Updates / UIKit updates

Article

UIKit updates

Learn about important changes to UIKit.

Overview

Browse notable changes in UIKit.

June 2025

General

- Provide seamless immersive visuals by using `UIBackgroundExtensionView` to extend a view's content under sidebars and inspectors.
- Apply Liquid Glass effects to views using `UIGlassEffect`.
- Organize views together for morph animations in `UIGlassContainerEffect`.
- Add or adjust effects at the edge of a scroll view with `UIScrollEdgeEffect`.
- Apply Liquid Glass effects to buttons with `glass()` and `prominentGlass()`.
- UIKit now supports Swift Observable objects. Use observable objects in `layoutSubviews()`; then UIKit automatically invalidates and updates the UI when those objects change.
- Add a badge to a `UIBarButtonItem` with `badge`.
- Notification payloads are now strongly typed: `NotificationCenter.MessageIdentifier`.

Menu bar in iPadOS

- Swipe from the top to reveal an iPad app’s full menu. Menus on iPad support images, submenus, inline sections, checkmarks, and more.
- Configure main menus with `UIMainMenuSystem`.

High dynamic range (HDR)

- `UIColorPickerViewController` supports picking HDR colors, with a maximum supported exposure value.
- Observe `UITraitHDRHeadroomUsageLimit` to automatically adjust HDR usage when a view with HDR content is not in focus.

June 2024

General

- Leverage automatic trait usage tracking inside key update methods such as `layoutSubviews()`, eliminating the need for manual trait change registration and invalidation.
- Add repeat, wiggle, breathe, and rotate effects to SF Symbols.
- Take advantage of enhancements to `UIListContentConfiguration`, which now automatically updates to match the style of the containing list by using the new `UIListEnvironment` trait from the trait collection, removing the need to instantiate a configuration for a specific list style yourself.
- Opt out or restrict collaboration on certain types of data through the share sheet using `UIActivityCollaborationMode`.
- Select a specific week of the year in `UICalendarView` using the new `UICalendarSelectionWeekOfYear` selection option.
- Observe, participate in, and affect the UI update process using `UIUpdateLink`.

Navigation

- Showcase your app and its unique identity with a new, customizable launch design for document-based apps. In UIKit, define `launchOptions` on your `UIDocumentViewController`.
- Make your app’s navigation more immersive by adopting the new tab bar on iPad. If your app presents a rich hierarchy of tab items, set the `mode` to `UITabBarController.Mode.tab`

Sidebar to automatically switch between the tab bar and sidebar representations. In SwiftUI, use sidebarAdaptable.

- Transition between views in a way that feels fluid and consistent using a systemwide zoom transition. In UIKit, configure your view controller's preferredTransition to zoom(options:sourceViewProvider:). In SwiftUI, use zoom(sourceID:in:).

Framework interoperability

- Reuse existing UIKit gesture recognizer code in SwiftUI. In SwiftUI, create UIKit gesture recognizers using UIGestureRecognizerRepresentable. In UIKit, refer to SwiftUI gestures by name using name.

visionOS

- Support more varieties of list layouts by configuring whether section headers stretch to fill the entire width of the list or shrink to tightly hug their content. For collection views, use contentHuggingElements on UICollectionLayoutListConfiguration. For table views, use contentHuggingElements on UITableView.
- Animate SF Symbols on visionOS using the symbol effects API and UIImageView.
- Apply hierarchical vibrant text color to labels using UIColor.Prominence.
- Specify an action to perform without shifting the focus away from the keyboard using keyboardAction.
- Push a new scene in place of an existing scene using UIWindowScenePushPlacement. The new scene appears in the same position as the original scene, hiding it. Closing the new scene makes the original scene reappear.

tvOS

- Create a unifying color theme in your app by specifying an accent color in your app's asset catalog, which is now supported in tvOS.

June 2023

General

- Preview your views and view controllers alongside your code using the new #Preview Swift macro.

- Take advantage of a new view controller appearance callback, `viewIsAppearing(_:)`, to run code that depends on the view's initial geometry. The system calls this method when both the view and view controller have an up-to-date trait collection, and after the superview adds the view to the hierarchy and lays it out. This method deploys back to iOS 13.
- Learn about enhancements to the trait system, which let you define custom traits for your own data, quickly change trait values throughout the view hierarchy, and register for trait changes in more flexible ways. For more information, see WWDC23 session 10057: Unleash the UIKit trait system.
- Display and manage empty state consistently in your app with `UIContentUnavailableConfiguration`, which provides new system standard styles and layouts for common empty states. Help people understand why no content is present, and when possible, provide guidance on how to add content.
- Create a powerful text experience in your app. Define richer interactions by changing the default tap or menu behavior when interacting with a text item. If you implement a custom UI for displaying text, support the redesigned text cursor by adopting the new text selection UI. Mark up text fields with additional text content types to help people fill out forms even faster. For more information, see WWDC23 session 10058: What's new with text and text interactions.
- Let people drop supported files and content onto your app icon on the Home Screen to open them in your app. To make sure your app is properly configured, verify that your `Info.plist` file specifies the file types your app supports using `CFBundleDocumentTypes`.

Accessibility and internationalization

- Simplify how you maintain your accessibility code with block-based setters for accessibility attributes. Make sure people receive the most important information first by specifying a default, low, or high priority for announcements. Enhance custom accessibility elements with the new toggle and zoom accessibility traits.
- Create a great text experience for international users by testing your UI in all languages. Adopt text styles to take advantage of enhancements to the font system, like improved wrapping and hyphenation for Chinese, German, Japanese, and Korean, as well as enhancements for variable line heights that improve legibility in several languages, including Arabic, Hindi, Thai, and Vietnamese. Access localized variants of symbol images by specifying a locale.

iPadOS

- Help people customize their Stage Manager configuration by including a larger target area for dragging windows. Leverage new resizing behavior for split view controllers to get the most out of your UI in Stage Manager.

- Support scrolling of your scroll view content with hardware keyboard shortcuts. This behavior is enabled by default, which you can override using `allowsKeyboardScrolling`.

- Simplify document management in your document-centric apps. Set your `UIDocument` subclass as the rename delegate of a navigation item to handle file renaming automatically. Build your content view controller from `UIDocumentViewController`, which provides a system default experience for managing documents: automatically configuring the title menu, sharing, drag and drop, key commands, and more. For more information, see WWDC23 session 10056: Build better document-centric apps.

- Enhance the Apple Pencil experience in your iPadOS app. Give your app a sense of depth by using `UIHoverGestureRecognizer` to draw a preview of the stroke. Support the beautiful new inks in PencilKit, including monoline, fountain pen, watercolor, and crayon.

Views and controls

- Animate symbol images with new symbol effects, including bounce, pulse, variable color, scale, appear, disappear, and replace.

- Build even more performant apps with flexible layouts using collection views. Apply diffable data source snapshots and perform batch updates with even better performance. Use the `uniformAcrossSiblings(estimate:)` dimension for compositional layouts to specify uniform size across sibling items, with smaller items increasing in size to match their largest sibling.

- Simplify spring animations by providing duration and bounce parameters for the new view animation method, `animate(springDuration:bounce:initialSpringVelocity:delay:options:animations:completion:)`.

- Represent fractional progress through a page of content with page controls.

- Display and manipulate high dynamic range (HDR) images.

- Display your menu as a palette with `displayAsPalette` for it to appear as a row of menu elements for choosing from a collection of items.

- Take advantage of the `UIStatusBarStyle.default` status bar style, which now automatically chooses a light or dark appearance that maintains contrast with the content underneath it.

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.

AppKit updates

Learn about important changes to AppKit.

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.