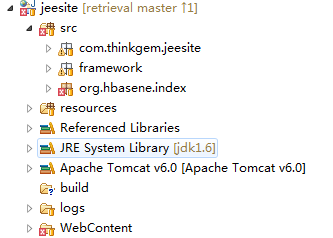
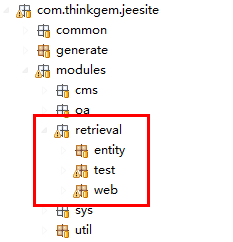
**全文检索项目**

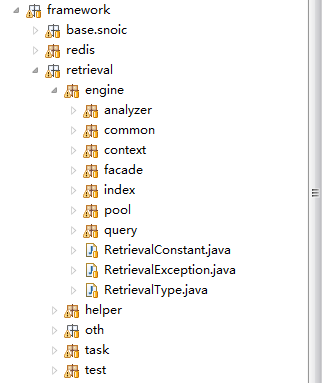
|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 作者 | 时间 | 备注 |
| V1.0 | 沈晓军 | 2013-12-23 |  |
|  |  |  |  |
|  |  |  |  |

# 开发指南

1. **项目结构**



1. **其中****中包括jeesite项目和文件检索页面展示，如下，所有全文检索相关的页面操作都放在这个retrieval下。**
2. **其中****中包括了全文检索，redis等所有内容，包括分词，建索引，搜索操作，都在这个下面**



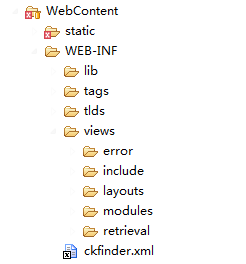
其中base.snoic包含了最基本的操作。

Redis包含了reids操作（暂时没有用，我集成了spring的redis）。

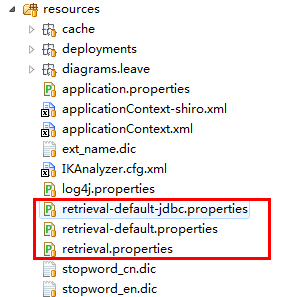
其中retrieval最为重要：

|  |  |  |
| --- | --- | --- |
| 主包 | 子包 | 作用 |
| Engine | analyzer | 分词包，例如将“中国人”分词分成“中国”和“中国人”，默认使用IK分词 |
| Common | Common中定义了lucene的版本等信息 |
| Context | 上下文，在检索和搜索时都通过这个上下文去获取基本信息 |
| Facade | 门面类，用于建索引和更新索引的入口 |
| Index | 建索引 |
| Pool | IndexRead和IndexWriter池，用来保证并发搜索的高效性 |
| query | 搜索 |
| Helper |  | 帮助类，用来分页查询 |
| Oth |  | 自己定义的类，包括，对象转json,对象转xml等等。今后会加入quartz |
| task |  | 任务模块，主要是建索引时的并发控制 |
| Test |  | 测试类（测试检索，建索引，删除、更新） |

1. **全文检索的所有页面都在retrieval下面：**



1. **配置文件**



全文检索用到的配置文件主要有这3个，

第一个jdbc方式的配置是用来配置建索引数据库的地址的，此文件今后要删掉的，数据库信息准备配置在redis中。

第二个是通用配置，配置了索引文件夹路径、redis等信息，今后所有需要配置的信息都存放到这个文件中。

第三个是全文检索建索引用到的配置文件，用来性能调优、配置合并因子和配置分词等等，这个可以不用管，应该用不到的。

1. **获取数据源的方式**

现在通过类DefaultDBGetMethodImpl这个实现类去完成从配置文件中加载数据源信息。如设置：

database.indexall.jdbc.driver=com.mysql.jdbc.Driver

database.indexall.jdbc.url=jdbc:mysql://127.0.0.1:3306/jeecg

database.indexall.jdbc.user=root

database.indexall.jdbc.password=11111

后续会将数据源存放到redis数据库中，通过redis方式去加载需要：

1.实现接口IDBGetMethod，例如  
**public** **class** RedisDBGetMethodImpl **implements** IDBGetMethod{

@Override

**public** DatabaseLink loadDatabaseLink() {

//doing some thing

**return** **null**;

}

}

2.在配置文件中retrieval-default.properties中配置

#配置采用获取数据源的方式，默认为framework.retrieval.engine.index.all.database.impl.DefaultDBGetMethodImpl:loadDatabaseLink

#其中冒号名前为类，冒号后为方法名，默认方法名为：loadDatabaseLink

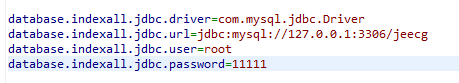
DEFAULT\_RETRIEVAL\_DATABASE\_CHOOSE\_CLASS=framework.retrieval.engine.index.all.database.impl.RedisDBGetMethodImpl:loadDatabaseLink

1. **测试类使用方法**

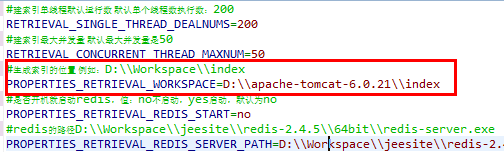
由于时间原因，我只测试了mysql数据库，使用方法简绍如下：

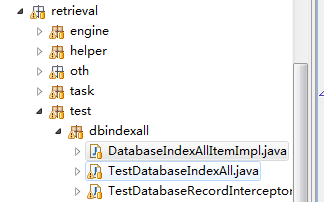
(1).配置retrieval-default-jdbc.properties中的数据库连接。（暂时只支持mysql）

配置参数如下：



(2)配置索引文件夹路径，在retrieval-default.properties中配置



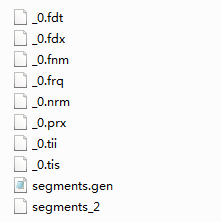
(3).测试建索引  


DatabaseIndexAllItemImpl用来设置数据库表，设置搜索出来的标题、正文等字段。

TestDatabaseIndexAll用来模拟索引的，配置好之后，右击run。

TestDatabaseRecordInterceptor拦截类，用来拦截数据库中查出的数据，进行调整，现在主要用来去除内容字段中的html标签。

当建完索引后，可以在文件夹中查看到索引：

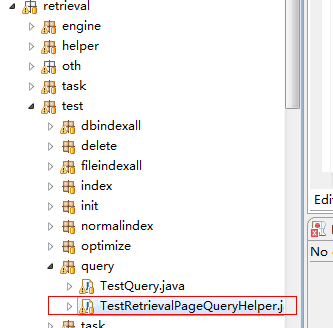


(3).测试搜索

有两种方式测试搜索，

第一种是直接页面访问：http://localhost:8080/jeesite/f/retrieval/search/

第二种是用测试类:TestRetrievalPageQueryHelper, 右击run.



具体使用可以参考下：  
双击打开这个pdf。

# 需求

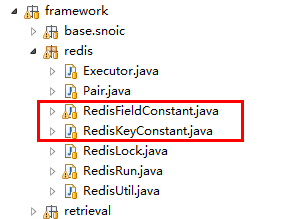
1. **需求一：数据源维护**

原数据源配置在retrieval-default-jdbc.properties这个文件中，现要将数据源配置到redis中。

修改入口DefaultRDatabaseIndexAllImpl类

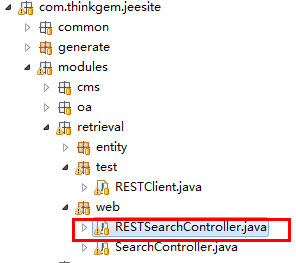


Reids数据库设计

数据源表用redis的hash表来存。键值对存放到这两个java文件中。  


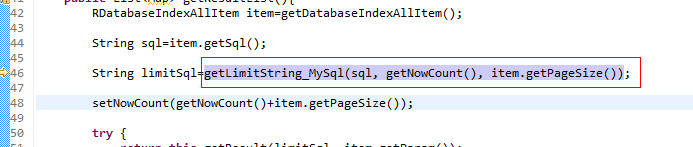
字段设计

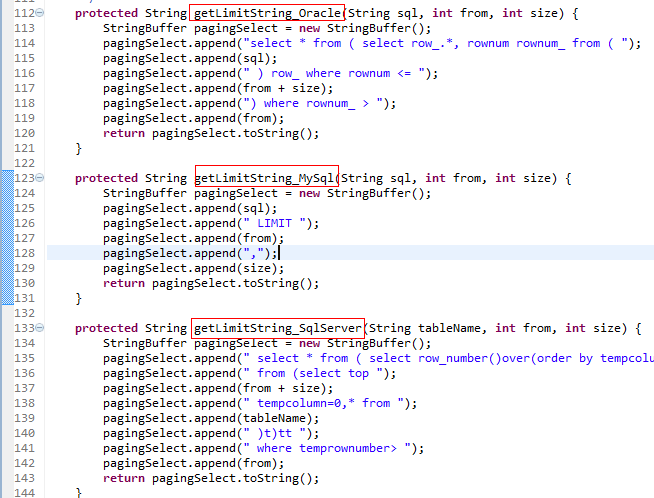
|  |  |  |
| --- | --- | --- |
| 主键 | 字段 | 值 |
| Database/类型/数据源名称/UUID  例如:  “Database /mysql/shujvyuan1/…” | type | 数据源类型： Oracle/Mysql/SqlServer |
| url | jdbc:mysql://127.0.0.1:3306/jeecg |
| driver | com.mysql.jdbc.Driver |
| user | root |
| password | 11111 |

页面可以做的简单点，但一定要能想rest一样能调用。参照下面这个类做：  


1. **需求二：针对不同数据库取数据(已修改——2013-12-24 沈晓军)**

现在只测了mysql方式，需要增加oracle和sqlserver方式读数据：

代码在在AbstractDefaultRDatabaseIndexAll类的第46行，我这边只做了mysql的分页：如下图所示：  


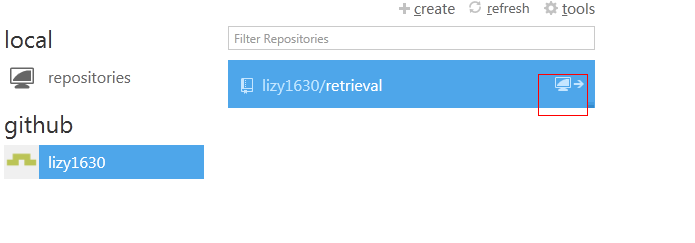
实际上应该包括Oracle和SqlServer的分页，分页代码我已经写了，在AbstractRDatabaseIndexAll类的第112行到144行，如下图：  


需要做一个配置项，确定取数据的数据源，然后测试下我上面写的分页是否正确。

# GitHub更新代码。

**Github上更新自己fork的代码**

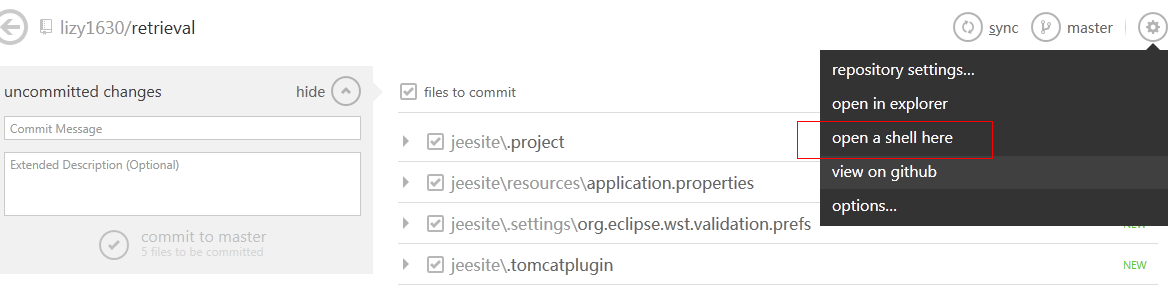
检出自己在github上fork别人的分支到rrest目录下，其中rrest目录之前是不存在的。



git clone git@github.com:yss/rrestjs.git rrest

或者如上图，直接clone这个分支到本地路径下；

然后打开shell增加远程分支（也就是你fork那个人的分支）名为bob（这个名字任意）到你本地。



git remote add bob <https://github.com/DoubleSpout/rrestjs.git>

如果你运行命令：git remote -v你会发现多出来了一个Bob的远程分支。如下：

bob https://github.com/DoubleSpout/rrestjs.git (fetch)

bob https://github.com/DoubleSpout/rrestjs.git (push)

origin git@github.com:yss/rrestjs.git (fetch)

origin git@github.com:yss/rrestjs.git (push)

然后，把对方的代码拉到你本地。

git fetch bob

最后，合并对方的代码。

git merge bob/master

最最后，把最新的代码推送到你的github上。

git push origin master

这样就完成了自己的代码更新。