
NSMutableDictionary Class Reference

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





Apple Inc.
© 2007 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, and Mac OS 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

NSMutableDictionary Class Reference 5

Overview 5

Tasks 6

 Creating and Initializing a Map Table 6

 Accessing Content 6

 Manipulating Content 6

 Creating a Dictionary Representation 7

 Accessing Pointer Functions 7

Class Methods 7

 NSMutableDictionaryWithOptions: 7

 NSMutableDictionaryWithStrongObjects 8

 NSMutableDictionaryWithStrongToWeakObjects 8

 NSMutableDictionaryWithWeakToStrongObjects 8

 NSMutableDictionaryWithWeakToWeakObjects 9

Instance Methods 9

 count 9

 dictionaryRepresentation 9

 initWithWithOptions:capacity: 10

 initWithKeyPointerFunctions:valuePointerFunctions:capacity: 10

 keyEnumerator 11

 keyPointerFunctions 12

 objectEnumerator 12

 objectForKey: 12

 removeAllObjects 13

 removeObjectForKey: 13

 setObject:forKey: 13

 valuePointerFunctions 14

Constants 14

 Personality Options 14

Document Revision History 17

Index 19

NSMutableDictionary Class Reference

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

Overview

`NSMutableDictionary` is a mutable collection modeled after `NSDictionary` but provides different options, in particular to support weak relationships in a garbage-collected environment.

`NSMutableDictionary` is modeled after `NSDictionary` but offers different behaviors:

- It can hold weak references to its keys and/or values.
Keys and/or values held "weakly" in a manner that entries are removed when one of the objects is collected under garbage collection.
If you are not using garbage collection, you must explicitly remove entries as you would from a dictionary. In addition to being held weakly, keys or values may be copied on input or may use pointer identity for equality and hashing.
- It can contain arbitrary pointers (its contents are not constrained to being objects).
You can configure an `NSMutableDictionary` instance to operate on arbitrary pointers and not just objects, although typically you are encouraged to use the C function API for `void *` pointers. The object-based API (such as `setObject:forKey:` (page 13)) will not work for non-object pointers without type-casting.

To configure an `NSMutableDictionary` instance for pointer use, you can: create or initialize it using `NSMutableDictionaryWithOptions:valueOptions:` (page 7) or `initWithKeyOptions:valueOptions:capacity:` (page 10) and the appropriate `NSPointerFunctionsOptions` options; or initialize it with `initWithKeyPointerFunctions:valuePointerFunctions:capacity:` (page 10) and appropriate instances of `NSPointerFunctions`. Note that only the options listed in “Personality Options” (page

14) guarantee that the rest of the API will work correctly—including copying, archiving, and fast enumeration. If you use other `NSPointerFunctions` options, the map table may not work correctly, or may not even be initialized correctly.

Tasks

Creating and Initializing a Map Table

- [initWithKeyOptions:valueOptions:capacity:](#) (page 10)
Returns a map table, initialized with the given options.
- + [mapTableWithKeyOptions:valueOptions:](#) (page 7)
Returns a new map table, initialized with the given options
- [initWithKeyPointerFunctions:valuePointerFunctions:capacity:](#) (page 10)
Returns a map table, initialized with the given functions.
- + [mapTableWithStrongToStrongObjects](#) (page 8)
Returns a new map table object which has strong references to the keys and values.
- + [mapTableWithWeakToStrongObjects](#) (page 8)
Returns a new map table object which has weak references to the keys and strong references to the values.
- + [mapTableWithStrongToWeakObjects](#) (page 8)
Returns a new map table object which has strong references to the keys and weak references to the values.
- + [mapTableWithWeakToWeakObjects](#) (page 9)
Returns a new map table object which has weak references to the keys and values.

Accessing Content

- [objectForKey:](#) (page 12)
Returns a the value associated with a given key.
- [keyEnumerator](#) (page 11)
Returns an enumerator object that lets you access each key in the receiver.
- [objectEnumerator](#) (page 12)
Returns an enumerator object that lets you access each value in the receiver.
- [count](#) (page 9)
Returns the number of key-value pairs in the receiver.

Manipulating Content

- [setObject:forKey:](#) (page 13)
Adds a given key-value pair to the receiver.
- [removeObjectForKey:](#) (page 13)
Removes a given key and its associated value from the receiver.

- [removeAllObjects](#) (page 13)
Empties the receiver of its entries.

Creating a Dictionary Representation

- [dictionaryRepresentation](#) (page 9)
Returns a dictionary representation of the receiver.

Accessing Pointer Functions

- [keyPointerFunctions](#) (page 12)
Returns the pointer functions the receiver uses to manage keys.
- [valuePointerFunctions](#) (page 14)
Returns the pointer functions the receiver uses to manage values.

Class Methods

mapTableWithOptions:capacity:

Returns a new map table, initialized with the given options

```
+ (id)mapTableWithOptions:(NSPointerFunctionsOptions)keyOptions
    valueOptions:(NSPointerFunctionsOptions)valueOptions
    capacity:(NSUInteger)capacity;
```

Parameters

keyOptions

A bit field that specifies the options for the keys in the map table.

Important: Not all values of `NSPointerFunctionsOptions` are valid for `NSMapTable`. For values that are guaranteed to work correctly, see “[Personality Options](#)” (page 14).

valueOptions

A bit field that specifies the options for the values in the map table.

Important: Not all values of `NSPointerFunctionsOptions` are valid for `NSMapTable`. For values that are guaranteed to work correctly, see “[Personality Options](#)” (page 14).

Return Value

A new map table, initialized with the given options.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [initWithKeyOptions:valueOptions:capacity:](#) (page 10)

- initWithKeyPointerFunctions:valuePointerFunctions:capacity: (page 10)

Declared In

NSMutableDictionary.h

NSMutableDictionaryWithStrongToStrongObjects

Returns a new map table object which has strong references to the keys and values.

+ (id)NSMutableDictionaryWithStrongToStrongObjects

Return Value

A new map table object which has strong references to the keys and values.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSMutableDictionary.h

NSMutableDictionaryWithStrongToWeakObjects

Returns a new map table object which has strong references to the keys and weak references to the values.

+ (id)NSMutableDictionaryWithStrongToWeakObjects

Return Value

A new map table object which has strong references to the keys and weak references to the values.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSMutableDictionary.h

NSMutableDictionaryWithWeakToStrongObjects

Returns a new map table object which has weak references to the keys and strong references to the values.

+ (id)NSMutableDictionaryWithWeakToStrongObjects

Return Value

A new map table object which has weak references to the keys and strong references to the values.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSMutableDictionary.h

mapTableWithWeakToWeakObjects

Returns a new map table object which has weak references to the keys and values.

```
+ (id)mapTableWithWeakToWeakObjects
```

Return Value

A new map table object which has weak references to the keys and values.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSMapTable.h

Instance Methods

count

Returns the number of key-value pairs in the receiver.

```
- (NSUInteger)count
```

Return Value

The number of key-value pairs in the receiver.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSMapTable.h

dictionaryRepresentation

Returns a dictionary representation of the receiver.

```
- (NSDictionary *)dictionaryRepresentation
```

Return Value

A dictionary representation of the receiver.

Discussion

The receiver's contents must be objects.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSMapTable.h

initWithKeyOptions:valueOptions:capacity:

Returns a map table, initialized with the given options.

```
- (id)initWithKeyOptions:(NSPointerFunctionsOptions)keyOptions
    valueOptions:(NSPointerFunctionsOptions)valueOptions
    capacity:(NSUInteger)initialCapacity
```

Parameters

keys

A bit field that specifies the options for the keys in the map table.

Important: Not all values of `NSPointerFunctionsOptions` are valid for `NSMapTable`. For values that are guaranteed to work correctly, see “[Personality Options](#)” (page 14).

values

A bit field that specifies the options for the values in the map table.

Important: Not all values of `NSPointerFunctionsOptions` are valid for `NSMapTable`. For values that are guaranteed to work correctly, see “[Personality Options](#)” (page 14).

capacity

The initial capacity of the receiver. This is just a hint; the map table may subsequently grow and shrink as required.

Return Value

A map table initialized using the given options.

Discussion

values must contain entries at all the indexes specified in *keys*.

Availability

Available in Mac OS X v10.5 and later.

See Also

+ [mapTableWithKeyOptions:valueOptions:](#) (page 7)

- [initWithKeyPointerFunctions:valuePointerFunctions:capacity:](#) (page 10)

Declared In

`NSMapTable.h`

initWithKeyPointerFunctions:valuePointerFunctions:capacity:

Returns a map table, initialized with the given functions.

```
- (id)initWithKeyPointerFunctions:(NSPointerFunctions *)keyFunctions
    valuePointerFunctions:(NSPointerFunctions *)valueFunctions
    capacity:(NSUInteger)initialCapacity
```

Parameters*keyFunctions*

The functions the receiver uses to manage keys.

Important: Not all values of `NSPointerFunctionsOptions` are valid for `NSMapTable`. For values that are guaranteed to work correctly, see [“Personality Options”](#) (page 14).

valueFunctions

The functions the receiver uses to manage values.

Important: Not all values of `NSPointerFunctionsOptions` are valid for `NSMapTable`. For values that are guaranteed to work correctly, see [“Personality Options”](#) (page 14).

initialCapacity

The initial capacity of the receiver. This is just a hint; the map table may subsequently grow and shrink as required.

Return Value

A map table, initialized with the given functions.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`NSMapTable.h`

keyEnumerator

Returns an enumerator object that lets you access each key in the receiver.

```
- (NSEnumerator *)keyEnumerator
```

Return Value

An enumerator object that lets you access each key in the receiver.

Discussion

The following code fragment illustrates how you might use the method.

```
NSEnumerator *enumerator = [myMapTable keyEnumerator];
id value;

while ((value = [enumerator nextObject])) {
    /* code that acts on the map table's keys */
}
```

See also `NSFastEnumeration`.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`NSMapTable.h`

keyPointerFunctions

Returns the pointer functions the receiver uses to manage keys.

- (NSPointerFunctions *)keyPointerFunctions

Return Value

The pointer functions the receiver uses to manage keys.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [valuePointerFunctions](#) (page 14)

Declared In

NSMapTable.h

objectEnumerator

Returns an enumerator object that lets you access each value in the receiver.

- (NSEnumerator *)objectEnumerator

Return Value

An enumerator object that lets you access each value in the receiver.

Discussion

The following code fragment illustrates how you might use the method.

```
NSEnumerator *enumerator = [myMapTable objectEnumerator];
id value;

while ((value = [enumerator nextObject])) {
    /* code that acts on the map table's values */
}
```

See also NSFastEnumeration.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSMapTable.h

objectForKey:

Returns a the value associated with a given key.

- (id)objectForKey:(id)aKey

Parameters

aKey

The key for which to return the corresponding value.

Return Value

The value associated with *aKey*, or `nil` if no value is associated with *aKey*.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSMapTable.h

removeAllObjects

Empties the receiver of its entries.

```
- (void)removeAllObjects
```

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSMapTable.h

removeObjectForKey:

Removes a given key and its associated value from the receiver.

```
- (void)removeObjectForKey:(id)aKey
```

Parameters

aKey

The key to remove.

Discussion

Does nothing if *aKey* does not exist.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSMapTable.h

setObject:forKey:

Adds a given key-value pair to the receiver.

```
- (void)setObject:(id)anObject
    forKey:(id)aKey
```

Parameters

anObject

The value for *aKey*. This value must not be `nil`.

aKey

The key for *anObject*. This value must not be `nil`.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSMapTable.h

valuePointerFunctions

Returns the pointer functions the receiver uses to manage values.

```
- (NSPointerFunctions *)valuePointerFunctions
```

Return Value

The pointer functions the receiver uses to manage values.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [keyPointerFunctions](#) (page 12)

Declared In

NSMapTable.h

Constants

Personality Options

Constants used as components in a bitfield to specify the behavior of elements (keys and values) in an NSMapTable object.

```
enum {
    NSMapTableStrongMemory           = 0,
    NSMapTableZeroingWeakMemory     = NSPointerFunctionsZeroingWeakMemory,
    NSMapTableCopyIn                = NSPointerFunctionsCopyIn,
    NSMapTableObjectPointerPersonality = NSPointerFunctionsObjectPointerPersonality
};
```

Constants

NSMapTableStrongMemory

Specifies a strong reference from the map table to its contents.

Equal to NSPointerFunctionsStrongMemory.

Available in Mac OS X v10.5 and later.

Declared in NSMapTable.h.

NSMutableDictionaryZeroingWeakMemory

Specifies a zeroing weak reference from the map table to its contents.

Equal to NSMutableDictionaryFunctionsZeroingWeakMemory.

Available in Mac OS X v10.5 and later.

Declared in NSMutableDictionary.h.

NSMutableDictionaryCopyIn

Use the memory acquire function to allocate and copy items on input (see acquireFunction [NSMutableDictionaryFunctions]).

Equal to NSMutableDictionaryFunctionsCopyIn.

Available in Mac OS X v10.5 and later.

Declared in NSMutableDictionary.h.

NSMutableDictionaryObjectPointerPersonality

Use shifted pointer hash and direct equality, object description.

Equal to NSMutableDictionaryFunctionsObjectPointerPersonality.

Available in Mac OS X v10.5 and later.

Declared in NSMutableDictionary.h.

Declared In

NSMutableDictionary.h

Document Revision History

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

Date	Notes
2007-07-22	New document that describes the Cocoa class used to contain an array of objects, optionally using weak references.

REVISION HISTORY

Document Revision History

Index

C

count [instance method 9](#)

D

dictionaryRepresentation [instance method 9](#)

I

initWithKeyOptions:valueOptions:capacity:
[instance method 10](#)
initWithKeyPointerFunctions:valuePointerFunctions:
capacity: [instance method 10](#)

K

keyEnumerator [instance method 11](#)
keyPointerFunctions [instance method 12](#)

M

mapTableWithKeyOptions:valueOptions: [class method 7](#)
mapTableWithStrongToStrongObjects [class method 8](#)
mapTableWithStrongToWeakObjects [class method 8](#)
mapTableWithWeakToStrongObjects [class method 8](#)
mapTableWithWeakToWeakObjects [class method 9](#)

N

NSMutableDictionaryCopyIn [constant 15](#)

NSMutableDictionaryObjectPointerPersonality [constant 15](#)
NSMutableDictionaryStrongMemory [constant 14](#)
NSMutableDictionaryZeroingWeakMemory [constant 15](#)

O

objectEnumerator [instance method 12](#)
objectForKey: [instance method 12](#)

P

Personality Options [14](#)

R

removeAllObjects [instance method 13](#)
removeObjectForKey: [instance method 13](#)

S

setObject:forKey: [instance method 13](#)

V

valuePointerFunctions [instance method 14](#)