

---

# NSMutableArray Class Reference

[Cocoa > Data Management](#)





Apple Inc.  
© 2009 Apple Inc.  
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

Apple Inc.  
1 Infinite Loop  
Cupertino, CA 95014  
408-996-1010

Apple, the Apple logo, Cocoa, Mac, Mac OS, and Objective-C are trademarks of Apple Inc., registered in the United States and other countries.

Simultaneously published in the United States and Canada.

**Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.**

**IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY**

**DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.**

**THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.**

**Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.**

# Contents

---

## **NSArray Class Reference 5**

---

Overview	5
Tasks	5
Creating and Initializing a New Pointer Array	5
Managing the Collection	6
Getting the Pointer Functions	6
Class Methods	6
pointerArrayWithOptions:	6
pointerArrayWithPointerFunctions:	7
pointerArrayWithStrongObjects	7
pointerArrayWithWeakObjects	8
Instance Methods	8
addPointer:	8
allObjects	9
compact	9
count	9
initWithOptions:	9
initWithPointerFunctions:	10
insertPointerAtIndex:	10
pointerAtIndex:	11
pointerFunctions	11
removePointerAtIndex:	12
replacePointerAtIndex:withPointer:	12
setCount:	12

---

## **Document Revision History 15**

---

## **Index 17**

---



# NSArray Class Reference

---

<b>Inherits from</b>	NSObject
<b>Conforms to</b>	NSCoding NSCopying NSFastEnumeration NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/Foundation.framework
<b>Declared in</b>	NSArray.h
<b>Companion guides</b>	Collections Programming Topics for Cocoa Garbage Collection Programming Guide
<b>Availability</b>	Available in Mac OS X v10.5 and later.

## Overview

`NSArray` is a mutable collection modeled after `NSMutableArray` but it can also hold `NULL` values, which can be inserted or extracted (and which contribute to the object's count). Moreover, unlike traditional arrays, you can set the count of the array directly. In a garbage collected environment, if you specify a zeroing weak memory configuration, if an element is collected it is replaced by a `NULL` value.

The copying and archiving protocols are applicable only when a pointer array is configured for object uses.

The fast enumeration protocol (that is, use a pointer array in the `for...in` language construct—see Fast Enumeration in *The Objective-C 2.0 Programming Language*) will yield `NULL` values that are present in the array. It is defined for all types of pointers although the language syntax doesn't directly support this.

## Tasks

### Creating and Initializing a New Pointer Array

- `initWithOptions:` (page 9)  
Initializes the receiver to use the given options.
- `initWithPointerFunctions:` (page 10)  
Initializes the receiver to use the given functions.

- + [pointerArrayWithOptions:](#) (page 6)  
Returns a new pointer array initialized to use the given options.
- + [pointerArrayWithPointerFunctions:](#) (page 7)  
A new pointer array initialized to use the given functions.
- + [pointerArrayWithStrongObjects](#) (page 7)  
Returns a new pointer array that maintains strong references to its elements.
- + [pointerArrayWithWeakObjects](#) (page 8)  
Returns a new pointer array that maintains weak references to its elements.

## Managing the Collection

- [count](#) (page 9)  
Returns the number of elements in the receiver.
- [setCount:](#) (page 12)  
Sets the count for the receiver.
- [allObjects](#) (page 9)  
Returns an array containing all the objects in the receiver.
- [pointerAtIndex:](#) (page 11)  
Returns the pointer at a given index.
- [addPointer:](#) (page 8)  
Adds a given pointer to the receiver.
- [removePointerAtIndex:](#) (page 12)  
Removes the pointer at a given index.
- [insertPointer:atIndex:](#) (page 10)  
Inserts a pointer at a given index.
- [replacePointerAtIndex:withPointer:](#) (page 12)  
Replaces the pointer at a given index.
- [compact](#) (page 9)  
Removes NULL values from the receiver.

## Getting the Pointer Functions

- [pointerFunctions](#) (page 11)  
Returns a new `NSPointerFunctions` object reflecting the functions in use by the receiver.

## Class Methods

### **pointerArrayWithOptions:**

Returns a new pointer array initialized to use the given options.

```
+ (id)pointerArrayWithOptions:(NSPointerFunctionsOptions)options
```

**Parameters***options*

The pointer functions options for the new instance.

**Return Value**

A new pointer array initialized to use the given options.

**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

+ [pointerArrayWithPointerFunctions:](#) (page 7)

+ [pointerArrayWithStrongObjects](#) (page 7)

+ [pointerArrayWithWeakObjects](#) (page 8)

**Declared In**

NSPointerArray.h

**pointerArrayWithPointerFunctions:**

A new pointer array initialized to use the given functions.

```
+ (id)pointerArrayWithPointerFunctions:(NSPointerFunctions *)functions
```

**Parameters***functions*

The pointer functions for the new instance.

**Return Value**

A new pointer array initialized to use the given pointer functions.

**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

+ [pointerArrayWithOptions:](#) (page 6)

+ [pointerArrayWithStrongObjects](#) (page 7)

+ [pointerArrayWithWeakObjects](#) (page 8)

**Declared In**

NSPointerArray.h

**pointerArrayWithStrongObjects**

Returns a new pointer array that maintains strong references to its elements.

```
+ (id)pointerArrayWithStrongObjects
```

**Return Value**

A new pointer array that maintains strong references to its elements.

**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

- + [pointerArrayWithWeakObjects](#) (page 8)
- + [pointerArrayWithOptions:](#) (page 6)
- + [pointerArrayWithPointerFunctions:](#) (page 7)

**Declared In**

NSPointerArray.h

**pointerArrayWithWeakObjects**

Returns a new pointer array that maintains weak references to its elements.

+ (id)pointerArrayWithWeakObjects

**Return Value**

A new pointer array that maintains weak references to its elements.

**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

- + [pointerArrayWithStrongObjects](#) (page 7)
- + [pointerArrayWithOptions:](#) (page 6)
- + [pointerArrayWithPointerFunctions:](#) (page 7)

**Declared In**

NSPointerArray.h

## Instance Methods

**addPointer:**

Adds a given pointer to the receiver.

- (void)addPointer:(void \*)*pointer***Parameters***pointer*

The pointer to add. This value may be NULL.

**Discussion***pointer* is added at index [count](#) (page 9).**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPointerArray.h



## **allObjects**

Returns an array containing all the objects in the receiver.

- (NSArray \*)allObjects

### **Return Value**

An array containing all the object in the receiver.

### **Availability**

Available in Mac OS X v10.5 and later.

### **See Also**

- [count](#) (page 9)

### **Declared In**

NSPointerArray.h

## **compact**

Removes NULL values from the receiver.

- (void)compact

### **Availability**

Available in Mac OS X v10.5 and later.

### **Declared In**

NSPointerArray.h

## **count**

Returns the number of elements in the receiver.

- (NSUInteger)count

### **Return Value**

The number of elements in the receiver.

### **Availability**

Available in Mac OS X v10.5 and later.

### **See Also**

- [setCount:](#) (page 12)

### **Declared In**

NSPointerArray.h

## **initWithOptions:**

Initializes the receiver to use the given options.

- (id)initWithOptions:(NSPointerFunctionsOptions)options

**Parameters***options*

The pointer functions options for the new instance.

**Return Value**

The receiver, initialized to use the given options.

**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

- [initWithPointerFunctions:](#) (page 10)
- + [pointerArrayWithOptions:](#) (page 6)
- + [pointerArrayWithPointerFunctions:](#) (page 7)

**Declared In**

NSPointerArray.h

**initWithPointerFunctions:**

Initializes the receiver to use the given functions.

- (id)initWithPointerFunctions:(NSPointerFunctions \*)*functions***Parameters***functions*

The pointer functions for the new instance.

**Return Value**

The receiver, initialized to use the given functions.

**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

- [initWithOptions:](#) (page 9)
- + [pointerArrayWithPointerFunctions:](#) (page 7)
- + [pointerArrayWithOptions:](#) (page 6)

**Declared In**

NSPointerArray.h

**insertPointer:atIndex:**

Inserts a pointer at a given index.

- (void)insertPointer:(void \*)*item* atIndex:(NSUInteger)*index***Parameters***item*

The pointer to add.

*index*

The index of an element in the receiver. This value must be less than the [count](#) (page 9) of the receiver.

#### Discussion

Elements at and above *index*, including NULL values, slide higher.

#### Availability

Available in Mac OS X v10.5 and later.

#### Declared In

NSPointerArray.h

## pointerAtIndex:

Returns the pointer at a given index.

```
- (void *)pointerAtIndex:(NSUInteger) index
```

#### Parameters

*index*

The index of an element in the receiver. This value must be less than the [count](#) (page 9) of the receiver.

#### Return Value

The pointer at *index*.

#### Discussion

The returned value may be NULL.

#### Availability

Available in Mac OS X v10.5 and later.

#### Declared In

NSPointerArray.h

## pointerFunctions

Returns a new NSPointerFunctions object reflecting the functions in use by the receiver.

```
- (NSPointerFunctions *)pointerFunctions
```

#### Return Value

A new NSPointerFunctions object reflecting the functions in use by the receiver.

#### Discussion

The returned object is a new NSPointerFunctions object that you can modify and/or use directly to create other pointer collections.

#### Availability

Available in Mac OS X v10.5 and later.

#### Declared In

NSPointerArray.h

**removePointerAtIndex:**

Removes the pointer at a given index.

```
- (void)removePointerAtIndex:(NSUInteger) index
```

**Parameters**

*index*

The index of an element in the receiver. This value must be less than the [count](#) (page 9) of the receiver.

**Discussion**

Elements above *index*, including NULL values, slide lower.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPointerArray.h

**replacePointerAtIndex:withPointer:**

Replaces the pointer at a given index.

```
- (void)replacePointerAtIndex:(NSUInteger) index withPointer:(void *) item
```

**Parameters**

*index*

The index of an element in the receiver. This value must be less than the [count](#) (page 9) of the receiver.

*item*

The item with which to replace the element at *index*. This value may be NULL.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPointerArray.h

**setCount:**

Sets the count for the receiver.

```
- (void)setCount:(NSUInteger) count
```

**Parameters**

*count*

The count for the receiver.

**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

- [count](#) (page 9)

**Discussion**

If *count* is greater than the [count](#) (page 9) of the receiver, NULL values are added; if *count* is less than the [count](#) (page 9) of the receiver, then elements at indexes *count* and greater are removed from the receiver.

**Declared In**

NSPointerArray.h



# Document Revision History

---

This table describes the changes to *NSArray Class Reference*.

Date	Notes
2009-03-04	Clarified behavior in fast enumeration.
2007-01-06	New document that describes the mutable collection that can hold NULL values.

**REVISION HISTORY**

Document Revision History



# Index

---

## A

---

addPointer: **instance method** [8](#)  
allObjects **instance method** [9](#)

## C

---

compact **instance method** [9](#)  
count **instance method** [9](#)

## I

---

initWithOptions: **instance method** [9](#)  
initWithPointerFunctions: **instance method** [10](#)  
insertPointer:atIndex: **instance method** [10](#)

## P

---

pointerArrayWithOptions: **class method** [6](#)  
pointerArrayWithPointerFunctions: **class method** [7](#)  
pointerArrayWithStrongObjects **class method** [7](#)  
pointerArrayWithWeakObjects **class method** [8](#)  
pointerAtIndex: **instance method** [11](#)  
pointerFunctions **instance method** [11](#)

## R

---

removePointerAtIndex: **instance method** [12](#)  
replacePointerAtIndex:withPointer: **instance method** [12](#)

## S

---

setCount: **instance method** [12](#)