

---

# NSNumber Class Reference

[Cocoa > Data Management](#)



2008-02-08



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, Mac, Mac OS, and Objective-C are trademarks of Apple Inc., registered in the United States and other countries.

iPhone and Numbers are trademarks of Apple Inc.

Times is a registered trademark of Heidelberger Druckmaschinen AG, available from Linotype Library GmbH.

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

## **NSNumber Class Reference 5**

---

Overview	5
Creating a Subclass of NSNumber	5
Tasks	6
Creating an NSNumber Object	6
Initializing an NSNumber Object	6
Accessing Numeric Values	7
Retrieving String Representations	8
Comparing NSNumber Objects	8
Accessing Type Information	8
Class Methods	9
numberWithBool:	9
numberWithChar:	9
numberWithDouble:	9
numberWithFloat:	10
numberWithInt:	10
numberWithInteger:	11
numberWithLong:	11
numberWithLongLong:	12
numberWithShort:	12
numberWithUnsignedChar:	13
numberWithUnsignedInt:	13
numberWithUnsignedInteger:	14
numberWithUnsignedLong:	14
numberWithUnsignedLongLong:	15
numberWithUnsignedShort:	15
Instance Methods	16
boolValue	16
charValue	16
compare:	16
decimalValue	17
descriptionWithLocale:	17
doubleValue	18
floatValue	19
initWithBool:	19
initWithChar:	20
initWithDouble:	20
initWithFloat:	20
initWithInt:	21
initWithInteger:	21
initWithLong:	21

initWithLongLong: 22  
initWithShort: 22  
initWithUnsignedChar: 22  
initWithUnsignedInt: 23  
initWithUnsignedInteger: 23  
initWithUnsignedLong: 24  
initWithUnsignedLongLong: 24  
initWithUnsignedShort: 24  
integerValue 25  
intValue 25  
isEqualToNumber: 25  
longLongValue 26  
longValue 26  
objCType 27  
shortValue 27  
stringValue 27  
unsignedCharValue 28  
unsignedIntegerValue 28  
unsignedIntValue 28  
unsignedLongLongValue 29  
unsignedLongValue 29  
unsignedShortValue 29

---

**Document Revision History 31**

---

**Index 33**

---

# NSNumber Class Reference

---

<b>Inherits from</b>	NSNumber : NSObject
<b>Conforms to</b>	NSCoding (NSNumber) NSCopying (NSNumber) NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/Foundation.framework
<b>Availability</b>	Available in Mac OS X v10.0 and later.
<b>Declared in</b>	NSDecimalNumber.h NSNumber.h
<b>Companion guides</b>	Number and Value Programming Topics for Cocoa Property List Programming Guide
<b>Related sample code</b>	Dicey QTCoreVideo301 Quartz Composer WWDC 2005 TextEdit SimpleScriptingObjects TextEditPlus

## Overview

NSNumber is a subclass of NSNumber that offers a value as any C scalar (numeric) type. It defines a set of methods specifically for setting and accessing the value as a signed or unsigned char, short int, int, long int, long long int, float, or double or as a BOOL. (Note that number objects do not necessarily preserve the type they are created with.) It also defines a [compare:](#) (page 16) method to determine the ordering of two NSNumber objects.

## Creating a Subclass of NSNumber

---

As with any class cluster, if you create a subclass of NSNumber, you have to override the primitive methods of its superclass, NSNumber. Furthermore, there is a restricted set of return values that your implementation of the NSNumber method objCType can return, in order to take advantage of the abstract implementations of the non-primitive methods. The valid return values are "c", "C", "s", "S", "i", "I", "l", "L", "q", "Q", "f", and "d".

## Tasks

### Creating an NSNumber Object

- + [numberWithBool:](#) (page 9)  
Creates and returns an `NSNumber` object containing a given value, treating it as a `BOOL`.
- + [numberWithChar:](#) (page 9)  
Creates and returns an `NSNumber` object containing a given value, treating it as a signed `char`.
- + [numberWithDouble:](#) (page 9)  
Creates and returns an `NSNumber` object containing a given value, treating it as a `double`.
- + [numberWithFloat:](#) (page 10)  
Creates and returns an `NSNumber` object containing a given value, treating it as a `float`.
- + [numberWithInt:](#) (page 10)  
Creates and returns an `NSNumber` object containing a given value, treating it as a signed `int`.
- + [numberWithInteger:](#) (page 11)  
Creates and returns an `NSNumber` object containing a given value, treating it as an `NSInteger`.
- + [numberWithLong:](#) (page 11)  
Creates and returns an `NSNumber` object containing a given value, treating it as a signed `long`.
- + [numberWithLongLong:](#) (page 12)  
Creates and returns an `NSNumber` object containing a given value, treating it as a signed `long long`.
- + [numberWithShort:](#) (page 12)  
Creates and returns an `NSNumber` object containing *value*, treating it as a signed `short`.
- + [numberWithUnsignedChar:](#) (page 13)  
Creates and returns an `NSNumber` object containing a given value, treating it as an unsigned `char`.
- + [numberWithUnsignedInt:](#) (page 13)  
Creates and returns an `NSNumber` object containing a given value, treating it as an unsigned `int`.
- + [numberWithUnsignedInteger:](#) (page 14)  
Creates and returns an `NSNumber` object containing a given value, treating it as an `NSUInteger`.
- + [numberWithUnsignedLong:](#) (page 14)  
Creates and returns an `NSNumber` object containing a given value, treating it as an unsigned `long`.
- + [numberWithUnsignedLongLong:](#) (page 15)  
Creates and returns an `NSNumber` object containing a given value, treating it as an unsigned `long long`.
- + [numberWithUnsignedShort:](#) (page 15)  
Creates and returns an `NSNumber` object containing a given value, treating it as an unsigned `short`.

### Initializing an NSNumber Object

- [initWithBool:](#) (page 19)  
Returns an `NSNumber` object initialized to contain a given value, treated as a `BOOL`.
- [initWithChar:](#) (page 20)  
Returns an `NSNumber` object initialized to contain a given value, treated as a signed `char`.

- [initWithDouble:](#) (page 20)  
Returns an NSNumber object initialized to contain *value*, treated as a double.
- [initWithFloat:](#) (page 20)  
Returns an NSNumber object initialized to contain a given value, treated as a float.
- [initWithInt:](#) (page 21)  
Returns an NSNumber object initialized to contain a given value, treated as a signed int.
- [initWithInteger:](#) (page 21)  
Returns an NSNumber object initialized to contain a given value, treated as an NSInteger.
- [initWithLong:](#) (page 21)  
Returns an NSNumber object initialized to contain a given value, treated as a signed long.
- [initWithLongLong:](#) (page 22)  
Returns an NSNumber object initialized to contain *value*, treated as a signed long long.
- [initWithShort:](#) (page 22)  
Returns an NSNumber object initialized to contain a given value, treated as a signed short.
- [initWithUnsignedChar:](#) (page 22)  
Returns an NSNumber object initialized to contain a given value, treated as an unsigned char.
- [initWithUnsignedInt:](#) (page 23)  
Returns an NSNumber object initialized to contain a given value, treated as an unsigned int.
- [initWithUnsignedInteger:](#) (page 23)  
Returns an NSNumber object initialized to contain a given value, treated as an NSUInteger.
- [initWithUnsignedLong:](#) (page 24)  
Returns an NSNumber object initialized to contain a given value, treated as an unsigned long.
- [initWithUnsignedLongLong:](#) (page 24)  
Returns an NSNumber object initialized to contain a given value, treated as an unsigned long long.
- [initWithUnsignedShort:](#) (page 24)  
Returns an NSNumber object initialized to contain a given value, treated as an unsigned short.

## Accessing Numeric Values

- [boolValue](#) (page 16)  
Returns the receiver's value as a BOOL.
- [charValue](#) (page 16)  
Returns the receiver's value as a char.
- [decimalValue](#) (page 17)  
Returns the receiver's value, expressed as an NSDecimal structure.
- [doubleValue](#) (page 18)  
Returns the receiver's value as a double.
- [floatValue](#) (page 19)  
Returns the receiver's value as a float.
- [intValue](#) (page 25)  
Returns the receiver's value as an int.
- [integerValue](#) (page 25)  
Returns the receiver's value as an NSInteger.

- [longLongValue](#) (page 26)  
Returns the receiver's value as a `long long`.
- [longValue](#) (page 26)  
Returns the receiver's value as a `long`.
- [shortValue](#) (page 27)  
Returns the receiver's value as a `short`.
- [unsignedCharValue](#) (page 28)  
Returns the receiver's value as an unsigned `char`.
- [unsignedIntegerValue](#) (page 28)  
Returns the receiver's value as an `NSUInteger`.
- [unsignedIntValue](#) (page 28)  
Returns the receiver's value as an unsigned `int`.
- [unsignedLongLongValue](#) (page 29)  
Returns the receiver's value as an unsigned `long long`.
- [unsignedLongValue](#) (page 29)  
Returns the receiver's value as an unsigned `long`.
- [unsignedShortValue](#) (page 29)  
Returns the receiver's value as an unsigned `short`.

## Retrieving String Representations

- [descriptionWithLocale:](#) (page 17)  
Returns a string that represents the contents of the receiver for a given locale.
- [stringValue](#) (page 27)  
Returns the receiver's value as a human-readable string.

## Comparing NSNumber Objects

- [compare:](#) (page 16)  
Returns an `NSComparisonResult` value that indicates whether the receiver is greater than, equal to, or less than a given number.
- [isEqualToNumber:](#) (page 25)  
Returns a Boolean value that indicates whether the receiver and a given number are equal.

## Accessing Type Information

- [objCType](#) (page 27)  
Returns a C string containing the Objective-C type of the data contained in the receiver.



## Class Methods

### **numberWithBool:**

Creates and returns an `NSNumber` object containing a given value, treating it as a `BOOL`.

```
+ (NSNumber *)numberWithBool:(BOOL) value
```

#### **Parameters**

*value*

The value for the new number.

#### **Return Value**

An `NSNumber` object containing *value*, treating it as a `BOOL`.

#### **Availability**

Available in Mac OS X v10.0 and later.

#### **Related Sample Code**

EnhancedAudioBurn

GridCalendar

Quartz Composer WWDC 2005 TextEdit

SMARTQuery

TextEditPlus

#### **Declared In**

`NSNumber.h`

### **numberWithChar:**

Creates and returns an `NSNumber` object containing a given value, treating it as a signed `char`.

```
+ (NSNumber *)numberWithChar:(char) value
```

#### **Parameters**

*value*

The value for the new number.

#### **Return Value**

An `NSNumber` object containing *value*, treating it as a signed `char`.

#### **Availability**

Available in Mac OS X v10.0 and later.

#### **Declared In**

`NSNumber.h`

### **numberWithDouble:**

Creates and returns an `NSNumber` object containing a given value, treating it as a `double`.

```
+ (NSNumber *)numberWithDouble:(double) value
```

**Parameters**

*value*

The value for the new number.

**Return Value**

An `NSNumber` object containing *value*, treating it as a double.

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

CIAnnotation

CocoaSOAP

SimpleScriptingObjects

TemperatureTester

TrackBall

**Declared In**

NSNumber.h

## **numberWithFloat:**

Creates and returns an `NSNumber` object containing a given value, treating it as a float.

```
+ (NSNumber *)numberWithFloat:(float) value
```

**Parameters**

*value*

The value for the new number.

**Return Value**

An `NSNumber` object containing *value*, treating it as a float.

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

CIAnnotation

Quartz Composer WWDC 2005 TextEdit

SampleScannerApp

SpeedometerView

TextEditPlus

**Declared In**

NSNumber.h

## **numberWithInt:**

Creates and returns an `NSNumber` object containing a given value, treating it as a signed int.

```
+ (NSNumber *)numberWithInt:(int) value
```

**Parameters**

*value*

The value for the new number.

**Return Value**

An `NSNumber` object containing *value*, treating it as a signed `int`.

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

Dicey

QTCoreVideo301

Quartz Composer WWDC 2005 TextEdit

StickiesExample

TextEditPlus

**Declared In**

`NSNumber.h`

## **numberWithInteger:**

Creates and returns an `NSNumber` object containing a given value, treating it as an `NSInteger`.

```
+ (NSNumber *)numberWithInteger:(NSInteger) value
```

**Parameters**

*value*

The value for the new number.

**Return Value**

An `NSNumber` object containing *value*, treating it as an `NSInteger`.

**Availability**

Available in Mac OS X v10.5 and later.

**Related Sample Code**

AutomatorHandsOn

Core Data HTML Store

CustomAtomicStoreSubclass

Mountains

**Declared In**

`NSNumber.h`

## **numberWithLong:**

Creates and returns an `NSNumber` object containing a given value, treating it as a signed `long`.

```
+ (NSNumber *)numberWithLong:(long) value
```

**Parameters**

*value*

The value for the new number.

**Return Value**

An `NSNumber` object containing *value*, treating it as a signed `long`.

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

AttachAScript

CocoaSpeechSynthesisExample

QTAudioExtractionPanel

QTKitPlayer

QTMetadataEditor

**Declared In**

`NSNumber.h`

**numberWithLongLong:**

Creates and returns an `NSNumber` object containing a given value, treating it as a signed `long long`.

+ (NSNumber \*)numberWithLongLong:(long long) *value*

**Parameters**

*value*

The value for the new number.

**Return Value**

An `NSNumber` object containing *value*, treating it as a signed `long long`.

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

QTKitMovieShuffler

**Declared In**

`NSNumber.h`

**numberWithShort:**

Creates and returns an `NSNumber` object containing *value*, treating it as a signed `short`.

+ (NSNumber \*)numberWithShort:(short) *value*

**Parameters**

*value*

The value for the new number.

### Return Value

An `NSNumber` object containing *value*, treating it as a signed short.

### Availability

Available in Mac OS X v10.0 and later.

### Related Sample Code

CocoaSpeechSynthesisExample

Core Data HTML Store

CoreRecipes

FunkyOverlayWindow

QTMetadataEditor

### Declared In

`NSNumber.h`

## `numberWithUnsignedChar:`

Creates and returns an `NSNumber` object containing a given value, treating it as an unsigned char.

```
+ (NSNumber *)numberWithUnsignedChar:(unsigned char)value
```

### Parameters

*value*

The value for the new number.

### Return Value

An `NSNumber` object containing *value*, treating it as an unsigned char.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

`NSNumber.h`

## `numberWithUnsignedInt:`

Creates and returns an `NSNumber` object containing a given value, treating it as an unsigned int.

```
+ (NSNumber *)numberWithUnsignedInt:(unsigned int)value
```

### Parameters

*value*

The value for the new number.

### Return Value

An `NSNumber` object containing *value*, treating it as an unsigned int.

### Availability

Available in Mac OS X v10.0 and later.

### Related Sample Code

EnhancedAudioBurn

OpenGLCaptureToMovie  
 Quartz Composer QCTV  
 Quartz Composer WWDC 2005 TextEdit  
 TextEditPlus

**Declared In**  
 NSValue.h

### **numberWithUnsignedInteger:**

Creates and returns an `NSNumber` object containing a given value, treating it as an `NSUInteger`.

```
+ (NSNumber *)numberWithUnsignedInteger:(NSUInteger) value
```

**Parameters**

*value*  
 The value for the new number.

**Return Value**

An `NSNumber` object containing *value*, treating it as an `NSUInteger`.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**  
 NSValue.h

### **numberWithUnsignedLong:**

Creates and returns an `NSNumber` object containing a given value, treating it as an unsigned `long`.

```
+ (NSNumber *)numberWithUnsignedLong:(unsigned long) value
```

**Parameters**

*value*  
 The value for the new number.

**Return Value**

An `NSNumber` object containing *value*, treating it as an unsigned `long`.

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

Apply Firmware Password  
 QTMetadataEditor  
 QTRecorder  
 Quartz Composer WWDC 2005 TextEdit  
 TextEditPlus

**Declared In**  
 NSValue.h

## numberWithUnsignedLongLong:

Creates and returns an `NSNumber` object containing a given value, treating it as an unsigned long long.

```
+ (NSNumber *)numberWithUnsignedLongLong:(unsigned long long) value
```

### Parameters

*value*

The value for the new number.

### Return Value

An `NSNumber` object containing *value*, treating it as an unsigned long long.

### Availability

Available in Mac OS X v10.0 and later.

### Related Sample Code

EnhancedAudioBurn

### Declared In

NSNumber.h

## numberWithUnsignedShort:

Creates and returns an `NSNumber` object containing a given value, treating it as an unsigned short.

```
+ (NSNumber *)numberWithUnsignedShort:(unsigned short) value
```

### Parameters

*value*

The value for the new number.

### Return Value

An `NSNumber` object containing *value*, treating it as an unsigned short.

### Availability

Available in Mac OS X v10.0 and later.

### Related Sample Code

AudioBurn

EnhancedDataBurn

QTMetadataEditor

Verification

### Declared In

NSNumber.h

## Instance Methods

### boolValue

Returns the receiver's value as a `BOOL`.

- (`BOOL`)boolValue

#### Return Value

The receiver's value as a `BOOL`, converting it as necessary.

#### Special Considerations

Prior to Mac OS X v10.3, the value returned isn't guaranteed to be one of `YES` or `NO`. A 0 value always means `NO` or `false`, but any nonzero value should be interpreted as `YES` or `true`.

#### Availability

Available in Mac OS X v10.0 and later.

#### Related Sample Code

CoreRecipes

#### Declared In

NSValue.h

### charValue

Returns the receiver's value as a `char`.

- (`char`)charValue

#### Return Value

The receiver's value as a `char`, converting it as necessary.

#### Availability

Available in Mac OS X v10.0 and later.

#### Declared In

NSValue.h

### compare:

Returns an `NSComparisonResult` value that indicates whether the receiver is greater than, equal to, or less than a given number.

- (`NSComparisonResult`)compare:(`NSNumber *`)aNumber



**Parameters***aNumber*

The number with which to compare the receiver.

This value must not be `nil`. If the value is `nil`, the behavior is undefined and may change in future versions of Mac OS X.

**Return Value**

`NSOrderedAscending` if the value of *aNumber* is greater than the receiver's, `NSOrderedSame` if they're equal, and `NSOrderedDescending` if the value of *aNumber* is less than the receiver's.

**Discussion**

The `compare:` method follows the standard C rules for type conversion. For example, if you compare an `NSNumber` object that has an integer value with an `NSNumber` object that has a floating point value, the integer value is converted to a floating-point value for comparison.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

`NSNumber.h`

**decimalValue**

Returns the receiver's value, expressed as an `NSDecimal` structure.

```
- (NSDecimal)decimalValue
```

**Return Value**

The receiver's value, expressed as an `NSDecimal` structure. The value returned isn't guaranteed to be exact for `float` and `double` values.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

`NSDecimalNumber.h`

**descriptionWithLocale:**

Returns a string that represents the contents of the receiver for a given locale.

```
- (NSString *)descriptionWithLocale:(id)aLocale
```

**Parameters***aLocale*

An object containing locale information with which to format the description. Use `nil` if you don't want the description formatted.

**Return Value**

A string that represents the contents of the receiver formatted using the locale information in *locale*.

**Discussion**

For example, if you have an `NSNumber` object that has the integer value 522, sending it the `descriptionWithLocale:` message returns the string “522”.

To obtain the string representation, this method invokes `NSString`’s `initWithFormat:locale:` method, supplying the format based on the type the `NSNumber` object was created with:

Data Type	Format Specification
<code>char</code>	<code>%i</code>
<code>double</code>	<code>%0.16g</code>
<code>float</code>	<code>%0.7g</code>
<code>int</code>	<code>%i</code>
<code>long</code>	<code>%li</code>
<code>long long</code>	<code>%lli</code>
<code>short</code>	<code>%hi</code>
<code>unsigned char</code>	<code>%u</code>
<code>unsigned int</code>	<code>%u</code>
<code>unsigned long</code>	<code>%lu</code>
<code>unsigned long long</code>	<code>%llu</code>
<code>unsigned short</code>	<code>%hu</code>

**Availability**

Available in Mac OS X v10.0 and later.

**See Also**

- [stringValue](#) (page 27)

**Declared In**

`NSNumber.h`

**doubleValue**

Returns the receiver’s value as a `double`.

- (double)doubleValue

**Return Value**

The receiver’s value as a `double`, converting it as necessary.

**Availability**

Available in Mac OS X v10.0 and later.

### Related Sample Code

CocoaSOAP  
UIKitMovieShuffler  
Quartz Composer QCTV  
SimpleScriptingObjects  
SimpleScriptingProperties

### Declared In

NSNumber.h

## floatValue

Returns the receiver's value as a `float`.

```
- (float)floatValue
```

### Return Value

The receiver's value as a `float`, converting it as necessary.

### Availability

Available in Mac OS X v10.0 and later.

### Related Sample Code

CIAnnotation  
MyPhoto  
Quartz Composer WWDC 2005 TextEdit  
TextEditPlus  
WebKitPluginWithJavaScript

### Declared In

NSNumber.h

## initWithBool:

Returns an `NSNumber` object initialized to contain a given value, treated as a `BOOL`.

```
- (id)initWithBool:(BOOL)value
```

### Parameters

*value*

The value for the new number.

### Return Value

An `NSNumber` object containing *value*, treating it as a `BOOL`.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

NSNumber.h

## initWithChar:

Returns an `NSNumber` object initialized to contain a given value, treated as a signed `char`.

```
- (id)initWithChar:(char) value
```

### Parameters

*value*

The value for the new number.

### Return Value

An `NSNumber` object containing *value*, treating it as a signed `char`.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

`NSNumber.h`

## initWithDouble:

Returns an `NSNumber` object initialized to contain *value*, treated as a `double`.

```
- (id)initWithDouble:(double) value
```

### Parameters

*value*

The value for the new number.

### Return Value

An `NSNumber` object containing *value*, treating it as a `double`.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

`NSNumber.h`

## initWithFloat:

Returns an `NSNumber` object initialized to contain a given value, treated as a `float`.

```
- (id)initWithFloat:(float) value
```

### Parameters

*value*

The value for the new number.

### Return Value

An `NSNumber` object containing *value*, treating it as a `float`.

### Availability

Available in Mac OS X v10.0 and later.

**Declared In**  
NSNumber.h

### **initWithInt:**

Returns an `NSNumber` object initialized to contain a given value, treated as a signed `int`.

- (id)initWithInt:(int) *value*

#### **Parameters**

*value*

The value for the new number.

#### **Return Value**

An `NSNumber` object containing *value*, treating it as a signed `int`.

#### **Availability**

Available in Mac OS X v10.0 and later.

**Declared In**  
NSNumber.h

### **initWithInteger:**

Returns an `NSNumber` object initialized to contain a given value, treated as an `NSInteger`.

- (id)initWithInteger:(NSInteger) *value*

#### **Parameters**

*value*

The value for the new number.

#### **Return Value**

An `NSNumber` object containing *value*, treating it as an `NSInteger`.

#### **Availability**

Available in Mac OS X v10.5 and later.

**Declared In**  
NSNumber.h

### **initWithLong:**

Returns an `NSNumber` object initialized to contain a given value, treated as a signed `long`.

- (id)initWithLong:(long) *value*

#### **Parameters**

*value*

The value for the new number.

#### **Return Value**

An `NSNumber` object containing *value*, treating it as a signed `long`.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSNumber.h

**initWithLongLong:**

Returns an NSNumber object initialized to contain *value*, treated as a signed long long.

- (id)initWithLongLong:(long long)*value*

**Parameters**

*value*

The value for the new number.

**Return Value**

An NSNumber object containing *value*, treating it as a signed long long.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSNumber.h

**initWithShort:**

Returns an NSNumber object initialized to contain a given value, treated as a signed short.

- (id)initWithShort:(short)*value*

**Parameters**

*value*

The value for the new number.

**Return Value**

An NSNumber object containing *value*, treating it as a signed short.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSNumber.h

**initWithUnsignedChar:**

Returns an NSNumber object initialized to contain a given value, treated as an unsigned char.

- (id)initWithUnsignedChar:(unsigned char)*value*

### Parameters

*value*

The value for the new number.

### Return Value

An `NSNumber` object containing *value*, treating it as an unsigned char.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

`NSNumber.h`

## **initWithUnsignedInt:**

Returns an `NSNumber` object initialized to contain a given value, treated as an unsigned int.

```
- (id)initWithUnsignedInt:(unsigned int) value
```

### Parameters

*value*

The value for the new number.

### Return Value

An `NSNumber` object containing *value*, treating it as an unsigned int.

### Availability

Available in Mac OS X v10.0 and later.

### Declared In

`NSNumber.h`

## **initWithUnsignedInteger:**

Returns an `NSNumber` object initialized to contain a given value, treated as an `NSUInteger`.

```
- (id)initWithUnsignedInteger:(NSUInteger) value
```

### Parameters

*value*

The value for the new number.

### Return Value

An `NSNumber` object containing *value*, treating it as an `NSUInteger`.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

`NSNumber.h`

**initWithUnsignedLong:**

Returns an `NSNumber` object initialized to contain a given value, treated as an unsigned long.

```
- (id)initWithUnsignedLong:(unsigned long) value
```

**Parameters**

*value*

The value for the new number.

**Return Value**

An `NSNumber` object containing *value*, treating it as an unsigned long.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

`NSNumber.h`

**initWithUnsignedLongLong:**

Returns an `NSNumber` object initialized to contain a given value, treated as an unsigned long long.

```
- (id)initWithUnsignedLongLong:(unsigned long long) value
```

**Parameters**

*value*

The value for the new number.

**Return Value**

An `NSNumber` object containing *value*, treating it as an unsigned long long.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

`NSNumber.h`

**initWithUnsignedShort:**

Returns an `NSNumber` object initialized to contain a given value, treated as an unsigned short.

```
- (id)initWithUnsignedShort:(unsigned short) value
```

**Parameters**

*value*

The value for the new number.

**Return Value**

An `NSNumber` object containing *value*, treating it as an unsigned short.

**Availability**

Available in Mac OS X v10.0 and later.



**Declared In**  
NSNumber.h

## integerValue

Returns the receiver's value as an NSInteger.

- (NSInteger)integerValue

### Return Value

The receiver's value as an NSInteger, converting it as necessary.

### Availability

Available in Mac OS X v10.5 and later.

### Related Sample Code

Mountains  
QTCoreVideo301

**Declared In**  
NSNumber.h

## intValue

Returns the receiver's value as an int.

- (int)intValue

### Return Value

The receiver's value as an int, converting it as necessary.

### Availability

Available in Mac OS X v10.0 and later.

### Related Sample Code

ABPresence  
Dicey  
EnhancedAudioBurn  
QTCoreVideo301  
WebKitPluginWithJavaScript

**Declared In**  
NSNumber.h

## isEqualToNumber:

Returns a Boolean value that indicates whether the receiver and a given number are equal.

- (BOOL)isEqualToNumber:(NSNumber \*)aNumber

**Parameters**

*aNumber*

The number with which to compare the receiver.

**Return Value**

YES if the receiver and *aNumber* are equal, otherwise NO

**Discussion**

Two NSNumber objects are considered equal if they have the same `id` values or if they have equivalent values (as determined by the [compare:](#) (page 16) method).

This method is more efficient than [compare:](#) (page 16) if you know the two objects are numbers.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSNumber.h

## longLongValue

Returns the receiver's value as a `long long`.

- (`long long`)longLongValue

**Return Value**

The receiver's value as a `long long`, converting it as necessary.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSNumber.h

## longValue

Returns the receiver's value as a `long`.

- (`long`)longValue

**Return Value**

The receiver's value as a `long`, converting it as necessary.

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

- CocoaSpeechSynthesisExample
- CustomAtomicStoreSubclass
- UIKitMovieShuffler
- Sketch-112
- WhackedTV

**Declared In**

NSValue.h

**objCType**

Returns a C string containing the Objective-C type of the data contained in the receiver.

```
- (const char *)objCType
```

**Return Value**

A C string containing the Objective-C type of the data contained in the receiver, as encoded by the `@encode()` compiler directive.

**Special Considerations**

The returned type does not necessarily match the method the receiver was created with.

**shortValue**

Returns the receiver's value as a `short`.

```
- (short)shortValue
```

**Return Value**

The receiver's value as a `short`, converting it as necessary.

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

CocoaSpeechSynthesisExample

CoreRecipes

**Declared In**

NSValue.h

**stringValue**

Returns the receiver's value as a human-readable string.

```
- (NSString *)stringValue
```

**Return Value**

The receiver's value as a human-readable string, created by invoking `descriptionWithLocale:` (page 17) where `locale` is `nil`.

**Availability**

Available in Mac OS X v10.0 and later.

**Related Sample Code**

AlbumToSlideshow

Audio Unit Effect Templates

Calculator  
VideoHardwareInfo

**Declared In**  
NSNumber.h

## unsignedCharValue

Returns the receiver's value as an unsigned char.

- (unsigned char)unsignedCharValue

### Return Value

The receiver's value as an unsigned char, converting it as necessary.

### Availability

Available in Mac OS X v10.0 and later.

**Declared In**  
NSNumber.h

## unsignedIntegerValue

Returns the receiver's value as an NSInteger.

- (NSInteger)unsignedIntegerValue

### Return Value

The receiver's value as an NSInteger, converting it as necessary.

### Availability

Available in Mac OS X v10.5 and later.

**Declared In**  
NSNumber.h

## unsignedIntValue

Returns the receiver's value as an unsigned int.

- (unsigned int)unsignedIntValue

### Return Value

The receiver's value as an unsigned int, converting it as necessary.

### Availability

Available in Mac OS X v10.0 and later.

### Related Sample Code

Dicey  
OpenGLCaptureToMovie  
Quartz Composer WWDC 2005 TextEdit

TextEditPlus  
WhackedTV

**Declared In**  
NSNumber.h

## unsignedLongLongValue

Returns the receiver's value as an unsigned long long.

- (unsigned long long)unsignedLongLongValue

### Return Value

The receiver's value as an unsigned long long, converting it as necessary.

### Availability

Available in Mac OS X v10.0 and later.

**Declared In**  
NSNumber.h

## unsignedLongValue

Returns the receiver's value as an unsigned long.

- (unsigned long)unsignedLongValue

### Return Value

The receiver's value as an unsigned long, converting it as necessary.

### Availability

Available in Mac OS X v10.0 and later.

### Related Sample Code

QTRecorder  
Quartz Composer WWDC 2005 TextEdit  
SampleScannerApp  
TextEditPlus

**Declared In**  
NSNumber.h

## unsignedShortValue

Returns the receiver's value as an unsigned short.

- (unsigned short)unsignedShortValue

### Return Value

The receiver's value as an unsigned short, converting it as necessary.

**Availability**

Available in Mac OS X v10.0 and later.

**Declared In**

NSNumber.h

# Document Revision History

---

This table describes the changes to *NSNumber Class Reference*.

Date	Notes
2008-02-08	Clarified the restricted return values that the (NSNumber) objCType method returns.
2007-10-31	Updated the description of the compare: method.
2007-01-08	Corrected definition of boolValue.
	Yes, this does look out of order, but the publication timeline is correct this way...
2006-12-01	Updated to include new API in Mac OS X v10.5.
2006-05-23	First publication of this content as a separate document.

## REVISION HISTORY

### Document Revision History



# Index

---

## B

---

boolValue [instance method 16](#)

## C

---

charValue [instance method 16](#)

compare: [instance method 16](#)

## D

---

decimalValue [instance method 17](#)

descriptionWithLocale: [instance method 17](#)

doubleValue [instance method 18](#)

## F

---

floatValue [instance method 19](#)

## I

---

initWithBool: [instance method 19](#)

initWithChar: [instance method 20](#)

initWithDouble: [instance method 20](#)

initWithFloat: [instance method 20](#)

initWithInt: [instance method 21](#)

initWithInteger: [instance method 21](#)

initWithLong: [instance method 21](#)

initWithLongLong: [instance method 22](#)

initWithShort: [instance method 22](#)

initWithUnsignedChar: [instance method 22](#)

initWithUnsignedInt: [instance method 23](#)

initWithUnsignedInteger: [instance method 23](#)

initWithUnsignedLong: [instance method 24](#)

initWithUnsignedLongLong: [instance method 24](#)

initWithUnsignedShort: [instance method 24](#)

integerValue [instance method 25](#)

intValue [instance method 25](#)

isEqualToNumber: [instance method 25](#)

## L

---

longLongValue [instance method 26](#)

longValue [instance method 26](#)

## N

---

numberWithBool: [class method 9](#)

numberWithChar: [class method 9](#)

numberWithDouble: [class method 9](#)

numberWithFloat: [class method 10](#)

numberWithInt: [class method 10](#)

numberWithInteger: [class method 11](#)

numberWithLong: [class method 11](#)

numberWithLongLong: [class method 12](#)

numberWithShort: [class method 12](#)

numberWithUnsignedChar: [class method 13](#)

numberWithUnsignedInt: [class method 13](#)

numberWithUnsignedInteger: [class method 14](#)

numberWithUnsignedLong: [class method 14](#)

numberWithUnsignedLongLong: [class method 15](#)

numberWithUnsignedShort: [class method 15](#)

## O

---

objCType [instance method 27](#)

## S

---

shortValue [instance method 27](#)

stringValue [instance method 27](#)

U

---

unsignedCharValue **instance method** [28](#)  
unsignedIntegerValue **instance method** [28](#)  
unsignedIntValue **instance method** [28](#)  
unsignedLongLongValue **instance method** [29](#)  
unsignedLongValue **instance method** [29](#)  
unsignedShortValue **instance method** [29](#)