

---

# NSPropertyDescription Class Reference

[Cocoa](#) > [Objective-C Language](#)



2006-10-03



Apple Inc.  
© 2006 Apple Computer, 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

---

## **NSPropertyDescription Class Reference 5**

---

Overview	5
Editing Property Descriptions	5
Tasks	6
Getting Features of a Property	6
Setting Features of a Property	6
Validation	6
Versioning Support	7
Instance Methods	7
entity	7
isIndexed	7
isOptional	8
isTransient	8
name	8
setIndexed:	9
setName:	9
setOptional:	10
setTransient:	10
setUserInfo:	11
setValidationPredicates:withValidationWarnings:	11
setVersionHashModifier:	12
userInfo	12
validationPredicates	13
validationWarnings	13
versionHash	13
versionHashModifier	14

---

## **Document Revision History 15**

---

## **Index 17**

---



# NSPropertyDescription Class Reference

---

<b>Inherits from</b>	NSObject
<b>Conforms to</b>	NSCoding NSCopying NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/CoreData.framework
<b>Availability</b>	Available in Mac OS X v10.4 and later.
<b>Companion guide</b>	Core Data Programming Guide
<b>Declared in</b>	NSPropertyDescription.h
<b>Related sample code</b>	Core Data HTML Store

## Overview

The `NSPropertyDescription` class is used to define properties of an entity in a Core Data managed object model. Properties are to entities what instance variables are to classes.

A property describes a single value within an object managed by the Core Data Framework. There are different types of property, each represented by a subclass which encapsulates the specific property behavior—see `NSAttributeDescription`, `NSRelationshipDescription`, and `NSFetchedPropertyDescription`.

Note that a property name cannot be the same as any no-parameter method name of `NSObject` or `NSManagedObject`. For example, you cannot give a property the name "description". There are hundreds of methods on `NSObject` which may conflict with property names—and this list can grow without warning from frameworks or other libraries. You should avoid very general words (like "font" and "color") and words or phrases which overlap with Cocoa paradigms (such as "isEditing" and "objectSpecifier").

Properties—relationships as well as attributes—may be transient. A managed object context knows about transient properties and tracks changes made to them. Transient properties are ignored by the persistent store, and not just during saves: you cannot fetch using a predicate based on transients (although you can use transient properties to filter in memory yourself).

## Editing Property Descriptions

---

Property descriptions are editable until they are used by an object graph manager (such as a persistent store coordinator). This allows you to create or modify them dynamically. However, once a description is used (when the managed object model to which it belongs is associated with a persistent store coordinator), it

*must not* (indeed cannot) be changed. This is enforced at runtime: any attempt to mutate a model or any of its sub-objects after the model is associated with a persistent store coordinator causes an exception to be thrown. If you need to modify a model that is in use, create a copy, modify the copy, and then discard the objects with the old model.

## Tasks

### Getting Features of a Property

- [entity](#) (page 7)  
Returns the entity description of the receiver.
- [isIndexed](#) (page 7)  
Returns a Boolean value that indicates whether the receiver is important for searching.
- [isOptional](#) (page 8)  
Returns a Boolean value that indicates whether the receiver is optional.
- [isTransient](#) (page 8)  
Returns a Boolean value that indicates whether the receiver is transient.
- [name](#) (page 8)  
Returns the name of the receiver.
- [userInfo](#) (page 12)  
Returns the user info dictionary of the receiver.

### Setting Features of a Property

- [setIndexed:](#) (page 9)  
Sets the optionality flag of the receiver.
- [setName:](#) (page 9)  
Sets the name of the receiver.
- [setOptional:](#) (page 10)  
Sets the optionality flag of the receiver.
- [setTransient:](#) (page 10)  
Sets the transient flag of the receiver.
- [setUserInfo:](#) (page 11)  
Sets the user info dictionary of the receiver.

### Validation

- [validationPredicates](#) (page 13)  
Returns the validation predicates of the receiver.
- [validationWarnings](#) (page 13)  
Returns the error strings associated with the receiver's validation predicates.

- [setValidationPredicates:withValidationWarnings:](#) (page 11)  
Sets the validation predicates and warnings of the receiver.

## Versioning Support

- [versionHash](#) (page 13)  
Returns the version hash for the receiver.
- [versionHashModifier](#) (page 14)  
Returns the version hash modifier for the receiver.
- [setVersionHashModifier:](#) (page 12)  
Sets the version hash modifier for the receiver.

## Instance Methods

### **entity**

Returns the entity description of the receiver.

- (NSEntityDescription \*)entity

#### **Return Value**

The entity description of the receiver.

#### **Availability**

Available in Mac OS X v10.4 and later.

#### **See Also**

[setProperty:\(NSEntityDescription\)](#)

#### **Declared In**

[NSPropertyDescription.h](#)

### **isIndexed**

Returns a Boolean value that indicates whether the receiver is important for searching.

- (BOOL)isIndexed

#### **Return Value**

YES if the receiver is important for searching, otherwise NO.

#### **Discussion**

Object stores can optionally use this information upon store creation for operations such as defining indexes.

#### **Availability**

Available in Mac OS X v10.5 and later.

**See Also**

- [setIndexed:](#) (page 9)

**Declared In**

NSObjectPropertyDescription.h

## isOptional

Returns a Boolean value that indicates whether the receiver is optional.

- (BOOL)isOptional

**Return Value**

YES if the receiver is optional, otherwise NO.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [setOptional:](#) (page 10)

**Declared In**

NSObjectPropertyDescription.h

## isTransient

Returns a Boolean value that indicates whether the receiver is transient.

- (BOOL)isTransient

**Return Value**

YES if the receiver is transient, otherwise NO.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [setTransient:](#) (page 10)

**Declared In**

NSObjectPropertyDescription.h

## name

Returns the name of the receiver.

- (NSString \*)name

**Return Value**

The name of the receiver.



**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [setName:](#) (page 9)

**Related Sample Code**

CoreRecipes

**Declared In**

NSPropertyDescription.h

**setIndexed:**

Sets the optionality flag of the receiver.

```
- (void)setIndexed:(BOOL)flag
```

**Parameters**

*flag*

A Boolean value that indicates whether whether the receiver is important for searching (YES) or not (NO).

**Discussion**

Object stores can optionally use this information upon store creation for operations such as defining indexes.

**Special Considerations**

This method raises an exception if the receiver's model has been used by an object graph manager.

**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

- [isIndexed](#) (page 7)

**Declared In**

NSPropertyDescription.h

**setName:**

Sets the name of the receiver.

```
- (void)setName:(NSString *)name
```

**Parameters**

*name*

The name of the receiver.

**Special Considerations**

A property name cannot be the same as any no-parameter method name of `NSObject` or `NSManagedObject`. Since there are hundreds of methods on `NSObject` which may conflict with property names, you should avoid very general words (like "font," and "color") and words or phrases which overlap with Cocoa paradigms (such as "isEditing" and "objectSpecifier").

This method raises an exception if the receiver's model has been used by an object graph manager.

#### Availability

Available in Mac OS X v10.4 and later.

#### See Also

- [name](#) (page 8)

#### Declared In

NSPropertyDescription.h

### setOptional:

Sets the optionality flag of the receiver.

```
- (void)setOptional:(BOOL)flag
```

#### Parameters

*flag*

A Boolean value that indicates whether whether the receiver is optional (YES) or not (NO).

#### Discussion

The optionality flag specifies whether a property's value can be `nil` before an object can be saved to a persistent store.

#### Special Considerations

This method raises an exception if the receiver's model has been used by an object graph manager.

#### Availability

Available in Mac OS X v10.4 and later.

#### See Also

- [isOptional](#) (page 8)

#### Declared In

NSPropertyDescription.h

### setTransient:

Sets the transient flag of the receiver.

```
- (void)setTransient:(BOOL)flag
```

#### Parameters

*flag*

A Boolean value that indicates whether whether the receiver is transient (YES) or not (NO).

#### Discussion

The transient flag specifies whether or not a property's value is ignored when an object is saved to a persistent store. Transient properties are not saved to the persistent store, but are still managed for undo, redo, validation, and so on.

**Special Considerations**

This method raises an exception if the receiver's model has been used by an object graph manager.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [isTransient](#) (page 8)

**Declared In**

NSPropertyDescription.h

**setUserInfo:**

Sets the user info dictionary of the receiver.

```
- (void)setUserInfo:(NSDictionary *)dictionary
```

**Parameters**

*dictionary*

The user info dictionary of the receiver.

**Special Considerations**

This method raises an exception if the receiver's model has been used by an object graph manager.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [userInfo](#) (page 12)

**Declared In**

NSPropertyDescription.h

**setValidationPredicates:withValidationWarnings:**

Sets the validation predicates and warnings of the receiver.

```
- (void)setValidationPredicates:(NSArray *)validationPredicates
withValidationWarnings:(NSArray *)validationWarnings
```

**Parameters**

*validationPredicates*

An array containing the validation predicates for the receiver.

*validationWarnings*

An array containing the validation warnings for the receiver.

**Discussion**

The *validationPredicates* and *validationWarnings* arrays should contain the same number of elements, and corresponding elements should appear at the same index in each array.

Instead of implementing individual validation methods, you can use this method to provide a list of predicates that are evaluated against the managed objects and a list of corresponding error messages (which can be localized).

### Special Considerations

This method raises an exception if the receiver's model has been used by an object graph manager.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [validationPredicates](#) (page 13)
- [validationWarnings](#) (page 13)

### Declared In

NSPropertyDescription.h

## setVersionHashModifier:

Sets the version hash modifier for the receiver.

```
- (void)setVersionHashModifier:(NSString *)modifierString
```

### Parameters

*modifierString*

The version hash modifier for the receiver.

### Discussion

This value is included in the version hash for the property. You use it to mark or denote a property as being a different "version" than another even if all of the values which affect persistence are equal. (Such a difference is important in cases where the attributes of a property are unchanged but the format or content of its data are changed.)

### Availability

Available in Mac OS X v10.5 and later.

### See Also

- [versionHash](#) (page 13)
- [versionHashModifier](#) (page 14)

### Declared In

NSPropertyDescription.h

## userInfo

Returns the user info dictionary of the receiver.

```
- (NSDictionary *)userInfo
```

### Return Value

The user info dictionary of the receiver.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [setUserInfo:](#) (page 11)

### Declared In

NSPropertyDescription.h

## validationPredicates

Returns the validation predicates of the receiver.

- (NSArray \*)validationPredicates

### Return Value

An array containing the receiver's validation predicates.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [validationWarnings](#) (page 13)

- [setValidationPredicates:withValidationWarnings:](#) (page 11)

### Declared In

NSPropertyDescription.h

## validationWarnings

Returns the error strings associated with the receiver's validation predicates.

- (NSArray \*)validationWarnings

### Return Value

An array containing the error strings associated with the receiver's validation predicates.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [validationPredicates](#) (page 13)

- [setValidationPredicates:withValidationWarnings:](#) (page 11)

### Declared In

NSPropertyDescription.h

## versionHash

Returns the version hash for the receiver.

- (NSData \*)versionHash

**Return Value**

The version hash for the receiver.

**Discussion**

The version hash is used to uniquely identify a property based on its configuration. The version hash uses only values which affect the persistence of data and the user-defined [versionHashModifier](#) (page 14) value. (The values which affect persistence are the name of the property, and the flags for `isOptional`, `isTransient`, and `isReadOnly`.) This value is stored as part of the version information in the metadata for stores, as well as a definition of a property involved in an `NSPropertyMapping` object.

**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

- [versionHashModifier](#) (page 14)
- [setVersionHashModifier:](#) (page 12)

**Declared In**

`NSPropertyDescription.h`

**versionHashModifier**

Returns the version hash modifier for the receiver.

```
- (NSString *)versionHashModifier
```

**Return Value**

The version hash modifier for the receiver.

**Discussion**

This value is included in the version hash for the property. See [setVersionHashModifier:](#) (page 12) for a full discussion.

**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

- [versionHash](#) (page 13)
- [setVersionHashModifier:](#) (page 12)

**Declared In**

`NSPropertyDescription.h`

# Document Revision History

---

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

Date	Notes
2006-10-03	Updated for Mac OS X v10.5.
2006-06-28	Enhanced the description of reserved attribute names.
2006-05-23	First publication of this content as a separate document.

**REVISION HISTORY**

Document Revision History



# Index

---

## E

---

entity [instance method 7](#)

## I

---

isIndexed [instance method 7](#)  
isOptional [instance method 8](#)  
isTransient [instance method 8](#)

## N

---

name [instance method 8](#)

## S

---

setIndexed: [instance method 9](#)  
setName: [instance method 9](#)  
setOptional: [instance method 10](#)  
setTransient: [instance method 10](#)  
setUserInfo: [instance method 11](#)  
setValidationPredicates:withValidationWarnings:  
    [instance method 11](#)  
setVersionHashModifier: [instance method 12](#)

## U

---

userInfo [instance method 12](#)

## V

---

validationPredicates [instance method 13](#)  
validationWarnings [instance method 13](#)

versionHash [instance method 13](#)  
versionHashModifier [instance method 14](#)