NSArchiver

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

Class Description

NSArchiver, a concrete subclass of NSCoder, defines an object that encodes Objective C objects into an architecture-neutral format that can be stored in a file. When objects are archived, their class information and the values of their instance variables are written to the archive. NSArchiver's companion class, NSUnarchiver, takes an archive file and decodes its contents into a set of objects equivalent to the original one.

Archiving is typically initiated by sending an NSArchiver an **encodeRootObject:** or **archiveRootObject:toFile:** message. These messages specify a single object that is the starting point for archiving. The root object receives an **encodeWithCoder:** message (see the NSCoding protocol) that allows it to begin archiving itself and the other objects that it's connected to. An object responds to an **encodeWithCoder:** message by writing its instance variables to the archiver.

An object doesn't have to archive the values of each of its instance variables. Some values may not be important to reestablish and others may be derivable from related state upon unarchiving. Other instance variables should be written to the archive only under certain conditions, as explained below.

NSArchiver overrides the inherited **encodeRootObject:** and **encodeConditionalObject:** methods to support the conditional archiving of members of a graph of objects. When an object receives an **encodeWithCoder:** message, it should respond by unconditionally archiving instance variables that are intrinsic to its nature (with the exceptions noted above) and conditionally archiving those that are not. For example, an NSView unconditionally archives its array of subviews (using **encodeObject:**, or the like) but conditionally archives its superview (using **encodeConditionalObject:**). The archiving system notes each reference to a conditional object, but doesn't actually archive the object unless some other object in the graph requests the object to be archived unconditionally. This ensures that an object is only archived once despite multiple references to it in the object graph. When the objects are extracted from the archive, the multiple references to objects are resolved, and an equivalent graph of objects is reestablished.

Note: As a consequence of being a subclass of NSObject, NSArchiver conforms to the NSCoding protocol. In practice, however, NSArchivers are not encoded nor archived.

Instance Variables

None declared in this class.

Method Types

Initializing an NSArchiver	– initForWritingWithMutableData:
Archiving Data	+ archivedDataWithRootObject: + archiveRootObject:toFile: – encodeArrayOfObjCType:count:at: – encodeConditionalObject: – encodeRootObject:
Getting Data from the NSArchiver	– archiverData
Substituting One Class for Another	+ classNameEncodedForTrueClassName: – encodeClassName:intoClassName:

Class Methods

archivedDataWithRootObject:

+ (NSData *)**archivedDataWithRootObject:**(id)*rootObject*

Creates and returns a data object after initializing an archiver with that object and encoding the archiver with *rootObject*.

archiveRootObject:toFile:

+ (BOOL)**archiveRootObject:**(id)*rootObject* **toFile:**(NSString *)*path*

Archives *rootObject* by encoding it as a data object in an archiver and writing that data object to file *path*. Returns YES upon success.

classNameEncodedForTrueClassName:

+ (NSString *)**classNameEncodedForTrueClassName:**(NSString *)*trueName*

Returns the string object representing the class name used to archive instances of the class. This class name might not be the original name (*trueName*).

See also: – encodeClassName:intoClassName.

Instance Methods

archiverData

– (NSMutableData *)**archiverData**

Returns the data object, in mutable form, that is associated with the receiving NSArchiver.

encodeArrayOfObjCType:count:at:

– (void)**encodeArrayOfObjCType:**(const char *)*type* **count:**(unsigned int)*count*
at:(const void *)*array*

Encodes an *array* of *count* data elements of the same Objective C data *type*.

encodeClassName:intoClassName:

– (void)**encodeClassName:**(NSString *)*trueName*
intoClassName:(NSString *)*inArchiveName*

Encodes in the archived data a substitute class name for the real class name (*trueName*).

encodeConditionalObject:

– (void)**encodeConditionalObject:**(id)*object*

Encodes into the linearized data a conditional *object* that points back toward a root object. If **nil** is specified for *object*, it encodes it as **nil** unconditionally. Raises an exception if no root object has been encoded.

encodeRootObject:

– (void)**encodeRootObject:**(id)*rootObject*

Encodes the *rootObject* at the start of the linearized data representing the object graph. Raises an exception if the root object has already been encoded.

initForWritingWithMutableData:

– (id)**initForWritingWithMutableData:**(NSMutableData *)*mdata*

Initializes an archiver, encoding stream and version information into mutable data *mdata*.