NSWindowController Class Reference

Developer

Contents

NSWindowController Class Reference

Inherits from	NSResponder : NSObject
Conforms to	NSCoding NSCoding (NSResponder) NSObject (NSObject)
Framework	/System/Library/Frameworks/AppKit.framework
Availability	Available in OS X v10.0 and later.
Companion guide	Document-Based App Programming Guide for Mac
Declared in	NSWindowController.h
Related sample code	BundleLoader Sketch Sketch+Accessibility SplitViews TextEdit

Overview

An `NSWindowController` object manages a window, usually a window stored in a nib file.

This management entails:

- Loading and displaying the window
- Closing the window when appropriate
- Customizing the window's title
- Storing the window's frame (size and location) in the defaults database
- Cascading the window in relation to other document windows of the application

A window controller can manage a window by itself or as a role player in the Application Kit's document-based architecture, which also includes `NSDocument` and `NSDocumentController` objects. In this architecture, a window controller is created and managed by a "document" (an instance of an `NSDocument` subclass) and, in turn, keeps a reference to the document.

The relationship between a window controller and a nib file is important. Although a window controller can manage a programmatically created window, it usually manages a window in a nib file. The nib file can contain other top-level objects, including other windows, but the window controller's responsibility is this primary window. The window controller is usually the owner of the nib file, even when it is part of a document-based application. Regardless of who is the file's owner, the window controller is responsible for freeing all top-level objects in the nib file it loads.

For simple documents—that is, documents with only one nib file containing a window—you need do little directly with `NSWindowController`. The Application Kit will create one for you. However, if the default window controller is not sufficient, you can create a custom subclass of `NSWindowController`. For documents with multiple windows or panels, your document must create separate instances of `NSWindowController` (or of custom subclasses of `NSWindowController`), one for each window or panel. An example is a CAD application that has different windows for side, top, and front views of drawn objects. What you do in your `NSDocument` subclass determines whether the default `NSWindowController` or separately created and configured `NSWindowController` objects are used.

Subclassing NSWindowController

You should create a subclass of `NSWindowController` when you want to augment the default behavior, such as to give the window a custom title or to perform some setup tasks before the window is loaded. In your class's initialization method, be sure to invoke on `super` either one of the `initWithWindowNibName:...` initializers or the `initWithWindow:` (page 9) initializer. Which one depends on whether the window object originates in a nib file or is programmatically created.

Three `NSWindowController` methods are most commonly overridden:

Method Name	Description
`windowWillLoad` (page 23)	Override to perform tasks before the window nib file is loaded.
`windowDidLoad` (page 20)	Override to perform tasks after the window nib file is loaded.
`windowTitleForDocumentDisplayName:` (page 22)	Override to customize the window title.

You can also override `loadWindow` (page 13) to get different nib-searching or nib-loading behavior, although there is usually no need to do this.

Adopted Protocols

NSCoding

- `encodeWithCoder:`
- `initWithCoder:`

Tasks

Initializing NSWindowControllers

- `initWithWindow:` (page 9)
  Returns a window controller initialized with a given window.
- `initWithWindowNibName:` (page 10)
  Returns a window controller initialized with a nib file.
- `initWithWindowNibName:owner:` (page 11)
  Returns a window controller initialized with a nib file and a specified owner for that nib file.
- `initWithWindowNibPath:owner:` (page 11)
  Returns a window controller initialized with a nib file at an absolute path and a specified owner.

Loading and Display the Window

- `loadWindow` (page 13)
  Loads the receiver's window from the nib file.
- `showWindow:` (page 18)
  Displays the window associated with the receiver.
- `isWindowLoaded` (page 12)
  Returns whether the nib file containing the receiver's window has been loaded.
- `window` (page 19)
  Returns the window owned by the receiver.

– `setWindow:` (page 16)
Sets the window controller's window.

– `windowDidLoad` (page 20)
Sent after the window owned by the receiver has been loaded.

– `windowWillLoad` (page 23)
Sent before the window owned by the receiver is loaded.

Setting and Getting the Document

– `setDocument:` (page 14)
Sets the document associated with the window managed by the receiver.

– `document` (page 9)
Returns the document associated with the receiver.

– `setDocumentEdited:` (page 14)
Sets the document edited flag for the window controller.

Closing the Window

– `close` (page 8)
Closes the window if it was loaded.

– `shouldCloseDocument` (page 17)
Returns whether the receiver necessarily closes the associated document when the window it manages is closed.

– `setShouldCloseDocument:` (page 15)
Sets whether the receiver should necessarily close the associated document when the window it manages is closed.

Getting Nib File Information

– `owner` (page 13)
Returns the owner of the nib file containing the window managed by the receiver.

– `windowNibName` (page 21)
Returns the name of the nib file that stores the window associated with the receiver.

– `windowNibPath` (page 22)

Returns the full path of the nib file that stores the window associated with the receiver.

Setting and Getting Window Attributes

– `setShouldCascadeWindows:` (page 15)

Sets whether the window should cascade in relation to other document windows.

– `shouldCascadeWindows` (page 17)

Returns whether the window will cascade in relation to other document windows when it is displayed.

– `setWindowFrameAutosaveName:` (page 16)

Sets the name under which the window's frame is saved in the defaults database.

– `windowFrameAutosaveName` (page 21)

Returns the name under which the frame rectangle of the window owned by the receiver is stored in the defaults database.

– `synchronizeWindowTitleWithDocumentName` (page 19)

Synchronizes the displayed window title and the represented filename with the information in the associated document.

– `windowTitleForDocumentDisplayName:` (page 22)

Returns the window title to be used for a given document display name.

Instance Methods

close

Closes the window if it was loaded.

```
- (void)close
```

Discussion

Because this method closes the window without asking the user for confirmation, you usually do not invoke it when the Close menu command is chosen. Instead invoke NSWindow's `performClose:` on the receiver's window.

Availability

Available in OS X v10.0 and later.

See Also

– `shouldCloseDocument` (page 17)

– `setShouldCloseDocument:` (page 15)

Related Sample Code
UIElementInspector

Declared in

`NSWindowController.h`

document

Returns the document associated with the receiver.

```
– (id)document
```

Return Value

The document associated with the receiver or `nil` if there is none.

Discussion

When a window controller is added to a document's list of window controllers, the document sets the window controller's document with `setDocument:`. The Application Kit uses this outlet to access the document for relevant next-responder messages.

Availability

Available in OS X v10.0 and later.

See Also

– `setDocument:` (page 14)

Related Sample Code
Sketch

Sketch+Accessibility

TextEdit

Declared in

`NSWindowController.h`

initWithWindow:

Returns a window controller initialized with a given window.

```
– (id)initWithWindow:(NSWindow *)window
```

Parameters

`window`

The window object to manage; can be `nil`.

Return Value

A newly initialized window controller.

Discussion

This method is the designated initializer for `NSWindowController`.

This initializer is useful when a window has been loaded but no window controller is assigned. The default initialization turns on cascading, sets the `shouldCloseDocument` (page 17) flag to `NO`, and sets the window frame autosave name to an empty string. As a side effect, the created window controller is added as an observer of the `NSWindowWillCloseNotification`s posted by that window object (which is handled by a private method). If you make the window controller a delegate of the window, you can implement NSWindow's `windowShouldClose:` delegate method.

Availability

Available in OS X v10.0 and later.

Related Sample Code

CustomMenus

UIElementInspector

Declared in

`NSWindowController.h`

initWithWindowNibName:

Returns a window controller initialized with a nib file.

```
- (id)initWithWindowNibName:(NSString *)windowNibName
```

Parameters

`windowNibName`

The name of the nib file (minus the "`.nib`" extension) that archives the receiver's window; cannot be `nil`.

Discussion

Sets the owner of the nib file to the receiver. The default initialization turns on cascading, sets the `shouldCloseDocument` (page 17) flag to `NO`, and sets the autosave name for the window's frame to an empty string.

Availability
Available in OS X v10.0 and later.

Related Sample Code
DictionaryController

Sketch

Sketch+Accessibility

SplitViews

UIElementInspector

Declared in
`NSWindowController.h`

initWithWindowNibName:owner:

Returns a window controller initialized with a nib file and a specified owner for that nib file.

```
- (id)initWithWindowNibName:(NSString *)windowNibName owner:(id)owner
```

Parameters
`windowNibName`

The name of the nib file (minus the "`.nib`" extension) that archives the receiver's window; cannot be `nil`.

`owner`

The nib file's owner; cannot be `nil`.

Discussion
The default initialization turns on cascading, sets the `shouldCloseDocument` (page 17) flag to `NO`, and sets the autosave name for the window's frame to an empty string.

Availability
Available in OS X v10.0 and later.

Declared in
`NSWindowController.h`

initWithWindowNibPath:owner:

Returns a window controller initialized with a nib file at an absolute path and a specified owner.

```
- (id)initWithWindowNibPath:(NSString *)windowNibPath owner:(id)owner
```

Parameters

`windowNibPath`

The full path to the nib file that archives the receiver's window; cannot be `nil`.

`owner`

The nib file's owner; cannot be `nil`.

Discussion

Use this method if your nib file is at a fixed location (which is not inside either the file's owner's class's bundle or in the application's main bundle). The default initialization turns on cascading, sets the `shouldCloseDocument` (page 17) flag to `NO`, and sets the autosave name for the window's frame to an empty string.

Availability

Available in OS X v10.0 and later.

Declared in

`NSWindowController.h`

isWindowLoaded

Returns whether the nib file containing the receiver's window has been loaded.

```
- (BOOL)isWindowLoaded
```

Return Value

`YES` if the nib file containing the receiver's window has been loaded, `NO` otherwise.

Availability

Available in OS X v10.0 and later.

See Also

- `loadWindow` (page 13)
- `window` (page 19)
- `windowDidLoad` (page 20)
- `windowWillLoad` (page 23)

Related Sample Code

TextEdit

Declared in

`NSWindowController.h`

loadWindow

Loads the receiver's window from the nib file.

```
- (void)loadWindow
```

Discussion

You should never directly invoke this method. Instead, invoke `window` (page 19) so the `windowDidLoad` (page 20) and `windowWillLoad` (page 23) methods are invoked. Subclasses can override this method if the way it finds and loads the window is not adequate. It uses NSBundle's `bundleForClass:` method to get the bundle, using the class of the nib file owner as argument. It then locates the nib file within the bundle and, if successful, loads it; if unsuccessful, it tries to find the nib file in the main bundle.

Availability

Available in OS X v10.0 and later.

See Also

– `isWindowLoaded` (page 12)

Declared in

`NSWindowController.h`

owner

Returns the owner of the nib file containing the window managed by the receiver.

```
- (id)owner
```

Return Value

The owner of the nib file containing the window managed by the receiver; usually `self`, but can be the receiver's document or some other object.

Availability

Available in OS X v10.0 and later.

See Also

– `windowNibName` (page 21)

Declared in

`NSWindowController.h`

setDocument:

Sets the document associated with the window managed by the receiver.

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

Parameters

`document`

The new document.

Discussion

Documents automatically call this method when they add a window controller to their list of window controllers; you should not call it directly.

Availability

Available in OS X v10.0 and later.

See Also

– `document` (page 9)

Declared in

`NSWindowController.h`

setDocumentEdited:

Sets the document edited flag for the window controller.

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

Parameters

`flag`

`YES` if the document has been edited since its last save, `NO` if it hasn't.

Discussion

The window controller uses this flag to control whether its associated window shows up as dirty. You should not call this method directly for window controllers with an associated document; the document calls this method on its window controllers as needed.

Availability

Available in OS X v10.0 and later.

Declared in

`NSWindowController.h`

setShouldCascadeWindows:

Sets whether the window should cascade in relation to other document windows.

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

Parameters

`flag`

`YES` if the window should cascade in relation to other document windows, `NO` otherwise.

Discussion

Cascading in relation to other document windows means having a slightly offset location so that the title bars of previously displayed windows are still visible.

The default is `YES`.

Availability

Available in OS X v10.0 and later.

See Also

– `shouldCascadeWindows` (page 17)

Related Sample Code

Sketch

Sketch+Accessibility

Declared in

`NSWindowController.h`

setShouldCloseDocument:

Sets whether the receiver should necessarily close the associated document when the window it manages is closed.

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

Parameters

`flag`

`YES` if the receiver necessarily closes the associated document when the window it manages is closed, `NO` otherwise.

Discussion

If `NO`, the document is closed only when the last remaining window of the document is closed.

The default is `NO`.

Availability
Available in OS X v10.0 and later.

See Also
– `shouldCloseDocument` (page 17)

Declared in
`NSWindowController.h`

setWindow:

Sets the window controller's window.

```
– (void)setWindow:(NSWindow *)aWindow
```

Parameters
`aWindow`

The new window.

Discussion
This method releases the old window and any associated top-level objects in its nib file and assumes ownership of the new window. You should generally create a new window controller for a new window and release the old window controller instead of using this method.

Availability
Available in OS X v10.0 and later.

Declared in
`NSWindowController.h`

setWindowFrameAutosaveName:

Sets the name under which the window's frame is saved in the defaults database.

```
– (void)setWindowFrameAutosaveName:(NSString *)name
```

Parameters
`name`

The name under which the window's frame is saved in the defaults database.

Discussion
By default, `name` is an empty string, causing no information to be stored in the defaults database.

Availability
Available in OS X v10.0 and later.

See Also
– `windowFrameAutosaveName` (page 21)
– `setFrameAutosaveName:` (NSWindow)

Related Sample Code
Sketch

Sketch+Accessibility

Declared in
`NSWindowController.h`

shouldCascadeWindows

Returns whether the window will cascade in relation to other document windows when it is displayed.

```
– (BOOL)shouldCascadeWindows
```

Return Value
`YES` if the window will cascade in relation to other document windows, `NO` otherwise.

Discussion
The default is `YES`.

Availability
Available in OS X v10.0 and later.

See Also
– `setShouldCascadeWindows:` (page 15)

Declared in
`NSWindowController.h`

shouldCloseDocument

Returns whether the receiver necessarily closes the associated document when the window it manages is closed.

```
– (BOOL)shouldCloseDocument
```

Return Value
YES if the receiver necessarily closes the associated document when the window it manages is closed, NO otherwise.

Discussion
If NO, the document is closed only when the last remaining window of the document is closed.

The default is NO.

Availability
Available in OS X v10.0 and later.

See Also
– setShouldCloseDocument: (page 15)

Declared in
NSWindowController.h

showWindow:

Displays the window associated with the receiver.

```
– (IBAction)showWindow:(id)sender
```

Parameters
sender

The control sending the message; can be nil.

Discussion
If the window is an NSPanel object and has its becomesKeyOnlyIfNeeded flag set to YES, the window is displayed in front of all other windows but is not made key; otherwise it is displayed in front and is made key. This method is useful for menu actions.

Availability
Available in OS X v10.0 and later.

See Also
– makeKeyAndOrderFront: (NSWindow)
– orderFront: (NSWindow)

Related Sample Code
BasicCocoaAnimations

CocoaSlides

Sketch
Sketch+Accessibility
SplitViews

Declared in
`NSWindowController.h`

synchronizeWindowTitleWithDocumentName

Synchronizes the displayed window title and the represented filename with the information in the associated document.

```
- (void)synchronizeWindowTitleWithDocumentName
```

Discussion
Does nothing if the window controller has no associated document or loaded window. This method queries the window controller's document to get the document's display name and full filename path, then calls `windowTitleForDocumentDisplayName:` (page 22) to get the display name to show in the window title.

Availability
Available in OS X v10.0 and later.

Declared in
`NSWindowController.h`

window

Returns the window owned by the receiver.

```
- (NSWindow *)window
```

Return Value
The window owned by the receiver or `nil` if there isn't one.

Discussion
If the window has not yet been loaded, this method attempts to load the window's nib file using `loadWindow` (page 13). Before it loads the window, it invokes `windowWillLoad` (page 23), and if the window controller has a document, it invokes the document's corresponding method `windowControllerWillLoadNib:` (if implemented). After loading the window, this method invokes `windowDidLoad` (page 20) and, if there is a document, the NSDocument method `windowControllerDidLoadNib:` (if implemented).

Availability
Available in OS X v10.0 and later.

See Also
– `windowControllerWillLoadNib:` (NSDocument)

Related Sample Code
AVSimpleEditorOSX

CustomMenus

Sketch

TextEdit

UIElementInspector

Declared in
`NSWindowController.h`

windowDidLoad

Sent after the window owned by the receiver has been loaded.

```
– (void)windowDidLoad
```

Discussion
The default implementation does nothing.

Availability
Available in OS X v10.0 and later.

See Also
– `loadWindow` (page 13)
– `window` (page 19)
– `windowWillLoad` (page 23)

Related Sample Code
Sketch

Sketch+Accessibility

TableViewPlayground

TextEdit

UIElementInspector

Declared in
`NSWindowController.h`

windowFrameAutosaveName

Returns the name under which the frame rectangle of the window owned by the receiver is stored in the defaults database.

```
- (NSString *)windowFrameAutosaveName
```

Return Value
The name under which the frame rectangle of the window owned by the receiver is stored in the defaults database.

Availability
Available in OS X v10.0 and later.

See Also
– `setWindowFrameAutosaveName:` (page 16)

Declared in
`NSWindowController.h`

windowNibName

Returns the name of the nib file that stores the window associated with the receiver.

```
- (NSString *)windowNibName
```

Return Value
The name of the nib file that stores the window associated with the receiver.

Discussion
If `initWithWindowNibPath:owner:` (page 11) was used to initialize the instance, `windowNibName` returns the last path component with the "`.nib`" extension stripped off. If `initWithWindowNibName:` (page 10) or `initWithWindowNibName:owner:` (page 11) was used, `windowNibName` returns the name without the "`.nib`" extension.

Availability
Available in OS X v10.0 and later.

See Also
– `owner` (page 13)

Related Sample Code
NSTableViewBinding

TableViewPlayground

Declared in
NSWindowController.h

windowNibPath

Returns the full path of the nib file that stores the window associated with the receiver.

```
- (NSString *)windowNibPath
```

Return Value
The full path of the nib file that stores the window associated with the receiver; `nil` if it cannot be located.

Discussion
If `initWithWindowNibPath:owner:` (page 11) was used to initialize the instance, the path is just returned. If `initWithWindowNibName:` (page 10) or `initWithWindowNibName:owner:` (page 11) was used, `windowNibPath` locates the nib in the file's owner's class' bundle or in the application's main bundle and returns the full path (or `nil` if it cannot be located). Subclasses can override this to augment the search behavior, but probably ought to call `super` first.

Availability
Available in OS X v10.0 and later.

Declared in
NSWindowController.h

windowTitleForDocumentDisplayName:

Returns the window title to be used for a given document display name.

```
- (NSString *)windowTitleForDocumentDisplayName:(NSString *)displayName
```

Parameters
`displayName`
The display name for the document. This is the last path component under which the document file is saved.

Discussion
The default implementation returns `displayName`. Subclasses can override this method to customize the window title. For example, a CAD application could append "-Top" or "-Side," depending on the view displayed by the window.

Availability
Available in OS X v10.0 and later.

See Also
synchronizeWindowTitleWithDocumentName (page 19)

Declared in
NSWindowController.h

windowWillLoad

Sent before the window owned by the receiver is loaded.

```
- (void)windowWillLoad
```

Discussion
The default implementation does nothing.

Availability
Available in OS X v10.0 and later.

See Also
- loadWindow (page 13)
- window (page 19)
- windowDidLoad (page 20)

Declared in
NSWindowController.h

Document Revision History

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

Date	Notes
2012-01-09	Updated companion guide.
2006-05-23	First publication of this content as a separate document.

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010