| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/AtomicReferenceArray.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/AtomicReference.html)   [**NEXT CLASS**](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceFieldUpdater.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/concurrent/atomic/AtomicReferenceArray.html)    [**NO FRAMES**](http://docs.google.com/AtomicReferenceArray.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#17dp8vu) |

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

Class AtomicReferenceArray<E>

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

**Type Parameters:**E - The base class of elements held in this array **All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html)

public class **AtomicReferenceArray<E>**extends [Object](http://docs.google.com/java/lang/Object.html)implements [Serializable](http://docs.google.com/java/io/Serializable.html)

An array of object references in which elements may be updated atomically. See the [java.util.concurrent.atomic](http://docs.google.com/java/util/concurrent/atomic/package-summary.html) package specification for description of the properties of atomic variables.

**Since:** 1.5 **See Also:**[Serialized Form](http://docs.google.com/serialized-form.html#java.util.concurrent.atomic.AtomicReferenceArray)

| **Constructor Summary** | |
| --- | --- |
| [**AtomicReferenceArray**](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html#AtomicReferenceArray(E%5B%5D))([E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html)[] array)            Creates a new AtomicReferenceArray with the same length as, and all elements copied from, the given array. |
| [**AtomicReferenceArray**](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html#AtomicReferenceArray(int))(int length)            Creates a new AtomicReferenceArray of given length. |

| **Method Summary** | |
| --- | --- |
| boolean | [**compareAndSet**](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html#compareAndSet(int,%20E,%20E))(int i, [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) expect, [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) update)            Atomically sets the element at position i to the given updated value if the current value == the expected value. |
| [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) | [**get**](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html#get(int))(int i)            Gets the current value at position i. |
| [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) | [**getAndSet**](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html#getAndSet(int,%20E))(int i, [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) newValue)            Atomically sets the element at position i to the given value and returns the old value. |
| void | [**lazySet**](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html#lazySet(int,%20E))(int i, [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) newValue)            Eventually sets the element at position i to the given value. |
| int | [**length**](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html#length())()            Returns the length of the array. |
| void | [**set**](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html#set(int,%20E))(int i, [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) newValue)            Sets the element at position i to the given value. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html#toString())()            Returns the String representation of the current values of array. |
| boolean | [**weakCompareAndSet**](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html#weakCompareAndSet(int,%20E,%20E))(int i, [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) expect, [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) update)            Atomically sets the element at position i to the given updated value if the current value == the expected value. |

| **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()), [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** |
| --- |

### AtomicReferenceArray

public **AtomicReferenceArray**(int length)

Creates a new AtomicReferenceArray of given length.

**Parameters:**length - the length of the array

### AtomicReferenceArray

public **AtomicReferenceArray**([E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html)[] array)

Creates a new AtomicReferenceArray with the same length as, and all elements copied from, the given array.

**Parameters:**array - the array to copy elements from **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if array is null

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

### length

public final int **length**()

Returns the length of the array.

**Returns:**the length of the array

### get

public final [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) **get**(int i)

Gets the current value at position i.

**Parameters:**i - the index **Returns:**the current value

### set

public final void **set**(int i,  
 [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) newValue)

Sets the element at position i to the given value.

**Parameters:**i - the indexnewValue - the new value

### lazySet

public final void **lazySet**(int i,  
 [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) newValue)

Eventually sets the element at position i to the given value.

**Parameters:**i - the indexnewValue - the new value**Since:** 1.6

### getAndSet

public final [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) **getAndSet**(int i,  
 [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) newValue)

Atomically sets the element at position i to the given value and returns the old value.

**Parameters:**i - the indexnewValue - the new value **Returns:**the previous value

### compareAndSet

public final boolean **compareAndSet**(int i,  
 [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) expect,  
 [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) update)

Atomically sets the element at position i to the given updated value if the current value == the expected value.

**Parameters:**i - the indexexpect - the expected valueupdate - the new value **Returns:**true if successful. False return indicates that the actual value was not equal to the expected value.

### weakCompareAndSet

public final boolean **weakCompareAndSet**(int i,  
 [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) expect,  
 [E](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceArray.html) update)

Atomically sets the element at position i to the given updated value if the current value == the expected value.

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:**i - the indexexpect - the expected valueupdate - the new value **Returns:**true if successful.

### toString

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

Returns the String representation of the current values of array.

**Overrides:**[toString](http://docs.google.com/java/lang/Object.html#toString()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**the String representation of the current values of array.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/AtomicReferenceArray.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/AtomicReference.html)   [**NEXT CLASS**](http://docs.google.com/java/util/concurrent/atomic/AtomicReferenceFieldUpdater.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/concurrent/atomic/AtomicReferenceArray.html)    [**NO FRAMES**](http://docs.google.com/AtomicReferenceArray.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#17dp8vu) |

[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).