

---

# NSPointerFunctions Class Reference

[Cocoa](#) > [Data Management](#)



2008-10-15



Apple Inc.  
© 2008 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

## **NSPointerFunctions Class Reference 5**

---

Overview	5
Tasks	5
Creating and Initializing an NSPointerFunctions Object	5
Personality Functions	6
Memory Configuration	6
Properties	6
acquireFunction	6
descriptionFunction	7
hashFunction	7
isEqualFunction	7
relinquishFunction	7
sizeFunction	8
usesStrongWriteBarrier	8
usesWeakReadAndWriteBarriers	8
Class Methods	9
pointerFunctionsWithOptions:	9
Instance Methods	9
initWithOptions:	9
Constants	10
NSPointerFunctionsOptions	10
Memory and Personality Options	10

## **Document Revision History 13**

---

## **Index 15**

---



# NSPointerFunctions Class Reference

---

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

## Overview

An instance of `NSPointerFunctions` defines callout functions appropriate for managing a pointer reference held somewhere else.

The functions specified by an instance of `NSPointerFunctions` are separated into two clusters—those that define “personality” such as “object” or “C-string”, and those that describe memory management issues such as a memory deallocation function. There are constants for common personalities and memory manager selections (see “[Memory and Personality Options](#)” (page 10)).

`NSHashTable`, `NSMapTable`, and `NSPointerArray` use an `NSPointerFunctions` object to define the acquisition and retention behavior for the pointers they manage. Note, however, that not all combinations of personality and memory management behavior are valid for these collections. The pointer collection objects copy the `NSPointerFunctions` object on input and output, so you cannot usefully subclass `NSPointerFunctions`.

## Tasks

### Creating and Initializing an NSPointerFunctions Object

- `initWithOptions:` (page 9)  
Returns an `NSPointerFunctions` object initialized with the given options.
- + `pointerFunctionsWithOptions:` (page 9)  
Returns a new `NSPointerFunctions` object initialized with the given options.

## Personality Functions

[hashFunction](#) (page 7) *property*

The hash function.

[isEqualFunction](#) (page 7) *property*

The function used to compare pointers.

[sizeFunction](#) (page 8) *property*

The function used to determine the size of pointers.

[descriptionFunction](#) (page 7) *property*

The function used to describe elements.

## Memory Configuration

[acquireFunction](#) (page 6) *property*

The function used to acquire memory.

[relinquishFunction](#) (page 7) *property*

The function used to relinquish memory.

[usesStrongWriteBarrier](#) (page 8) *property*

Specifies whether, in a garbage collected environment, pointers should be assigned using a strong write barrier.

[usesWeakReadAndWriteBarriers](#) (page 8) *property*

Specifies whether, in a garbage collected environment, pointers should use weak read and write barriers.

## Properties

For more about Objective-C properties, see “Properties” in *The Objective-C 2.0 Programming Language*.

### acquireFunction

The function used to acquire memory.

```
@property void *(*acquireFunction)(const void *src, NSUInteger (*size)(const void *item), BOOL shouldCopy)
```

#### Discussion

This specifies the function to use for copy-in operations.

#### Availability

Available in Mac OS X v10.5 and later.

#### See Also

[@property relinquishFunction](#) (page 7)

#### Declared In

NSPointerFunctions.h

## descriptionFunction

The function used to describe elements.

```
@property NSString *(*descriptionFunction)(const void *item)
```

### Discussion

This function is used by description methods for hash and map tables.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSPointerFunctions.h

## hashFunction

The hash function.

```
@property NSUInteger (*hashFunction)(const void *item, NSUInteger (*size)(const void *item))
```

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSPointerFunctions.h

## isEqualFunction

The function used to compare pointers.

```
@property BOOL (*isEqualFunction)(const void *item1, const void *item2, NSUInteger (*size)(const void *item))
```

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSPointerFunctions.h

## relinquishFunction

The function used to relinquish memory.

```
@property void (*relinquishFunction)(const void *item, NSUInteger (*size)(const void *item))
```

### Discussion

This specifies the function to use when an item is removed from a table or pointer array.

### Availability

Available in Mac OS X v10.5 and later.

**See Also**

[@property acquireFunction](#) (page 6)

**Declared In**

NSPointerFunctions.h

## sizeFunction

The function used to determine the size of pointers.

```
@property NSUInteger (*sizeFunction)(const void *item)
```

**Discussion**

This function is used for copy-in operations (unless the collection has an object personality).

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPointerFunctions.h

## usesStrongWriteBarrier

Specifies whether, in a garbage collected environment, pointers should be assigned using a strong write barrier.

```
@property BOOL usesStrongWriteBarrier
```

**Discussion**

If you use garbage collection, read and write barrier functions must be used when pointers are from memory scanned by the collector.

**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

[@property usesWeakReadAndWriteBarriers](#) (page 8)

**Declared In**

NSPointerFunctions.h

## usesWeakReadAndWriteBarriers

Specifies whether, in a garbage collected environment, pointers should use weak read and write barriers.

```
@property BOOL usesWeakReadAndWriteBarriers
```

**Discussion**

If you use garbage collection, read and write barrier functions must be used when pointers are from memory scanned by the collector.



**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

[@property usesStrongWriteBarrier](#) (page 8)

**Declared In**

NSPointerFunctions.h

## Class Methods

**pointerFunctionsWithOptions:**

Returns a new NSPointerFunctions object initialized with the given options.

+ (id)pointerFunctionsWithOptions:(NSPointerFunctionsOptions)options

**Parameters**

*options*

The options for the new NSPointerFunctions object.

**Return Value**

A new NSPointerFunctions object initialized with the given options.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPointerFunctions.h

## Instance Methods

**initWithOptions:**

Returns an NSPointerFunctions object initialized with the given options.

- (id)initWithOptions:(NSPointerFunctionsOptions)options

**Parameters**

*options*

The options for the new NSPointerFunctions object.

**Return Value**

The receiver, initialized with the given options.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSPointerFunctions.h

## Constants

### NSPointerFunctionsOptions

Defines the memory and personality options for an `NSPointerFunctions` object.

```
typedef NSUInteger NSPointerFunctionsOptions;
```

#### Discussion

For values, see [“Memory and Personality Options”](#) (page 10).

#### Availability

Available in Mac OS X v10.5 and later.

#### Declared In

`NSPointerFunctions.h`

### Memory and Personality Options

Specify memory and personality options for an `NSPointerFunctions` object.

```
enum {
    NSPointerFunctionsStrongMemory = (0 << 0),
    NSPointerFunctionsZeroingWeakMemory = (1 << 0),
    NSPointerFunctionsOpaqueMemory = (2 << 0),
    NSPointerFunctionsMallocMemory = (3 << 0),
    NSPointerFunctionsMachVirtualMemory = (4 << 0),
    NSPointerFunctionsObjectPersonality = (0 << 8),
    NSPointerFunctionsOpaquePersonality = (1 << 8),
    NSPointerFunctionsObjectPointerPersonality = (2 << 8),
    NSPointerFunctionsCStringPersonality = (3 << 8),
    NSPointerFunctionsStructPersonality = (4 << 8),
    NSPointerFunctionsIntegerPersonality = (5 << 8),
    NSPointerFunctionsCopyIn = (1 << 16),
};
```

#### Constants

`NSPointerFunctionsStrongMemory`

Use strong write-barriers to backing store; use garbage-collected memory on copy-in.

This is the default memory value.

As a special case, if you do not use garbage collection and specify this value in conjunction with [NSPointerFunctionsObjectPersonality](#) (page 11) then the `NSPointerFunctions` object uses retain and release.

Available in Mac OS X v10.5 and later.

Declared in `NSPointerFunctions.h`.

`NSPointerFunctionsZeroingWeakMemory`

Use weak read and write barriers; use garbage-collected memory on copyIn.

Available in Mac OS X v10.5 and later.

Declared in `NSPointerFunctions.h`.

NSPointerFunctionsOpaqueMemory

Take no action when pointers are deleted.

This is essentially a no-op relinquish function; the acquire function is only used for copy-in operations. This option is unlikely to be a good choice for objects.

Available in Mac OS X v10.5 and later.

Declared in `NSPointerFunctions.h`.

NSPointerFunctionsMallocMemory

Use `free()` on removal, `calloc()` on copy in.

Available in Mac OS X v10.5 and later.

Declared in `NSPointerFunctions.h`.

NSPointerFunctionsMachVirtualMemory

Use Mach memory.

Available in Mac OS X v10.5 and later.

Declared in `NSPointerFunctions.h`.

NSPointerFunctionsObjectPersonality

Use `hash` and `isEqual` methods for hashing and equality comparisons, use the `description` method for a description.

This is the default personality value.

As a special case, if you do not use garbage collection and specify this value in conjunction with [NSPointerFunctionsStrongMemory](#) (page 10) then the `NSPointerFunctions` object uses `retain` and `release`.

Available in Mac OS X v10.5 and later.

Declared in `NSPointerFunctions.h`.

NSPointerFunctionsOpaquePersonality

Use shifted pointer for the hash value and direct comparison to determine equality.

Available in Mac OS X v10.5 and later.

Declared in `NSPointerFunctions.h`.

NSPointerFunctionsObjectPointerPersonality

Use shifted pointer for the hash value and direct comparison to determine equality; use the `description` method for a description.

As a special case, if you do not use garbage collection and specify this value in conjunction with [NSPointerFunctionsStrongMemory](#) (page 10) then the `NSPointerFunctions` object uses `retain` and `release`.

Available in Mac OS X v10.5 and later.

Declared in `NSPointerFunctions.h`.

NSPointerFunctionsCStringPersonality

Use a string hash and `strcmp`; C-string '%s' style description.

Available in Mac OS X v10.5 and later.

Declared in `NSPointerFunctions.h`.

NSPointerFunctionsStructPersonality

Use a memory hash and `memcmp` (using a size function that you must set—see [sizeFunction](#) (page 8)).

Available in Mac OS X v10.5 and later.

Declared in `NSPointerFunctions.h`.

`NSPointerFunctionsIntegerPersonality`  
Use unshifted value as hash and equality.  
Available in Mac OS X v10.5 and later.  
Declared in `NSPointerFunctions.h`.

`NSPointerFunctionsCopyIn`  
Use the memory acquire function to allocate and copy items on input (see [acquireFunction](#) (page 6)).  
Available in Mac OS X v10.5 and later.  
Declared in `NSPointerFunctions.h`.

**Discussion**

Memory options are mutually exclusive and personality options are mutually exclusive.

**Declared In**

`NSPointerFunctions.h`

# Document Revision History

---

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

Date	Notes
2008-10-15	Added definition of <code>NSPointerFunctionsOpaqueMemory</code> .
2007-10-31	Clarified definition of <code>NSPointerFunctionsObjectPersonality</code> .
2007-06-26	New document that describes the class used to define callout functions appropriate for managing a pointer reference.

**REVISION HISTORY**

Document Revision History

# Index

---

## A

---

acquireFunction **instance property** [6](#)

## D

---

descriptionFunction **instance property** [7](#)

## H

---

hashFunction **instance property** [7](#)

## I

---

initWithOptions: **instance method** [9](#)

isEqualFunction **instance property** [7](#)

## M

---

Memory and Personality Options [10](#)

## N

---

NSPointerFunctionsCopyIn **constant** [12](#)

NSPointerFunctionsCStringPersonality **constant** [11](#)

NSPointerFunctionsIntegerPersonality **constant** [12](#)

NSPointerFunctionsMachVirtualMemory **constant** [11](#)

NSPointerFunctionsMallocMemory **constant** [11](#)

NSPointerFunctionsObjectPersonality **constant** [11](#)

NSPointerFunctionsObjectPointerPersonality **constant** [11](#)

NSPointerFunctionsOpaqueMemory **constant** [11](#)

NSPointerFunctionsOpaquePersonality **constant** [11](#)

NSPointerFunctionsOptions **data type** [10](#)

NSPointerFunctionsStrongMemory **constant** [10](#)

NSPointerFunctionsStructPersonality **constant** [11](#)

NSPointerFunctionsZeroingWeakMemory **constant** [10](#)

## P

---

pointerFunctionsWithOptions: **class method** [9](#)

## R

---

relinquishFunction **instance property** [7](#)

## S

---

sizeFunction **instance property** [8](#)

## U

---

usesStrongWriteBarrier **instance property** [8](#)

usesWeakReadAndWriteBarriers **instance property** [8](#)