

---

# NSTextStorage Class Reference

User Experience: Text Layout



2009-07-10



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

Helvetica is a registered trademark of Heidelberger Druckmaschinen AG, available from Linotype Library GmbH.

Java and all Java-based trademarks are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. 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

## **NSTextStorage Class Reference 5**

---

Overview	5
Tasks	6
Managing NSLayoutManager Objects	6
Handling Text Edited Messages	6
Determining the Nature of Changes	6
Determining the Extent of Changes	6
Setting the Delegate	7
Getting and Setting Scriptable Properties	7
Instance Methods	7
addLayoutManager:	7
attributeRuns	8
changeInLength	8
characters	9
delegate	9
edited:range:changeInLength:	9
editedMask	10
editedRange	11
ensureAttributesAreFixedInRange:	11
fixesAttributesLazily	12
font	12
foregroundColor	12
invalidateAttributesInRange:	13
layoutManagers	13
paragraphs	13
processEditing	14
removeLayoutManager:	14
setAttributeRuns:	15
setCharacters:	15
setDelegate:	15
setFont:	16
setForegroundColor:	16
setParagraphs:	16
setWords:	17
words	17
Constants	17
Change notifications	17
Notifications	18
NSTextStorageDidProcessEditingNotification	18
NSTextStorageWillProcessEditingNotification	18

**Document Revision History 19**

---

**Index 21**

---

# NSTextStorage Class Reference

---

<b>Inherits from</b>	NSMutableAttributedString : NSAttributedString : NSObject
<b>Conforms to</b>	NSCoding (NSAttributedString) NSCopying (NSAttributedString) NSMutableCopying (NSAttributedString) NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/AppKit.framework
<b>Availability</b>	Available in Mac OS X v10.0 and later.
<b>Declared in</b>	NSTextStorage.h NSTextStorageScripting.h
<b>Companion guides</b>	Text System Overview Text System Storage Layer Overview Cocoa Scripting Guide
<b>Related sample code</b>	Quartz Composer WWDC 2005 TextEdit QuickLookSketch Sketch+Accessibility Sketch-112 TextSizingExample

## Overview

`NSTextStorage` is a semiconcrete subclass of `NSMutableAttributedString` that manages a set of client `NSLayoutManager` objects, notifying them of any changes to its characters or attributes so that they can relay and redisplay the text as needed. `NSTextStorage` defines the fundamental storage mechanism of the Application Kit's extended text-handling system.

`NSTextStorage` also defines a set of methods, listed under "Getting and setting scriptable properties" in the Method Types section, useful for getting and setting scriptable properties of `NSTextStorage` objects. Unless you are dealing with scriptability, you do not normally need to invoke these methods directly. In particular, using the `characters`, `words` or `paragraphs` methods or their corresponding setter methods is an inefficient way to manipulate the text storage, since these methods create and return many objects. Instead, use the text access methods defined by `NSMutableAttributedString`, `NSAttributedString`, `NSMutableString`, and `NSString` to perform character-level manipulation.

## Tasks

### Managing NSLayoutManager Objects

- [addLayoutManager:](#) (page 7)  
Adds a layout manager to the receiver's set of layout managers.
- [removeLayoutManager:](#) (page 14)  
Removes a layout manager from the receiver's set of layout managers.
- [layoutManagers](#) (page 13)  
Returns the receiver's layout managers.

### Handling Text Edited Messages

- [edited:range:changeInLength:](#) (page 9)  
Tracks changes made to the receiver, allowing the text storage to record the full extent of changes made.
- [ensureAttributesAreFixedInRange:](#) (page 11)  
Ensures that attributes are fixed in the given range.
- [fixesAttributesLazily](#) (page 12)  
Returns whether the receiver fixes attributes lazily.
- [invalidateAttributesInRange:](#) (page 13)  
Invalidates attributes in the specified range.
- [processEditing](#) (page 14)  
Cleans up changes made to the receiver and notifies its delegate and layout managers of changes.

### Determining the Nature of Changes

- [editedMask](#) (page 10)  
Returns the kinds of edits pending for the receiver

### Determining the Extent of Changes

- [editedRange](#) (page 11)  
Returns the range of the receiver to which pending changes have been made, whether of characters or of attributes.
- [changeInLength](#) (page 8)  
Returns the difference between the current length of the edited range and its length before editing began.

## Setting the Delegate

- [setDelegate:](#) (page 15)  
Sets the receiver's delegate.
- [delegate](#) (page 9)  
Returns the receiver's delegate.

## Getting and Setting Scriptable Properties

- [attributeRuns](#) (page 8)  
Returns an array of the receiver's attribute runs.
- [setAttributeRuns:](#) (page 15)  
Sets the receiver's attribute runs.
- [characters](#) (page 9)  
Returns the receiver's text as an array of characters.
- [setCharacters:](#) (page 15)  
Sets the text storage's text.
- [font](#) (page 12)  
Returns the receiver's font.
- [setFont:](#) (page 16)  
Sets the text storage's font.
- [foregroundColor](#) (page 12)  
Returns the text storage's foreground color.
- [setForegroundColor:](#) (page 16)  
Sets the text storage's foreground color.
- [paragraphs](#) (page 13)  
Returns an array of the text storage's paragraphs.
- [setParagraphs:](#) (page 16)  
Sets the text storage's paragraphs.
- [words](#) (page 17)  
Returns an array of the text storage's words.
- [setWords:](#) (page 17)  
Sets the text storage's words.

## Instance Methods

### **addLayoutManager:**

Adds a layout manager to the receiver's set of layout managers.

- (void)addLayoutManager:(NSLayoutManager \*)aLayoutManager

**Parameters***aLayoutManager*

The layout manager to add.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [removeLayoutManager:](#) (page 14)
- [layoutManagers](#) (page 13)

**Related Sample Code**

Quartz Composer WWDC 2005 TextEdit

QuickLookSketch

Sketch+Accessibility

Sketch-112

TextSizingExample

**Declared In**

NSTextStorage.h

**attributeRuns**

Returns an array of the receiver's attribute runs.

- (NSArray \*)attributeRuns

**Return Value**

An array of the receiver's attribute runs.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSTextStorageScripting.h

**changeInLength**

Returns the difference between the current length of the edited range and its length before editing began.

- (NSInteger)changeInLength

**Return Value**

The difference between the current length of the edited range and its length before editing began. That is, before the receiver was sent the first `beginEditing` message or a single `edited:range:changeInLength:` (page 9) message.

**Discussion**

This difference is accumulated with each invocation of `edited:range:changeInLength:` (page 9), until a final message processes the changes.

The receiver's delegate and layout managers can use this information to determine the nature of edits in their respective notification methods.

### Availability

Available in Mac OS X v10.0 and later.

### See Also

- [editedRange](#) (page 11)
- [editedMask](#) (page 10)

### Declared In

NSTextStorage.h

## characters

Returns the receiver's text as an array of characters.

- (NSArray \*)characters

### Special Considerations

Do not use this method unless you are dealing directly with scriptability. For indexed access to characters, use NSAttributedString's `length` method to access the string, and NSString's `characterAtIndex:` method to access the individual characters.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

NSTextStorageScripting.h

## delegate

Returns the receiver's delegate.

- (id)delegate

### Return Value

The receiver's delegate.

### Availability

Available in Mac OS X v10.0 and later.

### See Also

- [setDelegate:](#) (page 15)

### Declared In

NSTextStorage.h

## edited:range:changeInLength:

Tracks changes made to the receiver, allowing the text storage to record the full extent of changes made.

- (void)edited:(NSUInteger)mask range:(NSRange)oldRange  
changeInLength:(NSUInteger)lengthChange

**Parameters***mask*

A mask specifying the nature of the changes. The value is made by combining with the C bitwise OR operator the options described in “[Change notifications](#)” (page 17).

*oldRange*

The extent of characters affected before the change took place.

*lengthChange*

The number of characters added to or removed from *oldRange*. If no characters were edited as noted by *mask*, its value is irrelevant and undefined. For example, when replacing “The” with “Several” in the string “The files couldn’t be saved”, *oldRange* is {0, 3} and *lengthChange* is 4.

**Discussion**

This method invokes [processEditing](#) (page 14). `NSTextStorage` invokes this method automatically each time it makes a change to its attributed string. Subclasses that override or add methods that alter their attributed strings directly should invoke this method after making those changes; otherwise you should not invoke this method. The information accumulated with this method is then used in an invocation of [processEditing](#) (page 14) to report the affected portion of the receiver.

The methods for querying changes, [editedRange](#) (page 11) and [changeInLength](#) (page 8), indicate the extent of characters affected after the change. This method expects the characters before the change because that information is readily available as the argument to whatever method performs the change (such as `replaceCharactersInRange:withString:`).

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

`ClipboardViewer`

**Declared In**

`NSTextStorage.h`

**editedMask**

Returns the kinds of edits pending for the receiver

- (NSUInteger)editedMask

**Return Value**

A mask describing the kinds of edits pending for the receiver.

**Discussion**

Use the C bitwise AND operator to test the mask; testing for equality will fail if additional mask flags are added later. The receiver’s delegate and layout managers can use this information to determine the nature of edits in their respective notification methods.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [editedRange](#) (page 11)
- [changeInLength](#) (page 8)

**Declared In**

NSTextStorage.h

**editedRange**

Returns the range of the receiver to which pending changes have been made, whether of characters or of attributes.

- (NSRange)editedRange

**Return Value**

The range of the receiver to which pending changes have been made, whether of characters or of attributes.

**Discussion**

The receiver's delegate and layout managers can use this information to determine the nature of edits in their respective notification methods.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [changeInLength](#) (page 8)
- [editedMask](#) (page 10)

**Declared In**

NSTextStorage.h

**ensureAttributesAreFixedInRange:**

Ensures that attributes are fixed in the given range.

- (void)ensureAttributesAreFixedInRange:(NSRange)range

**Parameters**

*range*

The range of characters whose attributes might be examined.

**Discussion**

An NSTextStorage object using lazy attribute fixing is required to call this method before accessing any attributes within *range*. This method gives attribute fixing a chance to occur if necessary. NSTextStorage subclasses wishing to support laziness must call this method from all attribute accessors they implement.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [fixesAttributesLazily](#) (page 12)
- [invalidateAttributesInRange:](#) (page 13)

**Declared In**

NSTextStorage.h

## **fixesAttributesLazily**

Returns whether the receiver fixes attributes lazily.

- (BOOL)fixesAttributesLazily

### **Return Value**

YES if the text storage fixes attributes lazily, NO otherwise.

### **Discussion**

By default, custom NSTextStorage subclasses are not lazy, but the provided concrete subclass is lazy by default.

### **Availability**

Available in Mac OS X v10.0 and later.

### **Declared In**

NSTextStorage.h

## **font**

Returns the receiver's font.

- (NSFont \*)font

### **Return Value**

The receiver's font.

### **Discussion**

In Mac OS X v10.5 and later, if the font has not been set, this method and font-related scripting commands assume Helvetica 12.

### **Availability**

Available in Mac OS X v10.0 and later.

### **Declared In**

NSTextStorageScripting.h

## **foregroundColor**

Returns the text storage's foreground color.

- (NSColor \*)foregroundColor

### **Return Value**

The text storage's foreground color.

### **Availability**

Available in Mac OS X v10.0 and later.

### **Declared In**

NSTextStorageScripting.h

## invalidateAttributesInRange:

Invalidates attributes in the specified range.

```
- (void)invalidateAttributesInRange:(NSRange)range
```

### Parameters

*range*

The range of characters whose attributes should be invalidated.

### Discussion

Called from [processEditing](#) (page 14) to invalidate attributes when the text storage changes. If the receiver is not lazy, this method simply calls `fixAttributesInRange:`. If lazy attribute fixing is in effect, this method instead records the range needing fixing.

### Availability

Available in Mac OS X v10.0 and later.

### See Also

- [ensureAttributesAreFixedInRange:](#) (page 11)
- [fixesAttributesLazily](#) (page 12)

### Declared In

NSTextStorage.h

## layoutManagers

Returns the receiver's layout managers.

```
- (NSArray *)layoutManagers
```

### Return Value

The receiver's layout managers.

### Availability

Available in Mac OS X v10.0 and later.

### See Also

- [addLayoutManager:](#) (page 7)
- [removeLayoutManager:](#) (page 14)

### Related Sample Code

Quartz Composer WWDC 2005 TextEdit

### Declared In

NSTextStorage.h

## paragraphs

Returns an array of the text storage's paragraphs.

```
- (NSArray *)paragraphs
```

**Return Value**

An array of the text storage's paragraphs.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

`NSTextStorageScripting.h`

**processEditing**

Cleans up changes made to the receiver and notifies its delegate and layout managers of changes.

- (void)processEditing

**Discussion**

This method is automatically invoked in response to an [edited:range:changeInLength:](#) (page 9) message. You should never need to invoke it directly.

This method begins by posting an [NSTextStorageWillProcessEditingNotification](#) (page 18) to the default notification center (which results in the delegate receiving a `textStorageWillProcessEditing:` message). After this, it posts an [NSTextStorageDidProcessEditingNotification](#) (page 18) to the default notification center (which results in the delegate receiving a `textStorageDidProcessEditing:` message). Finally, it sends a `textStorage:edited:range:changeInLength:invalidatedRange:` message to each of the receiver's layout managers using the argument values provided.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

`NSTextStorage.h`

**removeLayoutManager:**

Removes a layout manager from the receiver's set of layout managers.

- (void)removeLayoutManager:(NSLayoutManager \*)*aLayoutManager*

**Parameters**

*aLayoutManager*

The layout manager to remove.

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [addLayoutManager:](#) (page 7)
- [layoutManagers](#) (page 13)

**Related Sample Code**

Quartz Composer WWDC 2005 TextEdit

QuickLookSketch

Sketch+Accessibility

Sketch-112

**Declared In**

NSTextStorage.h

**setAttributeRuns:**

Sets the receiver's attribute runs.

```
- (void)setAttributeRuns:(NSArray *)attributeRuns
```

**Parameters**

*attributeRuns*

The array of attribute runs to set.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSTextStorageScripting.h

**setCharacters:**

Sets the text storage's text.

```
- (void)setCharacters:(NSArray *)characters
```

**Parameters**

*characters*

The characters to set as the text of the text storage.

**Special Considerations**

Do not use this method if you are not dealing directly with scriptability. Use `NSMutableAttributedString`'s `mutableString` method to return a string object that will be tracked by the corresponding attributed string for modifications.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSTextStorageScripting.h

**setDelegate:**

Sets the receiver's delegate.

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

**Parameters**

*anObject*

The new delegate.

### Availability

Available in Mac OS X v10.0 and later.

### See Also

- [delegate](#) (page 9)

### Declared In

NSTextStorage.h

## setFont:

Sets the text storage's font.

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

### Parameters

*font*

The new font.

### Availability

Available in Mac OS X v10.0 and later.

### Related Sample Code

DockTile

SpeedometerView

WebKitPluginStarter

WebKitPluginWithJavaScript

### Declared In

NSTextStorageScripting.h

## setForegroundColor:

Sets the text storage's foreground color.

```
- (void)setForegroundColor:(NSColor *)color
```

### Parameters

*color*

The new foreground color.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

NSTextStorageScripting.h

## setParagraphs:

Sets the text storage's paragraphs.

```
- (void)setParagraphs:(NSArray *)paragraphs
```

### Parameters

*paragraphs*

An array of strings to set as the text storage's paragraphs.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

NSTextStorageScripting.h

## setWords:

Sets the text storage's words.

- (void)setWords:(NSArray \*)words

### Parameters

*words*

An array of strings to set as the text storage's words.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

NSTextStorageScripting.h

## words

Returns an array of the text storage's words.

- (NSArray \*)words

### Return Value

An array of the text storage's words.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

NSTextStorageScripting.h

## Constants

### Change notifications

These constants are used in [edited:range:changeInLength:](#) (page 9).

```
enum {
    NSTextStorageEditedAttributes = 1,
    NSTextStorageEditedCharacters = 2
};
```

**Constants**

**NSTextStorageEditedAttributes**  
Attributes were added, removed, or changed.  
Available in Mac OS X v10.0 and later.  
Declared in `NSTextStorage.h`.

**NSTextStorageEditedCharacters**  
Characters were added, removed, or replaced.  
Available in Mac OS X v10.0 and later.  
Declared in `NSTextStorage.h`.

**Discussion**

These values are also OR'ed together in notifications to inform instances of `NSLayoutManager` was changed—see `textStorage:edited:range:changeInLength:invalidatedRange:.`

**Declared In**

`NSTextStorage.h`

## Notifications

### **NSTextStorageDidProcessEditingNotification**

Posted after a text storage finishes processing edits in [processEditing](#) (page 14).

Observers other than the delegate shouldn't make further changes to the text storage. The notification object is the text storage object that processed the edits. This notification does not contain a *userInfo* dictionary.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

`NSTextStorage.h`

### **NSTextStorageWillProcessEditingNotification**

Posted before a text storage finishes processing edits in [processEditing](#) (page 14).

Observers other than the delegate shouldn't make further changes to the text storage. The notification object is the text storage object that is about to process the edits. This notification does not contain a *userInfo* dictionary.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

`NSTextStorage.h`

# Document Revision History

---

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

Date	Notes
2009-07-10	Updated for Mac OS X v10.6. Delegate methods moved to <a href="#">NSTextStorageDelegate Protocol Reference</a> . Noted that unset fonts default to Helvetica 12.
	Added discussion to font method stating that the default font returned is Helvetica 12.
2006-05-23	First publication of this content as a separate document.
	First publication of this content as a separate document.

## REVISION HISTORY

### Document Revision History

# Index

---

## A

---

`addLayoutManager`: [instance method 7](#)  
`attributeRuns` [instance method 8](#)

## C

---

**Change notifications** [17](#)  
`changeInLength` [instance method 8](#)  
`characters` [instance method 9](#)

## D

---

`delegate` [instance method 9](#)

## E

---

`edited:range:changeInLength:` [instance method 9](#)  
`editedMask` [instance method 10](#)  
`editedRange` [instance method 11](#)  
`ensureAttributesAreFixedInRange:` [instance method 11](#)

## F

---

`fixesAttributesLazily` [instance method 12](#)  
`font` [instance method 12](#)  
`foregroundColor` [instance method 12](#)

## I

---

`invalidateAttributesInRange:` [instance method 13](#)

## L

---

`layoutManagers` [instance method 13](#)

## N

---

`NSTextStorageDidProcessEditingNotification` [notification 18](#)  
`NSTextStorageEditedAttributes` [constant 18](#)  
`NSTextStorageEditedCharacters` [constant 18](#)  
`NSTextStorageWillProcessEditingNotification` [notification 18](#)

## P

---

`paragraphs` [instance method 13](#)  
`processEditing` [instance method 14](#)

## R

---

`removeLayoutManager:` [instance method 14](#)

## S

---

`setAttributeRuns:` [instance method 15](#)  
`setCharacters:` [instance method 15](#)  
`setDelegate:` [instance method 15](#)  
`setFont:` [instance method 16](#)  
`setForegroundColor:` [instance method 16](#)  
`setParagraphs:` [instance method 16](#)  
`setWords:` [instance method 17](#)

## W

---

words [instance method](#) [17](#)