

---

# CFAttributedString Reference

Data Management: Strings, Text, & Fonts



2009-05-06



Apple Inc.  
© 2004, 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, and Mac OS are trademarks of Apple Inc., registered in the United States and other countries.

iPhone is a trademark of Apple Inc.

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

---

## **CFAttributedString Reference 5**

---

|   |    |
|---|----|
| Overview  | 5  |
| Functions by Task                                       | 6  |
| Creating a CFAttributedString                           | 6  |
| Accessing Attributes                                    | 6  |
| Getting Attributed String Properties                    | 6  |
| Functions   | 7  |
| CFAttributedStringCreate                                | 7  |
| CFAttributedStringCreateCopy                            | 7  |
| CFAttributedStringCreateWithSubstring                   | 8  |
| CFAttributedStringGetAttribute                          | 8  |
| CFAttributedStringGetAttributeAndLongestEffectiveRange  | 9  |
| CFAttributedStringGetAttributes                         | 10 |
| CFAttributedStringGetAttributesAndLongestEffectiveRange | 11 |
| CFAttributedStringGetLength                             | 11 |
| CFAttributedStringGetString                             | 12 |
| CFAttributedStringGetTypeID                             | 12 |
| Data Types  | 13 |
| CFAttributedStringRef                                   | 13 |

---

## **Document Revision History 15**

---

## **Index 17**

---



# CFAttributedString Reference

---

|                         |  |
|-------------------------|--|
| <b>Derived From:</b>    | <i>CType Reference</i>   |
| <b>Framework:</b>       | CoreFoundation/CoreFoundation.h  |
| <b>Declared in</b>      | CFAttributedString.h   |
| <b>Companion guides</b> | Property List Programming Topics for Core Foundation<br>Strings Programming Guide for Core Foundation<br>Data Formatting Guide for Core Foundation |

## Overview

Instances of `CFAttributedString` manage character strings and associated sets of attributes (for example, font and kerning information) that apply to individual characters or ranges of characters in the string. `CFAttributedString` as defined in `CoreFoundation` provides the basic container functionality, while higher levels provide definitions for standard attributes, their values, and additional behaviors involving these. `CFAttributedString` represents an immutable string—use `CFMutableAttributedString` to create and manage an attributed string that can be changed after it has been created.

**iPhone OS Note:** While Core Foundation on iPhone OS contains `CFAttributedString`, there are no additions to the APIs in UIKit to add specific attributes such as font, style, or color, and there are no APIs to draw attributed strings.

`CFAttributedString` is not a “subclass” of `CFString`; that is, it does not respond to `CFString` function calls. `CFAttributedString` conceptually contains a `CFString` to which it applies attributes. This protects you from ambiguities caused by the semantic differences between simple and attributed string.

Attributes are identified by key/value pairs stored in `CFDictionary` objects. Keys must be `CFString` objects, while the corresponding values are `CType` objects of an appropriate type. See the attribute constants in *NSAttributedString Application Kit Additions Reference* for standard attribute names.

**Important:** Attribute dictionaries set for an attributed string must always be created with `kCFCopyStringDictionaryKeyCallbacks` for their dictionary key callbacks and `kCFTypeDictionaryValueCallbacks` for their value callbacks; otherwise it's an error.

On Mac OS X, `CFAttributedString` is “toll-free bridged” with its Cocoa Foundation counterpart, `NSAttributedString`. This means that the Core Foundation type is interchangeable in function or method calls with the bridged Foundation object. Therefore, in a method where you see an `NSAttributedString *` parameter, you can pass in a `CFAttributedStringRef`, and in a function where you see a

`CFAttributedStringRef` parameter, you can pass in an `NSAttributedString` instance. This also applies to concrete subclasses of `NSAttributedString`. See [Interchangeable Data Types](#) for more information on toll-free bridging.

**iPhone OS Note:** `NSAttributedString` is not available on iPhone OS.

## Functions by Task

### Creating a CFAttributedString

[CFAttributedStringCreate](#) (page 7)

Creates an attributed string with specified string and attributes.

[CFAttributedStringCreateCopy](#) (page 7)

Creates an immutable copy of an attributed string.

[CFAttributedStringCreateWithSubstring](#) (page 8)

Creates a sub-attributed string from the specified range.

[CFAttributedStringGetLength](#) (page 11)

Returns the length of the attributed string in characters.

[CFAttributedStringGetString](#) (page 12)

Returns the string for an attributed string.

### Accessing Attributes

[CFAttributedStringGetAttribute](#) (page 8)

Returns the value of a given attribute of an attributed string at a specified location.

[CFAttributedStringGetAttributes](#) (page 10)

Returns the attributes of an attributed string at a specified location.

[CFAttributedStringGetAttributeAndLongestEffectiveRange](#) (page 9)

Returns the value of a given attribute of an attributed string at a specified location.

[CFAttributedStringGetAttributesAndLongestEffectiveRange](#) (page 11)

Returns the attributes of an attributed string at a specified location.

### Getting Attributed String Properties

[CFAttributedStringGetTypeID](#) (page 12)

Returns the type identifier for the `CFAttributedString` opaque type.

## Functions

### CFAttributedStringCreate

Creates an attributed string with specified string and attributes.

```
CFAttributedStringRef CFAttributedStringCreate (
    CFAllocatorRef alloc,
    CFStringRef str,
    CFDictionaryRef attributes
);
```

#### Parameters

*alloc*

The allocator to use to allocate memory for the new attributed string. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

*str*

A string that specifies the characters to use in the new attributed string. This value is copied.

*attributes*

A dictionary that contains the attributes to apply to the new attributed string. This value is copied.

#### Return Value

An attributed string that contains the characters from *str* and the attributes specified by *attributes*. The result is `NULL` if there was a problem in creating the attributed string. Ownership follows the Create Rule.

#### Discussion

Note that both the string and the attributes dictionary are copied. The specified attributes are applied to the whole string. If you want to apply different attributes to different ranges of the string, you should use a mutable attributed string.

#### Availability

Available in Mac OS X v10.4 and later.

#### Related Sample Code

CoreTextTest

#### Declared In

CFAttributedString.h

### CFAttributedStringCreateCopy

Creates an immutable copy of an attributed string.

```
CFAttributedStringRef CFAttributedStringCreateCopy (
    CFAllocatorRef alloc,
    CFAttributedStringRef aStr
);
```

#### Parameters

*alloc*

The allocator to use to allocate memory for the new attributed string. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

*aStr*

The attributed string to copy.

**Return Value**

An immutable attributed string with characters and attributes identical to those of *aStr*. Returns NULL if there was a problem copying the object. Ownership follows the Create Rule.

**Availability**

Available in Mac OS X v10.4 and later.

**Declared In**

CFAttributedString.h

## CFAttributedStringCreateWithSubstring

Creates a sub-attributed string from the specified range.

```
CFAttributedStringRef CFAttributedStringCreateWithSubstring (
    CFAllocatorRef alloc,
    CFAttributedStringRef aStr,
    CFRange range
);
```

**Parameters**

*alloc*

The allocator to use to allocate memory for the new attributed string. Pass NULL or `kCFAllocatorDefault` to use the current default allocator.

*theString*

The attributed string to copy.

*range*

The range of the attributed string to copy. *range* must not exceed the bounds of *aStr*.

**Return Value**

A new attributed string whose string and attributes are copied from from the specified range of the supplied attributed string. Returns NULL if there was a problem copying the object. Ownership follows the Create Rule.

**Availability**

Available in Mac OS X v10.4 and later.

**Declared In**

CFAttributedString.h

## CFAttributedStringGetAttribute

Returns the value of a given attribute of an attributed string at a specified location.

```

CTypeRef CFAttributedStringGetAttribute (
    CFAttributedStringRef aStr,
    CFIndex loc,
    CFStringRef attrName,
    CFRange *effectiveRange
);

```

**Parameters***str*

The attributed string to examine.

*loc*The location in *str* at which to determine the attributes. *loc* must not exceed the bounds of *str*.*attrName*

The name of the attribute whose value you want to determine.

*effectiveRange*If not NULL, upon return contains a range including *loc* over which exactly the same set of attributes apply as at *loc*.**Return Value**The value of the specified attribute at the specified location in *str*. Ownership follows the Get Rule.**Discussion**For performance reasons, a range returned in *effectiveRange* is not necessarily the maximal range. If you need the maximum range, you should use[CFAttributedStringGetAttributeAndLongestEffectiveRange](#) (page 9).**Availability**

Available in Mac OS X v10.4 and later.

**Declared In**

CFAttributedString.h

**CFAttributedStringGetAttributeAndLongestEffectiveRange**

Returns the value of a given attribute of an attributed string at a specified location.

```

CTypeRef CFAttributedStringGetAttributeAndLongestEffectiveRange (
    CFAttributedStringRef aStr,
    CFIndex loc,
    CFStringRef attrName,
    CFRange inRange,
    CFRange *longestEffectiveRange
);

```

**Parameters***str*

The attributed string to examine.

*loc*The location in *str* at which to determine the attributes. It is a programming error for *loc* to specify a location outside the bounds of *str*.*attrName*

The name of the attribute whose value you want to determine.

*inRange*

The range in *str* within which you want to find the longest effective range of the attributes at *loc*. *inRange* must not exceed the bounds of *str*.

*effectiveRange*

If not NULL, upon return contains the maximal range within *inRange* over which the exact same set of attributes apply. The returned range is clipped to *inRange*.

#### Return Value

A dictionary that contains the attributes of *str* at the specified location. Ownership follows the Get Rule.

#### Availability

Available in Mac OS X v10.4 and later.

#### Declared In

CFAttributedString.h

## CFAttributedStringGetAttributes

Returns the attributes of an attributed string at a specified location.

```
CFDictionaryRef CFAttributedStringGetAttributes (
    CFAttributedStringRef aStr,
    CFIndex loc,
    CFRange *effectiveRange
);
```

#### Parameters

*str*

The attributed string to examine.

*loc*

The location in *str* at which to determine the attributes. *loc* must not exceed the bounds of *str*.

*effectiveRange*

If not NULL, upon return contains a range including *loc* over which exactly the same set of attributes apply as at *loc*.

#### Return Value

A dictionary that contains the attributes of *str* at the specified location. Ownership follows the Get Rule.

#### Discussion

For performance reasons, a range returned in *effectiveRange* is not necessarily the maximal range. If you need the maximum range, you should use

[CFAttributedStringGetAttributesAndLongestEffectiveRange](#) (page 11).

Note that the returned attribute dictionary might change in unpredictable ways if the attributed string is edited after this call. If you want to preserve the state of the dictionary, you should make an actual copy of it rather than just retaining it. In addition, you should make no assumptions about the relationship of the actual dictionary returned by this call and the dictionary originally used to set the attributes, other than the fact that the values stored in the dictionaries will be identical (that is, ==) to those originally specified.

#### Availability

Available in Mac OS X v10.4 and later.

#### Declared In

CFAttributedString.h

## CFAttributedStringGetAttributesAndLongestEffectiveRange

Returns the attributes of an attributed string at a specified location.

```

CFDictionaryRef CFAttributedStringGetAttributesAndLongestEffectiveRange (
    CFAttributedStringRef aStr,
    CFIndex loc,
    CFRange inRange,
    CFRange *longestEffectiveRange
);

```

### Parameters

*str*

The attributed string to examine.

*loc*

The location in *str* at which to determine the attributes. *loc* must not exceed the bounds of *str*.

*inRange*

The range in *str* within to find the longest effective range of the attributes at *loc*. *inRange* must not exceed the bounds of *str*.

*effectiveRange*

If not NULL, upon return contains the maximal range within *inRange* over which the exact same set of attributes apply. The returned range is clipped to *inRange*.

### Return Value

A dictionary that contains the attributes of *str* at the specified location. Ownership follows the Get Rule.

### Availability

Available in Mac OS X v10.4 and later.

### Declared In

CFAttributedString.h

## CFAttributedStringGetLength

Returns the length of the attributed string in characters.

```

CFIndex CFAttributedStringGetLength (
    CFAttributedStringRef aStr
);

```

### Parameters

*str*

The attributed string to examine.

### Return Value

The length of the attributed string in characters; this is the same as `CFStringGetLength(CFAttributedStringGetString(aStr))`.

### Availability

Available in Mac OS X v10.4 and later.

### Related Sample Code

CoreTextTest

**Declared In**

CFAttributedString.h

**CFAttributedStringGetString**

Returns the string for an attributed string.

```
CFStringRef CFAttributedStringGetString (
    CFAttributedStringRef aStr
);
```

**Parameters***aStr*

The attributed string to examine.

**Return Value**An immutable string containing the characters from *aStr*, or NULL if there was a problem creating the object. Ownership follows the Get Rule.**Discussion**

For performance reasons, the string returned will often be the backing store of the attributed string, and it might therefore change if the attributed string is edited. However, this is an implementation detail, and you should not rely on this behavior.

**Availability**

Available in Mac OS X v10.4 and later.

**Declared In**

CFAttributedString.h

**CFAttributedStringGetTypeID**

Returns the type identifier for the CFAttributedString opaque type.

```
CTypeID CFAttributedStringGetTypeID (
    void
);
```

**Return Value**

The type identifier for the CFAttributedString opaque type.

**Discussion**

CFMutableAttributedString objects have the same type identifier as CFAttributedString objects.

**Availability**

Available in Mac OS X v10.4 and later.

**Declared In**

CFAttributedString.h

## Data Types

### CFAttributedStringRef

A reference to a CFAttributedString object.

```
typedef const struct __CFAttributedString *CFAttributedStringRef;
```

#### Discussion

The `CFAttributedStringRef` type refers to an object that combines a `CFString` object with a collection of attributes that specify how the characters in the string should be displayed. `CFAttributedString` is an opaque type that defines the characteristics and behavior of `CFAttributedString` objects.

Values of type `CFAttributedStringRef` may refer to immutable or mutable strings, as `CFMutableAttributedString` objects respond to all functions intended for immutable `CFAttributedString` objects. Functions which accept `CFAttributedStringRef` values, and which need to hold on to the values immutably, should call `CFAttributedStringCreateWithSubstring` (page 8) (instead of `CFRetain`) to do so.

#### Availability

Available in Mac OS X v10.4 and later.

#### Declared In

`CFAttributedString.h`



# Document Revision History

---

This table describes the changes to *CFAttributedString Reference*.

| Date       | Notes   |
|------------|---|
| 2009-05-06 | Added note that CFAttributedString is not toll-free bridged to NSAttributedString on iPhone.  |
| 2008-07-01 | Added note to introduction about attributed string support in iPhone OS. Enhanced warning about attribute dictionaries requiring key and value callbacks. |
| 2005-12-06 | Made minor changes to text to conform to reference consistency guidelines.  |
| 2005-11-09 | Updated link in Companion Documents.  |
| 2005-04-29 | First version of this document.   |

**REVISION HISTORY**

Document Revision History

# Index

---

## C

---

- CFAttributedStringCreate **function** [7](#)
- CFAttributedStringCreateCopy **function** [7](#)
- CFAttributedStringCreateWithSubstring **function** [8](#)
- CFAttributedStringGetAttribute **function** [8](#)
- CFAttributedStringGetAttributeAndLongestEffectiveRange **function** [9](#)
- CFAttributedStringGetAttributes **function** [10](#)
- CFAttributedStringGetAttributesAndLongestEffectiveRange **function** [11](#)
- CFAttributedStringGetLength **function** [11](#)
- CFAttributedStringGetString **function** [12](#)
- CFAttributedStringGetTypeID **function** [12](#)
- CFAttributedStringRef **data type** [13](#)