主 题：日报

汇报人：左超

日 期：2018年7月25日 星期3

内 容：

* 今日学习了的内容：

Spring Bean对象的2种最常用的构造方法：

<!-- 构造实例化（无参数构造器） 最标准 使用最多 -->

<bean id = *"cat1"* class = *"cn.wolfcode.creat.constructor.Cat1"*/>

<!--实现BeanFactory接口 -->

<beanid = *"cat2"* class=*"cn.wolfcode.creat.constructor.Cat2Factory"*/>

测试：

@SpringJUnitConfig

//如果没有路径参数，则将配置文件更改为类名-context.xml 且只适用于junit5

**public** **class** App {

@Autowired

**private** Cat1 cat1;

@Autowired

**private** Cat2 cat2;

@Test

**void** testCat1() **throws** Exception {

System.***out***.println(cat1);

}

@Test

**void** testCat2() **throws** Exception {

System.***out***.println(cat2);

}

}

lucence的初次使用：

public void testCreate() throws Exception {

Directory d =FSDirectory.*open*(Paths.*get*(***PATH***));

IndexWriterConfig conf = **new** IndexWriterConfig(**new** SimpleAnalyzer()); //对写入做配置--分词器

IndexWriter indexwriter = **new** IndexWriter(d,conf);

Document document1 = **new** Document(); //添加行

document1.add(**new** TextField("id","1",Store.***YES***));//添加列

document1.add(**new** TextField("title","doc1",Store.***YES***));

FieldType type = **new** FieldType(); //字段类型

type.setStored(**true**); //在数据区是否存储

type.setIndexOptions(IndexOptions.***DOCS\_AND\_FREQS\_AND\_POSITIONS\_AND\_OFFSETS***); //该字段是否创建索引

type.setTokenized(**true**); //创建索引时是否分词

Field contentField = **new** Field("content",doc1,type);

document1.add(contentField); //添加列1

indexwriter.addDocument(document1); //添加到缓冲区

indexwriter.commit();

indexwriter.close();

**package** cn.ElasticSearch.Demo1;

**public** **class** LucenceTest {

**private** **static** **final** String ***PATH*** = "D:/workspace02/Elasticsearch\_Deno/index/01\_hello";

**private** **static** String *doc1* = "hello world";

**private** **static** String *doc2* = "hello java world";

**private** **static** String *doc3* = "hello lucence world";

/\*\*

\* 创建索引

\* 1、创建InderWriter

\* 2、要把创建的索引的文本数据放入Document的字段中

\* 3、通过InderWriter将document写入

\*/

**public** **static** **void** testCreat()**throws** Exception {

//创建IndexWriter

Directory d =FSDirectory.*open*(Paths.*get*(***PATH***)); //索引目录

IndexWriterConfig conf = **new** IndexWriterConfig(**new** SimpleAnalyzer()); //对写入做配置--分词器

IndexWriter indexwriter = **new** IndexWriter(d,conf);

//要把创建索引的文本数据放入Document的字段中；

//放入doc1

Document document1 = **new** Document(); //添加行

document1.add(**new** TextField("id","1",Store.***YES***));//添加列

document1.add(**new** TextField("title","doc1",Store.***YES***));

document1.add(**new** TextField("content",*doc1*,Store.***YES***));

Document document2 = **new** Document();

document2.add(**new** TextField("id","2",Store.***YES***));

document2.add(**new** TextField("title","doc2",Store.***YES***));

document2.add(**new** TextField("content",*doc2*,Store.***YES***));

Document document3 = **new** Document();

document3.add(**new** TextField("id","3",Store.***YES***));

document3.add(**new** TextField("title","doc3",Store.***YES***));

document3.add(**new** TextField("content",*doc3*,Store.***YES***));

//通过indexwriter将document写入

indexwriter.addDocument(document1); //添加到缓冲区

indexwriter.addDocument(document2);

indexwriter.addDocument(document3);

indexwriter.commit();

indexwriter.close();

}

/\*\*

\* 搜索索引

\* 1)创建IndexSearcher

\* 2)创建Query对象

\* 3)使用IndexSearcher将搜索条件传入入Query进行搜索

\* 4)从结果获取documentId，在通过它获取document

\* 5)把documen转换成我们想要的对象进行返回

\* **@return**

\* **@throws** Exception

\*/

**public** **static** **void** testSearch() **throws** Exception {

Directory d = FSDirectory.*open*(Paths.*get*(***PATH***)); //索引目录

IndexReader reader = DirectoryReader.*open*(d); //索引读入器

IndexSearcher searcher = **new** IndexSearcher(reader); //索引搜索器

String parseStr = "content:hello";

Analyzer analyzer = **new** SimpleAnalyzer(); //用它对搜索条件进行分词

String defaultField = "content"; //默认搜索字段，如果搜索字符串中没有指定，就使用默认的

QueryParser parser = **new** QueryParser(defaultField,analyzer);

Query query = parser.parse(parseStr);

TopDocs topDocs = searcher.search(query, 100); //查询top n个数据

System.***out***.println("总长命中数: "+topDocs.totalHits);

ScoreDoc[] scoreDocs = topDocs.scoreDocs; //获取命中记录数组

**for**(ScoreDoc scoreDoc: scoreDocs) {

**int** docId = scoreDoc.doc; //f获取文档编号

//通过编号获取我们想要的对象进行返回

Document document = searcher.doc(docId);

System.***out***.println("id---->"+ document.get("id"));

System.***out***.println("title---->"+ document.get("title"));

System.***out***.println("content---->"+ document.get("content"));

}

}

算法没有完成..

【学习内容真实可靠，给自己的评分和完成度客观真实，不谦虚，不夸大】