| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/NavigableSet.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/util/NavigableMap.html)   [**NEXT CLASS**](http://docs.google.com/java/util/NoSuchElementException.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/NavigableSet.html)    [**NO FRAMES**](http://docs.google.com/NavigableSet.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#3dy6vkm) |

## **java.util**

Interface NavigableSet<E>

**Type Parameters:**E - the type of elements maintained by this set **All Superinterfaces:** [Collection](http://docs.google.com/java/util/Collection.html)<E>, [Iterable](http://docs.google.com/java/lang/Iterable.html)<E>, [Set](http://docs.google.com/java/util/Set.html)<E>, [SortedSet](http://docs.google.com/java/util/SortedSet.html)<E> **All Known Implementing Classes:** [ConcurrentSkipListSet](http://docs.google.com/java/util/concurrent/ConcurrentSkipListSet.html), [TreeSet](http://docs.google.com/java/util/TreeSet.html)

public interface **NavigableSet<E>**extends [SortedSet](http://docs.google.com/java/util/SortedSet.html)<E>

A [SortedSet](http://docs.google.com/java/util/SortedSet.html) extended with navigation methods reporting closest matches for given search targets. Methods lower, floor, ceiling, and higher return elements respectively less than, less than or equal, greater than or equal, and greater than a given element, returning null if there is no such element. A NavigableSet may be accessed and traversed in either ascending or descending order. The descendingSet method returns a view of the set with the senses of all relational and directional methods inverted. The performance of ascending operations and views is likely to be faster than that of descending ones. This interface additionally defines methods pollFirst and pollLast that return and remove the lowest and highest element, if one exists, else returning null. Methods subSet, headSet, and tailSet differ from the like-named SortedSet methods in accepting additional arguments describing whether lower and upper bounds are inclusive versus exclusive. Subsets of any NavigableSet must implement the NavigableSet interface.

The return values of navigation methods may be ambiguous in implementations that permit null elements. However, even in this case the result can be disambiguated by checking contains(null). To avoid such issues, implementations of this interface are encouraged to *not* permit insertion of null elements. (Note that sorted sets of [Comparable](http://docs.google.com/java/lang/Comparable.html) elements intrinsically do not permit null.)

Methods [subSet(E, E)](http://docs.google.com/java/util/NavigableSet.html#subSet(E,%20E)), [headSet(E)](http://docs.google.com/java/util/NavigableSet.html#headSet(E)), and [tailSet(E)](http://docs.google.com/java/util/NavigableSet.html#tailSet(E)) are specified to return SortedSet to allow existing implementations of SortedSet to be compatibly retrofitted to implement NavigableSet, but extensions and implementations of this interface are encouraged to override these methods to return NavigableSet.

This interface is a member of the  [Java Collections Framework](http://docs.google.com/technotes/guides/collections/index.html).

**Since:** 1.6

| **Method Summary** | |
| --- | --- |
| [E](http://docs.google.com/java/util/NavigableSet.html) | [**ceiling**](http://docs.google.com/java/util/NavigableSet.html#ceiling(E))([E](http://docs.google.com/java/util/NavigableSet.html) e)            Returns the least element in this set greater than or equal to the given element, or null if there is no such element. |
| [Iterator](http://docs.google.com/java/util/Iterator.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> | [**descendingIterator**](http://docs.google.com/java/util/NavigableSet.html#descendingIterator())()            Returns an iterator over the elements in this set, in descending order. |
| [NavigableSet](http://docs.google.com/java/util/NavigableSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> | [**descendingSet**](http://docs.google.com/java/util/NavigableSet.html#descendingSet())()            Returns a reverse order view of the elements contained in this set. |
| [E](http://docs.google.com/java/util/NavigableSet.html) | [**floor**](http://docs.google.com/java/util/NavigableSet.html#floor(E))([E](http://docs.google.com/java/util/NavigableSet.html) e)            Returns the greatest element in this set less than or equal to the given element, or null if there is no such element. |
| [SortedSet](http://docs.google.com/java/util/SortedSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> | [**headSet**](http://docs.google.com/java/util/NavigableSet.html#headSet(E))([E](http://docs.google.com/java/util/NavigableSet.html) toElement)            Returns a view of the portion of this set whose elements are strictly less than toElement. |
| [NavigableSet](http://docs.google.com/java/util/NavigableSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> | [**headSet**](http://docs.google.com/java/util/NavigableSet.html#headSet(E,%20boolean))([E](http://docs.google.com/java/util/NavigableSet.html) toElement, boolean inclusive)            Returns a view of the portion of this set whose elements are less than (or equal to, if inclusive is true) toElement. |
| [E](http://docs.google.com/java/util/NavigableSet.html) | [**higher**](http://docs.google.com/java/util/NavigableSet.html#higher(E))([E](http://docs.google.com/java/util/NavigableSet.html) e)            Returns the least element in this set strictly greater than the given element, or null if there is no such element. |
| [Iterator](http://docs.google.com/java/util/Iterator.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> | [**iterator**](http://docs.google.com/java/util/NavigableSet.html#iterator())()            Returns an iterator over the elements in this set, in ascending order. |
| [E](http://docs.google.com/java/util/NavigableSet.html) | [**lower**](http://docs.google.com/java/util/NavigableSet.html#lower(E))([E](http://docs.google.com/java/util/NavigableSet.html) e)            Returns the greatest element in this set strictly less than the given element, or null if there is no such element. |
| [E](http://docs.google.com/java/util/NavigableSet.html) | [**pollFirst**](http://docs.google.com/java/util/NavigableSet.html#pollFirst())()            Retrieves and removes the first (lowest) element, or returns null if this set is empty. |
| [E](http://docs.google.com/java/util/NavigableSet.html) | [**pollLast**](http://docs.google.com/java/util/NavigableSet.html#pollLast())()            Retrieves and removes the last (highest) element, or returns null if this set is empty. |
| [NavigableSet](http://docs.google.com/java/util/NavigableSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> | [**subSet**](http://docs.google.com/java/util/NavigableSet.html#subSet(E,%20boolean,%20E,%20boolean))([E](http://docs.google.com/java/util/NavigableSet.html) fromElement, boolean fromInclusive, [E](http://docs.google.com/java/util/NavigableSet.html) toElement, boolean toInclusive)            Returns a view of the portion of this set whose elements range from fromElement to toElement. |
| [SortedSet](http://docs.google.com/java/util/SortedSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> | [**subSet**](http://docs.google.com/java/util/NavigableSet.html#subSet(E,%20E))([E](http://docs.google.com/java/util/NavigableSet.html) fromElement, [E](http://docs.google.com/java/util/NavigableSet.html) toElement)            Returns a view of the portion of this set whose elements range from fromElement, inclusive, to toElement, exclusive. |
| [SortedSet](http://docs.google.com/java/util/SortedSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> | [**tailSet**](http://docs.google.com/java/util/NavigableSet.html#tailSet(E))([E](http://docs.google.com/java/util/NavigableSet.html) fromElement)            Returns a view of the portion of this set whose elements are greater than or equal to fromElement. |
| [NavigableSet](http://docs.google.com/java/util/NavigableSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> | [**tailSet**](http://docs.google.com/java/util/NavigableSet.html#tailSet(E,%20boolean))([E](http://docs.google.com/java/util/NavigableSet.html) fromElement, boolean inclusive)            Returns a view of the portion of this set whose elements are greater than (or equal to, if inclusive is true) fromElement. |

| **Methods inherited from interface java.util.**[**SortedSet**](http://docs.google.com/java/util/SortedSet.html) |
| --- |
| [comparator](http://docs.google.com/java/util/SortedSet.html#comparator()), [first](http://docs.google.com/java/util/SortedSet.html#first()), [last](http://docs.google.com/java/util/SortedSet.html#last()) |

| **Methods inherited from interface java.util.**[**Set**](http://docs.google.com/java/util/Set.html) |
| --- |
| [add](http://docs.google.com/java/util/Set.html#add(E)), [addAll](http://docs.google.com/java/util/Set.html#addAll(java.util.Collection)), [clear](http://docs.google.com/java/util/Set.html#clear()), [contains](http://docs.google.com/java/util/Set.html#contains(java.lang.Object)), [containsAll](http://docs.google.com/java/util/Set.html#containsAll(java.util.Collection)), [equals](http://docs.google.com/java/util/Set.html#equals(java.lang.Object)), [hashCode](http://docs.google.com/java/util/Set.html#hashCode()), [isEmpty](http://docs.google.com/java/util/Set.html#isEmpty()), [remove](http://docs.google.com/java/util/Set.html#remove(java.lang.Object)), [removeAll](http://docs.google.com/java/util/Set.html#removeAll(java.util.Collection)), [retainAll](http://docs.google.com/java/util/Set.html#retainAll(java.util.Collection)), [size](http://docs.google.com/java/util/Set.html#size()), [toArray](http://docs.google.com/java/util/Set.html#toArray()), [toArray](http://docs.google.com/java/util/Set.html#toArray(T%5B%5D)) |

| **Method Detail** |
| --- |

### lower

[E](http://docs.google.com/java/util/NavigableSet.html) **lower**([E](http://docs.google.com/java/util/NavigableSet.html) e)

Returns the greatest element in this set strictly less than the given element, or null if there is no such element.

**Parameters:**e - the value to match **Returns:**the greatest element less than e, or null if there is no such element **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified element cannot be compared with the elements currently in the set [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified element is null and this set does not permit null elements

### floor

[E](http://docs.google.com/java/util/NavigableSet.html) **floor**([E](http://docs.google.com/java/util/NavigableSet.html) e)

Returns the greatest element in this set less than or equal to the given element, or null if there is no such element.

**Parameters:**e - the value to match **Returns:**the greatest element less than or equal to e, or null if there is no such element **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified element cannot be compared with the elements currently in the set [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified element is null and this set does not permit null elements

### ceiling

[E](http://docs.google.com/java/util/NavigableSet.html) **ceiling**([E](http://docs.google.com/java/util/NavigableSet.html) e)

Returns the least element in this set greater than or equal to the given element, or null if there is no such element.

**Parameters:**e - the value to match **Returns:**the least element greater than or equal to e, or null if there is no such element **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified element cannot be compared with the elements currently in the set [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified element is null and this set does not permit null elements

### higher

[E](http://docs.google.com/java/util/NavigableSet.html) **higher**([E](http://docs.google.com/java/util/NavigableSet.html) e)

Returns the least element in this set strictly greater than the given element, or null if there is no such element.

**Parameters:**e - the value to match **Returns:**the least element greater than e, or null if there is no such element **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified element cannot be compared with the elements currently in the set [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified element is null and this set does not permit null elements

### pollFirst

[E](http://docs.google.com/java/util/NavigableSet.html) **pollFirst**()

Retrieves and removes the first (lowest) element, or returns null if this set is empty.

**Returns:**the first element, or null if this set is empty

### pollLast

[E](http://docs.google.com/java/util/NavigableSet.html) **pollLast**()

Retrieves and removes the last (highest) element, or returns null if this set is empty.

**Returns:**the last element, or null if this set is empty

### iterator

[Iterator](http://docs.google.com/java/util/Iterator.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> **iterator**()

Returns an iterator over the elements in this set, in ascending order.

**Specified by:**[iterator](http://docs.google.com/java/util/Collection.html#iterator()) in interface [Collection](http://docs.google.com/java/util/Collection.html)<[E](http://docs.google.com/java/util/NavigableSet.html)>**Specified by:**[iterator](http://docs.google.com/java/lang/Iterable.html#iterator()) in interface [Iterable](http://docs.google.com/java/lang/Iterable.html)<[E](http://docs.google.com/java/util/NavigableSet.html)>**Specified by:**[iterator](http://docs.google.com/java/util/Set.html#iterator()) in interface [Set](http://docs.google.com/java/util/Set.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> **Returns:**an iterator over the elements in this set, in ascending order

### descendingSet

[NavigableSet](http://docs.google.com/java/util/NavigableSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> **descendingSet**()

Returns a reverse order view of the elements contained in this set. The descending set is backed by this set, so changes to the set are reflected in the descending set, and vice-versa. If either set is modified while an iteration over either set is in progress (except through the iterator's own remove operation), the results of the iteration are undefined.

The returned set has an ordering equivalent to [Collections.reverseOrder](http://docs.google.com/java/util/Collections.html#reverseOrder(java.util.Comparator))(comparator()). The expression s.descendingSet().descendingSet() returns a view of s essentially equivalent to s.

**Returns:**a reverse order view of this set

### descendingIterator

[Iterator](http://docs.google.com/java/util/Iterator.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> **descendingIterator**()

Returns an iterator over the elements in this set, in descending order. Equivalent in effect to descendingSet().iterator().

**Returns:**an iterator over the elements in this set, in descending order

### subSet

[NavigableSet](http://docs.google.com/java/util/NavigableSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> **subSet**([E](http://docs.google.com/java/util/NavigableSet.html) fromElement,  
 boolean fromInclusive,  
 [E](http://docs.google.com/java/util/NavigableSet.html) toElement,  
 boolean toInclusive)

Returns a view of the portion of this set whose elements range from fromElement to toElement. If fromElement and toElement are equal, the returned set is empty unless fromExclusive and toExclusive are both true. The returned set is backed by this set, so changes in the returned set are reflected in this set, and vice-versa. The returned set supports all optional set operations that this set supports.

The returned set will throw an IllegalArgumentException on an attempt to insert an element outside its range.

**Parameters:**fromElement - low endpoint of the returned setfromInclusive - true if the low endpoint is to be included in the returned viewtoElement - high endpoint of the returned settoInclusive - true if the high endpoint is to be included in the returned view **Returns:**a view of the portion of this set whose elements range from fromElement, inclusive, to toElement, exclusive **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if fromElement and toElement cannot be compared to one another using this set's comparator (or, if the set has no comparator, using natural ordering). Implementations may, but are not required to, throw this exception if fromElement or toElement cannot be compared to elements currently in the set. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if fromElement or toElement is null and this set does not permit null elements [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if fromElement is greater than toElement; or if this set itself has a restricted range, and fromElement or toElement lies outside the bounds of the range.

### headSet

[NavigableSet](http://docs.google.com/java/util/NavigableSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> **headSet**([E](http://docs.google.com/java/util/NavigableSet.html) toElement,  
 boolean inclusive)

Returns a view of the portion of this set whose elements are less than (or equal to, if inclusive is true) toElement. The returned set is backed by this set, so changes in the returned set are reflected in this set, and vice-versa. The returned set supports all optional set operations that this set supports.

The returned set will throw an IllegalArgumentException on an attempt to insert an element outside its range.

**Parameters:**toElement - high endpoint of the returned setinclusive - true if the high endpoint is to be included in the returned view **Returns:**a view of the portion of this set whose elements are less than (or equal to, if inclusive is true) toElement **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if toElement is not compatible with this set's comparator (or, if the set has no comparator, if toElement does not implement [Comparable](http://docs.google.com/java/lang/Comparable.html)). Implementations may, but are not required to, throw this exception if toElement cannot be compared to elements currently in the set. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if toElement is null and this set does not permit null elements [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if this set itself has a restricted range, and toElement lies outside the bounds of the range

### tailSet

[NavigableSet](http://docs.google.com/java/util/NavigableSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> **tailSet**([E](http://docs.google.com/java/util/NavigableSet.html) fromElement,  
 boolean inclusive)

Returns a view of the portion of this set whose elements are greater than (or equal to, if inclusive is true) fromElement. The returned set is backed by this set, so changes in the returned set are reflected in this set, and vice-versa. The returned set supports all optional set operations that this set supports.

The returned set will throw an IllegalArgumentException on an attempt to insert an element outside its range.

**Parameters:**fromElement - low endpoint of the returned setinclusive - true if the low endpoint is to be included in the returned view **Returns:**a view of the portion of this set whose elements are greater than or equal to fromElement **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if fromElement is not compatible with this set's comparator (or, if the set has no comparator, if fromElement does not implement [Comparable](http://docs.google.com/java/lang/Comparable.html)). Implementations may, but are not required to, throw this exception if fromElement cannot be compared to elements currently in the set. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if fromElement is null and this set does not permit null elements [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if this set itself has a restricted range, and fromElement lies outside the bounds of the range

### subSet

[SortedSet](http://docs.google.com/java/util/SortedSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> **subSet**([E](http://docs.google.com/java/util/NavigableSet.html) fromElement,  
 [E](http://docs.google.com/java/util/NavigableSet.html) toElement)

Returns a view of the portion of this set whose elements range from fromElement, inclusive, to toElement, exclusive. (If fromElement and toElement are equal, the returned set is empty.) The returned set is backed by this set, so changes in the returned set are reflected in this set, and vice-versa. The returned set supports all optional set operations that this set supports.

The returned set will throw an IllegalArgumentException on an attempt to insert an element outside its range.

Equivalent to subSet(fromElement, true, toElement, false).

**Specified by:**[subSet](http://docs.google.com/java/util/SortedSet.html#subSet(E,%20E)) in interface [SortedSet](http://docs.google.com/java/util/SortedSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> **Parameters:**fromElement - low endpoint (inclusive) of the returned settoElement - high endpoint (exclusive) of the returned set **Returns:**a view of the portion of this set whose elements range from fromElement, inclusive, to toElement, exclusive **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if fromElement and toElement cannot be compared to one another using this set's comparator (or, if the set has no comparator, using natural ordering). Implementations may, but are not required to, throw this exception if fromElement or toElement cannot be compared to elements currently in the set. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if fromElement or toElement is null and this set does not permit null elements [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if fromElement is greater than toElement; or if this set itself has a restricted range, and fromElement or toElement lies outside the bounds of the range

### headSet

[SortedSet](http://docs.google.com/java/util/SortedSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> **headSet**([E](http://docs.google.com/java/util/NavigableSet.html) toElement)

Returns a view of the portion of this set whose elements are strictly less than toElement. The returned set is backed by this set, so changes in the returned set are reflected in this set, and vice-versa. The returned set supports all optional set operations that this set supports.

The returned set will throw an IllegalArgumentException on an attempt to insert an element outside its range.

Equivalent to headSet(toElement, false).

**Specified by:**[headSet](http://docs.google.com/java/util/SortedSet.html#headSet(E)) in interface [SortedSet](http://docs.google.com/java/util/SortedSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> **Parameters:**toElement - high endpoint (exclusive) of the returned set **Returns:**a view of the portion of this set whose elements are strictly less than toElement **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if toElement is not compatible with this set's comparator (or, if the set has no comparator, if toElement does not implement [Comparable](http://docs.google.com/java/lang/Comparable.html)). Implementations may, but are not required to, throw this exception if toElement cannot be compared to elements currently in the set. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if toElement is null and this set does not permit null elements [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if this set itself has a restricted range, and toElement lies outside the bounds of the range na

### tailSet

[SortedSet](http://docs.google.com/java/util/SortedSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> **tailSet**([E](http://docs.google.com/java/util/NavigableSet.html) fromElement)

Returns a view of the portion of this set whose elements are greater than or equal to fromElement. The returned set is backed by this set, so changes in the returned set are reflected in this set, and vice-versa. The returned set supports all optional set operations that this set supports.

The returned set will throw an IllegalArgumentException on an attempt to insert an element outside its range.

Equivalent to tailSet(fromElement, true).

**Specified by:**[tailSet](http://docs.google.com/java/util/SortedSet.html#tailSet(E)) in interface [SortedSet](http://docs.google.com/java/util/SortedSet.html)<[E](http://docs.google.com/java/util/NavigableSet.html)> **Parameters:**fromElement - low endpoint (inclusive) of the returned set **Returns:**a view of the portion of this set whose elements are greater than or equal to fromElement **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if fromElement is not compatible with this set's comparator (or, if the set has no comparator, if fromElement does not implement [Comparable](http://docs.google.com/java/lang/Comparable.html)). Implementations may, but are not required to, throw this exception if fromElement cannot be compared to elements currently in the set. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if fromElement is null and this set does not permit null elements [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if this set itself has a restricted range, and fromElement lies outside the bounds of the range

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/NavigableSet.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/java/util/NavigableMap.html)   [**NEXT CLASS**](http://docs.google.com/java/util/NoSuchElementException.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/NavigableSet.html)    [**NO FRAMES**](http://docs.google.com/NavigableSet.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#3znysh7) | DETAIL: FIELD | CONSTR | [METHOD](#3dy6vkm) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).