

Defined Types

NSArgumentInfo

DECLARED IN Foundation/NSMethodSignature.h

SYNOPSIS typedef struct {
 int offset;
 int size;
 char *type;
} NSArgumentInfo

NSBTreeComparator

DECLARED IN Foundation/NSByteStore.h

SYNOPSIS typedef int **NSBTreeComparator**(NSData *, NSData *, const void *)

NSComparisonResult

DECLARED IN Foundation/NSObject.h

SYNOPSIS typedef enum _NSComparisonResult {
 NSOrderedAscending = -1,
 NSOrderedSame,
 NSOrderedDescending
} NSComparisonResult

DESCRIPTION An **NSComparisonResult** indicates how items in a request are ordered, from the first one given in a method invocation or function call to the last (that is, left-to-right in code).

NSHashTableCallbacks

DECLARED IN Foundation/NSHashTable.h

SYNOPSIS typedef struct {
 unsigned(***hash**)(NSHashTable *table, const void *);
 BOOL(***isEqual**)(NSHashTable *table, const void *, const void *);
 void(***retain**)(NSHashTable *table, const void *);
 void(***release**)(NSHashTable *table, void *);
 NSString *(***describe**)(NSHashTable *table, const void *);
} **NSHashTableCallbacks**

NSMapTableKeyCallbacks

DECLARED IN Foundation/NSMapTable.h

SYNOPSIS typedef struct {
 unsigned(***hash**)(NSMapTable *table, const void *);
 BOOL(***isEqual**)(NSMapTable *table, const void *, const void *);
 void(***retain**)(NSMapTable *table, const void *);
 void(***release**)(NSMapTable *table, void *);
 NSString *(***describe**)(NSMapTable *table, const void *);
 const void***notAKeyMarker**;
} **NSMapTableKeyCallbacks**

NSMapTableValueCallbacks

DECLARED IN Foundation/NSMapTable.h

SYNOPSIS typedef struct {
 void(***retain**)(NSMapTable *table, const void *);
 void(***release**)(NSMapTable *table, void *);
 NSString *(***describe**)(NSMapTable *table, const void *);
} **NSMapTableValueCallbacks**

NSNotificationCoalescing

DECLARED IN Foundation/NSNotificationQueue.h

SYNOPSIS typedef enum {
 NSNotificationNoCoalescing = 0,
 NSNotificationCoalescingOnName = 1,
 NSNotificationCoalescingOnSender = 2,
 NSNotificationCoalescing

NSObjCValue

DECLARED IN Foundation/NSInvocation.h

SYNOPSIS typedef struct {
 enum {
 NSObjCNoType = 0,
 NSObjCVoidType = 'v',
 NSObjCCharType = 'c',
 NSObjCShortType = 's',
 NSObjCLongType = 'l',
 NSObjCLonglongType = 'q',
 NSObjCFloatType = 'f',
 NSObjCDoubleType = 'd',
 NSObjCSelectorType = ':',
 NSObjCObjectType = '@',
 NSObjCStructType = '{',
 NSObjCPointerType = '^',
 NSObjCStringType = '*',
 NSObjCArrayType = '[', // not used in return values
 NSObjCUnionType = '(', // not used in return values
 NSObjCBitfield = 'b', // not used in return values
 }
 type;
 union {
 char charValue;
 short shortValue;
 long longValue;
 long long longlongValue;
 float floatValue;
 double doubleValue;
 SEL selectorValue;
 id objectValue;
 void *pointerValue;
 void *structLocation;
 char *cStringLocation;
 }
 value;
}
NSObjCValue

NSPoint

DECLARED IN Foundation/NSGeometry.h

SYNOPSIS typedef struct _NSPoint {
 float **x**;
 float **y**;
} **NSPoint**

NSPosixFileActivity

DECLARED IN Foundation/NSPosixFileDescriptor.h

SYNOPSIS typedef enum {
 NSPosixNoActivity = 0,
 NSPosixReadableActivity = 1,
 NSPosixWritableActivity = 2,
 NSPosixExceptionalActivity = 4
} **NSPosixFileActivity**

NSPostingStyle

DECLARED IN Foundation/NSNotificationQueue.h

SYNOPSIS typedef enum {
 NSPostWhenIdle = 1,
 NSPostASAP = 2,
 NSPostNow = 3
} **NSPostingStyle**

NSRange

DECLARED IN Foundation/NSRange.h

SYNOPSIS typedef struct _NSRange {
 unsigned int **location**;
 unsigned int **length**;
} **NSRange**

DESCRIPTION An **NSRange** describes a portion of a series—such as characters in a string or objects in an NSArray. Its **location** member gives the start index (0 is the first, as in C arrays), and its **length** member gives the number of items in the range (and can be zero).

NSRect

DECLARED IN Foundation/NSGeometry.h

SYNOPSIS typedef struct _NSRect {
 NSPoint **origin**;
 NSSize **size**;
} **NSRect**

NSRectEdge

DECLARED IN Foundation/NSGeometry.h

SYNOPSIS typedef enum _NSRectEdge {
 NSMinXEdge = 0,
 NSMinYEdge = 1,
 NSMaxXEdge = 2,
 NSMaxYEdge = 3
} **NSRectEdge**

DESCRIPTION This enumeration identifies the sides of a rectangle represented by an NSRect.

NSSize

DECLARED IN Foundation/NSGeometry.h

SYNOPSIS typedef struct _NSSize {
 float **width**; /* should never be negative */
 float **height**; /* should never be negative */
} **NSSize**

NSStringEncoding

DECLARED IN Foundation/NSString.h

SYNOPSIS typedef unsigned **NSStringEncoding**

NSTimeInterval

DECLARED IN Foundation/NSDate.h

SYNOPSIS typedef double **NSTimeInterval**

DESCRIPTION Always in seconds; yields sub-millisecond precision over a range of 10000 years.

NSUncaughtExceptionHandler

DECLARED IN Foundation/NSException.h

SYNOPSIS typedef volatile void **NSUncaughtExceptionHandler**(NSException ***exception**)

Symbolic Constants

String Encodings

DECLARED IN Foundation/NSString.h

SYNOPSIS NSASCIIStringEncoding
NSNEXTSTEPStringEncoding
NSUnicodeStringEncoding
NSISOLatin1StringEncoding
NSISOLatin2StringEncoding
NSSymbolStringEncoding
NSJapaneseEUCStringEncoding
NSShiftJISStringEncoding
NSUTF8StringEncoding
NSNonLossyASCIIStringEncoding

The following encodings are NeXT additions to the OpenStep specification

NSWindowsCP1250StringEncoding
NSWindowsCP1251StringEncoding
NSWindowsCP1252StringEncoding
NSWindowsCP1253StringEncoding
NSWindowsCP1254StringEncoding
NSISO2022JPStringEncoding

DESCRIPTION These values represent the various character encodings supported by the NSString classes. These encodings are documented more fully elsewhere; here are some short descriptions:

Encoding	Description
NSASCIIStringEncoding	Strict 7-bit ASCII encoding within 8-bit chars
NSNEXTSTEPStringEncoding	8-bit ASCII encoding with NEXTSTEP extensions (see Appendix C of the NEXTSTEP General Reference)
NSUnicodeStringEncoding	The canonical Unicode encoding for string objects
NSISOLatin1StringEncoding	8-bit ISO Latin 1 encoding
NSISOLatin2StringEncoding	8-bit ISO Latin 2 encoding
NSSymbolStringEncoding	8-bit Adobe Symbol encoding vector
NSJapaneseEUCStringEncoding	8-bit EUC encoding for Japanese text
NSShiftJISStringEncoding	8-bit Shift-JIS encoding for Japanese text
NSUTF8StringEncoding	An 8-bit representation of Unicode characters, suitable for transmission or storage by ASCII-based systems
NSNonLossyASCIIStringEncoding	7-bit verbose ASCII to represent all Unicode characters. <u><<should we document this format?>></u>
NSWindowsCP1250StringEncoding	Microsoft Windows codepage 1250; equivalent to WinLatin2 <u><<defined by whom?>></u>
NSWindowsCP1251StringEncoding	Microsoft Windows codepage 1251, encoding Cyrillic characters; equivalent to AdobeStandardCyrillic font encoding <u><<what is AdobeStandardCyrillic? Windows-only?>></u>
NSWindowsCP1252StringEncoding	Microsoft Windows codepage 1252; equivalent to WinLatin1 <u><<defined by whom?>></u>
NSWindowsCP1253StringEncoding	Microsoft Windows codepage 1253, encoding Greek characters
NSWindowsCP1254StringEncoding	Microsoft Windows codepage 1254, encoding Turkish characters
NSISO2022JPStringEncoding	ISO 2022 Japanese encoding for electronic mail

Search Types

DECLARED IN Foundation/NSString.h

SYNOPSIS **NSCaseInsensitiveSearch**
NSLiteralSearch
NSBackwardsSearch
NSAnchoredSearch

DESCRIPTION These values represent the options available to many of the string classes' searching and comparison methods. See the NSString class cluster description for details on the effects of these options.

NSNotFound

DECLARED IN Foundation/NSObject.h

SYNOPSIS **NSNotFound**

DESCRIPTION A method or function return value of **NSNotFound** indicates that the item requested couldn't be found or doesn't exist. It's typically used by various methods and functions that search for items in serial data and return indices, such as characters in a string object or **ids** in an NSArray.

NSOpenStepUnicodeReservedBase

DECLARED IN Foundation/NSCharacterSet.h

SYNOPSIS **NSOpenStepUnicodeReservedBase** = 0xF400<<forthcoming.>>

<<This is actually in an anonymous enum...seems to be so that it's not a #define, and will remain a symbol in gdb....>>

Enumerations

NSByteOrder

DECLARED IN Foundation/NSByteOrder.h

SYNOPSIS enum **NSByteOrder** {
 NS_UnknownByteOrder,
 NS_LittleEndian,
 NS_BigEndian
 }

Structures

Global Variables

Exception Names

DECLARED IN Foundation/NSException.h

SYNOPSIS extern NSString ***NSGenericException**
extern NSString ***NSRangeException**
extern NSString ***NSInvalidArgumentException**
extern NSString ***NSInternalInconsistencyException**
extern NSString ***NSMallocException**
extern NSString ***NSObjectInaccessibleException**
extern NSString ***NSObjectNotAvailableException**
extern NSString ***NSDestinationInvalidException**
extern NSString ***NSPortTimeoutException**
extern NSString ***NSInvalidSendPortException**
extern NSString ***NSInvalidReceivePortException**
extern NSString ***NSPortSendException**
extern NSString ***NSPortReceiveException**
extern NSString ***NSOldStyleException**
extern NSString ***NSInvalidSendPort**
extern NSString ***NSInvalidReceivePort**
extern NSString ***NSPortSendError**
extern NSString ***NSPortReceiveError**
extern NSString ***NSCharacterConversionException**

DECLARED IN Foundation/NSArchiver.h

SYNOPSIS extern NSString ***NSInconsistentArchiveException**

DECLARED IN Foundation/NSByteStore.h

SYNOPSIS extern NSString ***NSByteStoreLockedException**;
extern NSString ***NSByteStoreVersionException**
extern NSString ***NSBTreeStoreKeyTooLargeException**
extern NSString ***NSByteStoreDamagedException**

DECLARED IN Foundation/NSConnection.h

extern NSString ***NSFailedAuthenticationException**

DECLARED IN Foundation/NSPosix.h

extern NSString ***NSPosixFileOperationException**

Notification Names

DECLARED IN Foundation/NSBundle.h

SYNOPSIS extern NSString ***NSBundleLoaded**

DECLARED IN Foundation/NSConnection.h

SYNOPSIS extern NSString ***NSConnectionDidDieNotification**

DECLARED IN Foundation/NSPort.h

SYNOPSIS extern NSString ***NSPortDidBecomeInvalidNotification**

DECLARED IN Foundation/NPPL.h

SYNOPSIS extern NSString ***NSPPLDidBecomeDirtyNotification**
extern NSString ***NSPPLDidSaveNotification**

DECLARED IN Foundation/NSThread.h

SYNOPSIS extern NSString ***NSBecomingMultiThreaded**
extern NSString ***NSThreadExiting**

File Attribute Keys

DECLARED IN Foundation/NSFileManager.h

SYNOPSIS extern NSString ***NSFileType**
extern NSString ***NSFileTypeDirectory**
extern NSString ***NSFileTypeRegular**
extern NSString ***NSFileTypeSymbolicLink**
extern NSString ***NSFileTypeSocket**
extern NSString ***NSFileTypeCharacterSpecial**
extern NSString ***NSFileTypeBlockSpecial**
extern NSString ***NSFileTypeUnknown**
extern NSString ***NSFileSize**
extern NSString ***NSFileModificationDate**
extern NSString ***NSFileReferenceCount**
extern NSString ***NSFileDeviceIdentifier**
extern NSString ***NSFileOwnerAccountNumber**
extern NSString ***NSFileGroupOwnerAccountNumber**
extern NSString ***NSFilePosixPermissions**
extern NSString ***NSFileSystemNumber**
extern NSString ***NSFileSystemFileNumber**

DESCRIPTION Keys to access the file attribute values contained in the NSDictionary returned from NSFileManager's **fileAttributesAtPath:follow:**. See the class specification for NSFileManager for details of usage. **NSFileDeviceIdentifier** is used to access the identifier of a remote device.

File System Attribute Keys

DECLARED IN Foundation/NSFileManager.h

SYNOPSIS extern NSString ***NSFileSystemSize**
extern NSString ***NSFileSystemFreeSize**
extern NSString ***NSFileSystemNodes**
extern NSString ***NSFileSystemFreeNodes**

Language-Dependent Information

DECLARED IN Foundation/NSUserDefaults.h

SYNOPSIS extern NSString *NSWeekDayNameArray
extern NSString *NSShortWeekDayNameArray
extern NSString *NSMonthNameArray
extern NSString *NSShortMonthNameArray
extern NSString *NSTimeFormatString
extern NSString *NSDateFormatString
extern NSString *NSTimeDateFormatString
extern NSString *NSShortTimeDateFormatString
extern NSString *NSCurrencySymbol
extern NSString *NSDecimalSeparator
extern NSString *NSThousandsSeparator
extern NSString *NSInternationalCurrencyString
extern NSString *NSCurrencyString
extern NSString *NSDecimalDigits
extern NSString *NSAMPMDesignation
extern NSString *NSHourNameDesignations
extern NSString *NSYearMonthWeekDesignations
extern NSString *NSEarlierTimeDesignations
extern NSString *NSLaterTimeDesignations
extern NSString *NSThisDayDesignations
extern NSString *NSNextDayDesignations
extern NSString *NSNextNextDayDesignations
extern NSString *NSPriorDayDesignations
extern NSString *NSDateTimeOrdering

User Defaults

DECLARED IN Foundation/NSUserDefaults.h

SYNOPSIS extern NSString *NSGlobalDomain
extern NSString *NSArgumentDomain
extern NSString *NSRegistrationDomain

NSConnectionReplyMode

DECLARED IN NSConnection.h

SYNOPSIS extern NSString *NSConnectionReplyMode

NSDefaultRunLoopMode

DECLARED IN Foundation/NSRunLoop.h

SYNOPSIS extern NSString *NSDefaultRunLoopMode

Common Hash Table Call Backs

DECLARED IN Foundation/NSHashTable.h

SYNOPSIS extern const NSHashTableCallbacks **NSIntHashCallbacks**
extern const NSHashTableCallbacks **NSNonOwnedPointerHashCallbacks**
extern const NSHashTableCallbacks **NSOwnedPointerHashCallbacks**
extern const NSHashTableCallbacks **NSPointerToStructHashCallbacks**
extern const NSHashTableCallbacks **NSObjectHashCallbacks**
extern const NSHashTableCallbacks **NSNonRetainedObjectHashCallbacks**;

DESCRIPTION **NSIntHashCallbacks** are for sets of pointer-sized quantities or samller (for example, ints, longs, or unichars).

NSNonOwnedPointerHashCallbacks are for sets of pointers, hashed by address.

NSOwnedPointerHashCallbacks are for sets of pointers, with transfer of ownership upon insertion.

NSPointerToStructHashCallbacks are for sets of pointers to structs, when the first field of the struct is int-sized.

NSObjectHashCallbacks are for sets of objects (similar to NSSet)

NSNonRetainedObjectHashCallbacks are for sets of objects, but without retaining/releasing.

Note that you can make your own call back by picking fields among the above call backs.

Common Map Table Key Call Backs

DECLARED IN Foundation/NSMapTable.h

SYNOPSIS FOUNDATION_EXPORT

SYNOPSIS extern const NSMapTableKeyCallbacks **NSIntMapKeyCallbacks**
extern const NSMapTableKeyCallbacks **NSNonOwnedPointerMapKeyCallbacks**
extern const NSMapTableKeyCallbacks **NSNonOwnedPointerOrNullMapKeyCallbacks**
extern const NSMapTableKeyCallbacks **NSOwnedPointerMapKeyCallbacks**
extern const NSMapTableKeyCallbacks **NSObjectMapKeyCallbacks**
extern const NSMapTableKeyCallbacks **NSNonRetainedObjectMapKeyCallbacks**

DESCRIPTION **NSIntMapKeyCallbacks** are for keys that are pointer-sized quantities or smaller (for example, ints, longs, or unichars).

NSNonOwnedPointerMapKeyCallbacks are for keys that are pointers not freed.

NSNonOwnedPointerOrNullMapKeyCallbacks are for keys that are pointers not freed, or NULL.

NSOwnedPointerMapKeyCallbacks are for keys that are pointers, with transfer of ownership upon insertion.

NSObjectMapKeyCallbacks are for keys that are objects

NSNonRetainedObjectMapKeyCallbacks are for sets of objects, but without retaining/releasing.

Note that you can make your own call back by picking fields among the above call backs.

Common Map Table Value Call Backs

DECLARED IN	Foundation/NSMapTable.h
SYNOPSIS	<pre>extern const NSMapTableValueCallbacks NSIntMapValueCallbacks extern const NSMapTableValueCallbacks NSNonOwnedPointerMapValueCallbacks extern const NSMapTableValueCallbacks NSOwnedPointerMapValueCallbacks extern const NSMapTableValueCallbacks NSObjectMapValueCallbacks - NSHashTable.h - Common Value Callbacks</pre>
DESCRIPTION	<p>NSIntMapValueCallbacks are for values that are pointer-sized quantities, such as ints.</p> <p>NSNonOwnedPointerMapValueCallbacks are for values that are not owned pointers.</p> <p>NSOwnedPointerMapValueCallbacks are for values that are owned pointers.</p> <p>NSObjectMapValueCallbacks are for values that are objects.</p> <p>Note that you can make your own call back by picking fields among the above call backs.</p>

Zero Constants

DECLARED IN	Foundation/NSGeometry.h
SYNOPSIS	<pre>extern const NSPoint NSZeroPoint extern const NSSize NSZeroSize extern const NSRect NSZeroRect</pre>