Guava学习之Multisets

今天谈谈Guava类库中的Multisets数据结构，虽然它不怎么经常用，但是还是有必要对它进行探讨。我们知道Java类库中的Set不能存放相同的元素，且里面的元素是无顺序的；而List是能存放相同的元素，而且是有顺序的。而今天要谈的Multisets是能存放相同的元素，但是元素之间的顺序是无序的。从这里也可以看出，Multisets肯定不是实现Java中Set接口的，因为Set接口是不能存放相同的元素！Java中的Set 里面的元素有点像 :[A, C, B]，而 Multiset 会是这样 : [A × 2, C × 3, B × 5]，这个是有区别的。
在以前，如果我们需要统计一篇文章中各个单词出现的次数，我们可能用下面的方法来实现：

```
public void wordCounts(List words) {
 Map<String, Integer> counts = new HashMap<String, Integer>();
 for (String word : words) {
  Integer count = counts.get(word);
  if (count == null) {
   counts.put(word, 1);
  } else {
   counts.put(word, count + 1);
  }
 }
}
```

如果我们需要得到某个单词（比如good）的出现次数，我们可能这么写：

```
int goodCount = counts.get("good");
```

很麻烦对吗？而且Map中的get(E key)的含义都不那么明显。那如果我们用Multisets来看看：

```
import com.google.common.collect.HashMultiset;
import com.google.common.collect.Multiset;

Multiset countMultiset = HashMultiset.create();
countMultiset.addAll(words);
```

很简单吧？甚至连循环都不需要。那么我们怎么得到某个单词（比如good）的出现次数？很简单：

int goodCount= countMultiset .count(“good”);

在Multiset中提供了一个count(Object element)方法，得到某个对象在Multiset中出现的次数，这个显然比在Map里面调get的可读性好多了，不是吗？

> 需要注意：Multiset提供setCount(E, int)方法，可以修改元素E在Multiset中的次数，但是不能把元素出现的次数修改为负数和大于Integer.MAX_VALUE的值。否则将会抛出异常

```
countMultiset .setCount("good", Integer.MAX_VALUE + 1);
或
countMultiset .setCount("good", -1);
```

将会抛出以下异常

```
Exception in thread "main" java.lang.IllegalArgumentException: count cannot be negative: -214
7483648（-1）
 at com.google.common.base.Preconditions.checkArgument(Preconditions.java:119)
 at com.google.common.collect.Multisets.checkNonnegative(Multisets.java:1061)
 at com.google.common.collect.AbstractMapBasedMultiset.setCount(AbstractMapBasedMultis
et.java:265)
 at com.google.common.collect.HashMultiset.setCount(HashMultiset.java:34)
 at com.wyp.test.testFiles(test.java:150)
 at com.wyp.test.main(test.java:156)
```

因为在Multiset中setCount先会判断所设置数据的状态

```
 static void checkNonnegative(int count, String name) {
 checkArgument(count >= 0, "%s cannot be negative: %s", name, count);
 }
```

可以看出，count是int类型的，最大的值为Integer.MAX_VALUE，在其基础只上加上1将会变成负数，所以不行。

Multiset也不是Map<E, Integer>类型的结构, 我们可以用Map<E, Integer>来实现Multiset，但是利用Map<E, Integer>实现Multiset和Google guava的实现还是有很大的区别的，主要表现如下：

1. Multiset中的元素出现的次数只能为正数，前面说了原因。如果E的出现次数为0，那么E将不出现在multiset中，是不能在elementSet()和entrySet()的视图中；
2. multiset.size()返回这个集合的大小，相当于在multiset中元素的出现的总数。如果想得到multiset中不同元素出现的总数，可以利用elementSet().size()来实现；
3. multiset.iterator()可以遍历multiset中的所有元素，所以iteration遍历的次数就等于multiset.size()；
4. Multiset支持添加、删除元素，设置元素出现的次数；setCount(elem, 0)相当于移除elem的所有元素；
5. multiset.count(elem)方法中的elem如果没有出现在Multiset中，那么它的返回值永远都是0。

常用的实现了Multiset 接口的类有：

1. HashMultiset: 元素存放于 HashMap
2. LinkedHashMultiset: 元素存放于 LinkedHashMap，即元素的排列顺序由第一次放入的顺序决定
3. TreeMultiset:元素被排序存放于TreeMap
4. EnumMultiset: 元素必须是 enum 类型
5. ImmutableMultiset: 不可修改的 Mutiset