| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/TreeMap.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/TooManyListenersException.html)   [**NEXT CLASS**](http://docs.google.com/java/util/TreeSet.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/TreeMap.html)    [**NO FRAMES**](http://docs.google.com/TreeMap.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | FIELD | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: FIELD | [CONSTR](#17dp8vu) | [METHOD](#1ksv4uv) |

## **java.util**

Class TreeMap<K,V>

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [java.util.AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<K,V>  
 **java.util.TreeMap<K,V>**

**Type Parameters:**K - the type of keys maintained by this mapV - the type of mapped values **All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html), [Cloneable](http://docs.google.com/java/lang/Cloneable.html), [Map](http://docs.google.com/java/util/Map.html)<K,V>, [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<K,V>, [SortedMap](http://docs.google.com/java/util/SortedMap.html)<K,V>

public class **TreeMap<K,V>**extends [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<K,V>implements [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<K,V>, [Cloneable](http://docs.google.com/java/lang/Cloneable.html), [Serializable](http://docs.google.com/java/io/Serializable.html)

A Red-Black tree based [NavigableMap](http://docs.google.com/java/util/NavigableMap.html) implementation. The map is sorted according to the [natural ordering](http://docs.google.com/java/lang/Comparable.html) of its keys, or by a [Comparator](http://docs.google.com/java/util/Comparator.html) provided at map creation time, depending on which constructor is used.

This implementation provides guaranteed log(n) time cost for the containsKey, get, put and remove operations. Algorithms are adaptations of those in Cormen, Leiserson, and Rivest's *Introduction to Algorithms*.

Note that the ordering maintained by a sorted map (whether or not an explicit comparator is provided) must be *consistent with equals* if this sorted map is to correctly implement the Map interface. (See Comparable or Comparator for a precise definition of *consistent with equals*.) This is so because the Map interface is defined in terms of the equals operation, but a map performs all key comparisons using its compareTo (or compare) method, so two keys that are deemed equal by this method are, from the standpoint of the sorted map, equal. The behavior of a sorted map *is* well-defined even if its ordering is inconsistent with equals; it just fails to obey the general contract of the Map interface.

**Note that this implementation is not synchronized.** If multiple threads access a map concurrently, and at least one of the threads modifies the map structurally, it *must* be synchronized externally. (A structural modification is any operation that adds or deletes one or more mappings; merely changing the value associated with an existing key is not a structural modification.) This is typically accomplished by synchronizing on some object that naturally encapsulates the map. If no such object exists, the map should be "wrapped" using the [Collections.synchronizedSortedMap](http://docs.google.com/java/util/Collections.html#synchronizedSortedMap(java.util.SortedMap)) method. This is best done at creation time, to prevent accidental unsynchronized access to the map:

SortedMap m = Collections.synchronizedSortedMap(new TreeMap(...));

The iterators returned by the iterator method of the collections returned by all of this class's "collection view methods" are *fail-fast*: if the map is structurally modified at any time after the iterator is created, in any way except through the iterator's own remove method, the iterator will throw a [ConcurrentModificationException](http://docs.google.com/java/util/ConcurrentModificationException.html). Thus, in the face of concurrent modification, the iterator fails quickly and cleanly, rather than risking arbitrary, non-deterministic behavior at an undetermined time in the future.

Note that the fail-fast behavior of an iterator cannot be guaranteed as it is, generally speaking, impossible to make any hard guarantees in the presence of unsynchronized concurrent modification. Fail-fast iterators throw ConcurrentModificationException on a best-effort basis. Therefore, it would be wrong to write a program that depended on this exception for its correctness: *the fail-fast behavior of iterators should be used only to detect bugs.*

All Map.Entry pairs returned by methods in this class and its views represent snapshots of mappings at the time they were produced. They do *not* support the Entry.setValue method. (Note however that it is possible to change mappings in the associated map using put.)

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

**Since:** 1.2 **See Also:**[Map](http://docs.google.com/java/util/Map.html), [HashMap](http://docs.google.com/java/util/HashMap.html), [Hashtable](http://docs.google.com/java/util/Hashtable.html), [Comparable](http://docs.google.com/java/lang/Comparable.html), [Comparator](http://docs.google.com/java/util/Comparator.html), [Collection](http://docs.google.com/java/util/Collection.html), [Serialized Form](http://docs.google.com/serialized-form.html#java.util.TreeMap)

| **Nested Class Summary** | |
| --- | --- |

| **Nested classes/interfaces inherited from class java.util.**[**AbstractMap**](http://docs.google.com/java/util/AbstractMap.html) |
| --- |
| [AbstractMap.SimpleEntry](http://docs.google.com/java/util/AbstractMap.SimpleEntry.html)<[K](http://docs.google.com/java/util/AbstractMap.SimpleEntry.html),[V](http://docs.google.com/java/util/AbstractMap.SimpleEntry.html)>, [AbstractMap.SimpleImmutableEntry](http://docs.google.com/java/util/AbstractMap.SimpleImmutableEntry.html)<[K](http://docs.google.com/java/util/AbstractMap.SimpleImmutableEntry.html),[V](http://docs.google.com/java/util/AbstractMap.SimpleImmutableEntry.html)> |

| **Constructor Summary** | |
| --- | --- |
| [**TreeMap**](http://docs.google.com/java/util/TreeMap.html#TreeMap())()            Constructs a new, empty tree map, using the natural ordering of its keys. |
| [**TreeMap**](http://docs.google.com/java/util/TreeMap.html#TreeMap(java.util.Comparator))([Comparator](http://docs.google.com/java/util/Comparator.html)<? super [K](http://docs.google.com/java/util/TreeMap.html)> comparator)            Constructs a new, empty tree map, ordered according to the given comparator. |
| [**TreeMap**](http://docs.google.com/java/util/TreeMap.html#TreeMap(java.util.Map))([Map](http://docs.google.com/java/util/Map.html)<? extends [K](http://docs.google.com/java/util/TreeMap.html),? extends [V](http://docs.google.com/java/util/TreeMap.html)> m)            Constructs a new tree map containing the same mappings as the given map, ordered according to the *natural ordering* of its keys. |
| [**TreeMap**](http://docs.google.com/java/util/TreeMap.html#TreeMap(java.util.SortedMap))([SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),? extends [V](http://docs.google.com/java/util/TreeMap.html)> m)            Constructs a new tree map containing the same mappings and using the same ordering as the specified sorted map. |

| **Method Summary** | |
| --- | --- |
| [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**ceilingEntry**](http://docs.google.com/java/util/TreeMap.html#ceilingEntry(K))([K](http://docs.google.com/java/util/TreeMap.html) key)            Returns a key-value mapping associated with the least key greater than or equal to the given key, or null if there is no such key. |
| [K](http://docs.google.com/java/util/TreeMap.html) | [**ceilingKey**](http://docs.google.com/java/util/TreeMap.html#ceilingKey(K))([K](http://docs.google.com/java/util/TreeMap.html) key)            Returns the least key greater than or equal to the given key, or null if there is no such key. |
| void | [**clear**](http://docs.google.com/java/util/TreeMap.html#clear())()            Removes all of the mappings from this map. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**clone**](http://docs.google.com/java/util/TreeMap.html#clone())()            Returns a shallow copy of this TreeMap instance. |
| [Comparator](http://docs.google.com/java/util/Comparator.html)<? super [K](http://docs.google.com/java/util/TreeMap.html)> | [**comparator**](http://docs.google.com/java/util/TreeMap.html#comparator())()            Returns the comparator used to order the keys in this map, or null if this map uses the [natural ordering](http://docs.google.com/java/lang/Comparable.html) of its keys. |
| boolean | [**containsKey**](http://docs.google.com/java/util/TreeMap.html#containsKey(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) key)            Returns true if this map contains a mapping for the specified key. |
| boolean | [**containsValue**](http://docs.google.com/java/util/TreeMap.html#containsValue(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) value)            Returns true if this map maps one or more keys to the specified value. |
| [NavigableSet](http://docs.google.com/java/util/NavigableSet.html)<[K](http://docs.google.com/java/util/TreeMap.html)> | [**descendingKeySet**](http://docs.google.com/java/util/TreeMap.html#descendingKeySet())()            Returns a reverse order [NavigableSet](http://docs.google.com/java/util/NavigableSet.html) view of the keys contained in this map. |
| [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**descendingMap**](http://docs.google.com/java/util/TreeMap.html#descendingMap())()            Returns a reverse order view of the mappings contained in this map. |
| [Set](http://docs.google.com/java/util/Set.html)<[Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>> | [**entrySet**](http://docs.google.com/java/util/TreeMap.html#entrySet())()            Returns a [Set](http://docs.google.com/java/util/Set.html) view of the mappings contained in this map. |
| [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**firstEntry**](http://docs.google.com/java/util/TreeMap.html#firstEntry())()            Returns a key-value mapping associated with the least key in this map, or null if the map is empty. |
| [K](http://docs.google.com/java/util/TreeMap.html) | [**firstKey**](http://docs.google.com/java/util/TreeMap.html#firstKey())()            Returns the first (lowest) key currently in this map. |
| [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**floorEntry**](http://docs.google.com/java/util/TreeMap.html#floorEntry(K))([K](http://docs.google.com/java/util/TreeMap.html) key)            Returns a key-value mapping associated with the greatest key less than or equal to the given key, or null if there is no such key. |
| [K](http://docs.google.com/java/util/TreeMap.html) | [**floorKey**](http://docs.google.com/java/util/TreeMap.html#floorKey(K))([K](http://docs.google.com/java/util/TreeMap.html) key)            Returns the greatest key less than or equal to the given key, or null if there is no such key. |
| [V](http://docs.google.com/java/util/TreeMap.html) | [**get**](http://docs.google.com/java/util/TreeMap.html#get(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) key)            Returns the value to which the specified key is mapped, or null if this map contains no mapping for the key. |
| [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**headMap**](http://docs.google.com/java/util/TreeMap.html#headMap(K))([K](http://docs.google.com/java/util/TreeMap.html) toKey)            Returns a view of the portion of this map whose keys are strictly less than toKey. |
| [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**headMap**](http://docs.google.com/java/util/TreeMap.html#headMap(K,%20boolean))([K](http://docs.google.com/java/util/TreeMap.html) toKey, boolean inclusive)            Returns a view of the portion of this map whose keys are less than (or equal to, if inclusive is true) toKey. |
| [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**higherEntry**](http://docs.google.com/java/util/TreeMap.html#higherEntry(K))([K](http://docs.google.com/java/util/TreeMap.html) key)            Returns a key-value mapping associated with the least key strictly greater than the given key, or null if there is no such key. |
| [K](http://docs.google.com/java/util/TreeMap.html) | [**higherKey**](http://docs.google.com/java/util/TreeMap.html#higherKey(K))([K](http://docs.google.com/java/util/TreeMap.html) key)            Returns the least key strictly greater than the given key, or null if there is no such key. |
| [Set](http://docs.google.com/java/util/Set.html)<[K](http://docs.google.com/java/util/TreeMap.html)> | [**keySet**](http://docs.google.com/java/util/TreeMap.html#keySet())()            Returns a [Set](http://docs.google.com/java/util/Set.html) view of the keys contained in this map. |
| [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**lastEntry**](http://docs.google.com/java/util/TreeMap.html#lastEntry())()            Returns a key-value mapping associated with the greatest key in this map, or null if the map is empty. |
| [K](http://docs.google.com/java/util/TreeMap.html) | [**lastKey**](http://docs.google.com/java/util/TreeMap.html#lastKey())()            Returns the last (highest) key currently in this map. |
| [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**lowerEntry**](http://docs.google.com/java/util/TreeMap.html#lowerEntry(K))([K](http://docs.google.com/java/util/TreeMap.html) key)            Returns a key-value mapping associated with the greatest key strictly less than the given key, or null if there is no such key. |
| [K](http://docs.google.com/java/util/TreeMap.html) | [**lowerKey**](http://docs.google.com/java/util/TreeMap.html#lowerKey(K))([K](http://docs.google.com/java/util/TreeMap.html) key)            Returns the greatest key strictly less than the given key, or null if there is no such key. |
| [NavigableSet](http://docs.google.com/java/util/NavigableSet.html)<[K](http://docs.google.com/java/util/TreeMap.html)> | [**navigableKeySet**](http://docs.google.com/java/util/TreeMap.html#navigableKeySet())()            Returns a [NavigableSet](http://docs.google.com/java/util/NavigableSet.html) view of the keys contained in this map. |
| [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**pollFirstEntry**](http://docs.google.com/java/util/TreeMap.html#pollFirstEntry())()            Removes and returns a key-value mapping associated with the least key in this map, or null if the map is empty. |
| [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**pollLastEntry**](http://docs.google.com/java/util/TreeMap.html#pollLastEntry())()            Removes and returns a key-value mapping associated with the greatest key in this map, or null if the map is empty. |
| [V](http://docs.google.com/java/util/TreeMap.html) | [**put**](http://docs.google.com/java/util/TreeMap.html#put(K,%20V))([K](http://docs.google.com/java/util/TreeMap.html) key, [V](http://docs.google.com/java/util/TreeMap.html) value)            Associates the specified value with the specified key in this map. |
| void | [**putAll**](http://docs.google.com/java/util/TreeMap.html#putAll(java.util.Map))([Map](http://docs.google.com/java/util/Map.html)<? extends [K](http://docs.google.com/java/util/TreeMap.html),? extends [V](http://docs.google.com/java/util/TreeMap.html)> map)            Copies all of the mappings from the specified map to this map. |
| [V](http://docs.google.com/java/util/TreeMap.html) | [**remove**](http://docs.google.com/java/util/TreeMap.html#remove(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) key)            Removes the mapping for this key from this TreeMap if present. |
| int | [**size**](http://docs.google.com/java/util/TreeMap.html#size())()            Returns the number of key-value mappings in this map. |
| [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**subMap**](http://docs.google.com/java/util/TreeMap.html#subMap(K,%20boolean,%20K,%20boolean))([K](http://docs.google.com/java/util/TreeMap.html) fromKey, boolean fromInclusive, [K](http://docs.google.com/java/util/TreeMap.html) toKey, boolean toInclusive)            Returns a view of the portion of this map whose keys range from fromKey to toKey. |
| [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**subMap**](http://docs.google.com/java/util/TreeMap.html#subMap(K,%20K))([K](http://docs.google.com/java/util/TreeMap.html) fromKey, [K](http://docs.google.com/java/util/TreeMap.html) toKey)            Returns a view of the portion of this map whose keys range from fromKey, inclusive, to toKey, exclusive. |
| [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**tailMap**](http://docs.google.com/java/util/TreeMap.html#tailMap(K))([K](http://docs.google.com/java/util/TreeMap.html) fromKey)            Returns a view of the portion of this map whose keys are greater than or equal to fromKey. |
| [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> | [**tailMap**](http://docs.google.com/java/util/TreeMap.html#tailMap(K,%20boolean))([K](http://docs.google.com/java/util/TreeMap.html) fromKey, boolean inclusive)            Returns a view of the portion of this map whose keys are greater than (or equal to, if inclusive is true) fromKey. |
| [Collection](http://docs.google.com/java/util/Collection.html)<[V](http://docs.google.com/java/util/TreeMap.html)> | [**values**](http://docs.google.com/java/util/TreeMap.html#values())()            Returns a [Collection](http://docs.google.com/java/util/Collection.html) view of the values contained in this map. |

| **Methods inherited from class java.util.**[**AbstractMap**](http://docs.google.com/java/util/AbstractMap.html) |
| --- |
| [equals](http://docs.google.com/java/util/AbstractMap.html#equals(java.lang.Object)), [hashCode](http://docs.google.com/java/util/AbstractMap.html#hashCode()), [isEmpty](http://docs.google.com/java/util/AbstractMap.html#isEmpty()), [toString](http://docs.google.com/java/util/AbstractMap.html#toString()) |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [wait](http://docs.google.com/java/lang/Object.html#wait()), [wait](http://docs.google.com/java/lang/Object.html#wait(long)), [wait](http://docs.google.com/java/lang/Object.html#wait(long,%20int)) |

| **Methods inherited from interface java.util.**[**Map**](http://docs.google.com/java/util/Map.html) |
| --- |
| [equals](http://docs.google.com/java/util/Map.html#equals(java.lang.Object)), [hashCode](http://docs.google.com/java/util/Map.html#hashCode()), [isEmpty](http://docs.google.com/java/util/Map.html#isEmpty()) |

| **Constructor Detail** |
| --- |

### TreeMap

public **TreeMap**()

Constructs a new, empty tree map, using the natural ordering of its keys. All keys inserted into the map must implement the [Comparable](http://docs.google.com/java/lang/Comparable.html) interface. Furthermore, all such keys must be *mutually comparable*: k1.compareTo(k2) must not throw a ClassCastException for any keys k1 and k2 in the map. If the user attempts to put a key into the map that violates this constraint (for example, the user attempts to put a string key into a map whose keys are integers), the put(Object key, Object value) call will throw a ClassCastException.

### TreeMap

public **TreeMap**([Comparator](http://docs.google.com/java/util/Comparator.html)<? super [K](http://docs.google.com/java/util/TreeMap.html)> comparator)

Constructs a new, empty tree map, ordered according to the given comparator. All keys inserted into the map must be *mutually comparable* by the given comparator: comparator.compare(k1, k2) must not throw a ClassCastException for any keys k1 and k2 in the map. If the user attempts to put a key into the map that violates this constraint, the put(Object key, Object value) call will throw a ClassCastException.

**Parameters:**comparator - the comparator that will be used to order this map. If null, the [natural ordering](http://docs.google.com/java/lang/Comparable.html) of the keys will be used.

### TreeMap

public **TreeMap**([Map](http://docs.google.com/java/util/Map.html)<? extends [K](http://docs.google.com/java/util/TreeMap.html),? extends [V](http://docs.google.com/java/util/TreeMap.html)> m)

Constructs a new tree map containing the same mappings as the given map, ordered according to the *natural ordering* of its keys. All keys inserted into the new map must implement the [Comparable](http://docs.google.com/java/lang/Comparable.html) interface. Furthermore, all such keys must be *mutually comparable*: k1.compareTo(k2) must not throw a ClassCastException for any keys k1 and k2 in the map. This method runs in n\*log(n) time.

**Parameters:**m - the map whose mappings are to be placed in this map **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the keys in m are not [Comparable](http://docs.google.com/java/lang/Comparable.html), or are not mutually comparable [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified map is null

### TreeMap

public **TreeMap**([SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),? extends [V](http://docs.google.com/java/util/TreeMap.html)> m)

Constructs a new tree map containing the same mappings and using the same ordering as the specified sorted map. This method runs in linear time.

**Parameters:**m - the sorted map whose mappings are to be placed in this map, and whose comparator is to be used to sort this map **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified map is null

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

### size

public int **size**()

Returns the number of key-value mappings in this map.

**Specified by:**[size](http://docs.google.com/java/util/Map.html#size()) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Overrides:**[size](http://docs.google.com/java/util/AbstractMap.html#size()) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**the number of key-value mappings in this map

### containsKey

public boolean **containsKey**([Object](http://docs.google.com/java/lang/Object.html) key)

Returns true if this map contains a mapping for the specified key.

**Specified by:**[containsKey](http://docs.google.com/java/util/Map.html#containsKey(java.lang.Object)) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Overrides:**[containsKey](http://docs.google.com/java/util/AbstractMap.html#containsKey(java.lang.Object)) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**key - key whose presence in this map is to be tested **Returns:**true if this map contains a mapping for the specified key **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified key cannot be compared with the keys currently in the map [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified key is null and this map uses natural ordering, or its comparator does not permit null keys

### containsValue

public boolean **containsValue**([Object](http://docs.google.com/java/lang/Object.html) value)

Returns true if this map maps one or more keys to the specified value. More formally, returns true if and only if this map contains at least one mapping to a value v such that (value==null ? v==null : value.equals(v)). This operation will probably require time linear in the map size for most implementations.

**Specified by:**[containsValue](http://docs.google.com/java/util/Map.html#containsValue(java.lang.Object)) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Overrides:**[containsValue](http://docs.google.com/java/util/AbstractMap.html#containsValue(java.lang.Object)) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**value - value whose presence in this map is to be tested **Returns:**true if a mapping to value exists; false otherwise**Since:** 1.2

### get

public [V](http://docs.google.com/java/util/TreeMap.html) **get**([Object](http://docs.google.com/java/lang/Object.html) key)

Returns the value to which the specified key is mapped, or null if this map contains no mapping for the key.

More formally, if this map contains a mapping from a key k to a value v such that key compares equal to k according to the map's ordering, then this method returns v; otherwise it returns null. (There can be at most one such mapping.)

A return value of null does not *necessarily* indicate that the map contains no mapping for the key; it's also possible that the map explicitly maps the key to null. The [containsKey](http://docs.google.com/java/util/TreeMap.html#containsKey(java.lang.Object)) operation may be used to distinguish these two cases.

**Specified by:**[get](http://docs.google.com/java/util/Map.html#get(java.lang.Object)) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Overrides:**[get](http://docs.google.com/java/util/AbstractMap.html#get(java.lang.Object)) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**key - the key whose associated value is to be returned **Returns:**the value to which the specified key is mapped, or null if this map contains no mapping for the key **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified key cannot be compared with the keys currently in the map [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified key is null and this map uses natural ordering, or its comparator does not permit null keys

### comparator

public [Comparator](http://docs.google.com/java/util/Comparator.html)<? super [K](http://docs.google.com/java/util/TreeMap.html)> **comparator**()

**Description copied from interface:** [**SortedMap**](http://docs.google.com/java/util/SortedMap.html#comparator()) Returns the comparator used to order the keys in this map, or null if this map uses the [natural ordering](http://docs.google.com/java/lang/Comparable.html) of its keys.

**Specified by:**[comparator](http://docs.google.com/java/util/SortedMap.html#comparator()) in interface [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**the comparator used to order the keys in this map, or null if this map uses the natural ordering of its keys

### firstKey

public [K](http://docs.google.com/java/util/TreeMap.html) **firstKey**()

**Description copied from interface:** [**SortedMap**](http://docs.google.com/java/util/SortedMap.html#firstKey()) Returns the first (lowest) key currently in this map.

**Specified by:**[firstKey](http://docs.google.com/java/util/SortedMap.html#firstKey()) in interface [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**the first (lowest) key currently in this map **Throws:** [NoSuchElementException](http://docs.google.com/java/util/NoSuchElementException.html) - if this map is empty

### lastKey

public [K](http://docs.google.com/java/util/TreeMap.html) **lastKey**()

**Description copied from interface:** [**SortedMap**](http://docs.google.com/java/util/SortedMap.html#lastKey()) Returns the last (highest) key currently in this map.

**Specified by:**[lastKey](http://docs.google.com/java/util/SortedMap.html#lastKey()) in interface [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**the last (highest) key currently in this map **Throws:** [NoSuchElementException](http://docs.google.com/java/util/NoSuchElementException.html) - if this map is empty

### putAll

public void **putAll**([Map](http://docs.google.com/java/util/Map.html)<? extends [K](http://docs.google.com/java/util/TreeMap.html),? extends [V](http://docs.google.com/java/util/TreeMap.html)> map)

Copies all of the mappings from the specified map to this map. These mappings replace any mappings that this map had for any of the keys currently in the specified map.

**Specified by:**[putAll](http://docs.google.com/java/util/Map.html#putAll(java.util.Map)) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Overrides:**[putAll](http://docs.google.com/java/util/AbstractMap.html#putAll(java.util.Map)) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**map - mappings to be stored in this map **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the class of a key or value in the specified map prevents it from being stored in this map [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified map is null or the specified map contains a null key and this map does not permit null keys

### put

public [V](http://docs.google.com/java/util/TreeMap.html) **put**([K](http://docs.google.com/java/util/TreeMap.html) key,  
 [V](http://docs.google.com/java/util/TreeMap.html) value)

Associates the specified value with the specified key in this map. If the map previously contained a mapping for the key, the old value is replaced.

**Specified by:**[put](http://docs.google.com/java/util/Map.html#put(K,%20V)) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Overrides:**[put](http://docs.google.com/java/util/AbstractMap.html#put(K,%20V)) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**key - key with which the specified value is to be associatedvalue - value to be associated with the specified key **Returns:**the previous value associated with key, or null if there was no mapping for key. (A null return can also indicate that the map previously associated null with key.) **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified key cannot be compared with the keys currently in the map [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified key is null and this map uses natural ordering, or its comparator does not permit null keys

### remove

public [V](http://docs.google.com/java/util/TreeMap.html) **remove**([Object](http://docs.google.com/java/lang/Object.html) key)

Removes the mapping for this key from this TreeMap if present.

**Specified by:**[remove](http://docs.google.com/java/util/Map.html#remove(java.lang.Object)) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Overrides:**[remove](http://docs.google.com/java/util/AbstractMap.html#remove(java.lang.Object)) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**key - key for which mapping should be removed **Returns:**the previous value associated with key, or null if there was no mapping for key. (A null return can also indicate that the map previously associated null with key.) **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified key cannot be compared with the keys currently in the map [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified key is null and this map uses natural ordering, or its comparator does not permit null keys

### clear

public void **clear**()

Removes all of the mappings from this map. The map will be empty after this call returns.

**Specified by:**[clear](http://docs.google.com/java/util/Map.html#clear()) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Overrides:**[clear](http://docs.google.com/java/util/AbstractMap.html#clear()) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>

### clone

public [Object](http://docs.google.com/java/lang/Object.html) **clone**()

Returns a shallow copy of this TreeMap instance. (The keys and values themselves are not cloned.)

**Overrides:**[clone](http://docs.google.com/java/util/AbstractMap.html#clone()) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**a shallow copy of this map**See Also:**[Cloneable](http://docs.google.com/java/lang/Cloneable.html)

### firstEntry

public [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **firstEntry**()

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#firstEntry()) Returns a key-value mapping associated with the least key in this map, or null if the map is empty.

**Specified by:**[firstEntry](http://docs.google.com/java/util/NavigableMap.html#firstEntry()) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**an entry with the least key, or null if this map is empty**Since:** 1.6

### lastEntry

public [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **lastEntry**()

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#lastEntry()) Returns a key-value mapping associated with the greatest key in this map, or null if the map is empty.

**Specified by:**[lastEntry](http://docs.google.com/java/util/NavigableMap.html#lastEntry()) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**an entry with the greatest key, or null if this map is empty**Since:** 1.6

### pollFirstEntry

public [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **pollFirstEntry**()

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#pollFirstEntry()) Removes and returns a key-value mapping associated with the least key in this map, or null if the map is empty.

**Specified by:**[pollFirstEntry](http://docs.google.com/java/util/NavigableMap.html#pollFirstEntry()) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**the removed first entry of this map, or null if this map is empty**Since:** 1.6

### pollLastEntry

public [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **pollLastEntry**()

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#pollLastEntry()) Removes and returns a key-value mapping associated with the greatest key in this map, or null if the map is empty.

**Specified by:**[pollLastEntry](http://docs.google.com/java/util/NavigableMap.html#pollLastEntry()) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**the removed last entry of this map, or null if this map is empty**Since:** 1.6

### lowerEntry

public [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **lowerEntry**([K](http://docs.google.com/java/util/TreeMap.html) key)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#lowerEntry(K)) Returns a key-value mapping associated with the greatest key strictly less than the given key, or null if there is no such key.

**Specified by:**[lowerEntry](http://docs.google.com/java/util/NavigableMap.html#lowerEntry(K)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**key - the key **Returns:**an entry with the greatest key less than key, or null if there is no such key **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified key cannot be compared with the keys currently in the map [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified key is null and this map uses natural ordering, or its comparator does not permit null keys**Since:** 1.6

### lowerKey

public [K](http://docs.google.com/java/util/TreeMap.html) **lowerKey**([K](http://docs.google.com/java/util/TreeMap.html) key)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#lowerKey(K)) Returns the greatest key strictly less than the given key, or null if there is no such key.

**Specified by:**[lowerKey](http://docs.google.com/java/util/NavigableMap.html#lowerKey(K)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**key - the key **Returns:**the greatest key less than key, or null if there is no such key **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified key cannot be compared with the keys currently in the map [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified key is null and this map uses natural ordering, or its comparator does not permit null keys**Since:** 1.6

### floorEntry

public [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **floorEntry**([K](http://docs.google.com/java/util/TreeMap.html) key)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#floorEntry(K)) Returns a key-value mapping associated with the greatest key less than or equal to the given key, or null if there is no such key.

**Specified by:**[floorEntry](http://docs.google.com/java/util/NavigableMap.html#floorEntry(K)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**key - the key **Returns:**an entry with the greatest key less than or equal to key, or null if there is no such key **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified key cannot be compared with the keys currently in the map [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified key is null and this map uses natural ordering, or its comparator does not permit null keys**Since:** 1.6

### floorKey

public [K](http://docs.google.com/java/util/TreeMap.html) **floorKey**([K](http://docs.google.com/java/util/TreeMap.html) key)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#floorKey(K)) Returns the greatest key less than or equal to the given key, or null if there is no such key.

**Specified by:**[floorKey](http://docs.google.com/java/util/NavigableMap.html#floorKey(K)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**key - the key **Returns:**the greatest key less than or equal to key, or null if there is no such key **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified key cannot be compared with the keys currently in the map [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified key is null and this map uses natural ordering, or its comparator does not permit null keys**Since:** 1.6

### ceilingEntry

public [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **ceilingEntry**([K](http://docs.google.com/java/util/TreeMap.html) key)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#ceilingEntry(K)) Returns a key-value mapping associated with the least key greater than or equal to the given key, or null if there is no such key.

**Specified by:**[ceilingEntry](http://docs.google.com/java/util/NavigableMap.html#ceilingEntry(K)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**key - the key **Returns:**an entry with the least key greater than or equal to key, or null if there is no such key **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified key cannot be compared with the keys currently in the map [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified key is null and this map uses natural ordering, or its comparator does not permit null keys**Since:** 1.6

### ceilingKey

public [K](http://docs.google.com/java/util/TreeMap.html) **ceilingKey**([K](http://docs.google.com/java/util/TreeMap.html) key)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#ceilingKey(K)) Returns the least key greater than or equal to the given key, or null if there is no such key.

**Specified by:**[ceilingKey](http://docs.google.com/java/util/NavigableMap.html#ceilingKey(K)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**key - the key **Returns:**the least key greater than or equal to key, or null if there is no such key **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified key cannot be compared with the keys currently in the map [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified key is null and this map uses natural ordering, or its comparator does not permit null keys**Since:** 1.6

### higherEntry

public [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **higherEntry**([K](http://docs.google.com/java/util/TreeMap.html) key)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#higherEntry(K)) Returns a key-value mapping associated with the least key strictly greater than the given key, or null if there is no such key.

**Specified by:**[higherEntry](http://docs.google.com/java/util/NavigableMap.html#higherEntry(K)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**key - the key **Returns:**an entry with the least key greater than key, or null if there is no such key **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified key cannot be compared with the keys currently in the map [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified key is null and this map uses natural ordering, or its comparator does not permit null keys**Since:** 1.6

### higherKey

public [K](http://docs.google.com/java/util/TreeMap.html) **higherKey**([K](http://docs.google.com/java/util/TreeMap.html) key)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#higherKey(K)) Returns the least key strictly greater than the given key, or null if there is no such key.

**Specified by:**[higherKey](http://docs.google.com/java/util/NavigableMap.html#higherKey(K)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**key - the key **Returns:**the least key greater than key, or null if there is no such key **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the specified key cannot be compared with the keys currently in the map [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified key is null and this map uses natural ordering, or its comparator does not permit null keys**Since:** 1.6

### keySet

public [Set](http://docs.google.com/java/util/Set.html)<[K](http://docs.google.com/java/util/TreeMap.html)> **keySet**()

Returns a [Set](http://docs.google.com/java/util/Set.html) view of the keys contained in this map. The set's iterator returns the keys in ascending order. The set is backed by the map, so changes to the map are reflected in the set, and vice-versa. If the map is modified while an iteration over the set is in progress (except through the iterator's own remove operation), the results of the iteration are undefined. The set supports element removal, which removes the corresponding mapping from the map, via the Iterator.remove, Set.remove, removeAll, retainAll, and clear operations. It does not support the add or addAll operations.

**Specified by:**[keySet](http://docs.google.com/java/util/Map.html#keySet()) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Specified by:**[keySet](http://docs.google.com/java/util/SortedMap.html#keySet()) in interface [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Overrides:**[keySet](http://docs.google.com/java/util/AbstractMap.html#keySet()) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**a set view of the keys contained in this map

### navigableKeySet

public [NavigableSet](http://docs.google.com/java/util/NavigableSet.html)<[K](http://docs.google.com/java/util/TreeMap.html)> **navigableKeySet**()

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#navigableKeySet()) Returns a [NavigableSet](http://docs.google.com/java/util/NavigableSet.html) view of the keys contained in this map. The set's iterator returns the keys in ascending order. The set is backed by the map, so changes to the map are reflected in the set, and vice-versa. If the map is modified while an iteration over the set is in progress (except through the iterator's own remove operation), the results of the iteration are undefined. The set supports element removal, which removes the corresponding mapping from the map, via the Iterator.remove, Set.remove, removeAll, retainAll, and clear operations. It does not support the add or addAll operations.

**Specified by:**[navigableKeySet](http://docs.google.com/java/util/NavigableMap.html#navigableKeySet()) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**a navigable set view of the keys in this map**Since:** 1.6

### descendingKeySet

public [NavigableSet](http://docs.google.com/java/util/NavigableSet.html)<[K](http://docs.google.com/java/util/TreeMap.html)> **descendingKeySet**()

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#descendingKeySet()) Returns a reverse order [NavigableSet](http://docs.google.com/java/util/NavigableSet.html) view of the keys contained in this map. The set's iterator returns the keys in descending order. The set is backed by the map, so changes to the map are reflected in the set, and vice-versa. If the map is modified while an iteration over the set is in progress (except through the iterator's own remove operation), the results of the iteration are undefined. The set supports element removal, which removes the corresponding mapping from the map, via the Iterator.remove, Set.remove, removeAll, retainAll, and clear operations. It does not support the add or addAll operations.

**Specified by:**[descendingKeySet](http://docs.google.com/java/util/NavigableMap.html#descendingKeySet()) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**a reverse order navigable set view of the keys in this map**Since:** 1.6

### values

public [Collection](http://docs.google.com/java/util/Collection.html)<[V](http://docs.google.com/java/util/TreeMap.html)> **values**()

Returns a [Collection](http://docs.google.com/java/util/Collection.html) view of the values contained in this map. The collection's iterator returns the values in ascending order of the corresponding keys. The collection is backed by the map, so changes to the map are reflected in the collection, and vice-versa. If the map is modified while an iteration over the collection is in progress (except through the iterator's own remove operation), the results of the iteration are undefined. The collection supports element removal, which removes the corresponding mapping from the map, via the Iterator.remove, Collection.remove, removeAll, retainAll and clear operations. It does not support the add or addAll operations.

**Specified by:**[values](http://docs.google.com/java/util/Map.html#values()) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Specified by:**[values](http://docs.google.com/java/util/SortedMap.html#values()) in interface [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Overrides:**[values](http://docs.google.com/java/util/AbstractMap.html#values()) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**a collection view of the values contained in this map

### entrySet

public [Set](http://docs.google.com/java/util/Set.html)<[Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>> **entrySet**()

Returns a [Set](http://docs.google.com/java/util/Set.html) view of the mappings contained in this map. The set's iterator returns the entries in ascending key order. The set is backed by the map, so changes to the map are reflected in the set, and vice-versa. If the map is modified while an iteration over the set is in progress (except through the iterator's own remove operation, or through the setValue operation on a map entry returned by the iterator) the results of the iteration are undefined. The set supports element removal, which removes the corresponding mapping from the map, via the Iterator.remove, Set.remove, removeAll, retainAll and clear operations. It does not support the add or addAll operations.

**Specified by:**[entrySet](http://docs.google.com/java/util/Map.html#entrySet()) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Specified by:**[entrySet](http://docs.google.com/java/util/SortedMap.html#entrySet()) in interface [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Specified by:**[entrySet](http://docs.google.com/java/util/AbstractMap.html#entrySet()) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**a set view of the mappings contained in this map

### descendingMap

public [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **descendingMap**()

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#descendingMap()) Returns a reverse order view of the mappings contained in this map. The descending map is backed by this map, so changes to the map are reflected in the descending map, and vice-versa. If either map is modified while an iteration over a collection view of either map is in progress (except through the iterator's own remove operation), the results of the iteration are undefined.

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

**Specified by:**[descendingMap](http://docs.google.com/java/util/NavigableMap.html#descendingMap()) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Returns:**a reverse order view of this map**Since:** 1.6

### subMap

public [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **subMap**([K](http://docs.google.com/java/util/TreeMap.html) fromKey,  
 boolean fromInclusive,  
 [K](http://docs.google.com/java/util/TreeMap.html) toKey,  
 boolean toInclusive)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#subMap(K,%20boolean,%20K,%20boolean)) Returns a view of the portion of this map whose keys range from fromKey to toKey. If fromKey and toKey are equal, the returned map is empty unless fromExclusive and toExclusive are both true. The returned map is backed by this map, so changes in the returned map are reflected in this map, and vice-versa. The returned map supports all optional map operations that this map supports.

The returned map will throw an IllegalArgumentException on an attempt to insert a key outside of its range, or to construct a submap either of whose endpoints lie outside its range.

**Specified by:**[subMap](http://docs.google.com/java/util/NavigableMap.html#subMap(K,%20boolean,%20K,%20boolean)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**fromKey - low endpoint of the keys in the returned mapfromInclusive - true if the low endpoint is to be included in the returned viewtoKey - high endpoint of the keys in the returned maptoInclusive - true if the high endpoint is to be included in the returned view **Returns:**a view of the portion of this map whose keys range from fromKey to toKey **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if fromKey and toKey cannot be compared to one another using this map's comparator (or, if the map has no comparator, using natural ordering). Implementations may, but are not required to, throw this exception if fromKey or toKey cannot be compared to keys currently in the map. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if fromKey or toKey is null and this map uses natural ordering, or its comparator does not permit null keys [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if fromKey is greater than toKey; or if this map itself has a restricted range, and fromKey or toKey lies outside the bounds of the range**Since:** 1.6

### headMap

public [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **headMap**([K](http://docs.google.com/java/util/TreeMap.html) toKey,  
 boolean inclusive)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#headMap(K,%20boolean)) Returns a view of the portion of this map whose keys are less than (or equal to, if inclusive is true) toKey. The returned map is backed by this map, so changes in the returned map are reflected in this map, and vice-versa. The returned map supports all optional map operations that this map supports.

The returned map will throw an IllegalArgumentException on an attempt to insert a key outside its range.

**Specified by:**[headMap](http://docs.google.com/java/util/NavigableMap.html#headMap(K,%20boolean)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**toKey - high endpoint of the keys in the returned mapinclusive - true if the high endpoint is to be included in the returned view **Returns:**a view of the portion of this map whose keys are less than (or equal to, if inclusive is true) toKey **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if toKey is not compatible with this map's comparator (or, if the map has no comparator, if toKey does not implement [Comparable](http://docs.google.com/java/lang/Comparable.html)). Implementations may, but are not required to, throw this exception if toKey cannot be compared to keys currently in the map. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if toKey is null and this map uses natural ordering, or its comparator does not permit null keys [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if this map itself has a restricted range, and toKey lies outside the bounds of the range**Since:** 1.6

### tailMap

public [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **tailMap**([K](http://docs.google.com/java/util/TreeMap.html) fromKey,  
 boolean inclusive)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#tailMap(K,%20boolean)) Returns a view of the portion of this map whose keys are greater than (or equal to, if inclusive is true) fromKey. The returned map is backed by this map, so changes in the returned map are reflected in this map, and vice-versa. The returned map supports all optional map operations that this map supports.

The returned map will throw an IllegalArgumentException on an attempt to insert a key outside its range.

**Specified by:**[tailMap](http://docs.google.com/java/util/NavigableMap.html#tailMap(K,%20boolean)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**fromKey - low endpoint of the keys in the returned mapinclusive - true if the low endpoint is to be included in the returned view **Returns:**a view of the portion of this map whose keys are greater than (or equal to, if inclusive is true) fromKey **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if fromKey is not compatible with this map's comparator (or, if the map has no comparator, if fromKey does not implement [Comparable](http://docs.google.com/java/lang/Comparable.html)). Implementations may, but are not required to, throw this exception if fromKey cannot be compared to keys currently in the map. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if fromKey is null and this map uses natural ordering, or its comparator does not permit null keys [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if this map itself has a restricted range, and fromKey lies outside the bounds of the range**Since:** 1.6

### subMap

public [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **subMap**([K](http://docs.google.com/java/util/TreeMap.html) fromKey,  
 [K](http://docs.google.com/java/util/TreeMap.html) toKey)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#subMap(K,%20K)) Returns a view of the portion of this map whose keys range from fromKey, inclusive, to toKey, exclusive. (If fromKey and toKey are equal, the returned map is empty.) The returned map is backed by this map, so changes in the returned map are reflected in this map, and vice-versa. The returned map supports all optional map operations that this map supports.

The returned map will throw an IllegalArgumentException on an attempt to insert a key outside its range.

Equivalent to subMap(fromKey, true, toKey, false).

**Specified by:**[subMap](http://docs.google.com/java/util/NavigableMap.html#subMap(K,%20K)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Specified by:**[subMap](http://docs.google.com/java/util/SortedMap.html#subMap(K,%20K)) in interface [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**fromKey - low endpoint (inclusive) of the keys in the returned maptoKey - high endpoint (exclusive) of the keys in the returned map **Returns:**a view of the portion of this map whose keys range from fromKey, inclusive, to toKey, exclusive **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if fromKey and toKey cannot be compared to one another using this map's comparator (or, if the map has no comparator, using natural ordering). Implementations may, but are not required to, throw this exception if fromKey or toKey cannot be compared to keys currently in the map. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if fromKey or toKey is null and this map uses natural ordering, or its comparator does not permit null keys [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if fromKey is greater than toKey; or if this map itself has a restricted range, and fromKey or toKey lies outside the bounds of the range

### headMap

public [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **headMap**([K](http://docs.google.com/java/util/TreeMap.html) toKey)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#headMap(K)) Returns a view of the portion of this map whose keys are strictly less than toKey. The returned map is backed by this map, so changes in the returned map are reflected in this map, and vice-versa. The returned map supports all optional map operations that this map supports.

The returned map will throw an IllegalArgumentException on an attempt to insert a key outside its range.

Equivalent to headMap(toKey, false).

**Specified by:**[headMap](http://docs.google.com/java/util/NavigableMap.html#headMap(K)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Specified by:**[headMap](http://docs.google.com/java/util/SortedMap.html#headMap(K)) in interface [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**toKey - high endpoint (exclusive) of the keys in the returned map **Returns:**a view of the portion of this map whose keys are strictly less than toKey **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if toKey is not compatible with this map's comparator (or, if the map has no comparator, if toKey does not implement [Comparable](http://docs.google.com/java/lang/Comparable.html)). Implementations may, but are not required to, throw this exception if toKey cannot be compared to keys currently in the map. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if toKey is null and this map uses natural ordering, or its comparator does not permit null keys [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if this map itself has a restricted range, and toKey lies outside the bounds of the range

### tailMap

public [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **tailMap**([K](http://docs.google.com/java/util/TreeMap.html) fromKey)

**Description copied from interface:** [**NavigableMap**](http://docs.google.com/java/util/NavigableMap.html#tailMap(K)) Returns a view of the portion of this map whose keys are greater than or equal to fromKey. The returned map is backed by this map, so changes in the returned map are reflected in this map, and vice-versa. The returned map supports all optional map operations that this map supports.

The returned map will throw an IllegalArgumentException on an attempt to insert a key outside its range.

Equivalent to tailMap(fromKey, true).

**Specified by:**[tailMap](http://docs.google.com/java/util/NavigableMap.html#tailMap(K)) in interface [NavigableMap](http://docs.google.com/java/util/NavigableMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)>**Specified by:**[tailMap](http://docs.google.com/java/util/SortedMap.html#tailMap(K)) in interface [SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/TreeMap.html),[V](http://docs.google.com/java/util/TreeMap.html)> **Parameters:**fromKey - low endpoint (inclusive) of the keys in the returned map **Returns:**a view of the portion of this map whose keys are greater than or equal to fromKey **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if fromKey is not compatible with this map's comparator (or, if the map has no comparator, if fromKey does not implement [Comparable](http://docs.google.com/java/lang/Comparable.html)). Implementations may, but are not required to, throw this exception if fromKey cannot be compared to keys currently in the map. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if fromKey is null and this map uses natural ordering, or its comparator does not permit null keys [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if this map itself has a restricted range, and fromKey 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/TreeMap.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/TooManyListenersException.html)   [**NEXT CLASS**](http://docs.google.com/java/util/TreeSet.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/TreeMap.html)    [**NO FRAMES**](http://docs.google.com/TreeMap.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | FIELD | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: FIELD | [CONSTR](#17dp8vu) | [METHOD](#1ksv4uv) |

[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).