

---

# NSComboBox Class Reference

[Cocoa](#) > [User Experience](#)



2006-05-23



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, 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

## **NSComboBox Class Reference 5**

---

Overview	5
Tasks	6
Initializing a Combo Box	6
Setting Display Attributes	6
Setting a Data Source	6
Working with an Internal List	7
Manipulating the Displayed List	7
Manipulating the Selection	8
Completing the Text Field	8
Displaying and dismissing a combo box	8
Changing selection	8
Instance Methods	8
addItemWithObjectValues:	8
addItemWithObjectValue:	9
completes	9
dataSource	10
deselectItemAtIndex:	10
encodeWithCoder:	10
hasVerticalScroller	11
indexOfItemWithObjectValue:	11
indexOfSelectedItem	12
initWithCoder:	12
insertItemWithObjectValue:atIndex:	13
intercellSpacing	13
isButtonBordered	14
itemHeight	14
itemObjectValueAtIndex:	14
noteNumberOfItemsChanged	15
numberOfItems	15
numberOfVisibleItems	16
objectValueOfSelectedItem	16
objectValues	16
reloadData	17
removeAllItems	17
removeItemAtIndex:	17
removeItemWithObjectValue:	18
scrollItemAtIndexToTop:	18
scrollItemAtIndexToVisible:	19
selectItemAtIndex:	19
selectItemWithObjectValue:	19

- setButtonBordered: 20
- setCompletes: 20
- setDataSource: 21
- setHasVerticalScroller: 21
- setIntercellSpacing: 22
- setItemHeight: 22
- setNumberOfVisibleItems: 22
- setUsesDataSource: 23
- usesDataSource 23
- Delegate Methods 24
  - comboBoxSelectionDidChange: 24
  - comboBoxSelectionIsChanging: 24
  - comboBoxWillDismiss: 24
  - comboBoxWillPopUp: 25
- Notifications 25
  - NSComboBoxSelectionDidChangeNotification 25
  - NSComboBoxSelectionIsChangingNotification 25
  - NSComboBoxWillDismissNotification 25
  - NSComboBoxWillPopUpNotification 26

---

**Document Revision History 27**

---

**Index 29**

---

# NSComboBox Class Reference

---

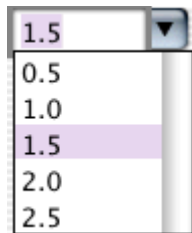
<b>Inherits from</b>	NSTextField : NSControl : NSView : NSResponder : NSObject
<b>Conforms to</b>	NSUserInterfaceValidations (NSTextField) NSAnimatablePropertyContainer (NSView) NSCoding (NSResponder) NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/AppKit.framework
<b>Availability</b>	Available in Mac OS X v10.0 and later.
<b>Companion guide</b>	Combo Box Programming Topics
<b>Declared in</b>	NSComboBox.h
<b>Related sample code</b>	LSMSmartCategorizer QTAudioExtractionPanel QTKitPlayer URL CacheInfo

## Overview

An `NSComboBox` is a kind of `NSControl` that allows you to either enter text directly (as you would with an `NSTextField`) or click the attached arrow at the right of the combo box and select from a displayed (“pop-up”) list of items. It normally looks like this:



When you click the downward-pointing arrow at the right side of the text field, the pop-up list appears, like this:



The `NSComboBox` class uses `NSComboBoxCell` to implement its user interface.

Also see the `NSComboBoxDataSource` informal protocol, which declares the methods that an `NSComboBox` uses to access the contents of its data source object.

## Tasks

### Initializing a Combo Box

- [initWithCoder:](#) (page 12)  
Initializes a newly allocated instance from the data in the given coder.
- [encodeWithCoder:](#) (page 10)  
Encodes the receiver using the specified coder.

### Setting Display Attributes

- [hasVerticalScroller](#) (page 11)  
Returns a Boolean value indicating whether the receiver will display a vertical scroller.
- [intercellSpacing](#) (page 13)  
Returns the horizontal and vertical spacing between cells in the receiver's pop-up list.
- [isButtonBordered](#) (page 14)  
Returns whether the combo box button is set to display a border.
- [itemHeight](#) (page 14)  
Returns the height of each item in the receiver's pop-up list.
- [numberOfVisibleItems](#) (page 16)  
Returns the maximum number of items visible in the pop-up list.
- [setButtonBordered:](#) (page 20)  
Determines whether the button in the combo box is displayed with a border.
- [setHasVerticalScroller:](#) (page 21)  
Determines whether the receiver displays a vertical scroller.
- [setIntercellSpacing:](#) (page 22)  
Sets the spacing between pop-up list items.
- [setItemHeight:](#) (page 22)  
Sets the height for items.
- [setNumberOfVisibleItems:](#) (page 22)  
Sets the maximum number of items that are visible in the receiver's pop-up list.

### Setting a Data Source

- [dataSource](#) (page 10)  
Returns the object that provides the data displayed in the receiver's pop-up list.

- [setDataSource:](#) (page 21)  
Sets the receiver's data source to *aSource*.
- [setUsesDataSource:](#) (page 23)  
Sets whether the receiver uses an external data source to populate the receiver's pop-up list.
- [usesDataSource](#) (page 23)  
Returns a Boolean value indicating whether the receiver uses an external data source to populate its pop-up list.

## Working with an Internal List

- [addItemWithObjectValues:](#) (page 8)  
Adds multiple objects to the end of the receiver's internal item list.
- [addItemWithObjectValue:](#) (page 9)  
Adds an object to the end of the receiver's internal item list.
- [insertItemWithObjectValue:atIndex:](#) (page 13)  
Inserts an object at the specified location in the receiver's internal item list.
- [objectValues](#) (page 16)  
Returns as an array the receiver's internal item list.
- [removeAllItems](#) (page 17)  
Removes all items from the receiver's internal item list.
- [removeItemAtIndex:](#) (page 17)  
Removes the object at the specified location from the receiver's internal item list.
- [removeItemWithObjectValue:](#) (page 18)  
Removes all occurrences of the given object from the receiver's internal item list.
- [numberOfItems](#) (page 15)  
Returns the total number of items in the pop-up list.

## Manipulating the Displayed List

- [indexOfItemWithObjectValue:](#) (page 11)  
Searches the receiver's internal item list for the specified object and returns the lowest matching index.
- [itemObjectValueAtIndex:](#) (page 14)  
Returns the object located at the given index within the receiver's internal item list.
- [noteNumberOfItemsChanged](#) (page 15)  
Informs the receiver that the number of items in its data source has changed.
- [reloadData](#) (page 17)  
Marks the receiver as needing redisplay, so that it will reload the data for visible pop-up items and draw the new values.
- [scrollItemAtIndexToTop:](#) (page 18)  
Scrolls the receiver's pop-up list vertically so that the item at the specified index is as close to the top as possible.
- [scrollItemAtIndexToVisible:](#) (page 19)  
Scrolls the receiver's pop-up list vertically so that the item at the specified index is visible.

## Manipulating the Selection

- [deselectItemAtIndex:](#) (page 10)  
Deselects the pop-up list item at the specified index if it's selected.
- [indexOfSelectedItem](#) (page 12)  
Returns the index of the last item selected from the pop-up list.
- [objectValueOfSelectedItem](#) (page 16)  
Returns the object corresponding to the last item selected from the pop-up list.
- [selectItemAtIndex:](#) (page 19)  
Selects the pop-up list row at the given index.
- [selectItemWithObjectValue:](#) (page 19)  
Selects the first pop-up list item that corresponds to the given object.

## Completing the Text Field

- [completes](#) (page 9)  
Returns a Boolean value indicating whether the receiver tries to complete what the user types in the text field.
- [setCompletes:](#) (page 20)  
Sets whether the receiver tries to complete what the user types in the text field.

## Displaying and dismissing a combo box

- [comboBoxWillDismiss:](#) (page 24) *delegate method*  
Informs the delegate that the pop-up list is about to be dismissed.
- [comboBoxWillPopUp:](#) (page 25) *delegate method*  
Informs the delegate that the pop-up list is about to be displayed.

## Changing selection

- [comboBoxSelectionDidChange:](#) (page 24) *delegate method*  
Informs the delegate that the pop-up list selection has finished changing.
- [comboBoxSelectionIsChanging:](#) (page 24) *delegate method*  
Informs the delegate that the pop-up list selection is changing.

## Instance Methods

### **addItemWithObjectValues:**

Adds multiple objects to the end of the receiver's internal item list.

- (void)addItemWithObjectValues:(NSArray \*)objects



### Parameters

*objects*

An array of the objects to add to the internal item list.

### Discussion

This method logs a warning if [usesDataSource](#) (page 23) returns YES.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

NSComboBox.h

## addItemWithObjectValue:

Adds an object to the end of the receiver's internal item list.

- (void)addItemWithObjectValue:(id)anObject

### Parameters

*anObject*

The object to add to the internal item list.

### Discussion

This method logs a warning if [usesDataSource](#) (page 23) returns YES.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

NSComboBox.h

## completes

Returns a Boolean value indicating whether the receiver tries to complete what the user types in the text field.

- (BOOL)completes

### Return Value

YES if the receiver tries to complete what the user types in the text field; otherwise NO.

### Availability

Available in Mac OS X v10.0 and later.

### See Also

[setCompletes:](#) (page 20)

### Declared In

NSComboBox.h

## dataSource

Returns the object that provides the data displayed in the receiver's pop-up list.

- (id)dataSource

### Return Value

The data source for the combo box's pop-up list.

### Discussion

This method logs a warning if [usesDataSource](#) (page 23) returns NO. See the class description and the [NSComboBoxDataSource](#) informal protocol specification for more information on combo box data source objects.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

NSComboBox.h

## deselectItemAtIndex:

Deselects the pop-up list item at the specified index if it's selected.

- (void)deselectItemAtIndex:(NSInteger) *index*

### Parameters

*index*

The index of the item to deselect.

### Discussion

If the selection does in fact change, this method posts an [NSComboBoxSelectionDidChangeNotification](#) (page 25) to the default notification center.

### Availability

Available in Mac OS X v10.0 and later.

### See Also

- [indexOfSelectedItem](#) (page 12)
- [numberOfItems](#) (page 15)
- [selectItemAtIndex:](#) (page 19)

### Declared In

NSComboBox.h

## encodeWithCoder:

Encodes the receiver using the specified coder.

- (void)encodeWithCoder:(NSCoder \*) *encoder*

**Parameters***encoder*

The coder to use to encode the receiver.

**Discussion**

If the receiver uses a data source, the data source is conditionally encoded as well.

**Availability**

Available in Mac OS X v10.0 through Mac OS X v10.2.

**See Also**

- [initWithCoder:](#) (page 12)

**Declared In**

NSComboBox.h

**hasVerticalScroller**

Returns a Boolean value indicating whether the receiver will display a vertical scroller.

- (BOOL)hasVerticalScroller

**Return Value**

YES if the receiver will display a vertical scroller; otherwise NO.

**Discussion**

Note that the scroller will be displayed even if the pop-up list contains fewer items than will fit in the area specified for display.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [numberOfItems](#) (page 15)

- [numberOfVisibleItems](#) (page 16)

**Declared In**

NSComboBox.h

**indexOfItemWithObjectValue:**

Searches the receiver's internal item list for the specified object and returns the lowest matching index.

- (NSInteger)indexOfItemWithObjectValue:(id)anObject

**Parameters***anObject*

The object for which to return the index.

**Return Value**

The lowest index in the internal item list whose corresponding value is equal to that of the specified object. Objects are considered equal if they have the same `id` or if `isEqual:` returns YES.

If none of the objects in the receiver's internal item list are equal to *anObject*, `indexOfItemWithObjectValue:` returns `NSNotFound`.

#### Discussion

This method logs a warning if `usesDataSource` (page 23) returns YES.

#### Availability

Available in Mac OS X v10.0 and later.

#### See Also

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

#### Declared In

NSComboBox.h

## indexOfSelectedItem

Returns the index of the last item selected from the pop-up list.

- (NSInteger)indexOfSelectedItem

#### Return Value

The index of the last item selected from the receiver's pop-up list or -1 if no item is selected.

#### Discussion

Note that nothing is initially selected in a newly initialized combo box.

#### Availability

Available in Mac OS X v10.0 and later.

#### See Also

- [objectValueOfSelectedItem](#) (page 16)

#### Declared In

NSComboBox.h

## initWithCoder:

Initializes a newly allocated instance from the data in the given coder.

- (id)initWithCoder:(NSCoder \*)*decoder*

#### Parameters

*decoder*

The coder object containing the data for the combo box. If the decoded instance uses a data source, `initWithCoder:` decodes the data source as well.

#### Return Value

The initialized `NSComboBox` object.

#### Availability

Available in Mac OS X v10.0 through Mac OS X v10.2.

**See Also**

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

**Declared In**

NSComboBox.h

**insertItemWithObjectValue:atIndex:**

Inserts an object at the specified location in the receiver's internal item list.

```
- (void)insertItemWithObjectValue:(id)anObject atIndex:(NSInteger)index
```

**Parameters**

*anObject*

The object to add to the internal item list.

*index*

The index in the list at which to add the new object. The previous item at *index*—along with all following items—is shifted down one slot to make room

**Discussion**

This method logs a warning if [usesDataSource](#) (page 23) returns YES.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

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

- [numberOfItems](#) (page 15)

**Declared In**

NSComboBox.h

**intercellSpacing**

Returns the horizontal and vertical spacing between cells in the receiver's pop-up list.

```
- (NSSize)intercellSpacing
```

**Return Value**

The space between cells in the pop-up list. The default spacing is (3.0, 2.0).

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [itemHeight](#) (page 14)

- [numberOfVisibleItems](#) (page 16)

**Declared In**

NSComboBox.h

## isButtonBordered

Returns whether the combo box button is set to display a border.

- (BOOL)isButtonBordered

### Return Value

YES if the button has a border; otherwise NO.

### Availability

Available in Mac OS X v10.3 and later.

### See Also

- [setButtonBordered:](#) (page 20)

### Declared In

NSComboBox.h

## itemHeight

Returns the height of each item in the receiver's pop-up list.

- (CGFloat)itemHeight

### Return Value

The height of items in the pop-up list. The default item height is 16.0.

### Availability

Available in Mac OS X v10.0 and later.

### See Also

- [intercellSpacing](#) (page 13)

- [numberOfVisibleItems](#) (page 16)

### Declared In

NSComboBox.h

## itemObjectValueAtIndex:

Returns the object located at the given index within the receiver's internal item list.

- (id)itemObjectValueAtIndex:(NSInteger) *index*

### Parameters

*index*

The index of the object to retrieve. If *index* is beyond the end of the list, an `NSRangeException` is raised.

### Return Value

The object located at the specified index in the internal item list.

### Discussion

This method logs a warning if [usesDataSource](#) (page 23) returns YES.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [objectValueOfSelectedItem](#) (page 16)

**Declared In**

NSComboBox.h

**noteNumberOfItemsChanged**

Informs the receiver that the number of items in its data source has changed.

- (void)noteNumberOfItemsChanged

**Discussion**

This method allows the receiver to update the scrollers in its displayed pop-up list without actually reloading data into the receiver. It is particularly useful for a data source that continually receives data in the background over a period of time, in which case the `NSComboBox` can remain responsive to the user while the data is received.

See the `NSComboBoxDataSource` informal protocol specification for information on the messages an `NSComboBox` sends to its data source.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [reloadData](#) (page 17)

**Declared In**

NSComboBox.h

**numberOfItems**

Returns the total number of items in the pop-up list.

- (NSInteger)numberOfItems

**Return Value**

The number of items in the list.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [numberOfVisibleItems](#) (page 16)  
 - `numberOfItemsInComboBox`: (`NSComboBoxDataSource` protocol)

**Declared In**

NSComboBox.h

## numberOfVisibleItems

Returns the maximum number of items visible in the pop-up list.

- (NSInteger)numberOfVisibleItems

### Return Value

The maximum number of items visible at any one time in the pop-up list.

### Availability

Available in Mac OS X v10.0 and later.

### See Also

- [numberOfItems](#) (page 15)

### Declared In

NSComboBox.h

## objectValueOfSelectedItem

Returns the object corresponding to the last item selected from the pop-up list.

- (id)objectValueOfSelectedItem

### Return Value

The object in the receiver's internal item list corresponding to the last item selected from the pop-up list, or `nil` if no item is selected.

### Discussion

Note that nothing is initially selected in a newly initialized combo box. This method logs a warning if [usesDataSource](#) (page 23) returns YES.

### Availability

Available in Mac OS X v10.0 and later.

### See Also

- [indexOfSelectedItem](#) (page 12)

- `comboBox:objectValueForItemAtIndex:` (NSComboBoxDataSource protocol)

### Declared In

NSComboBox.h

## objectValues

Returns as an array the receiver's internal item list.

- (NSArray \*)objectValues

### Return Value

The array containing the objects in the receiver's internal item list.

### Discussion

This method logs a warning if [usesDataSource](#) (page 23) returns YES.



**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSComboBox.h

**reloadData**

Marks the receiver as needing redisplay, so that it will reload the data for visible pop-up items and draw the new values.

- (void)reloadData

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [noteNumberOfItemsChanged](#) (page 15)

**Declared In**

NSComboBox.h

**removeAllItems**

Removes all items from the receiver's internal item list.

- (void)removeAllItems

**Discussion**

This method logs a warning if [usesDataSource](#) (page 23) returns YES.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [objectValues](#) (page 16)

**Declared In**

NSComboBox.h

**removeItemAtIndex:**

Removes the object at the specified location from the receiver's internal item list.

- (void)removeItemAtIndex:(NSInteger) *index*

**Parameters**

*index*

The index of the object to remove. All items beyond *index* are moved up one slot to fill the gap.

**Discussion**

The removed object receives a `release` message. This method raises an `NSRangeException` if `index` is beyond the end of the list and logs a warning if `usesDataSource` (page 23) returns YES.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

`NSComboBox.h`

**removeItemWithObjectValue:**

Removes all occurrences of the given object from the receiver's internal item list.

```
- (void)removeItemWithObjectValue:(id)anObject
```

**Parameters**

*anObject*

The object to remove from the internal item list. Objects are considered equal if they have the same `id` or if `isEqual:` returns YES.

**Discussion**

This method logs a warning if `usesDataSource` (page 23) returns YES.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

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

**Declared In**

`NSComboBox.h`

**scrollItemAtIndexToTop:**

Scrolls the receiver's pop-up list vertically so that the item at the specified index is as close to the top as possible.

```
- (void)scrollItemAtIndexToTop:(NSInteger)index
```

**Parameters**

*index*

The index of the item to scroll to the top.

**Discussion**

The pop-up list need not be displayed at the time this method is invoked.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

`NSComboBox.h`

**scrollItemAtIndexToVisible:**

Scrolls the receiver’s pop-up list vertically so that the item at the specified index is visible.

```
- (void)scrollItemAtIndexToVisible:(NSInteger) index
```

**Parameters**

*index*

The index of the item to make visible.

**Discussion**

The pop-up list need not be displayed at the time this method is invoked.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSComboBox.h

**selectItemAtIndex:**

Selects the pop-up list row at the given index.

```
- (void)selectItemAtIndex:(NSInteger) index
```

**Parameters**

*index*

The index of the item to select in the pop-up list.

**Discussion**

Posts an [NSComboBoxSelectionDidChangeNotification](#) (page 25) to the default notification center if the selection does in fact change. Note that this method does not alter the contents of the combo box’s text field—see “Setting the Combo Box’s Value” for more information.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

```
- setObjectValue: (NSControl)
```

**Declared In**

NSComboBox.h

**selectItemWithObjectValue:**

Selects the first pop-up list item that corresponds to the given object.

```
- (void)selectItemWithObjectValue:(id) anObject
```

**Parameters**

*anObject*

The object to select in the pop-up list. Objects are considered equal if they have the same `id` or if `isEqual:` returns YES.

**Discussion**

This method logs a warning if [usesDataSource](#) (page 23) returns YES. Posts an [NSComboBoxSelectionDidChangeNotification](#) (page 25) to the default notification center if the selection does in fact change. Note that this method doesn't alter the contents of the combo box's text field—see “Setting the Combo Box's Value” for more information.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [setObjectValue:](#) (NSControl)

**Declared In**

NSComboBox.h

**setButtonBordered:**

Determines whether the button in the combo box is displayed with a border.

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

**Parameters**

*flag*

YES to display a border; NO to display the button without a border. For example, it is often useful when using a combo box in an [NSTableView](#) to display the button without the border.

**Availability**

Available in Mac OS X v10.3 and later.

**See Also**

- [isButtonBordered](#) (page 14)

**Declared In**

NSComboBox.h

**setCompletes:**

Sets whether the receiver tries to complete what the user types in the text field.

```
- (void)setCompletes:(BOOL)completes
```

**Parameters**

*completes*

YES to indicate that the receiver should try to complete text entered by the user. If *completes* is YES, every time the user adds characters to the end of the text field, the combo box calls the [NSComboBoxCell](#) method `completedString:`.

**Discussion**

If `completedString:` returns a string that's longer than the existing string, the combo box replaces the existing string with the returned string and selects the additional characters. If the user is deleting characters or adds characters somewhere besides the end of the string, the combo box does not try to complete it.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

[completes](#) (page 9)

**Declared In**

NSComboBox.h

**setDataSource:**

Sets the receiver's data source to *aSource*.

```
- (void)setDataSource:(id)aSource
```

**Parameters**

*aSource*

The new data source for the receiver. The data source should implement the appropriate methods of the `NSComboBoxDataSource` informal protocol.

This method logs a warning if *aSource* doesn't respond to either `numberOfItemsInComboBox:` or `comboBox:objectValueForItemAtIndex:`.

**Discussion**

This method doesn't automatically set [usesDataSource](#) (page 23) to NO and in fact logs a warning if [usesDataSource](#) (page 23) returns NO.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

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

**Declared In**

NSComboBox.h

**setHasVerticalScroller:**

Determines whether the receiver displays a vertical scroller.

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

**Parameters**

*flag*

YES to display a vertical scroller; NO otherwise. By default, *flag* is YES.

**Discussion**

If *flag* is NO and the combo box has more list items (either in its internal item list or from its data source) than are allowed by [numberOfVisibleItems](#) (page 16), only a subset are displayed. The `NSComboBox` class' `scroll...` methods can be used to position this subset within the pop-up list.

Note that if *flag* is YES, a scroller will be displayed even if the combo box has fewer list items than are allowed by [numberOfVisibleItems](#) (page 16).

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [numberOfItems](#) (page 15)
- [scrollItemAtIndexToTop:](#) (page 18)
- [scrollItemAtIndexToVisible:](#) (page 19)

**Declared In**

NSComboBox.h

**setIntercellSpacing:**

Sets the spacing between pop-up list items.

```
- (void)setIntercellSpacing:(NSSize)aSize
```

**Parameters***aSize*

The new width and height between pop-up list items. The default intercell spacing is (3.0, 2.0).

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [setItemHeight:](#) (page 22)
- [setNumberOfVisibleItems:](#) (page 22)

**Declared In**

NSComboBox.h

**setItemHeight:**

Sets the height for items.

```
- (void)setItemHeight:(CGFloat)itemHeight
```

**Parameters***itemHeight*

The new height for items in the pop-up list.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [setIntercellSpacing:](#) (page 22)
- [setNumberOfVisibleItems:](#) (page 22)

**Declared In**

NSComboBox.h

**setNumberOfVisibleItems:**

Sets the maximum number of items that are visible in the receiver's pop-up list.

- (void)setNumberOfVisibleItems:(NSInteger)*visibleItems*

#### Parameters

*visibleItems*

The maximum number of items that are visible at one time in the pop-up list.

#### Availability

Available in Mac OS X v10.0 and later.

#### See Also

- [numberOfItems](#) (page 15)
- [setItemHeight:](#) (page 22)
- [setInterCellSpacing:](#) (page 22)

#### Declared In

NSComboBox.h

### setUsesDataSource:

Sets whether the receiver uses an external data source to populate the receiver's pop-up list.

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

#### Parameters

*flag*

YES if the receiver uses an external data source (specified by [setDataSource:](#) (page 21)); otherwise NO.

#### Availability

Available in Mac OS X v10.0 and later.

#### Declared In

NSComboBox.h

### usesDataSource

Returns a Boolean value indicating whether the receiver uses an external data source to populate its pop-up list.

- (BOOL)usesDataSource

#### Return Value

YES if the receiver uses an external data source to populate the receiver's pop-up list, NO if it uses an internal item list.

#### Availability

Available in Mac OS X v10.0 and later.

#### See Also

- [dataSource](#) (page 10)

#### Declared In

NSComboBox.h

## Delegate Methods

### **comboBoxSelectionDidChange:**

Informs the delegate that the pop-up list selection has finished changing.

```
- (void)comboBoxSelectionDidChange:(NSNotification *)notification
```

#### **Parameters**

*notification*

An [NSComboBoxSelectionDidChangeNotification](#) (page 25).

#### **Availability**

Available in Mac OS X v10.0 and later.

#### **Declared In**

NSComboBox.h

### **comboBoxSelectionIsChanging:**

Informs the delegate that the pop-up list selection is changing.

```
- (void)comboBoxSelectionIsChanging:(NSNotification *)notification
```

#### **Parameters**

*notification*

An [NSComboBoxSelectionIsChangingNotification](#) (page 25).

#### **Availability**

Available in Mac OS X v10.0 and later.

#### **Declared In**

NSComboBox.h

### **comboBoxWillDismiss:**

Informs the delegate that the pop-up list is about to be dismissed.

```
- (void)comboBoxWillDismiss:(NSNotification *)notification
```

#### **Parameters**

*notification*

An [NSComboBoxWillDismissNotification](#) (page 25).

#### **Availability**

Available in Mac OS X v10.0 and later.

#### **Declared In**

NSComboBox.h



**comboBoxWillPopUp:**

Informs the delegate that the pop-up list is about to be displayed.

```
- (void)comboBoxWillPopUp:(NSNotification *)notification
```

**Parameters**

*notification*

An [NSComboBoxWillPopUpNotification](#) (page 26).

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSComboBox.h

## Notifications

**NSComboBoxSelectionDidChangeNotification**

Posted after the pop-up list selection of the NSComboBox changes.

The notification object is the NSComboBox whose selection changed. This notification does not contain a *userInfo* dictionary.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSComboBox.h

**NSComboBoxSelectionIsChangingNotification**

Posted whenever the pop-up list selection of the NSComboBox is changing.

The notification object is the NSComboBox whose selection is changing. This notification does not contain a *userInfo* dictionary.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSComboBox.h

**NSComboBoxWillDismissNotification**

Posted whenever the pop-up list of the NSComboBox is about to be dismissed.

The notification object is the NSComboBox whose pop-up list will be dismissed. This notification does not contain a *userInfo* dictionary.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSComboBox.h

**NSComboBoxWillPopUpNotification**

Posted whenever the pop-up list of the NSComboBox is going to be displayed.

The notification object is the NSComboBox whose pop-up window will be displayed. This notification does not contain a *userInfo* dictionary.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSComboBox.h

# Document Revision History

---

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

Date	Notes
2006-05-23	First publication of this content as a separate document.

**REVISION HISTORY**

Document Revision History

# Index

---

## A

---

addItemWithObjectValues: [instance method 8](#)  
addItemWithObjectValue: [instance method 9](#)

## C

---

comboBoxSelectionDidChange: <NSObject> [delegate method 24](#)  
comboBoxSelectionIsChanging: <NSObject> [delegate method 24](#)  
comboBoxWillDismiss: <NSObject> [delegate method 24](#)  
comboBoxWillPopUp: <NSObject> [delegate method 25](#)  
completes [instance method 9](#)

## D

---

dataSource [instance method 10](#)  
deselectItemAtIndex: [instance method 10](#)

## E

---

encodeWithCoder: [instance method 10](#)

## H

---

hasVerticalScroller [instance method 11](#)

## I

---

indexOfItemWithObjectValue: [instance method 11](#)

indexOfSelectedItem [instance method 12](#)  
initWithCoder: [instance method 12](#)  
insertItemWithObjectValue:atIndex: [instance method 13](#)  
intercellSpacing [instance method 13](#)  
isButtonBordered [instance method 14](#)  
itemHeight [instance method 14](#)  
itemObjectValueAtIndex: [instance method 14](#)

## N

---

noteNumberOfItemsChanged [instance method 15](#)  
NSComboBoxSelectionDidChangeNotification [notification 25](#)  
NSComboBoxSelectionIsChangingNotification [notification 25](#)  
NSComboBoxWillDismissNotification [notification 25](#)  
NSComboBoxWillPopUpNotification [notification 26](#)  
numberOfItems [instance method 15](#)  
numberOfVisibleItems [instance method 16](#)

## O

---

objectValueOfSelectedItem [instance method 16](#)  
objectValues [instance method 16](#)

## R

---

reloadData [instance method 17](#)  
removeAllItems [instance method 17](#)  
removeItemAtIndex: [instance method 17](#)  
removeItemWithObjectValue: [instance method 18](#)

## S

---

scrollItemAtIndexToTop: **instance method** [18](#)  
scrollItemAtIndexToVisible: **instance method** [19](#)  
selectItemAtIndex: **instance method** [19](#)  
selectItemWithObjectValue: **instance method** [19](#)  
setButtonBordered: **instance method** [20](#)  
setCompletes: **instance method** [20](#)  
setDataSource: **instance method** [21](#)  
setHasVerticalScroller: **instance method** [21](#)  
setInterCellSpacing: **instance method** [22](#)  
setItemHeight: **instance method** [22](#)  
setNumberOfVisibleItems: **instance method** [22](#)  
setUsesDataSource: **instance method** [23](#)

## U

---

usesDataSource **instance method** [23](#)