

---

# NSAlert Class Reference

User Experience: Windows & Views



2009-05-04



Apple Inc.  
© 2009 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

## NSAlert Class Reference 7

---

Overview	7
Instance Attributes	8
Subclassing Notes	8
Tasks	8
Creating Alerts	8
Configuring Alerts	9
Displaying Alerts	9
Accessing Alert Text	9
Accessing Alert Icons	10
Accessing Alert Buttons	10
Getting Alert Panels	10
Class Methods	10
alertViewWithError:	10
alertViewWithMessageText:defaultButton:alternateButton:otherButton: informativeTextWithFormat:	11
Instance Methods	12
accessoryView	12
addButtonWithTitle:	13
alertStyle	13
beginSheetModalForWindow:modalDelegate:didEndSelector:contextInfo:	14
buttons	15
delegate	15
helpAnchor	16
icon	16
informativeText	16
layout	17
messageText	17
runModal	18
setAccessoryView:	18
setAlertStyle:	19
setDelegate:	20
setHelpAnchor:	20
setIcon:	21
setInformativeText:	21
setMessageText:	21
setShowsHelp:	22
setShowsSuppressionButton:	22
showsHelp	24
showsSuppressionButton	24
suppressionButton	24

## CONTENTS

window	25
Constants	25
NSAlertStyle	25
Button Return Values	26

### **Document Revision History 27**

---

### **Index 29**

---

# Figures and Listings

## **NSAlert Class Reference 7**

---

Figure 1	Alert dialog with an accessory view 19
Figure 2	Alert dialog with a suppression checkbox 23
Listing 1	Adding an accessory view to an alert 19
Listing 2	Creating an alert with a suppression checkbox 23



# NSAlert Class Reference

---

<b>Inherits from</b>	NSObject
<b>Conforms to</b>	NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/AppKit.framework
<b>Availability</b>	Available in Mac OS X v10.3 and later.
<b>Declared in</b>	NSAlert.h
<b>Companion guides</b>	Dialogs and Special Panels Sheet Programming Topics for Cocoa
<b>Related sample code</b>	ExtractMovieAudioToAIFF QTEExtractAndConvertToAIFF QTEExtractAndConvertToMovieFile QTKitTimeCode QTRecorder

## Overview

You use an `NSAlert` object to display an alert, either as an application-modal dialog or as a sheet attached to a document window. The methods of the `NSAlert` class allow you to specify alert level, icon, button titles, and alert text. The class also lets your alerts display help buttons and provides ways for applications to offer help specific to an alert. To display an alert as a sheet, invoke the [beginSheetModalForWindow:modalDelegate:didEndSelector:contextInfo:](#) (page 14) method; to display one as an application-modal dialog, use the [runModal](#) (page 18) method.

By design, an `NSAlert` object is intended for a single alert—that is, an alert with a unique combination of title, buttons, and so on—that is displayed upon a particular condition. You should create an `NSAlert` object for each alert dialog. Normally you should create an `NSAlert` object when you need to display an alert, and release it when you are done. If you have a particular alert dialog that you need to show repeatedly, you can retain and reuse an instance of `NSAlert` for this dialog.

After creating an alert using one of the alert creation methods, you can customize it further prior to displaying it by customizing its attributes. See [“Instance Attributes”](#) (page 8)

**Note:** The `NSAlert` class, which was introduced in Mac OS X v10.3, supersedes the functional Application Kit API for displaying alerts (`NSRunAlertPanel`, `NSBeginAlertSheet`, and so on). The former API is still supported, but you should use the `NSAlert` class for your application's alert dialogs.

## Instance Attributes

---

`NSAlert` objects have the following attributes:

- **Type.** An alert's type helps convey the importance or gravity of its message to the user. Specified with `setAlertStyle:` (page 19).
- **Message text.** The main message of the alert. Specified with `setMessageText:` (page 21).
- **Informative text.** Additional information about the alert. Specified with `informativeText` (page 16).
- **icon.** The icon displayed in the alert. Specified with `setIcon:` (page 21).
- **Help.** Alerts can let the user get help about them. Use `setHelpAnchor:` (page 20) and `setShowsHelp:` (page 22).
- **Response buttons.** By default an alert has one response button: the OK button. You can add more response buttons using `addButtonWithTitle:` (page 13).
- **Suppression checkbox.** A suppression checkbox allows the user to suppress the display of a particular alert in subsequent occurrences of the event that triggers it. Use `setShowsSuppressionButton:` (page 22), `suppressionButton` (page 24).
- **Accessory view.** An accessory view lets you add additional information to an alert; for example, a text field with contact information. Use `setAccessoryView:` (page 18), `layout` (page 17).

## Subclassing Notes

---

The `NSAlert` class is not designed for subclassing.

## Tasks

### Creating Alerts

+ `alertWithError:` (page 10)

Returns an alert initialized from information in an error object.

+ `alertWithMessageText:defaultButton:alternateButton:otherButton:informativeTextWithFormat:` (page 11)

Creates an alert compatible with alerts created using the `NSRunAlertPanel` function for display as a warning-style alert.

## Configuring Alerts

- [layout](#) (page 17)  
Specifies that the receiver must do immediate layout instead of lazily just before display.
- [alertStyle](#) (page 13)  
Returns the `NSAlertStyle` constant identifying the receiver's alert style.
- [setAlertStyle:](#) (page 19)  
Sets the alert style of the receiver.
- [accessoryView](#) (page 12)  
Returns the receiver's accessory view.
- [setAccessoryView:](#) (page 18)  
Sets the receiver's accessory view.
- [showsHelp](#) (page 24)  
Indicates whether the receiver has a help button.
- [setShowsHelp:](#) (page 22)  
Specifies whether the receiver has a help button.
- [helpAnchor](#) (page 16)  
Returns the receiver's HTML help anchor.
- [setHelpAnchor:](#) (page 20)  
Associates the receiver to a given anchor.
- [delegate](#) (page 15)  
Returns the receiver's delegate.
- [setDelegate:](#) (page 20)  
Sets the receiver's delegate.

## Displaying Alerts

- [runModal](#) (page 18)  
Runs the receiver as an application-modal dialog and returns the constant positionally identifying the button clicked.
- [beginSheetModalForWindow:modalDelegate:didEndSelector:contextInfo:](#) (page 14)  
Runs the receiver modally as an alert sheet attached to a specified window.
- [suppressionButton](#) (page 24)  
Returns the receiver's suppression checkbox.
- [showsSuppressionButton](#) (page 24)  
Indicates whether the receiver shows a suppression button.
- [setShowsSuppressionButton:](#) (page 22)  
Specifies whether the receiver includes a suppression checkbox.

## Accessing Alert Text

- [informativeText](#) (page 16)  
Returns the receiver's informative text.

- [setInformativeText:](#) (page 21)  
Sets the receiver's informative text to a given text.
- [messageText](#) (page 17)  
Returns the receiver's message text (or title).
- [setMessageText:](#) (page 21)  
Sets the receiver's message text, or title, to a given text.

## Accessing Alert Icons

- [icon](#) (page 16)  
Returns the icon displayed in the receiver.
- [setIcon:](#) (page 21)  
Sets the icon to be displayed in the alert to a given icon.

## Accessing Alert Buttons

- [buttons](#) (page 15)  
Returns the receiver's buttons.
- [addButtonWithTitle:](#) (page 13)  
Adds a button with a given title to the receiver.

## Getting Alert Panels

- [window](#) (page 25)  
Provides the application-modal panel associated with the receiver.

## Class Methods

### **alertViewWithError:**

Returns an alert initialized from information in an error object.

```
+ (NSAlert *)alertViewWithError:(NSError *)error
```

#### **Parameters**

*error*

Error information to display.

#### **Return Value**

Initialized alert.

#### **Discussion**

The `NSAlert` class extracts the localized error description, recovery suggestion, and recovery options from *error* and uses them as the alert's message text, informative text, and button titles, respectively.

**Availability**

Available in Mac OS X v10.4 and later.

**Related Sample Code**

AudioDataOutputToAudioUnit  
 QTExtractAndConvertToAIFF  
 QTRecorder  
 Quartz Composer WWDC 2005 TextEdit  
 StillMotion

**Declared In**

NSAlert.h

**alertViewWithTitle:defaultButton:alternateButton:otherButton:informativeTextWithFormat:**

Creates an alert compatible with alerts created using the `NSRunAlertPanel` function for display as a warning-style alert.

```
+ (NSAlert *)alertViewWithTitle:(NSString *)messageTitle defaultButton:(NSString *)defaultButtonTitle alternateButton:(NSString *)alternateButtonTitle otherButton:(NSString *)otherButtonTitle informativeTextWithFormat:(NSString *)informativeText, ...
```

**Parameters**

*messageTitle*

Title of the alert. When `nil` or an empty string, a default localized title is used (“Alert” in English).

*defaultButtonTitle*

Title for the default button. When `nil` or an empty string, a default localized button title (“OK” in English) is used.

*alternateButtonTitle*

Title for the alternate button. When `nil`, the alternate button is not created.

*otherButtonTitle*

Title for the other button. When `nil`, the other button is not created.

*informativeText*

Informative text, optional. Can embed variable values using a format string; list any necessary arguments for this formatted string at the end of the method’s argument list. For more information on format strings, see [Formatting String Objects](#).

**Return Value**

Initialized alert.

**Discussion**

For languages that read left to right, the buttons are laid out on the bottom-right corner of the alert sheet or window, with *defaultButtonTitle* on the right, *alternateButtonTitle* on the left, and *otherButtonTitle* in the middle. The return values identifying these buttons are constants—`NSAlertDefaultReturn`, `NSAlertAlternateReturn`, and `NSAlertOtherReturn`—that correspond to the keywords.

By default, the first button has a key equivalent of Return, any button with a title of “Cancel” has a key equivalent of Escape, and any button with the title “Don’t Save” has a key equivalent of Command-D (but only if it is not the first button). You can also assign different key equivalents for the buttons using the `setKeyEquivalent:` method of the `NSButton` class. To access the alert’s buttons, use the `buttons` (page 15) method.

### Special Considerations

This is a compatibility method. It is designed for easy adoption by applications migrating from the corresponding function-based API. This method uses earlier return values—`NSAlertDefaultReturn`, `NSAlertAlternateReturn`, and `NSAlertOtherReturn`—compatible with the earlier API, rather than the return values defined by the `NSAlert` class, described in “Constants” (page 25).

Unless you must maintain compatibility with existing alert-processing code that uses the function-based API, you should allocate (`alloc`) and initialize (`init`) the object, and then set its attributes using the appropriate methods of the `NSAlert` class.

### Availability

Available in Mac OS X v10.3 and later.

### Related Sample Code

Denoise  
 ExtractMovieAudioToAIFF  
 QTExtractAndConvertToAIFF  
 QTExtractAndConvertToMovieFile  
 QTKitTimeCode

### Declared In

`NSAlert.h`

## Instance Methods

### accessoryView

Returns the receiver’s accessory view.

```
- (NSView *)accessoryView
```

### Return Value

The alert’s accessory view.

### Availability

Available in Mac OS X v10.5 and later.

### See Also

- [setAccessoryView:](#) (page 18)

### Declared In

`NSAlert.h`

## addButtonWithTitle:

Adds a button with a given title to the receiver.

```
- (NSButton *)addButtonWithTitle:(NSString *)buttonTitle
```

### Parameters

*buttonTitle*

Title of the button to add to the alert. Must not be nil.

### Return Value

Button added to the alert.

### Discussion

Buttons are placed starting near the right side of the alert and going toward the left side (for languages that read left to right). The first three buttons are identified positionally as `NSAlertFirstButtonReturn`, `NSAlertSecondButtonReturn`, `NSAlertThirdButtonReturn` in the return-code parameter evaluated by the modal delegate. Subsequent buttons are identified as `NSAlertThirdButtonReturn + n`, where *n* is an integer.

By default, the first button has a key equivalent of Return, any button with a title of “Cancel” has a key equivalent of Escape, and any button with the title “Don’t Save” has a key equivalent of Command-D (but only if it is not the first button). You can also assign different key equivalents for the buttons using the `setKeyEquivalent:` method of the `NSButton` class. In addition, you can use the `setTag:` method of the `NSButton` class to set the return value.

### Availability

Available in Mac OS X v10.3 and later.

### See Also

- [buttons](#) (page 15)

### Related Sample Code

CoreRecipes

IdentitySample

### Declared In

NSAlert.h

## alertStyle

Returns the `NSAlertStyle` constant identifying the receiver’s alert style.

```
- (NSAlertStyle)alertStyle
```

### Return Value

Alert style for the alert. See [NSAlertStyle](#) (page 25) for the list of alert style constants.

### Availability

Available in Mac OS X v10.3 and later.

### See Also

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

**Declared In**  
NSAlert.h

## **beginSheetModalForWindow:modalDelegate:didEndSelector:contextInfo:**

Runs the receiver modally as an alert sheet attached to a specified window.

```
- (void)beginSheetModalForWindow:(NSWindow *)window modalDelegate:(id)modalDelegate
    didEndSelector:(SEL)alertDidEndSelector contextInfo:(void *)contextInfo
```

### **Parameters**

*window*

The parent window for the sheet.

*modalDelegate*

The delegate for the modal-dialog session.

*alertDidEndSelector*

Message the alert sends to *modalDelegate* after the user responds but before the sheet is dismissed.

*contextInfo*

Contextual data passed to *modalDelegate* in *didEndSelector* message.

### **Discussion**

You can create the required `NSAlert` object either through the standard allocate-initialize procedure or by using the compatibility method

[alertWithMessageText:defaultButton:alternateButton:otherButton:informativeTextWithFormat:](#) (page 11).

The *alertDidEndSelector* argument must be a selector that takes three arguments, and the corresponding method should have a declaration modeled on the following example:

```
- (void) alertDidEnd:(NSAlert *)alert returnCode:(NSInteger)returnCode
contextInfo:(void *)contextInfo;
```

where *alert* is the `NSAlert` object, *returnCode* specifies which button the user pressed, and *contextInfo* is the same *contextInfo* passed in the original message. The *returnCode* argument identifies which button was used to dismiss the alert (see this method's "Special Considerations" section). The modal delegate determines which button was clicked ("OK", "Cancel", and so on) and proceeds accordingly.

If you want to dismiss the sheet from within the *alertDidEndSelector* method before the modal delegate carries out an action in response to the return value, send *orderOut:* (`NSWindow`) to the window object obtained by sending [window](#) (page 25) to the *alert* argument. This allows you to chain sheets, for example, by dismissing one sheet before showing the next from within the *alertDidEndSelector* method. Note that you should be careful not to call *orderOut:* on the sheet from elsewhere in your program before the *alertDidEndSelector* method is invoked.

### **Special Considerations**

When you use [alertWithMessageText:defaultButton:alternateButton:otherButton:informativeTextWithFormat:](#) (page 11) to create an alert, these are the constants used to identify the button used to dismiss the alert: `NSAlertDefaultReturn`, `NSAlertAlternateReturn`, and `NSAlertOtherReturn`. Otherwise, the constants used are the ones described in ["Button Return Values"](#) (page 26).

### **Availability**

Available in Mac OS X v10.3 and later.

**See Also**

- [runModal](#) (page 18)

**Related Sample Code**

CoreRecipes

IdentitySample

QTExtractAndConvertToMovieFile

QTRecorder

XMLBrowser

**Declared In**

NSAlert.h

## buttons

Returns the receiver's buttons.

- (NSArray \*)buttons

**Return Value**

The alert's buttons. The rightmost button is at index 0.

**Availability**

Available in Mac OS X v10.3 and later.

**See Also**

- [addButtonWithTitle:](#) (page 13)

**Declared In**

NSAlert.h

## delegate

Returns the receiver's delegate.

- (id < NSAlertDelegate >)delegate

**Return Value**

The alert's delegate.

**Availability**

Available in Mac OS X v10.3 and later.

**See Also**

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

**Declared In**

NSAlert.h

## helpAnchor

Returns the receiver's HTML help anchor.

- (NSString \*)helpAnchor

### Return Value

The alert's help anchor. It's `nil` when the alert has no help anchor.

### Availability

Available in Mac OS X v10.3 and later.

### See Also

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

### Declared In

NSAlert.h

## icon

Returns the icon displayed in the receiver.

- (NSImage \*)icon

### Return Value

The alert's icon.

### Discussion

The default image is the application icon (`NSApplicationIcon` application property).

### Availability

Available in Mac OS X v10.3 and later.

### See Also

- [setIcon:](#) (page 21)

### Declared In

NSAlert.h

## informativeText

Returns the receiver's informative text.

- (NSString \*)informativeText

### Return Value

The alert's informative text.

### Availability

Available in Mac OS X v10.3 and later.

### See Also

- [setInformativeText:](#) (page 21)

- [messageText](#) (page 17)

**Declared In**

NSAlert.h

## layout

Specifies that the receiver must do immediate layout instead of lazily just before display.

- (void)layout

**Discussion**

You need to call this method only when you need to customize the alert's layout. Call this method after all the alert's attributes have been customized, including the suppression checkbox and the accessory layout. After the method returns, you can make the necessary layout changes; for example, adjusting the frame of the accessory view.

**Note:** The standard alert layout is subject to change in future system software versions. Therefore, if you rely on custom alert layout, you should make sure your layouts work as expected in future releases of Mac OS.

**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

- [setAccessoryView:](#) (page 18)

**Declared In**

NSAlert.h

## messageText

Returns the receiver's message text (or title).

- (NSString \*)messageText

**Return Value**

The alert's message text.

**Availability**

Available in Mac OS X v10.3 and later.

**See Also**

- [setMessageText:](#) (page 21)

- [informativeText](#) (page 16)

**Declared In**

NSAlert.h

## runModal

Runs the receiver as an application-modal dialog and returns the constant positionally identifying the button clicked.

- (NSInteger)runModal

### Return Value

Response to the alert. See this method's "Special Considerations" section for details.

### Discussion

You can create the alert either through the standard allocate–initialize procedure or by using the compatibility method [alertViewWithMessageText:defaultButton:alternateButton:otherButton:informativeTextWithFormat:](#) (page 11).

### Special Considerations

When you use [alertViewWithMessageText:defaultButton:alternateButton:otherButton:informativeTextWithFormat:](#) (page 11) to create an alert, these are the constants used to identify the button used to dismiss the alert: `NSAlertDefaultReturn`, `NSAlertAlternateReturn`, and `NSAlertOtherReturn`. Otherwise, the constants used are the ones described in "Button Return Values" (page 26).

### Availability

Available in Mac OS X v10.3 and later.

### See Also

- [beginSheetModalForWindow:modalDelegate:didEndSelector:contextInfo:](#) (page 14)

### Related Sample Code

AudioDataOutputToAudioUnit  
 Denoise  
 QTKitTimeCode  
 Quartz Composer WWDC 2005 TextEdit  
 StillMotion

### Declared In

NSAlert.h

## setAccessoryView:

Sets the receiver's accessory view.

- (void)setAccessoryView:(NSView \*)*accessoryView*

### Parameters

*accessoryView*

View that is to be the alert's accessory view.

### Discussion

The `NSAlert` class places the accessory view between the informative text or suppression checkbox (if present) and the response buttons. To change the location of the accessory view, you must first call the [layout](#) (page 17) method.

Listing 1 shows an example of adding an accessory view to an alert. Figure 1 shows the alert generated.

**Listing 1** Adding an accessory view to an alert

```

NSTextView *accessory = [[NSTextView alloc] initWithFrame:NSMakeRange(0,0,200,15)];
NSFont *font = [NSFont systemFontOfSize:[NSFont systemFontOfSize]];
NSDictionary *textAttributes = [NSDictionary dictionaryWithObject:font
forKey:NSFontAttributeName];
[accessory insertText:[NSAttributedString alloc] initWithString:@"Text in
accessory view"
attributes:textAttributes]];

[accessory setEditable:NO];
[accessory setDrawsBackground:NO];

NSAlert* alert = [NSAlert new];
[alert setInformativeText: @"Informative text"];
[alert setMessageText: @"Message text"];
[alert setAccessoryView:accessory];
[alert runModal];

```

**Figure 1** Alert dialog with an accessory view**Availability**

Available in Mac OS X v10.5 and later.

**See Also**

- [accessoryView](#) (page 12)

**Declared In**

NSAlert.h

**setAlertStyle:**

Sets the alert style of the receiver.

```
- (void)setAlertStyle:(NSAlertStyle)style
```

**Parameters**

*style*

Alert style for the alert. Indicates the severity level of the alert. See [NSAlertStyle](#) (page 25) for the list of alert style constants.

**Availability**

Available in Mac OS X v10.3 and later.

**See Also**

- [alertViewStyle](#) (page 13)

**Related Sample Code**

CocoaDVDPlayer

CoreRecipes

IdentitySample

**Declared In**

NSAlert.h

**setDelegate:**

Sets the receiver's delegate.

- (void)setDelegate:(id < NSAlertDelegate >)delegate

**Parameters**

delegate

Delegate for the alert. nil removes the delegate.

**Availability**

Available in Mac OS X v10.3 and later.

**See Also**

- [delegate](#) (page 15)

**Declared In**

NSAlert.h

**setHelpAnchor:**

Associates the receiver to a given anchor.

- (void)setHelpAnchor:(NSString \*)anchor

**Parameters**

anchor

Anchor to associate with the alert. nil removes the associated help anchor.

**Availability**

Available in Mac OS X v10.3 and later.

**See Also**

- [helpAnchor](#) (page 16),

- [setShowsHelp:](#) (page 22)

**Declared In**

NSAlert.h

**setIcon:**

Sets the icon to be displayed in the alert to a given icon.

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

**Parameters**

*icon*

Icon for the alert. `nil` restores the application icon.

**Discussion**

By default, the image is the application icon, accessed via the application bundle's `NSApplicationIcon` property.

**Availability**

Available in Mac OS X v10.3 and later.

**See Also**

- [icon](#) (page 16)

**Declared In**

NSAlert.h

**setInformativeText:**

Sets the receiver's informative text to a given text.

```
- (void)setInformativeText:(NSString *)informativeText
```

**Parameters**

*informativeText*

Informative text for the alert.

**Availability**

Available in Mac OS X v10.3 and later.

**See Also**

- [informativeText](#) (page 16)

- [setMessageText:](#) (page 21)

**Related Sample Code**

CocoaDVDPlayer

CoreRecipes

IdentitySample

XMLBrowser

**Declared In**

NSAlert.h

**setMessageText:**

Sets the receiver's message text, or title, to a given text.

- (void)setMessageText:(NSString \*)*messageText*

#### Parameters

*messageText*

Message text for the alert.

#### Availability

Available in Mac OS X v10.3 and later.

#### See Also

- [messageText](#) (page 17)
- [setInformativeText:](#) (page 21)

#### Related Sample Code

CocoaDVDPlayer  
CoreRecipes  
IdentitySample  
XMLBrowser

#### Declared In

NSAlert.h

## setShowsHelp:

Specifies whether the receiver has a help button.

- (void)setShowsHelp:(BOOL)*showsHelp*

#### Parameters

*showsHelp*

YES for a help button, NO for no help button.

#### Discussion

When the help button is pressed, the alert delegate ([delegate](#) (page 15)) is first sent a `alertShowHelp:` message. If there is no delegate, or the delegate does not implement `alertShowHelp:` or returns NO, then the `openHelpAnchor:inBook:` message is sent to the application's help manager with a nil book and the anchor specified by [setHelpAnchor:](#) (page 20), if any. An exception is raised if the delegate returns NO and no help anchor is set.

#### Availability

Available in Mac OS X v10.3 and later.

#### See Also

- [setDelegate:](#) (page 20)
- [showsHelp](#) (page 24)

#### Declared In

NSAlert.h

## setShowsSuppressionButton:

Specifies whether the receiver includes a suppression checkbox.

- (void)setShowsSuppressionButton:(BOOL)showButton

### Parameters

showButton

When YES the alert includes the suppression checkbox.

### Discussion

You can set the title of the checkbox with the following code:

```
[[alert suppressionButton] setTitle:title];
```

Listing 2 shows how to add a suppression checkbox (with the default suppression-checkbox title) to a modal alert. Figure 2 shows the corresponding dialog.

### Listing 2 Creating an alert with a suppression checkbox

```
NSString *exampleAlertSuppress = @"ExampleAlertSuppress";
NSUserDefaults *defaults = [NSUserDefaults standardUserDefaults];
if ([defaults boolForKey:exampleAlertSuppress]) {
    NSLog(@"ExampleAlert suppressed");
}
else {
    NSAlert* alert = [NSAlert new];
    [alert setInformativeText: @"Informative text"];
    [alert setMessageText: @"Message text"];
    [alert setShowsSuppressionButton:YES];
    [alert runModal];
    if ([[alert suppressionButton] state] == NSOnState) {
        // Suppress this alert from now on.
        [defaults setBool:YES forKey:exampleAlertSuppress];
    }
}
```

Figure 2 Alert dialog with a suppression checkbox



### Availability

Available in Mac OS X v10.5 and later.

### See Also

- [suppressionButton](#) (page 24)

### Declared In

NSAlert.h

## showsHelp

Indicates whether the receiver has a help button.

- (BOOL)showsHelp

### Return Value

YES if the alert has a help button, NO otherwise.

### Availability

Available in Mac OS X v10.3 and later.

### See Also

- [setShowsHelp:](#) (page 22)

### Declared In

NSAlert.h

## showsSuppressionButton

Indicates whether the receiver shows a suppression button.

- (BOOL)showsSuppressionButton

### Return Value

YES when the alert shows a suppression button, NO otherwise. The default is NO.

### Availability

Available in Mac OS X v10.5 and later.

### See Also

- [setShowsSuppressionButton:](#) (page 22)

### Declared In

NSAlert.h

## suppressionButton

Returns the receiver's suppression checkbox.

- (NSButton \*)suppressionButton

### Return Value

The alert's suppression button.

### Discussion

You can use this method to customize the alert's suppression checkbox before the alert is displayed. For example, you can change the title of the checkbox or specify its initial state, which is unselected by default.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

NSAlert.h

## window

Provides the application-modal panel associated with the receiver.

```
- (id)window
```

### Return Value

The receiver's associated `NSPanel` object.

### Discussion

This method is useful when you want to dismiss an alert created with [beginSheetModalForWindow:modalDelegate:didEndSelector:contextInfo:](#) (page 14) within the method identified by the `didEndSelector:` parameter.

### Availability

Available in Mac OS X v10.3 and later.

### Related Sample Code

BackgroundExporter  
QTKitTimeCode

### Declared In

NSAlert.h

## Constants

### NSAlertStyle

The `NSAlert` class defines these alert styles.

```
enum {
    NSWarningAlertStyle = 0,
    NSInformationalAlertStyle = 1,
    NSCriticalAlertStyle = 2
};
typedef NSUInteger NSAlertStyle;
```

#### Constants

`NSWarningAlertStyle`

An alert used to warn the user about a current or impending event. The purpose is more than informational but not critical. This is the default alert style.

Available in Mac OS X v10.3 and later.

Declared in `NSAlert.h`.

`NSInformationalAlertStyle`

An alert used to inform the user about a current or impending event.

Available in Mac OS X v10.3 and later.

Declared in `NSAlert.h`.

`NSCriticalAlertStyle`

Reserved this style for critical alerts, such as when there might be severe consequences as a result of a certain user response (for example, a “clean install” will erase all data on a volume). This style causes the icon to be badged with a caution icon.

Available in Mac OS X v10.3 and later.

Declared in `NSAlert.h`.

**Discussion**

Currently, there is no visual difference between informational and warning alerts. You should only use the critical (or “caution”) alert style if warranted, as specified in the “Alerts” chapter in *Apple Human Interface Guidelines*.

**Availability**

Available in Mac OS X v10.3 and later.

**Declared In**

`NSAlert.h`

## Button Return Values

An alert’s return values for buttons are position dependent. The following constants describe the return values for the first three buttons on an alert (assuming a language that reads left to right).

```
enum {
    NSAlertFirstButtonReturn = 1000,
    NSAlertSecondButtonReturn = 1001,
    NSAlertThirdButtonReturn = 1002
};
```

**Constants**`NSAlertFirstButtonReturn`

The user clicked the first (rightmost) button on the dialog or sheet.

Available in Mac OS X v10.3 and later.

Declared in `NSAlert.h`.

`NSAlertSecondButtonReturn`

The user clicked the second button from the right edge of the dialog or sheet.

Available in Mac OS X v10.3 and later.

Declared in `NSAlert.h`.

`NSAlertThirdButtonReturn`

The user clicked the third button from the right edge of the dialog or sheet.

Available in Mac OS X v10.3 and later.

Declared in `NSAlert.h`.

**Discussion**

If you have more than three buttons on your alert, the button-position return value is `NSAlertThirdButtonReturn + n`, where  $n$  is an integer. For languages that read right to left, the first button’s position is closest to the left edge of the dialog or sheet.

**Declared In**

`NSAlert.h`

# Document Revision History

---

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

Date	Notes
2009-05-04	Updated for Mac OS X v10.6. Moved delegate methods to <a href="#">NSAlertDelegate Protocol Reference</a> .
2007-04-25	Updated for Mac OS X v.10.5. Clarified use of constants in alert-display methods.
	Added information about allowing the user to selectively suppress alert display to <a href="#">“Displaying Alerts”</a> (page 9).
	Added information on customizing an alerts using accessory views in <a href="#">“Configuring Alerts”</a> (page 9).
	Specified that the alert button constants used in <a href="#">runModal</a> (page 18) and <a href="#">beginSheetModalForWindow:modalDelegate:didEndSelector:contextInfo:</a> (page 14) depend on how the alert was created.
2006-06-28	Added parameter descriptions to <a href="#">beginSheetModalForWindow:modalDelegate:didEndSelector:contextInfo:</a> .
2006-05-23	First publication of this content as a separate document.

**REVISION HISTORY**

Document Revision History

# Index

---

## A

---

accessoryView **instance method** [12](#)  
addButtonWithTitle: **instance method** [13](#)  
alertStyle **instance method** [13](#)  
alertWithError: **class method** [10](#)  
alertWithMessageText:defaultButton:  
    alternateButton:otherButton:  
    informativeTextWithFormat: **class method** [11](#)

## B

---

beginSheetModalForWindow:modalDelegate:  
    didEndSelector:contextInfo: **instance method**  
    [14](#)  
**Button Return Values** [26](#)  
buttons **instance method** [15](#)

## D

---

delegate **instance method** [15](#)

## H

---

helpAnchor **instance method** [16](#)

## I

---

icon **instance method** [16](#)  
informativeText **instance method** [16](#)

## L

---

layout **instance method** [17](#)

## M

---

messageText **instance method** [17](#)

## N

---

NSAlertFirstButtonReturn **constant** [26](#)  
NSAlertSecondButtonReturn **constant** [26](#)  
NSAlertStyle **data type** [25](#)  
NSAlertThirdButtonReturn **constant** [26](#)  
NSCriticalAlertStyle **constant** [26](#)  
NSInformationalAlertStyle **constant** [25](#)  
NSWarningAlertStyle **constant** [25](#)

## R

---

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

## S

---

setAccessoryView: **instance method** [18](#)  
setAlertStyle: **instance method** [19](#)  
setDelegate: **instance method** [20](#)  
setHelpAnchor: **instance method** [20](#)  
setIcon: **instance method** [21](#)  
setInformativeText: **instance method** [21](#)  
setMessageText: **instance method** [21](#)  
setShowsHelp: **instance method** [22](#)  
setShowsSuppressionButton: **instance method** [22](#)  
showsHelp **instance method** [24](#)  
showsSuppressionButton **instance method** [24](#)  
suppressionButton **instance method** [24](#)

## W

---

window instance method [25](#)