
NSTextAttachmentCell Protocol Reference

[Cocoa > Objective-C Language](#)



2006-05-23



Apple Inc.
© 2006 Apple Computer, Inc.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

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

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

Simultaneously published in the United States and Canada.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY

DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Contents

NSTextAttachmentCell Protocol Reference 5

Overview	5
Tasks	5
Drawing	5
Cell Size and Position	6
Event Handling	6
Setting the Attachment	6
Instance Methods	6
attachment	6
cellBaselineOffset	7
cellFrameForTextContainer:proposedLineFragment:glyphPosition:characterIndex:	7
cellSize	8
drawWithFrame:inView:	8
drawWithFrame:inView:characterIndex:	8
drawWithFrame:inView:characterIndex:layoutManager:	8
highlight:withFrame:inView:	9
setAttachment:	9
trackMouse:inRect:ofView:atCharacterIndex:untilMouseUp:	9
trackMouse:inRect:ofView:untilMouseUp:	10
wantsToTrackMouse	11
wantsToTrackMouseForEvent:inRect:ofView:atCharacterIndex:	11

Document Revision History 13

Index 15

NSTextAttachmentCell Protocol Reference

Adopted by	NSTextAttachmentCell
Conforms to	NSObject
Framework	/System/Library/Frameworks/AppKit.framework
Availability	Available in Mac OS X v10.0 and later.
Declared in	NSTextAttachment.h
Companion guides	Text System Overview Text Attachment Programming Topics for Cocoa

Overview

The NSTextAttachmentCell protocol declares the interface for objects that draw text attachment icons and handle mouse events on their icons. With the exceptions of [cellBaselineOffset](#) (page 7), [setAttachment:](#) (page 9), and [attachment](#) (page 6), all of these methods are implemented by the NSCell class and described in that class specification.

See the NSAttributedString and NSTextView class specifications for general information on text attachments.

Tasks

Drawing

- [drawWithFrame:inView:](#) (page 8)
Draws the receiver's image within *cellFrame* in *aView*, which is the view currently focused.
- [drawWithFrame:inView:characterIndex:](#) (page 8)
Draws the receiver's image within *cellFrame* in *aView*, which is the view currently focused. *charIndex* is the index of the attachment character within the text.
- [drawWithFrame:inView:characterIndex:layoutManager:](#) (page 8)
Draws the receiver's image within *cellFrame* in *controlView*, which is the view currently focused. *charIndex* is the index of the attachment character within the text. *layoutManager* is the layout manager for the text.

- [highlight:withFrame:inView:](#) (page 9)
Draws the receiver's image—with highlighting if *flag* is YES—within *cellFrame* in *aView*, which should be the focus view.

Cell Size and Position

- [cellSize](#) (page 8)
Returns the size of the attachment's icon.
- [cellBaselineOffset](#) (page 7)
Returns the position where the attachment cell's image should be drawn in text, relative to the current point established in the glyph layout.
- [cellFrameForTextContainer:proposedLineFragment:glyphPosition:characterIndex:](#) (page 7)
Returns the frame of the cell as it would be drawn as the character at the given glyph *position*, and character index, *charIndex*, in *textContainer*.

Event Handling

- [wantsToTrackMouse](#) (page 11)
Returns YES if the receiver will handle a mouse event occurring over its image (to support dragging, for example), NO otherwise.
- [wantsToTrackMouseForEvent:inRect:ofView:atCharacterIndex:](#) (page 11)
Allows an attachment to specify what events it would want to track the mouse for.
- [trackMouse:inRect:ofView:untilMouseUp:](#) (page 10)
Handles a mouse-down event on the receiver's image.
- [trackMouse:inRect:ofView:atCharacterIndex:untilMouseUp:](#) (page 9)
Handles a mouse-down event on the receiver's image.

Setting the Attachment

- [setAttachment:](#) (page 9)
Sets the text attachment object that owns the receiver to *anAttachment*, without retaining it (the text attachment, as the owner, retains the cell).
- [attachment](#) (page 6)
Returns the text attachment object that owns the receiver.

Instance Methods

attachment

Returns the text attachment object that owns the receiver.

- (NSTextAttachment *)attachment

Availability

Available in Mac OS X v10.0 and later.

See Also

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

Related Sample Code

Quartz Composer WWDC 2005 TextEdit
TextEditPlus

Declared In

NSTextAttachment.h

cellBaselineOffset

Returns the position where the attachment cell's image should be drawn in text, relative to the current point established in the glyph layout.

- (NSPoint)cellBaselineOffset

Discussion

The image should be drawn so its lower-left corner lies on this point.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [icon](#) (NSFileWrapper)

Declared In

NSTextAttachment.h

cellFrameForTextContainer:proposedLineFragment:glyphPosition:characterIndex:

Returns the frame of the cell as it would be drawn as the character at the given glyph *position*, and character index, *charIndex*, in *textContainer*.

```
- (NSRect)cellFrameForTextContainer:(NSTextContainer *)textContainer
    proposedLineFragment:(NSRect)lineFrag glyphPosition:(NSPoint)position
    characterIndex:(NSUInteger)charIndex
```

Discussion

The proposed line fragment is specified by *lineFrag*.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSTextAttachment.h

cellSize

Returns the size of the attachment's icon.

- (NSSize)cellSize

Availability

Available in Mac OS X v10.0 and later.

See Also

- icon (NSFileWrapper)
- fileWrapper (NSTextAttachment)

Declared In

NSTextAttachment.h

drawWithFrame:inView:

Draws the receiver's image within *cellFrame* in *aView*, which is the view currently focused.

- (void)drawWithFrame:(NSRect)cellFrame inView:(NSView *)aView

Availability

Available in Mac OS X v10.0 and later.

See Also

- drawWithFrame:inView: (NSCell)
- lockFocus (NSView)

Declared In

NSTextAttachment.h

drawWithFrame:inView:characterIndex:

Draws the receiver's image within *cellFrame* in *aView*, which is the view currently focused. *charIndex* is the index of the attachment character within the text.

- (void)drawWithFrame:(NSRect)cellFrame inView:(NSView *)aView
characterIndex:(NSUInteger)charIndex

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSTextAttachment.h

drawWithFrame:inView:characterIndex:layoutManager:

Draws the receiver's image within *cellFrame* in *controlView*, which is the view currently focused. *charIndex* is the index of the attachment character within the text. *layoutManager* is the layout manager for the text.


```
- (void)drawWithFrame:(NSRect)cellFrame inView:(NSView *)controlView
    characterIndex:(NSUInteger)charIndex layoutManager:(NSLayoutManager
    *)layoutManager
```

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSTextAttachment.h

highlight:withFrame:inView:

Draws the receiver's image—with highlighting if *flag* is YES—within *cellFrame* in *aView*, which should be the focus view.

```
- (void)highlight:(BOOL)flag withFrame:(NSRect)cellFrame inView:(NSView *)aView
```

Availability

Available in Mac OS X v10.0 and later.

See Also

- `highlight:withFrame:inView:` (NSCell)
- `lockFocus` (NSView)

Declared In

NSTextAttachment.h

setAttachment:

Sets the text attachment object that owns the receiver to *anAttachment*, without retaining it (the text attachment, as the owner, retains the cell).

```
- (void)setAttachment:(NSTextAttachment *)anAttachment
```

Availability

Available in Mac OS X v10.0 and later.

See Also

- [attachment](#) (page 6)
- `setAttachmentCell:` (NSTextAttachment)

Declared In

NSTextAttachment.h

trackMouse:inRect:ofView:atCharacterIndex:untilMouseUp:

Handles a mouse-down event on the receiver's image.

```
- (BOOL)trackMouse:(NSEvent *)theEvent inRect:(NSRect)cellFrame ofView:(NSView
    *)aTextView atCharacterIndex:(NSUInteger)charIndex untilMouseUp:(BOOL)flag
```

Discussion

theEvent is the mouse-down event. *cellFrame* is the region of *aTextView* in which you should track further mouse events. *charIndex* is the position in the text at which this attachment appears. *aTextView* is the view that received the event. It's assumed to be an `NSTextView`, and should be the focus view. If *flag* is YES, the receiver tracks the mouse until a mouse-up event occurs; if *flag* is NO, it stops tracking when a mouse-dragged event occurs outside of *cellFrame*. Returns YES if the receiver successfully finished tracking the mouse (typically through a mouse-up event), NO otherwise (such as when the mouse is dragged outside *cellFrame*).

`NSTextAttachmentCell`'s implementation of this method calls upon *aTextView*'s delegate to handle the event. If *theEvent* is a mouse-up event for a double click, the text attachment cell sends the delegate a `textView:doubleClickedOnCell:inRect:message` and returns YES. Otherwise, depending on whether the user clicks or drags the cell, it sends the delegate a `textView:clickedOnCell:inRect:or a textView:draggedCell:inRect:event:message` and returns YES. `NSTextAttachmentCell`'s implementation returns NO only if *flag* is NO and the mouse is dragged outside of *cellFrame*. The delegate methods are invoked only if the delegate responds.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSTextAttachment.h`

trackMouse:inRect:ofView:untilMouseUp:

Handles a mouse-down event on the receiver's image.

```
- (BOOL)trackMouse:(NSEvent *)theEvent inRect:(NSRect)cellFrame ofView:(NSView *)aTextView untilMouseUp:(BOOL)flag
```

Discussion

theEvent is the mouse-down event. *cellFrame* is the region of *aTextView* in which further mouse events should be tracked. *aTextView* is the view that received the event. It's assumed to be an `NSTextView` and should be the focus view. If *flag* is YES, the receiver tracks the mouse until a mouse-up event occurs; if *flag* is NO, it stops tracking when a mouse-dragged event occurs outside of *cellFrame*. Returns YES if the receiver successfully finished tracking the mouse (typically through a mouse-up event), NO otherwise (such as when the cursor is dragged outside *cellFrame*).

`NSTextAttachmentCell`'s implementation of this method calls upon the delegate of *aTextView* to handle the event. If *theEvent* is a mouse-up event for a double click, the text attachment cell sends the delegate a `textView:doubleClickedOnCell:inRect:message` and returns YES. Otherwise, depending on whether the user clicks or drags the cell, it sends the delegate a `textView:clickedOnCell:inRect:or a textView:draggedCell:inRect:event:message` and returns YES. `NSTextAttachmentCell`'s implementation returns NO only if *flag* is NO and the cursor is dragged outside of *cellFrame*. The delegate methods are invoked only if the delegate responds.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [wantsToTrackMouse](#) (page 11)
- `trackMouse:inRect:ofView:untilMouseUp:` (`NSCell`)
- `lockFocus` (`NSView`)

Declared In

NSTextAttachment.h

wantsToTrackMouse

Returns YES if the receiver will handle a mouse event occurring over its image (to support dragging, for example), NO otherwise.

- (BOOL)wantsToTrackMouse

Discussion

NSTextAttachmentCell's implementation of this method returns YES. The NSView containing the cell should invoke this method before sending a [trackMouse:inRect:ofView:untilMouseUp:](#) (page 10) message.

For an attachment in an attributed string, if the attachment cell returns NO its attachment character should be selected rather than the cell being asked to track the mouse. This results in the attachment icon behaving as any regular glyph in text.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSTextAttachment.h

wantsToTrackMouseForEvent:inRect:ofView:atCharacterIndex:

Allows an attachment to specify what events it would want to track the mouse for.

- (BOOL)wantsToTrackMouseForEvent:(NSEvent *)*theEvent* inRect:(NSRect)*cellFrame* ofView:(NSView *)*controlView* atCharacterIndex:(NSUInteger)*charIndex*

Discussion

theEvent is the event in question that occurred in *cellFrame* inside *controlView*. *charIndex* is the index of the attachment character within the text. If [wantsToTrackMouse](#) (page 11) returns YES, this method allows the attachment to decide whether it wishes to do so for particular events.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSTextAttachment.h

Document Revision History

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

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

REVISION HISTORY

Document Revision History

Index

A

attachment **protocol instance method 6**

C

cellBaselineOffset **protocol instance method 7**
cellFrameForTextContainer:proposedLineFragment:
 glyphPosition:characterIndex: **protocol
instance method 7**
cellSize **protocol instance method 8**

D

drawWithFrame:inView: **protocol instance method 8**
drawWithFrame:inView:characterIndex: **protocol
instance method 8**
drawWithFrame:inView:characterIndex:layoutManager:
protocol instance method 8

H

highlight:withFrame:inView: **protocol instance
method 9**

S

setAttachment: **protocol instance method 9**

T

trackMouse:inRect:ofView:atCharacterIndex:
 untilMouseUp: **protocol instance method 9**

trackMouse:inRect:ofView:untilMouseUp: **protocol
instance method 10**

W

wantsToTrackMouse **protocol instance method 11**
wantsToTrackMouseForEvent:inRect:ofView:
 atCharacterIndex: **protocol instance method 11**