Updates / AppKit updates

Article

AppKit updates

Learn about important changes to AppKit.

Overview

Browse notable changes in AppKit.

June 2025

General

- To use control metrics consistent with macOS 15 and earlier, use `prefersCompactControlSizeMetrics`.
- `NSControl.ControlSize` includes a new extra large size, `NSControl.ControlSize.extraLarge`.
- Provide seamless immersive visuals by using `NSBackgroundExtensionView` to extend a view's content under sidebars and inspectors.
- Apply Liquid Glass effects to your custom views using `NSGlassEffectView`. Use `NSGlassEffectContainerView` to efficiently merge these views when they're in proximity to one other.
- Configure buttons for Liquid Glass by setting `NSButton.BezelStyle` to `NSButton.BezelStyle.glass`.

Split views

- Add top and bottom accessory views in split views by adding one or more `NSSplitViewItemAccessoryViewController` objects to the `topAlignedAccessoryViewControllers` and `bottomAlignedAccessoryViewControllers` properties.

Toolbars

- Tint toolbar items to make them stand out and stand apart from other toolbar items by setting `NSToolbarItem.Style` to `NSToolbarItem.Style.prominent`, and setting `backgroundTintColor`.

April 2025

macOS pasteboard privacy

- Prepare your app for an upcoming feature in macOS that alerts a person using a device when your app programmatically reads the general pasteboard. The system shows the alert only if the pasteboard access wasn't a result of someone's input on a UI element that the system considers paste-related. This behavior is similar to how `UIPasteboard` behaves in iOS. New `detect` methods in `NSPasteboard` and `NSPasteboardItem` make it possible for an app to examine the kinds of data on the pasteboard without actually reading them and showing the alert. `NSPasteboard` also adds an `accessBehavior` property to determine if programmatic pasteboard access is always allowed, never allowed, or if it prompts an alert requesting permission. You can adopt these APIs ahead of the change, and set a user default to test the new behavior on your Mac. To do so, launch Terminal and enter the command `defaults write <your_app_bundle_id> EnablePasteboardPrivacyDeveloperPreview –bool yes` to enable the behavior for your app.

June 2024

General

- Organize your windows' display and layout with window tiling.

Swift and SwiftUI

- Use SwiftUI menus in AppKit with the `NSHostingMenu`.
- Animate AppKit views using SwiftUI animations using `animate(_:changes:completion:)`.

API refinements

- Use the keyboard to open context menus for UI elements on which you are focused currently.
- Add repeat, wiggle, bounce, and rotate effects to SF Symbols.
- Leverage predefined content types when saving files using the new format picker on `NSPanel`.
- Resize frames and zoom in and out with new `NSCursor` APIs such as `NSCursor.FrameResizeDirection` and `NSCursor.FrameResizePosition`.
- Control whether your toolbars display text as well as icons using the `allowsDisplayModeCustomization` property.
- Offer customized type-ahead suggestions in NSTextField using the `suggestionsDelegate`.

June 2023

Views and controls

- Use the new `userCanChangeVisibilityOf` delegate method on `NSTableView` to toggle the visibility of table columns.
- Use a new `NSProgressIndicator` property to observe progress of an ongoing task.
- Simplify how you display and style buttons with the new `.automatic` bezel style. This bezel style adapts to the most appropriate style based on the contents of the button, as well as its location in the view hierarchy.
- Display additional contextual information about currently selected documents with `NSSplitView` inspectors.
- New improvements to `NSPopover` enable you to anchor popovers from toolbar items, as well as support full-size popovers.
- Explore new UI elements in `NSMenu`. Group information more easily in section headers, lay out menu items in horizontal palettes, as well as display badge counts on menu items.

Cooperative app activation

- App activation is now driven by the user, preventing unexpected switches between apps.
- Take advantage of Cooperative Activation, where your apps can yield and accept activation from other apps on the system without interrupting the user’s workflows. For more information, see the `activate()` function on `NSApp` and `NSRunningApplication`.

Graphics

- `CGPath` and `NSBezierPath` are now interoperable. You can create a `CGPath` from a `NSBezierPath` and vice versa.
- Leverage `CADisplayLink` to synchronize your app's ability to draw to the refresh of the display.
- Create consistent, great visuals for your controls by taking advantage of standard system fill `NSColor` (`.systemFill`, `.secondarySystemFill`, `.tertiarySystemFill`, `.quaternarySystemFill`, and `.quinarySystemFill`).
- Views no longer clip their contents by default. This includes any drawing done by the view and its subviews. For more information, see the `clipsToBounds` property on `NSView`.
- Animate symbol images with the new `addSymbolEffect` function on `NSImageView`. Symbol effects include: bounce, pulse, variable color, scale, appear, disappear, and replace.
- Display and manipulate high dynamic range (HDR) images.

Swift and SwiftUI

- AppKit more fully integrates with Swift and SwiftUI with Sendable (`NSColor`, `NSColorSpace`, `NSGradient`, `NSShadow`, `NSTouch`) and Transferable (`NSImage`, `NSColor`, `NSSound`) types.
- Preview your views and view controllers alongside your code using the new `#Preview` Swift macro. Incrementally adopt SwiftUI into your AppKit life cycle by leveraging modifiers like toolbar and navigation title on `NSWindows`.
- Simplify your code with new attributes, `@ViewLoading` and `@WindowLoading`, to help with view and window loading.

Text improvements

- Help people enter text more effectively with the `NSTextInsertionIndicator` that adapts to the current accent color of the app. Cursor accessories also help users visualize where and how to enter text.
- Simplify `NSTextField` entry by leveraging the new `.contentType` AutoFill feature, making it more convenient to enter types such as contact information, birthdays, names, credit cards, and street addresses.
- Adopt text styles like `.body`, `largeTitle`, and `headline` on `NSFont.preferredFont` to take advantage of enhancements to the font system, like improved hyphenation for non-English languages and dynamic line-height adjustments for languages that require more vertical space. Access localized variants of symbol images by specifying a locale.

Related sessions from WWDC23

Session 10054: What’s new in AppKit

See Also

Technology updates

Accelerate updates

Learn about important changes to Accelerate.

Accessibility updates

Learn about important changes to Accessibility.

ActivityKit updates

Learn about important changes in ActivityKit.

AdAttributionKit Updates

Learn about important changes to AdAttributionKit.

App Clips updates

Learn about important changes in App Clips.

App Intents updates

Learn about important changes in App Intents.

Apple Intelligence updates

Learn about important changes to Apple Intelligence.

AppleMapsServerAPI Updates

Learn about important changes to AppleMapsServerAPI.

Apple Pencil updates

Learn about important changes to Apple Pencil.

ARKit updates

Learn about important changes to ARKit.

Audio Toolbox updates

Learn about important changes to Audio Toolbox.

AuthenticationServices updates

Learn about important changes to AuthenticationServices.

AVFAudio updates

Learn about important changes to AVFAudio.

AVFoundation updates

Learn about important changes to AVFoundation.

Background Tasks updates

Learn about important changes in Background Tasks.