| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/AtomicStampedReference.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/atomic/AtomicReferenceFieldUpdater.html)   NEXT CLASS | [**FRAMES**](http://docs.google.com/index.html?java/util/concurrent/atomic/AtomicStampedReference.html)    [**NO FRAMES**](http://docs.google.com/AtomicStampedReference.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#2s8eyo1) |

## **java.util.concurrent.atomic**

Class AtomicStampedReference<V>

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **java.util.concurrent.atomic.AtomicStampedReference<V>**

**Type Parameters:**V - The type of object referred to by this reference

public class **AtomicStampedReference<V>**extends [Object](http://docs.google.com/java/lang/Object.html)

An AtomicStampedReference maintains an object reference along with an integer "stamp", that can be updated atomically.

Implementation note. This implementation maintains stamped references by creating internal objects representing "boxed" [reference, integer] pairs.

**Since:** 1.5

| **Constructor Summary** | |
| --- | --- |
| [**AtomicStampedReference**](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html#AtomicStampedReference(V,%20int))([V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) initialRef, int initialStamp)            Creates a new AtomicStampedReference with the given initial values. |

| **Method Summary** | |
| --- | --- |
| boolean | [**attemptStamp**](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html#attemptStamp(V,%20int))([V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) expectedReference, int newStamp)            Atomically sets the value of the stamp to the given update value if the current reference is == to the expected reference. |
| boolean | [**compareAndSet**](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html#compareAndSet(V,%20V,%20int,%20int))([V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) expectedReference, [V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) newReference, int expectedStamp, int newStamp)            Atomically sets the value of both the reference and stamp to the given update values if the current reference is == to the expected reference and the current stamp is equal to the expected stamp. |
| [V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) | [**get**](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html#get(int%5B%5D))(int[] stampHolder)            Returns the current values of both the reference and the stamp. |
| [V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) | [**getReference**](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html#getReference())()            Returns the current value of the reference. |
| int | [**getStamp**](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html#getStamp())()            Returns the current value of the stamp. |
| void | [**set**](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html#set(V,%20int))([V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) newReference, int newStamp)            Unconditionally sets the value of both the reference and stamp. |
| boolean | [**weakCompareAndSet**](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html#weakCompareAndSet(V,%20V,%20int,%20int))([V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) expectedReference, [V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) newReference, int expectedStamp, int newStamp)            Atomically sets the value of both the reference and stamp to the given update values if the current reference is == to the expected reference and the current stamp is equal to the expected stamp. |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [clone](http://docs.google.com/java/lang/Object.html#clone()), [equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [hashCode](http://docs.google.com/java/lang/Object.html#hashCode()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [toString](http://docs.google.com/java/lang/Object.html#toString()), [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** |
| --- |

### AtomicStampedReference

public **AtomicStampedReference**([V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) initialRef,  
 int initialStamp)

Creates a new AtomicStampedReference with the given initial values.

**Parameters:**initialRef - the initial referenceinitialStamp - the initial stamp

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

### getReference

public [V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) **getReference**()

Returns the current value of the reference.

**Returns:**the current value of the reference

### getStamp

public int **getStamp**()

Returns the current value of the stamp.

**Returns:**the current value of the stamp

### get

public [V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) **get**(int[] stampHolder)

Returns the current values of both the reference and the stamp. Typical usage is int[1] holder; ref = v.get(holder); .

**Parameters:**stampHolder - an array of size of at least one. On return, stampholder[0] will hold the value of the stamp. **Returns:**the current value of the reference

### weakCompareAndSet

public boolean **weakCompareAndSet**([V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) expectedReference,  
 [V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) newReference,  
 int expectedStamp,  
 int newStamp)

Atomically sets the value of both the reference and stamp to the given update values if the current reference is == to the expected reference and the current stamp is equal to the expected stamp.

May [fail spuriously](http://docs.google.com/package-summary.html#Spurious) and does not provide ordering guarantees, so is only rarely an appropriate alternative to compareAndSet.

**Parameters:**expectedReference - the expected value of the referencenewReference - the new value for the referenceexpectedStamp - the expected value of the stampnewStamp - the new value for the stamp **Returns:**true if successful

### compareAndSet

public boolean **compareAndSet**([V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) expectedReference,  
 [V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) newReference,  
 int expectedStamp,  
 int newStamp)

Atomically sets the value of both the reference and stamp to the given update values if the current reference is == to the expected reference and the current stamp is equal to the expected stamp.

**Parameters:**expectedReference - the expected value of the referencenewReference - the new value for the referenceexpectedStamp - the expected value of the stampnewStamp - the new value for the stamp **Returns:**true if successful

### set

public void **set**([V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) newReference,  
 int newStamp)

Unconditionally sets the value of both the reference and stamp.

**Parameters:**newReference - the new value for the referencenewStamp - the new value for the stamp

### attemptStamp

public boolean **attemptStamp**([V](http://docs.google.com/java/util/concurrent/atomic/AtomicStampedReference.html) expectedReference,  
 int newStamp)

Atomically sets the value of the stamp to the given update value if the current reference is == to the expected reference. Any given invocation of this operation may fail (return false) spuriously, but repeated invocation when the current value holds the expected value and no other thread is also attempting to set the value will eventually succeed.

**Parameters:**expectedReference - the expected value of the referencenewStamp - the new value for the stamp **Returns:**true if successful

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/AtomicStampedReference.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/atomic/AtomicReferenceFieldUpdater.html)   NEXT CLASS | [**FRAMES**](http://docs.google.com/index.html?java/util/concurrent/atomic/AtomicStampedReference.html)    [**NO FRAMES**](http://docs.google.com/AtomicStampedReference.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#2s8eyo1) |

[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).