# NSTextInput Protocol Reference

Cocoa > Events & Other Input

2006-05-23

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

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

The Apple logo is a trademark of Apple Inc.

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

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

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

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

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

Simultaneously published in the United States and Canada.

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

**IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.**

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

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

# Contents

# NSTextInput Protocol Reference

| | |
|---|---|
| **Adopted by** | NSInputManager |
| **Framework** | /System/Library/Frameworks/AppKit.framework |
| **Availability** | Available in Mac OS X v10.0 and later. |
| **Declared in** | NSInputManager.h |
| **Companion guides** | Text System Overview<br>Text Input Management |

## Overview

The `NSTextInput` protocol defines the methods that Cocoa text views must implement in order to interact properly with the text input management system. `NSTextView` and its abstract superclass `NSText` are the only classes included in Cocoa that implement `NSTextInput`. To create another text view class, you can either subclass `NSTextView` (and not `NSText`, for historical reasons), or subclass `NSView` and implement the `NSTextInput` protocol.

**Important:** Methods specific to the `NSTextInput` protocol are intended for dealing with text input and generally are not suitable for other purposes.

## Tasks

### Marked Text

– `hasMarkedText` (page 8)
Returns a Boolean value indicating whether or not the receiver has marked text.

– `markedRange` (page 9)
Returns the range of the marked text.

– `selectedRange` (page 10)
Returns the range of selected text.

– `setMarkedText:selectedRange:` (page 10)
Replaces currently marked text in the receiver's text storage with the given string and sets the selection to the given range, computed from the beginning of the marked text.

- `unmarkText` (page 11)
  Removes any marking from pending input text and disposes of the marked text as it wishes. The text view should accept the marked text as if it had been inserted normally.
- `validAttributesForMarkedText` (page 11)
  Returns an array of names for the attributes supported by the receiver.

### Text Storage

- `attributedSubstringFromRange:` (page 6)
  Returns an attributed string derived from the given range in the receiver's text storage.
- `insertText:` (page 9)
  Inserts the given string into the receiver's text storage.

### Character Coordinates

- `characterIndexForPoint:` (page 7)
  Returns the index of the character whose frame rectangle includes the given point.
- `firstRectForCharacterRange:` (page 8)
  Returns the first frame rectangle for characters in the given range, in screen coordinates.

### Key Bindings

- `doCommandBySelector:` (page 8)
  Invokes the given selector if possible.

### Other

- `conversationIdentifier` (page 7)
  Returns a number used to identify the receiver's context to the input server.

## Instance Methods

### attributedSubstringFromRange:

Returns an attributed string derived from the given range in the receiver's text storage.

```
- (NSAttributedString *)attributedSubstringFromRange:(NSRange)theRange
```

**Parameters**

*theRange*
  The range in the text storage from which to create the returned string.

**Return Value**
The string created from the given range.

**Discussion**
This method allows input mangers to query any range in text storage.

An implementation of this method should be prepared for *theRange* to be out-of-bounds. For example, the InkWell text input service can ask for the contents of the text input client that extends beyond the document's range. In this case, you should return the intersection of the document's range and *theRange*. If the location of *theRange* is completely outside of the document's range, return `nil`.

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

**Declared In**
`NSInputManager.h`

## characterIndexForPoint:

Returns the index of the character whose frame rectangle includes the given point.

```
- (NSUInteger)characterIndexForPoint:(NSPoint)thePoint
```

**Parameters**

*thePoint*
A point, in screen coordinates.

**Return Value**
The character index, measured from the start of the receiver's text storage, of the character containing the given point. Returns `NSNotFound` if the cursor is not within a character.

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

**Declared In**
`NSInputManager.h`

## conversationIdentifier

Returns a number used to identify the receiver's context to the input server.

```
- (NSInteger)conversationIdentifier
```

**Return Value**
The identifying number of the receiver.

**Discussion**
Each text view within an application should return a unique identifier (typically its address). However, multiple text views sharing the same text storage must all return the same identifier.

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

**Declared In**
`NSInputManager.h`

## doCommandBySelector:

Invokes the given selector if possible.

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

### Parameters

*aSelector*

The selector to be invoked.

### Discussion

If *aSelector* cannot be invoked, then `doCommandBySelector:` should not pass this message up the responder chain. `NSResponder` also implements this method, and it does forward uninvokable commands up the responder chain, but a text view should not. A text view implementing the NSTextInput protocol inherits from `NSView`, which inherits from `NSResponder`, so your implementation of this method will override the one in `NSResponder`. It should not call `super`.

### Availability

Available in Mac OS X v10.0 and later.

### See Also

`- interpretKeyEvents:` (NSResponder)
`- doCommandBySelector:` (NSResponder)

### Declared In

`NSInputManager.h`

## firstRectForCharacterRange:

Returns the first frame rectangle for characters in the given range, in screen coordinates.

```
- (NSRect)firstRectForCharacterRange:(NSRange)theRange
```

### Parameters

*theRange*

The character range whose frame is returned.

### Return Value

The frame rectangle for the given range of characters.

### Discussion

If *theRange* spans multiple lines of text in the text view, the rectangle returned is the one for the characters in the first line. If the length of *theRange* is 0 (as it would be if there is nothing selected at the insertion point), the rectangle coincides with the insertion point, and its width is 0.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

`NSInputManager.h`

## hasMarkedText

Returns a Boolean value indicating whether or not the receiver has marked text.

```
- (BOOL)hasMarkedText
```

**Return Value**
YES if the receiver has marked text, NO if it doesn't.

**Discussion**
Unlike other methods in this protocol, this one is not called by an input server. The text view itself may call this method to determine whether there currently is marked text. NSTextView, for example, disables the Edit > Copy menu item when this method returns YES.

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

**See Also**
- markedRange (page 9)

**Declared In**
NSInputManager.h

## insertText:

Inserts the given string into the receiver's text storage.

```
- (void)insertText:(id)aString
```

**Parameters**

*aString*

Either an NSString or an NSAttributedString object.

**Discussion**
This method is the entry point for inserting text typed by the user and is generally not suitable for other purposes. Programmatic modification of the text is best done by operating on the text storage directly. Because this method pertains to the actions of the user, the text view must be editable for the insertion to work.

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

**Declared In**
NSInputManager.h

## markedRange

Returns the range of the marked text.

```
- (NSRange)markedRange
```

**Return Value**
The range of marked text.

**Discussion**
The returned range measures from the start of the receiver's text storage. The return value's location is NSNotFound, and its length is 0 if and only if hasMarkedText (page 8) returns NO.

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

**See Also**
- `setMarkedText:selectedRange:` (page 10)
- `unmarkText` (page 11)
- `hasMarkedText` (page 8)

**Declared In**
`NSInputManager.h`

## selectedRange

Returns the range of selected text.

```
- (NSRange)selectedRange
```

**Return Value**
The range of selected text.

**Discussion**
The returned range measures from the start of the receiver's text storage. If there is no selection, the return value's `location` is `NSNotFound`, and its `length` is 0.

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

**See Also**
- `setMarkedText:selectedRange:` (page 10)

**Declared In**
`NSInputManager.h`

## setMarkedText:selectedRange:

Replaces currently marked text in the receiver's text storage with the given string and sets the selection to the given range, computed from the beginning of the marked text.

```
- (void)setMarkedText:(id)aString selectedRange:(NSRange)selRange
```

**Parameters**

*aString*

Either an `NSString` or an `NSAttributedString` object; must not be `nil`.

*selRange*

The range within *aString* to set as the selection.

**Discussion**
If there is no marked text, the current selection is replaced. If there is no selection, the string is inserted at the insertion point.

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

### See Also

- `selectedRange` (page 10)
- `unmarkText` (page 11)

### Declared In

`NSInputManager.h`

## unmarkText

Removes any marking from pending input text and disposes of the marked text as it wishes. The text view should accept the marked text as if it had been inserted normally.

```
- (void)unmarkText
```

### Availability

Available in Mac OS X v10.0 and later.

### See Also

- `selectedRange` (page 10)
- `setMarkedText:selectedRange:` (page 10)

### Declared In

`NSInputManager.h`

## validAttributesForMarkedText

Returns an array of names for the attributes supported by the receiver.

```
- (NSArray *)validAttributesForMarkedText
```

### Return Value

An array of `NSString` objects representing names for the supported attributes.

### Discussion

The input server may choose to use some of these attributes in the text it inserts or in marked text. Returns an empty array if no attributes are supported. See NSAttributedString Additions for the set of string constants that you could return in the array.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

`NSInputManager.h`

# Document Revision History

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

| Date | Notes |
|---|---|
| 2006-05-23 | First publication of this content as a separate document. |
| | Revised description of setMarkedText:selectedRange: method. |

# Index