iOS SDK Release Notes for iOS 10 beta 8

Developer

Introduction

iOS SDK 10.0 beta 8 provides support for developing iOS apps. It’s packaged with a complete set of Xcode tools, compilers, and frameworks for creating apps for iOS and OS X. These tools include the Xcode IDE and the Instruments analysis tool, among many others.

With this software you can develop apps for iPhone, iPad, or iPod touch running iOS 10. You can also test your apps using the included Simulator, which supports iOS 10. iOS SDK 10.0 requires a Mac computer running OS X v10.10.3 (Yosemite) or later.

This version of iOS is intended for installation only on devices registered with the Apple Developer Program. Attempting to install this version of iOS in an unauthorized manner could put your device in an unusable state.

For more information and additional support resources, visit http://developer.apple.com/programs/ios/.

Bug Reporting

For issues not mentioned in the Notes and Known Issues section, please file bugs through the Apple Developer website https://developer.apple.com/bug-reporting/ios/. Additionally, you may discuss these issues and iOS SDK 10.0 in the Apple Developer Forums at https://forums.developer.apple.com/community/beta/ios-10-beta. To get more information about iCloud for Developers, go to http://developer.apple.com/icloud.

Autosubmission of Diagnostic and Usage Data

By default, the iOS 10.0 beta 8 automatically sends anonymous diagnostic and usage data back to Apple. This includes information about crashes, freezes, kernel panics, and information about how you use Apple and third-party software, hardware, and services. This information is used to help Apple improve the quality and performance of its products and services. You can stop autosubmission of diagnostics and usage data by going to Settings > Privacy > Diagnostics and Usage > Don’t Send.

Functionality not in iOS 10 beta 8

Functionality that requires adoption from App Store apps, such as iMessage apps, SiriKit, and Maps extensions, will not be available until those apps are able to adopt and submit to the Store.

Notes and Known Issues

AVFoundation

Note

In iOS 10 beta 3, the names of the optional `AVCapturePhotoCaptureDelegate` methods `didFinishProcessingPhoto` and `didFinishProcessingRawPhoto` were changed to `didFinishProcessingPhotoSampleBuffer:previewPhotoSampleBuffer:resolvedSettings:bracketSettings:error:` and `didFinishProcessingPhotoSampleBuffer:previewPhotoSampleBuffer:resolvedSettings:bracketSettings:error:`, respectively. Code that uses the previous names (such as the WWDC version of the *AVCamManual* sample project) will continue to compile, but the older delegate callbacks will not execute.

Binary Compatibility

- Apple reserves two-letter prefixes for use in framework classes. When naming your own classes, please use a three-letter prefix. The guidelines can be reviewed here:

  https://developer.apple.com/library/ios/documentation/Cocoa/Conceptual/ProgrammingWithObjectiveC/Conventions/Conventions.html

  Failure to follow these guidelines could result in your app crashing during beta software releases.

- Upon recompiling with iOS 10.0, calling `[NSObject valueForKey:]` with a `nil` key throws an exception. Previously, this led to undefined behavior; now, it causes your app to crash.
- Referencing a system font by name in a call to `+[UIFont fontWithName:size:]` causes a crash. For more information, see https://developer.apple.com/videos/play/wwdc2015/804/
- To improve customer privacy, `https://` URLs, `NSURLSession,` and `NSURLConnection` no longer support RC4 cipher suites during the TLS handshake. Affected apps and services should upgrade web servers to use more modern cipher suites.
- Apps may hang if they change a superview's geometry in `viewWillLayoutSubviews` or `layoutSubviews`.
- `NSURLConnection` disallows connections that use TLS protocol versions lower than the protocol version specified by an ATS policy via the `NSExceptionMinimumTLSVersion` or `NSThirdPartyExceptionMinimumTLSVersion` keys. Affected apps and services should upgrade web servers to use more modern TLS protocol versions.

CFNetwork HTTPProtocol

The `NSMutableURLRequest` class requires that the `HTTPBodyStream` property be an unopened stream, and the `NSURLConnection` and `NSURLSession` classes now strictly enforce this unopened stream requirement. Affected apps should ensure that any `NSInputStream` that is provided has not yet been opened.

CloudKit

When building and running from Xcode repeatedly, long-lived operations can fail with a "You don’t have permission to save the file” error because the container path is changing repeatedly.

FaceTime

FaceTime calls between this beta and older iOS and macOS betas are not supported. Please update your Mac and iPhone to the latest version.

HomeKit

Adding WAC HomeKit accessories might fail using the Home app if the network credentials are not first provided using the Settings > Wi-Fi > Set Up New Device option.

Workaround: If adding such a HomeKit accessory fails, provide the network credentials using Settings > Wi-Fi > Set Up New Device. After the accessory has joined the network, the accessory can be added using the Home app.

libdispatch

Libdispatch asserts if there is a hang detected due to a deadlock in `dispatch_barrier_sync`.

Messages

- When `UISearchController` and `UITableViewController` are used in Messages extensions, their content can be hidden below the top bar.

  Workaround: Use insets of around 80px on top and 40px on the bottom.
- In Simulator only, when `localizedChangeDescription` in the `insertMessage:localizedChangeDescription:completionHandler:` method is set to `$localParticipantIdentifier.UUIDString`, the `$localParticipantIdentifier.UUIDString` is not replaced with the user’s Messages ID, and the UUIDString is printed as-is.
- When the Messages app in Simulator is force quit, message history is lost.
- When a `UIAlertController` object is presented in a Messages extension, it is truncated by the bottom bar of the extension.

- If a sign-in dialog is displayed while performing an in-app purchase or attempting to buy content from the store, or the store or the extension will be dismissed.
- The local participant UUID is the same for both conversation participants in the Simulator only.

  Workaround: Use a device to test UUID comparisons.
- When reading messages, Siri does not accurately describe new message types or features.

Mobile Device Management

- Starting in iOS 10, SCEP payloads no longer default to MD5 if a SCEP server fails to return a CACaps or does not claim capability for SHA-1, SHA-256, or SHA-512 in CACaps.

  If a SCEP server does not respond to GetCACaps, SHA-1 will be assumed and used for the SCEP attempt.

  If the SCEP servers respond to GetCACaps, the server needs to note they have SHA-1, SHA-256, or SHA-512 capability or the SCEP enrollment request is failed due to insufficient capabilities.

  See the WWDC 2016 session What's New in Managing Apple Devices for more information.
- To encourage stronger passcode usage, iOS 10 will always prompt a user to create a password of at least 6 characters on a TouchID–capable device even if a passcode policy is in place that allows shorter passcodes. In this case users can still enter a shorter passcode that meets the passcode policy.

Music

- Recently Played in For You may not refresh instantly.
- `MPMediaPickerController` may not display as expected.
- Deleting Apple Music may prevent certain accessories from playing audiobooks, podcasts, and more. Download Apple Music from the App Store to restore playback.
- The today widget for music may not show all recently played music.

When connected to a car, certain features may not work as expected:

- Editing Up Next may unexpectedly disrupt playback.
- While using CarPlay, View All Stations does not yet display all stations.

Notes

Notes may quit unexpectedly.

Workaround: Toggle Notes off, then on, in Settings > General > iCloud.

NSUserActivity

An `NSUserActivity` object may not have any `userInfo` after handoff.

Workaround: Explicitly call `becomeCurrent` on the activity object.

Phone

Voicemail transcription (beta) is available on iPhone 6s, iPhone 6s Plus, and iPhone SE with Siri language set to English (United States or Canada).

Photos

- People syncing is not enabled via iCloud Photo Library in iOS 10.
- Memories, Related, People, and Scene are not supported on 32-bit devices.

Safari

- WebGeolocation now requires a secure (HTTPS) website to work on both iOS and macOS to prevent malicious use of location data.
- WKWebView now defaults to respecting user-scalable=no from a viewport. Clients of WKWebView can improve accessibility and allow users to pinch-to-zoom on all pages by setting the `WKWebViewConfiguration` property `ignoresViewportScaleLimits` to `YES`.
- The `SFSafariViewControllerConfiguration` and `–[SFSafariViewController initWithURL:configuration:]` APIs have been removed, and `–[SFSafariViewController initWithURL:entersReaderIfAvailable:]` is no longer marked as deprecated. The `preferredBarTintColor` property has been moved to `SFSafariViewController`, along with a new property `preferredControlTintColor` which clients should use instead of setting `tintColor` directly on the view. Apps linked on iOS 10 or later will no longer forward their view's tint color to `SFSafariViewController`.

Swift Playgrounds

Swift Playgrounds is a brand new app designed to help people learn to code with Swift 3. It offers downloadable content, including Learn to Code Part 1 and Learn to Code Part 2—which teaches the fundamentals of coding in Swift—and additional challenges you can explore. You can also experiment on your own by creating playgrounds based on the templates, create a blank playground, or open playgrounds created in Xcode.

Swift Playgrounds is included with iOS 10 beta 8.

Swift Playgrounds beta 8 contains Learn to Code Part 1 and Learn to Code Part 2.

Notes

- Playgrounds that are not stored in iCloud are not available when upgrading to the version of Swift Playgrounds available in the App Store this fall. To keep your playgrounds, either turn on iCloud, or use the share sheet to AirDrop, Message, or email your playgrounds for backup on another computer or device before moving to the final release Swift Playgrounds.

- Swift Playgrounds beta 6 uses Swift 3.0 preview 4 (swiftlang-800.0.41.1). Code written using other versions of Swift may not work in Swift Playgrounds beta 6-8.
- Playgrounds created with Swift Playgrounds beta 1 through 4 do not execute correctly in Swift Playgrounds beta 8. Similarly, playgrounds made available for Swift Playgrounds beta 7 do not execute correctly when run on Swift Playgrounds beta 1 through 4. This limitation includes the playgrounds in Learn to Code.
- Swift Playgrounds on iPad does not keep track of whether you've expanded an inline Quick Look for values in the editor. For example, if you edit a Swift 3 playground that had an expanded Quick Look, Quick Looks are collapsed when you save.
- AVFoundation is available, but input capture returns nothing.
  For example, `AVCapturePreviewLayer` is always a blank transparent layer.
- An issue prevents SceneKit views (`SCNView`) from being rendered inside Swift Playgrounds when the current camera (`SCNCamera`) uses new effects, such as color grading, color fringe, or saturation and contrast.
- GameController framework is available and appears to successfully connect to controllers, but button handler blocks are never called.

UIKit

Notes

- Prior to iOS 10, it was possible to override `[UIStackView initWithArrangedSubviews:]`, but this was intended as a convenience initializer and implemented as such. We have now enforced this in the headers. As a result, Swift clients can no longer override this method, because Swift initializer rules prevent the override of a convenience initializer.
- In iOS 10, UIKit has updated and unified background management for `UINavigationBar`, `UITabBar`, and `UIToolbar`. In particular, changes to background properties of these views (such as background or shadow images, or setting the bar style) may kick off a layout pass for the bar to resolve the new background appearance.

  In particular, this means that attempts to change the background appearance of these bars inside of `–[UIView layoutSubviews]`, `–[UIView updateConstraints]`, `–[UIViewController willLayoutSubviews]`, `–[UIViewController didLayoutSubviews]`, `–[UIViewController updateViewConstraints]`, or any other method that is called in response to layout may result in a layout loop.

  In some cases you can break these layout loops by ensuring that you always use the same object instance when objects (such as `UIImage` or `UIColor`) are required. But in general you should avoid doing this.

  Because all appearance parameters are now resolved at one time, there may be some cases where your bar's appearance has changed. In general, best results are obtained by specifying as little as possible for customizing your bar. For example, if you are specifying a `barTintColor` value and specifying an empty `UIImage` object for the `backgroundImage` property (as is the case when you call `[UIImage new]`, for example) then you should get better results by specifying only the `barTintColor`. Any changes you make to resolve these issues in iOS 10 should also work correctly in iOS 9—if this is not the case, please report bugs with a sample project and a screenshot indicating what the bars should look like.

- In iOS 10, there is a slight `UIGestureRecognizer` behavior change when removing a currently recognizing (that is, midflight) gesture recognizer from its `UIView`. Previously, removing the gesture recognizer midflight would not explicitly cancel the gesture recognizer, allowing you to re-add the gesture recognizer back to the same view or to a different view. In iOS 10, calling `–[UIView removeGestureRecognizer:]` on the view of a midflight gesture recognizer explicitly cancels the gesture recognizer. If a user desires to change the view of a midflight gesture recognizer, you can simply call `–[UIView addGestureRecognizer:]` on the view you wish to move the gesture recognizer to.
- Presented view controllers can now affect the status bar appearance even if they were presented from a view controller that did not affect the status bar (for example, a popover). By default, custom view controller presentations are assumed to not affect the status bar; use the `modalPresentationCapturesStatusBarAppearance` property on `UIViewController` to allow a presented view controller to participate in status bar appearance.
- It has always been a requirement that `UIViewController` subclasses call super's implementation of `–awakeFromNib` from their own overrides. Starting in iOS 10, `–awakeFromNib` is correctly annotated with the `NS_REQUIRES_SUPER` attribute to detect implementations that fail to obey this requirement. To fix this warning, ensure that all code paths of your override call `[super awakeFromNib]`.
- When running on iPad, the background color set for a `UITableViewCell` in a Storyboard is now respected.
- Starting in iOS 10, `UITableViewHeaderFooterView` supports `NSCoding` so if a view with this class is in a XIB, it now decodes correctly. The consequence is that apps may exhibit some extra content for these views which did not appear before due to the bug that was fixed.
- For very wide table views where cell layout margins have been automatically increased to follow the readable width, separator insets are now interpreted relative to these left and right margins instead of from the edges of the table view.
- The coalescing of `UITouch` delivery has been significantly improved, especially in cases where users would both touch the screen and use Apple Pencil at the same time on iPad Pro and the app wasn't able to process them at the incoming rate. In certain scenarios, events still can come in at a higher rate than the display refreshes. This is expected, and your app should anticipate this and handle accordingly.
- In iOS 10, windows that are not full screen do not affect status bar appearance.
- Sending `–layoutIfNeeded` to a view is not expected to move the view, but in earlier releases, if the view had `translatesAutoresizingMaskIntoConstraints == false`, and if it was being positioned by constraints, `–layoutIfNeeded` would move the view to match the layout engine before sending layout to the subtree.

  These changes correct this behavior, and the receiver's position and usually its size won't be affected by `–layoutIfNeeded`.

Some existing code may be relying on this incorrect behavior that is now corrected. There is no behavior change for binaries linked before iOS 10, but when building on iOS 10 you may need to correct some situations by sending `-layoutIfNeeded` to a superview of the `translatesAutoresizingMaskIntoConstraints == false` view that was the previous receiver, or else positioning and sizing it before (or after, depending on your desired behavior) `-layoutIfNeeded`.

- Third party apps with custom `UIView` subclasses using Auto Layout that override `layoutSubviews` and dirty layout on self before calling super are at risk of triggering a layout feedback loop when they rebuild on iOS 10. When they are correctly sent subsequent `-layoutSubviews` calls they must be sure to stop dirtying layout on self at some point (note that this call was skipped in release prior to iOS 10).
- Flippable images work by having two images in an asset, each with a different directionality trait. When you create a derived `UIImage` object using the `-imageWith…` methods, it is no longer associated with the image asset it came from. To create a flippable template image at runtime, use `UIImageAsset`.
- The source of the `UIContentSizeCategoryDidChangeNotification` notification is now `UIScreen.main()` instead of `UIApplication.shared()`.
- There are two properties in the `UIViewPropertyAnimator` class and one method in the `UIViewAnimating` protocol that are unavailable in iOS 10 beta 1.

  `UIViewPropertyAnimator`:

  ```
  @property(nonatomic, getter=isManualHitTestingEnabled) BOOL
  manualHitTestingEnabled;
  @property(nonatomic, readonly) NSTimeInterval delay;
  ```

  `UIViewAnimating`:

  ```
  (void)startAnimationAfterDelay:(NSTimeInterval)delay;
  ```

Known issue

For `UIImage` objects that are created from `CIImage` objects, the `UIImage` drawing methods (`drawInRect`, `drawAtPoint`) will always convert to the `DeviceRGB` color space before drawing. This results in loss of extended color information when drawing into a wide-color graphics context.

Workaround: You can retrieve the underlying `CIImage` via the `UIImage.CIImage` property and render it using a `CIContext` created with the appropriate color space (Extended sRGB) and pixel format (full-float).

A `UIImage` object that is created from `CGImageRef` is not affected, and will draw correctly without loss of color information.

Widgets

The first time you debug a widget (that is, a Today extension) on a device, it does not show as a possible extension.

Workaround: Debug again for the extension to show up.

Xcode

- Occasionally, using Command-Shift-HH from the Home screen does not invoke the app switcher.

  Workaround: Launch any app before using Command-Shift-HH.

- `Intents.framework` protocol methods require `@objc` annotation to be properly bridged between Obj-C and Swift 2.3.