

NSData

Inherits From:	NSObject
Conforms To:	NSCopying NSCoding NSMutableCopying NSObject
Declared In:	foundation/NSData.h

Allocating and Initializing a Data Object

+ (id) allocWithZone: (NSZone *) <i>zone</i>	Creates and returns an uninitialized data object from <i>zone</i> .
+ (id) data	Creates and returns an empty data object. This method is declared primarily for mutable subclasses of NSData.
+ (id) dataWithBytes: (const void *) <i>bytes</i> length: (unsigned int) <i>length</i>	Creates and returns a data object containing <i>length</i> bytes of data copied from the buffer <i>bytes</i> .
+ (id) dataWithBytesNoCopy: (void *) <i>bytes</i> length: (unsigned int) <i>length</i>	Creates and returns a data object containing <i>length</i> bytes from the buffer <i>bytes</i> .
+ (id) dataWithContentsOfFile: (NSString *) <i>path</i>	Creates and returns a data object by reading data from the file specified by <i>path</i> .
+ (id) dataWithContentsOfMappedFile: (NSString *) <i>path</i>	Creates and returns a data object whose contents come from the mapped file <i>path</i> , assuming mapped files are available on the underlying operating system. If mapped files are not available, this method is identical to dataWithContentsOfFile: .
- (id) initWithBytes: (const void *) <i>bytes</i> length: (unsigned int) <i>length</i>	Initializes a newly allocated data object by putting in it <i>length</i> bytes of data copied from the buffer <i>bytes</i> .
- (id) initWithBytesNoCopy: (void *) <i>bytes</i> length: (unsigned int) <i>length</i>	Initializes a newly allocated data object by putting in it <i>length</i> bytes of data from the buffer <i>bytes</i> .

- (id)**initWithContentsOfFile:**(NSString *)*path* Initializes a newly allocated data object by reading into it the data from the file specified by *path*.
- (id)**initWithContentsOfMappedFile:**(NSString *)*path* Initializes a newly allocated data object to contain the data residing in the mapped file *path*, assuming mapped files are available on the underlying operating system. If mapped files are not available, this method is identical to **initWithContentsOfFile:**.
- (id)**initWithData:**(NSData *)*data* Initializes a newly allocated data object by placing in it the contents of another data object, *data*.

Accessing Data

- (const void *)**bytes** Returns a pointer to the data object's contents. This method returns read-only access to the data.
- (NSString *)**description** Returns an NSString object that contains a hexadecimal representation of the the receiver's contents.
- (void)**getBytes:**(void *)*buffer* Copies a data object's contents into *buffer*.
- (void)**getBytes:**(void *)*buffer*
length:(unsigned int)*length* Copies *length* bytes in a data object's contents into *buffer*.
- (void)**getBytes:**(void *)*buffer*
range:(NSRange)*aRange* Copies into *buffer* the portion of the data object's contents within *aRange*. *aRange* must specify a range within the receiver's data; otherwise, an NSRangeException error is raised.
- (NSData *)**subdataWithRange:**(NSRange)*aRange* Returns a data object containing a copy of the receiver's bytes that fall within the limits specified by *aRange*. *aRange* must specify a range within the receiver's data; otherwise, an NSRangeException error is raised.

Querying a Data Object

- (BOOL)**isEqualToData:**(NSData *)*other* Compares the receiving data object to *other*. If the contents of *other* are equal to the contents of the receiver, this method returns YES. If not, it returns NO.
- (unsigned int)**length** Returns the number of bytes contained in a data object.

Storing Data

- (BOOL)**writeToFile:**(NSString *)*path*
atomically:(BOOL)*useAuxiliaryFile*

Writes the bytes in the receiving data object to the file specified by *path*. If *useAuxiliaryFile* is YES, the data is written to a backup file and then, assuming no errors occur, the backup file is renamed atomically to the intended file name.

Deserializing Data

- (unsigned int)**deserializeAlignedBytesLengthAtCursor:**(unsigned int*)*cursor*

Returns the length of the serialized bytes at the location referenced by *cursor*. If the bytes have been page-aligned, it also obtains the relevant "hole" information and adjusts the cursor. An invocation of this method must have a corresponding **serializeAlignedBytesLength:** invocation.

- (void)**deserializeBytes:**(void *)*buffer*
length:(unsigned int)*bytes*
atCursor:(unsigned int*)*cursor*

Deserializes *bytes* number of bytes in the buffer pointed at by *buffer*, places them internally starting at *cursor*, and advances the cursor.

- (void)**deserializeDataAt:**(void *)*data*
ofObjCType:(const char *)*type*
atCursor:(unsigned int*)*cursor*
context:(id <NSObjCTypeSerializationCallback>)element is an object other than an instance of

Deserializes the data pointed at by *cursor*, interpreting it by the Objective C type specifier *type* and writing it to the memory location referenced by *data*. If the data *callback*

NSDictionary, NSArray, NSString, or NSData, a callback from object *callback* can provide further definition of the object. All Objective C types are currently supported except **union** and **void ***. Pointers refer to a single item.

- (int)**deserializeIntAtCursor:**(unsigned int*)*cursor*

Deserializes and returns the integer encoded at *cursor*. Also advances the cursor.

- (int)**deserializeIntAtLocation:**(unsigned int)*location*

Deserializes and returns the integer encoded at offset *location*. Does not advance the cursor.

- (void)**deserializeInts:**(int *)*intBuffer*
count:(unsigned int)*numInts*
atCursor:(unsigned int*)*cursor*

Deserializes *numInts* integers encoded at the location referenced by *cursor* and puts them in the buffer *intBuffer*. Also advances the cursor.

- (void)**deserializeInts:**(int *)*intBuffer*
count:(unsigned int)*numInts*
atLocation:(unsigned int)*location*

Deserializes *numInts* integers encoded at offset *location* and puts them in the buffer *intBuffer*. Does not advance the cursor.

