
NSScriptKeyValueCoding Protocol Reference

[Cocoa > Scripting & Automation](#)



2007-10-31



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, 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

NSScriptKeyValueCoding Protocol Reference 5

Overview	5
Tasks	5
Indexed Access	5
Access by Name, Key, or ID	6
Coercion	6
Instance Methods	6
coerceValue:forKey:	6
insertValue:atIndex:inPropertyWithKey:	6
insertValue:inPropertyWithKey:	7
removeValueAtIndex:fromPropertyWithKey:	7
replaceValueAtIndex:inPropertyWithKey:withValue:	7
valueAtIndex:inPropertyWithKey:	8
valueWithName:inPropertyWithKey:	8
valueWithUniqueID:inPropertyWithKey:	8
Constants	9
NSScriptKeyValueCoding Exception Names	9

Document Revision History 11

Index 13

NSScriptKeyValueCoding Protocol Reference

(informal protocol)

Framework	/System/Library/Frameworks/Foundation.framework
Declared in	NSScriptKeyValueCoding.h
Companion guides	Cocoa Scripting Guide Key-Value Coding Programming Guide

Overview

Cocoa scripting takes advantage of key-value coding to get and set information in scriptable objects. The methods in this category provide additional capabilities for working with key-value coding, including getting and setting key values by index in multivalued keys and coercing (or converting) a key value. Additional methods allow the implementer of a scriptable container class to provide fast access to elements that are being referenced by name and unique ID.

Because Cocoa scripting invokes `setValue:forKey:` and `mutableArrayValueForKey:`, changes to model objects made by AppleScript scripts are observable using automatic key-value observing.

Note: In Mac OS X version 10.3 and earlier, Cocoa scripting did not invoke `setValue:forKey:` or `mutableArrayValueForKey:`, so automatic key-value observing notification was not always done for model object changes caused by scripts. Also, in Mac OS X version 10.4, for backward binary compatibility, Cocoa invokes the now-deprecated method `takeValue:forKey:` instead of `setValue:forKey:`, if `takeValue:forKey:` is overridden.

Tasks

Indexed Access

- [insertValue:atIndex:inPropertyWithKey:](#) (page 6)
Inserts an object at the specified index in the collection specified by the passed key.
- [removeValueAtIndex:fromPropertyWithKey:](#) (page 7)
Removes the object at the specified index from the collection specified by the passed key.
- [replaceValueAtIndex:inPropertyWithKey:withValue:](#) (page 7)
Replaces the object at the specified index in the collection specified by the passed key.
- [valueAtIndex:inPropertyWithKey:](#) (page 8)
Retrieves an indexed object from the collection specified by the passed key.

Access by Name, Key, or ID

- [insertValue:inPropertyWithKey:](#) (page 7)
Inserts an object in the collection specified by the passed key.
- [valueWithName:inPropertyWithKey:](#) (page 8)
Retrieves a named object from the collection specified by the passed key.
- [valueWithUniqueID:inPropertyWithKey:](#) (page 8)
Retrieves an object by ID from the collection specified by the passed key.

Coercion

- [coerceValue:forKey:](#) (page 6)
Uses type info from the class description and `NSScriptCoercionHandler` to attempt to convert *value* for *key* to the proper type, if necessary.

Instance Methods

coerceValue:forKey:

Uses type info from the class description and `NSScriptCoercionHandler` to attempt to convert *value* for *key* to the proper type, if necessary.

```
- (id)coerceValue:(id)value forKey:(NSString *)key
```

Discussion

The method `coerceValueFor<Key>:` is used if it exists.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSScriptKeyValueCoding.h`

insertValue:atIndex:inPropertyWithKey:

Inserts an object at the specified index in the collection specified by the passed key.

```
- (void)insertValue:(id)value atIndex:(NSUInteger)index inPropertyWithKey:(NSString *)key
```

Discussion

The method `insertIn<Key>:atIndex:` is invoked if it exists. If no corresponding scripting-KVC-compliant method (`insertIn<Key>:atIndex:`) is found, this method invokes `mutableArrayValueForKey:` and mutates the result.

Note: Prior to Mac OS X version 10.4, this method did not invoke `-mutableArrayValueForKey:`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSScriptKeyValueCoding.h

insertValue:inPropertyWithKey:

Inserts an object in the collection specified by the passed key.

```
- (void)insertValue:(id)value inPropertyWithKey:(NSString *)key
```

Discussion

The method `insertIn<Key>`: is used if it exists. Otherwise, raises an `NSUndefinedKeyException`. This is part of Cocoa's scripting support for inserting newly-created objects into containers without explicitly specifying a location.

Availability

Available in Mac OS X v10.2 and later.

Declared In

NSScriptKeyValueCoding.h

removeValueAtIndex:fromPropertyWithKey:

Removes the object at the specified index from the collection specified by the passed key.

```
- (void)removeValueAtIndex:(NSUInteger)index fromPropertyWithKey:(NSString *)key
```

Discussion

The method `removeFrom<Key>AtIndex:` is invoked if it exists. If no corresponding scripting-KVC-compliant method (`-removeFrom<Key>AtIndex:`) is found, this method invokes `-mutableArrayValueForKey:` and mutates the result.

Note: Prior to Mac OS X version 10.4, this method did not invoke `-mutableArrayValueForKey:`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSScriptKeyValueCoding.h

replaceValueAtIndex:inPropertyWithKey:withValue:

Replaces the object at the specified index in the collection specified by the passed key.

```
- (void)replaceValueAtIndex:(NSUInteger)index inPropertyWithKey:(NSString *)key
    withValue:(id)value
```

Discussion

The method `replaceIn<Key>:atIndex:` is invoked if it exists. If no corresponding scripting-KVC-compliant method (`-replaceIn<Key>atIndex:`) is found, this method invokes `-mutableArrayValueForKey:` and mutates the result.

Note: Prior to Mac OS X version 10.4, this method did not invoke `-mutableArrayValueForKey:`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSScriptKeyValueCoding.h

valueAtIndex:inPropertyWithKey:

Retrieves an indexed object from the collection specified by the passed key.

```
- (id)valueAtIndex:(NSUInteger)index inPropertyWithKey:(NSString *)key
```

Discussion

This actually works with a single-value key as well if `index` is 0. The method `valueIn<Key>AtIndex:` is used if it exists.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSScriptKeyValueCoding.h

valueWithName:inPropertyWithKey:

Retrieves a named object from the collection specified by the passed key.

```
- (id)valueWithName:(NSString *)name inPropertyWithKey:(NSString *)key
```

Discussion

The method `valueIn<Key>WithName:` is used if it exists. Otherwise, raises an `NSUndefinedKeyException`.

Availability

Available in Mac OS X v10.2 and later.

Declared In

NSScriptKeyValueCoding.h

valueWithUniqueID:inPropertyWithKey:

Retrieves an object by ID from the collection specified by the passed key.


```
- (id)valueWithUniqueID:(id)uniqueID inPropertyWithKey:(NSString *)key
```

Discussion

The method `valueIn<Key>WithUniqueID:` is invoked if it exists. Otherwise, raises an `NSUndefinedKeyException`. The declared type of `uniqueID` in the constructed method must be `id`, `NSNumber *`, `NSString *`, or one of the scalar types that can be encapsulated by `NSNumber`.

Availability

Available in Mac OS X v10.2 and later.

Declared In

`NSScriptKeyValueCoding.h`

Constants

NSScriptKeyValueCoding Exception Names

`NSScriptKeyValueCoding` defines the following exception.

```
extern NSString *NSOperationNotSupportedForKeyException;
```

Constants

`NSOperationNotSupportedForKeyException`

Can be raised by key-value coding methods that want to explicitly disallow certain manipulations or accesses.

For instance, a `setKey:` method for a read-only key can raise this exception.

Available in Mac OS X v10.0 and later.

Declared in `NSScriptKeyValueCoding.h`.

Declared In

`NSScriptKeyValueCoding.h`

Document Revision History

This table describes the changes to *NSScriptKeyValueCoding Protocol Reference*.

Date	Notes
2007-10-31	Changed occurrences of <code>NSUnknownKeyException</code> to <code>NSUndefinedKeyException</code> .
2006-05-23	First publication of this content as a separate document.

REVISION HISTORY

Document Revision History

Index

C

`coerceValue(forKey: <NSObject>)` instance method
[6](#)

I

`insertValue:atIndex:inPropertyWithKey:
<NSObject>` instance method [6](#)
`insertValue:inPropertyWithKey: <NSObject>`
instance method [7](#)

N

`NSOperationNotSupportedForKeyException`
constant [9](#)
`NSScriptKeyValueCodingExceptionNames` [9](#)

R

`removeValueAtIndex:fromPropertyWithKey:
<NSObject>` instance method [7](#)
`replaceValueAtIndex:inPropertyWithKey:withValue:
<NSObject>` instance method [7](#)

V

`valueAtIndex:inPropertyWithKey: <NSObject>`
instance method [8](#)
`valueWithName:inPropertyWithKey: <NSObject>`
instance method [8](#)
`valueWithUniqueID:inPropertyWithKey:
<NSObject>` instance method [8](#)