

---

# NSScriptClassDescription Class Reference

[Cocoa > Scripting & Automation](#)



2007-07-17



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, AppleScript, 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

## **NSScriptClassDescription Class Reference 5**

---

Overview 5

Tasks 6

- Initializing a Script Class Description 6
- Getting a Script Class Description 6
- Getting Basic Information About the Script Class 6
- Getting and Comparing Apple Event Codes 6
- Getting Attribute and Relationship Information 7
- Getting Command Information 7

Class Methods 7

classDescriptionForClass: 7

Instance Methods 8

- appleEventCode 8
- appleEventCodeForKey: 8
- classDescriptionForKey: 9
- className 9
- defaultSubcontainerAttributeKey 10
- hasOrderedToManyRelationshipForKey: 10
- hasPropertyForKey: 10
- hasReadablePropertyForKey: 11
- hasWritablePropertyForKey: 11
- implementationClassName 11
- initWithSuiteName:className:dictionary: 12
- isLocationRequiredToCreateForKey: 12
- keyWithAppleEventCode: 13
- matchesAppleEventCode: 13
- selectorForCommand: 14
- suiteName 14
- superclassDescription 15
- supportsCommand: 15
- typeForKey: 16

## **Appendix A**

## **Deprecated NSScriptClassDescription Methods 17**

---

Deprecated in Mac OS X v10.5 17

isReadOnlyKey: 17

## **Document Revision History 19**

---

## **Index 21**

---



# NSScriptClassDescription Class Reference

---

<b>Inherits from</b>	NSClassDescription : NSObject
<b>Conforms to</b>	NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/Foundation.framework
<b>Availability</b>	Available in Mac OS X v10.0 and later.
<b>Declared in</b>	NSScriptClassDescription.h
<b>Companion guides</b>	Cocoa Scripting Guide Key-Value Coding Programming Guide
<b>Related sample code</b>	Quartz Composer WWDC 2005 TextEdit TextEditPlus

## Overview

An instance of `NSScriptClassDescription` describes a script class that a Cocoa application supports.

A scriptable application provides scriptability information that describes the commands and objects scripters can use in scripts that target the application. That includes information about the classes those scriptable objects are created from.

An application's scriptability information is collected automatically by an instance of `NSScriptSuiteRegistry`. The registry object creates an `NSScriptClassDescription` for each class it finds and caches these objects in memory. Cocoa scripting uses registry information in handling scripting requests that target the application.

A class description instance stores the name, attributes, relationships, and supported commands for a class. For example, a scriptable document class for a drawing application might support attributes such as `file` and `file type`, relationships such as collections of `circles`, `rectangles`, and `lines`, and commands such as `align` and `rotate`.

As with many of the classes in Cocoa's built-in scripting support, your application may never need to directly work with instances of `NSScriptClassDescription`. However, one case where you might need access to a class description is if you override `objectSpecifier` in a scriptable class. For information on how to do this, see *Object Specifiers* in *Cocoa Scripting Guide*.

Another case where your application may need access to class description information is if you override `indicesOfObjectsByEvaluatingWithContainer:count:` in a specifier class.

Although you can subclass `NSScriptClassDescription`, it is unlikely that you would need to do so, or even to create instances of it.

## Tasks

### Initializing a Script Class Description

- `initWithSuiteName:className:dictionary:` (page 12)  
Initializes and returns a newly allocated instance of `NSScriptClassDescription`.

### Getting a Script Class Description

- + `classDescriptionForClass:` (page 7)  
Returns the class description for the specified class or, if it is not scriptable, for the first superclass that is.
- `classDescriptionForKey:` (page 9)  
Returns the class description instance for the class type of the specified attribute or relationship.
- `superclassDescription` (page 15)  
Returns the class description instance for the superclass of the receiver's class.

### Getting Basic Information About the Script Class

- `className` (page 9)  
Returns the name of the class the receiver describes, as provided at initialization time.
- `defaultSubcontainerAttributeKey` (page 10)  
Returns the value of the `DefaultSubcontainerAttribute` entry of the class dictionary from which the receiver was instantiated.
- `implementationClassName` (page 11)  
Returns the name of the Objective-C class instantiated to implement the scripting class.
- `isLocationRequiredToCreateForKey:` (page 12)  
Returns a Boolean value indicating whether an insertion location must be specified when creating a new object in the specified to-many relationship of the receiver.
- `suiteName` (page 14)  
Returns the name of the receiver's suite.

### Getting and Comparing Apple Event Codes

- `appleEventCode` (page 8)  
Returns the Apple event code associated with the receiver's class.
- `appleEventCodeForKey:` (page 8)  
Returns the Apple event code for the specified attribute or relationship in the receiver.

- [matchesAppleEventCode:](#) (page 13)  
Returns a Boolean value indicating whether a primary or secondary Apple event code in the receiver matches the passed code.

## Getting Attribute and Relationship Information

- [hasOrderedToManyRelationshipForKey:](#) (page 10)  
Returns a Boolean value indicating whether the described class has an ordered to-many relationship identified by the specified key.
- [hasPropertyForKey:](#) (page 10)  
Returns a Boolean value indicating whether the described class has a property identified by the specified key.
- [hasReadablePropertyForKey:](#) (page 11)  
Returns a Boolean value indicating whether the described class has a readable property identified by the specified key.
- [hasWritablePropertyForKey:](#) (page 11)  
Returns a Boolean value indicating whether the described class has a writable property identified by the specified key.
- [keyWithAppleEventCode:](#) (page 13)  
Given an Apple event code that identifies a property or element class, returns the key for the corresponding attribute, one-to-one relationship, or one-to-many relationship.
- [typeForKey:](#) (page 16)  
Returns the name of the declared type of the attribute or relationship identified by the passed key.
- [isReadOnlyKey:](#) (page 17) **Deprecated in Mac OS X v10.5**  
Returns a Boolean value indicating whether a specified property in the receiver is read-only. (**Deprecated.** Use [hasWritablePropertyForKey:](#) (page 11) instead.)

## Getting Command Information

- [selectorForCommand:](#) (page 14)  
Returns the selector associated with the receiver for the specified command description.
- [supportsCommand:](#) (page 15)  
Returns a Boolean value indicating whether the receiver or any superclass supports the specified command.

## Class Methods

### **classDescriptionForClass:**

Returns the class description for the specified class or, if it is not scriptable, for the first superclass that is.

```
+ (NSScriptClassDescription *)classDescriptionForClass:(Class)aClass
```

**Parameters***aClass*

The class whose description is needed.

**Return Value**The class description for the class specified by *aClass* or, if that class isn't scriptable, for the class description for the first superclass that is. Returns `nil` if it doesn't find a scriptable class.**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSScriptClassDescription.h

## Instance Methods

### appleEventCode

Returns the Apple event code associated with the receiver's class.

- (FourCharCode)appleEventCode

**Return Value**

The Apple event code associated with the receiver's class. This is the primary code used to identify the class in Apple events.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [appleEventCodeForKey:](#) (page 8)
- [matchesAppleEventCode:](#) (page 13)

**Declared In**

NSScriptClassDescription.h

### appleEventCodeForKey:

Returns the Apple event code for the specified attribute or relationship in the receiver.

- (FourCharCode)appleEventCodeForKey:(NSString \*)key

**Parameters***key*

The identifying key for an attribute or relationship of the receiver.

**Return Value**The four-character Apple event code associated with the attribute or relationship identified by *key* in the receiver or, if none exists, in the class description for the receiver's superclass. Returns 0 if no such attribute or relationship is found.



**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [appleEventCode](#) (page 8)
- [matchesAppleEventCode:](#) (page 13)

**Declared In**

NSScriptClassDescription.h

**classDescriptionForKey:**

Returns the class description instance for the class type of the specified attribute or relationship.

```
- (NSScriptClassDescription *)classDescriptionForKey:(NSString *)key
```

**Parameters**

*key*

The identifying key for an attribute or relationship of the receiver.

**Return Value**

The instance of `NSScriptClassDescription` for the type of the attribute or relationship specified by *key*. Returns `nil` if no scriptable property corresponds to *key*.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [superclassDescription](#) (page 15)

**Declared In**

NSScriptClassDescription.h

**className**

Returns the name of the class the receiver describes, as provided at initialization time.

```
- (NSString *)className
```

**Return Value**

A class name. This may be either the human-readable name for the class—that is, the name that is used in a script—or the name of the Objective-C class that is instantiated to implement the class. To reliably obtain the implementation name, use [implementationClassName](#) (page 11).

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [suiteName](#) (page 14)

**Declared In**

NSScriptClassDescription.h

## defaultSubcontainerAttributeKey

Returns the value of the `defaultSubcontainerAttribute` entry of the class dictionary from which the receiver was instantiated.

- (NSString \*)defaultSubcontainerAttributeKey

### Return Value

The value of the default subcontainer attribute entry. Returns `nil` if there was no such entry.

### Availability

Available in Mac OS X v10.2 and later.

### Declared In

NSScriptClassDescription.h

## hasOrderedToManyRelationshipForKey:

Returns a Boolean value indicating whether the described class has an ordered to-many relationship identified by the specified key.

- (BOOL)hasOrderedToManyRelationshipForKey:(NSString \*)key

### Parameters

*key*

The identifying key for a property of the receiver.

### Return Value

YES if the described class has an ordered to-many relationship identified by the specified key; otherwise, NO.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSScriptClassDescription.h

## hasPropertyForKey:

Returns a Boolean value indicating whether the described class has a property identified by the specified key.

- (BOOL)hasPropertyForKey:(NSString \*)key

### Parameters

*key*

The identifying key for a property of the receiver.

### Return Value

YES if the described class has a property identified by the specified key; otherwise, NO.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSScriptClassDescription.h

## hasReadablePropertyForKey:

Returns a Boolean value indicating whether the described class has a readable property identified by the specified key.

```
- (BOOL)hasReadablePropertyForKey:(NSString *)key
```

### Parameters

*key*

The identifying key for a property of the receiver.

### Return Value

YES if the described class has a readable property identified by the specified key; otherwise, NO.

### Discussion

To determine if a property is read-only, invoke [hasWritablePropertyForKey:](#) (page 11)/

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSScriptClassDescription.h

## hasWritablePropertyForKey:

Returns a Boolean value indicating whether the described class has a writable property identified by the specified key.

```
- (BOOL)hasWritablePropertyForKey:(NSString *)key
```

### Parameters

*key*

The identifying key for a property of the receiver.

### Return Value

YES if the described class has a writable property identified by the specified key; otherwise, NO.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSScriptClassDescription.h

## implementationClassName

Returns the name of the Objective-C class instantiated to implement the scripting class.

```
- (NSString *)implementationClassName
```

### Return Value

An Objective-C class name.

**Discussion**

The name returned by the `className` (page 9) method for an instance of `NSScriptClassDescription` resulting from an `sdef` class declaration is the human-readable name for the class—that is, the name that is used in a script. To obtain the name of the Objective-C class instantiated to implement the class, use `implementationClassName`.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

`NSScriptClassDescription.h`

**initWithSuiteName:className:dictionary:**

Initializes and returns a newly allocated instance of `NSScriptClassDescription`.

```
- (id)initWithSuiteName:(NSString *)suiteName className:(NSString *)className
  dictionary:(NSDictionary *)classDeclaration
```

**Parameters**

*suiteName*

The name of the suite (in the application's scriptability information) that the class belongs to. For example, "AppName Suite".

*className*

The name of the class that this instance describes.

*classDeclaration*

A class declaration dictionary of the sort that is valid in script suite property list files. This dictionary provides information about the class such as its attributes and relationships.

**Return Value**

The initialized instance. Returns `nil` if the event code value for the class description itself is missing or is not an `NSString`. Also returns `nil` if the superclass name or any of the subdictionaries of descriptions are not of the right type.

**Discussion**

This method registers `self` with the application's global instance of `NSScriptSuiteRegistry`.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

`NSScriptClassDescription.h`

**isLocationRequiredToCreateForKey:**

Returns a Boolean value indicating whether an insertion location must be specified when creating a new object in the specified to-many relationship of the receiver.

```
- (BOOL)isLocationRequiredToCreateForKey:(NSString *)toManyRelationshipKey
```

**Parameters***toManyRelationshipKey*

The key for the to-many relationship that may require an insertion location.

**Return Value**

YES if an insertion location must be specified; otherwise, NO.

**Discussion**

A script command object that creates a new object in a to-many relationship needs to know whether an explicitly specified insertion location is required. It can get this information from an instance of `NSScriptClassDescription`. For example, `NSMakeCommand` uses this method to determine whether or not a specific make AppleScript command must have an `at` parameter.

**Availability**

Available in Mac OS X v10.2 and later.

**Declared In**

`NSScriptClassDescription.h`

**keyWithAppleEventCode:**

Given an Apple event code that identifies a property or element class, returns the key for the corresponding attribute, one-to-one relationship, or one-to-many relationship.

```
- (NSString *)keyWithAppleEventCode:(FourCharCode)appleEventCode
```

**Parameters***appleEventCode*

An Apple event code that identifies a property or element class.

**Return Value**

The key that corresponds to the property or element class identified by *appleEventCode* in the receiver or, if none exists, in a class description in the receiver's superclasses.

The four-character Apple event code associated with the attribute or relationship identified by *key*. Returns 0 if no such attribute or relationship is found. Returns `nil` if it cannot find any such attribute or relationship.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [isReadOnlyKey:](#) (page 17)
- [typeForKey:](#) (page 16)

**Declared In**

`NSScriptClassDescription.h`

**matchesAppleEventCode:**

Returns a Boolean value indicating whether a primary or secondary Apple event code in the receiver matches the passed code.

```
- (BOOL)matchesAppleEventCode:(FourCharCode)appleEventCode
```

**Parameters***appleEventCode*

An Apple event code to compare against the receiver's primary or secondary codes.

**Return Value**

YES if the receiver's primary four-character Apple event code or any of its secondary codes (its synonyms) matches *code*; otherwise, NO.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [appleEventCode](#) (page 8)
- [appleEventCodeForKey:](#) (page 8)

**Declared In**

NSScriptClassDescription.h

**selectorForCommand:**

Returns the selector associated with the receiver for the specified command description.

```
- (SEL)selectorForCommand:(NSScriptCommandDescription *)commandDescription
```

**Parameters***commandDescription*

A description for a script command, such as `duplicate`, `make`, or `move`. Encapsulates the scriptability information for that command, such as its Objective-C selector, its argument names and types, and its return type (if any).

**Return Value**

The selector from the receiver for the command specified by *commandDescription*. Searches in the receiver first, then in any superclass. Returns NULL if no matching selector is found.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [supportsCommand:](#) (page 15)

**Declared In**

NSScriptClassDescription.h

**suiteName**

Returns the name of the receiver's suite.

```
- (NSString *)suiteName
```

**Return Value**

The receiver's suite name. Within an application's scriptability information, named suites contain related sets of information.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [className](#) (page 9)

**Declared In**

NSScriptClassDescription.h

**superclassDescription**

Returns the class description instance for the superclass of the receiver's class.

- (NSScriptClassDescription \*)superclassDescription

**Return Value**

A class description instance that describes the superclass of the receiver's class. Returns `nil` if the class has no superclass.

**Discussion**

The instance of `NSScriptClassDescription` that describes the superclass can be in the same suite as the receiver or in a different suite.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

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

**Declared In**

NSScriptClassDescription.h

**supportsCommand:**

Returns a Boolean value indicating whether the receiver or any superclass supports the specified command.

- (BOOL)supportsCommand:(NSScriptCommandDescription \*)commandDescription

**Parameters**

*commandDescription*

A description for a script command, such as `duplicate`, `make`, or `move`. Encapsulates the scriptability information for that command, such as its Objective-C selector, its argument names and types, and its return type (if any).

**Return Value**

YES if an the receiver or the instance of `NSScriptClassDescription` of any superclass of the receiver's class lists the command described by *commandDesc* among its supported commands; otherwise, NO.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [selectorForCommand:](#) (page 14)

**Declared In**

NSScriptClassDescription.h

**typeForKey:**

Returns the name of the declared type of the attribute or relationship identified by the passed key.

```
- (NSString *)typeForKey:(NSString *)key
```

**Parameters**

*key*

The identifying key for an attribute, one-to-one relationship, or one-to-many relationship of the receiver.

**Return Value**

The name of the declared type of the attribute or relationship identified by *key*; for example, “NSString”. Searches in the receiver first, then in any superclass. Returns *nil* if no match is found.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [isReadOnlyKey:](#) (page 17)
- [keyWithAppleEventCode:](#) (page 13)

**Declared In**

NSScriptClassDescription.h



# Deprecated NSScriptClassDescription Methods

---

A method identified as deprecated has been superseded and may become unsupported in the future.

## Deprecated in Mac OS X v10.5

### **isReadOnlyKey:**

Returns a Boolean value indicating whether a specified property in the receiver is read-only. (Deprecated in Mac OS X v10.5. Use [hasWritablePropertyForKey:](#) (page 11) instead.)

- (BOOL)isReadOnlyKey:(NSString \*)key

#### **Parameters**

*key*

The identifying key for a property of the receiver.

#### **Return Value**

YES if the property specified by *key* exists in the receiver or in the `NSScriptClassDescription` for any superclass, and is read only; otherwise, NO.

#### **Special Considerations**

This method could return NO either because *key* is unrecognized or because writing to the property is not supported. Use [hasWritablePropertyForKey:](#) (page 11) instead.

#### **Availability**

Available in in Mac OS X v10.0.

Deprecated in Mac OS X v10.5.

#### **See Also**

- [keyWithAppleEventCode:](#) (page 13)
- [typeForKey:](#) (page 16)

#### **Declared In**

`NSScriptClassDescription.h`



# Document Revision History

---

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

Date	Notes
2007-07-17	Added new methods for Mac OS X version 10.5.
	The new methods are <a href="#">implementationClassName</a> (page 11), <a href="#">hasPropertyForKey:</a> (page 10), <a href="#">hasOrderedToManyRelationshipForKey:</a> (page 10), <a href="#">hasReadablePropertyForKey:</a> (page 11), and <a href="#">hasWritablePropertyForKey:</a> (page 11).
	Noted that the method <a href="#">isReadOnlyKey:</a> (page 17) is deprecated in Mac OS X v10.5 and that you should use <a href="#">hasWritablePropertyForKey:</a> (page 11) instead.
	Clarified the description for the <a href="#">className</a> (page 9) method.
2006-05-23	First publication of this content as a separate document.

## REVISION HISTORY

### Document Revision History

# Index

---

## A

---

appleEventCode **instance method** [8](#)  
appleEventCodeForKey: **instance method** [8](#)

## C

---

classDescriptionForClass: **class method** [7](#)  
classDescriptionForKey: **instance method** [9](#)  
className **instance method** [9](#)

## D

---

defaultSubcontainerAttributeKey **instance method** [10](#)

## H

---

hasOrderedToManyRelationshipForKey: **instance method** [10](#)  
hasPropertyForKey: **instance method** [10](#)  
hasReadablePropertyForKey: **instance method** [11](#)  
hasWritablePropertyForKey: **instance method** [11](#)

## I

---

implementationClassName **instance method** [11](#)  
initWithSuiteName:className:dictionary:  
**instance method** [12](#)  
isLocationRequiredToCreateForKey: **instance method** [12](#)  
isReadOnlyKey: **instance method** [17](#)

## K

---

keyWithAppleEventCode: **instance method** [13](#)

## M

---

matchesAppleEventCode: **instance method** [13](#)

## S

---

selectorForCommand: **instance method** [14](#)  
suiteName **instance method** [14](#)  
superclassDescription **instance method** [15](#)  
supportsCommand: **instance method** [15](#)

## T

---

typeForKey: **instance method** [16](#)