# 配置MAVEN

在setting文件中添加私服：

<repository>

<id>releases</id>

<name>Releases</name>

<url>http://61.154.164.33:8907/nexus/content/repositories/releases/</url>

<snapshots>

<enabled>false</enabled>

</snapshots>

<releases>

<enabled>true</enabled>

<updatePolicy>always</updatePolicy>

</releases>

</repository>

在pom文件中添加如下依赖:

<dependency>

<groupId>org.jsearch</groupId>

<artifactId>jsearch</artifactId>

<version>1.2.1-beta</version>

</dependency>

# 注释说明

**Group**

**添加位置:** 类

**作用:** 确定where后面条件项的分组与其它分组的运算关系(AND或OR).

**属性说明:**

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 是否必须 | 默认值 | 说明 |
| groupId | 是 | 无 | 分组ID |
| operate | 是 | 无 | 运算符,具体参见: org.jsearch.LogicalOperateType |

**Criterion**

**添加位置:** 类的属性字段

**作用:** 确定where后面条件项的信息,如列明,运算符,所属分组等.

**属性说明:**

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名 | 是否必须 | 默认值 | 说明 |
| columnName | 是 |  | 数据库列名 |
| groupId | 否 | Constants.DEFAULT\_GROUP\_ID | 分组ID |
| operate | 否 | OperateType.EQ | 运算符,具体参见: org.jsearch.OperateType |
| expression | 否 | Constants.DEFAULT\_EXPRESSION\_LIKE | like查询的表达式 |

# 示例

**数据库表:**

CREATE TABLE `demo` (

`field1` varchar(512) NOT NULL,

`field2` varchar(512) DEFAULT NULL,

`field3` varchar(512) DEFAULT NULL,

`field5` int(11) DEFAULT NULL,

`field6` int(11) DEFAULT NULL,

PRIMARY KEY (`field1`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

**SearchBean.java（注释定义）**

@Group(groupId = { "0", "1" }, operate = { LogicalOperateType.OR, LogicalOperateType.OR })

public class SearchBean {

@Criterion(groupId = "0", columnName = "p.teat", operate = OperateType.LIKE, expression = "${P}%")

private String field1;

@Criterion(groupId = "0", operate = OperateType.IN)

private String[] field2 = { "0", "1", "2" };

@Criterion(columnName = "p.cloumn", groupId = "1", operate = OperateType.LIKE, expression = "${P}%")

private String field3;

@Criterion(groupId = "1", operate = OperateType.BETWEEN)

private Integer[] field5 = { 1, 2 };

@Criterion(groupId = "1", operate = OperateType.LT)

private Integer field6 = 111;

public String getField1() {

return this.field1;

}

public void setField1(String field1) {

this.field1 = field1;

}

public String[] getField2() {

return this.field2;

}

public void setField2(String[] field2) {

this.field2 = field2;

}

public String getField3() {

return this.field3;

}

public void setField3(String field3) {

this.field3 = field3;

}

public Integer[] getField5() {

return this.field5;

}

public void setField5(Integer[] field5) {

this.field5 = field5;

}

public Integer getField6() {

return this.field6;

}

public void setField6(Integer field6) {

this.field6 = field6;

}

}

**SearchTest.java（使用代码）**

public class JSearchTest {

@Test

public void test1(){

JSearch jsearch=JSearch.getInstance("select \* from demo where ${0} and ${1}", new SearchBean());

System.out.println(jsearch.getExpression()); **//输出: select \* from demo where (field2 in (:FIELD2\_0,:FIELD2\_1,:FIELD2\_2)) and (field5 between :FIELD5\_0 and :FIELD5\_1 or field6 < :FIELD6)**

System.out.println(jsearch.getParamValues()); **//输出: {FIELD6=111, FIELD5\_0=1, FIELD5\_1=2, FIELD2\_0=0, FIELD2\_1=1, FIELD2\_2=2}**

}

}