NSValue Class Reference

Cocoa > Data Management

2007-10-31

Apple Inc.
© 2007 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, and Objective-C are trademarks of Apple Inc., registered in the United States and other countries.

iPhone is a trademark of Apple Inc.

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

NSValue Class Reference 5

Overview 5
Adopted Protocols 5
Tasks 6
  Creating an NSValue 6
  Accessing Data 6
  Comparing Objects 7
Class Methods 7
  value:withObjCType: 7
  valueWithBytes:objCType: 7
  valueWithNonretainedObject: 8
  valueWithPoint: 9
  valueWithPointer: 9
  valueWithRange: 10
  valueWithRect: 10
  valueWithSize: 11
Instance Methods 11
  getValue: 11
  initWithBytes:objCType: 12
  isEqualToValue: 12
  nonretainedObjectValue 13
  objCType 13
  pointerValue 13
  pointValue 14
  rangeValue 14
  rectValue 14
  sizeValue 15

Document Revision History 17

Index 19

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

NSValue Class Reference

Inherits from	NSObject
Conforms to	NSCoding NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/Foundation.framework
Availability	Available in Mac OS X v10.0 and later.
Companion guide	Number and Value Programming Topics for Cocoa
Declared in	NSGeometry.h NSRange.h NSValue.h
Related sample code	iSpend QTAudioExtractionPanel QTKitMovieShuffler Quartz Composer WWDC 2005 TextEdit TextEditPlus

Overview

An `NSValue` object is a simple container for a single C or Objective-C data item. It can hold any of the scalar types such as `int`, `float`, and `char`, as well as pointers, structures, and object `id`s. The purpose of this class is to allow items of such data types to be added to collections such as instances of `NSArray` and `NSSet`, which require their elements to be objects. `NSValue` objects are always immutable.

Adopted Protocols

NSCoding

```
encodeWithCoder:
initWithCoder:
```

NSCopying

```
- copyWithZone:
```


2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

Tasks

Creating an NSValue

- `initWithBytes:objCType:` (page 12)
  Initializes and returns an `NSValue` object that contains a given value, which is interpreted as being of a given Objective-C type.
+ `valueWithBytes:objCType:` (page 7)
  Creates and returns an `NSValue` object that contains a given value, which is interpreted as being of a given Objective-C type.
+ `value:withObjCType:` (page 7)
  Creates and returns an `NSValue` object that contains a given value which is interpreted as being of a given Objective-C type.
+ `valueWithNonretainedObject:` (page 8)
  Creates and returns an `NSValue` object that contains a given object.
+ `valueWithPointer:` (page 9)
  Creates and returns an `NSValue` object that contains a given pointer.
+ `valueWithPoint:` (page 9)
  Creates and returns an `NSValue` object that contains a given `NSPoint` structure.
+ `valueWithRange:` (page 10)
  Creates and returns an `NSValue` object that contains a given `NSRange` structure.
+ `valueWithRect:` (page 10)
  Creates and returns an `NSValue` object that contains a given `NSRect` structure.
+ `valueWithSize:` (page 11)
  Creates and returns an `NSValue` object that contains a given `NSSize` structure.

Accessing Data

- `getValue:` (page 11)
  Copies the receiver's value into a given buffer.
- `nonretainedObjectValue` (page 13)
  Returns the receiver's value as an `id`.
- `objCType` (page 13)
  Returns a C string containing the Objective-C type of the data contained in the receiver.
- `pointValue` (page 14)
  Returns an `NSPoint` structure representation of the receiver.
- `pointerValue` (page 13)
  Returns the receiver's value as a pointer to `void`.
- `rangeValue` (page 14)
  Returns an `NSRange` structure representation of the receiver.
- `rectValue` (page 14)
  Returns an `NSRect` structure representation of the receiver.

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

- sizeValue (page 15)
  Returns an NSSize structure representation of the receiver.

Comparing Objects

- isEqualToValue: (page 12)
  Returns a Boolean value that indicates whether the receiver and another value are equal.

Class Methods

value:withObjCType:

Creates and returns an NSValue object that contains a given value which is interpreted as being of a given Objective-C type.

```
+ (NSValue *)value:(const void *)value withObjCType:(const char *)type
```

Parameters

value
  The value for the new NSValue object.

type
  The Objective-C type of *value*. *type* should be created with the Objective-C @encode() compiler directive; it should not be hard-coded as a C string.

Return Value
A new NSValue object that contains *value*, which is interpreted as being of the Objective-C type *type*.

Discussion
This method has the same effect as valueWithBytes:objCType: (page 7) and may be deprecated in a future release. You should use valueWithBytes:objCType: (page 7) instead.

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

See Also
+ valueWithBytes:objCType: (page 7)

Related Sample Code
VideoViewer

Declared In
NSValue.h

valueWithBytes:objCType:

Creates and returns an NSValue object that contains a given value, which is interpreted as being of a given Objective-C type.

```
+ (NSValue *)valueWithBytes:(const void *)value objCType:(const char *)type
```


2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

Parameters

value

The value for the new NSValue object.

type

The Objective-C type of *value*. *type* should be created with the Objective-C @encode() compiler directive; it should not be hard-coded as a C string.

Return Value

A new NSValue object that contains *value*, which is interpreted as being of the Objective-C type *type*.

Discussion

See *Number and Value Programming Topics for Cocoa* for other considerations in creating an NSValue object and code examples.

Availability

Available in Mac OS X v10.0 and later.

See Also

- initWithBytes:objCType: (page 12)

Declared In

NSValue.h

valueWithNonretainedObject:

Creates and returns an NSValue object that contains a given object.

```
+ (NSValue *)valueWithNonretainedObject:(id)anObject
```

Parameters

anObject

The value for the new object.

Return Value

A new NSValue object that contains *anObject*.

Discussion

This method is equivalent to invoking value:withObjCType: (page 7) in this manner:

```
NSValue *theValue = [NSValue value:&anObject withObjCType:@encode(void *)];
```

This method is useful for preventing an object from being retained when it's added to a collection object (such as an instance of NSArray or NSDictionary).

Availability

Available in Mac OS X v10.0 and later.

See Also

- nonretainedObjectValue (page 13)

Declared In

NSValue.h

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

valueWithPoint:

Creates and returns an `NSValue` object that contains a given `NSPoint` structure.

```
+ (NSValue *)valueWithPoint:(NSPoint)aPoint
```

Parameters

aPoint

The value for the new object.

Return Value

A new `NSValue` object that contains the value of *point*.

Availability

Available in Mac OS X v10.0 and later.

See Also

`- pointValue` (page 14)

Related Sample Code

Aperture Edit Plugin - Borders & Titles

Dicey

ImageMapExample

PDF Annotation Editor

TrackBall

Declared In

`NSGeometry.h`

valueWithPointer:

Creates and returns an `NSValue` object that contains a given pointer.

```
+ (NSValue *)valueWithPointer:(const void *)aPointer
```

Parameters

aPointer

The value for the new object.

Return Value

A new `NSValue` object that contains *aPointer*.

Discussion

This method is equivalent to invoking `value:withObjCType:` (page 7) in this manner:

```
NSValue *theValue = [NSValue value:&aPointer withObjCType:@encode(void *)];
```

This method does not copy the contents of *aPointer*, so you must not to deallocate the memory at the pointer destination while the `NSValue` object exists. `NSData` objects may be more suited for arbitrary pointers than `NSValue` objects.

Availability

Available in Mac OS X v10.0 and later.

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

See Also
- `pointerValue` (page 13)

Declared In
`NSValue.h`

valueWithRange:

Creates and returns an `NSValue` object that contains a given `NSRange` structure.

```
+ (NSValue *)valueWithRange:(NSRange)range
```

Parameters

range
The value for the new object.

Return Value
A new `NSValue` object that contains the value of *range*.

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

See Also
- `rangeValue` (page 14)

Declared In
`NSRange.h`

valueWithRect:

Creates and returns an `NSValue` object that contains a given `NSRect` structure.

```
+ (NSValue *)valueWithRect:(NSRect)rect
```

Parameters

rect
The value for the new object.

Return Value
A new `NSValue` object that contains the value of *rect*.

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

See Also
- `rectValue` (page 14)

Related Sample Code
IBFragmentView
iSpend
QTCoreVideo301
Reducer

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

Declared In
NSGeometry.h

valueWithSize:

Creates and returns an NSValue object that contains a given NSSize structure.

```
+ (NSValue *)valueWithSize:(NSSize)size
```

Parameters

size

The value for the new object.

Return Value
A new NSValue object that contains the value of *size*.

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

See Also
- sizeValue (page 15)

Related Sample Code
Dicey
ImageMapExample
QTKitMovieShuffler
Quartz Composer WWDC 2005 TextEdit
TextEditPlus

Declared In
NSGeometry.h

Instance Methods

getValue:

Copies the receiver's value into a given buffer.

```
- (void)getValue:(void *)buffer
```

Parameters

buffer

A buffer into which to copy the receiver's value. *buffer* must be large enough to hold the value.

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

Related Sample Code
VideoViewer

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

Declared In
NSValue.h

initWithBytes:objCType:

Initializes and returns an NSValue object that contains a given value, which is interpreted as being of a given Objective-C type.

```
- (id)initWithBytes:(const void *)value objCType:(const char *)type
```

Parameters

value

The value for the new NSValue object.

type

The Objective-C type of *value*. *type* should be created with the Objective-C @encode() compiler directive; it should not be hard-coded as a C string.

Return Value
An initialized NSValue object that contains *value*, which is interpreted as being of the Objective-C type *type*. The returned object might be different than the original receiver.

Discussion
See *Number and Value Programming Topics for Cocoa* for other considerations in creating an NSValue object.

This is the designated initializer for the NSValue class.

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

Declared In
NSValue.h

isEqualToValue:

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

```
- (BOOL)isEqualToValue:(NSValue *)value
```

Parameters

aValue

The value with which to compare the receiver.

Return Value
YES if the receiver and *aValue* are equal, otherwise NO. For NSValue objects, the class, type, and contents are compared to determine equality.

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

Declared In
NSValue.h

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

nonretainedObjectValue

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

```
- (id)nonretainedObjectValue
```

Return Value
The receiver's value as an `id`. If the receiver was not created to hold a pointer-sized data item, the result is undefined.

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

See Also
`- getValue:` (page 11)

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.

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

Declared In
`NSValue.h`

pointerValue

Returns the receiver's value as a pointer to `void`.

```
- (void *)pointerValue
```

Return Value
The receiver's value as a pointer to `void`. If the receiver was not created to hold a pointer-sized data item, the result is undefined.

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

See Also
`- getValue:` (page 11)

Declared In
`NSValue.h`

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

pointValue

Returns an `NSPoint` structure representation of the receiver.

```
- (NSPoint)pointValue
```

Return Value
An `NSPoint` structure representation of the receiver.

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

See Also
- `rectValue` (page 14)
- `sizeValue` (page 15)

Declared In
`NSGeometry.h`

rangeValue

Returns an `NSRange` structure representation of the receiver.

```
- (NSRange)rangeValue
```

Return Value
An `NSRange` structure representation of the receiver.

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

See Also
+ `valueWithRange:` (page 10)

Declared In
`NSRange.h`

rectValue

Returns an `NSRect` structure representation of the receiver.

```
- (NSRect)rectValue
```

Return Value
An `NSRect` structure representation of the receiver.

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

See Also
- `pointValue` (page 14)
- `sizeValue` (page 15)

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

Related Sample Code
IBFragmentView

Declared In
`NSGeometry.h`

sizeValue

Returns an `NSSize` structure representation of the receiver.

```
- (NSSize)sizeValue
```

Return Value
An `NSSize` structure representation of the receiver.

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

See Also
- `pointValue` (page 14)
- `rectValue` (page 14)

Related Sample Code
QTKitAdvancedDocument
QTKitCreateMovie
QTKitFrameStepper
QTKitMovieShuffler
QTKitTimeCode

Declared In
`NSGeometry.h`

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

Document Revision History

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

Date	Notes
2007-10-31	Corrected typographical errors.
2006-05-23	First publication of this content as a separate document.

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

Document Revision History

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.

Index

G

`getValue:` instance method 11

I

`initWithBytes:objCType:` instance method 12
`isEqualToValue:` instance method 12

N

`nonretainedObjectValue` instance method 13

O

`objCType` instance method 13

P

`pointerValue` instance method 13
`pointValue` instance method 14

R

`rangeValue` instance method 14
`rectValue` instance method 14

S

`sizeValue` instance method 15

V

`value:withObjCType:` class method 7
`valueWithBytes:objCType:` class method 7
`valueWithNonretainedObject:` class method 8
`valueWithPoint:` class method 9
`valueWithPointer:` class method 9
`valueWithRange:` class method 10
`valueWithRect:` class method 10
`valueWithSize:` class method 11

2007-10-31 | © 2007 Apple Inc. All Rights Reserved.