| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/IdentityHashMap.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/Hashtable.html)   [**NEXT CLASS**](http://docs.google.com/java/util/IllegalFormatCodePointException.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/IdentityHashMap.html)    [**NO FRAMES**](http://docs.google.com/IdentityHashMap.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | FIELD | [CONSTR](#3dy6vkm) | [METHOD](#1t3h5sf) | DETAIL: FIELD | [CONSTR](#17dp8vu) | [METHOD](#35nkun2) |

## **java.util**

Class IdentityHashMap<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.IdentityHashMap<K,V>**

**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>

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

This class implements the Map interface with a hash table, using reference-equality in place of object-equality when comparing keys (and values). In other words, in an IdentityHashMap, two keys k1 and k2 are considered equal if and only if (k1==k2). (In normal Map implementations (like HashMap) two keys k1 and k2 are considered equal if and only if (k1==null ? k2==null : k1.equals(k2)).)

**This class is *not* a general-purpose Map implementation! While this class implements the Map interface, it intentionally violates Map's general contract, which mandates the use of the equals method when comparing objects. This class is designed for use only in the rare cases wherein reference-equality semantics are required.**

A typical use of this class is *topology-preserving object graph transformations*, such as serialization or deep-copying. To perform such a transformation, a program must maintain a "node table" that keeps track of all the object references that have already been processed. The node table must not equate distinct objects even if they happen to be equal. Another typical use of this class is to maintain *proxy objects*. For example, a debugging facility might wish to maintain a proxy object for each object in the program being debugged.

This class provides all of the optional map operations, and permits null values and the null key. This class makes no guarantees as to the order of the map; in particular, it does not guarantee that the order will remain constant over time.

This class provides constant-time performance for the basic operations (get and put), assuming the system identity hash function ([System.identityHashCode(Object)](http://docs.google.com/java/lang/System.html#identityHashCode(java.lang.Object))) disperses elements properly among the buckets.

This class has one tuning parameter (which affects performance but not semantics): *expected maximum size*. This parameter is the maximum number of key-value mappings that the map is expected to hold. Internally, this parameter is used to determine the number of buckets initially comprising the hash table. The precise relationship between the expected maximum size and the number of buckets is unspecified.

If the size of the map (the number of key-value mappings) sufficiently exceeds the expected maximum size, the number of buckets is increased Increasing the number of buckets ("rehashing") may be fairly expensive, so it pays to create identity hash maps with a sufficiently large expected maximum size. On the other hand, iteration over collection views requires time proportional to the number of buckets in the hash table, so it pays not to set the expected maximum size too high if you are especially concerned with iteration performance or memory usage.

**Note that this implementation is not synchronized.** If multiple threads access an identity hash 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 a key that an instance already contains 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.synchronizedMap](http://docs.google.com/java/util/Collections.html#synchronizedMap(java.util.Map)) method. This is best done at creation time, to prevent accidental unsynchronized access to the map:

Map m = Collections.synchronizedMap(new IdentityHashMap(...));

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: *fail-fast iterators should be used only to detect bugs.*

Implementation note: This is a simple *linear-probe* hash table, as described for example in texts by Sedgewick and Knuth. The array alternates holding keys and values. (This has better locality for large tables than does using separate arrays.) For many JRE implementations and operation mixes, this class will yield better performance than [HashMap](http://docs.google.com/java/util/HashMap.html) (which uses *chaining* rather than linear-probing).

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

**Since:** 1.4 **See Also:**[System.identityHashCode(Object)](http://docs.google.com/java/lang/System.html#identityHashCode(java.lang.Object)), [Object.hashCode()](http://docs.google.com/java/lang/Object.html#hashCode()), [Collection](http://docs.google.com/java/util/Collection.html), [Map](http://docs.google.com/java/util/Map.html), [HashMap](http://docs.google.com/java/util/HashMap.html), [TreeMap](http://docs.google.com/java/util/TreeMap.html), [Serialized Form](http://docs.google.com/serialized-form.html#java.util.IdentityHashMap)

| **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** | |
| --- | --- |
| [**IdentityHashMap**](http://docs.google.com/java/util/IdentityHashMap.html#IdentityHashMap())()            Constructs a new, empty identity hash map with a default expected maximum size (21). |
| [**IdentityHashMap**](http://docs.google.com/java/util/IdentityHashMap.html#IdentityHashMap(int))(int expectedMaxSize)            Constructs a new, empty map with the specified expected maximum size. |
| [**IdentityHashMap**](http://docs.google.com/java/util/IdentityHashMap.html#IdentityHashMap(java.util.Map))([Map](http://docs.google.com/java/util/Map.html)<? extends [K](http://docs.google.com/java/util/IdentityHashMap.html),? extends [V](http://docs.google.com/java/util/IdentityHashMap.html)> m)            Constructs a new identity hash map containing the keys-value mappings in the specified map. |

| **Method Summary** | |
| --- | --- |
| void | [**clear**](http://docs.google.com/java/util/IdentityHashMap.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/IdentityHashMap.html#clone())()            Returns a shallow copy of this identity hash map: the keys and values themselves are not cloned. |
| boolean | [**containsKey**](http://docs.google.com/java/util/IdentityHashMap.html#containsKey(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) key)            Tests whether the specified object reference is a key in this identity hash map. |
| boolean | [**containsValue**](http://docs.google.com/java/util/IdentityHashMap.html#containsValue(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) value)            Tests whether the specified object reference is a value in this identity hash 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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.html)>> | [**entrySet**](http://docs.google.com/java/util/IdentityHashMap.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/IdentityHashMap.html#equals(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) o)            Compares the specified object with this map for equality. |
| [V](http://docs.google.com/java/util/IdentityHashMap.html) | [**get**](http://docs.google.com/java/util/IdentityHashMap.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. |
| int | [**hashCode**](http://docs.google.com/java/util/IdentityHashMap.html#hashCode())()            Returns the hash code value for this map. |
| boolean | [**isEmpty**](http://docs.google.com/java/util/IdentityHashMap.html#isEmpty())()            Returns true if this identity hash map contains no key-value mappings. |
| [Set](http://docs.google.com/java/util/Set.html)<[K](http://docs.google.com/java/util/IdentityHashMap.html)> | [**keySet**](http://docs.google.com/java/util/IdentityHashMap.html#keySet())()            Returns an identity-based set view of the keys contained in this map. |
| [V](http://docs.google.com/java/util/IdentityHashMap.html) | [**put**](http://docs.google.com/java/util/IdentityHashMap.html#put(K,%20V))([K](http://docs.google.com/java/util/IdentityHashMap.html) key, [V](http://docs.google.com/java/util/IdentityHashMap.html) value)            Associates the specified value with the specified key in this identity hash map. |
| void | [**putAll**](http://docs.google.com/java/util/IdentityHashMap.html#putAll(java.util.Map))([Map](http://docs.google.com/java/util/Map.html)<? extends [K](http://docs.google.com/java/util/IdentityHashMap.html),? extends [V](http://docs.google.com/java/util/IdentityHashMap.html)> m)            Copies all of the mappings from the specified map to this map. |
| [V](http://docs.google.com/java/util/IdentityHashMap.html) | [**remove**](http://docs.google.com/java/util/IdentityHashMap.html#remove(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) key)            Removes the mapping for this key from this map if present. |
| int | [**size**](http://docs.google.com/java/util/IdentityHashMap.html#size())()            Returns the number of key-value mappings in this identity hash map. |
| [Collection](http://docs.google.com/java/util/Collection.html)<[V](http://docs.google.com/java/util/IdentityHashMap.html)> | [**values**](http://docs.google.com/java/util/IdentityHashMap.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) |
| --- |
| [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)) |

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

### IdentityHashMap

public **IdentityHashMap**()

Constructs a new, empty identity hash map with a default expected maximum size (21).

### IdentityHashMap

public **IdentityHashMap**(int expectedMaxSize)

Constructs a new, empty map with the specified expected maximum size. Putting more than the expected number of key-value mappings into the map may cause the internal data structure to grow, which may be somewhat time-consuming.

**Parameters:**expectedMaxSize - the expected maximum size of the map **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if expectedMaxSize is negative

### IdentityHashMap

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

Constructs a new identity hash map containing the keys-value mappings in the specified map.

**Parameters:**m - the map whose mappings are to be placed into 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 identity hash 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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.html)> **Returns:**the number of key-value mappings in this map

### isEmpty

public boolean **isEmpty**()

Returns true if this identity hash 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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.html)> **Returns:**true if this identity hash map contains no key-value mappings

### get

public [V](http://docs.google.com/java/util/IdentityHashMap.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 == k), 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/IdentityHashMap.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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.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**See Also:**[put(Object, Object)](http://docs.google.com/java/util/IdentityHashMap.html#put(K,%20V))

### containsKey

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

Tests whether the specified object reference is a key in this identity hash map.

**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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.html)> **Parameters:**key - possible key **Returns:**true if the specified object reference is a key in this map**See Also:**[containsValue(Object)](http://docs.google.com/java/util/IdentityHashMap.html#containsValue(java.lang.Object))

### containsValue

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

Tests whether the specified object reference is a value in this identity hash map.

**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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.html)> **Parameters:**value - value whose presence in this map is to be tested **Returns:**true if this map maps one or more keys to the specified object reference**See Also:**[containsKey(Object)](http://docs.google.com/java/util/IdentityHashMap.html#containsKey(java.lang.Object))

### put

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

Associates the specified value with the specified key in this identity hash 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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.html)> **Parameters:**key - the key with which the specified value is to be associatedvalue - the 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.)**See Also:**[Object.equals(Object)](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [get(Object)](http://docs.google.com/java/util/IdentityHashMap.html#get(java.lang.Object)), [containsKey(Object)](http://docs.google.com/java/util/IdentityHashMap.html#containsKey(java.lang.Object))

### putAll

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

Copies all of the mappings from the specified map to this map. These mappings will 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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.html)> **Parameters:**m - mappings to be stored in this map **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the specified map is null

### remove

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

Removes the mapping for this 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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.html)> **Parameters:**key - key whose mapping is to be removed from the map **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.)

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

### 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 identical object-reference mappings. More formally, this map is equal to another map m if and only if this.entrySet().equals(m.entrySet()).

**Owing to the reference-equality-based semantics of this map it is possible that the symmetry and transitivity requirements of the Object.equals contract may be violated if this map is compared to a normal map. However, the Object.equals contract is guaranteed to hold among IdentityHashMap instances.**

**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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.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.equals(Object)](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object))

### hashCode

public int **hashCode**()

Returns the hash code value for this map. The hash code of a map is defined to be the sum of the hash codes of each entry in the map's entrySet() view. This ensures that m1.equals(m2) implies that m1.hashCode()==m2.hashCode() for any two IdentityHashMap instances m1 and m2, as required by the general contract of [Object.hashCode()](http://docs.google.com/java/lang/Object.html#hashCode()).

**Owing to the reference-equality-based semantics of the Map.Entry instances in the set returned by this map's entrySet method, it is possible that the contractual requirement of Object.hashCode mentioned in the previous paragraph will be violated if one of the two objects being compared is an IdentityHashMap instance and the other is a normal map.**

**Specified by:**[hashCode](http://docs.google.com/java/util/Map.html#hashCode()) in interface [Map](http://docs.google.com/java/util/Map.html)<[K](http://docs.google.com/java/util/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.html)>**Overrides:**[hashCode](http://docs.google.com/java/util/AbstractMap.html#hashCode()) in class [AbstractMap](http://docs.google.com/java/util/AbstractMap.html)<[K](http://docs.google.com/java/util/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.html)> **Returns:**the hash code value for this map**See Also:**[Object.equals(Object)](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [equals(Object)](http://docs.google.com/java/util/IdentityHashMap.html#equals(java.lang.Object))

### clone

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

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

### keySet

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

Returns an identity-based set view of the keys contained in this map. 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, 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 methods. It does not support the add or addAll methods.

**While the object returned by this method implements the Set interface, it does *not* obey Set's general contract. Like its backing map, the set returned by this method defines element equality as reference-equality rather than object-equality. This affects the behavior of its contains, remove, containsAll, equals, and hashCode methods.**

**The equals method of the returned set returns true only if the specified object is a set containing exactly the same object references as the returned set. The symmetry and transitivity requirements of the Object.equals contract may be violated if the set returned by this method is compared to a normal set. However, the Object.equals contract is guaranteed to hold among sets returned by this method.**

The hashCode method of the returned set returns the sum of the *identity hashcodes* of the elements in the set, rather than the sum of their hashcodes. This is mandated by the change in the semantics of the equals method, in order to enforce the general contract of the Object.hashCode method among sets returned by this method.

**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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.html)> **Returns:**an identity-based set view of the keys contained in this map**See Also:**[Object.equals(Object)](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [System.identityHashCode(Object)](http://docs.google.com/java/lang/System.html#identityHashCode(java.lang.Object))

### values

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

Returns a [Collection](http://docs.google.com/java/util/Collection.html) view of the values contained in this map. 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, 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 methods. It does not support the add or addAll methods.

**While the object returned by this method implements the Collection interface, it does *not* obey Collection's general contract. Like its backing map, the collection returned by this method defines element equality as reference-equality rather than object-equality. This affects the behavior of its contains, remove and containsAll methods.**

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

Returns a [Set](http://docs.google.com/java/util/Set.html) view of the mappings contained in this map. Each element in the returned set is a reference-equality-based Map.Entry. 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, 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 methods. It does not support the add or addAll methods.

Like the backing map, the Map.Entry objects in the set returned by this method define key and value equality as reference-equality rather than object-equality. This affects the behavior of the equals and hashCode methods of these Map.Entry objects. A reference-equality based Map.Entry e is equal to an object o if and only if o is a Map.Entry and e.getKey()==o.getKey() && e.getValue()==o.getValue(). To accommodate these equals semantics, the hashCode method returns System.identityHashCode(e.getKey()) ^ System.identityHashCode(e.getValue()).

**Owing to the reference-equality-based semantics of the Map.Entry instances in the set returned by this method, it is possible that the symmetry and transitivity requirements of the** [**Object.equals(Object)**](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)) **contract may be violated if any of the entries in the set is compared to a normal map entry, or if the set returned by this method is compared to a set of normal map entries (such as would be returned by a call to this method on a normal map). However, the Object.equals contract is guaranteed to hold among identity-based map entries, and among sets of such entries.**

**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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.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/IdentityHashMap.html),[V](http://docs.google.com/java/util/IdentityHashMap.html)> **Returns:**a set view of the identity-mappings contained in this map

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/IdentityHashMap.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/Hashtable.html)   [**NEXT CLASS**](http://docs.google.com/java/util/IllegalFormatCodePointException.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/IdentityHashMap.html)    [**NO FRAMES**](http://docs.google.com/IdentityHashMap.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2et92p0) | FIELD | [CONSTR](#3dy6vkm) | [METHOD](#1t3h5sf) | DETAIL: FIELD | [CONSTR](#17dp8vu) | [METHOD](#35nkun2) |

[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).