Java集合学习手册（5）：Java LinkedHashSet

一、概述

首先我们需要知道的是它是一个Set的实现，所以它其中存的肯定不是键值对，而是值。此实现与HashSet的不同之处在于，LinkedHashSet维护着一个运行于所有条目的双重链接列表。此链接列表定义了迭代顺序，该迭代顺序可为插入顺序或是访问顺序。

看到上面的介绍，是不是感觉其与HashMap和LinkedHashMap的关系很像?

注意，此实现不是同步的。如果多个线程同时访问链接的哈希Set，而其中至少一个线程修改了该Set，则它必须保持外部同步。

在【Java学习手册：LinkedHashMap】中，通过例子演示了HashMap和LinkedHashMap的区别。举一反三，我们现在学习的LinkedHashSet与之前的很相同，只不过之前存的是键值对，而现在存的只有值。

LinkedHashSet是可以按照插入顺序或者访问顺序进行迭代。

二、LinkedHashSet的实现

对于LinkedHashSet而言，它继承与HashSet、又基于LinkedHashMap来实现的。LinkedHashSet底层使用LinkedHashMap来保存所有元素，它继承与HashSet，其所有的方法操作上又与HashSet相同，因此LinkedHashSet的实现上非常简单，只提供了四个构造方法，并通过传递一个标识参数，调用父类的构造器，底层构造一个LinkedHashMap来实现，在相关操作上与父类HashSet的操作相同，直接调用父类HashSet的方法即可。LinkedHashSet的源代码如下：

```
public class LinkedHashSet<E>
    extends HashSet<E>
    implements Set<E>, Cloneable, java.io.Serializable {

    private static final long serialVersionUID = -2851667679971038690L;

    /**
     * 构造一个带有指定初始容量和加载因子的新空链接哈希set。
```

```
     *
     * 底层会调用父类的构造方法，构造一个有指定初始容量和加载因子的LinkedHashMap实例。
     * @param initialCapacity 初始容量。
     * @param loadFactor 加载因子。
     */
    public LinkedHashSet(int initialCapacity, float loadFactor) {
        super(initialCapacity, loadFactor, true);
    }

    /**
     * 构造一个带指定初始容量和默认加载因子0.75的新空链接哈希set。
     *
     * 底层会调用父类的构造方法，构造一个带指定初始容量和默认加载因子0.75的LinkedHashMap
实例。
     * @param initialCapacity 初始容量。
     */
    public LinkedHashSet(int initialCapacity) {
        super(initialCapacity, .75f, true);
    }

    /**
     * 构造一个带默认初始容量16和加载因子0.75的新空链接哈希set。
     *
     * 底层会调用父类的构造方法，构造一个带默认初始容量16和加载因子0.75的LinkedHashMap实
例。
     */
    public LinkedHashSet() {
        super(16, .75f, true);
    }

    /**
     * 构造一个与指定collection中的元素相同的新链接哈希set。
     *
     * 底层会调用父类的构造方法，构造一个足以包含指定collection
     * 中所有元素的初始容量和加载因子为0.75的LinkedHashMap实例。
     * @param c 其中的元素将存放在此set中的collection。
     */
    public LinkedHashSet(Collection<? extends E> c) {
        super(Math.max(2*c.size(), 11), .75f, true);
        addAll(c);
    }
}
```

以上几乎就是LinkedHashSet的全部代码了，那么读者可能就会怀疑了，不是说LinkedHashSet是基于LinkedHashMap实现的吗？那我为什么在源码中甚至都没有看到出现过LinkedHashMap。不要着急，我们可以看到在LinkedHashSet的构造方法中，其调用了父类的构造方法。我们可以进去看一下：

```
/**
     * 以指定的initialCapacity和loadFactor构造一个新的空链接哈希集合。
     * 此构造函数为包访问权限，不对外公开，实际只是是对LinkedHashSet的支持。
     *
     * 实际底层会以指定的参数构造一个空LinkedHashMap实例来实现。
     * @param initialCapacity 初始容量。
     * @param loadFactor 加载因子。
     * @param dummy 标记。
     */
HashSet(int initialCapacity, float loadFactor, boolean dummy) {
    map = new LinkedHashMap<E,Object>(initialCapacity, loadFactor);
}
```

在父类HashSet中，专为LinkedHashSet提供的构造方法如下，该方法为包访问权限，并未对外公开。
由上述源代码可见，LinkedHashSet通过继承HashSet，底层使用LinkedHashMap，以很简单明了的方式来实现了其自身的所有功能。

三、总结

以上就是关于LinkedHashSet的内容，我们只是从概述上以及构造方法这几个方面介绍了，并不是我们不想去深入其读取或者写入方法，而是其本身没有实现，只是继承于父类HashSet的方法。

所以我们需要注意的点是：

- LinkedHashSet是Set的一个具体实现，其维护着一个运行于所有条目的双重链接列表。此链接列表定义了迭代顺序，该迭代顺序可为插入顺序或是访问顺序。
- LinkedHashSet继承与HashSet，并且其内部是通过LinkedHashMap来实现的。有点类似于我们之前说的LinkedHashMap其内部是基于Hashmap实现一样，不过还是有一点点区别的（具体的区别大家可以自己去思考一下）。
- 如果我们需要迭代的顺序为插入顺序或者访问顺序，那么LinkedHashSet是需要你首先考虑的。