Core Foundation Reference Update

Core Foundation

2007-07-18

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

Simultaneously published in the United States and Canada.

Contents

Introduction to Core Foundation Reference Update

This document summarizes the symbols that have been added to the Core Foundation framework. The full reference documentation notes in what version a symbol was introduced, but sometimes it's useful to see only the new symbols for a given release.

If you are not familiar with this framework you should refer to the complete framework reference documentation.

Organization of This Document

Symbols are grouped by class or protocol for Objective-C and by header file for C. For each symbol there is a link to complete documentation, if available, and a brief description, if available.

See Also

For reference documentation on this framework, see *Core Foundation Framework Reference*.

10.5 Symbol Changes

This article lists the symbols added to `CoreFoundation.framework` in Mac OS X v10.5.

C Symbols

All of the header files with new symbols are listed alphabetically, with their new symbols described.

CFBase.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CF_EXTERN_C_BEGIN`	
`CF_EXTERN_C_END`	
`kCFCoreFoundationVersionNumber10_1_1`	
`kCFCoreFoundationVersionNumber10_1_3`	
`kCFCoreFoundationVersionNumber10_2_1`	The Core Foundation framework version in Mac OS X version 10.2.1.
`kCFCoreFoundationVersionNumber10_2_2`	The Core Foundation framework version in Mac OS X version 10.2.2.
`kCFCoreFoundationVersionNumber10_2_3`	The Core Foundation framework version in Mac OS X version 10.2.3.
`kCFCoreFoundationVersionNumber10_2_4`	The Core Foundation framework version in Mac OS X version 10.2.4.
`kCFCoreFoundationVersionNumber10_2_5`	The Core Foundation framework version in Mac OS X version 10.2.5.
`kCFCoreFoundationVersionNumber10_2_6`	The Core Foundation framework version in Mac OS X version 10.2.6.
`kCFCoreFoundationVersionNumber10_2_7`	The Core Foundation framework version in Mac OS X version 10.2.7.

kCFCoreFoundationVersionNumber10_2_8	The Core Foundation framework version in Mac OS X version 10.2.8.
kCFCoreFoundationVersionNumber10_3_1	The Core Foundation framework version in Mac OS X version 10.3.1.
kCFCoreFoundationVersionNumber10_3_2	The Core Foundation framework version in Mac OS X version 10.3.2.
kCFCoreFoundationVersionNumber10_3_5	The Core Foundation framework version in Mac OS X version 10.3.5.
kCFCoreFoundationVersionNumber10_3_6	The Core Foundation framework version in Mac OS X version 10.3.6.
kCFCoreFoundationVersionNumber10_3_7	The Core Foundation framework version in Mac OS X version 10.3.7.
kCFCoreFoundationVersionNumber10_3_8	The Core Foundation framework version in Mac OS X version 10.3.8.
kCFCoreFoundationVersionNumber10_3_9	The Core Foundation framework version in Mac OS X version 10.3.9.
kCFCoreFoundationVersionNumber10_4	The Core Foundation framework version in Mac OS X version 10.4.
kCFCoreFoundationVersionNumber10_4_1	The Core Foundation framework version in Mac OS X version 10.4.1.
kCFCoreFoundationVersionNumber10_4_10	
kCFCoreFoundationVersionNumber10_4_11	
kCFCoreFoundationVersionNumber10_4_2	The Core Foundation framework version in Mac OS X version 10.4.2.
kCFCoreFoundationVersionNumber10_4_3	The Core Foundation framework version in Mac OS X version 10.4.3.
kCFCoreFoundationVersionNumber10_4_4_Intel	The Core Foundation framework version in Mac OS X version 10.4.4 on Intel-based Macintosh computers.
kCFCoreFoundationVersionNumber10_4_4_PowerPC	The Core Foundation framework version in Mac OS X version 10.4.4 on PowerPC-based Macintosh computers.
kCFCoreFoundationVersionNumber10_4_5_Intel	The Core Foundation framework version in Mac OS X version 10.4.5 on Intel-based Macintosh computers.
kCFCoreFoundationVersionNumber10_4_5_PowerPC	The Core Foundation framework version in Mac OS X version 10.4.5 on PowerPC-based Macintosh computers.

kCFCoreFoundationVersionNumber10_4_6_Intel	The Core Foundation framework version in Mac OS X version 10.4.6 on Intel-based Macintosh computers.
kCFCoreFoundationVersionNumber10_4_6_PowerPC	The Core Foundation framework version in Mac OS X version 10.4.6 on PowerPC-based Macintosh computers.
kCFCoreFoundationVersionNumber10_4_7	The Core Foundation framework version in Mac OS X version 10.4.7.
kCFCoreFoundationVersionNumber10_4_8	
kCFCoreFoundationVersionNumber10_4_9	

CFBundle.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

CFBundleCopyExecutableArchitectures	Returns an array of CFNumbers representing the architectures a given bundle provides.
CFBundleCopyExecutableArchitecturesForURL	Returns an array of CFNumbers representing the architectures a given URL provides.
CFBundleLoadExecutableAndReturnError	Returns a Boolean value that indicates whether a given bundle is loaded, attempting to load it if necessary.
CFBundlePreflightExecutable	Returns a Boolean value that indicates whether a given bundle is loaded or appears to be loadable.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

CFBundleRefNum	Type that identifies a distinct reference number for a resource map.
kCFBundleExecutableArchitectureI386	Specifies the 32-bit Intel architecture.
kCFBundleExecutableArchitecturePPC	Specifies the 32-bit PowerPC architecture.
kCFBundleExecutableArchitecturePPC64	Specifies the 64-bit PowerPC architecture.
kCFBundleExecutableArchitectureX86_64	Specifies the 64-bit Intel architecture.

CFByteOrder.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CF_USE_OSBYTEORDER_H`	

CFCalendar.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFCalendarGetTimeRangeOfUnit`	Returns by reference the start time and duration of a given calendar unit that contains a given absolute time.

CFCharacterSet.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFCharacterSetNewline`	Newline character set (U000A ~ U000D, U0085, U2028, and U2029).

CFDateFormatter.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFDateFormatterGregorianStartDate`	Specifies the Gregorian start date property, a CFDate object.
`kCFDateFormatterLongEraSymbols`	Specifies the long era symbols property, a CFArray of CFString objects.

`kCFDateFormatterQuarterSymbols`	Specifies the quarter symbols property, a CFArray of CFString objects.
`kCFDateFormatterShortQuarterSymbols`	Specifies the short quarter symbols property, a CFArray of CFString objects.
`kCFDateFormatterShortStandaloneMonthSymbols`	Specifies the short standalone month symbols property, a CFArray of CFString objects.
`kCFDateFormatterShortStandaloneQuarterSymbols`	Specifies the short standalone quarter symbols property, a CFArray of CFString objects.
`kCFDateFormatterShortStandaloneWeekdaySymbols`	Specifies the short standalone weekday symbols property, a CFArray of CFString objects.
`kCFDateFormatterStandaloneMonthSymbols`	Specifies the standalone month symbols property, a CFArray of CFString objects.
`kCFDateFormatterStandaloneQuarterSymbols`	Specifies the standalone quarter symbols property, a CFArray of CFString objects.
`kCFDateFormatterStandaloneWeekdaySymbols`	Specifies the standalone weekday symbols property, a CFArray of CFString objects.
`kCFDateFormatterVeryShortMonthSymbols`	Specifies the very short month symbols property, a CFArray of CFString objects.
`kCFDateFormatterVeryShortStandaloneMonthSymbols`	Specifies the very short standalone month symbols property, a CFArray of CFString objects.
`kCFDateFormatterVeryShortStandaloneWeekdaySymbols`	Specifies the very short standalone weekday symbols property, a CFArray of CFString objects.
`kCFDateFormatterVeryShortWeekdaySymbols`	Specifies the very short weekday symbols property, a CFArray of CFString objects.

CFError.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFErrorCopyDescription`	Returns a human-presentable description for a given error.
`CFErrorCopyFailureReason`	Returns a human-presentable failure reason for a given error.
`CFErrorCopyRecoverySuggestion`	Returns a human presentable recovery suggestion for a given error.
`CFErrorCopyUserInfo`	Returns the user info dictionary for a given CFError.
`CFErrorCreate`	Creates a new CFError object.
`CFErrorCreateWithUserInfoKeysAndValues`	Creates a new CFError object using given keys and values to create the user info dictionary.
`CFErrorGetCode`	Returns the error code for a given CFError.
`CFErrorGetDomain`	Returns the error domain for a given CFError.
`CFErrorGetTypeID`	Returns the type identifier for the CFError opaque type.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFErrorRef`	A reference to a CFError object.
`kCFErrorDescriptionKey`	Key to identify the description in the userInfo dictionary.
`kCFErrorDomainCocoa`	A constant that specified the Cocoa domain.
`kCFErrorDomainMach`	A constant that specified the Mach domain.
`kCFErrorDomainOSStatus`	A constant that specified the OS domain.
`kCFErrorDomainPOSIX`	A constant that specified the POSIX domain.
`kCFErrorLocalizedDescriptionKey`	Key to identify the end user-presentable description in the userInfo dictionary.
`kCFErrorLocalizedFailureReasonKey`	Key to identify the end user-presentable failure reason in the userInfo dictionary.
`kCFErrorLocalizedRecoverySuggestionKey`	Key to identify the end user-presentable recovery suggestion in the userInfo dictionary.
`kCFErrorUnderlyingErrorKey`	Key to identify the underlying error in the userInfo dictionary.

CFFileDescriptor.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFFileDescriptorCreate`	Creates a new CFFileDescriptor.
`CFFileDescriptorCreateRunLoopSource`	Creates a new runloop source for a given CFFileDescriptor.
`CFFileDescriptorDisableCallBacks`	Disables callbacks for a given CFFileDescriptor.
`CFFileDescriptorEnableCallBacks`	Enables callbacks for a given CFFileDescriptor.
`CFFileDescriptorGetContext`	Gets the context for a given CFFileDescriptor.
`CFFileDescriptorGetNativeDescriptor`	Returns the native file descriptor for a given CFFileDescriptor.
`CFFileDescriptorGetTypeID`	Returns the type identifier for the CFFileDescriptor opaque type.
`CFFileDescriptorInvalidate`	Invalidates the native file descriptor for a given CFFileDescriptor.
`CFFileDescriptorIsValid`	Returns a Boolean value that indicates whether the native file descriptor for a given CFFileDescriptor is valid.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFFileDescriptorCallBack`	
`CFFileDescriptorContext`	Defines a structure for the context of a CFFileDescriptor.
`CFFileDescriptorNativeDescriptor`	Defines a type for the native file descriptor.
`CFFileDescriptorRef`	A reference to an CFFileDescriptor object.
`kCFFileDescriptorReadCallBack`	Identifies the read callback.
`kCFFileDescriptorWriteCallBack`	Identifies the write callback.

CFLocale.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFLocaleCopyCommonISOCurrencyCodes`	Returns an array of strings that represents ISO currency codes for currencies in common use.
`CFLocaleCopyPreferredLanguages`	Returns the array of canonicalized locale IDs that the user prefers.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFLocaleCurrentLocaleDidChangeNotification`	Identifier for the notification sent if the current locale changes.

CFNumber.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFNumberCGFloatType`	CGFloat value.
`kCFNumberNSIntegerType`	NSInteger value.

CFNumberFormatter.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFNumberFormatterCurrencyGroupingSeparator`	Specifies the grouping symbol to use when placing a currency value within a string, a CFString object.

kCFNumberFormatterIsLenient	Specifies whether the formatter is lenient, aCFBoolean object.
kCFNumberFormatterMaxSignificantDigits	Specifies the maximum number of significant digits to use, aCFNumber object.
kCFNumberFormatterMinSignificantDigits	Specifies the minimum number of significant digits to use, aCFNumber object.
kCFNumberFormatterUseSignificantDigits	Specifies the whether the formatter uses significant digits, a CFBoolean object.

CFRunLoop.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

CFRunLoopGetMain	Returns the main CFRunLoop object.

CFStream.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

CFReadStreamCopyError	Returns the error associated with a stream.
CFStreamCreateBoundPair	Creates a pair of read and write streams.
CFWriteStreamCopyError	Returns the error associated with a stream.

CFString.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

CFStringCompareWithOptionsAndLocale	Compares a range of the characters in one string with another string using a given locale.

`CFStringCreateWithBytesNoCopy`	Creates a string from a buffer, containing characters in a specified encoding, that might serve as the backing store for the new string.
`CFStringFindWithOptionsAndLocale`	Returns a Boolean value that indicates whether a given string was found in a given source string.
`CFStringFold`	Folds a given string into the form specified by optional flags.
`CFStringGetParagraphBounds`	Given a range of characters in a string, obtains the paragraph bounds—that is, the indexes of the first character and the final characters of the paragraph(s) containing the range.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFCompareDiacriticInsensitive`	Specifies that the comparison should ignore diacritic markers.
`kCFCompareForcedOrdering`	Specifies that the comparison is forced to return either kCFCompareLessThan or kCFCompareGreaterThan if the strings are equivalent but not strictly equal.
`kCFCompareWidthInsensitive`	Specifies that the comparison should ignore width differences.
`kCFStringTransformStripDiacritics`	The identifier of a transform to remove diacritic markings.

CFStringEncodingExt.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFStringEncodingShiftJIS_X0213`	

CFStringTokenizer.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFStringTokenizerAdvanceToNextToken`	Advances the tokenizer to the next token and sets that as the current token.
`CFStringTokenizerCopyBestStringLanguage`	Guesses a language of a given string and returns the guess as a BCP 47 string.
`CFStringTokenizerCopyCurrentTokenAttribute`	Returns a given attribute of the current token.
`CFStringTokenizerCreate`	Returns a tokenizer for a given string.
`CFStringTokenizerGetCurrentSubTokens`	Retrieves the subtokens or derived subtokens contained in the compound token.
`CFStringTokenizerGetCurrentTokenRange`	Returns the range of the current token.
`CFStringTokenizerGetTypeID`	Returns the type ID for CFStringTokenizer.
`CFStringTokenizerGoToTokenAtIndex`	Finds a token that includes the character at a given index, and set it as the current token.
`CFStringTokenizerSetString`	Sets the string for a tokenizer.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFStringTokenizerRef`	A reference to a CFStringTokenizer object.
`CFStringTokenizerTokenType`	Token types returned by CFStringTokenizerGoToTokenAtIndex and CFStringTokenizerAdvanceToNextToken.
`kCFStringTokenizerAttributeLanguage`	Tells the tokenizer to prepare the language (specified as an RFC 3066bis string) when it tokenizes the string.
`kCFStringTokenizerAttributeLatinTranscription`	Used with kCFStringTokenizerUnitWord, tells the tokenizer to prepare the Latin transcription when it tokenizes the string.
`kCFStringTokenizerTokenHasDerivedSubTokensMask`	Compound token which may contain derived subtokens.
`kCFStringTokenizerTokenHasHasNumbersMask`	Appears to contain a number.
`kCFStringTokenizerTokenHasNonLettersMask`	Contains punctuation, symbols, and so on.
`kCFStringTokenizerTokenHasSubTokensMask`	Compound token which may contain subtokens but with no derived subtokens.
`kCFStringTokenizerTokenIsCJWordMask`	Contains kana and/or ideographs.
`kCFStringTokenizerTokenNone`	Has no token.

kCFStringTokenizerTokenNormal	Has a normal token.
kCFStringTokenizerUnitLineBreak	Specifies that a string should be tokenized by line break. The locale parameter of CFStringTokenizerCreate is ignored.
kCFStringTokenizerUnitParagraph	Specifies that a string should be tokenized by paragraph. The locale parameter of CFStringTokenizerCreate is ignored.
kCFStringTokenizerUnitSentence	Specifies that a string should be tokenized by sentence. The locale parameter of CFStringTokenizerCreate is ignored.
kCFStringTokenizerUnitWord	Specifies that a string should be tokenized by word. The locale parameter of CFStringTokenizerCreate is ignored.
kCFStringTokenizerUnitWordBoundary	Specifies that a string should be tokenized by locale-sensitive word boundary.

CFTimeZone.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

CFTimeZoneCopyLocalizedName	Returns the localized name of a given time zone.
CFTimeZoneGetDaylightSavingTimeOffset	Returns the daylight saving time offset for a time zone at a given time.
CFTimeZoneGetNextDaylightSavingTimeTransition	Returns the time in a given time zone of the next daylight saving time transition after a given time.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

CFTimeZoneNameStyle	Index type for constants used to specify styles of time zone names.
kCFTimeZoneNameStyleDaylightSaving	Specifies the daylight saving name style for a time zone.

`kCFTimeZoneNameStyleShortDaylightSaving`	Specifies the short daylight saving name style for a time zone.
`kCFTimeZoneNameStyleShortStandard`	Specifies the short standard name style for a time zone.
`kCFTimeZoneNameStyleStandard`	Specifies the standard name style for a time zone.
`kCFTimeZoneSystemTimeZoneDidChangeNotification`	Name of the notification posted when the system time zone changes.

10.4 Symbol Changes

This article lists the symbols added to `CoreFoundation.framework` in Mac OS X v10.4.

C Symbols

All of the header files with new symbols are listed alphabetically, with their new symbols described.

CFAttributedString.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFAttributedStringBeginEditing`	Defers internal consistency-checking and coalescing for a mutable attributed string.
`CFAttributedStringCreate`	Creates an attributed string with specified string and attributes.
`CFAttributedStringCreateCopy`	Creates an immutable copy of an attributed string.
`CFAttributedStringCreateMutable`	Creates a mutable attributed string.
`CFAttributedStringCreateMutableCopy`	Creates a mutable copy of an attributed string.
`CFAttributedStringCreateWithSubstring`	Creates a sub-attributed string from the specified range.
`CFAttributedStringEndEditing`	Re-enables internal consistency-checking and coalescing for a mutable attributed string.
`CFAttributedStringGetAttribute`	Returns the value of a given attribute of an attributed string at a specified location.
`CFAttributedStringGetAttribute-AndLongestEffectiveRange`	Returns the value of a given attribute of an attributed string at a specified location.
`CFAttributedStringGetAttributes`	Returns the attributes of an attributed string at a specified location.

`CFAttributedStringGetAttributes-AndLongestEffectiveRange`	Returns the attributes of an attributed string at a specified location.
`CFAttributedStringGetLength`	Returns the length of the attributed string in characters.
`CFAttributedStringGetMutableString`	Gets as a mutable string the string for an attributed string.
`CFAttributedStringGetString`	Returns the string for an attributed string.
`CFAttributedStringGetTypeID`	Returns the type identifier for the CFAttributedString opaque type.
`CFAttributedStringRemoveAttribute`	Removes the value of a single attribute over a specified range.
`CFAttributedStringReplaceAttributedString`	Replaces the attributed substring over a range with another attributed string.
`CFAttributedStringReplaceString`	Modifies the string of an attributed string.
`CFAttributedStringSetAttribute`	Sets the value of a single attribute over the specified range.
`CFAttributedStringSetAttributes`	Sets the value of attributes of a mutable attributed string over a specified range.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFAttributedStringRef`	A reference to a CFAttributedString object.
`CFMutableAttributedStringRef`	A reference to a CFMutableAttributedString object.

CFBase.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFMakeCollectable`	Makes a newly-allocated Core Foundation object eligible for garbage collection.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFAllocatorMallocZone`	This allocator explicitly uses the default malloc zone, returned by malloc_default_zone().
`kCFCoreFoundationVersionNumber10_3`	The Core Foundation framework version in Mac OS X version 10.3.
`kCFCoreFoundationVersionNumber10_3_3`	The Core Foundation framework version in Mac OS X version 10.3.3.
`kCFCoreFoundationVersionNumber10_3_4`	The Core Foundation framework version in Mac OS X version 10.3.4.

CFCalendar.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFCalendarAddComponents`	Computes the absolute time when specified components are added to a given absolute time.
`CFCalendarComposeAbsoluteTime`	Computes the absolute time from components in a description string.
`CFCalendarCopyCurrent`	Returns a copy of the logical calendar for the current user.
`CFCalendarCopyLocale`	Returns a locale object for a specified calendar.
`CFCalendarCopyTimeZone`	Returns a time zone object for a specified calendar.
`CFCalendarCreateWithIdentifier`	Returns a calendar object for the calendar identified by a calendar identifier.
`CFCalendarDecomposeAbsoluteTime`	Computes the components which are indicated by the componentDesc description string for the given absolute time.
`CFCalendarGetComponentDifference`	Computes the difference between the two absolute times, in terms of specified calendrical components.
`CFCalendarGetFirstWeekday`	Returns the index of first weekday for a specified calendar.
`CFCalendarGetIdentifier`	Returns the given calendar's identifier.

`CFCalendarGetMaximumRangeOfUnit`	Returns the maximum range limits of the values that a specified unit can take on in a given calendar.
`CFCalendarGetMinimumDaysInFirstWeek`	Returns the minimum number of days in the first week of a specified calendar.
`CFCalendarGetMinimumRangeOfUnit`	Returns the minimum range limits of the values that a specified unit can take on in a given calendar.
`CFCalendarGetOrdinalityOfUnit`	Returns the ordinal number of a calendrical unit within a larger unit at a specified absolute time.
`CFCalendarGetRangeOfUnit`	Returns the range of values that one unit can take on within a larger unit during which a specific absolute time occurs.
`CFCalendarGetTypeID`	Returns the type identifier for the CFCalendar opaque type.
`CFCalendarSetFirstWeekday`	Sets the first weekday for a calendar.
`CFCalendarSetLocale`	Sets the locale for a calendar.
`CFCalendarSetMinimumDaysInFirstWeek`	Sets the minimum number of days in the first week of a specified calendar.
`CFCalendarSetTimeZone`	Sets the time zone for a calendar.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFCalendarRef`	A reference to a CFCalendar object.
`CFCalendarUnit`	CFCalendarUnit constants are used to specify calendrical units, such as day or month, in various calendar calculations.
`kCFCalendarComponentsWrap`	Specifies that the components specified for calendar components should be incremented and wrap around to zero/one on overflow, but should not cause higher units to be incremented.
`kCFCalendarUnitDay`	Specifies the day unit.
`kCFCalendarUnitEra`	Specifies the era unit.
`kCFCalendarUnitHour`	Specifies the hour unit.
`kCFCalendarUnitMinute`	Specifies the minute unit.
`kCFCalendarUnitMonth`	Specifies the month unit.
`kCFCalendarUnitSecond`	Specifies the second unit.

kCFCalendarUnitWeek	Specifies the week unit.
kCFCalendarUnitWeekday	Specifies the weekday unit.
kCFCalendarUnitWeekdayOrdinal	Specifies the ordinal weekday unit.
kCFCalendarUnitYear	Specifies the year unit.

CFDateFormatter.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

kCFDateFormatterAMSymbol	Specifies the AM symbol property, a CFString object.
kCFDateFormatterCalendar	Specifies the calendar property, a CFCalendar object.
kCFDateFormatterDefaultDate	Specifies the default date property, a CFDate object.
kCFDateFormatterEraSymbols	Specifies the era symbols property, a CFArray of CFString objects.
kCFDateFormatterMonthSymbols	Specifies the month symbols property, a CFArray of CFString objects.
kCFDateFormatterPMSymbol	Specifies the PM symbol property, a CFString object.
kCFDateFormatterShortMonthSymbols	Specifies the short month symbols property, a CFArray of CFString objects.
kCFDateFormatterShortWeekdaySymbols	Specifies the short weekday symbols property, a CFArray of CFString objects.
kCFDateFormatterTwoDigitStartDate	Specifies the property representing the date from which two-digit years start, a CFDate object.
kCFDateFormatterWeekdaySymbols	Specifies the weekday symbols property, a CFArray of CFString objects.

CFLocale.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFLocaleCopyAvailableLocaleIdentifiers`	Returns an array of CFString objects that represents all locales for which locale data is available.
`CFLocaleCopyDisplayNameForPropertyValue`	Returns the display name for the given value.
`CFLocaleCopyISOCountryCodes`	Returns an array of CFString objects that represents all known legal ISO country codes.
`CFLocaleCopyISOCurrencyCodes`	Returns an array of CFString objects that represents all known legal ISO currency codes.
`CFLocaleCopyISOLanguageCodes`	Returns an array of CFString objects that represents all known legal ISO language codes.
`CFLocaleCreateCanonicalLanguage-IdentifierFromString`	Returns a canonical language identifier by mapping an arbitrary locale identification string to the canonical identifier
`CFLocaleCreateComponentsFromLocaleIdentifier`	Returns a dictionary containing the result from parsing a locale ID consisting of language, script, country, variant, and keyword/value pairs.
`CFLocaleCreateLocaleIdentifierFromComponents`	Returns a locale identifier consisting of language, script, country, variant, and keyword/value pairs derived from a dictionary containing the source information.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFBuddhistCalendar`	Specifies the Buddhist calendar.
`kCFChineseCalendar`	Specifies the Chinese calendar.
`kCFHebrewCalendar`	Specifies the Hebrew calendar.
`kCFIslamicCalendar`	Specifies the Islamic calendar.
`kCFIslamicCivilCalendar`	Specifies the Islamic Civil calendar.
`kCFJapaneseCalendar`	Specifies the Japanese calendar.
`kCFLocaleCalendar`	Specifies the locale calendar.
`kCFLocaleCalendarIdentifier`	Specifies the locale calendar identifier.
`kCFLocaleCollationIdentifier`	Specifies the locale collation identifier.
`kCFLocaleCountryCode`	Specifies the locale country code.

kCFLocaleExemplarCharacterSet	Specifies the locale character set.
kCFLocaleIdentifier	Specifies locale identifier.
kCFLocaleLanguageCode	Specifies the locale language code.
kCFLocaleScriptCode	Specifies the locale script code.
kCFLocaleUsesMetricSystem	Specifies the whether the locale uses the metric system.
kCFLocaleVariantCode	Specifies the locale variant code.

CFNotificationCenter.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

CFNotificationCenterGetDarwinNotifyCenter	Returns the application's Darwin notification center.
CFNotificationCenterGetLocalCenter	Returns the application's local notification center.

CFNumberFormatter.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

kCFNumberFormatterInternationalCurrencySymbol	Specifies the international currency symbol to use when placing a formatted number within a string, a CFString object.
kCFNumberFormatterMultiplier	Specifies the multiplier to use when placing a formatted number within a string, a CFNumber object.
kCFNumberFormatterNegativePrefix	Specifies the minus sign prefix symbol to use when placing a formatted number within a string, a CFString object.
kCFNumberFormatterNegativeSuffix	Specifies the minus sign suffix symbol to use when placing a formatted number within a string, a CFString object.

kCFNumberFormatterPerMillSymbol	Specifies the per mill (1/1000) symbol to use when placing a formatted number within a string, a CFString object.
kCFNumberFormatterPositivePrefix	Specifies the plus sign prefix symbol to use when placing a formatted number within a string, a CFString object.
kCFNumberFormatterPositiveSuffix	Specifies the plus sign suffix symbol to use when placing a formatted number within a string, a CFString object.
kCFNumberFormatterSpellOutStyle	Specifies a spelled out format.

CFString.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

CFStringCreateWithFileSystemRepresentation	Creates a CFString from a zero-terminated POSIX file system representation.
CFStringGetFileSystemRepresentation	Extracts the contents of a string as a NULL-terminated 8-bit string appropriate for passing to POSIX APIs.
CFStringGetMaximumSizeOfFileSystemRepresentation	Determines the upper bound on the number of bytes required to hold the file system representation of the string.
CFStringTransform	Perform in-place transliteration on a mutable string.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

kCFStringEncodingUTF16	An encoding constant that identifies kTextEncodingUnicodeDefault + kUnicodeUTF16Format encoding (alias of kCFStringEncodingUnicode).
kCFStringEncodingUTF16BE	An encoding constant that identifies kTextEncodingUnicodeDefault + kUnicodeUTF16BEFormat encoding. This constant specifies big-endian byte order.

kCFStringEncodingUTF16LE	An encoding constant that identifies kTextEncodingUnicodeDefault + kUnicodeUTF16LEFormat encoding. This constant specifies little-endian byte order.
kCFStringEncodingUTF32	An encoding constant that identifies kTextEncodingUnicodeDefault + kUnicodeUTF32Format encoding.
kCFStringEncodingUTF32BE	An encoding constant that identifies kTextEncodingUnicodeDefault + kUnicodeUTF32BEFormat encoding. This constant specifies big-endian byte order.
kCFStringEncodingUTF32LE	An encoding constant that identifies kTextEncodingUnicodeDefault + kUnicodeUTF32LEFormat encoding. This constant specifies little-endian byte order.
kCFStringTransformFullwidthHalfwidth	The identifier of a reversible transform to convert full-width characters to their half-width equivalents.
kCFStringTransformHiraganaKatakana	The identifier of a reversible transform to transliterate text to Katakana from Hiragana.
kCFStringTransformLatinArabic	The identifier of a reversible transform to transliterate text to Arabic from Latin.
kCFStringTransformLatinCyrillic	The identifier of a reversible transform to transliterate text to Cyrillic from Latin.
kCFStringTransformLatinGreek	The identifier of a reversible transform to transliterate text to Greek from Latin.
kCFStringTransformLatinHangul	The identifier of a reversible transform to transliterate text to Hangul from Latin.
kCFStringTransformLatinHebrew	The identifier of a reversible transform to transliterate text to Hebrew from Latin.
kCFStringTransformLatinHiragana	The identifier of a reversible transform to transliterate text to Hiragana from Latin.
kCFStringTransformLatinKatakana	The identifier of a reversible transform to transliterate text to Katakana from Latin.
kCFStringTransformLatinThai	The identifier of a reversible transform to transliterate text to Thai from Latin.
kCFStringTransformMandarinLatin	The identifier of a reversible transform to transliterate text to Latin from ideographs interpreted as Mandarin Chinese.
kCFStringTransformStripCombiningMarks	The identifier of a transform to strip combining marks (accents or diacritics).

kCFStringTransformToLatin	The identifier of a transform to transliterate all text possible to Latin script. Ideographs are transliterated as Mandarin Chinese.
kCFStringTransformToUnicodeName	The identifier of a reversible transform to transliterate characters other than printable ASCII (minus braces) to their Unicode character name in braces.
kCFStringTransformToXMLHex	The identifier of a reversible transform to transliterate characters other than printable ASCII to XML/HTML numeric entities.

CFStringEncodingExt.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

kCFStringEncodingANSEL	ANSEL (ANSI Z39.47)
kCFStringEncodingBig5_E	Taiwan Big-5E standard
kCFStringEncodingISOLatin10	ISO 8859-16
kCFStringEncodingKOI8_U	RFC 2319, Ukrainian
kCFStringEncodingNextStepJapanese	NextStep Japanese encoding
kCFStringEncodingShiftJIS_X0213_MenKuTen	JIS X0213 in plane-row-column notation
kCFStringEncodingVISCII	RFC 1456, Vietnamese

10.3 Symbol Changes

This article lists the symbols added to `CoreFoundation.framework` in Mac OS X v10.3.

C Symbols

All of the header files with new symbols are listed alphabetically, with their new symbols described.

CFBase.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFCoreFoundationVersionNumber10_1_2`	The Core Foundation framework version in Mac OS X version 10.1.2.
`kCFCoreFoundationVersionNumber10_1_4`	The Core Foundation framework version in Mac OS X version 10.1.4.
`kCFCoreFoundationVersionNumber10_2`	The Core Foundation framework version in Mac OS X version 10.2.

CFCharacterSet.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFCharacterSetCreateCopy`	Creates a new character set with the values from a given character set.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFCharacterSetSymbol`	Symbol character set (Unicode General Category S*).

CFDateFormatter.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFDateFormatterCopyProperty`	Returns a copy of a date formatter's value for a given key.
`CFDateFormatterCreate`	Creates a new CFDateFormatter object, localized to the given locale, which will format dates to the given date and time styles.
`CFDateFormatterCreateDateFromString`	Returns a date object representing a given string.
`CFDateFormatterCreateStringWithAbsoluteTime`	Returns a string representation of the given absolute time using the specified date formatter.
`CFDateFormatterCreateStringWithDate`	Returns a string representation of the given date using the specified date formatter.
`CFDateFormatterGetAbsoluteTimeFromString`	Returns an absolute time object representing a given string.
`CFDateFormatterGetDateStyle`	Returns the date style used to create the given date formatter object.
`CFDateFormatterGetFormat`	Returns a format string for the given date formatter object.
`CFDateFormatterGetLocale`	Returns the locale object used to create the given date formatter object.
`CFDateFormatterGetTimeStyle`	Returns the time style used to create the given date formatter object.
`CFDateFormatterGetTypeID`	Returns the type identifier for CFDateFormatter.
`CFDateFormatterSetFormat`	Sets the format string of the given date formatter to the specified value.
`CFDateFormatterSetProperty`	Sets a date formatter property using a key-value pair.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFDateFormatterRef`	A reference to a CFDateFormatter object.
`CFDateFormatterStyle`	Data type for predefined date and time format styles.
`kCFDateFormatterCalendarName`	Specifies the calendar name, a CFString object.
`kCFDateFormatterDefaultFormat`	The original format string for the formatter (given the date & time style and locale specified at creation).
`kCFDateFormatterFullStyle`	Specifies a full style with complete details, such as "Tuesday, April 12, 1952 AD" or "3:30:42pm PST".
`kCFDateFormatterIsLenient`	Specifies the lenient property, a CFBoolean object where a true value indicates that the parsing of strings into date or absolute time values will be fuzzy.
`kCFDateFormatterLongStyle`	Specifies a long style, typically with full text, such as "November 23, 1937" or "3:30:32pm".
`kCFDateFormatterMediumStyle`	Specifies a medium style, typically with abbreviated text, such as "Nov 23, 1937".
`kCFDateFormatterNoStyle`	Specifies no output.
`kCFDateFormatterShortStyle`	Specifies a short style, typically numeric only, such as "11/23/37" or "3:30pm".
`kCFDateFormatterTimeZone`	Specifies the time zone property, a CFTimeZone object.
`kCFGregorianCalendar`	The name of the calendar currently supported by the kCFDateFormatterCalendarName property.

CFLocale.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFLocaleCopyCurrent`	Returns a copy of the logical locale for the current user.
`CFLocaleCreate`	Creates a locale for the given arbitrary locale identifier.

`CFLocaleCreateCanonicalLocaleIdentifier-FromScriptManagerCodes`	Returns a canonical locale identifier from given language and region codes.
`CFLocaleCreateCanonicalLocaleIdentifierFromString`	Returns a canonical locale identifier by mapping an arbitrary locale identification string to the canonical identifier.
`CFLocaleCreateCopy`	Returns a copy of a locale.
`CFLocaleGetIdentifier`	Returns the given locale's identifier.
`CFLocaleGetSystem`	Returns the root, canonical locale.
`CFLocaleGetTypeID`	Returns the type identifier for the CFLocale opaque type.
`CFLocaleGetValue`	Returns the corresponding value for the given key of a locale's key-value pair.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFLocaleRef`	A reference to a CFLocale object.
`kCFLocaleCurrencyCode`	Specifies the locale currency code.
`kCFLocaleCurrencySymbol`	Specifies the currency symbol.
`kCFLocaleDecimalSeparator`	Specifies the decimal point string.
`kCFLocaleGroupingSeparator`	Specifies the separator string between groups of digits.
`kCFLocaleMeasurementSystem`	Specifies the measurement system used.

CFNotificationCenter.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFNotificationCenterPostNotificationWithOptions`	Posts a notification for an object using specified options.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFNotificationDeliverImmediately`	Delivers the notification immediately.
`kCFNotificationPostToAllSessions`	Delivers the notification to all sessions.

CFNumberFormatter.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFNumberFormatterCopyProperty`	Returns a copy of a number formatter's value for a given key.
`CFNumberFormatterCreate`	Creates a new CFNumberFormatter object, localized to the given locale, which will format numbers to the given style.
`CFNumberFormatterCreateNumberFromString`	Returns a number object representing a given string.
`CFNumberFormatterCreateStringWithNumber`	Returns a string representation of the given number using the specified number formatter.
`CFNumberFormatterCreateStringWithValue`	Returns a string representation of the given number or value using the specified number formatter.
`CFNumberFormatterGetDecimalInfoForCurrencyCode`	Returns the number of fraction digits that should be displayed, and the rounding increment, for a given currency.
`CFNumberFormatterGetFormat`	Returns a format string for the given number formatter object.
`CFNumberFormatterGetLocale`	Returns the locale object used to create the given number formatter object.
`CFNumberFormatterGetStyle`	Returns the number style used to create the given number formatter object.
`CFNumberFormatterGetTypeID`	Returns the type identifier for the CFNumberFormatter opaque type.

CFNumberFormatterGetValueFromString	Returns a number or value representing a given string.
CFNumberFormatterSetFormat	Sets the format string of a number formatter.
CFNumberFormatterSetProperty	Sets a number formatter property using a key-value pair.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

CFNumberFormatterOptionFlags	Type for constants specifying how numbers should be parsed.
CFNumberFormatterPadPosition	Type for constants specifying how numbers should be padded.
CFNumberFormatterRef	A reference to a CFNumberFormatter object.
CFNumberFormatterRoundingMode	These constants are used to specify how numbers should be rounded.
CFNumberFormatterStyle	Type for constants specifying a formatter style.
kCFNumberFormatterAlwaysShowDecimalSeparator	Specifies if the result of converting a value to a string should always contain the decimal separator, even if the number is an integer.
kCFNumberFormatterCurrencyCode	Specifies the currency code, a CFString object.
kCFNumberFormatterCurrencyDecimalSeparator	Specifies the currency decimal separator, a CFString object.
kCFNumberFormatterCurrencyStyle	Specifies a currency style format.
kCFNumberFormatterCurrencySymbol	Specifies the symbol for the currency, a CFString object.
kCFNumberFormatterDecimalSeparator	Specifies the decimal separator, a CFString object.
kCFNumberFormatterDecimalStyle	Specifies a decimal style format.
kCFNumberFormatterDefaultFormat	The original format string for the formatter (given the date and time style and locale specified at creation), a CFString object.
kCFNumberFormatterExponentSymbol	Specifies the exponent symbol (“E” or “e”) in the scientific notation of numbers (for example, as in 1.0e+56), a CFString object.

kCFNumberFormatterFormatWidth	Specifies the width of a formatted number within a string that is either left justified or right justified based on the value of kCFNumberFormatterPaddingPosition, a CFNumber object.
kCFNumberFormatterGroupingSeparator	Specifies the grouping separator, a CFString object.
kCFNumberFormatterGroupingSize	Specifies how often the "thousands" or grouping separator appears, as in "10,000,000", a CFNumber object.
kCFNumberFormatterInfinitySymbol	Specifies the string that is used to represent the symbol for infinity, a CFString object.
kCFNumberFormatterMaxFractionDigits	Specifies the maximum number of digits after a decimal point, a CFNumber object.
kCFNumberFormatterMaxIntegerDigits	Specifies the maximum number of integer digits before a decimal point, a CFNumber object.
kCFNumberFormatterMinFractionDigits	Specifies the minimum number of digits after a decimal point, a CFNumber object.
kCFNumberFormatterMinIntegerDigits	Specifies the minimum number of integer digits before a decimal point, a CFNumber object.
kCFNumberFormatterMinusSign	Specifies the symbol for the minus sign, a CFString object.
kCFNumberFormatterNaNSymbol	Specifies the string that is used to represent NaN ("not a number") when values are converted to strings, a CFString object.
kCFNumberFormatterNoStyle	Specifies no style.
kCFNumberFormatterPadAfterPrefix	Specifies the number of padding characters after the prefix.
kCFNumberFormatterPadAfterSuffix	Specifies the number of padding characters after the suffix.
kCFNumberFormatterPadBeforePrefix	Specifies the number of padding characters before the prefix.
kCFNumberFormatterPadBeforeSuffix	Specifies the number of padding characters before the suffix.
kCFNumberFormatterPaddingCharacter	Specifies the padding character to use when placing a formatted number within a string, a CFString object.

kCFNumberFormatterPaddingPosition	Specifies the position of a formatted number within a string, a CFNumber object.
kCFNumberFormatterParseIntegersOnly	Specifies that only integers should be parsed.
kCFNumberFormatterPercentStyle	Specifies a percent style format.
kCFNumberFormatterPercentSymbol	Specifies the string that is used to represent the percent symbol, a CFString object.
kCFNumberFormatterPlusSign	Specifies the symbol for the plus sign, a CFString object.
kCFNumberFormatterRoundCeiling	Round up to next larger number with the proper number of fraction digits.
kCFNumberFormatterRoundDown	Round down to next larger number with the proper number of fraction digits.
kCFNumberFormatterRoundFloor	Round down to next larger number with the proper number of fraction digits.
kCFNumberFormatterRoundHalfDown	Round down when a 5 follows putative last digit.
kCFNumberFormatterRoundHalfEven	Round the last digit, when followed by a 5, toward an even digit (.25 -> .2, .35 -> .4)
kCFNumberFormatterRoundHalfUp	Round up when a 5 follows putative last digit.
kCFNumberFormatterRoundingIncrement	Specifies a positive rounding increment, or 0.0 to disable rounding, a CFNumber object.
kCFNumberFormatterRoundingMode	Specifies how the last digit is rounded, as when 3.1415926535... is rounded to three decimal places, as in 3.142, a CFNumber object. See “Rounding Modes” for possible values.
kCFNumberFormatterRoundUp	Round up to next larger number with the proper number of fraction digits.
kCFNumberFormatterScientificStyle	Specifies a scientific style format.
kCFNumberFormatterSecondaryGroupingSize	Specifies how often the secondary grouping separator appears, a CFNumber object. See Unicode Technical Standard #35 for more information.
kCFNumberFormatterUseGroupingSeparator	Specifies if the grouping separator should be used, a CFBoolean object.
kCFNumberFormatterZeroSymbol	Specifies the string that is used to represent zero, a CFString object.

CFStream.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFStreamPropertyFileCurrentOffset`	Value is a CFNumber object containing the current file offset.

CFURL.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFURLCreateAbsoluteURLWithBytes`	Creates a new CFURL object by resolving the relative portion of a URL, specified as bytes, against its given base URL.
`CFURLCreateStringByReplacing-PercentEscapesUsingEncoding`	Creates a new string by replacing any percent escape sequences with their character equivalent.
`CFURLGetByteRangeForComponent`	Returns the range of the specified component in the bytes of a URL.
`CFURLGetBytes`	Returns by reference the byte representation of a URL object.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFURLComponentType`	The types of components in a URL.
`kCFURLComponentFragment`	The URL's fragment.
`kCFURLComponentHost`	The URL's host.
`kCFURLComponentNetLocation`	The URL's network location.
`kCFURLComponentParameterString`	The URL's parameter string.
`kCFURLComponentPassword`	The user's password.
`kCFURLComponentPath`	The URL's path component.
`kCFURLComponentPort`	The URL's port.

kCFURLComponentQuery	The URL's query.
kCFURLComponentResourceSpecifier	The URL's resource specifier.
kCFURLComponentScheme	The URL's scheme.
kCFURLComponentUser	The URL's user.
kCFURLComponentUserInfo	The user's information.

CFUserNotification.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

kCFUserNotificationPopUpSelectionKey	The item that was selected from a pop-up menu.

CFXMLParser.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

CFXMLCreateStringByEscapingEntities	Given a CFString object containing XML source with unescaped entities, returns a string with specified XML entities escaped.
CFXMLCreateStringByUnescapingEntities	Given a CFString object containing XML source with escaped entities, returns a string with specified XML entities unescaped.
CFXMLTreeCreateFromDataWithError	Parses the given XML data and returns the resulting CFXMLTree object and any error information.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

kCFXMLTreeErrorDescription	Dictionary key whose value is a CFString containing a readable description of the error.

`kCFXMLTreeErrorLineNumber`	Dictionary key whose value is a CFNumber containing the line number where the error was detected. This may not be the line number where the actual XML error is located.
`kCFXMLTreeErrorLocation`	Dictionary key whose value is a CFNumber containing the byte location where the error was detected.
`kCFXMLTreeErrorStatusCode`	Dictionary key whose value is a CFNumber containing the error status code. See CFXMLParser Reference for possible status code values.

10.2 Symbol Changes

This article lists the symbols added to `CoreFoundation.framework` in Mac OS X v10.2.

C Symbols

All of the header files with new symbols are listed alphabetically, with their new symbols described.

CFBase.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFNullGetTypeID`	Returns the type identifier for the CFNull opaque type.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFNullRef`	A reference to a CFNull object.
`kCFCoreFoundationVersionNumber10_1`	The Core Foundation framework version in Mac OS X version 10.1.
`kCFNull`	The singleton CFNull object.
`NS_NEW_API`	

CFBundle.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFBundleCopyInfoDictionaryForURL`	Returns the information dictionary for a given URL location.
`CFBundleCopyLocalizationsForPreferences`	Given an array of possible localizations and preferred locations, returns the one or more of them that CFBundle would use, without reference to the current application context.
`CFBundleCopyLocalizationsForURL`	Returns an array containing the localizations for a bundle or executable at a particular location.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFCopyLocalizedStringWithDefaultValue`	Returns a localized version of a localization string.
`kCFBundleLocalizationsKey`	Allows an unbundled application that handles localization itself to specify which localizations it has available.

CFCharacterSet.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFCharacterSetCreateInvertedSet`	Creates a new immutable character set that is the invert of the specified character set.
`CFCharacterSetHasMemberInPlane`	Reports whether or not a character set contains at least one member character in the specified plane.
`CFCharacterSetIsLongCharacterMember`	Reports whether or not a given UTF-32 character is in a character set.
`CFCharacterSetIsSupersetOfSet`	Reports whether or not a character set is a superset of another set.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFCharacterSetCapitalizedLetter`	Titlecase character set (Unicode General Category Lt).

CFPreferences.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFPreferencesAppValueIsForced`	Determines whether or not a given key has been imposed on the user.

CFPropertyList.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFPropertyListCreateFromStream`	Creates a property list using data from a stream.
`CFPropertyListIsValid`	Determines if a property list is valid.
`CFPropertyListWriteToStream`	Writes the bytes of a property list serialization out to a stream.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFPropertyListFormat`	Specifies the format of a property list.
`kCFPropertyListBinaryFormat_v1_0`	Binary format version 1.0.
`kCFPropertyListOpenStepFormat`	OpenStep format (use of this format is discouraged).
`kCFPropertyListXMLFormat_v1_0`	XML format version 1.0.

CFSocket.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFSocketDisableCallBacks`	Disables the callback function of a CFSocket object for certain types of socket activity.
`CFSocketEnableCallBacks`	Enables the callback function of a CFSocket object for certain types of socket activity.
`CFSocketGetSocketFlags`	Returns flags that control certain behaviors of a CFSocket object.
`CFSocketSetSocketFlags`	Sets flags that control certain behaviors of a CFSocket object.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFSocketAutomaticallyReenableAcceptCallBack`	When enabled using CFSocketSetSocketFlags, the accept callback is called every time someone connects to your socket.
`kCFSocketAutomaticallyReenableDataCallBack`	When enabled using CFSocketSetSocketFlags, the data callback is called every time the socket has read some data.
`kCFSocketAutomaticallyReenableReadCallBack`	When enabled using CFSocketSetSocketFlags, the read callback is called every time the sockets has data to be read.
`kCFSocketAutomaticallyReenableWriteCallBack`	When enabled using CFSocketSetSocketFlags, the write callback is called every time more data can be written to the socket.
`kCFSocketCloseOnInvalidate`	When enabled using CFSocketSetSocketFlags, the native socket associated with a CFSocket object is closed when the CFSocket object is invalidated. When disabled, the native socket remains open. This option is enabled by default.
`kCFSocketWriteCallBack`	The callback is called when the socket is writable. This callback type may be useful when large amounts of data are being sent rapidly over the socket and you want a notification when there is space in the kernel buffers for more data.

CFStream.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFReadStreamSetProperty`	Sets the value of a property for a stream.
`CFStreamCreatePairWithPeerSocketSignature`	Creates readable and writable streams connected to a socket.
`CFWriteStreamSetProperty`	Sets the value of a property for a stream.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFStreamPropertyAppendToFile`	Value is a CFBoolean value that indicates whether to append the written data to a file, if it already exists, rather than to replace its contents.

CFString.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFStringFindAndReplace`	Replaces all occurrences of a substring within a given range.
`CFStringFindCharacterFromSet`	Query the range of the first character contained in the specified character set.
`CFStringGetRangeOfComposedCharactersAtIndex`	Returns the range of the composed character sequence at a specified index.
`CFStringNormalize`	Normalizes the string into the specified form as described in Unicode Technical Report #15.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFStringNormalizationForm`	Unicode normalization forms as described in Unicode Technical Report #15.
`kCFStringEncodingInvalidId`	Used as a function result to identify an encoding that is not supported or recognized by CFString.
`kCFStringNormalizationFormC`	Canonical decomposition followed by canonical composition.
`kCFStringNormalizationFormD`	Canonical decomposition.
`kCFStringNormalizationFormKC`	Compatibility decomposition followed by canonical composition.
`kCFStringNormalizationFormKD`	Compatibility decomposition.

CFStringEncodingExt.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFStringEncodingBig5_HKSCS_1999`	Big-5 with Hong Kong special char set supplement
`kCFStringEncodingGB_18030_2000`	
`kCFStringEncodingISO_2022_JP_1`	RFC 2237
`kCFStringEncodingISO_2022_JP_3`	JIS X0213
`kCFStringEncodingShiftJIS_X0213_00`	Shift-JIS format encoding of JIS X0213 planes 1 and 2

10.1 Symbol Changes

This article lists the symbols added to `CoreFoundation.framework` in Mac OS X v10.1.

C Symbols

All of the header files with new symbols are listed alphabetically, with their new symbols described.

CFBase.h

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`kCFCoreFoundationVersionNumber`	The current version of the Core Foundation framework.
`kCFCoreFoundationVersionNumber10_0`	The Core Foundation framework version in Mac OS X version 10.0.
`kCFCoreFoundationVersionNumber10_0_3`	The Core Foundation framework version in Mac OS X version 10.0.3.
`TYPE_BOOL`	

CFStream.h

Functions

All of the new functions in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFReadStreamClose`	Closes a readable stream.
`CFReadStreamCopyProperty`	Returns the value of a property for a stream.
`CFReadStreamCreateWithBytesNoCopy`	Creates a readable stream for a block of memory.
`CFReadStreamCreateWithFile`	Creates a readable stream for a file.

CFReadStreamGetBuffer	Returns a pointer to a stream's internal buffer of unread data, if possible.
CFReadStreamGetError	Returns the error status of a stream.
CFReadStreamGetStatus	Returns the current state of a stream.
CFReadStreamGetTypeID	Returns the type identifier the CFReadStream opaque type.
CFReadStreamHasBytesAvailable	Returns a Boolean value that indicates whether a readable stream has data that can be read without blocking.
CFReadStreamOpen	Opens a stream for reading.
CFReadStreamRead	Reads data from a readable stream.
CFReadStreamScheduleWithRunLoop	Schedules a stream into a run loop.
CFReadStreamSetClient	Assigns a client to a stream, which receives callbacks when certain events occur.
CFReadStreamUnscheduleFromRunLoop	Removes a read stream from a given run loop.
CFStreamCreatePairWithSocket	Creates readable and writable streams connected to a socket.
CFStreamCreatePairWithSocketToHost	Creates readable and writable streams connected to a TCP/IP port of a particular host.
CFWriteStreamCanAcceptBytes	Returns whether a writable stream can accept new data without blocking.
CFWriteStreamClose	Closes a writable stream.
CFWriteStreamCopyProperty	Returns the value of a property for a stream.
CFWriteStreamCreateWithAllocatedBuffers	Creates a writable stream for a growable block of memory.
CFWriteStreamCreateWithBuffer	Creates a writable stream for a fixed-size block of memory.
CFWriteStreamCreateWithFile	Creates a writable stream for a file.
CFWriteStreamGetError	Returns the error status of a stream.
CFWriteStreamGetStatus	Returns the current state of a stream.
CFWriteStreamGetTypeID	Returns the type identifier of all CFWriteStream objects.
CFWriteStreamOpen	Opens a stream for writing.
CFWriteStreamScheduleWithRunLoop	Schedules a stream into a run loop.

`CFWriteStreamSetClient`	Assigns a client to a stream, which receives callbacks when certain events occur.
`CFWriteStreamUnscheduleFromRunLoop`	Removes a stream from a particular run loop.
`CFWriteStreamWrite`	Writes data to a writable stream.

Data Types & Constants

All of the new data types and constants in this header file are listed alphabetically, with links to documentation and abstracts, if available.

`CFReadStreamClientCallBack`	Callback invoked when certain types of activity takes place on a readable stream.
`CFReadStreamRef`	A reference to a readable stream object.
`CFStreamClientContext`	A structure that contains program-defined data and callbacks with which you can configure a stream's client behavior.
`CFStreamError`	The structure returned by CFReadStreamGetError and CFWriteStreamGetError.
`CFStreamErrorDomain`	Defines constants for values returned in the domain field of the CFStreamError structure.
`CFStreamEventType`	Defines constants for stream-related events.
`CFStreamStatus`	Constants that describe the status of a stream.
`CFWriteStreamClientCallBack`	Callback invoked when certain types of activity takes place on a writable stream.
`CFWriteStreamRef`	A reference to a writable stream object.
`kCFStreamErrorDomainCustom`	The error code is a custom error code.
`kCFStreamErrorDomainMacOSStatus`	The error is an OSStatus value defined in MacErrors.h.
`kCFStreamErrorDomainPOSIX`	The error code is an error code defined in errno.h.
`kCFStreamEventCanAcceptBytes`	The stream can accept bytes for writing.
`kCFStreamEventEndEncountered`	The end of the stream has been reached.
`kCFStreamEventErrorOccurred`	An error has occurred on the stream.
`kCFStreamEventHasBytesAvailable`	The stream has bytes to be read.
`kCFStreamEventNone`	No event has occurred.
`kCFStreamEventOpenCompleted`	The open has completed successfully.

kCFStreamPropertyDataWritten	Value is a CFData object that contains all the bytes written to a writable memory stream. You cannot modify this value.
kCFStreamPropertySocketNativeHandle	Value is a CFData object that contains the native handle for a socket stream—of type CFSocketNativeHandle—to which the socket stream is connected.
kCFStreamPropertySocketRemoteHostName	Value is a CFString object containing the name of the host to which the socket stream is connected or NULL if unknown.
kCFStreamPropertySocketRemotePortNumber	Value is a CFNumber object containing the remote port number to which the socket stream is connected or NULL if unknown.
kCFStreamStatusAtEnd	There is no more data to read, or no more data can be written.
kCFStreamStatusClosed	The stream is closed.
kCFStreamStatusError	An error occurred on the stream.
kCFStreamStatusNotOpen	The stream is not open for reading or writing.
kCFStreamStatusOpen	The stream is open.
kCFStreamStatusOpening	The stream is being opened for reading or for writing.
kCFStreamStatusReading	The stream is being read from.
kCFStreamStatusWriting	The stream is being written to.

Document Revision History

This table describes the changes to *Core Foundation Reference Update*.

Date	Notes
2007-07-18	Updated for Mac OS X v10.5.
2005-11-09	Corrected article title.
2005-04-29	Added See Also section to Introduction.
	New document that summarizes the symbols added to the Core Foundation framework in Mac OS X v10.4.