Xcode 8.3 beta 2 Release Notes

Developer

About Xcode 8.3 beta 2

Supported Configurations

Xcode 8.3 beta 2 requires a Mac running macOS 10.12 or later.

Xcode 8.3 beta 2 includes SDKs for iOS 10.3, watchOS 3.2, macOS 10.12.4, and tvOS 10.2.

Installation

Xcode 8.3 2 beta can coexist with previous versions of Xcode. (Note - running multiple instances of Xcode simultaneously is not supported.)

Prerelease versions of Xcode are made available from developer.apple.com, packaged in a compressed XIP file. To install Xcode during the beta period, download the XIP file, double-click the file to expand it in place, then drag Xcode-beta.app to the Applications folder.

Upon final release, Xcode is installed via the Mac App Store.

Accessing Additional Developer Tools

To launch additional developer tools, such as Instruments and FileMerge, launch Xcode-beta and select Xcode > Open Developer Tool. You can keep these additional tools in your Dock for direct access when Xcode isn't running.

Configuring Xcode Server

macOS 10.12 or later and Server 5.2 or later are required to perform continuous integration with Xcode 8.3 beta 2.

Install macOS, Server, and Xcode 8.3 beta. Next, map Xcode 8.3 beta 2 to Server by performing the following steps:

1. Launch the Server app and set up for first time
2. In the Services list in the Server sidebar, select the Xcode service.
3. Click the Choose Xcode button, and select Xcode 8.3 beta 2.
4. Click the On/Off switch to enable Xcode Server.

For additional information, see Xcode Server and Continuous Integration Guide.

Technical Support and Learning Resources

Apple provides the following web resources to support your development with Xcode:

- Apple Developer Forums. Participate in discussions about developing for Apple platforms and using developer tools.
- Bug Reporter. Report issues, enhancement requests, and feedback to Apple. Provide detailed information, including the system and developer tools version information, and any relevant crash logs or console messages.
- Apple Developer website. Get the latest development information as well as technical documentation for Xcode.
- Xcode homepage. Get high-level information about the latest release of Xcode. Download current and beta Xcode releases.

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

Swift 2.3 Deprecation

- Xcode 8.3 beta 2 no longer supports Swift 2.3. Please migrate your projects containing Swift 2.3 code to Swift 3 syntax by opening the project and choosing Edit > Convert > To Current Swift Syntax.

Other Deprecations and Removal Notices

- OS X 10.11 was the last major release of macOS that supported the previously deprecated garbage collection runtime. Applications or features that depend upon garbage collection may not function properly or will not launch in macOS Sierra. Developers should use Automatic Reference Counting (ARC) or manual retain/release for memory management instead. (20589595)
- `Debugger()` and `DebugStr()` are deprecated and the scheme editor no longer provides the option to enable these functions. If your project uses these functions, enable them by setting the environment variable `USERBREAK` with a value of `1`. (24631170)
- The Automation instrument has been removed from Instruments. Use Xcode's UI Testing in its place. (26761665)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

New in Xcode 8.3 beta 2 – IDE

Debugging

- The view debugger now shows creation backtraces for many objects in the Object inspector. To show the backtraces, enable Malloc with the option "All Allocation and Free History" logging in the Diagnostics tab of the Manage Schemes dialog . (28715704)

Simulator

- The `simctl get_app_container` command can now return the path of an app's data container or App Group containers. (30224453)

Organizer

- The Xcode Organizer now supports exporting tvOS apps for Enterprise distribution. (30000839)

Testing

- Added the `XCUISiriService` class to XCTest for writing tests which activate Siri with a voice recognition string, and queries for elements in the Siri UI. Use the class to write UI tests for Intents and Intents UI extensions. (29958123)

FileMerge

- The FileMerge application now uses the same text infrastructure as Xcode. File navigation and syntax coloring are more consistent with Xcode. (8838481)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

Resolved in Xcode 8.3 beta 2 – IDE

Debugging

- Running a watchOS Intents extension will automatically trigger Siri for a voice command. (297144170)

Interface Builder

- Objects from the object library will be visible in the watchOS canvas during drag operations. (30108277)
- Typing in the Model Key Path field for a Cocoa binding no longer hangs Xcode. (26487347)
- The top of the library content no longer gets shifted under the header bar, which previously resulted in clipping. (27601428)
- Connecting @IBAction methods from IB to Swift code now works when the `sender` parameter type is `Any`. (29639217)

Testing

- XCTWaiter.h no longer produces a warning when modules are disabled and it is imported with the `-Weverything` flag. (30182239)

Simulator

- Resolved an issue where Siri on a simulated watchOS device did not respond after the first time Siri was enabled on a simulated iOS device. (29708878)
- Launching an application or executing tests should no longer generate "Application `<bundle>` is installing or uninstalling, and cannot be launched" errors in Simulator. (30172453)

Core Data

- Passing `-showBuildSettings` to `xcodebuild` for a project containing a Core Data model should no longer result in `xcodebuild` hanging. (26749142)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

Known Issues in Xcode 8.3 beta 2 – IDE

Debugging

- Siri may time out when debugging an intents extension for the first time. (30083076)
  Workaround: If this occurs, invoke Siri again in the same debugging session.

Simulator

- Siri may always ask for permission for location requests, ignoring the permission settings. Granting location permission to Siri has no effect. (28885210)
  Workaround: From the phone, approve Location Services for Siri. Then, use Siri on the Watch.
- Multiple entries for Siri may show up in Settings > Privacy > Location Services. Enable all entries to allow Siri to use the simulated user location. (29931932)

Instruments

- Time Profiler in Instruments has been disabled when targeting Simulator. (30318976)

  Workaround: Launch your app in Simulator manually, then target your Mac in Instruments using "All Processes". Filter the call tree view to focus on just your app as appropriate.

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

New in Xcode 8.3 beta 2 - Swift Compiler

Swift Compiler

- The Swift compiler can now automatically precompile Objective-C bridging headers, which can speed up Debug configuration builds (or other non-WMO builds) of mixed-source projects with large bridging headers. This feature is still experimental, and is disabled by default but can be enabled with the "Precompile Bridging Header" (`SWIFT_PRECOMPILE_BRIDGING_HEADER`) build configuration setting within Xcode. (29016063)
- Swift will now warn when an `NSObject` subclass attempts to override the class `initialize` method. Swift doesn't guarantee that references to class names trigger Objective-C class realization if they have no other side effects, leading to bugs when Swift code attempts to override `initialize`. (28954946)
- C functions that "return twice" are no longer imported into Swift and made explicitly unavailable. Attempting to reference a function will result in a compilation error.

  Examples of functions that "return twice" include `vfork` and `setjmp`. These functions change the control flow of a program in ways that that are supported in Swift. For example, definitive initialization of variables, a core Swift language feature, can not be guaranteed when these functions are used.

  Swift code that references these functions will no longer compile. Although this could be considered a source-breaking change, it's important to note that any use of these functions will most likely crash at runtime. Now, the compiler will prevent them from being used. (SR-2394)
- Indirect fields from C structures and unions are now always imported. Previously they weren't imported when they belonged to an union. This is done by naming anonymous fields. For example:

```
typedef struct foo_t {
  union {
    int a;
    double b;
  };
} foo_t;
```

  Is imported as:

```
struct foo_t {
  struct __Unnamed_union___Anonymous_field0 {
    var a : Int { get set }
    var b : Double { get set }
  }
  var __Anonymous_field0 : foo_t.__Unnamed_union___Anonymous_field0

  // a and b are computed properties accessing the content of __Anonymous_field0
  var a : Int { get set }
  var b : Double { get set }
}
```

  Since new symbols are exposed from imported structure/unions, this may conflict with existing code that extended C types in order to provide their own accessors to the indirect fields. (30221279)

Swift Language

- A new family of failable conversion initializers to all numeric types will either complete successfully without loss of information or return `nil`. (SE-0080)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

Swift Package Manager

- The package manager no longer uses the auto-linking feature of module maps for C language targets. Packages which use custom module maps for their C language targets should remove the link statement from the module map. The package manager will automatically add the link flag if required.

  For example, the following custom module map:

  ```
  module MyCLib {
     header "foo.h"
     link "MyCLib"
     export *
  }
  ```

  Should be changed to:

  ```
  module MyCLib {
     header "foo.h"
     export *
  }
  ```

  (30016942)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

Resolved Issues in Xcode 8.3 beta 2 - Swift and Apple LLVM Compilers

Linking

- Incremental Link-Time Optimization works properly with bitcode-enabled apps. (30109087)

Swift Compiler

- The Swift compiler correctly reports itself as Swift 3.1. (30081411)
- Swift REPL and interpreted scripts correctly determine the version of the OS that they are running on. (29433205)
- `#keyPath` respects the original name of properties imported from Objective-C. (28543037)
- The generated header for a mixed-source Swift target will now correctly reference Dispatch types using the usual typedefs (`dispatch_queue_t`) rather than their implementation details. (29790636)
- Properties that satisfy requirements in Objective-C protocols work properly when they are declared with a `willSet` or `didSet` observing accessor. (30101703)

Swift Package Manager

- When a dependency of a package is in edit mode, operations which trigger dependency resolution (such as `swift package update`) will fail. (29944464)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

Known Issues in Xcode 8.3 beta 2 - Swift Compiler

Swift Compiler

- Swift 3.1 includes a new warning of the form "implicit import of bridging header … via module … is deprecated and will be removed in a later version of Swift". This warning is reported too frequently in contexts where it may not be easy to correct. (30202509)
- AppKit's `NSView.animator()` method claims to return the `Self` type but really returns a proxy object. When called on a Swift subclass, this may lead to runtime crashes when trying to reference animatable properties of the view that were overridden in Swift:

```
public class MyView: NSView {
    public override var frame: NSRect {
           didSet {
                  Swift.print(self.frame)
           }
    }

}

let view = MyView()
view.animator().frame = NSMakeRect(0.0, 0.0, 120.0, 120.0) // crashes
```

Workaround: Cast the result of `view.animator()` to `AnyObject`, then do method dispatch on the `AnyObject` value to ensure that the animator proxy is invoked with Objective-C messaging instead of Swift messaging. For example:

```
(view.animator() as AnyObject).setFrame(NSMakeRect(0.0, 0.0, 120.0, 120.0))
```

(27655718)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

New in Xcode 8.3 beta – IDE

General

- Intents Extensions are available on watchOS. (28372004)

Debugging

- Autocompletion is supported in Breakpoint navigator text fields. Completion information for a file breakpoint uses information in the referenced file. Other breakpoints use general indexing information and may be slower. (28268983)
- The scheme editor now contains a text field to add an Intents Extension query instead of issuing an utterance vocally. (26456789)
- Added the ability to visually debug the view hierarchy of an iOS iMessage extension that is running on Simulator or an iOS device which is running iOS 10.3 or later. (28838632)

Testing

- Added `XCUIKeyModifierCapsLock` as a preferred alternative for `XCUIKeyModifierAlphaShift`. This aligns with `NSEventModifierFlagCapsLock` added in macOS 10.12, which replaces `NSAlphaShiftKeyMask`. (27375799)
- XCTest adds new APIs to simplify writing asynchronous tests. A new class, `XCTWaiter`, is used to wait for an `XCTestExpectation` object independent of any `XCTestCase` instance, making it possible to wait inside helper functions and to test support libraries. New subclasses of `XCTestExpectation` are used for tests that wait for `NSNotifications`, Key Value Observation, `NSPredicate` evaluations, and Darwin notifications. `XCTestExpectation` has a new writable description property to simplify failure diagnosis in asynchronous tests. Finally, `XCTestExpectation` adds two new behaviors. Set `.inverted` to `true` to fail if the expectation is fulfilled. Set `.expectedFulfillmentCount` to specify the required number of calls to `fulfill()`. See documentation in XCTWaiter.h, XCTestExpectation.h, XCTNSPredicateExpectation.h, XCTNSNotificationExpectation.h, XCTKVOExpectation.h, and XCTDarwinNotificationExpectation.h for more details. (22539853)
- The Test navigator now includes test classes and methods located in static libraries that are linked into test bundles. Note that it may be necessary to activate the test navigator before test diamonds for these tests appear in the source editor. (28173131)

Source Editor

- Xcode Source Editor Extensions will now assert if you try to create a command with an identifier containing invalid characters (characters outside the range 0-9, A-Z, a-z, -, or .). (26615933)

Provisioning

- Changed the user interface for managing signing certificates and provisioning profiles. Certificates are managed from the Accounts preferences pane by selecting a team and clicking Manage Certificates. Automatically managing signing is recommended, however if your app requires manually signing provisioning profiles are managed in the General tab of the project editor. Use the Provisioning Profile dropdown to import or download profiles. In addition it displays profiles that match the current signing configuration of the target. (28641027)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

Simulator

- The `simctl` tool of the xcrun command line tool has two new commands that provide additional information for filing bug reports.
  `simctl logverbose` enables verbose logging on the specified device.
  `simctl diagnose` collects all relevant Simulator logs on the host and on the device, and then packages them into an archive that can be attached to bug reports. For more information, see the help for each command by running `xcrun simctl <tool name> help` on the command line. (21001891)
- You can invoke Siri using Hardware > Siri after enabling Siri in the Settings app on Simulator. (25176537)

Instruments

Inspector Refresh

Instruments previously contained three inspector panes: Record Settings, Display Settings, and Extended Detail. There are now two inspector panes: Extended Detail and Run Info. Extended Detail remains unchanged and continues to provide additional information about selected data in the detail pane. The new Run Info inspector pane displays information about the currently selected run, including the recording device information and recording settings when the run was taken.

Record Settings are now located in a Recording Options window. To display this window, choose File > Recording Options, press Option-Command-R, or click-and-hold the Record button in the toolbar and choose Recording Options from the menu that appears. The Recording Options window includes global and instrument-specific recording options.

The Display Settings inspector pane was previously used to change the graphing style of the timeline pane and to perform filtering in the detail pane. These features have been moved to new locations. If an instrument supports multiple graphing options, a down arrow now appears beside the instrument's icon when the pointer is positioned above the instrument in the timeline pane. Click the instrument's icon to open a popover containing editable graphing options. The down arrow appears only for instruments that have graphing options. Detail pane filtering options, including the text filter field previously located in the navigation bar of the detail pane, are now found at the bottom of the detail pane.

Visual Run Comparison

The ability to view graphs for multiple runs at the same time in the timeline pane has been removed. To switch between runs, choose Previous Run or Next Run from the Instruments menu, or click a navigation arrow in the activity viewer in the toolbar. (28503173)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

Resolved in Xcode 8.3 beta – IDE

General

- Fixed an issue which caused the "update to recommended settings" warning to remain after updating to the recommended settings. (28447448)
- Fixed a problem which would cause Xcode to crash when removing certain file or framework references from certain projects. (29118448)

Interface Builder

- Using `NSColorPickerTouchBarItem` in Interface Builder no longer creates a Touch Bar item that is disabled at runtime. (28670596)
- Fixed a bug with inspector checkboxes that correspond to bitmask properties, such as `NSWindow.styleMask`, `UIAccessibility.accessibilityTraits`, and `UIPopoverPresentationController.permittedArrowDirections`. (26613034)
- Update frames works with ambiguous views that have an unambiguous horizontal or vertical axis. (16019988)
- Option for "Follow Readable Width" has been restored to Interface Builder's View Controller size inspector. (28246180)

Testing

- Fixed a bug in the handling of `XCTestExpectations` that waited on predicates evaluated against `XCUIElements`. As a result, some tests which were previously succeeding despite ambiguous (incorrect) queries will start to fail. If you start to see a UI test fail because of "Multiple matches found" and the element in question is the subject of a predicate expectation earlier in the test, your test is failing because of this fix and the element query will need to be updated to correct the ambiguity that is leading to multiple matches. (28139563)

Organizer

- Speed improvements have been made to the Organizer window. Opening the Organizer with many Archives and many iTunes Connect Products should open quickly. (28722427)

Simulator

- Videos recorded by the `simctl` tool of the xcrun command line tool now support scrubbing and import into video editors correctly. (29654098)

Build System

- When copying files the build system now preserves the extended attributes ('`xattrs`') used by `codesign` for signing standalone files. (29453501)
- The Mac will no longer sleep when a build is in progress. (29256278)
- Xcode no longer crashes when the build system copies a file with an invalid code signature. (29656173)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

- Changes to `xcconfig` files no longer require restarting Xcode to take effect. (29805284)
- Build issues from targets that are implicit dependencies of the active scheme are now shown in the issue navigator. (28719580)
- `xcodebuild` no longer hangs when invoked with a project or workspace containing no schemes. (20450296)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

New in Xcode 8.3 beta - Swift and Apple LLVM Compilers

Apple LLVM Compiler

- The Apple LLVM compiler now supports nullability annotations on function and method parameters with array type, and provides additional warnings for incomplete and suspicious nullability annotations. (25846421)

Swift Language

- The Swift compiler incorrectly reports a version of Swift 3.0. This may affect compiler directives which use the compiler version number as an argument, such as `#if swift(>=3.1)`. (30081411)
- Xcode 8.3 includes Swift 3.1 which is intended to be source compatible with Swift 3.0. If you find instances where your Swift 3.0 code no longer compiles, please file a bug report and include the affected code.
- Extended the `@available` attribute to include varying API availability by Swift language version. An annotated API is compiled as usual and added to the library, but is made available only to a client built with a matching compiler version or a matching `-swift-version` when built for version compatibility. (SE-0141)
- The `Sequence` protocol adds two new members, `prefix(while:)` and `drop(while:)`. `prefix(while:)` requests the longest subsequence satisfying a predicate. `drop(while:)` requests the remaining subsequence after dropping the longest subsequence satisfying a predicate. (SE-0045)
- Constrained extensions allow same-type constraints between generic parameters and concrete types. For example, the following code defines an extension on `Array` with `Int` elements:

  ```
  extension Array where Element == Int { }
  ```

  (SR-1009)
- Nested types may now appear inside generic types, and nested types may have their own generic parameters:

  ```
  struct OuterNonGeneric {
      struct InnerGeneric<T> {}
  }
  struct OuterGeneric<T> {
      struct InnerNonGeneric {}
      struct InnerGeneric<T> {}
  }
  extension OuterNonGeneric.InnerGeneric {}
  extension OuterGeneric.InnerNonGeneric {}
  extension OuterGeneric.InnerGeneric {}
  ```

  (SR-1446)

Swift Package Manager

- The Swift Package Manager now supports an edit mode for a package's dependencies. Putting a dependency in edit mode moves it from the managed dependency location to a local `Packages` directory and omits it from `swift package update` and related commands, allowing the user to manually manage checkout of the dependency (e.g. by committing local fixes they need). By default, dependencies are no longer stored in the `Packages` directory unless they are in edit mode. A package can be put in edit mode with the `swift package edit` command, and can leave edit mode with the `swift package unedit` command. (SE-0082)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

- The Swift Package Manager now supports pinning a dependency to a specific version, so that in a clean build, it resolves only to the pinned version. The `swift build` command will automatically record the version that is used for every dependency in a `Package.pins` file, unless autopinning is disabled with `swift package pin --disable-autopin`. Commit the `Package.pins` file to source control when you expect your package to use specific dependency versions without requiring those versions in your `Package.swift` manifest. The `Package.pins` file is used during initial dependency resolution to determine which dependency versions to use; `swift package update` will update all dependency versions and repin those dependencies to the new versions. If autopinning is off, pass the `--repin` flag to `swift package update` to override any manually-pinned versions. Pinning can be further controlled with the `swift package pin` and `swift package unpin` commands. (SE-0145)
- Replaced the `swift package clean=dist` command with `swift package reset` which resets a package back to a clean state, with no dependencies checked out or build artifacts present. (28384606)
- The `swift package test` command has a new `--parallel` option which will execute tests in parallel. This mode of testing may cause problems for tests which are not written to be able to safely run in parallel. (28402178)

Static Analyzer

- Added a check for synthesized `copy` properties of mutable types, such as `NSMutableArray`. Calling the setter for these properties will store an immutable copy of the value. (21022397)
- Added a check for calls to `dispatch_once()` that use an Objective-C instance variable as the predicate. Using an instance variable as a predicate may result in the passed-in block being executed multiple times or not at all. These calls should be rewritten either to use a lock or to store the predicate in a global or static variable. (23390767)
- Added a check for unintended comparisons between scalar values and `NSNumber`, `CFNumberRef`, and other number objects. (10228523)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.

Resolved Issues in Xcode 8.3 beta - Swift and Apple LLVM Compilers

Swift Compiler

- The LLVM and Swift compilers now emit correct diagnostics when using a custom module map that contains system headers (such as a module map with paths into the macOS SDK while building for iOS or headers from /usr/include). (26866326)
- The Swift compiler now raises an error for modifications of a `let` property or variable of protocol type after initialization. For example, the following code that compiled in previous versions will now raise an error:

```
protocol P {
  mutating func f()
}

extension Int: P {
  mutating func f() { self += 1 }
}

func foo(x: Int) {
  let y: P
  y = x
  y.f()
}
```

  Code that successfully compiled in previous releases may fail after upgrading. (29716016)

Debugging

- Activity and trace messages normally visible in LLDB thread summaries and thread detailed information are available when debugging processes running on beta versions of macOS, iOS, tvOS, or watchOS. (26370425)

Linking

- Incremental LTO performance is improved overall including fixing an issue that was creating duplicate cache files. (29526668)

Swift Package Manager

- Rebuilding a package in Swift Package Manager now avoids performing unnecessary work. (28037508)
- `swift package update` incrementally updates dependencies instead of re-cloning them from scratch. In addition, complex dependencies are resolved correctly. (26371453)

Copyright © 2017 Apple Inc. All rights reserved. Terms of Use. Privacy Policy.