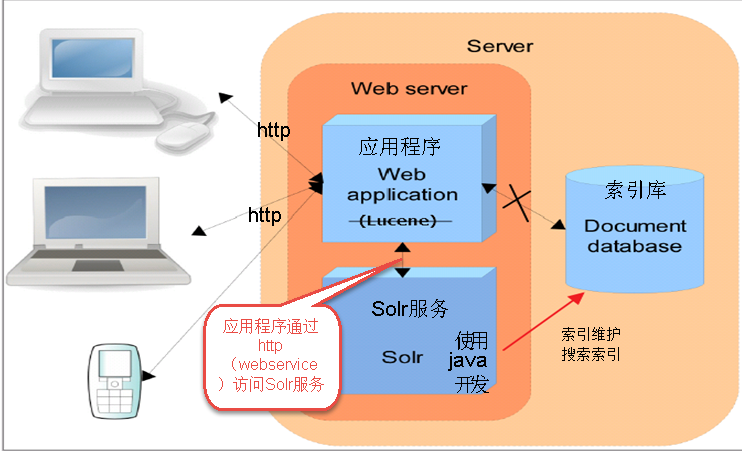
Solr

# Solr介绍

## Solr是什么

Solr 是Apache下的一个顶级开源项目，采用Java开发，它是基于Lucene的全文搜索服务。

solr可以实现全文检索功能（索引、搜索），solr是可以独立运行在tomcat等web容器中。



Solr不提供构建UI的功能，Solr提供了一个管理界面，通过管理界面可以查询Solr的配置和运行情况。

Solr主要对外提供索引和搜索服务。

## Solr和lucene的区别

Lucene是一个全文检索的工具包（是一堆的jar 包，不能独立运行，由应用程序通过lucene的jar实现全文检索）。

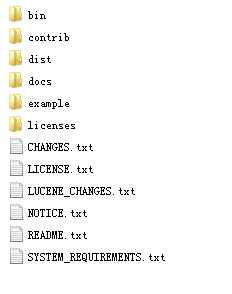
Solr 是一个全文检索的服务，solr是运行在tomcat等web容器中，对外提供http 的索引和搜索服务。

Solr提供了比Lucene更为丰富的查询语言，同时实现了可配置、可扩展，并对索引、搜索性能进行了优化。

## Solr下载

从Solr官方网站（http://lucene.apache.org/solr/ ）下载Solr4.10.3，根据Solr的运行环境，Linux下需要下载lucene-4.10.3.tgz，windows下需要下载lucene-4.10.3.zip。

Solr使用指南可参考：https://wiki.apache.org/solr/FrontPage。



bin：可执行程序、脚本

contrib：存放了一些扩展的包，用于索引和搜索

dist：其中有一个solr-4.10.3.war 将其部署到tomcat容器中，运行solr

docs：使用说明文档

example：存储了很多solr开发使用的例子工程及目录结构等。

* example/solr：

该目录是一个包含了默认配置信息的Solr的Core目录。

* example/multicore：

该目录包含了在Solr的multicore中设置的多个Core目录。

* example/webapps：

该目录中包括一个solr.war，该war可作为solr的运行实例工程。

licenses：solr相关的一些许可信息

licenses：许可

## Solr运行环境

Solr4.10.3在jdk1.7下运行

Solr：Solr4.10.3

Jdk：jdk1.7.0\_72

Tomcat：apache-tomcat-7.0.53

# Solr安装配置

## SolrCore的安装配置

### SolrHome和Solrcore

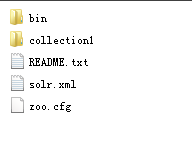
SolrHome是一个目录 ，是Solr运行的主目录 ，SolrHome目录 中包括多个SolrCore目录，每个solrcore目录 对应一个索引文件目录 。

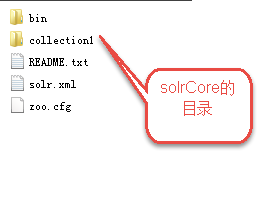
SolrCore即solr实例对外提供索引和搜索服务。

SolrCore是一个逻辑的概念，将一个solr工程（部署在tomcat中）逻辑区别开以solrcore为单位的服务。

### 目录结构

SolrHome目录 结构（solr-4.10.3\solr-4.10.3\example\solr例子）





SolrCore目录 结构

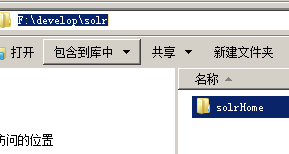


### 创建SolrCore

#### 创建目录

先创建SolrHome

F:\develop下创建solr目录 ，SolrHome目录放到F:\develop\solr下。



solrHOme目录如下：

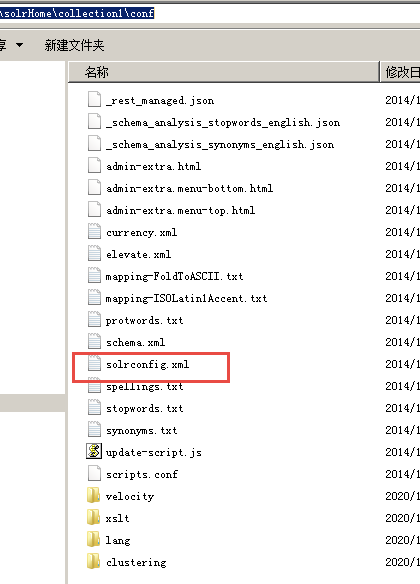


SolrCore目录 ：



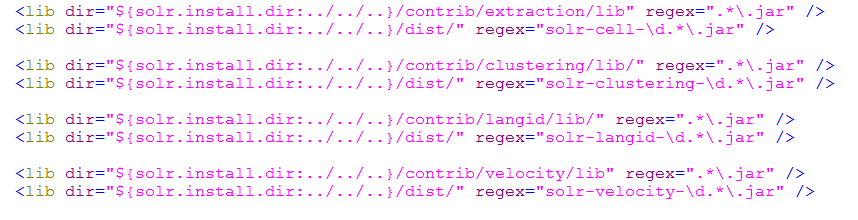
#### 配置文件

在solrCore的conf 下有一个solrconfig.xml文件，此文件就是solrCore的配置文件：



* 加载jar包

以下配置是solrcore运行所加载的jar包：

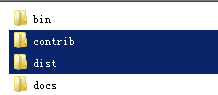


solr.install.dir 表示SolrCore所在目录。

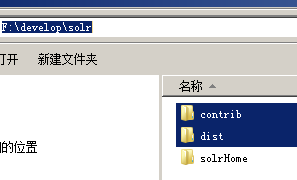
修改：



将solr下载包中的contrib、dist两个目录 拷贝到f:\develop\solr下

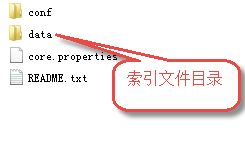


拷贝后结果：



* 更改索引文件目录

每个SolrCore都有自己的索引文件目录 ，默认在SolrCore目录下的data中。



data数据目录下包括了index索引目录 和tlog日志文件目录。

如果不想使用默认的目录也可以通过solrConfig.xml更改索引目录 ，如下：



* 配置请求处理器

客户端请求Solr请求处理器，进行索引和搜索。

在solrconfig.xml中配置了很多的请求处理器

requestHandler请求处理器，定义了索引和搜索的访问方式。

通过/update维护索引，可以完成索引的添加、修改、删除操作。



提交xml、json数据完成索引维护，索引维护小节详细介绍。

通过/select搜索索引。



设置搜索参数完成搜索，搜索参数也可以设置一些默认值，如下：

<requestHandler name="/select" class="solr.SearchHandler">

<!-- 设置默认的参数值，可以在请求地址中修改这些参数-->

<lst name="defaults">

<str name="echoParams">explicit</str>

<int name="rows">10</int><!--显示数量-->

<str name="wt">json</str><!--显示格式-->

<str name="df">text</str><!--默认搜索字段-->

</lst>

</requestHandler>

### Solr工程部署

将官方提供的Solr.war部署到tomcat下。

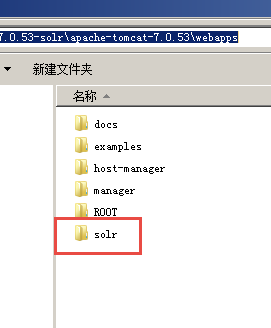
官方提供的solr的war包是dist\solr-4.10.3.war

#### 第一步：部署solr.war

将dist\solr-4.10.3.war拷贝到Tomcat的webapp目录下改名为solr.war

启动tomcat后，solr.war自动解压，将原来的solr.war删除

生成solr工程目录：



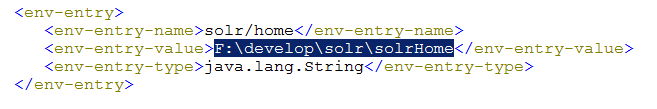
#### 第二步：拷贝扩展包到solr工程

拷贝example\lib\ext 目录下所有jar包拷贝到Tomcat的webapp\solr\WEB-INF\lib目录下

#### 第三步：设置solrHome

修改Tomcat目录 下webapp\solr\WEB-INF\web.xml文件，如下所示：

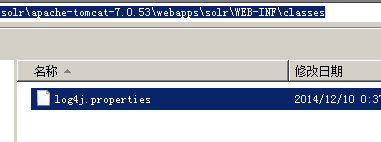
设置Solr home



#### 第四步：拷贝log4j.properties文件

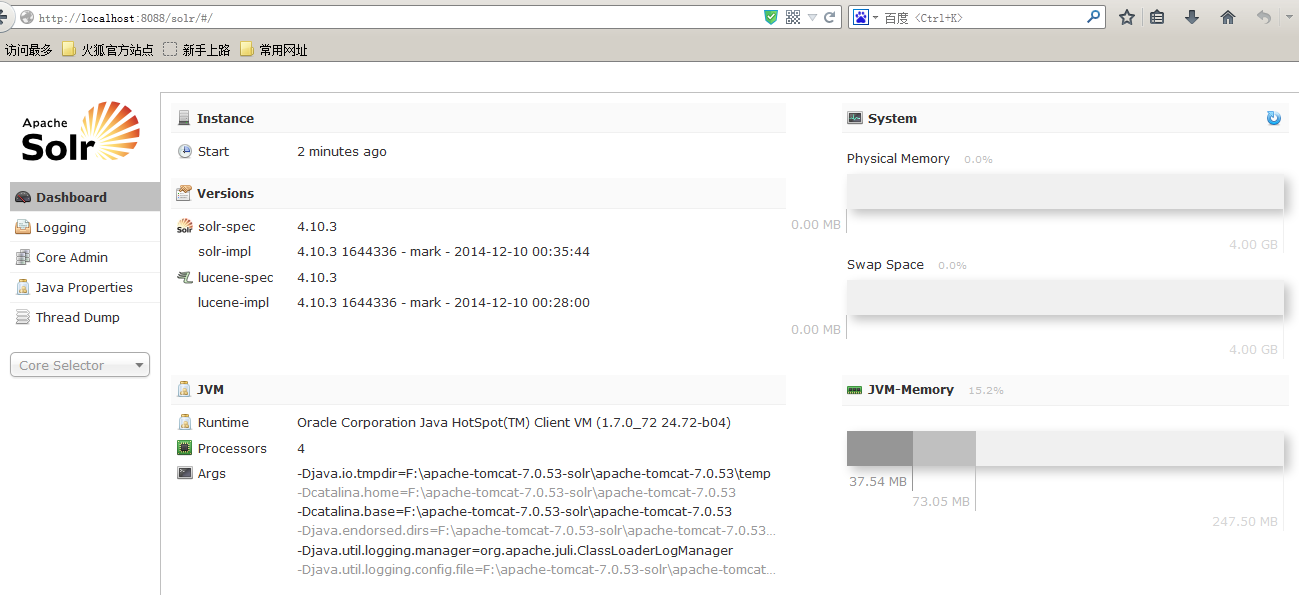
在 Tomcat下webapps\solr\WEB-INF目录中创建文件 classes文件夹，

复制Solr目录下example\resources\log4j.properties至Tomcat下webapps\solr\WEB-INF\classes目录



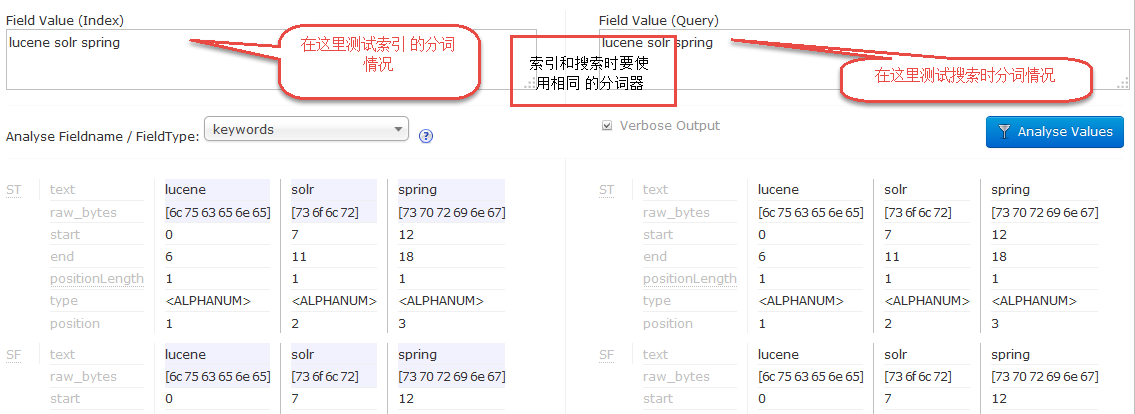
### 启动tomcat

solr的主界面

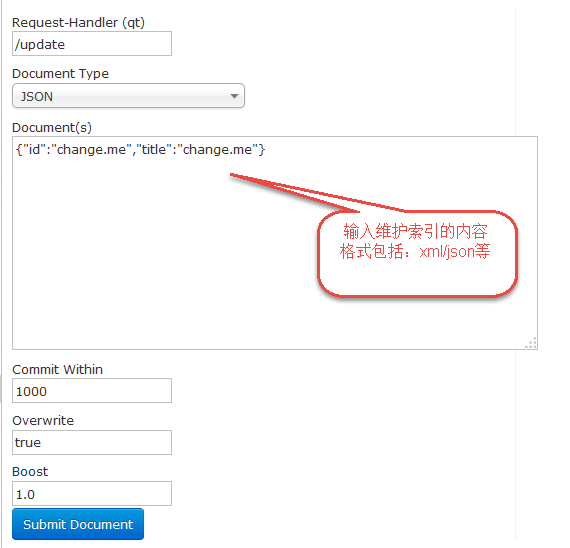


### solr工程测试

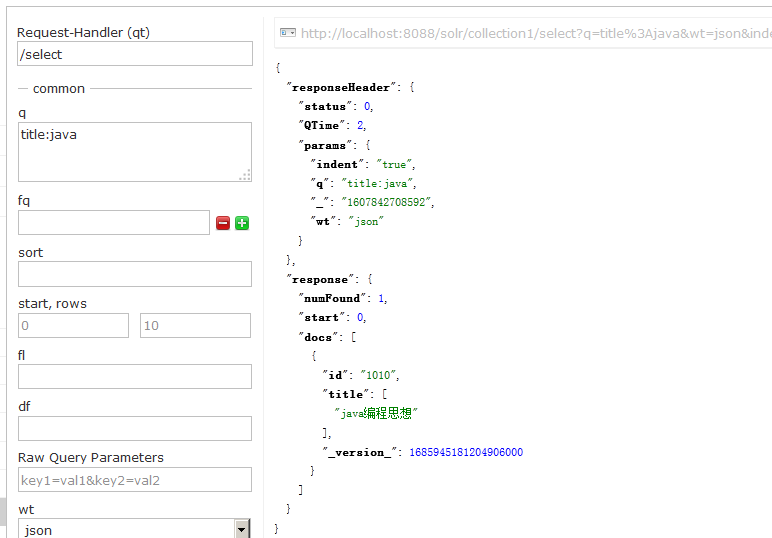
如下可以测试solr下各各Field的索引和搜索的分词效果。



如下是solrcore的索引维护界面：



如下是SolrCore的搜索界面：



### 多SolrCore配置

一个solr工程可以通过多个solrcore分别对外提供索引和搜索服务。

每个solrCore单独对外提供http的索引和搜索服务，相关于一个mysql数据（Solr工程）运行了多个数据库（solrcore）

配置多SolrCore的好处：

1. 一个solr工程对外通过SorlCore 提供服务，每个SolrCore相当于一个数据库，这个功能就相当于一个mysql可以运行多个数据库。
2. 将索引数据分SolrCore存储，方便对索引数据管理维护。
3. SolrCloud集群需要使用多core。

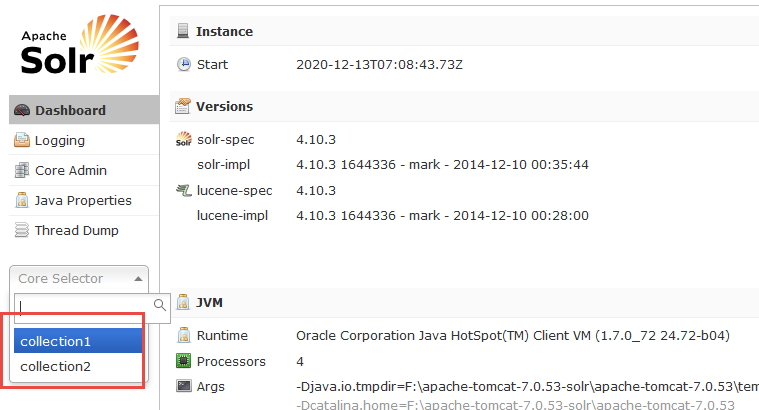
复制原来的core目录为collection2，目录结构如下：



修改collection2下的core.properties，如下：



启动tomcat：



测试多SolrCore是否分别对外提供服务？？

# Solr的基本使用

## schema.xml

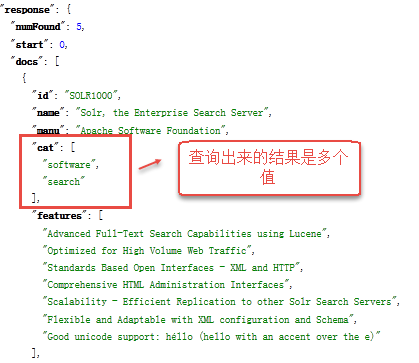
### Field和fieldType（重点）

schema.xml，在SolrCore的conf目录下，它是Solr数据表配置文件，它定义了加入索引的数据的数据类型的。主要包括FieldTypes、Fields和其他的一些缺省设置。

field：进行索引，需要创建document，document中包括 了很多的field（域）。

field属性：是否索引、是否存储、是否分词 ，是否多值multiValued

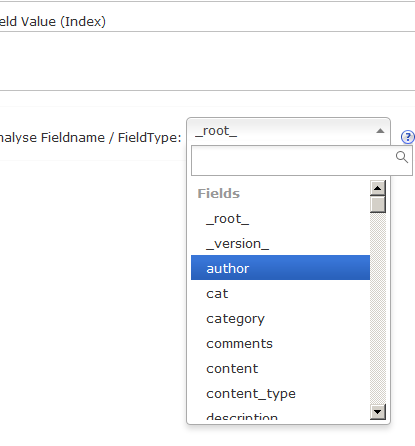
multiValued：该Field如果要存储多个值时设置为true，solr允许一个Field存储多个值，比如存储一个用户的好友id（多个），商品的图片（多个，大图和小图），通过使用solr查询要看出返回给客户端是数组：



Fieldtype：在solr中对每个Field都有一个Type类型。

**在Solr中进行索引、搜索时需要用哪些field需要提前在schema.xml文件中定义**！！！！

通过界面查看 solrCore中有哪些field被定义：



比如：

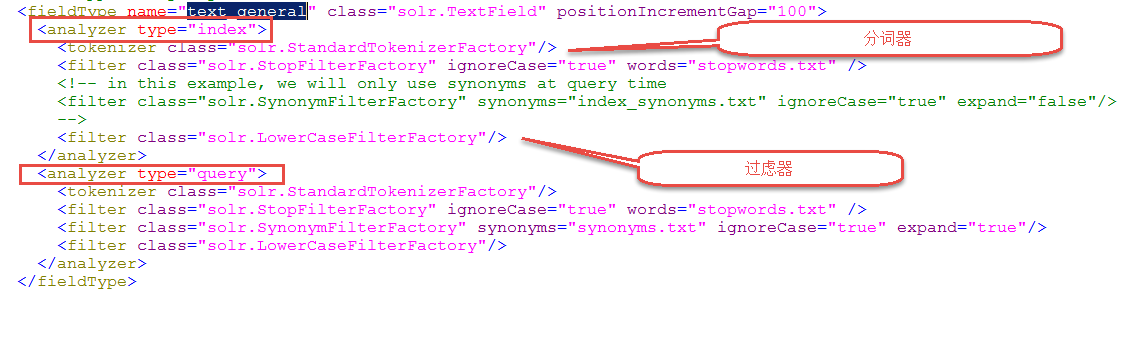
<field name="name" type="text\_general" indexed="true" stored="true"/>

filed的name：name

indexed：是否索引，true表示要索引，索引的目的为了搜索

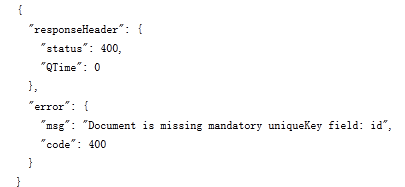
stored：是否存储，存储的目的是为了从索引文件读取数据

type：text\_general



### 主键（重点）

Solr在维护索引时要根据主键进行维护，比如创建索引没有指定主键，报如下错误：



如何指定solr的主键：

在schema.xml配置<uniqueKey>id</uniqueKey>

Solr中默认定义唯一主键key为id域，如下：



Solr在删除、更新索引时使用id域进行判断，也可以自定义唯一主键。

注意在创建索引时必须指定唯一约束。

### copyField复制域

copyField复制域，可以将多个Field复制到一个Field中，以便进行统一的检索：

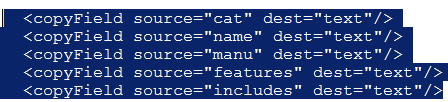
比如，输入关键字要搜索title标题和内容content这两个域时，要用到复制域

定义目标域：



必须要使用：multiValued="true"

将以下的域复制到目标域中：



如何在搜索时，搜索text域，solr会分别从以上cat、name、manu、features、includes这域中搜索，类似lucene中组合 域 搜索。

### dynamicField

使用 solr时要定义很多field比较麻烦，可以定义一种动态域，动态域没有具体的域名是定义一个规则，比如：name 为\*\_i。

当创建索引，指定field名称匹配上规则solr就执行成功，比如：动态域\*\_i，只要以\_i结尾 的就可以索引创建成功。

## 安装中文分词器

### IKAnalyzer部署

将 Ikanalyzer的jar拷贝到Solrcore中

将IKAnalyzer2012FF\_u1.jar拷贝到 solr\contrib\IKAnalyzer\lib下

在solrconfig.xml中配置加载IKanalyzer的jar，添加配置：



配置扩展词库文件：

在Tomcat的webapps/solr/WEB-INF/下创建classes目录

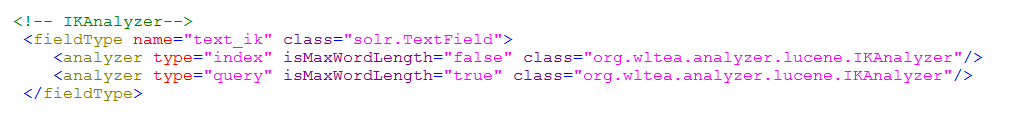
将IKAnalyzer.cfg.xml、ext\_stopword.dic mydict.dic copy到

Tomcat/webapps/solr/WEB-INF/classes下

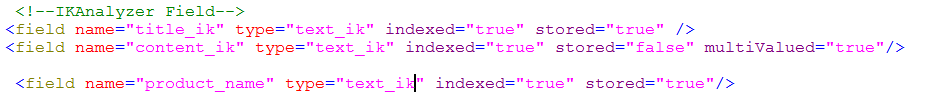
注意：ext\_stopword.dic 和mydict.dic必须保存成无BOM的utf-8类型。

### 修改schema.xml文件添加新的fieldType及Field测试

添加fieldType



自定义Field:



## 自定义业务系统Field

要使用Solr去实现业务系统站内搜索功能，需要在solrCore中schema.xml文件中自定义field。

### 需求

要使用solr实现电商网站中商品搜索。

电商中商品信息在mysql数据库中存储了，将mysql数据库中数据在solr中创建索引。

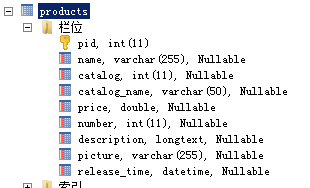
需要在solr的schema.xml文件定义商品Field。

### 定义步骤

先确定定义的商品document的field有哪些？

可以根据mysql数据库中商品表的字段来确定：

products商品表：



商品document的field包括：pid、name、catalog、catalog\_name、price、description、picture

先定义Fieldtype：

solr本身提供的fieldtype类型够用了不用定义新的了。

再定义Field：

pid：商品id主键

使用solr本身提供的：

<field name="id" type="string" indexed="true" stored="true" required="true" multiValued="false" />

name：商品名称

<field name="product\_name" type="text\_ik" indexed="true" stored="true"/>

catalog：商品分类

<field name="product\_catalog" type="string" indexed="true" stored="true"/>

catalog\_name：商品分类名称

<field name="product\_catalog\_name" type="text\_ik" indexed="true" stored="true"/>

price：商品价格

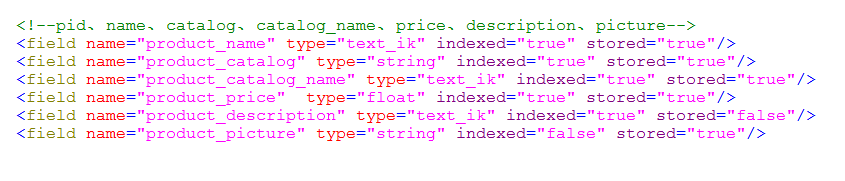
<field name="product\_price" type="float" indexed="true" stored="true"/>

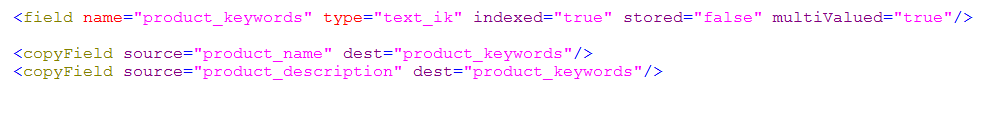
description：商品描述

<field name="product\_description" type="text\_ik" indexed="true" stored="false"/>

picture：商品图片

<field name="product\_picture" type="string" indexed="false" stored="true"/>



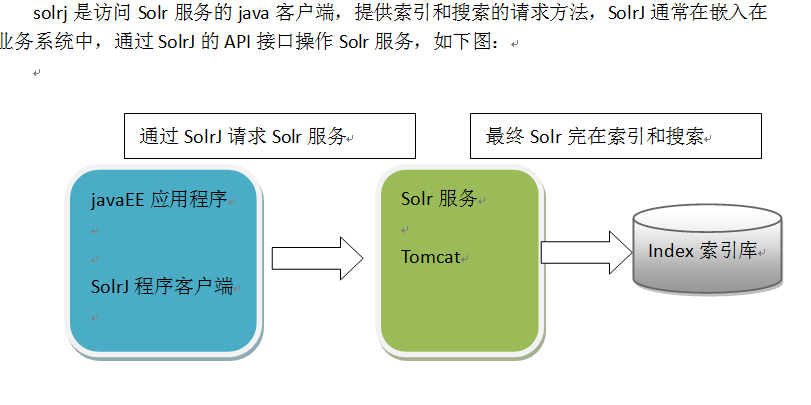


## SorlJ创建索引

### SorlJ介绍

Solr是一个全文检索的服务，对外提供webservice服务，webservice使用的http协议。

Solr作为一个服务端，同时也提供很多的不同编程语言的客户端，供程序员开发使用。

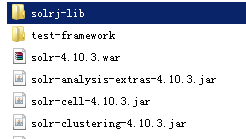


### SorlJ安装

solrj就是一推api，即jar包。

业务系统的应用程序（javaEE程序）通过solrJ提供的api调用solr服务。

下载的solr的zip包中就有solrj的jar:



要使用solrJ需要哪些jar包？

jar包明细：

solr-4.10.3\dist\solrj-lib\\*

solr-4.10.3\dist\ solr-solrj-4.10.3.jar

solr-4.10.3\example\lib\ext\\*

junit包

### Solrj索引维护

#### 添加及更新索引

//solr更新或添加索引，根据id判断，如果id存在进行更新索引，如果不存在进行添加索引

@Test

**public** **void** testUpdateIndex() **throws** SolrServerException, IOException{

//创建solr服务对象，通过此对象向solr服务发起请求

SolrServer server = **new** HttpSolrServer(solrUrl);

//创建document

SolrInputDocument doc =**new** SolrInputDocument();

//向文件中添加域

doc.addField("id", "8888");

doc.addField("product\_name", "lucene编程指南");

doc.addField("product\_price", 88.8f);

doc.addField("product\_catalog\_name", "计算机编程");

doc.addField("product\_description", "lucene编程指南是一本非常有用的书！");

doc.addField("product\_picture", "373737722.jpg");

// 创建索引

UpdateResponse response = server.add(doc);

//提交

server.commit();

}

#### 删除索引

//删除索引

@Test

**public** **void** testDeleteIndex() **throws** SolrServerException, IOException{

//创建solr服务对象，通过此对象向solr服务发起请求

SolrServer server = **new** HttpSolrServer(solrUrl);

//根据主键删除

// server.deleteById("1013");

//自定义查询条件删除

server.deleteByQuery("id:1012");

server.commit();

}

## 数据导入处理器

### dataimport-Handler介绍

solr提供dataimport-Handler数据导入处理器，工作流程：

1、solr通过dataimport-Handler查询关系数据库中的数据

2、对查询到的数据创建索引

上边的过程是自动化完成的。

可以通过此dataimport-Handler，批量将关系数据库中的数据创建索引到solr索引库中。

### 第一步：加载dataimport的jar包

将solr\dist\ solr-dataimporthandler-4.10.3.jar拷贝至solr\contrib\dataimporthandler

配置solrconfig.xml加载jar包：



### 第二步：加载mysql数据 驱动包

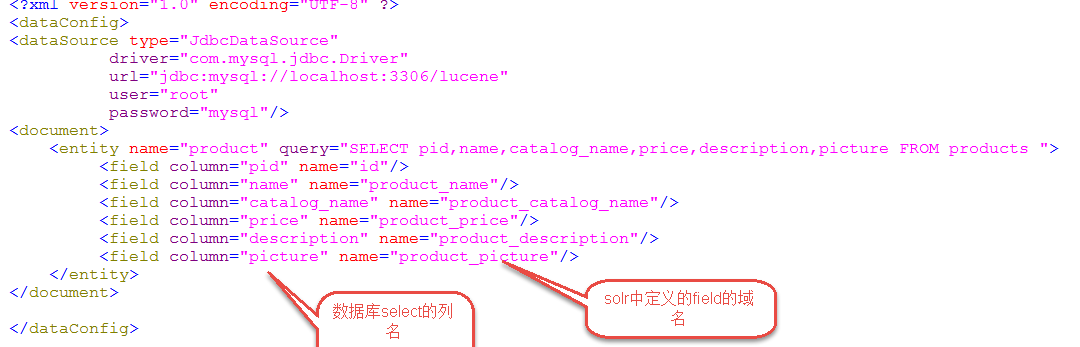
方法同上，将mysql数据库驱动的jar拷贝至solr\contrib\databaseDriver下

配置solrconfig.xml加载jar包：

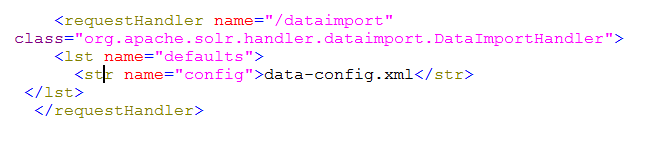


### 第三步：配置dataimport导入数据

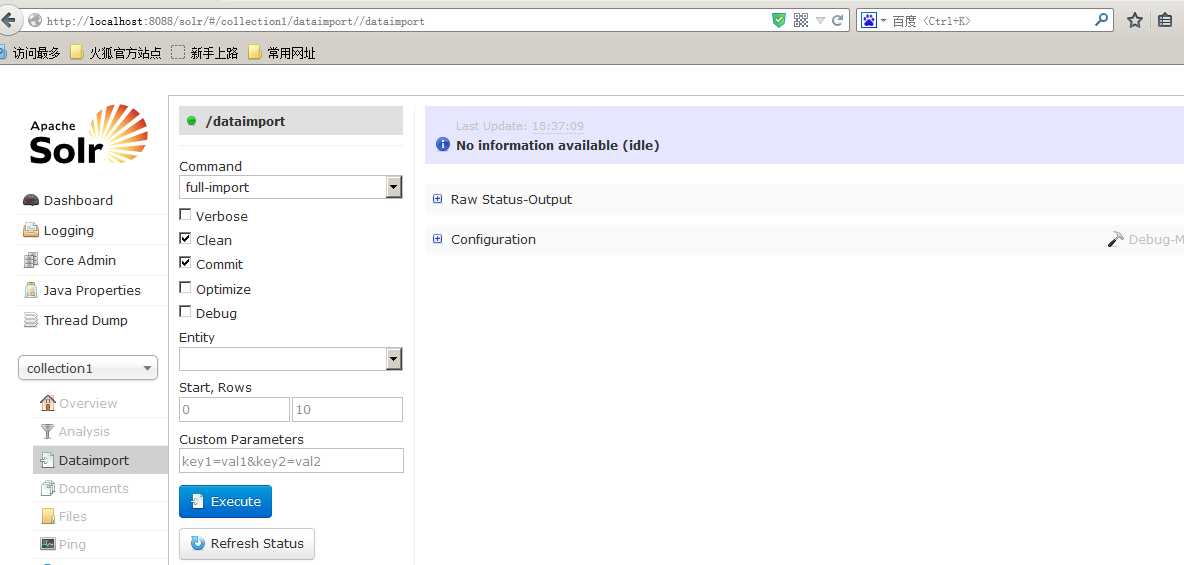
创建data-config.xml文件，存放在SolrCore的conf目录



### 第四步：修改solrconfig.xml，添加requestHandler



### 测试数据导入

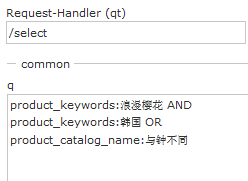


## solrj进行搜索

### solr的查询语法

1. q - 查询关键字，必须的，如果查询所有使用\*:\*。

请求的q是字符串



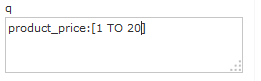
1. fq - （filter query）过虑查询，作用：在q查询符合结果中同时是fq查询符合的，例如：：

请求fq是一个数组（多个值）



过滤查询价格从1到20的记录。

也可以在“q”查询条件中使用product\_price:[1 TO 20]，如下：



也可以使用“\*”表示无限，例如：

20以上：product\_price:[20 TO \*]

20以下：product\_price:[\* TO 20]

1. sort - 排序，格式：sort=<field name>+<desc|asc>[,<field name>+<desc|asc>]… 。示例：

按价格降序

1. start - 分页显示使用，开始记录下标，从0开始
2. rows - 指定返回结果最多有多少条记录，配合start来实现分页。

实际开发时，知道当前页码和每页显示的个数最后求出开始下标。

1. fl - 指定返回那些字段内容，用逗号或空格分隔多个。

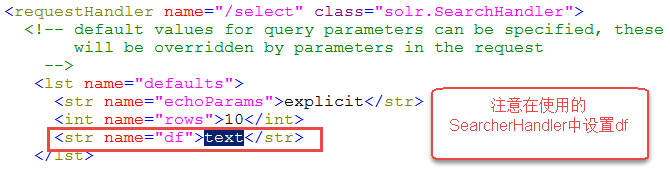


显示商品图片、商品名称、商品价格

1. df-指定一个搜索Field



也可以在SolrCore目录 中conf/solrconfig.xml文件中指定默认搜索Field，指定后就可以直接在“q”查询条件中输入关键字。



1. wt - (writer type)指定输出格式，可以有 xml, json, php, phps, 后面 solr 1.3增加的，要用通知我们，因为默认没有打开。
2. hl 是否高亮 ,设置高亮Field，设置格式前缀和后缀。



### solrj搜索方法

// 搜索索引

@Test

**public** **void** testSearch() **throws** SolrServerException {

// 创建solr服务对象，通过此对象向solr服务发起请求

SolrServer server = **new** HttpSolrServer(solrUrl);

//创建查询对象

SolrQuery query =**new** SolrQuery();

//查询关键字，q不能省略，必须写q

query.set("q", "product\_keywords:睡枕");

//指定过虑

// query.set("fq", "product\_price:[1 TO 100]");

//要添加多个 过虑

query.add("fq", "product\_price:[1 TO 100]");

query.add("fq", "product\_catalog\_name:四季用品");

//排序

//参数：field域，排序类型（asc、desc）

query.addSort("product\_price", ORDER.*desc*);

//加多个排序

// query.addSort(field, order)

//分页

//实际开发时，知道当前页码和每页显示的个数最后求出开始下标

**int** curPage=1;

**int** rows = 15;

//计算出开始记录下标

**int** start = rows \* (curPage -1);

//向 query中设置分页参数

query.setStart(start);

query.setRows(rows);

//指定显示的field

//query.addField("product\_name");

//query.addField("id");

//指定默认搜索域

query.set("df", "product\_keywords");

//开启高亮

query.setHighlight(**true**);

//设置高亮 参数

query.addHighlightField("product\_name");

//可以添加多个域高亮

// query.addHighlightField(f)

//设置高亮前缀和后缀

query.setHighlightSimplePre("<span style=\"color:red\">");

query.setHighlightSimplePost("</span>");

//执行查询

QueryResponse response = server.query(query);

//从响应中得到结果

SolrDocumentList docs = response.getResults();

//匹配到的总记录数

**long** numFound = docs.getNumFound();

System.*out*.println("匹配到的总记录数:"+numFound);

//从响应中获取高亮信息

Map<String, Map<String, List<String>>> highlighting = response.getHighlighting();

**for**(SolrDocument doc:docs){

System.*out*.println("商品id："+doc.get("id"));

System.*out*.println("商品名称："+doc.get("product\_name"));

System.*out*.println("商品价格："+doc.get("product\_name"));

System.*out*.println("商品分类名称："+doc.get("product\_catalog\_name"));

System.*out*.println("商品图片："+doc.get("product\_picture"));

//要获取高亮的信息

**if**(highlighting!=**null**){

//根据主键获取高亮信息

Map<String, List<String>> map = highlighting.get(doc.get("id"));

**if**(map!=**null**){

//获取高亮信息

List<String> list = map.get("product\_name");

**if**(list!=**null**){

System.*out*.println("高亮后："+list.get(0));

}

}

}

}

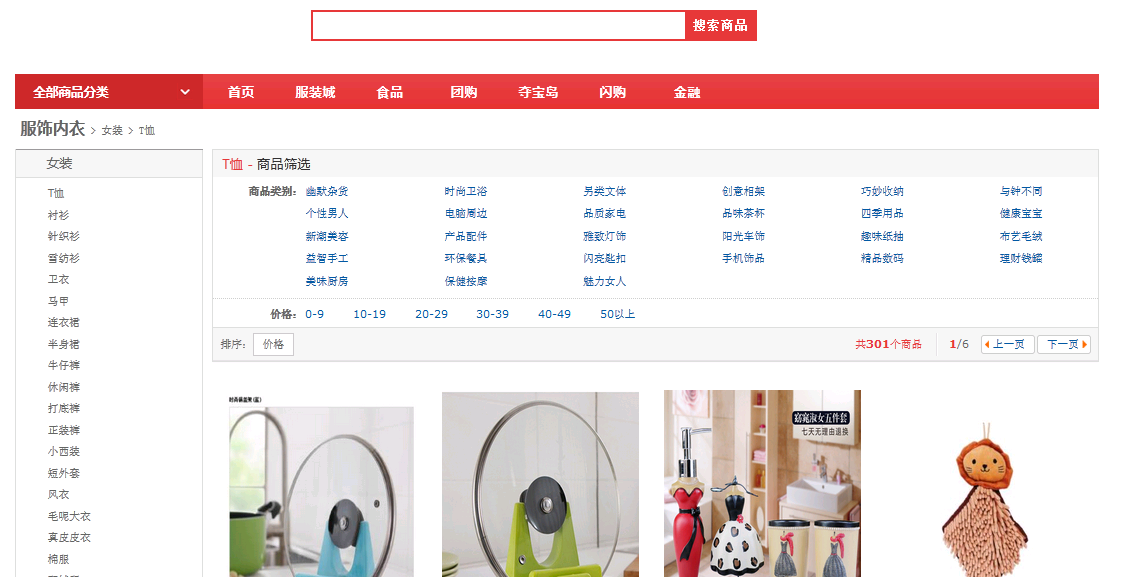
}

# Solr开发案例

## 案例需求

使用Solr实现电商网站中商品信息搜索功能，可以根据关键字、分类、价格搜索商品信息，也可以根据价格进行排序。

界面如下：



1、根据关键字搜索商品信息

使用solr中q查询关键字搜索

2、根据分类搜索

使用solr中q查询关键字搜索



可以使用过虑：



3、根据价格范围搜索

使用过虑



4、根据价格排序

使用过虑



点击一下升序，再点一下是降序。

5、分页实现

6、高亮

以上搜索条件是组合查询，可以同时满足这些条件进行搜索。

## 技术架构

spring容器

表现层

springmvc

Service层

SolrJ

Dao层

Solr索引库

mysql数据库

商品信息

Tomcat

Solr服务

索引、搜索请求

整个架构分三块儿：

* solr服务
* 业务系统(重点)

springmvc：实现用户搜索的交互（controller控制器和页面）

service：

接收controller的请求去搜索商品。

通过solrJ调用solr 服务搜索商品。

service调用dao进行商品信息维护时要更新mysql的同时将solr的索引更新。（保持mysql数据库中数据和solr索引同步）

Dao层：

对电商系统中，在dao中要对商品信息进行维护和管理。

* mysql数据库

存储了商品信息

索引流程如下：

1、管理员在业务系统中维护商品信息

2、将商品信息写入关系数据库

3、将商品信息写入Solr的索引库

（本例子不实现商品信息维护功能所以直接使用dataImport-handler处理器完成索引创建）

重点开发商品搜索，流程如下：

1、用户在页面中输入关键字及其它的查询条件

2、用户搜索请求到controller（使用springmvc实现）

3、controller调用service搜索商品信息

4、service内部通过solrJ调用solr服务搜索商品信息

## 工程环境

创建一个web工程

使用jdk1.7

开发工具：eclipse indigo

jar包：

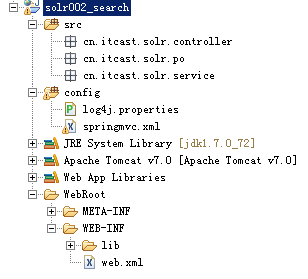
spring3.2.0(包括 springmvc)

jstl

junit

log4j

solrJ相关jar包



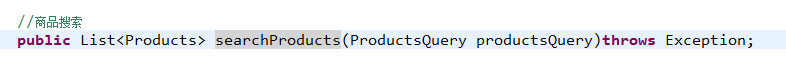
## 开发

### PO类

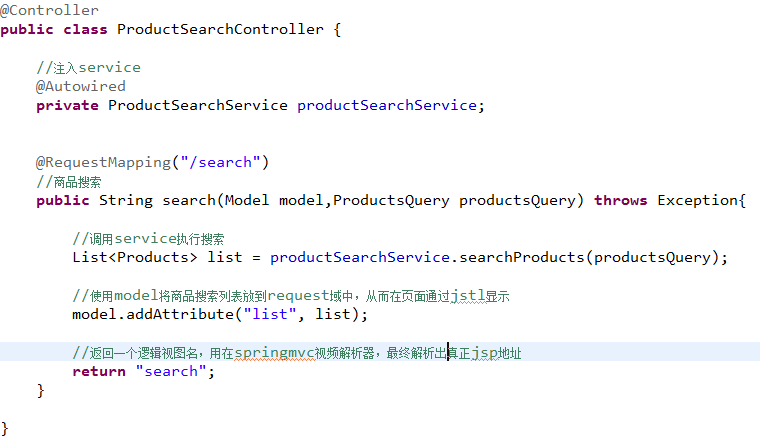
### Service

service接口是由controller调用，在service内部使用solrJ的方法搜索solr服务。

service和solrj耦合了，为了保证controller方法不和solrj进行耦合给service接口定义的方法参数不要用到solrj的类对象。



### controller



### jsp



### 配置springmvc.xml



### 配置web.xml

配置springmvc的前端控制器（servlet）

配置post乱码的过虑器

<!-- springmvc前端控制器 -->

<servlet>

<servlet-name>springmvc</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

<init-param>

<param-name>contextConfigLocation</param-name>

<param-value>classpath:springmvc.xml</param-value>

</init-param>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<!-- 以.action结尾的由springmvc前端控制器去解析 -->

<servlet-name>springmvc</servlet-name>

<url-pattern>\*.action</url-pattern>

</servlet-mapping>

<!-- post乱码 -->

<filter>

<filter-name>CharacterEncodingFilter</filter-name>

<filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>

<init-param>

<param-name>encoding</param-name>

<param-value>utf-8</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>CharacterEncodingFilter</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

### 主体功能调试

部署工程到tomcat

#### 图片的虚拟目录

切记不要将用户上传的图片放到工程目录下，不方便工程维护。

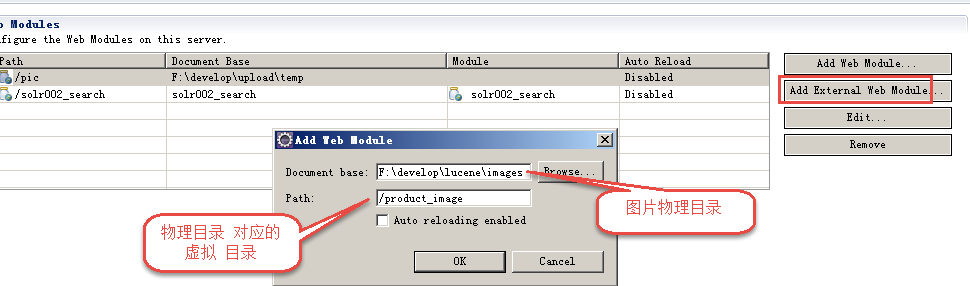
一般实际开发中，图片专门使用图片服务器存储图片。

创建图片虚拟目录提供图片服务：

物理图片地址：F:\develop\lucene\images

对应的虚拟目录：/product\_image

用户访问/product\_image就是访问F:\develop\lucene\images物理目录下的图片



也可以修改tomat下conf/server.xml文件定义虚拟目录 ：



### 功能完善