问题

（1）LinkedHashSet的底层使用什么存储元素?

（2）LinkedHashSet与HashSet有什么不同?

（3）LinkedHashSet是有序的吗?

（4）LinkedHashSet支持按元素访问顺序排序吗?

简介

上一节我们说HashSet中的元素是无序的，那么有没有什么办法保证Set中的元素是有序的呢?

答案是当然可以。

我们今天的主角LinkedHashSet就有这个功能，它是怎么实现有序的呢？让我们来一起学习吧。

源码分析

LinkedHashSet继承自HashSet，让我们直接上源码来看看它们有什么不同。

```
1.  package java.util;
2.
3.  // LinkedHashSet继承自HashSet
4.  public class LinkedHashSet<E>
5.      extends HashSet<E>
6.      implements Set<E>, Cloneable, java.io.Serializable {
7.
8.      private static final long serialVersionUID = -2851667679971038690L;
9.
10.     // 传入容量和装载因子
11.     public LinkedHashSet(int initialCapacity, float loadFactor) {
12.         super(initialCapacity, loadFactor, true);
13.     }
14.
15.     // 只传入容量，装载因子默认为0.75
16.     public LinkedHashSet(int initialCapacity) {
17.         super(initialCapacity, .75f, true);
18.     }
19.
20.     // 使用默认容量16，默认装载因子0.75
21.     public LinkedHashSet() {
22.         super(16, .75f, true);
23.     }
24.
25.     // 将集合c中的所有元素添加到LinkedHashSet中
26.     // 好奇怪，这里计算容量的方式又变了
27.     // HashSet中使用的是Math.max((int) (c.size()/.75f) + 1, 16)
28.     // 这一点有点不得其解，是作者偷懒？
29.     public LinkedHashSet(Collection<? extends E> c) {
30.         super(Math.max(2*c.size(), 11), .75f, true);
31.         addAll(c);
32.     }
33.
34.     // 可分割的迭代器，主要用于多线程并行迭代处理时使用
35.     @Override
36.     public Spliterator<E> spliterator() {
37.         return Spliterators.spliterator(this, Spliterator.DISTINCT | Spliterator.ORDERED);
38.     }
39. }
```

```
40.
41.
```

完了，结束了，就这么多，这是全部源码了，真的。

可以看到，LinkedHashSet中一共提供了5个方法，其中4个是构造方法，还有一个是迭代器。

4个构造方法都是调用父类的 `super(initialCapacity, loadFactor, true);` 这个方法。

这个方法长什么样呢?

还记得我们上一节说过一个不是public的构造方法吗？就是它。

```
1.      // HashSet的构造方法
2.      HashSet(int initialCapacity, float loadFactor, boolean dummy) {
3.          map = new LinkedHashMap<>(initialCapacity, loadFactor);
4.      }
5.
```

如上所示，这个构造方法里面使用了LinkedHashMap来初始化HashSet中的map。

现在这个逻辑应该很清晰了，LinkedHashSet继承自HashSet，它的添加、删除、查询等方法都是直接用的HashSet的，唯一的不同就是它使用LinkedHashMap存储元素。

那么，开篇那几个问题是否能回答了呢?

总结

（1）LinkedHashSet的底层使用LinkedHashMap存储元素。

（2）LinkedHashSet是有序的，它是按照插入的顺序排序的。

彩蛋

通过上面的学习，我们知道LinkedHashSet底层使用LinkedHashMap存储元素，而LinkedHashMap是支持按元素访问顺序遍历元素的，也就是可以用来实现LRU的，还记得吗？传送门【死磕 java集合之LinkedHashMap源码分析】

那么，LinkedHashSet支持按元素访问顺序排序吗?

让我们一起来分析下。

首先，LinkedHashSet所有的构造方法都是调用HashSet的同一个构造方法，如下:

```
1.      // HashSet的构造方法
2.      HashSet(int initialCapacity, float loadFactor, boolean dummy) {
3.          map = new LinkedHashMap<>(initialCapacity, loadFactor);
4.      }
5.
```

然后，通过调用LinkedHashMap的构造方法初始化map，如下所示:

```
1.      public LinkedHashMap(int initialCapacity, float loadFactor) {
2.          super(initialCapacity, loadFactor);
3.          accessOrder = false;
4.      }
5.
```

可以看到，这里把accessOrder写死为false了。

所以，LinkedHashSet是不支持按访问顺序对元素排序的，只能按插入顺序排序。