NSAlert Class Reference

Cocoa > User Experience

2007-04-25

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

Simultaneously published in the United States and Canada.

Contents

Figures and Listings

NSAlert Class Reference

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

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 22).
- **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 23), `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).

An alert also has a delegate; see "Displaying Help" (page 9).

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 23)
  Specifies whether the receiver includes a suppression checkbox.

Displaying Help

An alert's delegate is responsible for displaying help for the alert.

- `alertShowHelp:` (page 25) *delegate method*
  Sent to the delegate when the user clicks the alert's help button. The delegate causes help to be displayed for an alert, directly or indirectly.

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 22)
  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

alertWithError:

Returns an alert initialized from information in an error object.

```
+ (NSAlert *)alertWithError:(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
GridCalendar
QTExtractAndConvertToAIFF
QTRecorder
Quartz Composer WWDC 2005 TextEdit
TextEditPlus

Declared In
`NSAlert.h`

alertWithMessageText:defaultButton:alternateButton:otherButton: informativeTextWithFormat:

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

```
+ (NSAlert *)alertWithMessageText:(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 26).

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
BackgroundExporter
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 26) 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:(int)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
BackgroundExporter
CoreRecipes
IdentitySample
QTExtractAndConvertToAIFF
QTRecorder

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)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 22)

- 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 alertWithMessageText:defaultButton:alternateButton:otherButton:informativeTextWithFormat: (page 11).

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
- beginSheetModalForWindow:modalDelegate:didEndSelector:contextInfo: (page 14)

Related Sample Code
QTExtractAndConvertToAIFF
QTExtractAndConvertToMovieFile
QTKitTimeCode
Quartz Composer WWDC 2005 TextEdit
TextEditPlus

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.

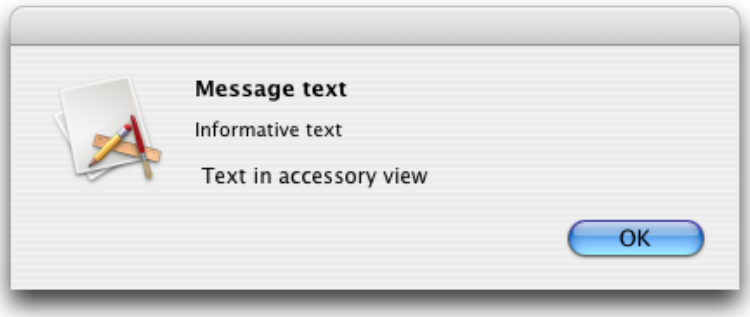
Listing 1 shows an example of adding an accessory view to an alert. Figure 1 shows the alert generated.

Listing 1 Adding an accessory view to an alert

```
NSTextView *accessory = [[NSTextView alloc] initWithFrame:NSMakeRect(0,0,200,15)];
NSFont *font = [NSFont systemFontOfSize:[NSFont systemFontSize]];
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 26) for the list of alert style constants.

Availability
Available in Mac OS X v10.3 and later.

See Also
- alertStyle (page 13)

Related Sample Code
CocoaDVDPlayer
CoreRecipes
IdentitySample

Declared In
NSAlert.h

setDelegate:

Sets the receiver's delegate.

```
- (void)setDelegate:(id)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 22)

Related Sample Code
CocoaDVDPlayer
CoreRecipes
IdentitySample

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

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: (page 25) 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:

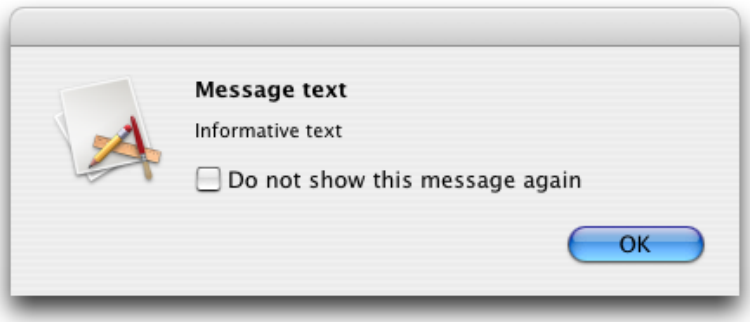
```
[[alert suppressionButton] setTitle:title];
```

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 23)

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`

Delegate Methods

alertShowHelp:

Sent to the delegate when the user clicks the alert's help button. The delegate causes help to be displayed for an alert, directly or indirectly.

```
- (BOOL)alertShowHelp:(NSAlert *)alert
```

Return Value
`YES` when the delegate displayed help directly, `NO` otherwise. When `NO` and the alert has a help anchor (`setHelpAnchor:` (page 20)), the application's help manager displays help using the help anchor.

Discussion
The delegate implements this method only to override the help-anchor lookup behavior.

Availability
Available in Mac OS X v10.3 and later.

See Also
`- setShowsHelp:` (page 22)

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
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 "Displaying Alerts" (page 9).
	Added information on customizing an alerts using accessory views in "Configuring Alerts" (page 9).
	Specified that the alert button constants used in `runModal` (page 18) and `beginSheetModalForWindow:modalDelegate:didEndSelector:contextInfo:` (page 14) depend on how the alert was created.
2006-06-28	Added parameter descriptions to beginSheetModalForWindow:modalDelegate: didEndSelector:contextInfo:.
2006-05-23	First publication of this content as a separate document.

Document Revision History

Index

W