
NSFileWrapper Class Reference

[Cocoa](#) > [Data Management](#)



2008-10-15



Apple Inc.
© 2008 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, Mac, Mac OS, and Quartz are trademarks of Apple Inc., registered in the United States and other countries.

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

NSFileWrapper Class Reference 5

Overview	5
Adopted Protocols	6
Tasks	6
Creating File Wrappers	6
Querying File Wrappers	6
Accessing File-Wrapper Information	6
Updating File Wrappers	7
Serializing	7
Accessing Files	7
Writing Files	8
Instance Methods	8
addFilePath:	8
addFileWrapper:	9
addRegularFileWithContents:preferredFilename:	9
addSymbolicLinkWithDestination:preferredFilename:	10
fileAttributes	11
filename	11
fileWrappers	11
icon	12
initWithDirectoryWithFileWrappers:	12
initWithRegularFileWithContents:	13
initWithSymbolicLinkWithDestination:	14
initWithPath:	14
initWithSerializedRepresentation:	15
isDirectory	15
isRegularFile	16
isSymbolicLink	16
keyForFileWrapper:	16
needsToBeUpdatedFromPath:	17
preferredFilename	18
regularFileContents	18
removeFileWrapper:	18
serializedRepresentation	19
setFileAttributes:	19
setFilename:	20
setIcon:	20
setPreferredFilename:	20
symbolicLinkDestination	21
updateFromPath:	21
writeToFile:atomically:updateFileNames:	22

Document Revision History 23

Index 25

NSFileWrapper Class Reference

Inherits from	NSObject
Conforms to	NSCoding NSObject (NSObject)
Framework	/System/Library/Frameworks/AppKit.framework
Availability	Available in Mac OS X v10.0 and later.
Companion guide	Application File Management
Declared in	NSFileWrapper.h
Related sample code	CoreRecipes File Wrappers with Core Data Documents Quartz Composer WWDC 2005 TextEdit StickiesExample TextEditPlus

Overview

The `NSFileWrapper` class provides access to the attributes and contents of filesystem nodes. A **filesystem node** is a file, directory, or symbolic link. Instances of this class are known as **file wrappers**.

File wrappers represent a filesystem node as an object that can be displayed as an image (and possibly edited in place), saved to the filesystem, or transmitted to another application. It can also store an icon for representing the node in a document or in a dragging operation.

There are three types of file wrappers:

- **Regular-file file wrapper:** Represents a regular file node.
- **Directory file wrapper:** Represents a directory node.
- **Symbolic-link file wrapper:** Represents a symbolic-link node.

A file wrapper has these attributes:

- **Filename.** Name of the filesystem node the file wrapper represents.
- **Icon:** Image that represents the file wrapper to the user.
- **Filesystem attributes.** See `NSFileManager` for information on the contents of the *attributes* dictionary.

- **Regular-file contents.** Applicable only to regular-file file wrappers.
- **File wrappers.** Applicable only to directory file wrappers.
- **Destination node.** Applicable only to symbolic-link file wrappers.

Adopted Protocols

NSCoding

`encodeWithCoder:`
`initWithCoder:`

Tasks

Creating File Wrappers

This class does not have a designated initializer.

- [initWithPath:](#) (page 14)
Initializes the receiver with a given node.
- [initWithDirectoryWithFileWrappers:](#) (page 12)
Initializes the receiver as a directory file wrapper, with a given file-wrapper list.
- [initWithRegularFileWithContents:](#) (page 13)
Initializes the receiver as a regular-file file wrapper.
- [initWithSymbolicLinkWithDestination:](#) (page 14)
Initializes the receiver as a symbolic-link file wrapper.
- [initWithSerializedRepresentation:](#) (page 15)
Initializes the receiver from given serialized data.

Querying File Wrappers

- [isRegularFile](#) (page 16)
Indicates whether the receiver is a regular-file file wrapper.
- [isDirectory](#) (page 15)
Indicates whether the receiver is a directory file wrapper.
- [isSymbolicLink](#) (page 16)
Indicates whether the receiver is a symbolic-link file wrapper.

Accessing File-Wrapper Information

- [fileWrappers](#) (page 11)
Provides the file wrappers contained by the receiver, which must be a directory file wrapper.

- [addFileWrapper:](#) (page 9)
Adds a file wrapper to the receiver, which must be a directory file wrapper.
- [removeFileWrapper:](#) (page 18)
Removes a file wrapper from the receiver, which must be a directory file wrapper.
- [addFileWithPath:](#) (page 8)
Creates a file wrapper from a given filesystem node and adds it to the receiver, which must be a directory file wrapper.
- [addRegularFileWithContents:preferredFilename:](#) (page 9)
Creates a regular-file file wrapper with the given contents and adds it to the receiver, which must be a directory file wrapper.
- [addSymbolicLinkWithDestination:preferredFilename:](#) (page 10)
Creates a symbolic-link file wrapper pointing to a given filesystem node and adds it to the receiver, which must be a directory file wrapper.
- [keyForFileWrapper:](#) (page 16)
Provides a key used by the receiver to identify a given file wrapper. The receiver must be a dictionary file wrapper.
- [symbolicLinkDestination](#) (page 21)
Provides the pathname referenced by the receiver, which must be a symbolic-link file wrapper.

Updating File Wrappers

- [needsToBeUpdatedFromPath:](#) (page 17)
Indicates whether the receiver needs to be updated to match a given filesystem node.
- [updateFromPath:](#) (page 21)
Updates the receiver to match a given filesystem node.

Serializing

- [serializedRepresentation](#) (page 19)
Provides the receiver's contents as an opaque collection of data.

Accessing Files

- [filename](#) (page 11)
Provides the receiver's filename.
- [setFilename:](#) (page 20)
Specifies the receiver's filename.
- [preferredFilename](#) (page 18)
Provides the receiver's preferred filename.
- [setPreferredFilename:](#) (page 20)
Specifies the receiver's preferred filename.
- [icon](#) (page 12)
Provides an image that represents the receiver to the user.

- [setIcon:](#) (page 20)
Specifies the image to be used to represent the receiver to the user.
- [fileAttributes](#) (page 11)
Provides the receiver's file attributes.
- [setFileAttributes:](#) (page 19)
Specifies the receiver's file attributes.
- [regularFileContents](#) (page 18)
Provides the contents of the receiver's filesystem node. The receiver must be a regular-file file wrapper.

Writing Files

- [writeToFile:atomically:updateFileNames:](#) (page 22)
Writes the receiver's contents to a given filesystem node.

Instance Methods

addFilePath:

Creates a file wrapper from a given filesystem node and adds it to the receiver, which must be a directory file wrapper.

```
- (NSString *)addFilePath:(NSString *)node
```

Parameters

node

Filesystem node from which to create the file wrapper to add to the receiver.

Return Value

Dictionary key used to store the new file wrapper in the receiver's list of file wrappers. See [Working With Directory Wrappers](#) for more information.

Discussion

This method raises `NSInternalInconsistencyException` when the receiver is not a directory file wrapper.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [addRegularFileWithContents:preferredFilename:](#) (page 9)
- [addSymbolicLinkWithDestination:preferredFilename:](#) (page 10)
- [removeFileWrapper:](#) (page 18)
- [fileWrappers](#) (page 11)

Related Sample Code

File Wrappers with Core Data Documents

Declared In

`NSFileWrapper.h`

addFileWrapper:

Adds a file wrapper to the receiver, which must be a directory file wrapper.

```
- (NSString *)addFileWrapper:(NSFileWrapper *)fileWrapper
```

Parameters

fileWrapper

File wrapper to add to the receiver. Raises `NSInvalidArgumentException` when the file wrapper doesn't have a preferred name.

Return Value

Dictionary key used to store *fileWrapper* in the receiver's list of file wrappers. See [Working With Directory Wrappers](#) for more information.

Discussion

This method raises `NSInternalInconsistencyException` when the receiver is not a directory file wrapper.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [addFileWithPath:](#) (page 8)
- [addRegularFileWithContents:preferredFilename:](#) (page 9)
- [addSymbolicLinkWithDestination:preferredFilename:](#) (page 10)
- [removeFileWrapper:](#) (page 18)
- [fileWrappers](#) (page 11)
- [preferredFilename](#) (page 18)

Related Sample Code

[File Wrappers with Core Data Documents](#)

Declared In

`NSFileWrapper.h`

addRegularFileWithContents:preferredFilename:

Creates a regular-file file wrapper with the given contents and adds it to the receiver, which must be a directory file wrapper.

```
- (NSString *)addRegularFileWithContents:(NSData *)regularFileContents
preferredFilename:(NSString *)preferredFilename
```

Parameters

regularFileContents

Contents for the new regular-file file wrapper.

preferredFilename

Preferred filename for the new regular-file file wrapper. A `nil` or empty value raises `NSInvalidArgumentException`.

Return Value

Dictionary key used to store the new file wrapper in the receiver's list of file wrappers.

Discussion

This method raises `NSInternalInconsistencyException` when the receiver is not a directory file wrapper.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [addFilePath:](#) (page 8)
- [addSymbolicLinkWithDestination:preferredFilename:](#) (page 10)
- [removeFileWrapper:](#) (page 18)
- [fileWrappers](#) (page 11)

Related Sample Code

StickiesExample

Declared In

NSFileWrapper.h

addSymbolicLinkWithDestination:preferredFilename:

Creates a symbolic-link file wrapper pointing to a given filesystem node and adds it to the receiver, which must be a directory file wrapper.

```
- (NSString *)addSymbolicLinkWithDestination:(NSString *)node
    preferredFilename:(NSString *)preferredFilename
```

Parameters

node

Pathname the new symbolic-link file wrapper is to reference.

preferredFilename

Preferred filename for the new symbolic-link file wrapper. A `nil` or empty value raises `NSInvalidArgumentException`.

Return Value

Dictionary key used to store the new file wrapper in the receiver's list of file wrappers. See [Working With Directory Wrappers](#) for more information.

Discussion

This method raises `NSInternalInconsistencyException` when the receiver is not a directory file wrapper.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [addFilePath:](#) (page 8)
- [addFileWrapper:](#) (page 9)
- [addRegularFileWithContents:preferredFilename:](#) (page 9)
- [removeFileWrapper:](#) (page 18)
- [fileWrappers](#) (page 11)

Declared In

NSFileWrapper.h

fileAttributes

Provides the receiver's file attributes.

- (NSDictionary *)fileAttributes

Discussion

See the `NSFileManager` class for information on the contents of the `attributes` dictionary.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setFileAttributes:](#) (page 19)

Declared In

NSFileWrapper.h

filename

Provides the receiver's filename.

- (NSString *)filename

Return Value

The receiver's filename; `nil` when the receiver has no corresponding node.

Discussion

The filename is used for record-keeping purposes only and is set automatically when the file wrapper is created from the filesystem using [initWithPath:](#) (page 14) and when it's saved to the filesystem using [writeToFile:atomically:updateFileNames:](#) (page 22) (although this method allows you to request that the filename not be updated).

Availability

Available in Mac OS X v10.0 and later.

See Also

- [preferredFilename](#) (page 18)

- [setFilename:](#) (page 20)

Related Sample Code

File Wrappers with Core Data Documents

Quartz Composer WWDC 2005 TextEdit

TextEditPlus

Declared In

NSFileWrapper.h

fileWrappers

Provides the file wrappers contained by the receiver, which must be a directory file wrapper.

- (NSDictionary *)fileWrappers

Return Value

Keyed list of file wrappers. See [Working With Directory Wrappers](#) for more information.

Discussion

This method raises `NSInternalInconsistencyException` when the receiver is not a directory file wrapper.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [filename](#) (page 11)
- [addFileWrapper:](#) (page 9)

Related Sample Code

File Wrappers with Core Data Documents

Declared In

`NSFileWrapper.h`

icon

Provides an image that represents the receiver to the user.

- (NSImage *)icon

Return Value

Image that represents the receiver; `nil` when the receiver has no icon.

Discussion

You don't have to use this image; for example, a file viewer typically looks up icons automatically based on file extensions, and so wouldn't need this image. Similarly, if a file wrapper represents an image file, you can display the image directly rather than a file icon.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setIcon:](#) (page 20)

Related Sample Code

JDragNDrop

Declared In

`NSFileWrapper.h`

initWithFileWrappers:

Initializes the receiver as a directory file wrapper, with a given file-wrapper list.

- (id)initWithFileWrappers:(NSDictionary *)fileWrappers

Parameters*fileWrappers*

Keyed list of file wrappers with which to initialize the receiver. See [Working With Directory Wrappers](#) for details about the file-wrapper list structure.

Return Value

Initialized file wrapper for *fileWrappers*.

Discussion

After initialization, the receiver is not associated to a filesystem node until you save it using [writeToFile:atomically:updateFileNames:](#) (page 22). It's also initialized with open permissions; anyone can read, write, or change directory to the disk representations that it saves.

If any file wrapper in *fileWrappers* doesn't have a preferred name, its preferred name is automatically set to its corresponding dictionary key in *fileWrappers*.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setPreferredFilename:](#) (page 20)
- [filename](#) (page 11)
- [setFileAttributes:](#) (page 19)

Related Sample Code

StickiesExample

Declared In

NSFileWrapper.h

initWithRegularFileWithContents:

Initializes the receiver as a regular-file file wrapper.

```
- (id)initWithRegularFileWithContents:(NSData *)regularFileContents
```

Parameters*regularFileContents*

Contents for the receiver.

Return Value

Initialized regular-file file wrapper containing *regularFileContents*.

Discussion

After initialization, the receiver is not associated to a filesystem node until you save it using [writeToFile:atomically:updateFileNames:](#) (page 22). It's also initialized with open permissions; anyone can read or write the disk representations that it saves.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setPreferredFilename:](#) (page 20)
- [filename](#) (page 11)

- [fileAttributes](#) (page 11)

Declared In

NSFileWrapper.h

initWithSymbolicLinkWithDestination:

Initializes the receiver as a symbolic-link file wrapper.

```
- (id)initWithSymbolicLinkWithDestination:(NSString *)node
```

Parameters

node

Pathname the receiver is to represent.

Return Value

Initialized symbolic-link file wrapper referencing *node*.

Discussion

The receiver is not associated to a filesystem node until you save it using [writeToFile:atomically:updateFileNames:](#) (page 22). It's also initialized with open permissions; anyone can read or write the disk representations it saves.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setPreferredFilename:](#) (page 20)
- [filename](#) (page 11)
- [fileAttributes](#) (page 11)

Declared In

NSFileWrapper.h

initWithPath:

Initializes the receiver with a given node.

```
- (id)initWithPath:(NSString *)node
```

Parameters

node

Pathname of the node the receiver is to represent.

Return Value

File wrapper for *node*.

Discussion

If *node* is a directory, this method recursively creates file wrappers for each node within that directory.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setPreferredFilename:](#) (page 20)
- [filename](#) (page 11)
- [fileAttributes](#) (page 11)

Declared In

NSFileWrapper.h

initWithSerializedRepresentation:

Initializes the receiver from given serialized data.

```
- (id)initWithSerializedRepresentation:(NSData *)serializedRepresentation
```

Parameters

serializedRepresentation

Serialized representation of a file wrapper in the format used for the NSFileContentsPboardType pasteboard type. Data of this format is returned by such methods as [serializedRepresentation](#) (page 19) and RTFDFromRange:documentAttributes:(NSAttributedString).

Return Value

File wrapper initialized from *serializedRepresentation*.

Discussion

The receiver is not associated to a filesystem node until you save it using [writeToFile:atomically:updateFileNames:](#) (page 22).

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setPreferredFilename:](#) (page 20)
- [filename](#) (page 11)
- [fileAttributes](#) (page 11)

Declared In

NSFileWrapper.h

isDirectory

Indicates whether the receiver is a directory file wrapper.

```
- (BOOL)isDirectory
```

Return Value

YES when the receiver is a directory file wrapper, NO otherwise.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [isRegularFile](#) (page 16)

- [isSymbolicLink](#) (page 16)

Declared In

NSFileWrapper.h

isRegularFile

Indicates whether the receiver is a regular-file file wrapper.

- (BOOL)isRegularFile

Return Value

YES when the receiver is a regular-file wrapper, NO otherwise.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [isDirectory](#) (page 15)

- [isSymbolicLink](#) (page 16)

Declared In

NSFileWrapper.h

isSymbolicLink

Indicates whether the receiver is a symbolic-link file wrapper.

- (BOOL)isSymbolicLink

Return Value

YES when the receiver is a symbolic-link file wrapper, NO otherwise.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [isDirectory](#) (page 15)

- [isRegularFile](#) (page 16)

Declared In

NSFileWrapper.h

keyForFileWrapper:

Provides a key used by the receiver to identify a given file wrapper. The receiver must be a dictionary file wrapper.

- (NSString *)keyForFileWrapper:(NSFileWrapper *)fileWrapper

Parameters*fileWrapper*

File wrapper in question.

Return ValueKey (not necessarily the filename) that identifies *fileWrapper* within the receiver's list of file wrappers. See [Working With Directory Wrappers](#) for more information.**Discussion**This method raises `NSInternalInconsistencyException` when the receiver is not a directory file wrapper.**Availability**

Available in Mac OS X v10.0 and later.

See Also- [filename](#) (page 11)**Declared In**

NSFileWrapper.h

needsToBeUpdatedFromPath:

Indicates whether the receiver needs to be updated to match a given filesystem node.

- (BOOL)needsToBeUpdatedFromPath:(NSString *)node

Parameters*node*

Filesystem node with which to compare the receiver.

Return ValueYES when the receiver needs to be updated to match *node*, NO otherwise.**Discussion**This table describes which attributes of the receiver and *node* are compared to determine whether the receiver needs to be updated:

File-wrapper type	Comparison determinants
Regular file	Modification date and access permissions.
Directory	Member hierarchy (recursive).
Symbolic link	Destination pathname.

Availability

Available in Mac OS X v10.0 and later.

See Also- [updateFromPath:](#) (page 21)- [fileAttributes](#) (page 11)**Declared In**

NSFileWrapper.h

preferredFilename

Provides the receiver's preferred filename.

- (NSString *)preferredFilename

Return Value

The receiver's preferred filename.

Discussion

This name is used as the key when a file wrapper is added to a directory wrapper. However, if another file wrapper with the same preferred name already exists in the directory file wrapper when the receiver is added, the dictionary key and filename assigned may differ from the preferred filename.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [filename](#) (page 11)
- [setPreferredFilename:](#) (page 20)

Declared In

NSFileWrapper.h

regularFileContents

Provides the contents of the receiver's filesystem node. The receiver must be a regular-file file wrapper.

- (NSData *)regularFileContents

Return Value

Contents of the filesystem node the receiver represents.

Discussion

This method raises `NSInternalInconsistencyException` when the receiver is not a regular-file file wrapper.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

File Wrappers with Core Data Documents
StickiesExample

Declared In

NSFileWrapper.h

removeFileWrapper:

Removes a file wrapper from the receiver, which must be a directory file wrapper.

- (void)removeFileWrapper:(NSFileWrapper *)fileWrapper

Parameters*fileWrapper*

File wrapper to remove from the receiver.

DiscussionThis method raises `NSInternalInconsistencyException` when the receiver is not a directory file wrapper.**Availability**

Available in Mac OS X v10.0 and later.

See Also

- [addFilePath:](#) (page 8)
- [addFileWrapper:](#) (page 9)
- [addRegularFileWithContents:preferredFilename:](#) (page 9)
- [addSymbolicLinkWithDestination:preferredFilename:](#) (page 10)
- [fileWrappers](#) (page 11)

Related Sample Code

File Wrappers with Core Data Documents

Declared In

NSFileWrapper.h

serializedRepresentation

Provides the receiver's contents as an opaque collection of data.

- (NSData *)serializedRepresentation

Return ValueThe receiver's contents in the format used for the pasteboard type `NSFileContentsPboardType`.**Availability**

Available in Mac OS X v10.0 and later.

See Also

- [initWithSerializedRepresentation:](#) (page 15)

Declared In

NSFileWrapper.h

setFileAttributes:

Specifies the receiver's file attributes.

- (void)setFileAttributes:(NSDictionary *)fileAttributes

Parameters*fileAttributes*

File attributes for the receiver.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [fileAttributes](#) (page 11)
- [writeToFile:atomically:updateFileNames:](#) (page 22)

Declared In

NSFileWrapper.h

setFilename:

Specifies the receiver's filename.

```
- (void)setFilename:(NSString *)filename
```

Parameters*filename*Filename for the receiver. A nil or empty value raises `NSInvalidArgumentException`.**Availability**

Available in Mac OS X v10.0 and later.

See Also

- [filename](#) (page 11)
- [setPreferredFilename:](#) (page 20)

Declared In

NSFileWrapper.h

setIcon:

Specifies the image to be used to represent the receiver to the user.

```
- (void)setIcon:(NSImage *)icon
```

Parameters*icon*

Image that is to represent the receiver to the user.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [icon](#) (page 12)

Related Sample Code

CoreRecipes

Declared In

NSFileWrapper.h

setPreferredFilename:

Specifies the receiver's preferred filename.

- (void)setPreferredFilename:(NSString *)preferredFilename

Parameters

preferredFilename

Preferred filename for the receiver. A nil or empty value raises `NSInvalidArgumentException`.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [preferredFilename](#) (page 18)
- [setFilename:](#) (page 20)
- [addFileWrapper:](#) (page 9)

Related Sample Code

File Wrappers with Core Data Documents

Declared In

NSFileWrapper.h

symbolicLinkDestination

Provides the pathname referenced by the receiver, which must be a symbolic-link file wrapper.

- (NSString *)symbolicLinkDestination

Return Value

Pathname the receiver references (the destination of the symbolic link the receiver represents).

Discussion

This method raises `NSInternalInconsistencyException` when the receiver is not a symbolic-link file wrapper.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSFileWrapper.h

updateFromPath:

Updates the receiver to match a given filesystem node.

- (BOOL)updateFromPath:(NSString *)path

Return Value

YES if the update is carried out, NO if it isn't needed.

Discussion

For a directory file wrapper, the contained file wrappers are also sent `updateFromPath:` messages. If nodes in the corresponding directory on the filesystem have been added or removed, corresponding file wrappers are released or created as needed.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [needsToBeUpdatedFromPath:](#) (page 17)
- [updateAttachmentsFromPath:](#) (NSAttributedString)

Declared In

NSFileWrapper.h

writeToFile:atomically:updateFileNames:

Writes the receiver's contents to a given filesystem node.

```
- (BOOL)writeToFile:(NSString *)node atomically:(BOOL)atomically
  updateFileNames:(BOOL)updateNames
```

Parameters

node

Pathname of the filesystem node to which the receiver's contents are written.

atomically

YES to write the file safely so that:

- An existing file is not overwritten
- The method fails if the file cannot be written in its entirety

NO to overwrite an existing file and ignore incomplete writes.

updateNames

YES to update the receiver's filenames (its filename and—for directory file wrappers—the filenames of its sub-file wrappers) be changed to the filenames of the corresponding nodes in the filesystem, after a successful write operation. Use this in Save or Save As operations.

NO to specify that the receiver's filenames not be updated. Use this in Save To operations.

Return Value

YES when the write operation is successful, NO otherwise.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [filename](#) (page 11)

Related Sample Code

File Wrappers with Core Data Documents

Quartz Composer WWDC 2005 TextEdit

TextEditPlus

Declared In

NSFileWrapper.h

Document Revision History

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

Date	Notes
2008-10-15	Corrected -initWithSerializedRepresentation: method description.
2007-03-27	Made editorial improvements.
2006-05-23	First publication of this content as a separate document.

REVISION HISTORY

Document Revision History

Index

A

addFileWithPath: **instance method** [8](#)
addFileWrapper: **instance method** [9](#)
addRegularFileWithContents:preferredFilename:
instance method [9](#)
addSymbolicLinkWithDestination:preferredFilename:
instance method [10](#)

F

fileAttributes **instance method** [11](#)
filename **instance method** [11](#)
fileWrappers **instance method** [11](#)

I

icon **instance method** [12](#)
initDirectoryWithFileWrappers: **instance method**
[12](#)
initRegularFileWithContents: **instance method** [13](#)
initSymbolicLinkWithDestination: **instance**
method [14](#)
initWithPath: **instance method** [14](#)
initWithSerializedRepresentation: **instance**
method [15](#)
isDirectory **instance method** [15](#)
isRegularFile **instance method** [16](#)
isSymbolicLink **instance method** [16](#)

K

keyForFileWrapper: **instance method** [16](#)

N

needsToBeUpdatedFromPath: **instance method** [17](#)

P

preferredFilename **instance method** [18](#)

R

regularFileContents **instance method** [18](#)
removeFileWrapper: **instance method** [18](#)

S

serializedRepresentation **instance method** [19](#)
setFileAttributes: **instance method** [19](#)
setFilename: **instance method** [20](#)
setIcon: **instance method** [20](#)
setPreferredFilename: **instance method** [20](#)
symbolicLinkDestination **instance method** [21](#)

U

updateFromPath: **instance method** [21](#)

W

writeToFile:atomically:updateFileNames:
instance method [22](#)