# NSTrackingArea Class Reference

**Cocoa > Events & Other Input**

**2007-03-09**

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



Simultaneously published in the United States and Canada.

# Contents

# NSTrackingArea Class Reference

| | |
|---|---|
| **Inherits from** | NSObject |
| **Conforms to** | NSCoding<br>NSCopying<br>NSObject (NSObject) |
| **Framework** | /System/Library/Frameworks/AppKit.framework |
| **Availability** | Available in Mac OS X v10.5 and later. |
| **Companion guide** | Cocoa Event-Handling Guide |
| **Declared in** | NSTrackingArea.h |
| **Related sample code** | BasicCocoaAnimations<br>MenuItemView<br>TrackIt |

## Overview

An `NSTrackingArea` object defines a region of view that generates mouse-tracking and cursor-update events when the mouse is over that region.

When creating a tracking-area object, you specify a rectangle (in the view's coordinate system), an owning object, and one or more options, along with (optionally) a dictionary of data. Once it's created, you add the tracking-area object to a view using the `addTrackingArea:` method. Depending on the options specified, the owner of the tracking area receives `mouseEntered:`, `mouseExited:`, `mouseMoved:`, and `cursorUpdate:` messages when the mouse cursor enters, moves within, and leaves the tracking area. Currently the tracking area is restricted to rectangles.

An `NSTrackingArea` object belongs to its view rather than to its window. Consequently, you can add and remove tracking rectangles without needing to worry if the view has been added to a window. In addition, this design makes it possible for the Application Kit to compute the geometry of tracking areas automatically when a view moves and, in some cases, when a view changes size.

With `NSTrackingArea`, you can configure the scope of activity for mouse tracking. There are four options:

- The tracking area is active only when the view is first responder.
- The tracking area is active when the view is in the key window.
- The tracking area is active when the application is active.
- The tracking area is active always (even when the application is inactive.

Other options for `NSTrackingArea` objects include specifying that the tracking area should be synchronized with the visible rectangle of the view (`visibleRect`) and for generating `mouseEntered:` and `mouseExited:` events when the mouse is dragged.

Other `NSView` methods related to `NSTrackingArea` objects (in addition to `addTrackingArea:`) include `removeTrackingArea:` and `updateTrackingAreas`. Views can override the latter method to recompute and replace their `NSTrackingArea` objects in certain situations, such as a change in the size of the `visibleRect`.

# Adopted Protocols

NSCoding
- `encodeWithCoder:`
- `initWithCoder:`

NSCopying
- `copyWithZone:`

# Tasks

## Initializing the Tracking-Area Object

- `initWithRect:options:owner:userInfo:` (page 7)
  Initializes and returns an object defining a region of a view to receive mouse-tracking events, mouse-moved events, cursor-update events, or possibly all these events.

## Getting Object Attributes

- `options` (page 8)
  Returns the options specified for the receiver.
- `owner` (page 8)
  Returns the object owning the receiver, which is the recipient of mouse-tracking, mouse-movement, and cursor-update messages.
- `rect` (page 8)
  Returns the rectangle defining the area encompassed by the receiver.
- `userInfo` (page 9)
  Returns the dictionary containing the data associated with the receiver when it was created.

# Instance Methods

## initWithRect:options:owner:userInfo:

Initializes and returns an object defining a region of a view to receive mouse-tracking events, mouse-moved events, cursor-update events, or possibly all these events.

```
- (NSTrackingArea *)initWithRect:(NSRect)rect options:(NSTrackingAreaOptions)options
    owner:(id)owner userInfo:(NSDictionary *)userInfo
```

### Parameters

*rect*

A rectangle that defines a region of a target view, in the view's coordinate system, for tracking events related to mouse tracking and cursor updating. The specified rectangle should not exceed the view's bounds rectangle.

*options*

One or more constants that specify the type of tracking area, the situations when the area is active, and special behaviors of the tracking area. See the description of `NSTrackingAreaOptions` (page 9) and related constants for details. You must specify one or more options for the initialized object, in particular the type of tracking area; zero is not a valid value.

*owner*

The object to receive the requested mouse-tracking, mouse-moved, or cursor-update messages. It does not necessarily have to be the view associated with the created `NSTrackingArea` object, but should be an object capable of responding to the `NSResponder` methods `mouseEntered:`, `mouseExited:`, `mouseMoved:`, and `cursorUpdate:`.

*userInfo*

A dictionary containing arbitrary data for each mouse-entered, mouse-exited, and cursor-update event. When handling such an event you can obtain the dictionary by sending `userData` to the `NSEvent` object. (The dictionary is not available for mouse-moved events.) This parameter may be `nil`.

### Discussion

After creating and initializing an `NSTrackingArea` object with this method, you must add it to a target view using the `addTrackingArea:` method. When changes in the view require changes in the geometry of its tracking areas, the Application Kit invokes `updateTrackingAreas`. The view should implement this method to replace the current `NSTrackingArea` object with one with a recomputed area.

### Special Considerations

Beginning with Mac OS X v10.5, the `initWithRect:options:owner:userInfo:`, along with the `addTrackingArea:` method of `NSView`, replaces the `NSView` method `addTrackingRect:owner:userData:assumeInside:`. The latter method will be deprecated, but supported for compatibility.

### Availability

Available in Mac OS X v10.5 and later.

### See Also

- `options` (page 8)
- `owner` (page 8)
- `rect` (page 8)

- userInfo (page 9)

**Related Sample Code**
BasicCocoaAnimations
MenuItemView
TrackIt

**Declared In**
NSTrackingArea.h

## options

Returns the options specified for the receiver.

```
- (NSTrackingAreaOptions)options
```

**Discussion**
The options for an NSTrackingArea object are specified when the object is created. To determine if a particular option is in effect, perform a bitwise-AND operation with an NSTrackingAreaOptions (page 9) constant and the value returned from this method, for example:

```
if ([trackingAreaObj options] & NSTrackingInVisibleRect != 0) {
    // do something appropriate
}
```

**Availability**
Available in Mac OS X v10.5 and later.

**Declared In**
NSTrackingArea.h

## owner

Returns the object owning the receiver, which is the recipient of mouse-tracking, mouse-movement, and cursor-update messages.

```
- (id)owner
```

**Availability**
Available in Mac OS X v10.5 and later.

**Declared In**
NSTrackingArea.h

## rect

Returns the rectangle defining the area encompassed by the receiver.

```
- (NSRect)rect
```

**Discussion**
The rectangle is specified in the local coordinate system of the associated view. if the `NSTrackingInVisibleRect` (page 11) option is specified, the receiver is automatically synchronized with changes in the view's visible area (`visibleRect`) and the value returned from this method is ignored.

**Availability**
Available in Mac OS X v10.5 and later.

**Declared In**
`NSTrackingArea.h`

### userInfo

Returns the dictionary containing the data associated with the receiver when it was created.

```
- (NSDictionary *)userInfo
```

**Discussion**
Returns `nil` if no data was specified when the receiver was initialized. You can obtain this dictionary per event in each `mouseEntered:` and `mouseExited:` method by querying the passed-in `NSEvent` object with `userData`.

**Availability**
Available in Mac OS X v10.5 and later.

**Declared In**
`NSTrackingArea.h`

## Constants

### NSTrackingAreaOptions

The data type defined for the constants specified in the *`options`* parameter of `initWithRect:options:owner:userInfo:` (page 7). These constants are described below; you may specify multiple constants by performing a bitwise-OR operation with them.

```
typedef NSUInteger NSTrackingAreaOptions
```

**Declared In**
`AppKit/NSTrackingArea.h`

The following constants specify the type of the tracking area defined by an `NSTrackingArea` object. They request the type of messages the owning object should receive.

| Constant | Description |
|---|---|
| `NSTrackingMouse-EnteredAndExited` | The owner of the tracking area receives `mouseEntered:` when the mouse cursor enters the area and `mouseExited:` events when the mouse leaves the area.<br>Available in Mac OS X v10.5 and later.<br>Declared in `NSTrackingArea.h`. |

| Constant | Description |
| --- | --- |
| `NSTrackingMouseMoved` | The owner of the tracking area receives `mouseMoved:` messages while the mouse cursor is within the area.<br>Available in Mac OS X v10.5 and later.<br>Declared in `NSTrackingArea.h`. |
| `NSTrackingCursorUpdate` | The owner of the tracking area receives `cursorUpdate:` messages when the mouse cursor enters the area; when the mouse leaves the area, the cursor is appropriately reset.<br>Available in Mac OS X v10.5 and later.<br>Declared in `NSTrackingArea.h`. |

The following constants specify when the tracking area defined by an `NSTrackingArea` object.is active. The owner receives all requested messages—which can include `mouseEntered:`, `mouseExited:`, `mouseMoved:`, and `cursorUpdate:`—unless otherwise noted.

| Constant | Description |
| --- | --- |
| `NSTrackingActiveWhen-FirstResponder` | The owner receives messages when the view is the first responder.<br>Available in Mac OS X v10.5 and later.<br>Declared in `NSTrackingArea.h`. |
| `NSTrackingActive-InKeyWindow` | The owner receives messages when the view in in the key window.<br>Available in Mac OS X v10.5 and later.<br>Declared in `NSTrackingArea.h`. |
| `NSTrackingActive-InActiveApp` | The owner receives messages when the application is active.<br>Available in Mac OS X v10.5 and later.<br>Declared in `NSTrackingArea.h`. |
| `NSTrackingActiveAlways` | The owner receives messages regardless of first-responder status, window status, or application status. The `cursorUpdate:` message is *not* sent when the `NSTrackingCursorUpdate` (page 10) option is specified along with this constant.<br>Available in Mac OS X v10.5 and later.<br>Declared in `NSTrackingArea.h`. |

The following constants specify various behaviors of the tracking defined by an `NSTrackingArea` object.

| Constant | Description |
| --- | --- |
| `NSTrackingAssume-Inside` | The first event is generated when the cursor leaves the tracking area, regardless if the cursor is inside the area when the `NSTrackingArea` is added to a view. If this option is not specified, the first event is generated when the cursor leaves the tracking area if the cursor is initially inside the area, or when the cursor enters the area if the cursor is initially outside it. Generally, you do not want to request this behavior.<br>Available in Mac OS X v10.5 and later.<br>Declared in `NSTrackingArea.h`. |

| Constant | Description |
| --- | --- |
| `NSTrackingInVisibleRect` | Mouse tracking occurs only in the visible rectangle of the view—in other words, that region of the tracking rectangle that is unobscured. Otherwise, the entire tracking area is active regardless of overlapping views. The `NSTrackingArea` object is automatically synchronized with changes in the view's visible area (`visibleRect`) and the value returned from `rect` (page 8) is ignored.<br>Available in Mac OS X v10.5 and later.<br>Declared in `NSTrackingArea.h`. |
| `NSTrackingEnabledDuringMouseDrag` | The owner receives `NSMouseEntered` events when the mouse cursor is dragged into the tracking area. If this option is not specified, the owner receives mouse-entered events when the mouse is moved (no buttons pressed) into the tracking area and on `NSLeftMouseUp` events after a mouse drag. `NSMouseExited` and `NSMouseEntered` events are paired so their delivery is indirectly affected. That is, if a `NSMouseEntered` event is generated and the mouse cursor subsequently moves out of the tracking area, a `NSMouseExited` event is generated regardless if the mouse is moved or dragged, independent of this constant.<br>Available in Mac OS X v10.5 and later.<br>Declared in `NSTrackingArea.h`. |

# Document Revision History

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

| Date | Notes |
| --- | --- |
| 2007-03-09 | New document that describes the class used to set up view regions for mouse tracking and cursor updating. |

Document Revision History

# Index