NSMenuItem Class Reference

Cocoa > User Experience

2008-02-08

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

Contents

NSMenuItem Class Reference

Inherits from	NSObject
Conforms to	NSValidatedUserInterfaceItem NSCoding NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/AppKit.framework
Availability	Available in Mac OS X v10.0 and later.
Companion guide	Application Menu and Pop-up List Programming Topics for Cocoa
Declared in	NSMenuItem.h
Related sample code	PDFKitLinker2 QTQuartzPlayer Quartz Composer WWDC 2005 TextEdit SearchField TextEditPlus

Overview

The `NSMenuItem` class defines objects that are used as command items in menus. Additionally, the `NSMenuItem` class also includes some private functionality needed to maintain binary compatibility with other components of Cocoa. Because of this fact, you cannot replace the `NSMenuItem` class with a different class. You may, however, subclass `NSMenuItem` if necessary.

Prior to Mac OS X v10.5, `NSMenuItem` conformed to the following protocols: `NSCopying` (see *NSCopying Protocol Reference*), `NSCoding` (see *NSCoding Protocol Reference*), and `NSValidatedUserInterfaceItem` (see *NSValidatedUserInterfaceItem Protocol Reference*).

Tasks

Creating a Menu Item

`- initWithTitle:action:keyEquivalent:` (page 13)
Returns an initialized instance of an `NSMenuItem`.

Enabling a Menu Item

- `setEnabled:` (page 21)
  Sets whether the receiver is enabled
- `isEnabled` (page 14)
  Returns a Boolean value that indicates whether the receiver is enabled.

Managing Hidden Status

- `setHidden:` (page 22)
  Sets whether the receiver is hidden.
- `isHidden` (page 14)
  Returns a Boolean value that indicates whether the receiver is hidden.
- `isHiddenOrHasHiddenAncestor` (page 15)
  Returns a Boolean value that indicates whether the receiver or any of its superitems is hidden.

Managing the Target and Action

- `setTarget:` (page 29)
  Sets the receiver's target.
- `target` (page 33)
  Returns the receiver's target.
- `setAction:` (page 19)
  Sets the receiver's action-method selector.
- `action` (page 11)
  Returns the receiver's action-method selector.

Managing the Title

- `setTitle:` (page 30)
  Sets the receiver's title.
- `title` (page 33)
  Returns the receiver's title.
- `setAttributedTitle:` (page 20)
  Specifies a custom string for a menu item.
- `attributedTitle` (page 11)
  Returns the custom title string for a menu item.

Managing the Tag

- `setTag:` (page 29)
  Sets the receiver's tag.

- `tag` (page 32)
  Returns the receiver's tag.

Managing the State

- `setState:` (page 28)
  Sets the state of the receiver.
- `state` (page 32)
  Returns the state of the receiver.

Managing the Image

- `setImage:` (page 22)
  Sets the receiver's image.
- `image` (page 12)
  Returns the image displayed by the receiver.
- `setOnStateImage:` (page 26)
  Sets the image of the receiver that indicates an "on" state.
- `onStateImage` (page 18)
  Returns the image used to depict the receiver's "on" state.
- `setOffStateImage:` (page 26)
  Sets the image of the receiver that indicates an "off" state.
- `offStateImage` (page 18)
  Returns the image used to depict the receiver's "off" state.
- `setMixedStateImage:` (page 25)
  Sets the image of the receiver that indicates a "mixed" state, that is, a state neither "on" nor "off."
- `mixedStateImage` (page 17)
  Returns the image used to depict a "mixed state."

Managing Submenus

- `setSubmenu:` (page 28)
  Sets the submenu of the receiver.
- `submenu` (page 32)
  Returns the submenu associated with the receiving menu item.
- `hasSubmenu` (page 12)
  Returns a Boolean value that indicates whether the receiver has a submenu.

Getting a Separator Item

+ `separatorItem` (page 10)
  Returns a menu item that is used to separate logical groups of menu commands.

- `isSeparatorItem` (page 15)
  Returns a Boolean value that indicates whether the receiver is a separator item.

Managing the Owning Menu

- `setMenu:` (page 24)
  Sets the receiver's menu.
- `menu` (page 16)
  Returns the menu to which the receiver belongs.

Managing Key Equivalents

- `setKeyEquivalent:` (page 23)
  Sets the receiver's unmodified key equivalent.
- `keyEquivalent` (page 16)
  Returns the receiver's unmodified keyboard equivalent.
- `setKeyEquivalentModifierMask:` (page 24)
  Sets the receiver's keyboard equivalent modifiers.
- `keyEquivalentModifierMask` (page 16)
  Returns the receiver's keyboard equivalent modifier mask.

Managing Mnemonics

- `setMnemonicLocation:` (page 25)
  Deprecated. Sets the character of the menu item title at location that is to be underlined.
- `mnemonicLocation` (page 17)
  Deprecated. Returns the position of the underlined character in the menu item title used as a mnemonic.
- `setTitleWithMnemonic:` (page 30)
  Deprecated. Sets the title of a menu item with a character denoting an access key.
- `mnemonic` (page 17)
  Deprecated. Returns the character in the menu item title that appears underlined for use as a mnemonic.

Managing User Key Equivalents

+ `setUsesUserKeyEquivalents:` (page 10)
  Sets whether menu items conform to user preferences for key equivalents.
+ `usesUserKeyEquivalents` (page 11)
  Returns a Boolean value that indicates whether menu items conform to user preferences for key equivalents.
- `userKeyEquivalent` (page 34)
  Returns the user-assigned key equivalent for the receiver.

Managing Alternates

- `setAlternate:` (page 19)
  Marks the receiver as an alternate to the previous menu item.
- `isAlternate` (page 14)
  Returns a Boolean value that indicates whether the receiver is an alternate to the previous menu item.

Managing Indentation Levels

- `setIndentationLevel:` (page 23)
  Sets the menu item indentation level for the receiver.
- `indentationLevel` (page 13)
  Returns the menu item indentation level for the receiver.

Managing Tool Tips

- `setToolTip:` (page 31)
  Sets a help tag for a menu item.
- `toolTip` (page 34)
  Returns the help tag for a menu item.

Representing an Object

- `setRepresentedObject:` (page 27)
  Sets the object represented by the receiver.
- `representedObject` (page 19)
  Returns the object that the receiving menu item represents.

Managing the View

- `setView:` (page 31)
  Sets the content view for the receiver.
- `view` (page 34)
  Returns the view for the receiver.

Getting Highlighted Status

- `isHighlighted` (page 15)
  Returns a Boolean value that indicates whether the receiver should be drawn highlighted.

Class Methods

separatorItem

Returns a menu item that is used to separate logical groups of menu commands.

```
+ (NSMenuItem *)separatorItem
```

Return Value
A menu item that is used to separate logical groups of menu commands.

Discussion
This menu item is disabled. The default separator item is blank space.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `isSeparatorItem` (page 15)
- `setEnabled:` (page 21)

Related Sample Code
Clock Control
QTAudioExtractionPanel
Quartz Composer WWDC 2005 TextEdit
SearchField
TextEditPlus

Declared In
`NSMenuItem.h`

setUsesUserKeyEquivalents:

Sets whether menu items conform to user preferences for key equivalents.

```
+ (void)setUsesUserKeyEquivalents:(BOOL)flag
```

Parameters

flag

If `YES`, menu items conform to user preferences for key equivalents; otherwise, the key equivalents originally assigned to the menu items are used.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `usesUserKeyEquivalents` (page 11)
- `userKeyEquivalent` (page 34)

Declared In
`NSMenuItem.h`

usesUserKeyEquivalents

Returns a Boolean value that indicates whether menu items conform to user preferences for key equivalents.

```
+ (BOOL)usesUserKeyEquivalents
```

Return Value
`YES` if menu items conform to user preferences for key equivalents, otherwise `NO`.

Availability
Available in Mac OS X v10.0 and later.

See Also
`+ setUsesUserKeyEquivalents:` (page 10)
`- userKeyEquivalent` (page 34)

Declared In
`NSMenuItem.h`

Instance Methods

action

Returns the receiver's action-method selector.

```
- (SEL)action
```

Return Value
The receiver's action-method selector.

Availability
Available in Mac OS X v10.0 and later.

See Also
`- target` (page 33)
`- setAction:` (page 19)

Related Sample Code
EnhancedAudioBurn
QTAudioExtractionPanel
QTKitPlayer
Quartz Composer WWDC 2005 TextEdit
TextEditPlus

Declared In
`NSMenuItem.h`

attributedTitle

Returns the custom title string for a menu item.

```
- (NSAttributedString *)attributedTitle
```

Return Value
The custom title string for a menu item.

Availability
Available in Mac OS X v10.3 and later.

See Also
- `setAttributedTitle:` (page 20)
- `title` (page 33)

Declared In
`NSMenuItem.h`

hasSubmenu

Returns a Boolean value that indicates whether the receiver has a submenu.

```
- (BOOL)hasSubmenu
```

Return Value
`YES` if the receiver has a submenu, otherwise `NO`.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `setSubmenu:forItem:` (`NSMenu`)

Declared In
`NSMenuItem.h`

image

Returns the image displayed by the receiver.

```
- (NSImage *)image
```

Return Value
The image displayed by the receiver, or `nil` if it displays no image.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `setImage:` (page 22)

Declared In
`NSMenuItem.h`

indentationLevel

Returns the menu item indentation level for the receiver.

```
- (NSInteger)indentationLevel
```

Discussion
The return value is from 0 to 15. The default indentation level is 0.

Availability
Available in Mac OS X v10.3 and later.

See Also
`- setIndentationLevel:` (page 23)

Declared In
`NSMenuItem.h`

initWithTitle:action:keyEquivalent:

Returns an initialized instance of an `NSMenuItem`.

```
- (id)initWithTitle:(NSString *)itemName action:(SEL)anAction keyEquivalent:(NSString
    *)charCode
```

Parameters

`itemName`
The title of the menu item. This value must not be `nil` (if there is no title, specify an empty `NSString`).

`anAction`
The action selector to be associated with the menu item. This value must be a valid selector or `NULL`.

`charCode`
A string representing a keyboard key to be used as the key equivalent. This value must not be `nil` (if there is no key equivalent, specify an empty `NSString`).

Return Value
An instance of `NSMenuItem`, or `nil` if the object couldn't be created.

Discussion
For instances of the `NSMenuItem` class, the default initial state is `NSOffState`, the default on-state image is a check mark, and the default mixed-state image is a dash.

Availability
Available in Mac OS X v10.0 and later.

Related Sample Code
DeskPictAppDockMenu
MenuItemView
ObjectPath
PDFKitLinker2
SearchField

Declared In
`NSMenuItem.h`

isAlternate

Returns a Boolean value that indicates whether the receiver is an alternate to the previous menu item.

```
- (BOOL)isAlternate
```

Return Value
`YES` if the receiver is an alternate to the previous menu item, otherwise `NO`.

Availability
Available in Mac OS X v10.3 and later.

See Also
`- setAlternate:` (page 19)

Declared In
`NSMenuItem.h`

isEnabled

Returns a Boolean value that indicates whether the receiver is enabled.

```
- (BOOL)isEnabled
```

Return Value
`YES` if the receiver is enabled, otherwise `NO`.

Availability
Available in Mac OS X v10.0 and later.

See Also
`- setEnabled:` (page 21)

Declared In
`NSMenuItem.h`

isHidden

Returns a Boolean value that indicates whether the receiver is hidden.

```
- (BOOL)isHidden
```

Return Value
`YES` if the receiver is hidden, otherwise `NO`.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- setHidden:` (page 22)
`- isHiddenOrHasHiddenAncestor` (page 15)

Declared In
`NSMenuItem.h`

isHiddenOrHasHiddenAncestor

Returns a Boolean value that indicates whether the receiver or any of its superitems is hidden.

```
- (BOOL)isHiddenOrHasHiddenAncestor
```

Return Value
`YES` if the receiver or any of its superitems is hidden, otherwise `NO`.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `setHidden:` (page 22)
- `isHidden` (page 14)

Declared In
`NSMenuItem.h`

isHighlighted

Returns a Boolean value that indicates whether the receiver should be drawn highlighted.

```
- (BOOL)isHighlighted
```

Return Value
`YES` if the receiver should be drawn highlighted, otherwise `NO`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`NSMenuItem.h`

isSeparatorItem

Returns a Boolean value that indicates whether the receiver is a separator item.

```
- (BOOL)isSeparatorItem
```

Return Value
`YES` if the receiver is a separator item (that is, a menu item used to visually segregate related menu items), otherwise `NO`.

Availability
Available in Mac OS X v10.0 and later.

Declared In
`NSMenuItem.h`

keyEquivalent

Returns the receiver's unmodified keyboard equivalent.

```
- (NSString *)keyEquivalent
```

Return Value
The receiver's unmodified keyboard equivalent, or the empty string if one hasn't been defined.

Discussion
Use `keyEquivalentModifierMask` (page 16) to determine the modifier mask for the key equivalent.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `userKeyEquivalent` (page 34)
- `mnemonic` (page 17)
- `setKeyEquivalent:` (page 23)

Declared In
`NSMenuItem.h`

keyEquivalentModifierMask

Returns the receiver's keyboard equivalent modifier mask.

```
- (NSUInteger)keyEquivalentModifierMask
```

Availability
Available in Mac OS X v10.0 and later.

See Also
- `setKeyEquivalentModifierMask:` (page 24)

Declared In
`NSMenuItem.h`

menu

Returns the menu to which the receiver belongs.

```
- (NSMenu *)menu
```

Return Value
The menu to which the receiver belongs, or `nil` if no menu has been set.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `setMenu:` (page 24)

Related Sample Code
WhackedTV

Declared In
NSMenuItem.h

mixedStateImage

Returns the image used to depict a “mixed state.”

```
- (NSImage *)mixedStateImage
```

Return Value
The image used to depict a “mixed state.”

Discussion
A mixed state is useful for indicating a mix of “off” and “on” attribute values in a group of selected objects, such as a selection of text containing boldface and plain (non-boldface) words. By default this is a horizontal line.

Availability
Available in Mac OS X v10.0 and later.

See Also
- setMixedStateImage: (page 25)

Declared In
NSMenuItem.h

mnemonic

Deprecated. Returns the character in the menu item title that appears underlined for use as a mnemonic.

```
- (NSString *)mnemonic
```

Discussion
If there is no mnemonic character, returns an empty string.

Availability
Available in Mac OS X v10.0 and later.

See Also
- setTitleWithMnemonic: (page 30)

Declared In
NSMenuItem.h

mnemonicLocation

Deprecated. Returns the position of the underlined character in the menu item title used as a mnemonic.

```
- (NSUInteger)mnemonicLocation
```

Discussion
The position is the zero-based index of that character in the title string. If the receiver has no mnemonic character, returns `NSNotFound`.

Availability
Available in Mac OS X v10.0 and later.

See Also
`- setMnemonicLocation:` (page 25)

Declared In
`NSMenuItem.h`

offStateImage

Returns the image used to depict the receiver's "off" state.

```
- (NSImage *)offStateImage
```

Return Value
The image used to depict the receiver's "off" state, or `nil` if the image has not been set.

Discussion
By default there is no off-state image.

Availability
Available in Mac OS X v10.0 and later.

See Also
`- setOffStateImage:` (page 26)

Declared In
`NSMenuItem.h`

onStateImage

Returns the image used to depict the receiver's "on" state.

```
- (NSImage *)onStateImage
```

Return Value
The image used to depict the receiver's "on" state, or `nil` if the image has not been set.

Discussion
By default this image is a check mark.

Availability
Available in Mac OS X v10.0 and later.

See Also
`- setOnStateImage:` (page 26)

Declared In
`NSMenuItem.h`

representedObject

Returns the object that the receiving menu item represents.

```
- (id)representedObject
```

Discussion
For example, you might have a menu list the names of views that are swapped into the same panel. The represented objects would be the appropriate `NSView` objects. The user would then be able to switch back and forth between the different views that are displayed by selecting the various menu items.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `tag` (page 32)
- `setRepresentedObject:` (page 27)

Declared In
`NSMenuItem.h`

setAction:

Sets the receiver's action-method selector.

```
- (void)setAction:(SEL)aSelector
```

Parameters
aSelector
A selector identifying the action method.

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

Availability
Available in Mac OS X v10.0 and later.

See Also
- `setTarget:` (page 29)
- `action` (page 11)

Related Sample Code
NumberInput_IMKit_Sample
Quartz Composer WWDC 2005 TextEdit
TextEditPlus
UIElementInspector

Declared In
`NSMenuItem.h`

setAlternate:

Marks the receiver as an alternate to the previous menu item.

```
- (void)setAlternate:(BOOL)isAlternate
```

Parameters

isAlternate

`YES` if the receiver is an alternate to the previous menu item, `NO` otherwise.

Discussion

If the receiver has the same key equivalent as the previous item, but has different key equivalent modifiers, the items are folded into a single visible item and the appropriate item shows while tracking the menu, depending on what modifier key (if any) is pressed. The menu items may also have no key equivalent as long as the key equivalent modifiers are different.

Consider the following example: `menuItem1` and `menuItem2` are menu items in the same menu, with `menuItem1` above `menuItem2`:

```
[menuItem1 setTitle:@"One"];
[menuItem1 setKeyEquivalent:@"t"];

[menuItem2 setTitle:@"Two"];
[menuItem2 setKeyEquivalent:@"T"];
[menuItem2 setAlternate:YES];
```

When the menu is displayed, it shows only `menuItem1` (with title "One") instead of two menu items. If the user presses the Shift key while the menu is displayed, `menuItem2` (with title "Two") replaces "One".

If there are two or more items with no key equivalent but different modifiers, then the only way to get access to the alternate items is with the mouse. In the following example,"Two" is shown only if the user presses the Alternate key.

```
[menuItem1 setKeyEquivalent:@""];
[menuItem1 setTitle:@"One"];

[menuItem2 setKeyEquivalent:@""];
[menuItem2 setKeyEquivalentModifierMask:NSAlternateKeyMask];
[menuItem2 setTitle:@"Two"];
```

If you mark items as alternates but their key equivalents don't match, they might be displayed as separate items. Marking the first item as an alternate has no effect.

The *isAlternate* value is archived.

Availability

Available in Mac OS X v10.3 and later.

See Also

- isAlternate (page 14)

Declared In

`NSMenuItem.h`

setAttributedTitle:

Specifies a custom string for a menu item.

```
- (void)setAttributedTitle:(NSAttributedString *)string
```

Parameters

string

An attributed string to use as the receiver's title.

Discussion

You can use this method to add styled text and embedded images to menu item strings. If you do not set a text color for the attributed string, it is black when not selected, white when selected, and gray when disabled. Colored text remains unchanged when selected.

When you call this method to set the menu title to an attributed string, the `setTitle:` (page 30) method is also called to set the menu title with a plain string. If you clear the attributed title, the plain title remains unchanged. To clear the attributed title, set the attributed string to either `nil` or an empty attributed string (`[attrStr length] == 0`).

The attributed string is not archived in the old nib format.

Availability

Available in Mac OS X v10.3 and later.

See Also

- `attributedTitle` (page 11)
- `setTitle:` (page 30)

Declared In

`NSMenuItem.h`

setEnabled:

Sets whether the receiver is enabled

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

Parameters

flag

`YES` if the receiver is to be enabled, otherwise `NO`.

Discussion

This method has no effect unless the menu in which the item will be added or is already a part of has been sent `setAutoenablesItems:NO`. If a menu item is disabled, its keyboard equivalent is also disabled. See the `NSMenuValidation` informal protocol specification for cautions regarding this method.

Availability

Available in Mac OS X v10.0 and later.

See Also

- `isEnabled` (page 14)

Related Sample Code

DeskPictAppDockMenu

MenuItemView

Quartz Composer WWDC 2005 TextEdit

TextEditPlus

WhackedTV

Declared In
NSMenuItem.h

setHidden:

Sets whether the receiver is hidden.

```
- (void)setHidden:(BOOL)hidden
```

Parameters

hidden

YES if the receiver is to be hidden, otherwise NO.

Discussion
Hidden menu items (or items with a hidden superitem) do not appear in a menu and do not participate in command key matching.

Availability
Available in Mac OS X v10.5 and later.

See Also
- isHidden (page 14)
- isHiddenOrHasHiddenAncestor (page 15)

Declared In
NSMenuItem.h

setImage:

Sets the receiver's image.

```
- (void)setImage:(NSImage *)menuImage
```

Parameters

menuImage

An NSImage object representing an image to be displayed in the menu item. If *menuImage* is nil, the current image (if any) is removed.

Discussion
The menu item's image is not affected by changes in its state.

Availability
Available in Mac OS X v10.0 and later.

See Also
- image (page 12)

Related Sample Code
MenuItemView

Declared In
NSMenuItem.h

setIndentationLevel:

Sets the menu item indentation level for the receiver.

```
- (void)setIndentationLevel:(NSInteger)indentationLevel
```

Parameters

indentationLevel

The value for *indentationLevel* may be from 0 to 15. If *indentationLevel* is greater than 15, the value is pinned to the maximum. If *indentationLevel* is less than 0, an exception is raised. The default indentation level is 0.

Discussion

The *indentationLevel* value is archived.

Availability

Available in Mac OS X v10.3 and later.

See Also

- indentationLevel (page 13)

Declared In

NSMenuItem.h

setKeyEquivalent:

Sets the receiver's unmodified key equivalent.

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

Parameters

aString

A string containing a character code representing a keyboard key. If you want to remove the key equivalent from a menu item, pass an empty string (@"") for *aString* (never pass nil).

Discussion

This method considers the case of the letter passed to determine if it has a Shift modifier added. That is, [item setKeyEquivalent:@"w"] sets the key equivalent to Command-w, while [item setKeyEquivalent:@"W"] is Command-Shift-w. You use setKeyEquivalentModifierMask: (page 24) to set the appropriate mask for the modifier keys for the key equivalent.

If you want to specify the Backspace key as the key equivalent for a menu item, use a single character string with NSBackspaceCharacter (defined in NSText.h as 0x08) and for the Forward Delete key, use NSDeleteCharacter (defined in NSText.h as 0x7F). Note that these are not the same characters you get from an NSEvent key-down event when pressing those keys.

Availability

Available in Mac OS X v10.0 and later.

See Also

- setMnemonicLocation: (page 25)
- keyEquivalent (page 16)

Related Sample Code

CocoaDVDPlayer

Declared In
NSMenuItem.h

setKeyEquivalentModifierMask:

Sets the receiver's keyboard equivalent modifiers.

```
- (void)setKeyEquivalentModifierMask:(NSUInteger)mask
```

Parameters

mask

The key masks indicate modifiers such as the Shift or Option keys. *mask* is an integer bit field containing any of these modifier key masks, combined using the C bitwise OR operator:

NSShiftKeyMask
NSAlternateKeyMask
NSCommandKeyMask
NSControlKeyMask

Discussion
In general, you are strongly encouraged to always set NSCommandKeyMask in *mask*, although there may be some conventions where this is not required. For example, in an application that plays media, the Play command may be mapped to just " " (space), without the command key. You can do this with the following code:

```
[menuItem setKeyEquivalent:@" "];
[menuItem setKeyEquivalentModifierMask:0];
```

NSShiftKeyMask is a valid modifier for any key equivalent in *mask*. This allows you to specify key-equivalents such as Command-Shift-1 that are consistent across all keyboards. However, with a few exceptions (such as the German "ß" character), a lowercase character with NSShiftKeyMask is interpreted the same as the uppercase character without that mask. For example, Command-Shift-c and Command-C are considered to be identical key equivalents.

See the NSEvent class specification for more information about modifier mask values.

Availability
Available in Mac OS X v10.0 and later.

See Also
- keyEquivalentModifierMask (page 16)

Declared In
NSMenuItem.h

setMenu:

Sets the receiver's menu.

```
- (void)setMenu:(NSMenu *)aMenu
```

Parameters

aMenu

The menu object that "owns" the receiver.

Discussion

This method is invoked by the owning `NSMenu` object when the receiver is added or removed. You shouldn't have to invoke this method in your own code, although it can be overridden to provide specialized behavior.

Availability

Available in Mac OS X v10.0 and later.

See Also

- `menu` (page 16)

Declared In

`NSMenuItem.h`

setMixedStateImage:

Sets the image of the receiver that indicates a "mixed" state, that is, a state neither "on" nor "off."

```
- (void)setMixedStateImage:(NSImage *)itemImage
```

Parameters

itemImage

The `NSImage` object to use for the "mixed" state of the menu item. If *itemImage* is `nil`, any current mixed-state image is removed.

Discussion

Changing state images is currently unsupported in Mac OS X.

Availability

Available in Mac OS X v10.0 and later.

See Also

- `mixedStateImage` (page 17)
- `setOffStateImage:` (page 26)
- `setOnStateImage:` (page 26)
- `setState:` (page 28)

Related Sample Code

MenuItemView

Declared In

`NSMenuItem.h`

setMnemonicLocation:

Deprecated. Sets the character of the menu item title at location that is to be underlined.

```
- (void)setMnemonicLocation:(NSUInteger)location
```

Parameters

location

An integer index into the character array of the title. *location* must be from 0 to 254.

Discussion

This character identifies the access key by which users can access the menu item.

Availability

Available in Mac OS X v10.0 and later.

See Also

- mnemonicLocation (page 17)

Declared In

NSMenuItem.h

setOffStateImage:

Sets the image of the receiver that indicates an “off” state.

```
- (void)setOffStateImage:(NSImage *)itemImage
```

Parameters

itemImage

The NSImage object to use for the "off" state of the menu item. If *itemImage* is nil, any current off-state image is removed.

Discussion

Changing state images is currently unsupported in Mac OS X.

Availability

Available in Mac OS X v10.0 and later.

See Also

- offStateImage (page 18)
- setMixedStateImage: (page 25)
- setOffStateImage: (page 26)
- setState: (page 28)

Declared In

NSMenuItem.h

setOnStateImage:

Sets the image of the receiver that indicates an “on” state.

```
- (void)setOnStateImage:(NSImage *)itemImage
```

Parameters

itemImage

The NSImage object to use for the "on" state of the menu item. If *itemImage* is nil, any current on-state image is removed.

Discussion
Changing state images is currently unsupported in Mac OS X.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `onStateImage` (page 18)
- `setMixedStateImage:` (page 25)
- `setOffStateImage:` (page 26)
- `setState:` (page 28)

Related Sample Code
MenuItemView

Declared In
`NSMenuItem.h`

setRepresentedObject:

Sets the object represented by the receiver.

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

Parameters

anObject
The object to be represented by the receiver.

Discussion
By setting a represented object for a menu item, you make an association between the menu item and that object. The represented object functions as a more specific form of tag that allows you to associate any object, not just an arbitrary integer, with the items in a menu.

For example, an `NSView` object might be associated with a menu item—when the user chooses the menu item, the represented object is fetched and displayed in a panel. Several menu items might control the display of multiple views in the same panel.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `setTag:` (page 29)
- `representedObject` (page 19)

Related Sample Code
DeskPictAppDockMenu
UIElementInspector

Declared In
`NSMenuItem.h`

setState:

Sets the state of the receiver.

```
- (void)setState:(NSInteger)itemState
```

Parameters

itemState

An integer constant representing a state; it should be one of `NSOffState`, `NSOnState`, or `NSMixedState`.

Discussion

The image associated with the new state is displayed to the left of the menu item.

Availability

Available in Mac OS X v10.0 and later.

See Also

- `state` (page 32)
- `setMixedStateImage:` (page 25)
- `setOffStateImage:` (page 26)
- `setOnStateImage:` (page 26)

Related Sample Code

QTAudioExtractionPanel

QTKitImport

QTKitPlayer

Sketch-112

WhackedTV

Declared In

`NSMenuItem.h`

setSubmenu:

Sets the submenu of the receiver.

```
- (void)setSubmenu:(NSMenu *)aSubmenu
```

Parameters

aSubmenu

The menu object to set as submenu.

Discussion

The default implementation of the `NSMenuItem` class raises an exception if *aSubmenu* already has a supermenu.

Availability

Available in Mac OS X v10.0 and later.

See Also

- `submenu` (page 32)
- `hasSubmenu` (page 12)

Related Sample Code
MenuItemView
PDFKitLinker2
ToolbarSample

Declared In
`NSMenuItem.h`

setTag:

Sets the receiver's tag.

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

Parameters

anInt
An integer tag to associate with the receiver.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `setRepresentedObject:` (page 27)
- `tag` (page 32)

Related Sample Code
QTAudioExtractionPanel
Quartz Composer WWDC 2005 TextEdit
SearchField
TextEditPlus
WhackedTV

Declared In
`NSMenuItem.h`

setTarget:

Sets the receiver's target.

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

Parameters

anObject
An object to be the target of action messages sent by the receiver.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `setAction:` (page 19)
- `target` (page 33)

Related Sample Code
MenuItemView
PDFKitLinker2
Quartz Composer WWDC 2005 TextEdit
TextEditPlus
UIElementInspector

Declared In
`NSMenuItem.h`

setTitle:

Sets the receiver's title.

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

Parameters
aString
The new title of the menu item. If you do not want a title, use an empty string (`@""`), not `nil`.

Availability
Available in Mac OS X v10.0 and later.

See Also
`- title` (page 33)
`- setAttributedTitle:` (page 20)

Related Sample Code
CocoaDVDPlayer
PDFKitLinker2
QTAudioExtractionPanel
ToolbarSample
WhackedTV

Declared In
`NSMenuItem.h`

setTitleWithMnemonic:

Deprecated. Sets the title of a menu item with a character denoting an access key.

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

Discussion
Use an ampersand character to mark the character (the one following the ampersand) to be designated.

Availability
Available in Mac OS X v10.0 and later.

See Also
`- mnemonic` (page 17)

- setMnemonicLocation: (page 25)

Related Sample Code
Quartz Composer WWDC 2005 TextEdit
TextEditPlus

Declared In
NSMenuItem.h

setToolTip:

Sets a help tag for a menu item.

```
- (void)setToolTip:(NSString *)toolTip
```

Parameters

toolTip
A short string that describes the menu item.

Discussion
You can invoke this method for any menu item, including items in the main menu bar. This string is not archived in the old nib format.

Availability
Available in Mac OS X v10.3 and later.

See Also
- toolTip (page 34)

Declared In
NSMenuItem.h

setView:

Sets the content view for the receiver.

```
- (void)setView:(NSView *)view
```

Parameters

view
The content view for the receiver.

Discussion
A menu item with a view does not draw its title, state, font, or other standard drawing attributes, and assigns drawing responsibility entirely to the view. Keyboard equivalents and type-select continue to use the key equivalent and title as normal. For more details, see *Application Menu and Pop-up List Programming Topics for Cocoa*.

Availability
Available in Mac OS X v10.5 and later.

See Also
- view (page 34)

Related Sample Code
MenuItemView

Declared In
`NSMenuItem.h`

state

Returns the state of the receiver.

```
- (NSInteger)state
```

Return Value
The state of the receiver—one of `NSOffState` (the default), `NSOnState`, or `NSMixedState`.

Availability
Available in Mac OS X v10.0 and later.

See Also
`- setState:` (page 28)

Declared In
`NSMenuItem.h`

submenu

Returns the submenu associated with the receiving menu item.

```
- (NSMenu *)submenu
```

Return Value
The submenu associated with the receiving menu item, or `nil` if no submenu is associated with it.

Discussion
If the receiver responds `YES` to `hasSubmenu` (page 12), the submenu is returned.

Availability
Available in Mac OS X v10.0 and later.

See Also
`- hasSubmenu` (page 12)
`- setSubmenu:` (page 28)

Related Sample Code
EnhancedAudioBurn

Declared In
`NSMenuItem.h`

tag

Returns the receiver's tag.

```
- (NSInteger)tag
```

Return Value
The receiver's tag.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `representedObject` (page 19)
- `setTag:` (page 29)

Related Sample Code
Quartz Composer WWDC 2005 TextEdit
TextEditPlus
ThreadsExporter
ThreadsImporter
ThreadsImportMovie

Declared In
`NSMenuItem.h`

target

Returns the receiver's target.

```
- (id)target
```

Return Value
The receiver's target.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `action` (page 11)
- `setTarget:` (page 29)

Related Sample Code
EnhancedDataBurn

Declared In
`NSMenuItem.h`

title

Returns the receiver's title.

```
- (NSString *)title
```

Return Value
The receiver's title.

Availability
Available in Mac OS X v10.0 and later.

See Also
- `setTitle:` (page 30)

Related Sample Code
WhackedTV

Declared In
`NSMenuItem.h`

toolTip

Returns the help tag for a menu item.

```
- (NSString *)toolTip
```

Availability
Available in Mac OS X v10.3 and later.

See Also
- `setToolTip:` (page 31)

Declared In
`NSMenuItem.h`

userKeyEquivalent

Returns the user-assigned key equivalent for the receiver.

```
- (NSString *)userKeyEquivalent
```

Availability
Available in Mac OS X v10.0 and later.

See Also
- `keyEquivalent` (page 16)

Declared In
`NSMenuItem.h`

view

Returns the view for the receiver.

```
- (NSView *)view
```

Return Value
The view for the receiver.

Discussion
By default, a menu item has a `nil` view.

See `setView:` (page 31) for more details.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- setView:` (page 31)

Declared In
`NSMenuItem.h`

Document Revision History

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

Date	Notes
2008-02-08	Corrected the discussion for setKeyEquivalentModifierMask:.
2007-01-19	Included API introduced in Mac OS X v10.5
2006-05-23	First publication of this content as a separate document.

Document Revision History

Index

T

U

V