
NSControl Class Reference

User Experience: Controls



2009-06-18



Apple Inc.
© 2009 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, Carbon, Cocoa, Mac, Mac OS, Objective-C, and Quartz are trademarks of Apple Inc., registered in the United States and other countries.

Finder and Shuffle are trademarks of Apple Inc.

OpenGL is a registered trademark of Silicon Graphics, Inc.

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

NSControl Class Reference 7

Overview	7
About Delegate Methods	7
Tasks	8
Initializing an NSControl	8
Setting the Control's Cell	8
Enabling and Disabling the Control	8
Identifying the Selected Cell	8
Setting the Control's Value	8
Interacting with Other Controls	9
Formatting Text	10
Managing the Field Editor	10
Resizing the Control	10
Displaying a Cell	10
Implementing the Target/action Mechanism	11
Getting and Setting Tags	11
Activating from the Keyboard	11
Tracking the Mouse	12
Control Editing Notifications	12
Class Methods	12
cellClass	12
setCellClass:	13
Instance Methods	13
abortEditing	13
action	14
alignment	14
attributedStringValue	15
baseWritingDirection	15
calcSize	16
cell	16
currentEditor	17
doubleValue	17
drawCell:	18
drawCellInside:	18
floatValue	19
font	19
formatter	20
ignoresMultiClick	20
initWithFrame:	20
integerValue	21
intValue	21

isContinuous	22
isEnabled	23
mouseDown:	23
objectValue	24
performClick:	24
refusesFirstResponder	25
selectCell:	25
selectedCell	25
selectedTag	26
sendAction:to:	26
sendActionOn:	27
setAction:	28
setAlignment:	28
setAttributedStringValue:	29
setBaseWritingDirection:	29
setCell:	30
setContinuous:	30
setDoubleValue:	31
setEnabled:	31
setFloatValue:	32
setFont:	33
setFormatter:	33
setIgnoresMultiClick:	33
setIntegerValue:	34
setIntValue:	34
setNeedsDisplay	35
setObjectValue:	35
setRefusesFirstResponder:	36
setStringValue:	37
setTag:	37
setTarget:	38
sizeToFit	39
stringValue	39
tag	40
takeDoubleValueFrom:	40
takeFloatValueFrom:	41
takeIntegerValueFrom:	41
takeIntValueFrom:	42
takeObjectValueFrom:	42
takeStringValueFrom:	43
target	43
updateCell:	44
updateCellInside:	44
validateEditing	44
Delegate Methods	45
controlTextDidBeginEditing:	45

controlTextDidChange: 45

controlTextDidEndEditing: 46

Notifications 47

NSControlTextDidBeginEditingNotification 47

NSControlTextDidChangeNotification 47

NSControlTextDidEndEditingNotification 47

Appendix A Deprecated NSControl Methods 49

Deprecated in Mac OS X v10.0 49

setFloatingPointFormat:left:right: 49

Document Revision History 51

Index 53

NSControl Class Reference

Inherits from	NSView : NSResponder : NSObject
Conforms to	NSAnimatablePropertyContainer (NSView) NSCoding (NSResponder) NSObject (NSObject)
Framework	/System/Library/Frameworks/AppKit.framework
Availability	Available in Mac OS X v10.0 and later.
Companion guide	Control and Cell Programming Topics for Cocoa
Declared in	NSControl.h
Related sample code	Clock Control ClockControl ImageClient QTMetadataEditor Quartz Composer QCTV

Overview

`NSControl` is an abstract superclass that provides three fundamental features for implementing user interface devices: drawing devices on the screen, responding to user events, and sending action messages. It works closely with the `NSCell` class.

About Delegate Methods

The `NSControl` class provides several delegate methods for its subclasses that allow text editing, such as `NSTextField` and `NSMatrix`. These include: [controlTextDidBeginEditing:](#) (page 45), [controlTextDidChange:](#) (page 45), and [controlTextDidEndEditing:](#) (page 46).

Note that although `NSControl` defines delegate methods, it does not itself have a delegate. Any subclass that uses these methods must have a delegate and the methods to get and set it. In addition, a format delegate protocol `NSControlTextEditingDelegate` also defines delegate methods used by control delegates.

Tasks

Initializing an NSControl

- `initWithFrame:` (page 20)
Returns an NSControl object initialized with the specified frame rectangle.

Setting the Control's Cell

- + `cellClass` (page 12)
Returns the type of cell used by the receiver.
- + `setCellClass:` (page 13)
Sets the type of cell used by the receiver.
- `cell` (page 16)
Returns the receiver's cell object.
- `setCell:` (page 30)
Sets the receiver's cell

Enabling and Disabling the Control

- `isEnabled` (page 23)
Returns whether the receiver reacts to mouse events.
- `setEnabled:` (page 31)
Sets whether the receiver (and its cell) reacts to mouse events.

Identifying the Selected Cell

- `selectedCell` (page 25)
Returns the receiver's selected cell.
- `selectedTag` (page 26)
Returns the tag of the receiver's selected cell.

Setting the Control's Value

- `doubleValue` (page 17)
Returns the value of the receiver's cell as a double-precision floating-point number.
- `setDoubleValue:` (page 31)
Sets the value of the receiver's cell using a double-precision floating-point number.
- `floatValue` (page 19)
Returns the value of the receiver's cell as a single-precision floating-point number.

- [setFloatValue:](#) (page 32)
Sets the value of the receiver's cell using a single-precision floating-point number.
- [intValue](#) (page 21)
Returns the value of the receiver's cell as an integer.
- [setIntValue:](#) (page 34)
Sets the value of the receiver's cell using an integer.
- [integerValue](#) (page 21)
Returns the value of the receiver's cell as an `NSInteger` value.
- [setIntegerValue:](#) (page 34)
Sets the value of the receiver's cell using an `NSInteger` value.
- [objectValue](#) (page 24)
Returns the value of the receiver's cell as an Objective-C object.
- [setObjectValue:](#) (page 35)
Sets the value of the receiver's cell using an Objective-C object.
- [stringValue](#) (page 39)
Returns the value of the receiver's cell as an `NSString` object.
- [setStringValue:](#) (page 37)
Sets the value of the receiver's cell using an `NSString` object.
- [setNeedsDisplay](#) (page 35)
Marks the receiver as needing redisplay (assuming automatic display is enabled).
- [attributedStringValue](#) (page 15)
Returns the value of the receiver's cell as an attributed string.
- [setAttributedStringValue:](#) (page 29)
Sets the value of the receiver's cell using an attributed string.

Interacting with Other Controls

- [takeDoubleValueFrom:](#) (page 40)
Sets the value of the receiver's cell to a double-precision floating-point value obtained from the specified object.
- [takeFloatValueFrom:](#) (page 41)
Sets the value of the receiver's cell to a single-precision floating-point value obtained from the specified object.
- [takeIntValueFrom:](#) (page 42)
Sets the value of the receiver's cell to an integer value obtained from the specified object.
- [takeIntegerValueFrom:](#) (page 41)
Sets the value of the receiver's cell to an `NSInteger` value obtained from the specified object.
- [takeObjectValueFrom:](#) (page 42)
Sets the value of the receiver's cell to the object value obtained from the specified object.
- [takeStringValueFrom:](#) (page 43)
Sets the value of the receiver's cell to the string value obtained from the specified object.

Formatting Text

- [alignment](#) (page 14)
Returns the alignment mode of the text in the receiver's cell.
- [setAlignment:](#) (page 28)
Sets the alignment of text in the receiver's cell.
- [font](#) (page 19)
Returns the font used to draw text in the receiver's cell.
- [setFont:](#) (page 33)
Sets the font used to draw text in the receiver's cell.
- [formatter](#) (page 20)
Returns the receiver's formatter.
- [setFormatter:](#) (page 33)
Sets the receiver's formatter
- [baseWritingDirection](#) (page 15)
Returns the initial writing direction used to determine the actual writing direction for text.
- [setBaseWritingDirection:](#) (page 29)
Sets the initial writing direction used to determine the actual writing direction for text .
- [setFloatingPointFormat:left:right:](#) (page 49) **Deprecated in Mac OS X v10.0**
Sets the autoranging and floating point number format of the receiver's cell.

Managing the Field Editor

- [abortEditing](#) (page 13)
Terminates the current editing operation and discards any edited text.
- [currentEditor](#) (page 17)
Returns the current field editor for the control.
- [validateEditing](#) (page 44)
Validates changes to any user-typed text.

Resizing the Control

- [calcSize](#) (page 16)
Recomputes any internal sizing information for the receiver, if necessary.
- [sizeToFit](#) (page 39)
Resizes the receiver's frame so that it is the minimum size needed to contain its cell.

Displaying a Cell

- [selectCell:](#) (page 25)
Selects the specified cell and redraws the control as needed.
- [drawCell:](#) (page 18)
Draws the specified cell, as long as it belongs to the receiver.

- [drawCellInside:](#) (page 18)
Draws the inside of the receiver's cell (the area within the bezel or border)
- [updateCell:](#) (page 44)
Marks the specified cell as in need of redrawing.
- [updateCellInside:](#) (page 44)
Marks the inside of the specified cell as in need of redrawing.

Implementing the Target/action Mechanism

- [action](#) (page 14)
Returns the default action-message selector associated with the control.
- [setAction:](#) (page 28)
Sets the receiver's action method to the specified selector.
- [target](#) (page 43)
Returns the target object of the receiver's cell.
- [setTarget:](#) (page 38)
Sets the target object to receive action messages from the receiver's cell.
- [isContinuous](#) (page 22)
Returns a Boolean value indicating whether the receiver's cell sends its action message continuously to its target during mouse tracking.
- [setContinuous:](#) (page 30)
Sets whether the receiver's cell sends its action message continuously to its target during mouse tracking.
- [sendAction:to:](#) (page 26)
Causes the specified action to be sent the target.
- [sendActionOn:](#) (page 27)
Sets the conditions on which the receiver sends action messages to its target.

Getting and Setting Tags

- [tag](#) (page 40)
Returns the tag identifying the receiver (not the tag of the receiver's cell).
- [setTag:](#) (page 37)
Sets the tag of the receiver.

Activating from the Keyboard

- [performClick:](#) (page 24)
Simulates a single mouse click on the receiver.
- [refusesFirstResponder](#) (page 25)
Returns a Boolean value indicating whether the receiver refuses the first responder role.
- [setRefusesFirstResponder:](#) (page 36)
Sets whether the receiver refuses first responder role.

Tracking the Mouse

- [mouseDown:](#) (page 23)
Informs the receiver that the user has pressed the left mouse button.
- [ignoresMultiClick:](#) (page 20)
Returns a Boolean value indicating whether the receiver ignores multiple clicks made in rapid succession.
- [setIgnoresMultiClick:](#) (page 33)
Sets whether the receiver ignores multiple clicks made in rapid succession.

Control Editing Notifications

- [controlTextDidBeginEditing:](#) (page 45) *delegate method*
Sent when a control with editable text begins an editing session.
- [controlTextDidChange:](#) (page 45) *delegate method*
Sent when the text in the receiving control changes.
- [controlTextDidEndEditing:](#) (page 46) *delegate method*
Sent when a control with editable text ends an editing session.

Class Methods

cellClass

Returns the type of cell used by the receiver.

```
+ (Class)cellClass
```

Return Value

The class of the cell used to manage the receiver's contents, or `nil` if no cell class has been set for the receiver or its superclasses (up to `NSControl`).

Availability

Available in Mac OS X v10.0 and later.

See Also

- [cell](#) (page 16)
- [setCell:](#) (page 30)
- + [setCellClass:](#) (page 13)

Related Sample Code

Clock Control
ClockControl
TrackBall

Declared In

NSControl.h

setCellClass:

Sets the type of cell used by the receiver.

```
+ (void)setCellClass:(Class)class
```

Parameters

class

The class of the cell to use with this control.

Discussion

If you have a custom cell subclass that you would like to substitute for the class of a cell object in a nib file, you should set the cell class in the `awakeFromNib` method (NSNibAwaking protocol). You cannot change the class programmatically after the cell object has been unarchived from the nib and instantiated, which occurs immediately after `awakeFromNib` returns. If you are going to be using your custom cell frequently, consider creating your own Interface Builder palette containing the cell.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [cell](#) (page 16)
- [setCell:](#) (page 30)
- + [cellClass](#) (page 12)

Declared In

NSControl.h

Instance Methods

abortEditing

Terminates the current editing operation and discards any edited text.

```
- (BOOL)abortEditing
```

Return Value

YES if there was a field editor associated with the control; otherwise, NO.

Discussion

If there was a field editor, this method removes the field editor's delegate.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [currentEditor](#) (page 17)
- [validateEditing](#) (page 44)

Related Sample Code

QTMetadataEditor

Declared In

NSControl.h

action

Returns the default action-message selector associated with the control.

- (SEL)action

Return Value

The selector associated with the receiver's cell.

Discussion

The `NSControl` implementation of this method returns the action message selector of the receiver's cell. Controls that support multiple cells (such as `NSMatrix` and `NSForm`) must override this method to return the appropriate action-message selector.

If you want the action-message selector for a control that has multiple cells, it is better to use the `getCell` selector directly from the cell's own `action` method, as shown in the following example:

```
SEL someAction = [[theControl selectedCell] action];
```

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setAction:](#) (page 28)
- [setTarget:](#) (page 38)
- [target](#) (page 43)

Related Sample Code

SimpleToolbar

Declared In

NSControl.h

alignment

Returns the alignment mode of the text in the receiver's cell.

- (NSTextAlignment)alignment

Return Value

One of the following constants: `NSLeftTextAlignment`, `NSRightTextAlignment`, `NSCenterTextAlignment`, `NSJustifiedTextAlignment`, or `NSNaturalTextAlignment`. The default value is `NSNaturalTextAlignment`.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setAlignment:](#) (page 28)

Declared In

NSControl.h

attributedStringValue

Returns the value of the receiver's cell as an attributed string.

- (NSAttributedString *)attributedStringValue

Return Value

The value of the cell interpreted as an attributed string, or an empty attributed string if the receiver has no cell.

Discussion

If the control contains many cells (for example, `NSMatrix`), then the value of the currently selected cell is returned. If the control is in the process of editing the affected cell, then it invokes the [validateEditing](#) (page 44) method before extracting and returning the value.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setAttributeStringValue:](#) (page 29)

Declared In

NSControl.h

baseWritingDirection

Returns the initial writing direction used to determine the actual writing direction for text.

- (NSWritingDirection)baseWritingDirection

Return Value

One of the following values: `NSWritingDirectionNatural`, `NSWritingDirectionLeftToRight`, or `NSWritingDirectionRightToLeft`. The default value is `NSWritingDirectionNatural`.

Discussion

The Text system uses this value as a hint for calculating the actual direction for displaying Unicode characters. You should not need to call this method directly.

Availability

Available in Mac OS X v10.4 and later.

See Also

- [setBaseWritingDirection:](#) (page 29)

Declared In

NSControl.h

calcSize

Recomputes any internal sizing information for the receiver, if necessary.

- (void)calcSize

Discussion

This method uses the `calcDrawInfo:` method of its cell to perform the calculations. Most controls maintain a flag that informs them if any of their cells have been modified in such a way that the location or size of the cell should be recomputed. If such a modification happens, this method is automatically invoked before the control is displayed. You should never need to invoke it yourself.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [sizeToFit](#) (page 39)

Declared In

NSControl.h

cell

Returns the receiver's cell object.

- (id)cell

Return Value

The cell object of the receiver.

Discussion

For controls with multiple cells (such as `NSMatrix` or `NSForm`), use the [selectedCell](#) (page 25) method or a similar method to retrieve a specific cell.

Availability

Available in Mac OS X v10.0 and later.

See Also

+ [cellClass](#) (page 12)

+ [setCellClass:](#) (page 13)

- [setCell:](#) (page 30)

Related Sample Code

ButtonMadness

Clock Control

DatePicker

FunHouse

Quartz Composer QCTV

Declared In

NSControl.h

currentEditor

Returns the current field editor for the control.

- (NSText *)currentEditor

Return Value

The field editor for the current control, or `nil` if the receiver does not have a field editor.

Discussion

When the receiver is a control displaying editable text (for example, a text field) and it is the first responder, it has a field editor, which is returned by this method. The field editor is a single `NSTextView` object that is shared among all the controls in a window for light text-editing needs. It is automatically instantiated when needed.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [abortEditing](#) (page 13)
- [validateEditing](#) (page 44)

Declared In

`NSControl.h`

doubleValue

Returns the value of the receiver's cell as a double-precision floating-point number.

- (double)doubleValue

Return Value

The value of the cell interpreted as a double-precision floating-point number.

Discussion

If the control contains many cells (for example, `NSMatrix`), then the value of the currently selected cell is returned. If the control is in the process of editing the affected cell, then it invokes the [validateEditing](#) (page 44) method before extracting and returning the value.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [floatValue](#) (page 19)
- [intValue](#) (page 21)
- [integerValue](#) (page 21)
- [objectValue](#) (page 24)
- [stringValue](#) (page 39)
- [setDoubleValue:](#) (page 31)

Related Sample Code

`CarbonCocoaTempConverter`
`CocoaSpeechSynthesisExample`

FunHouse

Declared In

NSControl.h

drawCell:

Draws the specified cell, as long as it belongs to the receiver.

```
- (void)drawCell:(NSCell *)aCell
```

Parameters

aCell

The cell to draw. If the cell does not belong to the receiver, this method does nothing.

Discussion

This method is provided primarily to support a consistent set of methods between `NSControl` objects with single and multiple cells, because a control with multiple cells needs to be able to draw individual cells.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [selectCell:](#) (page 25)
- [updateCell:](#) (page 44)
- [updateCellInside:](#) (page 44)

Declared In

NSControl.h

drawCellInside:

Draws the inside of the receiver's cell (the area within the bezel or border)

```
- (void)drawCellInside:(NSCell *)aCell
```

Parameters

aCell

The cell to draw. If the cell does not belong to the receiver, this method does nothing.

Discussion

If the receiver is transparent, the method causes the superview to draw itself. This method invokes the `drawInteriorWithFrame:inView:` method of `NSCell`. This method has no effect on controls (such as `NSMatrix` and `NSForm`) that have multiple cells.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [selectCell:](#) (page 25)
- [updateCell:](#) (page 44)
- [updateCellInside:](#) (page 44)

Declared In

NSControl.h

floatValue

Returns the value of the receiver's cell as a single-precision floating-point number.

```
- (float)floatValue
```

Return Value

The value of the cell interpreted as a single-precision floating-point number.

Discussion

If the control contains many cells (for example, `NSMatrix`), then the value of the currently selected cell is returned. If the control is in the process of editing the affected cell, then it invokes the [validateEditing](#) (page 44) method before extracting and returning the value.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [doubleValue](#) (page 17)
- [intValue](#) (page 21)
- [integerValue](#) (page 21)
- [objectValue](#) (page 24)
- [stringValue](#) (page 39)
- [setFloatValue:](#) (page 32)

Related Sample Code

CoreImageGLTextureFBO

Fireworks

GLUT

OpenALExample

WhackedTV

Declared In

NSControl.h

font

Returns the font used to draw text in the receiver's cell.

```
- (NSFont *)font
```

Return Value

The font object used for drawing text.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setFont:](#) (page 33)

Declared In

NSControl.h

formatter

Returns the receiver's formatter.

- (id)formatter

Return Value

The formatter object.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setFormatter:](#) (page 33)

Related Sample Code

TrackBall

Declared In

NSControl.h

ignoresMultiClick

Returns a Boolean value indicating whether the receiver ignores multiple clicks made in rapid succession.

- (BOOL)ignoresMultiClick

Return Value

YES if the receiver ignores multiple clicks; otherwise, NO.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setIgnoresMultiClick:](#) (page 33)

Declared In

NSControl.h

initWithFrame:

Returns an NSControl object initialized with the specified frame rectangle.

- (id)initWithFrame:(NSRect) *frameRect*

Parameters*frameRect*

The rectangle of the control, specified in points in the coordinate space of the enclosing view.

Return Value

An initialized control object, or `nil` if the object could not be initialized.

Discussion

If a cell has been specified for controls of this type, this method also creates an instance of the cell. Because `NSControl` is an abstract class, invocations of this method should appear only in the designated initializers of subclasses; that is, there should always be a more specific designated initializer for the subclass, as this method is the designated initializer for `NSControl`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSControl.h`

integerValue

Returns the value of the receiver's cell as an `NSInteger` value.

- (`NSInteger`)integerValue

Return Value

The value of the cell interpreted as an `NSInteger` value.

Discussion

If the control contains many cells (for example, `NSMatrix`), then the value of the currently selected cell is returned. If the control is in the process of editing the affected cell, then it invokes the [validateEditing](#) (page 44) method before extracting and returning the value.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [doubleValue](#) (page 17)
- [floatValue](#) (page 19)
- [intValue](#) (page 21)
- [objectValue](#) (page 24)
- [stringValue](#) (page 39)
- [setIntegerValue:](#) (page 34)

Declared In

`NSControl.h`

intValue

Returns the value of the receiver's cell as an integer.

- (`int`)intValue

Return Value

The value of the cell interpreted as an integer.

Discussion

If the control contains many cells (for example, `NSMatrix`), then the value of the currently selected cell is returned. If the control is in the process of editing the affected cell, then it invokes the `validateEditing` (page 44) method before extracting and returning the value.

Availability

Available in Mac OS X v10.0 and later.

See Also

- `doubleValue` (page 17)
- `floatValue` (page 19)
- `integerValue` (page 21)
- `objectValue` (page 24)
- `stringValue` (page 39)
- `setIntValue:` (page 34)

Related Sample Code

CocoaSpeechSynthesisExample

EnhancedAudioBurn

EnhancedDataBurn

GLUT

QTKitTimeCode

Declared In

`NSControl.h`

isContinuous

Returns a Boolean value indicating whether the receiver's cell sends its action message continuously to its target during mouse tracking.

- `(BOOL)isContinuous`

Return Value

YES if the action message is sent continuously; otherwise, NO.

Availability

Available in Mac OS X v10.0 and later.

See Also

- `setContinuous:` (page 30)

Related Sample Code

Cropped Image

Declared In

`NSControl.h`

isEnabled

Returns whether the receiver reacts to mouse events.

- (BOOL)isEnabled

Return Value

YES if the receiver responds to mouse events; otherwise, NO.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setEnabled:](#) (page 31)

Related Sample Code

EnhancedDataBurn

TrackBall

Declared In

NSControl.h

mouseDown:

Informs the receiver that the user has pressed the left mouse button.

- (void)mouseDown:(NSEvent *)theEvent

Parameters

theEvent

The event resulting from the user action.

Discussion

Invoked when the mouse button is pressed while the cursor is within the bounds of the receiver, generating *theEvent*. This method highlights the receiver's cell and sends it a `trackMouse:inRect:ofView:untilMouseUp:` message. Whenever the cell finishes tracking the mouse (for example, because the cursor has left the cell's bounds), the cell is unhighlighted. If the mouse button is still down and the cursor reenters the bounds, the cell is again highlighted and a new `trackMouse:inRect:ofView:untilMouseUp:` message is sent. This behavior repeats until the mouse button goes up. If it goes up with the cursor in the control, the state of the control is changed, and the action message is sent to the target. If the mouse button goes up when the cursor is outside the control, no action message is sent.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [ignoresMultiClick](#) (page 20)

- `trackMouse:inRect:ofView:untilMouseUp:` (NSCell)

Declared In

NSControl.h

objectValue

Returns the value of the receiver's cell as an Objective-C object.

- (id)objectValue

Return Value

The value of the cell interpreted as an Objective-C object.

Discussion

If the control contains many cells (for example, `NSMatrix`), then the value of the currently selected cell is returned. If the control is in the process of editing the affected cell, then it invokes the [validateEditing](#) (page 44) method before extracting and returning the value.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [doubleValue](#) (page 17)
- [floatValue](#) (page 19)
- [intValue](#) (page 21)
- [stringValue](#) (page 39)
- [setObjectValue:](#) (page 35)

Related Sample Code

FunHouse

PhotoSearch

PredicateEditorSample

TrackBall

Declared In

`NSControl.h`

performClick:

Simulates a single mouse click on the receiver.

- (void)performClick:(id)sender

Parameters

sender

The object requesting the action. This parameter is ignored.

Discussion

This method calls the `performClick:` method of the receiver's cell with the sender being the control itself. This method raises an exception if the action message cannot be successfully sent.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSControl.h`

refusesFirstResponder

Returns a Boolean value indicating whether the receiver refuses the first responder role.

- (BOOL)refusesFirstResponder

Return Value

YES if the receiver refuses the first responder role; otherwise, NO.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setRefusesFirstResponder:](#) (page 36)

Declared In

NSControl.h

selectCell:

Selects the specified cell and redraws the control as needed.

- (void)selectCell:(NSCell *)aCell

Parameters

aCell

The cell to select. The cell must belong to the receiver.

Discussion

If the cell is already selected (or does not belong to the receiver), this method does nothing. If the cell belongs to the receiver and is not selected, this method changes its state to `NSOnState` and redraws the cell.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [selectedCell](#) (page 25)

Declared In

NSControl.h

selectedCell

Returns the receiver's selected cell.

- (id)selectedCell

Return Value

The selected cell object.

Discussion

The default implementation of this method simply returns the control's associated cell (or `nil` if no cell has been set). Subclasses of `NSControl` that manage multiple cells (such as `NSMatrix` and `NSForm`) must override this method to return the cell selected by the user.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [cell](#) (page 16)
- [setCell:](#) (page 30)

Related Sample Code

Cropped Image
PhotoSearch

Declared In

NSControl.h

selectedTag

Returns the tag of the receiver's selected cell.

- (NSInteger)selectedTag

Return Value

The tag of the selected cell, or -1 if no cell is selected.

Discussion

When you set the tag of a control with a single cell in Interface Builder, it sets the tags of both the control and the cell with the same value as a convenience.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setTag:](#) (page 37)
- [tag](#) (page 40)

Related Sample Code

CalendarItems
WhackedTV

Declared In

NSControl.h

sendAction:to:

Causes the specified action to be sent the target.

- (BOOL)sendAction:(SEL)theAction to:(id)theTarget

Parameters

theAction

The selector to invoke on the target. If the selector is `NULL`, no message is sent.

theTarget

The target object to receive the message. If the object is `nil`, the application searches the responder chain for an object capable of handling the message. For more information on dispatching actions, see the class description for `NSActionCell`.

Return Value

YES if the message was successfully sent; otherwise, NO.

Discussion

This method uses the `sendAction:to:from:` method of `NSApplication` to invoke the specified method on an object. The receiver is passed as the parameter to the action message. This method is invoked primarily by the `trackMouse:inRect:ofView:untilMouseUp:` method of `NSCell`.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [action](#) (page 14)
- [target](#) (page 43)

Declared In

`NSControl.h`

sendActionOn:

Sets the conditions on which the receiver sends action messages to its target.

```
- (NSInteger)sendActionOn:(NSInteger)mask
```

Parameters

mask

A bit mask containing the conditions for sending the action. The only conditions that are actually checked are associated with the `NSLeftMouseDownMask`, `NSLeftMouseUpMask`, `NSLeftMouseDraggedMask`, and `NSPeriodicMask` bits.

Return Value

A bit mask containing the previous settings. This bit mask uses the same values as specified in the *mask* parameter.

Discussion

You use this method during mouse tracking when the mouse button changes state, the mouse moves, or if the cell is marked to send its action continuously while tracking. Because of this, the only bits checked in *mask* are `NSLeftMouseDownMask`, `NSLeftMouseUpMask`, `NSLeftMouseDraggedMask`, and `NSPeriodicMask`, which are declared in the `NSEvent` class reference.

The default implementation of this method simply invokes the `sendActionOn:` method of its associated cell.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [sendAction:to:](#) (page 26)
- `sendActionOn:(NSCell)`

Declared In

NSControl.h

setAction:

Sets the receiver's action method to the specified selector.

- (void)setAction:(SEL)aSelector

Parameters*aSelector*

The new action-message selector to associate with the receiver's cell. Specify `NULL` to prevent action messages from being sent to the receiver's target.

Discussion

See *Action Messages* for additional information on action messages.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [action](#) (page 14)
- [setTarget:](#) (page 38)
- [target](#) (page 43)

Related Sample Code

FunHouse

PrefsPane

QTAudioContextInsert

QTAudioExtractionPanel

Quartz Composer QCTV

Declared In

NSControl.h

setAlignment:

Sets the alignment of text in the receiver's cell.

- (void)setAlignment:(NSTextAlignment)mode

Parameters*mode*

One of the following constants: `NSLeftTextAlignment`, `NSRightTextAlignment`, `NSCenterTextAlignment`, `NSJustifiedTextAlignment`, or `NSNaturalTextAlignment`.

Discussion

If the cell is currently being edited, this method aborts the edits to change the alignment.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [alignment](#) (page 14)

Related Sample Code

Quartz Composer QCTV

Declared In

NSControl.h

setAttributedStringValue:

Sets the value of the receiver's cell using an attributed string.

```
- (void)setAttributedStringValue:(NSAttributedString *)object
```

Parameters

object

The value of the cell interpreted as an attributed string.

Discussion

If the cell is being edited, this method aborts all editing before setting the value. If the cell does not inherit from `NSActionCell`, the method marks the cell's interior as needing to be redisplayed; `NSActionCell` performs its own updating of cells.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [attributedStringValue](#) (page 15)

Declared In

NSControl.h

setBaseWritingDirection:

Sets the initial writing direction used to determine the actual writing direction for text .

```
- (void)setBaseWritingDirection:(NSWritingDirection)writingDirection
```

Parameters

writingDirection

One of the following values: `NSWritingDirectionNatural`, `NSWritingDirectionLeftToRight`, or `NSWritingDirectionRightToLeft`.

Discussion

If you know the base writing direction of the text you are rendering, you can use this method to specify that direction to the text system.

Availability

Available in Mac OS X v10.4 and later.

See Also

- [baseWritingDirection](#) (page 15)

Declared In

NSControl.h

setCell:

Sets the receiver's cell

```
- (void)setCell:(NSCell *)aCell
```

Parameters*aCell*

The new cell for the receiver.

Discussion

Use this method with great care as it can irrevocably damage the affected control; specifically, you should only use this method in initializers for subclasses of `NSControl`.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [cell](#) (page 16)
- [selectedCell](#) (page 25)

Declared In

NSControl.h

setContinuous:

Sets whether the receiver's cell sends its action message continuously to its target during mouse tracking.

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

Parameters*flag*

YES if the action message should be sent continuously; otherwise, NO.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [isContinuous](#) (page 22)

Related Sample Code

Cropped Image

SurfaceVertexProgram

Declared In

NSControl.h

setDoubleValue:

Sets the value of the receiver's cell using a double-precision floating-point number.

```
- (void)setDoubleValue:(double)aDouble
```

Parameters

aDouble

The value of the cell interpreted as a double-precision floating-point number.

Discussion

If the cell is being edited, this method aborts all editing before setting the value. If the cell does not inherit from `NSActionCell`, the method marks the cell's interior as needing to be redisplayed; `NSActionCell` performs its own updating of cells.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [doubleValue](#) (page 17)
- [setFloatValue:](#) (page 32)
- [setIntValue:](#) (page 34)
- [setIntegerValue:](#) (page 34)
- [setObjectValue:](#) (page 35)
- [setStringValue:](#) (page 37)

Related Sample Code

CocoaSpeechSynthesisExample

FunHouse

QTKitMovieShuffler

QTMetadataEditor

Quartz 2D Shadings

Declared In

`NSControl.h`

setEnabled:

Sets whether the receiver (and its cell) reacts to mouse events.

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

Parameters

flag

YES if you want the receiver to react to mouse events; otherwise, NO.

Discussion

If *flag* is NO, any editing is aborted. This method redraws the entire control if it is marked as needing redisplay. Subclasses may want to override this method to redraw only a portion of the control when the enabled state changes; `NSButton` and `NSSlider` do this.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [isEnabled](#) (page 23)

Related Sample Code

CocoaSpeechSynthesisExample

ColorMatching

GLUT

OpenALExample

WhackedTV

Declared In

NSControl.h

setFloatValue:

Sets the value of the receiver's cell using a single-precision floating-point number.

```
- (void)setFloatValue:(float)aFloat
```

Parameters

aFloat

The value of the cell interpreted as a single-precision floating-point number.

Discussion

If the cell is being edited, this method aborts all editing before setting the value. If the cell does not inherit from `NSActionCell`, the method marks the cell's interior as needing to be redisplayed; `NSActionCell` performs its own updating of cells.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [floatValue](#) (page 19)

- [setDoubleValue:](#) (page 31)

- [setIntValue:](#) (page 34)

- [setIntegerValue:](#) (page 34)

- [setObjectValue:](#) (page 35)

- [setStringValue:](#) (page 37)

Related Sample Code

FilterDemo

From A View to A Movie

From A View to A Picture

NineSlice

OpenALExample

Declared In

NSControl.h

setFont:

Sets the font used to draw text in the receiver's cell.

```
- (void)setFont:(NSFont *)fontObject
```

Parameters

fontObject

The font object to use.

Discussion

If the cell is being edited, the text in the cell is redrawn in the new font, and the cell's editor (the NSText object used globally for editing) is updated with the new font object.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setFont:](#) (page 33)

Related Sample Code

ObjectPath

Declared In

NSControl.h

setFormatter:

Sets the receiver's formatter

```
- (void)setFormatter:(NSFormatter *)newFormatter
```

Parameters

newFormatter

The new formatter object to use with the control.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [formatter](#) (page 20)

Declared In

NSControl.h

setIgnoresMultiClick:

Sets whether the receiver ignores multiple clicks made in rapid succession.

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

Parameters

flag

YES if the receiver should ignore multiple clicks; otherwise, NO.

Discussion

By default, controls treat double clicks as two distinct clicks, triple clicks as three distinct clicks, and so on. However, if you pass YES to this method, additional clicks (within a predetermined interval after the first) occurring after the first click are not processed by the receiver, and are instead passed on to `super`.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [ignoresMultiClick](#) (page 20)

Declared In

NSControl.h

setIntegerValue:

Sets the value of the receiver's cell using an `NSInteger` value.

```
- (void)setIntegerValue:(NSInteger)anInteger
```

Parameters

anInteger

The value of the cell interpreted as an `NSInteger` value.

Discussion

If the cell is being edited, this method aborts all editing before setting the value. If the cell does not inherit from `NSActionCell`, the method marks the cell's interior as needing to be redisplayed; `NSActionCell` performs its own updating of cells.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [integerValue](#) (page 21)
- [setDoubleValue:](#) (page 31)
- [setFloatValue:](#) (page 32)
- [setIntValue:](#) (page 34)
- [setObjectValue:](#) (page 35)
- [setStringValue:](#) (page 37)

Declared In

NSControl.h

setIntValue:

Sets the value of the receiver's cell using an integer.

```
- (void)setIntValue:(int)anInt
```

Parameters

anInt

The value of the cell interpreted as an integer.

Discussion

If the cell is being edited, this method aborts all editing before setting the value. If the cell does not inherit from `NSActionCell`, the method marks the cell's interior as needing to be redisplayed; `NSActionCell` performs its own updating of cells.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [intValue](#) (page 21)
- [setDoubleValue:](#) (page 31)
- [setIntegerValue:](#) (page 34)
- [setFloatValue:](#) (page 32)
- [setObjectValue:](#) (page 35)
- [setStringValue:](#) (page 37)

Related Sample Code

[AESendThreadSafe](#)

[Fireworks](#)

[From A View to A Movie](#)

[From A View to A Picture](#)

[ObjectPath](#)

Declared In

`NSControl.h`

setNeedsDisplay

Marks the receiver as needing redisplay (assuming automatic display is enabled).

```
- (void)setNeedsDisplay
```

Discussion

This method also recalculates the dimensions of the control as needed.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setNeedsDisplay:](#) (NSView)

Declared In

`NSControl.h`

setObjectValue:

Sets the value of the receiver's cell using an Objective-C object.

```
- (void)setObjectValue:(id < NSCopying >)object
```

Parameters*object*

The value of the cell interpreted as an Objective-C object.

Discussion

If the cell is being edited, this method aborts all editing before setting the value. If the cell does not inherit from `NSActionCell`, the method marks the cell's interior as needing to be redisplayed; `NSActionCell` performs its own updating of cells.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [objectValue](#) (page 24)
- [setDoubleValue:](#) (page 31)
- [setFloatValue:](#) (page 32)
- [setIntValue:](#) (page 34)
- [setStringValue:](#) (page 37)

Related Sample Code

EnhancedAudioBurn

EnhancedDataBurn

NewsReader

Quartz Composer QCTV

Declared In

`NSControl.h`

setRefusesFirstResponder:

Sets whether the receiver refuses first responder role.

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

Parameters*flag*

YES if the receiver should refuse the first responder role; otherwise, NO.

Discussion

By default, the user can advance the focus of keyboard events between controls by pressing the Tab key; when this focus—or first responder status—is indicated for a control (by the insertion point or, for nontext controls, a faint rectangle), the user can activate the control by pressing the Space bar.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [refusesFirstResponder](#) (page 25)
- [objectValue](#) (page 24)
- [setDoubleValue:](#) (page 31)
- [setFloatValue:](#) (page 32)

Declared In

NSControl.h

setStringValue:

Sets the value of the receiver's cell using an `NSString` object.

```
- (void)setStringValue:(NSString *)aString
```

Parameters*aString*

The value of the cell interpreted as an `NSString` object.

Discussion

If the cell is being edited, this method aborts all editing before setting the value. If the cell does not inherit from `NSActionCell`, the method marks the cell's interior as needing to be redisplayed; `NSActionCell` performs its own updating of cells.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setDoubleValue:](#) (page 31)
- [setFloatValue:](#) (page 32)
- [setIntValue:](#) (page 34)
- [setObjectValue:](#) (page 35)
- [stringValue](#) (page 39)

Related Sample Code

EnhancedAudioBurn

FunHouse

GLUT

OpenALExample

PDFKitLinker2

Declared In

NSControl.h

setTag:

Sets the tag of the receiver.

```
- (void)setTag:(NSInteger)anInt
```

Parameters*anInt*

The new tag.

Discussion

This method does not affect the tag of the receiver's cell. Tags allow you to identify particular cells. Tag values are not used internally; they are only changed by external invocations of [setTag:](#) (page 37). You typically set tag values in Interface Builder and use them at runtime in your application. When you set the tag of a control with a single cell in Interface Builder, it sets the tags of both the control and the cell to the same value as a convenience.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [tag](#) (page 40)
- [selectedTag](#) (page 26)

Related Sample Code

FunHouse

Quartz Composer QCTV

Declared In

NSControl.h

setTarget:

Sets the target object to receive action messages from the receiver's cell.

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

Parameters

anObject

The new target object to associate with the receiver's cell, or `nil` to remove the current target.

Discussion

If *anObject* is `nil` but the control still has a valid action message assigned, the application follows the responder chain looking for an object that can respond to the message. See the description of the `NSActionCell` class for details.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [action](#) (page 14)
- [setAction:](#) (page 28)
- [target](#) (page 43)
- [setTarget:](#) (NSCell)

Related Sample Code

AnimatedTableView

FunHouse

PrefsPane

Quartz Composer QCTV

SimpleCocoaApp

Declared In

NSControl.h

sizeToFit

Resizes the receiver's frame so that it is the minimum size needed to contain its cell.

```
- (void)sizeToFit
```

Discussion

If you want a multiple-cell custom subclass of `NSControl` to size itself to fit its cells, you must override this method. This method neither redisplay the receiver nor marks it as needing display. You must do this yourself with either `redisplay` or `setNeedsDisplay` (page 35) method.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [calcSize](#) (page 16)

Related Sample Code

Quartz Composer QCTV

Declared In

NSControl.h

stringValue

Returns the value of the receiver's cell as an `NSString` object.

```
- (NSString *)stringValue
```

Return Value

The value of the cell interpreted as an `NSString` object.

Discussion

If the control contains many cells (for example, `NSMatrix`), then the value of the currently selected cell is returned. If the control is in the process of editing the affected cell, then it invokes the `validateEditing` (page 44) method before extracting and returning the value.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [doubleValue](#) (page 17)

- [floatValue](#) (page 19)

- [intValue](#) (page 21)

- [objectValue](#) (page 24)

- [setStringValue:](#) (page 37)

Related Sample Code

CoreRecipes

EnhancedAudioBurn
 FinalCutPro_AppleEvents
 MovieAssembler
 WhackedTV

Declared In
 NSControl.h

tag

Returns the tag identifying the receiver (not the tag of the receiver's cell).

- (NSInteger)tag

Return Value

The tag of this control object.

Discussion

Tags allow you to identify particular controls. Tag values are not used internally; they are only changed by external invocations of `setTag:`. You typically set tag values in Interface Builder and use them at runtime in your application. When you set the tag of a control with a single cell in Interface Builder, it sets the tags of both the control and the cell to the same value as a convenience.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setTag:](#) (page 37)
- [selectedTag](#) (page 26)

Related Sample Code

Cropped Image
 EnhancedDataBurn
 OpenALExample
 Quartz Composer QCTV
 Quartz Composer WWDC 2005 TextEdit

Declared In
 NSControl.h

takeDoubleValueFrom:

Sets the value of the receiver's cell to a double-precision floating-point value obtained from the specified object.

- (void)takeDoubleValueFrom:(id)sender

Parameters

sender

The object from which to take the value. This object must respond to the [doubleValue](#) (page 17) message.

Discussion

You can use this method to link action messages between controls. It permits one control or cell (*sender*) to affect the value of another control (the receiver) by invoking this method in an action message to the receiver. For example, a text field can be made the target of a slider. Whenever the slider is moved, it sends this message to the text field. The text field then obtains the slider's value, turns it into a text string, and displays it.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSControl.h

takeFloatValueFrom:

Sets the value of the receiver's cell to a single-precision floating-point value obtained from the specified object.

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

Parameters

sender

The object from which to take the value. This object must respond to the [floatValue](#) (page 19) message.

Discussion

You can use this method to link action messages between controls. It permits one control or cell (*sender*) to affect the value of another control (the receiver) by invoking this method in an action message to the receiver. For example, a text field can be made the target of a slider. Whenever the slider is moved, it sends this message to the text field. The text field then obtains the slider's value, turns it into a text string, and displays it.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSControl.h

takeIntegerValueFrom:

Sets the value of the receiver's cell to an `NSInteger` value obtained from the specified object.

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

Parameters

sender

The object from which to take the value. This object must respond to the [integerValue](#) (page 21) message.

Discussion

You can use this method to link action messages between controls. It permits one control or cell (*sender*) to affect the value of another control (the receiver) by invoking this method in an action message to the receiver. For example, a text field can be made the target of a slider. Whenever the slider is moved, it sends this message to the text field. The text field then obtains the slider's value, turns it into a text string, and displays it.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSControl.h

takeIntValueFrom:

Sets the value of the receiver's cell to an integer value obtained from the specified object.

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

Parameters

sender

The object from which to take the value. This object must respond to the [intValue](#) (page 21) message.

Discussion

You can use this method to link action messages between controls. It permits one control or cell (*sender*) to affect the value of another control (the receiver) by invoking this method in an action message to the receiver. For example, a text field can be made the target of a slider. Whenever the slider is moved, it sends this message to the text field. The text field then obtains the slider's value, turns it into a text string, and displays it.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSControl.h

takeObjectValueFrom:

Sets the value of the receiver's cell to the object value obtained from the specified object.

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

Parameters

sender

The object from which to take the value. This object must respond to the [objectValue](#) (page 24) message.

Discussion

You can use this method to link action messages between controls. It permits one control or cell (*sender*) to affect the value of another control (the receiver) by invoking this method in an action message to the receiver. For example, a text field can be made the target of a slider. Whenever the slider is moved, it sends this message to the text field. The text field then obtains the slider's value, turns it into a text string, and displays it.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSControl.h

takeStringValueFrom:

Sets the value of the receiver's cell to the string value obtained from the specified object.

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

Parameters

sender

The object from which to take the value. This object must respond to the [stringValue](#) (page 39) message.

Discussion

You can use this method to link action messages between controls. It permits one control or cell (*sender*) to affect the value of another control (the receiver) by invoking this method in an action message to the receiver. For example, a text field can be made the target of a slider. Whenever the slider is moved, it sends this message to the text field. The text field then obtains the slider's value, turns it into a text string, and displays it.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSControl.h

target

Returns the target object of the receiver's cell.

```
- (id)target
```

Return Value

The target object that receives action messages from the cell.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [action](#) (page 14)
- [setAction:](#) (page 28)
- [setTarget:](#) (page 38)

Related Sample Code

RadiantColorPicker
SimpleToolbar
UIElementInspector

Declared In

NSControl.h

updateCell:

Marks the specified cell as in need of redrawing.

```
- (void)updateCell:(NSCell *)aCell
```

Parameters

aCell

The cell to redraw.

Discussion

If the cell currently has the focus, this method updates the cell's focus ring; otherwise, the entire cell is marked as needing redisplay. The cell is redrawn during the next update cycle.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSControl.h

updateCellInside:

Marks the inside of the specified cell as in need of redrawing.

```
- (void)updateCellInside:(NSCell *)aCell
```

Parameters

aCell

The cell to redraw.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [updateCell:](#) (page 44)

Declared In

NSControl.h

validateEditing

Validates changes to any user-typed text.

```
- (void)validateEditing
```

Discussion

Validation sets the object value of the cell to the current contents of the cell's editor (the `NSText` object used for editing), storing it as a simple `NSString` or an attributed string object based on the attributes of the editor.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [abortEditing](#) (page 13)
- [currentEditor](#) (page 17)

Declared In

`NSControl.h`

Delegate Methods

controlTextDidBeginEditing:

Sent when a control with editable text begins an editing session.

```
- (void)controlTextDidBeginEditing:(NSNotification *)aNotification
```

Parameters

aNotification

The notification object. The name of the notification is always [NSControlTextDidBeginEditingNotification](#) (page 47).

Discussion

This method is invoked when the user begins editing text in a control such as a text field or a form field. The control posts a [NSControlTextDidBeginEditingNotification](#) (page 47) notification, and if the control's delegate implements this method, it is automatically registered to receive the notification. Use the key `@\"NSFieldEditor\"` to obtain the field editor from the `userInfo` dictionary of the notification object.

See [controlTextDidEndEditing:](#) (page 46) for an explanation of why you may not always get one invocation of [controlTextDidBeginEditing:](#) (page 45) for each invocation of [controlTextDidEndEditing:](#) (page 46).

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSControl.h`

controlTextDidChange:

Sent when the text in the receiving control changes.

```
- (void)controlTextDidChange:(NSNotification *)aNotification
```

Parameters*aNotification*

The notification object. The name of the notification is always [NSControlTextDidChangeNotification](#) (page 47).

Discussion

This method is invoked when text in a control such as a text field or form changes. The control posts a [NSControlTextDidChangeNotification](#) (page 47) notification, and if the control's delegate implements this method, it is automatically registered to receive the notification. Use the key @"NSFieldEditor" to obtain the field editor from the `userInfo` dictionary of the notification object.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSControl.h

controlTextDidEndEditing:

Sent when a control with editable text ends an editing session.

```
- (void)controlTextDidEndEditing:(NSNotification *)aNotification
```

Parameters*aNotification*

The notification object. The name of the notification is always [NSControlTextDidEndEditingNotification](#) (page 47).

Discussion

This method is invoked when the user stops editing text in a control such as a text field or form. The control posts a [NSControlTextDidEndEditingNotification](#) (page 47) notification, and if the control's delegate implements this method, it is automatically registered to receive the notification. Use the key @"NSFieldEditor" to obtain the field editor from the `userInfo` dictionary of the notification object.



Warning: In some cases, such as when editing within an instance of `NSOutlineView`, this method may be invoked without a previous invocation of [controlTextDidBeginEditing:](#) (page 45). You will only get the `controlTextDidBeginEditing:` notification if the user actually types something, but you can get the `controlTextDidEndEditing:` notification if the user just double-clicks the field and then clicks outside the field, without typing.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSControl.h

Notifications

An `NSControl` object posts the following notifications to interested observers and its delegate. Note that although the `NSControl` class defines delegate methods, it does not itself have a delegate. Any subclass that uses these methods must have a delegate and the methods to get and set it.

NSControlTextDidBeginEditingNotification

Sent when a control with editable cells begins an edit session.

The field editor of the edited cell originally sends an `NSNotification` to the control, which passes it on in this form to its delegate. The notification object is the `NSControl` object posting the notification. The `userInfo` dictionary contains the following information:

Key	Value
@ <code>NSFieldEditor</code>	The edited cell's field editor

See the [controlTextDidEndEditing:](#) (page 46) method for details.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSControl.h`

NSControlTextDidChangeNotification

Sent when the text in the receiving control changes.

The field editor of the edited cell originally sends an `NSNotification` to the control, which passes it on in this form to its delegate. The notification object is the `NSControl` object posting the notification. The `userInfo` dictionary contains the following information:

Key	Value
@ <code>NSFieldEditor</code>	The edited cell's field editor

See the [controlTextDidChange:](#) (page 45) method for details.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSControl.h`

NSControlTextDidEndEditingNotification

Sent when a control with editable cells ends an editing session.

The field editor of the edited cell originally sends an [NSControlTextDidEndEditingNotification](#) (page 47) to the control, which passes it on in this form to its delegate. The notification object is the `NSControl` object posting the notification. The `userInfo` dictionary contains the following information:

Key	Value
@"NSFieldEditor"	The edited cell's field editor

See the [controlTextDidEndEditing:](#) (page 46) method for details.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSControl.h`

Deprecated NSControl Methods

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

Deprecated in Mac OS X v10.0

setFloatingPointFormat:left:right:

Sets the autoranging and floating point number format of the receiver's cell. (Deprecated in Mac OS X v10.0.)

```
- (void)setFloatingPointFormat:(BOOL)autoRange left:(NSUInteger)leftDigits
    right:(NSUInteger)rightDigits
```

Parameters

autoRange

YES to enable autoranging; otherwise, NO.

leftDigits

The number of digits to display to the left of the decimal point.

rightDigits

The number of digits to display to the right of the decimal point.

Discussion

For more information about autoranging and how it works, see the description of this method in the `NSCell` class specification. If the cell is being edited, the current edits are discarded and the cell's interior is redrawn.

Note: This method is being deprecated in favor of a new class of formatter objects. For more information, see `NSFormatter`. This documentation is provided only for developers who need to modify older applications.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.0.

See Also

- `setFloatingPointFormat:left:right:` (`NSCell`)

Declared In

`NSControl.h`

Document Revision History

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

Date	Notes
2009-06-18	Moved selected delegate methods back to class.
2009-04-15	Updated for Mac OS X v10.6. Delegate methods moved to NSControlTextEditingDelegate Protocol Reference .
2008-10-15	Revised wording of several methods and added descriptions of NSInteger-related "value" methods.
2007-07-13	Added warning about behavior of <code>controlTextDidBeginEditing:</code> and <code>controlTextDidEndEditing:</code> .
2006-05-23	First publication of this content as a separate document.

REVISION HISTORY

Document Revision History

Index

A

abortEditing [instance method 13](#)
action [instance method 14](#)
alignment [instance method 14](#)
attributedStringValue [instance method 15](#)

B

baseWritingDirection [instance method 15](#)

C

calcSize [instance method 16](#)
cell [instance method 16](#)
cellClass [class method 12](#)
controlTextDidBeginEditing: <NSObject> delegate [method 45](#)
controlTextDidChange: <NSObject> delegate [method 45](#)
controlTextDidEndEditing: <NSObject> delegate [method 46](#)
currentEditor [instance method 17](#)

D

doubleValue [instance method 17](#)
drawCell: [instance method 18](#)
drawCellInside: [instance method 18](#)

F

floatValue [instance method 19](#)
font [instance method 19](#)
formatter [instance method 20](#)

I

ignoresMultiClick [instance method 20](#)
initWithFrame: [instance method 20](#)
integerValue [instance method 21](#)
intValue [instance method 21](#)
isContinuous [instance method 22](#)
isEnabled [instance method 23](#)

M

mouseDown: [instance method 23](#)

N

NSControlTextDidBeginEditingNotification [notification 47](#)
NSControlTextDidChangeNotification [notification 47](#)
NSControlTextDidEndEditingNotification [notification 47](#)

O

objectValue [instance method 24](#)

P

performClick: [instance method 24](#)

R

refusesFirstResponder [instance method 25](#)

S

selectCell: instance method [25](#)
 selectedCell instance method [25](#)
 selectedTag instance method [26](#)
 sendAction:to: instance method [26](#)
 sendActionOn: instance method [27](#)
 setAction: instance method [28](#)
 setAlignment: instance method [28](#)
 setAttributedStringValue: instance method [29](#)
 setBaseWritingDirection: instance method [29](#)
 setCellClass: class method [13](#)
 setCell: instance method [30](#)
 setContinuous: instance method [30](#)
 setDoubleValue: instance method [31](#)
 setEnabled: instance method [31](#)
 setFloatingPointFormat:left:right: instance method [49](#)
 setFloatValue: instance method [32](#)
 setFont: instance method [33](#)
 setFormatter: instance method [33](#)
 setIgnoresMultiClick: instance method [33](#)
 setIntegerValue: instance method [34](#)
 setIntValue: instance method [34](#)
 setNeedsDisplay instance method [35](#)
 setObjectValue: instance method [35](#)
 setRefusesFirstResponder: instance method [36](#)
 setStringValue: instance method [37](#)
 setTag: instance method [37](#)
 setTarget: instance method [38](#)
 sizeToFit instance method [39](#)
 stringValue instance method [39](#)

T

tag instance method [40](#)
 takeDoubleValueFrom: instance method [40](#)
 takeFloatValueFrom: instance method [41](#)
 takeIntegerValueFrom: instance method [41](#)
 takeIntValueFrom: instance method [42](#)
 takeObjectValueFrom: instance method [42](#)
 takeStringValueFrom: instance method [43](#)
 target instance method [43](#)

U

updateCell: instance method [44](#)
 updateCellInside: instance method [44](#)

V

validateEditing instance method [44](#)