NSAttributedString Application Kit Additions Reference

Cocoa > Data Management

2007-12-04

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

Simultaneously published in the United States and Canada.

Contents

NSAttributedString Application Kit Additions Reference

Inherits from	NSObject
Framework	/System/Library/Frameworks/AppKit.framework
Companion guide	Attributed Strings Programming Guide
Declared in	NSAttributedString.h NSStringDrawing.h NSTextAttachment.h

Overview

The Application Kit extends Foundation's NSAttributedString class by adding support for RTF (with or without attachments), graphics attributes (including font and ruler attributes), methods for drawing attributed strings, and methods for calculating significant linguistic units.

Tasks

Creating an NSAttributedString

`+ attributedStringWithAttachment:` (page 8)
Creates an attributed string with an attachment.

`- initWithData:options:documentAttributes:error:` (page 14)
Initializes and returns a new `NSAttributedString` object from the data contained in the given `NSData` object.

`- initWithDocFormat:documentAttributes:` (page 14)
Initializes and returns a new `NSAttributedString` object from Microsoft Word format data contained in the given `NSData` object.

`- initWithHTML:documentAttributes:` (page 15)
Initializes and returns a new `NSAttributedString` object from HTML contained in the given data object.

`- initWithHTML:baseURL:documentAttributes:` (page 15)
Initializes and returns a new `NSAttributedString` object from the HTML contained in the given object and base URL.

`- initWithHTML:options:documentAttributes:` (page 16)
Initializes and returns a new `NSAttributedString` object from HTML contained in the given data object.

- `initWithPath:documentAttributes:` (page 16)
  Initializes a new `NSAttributedString` object from RTF or RTFD data contained in the file at the given path.
- `initWithRTF:documentAttributes:` (page 16)
  Initializes a new `NSAttributedString` object by decoding the stream of RTF commands and data contained in the given data object.
- `initWithRTFD:documentAttributes:` (page 17)
  Initializes a new `NSAttributedString` object by decoding the stream of RTFD commands and data contained in the given data object.
- `initWithRTFDFileWrapper:documentAttributes:` (page 17)
  Initializes a new `NSAttributedString` object from the given `NSFileWrapper` object containing an RTFD document.
- `initWithURL:documentAttributes:` (page 18)
  Initializes a new `NSAttributedString` object from the data at the given URL.
- `initWithURL:options:documentAttributes:error:` (page 18)
  Initializes a new `NSAttributedString` object from the contents of the given URL.

Retrieving Font Attribute Information

- `containsAttachments` (page 10)
  Returns `YES` if the receiver contains any attachment attributes, `NO` otherwise.
- `fontAttributesInRange:` (page 13)
  Returns the font attributes in effect for the character at the given location.
- `rulerAttributesInRange:` (page 23)
  Returns the ruler (paragraph) attributes in effect for the characters within the given range.

Calculating Linguistic Units

- `URLAtIndex:effectiveRange:` (page 24)
  Returns a URL, either from a link attribute or from text at the given location that appears to be a URL string, for use in automatic link detection.
- `doubleClickAtIndex:` (page 11)
  Returns the range of characters that form a word (or other linguistic unit) surrounding the given index, taking language characteristics into account.
- `lineBreakBeforeIndex:withinRange:` (page 19)
  Returns the index of the closest character before the given index, and within the given range, that can be placed on a new line when laying out text.
- `lineBreakByHyphenatingBeforeIndex:withinRange:` (page 19)
  Returns the index of the closest character before the given index, and within the given range, that can be placed on a new line by hyphenating.
- `nextWordFromIndex:forward:` (page 20)
  Returns the index of the first character of the word after or before the given index.

Calculating Ranges

- `itemNumberInTextList:atIndex:` (page 18)
  Returns the range of the item at the given index within the given list.
- `rangeOfTextBlock:atIndex:` (page 20)
  Returns the range of the individual text block that contains the given location.
- `rangeOfTextList:atIndex:` (page 20)
  Returns the range of the given text list that contains the given location.
- `rangeOfTextTable:atIndex:` (page 21)
  Returns the range of the given text table that contains the given location

Generating Data

- `dataFromRange:documentAttributes:error:` (page 10)
  Returns an `NSData` object that contains a text stream corresponding to the characters and attributes within the given range.
- `fileWrapperFromRange:documentAttributes:error:` (page 13)
  Returns an `NSFileWrapper` object that contains a text stream corresponding to the characters and attributes within the given range.
- `docFormatFromRange:documentAttributes:` (page 11)
  Returns an `NSData` object that contains a Microsoft Word–format stream corresponding to the characters and attributes within the specified range.
- `RTFFromRange:documentAttributes:` (page 22)
  Returns an `NSData` object that contains an RTF stream corresponding to the characters and attributes within the given range, omitting all attachment attributes.
- `RTFDFromRange:documentAttributes:` (page 22)
  Returns an NSData object that contains an RTFD stream corresponding to the characters and attributes within *`aRange`*.
- `RTFDFileWrapperFromRange:documentAttributes:` (page 21)
  Returns an `NSFileWrapper` object that contains an RTFD document corresponding to the characters and attributes within the given range.

Drawing the String

- `drawAtPoint:` (page 11)
  Draws the receiver with its font and other display attributes at the given point in the currently focused `NSView`.
- `drawInRect:` (page 12)
  Draws the receiver with its font and other display attributes within the given rectangle in the currently focused `NSView`, clipping the text layout to this rectangle.
- `drawWithRect:options:` (page 12)
  Draws the receiver with the specified options, within the given rectangle in the current graphics context.
- `size` (page 23)
  Returns the bounding box of the marks that the receiver draws.

Getting the Bounding Rectangle of Rendered Strings

`- boundingRectWithSize:options:` (page 9)
Calculates and returns bounding rectangle for the receiver drawn using the options specified, within the given rectangle in the current graphics context.

Testing String Data Sources

`+ textTypes` (page 9)
Returns an array of UTI strings identifying the file types supported by the receiver, either directly or through a user-installed filter service.

`+ textUnfilteredTypes` (page 9)
Returns an array of UTI strings identifying the file types supported directly by the receiver.

Deprecated Methods

`+ textFileTypes` (page 41) Deprecated in Mac OS X v10.5
Returns an array of strings representing those file types that can be loaded as text. (Deprecated. Use `textTypes` (page 9) instead.)

`+ textPasteboardTypes` (page 41) Deprecated in Mac OS X v10.5
Returns an array of pasteboard types that can be loaded as text. (Deprecated. Use `textTypes` (page 9) instead.)

`+ textUnfilteredFileTypes` (page 42) Deprecated in Mac OS X v10.5
Returns an array of strings representing those file types that can be loaded as a text. (Deprecated. Use `textUnfilteredTypes` (page 9) instead.)

`+ textUnfilteredPasteboardTypes` (page 42) Deprecated in Mac OS X v10.5
Returns an array of pasteboard types that can be loaded as text. (Deprecated. Use `textUnfilteredTypes` (page 9) instead.)

Class Methods

attributedStringWithAttachment:

Creates an attributed string with an attachment.

```
+ (NSAttributedString *)attributedStringWithAttachment:(NSTextAttachment *)attachment
```

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

Related Sample Code
CoreRecipes

Declared In
`NSTextAttachment.h`

textTypes

Returns an array of UTI strings identifying the file types supported by the receiver, either directly or through a user-installed filter service.

```
+ (NSArray *)textTypes
```

Return Value
An array of `NSString` objects, each of which contains a UTI identifying a supported file type.

Discussion
The returned list includes UTIs all file types supported by the receiver plus those that can be opened by the receiver after being converted by a user-installed filter service. You can use the returned UTI strings with any method that supports UTIs.

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

Declared In
`NSAttributedString.h`

textUnfilteredTypes

Returns an array of UTI strings identifying the file types supported directly by the receiver.

```
+ (NSArray *)textUnfilteredTypes
```

Return Value
An array of `NSString` objects, each of which contains a UTI identifying a supported file type.

Discussion
The returned list includes UTI strings only for those file types that are supported directly by the receiver. It does not include types that are supported through user-installed filter services. You can use the returned UTI strings with any method that supports UTIs.

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

Declared In
`NSAttributedString.h`

Instance Methods

boundingRectWithSize:options:

Calculates and returns bounding rectangle for the receiver drawn using the options specified, within the given rectangle in the current graphics context.

```
- (NSRect)boundingRectWithSize:(NSSize)size options:(NSStringDrawingOptions)options
```

Discussion
The origin of the rectangle returned from this method is the first glyph origin.

The values of `NSStringDrawingOptions` are listed in the "Constants" section of NSString Additions.

Availability
Available in Mac OS X v10.4 and later.

See Also
`- drawInRect:` (page 12)

Declared In
`NSStringDrawing.h`

containsAttachments

Returns `YES` if the receiver contains any attachment attributes, `NO` otherwise.

```
- (BOOL)containsAttachments
```

Discussion
This method checks only for attachment attributes, not for `NSAttachmentCharacter`.

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

Declared In
`NSAttributedString.h`

dataFromRange:documentAttributes:error:

Returns an `NSData` object that contains a text stream corresponding to the characters and attributes within the given range.

```
- (NSData *)dataFromRange:(NSRange)range documentAttributes:(NSDictionary *)dict
    error:(NSError **)error
```

Discussion
Requires a document attributes dictionary *`dict`* specifying at least the `NSDocumentTypeDocumentAttribute` to determine the format to write. Raises an `NSRangeException` if any part of *`range`* lies beyond the end of the receiver's characters. If unsuccessful, returns `nil` after setting *`error`* to point to an `NSError` object that encapsulates the reason why the object could not be created.

Availability
Available in Mac OS X v10.4 and later.

See Also
`- fileWrapperFromRange:documentAttributes:error:` (page 13)

Declared In
`NSAttributedString.h`

docFormatFromRange:documentAttributes:

Returns an `NSData` object that contains a Microsoft Word–format stream corresponding to the characters and attributes within the specified range.

```
- (NSData *)docFormatFromRange:(NSRange)range documentAttributes:(NSDictionary
    *)dict
```

Discussion
The range is passed in the *range* parameter. Also writes the document-level attributes in *dict*, as explained in "Constants" (page 24). If there are no document-level attributes, *dict* can be `nil`. Raises an `NSRangeException` if any part of *range* lies beyond the end of the receiver's characters.

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

Declared In
`NSAttributedString.h`

doubleClickAtIndex:

Returns the range of characters that form a word (or other linguistic unit) surrounding the given index, taking language characteristics into account.

```
- (NSRange)doubleClickAtIndex:(NSUInteger)index
```

Discussion
Raises an `NSRangeException` if *index* lies beyond the end of the receiver's characters.

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

See Also
`- nextWordFromIndex:forward:` (page 20)

Declared In
`NSAttributedString.h`

drawAtPoint:

Draws the receiver with its font and other display attributes at the given point in the currently focused `NSView`.

```
- (void)drawAtPoint:(NSPoint)point
```

Discussion
The width (height for vertical layout) of the rendering area is unlimited, unlike `drawInRect:` (page 12), which uses a bounding rectangle. As a result, this method renders the text in a single line.

Don't invoke this method while no `NSView` is focused.

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

See Also
- `lockFocus` (NSView)
- `size` (page 23)
- `drawInRect:` (page 12)

Related Sample Code
Aperture Edit Plugin - Borders & Titles

Declared In
`NSStringDrawing.h`

drawInRect:

Draws the receiver with its font and other display attributes within the given rectangle in the currently focused `NSView`, clipping the text layout to this rectangle.

```
- (void)drawInRect:(NSRect)rect
```

Discussion
Text is drawn within *rect* according to its line sweep direction; for example, Arabic text will begin at the right edge and potentially be clipped on the left.

The *rect* parameter determines how many glyphs are typeset within the width of a line, but it's possible for a portion of a glyph to appear outside the area of *rect* if the image bounding box of the particular glyph exceeds its typographic bounding box.

If the focus view is flipped, the text origin is set at the upper-left corner of the drawing bounding box; otherwise the origin is set at the lower-left corner. For text rendering, whether the view coordinates are flipped or not doesn't affect the flow of line layout, which goes from top to bottom. However, it affects the interpretation of the text origin. So, for example, if the *rect* argument is `{0.0, 0.0, 100.0, 100.0}`, the text origin is {0.0, 0.0} when the view coordinates are flipped and {0.0, 100.0} when not.

Don't invoke this method while no `NSView` is focused.

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

See Also
- `lockFocus` (NSView)
- `drawAtPoint:` (page 11)

Related Sample Code
IBFragmentView

Declared In
`NSStringDrawing.h`

drawWithRect:options:

Draws the receiver with the specified options, within the given rectangle in the current graphics context.

```
- (void)drawWithRect:(NSRect)rect options:(NSStringDrawingOptions)options
```

Discussion
The *rect* argument's origin field specifies the rendering origin. The point is interpreted as the baseline origin by default. With `NSStringDrawingUsesLineFragmentOrigin`, it is interpreted as the upper left corner of the line fragment rect. The size field specifies the text container size. The width part of the size field specifies the maximum line fragment width if larger than `0.0`. The height defines the maximum size that can be occupied with text if larger than `0.0` and `NSStringDrawingUsesLineFragmentOrigin` is specified. If `NSStringDrawingUsesLineFragmentOrigin` is not specified, height is ignored and considered to be single-line rendering (`NSLineBreakByWordWrapping` and `NSLineBreakByCharWrapping` are treated as `NSLineBreakByClipping`).

The values of `NSStringDrawingOptions` are listed in the "Constants" section of NSString Additions.

You should only invoke this method when there is a current graphics context.

Availability
Available in Mac OS X v10.4 and later.

See Also
- `drawAtPoint:` (page 11) (`NSView`)
- `lockFocus`

Declared In
`NSStringDrawing.h`

fileWrapperFromRange:documentAttributes:error:

Returns an `NSFileWrapper` object that contains a text stream corresponding to the characters and attributes within the given range.

```
- (NSFileWrapper *)fileWrapperFromRange:(NSRange)range
    documentAttributes:(NSDictionary *)dict error:(NSError **)error
```

Discussion
Requires a document attributes dictionary *dict* specifying at least the `NSDocumentTypeDocumentAttribute` to determine the format to write. Raises an `NSRangeException` if any part of *range* lies beyond the end of the receiver's characters. Returns a directory file wrapper for those document types for which it is appropriate; otherwise a regular file wrapper. If unsuccessful, returns `nil` after setting *error* to point to an `NSError` object that encapsulates the reason why the object could not be created.

Availability
Available in Mac OS X v10.4 and later.

See Also
- `dataFromRange:documentAttributes:error:` (page 10)

Declared In
`NSAttributedString.h`

fontAttributesInRange:

Returns the font attributes in effect for the character at the given location.

```
- (NSDictionary *)fontAttributesInRange:(NSRange)aRange
```

Discussion
Returns the font attributes in effect for the character at *aRange.location*. Font attributes are all those listed in "Standard Attributes" (page 24), except NSLinkAttributeName, NSParagraphStyleAttributeName, and NSAttachmentAttributeName. Use this method to obtain font attributes that are to be copied or pasted with "copy font" operations. Raises an NSRangeException if any part of *aRange* lies beyond the end of the receiver's characters.

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

See Also
- rulerAttributesInRange: (page 23)

Declared In
NSAttributedString.h

initWithData:options:documentAttributes:error:

Initializes and returns a new NSAttributedString object from the data contained in the given NSData object.

```
- (id)initWithData:(NSData *)data options:(NSDictionary *)options
    documentAttributes:(NSDictionary **)dict error:(NSError **)error
```

Discussion
The *options* dictionary can contain the values described in "Option keys for importing documents" (page 37) to specify how the document should be loaded. If NSDocumentTypeDocumentOption is specified, the document is treated as being in the specified format. If NSDocumentTypeDocumentOption is not specified, the method examines the document and loads it using whatever format it seems to contain. Also returns by reference in *dict* a dictionary containing document-level attributes described in "Constants" (page 24). The *dict* parameter may be nil, in which case no document attributes are returned. Returns nil if *data* can't be decoded, after setting *error* to point to an NSError that encapsulates the reason why the attributed string object could not be created.

Availability
Available in Mac OS X v10.4 and later.

Related Sample Code
CoreRecipes

Declared In
NSAttributedString.h

initWithDocFormat:documentAttributes:

Initializes and returns a new NSAttributedString object from Microsoft Word format data contained in the given NSData object.

```
- (id)initWithDocFormat:(NSData *)data documentAttributes:(NSDictionary
    **)docAttributes
```

Discussion
Also returns by reference in *docAttributes* a dictionary containing document-level attributes described in "Constants" (page 24). *docAttributes* may be NULL, in which case no document attributes are returned. Returns nil if *data* can't be decoded.

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

Declared In
NSAttributedString.h

initWithHTML:baseURL:documentAttributes:

Initializes and returns a new NSAttributedString object from the HTML contained in the given object and base URL.

```
- (id)initWithHTML:(NSData *)data baseURL:(NSURL *)aURL
    documentAttributes:(NSDictionary **)docAttributes
```

Discussion
Also returns by reference in *docAttributes* a dictionary containing document-level attributes described in "Constants" (page 24). *docAttributes* may be NULL, in which case no document attributes are returned. Returns an initialized object, or nil if the file at *aURL* can't be decoded.

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

Declared In
NSAttributedString.h

initWithHTML:documentAttributes:

Initializes and returns a new NSAttributedString object from HTML contained in the given data object.

```
- (id)initWithHTML:(NSData *)data documentAttributes:(NSDictionary **)docAttributes
```

Discussion
Also returns by reference in *docAttributes* a dictionary containing document-level attributes described in "Constants" (page 24). *docAttributes* may be NULL, in which case no document attributes are returned. Returns nil if *data* can't be decoded.

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

Related Sample Code
ObjectPath

Declared In
NSAttributedString.h

initWithHTML:options:documentAttributes:

Initializes and returns a new `NSAttributedString` object from HTML contained in the given data object.

```
- (id)initWithHTML:(NSData *)data options:(NSDictionary *)options
    documentAttributes:(NSDictionary **)dict
```

Discussion
The *options* dictionary can contain the values described in "Option keys for importing documents" (page 37).

Also returns by reference in *docAttributes* a dictionary containing document-level attributes described in "Constants" (page 24). *docAttributes* may be `NULL`, in which case no document attributes are returned. Returns `nil` if *data* can't be decoded.

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

Declared In
`NSAttributedString.h`

initWithPath:documentAttributes:

Initializes a new `NSAttributedString` object from RTF or RTFD data contained in the file at the given path.

```
- (id)initWithPath:(NSString *)path documentAttributes:(NSDictionary **)docAttributes
```

Discussion
The contents of *path* will be examined to best load the file in whatever format it's in. Filter services can be used to convert the file into a format recognized by Cocoa. Also returns by reference in *docAttributes* a dictionary containing document-level attributes described in "Constants" (page 24). *docAttributes* may be `NULL`, in which case no document attributes are returned. Returns an initialized object, or `nil` if the file at *path* can't be decoded.

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

Related Sample Code
iSpend

VertexPerformanceTest

Declared In
`NSAttributedString.h`

initWithRTF:documentAttributes:

Initializes a new `NSAttributedString` object by decoding the stream of RTF commands and data contained in the given data object.

```
- (id)initWithRTF:(NSData *)rtfData documentAttributes:(NSDictionary **)docAttributes
```

Discussion
Also returns by reference in *docAttributes* a dictionary containing document-level attributes described in "Constants" (page 24). *docAttributes* may be NULL, in which case no document attributes are returned. Returns an initialized object, or nil if *rtfData* can't be decoded.

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

Related Sample Code
CoreRecipes
iSpend
Spotlight

Declared In
NSAttributedString.h

initWithRTFD:documentAttributes:

Initializes a new NSAttributedString object by decoding the stream of RTFD commands and data contained in the given data object.

```
- (id)initWithRTFD:(NSData *)rtfdData documentAttributes:(NSDictionary
    **)docAttributes
```

Discussion
Also returns by reference in *docAttributes* a dictionary containing document-level attributes described in "Constants" (page 24). *docAttributes* may be NULL, in which case no document attributes are returned. Returns an initialized object, or nil if *rtfData* can't be decoded.

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

Declared In
NSAttributedString.h

initWithRTFDFileWrapper:documentAttributes:

Initializes a new NSAttributedString object from the given NSFileWrapper object containing an RTFD document.

```
- (id)initWithRTFDFileWrapper:(NSFileWrapper *)wrapper
    documentAttributes:(NSDictionary **)docAttributes
```

Discussion
Also returns by reference in *docAttributes* a dictionary containing document-level attributes described in "Constants" (page 24). *docAttributes* may be NULL, in which case no document attributes are returned. Returns an initialized object, or nil if *wrapper* can't be interpreted as an RTFD document.

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

Declared In
NSAttributedString.h

initWithURL:documentAttributes:

Initializes a new `NSAttributedString` object from the data at the given URL.

```
- (id)initWithURL:(NSURL *)aURL documentAttributes:(NSDictionary **)docAttributes
```

Discussion
The contents of *aURL* are examined to best load the file in whatever format it's in. Filter services can be used to convert the file into a format recognized by Cocoa. Also returns by reference in *docAttributes* a dictionary containing document-level attributes described in "Constants" (page 24). *docAttributes* may be `NULL`, in which case no document attributes are returned. Returns an initialized object, or `nil` if the file at *path* can't be decoded.

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

Declared In
`NSAttributedString.h`

initWithURL:options:documentAttributes:error:

Initializes a new `NSAttributedString` object from the contents of the given URL.

```
- (id)initWithURL:(NSURL *)url options:(NSDictionary *)options
    documentAttributes:(NSDictionary **)dict error:(NSError **)error
```

Discussion
Filter services can be used to convert the file into a format recognized by Cocoa. The *options* dictionary specifies how the document should be loaded and can contain the values described in "Option keys for importing documents" (page 37).

If `NSDocumentTypeDocumentOption` is specified, the document is treated as being in the specified format. If `NSDocumentTypeDocumentOption` is not specified, the method examines the document and loads it using whatever format it seems to contain.

Also returns by reference in *dict* a dictionary containing document-level attributes described in "Constants" (page 24). The *dict* parameter may be nil, in which case no document attributes are returned. Returns an initialized object, or `nil` if the file at *url* can't be decoded, after setting *error* to point to an `NSError` object that encapsulates the reason why the attributed string object could not be created.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`NSAttributedString.h`

itemNumberInTextList:atIndex:

Returns the range of the item at the given index within the given list.

```
- (NSInteger)itemNumberInTextList:(NSTextList *)list atIndex:(NSUInteger)location
```

Availability
Available in Mac OS X v10.4 and later.

See Also
- `rangeOfTextBlock:atIndex:` (page 20)
- `rangeOfTextList:atIndex:` (page 20)
- `rangeOfTextTable:atIndex:` (page 21)

Declared In
`NSAttributedString.h`

lineBreakBeforeIndex:withinRange:

Returns the index of the closest character before the given index, and within the given range, that can be placed on a new line when laying out text.

```
- (NSUInteger)lineBreakBeforeIndex:(NSUInteger)index withinRange:(NSRange)aRange
```

Discussion
In other words, finds the appropriate line break when the character at *index* won't fit on the same line as the character at the beginning of *aRange*. Returns `NSNotFound` if no line break is possible before *index*.Raises an `NSRangeException` if *index* or any part of *aRange* lies beyond the end of the receiver's characters.

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

See Also
- `lineBreakByHyphenatingBeforeIndex:withinRange:` (page 19)

Declared In
`NSAttributedString.h`

lineBreakByHyphenatingBeforeIndex:withinRange:

Returns the index of the closest character before the given index, and within the given range, that can be placed on a new line by hyphenating.

```
- (NSUInteger)lineBreakByHyphenatingBeforeIndex:(NSUInteger)location
    withinRange:(NSRange)aRange
```

Discussion
In other words, during text layout, finds the appropriate line break by hyphenation (the character index at which the hyphen glyph should be inserted) when the character at *index* won't fit on the same line as the character at the beginning of *aRange*. Returns `NSNotFound` if no line break by hyphenation is possible before *index*.Raises an `NSRangeException` if *index* or any part of *aRange* lies beyond the end of the receiver's characters.

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

See Also
- `lineBreakBeforeIndex:withinRange:` (page 19)

Declared In
`NSAttributedString.h`

nextWordFromIndex:forward:

Returns the index of the first character of the word after or before the given index.

```
- (NSUInteger)nextWordFromIndex:(NSUInteger)index forward:(BOOL)flag
```

Discussion
If *flag* is YES, this is the first character after *index* that begins a word; if *flag* is NO, it's the first character before *index* that begins a word, whether *index* is located within a word or not. If *index* lies at either end of the string and the search direction would progress past that end, it's returned unchanged. This method is intended for moving the insertion point during editing, not for linguistic analysis or parsing of text.Raises an NSRangeException if *index* lies beyond the end of the receiver's characters.

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

See Also
- lineBreakBeforeIndex:withinRange: (page 19)

Declared In
NSAttributedString.h

rangeOfTextBlock:atIndex:

Returns the range of the individual text block that contains the given location.

```
- (NSRange)rangeOfTextBlock:(NSTextBlock *)block atIndex:(NSUInteger)location
```

Discussion
The individual text is given by *block* and contains *location*.

Availability
Available in Mac OS X v10.4 and later.

See Also
- itemNumberInTextList:atIndex: (page 18)
- rangeOfTextList:atIndex: (page 20)
- rangeOfTextTable:atIndex: (page 21)

Related Sample Code
iSpend

Declared In
NSAttributedString.h

rangeOfTextList:atIndex:

Returns the range of the given text list that contains the given location.

```
- (NSRange)rangeOfTextList:(NSTextList *)list atIndex:(NSUInteger)location
```

Discussion
Returns the range of the *list* that contains *location*.

Availability
Available in Mac OS X v10.4 and later.

See Also
- `itemNumberInTextList:atIndex:` (page 18)
- `rangeOfTextBlock:atIndex:` (page 20)
- `rangeOfTextTable:atIndex:` (page 21)

Declared In
`NSAttributedString.h`

rangeOfTextTable:atIndex:

Returns the range of the given text table that contains the given location

```
- (NSRange)rangeOfTextTable:(NSTextTable *)table atIndex:(NSUInteger)location
```

Discussion
Returns the range of the text *table* that contains *location*.

Availability
Available in Mac OS X v10.4 and later.

See Also
- `itemNumberInTextList:atIndex:` (page 18)
- `rangeOfTextList:atIndex:` (page 20)
- `rangeOfTextBlock:atIndex:` (page 20)

Related Sample Code
iSpend

Declared In
`NSAttributedString.h`

RTFDFileWrapperFromRange:documentAttributes:

Returns an `NSFileWrapper` object that contains an RTFD document corresponding to the characters and attributes within the given range.

```
- (NSFileWrapper *)RTFDFileWrapperFromRange:(NSRange)aRange
    documentAttributes:(NSDictionary *)docAttributes
```

Discussion
The file wrapper also includes the document-level attributes in *docAttributes*, as explained in "RTF Files and Attributed Strings". If there are no document-level attributes, *docAttributes* can be `nil`. Raises an `NSRangeException` if any part of *aRange* lies beyond the end of the receiver's characters.You can save the file wrapper using the `NSFileWrapper` method `writeToFile:atomically:updateFilenames:`.

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

See Also
- `RTFFromRange:documentAttributes:` (page 22)
- `RTFDFromRange:documentAttributes:` (page 22)

Declared In
`NSAttributedString.h`

RTFDFromRange:documentAttributes:

Returns an NSData object that contains an RTFD stream corresponding to the characters and attributes within *aRange*.

```
- (NSData *)RTFDFromRange:(NSRange)aRange documentAttributes:(NSDictionary
    *)docAttributes
```

Discussion
Also writes the document-level attributes in *docAttributes*, as explained in "RTF Files and Attributed Strings". If there are no document-level attributes, *docAttributes* can be `nil`. Raises an `NSRangeException` if any part of *aRange* lies beyond the end of the receiver's characters.

When writing data to the pasteboard, you can use the `NSData` object as the first argument to the `NSPasteboard` method `setData:forType:`, with a second argument of `NSRTFDPboardType`.

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

See Also
- `RTFFromRange:documentAttributes:` (page 22)
- `RTFDFileWrapperFromRange:documentAttributes:` (page 21)

Declared In
`NSAttributedString.h`

RTFFromRange:documentAttributes:

Returns an `NSData` object that contains an RTF stream corresponding to the characters and attributes within the given range, omitting all attachment attributes.

```
- (NSData *)RTFFromRange:(NSRange)aRange documentAttributes:(NSDictionary
    *)docAttributes
```

Discussion
Also writes the document-level attributes in *docAttributes*, as explained in "RTF Files and Attributed Strings". If there are no document-level attributes, *docAttributes* can be `nil`. Raises an `NSRangeException` if any part of *aRange* lies beyond the end of the receiver's characters.When writing data to the pasteboard, you can use the `NSData` object as the first argument to the `NSPasteboard` method `setData:forType:`, with a second argument of `NSRTFPboardType`. Although this method strips attachments, it leaves the attachment characters in the text itself. The `NSText` method `RTFFromRange:`, on the other hand, does strip attachment characters when extracting RTF.

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

See Also
- `- RTFDFromRange:documentAttributes:` (page 22)
- `- RTFDFileWrapperFromRange:documentAttributes:` (page 21)

Related Sample Code
VertexPerformanceTest

Declared In
`NSAttributedString.h`

rulerAttributesInRange:

Returns the ruler (paragraph) attributes in effect for the characters within the given range.

```
- (NSDictionary *)rulerAttributesInRange:(NSRange)aRange
```

Discussion
The only ruler attribute currently defined is that named by `NSParagraphStyleAttributeName`. Use this method to obtain attributes that are to be copied or pasted with "copy ruler" operations. Raises an `NSRangeException` if any part of *aRange* lies beyond the end of the receiver's characters.

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

See Also
- `- fontAttributesInRange:` (page 13)

Declared In
`NSAttributedString.h`

size

Returns the bounding box of the marks that the receiver draws.

```
- (NSSize)size
```

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

See Also
- `- drawAtPoint:` (page 11)
- `- drawInRect:` (page 12)

Related Sample Code
Aperture Edit Plugin - Borders & Titles
IBFragmentView

Declared In
`NSStringDrawing.h`

URLAtIndex:effectiveRange:

Returns a URL, either from a link attribute or from text at the given location that appears to be a URL string, for use in automatic link detection.

```
- (NSURL *)URLAtIndex:(NSUInteger)location
    effectiveRange:(NSRangePointer)effectiveRange
```

Parameters

location

The character index in the string at which the method checks for a link.

effectiveRange

The actual range covered by the link attribute or URL string, or of non-URL text if no apparent URL is found.

Return Value

The URL found at *location*.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSAttributedString.h

Constants

Standard Attributes

Attributed strings support the following standard attributes for text. If the key is not in the dictionary, then use the default values described below.

```
NSString *NSFontAttributeName;
NSString *NSParagraphStyleAttributeName;
NSString *NSForegroundColorAttributeName;
NSString *NSUnderlineStyleAttributeName;
NSString *NSSuperscriptAttributeName;
NSString *NSBackgroundColorAttributeName;
NSString *NSAttachmentAttributeName;
NSString *NSLigatureAttributeName;
NSString *NSBaselineOffsetAttributeName;
NSString *NSKernAttributeName;
NSString *NSLinkAttributeName;
NSString *NSStrokeWidthAttributeName;
NSString *NSStrokeColorAttributeName;
NSString *NSUnderlineColorAttributeName;
NSString *NSStrikethroughStyleAttributeName;
NSString *NSStrikethroughColorAttributeName;
NSString *NSShadowAttributeName;
NSString *NSObliquenessAttributeName;
NSString *NSExpansionAttributeName;
NSString *NSCursorAttributeName;
NSString *NSToolTipAttributeName;
NSString *NSMarkedClauseSegmentAttributeName;
```

Constants

`NSFontAttributeName`
`NSFont`

Default Helvetica 12-point

Available in Mac OS X v10.0 and later.

Declared in `NSAttributedString.h`.

`NSParagraphStyleAttributeName`
`NSParagraphStyle`

Default as returned by the `NSParagraphStyle` method `defaultParagraphStyle`

Available in Mac OS X v10.0 and later.

Declared in `NSAttributedString.h`.

`NSForegroundColorAttributeName`
`NSColor`

Default `blackColor`

Available in Mac OS X v10.0 and later.

Declared in `NSAttributedString.h`.

`NSUnderlineStyleAttributeName`
`NSNumber` containing integer

Default `0`, no underline. See "Underlining Patterns" (page 29), "Underlining Styles" (page 28), and "Underline Masks" (page 30) for mask values.

Available in Mac OS X v10.0 and later.

Declared in `NSAttributedString.h`.

`NSSuperscriptAttributeName`
`NSNumber` containing integer
Default `0`
Available in Mac OS X v10.0 and later.
Declared in `NSAttributedString.h`.

`NSBackgroundColorAttributeName`
`NSColor`
Default `nil`, no background
Available in Mac OS X v10.0 and later.
Declared in `NSAttributedString.h`.

`NSAttachmentAttributeName`
`NSTextAttachment`
Default `nil`, no attachment
Available in Mac OS X v10.0 and later.
Declared in `NSAttributedString.h`.

`NSLigatureAttributeName`
`NSNumber` containing integer
Default `1`, standard ligatures; `0`, no ligatures; `2`, all ligatures
Available in Mac OS X v10.0 and later.
Declared in `NSAttributedString.h`.

`NSBaselineOffsetAttributeName`
`NSNumber` containing floating point value, as points offset from baseline
Default `0.0`
Available in Mac OS X v10.0 and later.
Declared in `NSAttributedString.h`.

`NSKernAttributeName`
`NSNumber` containing floating point value, as points by which to modify default kerning
Default `nil`, use default kerning specified in font file; `0.0`, kerning off; non-zero, points by which to modify default kerning
Available in Mac OS X v10.0 and later.
Declared in `NSAttributedString.h`.

`NSLinkAttributeName`
`NSURL` (preferred) or `NSString`
Default `nil`, no link
Available in Mac OS X v10.0 and later.
Declared in `NSAttributedString.h`.

`NSStrokeWidthAttributeName`
`NSNumber` containing floating point value, as percent of font point size
Default `0`, no stroke; positive, stroke alone; negative, stroke and fill (a typical value for outlined text would be `3.0`)
Available in Mac OS X v10.3 and later.
Declared in `NSAttributedString.h`.

`NSStrokeColorAttributeName`
`NSColor`

Default `nil`, same as foreground color

Available in Mac OS X v10.3 and later.

Declared in `NSAttributedString.h`.

`NSUnderlineColorAttributeName`
`NSColor`

Default `nil`, same as foreground color

Available in Mac OS X v10.3 and later.

Declared in `NSAttributedString.h`.

`NSStrikethroughStyleAttributeName`
`NSNumber` containing integer

Default `0`, no strikethrough. See "Underlining Patterns" (page 29), "Underlining Styles" (page 28), and "Underline Masks" (page 30) for mask values.

Available in Mac OS X v10.3 and later.

Declared in `NSAttributedString.h`.

`NSStrikethroughColorAttributeName`
`NSColor`

Default `nil`, same as foreground color

Available in Mac OS X v10.3 and later.

Declared in `NSAttributedString.h`.

`NSShadowAttributeName`
`NSShadow`

Default `nil`, no shadow

Available in Mac OS X v10.3 and later.

Declared in `NSAttributedString.h`.

`NSObliquenessAttributeName`
`NSNumber` containing floating point value, as skew to be applied to glyphs

Default `0.0`, no skew

Available in Mac OS X v10.3 and later.

Declared in `NSAttributedString.h`.

`NSExpansionAttributeName`
`NSNumber` containing floating point value, as log of expansion factor to be applied to glyphs

Default `0.0`, no expansion

Available in Mac OS X v10.3 and later.

Declared in `NSAttributedString.h`.

`NSCursorAttributeName`
`NSCursor`

Default as returned by the `NSCursor` method `IBeamCursor`

Available in Mac OS X v10.3 and later.

Declared in `NSAttributedString.h`.

`NSToolTipAttributeName`
`NSString`
Default `nil`, no tooltip
Available in Mac OS X v10.3 and later.
Declared in `NSAttributedString.h`.

`NSMarkedClauseSegmentAttributeName`
`NSNumber` containing an integer, as an index in marked text indicating clause segments
Available in Mac OS X v10.5 and later.
Declared in `NSAttributedString.h`.

Declared In
`NSAttributedString.h`

Underlining Styles

These constants define underlining style values for `NSUnderlineStyleAttributeName` (page 25) and `NSStrikethroughStyleAttributeName` (page 27).

```
enum {
    NSUnderlineStyleNone   = 0x00,
    NSUnderlineStyleSingle   = 0x01,
    NSUnderlineStyleThick   = 0x02,
    NSUnderlineStyleDouble   = 0x09
};
```

Constants
`NSUnderlineStyleNone`
Do not draw an underline.
Available in Mac OS X v10.3 and later.
Declared in `NSAttributedString.h`.

`NSUnderlineStyleSingle`
Draw an underline consisting of a single line.
Available in Mac OS X v10.3 and later.
Declared in `NSAttributedString.h`.

`NSUnderlineStyleThick`
Draw an underline consisting of a thick line.
Available in Mac OS X v10.3 and later.
Declared in `NSAttributedString.h`.

`NSUnderlineStyleDouble`
Draw an underline consisting of a double line.
Available in Mac OS X v10.3 and later.
Declared in `NSAttributedString.h`.

Discussion
See also "`Underline Masks`" (page 30) and "`Underlining Patterns`" (page 29). The style, pattern, and optionally by-word mask are OR'd together to produce the value for `NSUnderlineStyleAttributeName` (page 25) and `NSStrikethroughStyleAttributeName` (page 27).

Declared In
NSAttributedString.h

Underlining Patterns

These constants define underlining pattern values for NSUnderlineStyleAttributeName (page 25) and NSStrikethroughStyleAttributeName (page 27).

```
enum {
    NSUnderlinePatternSolid   = 0x0000,
    NSUnderlinePatternDot   = 0x0100,
    NSUnderlinePatternDash   = 0x0200,
    NSUnderlinePatternDashDot   = 0x0300,
    NSUnderlinePatternDashDotDot  = 0x0400
};
```

Constants

NSUnderlinePatternSolid
Draw a solid underline.
Available in Mac OS X v10.3 and later.
Declared in NSAttributedString.h.

NSUnderlinePatternDot
Draw an underline using a pattern of dots.
Available in Mac OS X v10.3 and later.
Declared in NSAttributedString.h.

NSUnderlinePatternDash
Draw an underline using a pattern of dashes.
Available in Mac OS X v10.3 and later.
Declared in NSAttributedString.h.

NSUnderlinePatternDashDot
Draw an underline using a pattern of alternating dashes and dots.
Available in Mac OS X v10.3 and later.
Declared in NSAttributedString.h.

NSUnderlinePatternDashDotDot
Draw an underline using a pattern of a dash followed by two dots.
Available in Mac OS X v10.3 and later.
Declared in NSAttributedString.h.

Discussion
See also "Underline Masks" (page 30) and "Underlining Styles" (page 28). The style, pattern, and optionally by-word mask are OR'd together to produce the value for NSUnderlineStyleAttributeName (page 25) and NSStrikethroughStyleAttributeName (page 27).

The following constants previously used for underline style are deprecated in Mac OS X v10.3 and later:

NSNoUnderlineStyle
Available in Mac OS X v10.0 and later.
Declared in NSAttributedString.h.

`NSSingleUnderlineStyle`
Available in Mac OS X v10.0 and later.

Declared in `NSAttributedString.h`.

`NSUnderlineStrikethroughMask`

Declared In
`NSAttributedString.h`

Underline Masks

This constant defines the underlining style for `NSUnderlineStyleAttributeName` (page 25) and `NSStrikethroughStyleAttributeName` (page 27).

```
unsigned NSUnderlineByWordMask;
```

Constants
`NSUnderlineByWordMask`
Draw the underline only underneath words, not underneath whitespace.

Available in Mac OS X v10.0 and later.

Declared in `NSAttributedString.h`.

Discussion
Use this constant with the desired underline style to create the given effect. For example, to get a thick underline only underneath words, set `NSUnderlineStyleAttribute` to `(NSUnderlineStyleThick | NSUnderlineByWordMask)`. Also see "Underlining Styles" (page 28) and "Underlining Patterns" (page 29).

Declared In
`NSAttributedString.h`

Glyph Info Attribute

This object provides a means to override the standard glyph generation.

```
NSString *NSGlyphInfoAttributeName;
```

Constants
`NSGlyphInfoAttributeName`
The name of an `NSGlyphInfo` object.

`NSLayoutManager` assigns the glyph specified by this glyph info to the entire attribute range, provided that its contents match the specified base string, and that the specified glyph is available in the font specified by `NSFontAttributeName`.

Available in Mac OS X v10.2 and later.

Declared in `NSAttributedString.h`.

Declared In
`NSAttributedString.h`

Character Shape Attribute

The character shape feature type (`kCharacterShapeType`) is used when a single font contains different appearances for the same character shape, and these shapes are not traditionally treated as swashes. It is needed for languages such as Chinese that have both traditional and simplified character sets.

```
NSString *NSCharacterShapeAttributeName;
```

Constants

`NSCharacterShapeAttributeName`

An integer value. The value is interpreted as Apple Type Services `kCharacterShapeType selector + 1`.

The default value is 0 (disable). 1 is `kTraditionalCharactersSelector`, and so on. Refer to `<ATS/SFNTLayoutTypes.h>` and Font Features in *ATSUI Programming Guide* for additional information.

Available in Mac OS X v10.0 and later.

Declared in `NSAttributedString.h`.

Declared In

`NSAttributedString.h`

Document Types

The following values can be returned for the `@"DocumentType"` key in the document attributes dictionary.

```
NSString *NSPlainTextDocumentType;
NSString *NSRTFTextDocumentType;
NSString *NSRTFDTextDocumentType;
NSString *NSMacSimpleTextDocumentType;
NSString *NSHTMLTextDocumentType;
NSString *NSDocFormatTextDocumentType;
NSString *NSWordMLTextDocumentType;
NSString *NSOfficeOpenXMLTextDocumentType;
NSString *NSOpenDocumentTextDocumentType;
```

Constants

`NSPlainTextDocumentType`

Plain text document.

Available in Mac OS X v10.0 and later.

Declared in `NSAttributedString.h`.

`NSRTFTextDocumentType`

Rich text format document.

Available in Mac OS X v10.0 and later.

Declared in `NSAttributedString.h`.

`NSRTFDTextDocumentType`

Rich text format with attachments document.

Available in Mac OS X v10.0 and later.

Declared in `NSAttributedString.h`.

`NSMacSimpleTextDocumentType`
Macintosh SimpleText document.

Available in Mac OS X v10.0 and later.

Declared in `NSAttributedString.h`.

`NSHTMLTextDocumentType`
Hypertext Markup Language (HTML) document.

Available in Mac OS X v10.0 and later.

Declared in `NSAttributedString.h`.

`NSDocFormatTextDocumentType`
Microsoft Word document.

Available in Mac OS X v10.3 and later.

Declared in `NSAttributedString.h`.

`NSWordMLTextDocumentType`
Microsoft Word XML (WordML schema) document.

Available in Mac OS X v10.3 and later.

Declared in `NSAttributedString.h`.

`NSWebArchiveTextDocumentType`
Web Kit WebArchive document.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSOfficeOpenXMLTextDocumentType`
ECMA Office Open XML text document format.

Available in Mac OS X v10.5 and later.

Declared in `NSAttributedString.h`.

`NSOpenDocumentTextDocumentType`
OASIS Open Document text document format.

Available in Mac OS X v10.5 and later.

Declared in `NSAttributedString.h`.

Discussion
See also NSDocumentTypeDocumentOption (page 38).

Declared In
`NSAttributedString.h`

Document Attributes

The `init...` methods can return a dictionary with the following document-wide attributes (attribute identifiers available on Mac OS X v10.4 and later; use actual string value keys for earlier systems):

```
NSString *NSAuthorDocumentAttribute;
NSString *NSBackgroundColorDocumentAttribute;
NSString *NSBottomMarginDocumentAttribute;
NSString *NSCharacterEncodingDocumentAttribute;
NSString *NSCocoaVersionDocumentAttribute;
NSString *NSCommentDocumentAttribute;
NSString *NSCompanyDocumentAttribute;
NSString *NSConvertedDocumentAttribute;
NSString *NSCopyrightDocumentAttribute;
NSString *NSCreationTimeDocumentAttribute;
NSString *NSDefaultTabIntervalDocumentAttribute;
NSString *NSDocumentTypeDocumentAttribute;
NSString *NSEditorDocumentAttribute;
NSString *NSHyphenationFactorDocumentAttribute;
NSString *NSKeywordsDocumentAttribute;
NSString *NSLeftMarginDocumentAttribute;
NSString *NSModificationTimeDocumentAttribute;
NSString *NSPaperSizeDocumentAttribute;
NSString *NSReadOnlyDocumentAttribute;
NSString *NSRightMarginDocumentAttribute;
NSString *NSSubjectDocumentAttribute;
NSString *NSTitleDocumentAttribute;
NSString *NSTopMarginDocumentAttribute;
NSString *NSViewModeDocumentAttribute;
NSString *NSViewSizeDocumentAttribute;
NSString *NSViewZoomDocumentAttribute;
```

Constants

`NSPaperSizeDocumentAttribute`
`@"PaperSize"`

NSValue, containing NSSize.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSLeftMarginDocumentAttribute`
`@"LeftMargin"`

NSNumber, containing a float, in points.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSRightMarginDocumentAttribute`
`@"RightMargin"`

NSNumber, containing a float, in points.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSTopMarginDocumentAttribute`
`@"TopMargin"`

NSNumber, containing a float, in points.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSBottomMarginDocumentAttribute`
`@"BottomMargin"`

NSNumber, containing a float, in points.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSHyphenationFactorDocumentAttribute`
`@"HyphenationFactor"`

NSNumber, containing a float; 0 = off, 1 = full hyphenation.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSDocumentTypeDocumentAttribute`
`@"DocumentType"`

How the document was interpreted; one of the values below.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSCharacterEncodingDocumentAttribute`
`@"CharacterEncoding"`

NSNumber, containing an int specifying the `NSStringEncoding` for the file; for reading and writing plain text files and writing HTML; default for plain text is the default encoding; default for HTML is UTF-8.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSViewSizeDocumentAttribute`
`@"ViewSize"`

NSValue, containing NSSize.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSViewZoomDocumentAttribute`
`@"ViewZoom"`

NSValue, containing a float; 100 = 100% zoom.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSViewModeDocumentAttribute`
`@"ViewMode"`

NSValue, containing an int; 0 = normal; 1 = page layout (use value of `@"PaperSize"`.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSBackgroundColorDocumentAttribute`
`@"BackgroundColor"`

NSColor, representing the document-wide page background color.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSCocoaVersionDocumentAttribute`
`@"CocoaRTFVersion"`

NSNumber, containing a float. For RTF files only, stores the version of Cocoa with which the file was created. Absence of this value indicates RTF file not created by Cocoa or its predecessors. Values less than 100 are pre–Mac OS X; 100 is Mac OS X v10.0 or v10.1; 102 is Mac OS X v10.2 and 10.3; values greater than 102 correspond to values of `NSAppKitVersionNumber` on Mac OS X v10.4 and later.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSReadOnlyDocumentAttribute`
`@"ReadOnly"`

NSNumber, containing int. If missing or 0 or negative, not read only; 1 or more, read only. Note that this has nothing to do with the file system protection on the file, but instead can affect how the file should be displayed to the user.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSConvertedDocumentAttribute`
`@"Converted"`

NSNumber, containing an int. Indicates whether the file was converted by a filter service. If missing or 0, the file was originally in the format specified by document type. If negative, the file was originally in the format specified by document type, but the conversion to NSAttributedString may have been lossy. If 1 or more, it was converted to this type by a filter service.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSDefaultTabIntervalDocumentAttribute`
`@"DefaultTabInterval"`

NSNumber containing a float. Represents the document-wide default tab stop interval.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSTitleDocumentAttribute`

NSString containing document title.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSCompanyDocumentAttribute`

NSString containing company or organization name.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSCopyrightDocumentAttribute`

NSString containing document copyright info.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSSubjectDocumentAttribute`

NSString containing subject of document.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSAuthorDocumentAttribute`
: NSString containing author name.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSKeywordsDocumentAttribute`
: NSArray of NSString, containing keywords.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSCommentDocumentAttribute`
: NSString containing document comments.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSEditorDocumentAttribute`
: NSString containing name of person who last edited the document.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSCreationTimeDocumentAttribute`
: NSDate containing the creation date of the document; note that this is not the file system creation date of the file, but of the document.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSModificationTimeDocumentAttribute`
: NSDate containing the modification date of the document contents.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

Declared In
`NSAttributedString.h`

Attributes for generating HTML

These document-wide attributes provide control over the form of generated HTML—you use them only for writing HTML

```
NSString *NSExcludedElementsDocumentAttribute;
NSString *NSTextEncodingNameDocumentAttribute;
NSString *NSPrefixSpacesDocumentAttribute;
```

Constants
`NSExcludedElementsDocumentAttribute`
: An `NSArray` object containing `NSString` objects, representing HTML elements not to be used in generated HTML.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSTextEncodingNameDocumentAttribute`

An `NSString` object containing the name, IANA or otherwise, of a text encoding to be used; mutually exclusive with `NSCharacterEncodingDocumentAttribute`.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSPrefixSpacesDocumentAttribute`

An `NSNumber` containing an integer (default 0) representing the number of spaces per level by which to indent certain nested HTML elements.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

Discussion

`NSExcludedElementsDocumentAttribute` allows control over the tags used. The recognized values in the `NSExcludedElementsDocumentAttribute` array are (case-insensitive) HTML tags, plus DOCTYPE (representing a doctype declaration) and XML (representing an XML declaration). By default, if this attribute is not present, the excluded elements will be those deprecated in HTML 4 (APPLET, BASEFONT, CENTER, DIR, FONT, ISINDEX, MENU, S, STRIKE, and U) plus XML. If XML is on the list, HTML forms are used; if XML is not on the list, XHTML forms are used where there is a distinction. Either `NSCharacterEncodingDocumentAttribute` or `NSTextEncodingNameDocumentAttribute` may be used to control the encoding used for generated HTML; character entities are used for characters not representable in the specified encoding. `NSPrefixSpacesDocumentAttribute` allows some control over formatting.

Declared In

`NSAttributedString.h`

Option keys for importing documents

These option keys are recognized for importing documents using initWithData:options:documentAttributes:error: (page 14), initWithHTML:options:documentAttributes: (page 16), initWithURL:options:documentAttributes:error: (page 18), or the `readFrom...` methods (such as `readFromData:options:documentAttributes:`) implemented by `NSMutableAttributedString`.

```
NSString *NSBaseURLDocumentOption;
NSString *NSCharacterEncodingDocumentOption;
NSString *NSDefaultAttributesDocumentOption;
NSString *NSDocumentTypeDocumentOption;
NSString *NSTextEncodingNameDocumentOption;
NSString *NSTextSizeMultiplierDocumentOption;
NSString *NSTimeoutDocumentOption;
NSString *NSWebPreferencesDocumentOption;
NSString *NSWebResourceLoadDelegateDocumentOption;
```

Constants

`NSCharacterEncodingDocumentOption`

`@"CharacterEncoding"`

For plain text documents; NSNumber containing the unsigned int `NSStringEncoding` to override any encoding specified in an HTML document.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSBaseURLDocumentOption`
`@"BaseURL"`

For HTML documents; NSURL containing base URL.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSDefaultAttributesDocumentOption`
`@"DefaultAttributes"`

For plain text documents; NSDictionary containing attributes to be applied to plain files.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSDocumentTypeDocumentOption`
`@"DocumentType"`

One of the document types described in "Document Types" (page 31), indicating a document type to be forced when loading the document.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSTextEncodingNameDocumentOption`
`@"TextEncodingName"`

NSString containing the name, IANA or otherwise, of a text encoding to override any encoding specified in an HTML document. Mutually exclusive with `@"CharacterEncoding"`.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSTimeoutDocumentOption`
`@"Timeout"`

NSNumber containing float. Time in seconds to wait for a document to finish loading.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSWebPreferencesDocumentOption`
`@"WebPreferences"`

WebPreferences; for HTML only, specifies a WebPreferences object. If not present, a default set of preferences is used.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSWebResourceLoadDelegateDocumentOption`
`@"WebResourceLoadDelegate"`

NSObject; for HTML only, specifies an object to serve as the WebResourceLoadDelegate. If not present, a default delegate is used that permits the loading of subsidiary resources but does not respond to authentication challenges.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

`NSTextSizeMultiplierDocumentOption`

Specifies a scale factor for font sizes.

NSNumber containing float, default 1.0; for HTML only, corresponding to WebView's `textSizeMultiplier`.

There is no textual equivalent for Mac OS X v10.3.

Available in Mac OS X v10.4 and later.

Declared in `NSAttributedString.h`.

Discussion

In Mac OS X v10.3, the options key `@"UseWebKit"` specifies that WebKit–based HTML importing be used (and must be specified for the other options to be recognized). In Mac OS X v10.4 and later, WebKit is always used for HTML documents, and all of the options except `@"UseWebKit"` are recognized (attribute identifiers are available on Mac OS X v10.4 and later; use actual string value keys for Mac OS X v10.3):

Declared In

`NSAttributedString.h`

NSSpellingStateAttributeName

These constants control the display of the spelling and grammar indicators on text, highlighting portions of the text that are flagged for spelling or grammar issues. These regions are denoted by a temporary attribute on the layout manager, using the `NSSpellingStateAttributeName` key.

```
NSString *NSSpellingStateAttributeName;

enum {
    NSSpellingStateSpellingFlag = (1 << 0),
    NSSpellingStateGrammarFlag  = (1 << 1)
};
```

Constants

`NSSpellingStateAttributeName`

This key is available in Mac OS X v10.2 and later, but its interpretation changed in Mac OS X v10.5. Previously, any non-zero value caused the spelling indicator to be displayed. For Mac OS X v10.5 and later, the (integer) value is treated as being composed of the spelling and grammar flags.

`NSSpellingStateSpellingFlag`

Flag for spelling issues.

Available in Mac OS X v10.5 and later.

Declared in `NSAttributedString.h`.

`NSSpellingStateGrammarFlag`

Flag for grammar issues.

Available in Mac OS X v10.5 and later.

Declared in `NSAttributedString.h`.

Declared In

`NSAttributedString.h`

Deprecated NSAttributedString Application Kit Additions Methods

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

Deprecated in Mac OS X v10.5

textFileTypes

Returns an array of strings representing those file types that can be loaded as text. (Deprecated in Mac OS X v10.5. Use `textTypes` (page 9) instead.)

```
+ (NSArray *)textFileTypes
```

Discussion
This list includes all file types supported by text classes, plus those types that can be converted to supported file types through a user-installed filter service. The array returned by this method may be passed directly to *NSOpenPanel* method `runModalForTypes:`.

File types are identified by extension and HFS file types. By default, the list returned by this method includes "`txt`", "`rtf`", "`rtfd`", and "`html`".

When creating a subclass of `NSAttributedString` that accepts text data from nondefault file types, override `textUnfilteredFileTypes` (page 42) to notify `NSAttributedString` of the file types your class supports.

Availability
Available in Mac OS X v10.1 and later.
Deprecated in Mac OS X v10.5.

See Also
`+ textUnfilteredFileTypes` (page 42)

Declared In
`NSAttributedString.h`

textPasteboardTypes

Returns an array of pasteboard types that can be loaded as text. (Deprecated in Mac OS X v10.5. Use `textTypes` (page 9) instead.)

```
+ (NSArray *)textPasteboardTypes
```

Discussion
This list includes all pasteboard types supported by text classes and those that can be converted to supported pasteboard types through a user-installed filter service.

By default, the list returned by this method includes `NSHTMLPboardType`, `NSRTFPboardType`, `NSRTFDPboardType`, and `NSStringPboardType`.

When creating a subclass of `NSAttributedString` that accepts text data from nondefault pasteboard types, override `textUnfilteredPasteboardTypes` (page 42) to notify `NSAttributedString` of the pasteboard types your class supports.

Availability
Available in Mac OS X v10.1 and later.
Deprecated in Mac OS X v10.5.

See Also
`+ textUnfilteredPasteboardTypes` (page 42)

Declared In
`NSAttributedString.h`

textUnfilteredFileTypes

Returns an array of strings representing those file types that can be loaded as a text. (Deprecated in Mac OS X v10.5. Use `textUnfilteredTypes` (page 9) instead.)

```
+ (NSArray *)textUnfilteredFileTypes
```

Discussion
This list consists of all file types supported by text classes, but does not include those types that can be converted to supported file types through a user-installed filter service. The array returned by this method may be passed directly to `NSOpenPanel` method `runModalForTypes:`.

Availability
Available in Mac OS X v10.1 and later.
Deprecated in Mac OS X v10.5.

See Also
`+ textFileTypes` (page 41)

Declared In
`NSAttributedString.h`

textUnfilteredPasteboardTypes

Returns an array of pasteboard types that can be loaded as text. (Deprecated in Mac OS X v10.5. Use `textUnfilteredTypes` (page 9) instead.)

```
+ (NSArray *)textUnfilteredPasteboardTypes
```

Discussion
This list consists of all pasteboard types supported by text classes, but does not include those that can be converted to supported pasteboard types through a user-installed filter service.

Availability
Available in Mac OS X v10.1 and later.
Deprecated in Mac OS X v10.5.

See Also
`+ textPasteboardTypes` (page 41)

Declared In
`NSAttributedString.h`

Document Revision History

This table describes the changes to *NSAttributedString Application Kit Additions Reference*.

Date	Notes
2007-12-04	Enhanced descriptions of NSStrokeWidthAttributeName, NSKernAttributeName, and other standard attribute constants.
2007-03-30	Added new methods introduced with Mac OS X v10.5. Corrected minor typographical errors. Fixed link to standard font attribute constants.
	Clarified clipping behavior and text flow in `drawInRect:` (page 12) and `drawAtPoint:` (page 11) method descriptions.
2006-05-23	Corrected return type for itemNumberInTextList:atIndex: to int and noted additional attribute not returned by fontAttributesInRange:.
	Corrected return type for itemNumberInTextList:atIndex: to int and noted additional attribute not returned by fontAttributesInRange:.
	First publication of this content as a separate document.

Document Revision History

Index

N

O

R

S

T

U