《电影天堂》站内检索
设计文档

——2014年秋Java课程作业

小组成员：赵艺轩　卢晓航　戴丛蔚
罗　尖　罗　晶　浦隽瑾

指导教师：李广建

2015年1月

目　　录

《电影天堂》站内检索设计文档

1. 引言

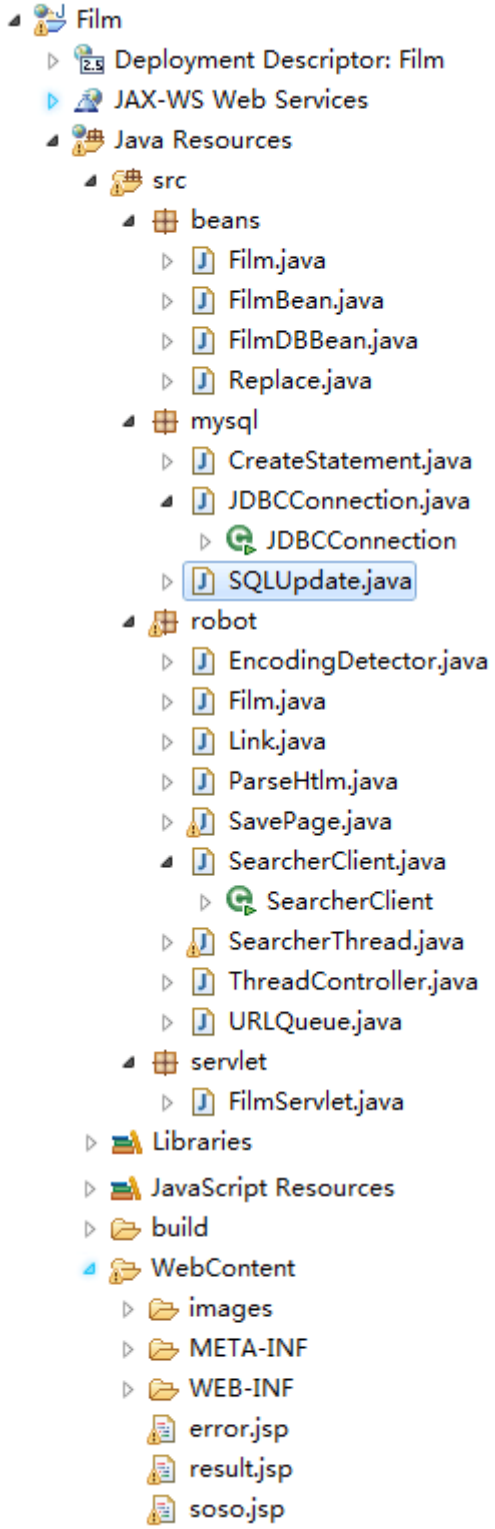
本设计文档编写的目的是说明本项目（即网站《电影天堂》站内检索）各模块的设计考虑，包括程序描述、输入/输出、算法和流程逻辑等，为系统、项目维护提供基础。

下面，本设计文档将主要介绍模块设计、接口设计、出错处理及未来的优化方向等方面。

2. 背景介绍

《电影天堂》是国内较大的电影在线观看和下载平台。主要有迅雷下载和快车下载和电驴以及手机视频格式下载。数据已经达到超出 100 多万部资源，千万小时时长。还能提供在线观看和下载最新的国内外电影和连续剧的服务。电影天堂提供其他剧情电影，大多都是都是高清电影，保持每天海量数据的更新。

3. 模块设计


Film
Deployment Descriptor: Film
JAX-WS Web Services
Java Resources
src
beans
Film.java
FilmBean.java
FilmDBBean.java
Replace.java
mysql
CreateStatement.java
JDBCConnection.java
JDBCConnection
SQLUpdate.java
robot
EncodingDetector.java
Film.java
Link.java
ParseHtlm.java
SavePage.java
SearcherClient.java
SearcherClient
SearcherThread.java
ThreadController.java
URLQueue.java
servlet
FilmServlet.java
Libraries
JavaScript Resources
build
WebContent
images
META-INF
WEB-INF
error.jsp
result.jsp
soso.jsp

3.1 Mysql 包

mysql 包包含三个.java 文件：JDBCConnection.java 、CreateStatement.java 和 SQLUpdate.java。JDBCConnection.java 用于建立与 MySQL 数据库的连接，包括两个方法：public JDBCConnection()和 public void close()，前者用于建立数据库连接，后者用于关闭数据库连接；CreateStatement.java 用于创建 sql 语句对象，包括两个方法：public CreateStatement(JDBCConnection dc)和 public void close()，前者用于创建语句对象，后者用于关闭语句对象；SQLUpdate.java 用于发送 SQL 语句，主要包括两个方法：public void createTable()和 public void insertData(Film film)，前者用于创建数据表，后者用于插入数据。

3.2 Beans 包

beans 包包含三个.java 文件：Film.java、FilmBean.java 和 FilmDBBean.java。Film.java 定义一个电影的构成：name 是片名，transname 是译名，role 是主演，introduction 是简介，site 是下载地址，director 是导演，time 是上映时间，score 是评分，country 是国家，type 是类型，picture 是图片，language 是语言，minute 是片长，href 是电影天堂每个电影的链接，Film.java 还包括了这些项的值的存取。FilmBean.java 作为电影的队列，主要起添加电影到队列以及清空队列的作用，同时，为对搜索结果进行展示，定义了页码、页数以及数量的设置。FilmDBBean.java 主要用于处理用户的检索，处理检索的方法是 public FilmDBBean(int judge, String name, String director, String role, String score, String time, String type)，此外还有显示特定页的搜索结果的方法 public FilmBean getResult(String toPage)。

3.3 Servlet 包

servlet 包用于负责对 jsp 的调度，只有一个.java 文件：FilmServlet.java。先判断是哪一种搜索，并调用 FilmDBBean.java 进行搜索，无搜索结果调用 error.jsp，有搜索结果调用 result.jsp。

3.4 Robot 包

robot 包是爬虫程序，包括爬虫与数据传入。分三个部分来介绍：

一是爬虫的基本框架。

在构建一个爬虫时，最为关键的部分莫过于爬虫本身实现功能的部分。为了提高爬虫爬行的速率，增加了多线程，但是爬虫最核心的框架还是爬虫的本身。

在这里，爬虫的本身分为几个部分。首先是检测爬来的网页的编码形式，只有这样你才能对网页进行解析，紧接着对于网页根据特定的编码进行解析。这部分的解析有两个重要的工作，第一，将里面的链接拿出来保存在队列中，以便进行再次爬行；第二，将里面有用的信息拿出来存在数据库里。那么对于爬行出来的链接要进行管理，保存，为后面继续爬做铺垫。这里面的方法就有深度优先，广度优先等等搜索方法，对于不同的方法，队列的数据结构选取不同。最后，为了方便的进行处理，建立一些特定的数据结构。

找到编码----------解析网页（保存内容到数据库）-----------找出链接------（深度/广度）---------保存链接（出去相同）-----------接续解析下一个

当然了，这里使用的是单线程，为了提高检索速度，建立多线程，每一个线程实现的功

能都如上图所示。但需要对这些线程进行管理。

主函数----------线程管理器 ---------各个线程

二是系统调用的具体实现。

1） 数据结构

Film 数据结构保存的所有的重要的数据项。有 14 项信息，与数据库的项数一致。记录了所需要的所有信息。

```
public class Film {
    //电影元素
    public String chineseName;//中文名
    public String Name;//英文名
    public String onTime;//上映时间
    public String country;//国家
    public String classify;//分类
    public String language;//语言
    public String score;//评分
    public String filmTime;//电影时长
    public String director;//导演
    public String leadingRole;//主演
    public String intoduction;//简介
    public String site;//地址
    public String picture;//图片
    public String href;//自己
}
```

Link 数据结构记录链接的层数和 url。

```
private URL pageUrl; //待爬行的 URL
private int depth;// URL 的层数
```

urlQueue 结构记录了队列的结构，与特定的查找方式有关。其中记录了层数以及对于所得到的 url 进行查重。另外，可以灵活设置层数和网页抓取限制。

```
public class URLQueue {      //10 行
    //待爬行链接队列
    private LinkedList<Link> toVisitURL;
    //已获得 URL 队列,用于查重
    private Set<URL> duplicateCheckingURL;
    //最大搜索层数限制   //15 行
    private int maxLevel;
    //最大抓取网页数限制
    private int maxPageNum;
    //构造方法，缺省不限制爬行网页数量和最大爬行层数
    public URLQueue() {    //20 行
        toVisitURL=new LinkedList<Link>();
        duplicateCheckingURL= new HashSet<URL>();
        maxLevel =-1;//缺省值，不限制最大搜索层数限制
        maxPageNum = -1;//缺省值，不限制抓取页面数
    } //25 行
```

```
    //设置最大抓取网页数
    public void setMaxElements(int maxPageNum) {
        this.maxPageNum= maxPageNum;
    }
    //设置最大访问层数   //30 行
    public void setMaxLevel(int maxLevel) {
        this.maxLevel = maxLevel;
    }
    //返回最大访问层数
    public int getMaxLevel() {    //35 行
        return maxLevel;
    }
    //返回爬行队列中的元素个数
    public int getQueueSize() {
        return toVisitURL.size();//40 行
    }
    //入队的方法
    public boolean push(Link link) {
        //获得当前链接的 URL
        URL url = link.getURL(); //45 行
        //获得当前链接的层数
        int level = link.getDepth();
        // 实现最大抓取数量控制
        if (maxPageNum != -1 && maxPageNum<= duplicateCheckingURL.size())
            return false; //50 行
        // 实现最大访问层数控制
        if (maxLevel != -1 && level > maxLevel)
            return false;
        // 实现重复访问控制，如果不重复追加到爬行队列的末尾
        if (duplicateCheckingURL.add(url)) {//55 行
            toVisitURL.addLast(link);//追加到爬行队列的末尾
            return true;
        }else return false;
    }
```

另外，不同的方法对队列的操作也不同。（实际上一个是栈，一个是队列）

```
public Link popBFS() {
        //出队操作，移除并返回待爬行队列的第一个元素
        Link link = toVisitURL.removeFirst();
        return link;
    }//65 行
    //深度优先出队方法
    public Link popDFS() {
        //出队操作，移除并返回待爬行队列的最后一个元素
        Link link = toVisitURL.removeLast();
```

```
            return link;//70
        }
```

ThreadController 数据结构，记录现有线程数以及最大线程数。

```
private int maxThreads;//最大线程数

    private URLQueue queue;//URL 队列

    private Class<SearcherThread> threadClass; //网页处理线程类

    private int nThreads;//当前运行的线程数量
```

2） 处理程序。

robot 处理程序包主要有 9 个类。

SearcherClient 是函数入口，负责初始化参数，连接数据库，开启线程管理器，调用 ThreadController 类。

ThreadController 类是用来管理各个线程的，线程不够时候添加。使线程维持在一定的水平，调用多个 SearcherThread 类。

SearcherThread 类，线程主类，调用各个线程实现函数来实现功能。

EncodingDetector 类用编码解析器解析编码。

ParserHtlm 类用来解析网页，提取信息。

SavePage 类用来保存网页。

剩下的 3 个类 Link，urlQUEUE 和 Film 记录数据结构。

另外需要说的是 ParserHtlm 解析类，用来解析网页，这里面主要是解析有用内容，用的方法是正则表达式。例如(主.*?演).(.*)(.)(简.*?介)，从中找到匹配。比较难的是匹配长段话的时候.的属性要改为 Pattern.DOTALL，因为.默认不会匹配回车的。

3.5 Jsp

主要通过三个 jsp 开发用户界面，分别是 soso.jsp，result.jsp，error.jsp。

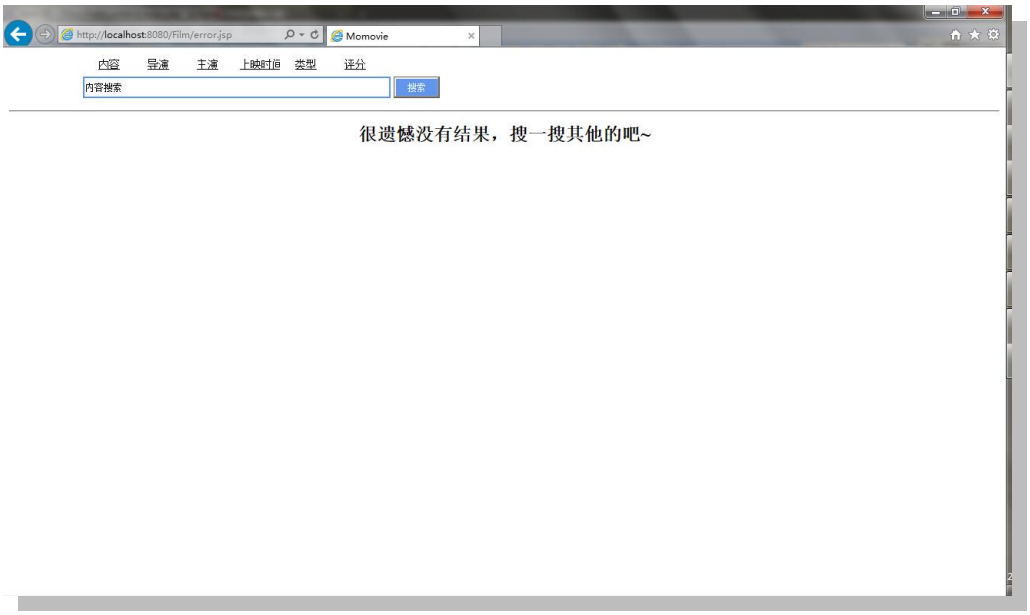
soso.jsp 是我们的主页，分为六种检索，检索后调用 servlet 转向 result 界面（如果结果非空的话）。我们的布局用的是 div 自动居中布局，结果展示通过表格完成。

soso. jsp

result.jsp

4. 出错处理

为了在系统出现异常情况下给用户以明确的提示，可采用下述方式提示：



5. 未来的优化方向及相关体会

首先，在做这种大型程序的时候，接口并不是一个容易处理的问题，尤其是分工合作时需要先有计划再开始具体写代码。

其次，在网页爬虫部分，解析网页的时候，编码后网页有很多乱码现象，这是非常困难的，而且网页格式也有很多区别，导致了许多问题。此处，还存在大量重复写解析句的问题，可以考虑是否可以用泛型或者数据库中预编译的方法。

另外，我们的检索不够“高级”，目前还无法实现带布尔逻辑的检索，只能进行全文或某些部分的检索。

第四，本项目的高亮实现还比较简省，目前只是简单地在前端进行格式上的变化，更好的方式应该是用索引向量实现。

第五，由于缺乏 html 编码的经验，网页的页面设计较为简陋，如果能熟练使用 css、js 等工具，相信本项目会有很好的发展前景。

本项目采用 javabean、jsp、servlet 三者结合的方式进行开发，业务逻辑清晰，因此小组成员分工详细，各展其长，是真切体会到了这一套体系的优越性。可惜才疏学浅，本项目还有种种需要改进之处，望指教。

2015 年 1 月