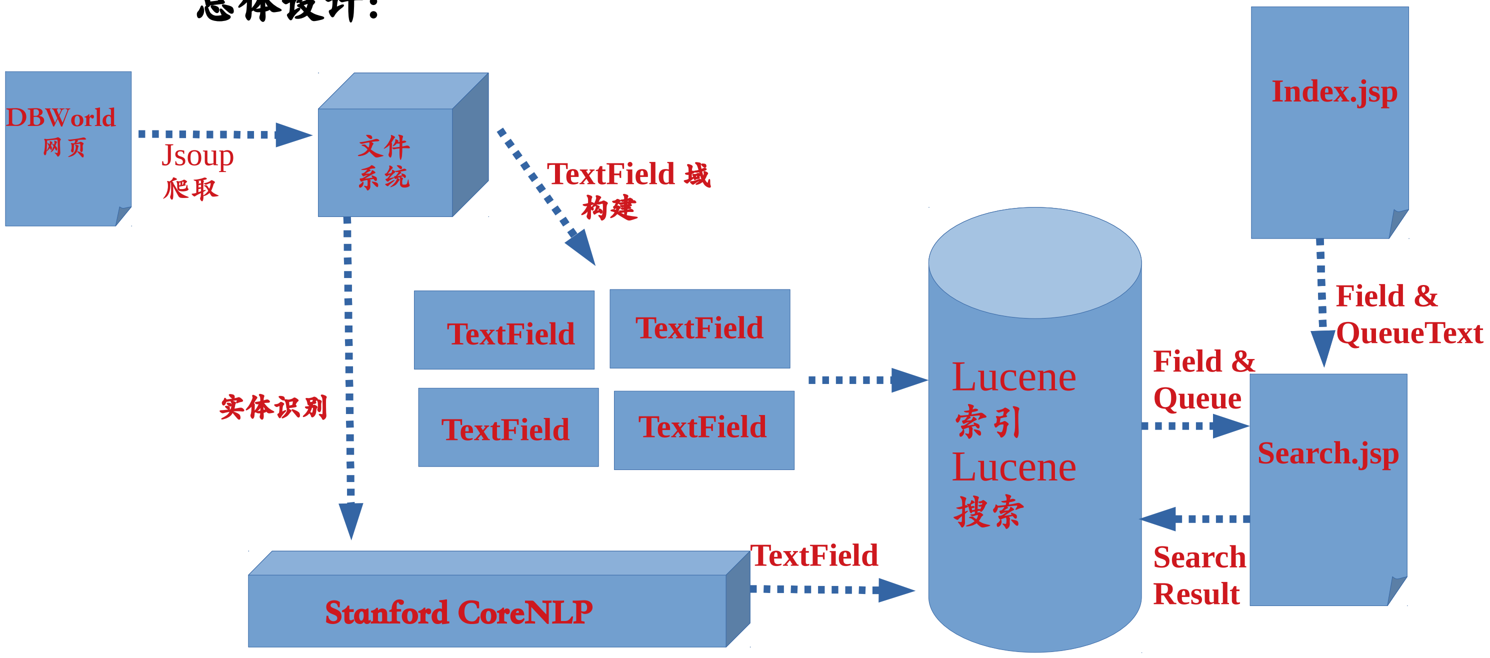
2nd CFP: 1st Int. WS on Intelligence & Interaction in Knowledge Engineering (IIKE2019)
截止时间: 26-Nov-2018
发布者: Christophe Debruyne
相关链接: 2nd CFP: 1st Int. WS on Intelligence & Interaction in Knowledge Engineering (IIKE2019)

CFP: The 1st International Workshop on Intelligence & Interaction in Knowledge Engineering (IIKE19)
截止时间: 26-Nov-2018
发布者: Christophe Debruyne
相关链接: CFP: The 1st International Workshop on Intelligence & Interaction in Knowledge Engineering (IIKE19)

Call For Participants: HICSS-52 Symposia on "Computing in Companion Robots and Smart Toys"
截止时间: 8-Jan-2019

【实验总结】

总体设计:



1. **亮点：**

· 基于 Stanford CoreNLP 的地点实体识别

· 在爬虫部分，自己完成，没有使用开源工具，在实验过程中，发现从起始网页中提取的 URL 指向的都是 302 重定向网页。为了加快爬取效率，不采用爬取重定向网页，再爬取原网页的方法，而是直接根据两者之间的转换规则，把重定向网页的 URL 直接转换为原网页的 URL。根据观察，两者间的区别在于前缀的网址不同，如 302 网页是 http://www.cs.wisc.edu/dbworld/messages/2018-12/1544218985.html，而实际是 https://research.cs.wisc.edu/dbworld/messages/2018-12/1544218985.html。显然，只要简单修改前缀即可。因此对于 DBWorld 中的 302 网页，直接按规则转换为正常网页，加快了爬取速度。

· 提供多种搜多选项和搜索结果分页

· 实现定时更新功能

· 对于 lucene 多次返回相同文件进行管理和去重

2. **不足和需要改进的地方：**

nlp 实现过于简单，对于时间和主题实体识别都没有实现；搜索结果基本依靠 lucene 接口，没有自己实现排序输出；搜索界面和定时更新实现过于粗糙暴力。

3. 收获与建议：

通过这次实验，学习了很多知识技能，提高了自己的工程实践能力，学习了 Tomcat 部署和后台 Java 运行，简单的 Jsp 和 HTML 脚本，学习了 lucene 和 Stanford CoreNLP 等工具的使用，收获很大，但时间安排上有些仓促，希望之后能够分阶段布置任务，由浅入深，逐步实现，提高实验对于学生的锻炼和学习效果。