

---

# NSTreeController Class Reference

[Cocoa > Data Management](#)



2007-04-10



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

## **NSTreeController Class Reference 5**

---

Overview	5
Adopted Protocols	5
Tasks	6
Managing Sort Descriptors	6
Setting the Content	6
Arranging Objects	6
Getting the Current Selection	6
Managing Selections	7
Adding, Inserting and Removing Objects	7
Specifying Model Attributes	8
Instance Methods	9
add:	9
addChild:	9
addSelectionIndexPaths:	10
alwaysUsesMultipleValuesMarker	10
arrangedObjects	10
avoidsEmptySelection	11
canAddChild	11
canInsert	12
canInsertChild	12
childrenKeyPath	12
childrenKeyPathForNode:	13
content	13
countKeyPath	13
countKeyPathForNode:	14
insert:	14
insertChild:	14
insertObject:atArrangedObjectIndexPath:	15
insertObjects:atArrangedObjectIndexPaths:	15
leafKeyPath	16
leafKeyPathForNode:	16
moveNode:toIndexPath:	16
moveNodes:toIndexPath:	17
preservesSelection	17
rearrangeObjects	17
remove:	18
removeObjectAtArrangedObjectIndexPath:	18
removeObjectsAtArrangedObjectIndexPaths:	18
removeSelectionIndexPaths:	19
selectedNodes	19

- selectedObjects 19
- selectionIndexPath 20
- selectionIndexPaths 20
- selectsInsertedObjects 20
- setAlwaysUsesMultipleValuesMarker: 21
- setAvoidsEmptySelection: 21
- setChildrenKeyPath: 21
- setContent: 22
- setCountKeyPath: 22
- setLeafKeyPath: 22
- setPreservesSelection: 23
- setSelectionIndexPath: 23
- setSelectionIndexPaths: 24
- setSelectsInsertedObjects: 24
- setSortDescriptors: 24
- sortDescriptors 25

---

**Document Revision History 27**

---

**Index 29**

---

# NSTreeController Class Reference

---

<b>Inherits from</b>	NSObjectController : NSController : NSObject
<b>Conforms to</b>	NSCoding (NSController) NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/AppKit.framework
<b>Availability</b>	Available in Mac OS X v10.4 and later.
<b>Companion guide</b>	Cocoa Bindings Programming Topics
<b>Declared in</b>	NSTreeController.h
<b>Related sample code</b>	CoreRecipes QTMetadataEditor

## Overview

The NSTreeController is a bindings compatible controller that manages a tree of objects. It provides selection and sort management. Its primary purpose is to act as the controller when binding NSOutlineView and NSBrowser instances to a hierarchial collection of objects.

An NSTreeController requires that you describe how the tree of objects is traversed by specifying the key path for child objects. All child objects for the tree must be key-value-coding compliant for the same child key path. If necessary you should implement accessor methods in your model classes, or categories on those classes, that map the child key to the appropriate class-specific method name.

An optional count key path can be specified that, if provided, returns the number of child objects available. Your model objects are expected to update the value of the count key path in a key-value-observing compliant method. You can optionally provide a leaf key path that specifies a key in your model object that returns YES if the object is a leaf node, and NO if it is not. Providing this key path prevents the NSTreeController from having to determine if a child object is a leaf node by examining the child object and as a result improve performance.

The root content object can be a single object, or an array of objects.

## Adopted Protocols

NSCoding  
- encodeWithCoder:

- initWithCoder:

#### NSCopying

- copyWithZone:

## Tasks

### Managing Sort Descriptors

- [setSortDescriptors:](#) (page 24)  
Sets the sort descriptors used to arrange the receiver's contents.
- [sortDescriptors](#) (page 25)  
Returns an array containing the sort descriptors used to arrange the receiver's content.

### Setting the Content

- [setContent:](#) (page 22)  
Sets the receiver's content to *content*.
- [content](#) (page 13)  
Returns the receiver's content object.

### Arranging Objects

- [arrangedObjects](#) (page 10)  
Returns a proxy root tree node containing the receiver's sorted content objects.
- [rearrangeObjects](#) (page 17)  
Use this method to trigger reordering of the receiver's content.

### Getting the Current Selection

- [setSelectionIndexPath:](#) (page 23)  
Sets the receiver's current selection to *indexPath*, returning YES if the selection was changed.
- [selectionIndexPath](#) (page 20)  
Returns the index path of the first object in the receiver's selection, or *nil* if there is no selection.
- [setSelectionIndexPaths:](#) (page 24)  
Sets the receiver's current selection to *indexPaths*, returning YES if the selection was changed.
- [selectionIndexPaths](#) (page 20)  
Returns an array containing the index paths of the receiver's currently selected objects in the content.
- [selectedObjects](#) (page 19)  
Returns an array containing the receiver's selected objects.

- [selectedNodes](#) (page 19)  
Returns an array of the receiver's selected tree nodes.

## Managing Selections

- [setSelectsInsertedObjects:](#) (page 24)  
Sets whether the receiver will automatically select objects as they are inserted.
- [selectsInsertedObjects](#) (page 20)  
Returns whether the receiver selects inserted objects automatically.
- [addSelectionIndexPaths:](#) (page 10)  
Adds the objects at the specified *indexPaths* in the receiver's content to the current selection.
- [removeSelectionIndexPaths:](#) (page 19)  
Removes the objects at the specified *indexPaths* from the receiver's current selection, returning YES if the selection was changed.
- [setAvoidsEmptySelection:](#) (page 21)  
Sets whether the receiver will attempt to avoid an empty selection.
- [avoidsEmptySelection](#) (page 11)  
Returns whether the receiver requires that the content array attempt to maintain a selection at all times.
- [setPreservesSelection:](#) (page 23)  
Sets whether the receiver will attempt to preserve selection when the content changes.
- [preservesSelection](#) (page 17)  
Returns whether the receiver will attempt to preserve the current selection when the content changes.
- [setAlwaysUsesMultipleValuesMarker:](#) (page 21)  
Sets whether the receiver always returns the multiple values marker when multiple objects are selected, even if they have the same value.
- [alwaysUsesMultipleValuesMarker](#) (page 10)  
Returns whether the receiver always returns the multiple values marker when multiple objects are selected, even if the selected items have the same value.

## Adding, Inserting and Removing Objects

- [add:](#) (page 9)  
Adds an object to the receiver after the current selection.
- [addChild:](#) (page 9)  
Adds a child object to the currently selected item.
- [canAddChild](#) (page 11)  
Returns YES if a child object can be added to the receiver's content.
- [canInsert](#) (page 12)  
Returns YES if an object can be inserted into the receiver's content.
- [canInsertChild](#) (page 12)  
Returns YES if a child object can be inserted into the receiver's content.
- [insert:](#) (page 14)  
Creates a new object of the class specified by *objectClass* and inserts it into the receiver's content.

- [insertChild:](#) (page 14)  
Creates a new object of the class specified by `objectClass` and inserts it into the receiver's content as a child of the current selection.
- [insertObject:atArrangedObjectIndexPath:](#) (page 15)  
Inserts `object` into the receiver's arranged objects array at the location specified by `indexPath`, and adds it to the receiver's content.
- [insertObjects:atArrangedObjectIndexPaths:](#) (page 15)  
Inserts `objects` into the receiver's arranged objects array at the locations specified in `indexPaths`, and adds them to the receiver's content.
- [remove:](#) (page 18)  
Removes the receiver's selected objects from the content.
- [removeObjectAtArrangedObjectIndexPath:](#) (page 18)  
Removes the object at the specified `indexPath` in the receiver's arranged objects from the receiver's content.
- [removeObjectsAtArrangedObjectIndexPaths:](#) (page 18)  
Removes the objects at the specified `indexPaths` in the receiver's arranged objects from the receiver's content.
- [moveNode:toIndexPath:](#) (page 16)  
Moves the specified tree node to the new index path.
- [moveNodes:toIndexPath:](#) (page 17)  
Moves the specified tree nodes to the new index path.

## Specifying Model Attributes

- [setChildrenKeyPath:](#) (page 21)  
Sets the key path used by the receiver to access child objects to `key`.
- [childrenKeyPath](#) (page 12)  
Returns the key path used to find the children in the receiver's objects.
- [setCountKeyPath:](#) (page 22)  
Sets the key path used by the receiver to determine the number of objects at a node to `key`.
- [childrenKeyPathForNode:](#) (page 13)  
Returns the key path used to find the children in the specified tree node.
- [countKeyPath](#) (page 13)  
Returns the key path used to find the number of children for a node.
- [countKeyPathForNode:](#) (page 14)  
Returns the key path that provides the number of children for a specified node.
- [setLeafKeyPath:](#) (page 22)  
Sets the key path used by the receiver to determine if an object is a leaf node to `key`.
- [leafKeyPath](#) (page 16)  
Returns the key path used by the receiver to determine if a node is a leaf key.
- [leafKeyPathForNode:](#) (page 16)  
Returns the key path that specifies whether the node is a leaf node.



## Instance Methods

### add:

Adds an object to the receiver after the current selection.

- (void)add:(id)sender

#### Discussion

The *sender* is typically the object that invoked this method.

If the receiver is in object mode `newObject` is called and the returned object is added to the collection. If the receiver is in entity mode a new object is created that is appropriate as specified by the entity, and `newObject` is not used.

#### Special Considerations

Beginning with Mac OS X v10.4 the result of this method is deferred until the next iteration of the runloop so that the error presentation mechanism can provide feedback as a sheet.

#### Availability

Available in Mac OS X v10.4 and later.

#### See Also

- [remove:](#) (page 18)

#### Declared In

NSTreeController.h

### addChild:

Adds a child object to the currently selected item.

- (void)addChild:(id)sender

#### Discussion

The *sender* is typically the object that invoked this method.

If the receiver is in object mode `newObject` is called and the returned object is added as a child. If the receiver is in entity mode a new object is created that is appropriate for the relationship as specified by the entity, and `newObject` is not used.

#### Special Considerations

Beginning with Mac OS X v10.4 the result of this method is deferred until the next iteration of the runloop so that the error presentation mechanism can provide feedback as a sheet.

#### Availability

Available in Mac OS X v10.4 and later.

#### See Also

- [remove:](#) (page 18)

**Declared In**

NSTreeController.h

**addSelectionIndexPaths:**

Adds the objects at the specified *indexPaths* in the receiver's content to the current selection.

- (BOOL)addSelectionIndexPaths:(NSArray \*)*indexPaths*

**Discussion**

Attempting to change the selection may cause a `commitEditing` message which fails, thus denying the selection change.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [removeSelectionIndexPaths:](#) (page 19)

**Declared In**

NSTreeController.h

**alwaysUsesMultipleValuesMarker**

Returns whether the receiver always returns the multiple values marker when multiple objects are selected, even if the selected items have the same value.

- (BOOL)alwaysUsesMultipleValuesMarker

**Discussion**

The default is NO.

This property is observable using key-value observing.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [setAlwaysUsesMultipleValuesMarker:](#) (page 21)

**Declared In**

NSTreeController.h

**arrangedObjects**

Returns a proxy root tree node containing the receiver's sorted content objects.

- (id)arrangedObjects

**Discussion**

This property is observable using key-value observing.

**Special Considerations**

Prior to Mac OS X v10.5 this method returned an opaque root node representing all the currently displayed objects. This method should be used for binding, no assumption should be made about what methods this object supports.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [rearrangeObjects](#) (page 17)

**Declared In**

NSTreeController.h

**avoidsEmptySelection**

Returns whether the receiver requires that the content array attempt to maintain a selection at all times.

- (BOOL)avoidsEmptySelection

**Discussion**

The default is YES.

This property is observable using key-value observing.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [setAvoidsEmptySelection:](#) (page 21)

**Declared In**

NSTreeController.h

**canAddChild**

Returns YES if a child object can be added to the receiver's content.

- (BOOL)canAddChild

**Discussion**

This property is observable using key-value observing.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [canInsertChild](#) (page 12)

**Declared In**

NSTreeController.h

## canInsert

Returns YES if an object can be inserted into the receiver's content.

- (BOOL)canInsert

### Discussion

This property is observable using key-value observing.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [canInsertChild](#) (page 12)

### Declared In

NSTreeController.h

## canInsertChild

Returns YES if a child object can be inserted into the receiver's content.

- (BOOL)canInsertChild

### Discussion

This property is observable using key-value observing.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

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

### Declared In

NSTreeController.h

## childrenKeyPath

Returns the key path used to find the children in the receiver's objects.

- (NSString \*)childrenKeyPath

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [setChildrenKeyPath:](#) (page 21)

### Declared In

NSTreeController.h

**childrenKeyPathForNode:**

Returns the key path used to find the children in the specified tree node.

- (NSString \*)childrenKeyPathForNode:(NSTreeNode \*)*node*

**Parameters**

*node*

A tree node in the receiver.

**Return Value**

A string containing the key path in *node* that provides the child nodes.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSTreeController.h

**content**

Returns the receiver's content object.

- (id)content

**Discussion**

This property is observable using key-value observing.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [setContent:](#) (page 22)

**Declared In**

NSTreeController.h

**countKeyPath**

Returns the key path used to find the number of children for a node.

- (NSString \*)countKeyPath

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [setCountKeyPath:](#) (page 22)

**Declared In**

NSTreeController.h

**countKeyPathForNode:**

Returns the key path that provides the number of children for a specified node.

```
- (NSString *)countKeyPathForNode:(NSTreeNode *)node
```

**Parameters**

*node*

A tree node in the receiver.

**Return Value**

A string containing the key path in *node* that provides the number of children.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSTreeController.h

**insert:**

Creates a new object of the class specified by `objectClass` and inserts it into the receiver's content.

```
- (void)insert:(id)sender
```

**Discussion**

The *sender* is typically the object that invoked this method.

If the receiver is in object mode `newObject` is called and the returned object is inserted into the collection. If the receiver is in entity mode a new object is created that is appropriate as specified by the entity, and `newObject` is not used.

**Special Considerations**

Beginning with Mac OS X v10.4 the result of this method is deferred until the next iteration of the runloop so that the error presentation mechanism can provide feedback as a sheet.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

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

**Declared In**

NSTreeController.h

**insertChild:**

Creates a new object of the class specified by `objectClass` and inserts it into the receiver's content as a child of the current selection.

```
- (void)insertChild:(id)sender
```

**Discussion**

The *sender* is typically the object that invoked this method.

If the receiver is in object mode `newObject` is called and the returned object is inserted as a child. If the receiver is in entity mode a new object is created that is appropriate for the relationship as specified by the entity, and `newObject` is not used.

### Special Considerations

Beginning with Mac OS X v10.4 the result of this method is deferred until the next iteration of the runloop so that the error presentation mechanism can provide feedback as a sheet.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

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

### Declared In

NSTreeController.h

## insertObject:atArrangedObjectIndexPath:

Inserts *object* into the receiver's arranged objects array at the location specified by *indexPath*, and adds it to the receiver's content.

```
- (void)insertObject:(id)object atArrangedObjectIndexPath:(NSIndexPath *)indexPath
```

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [insertObjects:atArrangedObjectIndexPaths:](#) (page 15)

### Declared In

NSTreeController.h

## insertObjects:atArrangedObjectIndexPaths:

Inserts *objects* into the receiver's arranged objects array at the locations specified in *indexPaths*, and adds them to the receiver's content.

```
- (void)insertObjects:(NSArray *)objects atArrangedObjectIndexPaths:(NSArray *)indexPaths
```

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [removeObjectAtArrangedObjectIndexPath:](#) (page 18)

### Declared In

NSTreeController.h

## leafKeyPath

Returns the key path used by the receiver to determine if a node is a leaf key.

- (NSString \*)leafKeyPath

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [setLeafKeyPath:](#) (page 22)

### Declared In

NSTreeController.h

## leafKeyPathForNode:

Returns the key path that specifies whether the node is a leaf node.

- (NSString \*)leafKeyPathForNode:(NSTreeNode \*)node

### Parameters

*node*

A tree node in the receiver.

### Return Value

A string containing the key path in *node* that specifies that the node is a leaf node.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSTreeController.h

## moveNode:toIndexPath:

Moves the specified tree node to the new index path.

- (void)moveNode:(NSTreeNode \*)node toIndexPath:(NSIndexPath \*)indexPath

### Parameters

*node*

A tree node.

*indexPath*

An index path specifying the new position in the receiver's content.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSTreeController.h



## moveNodes:toIndexPath:

Moves the specified tree nodes to the new index path.

```
- (void)moveNodes:(NSArray *)nodes toIndexPath:(NSIndexPath *)startingIndexPath
```

### Parameters

*nodes*

An array of tree nodes.

*startingIndexPath*

An index path specifying the starting position to move the tree nodes to in the receiver's content.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSTreeController.h

## preservesSelection

Returns whether the receiver will attempt to preserve the current selection when the content changes.

```
- (BOOL)preservesSelection
```

### Discussion

The default is YES.

This property is observable using key-value observing.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [setPreservesSelection:](#) (page 23)

### Declared In

NSTreeController.h

## rearrangeObjects

Use this method to trigger reordering of the receiver's content.

```
- (void)rearrangeObjects
```

### Discussion

Subclasses should invoke this method if any parameter that affects the arranged objects changes.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [arrangedObjects](#) (page 10)

**Declared In**

NSTreeController.h

**remove:**

Removes the receiver's selected objects from the content.

- (void)remove:(id)sender

**Discussion**

The *sender* is typically the object that invoked this method.

**Special Considerations**

Beginning with Mac OS X v10.4 the result of this method is deferred until the next iteration of the runloop so that the error presentation mechanism can provide feedback as a sheet.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

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

**Declared In**

NSTreeController.h

**removeObjectAtArrangedObjectIndexPath:**

Removes the object at the specified *indexPath* in the receiver's arranged objects from the receiver's content.

- (void)removeObjectAtArrangedObjectIndexPath:(NSIndexPath \*)indexPath

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [removeObjectsAtArrangedObjectIndexPaths:](#) (page 18)

**Declared In**

NSTreeController.h

**removeObjectsAtArrangedObjectIndexPaths:**

Removes the objects at the specified *indexPaths* in the receiver's arranged objects from the receiver's content.

- (void)removeObjectsAtArrangedObjectIndexPaths:(NSArray \*)indexPaths

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [removeObjectAtArrangedObjectIndexPath:](#) (page 18)

**Declared In**

NSTreeController.h

**removeSelectionIndexPaths:**

Removes the objects at the specified `indexPaths` from the receiver's current selection, returning YES if the selection was changed.

```
- (BOOL)removeSelectionIndexPaths:(NSArray *)indexPaths
```

**Discussion**

Attempting to change the selection may cause a `commitEditing` message which fails, thus denying the selection change.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

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

**Declared In**

NSTreeController.h

**selectedNodes**

Returns an array of the receiver's selected tree nodes.

```
- (NSArray *)selectedNodes
```

**Return Value**

An array containing the receiver's selected tree nodes

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

NSTreeController.h

**selectedObjects**

Returns an array containing the receiver's selected objects.

```
- (NSArray *)selectedObjects
```

**Discussion**

This property is observable using key-value observing.

**Availability**

Available in Mac OS X v10.4 and later.

**Related Sample Code**

QTMetadataEditor

**Declared In**

NSTreeController.h

**selectionIndexPath**Returns the index path of the first object in the receiver's selection, or `nil` if there is no selection.

- (NSIndexPath \*)selectionIndexPath

**Discussion**

This property is observable using key-value observing.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**- [selectionIndexPaths](#) (page 20)**Declared In**

NSTreeController.h

**selectionIndexPaths**

Returns an array containing the index paths of the receiver's currently selected objects in the content.

- (NSArray \*)selectionIndexPaths

**Discussion**

This property is observable using key-value observing.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**- [selectionIndexPath](#) (page 20)**Declared In**

NSTreeController.h

**selectsInsertedObjects**

Returns whether the receiver selects inserted objects automatically.

- (BOOL)selectsInsertedObjects

**Discussion**

The default is YES.

This property is observable using key-value observing.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [setSelectsInsertedObjects:](#) (page 24)

**Declared In**

NSTreeController.h

**setAlwaysUsesMultipleValuesMarker:**

Sets whether the receiver always returns the multiple values marker when multiple objects are selected, even if they have the same value.

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

**Discussion**

Setting *flag* to YES can increase performance if your application doesn't allow editing multiple values. The default is NO.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [alwaysUsesMultipleValuesMarker](#) (page 10)

**Declared In**

NSTreeController.h

**setAvoidsEmptySelection:**

Sets whether the receiver will attempt to avoid an empty selection.

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

**Discussion**

If *flag* is YES then the receiver will maintain a selection unless there are no objects in the content. The default is YES.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [avoidsEmptySelection](#) (page 11)

**Declared In**

NSTreeController.h

**setChildrenKeyPath:**

Sets the key path used by the receiver to access child objects to *key*.

```
- (void)setChildrenKeyPath:(NSString *)key
```

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [childrenKeyPath](#) (page 12)

**Declared In**

NSTreeController.h

**setContent:**

Sets the receiver's content to *content*.

- (void)setContent:(id)*content*

**Discussion**

The *content* can be an array of objects, or a single root object.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [content](#) (page 13)

**Declared In**

NSTreeController.h

**setCountKeyPath:**

Sets the key path used by the receiver to determine the number of objects at a node to *key*.

- (void)setCountKeyPath:(NSString \*)*key*

**Discussion**

Specifying this key path, if the data is available in the model object, can increase performance, but disables insert and remove functionality.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [countKeyPath](#) (page 13)

**Declared In**

NSTreeController.h

**setLeafKeyPath:**

Sets the key path used by the receiver to determine if an object is a leaf node to *key*.

- (void)setLeafKeyPath:(NSString \*)*key*

**Discussion**

Specifying this key path is optional. If the receiver is able to determine that a node is a leaf node, it can disable inserting or adding children to those nodes.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [leafKeyPath](#) (page 16)

**Declared In**

NSTreeController.h

**setPreservesSelection:**

Sets whether the receiver will attempt to preserve selection when the content changes.

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

**Discussion**

If *flag* is YES then the selection will be preserved, if possible. The default is YES.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [preservesSelection](#) (page 17)

**Declared In**

NSTreeController.h

**setSelectionIndexPath:**

Sets the receiver's current selection to *indexPath*, returning YES if the selection was changed.

```
- (BOOL)setSelectionIndexPath:(NSIndexPath *)indexPath
```

**Discussion**

Attempting to change the selection may cause a `commitEditing` message which fails, thus denying the selection change.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [selectionIndexPaths](#) (page 20)

**Declared In**

NSTreeController.h

## setSelectionIndexPaths:

Sets the receiver's current selection to *indexPaths*, returning YES if the selection was changed.

- (BOOL)setSelectionIndexPaths:(NSArray \*)*indexPaths*

### Discussion

Attempting to change the selection may cause a `commitEditingStyle` message which fails, thus denying the selection change.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [setSelectionIndexPath:](#) (page 23)

### Declared In

NSTreeController.h

## setSelectsInsertedObjects:

Sets whether the receiver will automatically select objects as they are inserted.

- (void)setSelectsInsertedObjects:(BOOL) *flag*

### Discussion

If *flag* is YES then items will be selected upon insertion. The default is YES.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [selectsInsertedObjects](#) (page 20)

### Declared In

NSTreeController.h

## setSortDescriptors:

Sets the sort descriptors used to arrange the receiver's contents.

- (void)setSortDescriptors:(NSArray \*)*sortDescriptors*

### Discussion

The *sortDescriptors* parameter contains an array of `NSSortDescriptor` objects. Setting *sortDescriptors* to `nil` causes the contents to be arranged in their natural order.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [sortDescriptors](#) (page 25)



**Declared In**

NSTreeController.h

**sortDescriptors**

Returns an array containing the sort descriptors used to arrange the receiver's content.

- (NSArray \*)sortDescriptors

**Discussion**

Returns `nil` if the receiver has no sort descriptors configured. This property is observable using key-value observing.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [setSortDescriptors:](#) (page 24)

**Declared In**

NSTreeController.h



# Document Revision History

---

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

Date	Notes
2007-04-10	Updated for Mac OS X v10.5.
2006-05-23	First publication of this content as a separate document.

## REVISION HISTORY

### Document Revision History

# Index

---

## A

---

addChild: [instance method 9](#)  
add: [instance method 9](#)  
addSelectionIndexPaths: [instance method 10](#)  
alwaysUsesMultipleValuesMarker [instance method 10](#)  
arrangedObjects [instance method 10](#)  
avoidsEmptySelection [instance method 11](#)

## C

---

canAddChild [instance method 11](#)  
canInsert [instance method 12](#)  
canInsertChild [instance method 12](#)  
childrenKeyPath [instance method 12](#)  
childrenKeyPathForNode: [instance method 13](#)  
content [instance method 13](#)  
countKeyPath [instance method 13](#)  
countKeyPathForNode: [instance method 14](#)

## I

---

insertChild: [instance method 14](#)  
insert: [instance method 14](#)  
insertObject:atArrangedObjectIndexPath:  
[instance method 15](#)  
insertObjects:atArrangedObjectIndexPaths:  
[instance method 15](#)

## L

---

leafKeyPath [instance method 16](#)  
leafKeyPathForNode: [instance method 16](#)

## M

---

moveNode:toIndexPath: [instance method 16](#)  
moveNodes:toIndexPath: [instance method 17](#)

## P

---

preservesSelection [instance method 17](#)

## R

---

rearrangeObjects [instance method 17](#)  
remove: [instance method 18](#)  
removeObjectAtArrangedObjectIndexPath: [instance method 18](#)  
removeObjectsAtArrangedObjectIndexPaths:  
[instance method 18](#)  
removeSelectionIndexPaths: [instance method 19](#)

## S

---

selectedNodes [instance method 19](#)  
selectedObjects [instance method 19](#)  
selectionIndexPath [instance method 20](#)  
selectionIndexPaths [instance method 20](#)  
selectsInsertedObjects [instance method 20](#)  
setAlwaysUsesMultipleValuesMarker: [instance method 21](#)  
setAvoidsEmptySelection: [instance method 21](#)  
setChildrenKeyPath: [instance method 21](#)  
setContent: [instance method 22](#)  
setCountKeyPath: [instance method 22](#)  
setLeafKeyPath: [instance method 22](#)  
setPreservesSelection: [instance method 23](#)  
setSelectionIndexPath: [instance method 23](#)  
setSelectionIndexPaths: [instance method 24](#)  
setSelectsInsertedObjects: [instance method 24](#)

setSortDescriptors: instance method [24](#)  
sortDescriptors instance method [25](#)