NSNumber Class Reference

Cocoa > Data Management

2008-02-08

Contents

NSNumber Class Reference

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

Overview

`NSNumber` is a subclass of `NSValue` 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, `NSValue`. Furthermore, there is a restricted set of return values that your implementation of the `NSValue` 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

`NSValue.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

`NSValue.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
`NSValue.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
`NSValue.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

`NSValue.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

`NSValue.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

`NSValue.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

`NSValue.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
`NSValue.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
`NSValue.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

`NSValue.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

`NSValue.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

`NSValue.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
`char`	%i
`double`	%0.16g
`float`	%0.7g
`int`	%i
`long`	%li
`long long`	%lli
`short`	%hi
`unsigned char`	%u
`unsigned int`	%u
`unsigned long`	%lu
`unsigned long long`	%llu
`unsigned short`	%hu

Availability
Available in Mac OS X v10.0 and later.

See Also
`- stringValue` (page 27)

Declared In
`NSValue.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
QTKitMovieShuffler
Quartz Composer QCTV
SimpleScriptingObjects
SimpleScriptingProperties

Declared In
`NSValue.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
`NSValue.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
`NSValue.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

`NSValue.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

`NSValue.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
`NSValue.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
`NSValue.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
`NSValue.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
`NSValue.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
`NSValue.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
`NSValue.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

`NSValue.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

`NSValue.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

`NSValue.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

`NSValue.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

`NSValue.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
`NSValue.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
`NSValue.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
`NSValue.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`r

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

`NSValue.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

`NSValue.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

QTKitMovieShuffler

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
`NSValue.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
`NSValue.h`

unsignedIntegerValue

Returns the receiver's value as an `NSUInteger`.

```
- (NSUInteger)unsignedIntegerValue
```

Return Value
The receiver's value as an `NSUInteger`, converting it as necessary.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`NSValue.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
`NSValue.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
`NSValue.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
`NSValue.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
`NSValue.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 (NSValue) 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.

Index