NSDocumentController Class Reference

Cocoa > Design Guidelines

2008-10-15

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

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

NSDocumentController Class Reference 5

Overview 5
Adopted Protocols 5
Tasks 6
Obtaining the Shared Document Controller 6
Initializing a New NSDocumentController 6
Creating and Opening Documents 6
Managing Documents 6
Managing Document Types 7
Autosaving 7
Closing Documents 7
Responding to Action Messages 8
Managing the Open Panel 8
Managing the Open Recent Menu 8
Validating User Interface Items 9
Handling Errors 9
Deprecated Methods 9
Class Methods 10
sharedDocumentController 10
Instance Methods 11
addDocument: 11
autosavingDelay 11
clearRecentDocuments: 11
closeAllDocumentsWithDelegate:didCloseAllSelector:contextInfo: 12
currentDirectory 12
currentDocument 13
defaultType 13
displayNameForType: 14
documentClassForType: 14
documentClassNames 15
documentForURL: 15
documentForWindow: 16
documents 16
hasEditedDocuments 17
init 17
makeDocumentForURL:withContentsOfURL:ofType:error: 17
makeDocumentWithContentsOfURL:ofType:error: 18
makeUntitledDocumentOfType:error: 18
maximumRecentDocumentCount 19
newDocument: 19
noteNewRecentDocument: 20

noteNewRecentDocumentURL: 20
openDocument: 20
openDocumentWithContentsOfURL:display:error: 21
openUntitledDocumentAndDisplay:error: 21
presentError: 22
presentError:modalForWindow:delegate:didPresentSelector:contextInfo: 23
recentDocumentURLs 23
removeDocument: 23
reopenDocumentForURL:withContentsOfURL:error: 24
reviewUnsavedDocumentsWithAlertTitle:cancellable:delegate:didReviewAllSelector: contextInfo: 24
runModalOpenPanel:forTypes: 25
saveAllDocuments: 25
setAutosavingDelay: 25
typeForContentsOfURL:error: 26
URLsFromRunningOpenPanel 26
validateUserInterfaceItem: 27
willPresentError: 27

Appendix A Deprecated NSDocumentController Methods 29

Deprecated in Mac OS X v10.4 29
documentForFileName: 29
fileNamesFromRunningOpenPanel 29
makeDocumentWithContentsOfFile:ofType: 30
makeDocumentWithContentsOfURL:ofType: 30
makeUntitledDocumentOfType: 31
openDocumentWithContentsOfFile:display: 31
openDocumentWithContentsOfURL:display: 32
openUntitledDocumentOfType:display: 33
setShouldCreateUI: 33
shouldCreateUI 34
Deprecated in Mac OS X v10.5 34
fileExtensionsFromType: 34
typeFromFileExtension: 35

Document Revision History 37

Index 39

NSDocumentController Class Reference

Inherits from	NSObject
Conforms to	NSUserInterfaceValidations NSCoding NSObject (NSObject)
Framework	/System/Library/Frameworks/AppKit.framework
Availability	Available in Mac OS X v10.0 and later.
Companion guide	Document-Based Applications Overview
Declared in	NSDocumentController.h
Related sample code	iSpend QTAudioExtractionPanel QTKitPlayer Quartz Composer WWDC 2005 TextEdit TextEditPlus

Overview

An `NSDocumentController` object manages an application's documents. As the first-responder target of New and Open menu commands, it creates and opens documents and tracks them throughout a session of the application. When opening documents, an `NSDocumentController` runs and manages the modal Open panel. `NSDocumentController` objects also maintain and manage the mappings of document types, extensions, and `NSDocument` subclasses as specified in the `CFBundleDocumentTypes` property loaded from the information property list (`Info.plist`).

You can use various `NSDocumentController` methods to get a list of the current documents; get the current document (which is the document whose window is currently key); get documents based on a given filename or window; and find out about a document's extension, type, display name, and document class.

In some situations, it is worthwhile to subclass `NSDocumentController` in non-`NSDocument`-based applications to get some of its features. For example, the `NSDocumentController` management of the Open Recent menu is useful in applications that don't use subclasses of `NSDocument`.

Adopted Protocols

NSCoding

`encodeWithCoder:`
`initWithCoder:`

Tasks

Obtaining the Shared Document Controller

`+ sharedDocumentController` (page 10)
Returns the shared `NSDocumentController` instance.

Initializing a New NSDocumentController

`- init` (page 17)
This method is the designated initializer for `NSDocumentController`.

Creating and Opening Documents

`- documentForURL:` (page 15)
Returns, for a given URL, the open document whose file or file package is located by the URL, or `nil` if there is no such open document.

`- openUntitledDocumentAndDisplay:error:` (page 21)
Creates a new untitled document, presents its user interface if *`displayDocument`* is `YES`, and returns the document if successful.

`- makeUntitledDocumentOfType:error:` (page 18)
Instantiates a new untitled document of the specified type and returns it if successful.

`- openDocumentWithContentsOfURL:display:error:` (page 21)
Opens a document located by the given URL presents its user interface if requested, and returns the document if successful.

`- makeDocumentWithContentsOfURL:ofType:error:` (page 18)
Instantiates a document located by a URL, of a specified type, and returns it if successful.

`- reopenDocumentForURL:withContentsOfURL:error:` (page 24)
Reopens an autosaved document located by a URL, by reading the contents for the document from another URL, presents its user interface, and returns `YES` if successful.

`- makeDocumentForURL:withContentsOfURL:ofType:error:` (page 17)
Instantiates a document located by a URL, of a specified type, but by reading the contents for the document from another URL, and returns it if successful.

Managing Documents

`- documents` (page 16)
Returns the `NSDocument` objects managed by the receiver.

- `addDocument:` (page 11)

  Adds the given document to the list of open documents.

- `currentDocument` (page 13)

  Returns the `NSDocument` object associated with the main window.

- `documentForWindow:` (page 16)

  Returns the document object whose window controller owns a specified window.

- `hasEditedDocuments` (page 17)

  Returns a Boolean value that indicates whether the receiver has any documents with unsaved changes.

- `removeDocument:` (page 23)

  Removes the given document from the list of open documents.

Managing Document Types

- `documentClassNames` (page 15)

  Returns the names of `NSDocument` subclasses supported by this application.

- `defaultType` (page 13)

  Returns the name of the document type that should be used when creating new documents.

- `documentClassForType:` (page 14)

  Returns the `NSDocument` subclass associated with a given document type.

- `displayNameForType:` (page 14)

  Returns the descriptive name for the specified document type, which is used in the File Format pop-up menu of the Save As dialog.

- `typeForContentsOfURL:error:` (page 26)

  Returns, for a specified URL, the name of the document type that should be used when opening the document at that location, if successful.

- `fileExtensionsFromType:` (page 34) Deprecated in Mac OS X v10.5

  Returns the allowable file extensions for the given document type.

- `typeFromFileExtension:` (page 35) Deprecated in Mac OS X v10.5

  Returns the document type associated with files having extension *`fileExtensionOrHFSFileType`*.

Autosaving

- `autosavingDelay` (page 11)

  Returns the time interval in seconds for periodic autosaving.

- `setAutosavingDelay:` (page 25)

  Sets the time interval in seconds for periodic autosaving.

Closing Documents

- `closeAllDocumentsWithDelegate:didCloseAllSelector:contextInfo:` (page 12)

  Iterates through all the open documents and tries to close them one by one using the specified delegate.

- reviewUnsavedDocumentsWithAlertTitle:cancellable:delegate:didReviewAllSelector:contextInfo: (page 24)
  Displays an alert dialog asking if the user wants to review unsaved documents (only if there are two or more unsaved documents), quit regardless of unsaved documents, or (if the choice is allowed) cancel the impending save operation.

Responding to Action Messages

- newDocument: (page 19)
  An action method invoked by the New menu command, this method creates a new `NSDocument` object and adds it to the list of such objects managed by the receiver.
- openDocument: (page 20)
  An action method invoked by the Open menu command, it runs the modal Open panel and, based on the selected filenames, creates one or more `NSDocument` objects from the contents of the files.
- saveAllDocuments: (page 25)
  As the action method invoked by the Save All command, saves all open documents of the application that need to be saved.

Managing the Open Panel

- runModalOpenPanel:forTypes: (page 25)
  Invokes the `NSOpenPanel` `runModalForTypes:` method, passing the *`openPanel`* object and the file *`extensions`* associated with a document type.
- currentDirectory (page 12)
  Returns the directory path to be used as the starting point in the Open panel.
- URLsFromRunningOpenPanel (page 26)
  Creates an `NSOpenPanel` instance and initializes it appropriately.

Managing the Open Recent Menu

- maximumRecentDocumentCount (page 19)
  Returns the maximum number of items that may be presented in the standard Open Recent menu.
- clearRecentDocuments: (page 11)
  Empties the recent documents list for the application.
- noteNewRecentDocumentURL: (page 20)
  This method should be called by applications not based on `NSDocument` when they open or save documents identified by the given URL.
- noteNewRecentDocument: (page 20)
  This method is called by `NSDocument` objects at appropriate times for managing the recent-documents list.
- recentDocumentURLs (page 23)
  Returns the list of recent-document URLs.

Validating User Interface Items

- `validateUserInterfaceItem:` (page 27)
  Returns a Boolean value that indicates whether a given user interface item should be enabled.

Handling Errors

- `presentError:` (page 22)
  Presents an error alert to the user as a modal panel.
- `presentError:modalForWindow:delegate:didPresentSelector:contextInfo:` (page 23)
  Presents an error alert to the user as a modal panel.
- `willPresentError:` (page 27)
  Called when the receiver is about to present an error, returns the error that should actually be presented.

Deprecated Methods

- `documentForFileName:` (page 29) Deprecated in Mac OS X v10.4
  Returns the document object for the file in which the document data is stored. (Deprecated. Use `documentForURL:` (page 15) instead.)
- `fileNamesFromRunningOpenPanel` (page 29) Deprecated in Mac OS X v10.4
  Returns a selection of files chosen by the user in the Open panel. (Deprecated. Use `URLsFromRunningOpenPanel` (page 26) instead.)
- `makeDocumentWithContentsOfFile:ofType:` (page 30) Deprecated in Mac OS X v10.4
  Creates and returns a document object of a given document type from the contents of a file. (Deprecated. Use `makeDocumentWithContentsOfURL:ofType:error:` (page 18) instead.)
- `makeDocumentWithContentsOfURL:ofType:` (page 30) Deprecated in Mac OS X v10.4
  Creates and returns a ocument object for the given document type from the contents of a given URL. (Deprecated. Use `makeDocumentWithContentsOfURL:ofType:error:` (page 18) instead.)
- `makeUntitledDocumentOfType:` (page 31) Deprecated in Mac OS X v10.4
  Creates and returns a document object for document type. (Deprecated. Use `makeUntitledDocumentOfType:error:` (page 18) instead.)
- `openDocumentWithContentsOfFile:display:` (page 31) Deprecated in Mac OS X v10.4
  Returns a document object created from the contents of a given file and optionally displays it. (Deprecated. Use `openDocumentWithContentsOfURL:display:error:` (page 21) instead.)
- `openDocumentWithContentsOfURL:display:` (page 32) Deprecated in Mac OS X v10.4
  Returns a document object created from the contents of a given URL and optionally displays it. (Deprecated. Use `openDocumentWithContentsOfURL:display:error:` (page 21) instead.)
- `openUntitledDocumentOfType:display:` (page 33) Deprecated in Mac OS X v10.4
  Returns a document object instantiated from the subclass of the given document type and optionally displays it. (Deprecated. Use `openUntitledDocumentAndDisplay:error:` (page 21) with `defaultType` (page 13) instead.)

- `setShouldCreateUI:` (page 33) Deprecated in Mac OS X v10.4
  Sets whether the window controllers of a document should be created when the document is created. (Deprecated. Use the display parameter of `openUntitledDocumentAndDisplay:error:` (page 21) or `openDocumentWithContentsOfURL:display:error:` (page 21) instead.)

- `shouldCreateUI` (page 34) Deprecated in Mac OS X v10.4
  Returns a Boolean value that indicates whether the window controllers of a document should be created when the document is created. (Deprecated. Use the *`display`* parameter of `openUntitledDocumentAndDisplay:error:` (page 21) or `openDocumentWithContentsOfURL:display:error:` (page 21) instead.)

Class Methods

sharedDocumentController

Returns the shared `NSDocumentController` instance.

```
+ (id)sharedDocumentController
```

Return Value
The shared `NSDocumentController` instance.

Discussion
If an `NSDocumentController` instance doesn't exist yet, it is created.

Initialization reads in the document types from the `CFBundleDocumentTypes` property list (in `Info.plist`), registers the instance for `NSWorkspaceWillPowerOffNotification`s, and turns on the flag indicating that document user interfaces should be visible. You should always obtain your application's `NSDocumentController` using this method.

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

See Also
- `setShouldCreateUI:` (page 33)

Related Sample Code
iSpend
QTAudioExtractionPanel
QTKitPlayer
Quartz Composer WWDC 2005 TextEdit
TextEditPlus

Declared In
`NSDocumentController.h`

Instance Methods

addDocument:

Adds the given document to the list of open documents.

```
- (void)addDocument:(NSDocument *)document
```

Discussion

The `open...` methods automatically call `addDocument:`. This method is mostly provided for subclasses that want to know when documents arrive.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

QTAudioExtractionPanel

QTKitAdvancedDocument

QTKitImport

QTKitPlayer

QTMetadataEditor

Declared In

`NSDocumentController.h`

autosavingDelay

Returns the time interval in seconds for periodic autosaving.

```
- (NSTimeInterval)autosavingDelay
```

Discussion

A value of 0 indicates that periodic autosaving should not be done at all. `NSDocumentController` uses this number as the amount of time to wait between detecting that a document has unautosaved changes and sending the document an `autosaveDocumentWithDelegate:didAutosaveSelector:contextInfo:` message. The default value is 0.

Availability

Available in Mac OS X v10.4 and later.

See Also

`– setAutosavingDelay:` (page 25)

Declared In

`NSDocumentController.h`

clearRecentDocuments:

Empties the recent documents list for the application.

```
- (IBAction)clearRecentDocuments:(id)sender
```

Discussion
This is the action for the Clear menu command, but it can be invoked directly if necessary.

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

Declared In
`NSDocumentController.h`

closeAllDocumentsWithDelegate:didCloseAllSelector:contextInfo:

Iterates through all the open documents and tries to close them one by one using the specified delegate.

```
- (void)closeAllDocumentsWithDelegate:(id)delegate
    didCloseAllSelector:(SEL)didCloseAllSelector contextInfo:(void *)contextInfo
```

Discussion
Each `NSDocument` object is sent `canCloseDocumentWithDelegate:shouldCloseSelector:contextInfo:`, which, if the document is dirty, gives it a chance to refuse to close or to save itself first. This method may ask whether to save or to perform a save.

The *didCloseAllSelector* callback method is invoked with `YES` if all documents are closed, and `NO` otherwise. Pass the *contextInfo* object with the callback. The *didCloseAllSelector* callback method should have the following signature:

```
- (void)documentController:(NSDocumentController *)docController  didCloseAll:
 (BOOL)didCloseAll contextInfo:(void *)contextInfo
```

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

Declared In
`NSDocumentController.h`

currentDirectory

Returns the directory path to be used as the starting point in the Open panel.

```
- (NSString *)currentDirectory
```

Discussion
The first valid directory from the following list is returned:

- The directory location where the current document was last saved
- The last directory visited in the Open panel
- The user's home directory

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

See Also

- documentForFileName: (page 29)

Declared In

NSDocumentController.h

currentDocument

Returns the NSDocument object associated with the main window.

```
- (id)currentDocument
```

Discussion

This method returns nil if it is called when its application is not active. This can occur during processing of a drag-and-drop operation, for example, in an implementation of readSelectionFromPasteboard:. In such a case, send the following message instead from an NSView subclass associated with the document:

```
[[[self window] windowController] document];
```

Availability

Available in Mac OS X v10.0 and later.

See Also

- documentForFileName: (page 29)
- documentForWindow: (page 16)
- documents (page 16)

Related Sample Code

QTAudioExtractionPanel

QTKitPlayer

Declared In

NSDocumentController.h

defaultType

Returns the name of the document type that should be used when creating new documents.

```
- (NSString *)defaultType
```

Discussion

The default implementation of this method returns the first Editor type declared by the CFBundleDocumentTypes array in the application's Info.plist, or returns nil if no Editor type is declared. You can override it to customize the type of document that is created when, for instance, the user chooses New in the File menu.

Availability

Available in Mac OS X v10.4 and later.

Declared In

NSDocumentController.h

displayNameForType:

Returns the descriptive name for the specified document type, which is used in the File Format pop-up menu of the Save As dialog.

```
- (NSString *)displayNameForType:(NSString *)documentTypeName
```

Parameters

documentTypeName

The name of a document type, specified by `CFBundleTypeName` in the application's `Info.plist` file.

Return Value

The descriptive name for the document type specified by *documentTypeName*. If there is no descriptive name, returns *documentTypeName*.

Discussion

For a document-based application, supported document types are specified in the `Info.plist` file by the `CFBundleDocumentTypes` array. Each document type is specified by a dictionary in this array, and is named by the `CFBundleTypeName` attribute. You can provide a descriptive, localized, representation of this name by providing a corresponding entry in the `InfoPlist.strings` file(s). For example, given an `Info.plist` file that contains the following fragment:

```
<dict>
    <key>CFBundleDocumentTypes</key>
    <array>
        <dict>
            <key>CFBundleTypeName</key>
            <string>BinaryFile</string>
            <key>CFBundleTypeExtensions</key>
            <array>
                <string>binary</string>
            </array>
```

you could provide a descriptive name by adding an entry in the `InfoPlist.strings` file:

```
BinaryFile = "Binary file format";
```

Availability

Available in Mac OS X v10.0 and later.

See Also

- `- fileExtensionsFromType:` (page 34)
- `- typeFromFileExtension:` (page 35)

Declared In

`NSDocumentController.h`

documentClassForType:

Returns the `NSDocument` subclass associated with a given document type.

```
- (Class)documentClassForType:(NSString *)documentTypeName
```

Parameters

documentTypeName

The name of a document type, specified by `CFBundleTypeName` in the application's `Info.plist` file.

The document type must be one the receiver can read.

Return Value

Returns the `NSDocument` subclass associated with *documentTypeName*. If the class cannot be found, returns `nil`.

Discussion

Para

Availability

Available in Mac OS X v10.0 and later.

See Also

`- displayNameForType:` (page 14)

`- fileExtensionsFromType:` (page 34)

`- typeFromFileExtension:` (page 35)

Declared In

`NSDocumentController.h`

documentClassNames

Returns the names of `NSDocument` subclasses supported by this application.

```
- (NSArray *)documentClassNames
```

Return Value

The names of `NSDocument` subclasses supported by this application.

Discussion

The default implementation of this method returns information derived from the application's `Info.plist` property list file. You can override it to return the names of document classes that are dynamically loaded from plugins.

Availability

Available in Mac OS X v10.4 and later.

Declared In

`NSDocumentController.h`

documentForURL:

Returns, for a given URL, the open document whose file or file package is located by the URL, or `nil` if there is no such open document.

```
- (id)documentForURL:(NSURL *)absoluteURL
```

Discussion

The default implementation of this method queries each open document to find one whose URL matches, and returns the first one whose URL does match.

For backward binary compatibility with Mac OS X v10.3 and earlier, the default implementation of this method instead invokes `documentForFileName:` (page 29) if it is overridden and the URL uses the `file:` scheme.

Availability

Available in Mac OS X v10.4 and later.

Related Sample Code

QTKitCreateMovie

QTKitFrameStepper

Declared In

`NSDocumentController.h`

documentForWindow:

Returns the document object whose window controller owns a specified window.

```
- (id)documentForWindow:(NSWindow *)window
```

Return Value

The document object whose window controller owns *window*. Returns `nil` if *window* is `nil`, if *window* has no window controller, or if the window controller does not have an association with an instance of `NSDocument`.

Availability

Available in Mac OS X v10.0 and later.

See Also

- `currentDocument` (page 13)
- `documentForFileName:` (page 29)
- `documents` (page 16)

Declared In

`NSDocumentController.h`

documents

Returns the `NSDocument` objects managed by the receiver.

```
- (NSArray *)documents
```

Return Value

The `NSDocument` objects managed by the receiver. If there are currently no documents, returns an empty `NSArray` object.

Availability

Available in Mac OS X v10.0 and later.

See Also
- `currentDocument` (page 13)
- `documentForFileName:` (page 29)
- `documentForWindow:` (page 16)

Related Sample Code
QTAudioExtractionPanel

Declared In
`NSDocumentController.h`

hasEditedDocuments

Returns a Boolean value that indicates whether the receiver has any documents with unsaved changes.

```
- (BOOL)hasEditedDocuments
```

Return Value
`YES` if the receiver has any documents with unsaved changes, otherwise `NO`.

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

See Also
- `documents` (page 16)

Declared In
`NSDocumentController.h`

init

This method is the designated initializer for `NSDocumentController`.

```
- (id)init
```

Discussion
The first instance of `NSDocumentController` or any of its subclasses that is created becomes the shared instance.

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

Declared In
`NSDocumentController.h`

makeDocumentForURL:withContentsOfURL:ofType:error:

Instantiates a document located by a URL, of a specified type, but by reading the contents for the document from another URL, and returns it if successful.

```
- (id)makeDocumentForURL:(NSURL *)absoluteDocumentURL withContentsOfURL:(NSURL
    *)absoluteDocumentContentsURL ofType:(NSString *)typeName error:(NSError
    **)outError
```

Discussion

The URL is specified by *absoluteDocumentURL*, the type by *typeName*, and the other URL providing the contents by *absoluteDocumentContentsURL*. If not successful, the method returns `nil` after setting *outError* to point to an `NSError` object that encapsulates the reason why the document could not be instantiated. The default implementation of this method invokes `documentClassForType:` (page 14) to find out the class of document to instantiate, allocates a document object, and initializes it by sending it an `initForURL:withContentsOfURL:ofType:error:` message.

Availability

Available in Mac OS X v10.4 and later.

Declared In

`NSDocumentController.h`

makeDocumentWithContentsOfURL:ofType:error:

Instantiates a document located by a URL, of a specified type, and returns it if successful.

```
- (id)makeDocumentWithContentsOfURL:(NSURL *)absoluteURL ofType:(NSString *)typeName
     error:(NSError **)outError
```

Discussion

The URL is specified by *absoluteURL* and the document type by *typeName*. If not successful, the method returns `nil` after setting *outError* to point to an NSError that encapsulates the reason why the document could not be instantiated. The default implementation of this method invokes `documentClassForType:` (page 14) to find out the class of document to instantiate, allocates a document object, and initializes it by sending it an `initWithContentsOfURL:ofType:error:` message.

For backward binary compatibility with Mac OS X v10.3 and earlier, the default implementation of this method instead invokes `makeDocumentWithContentsOfFile:ofType:` (page 30) if it is overridden and the URL uses the `file:` scheme.

Availability

Available in Mac OS X v10.4 and later.

Related Sample Code

QTKitPlayer

Declared In

`NSDocumentController.h`

makeUntitledDocumentOfType:error:

Instantiates a new untitled document of the specified type and returns it if successful.

```
- (id)makeUntitledDocumentOfType:(NSString *)typeName error:(NSError **)outError
```

Discussion

The document type is specified by *typeName*. If not successful, the method returns nil after setting *outError* to point to an NSError object that encapsulates the reason why a new untitled document could not be instantiated. The default implementation of this method invokes documentClassForType: (page 14) to find out the class of document to instantiate, then allocates and initializes a document by sending it initWithType:error:.

For backward binary compatibility with Mac OS X v10.3 and earlier, the default implementation of this method instead invokes makeUntitledDocumentOfType: (page 31) if it is overridden.

Availability

Available in Mac OS X v10.4 and later.

Declared In

NSDocumentController.h

maximumRecentDocumentCount

Returns the maximum number of items that may be presented in the standard Open Recent menu.

```
- (NSUInteger)maximumRecentDocumentCount
```

Discussion

A value of 0 indicates that NSDocumentController will not attempt to add an Open Recent menu to your application's File menu, although NSDocumentController will not attempt to remove any preexisting Open Recent menu item. The default implementation returns a value that is subject to change and may or may not be derived from a setting made by the user in System Preferences.

Availability

Available in Mac OS X v10.4 and later.

Declared In

NSDocumentController.h

newDocument:

An action method invoked by the New menu command, this method creates a new NSDocument object and adds it to the list of such objects managed by the receiver.

```
- (IBAction)newDocument:(id)sender
```

Discussion

It invokes openUntitledDocumentAndDisplay:error: (page 21) with the document type (first argument) being the first one specified in the CFBundleDocumentTypes property (defined in Info.plist); the document type determines the NSDocument subclass used to instantiate the document object.

Availability

Available in Mac OS X v10.0 and later.

See Also

- openDocument: (page 20)

Declared In
`NSDocumentController.h`

noteNewRecentDocument:

This method is called by `NSDocument` objects at appropriate times for managing the recent-documents list.

```
- (void)noteNewRecentDocument:(NSDocument *)aDocument
```

Discussion
This method constructs a URL and calls `noteNewRecentDocumentURL:` (page 20). Subclasses might override this method to prevent certain documents or kinds of documents from getting into the list.

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

Declared In
`NSDocumentController.h`

noteNewRecentDocumentURL:

This method should be called by applications not based on `NSDocument` when they open or save documents identified by the given URL.

```
- (void)noteNewRecentDocumentURL:(NSURL *)aURL
```

Discussion
`NSDocument` automatically calls this method when appropriate for `NSDocument`-based applications. Applications not based on `NSDocument` must also implement the `application:openFile:` method in the application delegate to handle requests from the Open Recent menu command. You can override this method in an `NSDocument`-based application to prevent certain kinds of documents from getting into the list (but you have to identify them by URL).

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

Related Sample Code
Quartz Composer WWDC 2005 TextEdit

TextEditPlus

Declared In
`NSDocumentController.h`

openDocument:

An action method invoked by the Open menu command, it runs the modal Open panel and, based on the selected filenames, creates one or more `NSDocument` objects from the contents of the files.

```
- (IBAction)openDocument:(id)sender
```

Discussion

The method adds the newly created objects to the list of `NSDocument` objects managed by the receiver. This method invokes `openDocumentWithContentsOfURL:display:error:` (page 21), which actually creates the `NSDocument` objects.

Availability

Available in Mac OS X v10.0 and later.

See Also

- `– fileNamesFromRunningOpenPanel` (page 29)
- `– newDocument:` (page 19)

Declared In

`NSDocumentController.h`

openDocumentWithContentsOfURL:display:error:

Opens a document located by the given URL presents its user interface if requested, and returns the document if successful.

```
- (id)openDocumentWithContentsOfURL:(NSURL *)absoluteURL
    display:(BOOL)displayDocument error:(NSError **)outError
```

Discussion

If not successful, the method returns `nil` after setting *`outError`* to point to an `NSError` object that encapsulates the reason why the document could not be opened.

The default implementation of this method checks to see if the document is already open according to `documentForURL:` (page 15), and if it is not open determines the type of the document, invokes `makeDocumentWithContentsOfURL:ofType:error:` (page 18) to instantiate it, then invokes `addDocument:` (page 11) to record its opening, and sends the document `makeWindowControllers` and `showWindows` messages if *`displayDocument`* is `YES`. If the document is already open it is just sent a `showWindows` message if *`displayDocument`* is `YES`.

For backward binary compatibility with Mac OS X v10.3 and earlier, the default implementation of this method instead invokes `openDocumentWithContentsOfFile:display:` (page 31), if it is overridden and the URL uses the `file:` scheme.

Availability

Available in Mac OS X v10.4 and later.

Declared In

`NSDocumentController.h`

openUntitledDocumentAndDisplay:error:

Creates a new untitled document, presents its user interface if *`displayDocument`* is `YES`, and returns the document if successful.

```
- (id)openUntitledDocumentAndDisplay:(BOOL)displayDocument error:(NSError **)outError
```

Discussion

If not successful, the method returns `nil` after setting *`outError`* to point to an NSError that encapsulates the reason why a new untitled document could not be created.

The default implementation of this method invokes `defaultType` (page 13) to determine the type of new document to create, invokes `makeUntitledDocumentOfType:error:` (page 18) to create it, then invokes `addDocument:` (page 11) to record its opening. If *`displayDocument`* is `YES`, it then sends the new document `makeWindowControllers` and `showWindows` messages.

For backward binary compatibility with Mac OS X v10.3 and earlier, the default implementation of this method instead invokes `openUntitledDocumentOfType:display:` (page 33) if it is overridden.

Availability

Available in Mac OS X v10.4 and later.

Related Sample Code

iSpend

Declared In

`NSDocumentController.h`

presentError:

Presents an error alert to the user as a modal panel.

```
- (BOOL)presentError:(NSError *)error
```

Discussion

Returns `YES` if error recovery was done, `NO` otherwise. This method does not return until the user dismisses the alert and, if the error has recovery options and a recovery delegate, the error's recovery delegate is sent an `attemptRecoveryFromError:optionIndex:` message.

The default `NSDocumentController` implementation of this method is equivalent to that of `NSResponder` while treating the application object as the next responder and forwarding error presentation messages to it. (The default `NSDocument` implementation of this method treats the shared `NSDocumentController` instance as the next responder and forwards these messages to it.) The default implementations of several `NSDocumentController` methods invoke this method.

The default implementation of this method invokes `willPresentError:` (page 27) to give subclasses an opportunity to customize error presentation. You should not override this method but should instead override `willPresentError:` (page 27).

Availability

Available in Mac OS X v10.4 and later.

See Also

- `willPresentError:` (page 27)
- `presentError:modalForWindow:delegate:didPresentSelector:contextInfo:` (page 23)

Declared In

`NSDocumentController.h`

presentError:modalForWindow:delegate:didPresentSelector:contextInfo:

Presents an error alert to the user as a modal panel.

```
- (void)presentError:(NSError *)error modalForWindow:(NSWindow *)window
    delegate:(id)delegate didPresentSelector:(SEL)didPresentSelector
    contextInfo:(void *)contextInfo
```

Discussion

When the user dismisses the alert and any recovery possible for the error and chosen by the user has been attempted, sends the message `didPresentSelector` to the specified `delegate`. The method selected by `didPresentSelector` must have the same signature as:

```
- (void)didPresentErrorWithRecovery:(BOOL)didRecover contextInfo:(void
*)contextInfo;
```

The default `NSDocumentController` implementation of this method is equivalent to that of `NSResponder` while treating the application object as the next responder and forwarding error presentation messages to it. (The default `NSDocument` implementation of this method treats the shared `NSDocumentController` instance as the next responder and forwards these messages to it.)

The default implementation of this method invokes `willPresentError:` (page 27) to give subclasses an opportunity to customize error presentation. You should not override this method but should instead override `willPresentError:` (page 27).

Availability

Available in Mac OS X v10.4 and later.

See Also

`- willPresentError:` (page 27)
`- presentError:` (page 22)

Declared In

`NSDocumentController.h`

recentDocumentURLs

Returns the list of recent-document URLs.

```
- (NSArray *)recentDocumentURLs
```

Discussion

This method is not a good one to override since the internals of `NSDocumentController` do not generally use it.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSDocumentController.h`

removeDocument:

Removes the given document from the list of open documents.

```
- (void)removeDocument:(NSDocument *)document
```

Discussion
A document will automatically call `removeDocument:` (page 23) when it closes. This method is mostly provided for subclasses that want to know when documents close.

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

Related Sample Code
iSpend

Declared In
`NSDocumentController.h`

reopenDocumentForURL:withContentsOfURL:error:

Reopens an autosaved document located by a URL, by reading the contents for the document from another URL, presents its user interface, and returns `YES` if successful.

```
- (BOOL)reopenDocumentForURL:(NSURL *)absoluteDocumentURL withContentsOfURL:(NSURL
    *)absoluteDocumentContentsURL error:(NSError **)outError
```

Discussion
The document is located by *`absoluteDocumentURL`* and the contents are read from *`absoluteDocumentContentsURL`*. If not successful, the method returns `NO` after setting *`outError`* to point to an `NSError` object that encapsulates the reason why the document could not be reopened.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`NSDocumentController.h`

reviewUnsavedDocumentsWithAlertTitle:cancellable:delegate:didReviewAllSelector:contextInfo:

Displays an alert dialog asking if the user wants to review unsaved documents (only if there are two or more unsaved documents), quit regardless of unsaved documents, or (if the choice is allowed) cancel the impending save operation.

```
- (void)reviewUnsavedDocumentsWithAlertTitle:(NSString *)title
    cancellable:(BOOL)cancellable delegate:(id)delegate
    didReviewAllSelector:(SEL)didReviewAllSelector contextInfo:(void *)contextInfo
```

Discussion
Assigns *`delegate`* to the panel. Invokes *`didReviewAllSelector`* with `YES` if quit without saving is chosen or if there are no dirty documents, and `NO` otherwise. If the user selects the "Review Unsaved" option, `closeAllDocumentsWithDelegate:didCloseAllSelector:contextInfo:` (page 12) is invoked. This method is invoked when the user chooses the Quit menu command, and also when the computer power is being turned off. Note that *`title`* is ignored. Pass the *`contextInfo`* object with the callback.

The *`didReviewAllSelector`* callback method should have the following signature:

```
- (void)documentController:(NSDocumentController *)docController  didReviewAll:
 (BOOL)didReviewAll contextInfo:(void *)contextInfo
```

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

Declared In
`NSDocumentController.h`

runModalOpenPanel:forTypes:

Invokes the `NSOpenPanel` `runModalForTypes:` method, passing the *`openPanel`* object and the file *`extensions`* associated with a document type.

```
- (NSInteger)runModalOpenPanel:(NSOpenPanel *)openPanel forTypes:(NSArray
    *)extensions
```

Discussion
This method is invoked by the `fileNamesFromRunningOpenPanel` (page 29) method. *`extensions`* may also contain encoded HFS file types as well as filename extensions.

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

Declared In
`NSDocumentController.h`

saveAllDocuments:

As the action method invoked by the Save All command, saves all open documents of the application that need to be saved.

```
- (IBAction)saveAllDocuments:(id)sender
```

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

See Also
`- saveDocument:` (`NSDocument`)

Declared In
`NSDocumentController.h`

setAutosavingDelay:

Sets the time interval in seconds for periodic autosaving.

```
- (void)setAutosavingDelay:(NSTimeInterval)autosavingDelay
```

Discussion
A value of 0 indicates that periodic autosaving should not be done at all. `NSDocumentController` uses this number as the amount of time to wait between detecting that a document has unautosaved changes and sending the document an `autosaveDocumentWithDelegate:didAutosaveSelector:contextInfo:` message. The default value is 0.

Availability
Available in Mac OS X v10.4 and later.

See Also
`– autosavingDelay` (page 11)

Declared In
`NSDocumentController.h`

typeForContentsOfURL:error:

Returns, for a specified URL, the name of the document type that should be used when opening the document at that location, if successful.

```
- (NSString *)typeForContentsOfURL:(NSURL *)inAbsoluteURL error:(NSError **)outError
```

Discussion
The URL is represented by *`absoluteURL`*. If not successful, the method returns `nil` after setting *`outError`* to point to an `NSError` object that encapsulates the reason why the document type could not be determined, or the fact that the document type is unrecognized.

You can override this method to customize type determination for documents being opened.

Availability
Available in Mac OS X v10.4 and later.

Declared In
`NSDocumentController.h`

URLsFromRunningOpenPanel

Creates an `NSOpenPanel` instance and initializes it appropriately.

```
- (NSArray *)URLsFromRunningOpenPanel
```

Discussion
This method uses `runModalOpenPanel:forTypes:` (page 25) to run the open panel. Returns the chosen files as an array of URLs. Returns `nil` if the user cancels the Open panel or makes no selection.

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

Declared In
`NSDocumentController.h`

validateUserInterfaceItem:

Returns a Boolean value that indicates whether a given user interface item should be enabled.

```
- (BOOL)validateUserInterfaceItem:(id < NSValidatedUserInterfaceItem >)anItem
```

Parameters

anItem

The user interface item to validate. You can send *anItem* the `action` and `tag` messages.

Return Value

`YES` if *anItem* should be enabled, otherwise `NO`.

Discussion

Subclasses can override this method to perform additional validations. Subclasses should call `super` in their implementation for items they don’t handle themselves.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSDocumentController.h`

willPresentError:

Called when the receiver is about to present an error, returns the error that should actually be presented.

```
- (NSError *)willPresentError:(NSError *)error
```

Discussion

The default implementation of this method merely returns the passed-in error. The returned error may simply be forwarded to the application object.

You can override this method to customize the presentation of errors by examining the passed-in error and, for example, returning more specific information. When you override this method always check the `NSError` object's domain and code to discriminate between errors whose presentation you want to customize and those you don't. For errors you don't want to customize, call the superclass implementation, passing the original error.

Availability

Available in Mac OS X v10.4 and later.

See Also

`– presentError:` (page 22)

`– presentError:modalForWindow:delegate:didPresentSelector:contextInfo:` (page 23)

Declared In

`NSDocumentController.h`

Deprecated NSDocumentController Methods

A method identified as deprecated has been superseded and may become unsupported in the future.

Deprecated in Mac OS X v10.4

documentForFileName:

Returns the document object for the file in which the document data is stored. (Deprecated in Mac OS X v10.4. Use `documentForURL:` (page 15) instead.)

```
- (id)documentForFileName:(NSString *)fileName
```

Discussion
The *`fileName`* argument is a fully qualified path in the file system. Returns `nil` if no document can be found.

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

Deprecated in Mac OS X v10.4.

See Also
`– documentForWindow:` (page 16)

`– documents` (page 16)

Related Sample Code
QTAudioExtractionPanel

QTKitAdvancedDocument

QTKitImport

QTKitPlayer

QTMetadataEditor

Declared In
`NSDocumentController.h`

fileNamesFromRunningOpenPanel

Returns a selection of files chosen by the user in the Open panel. (Deprecated in Mac OS X v10.4. Use `URLsFromRunningOpenPanel` (page 26) instead.)

```
- (NSArray *)fileNamesFromRunningOpenPanel
```

Discussion

Each file in the returned `NSArray` is a fully qualified path to the file's location in the file system. This method is invoked by `openDocument:` (page 20), and it invokes `runModalOpenPanel:forTypes:` (page 25) after initializing the Open panel (which includes getting the starting directory with `currentDirectory` (page 12)). Returns `nil` if the user cancels the Open panel or makes no selection.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

Declared In

`NSDocumentController.h`

makeDocumentWithContentsOfFile:ofType:

Creates and returns a document object of a given document type from the contents of a file. (Deprecated in Mac OS X v10.4. Use `makeDocumentWithContentsOfURL:ofType:error:` (page 18) instead.)

```
- (id)makeDocumentWithContentsOfFile:(NSString *)fileName ofType:(NSString *)docType
```

Discussion

Creates and returns an `NSDocument` object for document type *docType* from the contents of the file *fileName*, which must be a fully qualified path. The returned object is not retained. Returns `nil` if the `NSDocument` subclass for *docType* couldn't be determined or if the object couldn't be created. This method invokes the `NSDocument` method `initWithContentsOfFile:ofType:` and is invoked by `openDocumentWithContentsOfFile:display:` (page 31).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

See Also

- `makeUntitledDocumentOfType:` (page 31)
- `openDocument:` (page 20)

Related Sample Code

QTKitImport

Declared In

`NSDocumentController.h`

makeDocumentWithContentsOfURL:ofType:

Creates and returns a ocument object for the given document type from the contents of a given URL. (Deprecated in Mac OS X v10.4. Use `makeDocumentWithContentsOfURL:ofType:error:` (page 18) instead.)

```
- (id)makeDocumentWithContentsOfURL:(NSURL *)aURL ofType:(NSString *)docType
```

Discussion

Creates and returns an `NSDocument` object for document type *`docType`* from the contents of *`aURL`*. The returned object is not retained. Returns `nil` if the `NSDocument` subclass for *`docType`* couldn't be determined or if the object couldn't be created. This method invokes the `NSDocument` method `initWithContentsOfURL:ofType:` and is invoked by `openDocumentWithContentsOfURL:display:` (page 32).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

See Also

`– makeUntitledDocumentOfType:` (page 31)

`– makeUntitledDocumentOfType:error:` (page 18)

`– openDocument:` (page 20)

Related Sample Code

QTAudioExtractionPanel

Declared In

`NSDocumentController.h`

makeUntitledDocumentOfType:

Creates and returns a document object for document type. (Deprecated in Mac OS X v10.4. Use `makeUntitledDocumentOfType:error:` (page 18) instead.)

```
- (id)makeUntitledDocumentOfType:(NSString *)type
```

Discussion

Creates and returns an `NSDocument` object for document type *`type`*. The returned object is not retained. Returns `nil` if the `NSDocument` subclass for *`type`* couldn't be determined or if the object couldn't be created. This method invokes the `NSDocument` `init` method and is invoked by `openUntitledDocumentOfType:display:` (page 33).

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

See Also

`– makeDocumentWithContentsOfFile:ofType:` (page 30)

`– newDocument:` (page 19)

Declared In

`NSDocumentController.h`

openDocumentWithContentsOfFile:display:

Returns a document object created from the contents of a given file and optionally displays it. (Deprecated in Mac OS X v10.4. Use `openDocumentWithContentsOfURL:display:error:` (page 21) instead.)

```
- (id)openDocumentWithContentsOfFile:(NSString *)fileName display:(BOOL)flag
```

Discussion
Returns an `NSDocument` object created from the contents of the file *fileName* (an absolute path) and displays it if *flag* is `YES`. The returned object is not retained, but is added to the receiver's list of managed documents. Returns `nil` if the object could not be created, typically because *fileName* does not point to a valid file or because there is no `NSDocument` subclass for the document type (as indicated by the file extension or HFS file type). Even if *flag* is `YES`, the document is not displayed if `shouldCreateUI` (page 34) returns `NO`. This method invokes `makeDocumentWithContentsOfFile:ofType:` (page 30) to obtain the created NSDocument object. If you override this method, your implementation should be prepared to handle either `YES` or `NO`.

To handle an Open Documents Apple event, the Application Kit's built-in Apple event handling automatically invokes this method with the path to the file to open and a display argument.

Invoked with a display argument of `YES` instead of `NO` when a Print Documents Apple event is handled. This may have been handled differently in versions of Mac OS X prior to version 10.3.

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

Deprecated in Mac OS X v10.4.

See Also
- `openDocument:` (page 20)
- `openUntitledDocumentOfType:display:` (page 33)
- `setShouldCreateUI:` (page 33)

Related Sample Code
ColorMatching

QTAudioExtractionPanel

QTKitImport

QTKitPlayer

QTMetadataEditor

Declared In
`NSDocumentController.h`

openDocumentWithContentsOfURL:display:

Returns a document object created from the contents of a given URL and optionally displays it. (Deprecated in Mac OS X v10.4. Use `openDocumentWithContentsOfURL:display:error:` (page 21) instead.)

```
- (id)openDocumentWithContentsOfURL:(NSURL *)aURL display:(BOOL)flag
```

Discussion
Returns an `NSDocument` object created from the contents of *aURL* and displays it if *flag* is `YES`. The returned object is not retained, but is added to the receiver's list of managed documents. Returns `nil` if the object could not be created, typically because *aURL* does not point to a valid location or because there is no `NSDocument` subclass for the document type. Even if *flag* is `YES`, the document is not displayed if `shouldCreateUI` (page 34) returns `NO`. This method invokes `makeDocumentWithContentsOfURL:ofType:` (page 30) to obtain the created `NSDocument` object.

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

Deprecated in Mac OS X v10.4.

See Also

- `openDocument:` (page 20)
- `openUntitledDocumentOfType:display:` (page 33)
- `setShouldCreateUI:` (page 33)

Declared In

`NSDocumentController.h`

openUntitledDocumentOfType:display:

Returns a document object instantiated from the subclass of the given document type and optionally displays it. (Deprecated in Mac OS X v10.4. Use `openUntitledDocumentAndDisplay:error:` (page 21) with `defaultType` (page 13) instead.)

```
- (id)openUntitledDocumentOfType:(NSString *)docType display:(BOOL)display
```

Discussion

Returns an `NSDocument` object instantiated from the `NSDocument` subclass required by document type *docType* and displays it if *flag* is `YES`. The returned object is not retained, but is added to the receiver's list of managed documents. Returns `nil` if the object could not be created, typically because no `NSDocument` subclass could be found for *docType*. Even if *flag* is `YES`, the document is not displayed if `shouldCreateUI` (page 34) returns `NO`. This method invokes `makeUntitledDocumentOfType:` (page 31) to obtain the created `NSDocument` object.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.4.

See Also

- `newDocument:` (page 19)
- `openDocumentWithContentsOfFile:display:` (page 31)
- `setShouldCreateUI:` (page 33)

Declared In

`NSDocumentController.h`

setShouldCreateUI:

Sets whether the window controllers of a document should be created when the document is created. (Deprecated in Mac OS X v10.4. Use the display parameter of `openUntitledDocumentAndDisplay:error:` (page 21) or `openDocumentWithContentsOfURL:display:error:` (page 21) instead.)

```
- (void)setShouldCreateUI:(BOOL)flag
```

Discussion
Sets whether the window controllers (`NSWindowController` instances) of a document should be created when the document is created. When a window controller is created, it loads the nib file containing the window it manages. Often *`flag`* is set to `NO` for scripting or searching operations involving the document's data.

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

Deprecated in Mac OS X v10.4.

See Also
`- shouldCreateUI` (page 34)

Declared In
`NSDocumentController.h`

shouldCreateUI

Returns a Boolean value that indicates whether the window controllers of a document should be created when the document is created. (Deprecated in Mac OS X v10.4. Use the *`display`* parameter of `openUntitledDocumentAndDisplay:error:` (page 21) or `openDocumentWithContentsOfURL:display:error:` (page 21) instead.)

```
- (BOOL)shouldCreateUI
```

Return Value
A Boolean value that indicates whether the window controllers (`NSWindowController` instances) of a document should be created when the document is created.

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

Deprecated in Mac OS X v10.4.

See Also
`- setShouldCreateUI:` (page 33)

Declared In
`NSDocumentController.h`

Deprecated in Mac OS X v10.5

fileExtensionsFromType:

Returns the allowable file extensions for the given document type. (Deprecated in Mac OS X v10.5.)

```
- (NSArray *)fileExtensionsFromType:(NSString *)documentTypeName
```

Parameters

documentTypeName

The name of a document type, specified by `CFBundleTypeName` in the application's `Info.plist` file.

Return Value

The allowable file extensions (as `NSString` objects) for *documentTypeName*.

Discussion

Type extensions are specified by the `CFBundleTypeExtensions` array for the given type in the `Info.plist` file.

The first string in the returned array is typically the most common extension. The array may also contain encoded HFS file types as will as filename extensions.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

See Also

`- displayNameForType:` (page 14)

`- typeFromFileExtension:` (page 35)

Declared In

`NSDocumentController.h`

typeFromFileExtension:

Returns the document type associated with files having extension *fileExtensionOrHFSFileType*. (Deprecated in Mac OS X v10.5.)

```
- (NSString *)typeFromFileExtension:(NSString *)fileExtensionOrHFSFileType
```

Discussion

fileExtensionOrHFSFileType may also be an encoded HFS file type, as well as a filename extension.

Availability

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

See Also

`- displayNameForType:` (page 14)

`- fileExtensionsFromType:` (page 34)

Declared In

`NSDocumentController.h`

Document Revision History

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

Date	Notes
2008-10-15	Added defaultType to replacement information for deprecated openUntitledDocumentOfType:display: method.
2008-03-11	Removed false statement that reviewUnsavedDocumentsWithAlertTitle:cancellable: delegate:didReviewAllSelector:contextInfo: can ever return NO for cancellable.
2007-10-31	Enhanced the description of the displayNameForType: method.
	Updated for Mac OS X v10.5. Corrected deprecation information.
2006-05-23	Removed reference to deprecated method from newDocument: description.
	First publication of this content as a separate document.

Index

A

addDocument: instance method 11
autosavingDelay instance method 11

C

clearRecentDocuments: instance method 11
closeAllDocumentsWithDelegate:didCloseAllSelector:
contextInfo: instance method 12
currentDirectory instance method 12
currentDocument instance method 13

D

defaultType instance method 13
displayNameForType: instance method 14
documentClassForType: instance method 14
documentClassNames instance method 15
documentForFileName: instance method 29
documentForURL: instance method 15
documentForWindow: instance method 16
documents instance method 16

F

fileExtensionsFromType: instance method 34
fileNamesFromRunningOpenPanel instance method 29

H

hasEditedDocuments instance method 17

I

init instance method 17

M

makeDocumentForURL:withContentsOfURL:ofType:error: instance method 17
makeDocumentWithContentsOfFile:ofType: instance method 30
makeDocumentWithContentsOfURL:ofType: instance method 30
makeDocumentWithContentsOfURL:ofType:error: instance method 18
makeUntitledDocumentOfType: instance method 31
makeUntitledDocumentOfType:error: instance method 18
maximumRecentDocumentCount instance method 19

N

newDocument: instance method 19
noteNewRecentDocument: instance method 20
noteNewRecentDocumentURL: instance method 20

O

openDocument: instance method 20
openDocumentWithContentsOfFile:display: instance method 31
openDocumentWithContentsOfURL:display: instance method 32
openDocumentWithContentsOfURL:display:error: instance method 21
openUntitledDocumentAndDisplay:error: instance method 21

openUntitledDocumentOfType:display: instance method 33

P

presentError: instance method 22
presentError:modalForWindow:delegate:didPresentSelector:contextInfo: instance method 23

R

recentDocumentURLs instance method 23
removeDocument: instance method 23
reopenDocumentForURL:withContentsOfURL:error: instance method 24
reviewUnsavedDocumentsWithAlertTitle:cancellable:delegate:didReviewAllSelector:contextInfo: instance method 24
runModalOpenPanel:forTypes: instance method 25

S

saveAllDocuments: instance method 25
setAutosavingDelay: instance method 25
setShouldCreateUI: instance method 33
sharedDocumentController class method 10
shouldCreateUI instance method 34

T

typeForContentsOfURL:error: instance method 26
typeFromFileExtension: instance method 35

U

URLsFromRunningOpenPanel instance method 26

V

validateUserInterfaceItem: instance method 27

W

willPresentError: instance method 27