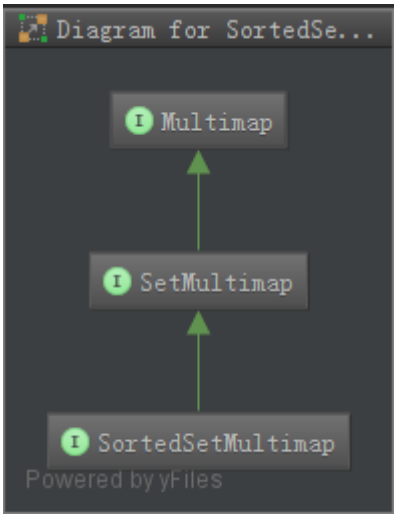
Gauva 学习之 SortedSetMultimap



Gauva 学习之 SortedSetMultimap

SortedSetMultimap 是一个接口，它的继承关系如上所示。继承了 SortedSetMultimap 接口的类中 key 所对应的 value 是有序的。因为 SortedSetMultimap 的子类中 key 所对应的 value 是有序的，所以 SortedSetMultimap 重写了 SetMultimap 中的以下四个方法：

```
01 @Override
02 SortedSet<V> get(@Nullable K key);
03
04 @Override
05 SortedSet<V> removeAll(@Nullable Object key);
06
07 @Override
08 SortedSet<V> replaceValues(K key, Iterable<? extends V> values);
09
10 @Override Map<K, Collection<V>> asMap();
```

前三个函数都是将 SetMultimap 接口中相应的函数返回类型（Set）修改成了 SortedSet 类型。而 SortedSet 接口是继承自 Set 的，只不过 SortedSet 的子类可以保证其中的 value 是有序的，而 SortedSetMultimap 接口就是需要保证 key 中所对应的 value 是有序的，所以可以使用 SortedSet。

但是既然 asMap()也是重载了 SetMultimap 中相应的函数，为什么它就返回 Map<k , Collection>，而不返回 Map<k , SortedSet>呢？这是因为 asMap()函数能够保

证其返回的类型为一定是 SortedSet 的子类。看看代码就知道了，TreeMultimap 类是 SetMultimap 的实现类，它的 asMap()实现如下：

```
1 @Override
2 @GwtIncompatible("NavigableMap")
3 public NavigableMap<K, Collection<V>> asMap() {
4         return (NavigableMap<K, Collection<V>>) super.asMap();
5 }
```

而 super.asMap();最上层类（AbstractMultimap 接口）的实现如下：

```
1 private transient Map<K, Collection<V>> asMap;
2
3 @Override
4 public Map<K, Collection<V>> asMap() {
5   Map<K, Collection<V>> result = asMap;
6   return (result == null) ? asMap = createAsMap() : result;
7 }
```

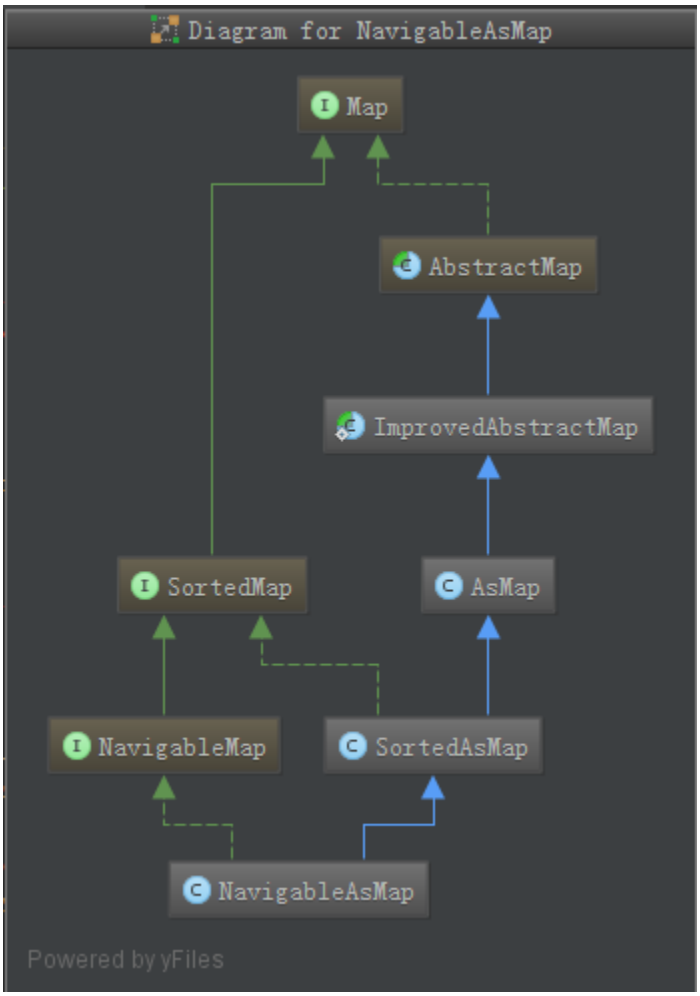
而 createAsMap()函数在 AbstractMultimap 中的定义如下：

```
1 abstract Map<K, Collection<V>> createAsMap();
```

这是一个抽象的函数，需要其子类（这里是指 TreeMultimap）实现，那么 TreeMultimap 中对 createAsMap()函数是怎么实现的呢？看下它的实现代码：

```
1 @Override
2 @GwtIncompatible("NavigableMap")
3 NavigableMap<K, Collection<V>> createAsMap() {
4   return new NavigableAsMap(backingMap());
5 }
```

我们不需要知道 NavigableAsMap 类的实现方式，只需要知道 NavigableAsMap 类的继承关系，如下所示：



Gauva 学习之 SortedSetMultimap

从上面 NavigableAsMap 类的继承关系图可以看出，NavigableAsMap 类是 SortedMsp 的实现类，它能保证的 Map 中的 key 和 value 都是有序的。所以，SortedSetMultimap 中 asMap()函数返回类型为 Map<k , Collection>。

SortedSetMultimap 接口中还定义了 valueComparator 函数，其原型如下：

```
1 Comparator<? super V> valueComparator();
```

valueComparator 函数返回对 multimap 中 value 排序的对象，如果返回为 null，则说明 multimap 中 value 是按照自然排序的。

云帆教育大数据学院

关注云帆教育微信公众号 yfteach，第一时间获取公开课信息。