
Foundation Data Types Reference

[Cocoa > Data Management](#)



2008-09-09



Apple Inc.
© 2008 Apple Inc.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

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

Apple, the Apple logo, Cocoa, Leopard, Mac, and Mac OS are trademarks of Apple Inc., registered in the United States and other countries.

Simultaneously published in the United States and Canada.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY

DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Contents

Foundation Data Types Reference 5

Overview	5
Data Types	5
NSAppleEventManagerSuspensionID	5
NSByteOrder	6
NSComparisonResult	6
NSDecimal	7
NSHashEnumerator	7
NSHashTable	8
NSHashTableCallbacks	8
NSHashTableOptions	9
NSInteger	9
NSMapEnumerator	9
NSMapTable	10
NSMapTableKeyCallbacks	10
NSMapTableOptions	11
NSMapTableValueCallbacks	11
NSObjCValue	12
NSPoint	12
NSPointArray	13
NSPointPointer	13
NSRange	13
NSRangePointer	14
NSRect	14
NSRectArray	15
NSRectEdge	15
NSRectPointer	16
NSSearchPathDirectory	16
NSSearchPathDomainMask	19
NSSize	19
NSSizeArray	20
NSSizePointer	20
NSStringEncoding	21
NSSwappedDouble	21
NSSwappedFloat	21
NSTimeInterval	21
NSUncaughtExceptionHandler	22
NSUInteger	22
NSZone	22

Document Revision History 25

Index 27

Foundation Data Types Reference

Framework:	Foundation/Foundation.h
Declared in	IKPictureTaker.h NSAppleEventManager.h NSByteOrder.h NSDate.h NSDecimal.h NSException.h NSGeometry.h NSHashTable.h NSInvocation.h NSMapTable.h NSObjCRuntime.h NSPathUtilities.h NSRange.h NSString.h NSZone.h UIKit/UIKitDefines.h

Overview

This document describes the data types and constants found in the Foundation framework.

Data Types

NSAppleEventManagerSuspensionID

Identifies an Apple event whose handling has been suspended. Can be used to resume handling of the Apple event.

```
typedef const struct __NSAppleEventManagerSuspension  
*NSAppleEventManagerSuspensionID;
```

Availability

Available in Mac OS X v10.3 and later.

Declared In

NSAppleEventManager.h

NSByteOrder

These constants specify an endian format.

```
enum _NSByteOrder {
    NS_UnknownByteOrder = CByteOrderUnknown,
    NS_LittleEndian = CByteOrderLittleEndian,
    NS_BigEndian = CByteOrderBigEndian
};
```

Constants

NS_UnknownByteOrder

The byte order is unknown.

Available in Mac OS X v10.0 and later.

Declared in NSByteOrder.h.

NS_LittleEndian

The byte order is little endian.

Available in Mac OS X v10.0 and later.

Declared in NSByteOrder.h.

NS_BigEndian

The byte order is big endian.

Available in Mac OS X v10.0 and later.

Declared in NSByteOrder.h.

Discussion

These constants are returned by NSHostByteOrder.

Declared In

NSByteOrder.h

NSComparisonResult

These constants are used to indicate how items in a request are ordered.

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

Constants

NSOrderedAscending

The left operand is smaller than the right operand.

Available in Mac OS X v10.0 and later.

Declared in NSObjCRuntime.h.

NSOrderedSame

The two operands are equal.

Available in Mac OS X v10.0 and later.

Declared in NSObjCRuntime.h.

`NSOrderedDescending`

The left operand is greater than the right operand.

Available in Mac OS X v10.0 and later.

Declared in `NSObjCRuntime.h`.

Discussion

These constants are used to indicate 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).

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSObjCRuntime.h`

NSDecimal

Used to describe a decimal number.

```
typedef struct {
    signed int _exponent:8;
    unsigned int _length:4;
    unsigned int _isNegative:1;
    unsigned int _isCompact:1;
    unsigned int _reserved:18;
    unsigned short _mantissa[NSDecimalMaxSize];
} NSDecimal;
```

Discussion

The fields of `NSDecimal` are private.

Used by the functions described in "Decimals".

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSDecimal.h`

NSHashEnumerator

Allows successive elements of a hash table to be returned each time this structure is passed to `NSNextHashEnumeratorItem`.

```
typedef struct {
    unsigned _pi;
    unsigned _si void *_bs;
} NSHashEnumerator;
```

Discussion

The fields of `NSHashEnumerator` are private.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSHashTable.h

NSHashTable

The opaque data type used by the functions described in "Hash Tables".

```
typedef struct _NSHashTable NSHashTable;
```

Discussion

For Mac OS X v10.5 and later, see also NSHashTable.

Availability

Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared In

NSHashTable.h

NSHashTableCallbacks

Defines a structure that contains the function pointers used to configure behavior of NSHashTable with respect to elements within a hash table.

```
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;
```

Fields

hash

Points to the function that must produce hash code for elements of the hash table. If `NULL`, the pointer value is used as the hash code. Second parameter is the element for which hash code should be produced.

isEqual

Points to the function that compares second and third parameters. If `NULL`, then `==` is used for comparison.

retain

Points to the function that increments a reference count for the given element. If `NULL`, then nothing is done for reference counting.

release

Points to the function that decrements a reference count for the given element, and if the reference count becomes 0, frees the given element. If `NULL`, then nothing is done for reference counting or releasing.

describe

Points to the function that produces an autoreleased `NSString *` describing the given element. If `NULL`, then the hash table produces a generic string description.

Discussion

All functions must know the types of things in the hash table to be able to operate on them. Sets of predefined call backs are described in "NSHashTable Callbacks".

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSHashTable.h

NSHashTableOptions

Specifies a bitfield used to configure the behavior of elements in an instance of NSHashTable.

```
typedef NSUInteger NSHashTableOptions
```

Declared In

NSHashTable.h

NSInteger

Used to describe an integer.

```
#if __LP64__
typedef long NSInteger;
#else
typedef int NSInteger;
#endif
```

Discussion

When building 32-bit applications, NSInteger is a 32-bit integer. A 64-bit application treats NSInteger as a 64-bit integer.

Availability

Available in Mac OS X v10.5 and later.

Declared In

IKPictureTaker.h

NSMapEnumerator

Allows successive elements of a map table to be returned each time this structure is passed to NSNextMapEnumeratorPair.

```
typedef struct {
    unsigned _pi;
    unsigned _si;
    void *_bs;
} NSMapEnumerator;
```

Discussion

The fields of NSMapEnumerator are private.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSMapTable.h

NSMapTable

The opaque data type used by the functions described in "Map Tables".

```
typedef struct _NSMapTable NSMapTable;
```

Discussion

For Mac OS X v10.5 and later, see also NSMapTable.

Availability

Available in Mac OS X v10.0 through Mac OS X v10.4.

Declared In

NSMapTable.h

NSMapTableKeyCallbacks

The function pointers used to configure behavior of NSMapTable with respect to key elements within a map table.

```
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;
```

Fields

hash

Points to the function which must produce hash code for key elements of the map table. If NULL, the pointer value is used as the hash code. Second parameter is the element for which hash code should be produced.

isEqual

Points to the function which compares second and third parameters. If NULL, then == is used for comparison.

retain

Points to the function which increments a reference count for the given element. If NULL, then nothing is done for reference counting.

release

Points to the function which decrements a reference count for the given element, and if the reference count becomes zero, frees the given element. If NULL, then nothing is done for reference counting or releasing.

describe

Points to the function which produces an autoreleased NSString * describing the given element. If NULL, then the map table produces a generic string description.

notAKeyMarker

No key put in map table can be this value. An exception is raised if attempt is made to use this value as a key

Discussion

All functions must know the types of things in the map table to be able to operate on them. Sets of predefined call backs are described in "NSMutableDictionary Key Call Backs".

Two predefined values to use for notAKeyMarker are NSNotAnIntMapKey and NSNotAPointerMapKey.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSMutableDictionary.h

NSMutableDictionaryOptions

Specifies a bitfield used to configure the behavior of elements in an instance of NSMutableDictionary.

```
typedef NSUInteger NSMutableDictionaryOptions
```

Declared In

NSMutableDictionary.h

NSMutableDictionaryValueCallbacks

The function pointers used to configure behavior of NSMutableDictionary with respect to value elements within a map table.

```
typedef struct {
    void (*retain)(NSMutableDictionary *table, const void *);
    void (*release)(NSMutableDictionary *table, void *);
    NSString *(*describe)(NSMutableDictionary *table, const void *);
} NSMutableDictionaryValueCallbacks;
```

Fields

retain

Points to the function that increments a reference count for the given element. If NULL, then nothing is done for reference counting.

release

Points to the function that decrements a reference count for the given element, and if the reference count becomes zero, frees the given element. If NULL, then nothing is done for reference counting or releasing.

describe

Points to the function that produces an autoreleased NSString * describing the given element. If NULL, then the map table produces a generic string description.

Discussion

All functions must know the types of things in the map table to be able to operate on them. Sets of predefined call backs are described in "NSMutableDictionary Value Callbacks".

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSMutableDictionary.h

NSObjCValue

This structure is defined for use by `NSInvocation`—you should not use it directly.

```
typedef struct {
    enum _NSObjCValueType type;
    union {
        char charValue;
        short shortValue;
        long longValue;
        long long longlongValue;
        float floatValue;
        double doubleValue;
        bool boolValue;
        SEL selectorValue;
        id objectValue;
        void *pointerValue;
        void *structLocation;
        char *cStringLocation;
    } value;
} NSObjCValue;
```

Discussion

The fields of `NSObjCValue` are private.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSInvocation.h

NSPoint

Represents a point in a Cartesian coordinate system.

```
typedef struct _NSPoint {
    CGFloat x;
    CGFloat y;
} NSPoint;
```

Fields

x

The x coordinate.

y

The y coordinate.

Special Considerations

Prior to Mac OS X v10.5 the coordinates were represented by `float` values rather than `CGFloat` values.

When building for 64 bit systems, or building 32 bit like 64 bit, `NSPoint` is typedef'd to `CGPoint`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSGeometry.h`

NSPointArray

Type indicating a parameter is array of `NSPoint` structures.

```
typedef NSPoint *NSPointArray;
```

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSGeometry.h`

NSPointPointer

Type indicating a parameter is a pointer to an `NSPoint` structure.

```
typedef NSPoint *NSPointPointer;
```

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSGeometry.h`

NSRange

A structure used to describe a portion of a series—such as characters in a string or objects in an `NSArray` object.

```
typedef struct _NSRange {
    NSUInteger location;
    NSUInteger length;
} NSRange;
```

Fields

`location`

The start index (0 is the first, as in C arrays).

length

The number of items in the range (can be 0).

Discussion

Foundation functions that operate on ranges include the following:

- NSEqualRanges
- NSIntersectionRange
- NSLocationInRange
- NSMakeRange
- NSMaxRange
- NSRangeFromString
- NSStringFromRange
- NSUnionRange

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSRange.h

NSRangePointer

Type indicating a parameter is a pointer to an NSRange structure.

```
typedef NSRange *NSRangePointer;
```

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSRange.h

NSRect

Represents a rectangle.

```
typedef struct _NSRect {  
    NSPoint origin;  
    NSSize size;  
} NSRect;
```

Fields

origin

The origin of the rectangle (its starting x coordinate and y coordinate).

size

The width and height of the rectangle, as measured from the origin.

Special Considerations

When building for 64 bit systems, or building 32 bit like 64 bit, NSRect is typedef'd to CGRect.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSGeometry.h

NSRectArray

Type indicating a parameter is array of NSRect structures.

```
typedef NSRect *NSRectArray;
```

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSGeometry.h

NSRectEdge

Identifiers used by NSDivideRect to specify the edge of the input rectangle from which the division is measured.

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

Constants

NSMinXEdge

Specifies the left edge of the input rectangle.

The input rectangle is divided vertically, and the leftmost rectangle with the width of `amount` is placed in `slice`.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in NSGeometry.h.

NSMinYEdge

Specifies the bottom edge of the input rectangle.

The input rectangle is divided horizontally, and the bottom rectangle with the height of `amount` is placed in `slice`.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in NSGeometry.h.

`NSMaxXEdge`

Specifies the right edge of the input rectangle.

The input rectangle is divided vertically, and the rightmost rectangle with the width of `amount` is placed in `slice`.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `NSGeometry.h`.

`NSMaxYEdge`

Specifies the top edge of the input rectangle.

The input rectangle is divided horizontally, and the top rectangle with the height of `amount` is placed in `slice`.

Available in Mac OS X v10.0 and later.

Not available to 64-bit applications.

Declared in `NSGeometry.h`.

Discussion

The parameters `amount` and `slice` are defined by `NSDivideRect`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSGeometry.h`

NSRectPointer

Type indicating a parameter is a pointer to an `NSRect` structure.

```
typedef NSRect *NSRectPointer;
```

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSGeometry.h`

NSSearchPathDirectory

These constants specify the location of a variety of directories.


```
typedef enum {
    NSApplicationDirectory = 1,
    NSDemoApplicationDirectory,
    NSDeveloperApplicationDirectory,
    NSAdminApplicationDirectory,
    NSLibraryDirectory,
    NSDeveloperDirectory,
    NSUserDirectory,
    NSDocumentationDirectory,
    NSDocumentDirectory,
    NSCoreServiceDirectory,
    NSDesktopDirectory = 12,
    NSCachesDirectory = 13,
    NSApplicationSupportDirectory = 14,
    NSDownloadsDirectory = 15,
    NSAllApplicationsDirectory = 100,
    NSAllLibrariesDirectory = 101
} NSSearchPathDirectory;
```

Constants

NSApplicationDirectory

Supported applications (/Applications).

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

NSDemoApplicationDirectory

Unsupported applications and demonstration versions.

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

NSDeveloperApplicationDirectory

Developer applications (/Developer/Applications).

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

NSAdminApplicationDirectory

System and network administration applications.

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

NSLibraryDirectory

Various user-visible documentation, support, and configuration files (/Library).

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

NSDeveloperDirectory

Developer resources (/Developer).

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

NSUserDirectory

User home directories (/Users).

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

`NSDocumentationDirectory`

Documentation.

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

`NSDocumentDirectory`

Document directory.

Available in Mac OS X v10.2 and later.

Declared in `NSPathUtilities.h`.

`NSCoreServiceDirectory`

Location of core services (`System/Library/CoreServices`).

Available in Mac OS X v10.4 and later.

Declared in `NSPathUtilities.h`.

`NSDesktopDirectory`

Location of user's desktop directory.

Available in Mac OS X v10.4 and later.

Declared in `NSPathUtilities.h`.

`NSCachesDirectory`

Location of discardable cache files (`Library/Caches`).

Available in Mac OS X v10.4 and later.

Declared in `NSPathUtilities.h`.

`NSApplicationSupportDirectory`

Location of application support files (`Library/Application Support`).

Available in Mac OS X v10.4 and later.

Declared in `NSPathUtilities.h`.

`NSDownloadsDirectory`

Location of the user's downloads directory.

The `NSDownloadsDirectory` flag will only produce a path only when the `NSUserDomainMask` is provided.

Available in Mac OS X v10.5 and later.

Declared in `NSPathUtilities.h`.

`NSAllApplicationsDirectory`

All directories where applications can occur.

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

`NSAllLibrariesDirectory`

All directories where resources can occur.

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSPathUtilities.h`

NSSearchPathDomainMask

Search path domain constants specifying base locations for the [NSSearchPathDirectory](#) (page 16) type.

```
typedef enum {
    NSUserDomainMask = 1,
    NSLocalDomainMask = 2,
    NSNetworkDomainMask = 4,
    NSSystemDomainMask = 8,
    NSAllDomainsMask = 0xffff,
} NSSearchPathDomainMask;
```

Constants

NSUserDomainMask

The user's home directory—the place to install user's personal items (~).

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

NSLocalDomainMask

Local to the current machine—the place to install items available to everyone on this machine.

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

NSNetworkDomainMask

Publicly available location in the local area network—the place to install items available on the network (/Network).

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

NSSystemDomainMask

Provided by Apple — can't be modified (/System).

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

NSAllDomainsMask

All domains.

Includes all of the above and future items.

Available in Mac OS X v10.0 and later.

Declared in `NSPathUtilities.h`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSPathUtilities.h`

NSSize

Represents a two-dimensional size.

```
typedef struct _NSSize {
    CGFloat width;
    CGFloat height;
} NSSize;
```

Fields

width

The width.

height

The height.

Discussion

Normally, the values of `width` and `height` are non-negative. The functions that create an `NSSize` structure do not prevent you from setting a negative value for these attributes. If the value of `width` or `height` is negative, however, the behavior of some methods may be undefined.

Special Considerations

Prior to Mac OS X v10.5 the width and height were represented by `float` values rather than `CGFloat` values.

When building for 64 bit systems, or building 32 bit like 64 bit, `NSSize` is typedef'd to `CGSize`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSGeometry.h`

NSSizeArray

Type indicating a parameter is array of `NSSize` structures.

```
typedef NSSize *NSSizeArray;
```

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSGeometry.h`

NSSizePointer

Type indicating parameter is a pointer to an `NSSize` structure.

```
typedef NSSize *NSSizePointer;
```

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSGeometry.h`

NSStringEncoding

Type representing string-encoding values.

```
typedef NSUInteger NSStringEncoding;
```

Discussion

See String Encodings for a list of values.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSString.h

NSSwappedDouble

Opaque structure containing endian-independent double value.

```
typedef struct {  
    unsigned long long v;  
} NSSwappedDouble;
```

Discussion

The fields of an NSSwappedDouble are private.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSByteOrder.h

NSSwappedFloat

Opaque type containing an endian-independent float value.

```
typedef struct {  
    unsigned long v;  
} NSSwappedFloat;
```

Discussion

The fields of an NSSwappedFloat are private.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSByteOrder.h

NSTimeInterval

Used to specify a time interval, in seconds.

```
typedef double NSTimeInterval;
```

Discussion

`NSTimeInterval` is always specified in seconds; it yields sub-millisecond precision over a range of 10,000 years.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSDate.h`

NSUncaughtExceptionHandler

Used for the function handling exceptions outside of an exception-handling domain.

```
typedef volatile void NSUncaughtExceptionHandler(NSException *exception);
```

Discussion

You can set exception handlers using `NSSetUncaughtExceptionHandler`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSException.h`

NSUInteger

Used to describe an unsigned integer.

```
#if __LP64__
typedef long NSUInteger;
#else
typedef int NSUInteger;
#endif
```

Discussion

When building 32-bit applications, `NSUInteger` is a 32-bit unsigned integer. A 64-bit application treats `NSUInteger` as a 64-bit unsigned integer.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`UIKitDefines.h`

NSZone

Used to identify and manage memory zones.

```
typedef struct _NSZone NSZone;
```

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSZone.h

Document Revision History

This table describes the changes to *Foundation Data Types Reference*.

Date	Notes
2008-03-11	Updated float to CGFloat where appropriate.
2007-12-11	Enhanced the definitions for constants in NSRectEdge.
2007-10-31	Added NSDownloadsDirectory constant for obtaining the user's Downloads location; corrected the definitions for constants in NSRectEdge.
2007-07-19	Corrected minor typographical errors.
Leopard WWDC	Updated for Mac OS X v10.5.
2006-06-28	Removed references to retired document.
2006-05-23	First publication of this content as a separate document.

REVISION HISTORY

Document Revision History

Index

N

- NSAdminApplicationDirectory **constant** 17
- NSAllApplicationsDirectory **constant** 18
- NSAllDomainsMask **constant** 19
- NSAllLibrariesDirectory **constant** 18
- NSAppleEventManagerSuspensionID **data type** 5
- NSApplicationDirectory **constant** 17
- NSApplicationSupportDirectory **constant** 18
- NSByteOrder **data type** 6
- NSCachesDirectory **constant** 18
- NSComparisonResult **data type** 6
- NSCoreServiceDirectory **constant** 18
- NSDecimal **data type** 7
- NSDemoApplicationDirectory **constant** 17
- NSDesktopDirectory **constant** 18
- NSDeveloperApplicationDirectory **constant** 17
- NSDeveloperDirectory **constant** 17
- NSDocumentationDirectory **constant** 18
- NSDocumentDirectory **constant** 18
- NSDownloadsDirectory **constant** 18
- NSHashEnumerator **data type** 7
- NSHashTable **data type** 8
- NSHashTableCallbacks **data type** 8
- NSHashTableOptions **data type** 9
- NSInteger **data type** 9
- NSLibraryDirectory **constant** 17
- NSLocalDomainMask **constant** 19
- NSMapEnumerator **data type** 9
- NSMapTable **data type** 10
- NSMapTableKeyCallbacks **data type** 10
- NSMapTableOptions **data type** 11
- NSMapTableValueCallbacks **data type** 11
- NSMaxXEdge **constant** 16
- NSMaxYEdge **constant** 16
- NSMinXEdge **constant** 15
- NSMinYEdge **constant** 15
- NSNetworkDomainMask **constant** 19
- NSObjCValue **data type** 12
- NSOrderedAscending **constant** 6
- NSOrderedDescending **constant** 7
- NSOrderedSame **constant** 6
- NSPoint **data type** 12
- NSPointArray **data type** 13
- NSPointPointer **data type** 13
- NSRange **data type** 13
- NSRangePointer **data type** 14
- NSRect **data type** 14
- NSRectArray **data type** 15
- NSRectEdge **data type** 15
- NSRectPointer **data type** 16
- NSSearchPathDirectory **data type** 16
- NSSearchPathDomainMask **data type** 19
- NSSize **data type** 19
- NSSizeArray **data type** 20
- NSSizePointer **data type** 20
- NSStringEncoding **data type** 21
- NSSwappedDouble **data type** 21
- NSSwappedFloat **data type** 21
- NSSystemDomainMask **constant** 19
- NSTimeInterval **data type** 21
- NSUInteger **data type** 22
- NSUncaughtExceptionHandler **data type** 22
- NSUserDirectory **constant** 17
- NSUserDomainMask **constant** 19
- NSZone **data type** 22
- NS_BigEndian **constant** 6
- NS_LittleEndian **constant** 6
- NS_UnknownByteOrder **constant** 6