Documentation

Language: Swift

AppKit / Mouse, Keyboard, and Trackpad

API Collection

Mouse, Keyboard, and Trackpad

Handle events related to mouse, keyboard, and trackpad input.

Overview

The `NSResponder` class defines the responder chain, an ordered list of objects that respond to user events. When the user clicks the mouse button, taps on the trackpad, or presses a key, an event is generated and passed up the responder chain in search of an object that can respond to it. Any object that handles events must inherit from the `NSResponder` class. The core AppKit classes, `NSApplication`, `NSWindow`, and `NSView`, inherit from `NSResponder`.

An `NSApplication` object maintains a list of `NSWindow` objects—one for each window belonging to the app—and each `NSWindow` object maintains a hierarchy of `NSView` objects. This view hierarchy is used for both drawing the user interface and for handling events.

An `NSWindow` object handles window-level events and distributes other events to its views. An `NSWindow` object also has a delegate allowing you to customize its behavior.

Topics

Responder Objects

`class NSResponder`

An abstract class that forms the basis of event and command processing in AppKit.

Mouse, Keyboard, and Touch Events

`class NSEvent`

An object that contains information about an input action, such as a mouse click or a key press.

`class NSTouch`

A snapshot of a particular touch at an instant in time.

Trackpad

`class NSPressureConfiguration`

An encapsulation of the behavior and progression of a Force Touch trackpad as it responds to specific events.

`class NSHapticFeedbackManager`

An object that provides access to the haptic feedback management attributes on a system with a Force Touch trackpad.

Constants

`struct EventTypeMask`

Constants that you use to filter out specific event types from the stream of incoming events.

`struct ButtonMask`

Constants you use to identify the activated tablet buttons in an event.

`struct ModifierFlags`

Flags that represent key states in an event object.

`struct Phase`

Constants that represent the possible phases during an event phase.

`struct SwipeTrackingOptions`

Constants that specify swipe-tracking options.

`init(type: NSEvent.EventType)`

Returns the event mask for the specified type.

See Also

User Interactions

- Menus, Cursors, and the Dock

  Implement menus and cursors to facilitate interactions with your app, and use your app’s Dock tile to convey updated information.

- Gestures

  Encapsulate your app’s event-handling logic in gesture recognizers so that you can reuse that code throughout your app.

- Touch Bar

  Display interactive content and controls in the Touch Bar.

- Drag and Drop

  Support the direct manipulation of your app’s content using drag and drop.

- Accessibility for AppKit

  Make your AppKit apps accessible to everyone who uses macOS.