

## Solr的安装

### 1. 环境要求

jdk1.7+tomcat8+solr5.5.0

### 2. 将以上的软件包上传到服务器

### 3.

安装tomcat8(解压文件)

```
tar -zxvf apache-tomcat-8.5.24.tar.gz
```

### 4.

解压solr

```
tar -zxvf solr-5.5.0.tgz
```

#### 1. 将solr的服务拷贝到tomcat的webapps下

```
cp -r solr-5.5.0/server/solr-webapp/webapp tomcat8/webapps/solr
```

#### 2. 需要将solr需要日志的jar拷贝到tomcat8/webapp/solr/WEB-INF/lib

```
cp -r solr-5.5.0/server/lib/ext/*.jar tomcat8/webapps/solr/WEB-INF/lib
```

#### 3. 需要将日志的log4j配置文件拷贝到项目中

```
mkdir -p tomcat8/webapps/solr/WEB-INF/classes
```

```
cp -r solr-5.5.0/server/resource/log4j.properties tomcat8/webapps/WEB-INF/classes
```

#### 4. 配置solrHome (存放索引)

```
mkdir solr-home
```

```
vi tomcat8/webapps/solr/WEB-INF/web.xml
```

打开如下配置

```
<env-entry>
```

```
    <env-entry-name>solr/home</env-entry-name>
```

```
    <env-entry-value>/put/your/solr/solr-home</env-entry-value>
```

```
    <env-entry-type>java.lang.String</env-entry-type>
```

```
</env-entry>
```

拷贝solr的配置文件到solr-home

```
cp -r solr-5.5.0/server/solr/* /root/solr/solr-home
```

### 5. 启动tomcat

```
tomcat8/bin/startup.sh
```

### 6. <http://192.168.137.129:8080/solr/admin.html>

## 核心概念

1. solr服务器 --> 理解为数据库

2. 核 --> 表

3. field --> 表中字段

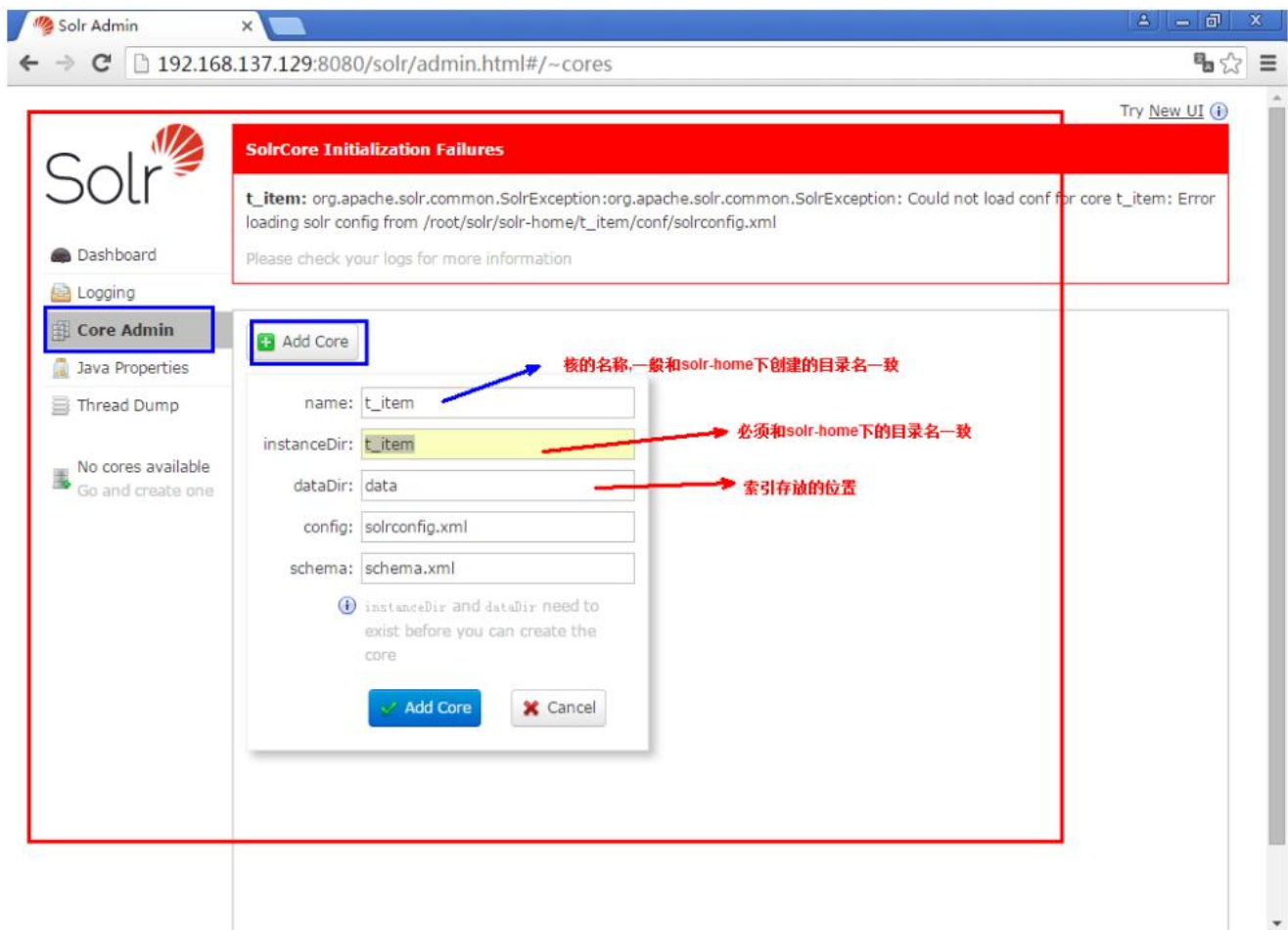
## 创建核

1. 在solr-home下创建一个目录 (目录名随意, 例如t\_item)

2. 拷贝核中所需要的配置文件(solr-home下configsets里面)

```
cp -r configsets/sample-techproducts-configs/* t_item
```

3. 主界面配置 (见下图)



系统界面操作

Solr

Dashboard  
Logging  
Core Admin  
Java Properties  
Thread Dump

t\_item

Overview  
Analysis  
Dataimport  
**Documents**  
Files  
Ping  
Plugins / Stats  
Query  
Replication  
Schema Browser

/update

Document Type  
XML

Document(s)

```
<doc>  
<field name="id">change.me</field><field name="title">change.me</field></doc>
```

Commit Within  
1000

Overwrite  
true

Submit Document

Documentation Issue Tracker IRC Channel Community forum

新增doc 修改doc 删除doc  
Json XML

1. 新增和修改（id在索引库中存在就是修改，不存在就是新增）  
{"id":"change.me","title":"change.me"}

id必须存在，可以理解为主键

key必须先配置（managed-schema）再使用

```
1 2. 删除使用xml  
2 <delete>  
3   <id>change.me</id>  
4 </delete>  
5 <commit/>
```

3. 查询(见下图)

The screenshot shows the Solr Admin interface with a query configuration on the left and the resulting JSON response on the right. Red arrows and annotations explain the mapping:

- 查询关键字** (Query Keyword): Points to the `q` parameter field containing `title:change.me1`.
- 附带条件相当于我们的AND** (Additional conditions are equivalent to our AND): Points to the `fq` (filter query) field.
- 排序 和数据库语法一致** (Sorting is consistent with database syntax): Points to the `sort` field.
- 分页和limit参数一致** (Pagination is consistent with limit parameters): Points to the `start` and `rows` fields.
- 指定查询的域** (Specify the domain to query): Points to the `df` (default field) field.
- 默认查询域** (Default query domain): Points to the `wt` (write method) field set to `json`.
- 高亮展示** (Highlighting display): Points to the `hl` (highlighting) field.

The JSON response on the right shows the query results, including the highlighted text: `<span style='color:red;'>change.me1</span>`.

## java中操作Solr

```

1  1.添加依赖
2  <dependency>
3    <groupId>org.apache.solr</groupId>
4    <artifactId>solr-solrj</artifactId>
5    <version>5.5.0</version>
6  </dependency>
7  2.使用API
8    1.获取和服务端连接
9      String baseURL = "http://192.168.137.129:8080/solr/t_item"; //服务器地址，最后一个表
      示核
10     SolrClient solrClient = new HttpSolrClient(baseURL);
11     2.调用
12     solrClient中的增删改查操作

```

## Solr添加域

```

1 1. 需要在solr-home，找到对应的核的目录，conf/managed-schema
2      vi conf/managed-schema
3      <!--配置自定义的域-->
4      <field name="content_ik" type="text_ik" indexed="true" stored="true"/>
5      <fieldType name="text_ik" class="solr.TextField">
6          <analyzer class="org.wltea.analyzer.lucene.IKAnalyzer"></analyzer>
7      </fieldType>
8
9 2. 将IK分词器拷贝到tomcat8/webapps/solr/WEB-INF/lib
10      cp ik-analyzer-5.3.0.jar tomcat8/webapps/solr/WEB-INF/lib
11
12 3. 重启服务器
13
14
15 IK分词器需要添加扩展词典
16     将配置文件IKAnalyzer.cfg.xml和词典上到 tomcat8/webapps/solr/WEB-INF/classes
17

```

## 在工作中使用Solr

```

1 1. 创建业务域
2      <!--业务域类型定义-->
3      <field name="content_ik_nostore" type="text_ik" indexed="true" stored="false"/>
4      <fieldType name="text_ik" class="solr.TextField">
5          <analyzer class="org.wltea.analyzer.lucene.IKAnalyzer">
6          </analyzer>
7      </fieldType>
8
9      <!--业务域名-->
10     <field name="item_title" type="text_ik" indexed="true" stored="true"/>
11     <field name="item_sell_point" type="text_ik" indexed="true" stored="true"/>
12     <field name="item_price" type="long" indexed="true" stored="true"/>
13     <field name="item_image" type="string" indexed="false" stored="true"/>
14     <field name="item_created" type="date" indexed="true" stored="true"/>
15
16     <!--拷贝域-->
17     <field name="item_keywords" type="text_ik" indexed="true" stored="true"
18     multiValued="true"/>
19     <copyField source="item_title" dest="item_keywords"/>
20     <copyField source="item_sell_point" dest="item_keywords"/>
21
22 2. 加载数据
23     使用加载的数据，包裹以上域名包装为SolrInputDocument
24     调用SolrClient中add方法添加Solr库中

```

## java调用solr服务器的相关代码

### managed-schema配置文件修改

solr-home/某核心/conf/managed-schema文件中添加的配置

```

1 <!-- 自定义 field -->
2 <field name="content_ik" type="text_ik" indexed="true" stored="true" />
3
4 <field name="title_ik" type="text_ik" indexed="true" stored="true" />
5 <field name="info_ik" type="text_ik" indexed="true" stored="true" />
6
7 <!-- 自定义 fieldType -->
8 <fieldType name="text_ik" class="solr.TextField" >
9   <analyzer class="org.wltea.analyzer.lucene.IKAnalyzer" ></analyzer>
10 </fieldType>
11
12 <!-- copyField的使用 -->
13 <field name="keywords_ik" type="text_ik" indexed="true" stored="true" multiValued="true" />
14 <copyField source="title_ik" dest="keywords_ik" />
15 <copyField source="info_ik" dest="keywords_ik" />

```

## Properties配置文件:

```

1 solr.url=http://localhost:8181/solr/t_user
2 solr.connectionTimeout=500

```

## 在junit测试类中写的代码

### 初始化

```

1 //相关初始化
2 private Properties p = new Properties();
3 private HttpSolrClient solrClient;
4 //private SolrClient solrClient; //使用此类时无法设置connectionTimeout等属性
5 @Before
6 public void before(){
7   try {
8     p.load(SolrTest.class.getClassLoader().getResourceAsStream("solrclient.properties"));
9   } catch (IOException e) {
10     e.printStackTrace();
11   }
12   //HttpSolrClient:启动web服务器使用的,通过http请求的
13   solrClient = new HttpSolrClient(p.getProperty("solr.url"));
14   int connectionTimeout = Integer.parseInt(p.getProperty("solr.connectionTimeout"));
15   solrClient.setConnectionTimeout(connectionTimeout);
16 }

```

### 高亮查询

```

1 public void queryHighlight(){
2     //关键字查询 域:值 *:匹配所有
3     //想完全匹配必须要用双引号
4     //String q = "title_ik:是 AND info_ik:\"中国人\"";
5     String q = "keywords_ik:国人";//keywords_ik是copyField
6     SolrQuery sq = new SolrQuery();
7     sq.setQuery(q);
8     sq.setHighlight(true);//开启高亮显示
9     sq.setHighlightSimplePre("<span color='red' >");//高亮格式开头
10    sq.setHighlightSimplePost("</span>");//高亮格式结尾
11    //sq.setStart(0).setRows(100);//分页
12
13    //hl.fl表示属性高亮, String... values表示哪些field需要高亮, copyField无法高亮
14    sq.setParam("hl.fl", "title_ik", "info_ik", "keywords_ik");
15    try {
16        QueryResponse response = solrClient.query(sq);
17
18        SolrDocumentList results = response.getResults();
19        System.out.println("总数:" + results.getNumFound());
20        int size = results.size();
21        System.out.println("当前总数:" + size);
22        for (int i = 0; i < size; i++) {
23            SolrDocument doc = results.get(i);
24            System.out.println(doc);
25            String id = (String) doc.get("id");
26            //在solr这里对需要加高亮的字段必须要在索引中的store=true才行
27            //获取对应的document高亮的key=value
28            Map<String, List<String>> map = response.getHighlighting().get(id);
29            List<String> titles = map.get("title_ik");
30            List<String> infos = map.get("info_ik");
31            if(null == titles){
32                titles = new ArrayList<String>();
33                String title = (String) doc.get("title_ik");
34                if(null != title){
35                    titles.add(title);
36                }
37            }
38            if(null == infos){
39                infos = new ArrayList<String>();
40                String info = (String) doc.get("info_ik");
41                if(null != info){
42                    infos.add(info);
43                }
44            }
45            System.out.println(id + " ,title_ik= " + titles + " ,info_ik= " + infos + "
,=keywords_ik= " + doc.get("keywords_ik"));
46        }
47    } catch (SolrServerException e) {
48        e.printStackTrace();
49    } catch (IOException e) {
50        e.printStackTrace();
51    }
52 }

```

## 添加Document

```
1 public void addDoc(SolrInputDocument doc){
2     //id是必填的,并且是String类型的
3     //id是唯一的主键,当多次添加的时候,最后添加的相同id会覆盖前面的域
4     try {
5         solrClient.add(doc);
6         solrClient.commit();
7     } catch (SolrServerException e) {
8         e.printStackTrace();
9     } catch (IOException e) {
10        e.printStackTrace();
11    }
12 }
```

## 修改Document

```
1 public void updateDocById(SolrInputDocument doc){
2     //修改实际上还是增加索引,只不过指定id,把相同id的filed覆盖掉
3     try {
4         solrClient.add(doc);
5         solrClient.commit();
6     } catch (SolrServerException e) {
7         e.printStackTrace();
8     } catch (IOException e) {
9         e.printStackTrace();
10    }
11 }
```

## 删除Document



```
1 public void deleteByIds(List<String> ids){
2     //根据多个id删除多个document
3     //List<String> ids = new ArrayList<String>();
4     try {
5         solrClient.deleteById(ids);
6         solrClient.commit();
7     } catch (SolrServerException e) {
8         e.printStackTrace();
9     } catch (IOException e) {
10        e.printStackTrace();
11    }
12 }
13
14 public void deleteById(String id){
15     //根据id删除document
16     try {
17         solrClient.deleteById(id);
18         solrClient.commit();
19     } catch (SolrServerException e) {
20         e.printStackTrace();
21     } catch (IOException e) {
22         e.printStackTrace();
23     }
24 }
25
26 public void deleteByQuery(String query){
27     //删除查询的结果
28     //String query = "title_ik:人";
29     try {
30         solrClient.deleteByQuery(query);
31         solrClient.commit();
32     } catch (SolrServerException e) {
33         e.printStackTrace();
34     } catch (IOException e) {
35         e.printStackTrace();
36     }
37 }
```