| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ConcurrentSkipListMap.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/concurrent/ConcurrentNavigableMap.html)   [**NEXT CLASS**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListSet.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/concurrent/ConcurrentSkipListMap.html)    [**NO FRAMES**](http://docs.google.com/ConcurrentSkipListMap.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | FIELD | [CONSTR](#3dy6vkm) | [METHOD](#1t3h5sf) | DETAIL: FIELD | [CONSTR](#3rdcrjn) | [METHOD](#44sinio) |

## **java.util.concurrent**

Class ConcurrentSkipListMap<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.concurrent.ConcurrentSkipListMap<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), [ConcurrentMap](http://docs.google.com/java/util/concurrent/ConcurrentMap.html)<K,V>, [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<K,V>, [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 **ConcurrentSkipListMap<K,V>**extends [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<K,V>implements [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<K,V>, [Cloneable](http://docs.google.com/java/lang/Cloneable.html), [Serializable](http://docs.google.com/java/io/Serializable.html)

A scalable concurrent [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.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 class implements a concurrent variant of [SkipLists](http://www.cs.umd.edu/~pugh/) providing expected average *log(n)* time cost for the containsKey, get, put and remove operations and their variants. Insertion, removal, update, and access operations safely execute concurrently by multiple threads. Iterators are *weakly consistent*, returning elements reflecting the state of the map at some point at or since the creation of the iterator. They do *not* throw [ConcurrentModificationException](http://docs.google.com/java/util/ConcurrentModificationException.html), and may proceed concurrently with other operations. Ascending key ordered views and their iterators are faster than descending ones.

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, putIfAbsent, or replace, depending on exactly which effect you need.)

Beware that, unlike in most collections, the size method is *not* a constant-time operation. Because of the asynchronous nature of these maps, determining the current number of elements requires a traversal of the elements. Additionally, the bulk operations putAll, equals, and clear are *not* guaranteed to be performed atomically. For example, an iterator operating concurrently with a putAll operation might view only some of the added elements.

This class and its views and iterators implement all of the *optional* methods of the [Map](http://docs.google.com/java/util/Map.html) and [Iterator](http://docs.google.com/java/util/Iterator.html) interfaces. Like most other concurrent collections, this class does *not* permit the use of null keys or values because some null return values cannot be reliably distinguished from the absence of elements.

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

**Since:** 1.6 **See Also:**[Serialized Form](http://docs.google.com/serialized-form.html#java.util.concurrent.ConcurrentSkipListMap)

| **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)> |

| **Nested classes/interfaces inherited from interface java.util.**[**Map**](http://docs.google.com/java/util/Map.html) |
| --- |
| [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/Map.Entry.html),[V](http://docs.google.com/java/util/Map.Entry.html)> |

| **Constructor Summary** | |
| --- | --- |
| [**ConcurrentSkipListMap**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#ConcurrentSkipListMap())()            Constructs a new, empty map, sorted according to the [natural ordering](http://docs.google.com/java/lang/Comparable.html) of the keys. |
| [**ConcurrentSkipListMap**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#ConcurrentSkipListMap(java.util.Comparator))([Comparator](http://docs.google.com/java/util/Comparator.html)<? super [K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> comparator)            Constructs a new, empty map, sorted according to the specified comparator. |
| [**ConcurrentSkipListMap**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#ConcurrentSkipListMap(java.util.Map))([Map](http://docs.google.com/java/util/Map.html)<? extends [K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),? extends [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> m)            Constructs a new map containing the same mappings as the given map, sorted according to the [natural ordering](http://docs.google.com/java/lang/Comparable.html) of the keys. |
| [**ConcurrentSkipListMap**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#ConcurrentSkipListMap(java.util.SortedMap))([SortedMap](http://docs.google.com/java/util/SortedMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),? extends [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> m)            Constructs a new 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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**ceilingEntry**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#ceilingEntry(K))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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 entry. |
| [K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) | [**ceilingKey**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#ceilingKey(K))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html#clear())()            Removes all of the mappings from this map. |
| [ConcurrentSkipListMap](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**clone**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#clone())()            Returns a shallow copy of this ConcurrentSkipListMap instance. |
| [Comparator](http://docs.google.com/java/util/Comparator.html)<? super [K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**comparator**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html)> | [**descendingKeySet**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#descendingKeySet())()            Returns a reverse order [NavigableSet](http://docs.google.com/java/util/NavigableSet.html) view of the keys contained in this map. |
| [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**descendingMap**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)>> | [**entrySet**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#entrySet())()            Returns a [Set](http://docs.google.com/java/util/Set.html) view of the mappings contained in this map. |
| boolean | [**equals**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#equals(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) o)            Compares the specified object with this map for equality. |
| [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**firstEntry**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html) | [**firstKey**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**floorEntry**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#floorEntry(K))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html) | [**floorKey**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#floorKey(K))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html) | [**get**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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. |
| [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**headMap**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#headMap(K))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) toKey)            Returns a view of the portion of this map whose keys are strictly less than toKey. |
| [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**headMap**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#headMap(K,%20boolean))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**higherEntry**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#higherEntry(K))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html) | [**higherKey**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#higherKey(K))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) key)            Returns the least key strictly greater than the given key, or null if there is no such key. |
| boolean | [**isEmpty**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#isEmpty())()            Returns true if this map contains no key-value mappings. |
| [NavigableSet](http://docs.google.com/java/util/NavigableSet.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**keySet**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#keySet())()            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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**lastEntry**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html) | [**lastKey**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**lowerEntry**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#lowerEntry(K))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html) | [**lowerKey**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#lowerKey(K))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html)> | [**navigableKeySet**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**pollFirstEntry**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**pollLastEntry**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html) | [**put**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#put(K,%20V))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) key, [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) value)            Associates the specified value with the specified key in this map. |
| [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) | [**putIfAbsent**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#putIfAbsent(K,%20V))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) key, [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) value)            If the specified key is not already associated with a value, associate it with the given value. |
| [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) | [**remove**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#remove(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) key)            Removes the mapping for the specified key from this map if present. |
| boolean | [**remove**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#remove(java.lang.Object,%20java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) key, [Object](http://docs.google.com/java/lang/Object.html) value)            Removes the entry for a key only if currently mapped to a given value. |
| [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) | [**replace**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#replace(K,%20V))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) key, [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) value)            Replaces the entry for a key only if currently mapped to some value. |
| boolean | [**replace**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#replace(K,%20V,%20V))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) key, [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) oldValue, [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) newValue)            Replaces the entry for a key only if currently mapped to a given value. |
| int | [**size**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#size())()            Returns the number of key-value mappings in this map. |
| [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**subMap**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#subMap(K,%20boolean,%20K,%20boolean))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) fromKey, boolean fromInclusive, [K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) toKey, boolean toInclusive)            Returns a view of the portion of this map whose keys range from fromKey to toKey. |
| [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**subMap**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#subMap(K,%20K))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) fromKey, [K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) toKey)            Returns a view of the portion of this map whose keys range from fromKey, inclusive, to toKey, exclusive. |
| [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**tailMap**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#tailMap(K))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) fromKey)            Returns a view of the portion of this map whose keys are greater than or equal to fromKey. |
| [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> | [**tailMap**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html#tailMap(K,%20boolean))([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html)> | [**values**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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) |
| --- |
| [hashCode](http://docs.google.com/java/util/AbstractMap.html#hashCode()), [putAll](http://docs.google.com/java/util/AbstractMap.html#putAll(java.util.Map)), [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) |
| --- |
| [hashCode](http://docs.google.com/java/util/Map.html#hashCode()), [putAll](http://docs.google.com/java/util/Map.html#putAll(java.util.Map)) |

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

### ConcurrentSkipListMap

public **ConcurrentSkipListMap**()

Constructs a new, empty map, sorted according to the [natural ordering](http://docs.google.com/java/lang/Comparable.html) of the keys.

### ConcurrentSkipListMap

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

Constructs a new, empty map, sorted according to the specified comparator.

**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.

### ConcurrentSkipListMap

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

Constructs a new map containing the same mappings as the given map, sorted according to the [natural ordering](http://docs.google.com/java/lang/Comparable.html) of the keys.

**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 or any of its keys or values are null

### ConcurrentSkipListMap

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

Constructs a new map containing the same mappings and using the same ordering as the specified sorted map.

**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 sorted map or any of its keys or values are null

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

### clone

public [ConcurrentSkipListMap](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **clone**()

Returns a shallow copy of this ConcurrentSkipListMap 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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Returns:**a shallow copy of this map**See Also:**[Cloneable](http://docs.google.com/java/lang/Cloneable.html)

### 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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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

### get

public [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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.)

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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

### put

public [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) **put**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) key,  
 [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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 the specified key, or null if there was 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 or value is null

### remove

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

Removes the mapping for the specified key from this map 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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Parameters:**key - key for which mapping should be removed **Returns:**the previous value associated with the specified key, or null if there was 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

### 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. This operation requires time linear in the map size.

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Parameters:**value - value whose presence in this map is to be tested **Returns:**true if a mapping to value exists; false otherwise **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified value is null

### size

public int **size**()

Returns the number of key-value mappings in this map. If this map contains more than Integer.MAX\_VALUE elements, it returns Integer.MAX\_VALUE.

Beware that, unlike in most collections, this method is *NOT* a constant-time operation. Because of the asynchronous nature of these maps, determining the current number of elements requires traversing them all to count them. Additionally, it is possible for the size to change during execution of this method, in which case the returned result will be inaccurate. Thus, this method is typically not very useful in concurrent applications.

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Returns:**the number of elements in this map

### isEmpty

public boolean **isEmpty**()

Returns true if this map contains no key-value mappings.

**Specified by:**[isEmpty](http://docs.google.com/java/util/Map.html#isEmpty()) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)>**Overrides:**[isEmpty](http://docs.google.com/java/util/AbstractMap.html#isEmpty()) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Returns:**true if this map contains no key-value mappings

### clear

public void **clear**()

Removes all of the mappings from this map.

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)>

### keySet

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

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. 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.

The view's iterator is a "weakly consistent" iterator that will never throw [ConcurrentModificationException](http://docs.google.com/java/util/ConcurrentModificationException.html), and guarantees to traverse elements as they existed upon construction of the iterator, and may (but is not guaranteed to) reflect any modifications subsequent to construction.

This method is equivalent to method navigableKeySet.

**Specified by:**[keySet](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html#keySet()) in interface [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)>**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Returns:**a navigable set view of the keys in this map

### navigableKeySet

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

**Description copied from interface:** [**ConcurrentNavigableMap**](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.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. 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.

The view's iterator is a "weakly consistent" iterator that will never throw [ConcurrentModificationException](http://docs.google.com/java/util/ConcurrentModificationException.html), and guarantees to traverse elements as they existed upon construction of the iterator, and may (but is not guaranteed to) reflect any modifications subsequent to construction.

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

### values

public [Collection](http://docs.google.com/java/util/Collection.html)<[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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. 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.

The view's iterator is a "weakly consistent" iterator that will never throw [ConcurrentModificationException](http://docs.google.com/java/util/ConcurrentModificationException.html), and guarantees to traverse elements as they existed upon construction of the iterator, and may (but is not guaranteed to) reflect any modifications subsequent to construction.

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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. 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.

The view's iterator is a "weakly consistent" iterator that will never throw [ConcurrentModificationException](http://docs.google.com/java/util/ConcurrentModificationException.html), and guarantees to traverse elements as they existed upon construction of the iterator, and may (but is not guaranteed to) reflect any modifications subsequent to construction.

The Map.Entry elements returned by iterator.next() do *not* support the setValue operation.

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Returns:**a set view of the mappings contained in this map, sorted in ascending key order

### descendingMap

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

**Description copied from interface:** [**ConcurrentNavigableMap**](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.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.

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/concurrent/ConcurrentNavigableMap.html#descendingMap()) in interface [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)>**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Returns:**a reverse order view of this map

### descendingKeySet

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

**Description copied from interface:** [**ConcurrentNavigableMap**](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.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. 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.

The view's iterator is a "weakly consistent" iterator that will never throw [ConcurrentModificationException](http://docs.google.com/java/util/ConcurrentModificationException.html), and guarantees to traverse elements as they existed upon construction of the iterator, and may (but is not guaranteed to) reflect any modifications subsequent to construction.

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

### equals

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

Compares the specified object with this map for equality. Returns true if the given object is also a map and the two maps represent the same mappings. More formally, two maps m1 and m2 represent the same mappings if m1.entrySet().equals(m2.entrySet()). This operation may return misleading results if either map is concurrently modified during execution of this method.

**Specified by:**[equals](http://docs.google.com/java/util/Map.html#equals(java.lang.Object)) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)>**Overrides:**[equals](http://docs.google.com/java/util/AbstractMap.html#equals(java.lang.Object)) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Parameters:**o - object to be compared for equality with this map **Returns:**true if the specified object is equal to this map**See Also:**[Object.hashCode()](http://docs.google.com/java/lang/Object.html#hashCode()), [Hashtable](http://docs.google.com/java/util/Hashtable.html)

### putIfAbsent

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

If the specified key is not already associated with a value, associate it with the given value. This is equivalent to

if (!map.containsKey(key))  
 return map.put(key, value);  
 else  
 return map.get(key);

except that the action is performed atomically.

**Specified by:**[putIfAbsent](http://docs.google.com/java/util/concurrent/ConcurrentMap.html#putIfAbsent(K,%20V)) in interface [ConcurrentMap](http://docs.google.com/java/util/concurrent/ConcurrentMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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 the specified key, or null if there was 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 or value is null

### remove

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

Removes the entry for a key only if currently mapped to a given value. This is equivalent to

if (map.containsKey(key) && map.get(key).equals(value)) {  
 map.remove(key);  
 return true;  
 } else return false;

except that the action is performed atomically.

**Specified by:**[remove](http://docs.google.com/java/util/concurrent/ConcurrentMap.html#remove(java.lang.Object,%20java.lang.Object)) in interface [ConcurrentMap](http://docs.google.com/java/util/concurrent/ConcurrentMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Parameters:**key - key with which the specified value is associatedvalue - value expected to be associated with the specified key **Returns:**true if the value was removed **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

### replace

public boolean **replace**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) key,  
 [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) oldValue,  
 [V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) newValue)

Replaces the entry for a key only if currently mapped to a given value. This is equivalent to

if (map.containsKey(key) && map.get(key).equals(oldValue)) {  
 map.put(key, newValue);  
 return true;  
 } else return false;

except that the action is performed atomically.

**Specified by:**[replace](http://docs.google.com/java/util/concurrent/ConcurrentMap.html#replace(K,%20V,%20V)) in interface [ConcurrentMap](http://docs.google.com/java/util/concurrent/ConcurrentMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Parameters:**key - key with which the specified value is associatedoldValue - value expected to be associated with the specified keynewValue - value to be associated with the specified key **Returns:**true if the value was replaced **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 any of the arguments are null

### replace

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

Replaces the entry for a key only if currently mapped to some value. This is equivalent to

if (map.containsKey(key)) {  
 return map.put(key, value);  
 } else return null;

except that the action is performed atomically.

**Specified by:**[replace](http://docs.google.com/java/util/concurrent/ConcurrentMap.html#replace(K,%20V)) in interface [ConcurrentMap](http://docs.google.com/java/util/concurrent/ConcurrentMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Parameters:**key - key with which the specified value is associatedvalue - value to be associated with the specified key **Returns:**the previous value associated with the specified key, or null if there was 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 or value is null

### comparator

public [Comparator](http://docs.google.com/java/util/Comparator.html)<? super [K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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

### subMap

public [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **subMap**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) fromKey,  
 boolean fromInclusive,  
 [K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentNavigableMap.html#subMap(K,%20boolean,%20K,%20boolean)) in interface [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)>**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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 [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 [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **headMap**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentNavigableMap.html#headMap(K,%20boolean)) in interface [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)>**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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 [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 [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **tailMap**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentNavigableMap.html#tailMap(K,%20boolean)) in interface [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)>**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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 [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

### subMap

public [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **subMap**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) fromKey,  
 [K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentNavigableMap.html#subMap(K,%20K)) in interface [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)>**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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 [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 [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **headMap**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentNavigableMap.html#headMap(K)) in interface [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)>**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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 [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 [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **tailMap**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentNavigableMap.html#tailMap(K)) in interface [ConcurrentNavigableMap](http://docs.google.com/java/util/concurrent/ConcurrentNavigableMap.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)>**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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 [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

### lowerEntry

public [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **lowerEntry**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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. The returned entry does *not* support the Entry.setValue method.

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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

### lowerKey

public [K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) **lowerKey**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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

### floorEntry

public [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **floorEntry**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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. The returned entry does *not* support the Entry.setValue method.

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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

### floorKey

public [K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) **floorKey**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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

### ceilingEntry

public [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **ceilingEntry**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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 entry. The returned entry does *not* support the Entry.setValue method.

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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

### ceilingKey

public [K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) **ceilingKey**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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

### higherEntry

public [Map.Entry](http://docs.google.com/java/util/Map.Entry.html)<[K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **higherEntry**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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. The returned entry does *not* support the Entry.setValue method.

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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

### higherKey

public [K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html) **higherKey**([K](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.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

### firstEntry

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

Returns a key-value mapping associated with the least key in this map, or null if the map is empty. The returned entry does *not* support the Entry.setValue method.

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Returns:**an entry with the least key, or null if this map is empty

### lastEntry

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

Returns a key-value mapping associated with the greatest key in this map, or null if the map is empty. The returned entry does *not* support the Entry.setValue method.

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Returns:**an entry with the greatest key, or null if this map is empty

### pollFirstEntry

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

Removes and returns a key-value mapping associated with the least key in this map, or null if the map is empty. The returned entry does *not* support the Entry.setValue method.

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Returns:**the removed first entry of this map, or null if this map is empty

### pollLastEntry

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

Removes and returns a key-value mapping associated with the greatest key in this map, or null if the map is empty. The returned entry does *not* support the Entry.setValue method.

**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/concurrent/ConcurrentSkipListMap.html),[V](http://docs.google.com/java/util/concurrent/ConcurrentSkipListMap.html)> **Returns:**the removed last entry of this map, or null if this map is empty

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ConcurrentSkipListMap.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/concurrent/ConcurrentNavigableMap.html)   [**NEXT CLASS**](http://docs.google.com/java/util/concurrent/ConcurrentSkipListSet.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/concurrent/ConcurrentSkipListMap.html)    [**NO FRAMES**](http://docs.google.com/ConcurrentSkipListMap.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | FIELD | [CONSTR](#3dy6vkm) | [METHOD](#1t3h5sf) | DETAIL: FIELD | [CONSTR](#3rdcrjn) | [METHOD](#44sinio) |

[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).