1.接口Set

public interface **Set<E>**extends [Collection](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Collection.html" \o "java.util 中的接口)<E>

Set集合不包含重复的元素，并且只能有一个null元素。

|  |  |
| --- | --- |
| boolean | **[add](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "add(E))**([E](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \o "Set 中的类型参数) e)            如果 set 中尚未存在指定的元素，则添加此元素（可选操作）。 |
| boolean | **[addAll](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "addAll(java.util.Collection))**([Collection](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Collection.html" \o "java.util 中的接口)<? extends [E](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \o "Set 中的类型参数)> c)            如果 set 中没有指定 collection 中的所有元素，则将其添加到此 set 中（可选操作）。 |
| void | **[clear](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "clear())**()            移除此 set 中的所有元素（可选操作）。 |
| boolean | **[contains](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "contains(java.lang.Object))**([Object](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/lang/Object.html" \o "java.lang 中的类) o)            如果 set 包含指定的元素，则返回 true。 |
| boolean | **[containsAll](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "containsAll(java.util.Collection))**([Collection](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Collection.html" \o "java.util 中的接口)<?> c)            如果此 set 包含指定 collection 的所有元素，则返回 true。 |
| boolean | **[equals](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "equals(java.lang.Object))**([Object](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/lang/Object.html" \o "java.lang 中的类) o)            比较指定对象与此 set 的相等性。 |
| int | **[hashCode](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "hashCode())**()            返回 set 的哈希码值。 |
| boolean | **[isEmpty](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "isEmpty())**()            如果 set 不包含元素，则返回 true。 |
| [Iterator](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Iterator.html" \o "java.util 中的接口)<[E](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \o "Set 中的类型参数)> | **[iterator](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "iterator())**()            返回在此 set 中的元素上进行迭代的迭代器。 |
| boolean | **[remove](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "remove(java.lang.Object))**([Object](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/lang/Object.html" \o "java.lang 中的类) o)            如果 set 中存在指定的元素，则将其移除（可选操作）。 |
| boolean | **[removeAll](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "removeAll(java.util.Collection))**([Collection](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Collection.html" \o "java.util 中的接口)<?> c)            移除 set 中那些包含在指定 collection 中的元素（可选操作）。 |
| boolean | **[retainAll](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "retainAll(java.util.Collection))**([Collection](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Collection.html" \o "java.util 中的接口)<?> c)            仅保留 set 中那些包含在指定 collection 中的元素（可选操作）。 |
| int | **[size](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "size())**()            返回 set 中的元素数（其容量）。 |
| [Object](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/lang/Object.html" \o "java.lang 中的类)[] | **[toArray](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "toArray())**()            返回一个包含 set 中所有元素的数组。 |
| |  | | --- | | <T> T[] | | **[toArray](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \l "toArray(T[]))**(T[] a)            返回一个包含此 set 中所有元素的数组；返回数组的运行时类型是指定数组的类型。 |

2.类HashSet

  public class **HashSet<E>** extends [AbstractSet](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/AbstractSet.html" \o "java.util 中的类)<E>

            implements [Set](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Set.html" \o "java.util 中的接口)<E>, [Cloneable](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/lang/Cloneable.html" \o "java.lang 中的接口), [Serializable](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/io/Serializable.html" \o "java.io 中的接口)

此类实现了Set接口，由哈希表(实际上是一个HashMap实例)支持。它不保证set的迭代顺序；此类允许使用null元素。

如果多个线程同时访问一个哈希 set，而其中至少一个线程修改了该 set，那么它*必须* 保持外部同步。这通常是通过对自然封装该 set 的对象执行同步操作来完成的。如果不存在这样的对象，则应该使用 [Collections.synchronizedSet](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Collections.html" \l "synchronizedSet(java.util.Set)) 方法来“包装” set。最好在创建时完成这一操作，以防止对该 set 进行意外的不同步访问：

   Set s = Collections.synchronizedSet(new HashSet(...));

|  |  |
| --- | --- |
| boolean | **[add](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/HashSet.html" \l "add(E))**([E](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/HashSet.html" \o "HashSet 中的类型参数) e)            如果此 set 中尚未包含指定元素，则添加指定元素。 |
| void | **[clear](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/HashSet.html" \l "clear())**()            从此 set 中移除所有元素。 |
| [Object](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/lang/Object.html" \o "java.lang 中的类) | **[clone](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/HashSet.html" \l "clone())**()            返回此 HashSet 实例的浅表副本：并没有复制这些元素本身。 |
| boolean | **[contains](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/HashSet.html" \l "contains(java.lang.Object))**([Object](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/lang/Object.html" \o "java.lang 中的类) o)            如果此 set 包含指定元素，则返回 true。 |
| boolean | **[isEmpty](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/HashSet.html" \l "isEmpty())**()            如果此 set 不包含任何元素，则返回 true。 |
| [Iterator](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Iterator.html" \o "java.util 中的接口)<[E](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/HashSet.html" \o "HashSet 中的类型参数)> | **[iterator](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/HashSet.html" \l "iterator())**()            返回对此 set 中元素进行迭代的迭代器。 |
| boolean | **[remove](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/HashSet.html" \l "remove(java.lang.Object))**([Object](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/lang/Object.html" \o "java.lang 中的类) o)            如果指定元素存在于此 set 中，则将其移除。 |
| int | **[size](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/HashSet.html" \l "size())**()            返回此 set 中的元素的数量（set 的容量）。 |

3.TreeSet

public class **TreeSet<E>** extends [AbstractSet](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/AbstractSet.html" \o "java.util 中的类)<E>

     implements [NavigableSet](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/NavigableSet.html" \o "java.util 中的接口)<E>, [Cloneable](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/lang/Cloneable.html" \o "java.lang 中的接口), [Serializable](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/io/Serializable.html" \o "java.io 中的接口)

     基于TreeMap实现，使用元素的自然顺序对元素进行排序，或者根据创建 set 时提供的 [Comparator](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Comparator.html) 进行排序，具体取决于使用的构造方法。此实现为基本操作（add、remove 和 contains）提供受保证的 log(n) 时间开销。

如果多个线程同时访问一个 TreeSet，而其中至少一个线程修改了该 set，那么它*必须* 外部同步。这一般是通过对自然封装该 set 的对象执行同步操作来完成的。如果不存在这样的对象，则应该使用 [Collections.synchronizedSortedSet](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/Collections.html" \l "synchronizedSortedSet(java.util.SortedSet)) 方法来“包装”该 set。此操作最好在创建时进行，以防止对 set 的意外非同步访问：

   SortedSet s = Collections.synchronizedSortedSet(new TreeSet(...));

快速失败机制：

     此类的 iterator 方法返回的迭代器是*快速失败* 的：在创建迭代器之后，如果从结构上对 set 进行修改，除非通过迭代器自身的 remove 方法，否则在其他任何时间以任何方式进行修改都将导致迭代器抛出 [ConcurrentModificationException](mk:@MSITStore:C:\\Users\\张志鹏\\Desktop\\JDK_API_1_6_zh_CN.CHM::/java/util/mk:java/util/ConcurrentModificationException.html)。因此，对于并发的修改，迭代器很快就完全失败，而不会冒着在将来不确定的时间发生不确定行为的风险。

注意，迭代器的快速失败行为无法得到保证，一般来说，存在不同步的并发修改时，不可能作出任何肯定的保证。快速失败迭代器尽最大努力抛出 ConcurrentModificationException。因此，编写依赖于此异常的程序的做法是错误的，正确做法是：迭代器的快速失败行为应该仅用于检测 bug。