Core Services Reference Update

Carbon

2007-07-18

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

Contents

Introduction to Core Services Reference Update

This document summarizes the symbols that have been added to the Core Services 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 Services Framework Reference*.

10.5 Symbol Changes

This article lists the symbols added to `CoreServices.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.

AE

AEDataModel.h

Functions

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

`AECheckIsRecord`	Determines whether a descriptor is truly an AERecord.
`AECoerceDesc`	Coerces the data in a descriptor to another descriptor type and creates a descriptor containing the newly coerced data.
`AECoercePtr`	Coerces data to a desired descriptor type and creates a descriptor containing the newly coerced data.
`AECountItems`	Counts the number of descriptors in a descriptor list.
`AECreateAppleEvent`	Creates an Apple event with several important attributes but no parameters.
`AECreateDesc`	Creates a new descriptor that incorporates the specified data.
`AECreateDescFromExternalPtr`	Creates a new descriptor that uses a memory buffer supplied by the caller.
`AECreateList`	Creates an empty descriptor list or Apple event record.
`AEDeleteItem`	Deletes a descriptor from a descriptor list, causing all subsequent descriptors to move up one place.
`AEDeleteParam`	Deletes a keyword-specified parameter from an Apple event record.
`AEDisposeDesc`	Deallocates the memory used by a descriptor.

AEDuplicateDesc	Creates a copy of a descriptor.
AEFlattenDesc	Flattens the specified descriptor and stores the data in the supplied buffer.
AEGetArray	Extracts data from an Apple event array created with the AEPutArray function and stores it as a standard array of fixed size items in the specified buffer.
AEGetAttributeDesc	Gets a copy of the descriptor for a specified Apple event attribute from an Apple event; typically used when your application needs to pass the descriptor on to another function.
AEGetAttributePtr	Gets a copy of the data for a specified Apple event attribute from an Apple event; typically used when your application needs to work with the data directly.
AEGetCoercionHandler	Gets the coercion handler for a specified descriptor type.
AEGetDescData	Gets the data from the specified descriptor.
AEGetDescDataRange	Retrieves a specified series of bytes from the specified descriptor.
AEGetDescDataSize	Gets the size, in bytes, of the data in the specified descriptor.
AEGetNthDesc	Copies a descriptor from a specified position in a descriptor list into a specified descriptor; typically used when your application needs to pass the extracted data to another function as a descriptor.
AEGetNthPtr	Gets a copy of the data from a descriptor at a specified position in a descriptor list; typically used when your application needs to work with the extracted data directly.
AEGetParamDesc	Gets a copy of the descriptor for a keyword-specified Apple event parameter from an Apple event or an Apple event record.
AEGetParamPtr	Gets a copy of the data for a specified Apple event parameter from an Apple event or an Apple event record.
AEInitializeDesc	Initializes a new descriptor.
AEInitializeDescInline	
AEInstallCoercionHandler	Installs a coercion handler in either the application or system coercion handler dispatch table.
AEPutArray	Inserts the data for an Apple event array into a descriptor list, replacing any previous descriptors in the list.
AEPutAttributeDesc	Adds a descriptor and a keyword to an Apple event as an attribute.
AEPutAttributePtr	Adds a pointer to data, a descriptor type, and a keyword to an Apple event as an attribute.

AEPutDesc	Adds a descriptor to any descriptor list, possibly replacing an existing descriptor in the list.
AEPutParamDesc	Inserts a descriptor and a keyword into an Apple event or Apple event record as an Apple event parameter.
AEPutParamPtr	Inserts data, a descriptor type, and a keyword into an Apple event or Apple event record as an Apple event parameter.
AEPutPtr	Inserts data specified in a buffer into a descriptor list as a descriptor, possibly replacing an existing descriptor in the list.
AERemoveCoercionHandler	Removes a coercion handler from a coercion handler dispatch table.
AEReplaceDescData	Copies the specified data into the specified descriptor, replacing any previous data.
AESizeOfAttribute	Gets the size and descriptor type of an Apple event attribute from a descriptor of type AppleEvent.
AESizeOfFlattenedDesc	Returns the amount of buffer space needed to store the descriptor after flattening it.
AESizeOfNthItem	Gets the data size and descriptor type of the descriptor at a specified position in a descriptor list.
AESizeOfParam	Gets the size and descriptor type of an Apple event parameter from a descriptor of type AERecord or AppleEvent.
AEUnflattenDesc	Unflattens the data in the passed buffer and creates a descriptor from it.
DisposeAECoerceDescUPP	Disposes of a universal procedure pointer to a function that coerces data stored in a descriptor.
DisposeAECoercePtrUPP	Disposes of a universal procedure pointer to a function that coerces data stored in a buffer.
DisposeAEDisposeExternalUPP	Disposes of a universal procedure pointer to a function that disposes of data supplied to the AECreateDescFromExternalPtr function.
DisposeAEEventHandlerUPP	Disposes of a universal procedure pointer to an event handler function.
InvokeAECoerceDescUPP	Calls a universal procedure pointer to a function that coerces data stored in a descriptor.
InvokeAECoercePtrUPP	Calls a universal procedure pointer to a function that coerces data stored in a buffer.
InvokeAEDisposeExternalUPP	Calls a dispose external universal procedure pointer.
InvokeAEEventHandlerUPP	Calls an event handler universal procedure pointer.

NewAECoerceDescUPP	Creates a new universal procedure pointer to a function that coerces data stored in a descriptor.
NewAECoercePtrUPP	Creates a new universal procedure pointer to a function that coerces data stored in a buffer.
NewAEDisposeExternalUPP	Creates a new universal procedure pointer to a function that disposes of data stored in a buffer.
NewAEEventHandlerUPP	Creates a new universal procedure pointer to an event handler function.

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.

AEAddressDesc	A descriptor that contains the address of an application. Typically used to describe the target application for an Apple event.
AEArrayData	Stores array information to be put into a descriptor list with the AEPutArray function or extracted from a descriptor list with the AEGetArray function.
AEArrayDataPointer	
AEArrayType	Stores a value that specifies an array type.
AECoerceDescProcPtr	Defines a pointer to a function that coerces data stored in a descriptor. Your descriptor coercion callback function coerces the data from the passed descriptor to the specified type, returning the coerced data in a second descriptor.
AECoerceDescUPP	Defines a data type for the universal procedure pointer for the AECoerceDescProcPtr callback function pointer.
AECoercePtrProcPtr	Defines a pointer to a function that coerces data stored in a buffer. Your pointer coercion callback routine coerces the data from the passed buffer to the specified type, returning the coerced data in a descriptor.
AECoercePtrUPP	Defines a data type for the universal procedure pointer for the AECoercePtrProcPtr callback function pointer.
AECoercionHandlerUPP	Defines a data type for the universal procedure pointer for the AECoercionHandlerUPP callback function pointer.
AEDataStorage	A pointer to an opaque data type that provides storage for an AEDesc descriptor.
AEDataStorageType	An opaque data type used to store data in Apple event descriptors.

AEDeleteKeyDesc	
AEDesc	Stores data and an accompanying descriptor type to form the basic building block of all Apple Events.
AEDescList	A descriptor whose data consists of a list of one or more descriptors.
AEDescPtr	
AEDisposeExternalProcPtr	Defines a pointer to a function the Apple Event Manager calls to dispose of a descriptor created by the AECreateDescFromExternalPtr function. Your callback function disposes of the buffer you originally passed to that function.
AEDisposeExternalUPP	Defines a universal procedure pointer to a function the Apple Event Manager calls to dispose of a descriptor created by the AECreateDescFromExternalPtr function.
AEEventClass	Specifies the event class of an Apple event.
AEEventHandlerProcPtr	Defines a pointer to a function that handles one or more Apple events. Your Apple event handler function performs any action requested by the Apple event, adds parameters to the reply Apple event if appropriate (possibly including error information), and returns a result code.
AEEventHandlerUPP	Defines a data type for the universal procedure pointer for the AEEventHandlerUPP callback function pointer.
AEEventID	Specifies the event ID of an Apple event.
AEGetKeyDesc	
AEGetKeyPtr	
AEKeyDesc	Associates a keyword with a descriptor to form a keyword-specified descriptor.
AEKeyword	A four-character code that uniquely identifies a descriptor in an Apple event record or an Apple event.
AEPutKeyDesc	
AEPutKeyPtr	
AERecord	A descriptor whose data is a list of keyword-specified descriptors.
AEReturnID	Specifies a return ID for a created Apple event.
AESendMode	Specify send preferences to the AESend function.
AESendPriority	Specifies the processing priority for a sent Apple event.

AESizeOfKeyDesc	
AETransactionID	Specifies a transaction ID.
AppleEvent	A descriptor whose data is a list of descriptors containing both attributes and parameters that make up an Apple event.
AppleEventPtr	
DescType	Specifies the type of the data stored in an AEDesc descriptor.
kAEAlwaysInteract	The user interaction preference—the server application should always interact with the user in response to the Apple event.
kAECanInteract	The user interaction preference—the server application can interact with the user in response to the Apple event.
kAECanSwitchLayer	The application switch preference—if both the client and server allow interaction, and if the client application is the active application on the local computer and is waiting for a reply (that is, it has set the kAEWaitReply flag), AEInteractWithUser brings the server directly to the foreground.
kAEDataArray	Array items consist of data of the same size and same type, and are aligned on word boundaries.
kAEDebugPOSTHeader	
kAEDebugReplyHeader	
kAEDebugXMLDebugAll	
kAEDebugXMLRequest	
kAEDebugXMLResponse	
kAEDefaultTimeout	The timeout value is determined by the Apple Event Manager. The default timeout value is about one minute.
kAEDescArray	Array items consist of descriptors of different descriptor types with data of variable size.
kAEDescListFactorNone	
kAEDescListFactorType	
kAEDescListFactorTypeAndSize	
kAEDontExecute	The execution preference—your application is sending an Apple event to itself for recording purposes only—that is, you want the Apple Event Manager to send a copy of the event to the recording process but you do not want your application actually to receive the event.
kAEDontReconnect	Deprecated and unsupported in Mac OS X.

kAEDontRecord	The recording preference—your application is sending an event to itself but does not want the event recorded.
kAEHandleArray	Array items consist of handles to data of the same type and possibly variable size.
kAEHighPriority	The Apple Event Manager posts the event at the beginning of the event queue of the server process.
kAEHTTPProxyHostAttr	A value of type typeChar or typeUTF8Text.
kAEHTTPProxyPortAttr	A value of type typeSInt32.
kAEKeyDescArray	Array items consist of keyword-specified descriptors with different keywords, different descriptor types, and data of variable size.
kAENeverInteract	The user interaction preference—the server application should never interact with the user in response to the Apple event.
kAENoReply	The reply preference—your application does not want a reply Apple event. If you set the bit specified by this constant, the server processes the Apple event as soon as it has the opportunity.
kAENormalPriority	The Apple Event Manager posts the event at the end of the event queue of the server process and the server processes the Apple event as soon as it has the opportunity.
kAEPackedArray	Array items consist of data of the same size and same type, and are packed without regard for word boundaries.
kAEProcessNonReplyEvents	Allow processing of non-reply Apple events while awaiting a synchronous Apple event reply (you specified kAEWaitReply for the reply preference).
kAEQueueReply	The reply preference—your application wants a reply Apple event. If you set the bit specified by this constant, the reply appears in your event queue as soon as the server has the opportunity to process and respond to your Apple event.
kAERPCClass	
kAESharedScriptHandler	
kAESOAPScheme	
kAESocks4Protocol	
kAESocks5Protocol	
kAESocksHostAttr	
kAESocksPasswordAttr	

kAESocksPortAttr	
kAESocksProxyAttr	
kAESocksUserAttr	
kAEUseHTTPProxyAttr	A value of type typeBoolean. Specifies whether to manually specify the proxy host and port. Defaults to true.
kAEUseSocksAttr	
kAEWaitReply	The reply preference—your application wants a reply Apple event and is willing to give up the processor while waiting for the reply.
kAEWantReceipt	Deprecated and unsupported in Mac OS X.
kAEXMLRPCScheme	
kAnyTransactionID	You pass this value for the transactionID parameter of the AECreateAppleEvent function if the Apple event is not one of a series of interdependent Apple events.
kAutoGenerateReturnID	If you pass this value for the returnID parameter of the AECreateAppleEvent function, the Apple Event Manager assigns to the created Apple event a return ID that is unique to the current session.
keyAcceptTimeoutAttr	
keyAdditionalHTTPHeaders	
keyAddressAttr	Address of a target or client application. See also AEAddressDesc.
keyAEPOSTHeaderData	
keyAEReplyHeaderData	
keyAEXMLReplyData	
keyAEXMLRequestData	
keyDisableAuthenticationAttr	
keyEventClassAttr	Event class of an Apple event. See AEAddressDesc.
keyEventIDAttr	Event ID of an Apple event. See AEAddressDesc.
keyEventSourceAttr	Nature of the source application. (Read only.)
keyInteractLevelAttr	Settings for when to allow the Apple Event Manager to bring a server application to the foreground, if necessary, to interact with the user. See AEAddressDesc. (Read only.)

keyMissedKeywordAttr	Keyword for first required parameter remaining in an Apple event.
keyOptionalKeywordAttr	List of keywords for parameters of an Apple event that should be treated as optional by the target application.
keyOriginalAddressAttr	Address of original source of Apple event if the event has been forwarded (available only in version 1.01 or later versions of the Apple Event Manager). See also AEAddressDesc.
keyReplyRequestedAttr	A Boolean value indicating whether the Apple event expects to be replied to.
keyReturnIDAttr	Return ID for a reply Apple event.
keyRPCMethodName	
keyRPCMethodParam	
keyRPCMethodParamOrder	
keySOAPAction	
keySOAPMethodNameSpace	
keySOAPMethodNameSpaceURI	
keySOAPSchemaVersion	
keySOAPSMDNamespace	
keySOAPSMDNamespaceURI	
keySOAPSMDType	
keySOAPStructureMetaData	
keyTimeoutAttr	Length of time, in ticks, that the client will wait for a reply or a result from the server.
keyTransactionIDAttr	Transaction ID identifying a series of Apple events that are part of one transaction.
keyUserNameAttr	
keyUserPasswordAttr	
keyXMLDebuggingAttr	
kNoTimeOut	Your application is willing to wait indefinitely. Most commonly, you instead provide a timeout value (in ticks) that will provide a reasonable amount of time for the current operation.
kSOAP1999Schema	
kSOAP2001Schema	

typeLongFloatingPoint	128-bit floating point value.
typeAEList	List of descriptors.
typeAERecord	List of keyword-specified descriptors.
typeAlias	Alias.
typeAppleEvent	Apple event.
typeApplicationBundleID	Indicates a descriptor containing UTF-8 characters that specify the bundle ID of an application. Bundle IDs should be constructed similarly to "com.company.directorylocation.ApplicationName".
typeApplicationURL	For specifying an application by URL. See Discussion section below for important information.
typeApplSignature	Application signature.
typeAppParameters	Process Manager launch parameters.
typeBoolean	Boolean value—single byte with value 0 or 1.
typeCFArrayRef	A Core Foundation array.
typeCFAttributedStringRef	A Core Foundation attributed string.
typeCFBooleanRef	A Core Foundation Boolean value.
typeCFDictionaryRef	A Core Foundation dictionary.
typeCFMutableArrayRef	A Core Foundation mutable array.
typeCFMutableAttributedStringRef	A Core Foundation mutable attributed string.
typeCFMutableDictionaryRef	A Core Foundation mutable dictionary.
typeCFMutableStringRef	A Core Foundation mutable string.
typeCFNumberRef	A Core Foundation number.
typeCFStringRef	A Core Foundation string.
typeCFTypeRef	A Core Foundation type.
typeChar	Unterminated string of system script characters.
typeComp	Standard Apple Numerics Environment (SANE) comparison operator.
typeCString	C string—Mac OS Roman characters followed by a NULL byte. Deprecated.
typeDecimalStruct	Decimal.

typeEncodedString	Styled Unicode text. Not implemented.
typeEnumerated	Enumerated data.
typeEventRecord	
typeExtended	SANE extended.
typeFalse	FALSE Boolean value.
typeFileURL	A file URL. That is, the associated data consists of the bytes of a UTF-8 encoded URL with a scheme of "file". This type is appropriate for describing a file that may not yet exist—see Technical Note 2022 for more information.
typeFixed	
typeFloat	SANE double.
typeFSRef	File system reference. Use in preference to file system specifications (typeFSS).
typeFSS	File system specification. Deprecated in Mac OS X. Use file system references (typeFSRef) instead.
typeIEEE32BitFloatingPoint	32-bit floating point value.
typeIEEE64BitFloatingPoint	64-bit floating point value.
typeInteger	32-bit integer.
typeKernelProcessID	Indicates a descriptor containing a UNIX process ID. A process ID is similar to a PSN (processor serial number) but does not require a Process Manager connection. It is analogous to a 32-bit unsigned integer.
typeKeyword	Apple event keyword.
typeLongFloat	SANE double.
typeLongInteger	32-bit integer.
typeMachPort	Indicates a descriptor that specifies a Mach port.
typeMagnitude	Unsigned 32-bit integer.
typeNull	A null data storage pointer. When resolving an object specifier, an object with a null storage pointer specifies the default container at the top of the container hierarchy.
typeProcessSerialNumber	A process serial number. See also AEAddressDesc.
typeProperty	Apple event object property.
typePString	Pascal string—unsigned length byte followed by Mac OS Roman characters. Deprecated.

typeQDRectangle	
typeSectionH	Handle to a section record. (Deprecated.)
typeShortFloat	SANE single.
typeShortInteger	16-bit integer.
typeSInt16	16-bit signed integer.
typeSInt32	32-bit signed integer.
typeSInt64	64-bit signed integer.
typeSMFloat	SANE single.
typeSMInt	16-bit integer.
typeStyledUnicodeText	Styled Unicode text. Not implemented.
typeTrue	TRUE Boolean value.
typeType	Four-character code for event class or event ID
typeUInt16	16-bit unsigned integer.
typeUInt32	32-bit unsigned integer.
typeUInt64	64-bit unsigned integer.
typeUnicodeText	Unicode text. Native byte ordering, optional BOM.
typeUTF16ExternalRepresentation	Unicode text in 16-bit external representation with byte-order-mark (BOM).
typeUTF8Text	8-bit Unicode (UTF-8 encoding).
typeWildCard	Matches any type.

AEHelpers.h

Functions

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

AEBuildAppleEvent	Constructs an entire Apple event in a single call.
AEBuildDesc	Provides a facility for compiling AEBuild descriptor strings into Apple event descriptors (AEDesc).
AEBuildParameters	Adds additional parameters or attributes to an existing Apple event.

AEPrintDescToHandle	Provides a pretty printer facility for displaying the contents of Apple event descriptors.
AEStreamClose	Closes and deallocates an AEStreamRef.
AEStreamCloseDesc	Marks the end of a descriptor in an AEStreamRef.
AEStreamCloseList	Marks the end of a list of descriptors in an AEStreamRef.
AEStreamCloseRecord	Marks the end of a record in an AEStreamRef.
AEStreamCreateEvent	Creates a new Apple event and opens a stream for writing data to it.
AEStreamOpen	Opens a new AEStreamRef for use in building a descriptor.
AEStreamOpenDesc	Marks the beginning of a descriptor in an AEStreamRef.
AEStreamOpenEvent	Opens a stream for an existing Apple event.
AEStreamOpenKeyDesc	Marks the beginning of a key descriptor in an AEStreamRef.
AEStreamOpenList	Marks the beginning of a descriptor list in an AEStreamRef.
AEStreamOpenRecord	Marks the beginning of an Apple event record in an AEStreamRef.
AEStreamOptionalParam	Designates a parameter in an Apple event as optional.
AEStreamSetRecordType	Sets the type of the most recently created record in an AEStreamRef.
AEStreamWriteAEDesc	Copies an existing descriptor into an AEStreamRef.
AEStreamWriteData	Appends data to the current descriptor in an AEStreamRef.
AEStreamWriteDesc	Appends the data for a complete descriptor to an AEStreamRef.
AEStreamWriteKey	Marks the beginning of a keyword/descriptor pair for a descriptor in an AEStreamRef.
AEStreamWriteKeyDesc	Writes a complete keyword/descriptor pair to an AEStreamRef.
vAEBuildAppleEvent	Allows you to encapsulate calls to AEBuildAppleEvent in a wrapper routine.
vAEBuildDesc	Allows you to encapsulate calls to AEBuildDesc in your own wrapper routines.
vAEBuildParameters	Allows you to encapsulate calls to AEBuildParameters in your own stdarg-style wrapper routines, using techniques similar to those allowed by vsprintf.

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.

AEBuildError	Defines a structure for storing additional error code information for “AEBuild” routines.
AEBuildErrorCode	Represents syntax errors found by an “AEBuild” routine.
aeBuildSyntaxBadData	Bad data was found inside a variable argument list.
aeBuildSyntaxBadDesc	An illegal descriptor was specified.
aeBuildSyntaxBadEOF	An unexpected end of format string was encountered.
aeBuildSyntaxBadHex	A hex string contained characters other than hexadecimal digits.
aeBuildSyntaxBadNegative	A minus sign “-” was not followed by digits.
aeBuildSyntaxBadToken	An illegal character was specified.
aeBuildSyntaxCoercedList	Cannot coerce a list.
aeBuildSyntaxMissingQuote	A string was not terminated by a closing quotation mark.
aeBuildSyntaxNoCloseBrace	A comma or closing brace “}” was expected.
aeBuildSyntaxNoCloseBracket	A comma or closing bracket “]” was expected.
aeBuildSyntaxNoCloseHex	A hex string was missing a “$” or “»” character.
aeBuildSyntaxNoCloseParen	A data value was missing a closing parenthesis.
aeBuildSyntaxNoCloseString	A string was missing a closing quote.
aeBuildSyntaxNoColon	In a descriptor, one of the keywords was not followed by a colon.
aeBuildSyntaxNoEOF	There were unexpected characters beyond the end of the format string.
aeBuildSyntaxNoErr	No error.
aeBuildSyntaxNoKey	A keyword was missing from a descriptor.
aeBuildSyntaxOddHex	A hex string contained an odd number of digits.
aeBuildSyntaxUncoercedDoubleAt	You must coerce a “@@” substitution.
aeBuildSyntaxUncoercedHex	A hex string must be coerced to a type.
AEStreamRef	An opaque data structure for storing stream-based descriptor data.

AEMach.h

Functions

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

`AEDecodeMessage`	Decodes a Mach message and converts it into an Apple event and its related reply.
`AEGetRegisteredMachPort`	Returns the Mach port (in the form of a mach_port_t) that was registered with the bootstrap server for this process.
`AEProcessMessage`	Decodes and dispatches a low level Mach message event to an event handler, including packaging and returning the reply to the sender.
`AESendMessage`	Sends an AppleEvent to a target process without some of the overhead required by AESend.

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.

`keyReplyPortAttr`	
`typeReplyPortAttr`	

AEObjects.h

Functions

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

`AECallObjectAccessor`	Invokes the appropriate object accessor function for a specific desired type and container type.
`AEDisposeToken`	Deallocates the memory used by a token.
`AEGetObjectAccessor`	Gets an object accessor function from an object accessor dispatch table.
`AEInstallObjectAccessor`	Adds or replaces an entry for an object accessor function to an object accessor dispatch table.
`AEObjectInit`	Initializes the Object Support Library.
`AERemoveObjectAccessor`	Removes an object accessor function from an object accessor dispatch table.

AEResolve	Resolves an object specifier.
AESetObjectCallbacks	Specifies the object callback functions for your application.
DisposeOSLAccessorUPP	Disposes of a universal procedure pointer to an object accessor function.
DisposeOSLAdjustMarksUPP	Disposes of a universal procedure pointer to an object callback adjust marks function.
DisposeOSLCompareUPP	Disposes of a universal procedure pointer to an object callback comparison function.
DisposeOSLCountUPP	Disposes of a universal procedure pointer to an object callback count function.
DisposeOSLDisposeTokenUPP	Disposes of a universal procedure pointer to an object callback dispose token function.
DisposeOSLGetErrDescUPP	Disposes of a universal procedure pointer to an object callback get error descriptor function.
DisposeOSLGetMarkTokenUPP	Disposes of a universal procedure pointer to an object callback get mark function.
DisposeOSLMarkUPP	Disposes of a universal procedure pointer to an object callback mark function.
InvokeOSLAccessorUPP	Calls an object accessor universal procedure pointer.
InvokeOSLAdjustMarksUPP	Calls an object callback adjust marks universal procedure pointer.
InvokeOSLCompareUPP	Calls an object callback comparison universal procedure pointer.
InvokeOSLCountUPP	Calls an object callback count universal procedure pointer.
InvokeOSLDisposeTokenUPP	Calls an object callback dispose token universal procedure pointer.
InvokeOSLGetErrDescUPP	Calls an object callback get error descriptor universal procedure pointer.
InvokeOSLGetMarkTokenUPP	Calls an object callback get mark universal procedure pointer.
InvokeOSLMarkUPP	Calls an object callback mark universal procedure pointer.
NewOSLAccessorUPP	Creates a new universal procedure pointer to an object accessor function.
NewOSLAdjustMarksUPP	Creates a new universal procedure pointer to an object callback adjust marks function.
NewOSLCompareUPP	Creates a new universal procedure pointer to an object callback comparison function.

NewOSLCountUPP	Creates a new universal procedure pointer to an object callback count function.
NewOSLDisposeTokenUPP	Creates a new universal procedure pointer to an object callback dispose token function.
NewOSLGetErrDescUPP	Creates a new universal procedure pointer to an object callback get error descriptor function.
NewOSLGetMarkTokenUPP	Creates a new universal procedure pointer to an object callback get mark function.
NewOSLMarkUPP	Creates a new universal procedure pointer to an object callback mark function.

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.

ccntTokenRecHandle	
ccntTokenRecord	Stores token information used by the AEResolve function while locating a range of objects.
ccntTokenRecPtr	
formAbsolutePosition	An integer or other constant indicating the position of one or more elements in relation to the beginning or end of their container.
formName	Specifies the Apple event object by name.
formPropertyID	Specifies the property ID for an element's property.
formRange	Specifies a group of elements between two other elements.
formRelativePosition	Specifies an element position either immediately before or immediately after a container, not inside it.
formTest	Specifies a test.
formUniqueID	Specifies a value that uniquely identifies an object within its container or across an application.
formWhose	Specifies a container of one or more objects and a test to perform on the objects.
kAEAll	Specifies all the elements in the container.
kAEAND	Specifies a logical AND operation.
kAEAny	Specifies a single element chosen at random from the container.
kAEFirst	The first element in the specified container.

kAEHandleSimpleRanges	
kAEIDoMarking	The application provides marking callback functions. Marking callback functions are described in “Object Callback Functions.”
kAEIDoMinimum	The application does not handle whose tests or provide marking callbacks.
kAEIDoWhose	The application supports whose tests (supports key form formWhose).
kAELast	Specifies the last element in the container.
kAEMiddle	Specifies the middle element in the container.
kAENext	Specifies the Apple event object after the container.
kAENOT	Specifies a logical NOT operation.
kAEOR	Specifies a logical OR operation.
kAEPassSubDescs	
kAEPrevious	Specifies the Apple event object before the container.
kAEResolveNestedLists	
kAEUseRelativeIterators	
keyAEAdjustMarksProc	Mark-adjusting function. See OSLAdjustMarksProcPtr.
keyAECompareProc	Object-comparison function. See OSLCompareProcPtr.
keyAECompOperator	Specifies a descriptor of typeType, whose data consists of one of the constant values described in “Key Form and Descriptor Type Object Specifier Constants.”
keyAEContainer	Specifies the container for the requested object or objects. The data is an object specifier (or in some cases a null descriptor).
keyAECountProc	Object-counting function. See OSLCountProcPtr.
keyAEDesiredClass	A four-character code that identifies the object class of the specified object or objects.
keyAEGetErrDescProc	Get error descriptor callback function. See OSLGetErrDescProcPtr.
keyAEIndex	
keyAEKeyData	Data or nested descriptors that specify a property, name, position, range, or test, depending on the key form.
keyAEKeyForm	A four-character code that identifies the key form for the specified object or objects.

keyAELogicalOperator	Specifies a descriptor of type typeEnumerated whose data is one of the logical operators (such as kAEAND) defined in "Key Form and Descriptor Type Object Specifier Constants."
keyAELogicalTerms	Specifies a descriptor of type typeAEList containing one or more comparison or logical descriptors.
keyAEMarkProc	Object-marking function. See OSLMarkProcPtr.
keyAEMarkTokenProc	Mark token function. See OSLGetMarkTokenProcPtr.
keyAEObject1	Identifies a descriptor for the element that is currently being compared to the object or data specified by the descriptor for the keyword keyAEObject2.
keyAEObject2	Identifies a descriptor for the element that is currently being compared to the object or data specified by the descriptor for the keyword keyAEObject1.
keyAERangeStart	Specifies the first Apple event object in a desired range.
keyAERangeStop	Specifies the last Apple event object in the desired range.
keyAETest	
keyAEWhoseRangeStart	
keyAEWhoseRangeStop	
keyDisposeTokenProc	Token disposal function. See OSLDisposeTokenProcPtr.
OSLAccessorProcPtr	Your object accessor function either finds elements or properties of an Apple event object.
OSLAccessorUPP	Defines a data type for the universal procedure pointer for the OSLAccessorProcPtr callback function pointer.
OSLAdjustMarksProcPtr	Defines a pointer to an adjust marks callback function. Your adjust marks function unmarks objects previously marked by a call to your marking function.
OSLAdjustMarksUPP	Defines a data type for the universal procedure pointer for the OSLAdjustMarksProcPtr callback function pointer.
OSLCompareProcPtr	Defines a pointer to an object comparison callback function. Your object comparison function compares one Apple event object to another or to the data for a descriptor.
OSLCompareUPP	Defines a data type for the universal procedure pointer for the OSLCompareProcPtr callback function pointer.
OSLCountProcPtr	Defines a pointer to an object counting callback function. Your object counting function counts the number of Apple event objects of a specified class in a specified container object.

`OSLCountUPP`	Defines a data type for the universal procedure pointer for the OSLCountProcPtr callback function pointer.
`OSLDisposeTokenProcPtr`	Defines a pointer to a dispose token callback function. Your dispose token function, required only if you use a complex token format, disposes of the specified token.
`OSLDisposeTokenUPP`	Defines a data type for the universal procedure pointer for the OSLDisposeTokenProcPtr callback function pointer.
`OSLGetErrDescProcPtr`	Defines a pointer to an error descriptor callback function. Your error descriptor callback function supplies a pointer to an address where the Apple Event Manager can store the current descriptor if an error occurs during a call to the AEResolve function.
`OSLGetErrDescUPP`	Defines a data type for the universal procedure pointer for the OSLGetErrDescProcPtr callback function pointer.
`OSLGetMarkTokenProcPtr`	Defines a pointer to a mark token callback function. Your mark token function returns a mark token.
`OSLGetMarkTokenUPP`	Defines a data type for the universal procedure pointer for the OSLGetMarkTokenProcPtr callback function pointer.
`OSLMarkProcPtr`	Defines a pointer to an object marking callback function. Your object-marking function marks a specific Apple event object.
`OSLMarkUPP`	Defines a data type for the universal procedure pointer for the OSLMarkProcPtr callback function pointer.
`typeAbsoluteOrdinal`	Specifies a descriptor whose data consists of one of the constants kAEFirst, kAEMiddle, kAELast, kAEAny, or kAEAll, which are described in AEDisposeToken. Used with formAbsolutePosition.
`typeCompDescriptor`	Specifies a comparison descriptor. Data is one of the constants described in AEDisposeToken.
`typeCurrentContainer`	Specifies a container for an element that demarcates one boundary in a range. The descriptor has a null data storage pointer. This descriptor type is used only with formRange.
`typeIndexDescriptor`	Specifies a descriptor whose data indicates an indexed position within a range of values.
`typeLogicalDescriptor`	Specifies a logical descriptor. Data is one of the constants described in AEDisposeToken.
`typeObjectBeingExamined`	Specifies a descriptor that acts as a placeholder for each of the successive elements in a container when the Apple Event Manager tests those elements one at a time.
`typeObjectSpecifier`	Specifies a descriptor used with the keyAEContainer keyword in a keyword-specified descriptor. The key data for the descriptor is an object specifier.

typeOSLTokenList	Specifies a descriptor whose data consists of a list of tokens. (Token is defined in AEDisposeToken.)
typeRangeDescriptor	Specifies a range descriptor that identifies two Apple event objects marking the beginning and end of a range of elements.
typeRelativeDescriptor	Specifies a descriptor whose data consists of one of the constants kAENext or kAEPrevious, which are described in AEDisposeToken. Used with formRelativePosition.
typeToken	Specifies a descriptor whose data storage pointer refers to a structure of type AEDisposeToken.
typeWhoseDescriptor	
typeWhoseRange	

AEPackObject.h

Functions

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

CreateCompDescriptor	Creates a comparison descriptor that specifies how to compare one or more Apple event objects with either another Apple event object or a descriptor.
CreateLogicalDescriptor	Creates a logical descriptor that specifies a logical operator and one or more logical terms for the Apple Event Manager to evaluate.
CreateObjSpecifier	Assembles an object specifier that identifies one or more Apple event objects, from other descriptors.
CreateOffsetDescriptor	Creates an offset descriptor that specifies the position of an element in relation to the beginning or end of its container.
CreateRangeDescriptor	Creates a range descriptor that specifies a series of consecutive elements in the same container.

AERegistry.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.

cADBAddress	
cAddressSpec	

cAEList	
cAppleTalkAddress	
cApplication	
cArc	
cBoolean	
cBusAddress	
cCell	
cChar	
cColorTable	
cColumn	
cDevSpec	
cDocument	
cDrawingArea	
cEnumeration	
cEthernetAddress	
cFile	
cFireWireAddress	
cFixed	
cFixedPoint	
cFixedRectangle	
cFTPItem	Specifies FTP (File Transfer Protocol) protocol.
cGraphicLine	
cGraphicObject	
cGraphicShape	
cGraphicText	
cGroupedGraphic	
cHTML	Specifies HTML (HyperText Markup Language) format.

cInsertionLoc	
cInsertionPoint	
cInternetAddress	Specifies an Internet or Intranet address for the TCP/IP protocol.
cIntlText	
cIntlWritingCode	
cIPAddress	
cItem	
cKeystroke	
cLine	
cLocalTalkAddress	
cLongDateTime	
cLongFixed	
cLongFixedPoint	
cLongFixedRectangle	
cLongInteger	
cLongPoint	
cLongRectangle	
cMachineLoc	
cMenu	
cMenuItem	
cObject	
cObjectSpecifier	
cOpenableObject	
cOval	
cParagraph	A paragraph of text.
cPICT	A PICT format figure.
cPixel	
cPixelMap	

cPolygon	
cProperty	A property of any object class.
cQDPoint	
cQDRectangle	
cRectangle	
cRGBColor	An RGB color value.
cRotation	
cRoundedRectangle	
cRow	
cSCSIAddress	
cSelection	
cShortInteger	
cTable	
cText	
cTextFlow	
cTextStyles	
cTokenRingAddress	
cType	
cURL	Specifies a Uniform Resource Locator or Uniform Resource ID (URI).
cUSBAddress	
cVersion	
cWindow	
cWord	
eADB	
eAddressSpec	
eAnalogAudio	
eAppleTalk	

eAudioLineIn	
eAudioLineOut	
eAudioOut	
eBus	
eCapsLockDown	
eCDROM	
eClearKey	
eCommandDown	
eCommSlot	
eConduit	
eControlDown	
eDeleteKey	
eDeviceType	
eDigitalAudio	
eDisplay	
eDownArrowKey	
eDVD	
eEndKey	
eEnterKey	
eEscapeKey	
eEthernet	
eF10Key	
eF11Key	
eF12Key	
eF13Key	
eF14Key	
eF15Key	
eF1Key	

eF2Key	
eF3Key	
eF4Key	
eF5Key	
eF6Key	
eF7Key	
eF8Key	
eF9Key	
eFireWire	
eFloppy	
eForwardDelKey	
eHD	
eHelpKey	
eHomeKey	
eInfrared	
eIP	
eIrDA	
eIRTalk	
eKeyboard	
eKeyKind	
eLCD	
eLeftArrowKey	
eLocalTalk	
eMacIP	
eMacVideo	
eMicrophone	
eModem	
eModemPort	

eModemPrinterPort	
eModifiers	
eMonitorOut	
eMouse	
eNuBus	
eNuBusCard	
enumArrows	
enumJustification	
enumKeyForm	
enumPosition	
enumProtection	
enumQuality	
enumSaveOptions	
enumStyle	
enumTransferMode	
eOptionDown	
ePageDownKey	
ePageUpKey	
ePCcard	
ePCIbus	
ePCIcard	
ePDScard	
ePDSslot	
ePointingDevice	
ePostScript	
ePPP	
ePrinter	
ePrinterPort	

eProtocol	
eReturnKey	
eRightArrowKey	
eScheme	
eSCSI	
eSerial	
eShiftDown	
eSpeakers	
eStorageDevice	
eSVGA	
eSvideo	
eTabKey	
eTokenRing	
eTrackball	
eTrackpad	
eUpArrowKey	
eurlAFP	
eurlAT	
eurlEPPC	
eurlFile	
eurlFTP	
eurlGopher	
eurlHTTP	
eurlHTTPS	
eurlIMAP	
eurlLaunch	
eurlLDAP	
eurlMail	

eurlMailbox	
eurlMessage	
eurlMulti	
eurlNews	
eurlNFS	
eurlNNTP	
eurlPOP	
eurlRTSP	
eurlSNews	
eurlTelnet	
eurlUnknown	
eUSB	
eVideoIn	
eVideoMonitor	
eVideoOut	
IntlText	International text consists of an ordered series of bytes, beginning with a 4-byte language code and a 4-byte script code that together determine the format of the bytes that follow.
kAEAbout	
kAEActivate	
kAEAfter	
kAEAliasSelection	
kAEAllCaps	
kAEApplicationClass	
kAEArrowAtEnd	
kAEArrowAtStart	
kAEArrowBothEnds	
kAEAsk	

kAEAutoDown	
kAEBefore	
kAEBeginning	
kAEBeginsWith	The value of operand1 begins with the value of operand2 (for example, the string "operand" begins with the string "opera").
kAEBeginTransaction	
kAEBold	
kAECaseSensEquals	
kAECentered	
kAEChangeView	
kAEClone	
kAEClose	
kAECommandClass	
kAECondensed	
kAEContains	The value of operand1 contains the value of operand2 (for example, the string "operand" contains the string "era").
kAECopy	
kAECoreSuite	An Apple event in the Standard Suite.
kAECountElements	
kAECreateElement	
kAECreatePublisher	
kAECut	
kAEDeactivate	
kAEDelete	
kAEDiskEvent	
kAEDoObjectsExist	
kAEDoScript	

kAEDown	
kAEDrag	
kAEDuplicateSelection	
kAEEditGraphic	
kAEEmptyTrash	
kAEEnd	
kAEEndsWith	The value of operand1 ends with the value of operand2 (for example, the string "operand" ends with the string "and").
kAEEndTransaction	
kAEEquals	The value of operand1 is equal to the value of operand2
kAEExpanded	
kAEFast	
kAEFinderEvents	An event that the Finder accepts.
kAEFormulaProtect	
kAEFullyJustified	
kAEGetClassInfo	
kAEGetData	
kAEGetDataSize	
kAEGetEventInfo	
kAEGetInfoSelection	
kAEGetPrivilegeSelection	
kAEGetSuiteInfo	
kAEGreaterThan	The value of operand1 is greater than the value of operand2.
kAEGreaterThanEquals	The value of operand1 is greater than or equal to the value of operand2.
kAEGrow	
kAEHidden	

kAEHighLevel	
kAEHiQuality	
kAEImageGraphic	
kAEInfo	
kAEInternetSuite	
kAEISAction	
kAEISActionPath	
kAEISClientAddress	
kAEISClientIP	
kAEISContentType	
kAEISFromUser	
kAEISFullRequest	
kAEISGetURL	
KAEISHandleCGI	
kAEISHTTPSearchArgs	
kAEISMethod	
kAEISPassword	
kAEISPostArgs	
kAEISReferrer	
kAEISScriptName	
kAEISServerName	
kAEISServerPort	
kAEIsUniform	
kAEISUserAgent	
kAEISUserName	
kAEISWebStarSuite	
kAEItalic	
kAEKeyClass	

kAEKeyDown	
kAELeftJustified	
kAELessThan	
kAELessThanEquals	The value of operand1 is less than or equal to the value of operand2.
kAELogOut	
kAELowercase	
kAEMain	
kAEMakeObjectsVisible	
kAEMenuClass	
kAEMenuSelect	
kAEMiscStandards	
kAEModifiable	
kAEMouseClass	
kAEMouseDown	
kAEMouseDownInBack	
kAEMove	
kAEMoved	
kAENavigationKey	
kAENo	
kAENoArrow	
kAENonmodifiable	
kAENullEvent	
kAEOpen	
kAEOpenSelection	
kAEOutline	
kAEPageSetup	
kAEPaste	
kAEPlain	

kAEPrint	
kAEPrintSelection	
kAEPrintWindow	
kAEPromise	
kAEPutAwaySelection	
kAEQDAddOver	
kAEQDAddPin	
kAEQDAdMax	
kAEQDAdMin	
kAEQDBic	
kAEQDBlend	
kAEQDCopy	
kAEQDNotBic	
kAEQDNotCopy	
kAEQDNotOr	
kAEQDNotXor	
kAEQDOr	
kAEQDSubOver	
kAEQDSubPin	
kAEQDSupplementalSuite	
kAEQDXor	
kAEQuickdrawSuite	
kAEQuitAll	
kAERawKey	
kAEReallyLogOut	
kAERedo	
kAERegular	

kAEReopenApplication	Event that reopens an application. Sent, for example, when your application is running and a user clicks your application icon in the Dock.
kAEReplace	
kAERequiredSuite	
kAEResized	
kAERestart	
kAEResume	
kAERevealSelection	
kAERevert	
kAERightJustified	
kAESave	
kAEScrapEvent	
kAESelect	
kAESetData	
kAESetPosition	
kAEShadow	
kAESharing	
kAEShowClipboard	
kAEShowRestartDialog	
kAEShowShutdownDialog	
kAEShutDown	
kAESleep	
kAESmallCaps	
kAESpecialClassProperties	
kAEStoppedMoving	
kAEStrikethrough	
kAESubscript	

kAESuperscript	
kAESuspend	
kAETableSuite	
kAETextSuite	
kAETransactionTerminated	
kAEUnderline	
kAEUndo	
kAEUp	
kAEUpdate	
kAEVirtualKey	
kAEWakeUpEvent	
kAEWholeWordEquals	
kAEWindowClass	
kAEYes	
kAEZoom	
kAEZoomIn	
kAEZoomOut	
kByCommentView	
kByDateView	
kByIconView	
kByKindView	
kByLabelView	
kByNameView	
kBySizeView	
kBySmallIcon	
kByVersionView	
kConnSuite	
kDoFolderActionEvent	

keyAEAngle	
keyAEArcAngle	
keyAEBaseAddr	
keyAEBestType	
keyAEBgndColor	
keyAEBgndPattern	
keyAEBounds	
keyAEBufferSize	
keyAECellList	
keyAEClassID	
keyAEClauseOffsets	
keyAEColor	
keyAEColorTable	
keyAECurrentPoint	
keyAECurveHeight	
keyAECurveWidth	
keyAEDashStyle	
keyAEData	
keyAEDefaultType	
keyAEDefinitionRect	
keyAEDescType	
keyAEDestination	
keyAEDoAntiAlias	
keyAEDoDithered	
keyAEDoRotate	
keyAEDoScale	
keyAEDoTranslate	
keyAEDragging	

keyAEEditionFileLoc	
keyAEElements	
keyAEEndPoint	
keyAEEventClass	
keyAEEventID	
keyAEFile	
keyAEFileType	
keyAEFillColor	
keyAEFillPattern	
keyAEFixLength	
keyAEFlipHorizontal	
keyAEFlipVertical	
keyAEFont	
keyAEFormula	
keyAEGraphicObjects	
keyAEHiliteRange	
keyAEID	
keyAEImageQuality	
keyAEInsertHere	
keyAEKeyForms	
keyAEKeyword	
keyAELaunchedAsLogInItem	If present in a kAEOpenApplication event, the receiving application was launched as a login item and should only perform actions suitable to that environment—for example, it probably shouldn't open an untitled document.

keyAELaunchedAsServiceItem	If present in a kAEOpenApplication event, the receiving application was launched as a service item and should only perform actions suitable to that environment—for example, it probably shouldn't open an untitled document.
keyAELeftSide	
keyAELevel	
keyAELineArrow	
keyAEMoveView	
keyAEName	
keyAENewElementLoc	
keyAENextBody	
keyAEObject	
keyAEObjectClass	
keyAEOffset	
keyAEOffStyles	
keyAEOnStyles	
keyAEParameters	
keyAEParamFlags	
keyAEPenColor	
keyAEPenPattern	
keyAEPenWidth	
keyAEPinRange	
keyAEPixelDepth	
keyAEPixMapMinus	
keyAEPMTable	
keyAEPoint	
keyAEPointList	
keyAEPointSize	

keyAEPosition	
keyAEPropData	
keyAEProperties	
keyAEProperty	
keyAEPropFlags	
keyAEPropID	
keyAEProtection	
keyAERegionClass	
keyAERenderAs	
keyAERequestedType	
keyAEResult	
keyAEResultInfo	
keyAERotation	
keyAERotPoint	
keyAERowList	
keyAESaveOptions	
keyAEScale	
keyAEScriptTag	
keyAESearchText	Identifies an optional parameter to the open documents Apple event, described in "Event ID Constants." The parameter contains the search text from the Spotlight search that identified the documents to be opened.
keyAEServerInstance	
keyAEShowWhere	
keyAEStartAngle	
keyAEStartPoint	
keyAEStyles	
keyAESuiteID	

keyAEText	
keyAETextColor	
keyAETextFont	
keyAETextLineAscent	
keyAETextLineHeight	
keyAETextPointSize	
keyAETextServiceEncoding	
keyAETextServiceMacEncoding	
keyAETextStyles	
keyAETheData	
keyAETheText	
keyAETransferMode	
keyAETranslation	
keyAETryAsStructGraf	
keyAETSMDocumentRefcon	
keyAETSMEventRecord	
keyAETSMEventRef	
keyAETSMGlyphInfoArray	
keyAETSMScriptTag	
keyAETSMTextFMFont	
keyAETSMTextFont	
keyAETSMTextPointSize	
keyAEUniformStyles	
keyAEUpdateOn	
keyAEUpdateRange	
keyAEUserTerm	
keyAEWindow	
keyAEWritingCode	

keyCloseAllWindows	
keyDriveNumber	
keyErrorCode	
keyHighLevelClass	
keyHighLevelID	
keyKey	
keyKeyboard	
keyKeyCode	
keyLocalWhere	
keyMenuID	
keyMenuItem	
keyMiscellaneous	
keyModifiers	
keyNewBounds	
keyOriginalBounds	
keySelection	
keyWhen	
keyWhere	
keyWindow	
kFAAttachCommand	
kFAEditCommand	
kFAFileParam	
kFAIndexParam	
kFARemoveCommand	
kFAServerApp	
kFASuiteCode	
kFolderActionCode	
kFolderClosedEvent	

kFolderItemsAddedEvent	
kFolderItemsRemovedEvent	
kFolderOpenedEvent	
kFolderWindowMovedEvent	
kGetSelectedText	
kItemList	
kNewSizeParameter	
kNextBody	
kOffset2Pos	
kPos2Offset	
kPreviousBody	
kShowHideInputWindow	
kTextServiceClass	
kTSMHiliteBlockFillText	Specifies block fill highlight style.
kTSMHiliteCaretPosition	Specifies caret position.
kTSMHiliteConvertedText	Specifies range of converted text.
kTSMHiliteNoHilite	Specifies range of non-highlighted text.
kTSMHiliteOutlineText	Specifies outline highlight style.
kTSMHiliteRawText	Specifies range of raw text.
kTSMHiliteSelectedConvertedText	Specifies range of selected converted text.
kTSMHiliteSelectedRawText	Specifies range of selected raw text.
kTSMHiliteSelectedText	Specifies selected highlight style.
kTSMInsideOfActiveInputArea	
kTSMInsideOfBody	
kTSMOutsideOfBody	
kUnicodeNotFromInputMethod	
kUpdateActiveInputArea	

OffsetArray	Specifies offsets of ranges of text. Not typically used by developers.
OffsetArrayHandle	Defines a data type that points to an OffsetArray. Not typically used by developers.
OffsetArrayPtr	
pArcAngle	
pATMachine	
pATType	
pATZone	
pBackgroundColor	
pBackgroundPattern	
pBestType	
pBounds	
pClass	
pClipboard	
pColor	
pColorTable	
pConduit	
pContents	
pCornerCurveHeight	
pCornerCurveWidth	
pDashStyle	
pDefaultType	
pDefinitionRect	
pDeviceAddress	
pDeviceType	
pDNS	
pDNSForm	
pDottedDecimal	

pEnabled	
pEndPoint	
pFillColor	
pFillPattern	
pFont	
pFormula	
pFTPKind	
pGraphicObjects	
pHasCloseBox	
pHasTitleBar	
pHost	
pID	
pIndex	
pInsertionLoc	
pIsFloating	
pIsFrontProcess	
pIsModal	
pIsModified	
pIsResizable	
pIsStationeryPad	
pIsZoomable	
pIsZoomed	
pItemNumber	
pJustification	
pKeyKind	
pKeystrokeKey	
pLineArrow	
pMenuID	

pModifiers	
pName	
pNetwork	
pNewElementLoc	
pNode	
pPath	
pPenColor	
pPenPattern	
pPenWidth	
pPixelDepth	
pPointList	
pPointSize	
pPort	
pProtection	
pProtocol	
pRotation	
pScale	
pScheme	
pScript	
pScriptTag	
pSCSIBus	
pSCSILUN	
pSelected	
pSelection	
pSocket	
pStartAngle	
pStartPoint	
pTextColor	

pTextEncoding	
pTextFont	
pTextItemDelimiters	
pTextPointSize	
pTextStyles	
pTransferMode	
pTranslation	
pUniformStyles	
pUpdateOn	
pURL	
pUserName	
pUserPassword	
pUserSelection	
pVersion	
pVisible	
TextRange	Specifies a range of text. Not typically used by developers.
TextRangeArray	Specifies an array of text ranges. Not typically used by developers.
TextRangeArrayHandle	
TextRangeArrayPtr	
TextRangeHandle	
TextRangePtr	
typeAEText	
typeArc	
typeBest	
typeCell	
typeCentimeters	
typeCFAbsoluteTime	

typeClassInfo	
typeColorTable	
typeColumn	
typeComponentInstance	
typeCubicCentimeter	
typeCubicFeet	
typeCubicInches	
typeCubicMeters	
typeCubicYards	
typeDashStyle	
typeData	
typeDegreesC	
typeDegreesF	
typeDegreesK	
typeDrawingArea	
typeElemInfo	
typeEnumeration	
typeEPS	
typeEventInfo	
typeEventRef	
typeFeet	
typeFinderWindow	
typeFixedPoint	
typeFixedRectangle	
typeGallons	
typeGIF	
typeGlyphInfoArray	
typeGrams	

typeGraphicLine	
typeGraphicText	
typeGroupedGraphic	
typeHIMenu	
typeHIWindow	
typeInches	
typeInsertionLoc	
typeIntlText	For important information, see the Version Notes section of the "typeUnicodeText" enum.
typeIntlWritingCode	
typeISO8601DateTime	
typeJPEG	
typeKilograms	
typeKilometers	
typeLiters	
typeLongDateTime	
typeLongFixed	
typeLongFixedPoint	
typeLongFixedRectangle	
typeLongPoint	
typeLongRectangle	
typeLowLevelEventRecord	
typeMachineLoc	
typeMeters	
typeMiles	
typeOffsetArray	
typeOunces	
typeOval	

typeParamInfo	
typePict	
typePixelMap	
typePixMapMinus	
typePolygon	
typePounds	
typePropInfo	
typePtr	
typeQDPoint	
typeQDRegion	
typeQuarts	
typeRectangle	
typeRGB16	
typeRGB96	
typeRGBColor	
typeRotation	
typeRoundedRectangle	
typeRow	
typeScrapStyles	
typeScript	
typeSquareFeet	
typeSquareKilometers	
typeSquareMeters	
typeSquareMiles	
typeSquareYards	
typeStyledText	Text that includes style information.
typeSuiteInfo	
typeTable	

typeText	
typeTextRange	
typeTextRangeArray	
typeTextStyles	
typeTIFF	
typeVersion	
typeYards	
WritingCode	

AEUserTermTypes.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.

kAEOSAXSizeResource	
kAEScriptingSizeResource	
kAETerminologyExtension	
kAEUserTerminology	
kAEUTApostrophe	
kAEUTChangesState	
kAEUTDirectParamIsReference	
kAEUTEnumerated	
kAEUTEnumListIsExclusive	
kAEUTEnumsAreTypes	
kAEUTFeminine	
kAEUTHasReturningParam	
kAEUTlistOfItems	
kAEUTMasculine	
kAEUTNotDirectParamIsTarget	
kAEUTOptional	

kAEUTParamIsReference	
kAEUTParamIsTarget	
kAEUTPlural	
kAEUTPropertyIsReference	
kAEUTReadWrite	
kAEUTReplyIsReference	
kAEUTTightBindingFunction	
kAlwaysSendSubject	
kDontFindAppBySignature	
kLaunchToGetTerminology	
kOSIZCodeInSharedLibraries	
kOSIZdontAcceptRemoteEvents	
kOSIZDontOpenResourceFile	
kOSIZOpenWithReadPermission	
kReadExtensionTermsMask	
TScriptingSizeResource	Defines a data type to store stack and heap information. Not typically used by developers.

AppleEvents.h

Functions

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

AECreateRemoteProcessResolver	Creates an object for resolving a list of remote processes.
AEDisposeRemoteProcessResolver	Disposes of an AERemoteProcessResolverRef.
AEGetEventHandler	Gets an event handler from an Apple event dispatch table.
AEGetSpecialHandler	Gets a specified handler from a special handler dispatch table.

`AEInstallEventHandler`	Adds an entry for an event handler to an Apple event dispatch table.
`AEInstallSpecialHandler`	Installs a callback function in a special handler dispatch table.
`AEManagerInfo`	Provides information about the version of the Apple Event Manager currently available or the number of processes that are currently recording Apple events.
`AERemoteProcessResolverGetProcesses`	Returns an array of objects containing information about processes running on a remote machine.
`AERemoteProcessResolverScheduleWithRunLoop`	Schedules a resolver for execution on a given run loop in a given mode.
`AERemoveEventHandler`	Removes an event handler entry from an Apple event dispatch table.
`AERemoveSpecialHandler`	Removes a handler from a special handler dispatch table.

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.

`AEEventSource`	A data type for values that specify how an Apple event was delivered.
`AERemoteProcessResolverCallback`	Defines a pointer to a function the Apple Event Manager calls when the asynchronous execution of a remote process resolver completes, either due to success or failure, after a call to the AERemoteProcessResolverScheduleWithRunLoop function. Your callback function can use the reference passed to it to get the remote process information.
`AERemoteProcessResolverContext`	Supplied as a parameter when performing asynchronous resolution of remote processes.
`AERemoteProcessResolverRef`	An opaque reference to an object that encapsulates the mechanism for obtaining a list of processes running on a remote machine.
`kAEAnswer`	Event that is a reply Apple event.
`kAEApplicationDied`	Event sent by the Process Manager to an application that launched another application when the launched application quits or terminates.

kAEDirectCall	The source of the Apple event is a direct call that bypassed the PPC Toolbox.
kAELocalProcess	The source application is another process on the same computer as the target application.
kAENotifyRecording	Wildcard event class and event ID handled by a recording process in order to receive and record copies of recordable events sent to it by the Apple Event Manager.
kAENotifyStartRecording	An event that notifies an application that recording has been turned on.
kAENotifyStopRecording	An event that notifies an application that recording has been turned off.
kAEOpenApplication	Event that launches an application.
kAEOpenContents	Event that provides an application with dragged content, such as text or an image.
kAEOpenDocuments	Event that provides an application with a list of documents to open.
kAEPrintDocuments	Event that provides an application with a list of documents to print.
kAEQuitApplication	Event that causes the application to quit.
kAERemoteProcess	The source application is a process on a remote computer on the network.
kAERemoteProcessNameKey	Use this key to obtain the visible name of the remote process, in the localization supplied by the server, as a CFStringRef.
kAERemoteProcessProcessIDKey	Use this key to obtain the process ID of the remote process, if available; if so, returned as a CFNumberRef.
kAERemoteProcessURLKey	Use this key to obtain the full URL to the remote process, as a CFURLRef.
kAERemoteProcessUserIDKey	Use this key to obtain the user ID of the remote process, if available; if so, returned as a CFNumberRef.
kAESameProcess	The source of the Apple event is the same application that received the event (the target application and the source application are the same).
kAEShowPreferences	Event sent by the Mac OS X to a process when the user chooses the Preferences item for that process.
kAEStartRecording	Event ID for an event by a scripting component to the recording process (or to any running process on the local computer), but handled by the Apple Event Manager.

kAEStopRecording	Event ID for an event sent by a scripting component to the recording process (or to any running process on the local computer), but handled by the Apple Event Manager.
kAEUnknownSource	The source of the Apple event is unknown.
kCoreEventClass	An Apple event sent by the Mac OS; applications that present a graphical interface to the user should be able to any events sent by the Mac OS that apply to the application.
keyAERecorderCount	Used with the keyword parameter of the AEManagerInfo function. If you pass this value, on return, the result parameter supplies the number of processes that are currently recording Apple events.
keyAEVersion	Used with the keyword parameter of the AEManagerInfo function. If you pass this value, on return, the result parameter supplies version information for the Apple Event Manager, in NumVersion format.
keyDirectObject	Direct parameter. Usually specifies the data to be acted upon by the target application.
keyErrorNumber	Error number. Often used to extract error information from a reply Apple event.
keyErrorString	Error string. Often used to extract error information from a reply Apple event to display to the user.
keyPreDispatch	A predispatch handler (an Apple event handler that the Apple Event Manager calls immediately before it dispatches an Apple event). See also “Managing Special Handler Dispatch Tables.”
keyProcessSerialNumber	Process serial number. See also AEManagerInfo.
keySelectProc	You pass this value in the functionClass parameter of the AEManagerInfo function to disable the Object Support Library. Disabling the Object Support Library is not recommended.

CFNetwork

CFHTTPMessage.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.

kCFHTTPAuthenticationSchemeNegotiate	Request an automatically negotiated authentication scheme.

kCFHTTPAuthenticationSchemeNTLM	Request the HTTP NTLM authentication scheme.

CFHTTPStream.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.

kCFStreamPropertyHTTPFinalRequest	HTTP Final Request property. A value of type CFHTTPMessage containing the final message transmitted by the stream after all modifications (including authentication, connection policy, redirects, and so on) have been made. This property cannot be set.

CFNetServices.h

Functions

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

CFNetServiceGetPortNumber	This function gets the port number from a CFNetService.

CFNetworkDefs.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.

APPLE_WIN_API	

CFNetworkErrors.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.

CFNetworkErrors	
kCFDNSServiceFailureKey	
kCFErrorDomainCFNetwork	

kCFErrorDomainWinSock	
kCFErrorHTTPAuthenticationTypeUnsupported	
kCFErrorHTTPBadCredentials	
kCFErrorHTTPBadProxyCredentials	
kCFErrorHTTPBadURL	
kCFErrorHTTPConnectionLost	
kCFErrorHTTPParseFailure	
kCFErrorHTTPProxyConnectionFailure	
kCFErrorHTTPRedirectionLoopDetected	
kCFFTPErrorUnexpectedStatusCode	
kCFFTPStatusCodeKey	
kCFGetAddrInfoFailureKey	
kCFHostErrorHostNotFound	
kCFHostErrorUnknown	
kCFNetServiceErrorBadArgument	
kCFNetServiceErrorCancel	
kCFNetServiceErrorCollision	
kCFNetServiceErrorDNSServiceFailure	
kCFNetServiceErrorInProgress	
kCFNetServiceErrorInvalid	
kCFNetServiceErrorNotFound	
kCFNetServiceErrorTimeout	
kCFNetServiceErrorUnknown	
kCFSOCKS4ErrorIdConflict	
kCFSOCKS4ErrorIdentdFailed	
kCFSOCKS4ErrorRequestFailed	
kCFSOCKS4ErrorUnknownStatusCode	
kCFSOCKS5ErrorBadCredentials	

kCFSOCKS5ErrorBadResponseAddr	
kCFSOCKS5ErrorBadState	
kCFSOCKS5ErrorNoAcceptableMethod	
kCFSOCKS5ErrorUnsupportedNegotiationMethod	
kCFSOCKSErrorUnknownClientVersion	
kCFSOCKSErrorUnsupportedServerVersion	
kCFSOCKSNegotiationMethodKey	
kCFSOCKSStatusCodeKey	
kCFSOCKSVersionKey	

CFProxySupport.h

Functions

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

CFNetworkCopyProxiesForAutoConfigurationScript	
CFNetworkCopyProxiesForURL	
CFNetworkExecuteProxyAutoConfigurationURL	

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.

CFProxyAutoConfigurationResultCallback	
kCFProxyAutoConfigurationURLKey	
kCFProxyHostNameKey	
kCFProxyPasswordKey	
kCFProxyPortNumberKey	
kCFProxyTypeAutoConfigurationURL	
kCFProxyTypeFTP	
kCFProxyTypeHTTP	

kCFProxyTypeHTTPS	
kCFProxyTypeKey	
kCFProxyTypeNone	
kCFProxyTypeSOCKS	
kCFProxyUsernameKey	

CFSocketStream.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.

kCFStreamErrorDomainWinSock	

CarbonCore

Aliases.h

Functions

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

FSMatchAliasBulk	Identifies a list of possible matches for an alias.
FSNewAliasFromPath	Creates a new alias record, given the pathname of the target file or directory.

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.

FSAliasFilterProcPtr	Defines a pointer to an alias filtering callback function that filters out possible targets identified by the FSMatchAliasBulk function.

BackupCore.h

Functions

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

`CSBackupIsItemExcluded`	Returns a Boolean value indicating whether an item is currently excluded from the backup.
`CSBackupSetItemExcluded`	Includes or excludes an item from the backup.

Components.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.

`platformPowerPC64NativeEntryPoint`	
`platformX86_64NativeEntryPoint`	

FSEvents.h

Functions

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

`FSEventsCopyUUIDForDevice`	
`FSEventsGetCurrentEventId`	
`FSEventsGetLastEventIdForDeviceBeforeTime`	
`FSEventsPurgeEventsForDeviceUpToEventId`	
`FSEventStreamCopyDescription`	
`FSEventStreamCopyPathsBeingWatched`	
`FSEventStreamCreate`	
`FSEventStreamCreateRelativeToDevice`	
`FSEventStreamFlushAsync`	
`FSEventStreamFlushSync`	

FSEventStreamGetDeviceBeingWatched	
FSEventStreamGetLatestEventId	
FSEventStreamInvalidate	
FSEventStreamRelease	
FSEventStreamRetain	
FSEventStreamScheduleWithRunLoop	
FSEventStreamShow	
FSEventStreamStart	
FSEventStreamStop	
FSEventStreamUnscheduleFromRunLoop	

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.

ConstFSEventStreamRef	
FSEventStreamCallback	
FSEventStreamContext	
FSEventStreamCreateFlags	
FSEventStreamEventFlags	
FSEventStreamEventId	
FSEventStreamRef	
kFSEventStreamCreateFlagNoDefer	
kFSEventStreamCreateFlagNone	
kFSEventStreamCreateFlagUseCFTypes	
kFSEventStreamCreateFlagWatchRoot	
kFSEventStreamEventFlagEventIdsWrapped	
kFSEventStreamEventFlagHistoryDone	
kFSEventStreamEventFlagKernelDropped	
kFSEventStreamEventFlagMount	

kFSEventStreamEventFlagMustScanSubDirs	
kFSEventStreamEventFlagNone	
kFSEventStreamEventFlagRootChanged	
kFSEventStreamEventFlagUnmount	
kFSEventStreamEventFlagUserDropped	
kFSEventStreamEventIdSinceNow	

Files.h

Functions

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

FSFlushVolume	For the specified volume, writes all open and modified files in the current process to permanent storage.
FSGetTemporaryDirectoryForReplaceObject	
FSGetVolumeMountInfo	Retrieves the mounting information associated with the specified volume.
FSGetVolumeMountInfoSize	Determines the size of the mounting information associated with the specified volume.
FSGetVolumeParms	Retrieves information about the characteristics of a volume.
FSMoveObjectToTrashAsync	Starts an asynchronous file operation to move a source object to the Trash.
FSMoveObjectToTrashSync	Moves a source object to the Trash.
FSPathGetTemporaryDirectoryForReplaceObject	
FSPathMoveObjectToTrashAsync	Starts an asynchronous file operation to move a source object, specified using a pathname, to the Trash.
FSPathMoveObjectToTrashSync	Moves a source object, specified using a pathname, to the Trash.
FSPathReplaceObject	
FSReplaceObject	

FSResolveNodeID	
FSUnlinkObject	
FSVolumeMount	Mounts a volume using the specified mounting information.
PBFlushVolumeAsync	For the specified volume, writes all open and modified files in the current process to permanent storage.
PBFlushVolumeSync	For the specified volume, writes all open and modified files in the current process to permanent storage.
PBFSCopyFileAsync	Duplicates a file and optionally renames it.
PBFSCopyFileSync	Duplicates a file and optionally renames it.
PBFSResolveNodeIDAsync	
PBFSResolveNodeIDSync	
PBUnlinkObjectAsync	
PBUnlinkObjectSync	

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.

FSIORefNum	
fsRdAccessPerm	
fsWrAccessPerm	
kFSReplaceObjectDefaultOptions	
kFSReplaceObjectDoNotCheckObjectWriteAccess	
kFSReplaceObjectPreservePermissionInfo	
kFSReplaceObjectReplaceMetadata	
kFSReplaceObjectReplacePermissionInfo	
kFSReplaceObjectSaveOriginalAsABackup	

Folders.h

Functions

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

GetFolderNameUnicode	

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.

kAudioDigidesignFolderType	
kAudioVSTFolderType	
kColorPickersFolderType	
kCompositionsFolderType	
kDownloadsFolderType	
kFolderManagerLastDomain	
kFontCollectionsFolderType	
kiMovieFolderType	
kiMoviePlugInsFolderType	
kiMovieSoundEffectsFolderType	
kInputManagersFolderType	
kInputMethodsFolderType	
kLibraryAssistantsFolderType	
kScreenSaversFolderType	
kSpotlightImportersFolderType	
kSpotlightMetadataCacheFolderType	
kWidgetsFolderType	

Gestalt.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.

gestaltCPU970MP	

MacErrors.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.

kPOSIXErrorBase	
kPOSIXErrorE2BIG	
kPOSIXErrorEACCES	
kPOSIXErrorEADDRINUSE	
kPOSIXErrorEADDRNOTAVAIL	
kPOSIXErrorEAFNOSUPPORT	
kPOSIXErrorEAGAIN	
kPOSIXErrorEALREADY	
kPOSIXErrorEAUTH	
kPOSIXErrorEBADARCH	
kPOSIXErrorEBADEXEC	
kPOSIXErrorEBADF	
kPOSIXErrorEBADMACHO	
kPOSIXErrorEBADMSG	
kPOSIXErrorEBADRPC	
kPOSIXErrorEBUSY	
kPOSIXErrorECANCELED	
kPOSIXErrorECHILD	
kPOSIXErrorECONNABORTED	

kPOSIXErrorECONNREFUSED	
kPOSIXErrorECONNRESET	
kPOSIXErrorEDEADLK	
kPOSIXErrorEDESTADDRREQ	
kPOSIXErrorEDEVERR	
kPOSIXErrorEDOM	
kPOSIXErrorEDQUOT	
kPOSIXErrorEEXIST	
kPOSIXErrorEFAULT	
kPOSIXErrorEFBIG	
kPOSIXErrorEFTYPE	
kPOSIXErrorEHOSTDOWN	
kPOSIXErrorEHOSTUNREACH	
kPOSIXErrorEIDRM	
kPOSIXErrorEILSEQ	
kPOSIXErrorEINPROGRESS	
kPOSIXErrorEINTR	
kPOSIXErrorEINVAL	
kPOSIXErrorEIO	
kPOSIXErrorEISCONN	
kPOSIXErrorEISDIR	
kPOSIXErrorELOOP	
kPOSIXErrorEMFILE	
kPOSIXErrorEMLINK	
kPOSIXErrorEMSGSIZE	
kPOSIXErrorEMULTIHOP	
kPOSIXErrorENAMETOOLONG	
kPOSIXErrorENEEDAUTH	

kPOSIXErrorENETDOWN	
kPOSIXErrorENETRESET	
kPOSIXErrorENETUNREACH	
kPOSIXErrorENFILE	
kPOSIXErrorENOATTR	
kPOSIXErrorENOBUFS	
kPOSIXErrorENODATA	
kPOSIXErrorENODEV	
kPOSIXErrorENOENT	
kPOSIXErrorENOEXEC	
kPOSIXErrorENOLCK	
kPOSIXErrorENOLINK	
kPOSIXErrorENOMEM	
kPOSIXErrorENOMSG	
kPOSIXErrorENOPROTOOPT	
kPOSIXErrorENOSPC	
kPOSIXErrorENOSR	
kPOSIXErrorENOSTR	
kPOSIXErrorENOSYS	
kPOSIXErrorENOTBLK	
kPOSIXErrorENOTCONN	
kPOSIXErrorENOTDIR	
kPOSIXErrorENOTEMPTY	
kPOSIXErrorENOTSOCK	
kPOSIXErrorENOTSUP	
kPOSIXErrorENOTTY	
kPOSIXErrorENXIO	
kPOSIXErrorEOPNOTSUPP	

kPOSIXErrorEOVERFLOW	
kPOSIXErrorEPERM	
kPOSIXErrorEPFNOSUPPORT	
kPOSIXErrorEPIPE	
kPOSIXErrorEPROCLIM	
kPOSIXErrorEPROCUNAVAIL	
kPOSIXErrorEPROGMISMATCH	
kPOSIXErrorEPROGUNAVAIL	
kPOSIXErrorEPROTO	
kPOSIXErrorEPROTONOSUPPORT	
kPOSIXErrorEPROTOTYPE	
kPOSIXErrorEPWROFF	
kPOSIXErrorERANGE	
kPOSIXErrorEREMOTE	
kPOSIXErrorEROFS	
kPOSIXErrorERPCMISMATCH	
kPOSIXErrorESHLIBVERS	
kPOSIXErrorESHUTDOWN	
kPOSIXErrorESOCKTNOSUPPORT	
kPOSIXErrorESPIPE	
kPOSIXErrorESRCH	
kPOSIXErrorESTALE	
kPOSIXErrorETIME	
kPOSIXErrorETIMEDOUT	
kPOSIXErrorETOOMANYREFS	
kPOSIXErrorETXTBSY	
kPOSIXErrorEUSERS	
kPOSIXErrorEXDEV	

MacTypes.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.

`PRefCon`	
`SRefCon`	
`URefCon`	

MachineExceptions.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.

`FPUInformationIntel64`	
`MachineInformationIntel64`	
`RegisterInformationIntel64`	
`VectorInformationIntel64`	

Resources.h

Functions

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

`FSOpenOrphanResFile`	Opens a resource file that is persistent across all contexts.

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.

`ResourceCount`	
`ResourceIndex`	

TextCommon.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.

`kTextCenter`	
`kTextEncodingShiftJIS_X0213`	
`kTextEncodingUnicodeV5_0`	
`kTextFlushDefault`	
`kTextFlushLeft`	
`kTextFlushRight`	

DictionaryServices

DictionaryServices.h

Functions

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

`DCSCopyTextDefinition`	Returns the definition associated with the provided text range.
`DCSGetTermRangeInString`	Determines the range of the longest word or phrase with respect to an offset.

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.

`DCSDictionaryRef`	An opaque object that represents a dictionary file.

LaunchServices

IconsCore.h

Functions

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

`AcquireIconRef`	Increments the reference count for an IconRef.
`CompositeIconRef`	Superimposes one IconRef onto another.
`FlushIconRefs`	Reclaims memory used by the specified icon if the memory is purgeable.
`FlushIconRefsByVolume`	On a given volume, reclaims memory used by purgeable icons.
`GetCustomIconsEnabled`	Determines whether custom icons are enabled or disabled on a specified volume.
`GetIconRef`	Provides an IconRef object for an icon in the desktop database or for a registered icon.
`GetIconRefFromComponent`	Provides an IconRef object based on a specified component.
`GetIconRefFromFile`	Provides an IconRef object for a file, folder or volume.
`GetIconRefFromFileInfo`	Provides an IconRef object for a file with minimal file I/O.
`GetIconRefFromFolder`	Provides an IconRef object for a folder with no custom icon.
`GetIconRefFromIconFamilyPtr`	Provides an IconRef object from a specified icon family.
`GetIconRefFromTypeInfo`	Provides an IconRef object with the specified type information.
`GetIconRefOwners`	Provides the current reference count for an IconRef.
`IsDataAvailableInIconRef`	Indicates whether an IconRef has the specified data.
`IsIconRefComposite`	Reports whether a specified IconRef has been composited.
`IsValidIconRef`	Reports whether a specified IconRef is valid.
`OverrideIconRef`	Replaces the bitmaps of one IconRef with those of another IconRef.
`OverrideIconRefFromResource`	Replaces the bitmaps in an IconRef with bitmaps from a specified resource file.
`ReadIconFile`	Copies data from a given file into an icon family.
`ReadIconFromFSRef`	Reads an icon ('icns') file into memory.

`RegisterIconRefFromFSRef`	Registers an IconRef from a .icns file and associates it with a creator and type pair.
`RegisterIconRefFromIconFamily`	Adds an iconFamily-derived IconRef to the Icon Services registry.
`RegisterIconRefFromIconFile`	Adds a file-derived IconRef to the Icon Services registry.
`RegisterIconRefFromResource`	Adds a resource-derived IconRef to the Icon Services registry.
`ReleaseIconRef`	Decrements the reference count for an IconRef.
`RemoveIconRefOverride`	Restores the original bitmaps of an overridden IconRef.
`SetCustomIconsEnabled`	Enables or disables custom icons on a specified volume.
`UnregisterIconRef`	Removes the specified icon data from the icon registry.
`UpdateIconRef`	Forces an update of IconRef data.
`WriteIconFile`	Copies data from a given icon family into a file.

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.

`appleMenuFolderIconResource`	
`controlPanelFolderIconResource`	
`desktopIconResource`	
`dropFolderIconResource`	
`extensionsFolderIconResource`	
`floppyIconResource`	
`fontsFolderIconResource`	
`fullTrashIconResource`	
`genericApplicationIconResource`	
`genericCDROMIconResource`	
`genericDeskAccessoryIconResource`	
`genericDocumentIconResource`	
`genericEditionFileIconResource`	
`genericExtensionIconResource`	
`genericFileServerIconResource`	

genericFolderIconResource	
genericHardDiskIconResource	
genericMoverObjectIconResource	
genericPreferencesIconResource	
genericQueryDocumentIconResource	
genericRAMDiskIconResource	
genericStationeryIconResource	
genericSuitcaseIconResource	
IconRef	Defines an icon reference.
IconServicesUsageFlags	
kAFPServerIcon	
kAlertCautionBadgeIcon	
kAlertCautionIcon	
kAlertNoteIcon	
kAlertStopIcon	
kAliasBadgeIcon	
kAppearanceFolderIcon	
kAppleExtrasFolderIcon	
kAppleLogoIcon	
kAppleMenuFolderIcon	
kAppleMenuFolderIconResource	
kAppleMenuIcon	
kAppleScriptBadgeIcon	
kAppleTalkIcon	
kAppleTalkZoneIcon	
kApplicationsFolderIcon	
kApplicationSupportFolderIcon	
kAssistantsFolderIcon	

kBackwardArrowIcon	
kBurningIcon	
kClipboardIcon	
kClippingPictureTypeIcon	
kClippingSoundTypeIcon	
kClippingTextTypeIcon	
kClippingUnknownTypeIcon	
kColorSyncFolderIcon	
kComputerIcon	
kConnectToIcon	
kContextualMenuItemsFolderIcon	
kControlPanelDisabledFolderIcon	
kControlPanelFolderIcon	
kControlPanelFolderIconResource	
kControlStripModulesFolderIcon	
kDeleteAliasIcon	
kDesktopIcon	
kDesktopIconResource	
kDocumentsFolderIcon	
kDropFolderIcon	
kDropFolderIconResource	
kEjectMediaIcon	
kExtensionsDisabledFolderIcon	
kExtensionsFolderIcon	
kExtensionsFolderIconResource	
kFavoriteItemsIcon	
kFavoritesFolderIcon	
kFinderIcon	

kFloppyIconResource	
kFontsFolderIcon	
kFontsFolderIconResource	
kFontSuitcaseIcon	
kForwardArrowIcon	
kFTPServerIcon	
kFullTrashIcon	
kFullTrashIconResource	
kGenericApplicationIcon	
kGenericApplicationIconResource	
kGenericCDROMIcon	
kGenericCDROMIconResource	
kGenericComponentIcon	
kGenericControlPanelIcon	
kGenericControlStripModuleIcon	
kGenericDeskAccessoryIcon	
kGenericDeskAccessoryIconResource	
kGenericDocumentIcon	
kGenericDocumentIconResource	
kGenericEditionFileIcon	
kGenericEditionFileIconResource	
kGenericExtensionIcon	
kGenericExtensionIconResource	
kGenericFileServerIcon	
kGenericFileServerIconResource	
kGenericFloppyIcon	
kGenericFolderIcon	
kGenericFolderIconResource	

kGenericFontIcon	
kGenericFontScalerIcon	
kGenericHardDiskIcon	
kGenericHardDiskIconResource	
kGenericIDiskIcon	
kGenericMoverObjectIcon	
kGenericMoverObjectIconResource	
kGenericNetworkIcon	
kGenericPCCardIcon	
kGenericPreferencesIcon	
kGenericPreferencesIconResource	
kGenericQueryDocumentIcon	
kGenericQueryDocumentIconResource	
kGenericRAMDiskIcon	
kGenericRAMDiskIconResource	
kGenericRemovableMediaIcon	
kGenericSharedLibaryIcon	
kGenericStationeryIcon	
kGenericStationeryIconResource	
kGenericSuitcaseIcon	
kGenericSuitcaseIconResource	
kGenericURLIcon	
kGenericWindowIcon	
kGenericWORMIcon	
kGridIcon	
kGroupIcon	
kGuestUserIcon	
kHelpFolderIcon	

kHelpIcon	
kHelpIconResource	
kHTTPServerIcon	
kIconServicesCatalogInfoMask	Use this mask with the File Manager function FSGetCatalogInfo before calling GetIconRefFromFileInfo.
kIconServicesNoBadgeFlag	
kIconServicesNormalUsageFlag	
kIconServicesUpdateIfNeededFlag	
kInternationalResourcesIcon	
kInternationResourcesIcon	
kInternetFolderIcon	
kInternetLocationAppleShareIcon	
kInternetLocationAppleTalkZoneIcon	
kInternetLocationFileIcon	
kInternetLocationFTPIcon	
kInternetLocationGenericIcon	
kInternetLocationHTTPIcon	
kInternetLocationMailIcon	
kInternetLocationNewsIcon	
kInternetLocationNSLNeighborhoodIcon	
kInternetPlugInFolderIcon	
kInternetSearchSitesFolderIcon	
kIPFileServerIcon	
kKeepArrangedIcon	
kKeyboardLayoutIcon	
kLocalesFolderIcon	
kLockedBadgeIcon	

kLockedIcon	
kMacOSReadMeFolderIcon	
kMountedBadgeIcon	
kMountedFolderIcon	
kMountedFolderIconResource	
kNoFilesIcon	
kNoFolderIcon	
kNoWriteIcon	
kOpenFolderIcon	
kOpenFolderIconResource	
kOwnedFolderIcon	
kOwnedFolderIconResource	
kOwnerIcon	
kPreferencesFolderIcon	
kPreferencesFolderIconResource	
kPrinterDescriptionFolderIcon	
kPrinterDriverFolderIcon	
kPrintMonitorFolderIcon	
kPrintMonitorFolderIconResource	
kPrivateFolderIcon	
kPrivateFolderIconResource	
kProtectedApplicationFolderIcon	
kProtectedSystemFolderIcon	
kPublicFolderIcon	
kQuestionMarkIcon	
kRecentApplicationsFolderIcon	
kRecentDocumentsFolderIcon	
kRecentItemsIcon	

kRecentServersFolderIcon	
kRightContainerArrowIcon	
kScriptingAdditionsFolderIcon	
kScriptsFolderIcon	
kSharedBadgeIcon	
kSharedFolderIcon	
kSharedFolderIconResource	
kSharedLibrariesFolderIcon	
kSharingPrivsNotApplicableIcon	
kSharingPrivsReadOnlyIcon	
kSharingPrivsReadWriteIcon	
kSharingPrivsUnknownIcon	
kSharingPrivsWritableIcon	
kShortcutIcon	
kShutdownItemsDisabledFolderIcon	
kShutdownItemsFolderIcon	
kSortAscendingIcon	
kSortDescendingIcon	
kSoundFileIcon	
kSpeakableItemsFolder	
kStartupFolderIconResource	
kStartupItemsDisabledFolderIcon	
kStartupItemsFolderIcon	
kSystemExtensionDisabledFolderIcon	
kSystemFolderIcon	
kSystemFolderIconResource	
kSystemIconsCreator	
kSystemSuitcaseIcon	

kTextEncodingsFolderIcon	
kToolbarCustomizeIcon	
kToolbarDeleteIcon	
kToolbarFavoritesIcon	
kToolbarHomeIcon	
kTrashIcon	
kTrashIconResource	
kTrueTypeFlatFontIcon	
kTrueTypeFontIcon	
kTrueTypeMultiFlatFontIcon	
kUnknownFSObjectIcon	
kUnlockedIcon	
kUserFolderIcon	
kUserIcon	
kUserIDiskIcon	
kUsersFolderIcon	
kUtilitiesFolderIcon	
kVoicesFolderIcon	
kWorkgroupFolderIcon	
mountedFolderIconResource	
openFolderIconResource	
ownedFolderIconResource	
preferencesFolderIconResource	
printMonitorFolderIconResource	
privateFolderIconResource	
sharedFolderIconResource	
startupFolderIconResource	
systemFolderIconResource	

`trashIconResource`	

LSInfo.h

Functions

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

`LSCanRefAcceptItem`	Tests whether an application can accept (open) an item designated by file-system reference.
`LSCanURLAcceptURL`	Tests whether an application can accept (open) an item designated by URL.
`LSCopyAllHandlersForURLScheme`	Returns an array of application bundle identifiers for applications capable of handling the specified URL scheme.
`LSCopyAllRoleHandlersForContentType`	Returns an array of application bundle identifiers for applications capable of handling a specified content type with the specified roles.
`LSCopyApplicationForMIMEType`	Locates the preferred application for opening items with a specified MIME type.
`LSCopyApplicationURLsForURL`	Locates all known applications suitable for opening an item designated by URL.
`LSCopyDefaultHandlerForURLScheme`	Returns the application bundle identifier of the user's preferred default handler for the specified URL scheme.
`LSCopyDefaultRoleHandlerForContentType`	Returns the application bundle identifier of the user's preferred default handler for the specified content type with the specified role.
`LSCopyDisplayNameForRef`	Obtains the display name for an item designated by file-system reference.
`LSCopyDisplayNameForURL`	Obtains the display name for an item designated by URL.
`LSCopyItemAttribute`	Obtains the value of an item's attribute.
`LSCopyItemAttributes`	Obtains multiple item attribute values as a dictionary.
`LSCopyItemInfoForRef`	Obtains requested information about an item designated by file-system reference.
`LSCopyItemInfoForURL`	Obtains requested information about an item designated by URL.
`LSCopyKindStringForMIMEType`	Obtains the kind string for a specified MIME type.

LSCopyKindStringForRef	Obtains the kind string for an item designated by file-system reference.
LSCopyKindStringForTypeInfo	Obtains a kind string for items with a specified file type, creator signature, filename extension, or any combination of these characteristics.
LSCopyKindStringForURL	Obtains the kind string for an item designated by URL.
LSFindApplicationForInfo	Locates an application with a specified creator signature, bundle ID, filename, or any combination of these characteristics.
LSGetApplicationForInfo	Locates the preferred application for opening items with a specified file type, creator signature, filename extension, or any combination of these characteristics.
LSGetApplicationForItem	Locates the preferred application for opening an item designated by file-system reference.
LSGetApplicationForURL	Locates the preferred application for opening an item designated by URL.
LSGetExtensionInfo	Obtains the starting index of the extension within a filename.
LSGetHandlerOptionsForContentType	Gets the handler options for the specified content type.
LSInit	Calling this function was formerly required in order to initialize Launch Services; it is no longer needed, however, because Launch Services is now initialized automatically the first time one of its functions is called.
LSRegisterFSRef	Registers an application, designated by file-system reference, in the Launch Services database.
LSRegisterURL	Registers an application, designated by URL, in the Launch Services database.
LSSetDefaultHandlerForURLScheme	Sets the user's preferred default handler for the specified URL scheme.
LSSetDefaultRoleHandlerForContentType	Sets the user's preferred default handler for the specified content type in the specified roles.
LSSetExtensionHiddenForRef	Specifies whether the filename extension for an item designated by file-system reference should be hidden or shown.
LSSetExtensionHiddenForURL	Specifies whether the filename extension for an item designated by URL should be hidden or shown.
LSSetHandlerOptionsForContentType	Sets the handler option for the specified content type.

LSSetItemAttribute	
LSTerm	Calling this function was formerly required in order to terminate Launch Services; however, it is no longer needed and so should not be called.

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.

kLSAcceptAllowLoginUI	Requests that the user interface to log in be presented if necessary.
kLSAcceptDefault	Requests the default behavior (as opposed to the behavior specified by the kLSAcceptAllowLoginUI flag).
kLSAppDoesNotClaimTypeErr	Not currently used.
kLSAppDoesNotSupportSchemeWarning	Not currently used.
kLSAppInTrashErr	The application cannot be run because it is inside a Trash folder.
kLSApplicationNotFoundErr	No application in the Launch Services database matches the input criteria.
kLSAttributeNotFoundErr	
kLSAttributeNotSettableErr	
kLSCannotSetInfoErr	The filename extension to be hidden cannot be hidden.
kLSDataErr	Data is structured improperly (for example, an item's information property list is malformed). Not used in Mac OS X v10.4.
kLSDataTooOldErr	Not currently used.
kLSDataUnavailableErr	Data of the desired type is not available (for example, there is no kind string).
kLSExecutableIncorrectFormat	
kLSHandlerOptionsDefault	When set, causes Launch Services to use a content item's creator (when available) to select a handler. This is the default setting.
kLSHandlerOptionsIgnoreCreator	When set, causes Launch Services to ignore the content item's creator when selecting a role handler for the specified content type.
kLSIncompatibleSystemVersionErr	The application to be launched cannot run on the current Mac OS version.

kLSInitializeDefaults	Formerly passed to the LSInit function, which is no longer used.
kLSInvalidExtensionIndex	The value obtained by the LSGetExtensionInfo function if the filename does not contain a valid extension.
kLSItemContentType	The item's content type identifier, which is a uniform type identifier string. The value type of this attribute is CFStringRef.
kLSItemDisplayKind	The localized kind string describing the item's type. The value type of this attribute is CFStringRef.
kLSItemDisplayName	The item's name as displayed to the user. The display name reflects localization and extension hiding that may be in effect. The value type of this attribute is CFStringRef.
kLSItemExtension	The item's filename extension. The value type of this attribute is CFStringRef.
kLSItemExtensionIsHidden	A value of kCFBooleanTrue if the item's extension is set to be hidden; otherwise, kCFBooleanFalse. The value type of this attribute is CFBooleanRef.
kLSItemFileCreator	The item's file creator (OSType). The value type of this attribute is CFStringRef.
kLSItemFileType	The item's file type (OSType). The value type of this attribute is CFStringRef.
kLSItemInfoAppIsScriptable	Item is an application that can be scripted.
kLSItemInfoAppPrefersClassic	Item is an application that can run either natively or in the Classic emulation environment, but prefers to be launched in the Classic environment. This flag is valid only when kLSItemInfoIsNativeApp is set.
kLSItemInfoAppPrefersNative	Item is an application that can run either natively or in the Classic emulation environment, but prefers to be launched natively. This flag is valid only when kLSItemInfoIsNativeApp is set.
kLSItemInfoExtensionIsHidden	Item has a hidden filename extension.
kLSItemInfoIsAliasFile	Item is an alias file (includes symbolic links).
kLSItemInfoIsApplication	Item is a single-file or packaged application.
kLSItemInfoIsClassicApp	Item is an application that cannot run natively and must be launched in the Classic emulation environment.
kLSItemInfoIsContainer	Item is a directory (includes packages) or volume.
kLSItemInfoIsInvisible	Item is invisible, because either its name begins with a period or its isInvisible Finder flag is set.

kLSItemInfoIsNativeApp	Item is an application that can run natively in Mac OS X.
kLSItemInfoIsPackage	Item is a packaged directory.
kLSItemInfoIsPlainFile	Item is a data file (and not, for example, a directory, volume, or UNIX symbolic link).
kLSItemInfoIsSymlink	Item is a UNIX symbolic link.
kLSItemInfoIsVolume	Item is the root directory of a volume or mount point.
kLSItemIsInvisible	A value of kCFBooleanTrue if the item is normally hidden from users; otherwise, kCFBooleanFalse. The value type of this attribute is CFBooleanRef.
kLSItemQuarantineProperties	
kLSItemRoleHandlerDisplayName	The display name of the application that is set to handle this item, subject to the role mask. The value type of this attribute is CFStringRef.
kLSLaunchInProgressErr	A launch of the application is already in progress.
kLSMinCatInfoBitmap	A minimal catalog information bitmap; no longer used.
kLSMultipleSessionsNotSupportedErr	The application to be launched cannot run simultaneously in two different user sessions.
kLSNoClassicEnvironmentErr	The Classic emulation environment was required but is not available.
kLSNoExecutableErr	The executable file is missing or has an unusable format.
kLSNoLaunchPermissionErr	The user does not have permission to launch the application (on a managed network).
kLSNoRegistrationInfoErr	Not currently used.
kLSNotAnApplicationErr	The item to be registered is not an application.
kLSNotInitializedErr	Formerly returned by LSInit on initialization failure; no longer used.
kLSNotRegisteredErr	Not currently used.
kLSRequestAllFlags	Requests all item-information flags.
kLSRequestAllInfo	Requests all available item information.
kLSRequestAppTypeFlags	Requests all application-specific item-information flags: that is, kLSItemInfoIsNativeApp through kLSItemInfoAppIsScriptable.

kLSRequestBasicFlagsOnly	Requests all item-information flags that are not application-specific: that is, all except kLSItemInfoIsNativeApp through kLSItemInfoAppIsScriptable.
kLSRequestExtension	Requests the item's filename extension.
kLSRequestExtensionFlagsOnly	Requests only the kLSItemInfoExtensionIsHidden item-information flag.
kLSRequestIconAndKind	Not used.
kLSRequestTypeCreator	Requests the item's file type and creator signature.
kLSRolesAll	Accepts any role with respect to the item.
kLSRolesEditor	Requests the role Editor (the application can read, present, manipulate, and save the item).
kLSRolesNone	Requests the role None (the application cannot open the item, but provides an icon and a kind string for it).
kLSRolesShell	Requests the role Shell (the application can execute the item).
kLSRolesViewer	Requests the role Viewer (the application can read and present the item, but cannot manipulate or save it).
kLSServerCommunicationErr	There is a problem communicating with the server process that maintains the Launch Services database.
kLSUnknownCreator	The value to supply as the creator signature if no file creator information is available.
kLSUnknownErr	An unknown error has occurred.
kLSUnknownKindID	A possible value of the kindID field of an item-information record, which is no longer used.
kLSUnknownType	The value to supply as the file type (for example, to the LSGetApplicationForInfo function) if no file type information is available.
kLSUnknownTypeErr	Not currently used.
LSAcceptanceFlags	Specify behavior to observe when testing whether an application can accept (open) an item.
LSHandlerOptions	Specify the options for controlling how content handlers are selected.
LSInitializeFlags	The following constants are no longer used.
LSItemInfoFlags	Provide information about an item.

`LSItemInfoRecord`	Contains requested information about an item.
`LSKindID`	Data type of the kindID field of an item-information record (LSItemInfoRecord); no longer used.
`LSRequestedInfo`	Specify what information to obtain about an item.
`LSRolesMask`	Specify the desired role or roles for an application to claim with respect to an item or family of items.

LSOpen.h

Functions

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

`LSOpenApplication`	Launches the specified application.
`LSOpenCFURLRef`	Opens an item designated by URL, in the default manner in its preferred application.
`LSOpenFromRefSpec`	Opens one or more items designated by file-system reference, in either their preferred applications or a designated application.
`LSOpenFromURLSpec`	Opens one or more items designated by URL, in either their preferred applications or a designated application.
`LSOpenFSRef`	Opens an item designated by file-system reference, in the default manner in its preferred application.
`LSOpenItemsWithRole`	Opens items specified as an array of values of type FSRef with a specified role.
`LSOpenURLsWithRole`	Opens one or more URLs with the specified roles.

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.

`kLSLaunchAndDisplayErrors`	Requests that launch and open failures be displayed in the UI.
`kLSLaunchAndHide`	Requests that the application be hidden as soon as it completes its launch sequence.
`kLSLaunchAndHideOthers`	Requests that other applications be hidden as soon as the opened application completes its launch sequence.
`kLSLaunchAndPrint`	Requests that documents opened in the application be printed.

kLSLaunchAsync	Requests that the application be launched asynchronously: that is, the Launch Services function launching it return control immediately, without waiting for it to complete its launch sequence (indicated visually to the user when the application's icon stops "bouncing" in the Dock).
kLSLaunchDefaults	Requests launching in the default manner (as if the only flags set were kLSLaunchNoParams, kLSLaunchAsync, and kLSLaunchStartClassic).
kLSLaunchDontAddToRecents	Requests that the application or documents not be added to the Finder's Recent Items menu.
kLSLaunchDontSwitch	Requests that the application be launched without being brought to the foreground.
kLSLaunchHasUntrustedContents	Requests that the items to be launched should be marked as untrusted.
kLSLaunchInClassic	Requests that the application be forced to launch in the Classic emulation environment.
kLSLaunchInhibitBGOnly	Requests that the launch be made to fail if the application is background-only.
kLSLaunchNewInstance	Requests that a new instance of the application be started, even if one is already running.
kLSLaunchNoParams	Requests that the application's information property list be used to determine the launch parameters.
kLSLaunchReserved2	Reserved for future use.
kLSLaunchReserved3	Reserved for future use.
kLSLaunchReserved4	Reserved for future use.
kLSLaunchReserved5	Reserved for future use.
kLSLaunchStartClassic	Requests that the Classic emulation environment be started up if the application requires it. If this flag is not set and the application requires the Classic environment, the launch will fail.
LSApplicationParameters	Specifies the application, launch flags, and additional parameters that control how an application is launched.
LSLaunchFlags	Specify how to launch an application.
LSLaunchFSRefSpec	Specifies, by file-system reference, an application to launch, items to open, or both, along with related information.
LSLaunchURLSpec	Specifies, by URL, an application to launch, items to open, or both, along with related information.

LSQuarantine.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.

kLSQuarantineAgentBundleIdentifierKey	
kLSQuarantineAgentNameKey	
kLSQuarantineDataURLKey	
kLSQuarantineOriginURLKey	
kLSQuarantineTimeStampKey	
kLSQuarantineTypeCalendarEventAttachment	
kLSQuarantineTypeEmailAttachment	
kLSQuarantineTypeInstantMessageAttachment	
kLSQuarantineTypeKey	
kLSQuarantineTypeOtherAttachment	
kLSQuarantineTypeOtherDownload	
kLSQuarantineTypeWebDownload	

LSSharedFileList.h

Functions

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

LSSharedFileListAddObserver	
LSSharedFileListCopyProperty	
LSSharedFileListCopySnapshot	
LSSharedFileListCreate	
LSSharedFileListGetSeedValue	
LSSharedFileListGetTypeID	
LSSharedFileListInsertItemFSRef	
LSSharedFileListInsertItemURL	

LSSharedFileListItemCopyDisplayName	
LSSharedFileListItemCopyIconRef	
LSSharedFileListItemCopyProperty	
LSSharedFileListItemGetID	
LSSharedFileListItemGetTypeID	
LSSharedFileListItemMove	
LSSharedFileListItemRemove	
LSSharedFileListItemResolve	
LSSharedFileListItemSetProperty	
LSSharedFileListRemoveAllItems	
LSSharedFileListRemoveObserver	
LSSharedFileListSetAuthorization	
LSSharedFileListSetProperty	

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.

kLSSharedFileListDoNotMountVolumes	
kLSSharedFileListFavoriteItems	
kLSSharedFileListFavoriteVolumes	
kLSSharedFileListGlobalLoginItems	
kLSSharedFileListItemBeforeFirst	
kLSSharedFileListItemHidden	
kLSSharedFileListItemLast	
kLSSharedFileListNoUserInteraction	
kLSSharedFileListRecentApplicationItems	
kLSSharedFileListRecentDocumentItems	
kLSSharedFileListRecentItemsMaxAmount	
kLSSharedFileListRecentServerItems	

kLSSharedFileListSessionLoginItems	
kLSSharedFileListVolumesComputerVisible	
kLSSharedFileListVolumesIDiskVisible	
kLSSharedFileListVolumesNetworkVisible	
LSSharedFileListChangedProcPtr	
LSSharedFileListItemRef	
LSSharedFileListRef	

UTCoreTypes.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.

kUTTypeAliasFile	
kUTTypeAliasRecord	
kUTTypeAppleICNS	
kUTTypeAppleProtectedMPEG4Audio	
kUTTypeApplication	
kUTTypeApplicationBundle	
kUTTypeApplicationFile	
kUTTypeArchive	
kUTTypeAudio	
kUTTypeAudiovisualContent	
kUTTypeBMP	
kUTTypeBundle	
kUTTypeCHeader	
kUTTypeCompositeContent	
kUTTypeContact	
kUTTypeContent	
kUTTypeCPlusPlusHeader	

kUTTypeCPlusPlusSource	
kUTTypeCSource	
kUTTypeData	
kUTTypeDirectory	
kUTTypeDiskImage	
kUTTypeFileURL	
kUTTypeFlatRTFD	
kUTTypeFolder	
kUTTypeFramework	
kUTTypeGIF	
kUTTypeHTML	
kUTTypeICO	
kUTTypeImage	
kUTTypeInkText	
kUTTypeItem	
kUTTypeJavaSource	
kUTTypeJPEG	
kUTTypeJPEG2000	
kUTTypeMessage	
kUTTypeMountPoint	
kUTTypeMovie	
kUTTypeMP3	
kUTTypeMPEG	
kUTTypeMPEG4	
kUTTypeMPEG4Audio	
kUTTypeObjectiveCPlusPlusSource	
kUTTypeObjectiveCSource	
kUTTypePackage	

kUTTypePDF	
kUTTypePICT	
kUTTypePlainText	
kUTTypePNG	
kUTTypeQuickTimeImage	
kUTTypeQuickTimeMovie	
kUTTypeResolvable	
kUTTypeRTF	
kUTTypeRTFD	
kUTTypeSourceCode	
kUTTypeSymLink	
kUTTypeText	
kUTTypeTIFF	
kUTTypeTXNTextAndMultimediaData	
kUTTypeURL	
kUTTypeUTF16ExternalPlainText	
kUTTypeUTF16PlainText	
kUTTypeUTF8PlainText	
kUTTypeVCard	
kUTTypeVideo	
kUTTypeVolume	
kUTTypeWebArchive	
kUTTypeXML	

UTType.h

Functions

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

UTCreateStringForOSType	

UTGetOSTypeFromString	
UTTypeConformsTo	
UTTypeCopyDeclaration	
UTTypeCopyDeclaringBundleURL	
UTTypeCopyDescription	
UTTypeCopyPreferredTagWithClass	
UTTypeCreateAllIdentifiersForTag	
UTTypeCreatePreferredIdentifierForTag	
UTTypeEqual	

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.

kUTExportedTypeDeclarationsKey	
kUTImportedTypeDeclarationsKey	
kUTTagClassFilenameExtension	
kUTTagClassMIMEType	
kUTTagClassNSPboardType	
kUTTagClassOSType	
kUTTypeConformsToKey	
kUTTypeDescriptionKey	
kUTTypeIconFileKey	
kUTTypeIdentifierKey	
kUTTypeReferenceURLKey	
kUTTypeTagSpecificationKey	
kUTTypeVersionKey	

Metadata

MDExternalDatastore.h

Functions

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

`MDOIDEnumerationHasMoreOIDs`	
`MDOIDEnumerationNextOID`	
`MDResponseChannelSendObject`	
`MDResponseChannelSendOID`	

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.

`kMDExternalDatastoreStoreInterfaceID`	
`kMDExternalDatastoreTypeID`	
`MDExternalDatastoreQueryRef`	
`MDExternalDatastoreRef`	
`MDOIDEnumerationRef`	
`MDResponseChannelRef`	

MDImporter.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.

`kMDExporterInterfaceID`	

MDItem.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.

kMDItemAltitude	
kMDItemAuthorEmailAddresses	
kMDItemCFBundleIdentifier	
kMDItemContentTypeTree	
kMDItemDirector	
kMDItemEditors	
kMDItemEXIFGPSVersion	
kMDItemFSHasCustomIcon	
kMDItemFSIsStationery	
kMDItemGenre	
kMDItemGPSTrack	
kMDItemImageDirection	
kMDItemInformation	
kMDItemLatitude	
kMDItemLongitude	
kMDItemOriginalFormat	
kMDItemOriginalSource	
kMDItemPerformers	
kMDItemProducer	
kMDItemRecipientEmailAddresses	
kMDItemSpeed	
kMDItemSupportFileType	
kMDItemTheme	
kMDItemTimestamp	

kMDItemURL	
MD_AVAIL_LEOPARD	
MD_DEPRECATED	

MDLineage.h

Functions

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

MDLineageCopyChildren	
MDLineageCopyLeafDescendants	
MDLineageCopyParents	
MDLineageCopyRootAncestors	
MDLineageCreate	Creates and returns a new, unique lineage.
MDLineageCreateBranch	
MDLineageCreateBranchFromFile	Creates and returns a new lineage descending from a given file.
MDLineageCreateFromFile	Creates and returns a new, unique lineage using data from a given file.
MDLineageCreateQueryString	
MDLineageRemoveFromFile	Removes any existing lineage from a given file.
MDLineageSetOnFile	Writes the given lineage data to a given file.

MDQuery.h

Functions

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

MDQuerySetMaxCount	

MDSchema.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.

`kMDAttributeReadOnlyValues`	
`kMDExporterAvaliable`	

OSServices

CSIdentity.h

Functions

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

`CSIdentityAddAlias`	
`CSIdentityAddMember`	
`CSIdentityAuthenticateUsingPassword`	
`CSIdentityCommit`	
`CSIdentityCommitAsynchronously`	
`CSIdentityCreate`	
`CSIdentityCreateCopy`	
`CSIdentityCreateGroupMembershipQuery`	
`CSIdentityCreatePersistentReference`	
`CSIdentityDelete`	
`CSIdentityGetAliases`	
`CSIdentityGetAuthority`	
`CSIdentityGetCertificate`	
`CSIdentityGetClass`	
`CSIdentityGetEmailAddress`	
`CSIdentityGetFullName`	

CSIdentityGetImageData	
CSIdentityGetImageDataType	
CSIdentityGetImageURL	
CSIdentityGetPosixID	
CSIdentityGetPosixName	
CSIdentityGetTypeID	
CSIdentityGetUUID	
CSIdentityIsCommitting	
CSIdentityIsEnabled	
CSIdentityIsHidden	
CSIdentityIsMemberOfGroup	
CSIdentityRemoveAlias	
CSIdentityRemoveClient	
CSIdentityRemoveMember	
CSIdentitySetCertificate	
CSIdentitySetEmailAddress	
CSIdentitySetFullName	
CSIdentitySetImageData	
CSIdentitySetImageURL	
CSIdentitySetIsEnabled	
CSIdentitySetPassword	

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.

CSIdentityClass	
CSIdentityClientContext	
CSIdentityFlags	
CSIdentityQueryRef	

CSIdentityRef	
CSIdentityStatusUpdatedCallback	
kCSIdentityAuthorityNotAccessibleErr	
kCSIdentityClassGroup	
kCSIdentityClassUser	
kCSIdentityCommitCompleted	
kCSIdentityDeletedErr	
kCSIdentityDuplicateFullNameErr	
kCSIdentityDuplicatePosixNameErr	
kCSIdentityErrorDomain	
kCSIdentityFlagHidden	
kCSIdentityFlagNone	
kCSIdentityGeneratePosixName	
kCSIdentityInvalidFullNameErr	
kCSIdentityInvalidPosixNameErr	
kCSIdentityPermissionErr	
kCSIdentityUnknownAuthorityErr	

CSIdentityAuthority.h

Functions

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

CSGetDefaultIdentityAuthority	
CSGetLocalIdentityAuthority	
CSGetManagedIdentityAuthority	
CSIdentityAuthorityCopyLocalizedName	
CSIdentityAuthorityGetTypeID	

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.

`CSIdentityAuthorityRef`	

CSIdentityQuery.h

Functions

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

`CSIdentityQueryCopyResults`	
`CSIdentityQueryCreate`	
`CSIdentityQueryCreateForCurrentUser`	
`CSIdentityQueryCreateForName`	
`CSIdentityQueryCreateForPersistentReference`	
`CSIdentityQueryCreateForPosixID`	
`CSIdentityQueryCreateForUUID`	
`CSIdentityQueryExecute`	
`CSIdentityQueryExecuteAsynchronously`	
`CSIdentityQueryGetTypeID`	
`CSIdentityQueryStop`	

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.

`CSIdentityQueryClientContext`	
`CSIdentityQueryEvent`	
`CSIdentityQueryFlags`	
`CSIdentityQueryReceiveEventCallback`	
`CSIdentityQueryStringComparisonMethod`	
`kCSIdentityQueryEventErrorOccurred`	

kCSIdentityQueryEventResultsAdded	
kCSIdentityQueryEventResultsChanged	
kCSIdentityQueryEventResultsRemoved	
kCSIdentityQueryEventSearchPhaseFinished	
kCSIdentityQueryGenerateUpdateEvents	
kCSIdentityQueryIncludeHiddenIdentities	
kCSIdentityQueryStringBeginsWith	
kCSIdentityQueryStringEquals	

IconStorage.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.

kIconServices128PixelDataARGB	
kIconServices16PixelDataARGB	
kIconServices32PixelDataARGB	
kIconServices48PixelDataARGB	
kIconServices512PixelDataARGB	

KeychainCore.h

Functions

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

KCMakeKCRefFromFSRef	

10.4 Symbol Changes

This article lists the symbols added to `CoreServices.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.

CFNetwork

CFHTTPAuthentication.h

Functions

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

`CFHTTPAuthenticationAppliesToRequest`	Returns a Boolean value that indicates whether a CFHTTPAuthentication object is associated with a CFHTTPMessage object.
`CFHTTPAuthenticationCopyDomains`	Returns an array of domain URLs to which a given CFHTTPAuthentication object can be applied.
`CFHTTPAuthenticationCopyMethod`	Gets the strongest authentication method that will be used when a CFHTTPAuthentication object is applied to a request.
`CFHTTPAuthenticationCopyRealm`	Gets an authentication information's namespace.
`CFHTTPAuthenticationCreateFromResponse`	Uses an authentication failure response to create a CFHTTPAuthentication object.
`CFHTTPAuthenticationGetTypeID`	Gets the Core Foundation type identifier for the CFHTTPAuthentication opaque type.

`CFHTTPAuthenticationIsValid`	Returns a Boolean value that indicates whether a CFHTTPAuthentication object is valid.
`CFHTTPAuthenticationRequiresAccountDomain`	Returns a Boolean value that indicates whether a CFHTTPAuthentication object uses an authentication method that requires an account domain.
`CFHTTPAuthenticationRequiresOrderedRequests`	Returns a Boolean value that indicates whether authentication requests should be made one at a time.
`CFHTTPAuthenticationRequiresUserNameAndPassword`	Returns a Boolean value that indicates whether a CFHTTPAuthentication object uses an authentication method that requires a username and a password.
`CFHTTPMessageApplyCredentialDictionary`	Use a dictionary containing authentication credentials to perform the authentication method specified by a CFHTTPAuthentication object.
`CFHTTPMessageApplyCredentials`	Performs the authentication method specified by a CFHTTPAuthentication 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.

`CFHTTPAuthenticationRef`	An opaque reference representing HTTP authentication information.
`CFStreamErrorHTTPAuthentication`	Authentication error codes that may be returned when trying to apply authentication to a request.
`kCFHTTPAuthenticationAccountDomain`	Account domain to use for authentication.
`kCFHTTPAuthenticationPassword`	Password to use for authentication.
`kCFHTTPAuthenticationUsername`	
`kCFStreamErrorHTTPAuthenticationBadPassword`	Password is in a format that is not suitable for the request. Currently, passwords are decoded using kCFStringEncodingISOLatin1.

kCFStreamErrorHTTPAuthenticationBadUserName	User name is in a format that is not suitable for the request. Currently, user names are decoded using kCFStringEncodingISOLatin1.
kCFStreamErrorHTTPAuthenticationTypeUnsupported	Specified authentication type is not supported.

CFNetDiagnostics.h

Functions

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

CFNetDiagnosticCopyNetworkStatusPassively	Gets a network status value.
CFNetDiagnosticCreateWithStreams	Creates a network diagnostic object from a pair of CFStreams.
CFNetDiagnosticCreateWithURL	Creates a CFNetDiagnosticRef from a CFURLRef.
CFNetDiagnosticDiagnoseProblemInteractively	Opens a Network Diagnostics window.
CFNetDiagnosticSetName	Overrides the displayed application name.

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.

CFNetDiagnosticRef	An opaque reference representing a CFNetDiagnostic.
CFNetDiagnosticStatus	A CFIndex type that is used to return status values from CFNetDiagnostic status and diagnostic functions. For a list of possible values, see "CFNetDiagnosticStatusValues Constants."
CFNetDiagnosticStatusValues	Constants for diagnostic status values.
kCFNetDiagnosticConnectionDown	The connection does not appear to be working.
kCFNetDiagnosticConnectionIndeterminate	The status of the connection is not known.
kCFNetDiagnosticConnectionUp	The connection appears to be working.
kCFNetDiagnosticErr	An error occurred that prevented the call from completing.
kCFNetDiagnosticNoErr	No error occurred but there is no status.

CFNetServices.h

Functions

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

`CFNetServiceCreateDictionaryWithTXTData`	Uses TXT record data to create a dictionary.
`CFNetServiceCreateTXTDataWithDictionary`	Flattens a set of key/value pairs into a CFDataRef suitable for passing to CFNetServiceSetTXTData.
`CFNetServiceGetTargetHost`	Queries a CFNetService for its target hosts.
`CFNetServiceGetTXTData`	Queries a network service for the contents of its TXT records.
`CFNetServiceMonitorCreate`	Creates an instance of a NetServiceMonitor object that watches for record changes.
`CFNetServiceMonitorGetTypeID`	Gets the Core Foundation type identifier for all CFNetServiceMonitor instances.
`CFNetServiceMonitorInvalidate`	Invalidates an instance of a Network Service monitor object.
`CFNetServiceMonitorScheduleWithRunLoop`	Schedules a CFNetServiceMonitor on a run loop.
`CFNetServiceMonitorStart`	Starts monitoring.
`CFNetServiceMonitorStop`	Stops a CFNetServiceMonitor.
`CFNetServiceMonitorUnscheduleFromRunLoop`	Unschedules a CFNetServiceMonitor from a run loop.
`CFNetServiceRegisterWithOptions`	Makes a CFNetService available on the network.
`CFNetServiceResolveWithTimeout`	Gets the IP address or addresses for a CFNetService.
`CFNetServiceSetTXTData`	Sets the TXT record for a CFNetService.

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.

`CFNetServiceMonitorClientCallBack`	Defines a pointer to the callback function that is to be called when a monitored record type changes.
`CFNetServiceMonitorRef`	An opaque reference for a service monitor.
`CFNetServiceMonitorType`	Record type specifier used to tell a service monitor the type of record changes to watch for.

kCFNetServiceFlagIsDefault	If set, the resulting domain is the default registration or browse domain, depending on the context.
kCFNetServiceFlagNoAutoRename	Causes registrations to fail if a name conflict occurs.
kCFNetServiceMonitorTXT	Watch for TXT record changes.
kCFNetServicesErrorTimeout	Resolution failed because the timeout was reached.

CFNetworkDefs.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.

CFNetwork_EXPORT	

CFSocketStream.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.

kCFStreamPropertyProxyLocalBypass	Proxy Local Bypass property key.
kCFStreamPropertySSLPeerCertificates	SSL Peer Certificates property key for copy operations, which return a CFArray object containing SecCertificateRefs.
kCFStreamPropertySSLSettings	SSL Settings property key for set operations.
kCFStreamSSLAllowsAnyRoot	
kCFStreamSSLAllowsExpiredCertificates	
kCFStreamSSLAllowsExpiredRoots	
kCFStreamSSLCertificates	
kCFStreamSSLIsServer	
kCFStreamSSLLevel	
kCFStreamSSLPeerName	
kCFStreamSSLValidatesCertificateChain	

CarbonCore

Aliases.h

Functions

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

`GetAliasSize`	Gets the size of an alias record referenced by a handle.
`GetAliasSizeFromPtr`	Gets the size of an alias record referenced by a pointer.
`GetAliasUserType`	Gets the user type for an alias record referenced by a handle.
`GetAliasUserTypeFromPtr`	Gets the user type for the alias record referenced by a pointer.
`SetAliasUserType`	Sets the user type for an alias record referenced by a handle.
`SetAliasUserTypeWithPtr`	Sets the user type for the alias record referenced by a pointer.

Components.h

Functions

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

`GetComponentInstanceA5`	Obsolete.

Debugging.h

Functions

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

`GetMacOSStatusCommentString`	
`GetMacOSStatusErrorString`	

Devices.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.

DCtlHandle	

Endian.h

Functions

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

CoreEndianFlipData	Calls the flipper callback associated with the specified data type.
CoreEndianGetFlipper	Obtains the flipper callback that is installed for the specified data type.
CoreEndianInstallFlipper	Installs a flipper callback for the specified data 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.

CoreEndianFlipProc	Defines a pointer to a callback function that byte-swaps your custom data.
kCoreEndianAppleEventManagerDomain	Specifies that the domain is limited to Apple events.
kCoreEndianResourceManagerDomain	Specifies that the domain is limited to the resources for a specific application.

Files.h

Functions

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

FSCopyDADiskForVolume	
FSCopyObjectAsync	Starts an asynchronous file operation to copy a source object to a destination directory.
FSCopyObjectSync	Copies a source object to a destination directory.
FSCreateFileAndOpenForkUnicode	
FSCreateStringFromHFSUniStr	
FSFileOperationCancel	Cancels an asynchronous file operation.

FSFileOperationCopyStatus	Gets a copy of the current status information for an asynchronous file operation.
FSFileOperationCreate	Creates an object that represents an asynchronous file operation.
FSFileOperationGetTypeID	Returns the Core Foundation type identifier for the FSFileOperation opaque type.
FSFileOperationScheduleWithRunLoop	Schedules an asynchronous file operation with the specified run loop and mode.
FSFileOperationUnscheduleFromRunLoop	Unschedules an asynchronous file operation from the specified run loop and mode.
FSFileSecurityCopyAccessControlList	
FSFileSecurityCreate	
FSFileSecurityCreateWithFSPermissionInfo	
FSFileSecurityGetGroup	
FSFileSecurityGetGroupUUID	
FSFileSecurityGetMode	
FSFileSecurityGetOwner	
FSFileSecurityGetOwnerUUID	
FSFileSecurityGetTypeID	
FSFileSecurityRefCreateCopy	
FSFileSecuritySetAccessControlList	
FSFileSecuritySetGroup	
FSFileSecuritySetGroupUUID	
FSFileSecuritySetMode	
FSFileSecuritySetOwner	
FSFileSecuritySetOwnerUUID	
FSGetHFSUniStrFromString	
FSGetVolumeForDADisk	
FSGetVolumeForDiskID	
FSIsFSRefValid	
FSLockRange	Locks a range of bytes of the specified fork.

FSMoveObjectAsync	Starts an asynchronous file operation to move a source object to a destination directory.
FSMoveObjectSync	Moves a source object to a destination directory.
FSPathCopyObjectAsync	Starts an asynchronous file operation to copy a source object to a destination directory using pathnames.
FSPathCopyObjectSync	Copies a source object to a destination directory using pathnames.
FSPathFileOperationCopyStatus	Gets a copy of the current status information for an asynchronous file operation that uses pathnames.
FSPathMakeRefWithOptions	Converts a POSIX-style pathname into an FSRef structure with options.
FSPathMoveObjectAsync	Starts an asynchronous file operation to move a source object to a destination directory using pathnames.
FSPathMoveObjectSync	Moves a source object to a destination directory using pathnames.
FSUnlockRange	Unlocks a range of bytes of the specified fork.
PBCreateFileAndOpenForkUnicodeAsync	
PBCreateFileAndOpenForkUnicodeSync	
PBXLockRangeAsync	Locks a range of bytes of the specified fork.
PBXLockRangeSync	Locks a range of bytes of the specified fork.
PBXUnlockRangeAsync	Unlocks a range of bytes of the specified fork.
PBXUnlockRangeSync	Unlocks a range of bytes of the specified fork.

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.

bIsOnExternalBus	
bIsOnInternalBus	
bIsRemovable	
bNoRootTimes	
bSupportsExtendedFileSecurity	

FSFileOperationClientContext	Specifies user-defined data and callbacks associated with an asynchronous file operation.
FSFileOperationRef	Defines an opaque type that represents an asynchronous file operation.
FSFileOperationStage	Constants used by the File Manager to indicate the current stage of an asynchronous file operation.
FSFileOperationStatusProcPtr	Defines a status callback function for an asynchronous file operation on an FSRef object.
FSFileSecurityRef	
FSPathFileOperationStatusProcPtr	Defines a status callback function for an asynchronous file operation on an object specified with a pathname.
FSRangeLockParam	Defines a parameter block for use with 64-bit range locking functions.
FSRangeLockParamPtr	Defines a pointer to a range lock parameter block.
FSRefForkIOParam	
FSRefForkIOParamPtr	
kFSAllowConcurrentAsyncIOBit	
kFSAllowConcurrentAsyncIOMask	
kFSCatInfoFSFileSecurityRef	
kFSEjectVolumeForceEject	
kFSFileOperationDefaultOptions	Use the following default options:
kFSFileOperationDoNotMoveAcrossVolumes	If asked to move an object across volume boundaries, abort the operation.
kFSFileOperationOverwrite	If the destination directory contains an object with the same name as a source object, overwrite the destination object.
kFSFileOperationSkipPreflight	Skip the preflight stage for a directory move or copy operation.
kFSFileOperationSkipSourcePermissionErrors	If a source object cannot be read, skip the object and continue the operation.
kFSFileSecurityRemoveACL	
kFSForceReadBit	

kFSForceReadMask	
kFSNewLineBit	
kFSNewLineCharMask	
kFSNewLineMask	
kFSNoCacheBit	
kFSNoCacheMask	
kFSOperationBytesCompleteKey	The value for this key is a CFNumber that represents the total number of bytes that have already been moved or copied by this file operation.
kFSOperationBytesRemainingKey	The value for this key is a CFNumber that represents the total number of bytes that remain to be moved or copied by this file operation.
kFSOperationObjectsCompleteKey	The value for this key is a CFNumber that represents the total number of objects that have already been moved or copied by this file operation.
kFSOperationObjectsRemainingKey	The value for this key is a CFNumber that represents the total number of objects that remain to be moved or copied by this file operation.
kFSOperationStageComplete	The file operation is complete.
kFSOperationStagePreflighting	The File Manager is performing tasks such as calculating the sizes and number of objects in the operation, and checking to make sure there is enough space on the destination volume to complete the operation.
kFSOperationStageRunning	The File Manager is copying or moving the file or directory.
kFSOperationStageUndefined	The File Manager has not started the file operation.
kFSOperationThroughputKey	The value for this key is a CFNumber that represents the current throughput of this file operation in bytes per second.
kFSOperationTotalBytesKey	The value for this key is a CFNumber that represents the total number of bytes that will be moved or copied by this file operation.

kFSOperationTotalObjectsKey	The value for this key is a CFNumber that represents the total number of objects that will be moved or copied by this file operation.
kFSOperationTotalUserVisibleObjectsKey	The value for this key is a CFNumber that represents the total number of user-visible objects that will be moved or copied by this file operation.
kFSOperationUserVisibleObjectsCompleteKey	The value for this key is a CFNumber that represents the total number of user-visible objects that have already been moved or copied by this file operation.
kFSOperationUserVisibleObjectsRemainingKey	The value for this key is a CFNumber that represents the total number of user-visible objects that remain to be moved or copied by this file operation.
kFSPathMakeRefDefaultOptions	Use the default options.
kFSPathMakeRefDoNotFollowLeafSymlink	When converting a path that refers to a symbolic link, do not follow the link. The new FSRef should refer to the link itself.
kFSPleaseCacheBit	
kFSPleaseCacheMask	
kFSRdVerifyBit	
kFSRdVerifyMask	
kFSUnmountVolumeForceUnmount	

Finder.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.

kExtendedFlagObjectIsBusy	

FixMath.h

Functions

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

`UnsignedFixedMulDiv`	Performs multiplications and divisions on unsigned fixed-point numbers.

Folders.h

Functions

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

`DetermineIfPathIsEnclosedByFolder`	Determines whether a file path is enclosed inside a special folder type for the given domain.
`FSDetermineIfRefIsEnclosedByFolder`	Determines whether a file of type FSRef is enclosed inside a special folder type for the given domain.
`FSpDetermineIfSpecIsEnclosedByFolder`	Determines whether a file of type FSSpec is enclosed inside a special folder type for the given domain.

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.

`kAudioPresetsFolderType`	
`kAutomatorWorkflowsFolderType`	
`kAutosaveInformationFolderType`	
`kFolderManagerNewlyCreatedFolderIsLocalizedBit`	
`kFolderManagerNewlyCreatedFolderShouldHaveDotLocalizedCreatedWithinMask`	
`kFolderManagerNotCreatedOnRemoteVolumesBit`	
`kFolderManagerNotCreatedOnRemoteVolumesMask`	
`kSpotlightSavedSearchesFolderType`	
`kTemporaryItemsInUserDomainFolderType`	

Gestalt.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.

gestaltAliasMgrRequiresAccessors	
gestaltBusClkSpeedMHz	
gestaltCountOfCPUs	
gestaltCPU970FX	
gestaltCPUPentium4	
gestaltFSAllowsConcurrentAsyncIO	
gestaltPortableUSBANSIKbd	
gestaltPortableUSBISOKbd	
gestaltPortableUSBJISKbd	
gestaltThirdPartyANSIKbd	
gestaltThirdPartyISOKbd	
gestaltThirdPartyJISKbd	
gestaltTSMgr23	
gestaltUnknownThirdPartyKbd	
gestaltX86AdditionalFeatures	
gestaltX86HasCID	
gestaltX86HasCX16	
gestaltX86HasDSCPL	
gestaltX86HasEST	
gestaltX86HasMONITOR	
gestaltX86HasSMX	
gestaltX86HasSSE3	
gestaltX86HasSupplementalSSE3	
gestaltX86HasTM2	
gestaltX86HasVMX	
gestaltX86HasxTPR	

MacErrors.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.

cfragCFragRsrcErr	'cfrg' resource could not be loaded.
cfragExecFileRefErr	Bundle does not have valid executable file.
cfragMapFileErr	A file could not be mapped.
cfragRsrcForkErr	Resource fork could not be opened.
cfragStdFolderErr	Could not find standard CFM folder.
errCoreEndianDataDoesNotMatchFormat	
errCoreEndianDataTooLongForFormat	
errCoreEndianDataTooShortForFormat	
errCppbad_alloc	
errCppbad_cast	
errCppbad_exception	
errCppbad_typeid	
errCppdomain_error	
errCppGeneral	
errCppinvalid_argument	
errCppios_base_failure	
errCppLastSystemDefinedError	
errCppLastUserDefinedError	
errCpplength_error	
errCpplogic_error	
errCppout_of_range	
errCppoverflow_error	

errCpprange_error	
errCppruntime_error	
errCppunderflow_error	
errFSAttributeNotFound	
errFSNotEnoughSpaceForOperation	
errFSOperationNotSupported	
errFSPropertyNotValid	

MacTypes.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.

ALLOW_OBSOLETE_CARBON_MACMEMORY	
ALLOW_OBSOLETE_CARBON_OSUTILS	
ProcessSerialNumber	Defines the unique identifier for an open process.
ProcessSerialNumberPtr	

MachineExceptions.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.

FPRegIntel	
FPUInformationIntel	
MachineInformationIntel	
RegisterInformationIntel	
Vector128Intel	
VectorInformationIntel	

Math64.h

Functions

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

`LongDoubleToSInt64`	
`LongDoubleToUInt64`	
`SInt64ToLongDouble`	
`SInt64ToWide`	
`UInt64ToLongDouble`	
`UInt64ToUnsignedWide`	
`UnsignedWideToUInt64`	
`WideToSInt64`	

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.

`LongDoubleToSInt64`	
`LongDoubleToUInt64`	
`SInt64ToLongDouble`	
`UInt64ToLongDouble`	

Multiprocessing.h

Functions

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

`MPRemoteCallCFM`	Calls a non-reentrant function and blocks the current task.

MultiprocessingInfo.h

Functions

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

`MPGetNextCpuID`	Obtains the next CPU ID in the list of physical processors of the specified memory coherence group.
`MPGetNextTaskID`	Obtains the next task ID in the list of available tasks.

NumberFormatting.h

Functions

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

`numtostring`	

OSUtils.h

Functions

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

`DTUninstall`	

TextCommon.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.

`kUCCharPropTypeDecimalDigitValue`	
`kUnicodeNormalizationFormC`	
`kUnicodeNormalizationFormD`	

TextUtils.h

Functions

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

`c2pstr`	Converts a C string to a Pascal string.
`C2PStr`	Converts a C string to a Pascal string.
`p2cstr`	Converts a Pascal string to a C string.
`P2CStr`	Converts a Pascal string to a C string.

UnicodeUtilities.h

Functions

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

`DisposeIndexToUCStringUPP`	
`InvokeIndexToUCStringUPP`	
`NewIndexToUCStringUPP`	
`UCTypeSelectAddKeyToSelector`	
`UCTypeSelectCompare`	
`UCTypeSelectCreateSelector`	
`UCTypeSelectFindItem`	
`UCTypeSelectFlushSelectorData`	
`UCTypeSelectReleaseSelector`	
`UCTypeSelectWalkList`	
`UCTypeSelectWouldResetBuffer`	

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.

`IndexToUCStringProcPtr`	

IndexToUCStringUPP	
kUCTSDirectionNext	
kUCTSDirectionPrevious	
kUCTSOptionsDataIsOrderedMask	
kUCTSOptionsNoneMask	
kUCTSOptionsReleaseStringMask	
kUCTypeSelectMaxListSize	
UCTSWalkDirection	
UCTypeSelectCompareResult	
UCTypeSelectOptions	
UCTypeSelectRef	

fp.h

Functions

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

ldtox80	
x80told	

Metadata

MDImporter.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.

kMDImporterInterfaceID	Importers must implement an interface corresponding to this UUID.
kMDImporterTypeID	Only importers with this type ID are loaded.

MDItem.h

Functions

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

`MDItemCopyAttribute`	Returns the value of the specified attribute in the metadata item.
`MDItemCopyAttributeList`	Returns the values of the specified attributes in the metadata item.
`MDItemCopyAttributeNames`	Returns an array containing the attribute names existing in the metadata item.
`MDItemCopyAttributes`	Returns the values of the specified attributes in the metadata item.
`MDItemCreate`	Creates an MDItem object for a file at the specified path.
`MDItemGetTypeID`	Returns the type identifier of all MDItem instances.

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.

`kMDItemAcquisitionMake`	The manufacturer of the device used to aquire the document contents. A CFString.
`kMDItemAcquisitionModel`	The model of the device used to aquire the document contents. For example, 100, 200, 400, etc. A CFString.
`kMDItemAlbum`	The title for a collection of media. This is analagous to a record album, or photo album. A CFString.
`kMDItemAperture`	The aperature setting used to aqure the document contents. This unit is the APEX value. A CFNumber.
`kMDItemAppleLoopDescriptors`	Specifies multiple pieces of descriptive information about a loop. Besides genre and instrument, files can contain descriptive information that help users in refining searches. A CFArray of CFStrings.
`kMDItemAppleLoopsKeyFilterType`	Specifies key filtering information about a loop.
`kMDItemAppleLoopsLoopMode`	Specifies how a file should be played.
`kMDItemAppleLoopsRootKey`	Specifies the loop's original key. The key is the root note or tonic for the loop, and does not include the scale type. A CFString.
`kMDItemAttributeChangeDate`	The date and time of the last change made to a metadata attribute. A CFDate.

kMDItemAudiences	The audience for which the file is intended. The audience may be determined by the creator or the publisher or by a third party. A CFArray of CFStrings.
kMDItemAudioBitRate	The audio bit rate. A CFNumber.
kMDItemAudioChannelCount	Number of channels in the audio data contained in the file.
kMDItemAudioEncodingApplication	The name of the application that encoded the data contained in the audio file. A CFString.
kMDItemAudioSampleRate	Sample rate of the audio data contained in the file. The sample rate is a float value representing hz (audio_frames/second). For example: 44100.0, 22254.54. A CFNumber.
kMDItemAudioTrackNumber	The track number of a song or composition when it is part of an album. A CFNumber (integer).
kMDItemAuthors	The author, or authors, of the contents of the file. The order of the authors is preserved, but does not represent the main author or relative importance of the authors. A CFArray of CFStrings.
kMDItemBitsPerSample	The number of bits per sample. For example, the bit depth of an image (8-bit, 16-bit etc...) or the bit depth per audio sample of uncompressed audio data (8, 16, 24, 32, 64, etc..). A CFNumber.
kMDItemCity	Identifies city of origin according to guidelines established by the provider. For example, "New York", "Cupertino", or "Toronto". A CFString.
kMDItemCodecs	The codecs used to encode/decode the media. A CFArray of CFStrings.
kMDItemColorSpace	The color space model used by the document contents. For example, “RGB”, “CMYK”, “YUV”, or “YCbCr”. A CFString.
kMDItemComment	A comment related to the file. This differs from the Finder comment, kMDItemFinderComment. A CFString.
kMDItemComposer	The composer of the music contained in the audio file. A CFString.
kMDItemContactKeywords	A list of contacts that are associated with this document, not including the authors. A CFArray of CFStrings.
kMDItemContentCreationDate	The date that the contents of the file were created. This is different than the file creation date. Its can be used to store when the file contents were first created, or first modified. A CFDate.
kMDItemContentModificationDate	The date and time that the contents of the file were last modified. This is not necessarily the file modification date. A CFDate.

kMDItemContentType	The UTI pedigree of a file.
kMDItemContributors	The entities responsible for making contributions to the content of the resource. Examples of a contributor include a person, an organization or a service. A CFArray of CFStrings.
kMDItemCopyright	The copyright owner of the file contents. A CFString.
kMDItemCountry	The full, publishable name of the country or primary location where the intellectual property of the item was created, according to guidelines of the provider. A CFString.
kMDItemCoverage	The extent or scope of the content of the resource.
kMDItemCreator	Application used to create the document content (e.g. "Word", "AppleWorks", etc.). A CFString.
kMDItemDeliveryType	The delivery type. Values are “Fast start” or “RTSP”. A CFString.
kMDItemDescription	A description of the content of the resource. The description may include an abstract, table of contents, reference to a graphical representation of content or a free-text account of the content. A CFString.
kMDItemDisplayName	The localized version of the file name. This is the localized version of the LaunchServices call LSCopyDisplayNameForURL()/LSCopyDisplayNameForRef(). A CFString.
kMDItemDueDate	Date this item is due. A CFDate.
kMDItemDurationSeconds	The duration, in seconds, of the content of file. A value of 10.5 represents media that is 10 and 1/2 seconds long. A CFNumber.
kMDItemEmailAddresses	Email addresses related to this item. A CFArray of CFStrings.
kMDItemEncodingApplications	Application used to convert the original content into it's current form. For example, a PDF file might have an encoding application set to "Distiller". A CFArray of CFStrings.
kMDItemEXIFVersion	The version of the EXIF header used to generate the metadata. A CFString.
kMDItemExposureMode	The exposure mode used to aquire the document contents. Possible values are 0 (auto exposure), 1 (manual exposure) and 2 (auto bracket). A CFNumber.
kMDItemExposureProgram	The class of the exposure program used by the camera to set exposure when the image is taken. Possible values include: Manual, Normal, and Aperture priority.
kMDItemExposureTimeSeconds	The exposure time, in seconds, used to aquire the document contents. A CFNumber.
kMDItemExposureTimeString	The time of the exposure. A CFString.

kMDItemFinderComment	Finder comments for this file. A CFString.
kMDItemFlashOnOff	Indicates if a camera flash was used. A CFNumber.
kMDItemFNumber	The diameter of the diaphragm aperture in terms of the effective focal length of the lens.
kMDItemFocalLength	The actual focal length of the lens, in millimeters. A CFNumber.
kMDItemFonts	Fonts used in this item. You should store the font's full name, the postscript name, or the font family name, based on the available information. A CFArray of CFStrings.
kMDItemFSContentChangeDate	The date the file contents last changed. A CFDate.
kMDItemFSCreationDate	The date and time that the file was created. A CFDate.
kMDItemFSExists	This attribute is deprecated and was never implemented.
kMDItemFSInvisible	Indicates whether the file is invisible. A CFBoolean.
kMDItemFSIsExtensionHidden	Indicates whether the file extension of the file is hidden. A CFBoolean.
kMDItemFSIsReadable	This attribute is deprecated and was never implemented.
kMDItemFSIsWriteable	This attribute is deprecated and was never implemented.
kMDItemFSLabel	Index of the Finder label of the file. Possible values are 0 through 7. A CFNumber.
kMDItemFSName	The file name of the item. A CFString.
kMDItemFSNodeCount	Number of files in a directory. A CFNumber.
kMDItemFSOwnerGroupID	The group ID of the owner of the file. A CFNumber.
kMDItemFSOwnerUserID	The user ID of the owner of the file. A CFNumber.
kMDItemFSSize	The size, in bytes, of the file on disk. A CFNumber.
kMDItemHasAlphaChannel	Indicates if this image file has an alpha channel. A CFBoolean.
kMDItemHeadline	A publishable entry providing a synopsis of the contents of the file. For example, "Apple Introduces the iPod Photo". A CFString.
kMDItemIdentifier	A formal identifier used to reference the resource within a given context. A CFString.
kMDItemInstantMessageAddresses	Instant message addresses related to this item. A CFArray of CFStrings.
kMDItemInstructions	Editorial instructions concerning the use of the item, such as embargoes and warnings. For example, "Second of four stories". A CFString.

kMDItemIsGeneralMIDISequence	Indicates whether the MIDI sequence contained in the file is setup for use with a General MIDI device. A CFBoolean.
kMDItemISOSpeed	The ISO speed used to aquire the document contents. A CFNumber.
kMDItemKeySignature	The key of the music contained in the audio file. For example: C, Dm, F#m, Bb. A CFString.
kMDItemKeywords	Keywords associated with this file. For example, "Birthday", "Important", etc. A CFArray of CFStrings.
kMDItemKind	A description of the kind of item this file represents. A CFString.
kMDItemLanguages	Indicates the languages of the intellectual content of the resource. Recommended best practice for the values of the Language element is defined by RFC 3066. A CFArray of CFStrings.
kMDItemLastUsedDate	The date and time that the file was last used. This value is updated automatically by LaunchServices everytime a file is opened by double clicking, or by asking LaunchServices to open a file. A CFDate.
kMDItemLayerNames	The names of the layers in the file. A CFArray of CFStrings.
kMDItemLyricist	The lyricist, or text writer, of the music contained in the audio file. A CFString.
kMDItemMaxAperture	The smallest f-number of the lens. The unit is the APEX?? value. Ordinarily it is given in the range of 00.00 to 99.99. A CFNumber.
kMDItemMediaTypes	The media types present in the content. A CFArray of CFStrings.
kMDItemMeteringMode	The metering mode used to take the image. Possible values are: Unknown, Average, CenterWeightedAverage, Spot, MultiSpot, Pattern, and Partial. A CFString.
kMDItemMusicalGenre	The musical genre of the song or composition contained in the audio file. For example: Jazz, Pop, Rock, Classical. A CFString.
kMDItemMusicalInstrumentCategory	Specifies the category of an instrument.
kMDItemMusicalInstrumentName	Specifies the name of instrument relative to the instrument category.
kMDItemNumberOfPages	Number of pages in the document. A CFNumber.
kMDItemOrganizations	The company or organization that created the document. A CFArray of CFStrings.
kMDItemOrientation	The orientation of the document contents. Possible values are 0 (landscape) and 1 (portrait). A CFNumber.

kMDItemPageHeight	Height of the document page, in points (72 points per inch). For PDF files this indicates the height of the first page only. A CFNumber.
kMDItemPageWidth	Width of the document page, in points (72 points per inch). For PDF files this indicates the width of the first page only. A CFNumber.
kMDItemPath	The complete path to the file. A CFString.
kMDItemPhoneNumbers	Phone numbers related to this item. A CFArray of CFStrings.
kMDItemPixelHeight	The height, in pixels, of the contents. For example, the image height or the video frame height. A CFNumber.
kMDItemPixelWidth	The width, in pixels, of the contents. For example, the image width or the video frame width. A CFNumber.
kMDItemProfileName	The name of the color profile used by the document contents. A CFString.
kMDItemProjects	The list of projects that this file is part of. For example, if you were working on a movie all of the files could be marked as belonging to the project “My Movie”. A CFArray of CFStrings.
kMDItemPublishers	The entity responsible for making the resource available. For example, a person, an organization, or a service. Typically, the name of a publisher should be used to indicate the entity. A CFArray of CFStrings.
kMDItemRecipients	Recipients of this item. A CFArray of CFStrings.
kMDItemRecordingDate	The recording date of the song or composition. This is in contrast to kMDItemContentCreationDate which, could indicate the creation date of an edited or 'mastered' version of the original art. A CFDate.
kMDItemRecordingYear	Indicates the year the item was recorded. For example, 1964, 2003, etc. A CFNumber.
kMDItemRedEyeOnOff	Indicates if red-eye reduction was used to take the picture. Possible values are 0 (no red-eye reduction mode or unknown) and 1 (red-eye reduction used). A CFBoolean.
kMDItemResolutionHeightDPI	Resolution height, in DPI, of this image. A CFNumber.
kMDItemResolutionWidthDPI	Resolution width, in DPI, of this image. A CFNumber.
kMDItemRights	Provides a link to information about rights held in and over the resource.
kMDItemSecurityMethod	The security or encryption method used for the file. A CFNumber.

kMDItemStarRating	User rating of this item. For example, the stars rating of an iTunes track. A CFNumber.
kMDItemStateOrProvince	Identifies the province or state of origin according to guidelines established by the provider. For example, "CA", "Ontario", or "Sussex". A CFString.
kMDItemStreamable	Whether the content is prepared for streaming. A CFBoolean.
kMDItemSubject	
kMDItemTempo	A float value that specifies the beats per minute of the music contained in the audio file. A CFNumber.
kMDItemTextContent	Contains a text representation of the content of the document.
kMDItemTimeSignature	The time signature of the musical composition contained in the audio/MIDI file. For example: "4/4", "7/8". A CFString.
kMDItemTitle	The title of the file. For example, this could be the title of a document, the name of an song, or the subject of an email message. A CFString.
kMDItemTotalBitRate	The total bit rate, audio and video combined, of the media. A CFNumber.
kMDItemVersion	The version number of this file. A CFString
kMDItemVideoBitRate	The video bit rate. A CFNumber.
kMDItemWhereFroms	Describes where the file was obtained from. For example, a downloaded file may refer to the URL, files received by email may indicate the sender's email address, message subject, etc. A CFArray of CFStrings.
kMDItemWhiteBalance	The white balance setting used to aquire the document contents. Possible values are 0 (auto white balance) and 1 (manual). A CFNumber.
MD_AVAIL	
MD_BEGIN_C_DECLS	
MD_END_C_DECLS	
MD_EXPORT	
MDItemCopyAttributeList	
MDItemRef	A reference to a MDItem object.

MDQuery.h

Functions

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

`MDQueryCopyQueryString`	Returns the query string of the query.
`MDQueryCopySortingAttributes`	Returns the list of attribute names used to sort the results.
`MDQueryCopyValueListAttributes`	Returns the list of attribute names for which values are being collected by the query.
`MDQueryCopyValuesOfAttribute`	Returns the list of values from the results of the query for the specified attribute.
`MDQueryCreate`	Creates a new query instance.
`MDQueryCreateSubset`	Creates a new query that is a subset of the specified parentquery.
`MDQueryDisableUpdates`	Disables updates to the query result list.
`MDQueryEnableUpdates`	Enables updates to the query result list.
`MDQueryExecute`	Run the query, and populate the query with the results.
`MDQueryGetAttributeValueOfResultAtIndex`	Returns the value of the named attribute for the result at the given index.
`MDQueryGetBatchingParameters`	Returns the current parameters that control the batching of progress notifications.
`MDQueryGetCountOfResultsWithAttributeValue`	Returns the number of results which have the given attribute and attribute value.
`MDQueryGetIndexOfResult`	Returns the current index of the given result.
`MDQueryGetResultAtIndex`	Returns the current result at the given index.
`MDQueryGetResultCount`	Returns the number of results currently collected by the query.
`MDQueryGetTypeID`	Returns the type identifier of all MDQuery instances
`MDQueryIsGatheringComplete`	Returns true if the first phase of a query, the initial result gathering, has finished.
`MDQuerySetBatchingParameters`	Set the query batching parameters.

MDQuerySetCreateResultFunction	Sets the function used to create the result objects of the MDQuery.
MDQuerySetCreateValueFunction	Sets the function used to create the value objects of the MDQuery.
MDQuerySetSearchScope	Sets the search scope for a query instance.
MDQuerySetSortComparator	Sets the function used to sort the results of an MDQuery.
MDQueryStop	Stops the query from generating more results.

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.

kMDQueryDidFinishNotification	Posted to indicate that the query has finished the initial result-gathering phase.
kMDQueryDidUpdateNotification	Notification posted to indicate that a change has occured to the query's resuls list during the live-update phase of a query's execution.
kMDQueryProgressNotification	Notification posted to indicate that a change has occurred to the query's results list during the initial result-gathering phase of execution.
kMDQueryResultContentRelevance	A CFNumberRef with a floating point value between 0.0 and 1.0 inclusive.
kMDQueryScopeComputer	Specifies that the query should be restricted to all locally mounted volumes, plus the user's home directory (which may be on a remote volume).
kMDQueryScopeHome	Specifies that the query should be restricted to the volume and directory that contains the current user's home directory.
kMDQueryScopeNetwork	Specifies that the query should include all user mounted remote volumes.
kMDQuerySynchronous	Specifies that a query should block during the initial gather phase. The query's run loop will run in the default mode. If this option is not specified the query function returns immediately after starting the query asynchronously.
kMDQueryUpdateAddedItems	An array that identifies the items that have been added to the query results. This list only contains result objects that have previously been created, result objects that have not been created are not included.

`kMDQueryUpdateChangedItems`	An array that identifies the items that have changed in the query results. This list only contains result objects that have previously been created, result objects that have not been created are not included.
`kMDQueryUpdateRemovedItems`	An array that identifies the items that have been removed from the query results. This list only contains result objects that have previously been created, result objects that have not been created are not included.
`kMDQueryWantsUpdates`	Specifies that a query should provide live-updates to the result list after the initial gathering phase.
`MDQueryBatchingParams`	Structure containing the progress notification batching parameters of a MDQuery.
`MDQueryCreateResultFunction`	Callback function used to create the result objects stored and returned by a query.
`MDQueryCreateValueFunction`	Callback function used to create the value objects stored and returned by a query.
`MDQueryOptionFlags`	Specify the execution mode for a query.
`MDQueryRef`	A reference to a MDQuery object.
`MDQuerySortComparatorFunction`	Callback function used to sort the results of a query.

MDSchema.h

Functions

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

`MDSchemaCopyAllAttributes`	Returns an array containing all the metadata attributes defined in the schema.
`MDSchemaCopyAttributesForContentType`	Returns a dictionary containing the metadata attributes for the specified UTI type.
`MDSchemaCopyDisplayDescriptionForAttribute`	Returns the localized description of a metadata attribute key.
`MDSchemaCopyDisplayNameForAttribute`	Returns the localized display name of a metadata attribute key.
`MDSchemaCopyMetaAttributesForAttribute`	Returns a dictionary describing the values for the specified metadata attribute key.

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.

`kMDAttributeAllValues`	An array of strings containing available the metadata attribute keys, or NULL if the type is not known by the system.
`kMDAttributeDisplayValues`	An array of strings containing the available display metadata attribute keys, or NULL if the type is not known by the system.
`kMDAttributeMultiValued`	A boolean that indicates if the metadata attribute value is multi-valued. If this is TRUE, the metadata attribute value is an array of the types specified in kMDAttributeType.
`kMDAttributeName`	A string containing the name of the metadata attribute key.
`kMDAttributeType`	A CFNumberRef or CFTypeId describing the type of data returned as the value of the metadata attribute key.

OSServices

IconStorage.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.

`kIconServices256PixelDataARGB`	

NSLCore.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.

`NSLLibraryPresent`	

SystemSound.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.

kSystemSoundClientTimedOutError	

SearchKit

SKAnalysis.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.

kSKEndTermChars	Additional valid last-position “word” characters for indexing and querying.
kSKMaximumTerms	The maximum number of number unique terms to index in each document.
kSKProximityIndexing	A Boolean flag indicating whether or not Search Kit should use proximity indexing.
kSKStartTermChars	Additional valid starting-position “word” characters for indexing and querying.
kSKTermChars	Additional valid starting-position “word” characters for indexing and querying.

SKIndex.h

Functions

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

SKIndexClose	Closes an index.

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.

SKDocumentID	Defines an opaque data type representing a lightweight document identifier.

SKSearch.h

Functions

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

`SKIndexCopyDocumentRefsForDocumentIDs`	Gets document URL objects based on document IDs.
`SKIndexCopyDocumentURLsForDocumentIDs`	Gets document URLs based on document IDs.
`SKIndexCopyInfoForDocumentIDs`	Gets document names and parent IDs based on document IDs.
`SKSearchCancel`	Cancels an asynchronous search request.
`SKSearchCreate`	Creates an asynchronous search object for querying an index, and initiates search.
`SKSearchFindMatches`	Extracts search result information from a search object.
`SKSearchGetTypeID`	Gets the type identifier for Search Kit search objects.

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.

`kSKSearchOptionDefault`	Default search options include:
`kSKSearchOptionFindSimilar`	This option alters query behavior so that Search Kit returns references to documents that are similar to an example text string. When this option is specified, Search Kit ignores all query operators.
`kSKSearchOptionNoRelevanceScores`	This option saves time during a search by suppressing the computation of relevance scores.
`kSKSearchOptionSpaceMeansOR`	This option alters query behavior so that spaces are interpreted as Boolean OR operators.
`SKSearchOptions`	Specifies the search options available for the SKSearchCreate function.
`SKSearchRef`	Defines an opaque data type representing a an asynchronous search.

SKSummary.h

Functions

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

`SKSummaryCopyParagraphAtIndex`	Gets a specified paragraph from the text in a summarization object.
`SKSummaryCopyParagraphSummaryString`	Gets a text string consisting of a summary with, at most, the requested number of paragraphs.
`SKSummaryCopySentenceAtIndex`	Gets a specified sentence from the text in a summarization object.
`SKSummaryCopySentenceSummaryString`	Gets a text string consisting of a summary with, at most, the requested number of sentences.
`SKSummaryCreateWithString`	Creates a summary object based on a text string.
`SKSummaryGetParagraphCount`	Gets the number of paragraphs in a summarization object.
`SKSummaryGetParagraphSummaryInfo`	Gets detailed information about a body of text for constructing a custom paragraph-based summary string.
`SKSummaryGetSentenceCount`	Gets the number of sentences in a summarization object.
`SKSummaryGetSentenceSummaryInfo`	Gets detailed information about a body of text for constructing a custom sentence-based summary string.
`SKSummaryGetTypeID`	Gets the type identifier for Search Kit summarization objects.

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.

`SKSummaryRef`	Defines an opaque data type representing summarization information.

10.3 Symbol Changes

This article lists the symbols added to `CoreServices.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.

CFNetwork

CFFTPStream.h

Functions

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

`CFFTPCreateParsedResourceListing`	Parses an FTP listing to a dictionary.
`CFReadStreamCreateWithFTPURL`	Creates an FTP read stream.
`CFWriteStreamCreateWithFTPURL`	Creates an FTP write 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.

`kCFFTPResourceGroup`	CFDictionary key for getting the CFString containing the name of a group that shares the FTP resource.
`kCFFTPResourceLink`	CFDictionary key for getting the CFString containing the symbolic link information. If the item is a symbolic link, the CFString contains the path to the item that the link references.
`kCFFTPResourceModDate`	CFDictionary key for getting the CFDate containing the last date and time the FTP resource was modified.

kCFFTPResourceMode	CFDictionary key for getting the CFNumber containing the access permissions, defined in sys/types.h, of the FTP resource.
kCFFTPResourceName	CFDictionary key for getting the CFString containing the name of the FTP resource.
kCFFTPResourceOwner	CFDictionary key for getting the CFString containing the name of the owner of the FTP resource.
kCFFTPResourceSize	CFDictionary key for getting the CFNumber containing the size in bytes of the FTP resource.
kCFFTPResourceType	CFDictionary key for getting the CFNumber containing the type of the FTP resource as defined in sys/dirent.h.
kCFStreamErrorDomainFTP	Error domain that returns the last result code returned by the FTP server.
kCFStreamPropertyFTPAttemptPersistentConnection	FTP Attempt Persistent Connection stream property key for set and copy operations.
kCFStreamPropertyFTPFetchResourceInfo	FTP Fetch Resource Information stream property key for set and copy operations.
kCFStreamPropertyFTPFileTransferOffset	FTP File Transfer Offset stream property key for set and copy operations. The value of this property is a CFNumber of type kCFNumberLongLongType representing the file offset at which to start the transfer.
kCFStreamPropertyFTPPassword	FTP Password stream property key for set and copy operations. A value of type CFString for storing the login password. Don't set this property when anonymous FTP is desired.
kCFStreamPropertyFTPProxy	FTP Proxy stream property key for set and copy operations. The property is a value of type CFDictionary that holds proxy dictionary key-value pairs. The dictionary returned by SystemConfiguration can also be set as the value of this property.
kCFStreamPropertyFTPProxyHost	FTP Proxy Host stream property key or an FTP Proxy dictionary key for set and copy operations.

kCFStreamPropertyFTPProxyPassword	FTP Proxy Port stream property key or FTP Proxy dictionary key for set and copy operations.
kCFStreamPropertyFTPProxyPort	FTP Proxy Port stream property key or an FTP Proxy dictionary key for set and copy operations.
kCFStreamPropertyFTPProxyUser	FTP Proxy Host stream property key or FTP Proxy dictionary key for set and copy operations.
kCFStreamPropertyFTPResourceSize	FTP Resource Size read stream property key copy operations. This property stores a CFNumber of type kCFNumberLongLongType representing the size of a resource in bytes.
kCFStreamPropertyFTPUsePassiveMode	FTP Passive Mode stream property key for set and copy operations. Set this property to kCFBooleanTrue to enable passive mode; set this property to kCFBooleanFalse to disable passive mode.
kCFStreamPropertyFTPUserName	FTP User Name stream property key for set and copy operations. A value of type CFString for storing the login user name. Don't set this property when anonymous FTP is desired.

CFHTTPStream.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.

kCFStreamPropertyHTTPRequestBytesWrittenCount	HTTP Request Bytes Written property.

CFHost.h

Functions

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

CFHostCancelInfoResolution	Cancels the resolution of a host.
CFHostCreateCopy	Creates a new host object by copying.

`CFHostCreateWithAddress`	Uses an address to create an instance of a host object.
`CFHostCreateWithName`	Uses a name to create an instance of a host object.
`CFHostGetAddressing`	Gets the addresses from a host.
`CFHostGetNames`	Gets the names from a CFHost.
`CFHostGetReachability`	Gets reachability information from a host.
`CFHostGetTypeID`	Gets the Core Foundation type identifier for the CFHost opaque type.
`CFHostScheduleWithRunLoop`	Schedules a CFHost on a run loop.
`CFHostSetClient`	Associates a client context and a callback function with a CFHost object or disassociates a client context and callback function that were previously set.
`CFHostStartInfoResolution`	Starts resolution for a host object.
`CFHostUnscheduleFromRunLoop`	Unschedules a CFHost from a run loop.

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.

`CFHostClientCallBack`	Defines a pointer to the callback function that is called when an asynchronous resolution of a CFHost completes or an error occurs for an asynchronous CFHost resolution.
`CFHostClientContext`	A structure containing user-defined data and callbacks for CFHost objects.
`CFHostInfoType`	Values indicating the type of data that is to be resolved or the type of data that was resolved.
`CFHostRef`	An opaque reference representing an CFHost object.
`kCFHostAddresses`	Specifies that addresses are to be resolved or that addresses were resolved.
`kCFHostNames`	Specifies that names are to be resolved or that names were resolved.
`kCFHostReachability`	Specifies that reachability information is to be resolved or that reachability information was resolved.
`kCFStreamErrorDomainNetDB`	The error domain that returns errors from the network database (DNS resolver) layer (described in /usr/include/netdb.h).

`kCFStreamErrorDomainSystemConfiguration`	The error domain that returns errors from the system configuration layer (described in System Configuration Framework Reference).

CFNetServices.h

Functions

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

`CFNetServiceCreateCopy`	Creates a copy of a CFNetService object.

CFSocketStream.h

Functions

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

`CFStreamCreatePairWithSocketToCFHost`	Creates readable and writable streams connected to a given CFHost object.
`CFStreamCreatePairWithSocketToNetService`	Creates a pair of streams for a CFNetService.

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.

`kCFStreamPropertySocketRemoteHost`	
`kCFStreamPropertySocketRemoteNetService`	

CarbonCore

Components.h

Functions

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

`CSGetComponentsThreadMode`	Indicates whether using thread-unsafe components is allowed in the current thread.

CSSetComponentsThreadMode	Sets whether or not using thread-unsafe components is allowed in the current thread.

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.

cmpThreadSafe	Available in Mac OS X v10.3 and later.
CSComponentsThreadMode	kCSAcceptAllComponentsMode Available in Mac OS X v10.3 and later.Components.h kCSAcceptThreadSafe-ComponentsOnlyMode Available in Mac OS X v10.3 and later.Components.h
kCSAcceptAllComponentsMode	Available in Mac OS X v10.3 and later.
kCSAcceptThreadSafeComponentsOnlyMode	Available in Mac OS X v10.3 and later.
platformIA32NativeEntryPoint	Available in Mac OS X v10.3 and later.

Files.h

Functions

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

FSCopyURLForVolume	Returns a copy of the URL for a volume.

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.

bDoNotDisplay	The volume should not be displayed in the user interface.
bIsCasePreserving	The volume is preserves case.
bIsCaseSensitive	The volume is case-sensitive.
bNoVolumeSizes	The volume is unable to report volume size or free space.
bSupportsJournaling	The volume supports journaling. This does not indicate whether journaling is currently enabled on the volume.
fsSBGroupID	

fsSBGroupIDBit	
fsSBSkipHiddenItems	
fsSBSkipHiddenItemsBit	
fsSBSkipPackageContents	
fsSBSkipPackageContentsBit	
fsSBUserID	
fsSBUserIDBit	
kFNNotifyInBackground	Specify this option if you want to receive notifications on this subscription when your application is in background.
kFSCatInfoSetOwnership	Attempt to set the file's user and group (UID and GID). If the File Manager cannot set the the user or group ID, the call fails. (Mac OS X only).
kFSMountServerMarkDoNotDisplay	
kFSMountServerMountOnMountDir	
kFSMountServerMountWithoutNotification	
kFSVolFlagJournalingActiveBit	
kFSVolFlagJournalingActiveMask	

FixMath.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.

ColorToFract	
ff	
FixedRound	
FixedSquareRoot	
FixedToFloat	
FixedToFract	
FixedToInt	
FixedTruncate	

fl	
FloatToFixed	
FloatToFract	
FractToColor	
FractToFixed	
FractToFloat	
IntToFixed	

Folders.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.

kClassicPreferencesFolderType	
kDeveloperApplicationsFolderType	
kDictionariesFolderType	
kLogsFolderType	
kMagicTemporaryItemsFolderType	
kPreferencePanesFolderType	
kRedirectedRelativeFolder	
kTemporaryItemsInCacheDataFolderType	

Gestalt.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.

gestaltATSUBiDiCursorPositionFeature	If the bit specified by this mask is set, support for bidirectional cursor positioning is available.
gestaltATSUDecimalTabFeature	If the bit specified by this mask is set, your application can set a decimal tab character.

gestaltATSUDropShadowStyleFeature	If the bit specified by this mask is set, drop shadow features are available.
gestaltATSUHighlightColorControlFeature	If the bit specified by this mask is set, your application can control highlight color.
gestaltATSUNearestCharLineBreakFeature	If the bit specified by this mask is set, the nearest character line break feature is available.
gestaltATSUStrikeThroughStyleFeature	If the bit specified by this mask is set, strike through styles are available.
gestaltATSUUnderlineOptionsStyleFeature	If the bit specified by this mask is set, underline options are available.
gestaltATSUUpdate7	Indicates that version 2.5 of ATSUI is installed on the user's system. Available beginning with ATSUI 2.5, in Mac OS X version 10.3.
gestaltCPU970	
gestaltCPUG47447	
gestaltMenuMgrCGImageMenuTitleBit	
gestaltMenuMgrCGImageMenuTitleMask	
gestaltPhysicalRAMSizeInMegabytes	
gestaltPowerPCASArchitecture	
gestaltPowerPCHas64BitSupport	
gestaltPowerPCHasDCBTStreams	
gestaltPowerPCIgnoresDCBST	
gestaltProcClkSpeedMHz	
gestaltProcessorCacheLineSize	
gestaltProF16ANSIKbd	
gestaltProF16ISOKbd	
gestaltProF16JISKbd	
gestaltQuickTimeThreadSafeFeaturesAttr	
gestaltQuickTimeThreadSafeGraphicsExport	
gestaltQuickTimeThreadSafeGraphicsImport	

gestaltQuickTimeThreadSafeICM	
gestaltQuickTimeThreadSafeMovieExport	
gestaltQuickTimeThreadSafeMovieImport	
gestaltQuickTimeThreadSafeMoviePlayback	
gestaltQuickTimeThreadSafeMovieToolbox	
gestaltSystemVersionBugFix	The bug fix version number.
gestaltSystemVersionMajor	The major system version number.
gestaltSystemVersionMinor	The minor system version number.
gestaltTSMgr22	
gestaltUSBProF16ANSIKbd	
gestaltUSBProF16ISOKbd	
gestaltUSBProF16JISKbd	
gestaltX86Features	
gestaltX86HasAPIC	
gestaltX86HasCLFSH	
gestaltX86HasCMOV	
gestaltX86HasCX8	
gestaltX86HasDE	
gestaltX86HasDS	
gestaltX86HasFPU	
gestaltX86HasFXSR	
gestaltX86HasHTT	
gestaltX86HasMCA	
gestaltX86HasMCE	
gestaltX86HasMMX	
gestaltX86HasMSR	
gestaltX86HasMTRR	
gestaltX86HasPAE	

gestaltX86HasPAT	
gestaltX86HasPGE	
gestaltX86HasPSE	
gestaltX86HasPSE36	
gestaltX86HasPSN	
gestaltX86HasSEP	
gestaltX86HasSS	
gestaltX86HasSSE	
gestaltX86HasSSE2	
gestaltX86HasTM	
gestaltX86HasTSC	
gestaltX86HasVME	
gestaltX86ResACPI	
gestaltX86Serviced20	
gestaltX86VectorUnit	
gestaltX86VectorUnitMMX	
gestaltX86VectorUnitNone	
gestaltX86VectorUnitSSE	
gestaltX86VectorUnitSSE2	

MacErrors.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.

componentNotThreadSafeErr	
kATSUNoFontNameErr	
kUCTokenizerIterationFinished	
kUCTokenizerUnknownLang	
kUCTokenNotFound	

tsmComponentPropertyNotFoundErr	
tsmComponentPropertyUnsupportedErr	
tsmInputModeChangeFailedErr	

Script.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.

langGreekAncient	
langNynorsk	
verEastAsiaGeneric	
verFlemishPoint	

TextCommon.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.

kBig5_DOSVariant	
kDOSJapanesePalmVariant	
kDOSJapaneseStandardVariant	
kEUC_CN_BasicVariant	
kEUC_CN_DOSVariant	
kEUC_KR_BasicVariant	
kEUC_KR_DOSVariant	
kISOLatin1MusicCDVariant	
kISOLatin1StandardVariant	
kISOLatinArabicExplicitOrderVariant	
kISOLatinArabicImplicitOrderVariant	
kISOLatinArabicVisualOrderVariant	

kISOLatinHebrewExplicitOrderVariant	
kISOLatinHebrewImplicitOrderVariant	
kISOLatinHebrewVisualOrderVariant	
kMacGreekDefaultVariant	
kMacGreekEuroSignVariant	
kMacGreekNoEuroSignVariant	
kShiftJIS_BasicVariant	
kShiftJIS_DOSVariant	
kShiftJIS_MusicCDVariant	
kTextEncodingANSEL	
kTextEncodingBig5_E	
kTextEncodingISOLatin10	
kTextEncodingJIS_X0213_MenKuTen	
kTextEncodingKOI8_U	
kTextEncodingUnicodeV4_0	
kTextEncodingVISCII	
kUnicodeSCSUFormat	
kUnicodeUTF16BEFormat	
kUnicodeUTF16Format	
kUnicodeUTF16LEFormat	
kUnicodeUTF32BEFormat	
kUnicodeUTF32Format	
kUnicodeUTF32LEFormat	
kWindowsLatin1PalmVariant	
kWindowsLatin1StandardVariant	

TextEncodingConverter.h

Functions

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

TECCopyTextEncodingInternetNameAndMIB	
TECGetTextEncodingFromInternetNameOrMIB	
TECSetBasicOptions	

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.

kTEC_MIBEnumDontCare	
kTECInternetNameDefaultUsageMask	
kTECInternetNameStrictUsageMask	
kTECInternetNameTolerantUsageMask	
TECInternetNameUsageMask	

TextEncodingPlugin.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.

kTECAvailableEncodingsResType	
kTECAvailableSniffersResType	
kTECConversionInfoResType	
kTECInternetNamesResType	
kTECMailEncodingsResType	
kTECPluginCreator	
kTECPluginManyToOne	
kTECPluginOneToMany	

kTECPluginOneToOne	
kTECPluginSniffObj	
kTECPluginType	
kTECResourceID	
kTECSubTextEncodingsResType	
kTECWebEncodingsResType	
TECEncodingPairRec	
TECEncodingPairs	
TECEncodingPairsHandle	
TECEncodingPairsPtr	
TECEncodingPairsRec	
TECEncodingsListHandle	
TECEncodingsListPtr	
TECEncodingsListRec	
TECInternetNameRec	
TECInternetNamesHandle	
TECInternetNamesPtr	
TECInternetNamesRec	
TECLocaleListToEncodingListPtr	
TECLocaleListToEncodingListRec	
TECLocaleToEncodingsListHandle	
TECLocaleToEncodingsListPtr	
TECLocaleToEncodingsListRec	
TECSubTextEncodingRec	
TECSubTextEncodingsHandle	
TECSubTextEncodingsPtr	
TECSubTextEncodingsRec	
TextEncodingRec	

verUnspecified	

UnicodeConverter.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.

kUnicodeUseExternalEncodingFormBit	
kUnicodeUseExternalEncodingFormMask	

UnicodeUtilities.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.

kUCTextBreakParagraphMask	

fp.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.

ldtox80	
x80told	

OSServices

SystemSound.h

Functions

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

AlertSoundPlayCustomSound	

DisposeSystemSoundCompletionUPP	
InvokeSystemSoundCompletionUPP	
NewSystemSoundCompletionUPP	
SystemSoundRemoveCompletionRoutine	
SystemSoundSetCompletionRoutine	

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.

SystemSoundCompletionProcPtr	
SystemSoundCompletionUPP	

SearchKit

SKAnalysis.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.

kSKLanguageTypes	Deprecated—Search Kit ignores this constant.
kSKMinTermLength	The minimum term length to index. Specified as a CFNumber object. If this optional key is not present, Search Kit indexing defaults to a minimum term length of 1.
kSKStopWords	A set of stopwords—words not to index. Specified as a CFSet object. There is no default stopword list. You must supply your own.
kSKSubstitutions	A dictionary of term substitutions—terms that differ in their character strings but that match during a search. Specified as a CFDictionary object.

SKDocument.h

Functions

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

SKDocumentCopyURL	Builds a CFURL object from a document URL object.
SKDocumentCreate	Creates a document URL object based on a scheme, parent, and name.
SKDocumentCreateWithURL	Creates a document URL object from a CFURL object.
SKDocumentGetName	Gets the name of a document URL object.
SKDocumentGetParent	Gets a document URL object's parent.
SKDocumentGetSchemeName	Gets the scheme name for a document URL object.
SKDocumentGetTypeID	Gets the type identifier for Search Kit document URL objects.

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.

SKDocumentRef	Defines an opaque data type representing a document's URL.

SKIndex.h

Functions

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

SKIndexAddDocument	Adds location information for a file-based document, and the document's textual content, to an index.
SKIndexAddDocumentWithText	Adds a document URL object, and the associated document's textual content, to an index.
SKIndexCompact	Invokes all pending updates associated with an index, compacts the index if compaction is needed, and commits all changes to backing store.
SKIndexCopyDocumentForDocumentID	Obtains a document URL object from an index.
SKIndexCopyDocumentIDArrayForTermID	Obtains document IDs for documents that contain a given term.
SKIndexCopyDocumentProperties	Obtains the application-defined properties of an indexed document.
SKIndexCopyTermIDArrayForDocumentID	Obtains the IDs for the terms of an indexed document.
SKIndexCopyTermStringForTermID	Obtains a term, specified by ID, from an index.
SKIndexCreateWithMutableData	Creates a named index stored in a CFMutableData object.

SKIndexCreateWithURL	Creates a named index in a file whose location is specified with a CFURL object.
SKIndexDocumentIteratorCopyNext	Obtains the next document URL object from an index using a document iterator.
SKIndexDocumentIteratorCreate	Creates an index-based iterator for document URL objects owned by a parent document URL object.
SKIndexDocumentIteratorGetTypeID	Gets the type identifier for Search Kit document iterators.
SKIndexFlush	Invokes all pending updates associated with an index and commits them to backing store.
SKIndexGetAnalysisProperties	Gets the text analysis properties of an index.
SKIndexGetDocumentCount	Gets the total number of documents represented in an index.
SKIndexGetDocumentID	Gets the ID of a document URL object in an index.
SKIndexGetDocumentState	Gets the current indexing state of a document URL object in an index.
SKIndexGetDocumentTermCount	Gets the number of terms for a document in an index.
SKIndexGetDocumentTermFrequency	Gets the number of occurrences of a term in a document.
SKIndexGetIndexType	Gets the category of an index.
SKIndexGetMaximumBytesBeforeFlush	Not recommended. Gets the memory size limit for updates to an index, measured in bytes.
SKIndexGetMaximumDocumentID	Gets the highest-numbered document ID in an index.
SKIndexGetMaximumTermID	Gets the highest-numbered term ID in an index.
SKIndexGetTermDocumentCount	Gets the number of documents containing a given term represented in an index.
SKIndexGetTermIDForTermString	Gets the ID for a term in an index.
SKIndexGetTypeID	Gets the type identifier for Search Kit indexes.
SKIndexMoveDocument	Changes the parent of a document URL object in an index.
SKIndexOpenWithData	Opens an existing, named index for searching only.
SKIndexOpenWithMutableData	Opens an existing, named index for searching and updating.
SKIndexOpenWithURL	Opens an existing, named index stored in a file whose location is specified with a CFURL object.
SKIndexRemoveDocument	Removes a document URL object and its children, if any, from an index.

`SKIndexRenameDocument`	Changes the name of a document URL object in an index.
`SKIndexSetDocumentProperties`	Sets the application-defined properties of a document URL object.
`SKIndexSetMaximumBytesBeforeFlush`	Not recommended. Sets the memory size limit for updates to an index, measured in bytes.
`SKLoadDefaultExtractorPlugIns`	Tells Search Kit to use the Spotlight metadata importers.

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.

`kSKDocumentStateAddPending`	Specifies that the document is not in the index but will be added after the index is flushed or closed.
`kSKDocumentStateDeletePending`	Specifies that the document is in the index but will be deleted after the index is flushed or closed.
`kSKDocumentStateIndexed`	Specifies that the document is indexed.
`kSKDocumentStateNotIndexed`	Specifies that the document is not indexed.
`kSKIndexInverted`	Specifies an inverted index, mapping terms to documents.
`kSKIndexInvertedVector`	Specifies an index type with all the capabilities of an inverted and a vector index.
`kSKIndexUnknown`	Specifies an unknown index type.
`kSKIndexVector`	Specifies a vector index, mapping documents to terms.
`SKDocumentIndexState`	The indexing state of a document.
`SKIndexDocumentIteratorRef`	Defines an opaque data type representing an index-based document iterator.
`SKIndexRef`	Defines an opaque data type representing an index.
`SKIndexType`	Specifies the category of an index.

SKSearch.h

Functions

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

`SKSearchGroupCopyIndexes`	Obtains the indexes for a search group.

SKSearchGroupCreate	Creates a search group as an array of references to indexes.
SKSearchGroupGetTypeID	Deprecated. Use asynchronous searching with SKSearchCreate instead, which does not employ search groups.
SKSearchResultsCopyMatchingTerms	Obtains the terms in a document that match a query.
SKSearchResultsCreateWithDocuments	Finds documents similar to given example documents.
SKSearchResultsCreateWithQuery	Queries the indexes in a search group.
SKSearchResultsGetCount	Gets the total number of found items in a search.
SKSearchResultsGetInfoInRange	Extracts information from a Search Kit query result.
SKSearchResultsGetTypeID	Gets the type identifier for Search Kit search results.

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.

kSKSearchBooleanRanked	Deprecated. Specifies a query that can include Boolean operators including '|', '&', '!', '(', and ')'.
kSKSearchPrefixRanked	Deprecated. Specifies a prefix-based search, which matches terms that begin with the query string.
kSKSearchRanked	Deprecated. Specifies a basic ranked search.
kSKSearchRequiredRanked	Deprecated. Specifies a query that can include required ('+') or excluded ('-') terms.
SKSearchGroupRef	Deprecated. Use asynchronous searching with SKSearchCreate instead, which does not employ search groups.
SKSearchResultsFilterCallBack	Deprecated. Use SKSearchCreate and SKSearchFindMatches instead, which do not use a callback.
SKSearchResultsRef	Deprecated. Use asynchronous searching with SKSearchCreate instead, which does not employ search groups.
SKSearchType	Search Kit ignores the constants in this group. Use asynchronous searching with SKSearchCreate instead, which uses query syntax to determine search type.

WebServicesCore

WSProtocolHandler.h

Functions

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

`WSProtocolHandlerCopyFaultDocument`	Creates a Fault XML response for a given WSProtocolHandler and fault details dictionary.
`WSProtocolHandlerCopyProperty`	Returns a copy of a property from a protocol handler reference.
`WSProtocolHandlerCopyReplyDictionary`	Parses an incoming XML document as if it were the reply of a method.
`WSProtocolHandlerCopyReplyDocument`	Creates a Reply XML document for a given WS ProtocolHandler and context dictionary.
`WSProtocolHandlerCopyRequestDictionary`	Parses an incoming XML document for the method name and parameters.
`WSProtocolHandlerCopyRequestDocument`	Creates an XML request for a given WSProtocolHandler and parameter list.
`WSProtocolHandlerCreate`	Creates a WSProtocolHandlerRef for use in translating an XML document.
`WSProtocolHandlerGetTypeID`	Returns a CFTypeID for the current WSProtocolHandlerRef.
`WSProtocolHandlerSetDeserializationOverride`	Specifies a callback to be made when parsing an XML method response.
`WSProtocolHandlerSetProperty`	Sets a property in a specified protocol handler.
`WSProtocolHandlerSetSerializationOverride`	Specifies a callback which will be called to produce the XML that represents the serialization of a given type ref.

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.

`kWSMethodName`	
`kWSMethodParameterOrder`	

`kWSMethodParameters`	
`WSProtocolHandlerDeserializationProcPtr`	This is an optional callback that handles custom deserialization of a particular data type for a protocol handler.
`WSProtocolHandlerRef`	An opaque reference to a web services protocol handler.
`WSProtocolHandlerSerializationProcPtr`	This is an optional callback that handles custom serialization of a particular data type for a protocol handler.

10.2 Symbol Changes

This article lists the symbols added to `CoreServices.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.

CFNetwork

CFHTTPStream.h

Functions

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

`CFReadStreamCreateForStreamedHTTPRequest`	Creates a read stream for a CFHTTP request message object whose body is too long to keep in memory.

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.

`kCFStreamPropertyHTTPAttemptPersistentConnection`	HTTP Attempt Persistent Connection property.
`kCFStreamPropertyHTTPFinalURL`	HTTP Final URL property. A value of type CFURL containing the final HTTP URL. This value differs from the URL in the original HTTP request if an autoredirection occurred. This property cannot be set.
`kCFStreamPropertyHTTPProxy`	HTTP Proxy property.
`kCFStreamPropertyHTTPProxyHost`	HTTP Proxy Host property.
`kCFStreamPropertyHTTPProxyPort`	HTTP Proxy Host property.

kCFStreamPropertyHTTPShouldAutoredirect	HTTP Should Auto Redirect property. Set this property to kCFBooleanTrue to enable autoredirection; set this property to kCFBooleanFalse to disable autoredirection.
kCFStreamPropertyHTTPSProxyHost	HTTPS Proxy Host property.
kCFStreamPropertyHTTPSProxyPort	HTTPS Proxy Host property.

CFNetServices.h

Functions

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

CFNetServiceBrowserCreate	Creates an instance of a Network Service browser object.
CFNetServiceBrowserGetTypeID	Gets the Core Foundation type identifier for the Network Service browser object.
CFNetServiceBrowserInvalidate	Invalidates an instance of a Network Service browser object.
CFNetServiceBrowserScheduleWithRunLoop	Schedules a CFNetServiceBrowser on a run loop.
CFNetServiceBrowserSearchForDomains	Searches for domains.
CFNetServiceBrowserSearchForServices	Searches a domain for services of a specified type.
CFNetServiceBrowserStopSearch	Stops a search for domains or services.
CFNetServiceBrowserUnscheduleFromRunLoop	Unschedules a CFNetServiceBrowser from a run loop and mode.
CFNetServiceCancel	Cancels a service registration or a service resolution.
CFNetServiceCreate	Creates an instance of a Network Service object.
CFNetServiceGetAddressing	Gets the IP addressing from a CFNetService.
CFNetServiceGetDomain	Gets the domain from a CFNetService.
CFNetServiceGetName	Gets the name from a CFNetService.
CFNetServiceGetProtocolSpecificInformation	This function gets protocol-specific information from a CFNetService.
CFNetServiceGetType	Gets the type from a CFNetService.

`CFNetServiceGetTypeID`	Gets the Core Foundation type identifier for the Network Service object.
`CFNetServiceRegister`	Makes a CFNetService available on the network.
`CFNetServiceResolve`	This function updates the specified CFNetService with the IP address or addresses associated with the service. Call CFNetServiceGetAddressing to get the addresses.
`CFNetServiceScheduleWithRunLoop`	Schedules a CFNetService on a run loop.
`CFNetServiceSetClient`	Associates a callback function with a CFNetService or disassociates a callback function from a CFNetService.
`CFNetServiceSetProtocolSpecificInformation`	Sets protocol-specific information for a CFNetService.
`CFNetServiceUnscheduleFromRunLoop`	Unschedules a CFNetService from a run loop.

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.

`CFNetServiceBrowserClientCallBack`	Defines a pointer to the callback function for a CFNetServiceBrowser.
`CFNetServiceBrowserRef`	An opaque reference representing a CFNetServiceBrowser.
`CFNetServiceClientCallBack`	Defines a pointer to the callback function for a CFNetService.
`CFNetServiceClientContext`	A structure provided when a CFNetService is associated with a callback function or when a CFNetServiceBrowser is created.
`CFNetServiceRef`	An opaque reference representing a CFNetService.
`CFNetServicesError`	Error codes that may be returned by CFNetServices functions or passed to CFNetServices callback functions.
`kCFNetServiceFlagIsDomain`	If set, the results pertain to a search for domains. If not set, the results pertain to a search for services.
`kCFNetServiceFlagIsRegistrationDomain`	If set, the resulting domain is the default registration domain.
`kCFNetServiceFlagMoreComing`	If set, a hint that the client's callback function will be called again soon; therefore, the client should not do anything time-consuming, such as updating the screen.

kCFNetServiceFlagRemove	If set, the client should remove the result item instead of adding it.
kCFNetServicesErrorBadArgument	A required argument was not provided.
kCFNetServicesErrorCancel	The search or service was canceled.
kCFNetServicesErrorCollision	An attempt was made to use a name that is already in use.
kCFNetServicesErrorInProgress	A search is already in progress.
kCFNetServicesErrorInvalid	Invalid data was passed to a CFNetServices function.
kCFNetServicesErrorNotFound	Not used.
kCFNetServicesErrorUnknown	An unknown CFNetService error occurred.
kCFStreamErrorDomainMach	Error domain returning errors reported by Mach. For more information, see the header file /usr/include/mach/error.h.
kCFStreamErrorDomainNetServices	Error domain returning errors reported by the service discovery APIs. These errors are only returned if you use the CFNetServiceBrowser API or any APIs introduced in Mac OS X v10.4 or later.

CFSocketStream.h

Functions

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

CFSocketStreamSOCKSGetError	
CFSocketStreamSOCKSGetErrorSubdomain	

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.

kCFStreamErrorDomainSOCKS	The error code is a SOCKS proxy error.
kCFStreamErrorSOCKS4IdConflict	The request was rejected because the client and identd reported different user IDs.
kCFStreamErrorSOCKS4IdentdFailed	The request was rejected because the SOCKS server cannot connect to identd on the client.
kCFStreamErrorSOCKS4RequestFailed	The request was rejected or failed.

kCFStreamErrorSOCKS4SubDomainResponse	The error code is the status code returned by the server.
kCFStreamErrorSOCKS5BadResponseAddr	The server responded with an address type that is not supported.
kCFStreamErrorSOCKS5BadState	At some point in the SOCKS protocol, a bad state was encountered.
kCFStreamErrorSOCKS5SubDomainMethod	The error code is the negotiation method the server wants to use.
kCFStreamErrorSOCKS5SubDomainResponse	The error code is the value the server returned in response to a connection request.
kCFStreamErrorSOCKS5SubDomainUserPass	The error code indicates that the server returned an error during username and password negotiation.
kCFStreamErrorSOCKSSubDomainNone	The error code is a general SOCKS error.
kCFStreamErrorSOCKSSubDomainVersionCode	The error code is the version of SOCKS the server wants to use.
kCFStreamErrorSOCKSUnknownClientVersion	The stream was told to perform a SOCKS protocol other than version 4 or 5.
kCFStreamPropertyShouldCloseNativeSocket	Should Close Native Socket property key.
kCFStreamPropertySocketSecurityLevel	Socket Security Level property key.
kCFStreamPropertySOCKSPassword	
kCFStreamPropertySOCKSProxy	SOCKS proxy property key.
kCFStreamPropertySOCKSProxyHost	
kCFStreamPropertySOCKSProxyPort	
kCFStreamPropertySOCKSUser	
kCFStreamPropertySOCKSVersion	
kCFStreamSocketSecurityLevelNegotiatedSSL	
kCFStreamSocketSecurityLevelNone	
kCFStreamSocketSecurityLevelSSLv2	
kCFStreamSocketSecurityLevelSSLv3	
kCFStreamSocketSecurityLevelTLSv1	
kCFStreamSocketSOCKSVersion4	
kCFStreamSocketSOCKSVersion5	

kSOCKS5NoAcceptableMethod	Other values indicate the server's desired method.

CarbonCore

Aliases.h

Functions

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

FSCopyAliasInfo	Returns information from an alias handle.
FSMatchAlias	Identifies a list of possible matches for an alias.
FSMatchAliasNoUI	Identifies a list of possible matches for an alias without any user interaction.
FSNewAliasMinimalUnicode	Creates a minimal alias, given the Unicode name and parent directory of the target.
FSNewAliasUnicode	Creates a new alias record, given the Unicode name and parent directory of the target.

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.

FSAliasInfo	Defines an information block passed to the FSCopyAliasInfo function.
FSAliasInfoBitmap	Returned by the FSCopyAliasInfo function to indicate which fields of the alias information structure contain valid data.
FSAliasInfoPtr	
kARMTryFileIDFirst	Perform a search using the file ID of the target before searching using the path.
kFSAliasInfoFinderInfo	The file type and creator information, in the fileType and fileCreator fields, is valid.
kFSAliasInfoFSInfo	The filesystem ID and signature, in the filesystemID and signature fields, are valid.
kFSAliasInfoIDs	The parent directory ID and alias target ID, in the parentDirID and nodeID fields, are valid.
kFSAliasInfoIsDirectory	The information in the isDirectory field is valid.

kFSAliasInfoNone	None of the alias information is valid.
kFSAliasInfoTargetCreateDate	The creation date of the alias target, in the targetCreateDate field, is valid.
kFSAliasInfoVolumeCreateDate	The volume creation date in the volumeCreateDate field is valid.
kFSAliasInfoVolumeFlags	The volume information, in the volumeIsBootVolume, volumeIsAutomounted, volumeIsEjectable, and volumeHasPersistentFileIDs fields, is valid.
kResolveAliasTryFileIDFirst	The Alias Manager should search for the alias target using file IDs before searching using the path.

DateTimeUtils.h

Functions

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

UCConvertCFAbsoluteTimeToLongDateTime	Converts a value of type CFAbsoluteTime to LongDateTime.
UCConvertCFAbsoluteTimeToSeconds	Converts a value of type CFAbsoluteTime to seconds.
UCConvertCFAbsoluteTimeToUTCDateTime	Converts a value of type CFAbsoluteTime to UTCDateTime.
UCConvertLongDateTimeToCFAbsoluteTime	Converts a value of type LongDateTime to CFAbsoluteTime.
UCConvertSecondsToCFAbsoluteTime	Converts a value from the normal seconds time representation to CFAbsoluteTime.
UCConvertUTCDateTimeToCFAbsoluteTime	Converts a value of type UTCDateTime time to CFAbsoluteTime.

Debugging.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.

DEBUG_ASSERT_COMPONENT_NAME_STRING	
DEBUG_ASSERT_MESSAGE	
DEBUG_ASSERT_PRODUCTION_CODE	

require_tasklevel0_action_quiet	
require_tasklevel0_quiet	
verify_tasklevel0	
verify_tasklevel0_string	

Files.h

Functions

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

DisposeFSVolumeEjectUPP	Deletes a universal procedure pointer (UPP) to your volume ejection callback function.
DisposeFSVolumeMountUPP	Deletes a universal procedure pointer (UPP) to your volume mount callback function.
DisposeFSVolumeUnmountUPP	Deletes a universal procedure pointer (UPP) to your volume unmount callback function.
FSCancelVolumeOperation	Cancels an outstanding asynchronous volume mounting operation.
FSCopyDiskIDForVolume	Returns a copy of the disk ID for a volume.
FSCreateVolumeOperation	Returns an FSVolumeOperation which can be used for an asynchronous volume operation.
FSDisposeVolumeOperation	Releases the memory associated with a volume operation.
FSEjectVolumeAsync	Asynchronously ejects a volume.
FSEjectVolumeSync	Ejects a volume.
FSGetAsyncEjectStatus	Returns the current status of an asynchronous eject operation.
FSGetAsyncMountStatus	Returns the current status of an asynchronous mount operation.
FSGetAsyncUnmountStatus	Returns the current status of an asynchronous unmount operation.
FSMountLocalVolumeAsync	Mounts a volume asynchronously.
FSMountLocalVolumeSync	Mounts a volume.
FSMountServerVolumeAsync	Mounts a server volume asynchronously.
FSMountServerVolumeSync	Mounts a server volume.

FSUnmountVolumeAsync	Unmounts a volume asynchronously.
FSUnmountVolumeSync	Unmounts a volume.
InvokeFSVolumeEjectUPP	Calls your volume ejection callback function.
InvokeFSVolumeMountUPP	Calls your volume mount callback function.
InvokeFSVolumeUnmountUPP	Calls your volume unmount callback function.
NewFSVolumeEjectUPP	Creates a new universal procedure pointer (UPP) to your volume ejection callback function.
NewFSVolumeMountUPP	Creates a new universal procedure pointer (UPP) to your volume mount callback function.
NewFSVolumeUnmountUPP	Creates a new universal procedure pointer (UPP) to your volume unmount callback function.

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.

bSupportsExclusiveLocks	The volume supports exclusive access to files opened for writing.
FSEjectStatus	
FSMountStatus	
FSUnmountStatus	
FSVolumeEjectProcPtr	
FSVolumeEjectUPP	For more information, see the description of the FSVolumeEjectProcPtr callback function.
FSVolumeMountProcPtr	
FSVolumeMountUPP	For more information, see the description of the FSVolumeMountProcPtr callback function.
FSVolumeOperation	
FSVolumeUnmountProcPtr	
FSVolumeUnmountUPP	For more information, see the description of the FSVolumeUnmountProcPtr callback function.
kAsyncEjectComplete	Available in Mac OS X v10.2 and later.
kAsyncEjectInProgress	Available in Mac OS X v10.2 and later.

kAsyncMountComplete	Available in Mac OS X v10.2 and later.
kAsyncMountInProgress	Available in Mac OS X v10.2 and later.
kAsyncUnmountComplete	Available in Mac OS X v10.2 and later.
kAsyncUnmountInProgress	Available in Mac OS X v10.2 and later.
kFSNodeHardLinkBit	Available in Mac OS X v10.2 and later.
kFSNodeHardLinkMask	Available in Mac OS X v10.2 and later.

Folders.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.

kBootTimeStartupItemsFolderType	
kFindByContentIndexesFolderType	
kIndexFilesFolderType	
kKeyboardLayoutsFolderType	
kManagedItemsFolderType	

Gestalt.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.

gestaltAliasMgrPrefersPath	
gestaltAltivecRegistersSwappedCorrectlyBit	Available in Mac OS X v10.2 and later.
gestaltATSUAscentDescentControlsFeature	If the bit specified by this mask is set, ascent and descent controls (kATSUDescentTag and kATSUAscentTag) are available.
gestaltATSUBatchBreakLinesFeature	If the bit specified by this mask is set, the ATSUBatchBreakLines function is available.
gestaltATSUByCharacterClusterFeature	If the bit specified by this mask is set, ATSUI cursor movement types are available.

gestaltATSUDirectAccess	If the bit specified by this mask is set, ATSU direct-access functions are available. These functions let you access glyph information directly.
gestaltATSUHighlightInactiveTextFeature	If the bit specified by this mask is set, the highlight inactive text feature is available.
gestaltATSUPositionToCursorFeature	If the bit specified by this mask is set, the position-to-cursor feature is available.
gestaltATSUTabSupportFeature	If the bit specified by this mask is set, support for tabs is available.
gestaltATSUUpdate6	Indicates that version 2.4 of ATSUI is installed on the user's system. Available beginning with ATSUI 2.4, in Mac OS X version 10.2.
gestaltAVLTreeSupportsHandleBasedTreeBit	
gestaltAVLTreeSupportsTreeLockingBit	
gestaltCanUseCGTextRendering	Available in Mac OS X v10.2 and later.
gestaltCPU750FX	
gestaltCPUG4745O	Available in Mac OS X v10.2 and later.
gestaltDisplayMgrSleepNotifies	Available in Mac OS X v10.2 and later.
gestaltFrontWindowMayBeHiddenBit	Available in Mac OS X v10.2 and later.
gestaltFrontWindowMayBeHiddenMask	Available in Mac OS X v10.2 and later.
gestaltFSSupportsExclusiveLocks	Available in Mac OS X v10.2 and later.
gestaltFSSupportsHardLinkDetection	Available in Mac OS X v10.2 and later.
gestaltMenuMgrMoreThanFiveMenusDeepBit	Available in Mac OS X v10.2 and later.
gestaltMenuMgrMoreThanFiveMenusDeepMask	Available in Mac OS X v10.2 and later.

MacErrors.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.

badCodecCharacterizationErr	
errFSQuotaExceeded	The user's quota of disk blocks has been exhausted.

kATSUBadStreamErr	The data is not formatted as specified by the data-flattening format constant, or the data is corrupt.
kATSUInvalidCallInsideCallbackErr	Your callback is making a call that could cause an infinite recursion.
kATSUOutputBufferTooSmallErr	The output buffer is too small to contain the data output by the function.
kATSUUnsupportedStreamFormatErr	The data-flattening format is invalid or is not supported by this version of ATSUI.
kQDCorruptPICTDataErr	Available in Mac OS X v10.2 and later.
kQDCursorAlreadyRegistered	Available in Mac OS X v10.2 and later.
kQDCursorNotRegistered	Available in Mac OS X v10.2 and later.
kQDNoColorHWCursorSupport	Available in Mac OS X v10.2 and later.
kQDNoPalette	Available in Mac OS X v10.2 and later.
noThumbnailFoundErr	
themeNoAppropriateBrushErr	Theme brush has no corresponding theme text color
tsmCantChangeForcedClassStateErr	Enabled state of a TextService class has been forced and cannot be changed
tsmDocPropertyBufferTooSmallErr	Buffer passed for property value is too small
tsmDocPropertyNotFoundErr	Requested TSM document property not found

Math64.h

Functions

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

S32Set	
S64Add	
S64And	
S64BitwiseAnd	
S64BitwiseEor	
S64BitwiseNot	
S64BitwiseOr	

S64Div	
S64Divide	
S64Eor	
S64Max	
S64Min	
S64Mod	
S64Multiply	
S64Negate	
S64Not	
S64Or	
S64Set	
S64SetU	
S64ShiftLeft	
S64ShiftRight	
S64Subtract	
SInt64ToUInt64	
U32SetU	
U64Add	
U64And	
U64BitwiseAnd	
U64BitwiseEor	
U64BitwiseNot	
U64BitwiseOr	
U64Div	
U64Divide	
U64Eor	
U64Max	
U64Mod	

U64Multiply	
U64Not	
U64Or	
U64Set	
U64SetU	
U64ShiftLeft	
U64ShiftRight	
U64Subtract	
UInt64ToSInt64	

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.

S64Divide	
U64Divide	

Resources.h

Functions

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

FSCreateResourceFork	Creates a named fork for storing resource data.

Script.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.

verIrelandEnglish	

TextCommon.h

Functions

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

`GetScriptInfoFromTextEncoding`	
`GetTextEncodingFromScriptInfo`	
`UCGetUnicodeScalarValueForSurrogatePair`	
`UCIsSurrogateHighCharacter`	
`UCIsSurrogateLowCharacter`	

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.

`kTextEncodingNextStepJapanese`	
`kUCHighSurrogateRangeEnd`	
`kUCHighSurrogateRangeStart`	
`kUCLowSurrogateRangeEnd`	
`kUCLowSurrogateRangeStart`	
`kUnicodeCanonicalCompVariant`	This is the normal canonical composition according to Unicode 3.2 rules.
`kUnicodeHFSPlusCompVariant`	Specifies canonical composition according to Unicode 3.2 rules, but using the HFS+ decomposition exclusions.
`kUnicodeHFSPlusDecompVariant`	Specifies canonical decomposition according to Unicode 3.2 rules, with HFS+ exclusions ("HFS+ decomposition 3.2"). That is, it doesn't decompose in 2000-2FFF, F900-FAFF, 2F800-2FAFF. You can use ths option when converting HFS file names.
`UC_INLINE`	

UnicodeConverter.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.

kUnicodeMapLineFeedToReturnBit	Available in Mac OS X v10.2 and later.
kUnicodeMapLineFeedToReturnMask	Specifies mapping o the LF (LineFeed) character used in Unix to represent new lines to the CR (CarriageReturn) used in Mac encodings.

OSServices

IconStorage.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.

IconFamilyElement	
IconFamilyHandle	
IconFamilyPtr	
IconFamilyResource	
kDropIconVariant	
kHuge1BitMask	
kHuge32BitData	
kHuge4BitData	
kHuge8BitData	
kHuge8BitMask	
kIconFamilyType	
kLarge1BitMask	
kLarge32BitData	
kLarge4BitData	
kLarge8BitData	
kLarge8BitMask	
kMini1BitMask	
kMini4BitData	
kMini8BitData	

kOpenDropIconVariant	
kOpenIconVariant	
kRolloverIconVariant	
kSmall1BitMask	
kSmall32BitData	
kSmall4BitData	
kSmall8BitData	
kSmall8BitMask	
kThumbnail32BitData	
kThumbnail8BitMask	
kTileIconVariant	
large1BitMask	
large4BitData	
large8BitData	
mini1BitMask	
mini4BitData	
mini8BitData	
small1BitMask	
small4BitData	
small8BitData	

SystemSound.h

Functions

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

AlertSoundPlay	
SystemSoundGetActionID	
SystemSoundPlay	
SystemSoundRemoveActionID	

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.

`kSystemSoundNoError`	
`kSystemSoundUnspecifiedError`	
`SystemSoundActionID`	

WebServicesCore

WSMethodInvocation.h

Functions

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

`WSGetCFTypeIDFromWSTypeID`	Gets the CFType associated with a given WSType
`WSGetWSTypeIDFromCFType`	Returns the WSTypeID associated with a given CFTypeRef.
`WSMethodInvocationAddDeserializationOverride`	Specifies a callback to be made when parsing the XML in a method response.
`WSMethodInvocationAddSerializationOverride`	Specifies a callback to be made when creating the XML for an method invocation.
`WSMethodInvocationCopyParameters`	Creates a copy of the parameters dictionary and sets the order in an array.
`WSMethodInvocationCopyProperty`	Creates a copy of a named property of the invocation reference.
`WSMethodInvocationCopySerialization`	Creates an XML serialization of a method invocation.
`WSMethodInvocationCreate`	Creates a reference to a method invocation, containing the URL of the service, the operation name, and the protocol.
`WSMethodInvocationCreateFromSerialization`	Creates a method invocation object from an XML serialization.
`WSMethodInvocationGetTypeID`	Returns the type ID of the current method invocation.

`WSMethodInvocationInvoke`	Invokes an web services operation synchronously.
`WSMethodInvocationScheduleWithRunLoop`	Schedule a method invocation for asynchronous execution on a run loop.
`WSMethodInvocationSetCallBack`	Set a callback to handle the response to an asynchronous method invocation.
`WSMethodInvocationSetParameters`	Set the parameter names, types, and order for a method invocation.
`WSMethodInvocationSetProperty`	Sets a named property of the method invocation.
`WSMethodInvocationUnscheduleFromRunLoop`	Unschedules a method invocation from a run loop.
`WSMethodResultIsFault`	Tests a method result dictionary for a fault condition.

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.

`errWSInternalError`	An internal framework error occured.
`errWSParseError`	The server response was not valid XML.
`errWSTimeoutError`	The method invocation timed out.
`errWSTransportError`	A network error occured.
`eWSArrayType`	Maps to CFArrayRef.
`eWSBooleanType`	Maps to CFBooleanRef.
`eWSDataType`	Maps to CFDataRef.
`eWSDateType`	Maps to CFDateRef.
`eWSDictionaryType`	Maps to CFDictionaryRef.
`eWSDoubleType`	Maps to CFNumberRef for long, double, or real numbers.
`eWSIntegerType`	Maps to CFNumberRef for 8, 16, 32 bit integers.
`eWSNullType`	Maps to CFNullRef.
`eWSStringType`	Maps to CFStringRef.

eWSUnknownType	No mapping is known for this type.
kWSDebugIncomingBody	If this flag is set, the result includes the incoming message body.
kWSDebugIncomingHeaders	If this flag is set, the result includes the incoming message headers.
kWSDebugOutgoingBody	If this flag is set, the result includes the outgoing message body.
kWSDebugOutgoingHeaders	If this flag is set, the result includes the outgoing message headers.
kWSFaultCode	If the result is a fault, this key returns a CFNumber with the fault code, unless the fault is a network error, in which case this field should be ignored.
kWSFaultExtra	If the result is a fault, and the fault is a network error, the key returns a CFDictionary with the network error. This key may also return a CFString, or NULL.
kWSFaultString	If the result is a fault, this key returns a CFString with the fault type. If the fault type is kWSNetworkStreamFaultString, then the fault is a network error. In the case of a network error, kWSFaultCode should be ignored, and kWSFaultExtra returns a dictionary indicating the network error.
kWSHTTPExtraHeaders	A CFDictionary of { key (CFString), val (CFString) } pairs.
kWSHTTPFollowsRedirects	A CFBoolean that controls whether the invocation follows redirects (default is false).
kWSHTTPMessage	The message.
kWSHTTPProxy	The CFURLRef of the SOCKS proxy.
kWSHTTPResponseMessage	The response.
kWSHTTPVersion	The CFHTTPMessageRef version such as “http/1.1”.

kWSMethodInvocationResult	Dictionary entry if the invocation result is not a fault. If you don't know what field to ask for, you can ask for this key. You can also specify the name of a reply parameter in the invocation using kWSMethodInvocationResultParameterName. This will add an alias for the given name to the result dictionary so that this key will return the named parameter.
kWSMethodInvocationResultParameterName	Set this property to create an alias to a parameter to be returned by kWSMethodInvocationResult. Pass in the parameter name as a CFStringRef.
kWSMethodInvocationTimeoutValue	
kWSNetworkStreamFaultString	If kWSFaultExtra is a dictionary, this key returns a CFString from that dictionary for debug purposes.
kWSRecordNamespaceURI	A CFStringRef containing the namespace.
kWSRecordParameterOrder	A CFArrayRef of CFStringRefs containing the parameter names, in order.
kWSRecordType	A CFStringRef containing the record type.
kWSSOAP1999Protocol	SOAP v1.1 protocol.
kWSSOAP2001Protocol	SOAP v1.2 protocol.
kWSSOAPBodyEncodingStyle	
kWSSOAPMessageHeaders	A CFArrayRef of XML header elements, as CFStringRefs.
kWSSOAPMethodNamespaceURI	
kWSSOAPStyleDoc	
kWSSOAPStyleRPC	
kWSStreamErrorDomain	If kWSFaultExtra is a dictionary, this key returns a CFNumberRef from that dictionary containing domain number. See CFStream.h for domain numbers.
kWSStreamErrorError	If kWSFaultExtra is a dictionary, this key returns a CFNumberRef from that dictionary containing error number. See CFStream.h for error numbers.

kWSStreamErrorMessage	If kWSFaultExtra is a dictionary, this key returns a CFString from that dictionary containing the stream error message.
kWSXMLRPCProtocol	XML-RPC protocol.
WSClientContext	An optional context that can contain data you want passed to your callback.
WSClientContextCopyDescriptionCallBackProcPtr	This is the callback that copies the information.
WSClientContextReleaseCallBackProcPtr	This is the callback that releases the information.
WSClientContextRetainCallBackProcPtr	This is the callback that retains the information.
WSMethodInvocationCallBackProcPtr	This is the callback that handles method invocation completion when the method is invoked asynchronously.
WSMethodInvocationDeserializationProcPtr	This is an optional callback that handles custom deserialization of a particular data type for a method response.
WSMethodInvocationRef	An opaque reference to a web services method invocation.
WSMethodInvocationSerializationProcPtr	This is an optional callback that handles custom serialization of a particular data type for method invocation.
WSTypeID	Web Services Core uses the following enumeration when serializing between Core Foundation and XML types. Because CFTypes are defined at runtime, it isn't always possible to produce a static mapping to a particular CFTypeRef. This enum and associated API allows for static determination of the expected serialization.

10.1 Symbol Changes

This article lists the symbols added to `CoreServices.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.

CFNetwork

CFHTTPMessage.h

Functions

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

`CFHTTPMessageAddAuthentication`	Adds authentication information to a request.
`CFHTTPMessageAppendBytes`	Appends data to a CFHTTPMessage object.
`CFHTTPMessageCopyAllHeaderFields`	Gets all header fields from a CFHTTPMessage object.
`CFHTTPMessageCopyBody`	Gets the body from a CFHTTPMessage object.
`CFHTTPMessageCopyHeaderFieldValue`	Gets the value of a header field from a CFHTTPMessage object.
`CFHTTPMessageCopyRequestMethod`	Gets the request method from a CFHTTPMessage object.
`CFHTTPMessageCopyRequestURL`	Gets the URL from a CFHTTPMessage object.
`CFHTTPMessageCopyResponseStatusLine`	Gets the status line from a CFHTTPMessage object.
`CFHTTPMessageCopySerializedMessage`	Serializes a CFHTTPMessage object.
`CFHTTPMessageCopyVersion`	Gets the HTTP version from a CFHTTPMessage object.
`CFHTTPMessageCreateCopy`	Gets a copy of a CFHTTPMessage object.
`CFHTTPMessageCreateEmpty`	Creates and returns a new, empty CFHTTPMessage object.

`CFHTTPMessageCreateRequest`	Creates and returns a CFHTTPMessage object for an HTTP request.
`CFHTTPMessageCreateResponse`	Creates and returns a CFHTTPMessage object for an HTTP response.
`CFHTTPMessageGetResponseStatusCode`	Gets the status code from a CFHTTPMessage object representing an HTTP response.
`CFHTTPMessageGetTypeID`	Returns the Core Foundation type identifier for the CFHTTPMessage opaque type.
`CFHTTPMessageIsHeaderComplete`	Determines whether a message header is complete.
`CFHTTPMessageIsRequest`	Returns a boolean indicating whether the CFHTTPMessage is a request or a response.
`CFHTTPMessageSetBody`	Sets the body of a CFHTTPMessage object.
`CFHTTPMessageSetHeaderFieldValue`	Sets the value of a header field in an HTTP message.

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.

`CFHTTPMessageRef`	An opaque reference representing an HTTP message.
`kCFHTTPAuthenticationSchemeBasic`	Request the HTTP basic authentication scheme.
`kCFHTTPAuthenticationSchemeDigest`	Request the HTTP digest authentication scheme.
`kCFHTTPVersion1_0`	Specifies HTTP version 1.0.
`kCFHTTPVersion1_1`	Specifies HTTP version 1.1.

CFHTTPStream.h

Functions

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

`CFHTTPReadStreamSetProxy`	This function sets the proxy host for a read stream.
`CFHTTPReadStreamSetRedirectsAutomatically`	This function enables or disables automatic redirection for a read stream.
`CFReadStreamCreateForHTTPRequest`	Creates a read stream for a CFHTTP request message.

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.

`CFStreamErrorHTTP`	Error codes that a read stream for an HTTP request may return.
`kCFStreamErrorDomainHTTP`	Error domain that returns errors associated with the CFHTTPStream layer.
`kCFStreamErrorHTTPBadURL`	The URL is not properly formatted.
`kCFStreamErrorHTTPParseFailure`	A parsing error occurred while an incoming message was being deserialized and appended to a message object. The headers of the incoming message may be formatted improperly.
`kCFStreamErrorHTTPRedirectionLoop`	A redirection loop has been detected.
`kCFStreamPropertyHTTPResponseHeader`	HTTP Response Header property. When copied by CFReadStreamCopyProperty, the header of an HTTP response message is returned.

CFSocketStream.h

Functions

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

`CFSocketStreamPairSetSecurityProtocol`	This function sets the security protocol for the specified pair of socket streams.

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.

`CFStreamSocketSecurityProtocol`	Specifies constants for setting the security protocol for a socket stream.
`kCFStreamErrorDomainSSL`	The error code is an SSL error code as defined in Security/SecureTransport.h.
`kCFStreamSocketSecurityNone`	Specifies that no security protocol be set for a socket stream.
`kCFStreamSocketSecuritySSLv2`	Specifies that SSL version 2 be set as the security protocol for a socket stream.

`kCFStreamSocketSecuritySSLv23`	Specifies that SSL version 3 be set as the security protocol for a socket stream pair. If that version is not available, specifies that SSL version 2 be set as the security protocol for a socket stream.
`kCFStreamSocketSecuritySSLv3`	Specifies that SSL version 3 be set as the security protocol for a socket stream.
`kCFStreamSocketSecurityTLSv1`	Specifies that TLS version 1 be set as the security protocol for a socket stream.

CarbonCore

CodeFragments.h

Functions

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

`ConvertBundlePreLocator`	Converts a bundle prelocator to a Core Foundation bundle locator.

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.

`CFMOffsetOf`	
`kCFBundlePreCFragLocator`	Indicates it was passed to the initialization routines in lieu of kCFBundleCFragLocator

Collections.h

Functions

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

`GetCollectionRetainCount`	Obtains the owner count (the number of existing references) for a collection object.
`ReleaseCollection`	Decrements the owner count (the number of existing references) for a collection object.
`RetainCollection`	Increments the owner count (the number of existing references) for a collection object.

DriverServices.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.

kMPTaskLevel	

Files.h

Functions

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

DisposeFNSubscriptionUPP	Deletes a universal procedure pointer (UPP) to your directory change callback function.
FNGetDirectoryForSubscription	Fetches the directory for which this subscription was originally entered.
FNSubscribe	Subscribes to change notifications for the specified directory.
FNSubscribeByPath	Subscribes to change notifications for the specified directory.
FNUnsubscribe	Releases a subscription which is no longer needed.
InvokeFNSubscriptionUPP	Calls your directory change callback function.
NewFNSubscriptionUPP	Creates a new universal procedure pointer (UPP) to your directory change callback function.

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.

bAllowCDiDataHandler	QuickTime's CDi data handler is allowed to examine the volume.
FNSubscriptionProcPtr	Callback delivered for directory notifications.
FNSubscriptionRef	
FNSubscriptionUPP	
kFNNoImplicitAllSubscription	Specify this option if you do not want to receive notifications on this subscription when FNNotifyAll is called.
kFSCatInfoUserAccess	Available in Mac OS X v10.1 and later.

Folders.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.

kAppleShareAuthenticationFolderType	
kAppleShareSupportFolderType	
kDirectoryServicesFolderType	
kFileSystemSupportFolderType	
kInstallerReceiptsFolderType	
kMIDIDriversFolderType	

Gestalt.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.

gestaltATSUFallbacksObjFeatures	If the bit specified by this mask is set, ATSUFontFallbacks objects are available.
gestaltATSUIgnoreLeadingFeature	If the bit specified by this mask is set, the line layout option (kATSIgnoreFontLeadingTag) to ignore the font leading value is available.
gestaltATSUUpdate4	Indicates that ATSUI for a version of Mac OS X from 10.0.1 through 10.0.4 is installed on the user's system.
gestaltATSUUpdate5	Indicates that version 2.3 of ATSUI is installed on the user's system. Available beginning with ATSUI 2.3, in Mac OS X version 10.1.
gestaltControlMgrVersion	Available in Mac OS X v10.1 and later.
gestaltDialogMgrHasAquaAlertBit	Available in Mac OS X v10.1 and later.
gestaltDialogMgrHasAquaAlertMask	Available in Mac OS X v10.1 and later.
gestaltDialogMgrPresentMask	Available in Mac OS X v10.1 and later.
gestaltDrawSprocketVersion	
gestaltFileMappingMultipleFilesFix	

gestaltPortable2001ANSIKbd	
gestaltPortable2001ISOKbd	
gestaltPortable2001JISKbd	
gestaltQTVRCubicPanosPresent	
gestaltSetDragImageUpdates	Available in Mac OS X v10.1 and later.
gestaltSupportsApplicationURL	Available in Mac OS X v10.1 and later.

HFSVolumes.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.

kHFSCatalogNodeIDsReusedBit	Available in Mac OS X v10.0 through Mac OS X v10.3.
kHFSCatalogNodeIDsReusedMask	Available in Mac OS X v10.0 through Mac OS X v10.3.

MacErrors.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.

kATSUBusyObjectErr	An ATSUI object is being used by another thread.
kATSUInvalidFontFallbacksErr	The ATSUFontFallback object is not initialized or is otherwise in an in valid state.
windowAppModalStateAlreadyExistsErr	The window is already application modal.
windowGroupInvalidErr	The window group is not valid.
windowNoAppModalStateErr	The window is not currently application modal.

Multiprocessing.h

Functions

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

MPModifyNotificationParameters	A result code.

MPSetTaskType	Sets the type of the task.

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.

kMPAllocateNoCreateMask	Do not attempt to create the pool if it does not yet exist.
kMPAsyncInterruptRemoteContext	Unsupported in Mac OS X.
kMPCreateTaskNotDebuggableMask	Unsupported in Mac OS X.
kMPHighLevelDebugger	The high-level debugger.
kMPInterruptRemoteContext	Unsupported in Mac OS X.
kMPLowLevelDebugger	The low-level debugger.
kMPMidLevelDebugger	The mid-level debugger.

MultiprocessingInfo.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.

kMPAddressSpaceInfoVersion	The MPAddressSpaceInfo structure version.
MPAddressSpaceInfo	

TextCommon.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.

kTextEncodingBig5_HKSCS_1999	Available in Mac OS X v10.1 and later.
kTextEncodingGB_18030_2000	Available in Mac OS X v10.1 and later.
kTextEncodingISO_2022_JP_3	JIS X0213
kTextEncodingShiftJIS_X0213_00	Shift-JIS format encoding of JIS X0213 planes 1 and 2
kTextEncodingUnicodeV3_1	Adds characters requiring surrogate pairs in UTF-16

kTextEncodingUnicodeV3_2	Available in Mac OS X v10.1 and later.

TextEncodingPlugin.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.

kTECMacOSXDispatchTableNameString	
TECPluginGetPluginDispatchTablePtr	Defines a pointer to a function that returns a pointer to a plug-in dispatch table.

fenv.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.

FE_ALL_EXCEPT	Available in Mac OS X v10.1 through Mac OS X v10.1.

OSServices

AppleDiskPartitions.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.

kATADriverSignature	Available in Mac OS X v10.1 and later.
kATAPIDriverSignature	Available in Mac OS X v10.1 and later.
kDriverTypeMacATA	Available in Mac OS X v10.1 and later.
kDriverTypeMacATAChained	Available in Mac OS X v10.1 and later.
kDriverTypeMacSCSI	Available in Mac OS X v10.1 and later.
kDriverTypeMacSCSIChained	Available in Mac OS X v10.1 and later.
kDriveSetupHFSSignature	Available in Mac OS X v10.1 and later.
kPartitionAUXIsAllocated	Available in Mac OS X v10.1 and later.

kPartitionAUXIsBootCodePositionIndependent	Available in Mac OS X v10.1 and later.
kPartitionAUXIsBootValid	Available in Mac OS X v10.1 and later.
kPartitionAUXIsInUse	Available in Mac OS X v10.1 and later.
kPartitionAUXIsReadable	Available in Mac OS X v10.1 and later.
kPartitionAUXIsValid	Available in Mac OS X v10.1 and later.
kPartitionAUXIsWriteable	Available in Mac OS X v10.1 and later.
kPartitionCanChainToNext	Available in Mac OS X v10.1 and later.
kPartitionIsChainCompatible	Available in Mac OS X v10.1 and later.
kPartitionIsMountedAtStartup	Available in Mac OS X v10.1 and later.
kPartitionIsRealDeviceDriver	Available in Mac OS X v10.1 and later.
kPartitionIsStartup	Available in Mac OS X v10.1 and later.
kPartitionIsWriteable	Available in Mac OS X v10.1 and later.
kPatchDriverSignature	Available in Mac OS X v10.1 and later.
kSCSICDDriverSignature	Available in Mac OS X v10.1 and later.
kSCSIDriverSignature	Available in Mac OS X v10.1 and later.

KeychainCore.h

Functions

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

KCDeleteItem	Deletes a password or other keychain item from the default keychain.
kcfindapplesharepassword	This function is available for convenience only and may be removed.
KCFindAppleSharePassword	Finds the first AppleShare password in the default keychain that matches the specified parameters.
KCFindGenericPassword	Finds the first generic password in the default keychain matching the specified parameters.
kcfindgenericpassword	This function is available for convenience only and may be removed.

KCFindInternetPassword	Finds the first Internet password in the default keychain that matches the specified parameters.
kcfindinternetpassword	This function is available for convenience only and may be removed.
kcfindinternetpasswordwithpath	This function is available for convenience only and may be removed.
KCFindInternetPasswordWithPath	Finds the first Internet password in the default keychain that matches the specified parameters, including path information.
KCGetData	Determines keychain item data.
KCIsInteractionAllowed	Indicates whether Keychain Manager functions that display a user interaction will do so.
KCLock	Locks a keychain.
KCSetInteractionAllowed	Enables or disables Keychain Manager functions that display a user interface.

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.

kAnyAuthType	Indicates that any Internet authentication type can be used.
kAnyPort	Indicates that any Internet port can be used.
kAnyProtocol	Indicates that any Internet protocol can be used.

Power.h

Functions

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

MinimumProcessorSpeed	

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.

hasAggressiveIdling	When this bit is set, it indicates Power Manager only resets OverallAct on UsrActivity.

`IdleActivity`	Delays idle sleep by timeout time.
`kEnterIdle`	Idle queue only.
`kExitIdle`	Idle queue only.
`kIdleQueueDeviceType`	
`kStillIdle`	Idle queue only.
`supportsIdleQueue`	When this bit is set, it indicates Power Manager supports the idle queue.

SCSI.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.

`DeviceIdentATA`	
`kBusTypeATA`	DeviceIdent holds information about an ATA device.
`kBusTypeMediaBay`	Not recommended.
`kBusTypePCMCIA`	Not recommended.
`kBusTypeSCSI`	DeviceIdent holds information about a SCSI device.

Document Revision History

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

Date	Notes
2007-07-18	Updated with the symbols added to the Core Services framework in Mac OS X v10.5.
2005-07-07	Made minor additions and corrections.
2005-04-29	Made minor editorial corrections.
	New document that summarizes the symbols added to the Core Services framework in Mac OS X v10.4.