

# NSWindow

**Inherits From:** NSResponder : NSObject  
**Declared In:** appkit/NSWindow.h

## Initializing and Getting a New Window Object

- (id)**initWithContentRect:**(NSRect)*contentRect* **styleMask:**(unsigned int)*aStyle* **backing:**(NSBackingStoreType)*bufferingType* **defer:**(BOOL)*flag* Initializes the new window object with a location and size for content of *contentRect*, a window style and buttons as indicated in the bitmap mask *aStyle*, drawing buffering as indicated by *bufferingType*. If *flag* is YES, the window server defers creating the window until it's needed.
- (id)**initWithContentRectper:**(NSRect)*contentRect* **styleMask:**(unsigned int)*aStyle* **backing:**(NSBackingStoreType)*bufferingType* **defer:**(BOOL)*flag* **screen:**(NSScreen \*)*aScreen* Initializes the new window object for *aScreen* as specified above.

## Computing Frame and Content Rectangles

- + (NSRect)**contentRectForFrameRect:**(NSRect)*aRect* **styleMask:**(unsigned int)*aStyle* Gets the content rectangle for frame rectangle *aRect* in a window of type *aStyle*.
- + (NSRect)**frameRectForContentRect:**(NSRect)*aRect* **styleMask:**(unsigned int)*aStyle* Gets the frame rectangle for content rectangle *aRect* in a window of type *aStyle*.

	<i>aStyle</i> .
+ (float) <b>minFrameWidthWithTitle:(NSString *)aTitle</b>	
<b>styleMask:(unsigned int)aStyle</b>	Returns the minimum frame width needed for <i>aTitle</i> in a window of type <i>aStyle</i> .

## Accessing the Content View

- (id) <b>contentView</b>	Returns the NSWindow's content view.
- (void) <b>setContentView:(NSView *)aView</b>	Makes <i>aView</i> the NSWindow's content view.

## Window Graphics

- (NSColor *) <b>backgroundColor</b>	Returns the window's background color.
- (void) <b>setBackgroundColor:(NSColor *)color</b>	Sets the window's background color to <i>color</i> .
- (void) <b>setTitle:(NSString *)aString</b>	Makes <i>aString</i> the window's title.
- (void) <b>setTitleAsFilename:(NSString *)aString</b>	Formats <i>aString</i> as a path and file name and makes it the window's title
- (unsigned int) <b>styleMask</b>	Returns the window's border and title-bar style.
- (NSString *) <b>title</b>	Returns the window's title string.

## Window Device Attributes

- (NSBackingStoreType) <b>backingType</b>	Returns the type of the window device's backing store.
- (NSDictionary *) <b>deviceDescription</b>	Returns the window device's attributes as key/value pairs.
- (int) <b>gState</b>	Returns the graphics-state object for the window object.
- (BOOL) <b>isOneShot</b>	Returns whether backing-store memory for the window is freed when the window is ordered off-screen.
- (void) <b>setBackingType:(NSBackingStoreType)type</b>	Sets the type of window-device backing store.
- (void) <b>setOneShot:(BOOL)flag</b>	Sets whether backing-store memory for the window should be freed when the window is ordered off-screen.
- (int) <b>windowNumber</b>	Returns the window number.

## The Miniwindow

- (NSImage \*)**miniwindowImage** Returns the image that's displayed in the miniwindow.
- (NSString \*)**miniwindowTitle** Returns the title that's displayed in the miniwindow.
- (void)**setMiniwindowImage:**(NSImage \*)*image* Sets the *image* that's displayed in the miniwindow.
- (void)**setMiniwindowTitle:**(NSString \*)*title* Sets the *title* that's displayed in the miniwindow.

## The Field Editor

- (void)**endEditingFor:**(id)*anObject* Ends the field editor's editing assignment for *anObject*.
- (NSText \*)**fieldEditor:**(BOOL)*createFlag*  
**forObject:**(id)*anObject* Returns the window object's field editor for *anObject*.  
If the field editor does not exist and *createFlag* is YES, creates a field editor.

## Window Status and Ordering

- (void)**becomeKeyWindow** Records the window's new status as the key window.
- (void)**becomeMainWindow** Records the window's new status as the main window.
- (BOOL)**canBecomeKeyWindow** Returns whether the receiving window object can be the key window.
- (BOOL)**canBecomeMainWindow** Returns whether the receiving window object can be the main window.
- (BOOL)**doesHideOnDeactivate** Returns whether deactivation hides the window.
- (BOOL)**isKeyWindow** Returns whether the receiving window object is the key window.
- (BOOL)**isMainWindow** Returns whether the receiving window object is the main window.
- (BOOL)**isVisible** Returns whether the window object is in the screen list (and thus visible).
- (int)**level** Returns the current window level.
- (void)**makeKeyAndOrderFront:**(id)*sender* Makes the receiving window object the key window and brings it forward.
- (void)**makeKeyWindow** Makes the receiving window object the key window.
- (void)**orderBack:**(id)*sender* Puts the window object at the back of its tier.
- (void)**orderFront:**(id)*sender* Puts the window object at the front of its tier.
- (void)**orderFrontRegardless** Puts the window object at the front even if the application is inactive.
- (void)**orderOut:**(id)*sender* Removes the window object from the screen list.
- (void)**orderWindow:**(NSWindowOrderingMode)*place*

**relativeTo:**(int)*otherWin*

- (void)**resignKeyWindow**
- (void)**resignMainWindow**
- (void)**setHideOnDeactivate:**(BOOL)*flag*
- (void)**setLevel:**(int)*newLevel*

Repositions the window object in the screen list in position *place* relative to another window..

Records that the window object is no longer the key window.

Records that the window object is no longer the main window.

Sets whether deactivation hides the window.

Resets the window level to *newLevel*.

## Moving and Resizing the Window

- (NSPoint)**cascadeTopLeftFromPoint:**(NSPoint)*topLeftPoint*

When successively invoked, tiles windows by offsetting them slightly to the right and down from the previous window. Returns the top left point of the placed window, which is typically used for *topLeftPoint* in the next invocation. If you specify (0,0), places the window as is, and returns its top left point.

- (void)**center**

Centers the window on the screen.

- (NSRect)**constrainFrameRect:**(NSRect)*frameRect*  
**toScreen:**(NSScreen \*)*screen*

Constrains the window's frame rectangle *frameRect* to *screen*. Returns the frame rectangle.

- (NSRect)**frame**

Returns the window's frame rectangle

- (NSSize)**minSize**

Returns the window's minimum size.

- (NSSize)**maxSize**

Returns the window's maximum size

- (void)**setContentSize:**(NSSize)*aSize*

Resizes the window's content area to *aSize*.

- (void)**setFrame:**(NSRect)*frameRect*  
**display:**(BOOL)*flag*

Moves and/or resizes the window frame to *frameRect*. *flag* determines whether the window is displayed.

- (void)**setFrameOrigin:**(NSPoint)*aPoint*

Moves the window by changing its frame origin to *aPoint*.

- (void)**setFrameTopLeftPoint:**(NSPoint)*aPoint*

Moves the window by changing its top-left corner to *aPoint*.

- (void)**setMinSize:**(NSSize)*aSize*

Sets the window's minimum size.

- (void)**setMaxSize:**(NSSize)*aSize*

Sets the window's maximum size.

## Converting Coordinates

- (NSPoint)**convertBaseToScreen**:(NSPoint)*aPoint*  
Converts *aPoint* from base to screen coordinates.
- (NSPoint)**convertScreenToBase**:(NSPoint)*aPoint*  
Converts *aPoint* from screen to base coordinates.

## Managing the Display

- (void)**display**  
Displays all the window's views.
- (void)**disableFlushWindow**  
Disables flushing for a buffered window.
- (void)**displayIfNeeded**  
Displays all the window's views that need to be redrawn.
- (void)**enableFlushWindow**  
Enables flushing for a buffered window.
- (void)**flushWindow**  
Flushes the window's buffer to the screen.
- (void)**flushWindowIfNeeded**  
Conditionally flushes the window's buffer to the screen.
- (BOOL)**isAutodisplay**  
Returns whether the window displays all views requiring redrawing when *update* is invoked.
- (BOOL)**isFlushWindowDisabled**  
Returns whether flushing is disabled.
- (void)**setAutodisplay**:(BOOL)*flag*  
Sets whether the window displays all views requiring redrawing when **update** is invoked.
- (void)**setViewsNeedDisplay**:(BOOL)*flag*  
Sets whether some views of the receiving window object should be redrawn.
- (void)**update**  
Update's the window's display and cursor rectangles. Sent after every event.
- (void)**useOptimizedDrawing**:(BOOL)*flag*  
Sets whether the window's views should optimize drawing.
- (BOOL)**viewsNeedDisplay**  
Returns whether some views of the receiving NSWindow object should be redrawn.

## Screens and Window Depths

- + (NSWindowDepth)**defaultDepthLimit**  
Returns the default depth limit for all windows.
- (BOOL)**canStoreColor**  
Returns whether the window is deep enough to store colors.
- (NSScreen \*)**deepestScreen**  
Returns the deepest screen that the window is on.
- (NSWindowDepth)**depthLimit**  
Returns the window's depth limit.
- (BOOL)**hasDynamicDepthLimit**  
Returns whether the depth limit depends on the screen.

- (NSScreen \*)**screen**
- (void)**setDepthLimit:**(NSWindowDepth)*limit*
- (void)**setDynamicDepthLimit:**(BOOL)*flag*

Returns the screen that (most of) the window is on.  
Sets the window's depth limit to *limit*  
Sets whether the depth limit will depend on the screen.

## Cursor Management

- (BOOL)**areCursorRectsEnabled**
- (void)**disableCursorRects**
- (void)**discardCursorRects**
- (void)**enableCursorRects**
- (void)**invalidateCursorRectsForView:**(NSView \*)*aView*
- (void)**resetCursorRects**

Returns whether cursor rectangles are enabled.  
Disables all cursor rectangles in the window object.  
Removes all cursor rectangles in the window object.  
Enables cursor rectangles in the window object.  
Marks cursor rectangles invalid for *aView*.  
Resets cursor rectangles for the window object.

## Handling User Actions and Events

- (void)**close**
- (void)**deminiaturize:**(id)*sender*
- (BOOL)**isDocumentEdited**
- (BOOL)**isReleasedWhenClosed**
- (void)**miniaturize:**(id)*sender*
- (void)**performClose:**(id)*sender*
- (void)**performMiniaturize:**(id)*sender*
- (int)**resizeFlags**
- (void)**setDocumentEdited:**(BOOL)*flag*
- (void)**setReleasedWhenClosed:**(BOOL)*flag*

Closes the window.  
Hides the miniwindow and redisplay the window.  
Returns whether the window's document has been edited.  
Returns whether the window object is released when it is closed.  
Hides the window and displays its miniwindow.  
Simulates user clicking the close button.  
Simulates user clicking the miniaturize button.  
Returns the event modifier flags during resizing.  
Sets whether the window's document has been edited.  
Sets whether closing the window object also releases it.

## Aiding Event Handling

- (BOOL)**acceptsMouseMovedEvents**
- (NSEvent \*)**currentEvent**
- (void)**discardEventsMatchingMask:**(unsigned int)*mask*

Returns whether the NSWindow accepts mouse-moved events.  
Returns the current event object for the application.

<b>beforeEvent:(NSEvent *)lastEvent</b>	Discards any events in the event queue that have a type indicated by bitmap <i>mask</i> until the method encounters the event <i>lastEvent</i> .
- (NSResponder *) <b>firstResponder</b>	Returns the first responder to user events.
- (void) <b>keyDown:(NSEvent *)theEvent</b>	Handles key-down events.
- (BOOL) <b>makeFirstResponder:(NSResponder *)aResponder</b>	Makes <i>aResponder</i> the first responder to user events.
- (NSPoint) <b>mouseLocationOutsideOfEventStream</b>	Provides current location of the cursor.
- (NSEvent *) <b>nextEventMatchingMask:(unsigned int)mask</b>	Returns the next event object for the application that matches the events indicated by event mask <i>mask</i> .
- (NSEvent *) <b>nextEventMatchingMask:(unsigned int)mask untilDate:(NSDate *)expiration inMode:(NSString *)mode dequeue:(BOOL)deqFlag</b>	Returns the next event object for the application that matches the events indicated by event mask <i>mask</i> , and that occurs before time <i>expiration</i> ; until <i>expiration</i> , the run loop runs in <i>mode</i> .
- (void) <b>postEvent:(NSEvent *)event atStart:(BOOL)flag</b>	Post an <i>event</i> for the application; if <i>atStart</i> is YES, the event goes to the beginning of the event queue.
- (void) <b>setAcceptsMouseMovedEvents:(BOOL)flag</b>	Sets whether the NSWindow accepts mouse-moved events.
- (void) <b>sendEvent:(NSEvent *)theEvent</b>	Dispatches mouse and keyboard events.
- (BOOL) <b>tryToPerform:(SEL)anAction with:(id)anObject</b>	Aids in dispatching action messages ( <i>anAction</i> ) to <i>anObject</i> .
- (BOOL) <b>worksWhenModal</b>	Override to return whether the window object accepts events when a modal panel is being run. Default is NO.

## Dragging

- (void) <b>dragImage:(NSImage *)anImage at:(NSPoint)baseLocation offset:(NSSize)initialOffset event:(NSEvent *)event pasteboard:(NSPasteboard *)pboard</b>	Initiates an image-dragging session. NSView invokes this method inside its implementation of <b>mouseDown:</b> .
---	--

- source:**(id)*sourceObject*
- slideBack:**(BOOL)*slideFlag*
- (void)**registerForDraggedTypes:**(NSArray \*)*newTypes* Registers the NSPasteboard types (*newTypes*) that the window object accepts in an image-dragging session.
- (void)**unregisterDraggedTypes** Unregisters the window object as a recipient of dragged images.

## Services and Windows Menu Support

- (BOOL)**isExcludedFromWindowsMenu** Returns whether the receiving window object is omitted from the Windows menu.
- (void)**setExcludedFromWindowsMenu:**(BOOL)*flag* Sets whether the receiving window object is omitted from the Windows menu.
- (id)**validRequestorForSendType:**(NSString \*)*sendType*  
**returnType:**(NSString \*)*returnType* Returns whether the window can respond to a service with send and receive types *sendType* and *returnType*.

## Saving and Restoring the Frame

- + (void)**removeFrameUsingName:**(NSString \*)*name* Removes the named frame rectangle from the system defaults.
- (NSString \*)**frameAutosaveName** Returns the name that's used to autosave the frame rectangle as a system default.
- (void)**saveFrameUsingName:**(NSString \*)*name* Saves the frame rectangle as a system default.
- (BOOL)**setFrameAutosaveName:**(NSString \*)*name* Sets the *name* that's used to autosave the frame rectangle as a system default.
- (void)**setFrameFromString:**(NSString \*)*string* Sets the frame rectangle from *string*, which encodes the position and dimensions of the frame rectangle and the position and dimensions of the screen.
- (BOOL)**setFrameUsingName:**(NSString \*)*name* Sets the frame rectangle from the named default.



- (NSString \*)**stringWithSavedFrame** Returns a string encoding the position and dimensions of the frame rectangle and the position and dimensions of the screen.

## Printing and PostScript

- (NSData \*)**dataWithEPSInsideRect:**(NSRect)*rect* Returns the encapsulated PostScript inside *rect* as a data object.
- (void)**print:**(id)*sender* Prints all the window's views.

## Assigning a Delegate

- (id)**delegate** Returns the window object's delegate.
- (void)**setDelegate:**(id)*anObject* Makes *anObject* the window object's delegate.

## Implemented by the Delegate

- (BOOL)**windowShouldClose:**(id)*sender* Notifies delegate that the window is about to close.
- (NSSize)**windowWillResize:**(NSWindow \*)*sender*  
**toSize:**(NSSize)*frameSize* Lets delegate constrain resizing to *frameSize*.
- (id)**windowWillReturnFieldEditor:**(NSWindow \*)*sender*  
**toObject:**(id)*client* Lets delegate provide another text object for field editor.

## Implemented by Observers

- (void)**windowDidBecomeKey:**(NSNotification \*)*notification* Notifies observer that the window is the key window.
- (void)**windowDidBecomeMain:**(NSNotification \*)*notification* Notifies observer that the window is the main window.
- (void)**windowDidChangeScreen:**(NSNotification \*)*notification* Notifies observer that the window changed screens.
- (void)**windowDidDeminiaturize:**(NSNotification \*)*notification*

- (void)**windowDidExpose:**(NSNotification \*)*notification*  
Notifies observer that the window was restored to screen.
- (void)**windowDidMiniaturize:**(NSNotification \*)*notification*  
Notifies observer that the window was exposed.
- (void)**windowDidMove:**(NSNotification \*)*notification*  
Notifies observer that the window was miniaturized.
- (void)**windowDidResignKey:**(NSNotification \*)*notification*  
Notifies observer that the window did move.
- (void)**windowDidResignMain:**(NSNotification \*)*notification*  
Notifies observer that the window isn't the key window.
- (void)**windowDidResize:**(NSNotification \*)*notification*  
Notifies observer that the window isn't the main window.
- (void)**windowDidUpdate:**(NSNotification \*)*notification*  
Notifies observer that the window was resized.
- (void)**windowWillMiniaturize:**(NSNotification \*)*notification*  
Notifies observer that the window was updated.
- (void)**windowWillMove:**(NSNotification \*)*notification*  
Notifies observer that the window will be miniaturized.
- (void)**windowWillMove:**(NSNotification \*)*notification*  
Notifies observer that the window will move.