

NSUnarchiver

Inherits From:	NSCoder : NSObject
Conforms To:	NSCoding NSObject
Declared In:	foundation/NSArchiver.h

Class Description

NSUnarchiver, a concrete subclass of NSCoder, defines objects that can decode a data structure, such as a graph of Objective C objects, from an archive. Such archives are produced by objects of the NSArchiver class. See the NSArchiver specification for an introduction to archiving.

General Exception Conditions

While unarchiving, NSUnarchiver performs a variety of consistency checks on the incoming data stream.

NSUnarchiver raises an `NSInconsistentArchiveException` for a variety of reasons. Possible data errors leading to this exception are: unknown type descriptors in the data file; an array type descriptor is incorrectly terminated (missing `]`); excess characters in a type descriptor; a null class found where a concrete class was expected class not loaded.

Instance Variables

None declared in this class.

Method Types

Initializing an NSUnarchiver

- `initWithReadingWithData:`

Decoding Objects

+ `unarchiveObjectWithData:`

+ `unarchiveObjectWithFile:`

- `decodeArrayOfObjCType:count:at:`

Managing an NSUnarchiver

- `isAtEnd`

- `objectZone`

- `setObjectZone:`

- `systemVersion`

Substituting One Class for Another

+ `classNameDecodedForArchiveClassName:`

+ `decodeClassName:asClassName:`

- `classNameDecodedForArchiveClassName:`

- `decodeClassName:asClassName:`

Class Methods

classNameDecodedForArchiveClassName:

+ (NSString *)**classNameDecodedForArchiveClassName:**
 (NSString *)*nameInArchive*

Returns the class name used to archive instances of the class (*nameInArchive*). This may not be the original class name but another name encoded with NSArchiver's **encodeClassName:intoClassName**.

decodeClassName:asClassName:

+ (void)**decodeClassName:**(NSString *)*nameInArchive*
 asClassName:(NSString *)*trueName*

Decodes from the archived data a class name (*nameInArchive*) substituted for the real class name (*trueName*). This method enables easy conversion of unarchived data when there are name changes in classes.

unarchiveObjectWithData:

+ (id)**unarchiveObjectWithData:**(NSData *)*data*

Decodes an archived object stored in *data*.

unarchiveObjectWithFile:

+ (id)**unarchiveObjectWithFile:**(NSString *)*path*

Decodes an archived object stored in the file *path*.

Instance Methods

classNameDecodedForArchiveClassName:

- (NSString *)**classNameDecodedForArchiveClassName:**
 (NSString *)*nameInArchive*

Returns the class name used to archive instances of the class (*nameInArchive*). This may not be the original class name but another name encoded with NSArchiver's **encodeClassName:intoClassName**.

decodeArrayOfObjCType:count:at:

- (void)**decodeArrayOfObjCType:**(const char *)*itemType* **count:**(unsigned int)*count* **at:**(void *)*array*

Decodes an *array* of *count* data elements of the same Objective C data *itemType*. It is your responsibility to release any objects derived in this way.

decodeClassName:

- (void)**decodeClassName:**(NSString *)*nameInArchive*
 asClassName:(NSString *)*trueName*

Decodes from the archived data a class name (*nameInArchive*) substituted for the real class name (*trueName*). This method enables easy conversion of unarchived data when there are name changes in classes.

initWithReadingWithData:

- (id) **initWithReadingWithData:** (NSData *)*data*

Initializes an NSUnarchiver object from data object *data*. Raises **NSInvalidArgumentException** if the *data* argument is **nil**.

isAtEnd

- (BOOL) **isAtEnd**

Returns YES if the end of data is reached, NO if more data follows.

objectZone

- (NSZone *) **objectZone**

Returns the allocation zone for the unarchiver object.

setObjectZone:

- (void) **setObjectZone:** (NSZone *)*zone*

Sets the allocation zone for the unarchiver object to *zone*. If *zone* is **nil**, it sets it to the default zone.

systemVersion

- (unsigned int) **systemVersion**

Returns the system version number for the unarchived data.

