Java中的Set体系集合

1. **Set接口**：存在于**java.util包**中，继承于**Collection接口**。**Set接口中的方法与父接口Collection接口方法一样。Set子接口**（与List接口并列）：**元素不可以重复，是无序的**。重复的对象存不进去，保证了元素的**唯一性（不可重复性）**；对象的顺序**并不是按存进去的顺序排列**的，即所谓的**无序的**。
2. **Set接口有两个主要的实现类：HashSet 和 TreeSet。都是不同步的，且都是无序的，但是TreeSet是排序的，而HashSet没有排序。都保证了元素的唯一性，HashSet是利用对象元素的hashCode和equals方法，而TreeSet是利用对象元素类实现Comparable接口中的compareTo方法或者比较器中的compare方法保证唯一性的。TreeSet集合中的元素唯一性和排序都是利用对象元素类实现Comparable接口中的compareTo方法或者比较器中的compare方法。**
3. **HashSet集合**的内部数据结构是**哈希表**，是**不同步的**。元素存储到**HashSet集合**中，必须覆盖**hashCode方法和equals方法**，这两个方法就是**判断对象元素是否相同的依据，从而保证元素的唯一性**。首先调用元素的hashCode方法，来确定**位置**，如果存在两个元素的哈希值相同，再通过调用元素的equals方法来确定这两个元素的内容是否相同。**区别**：ArrayList、LinkedList中的 contains、remove、indexOf方法等都是先找到要**操作的元素**，寻找的依据利用**equals方法，没有利用hashCode方法**。哈希值相同，称为**哈希冲突**。
4. **TreeSet集合的内部数据结构是二叉树结构，是不同步的，可以对集合中的元素按照指定的规则进行排序。（注意：HashSet集合中的元素并没有排序，是随意排列的）排序的规则有两种：**
5. **自然排序法：就是让元素自身具备比较功能，即让元素对象的类实现Comparable接口，重写compareTo（）方法即可。**
6. **比较器的方法**：**就是让TreeSet集合自身具有排序功能**，即创建一个**比较器**实现Comparator接口，覆写**compare方法**，然后new出一个比较器对象作为参数传进**TreeSet集合**即可。



**两种方法的区别**：**自然排序法具有局限性**，如果某些类是已经封装好的，但是其没有实现Comparable接口且没有实现compareTo方法（即不具备自然排序方式），或者实现了但不是所需要的排序方式（即自然排序方式不满足所需的排序要求）怎么办，此时，只能通过**添加比较器的方法**进行解决。**比较器可以根据不同的需求创建多个，然后选择需要的作为参数传进TreeSet集合即可。实际开发中，比较器方法比较常用。**

1. **Comparable接口**：存在于**java.lang包**中，实现此接口可以使实现类具有排序功能，这种排序称为**自然排序**。如**String类**。**接口中只有一个方法：compareTo(T o)。**



实现了Comparable接口中的类有哪些：**String、Date、Calendar类**。

1. **Comparator接口**：存在于java.util包中，是一个**功能接口**。创建一个比较器必须实现此接口，且覆写其中的**compare方法：返回int值，0，正数，负数**。



1. **HashSet集合**如何保证集合中的**元素的唯一性**？

**hashCode方法用来获取哈希值，equals方法用来判断对象内容是否相同。**

答： 通过**hashCode和equals这两个方法**来完成元素的唯一性。

1. 如果**对象元素的hashCode值**不同，则不需要判断equals方法，就直接存储到哈希表中 ；
2. 如果**对象元素hashCode值**相同，则需要利用**equals方法**判断对象内容是否相同，如果返回true，则视为相同的对象元素，不存储；如果返回false，则视为不同的对象元素，就会存储到**哈希表**中。
3. **TreeSet集合**如何保证集合中的**元素的唯一性**？

**利用对象元素类实现Comparable接口中的compareTo方法或者比较器中的compare方法。**

**注意**：如果对象元素需要存储到HashSet集合中，必须覆盖hashCode和equals方法；如果多个元素存储到**Treeset集合**中，那么TreeSet集合必须传入比较器或者元素对象类必须实现Comparable接口中的**compareTo方法**。

1. **TreeSet集合**如何保证集合中的**元素是排序的**？

**利用对象元素类实现Comparable接口中的compareTo方法或者比较器中的compare方法。**

1. **HashSet集合**：
2. **构造方法：**
3. **空参数**



1. 参数是个集合：



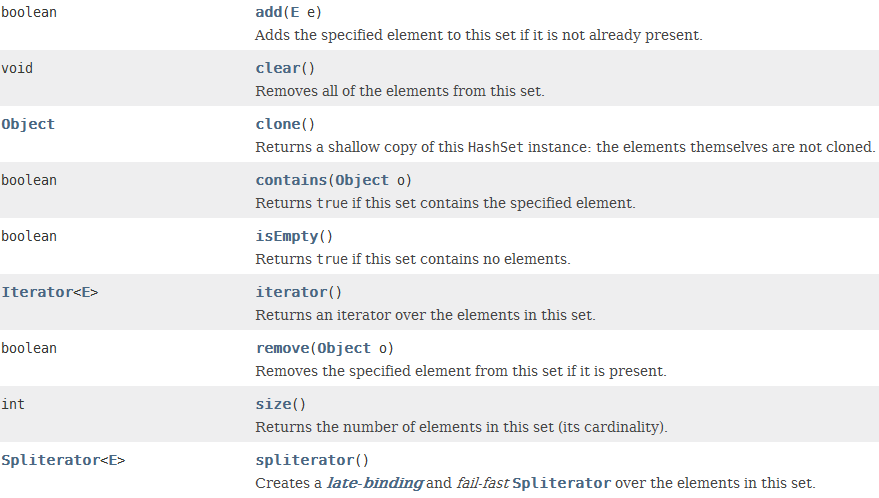
1. 参数是设定大小：



1. 参数为：



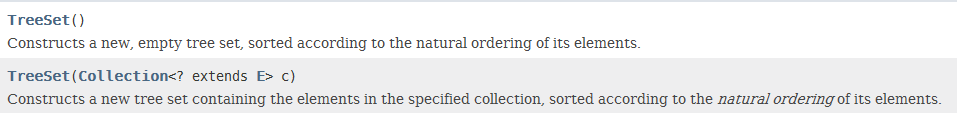
1. **特有方法：比较少**



1. **LinkedHashSet集合**：存在于java.util包中，是**HashSet集合**的一个子类。由于HashSet集合中的元素存和取都是无序的，且不像TreeSet集合会排序，因此为了克服这个缺点，HashSet有个子类**LinkedHashSet**，可以实现集合元素的**有序性和唯一性**。LinkedHashSet集合内部使用了**哈希表和链表相结合的数据结构**，既实现了**元素的唯一性**，又实现了**元素的有序存取**。

**LinkedHashSet集合和HashSet集合构造方法和成员方法一样，不再赘述。**

1. **TreeSet集合**：
2. **构造方法：**
3. **空参数**和**集合**为参数



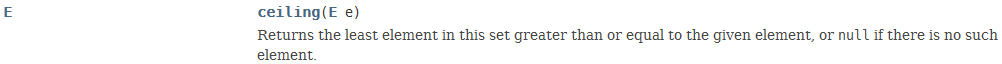
1. 添加比较器：**使TreeSet集合本身具有比较功能**。



1. SortedSet为参数：



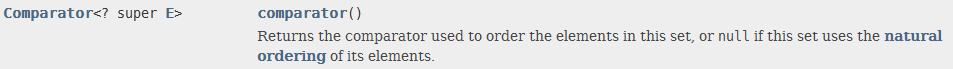
1. **特有方法：**
2. **ceiling（E e）： 返回一个集合中比给定元素大的或相等的元素或者null。**



1. **floor（E e）**: **返回一个集合中比给定元素小的或相等的元素或者null。**



1. comparator（）： 返回集合的比较器或者返回null（如果使用自然排序法）。



1. **Iterator方法**：TreeSet集合的迭代器。

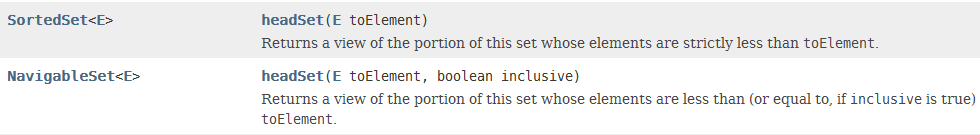


1. first方法和last方法

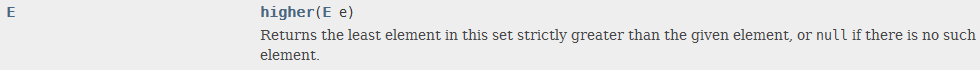


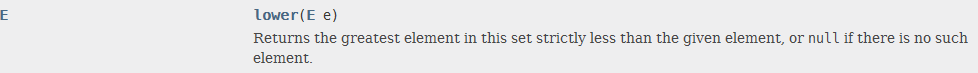


1. headset方法：

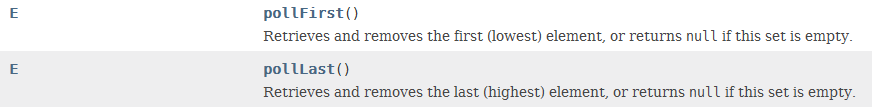


1. higher和lower

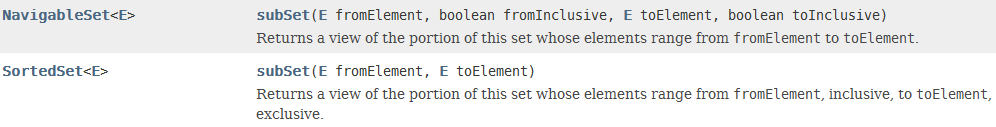




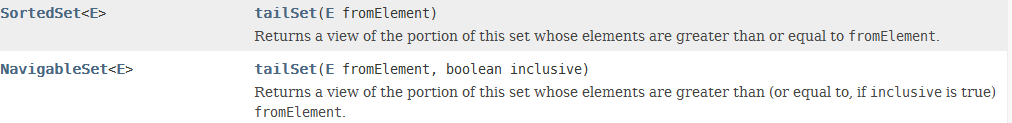
1. pollFirst和pollLast



1. subset



1. tailSet



1. **TreeSet**内部的排序方法效率比较高，采用的是二叉树的结构，查找方式采用二分查找的方式。故称为TreeSet。二叉树： 每个元素（节点）最多记录着三个引用，父引用，左引用，右引用。
2. **ArrayList 等集合**中的contains、remove等方法，在判断是否含有或要删除某个元素的时候，首先要找到这个元素（若找不到，方法直接返回false，这就是大部分方法返回boolean变量的原因），查找就是利用**equals方法**，此时**equals方法**的重写就关系到其他方法的使用功能。然而**HashSet**中的contains、remove等方法，寻找这个元素，需要利用**hashCode（）和equals（）**两个方法。所以对hashCode和equals方法的重写很重要，涉及到很多其他的方法。