Atitit 常见集合的操作 dsl表达式 选择器 多属性条件检索 ，排序等

目录

[1. Ognl 集合操作 1](#_Toc24055)

[2. /bookmarksHtmlEverythingIndexPrj/src/ListMapWhereSearch.java 1](#_Toc11377)

[2.1. //查找所有遍历 2](#_Toc7090)

[2.2. //条件where 过滤 2](#_Toc21521)

[2.3. //多条件组合过滤 2](#_Toc5634)

[2.4. //first 过滤 2](#_Toc22197)

[2.5. //过滤(filtering), 获取到集合的第一个元素,collection.{^ expression} 3](#_Toc27129)

[2.6. //过滤(filtering), 获取到集合的最后一个元素,collection.{$ expression} 3](#_Toc13631)

[2.7. In vs not in 3](#_Toc29866)

[3. Groovy集合过滤操作 4](#_Toc356)

# Ognl 集合操作

# /bookmarksHtmlEverythingIndexPrj/src/ListMapWhereSearch.java

**public** **class** ListMapWhereSearch {

**public** **static** **void** main(String[] args) **throws** OgnlException {

List<Map> list=**new** ArrayList(){{

add(**new** LinkedHashMap() {{

put("age",12);put("name","jcc");

}});

add(**new** LinkedHashMap() {{

put("age",15);put("name","aaa");

}});

}};

OgnlContext context = **new** OgnlContext();

/\*OGNL 过滤集合

\* 表达式：collection.{? expression}\*/

## //查找所有遍历

// System.out.println( Ognl.getValue("#this", list)); //[{age=12, name=jcc}, {age=15, name=aaa}]

// System.out.println( Ognl.getValue("#root", list)); //ok as same to #this

## //条件where 过滤

//Object filteredCollection = Ognl.getValue("#root.{? #this.age > 13}", list); // [{age=15, name=aaa}]

// System.out.println(filteredCollection);

## //多条件组合过滤

System.***out***.println( Ognl.*getValue*("#root.{? #this.age > 13 && #this.age <15}", list));

System.***out***.println( Ognl.*getValue*("#root.{? #this.age > 13 and #this.age <19}", list));

System.***out***.println( Ognl.*getValue*("#root.{? #this.age > 13 & #this.age <19}", list));

## //first 过滤

// System.out.println( Ognl.getValue("[0]", list)); //{age=12, name=jcc}

// System.out.println( Ognl.getValue("[1].age", list)); // ok ret 15

## //过滤(filtering), 获取到集合的第一个元素,collection.{^ expression}

System.out.println(Ognl.getValue("#persons.{^ #this.name.length() > 4}[0].name",context,context.getRoot()));

## //过滤(filtering), 获取到集合的最后一个元素,collection.{$ expression}

System.out.println(Ognl.getValue("#persons.{$ #this.name.length() > 4}[0].name",context,context.getRoot()));

---------------------

## In vs not in

对于集合类型，OGNL表达式可以使用in和not in两个元素符号。其中，in表达式用来判断某个元素是否在指定的集合对象中；not in判断某个元素是否不在指定的集合对象中，如代码8.3所示。

<s:if test="'aaa' in {'aaa','bbb'}">

aaa 在 集合{'aaa','bbb'}中；

</s:if>

<s:else>

aaa 不在 集合{'aaa','bbb'}中；

</s:else>

<br/>

<s:if test="#request.req not in #list">

# Groovy集合过滤操作

是的，在Groovy语言中，我们对集合元素的过滤一般都使用"grep"方法，这样的代码更加具有Groovy风格一些。具有同样一些功能的方法还有"find(Closure closure)"，"findAll(Closure closure)"。

---------------------

List<Map> list=**new** ArrayList(){{

add(**new** LinkedHashMap() {{

put("age",12);put("name","jcc");

}});

add(**new** LinkedHashMap() {{

put("age",15);put("name","aaa");

}});

add(**new** LinkedHashMap() {{

put("age",18);put("name","aaa");

}});

}};

**def** result = list.grep{ it.age>13 && it.age<20 }

println result

(9+条消息)Groovy探索 使用集合方法，写出更加Groovy风格的代码 - 软件的信雅达 - CSDN博客.mhtml

(9+条消息)groovy中List集合的使用 - 技术成就梦想 - CSDN博客