#WWDC19

Binary Frameworks in Swift

Harlan Haskins, Swift Team
Jordan Rose, Swift Team

Swift Packages

Swift Packages

Swift Packages

Dependencies managed by Xcode

Swift Packages

Dependencies managed by Xcode

Versions managed by Xcode

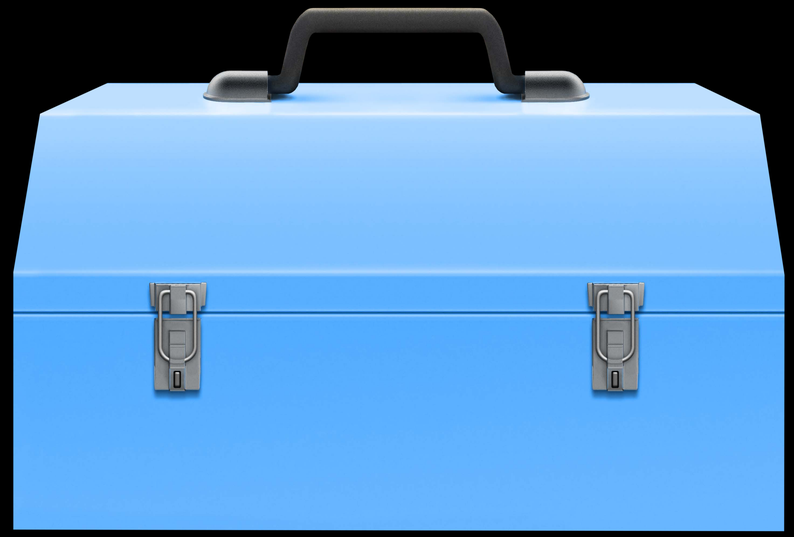
Swift Packages

Dependencies managed by Xcode

Versions managed by Xcode

No binary compatibility requirements

Swift Packages



XCFrameworks

Agenda

Agenda

Introducing XCFrameworks

Agenda

Introducing XCFrameworks

Creating an XCFramework

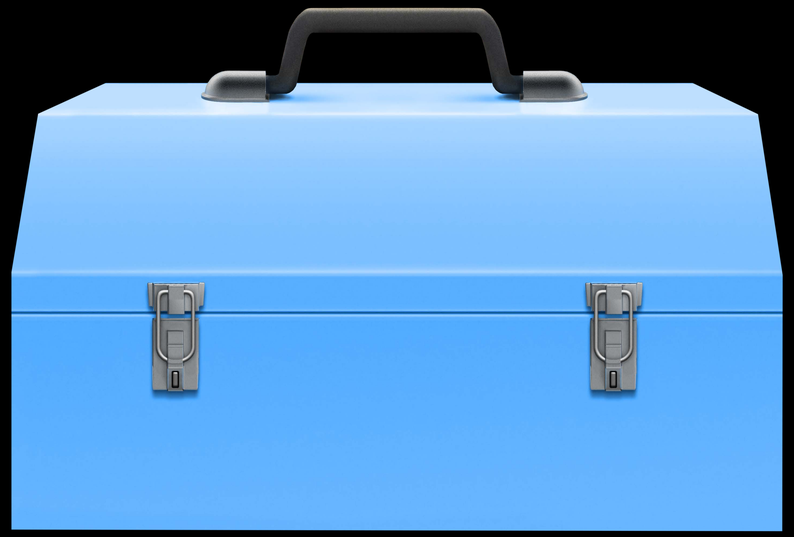
Agenda

Introducing XCFrameworks

Creating an XCFramework

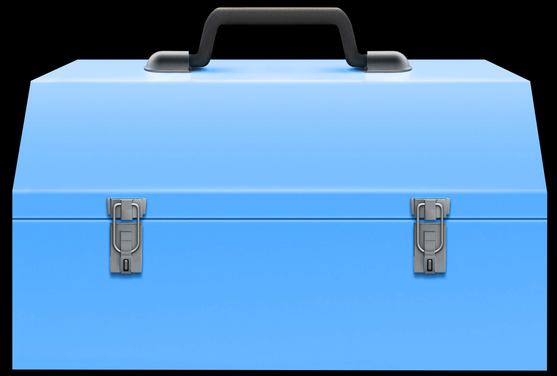
Framework author considerations

NEW

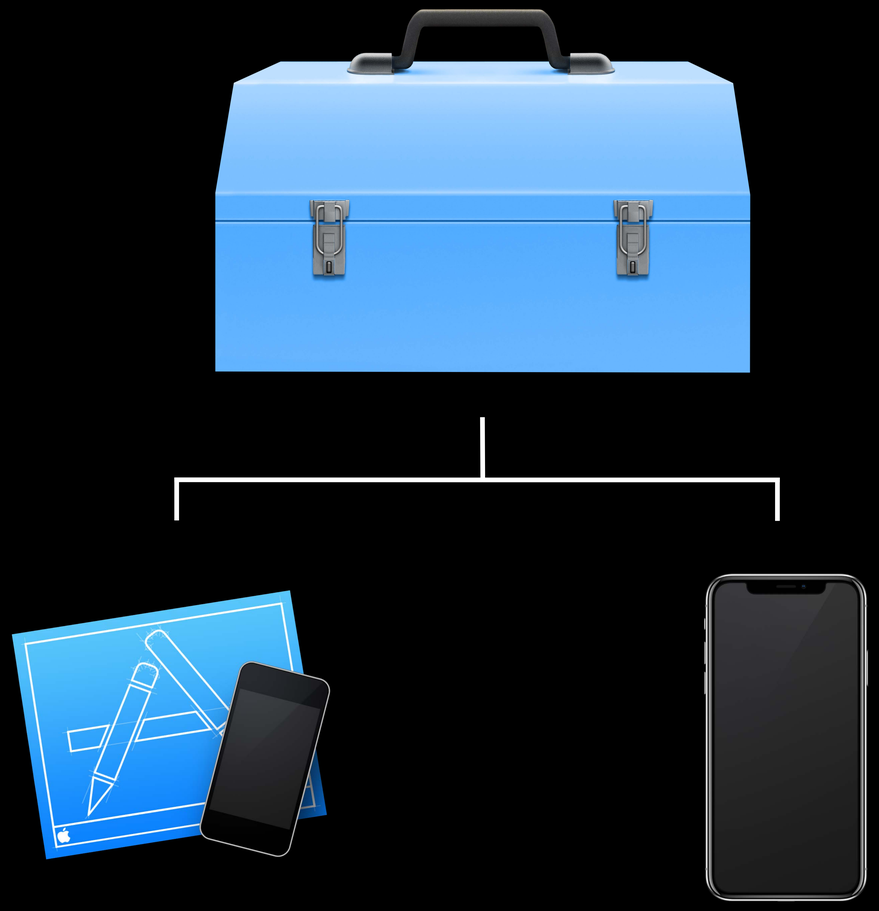


XCFramework

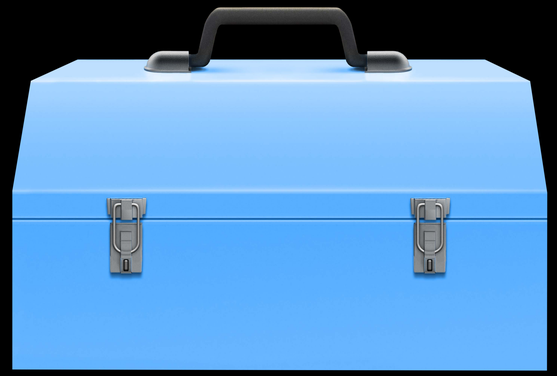
NEW



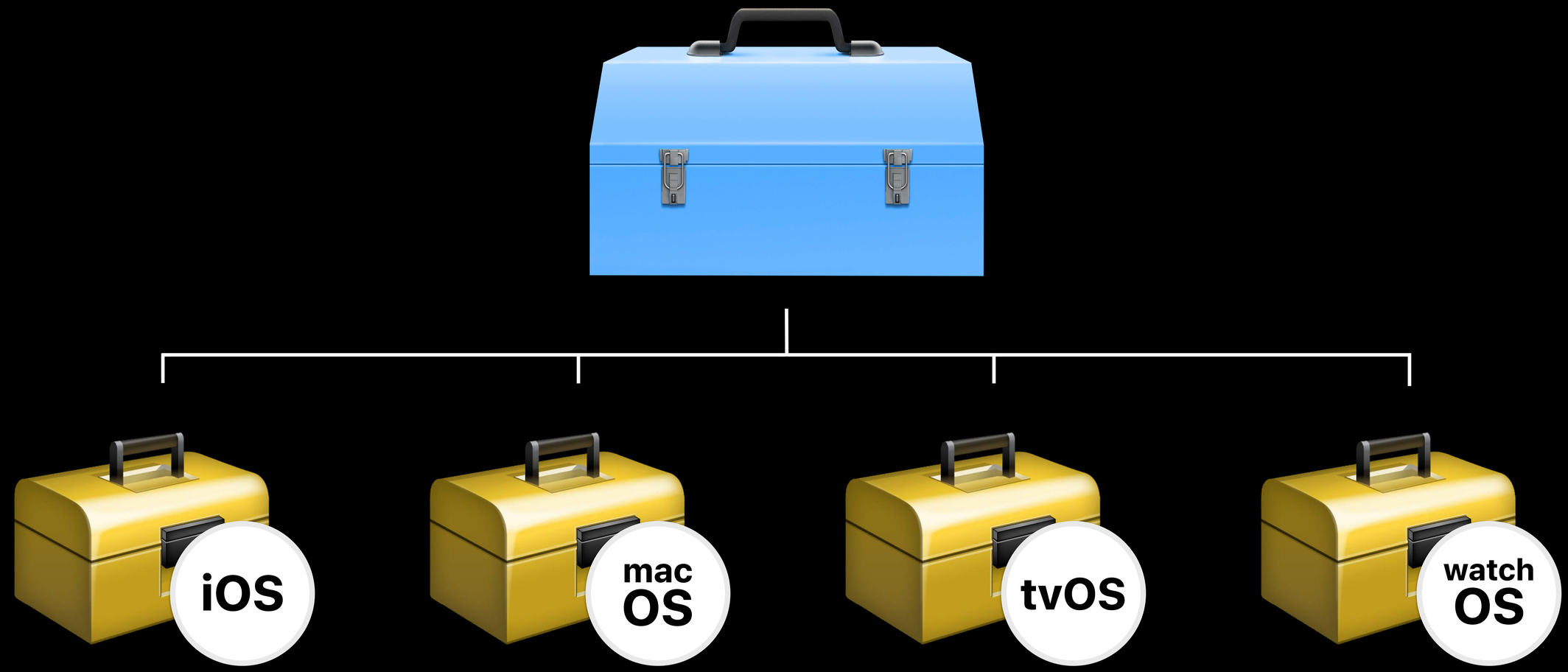
NEW



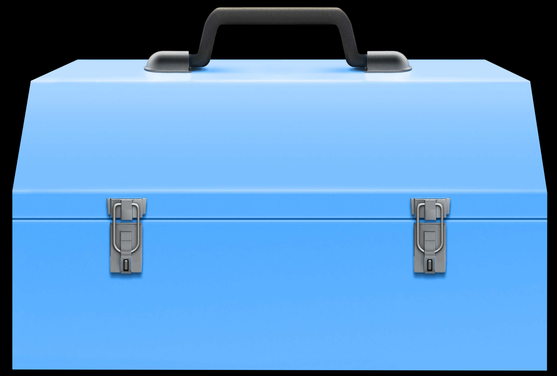
NEW



NEW



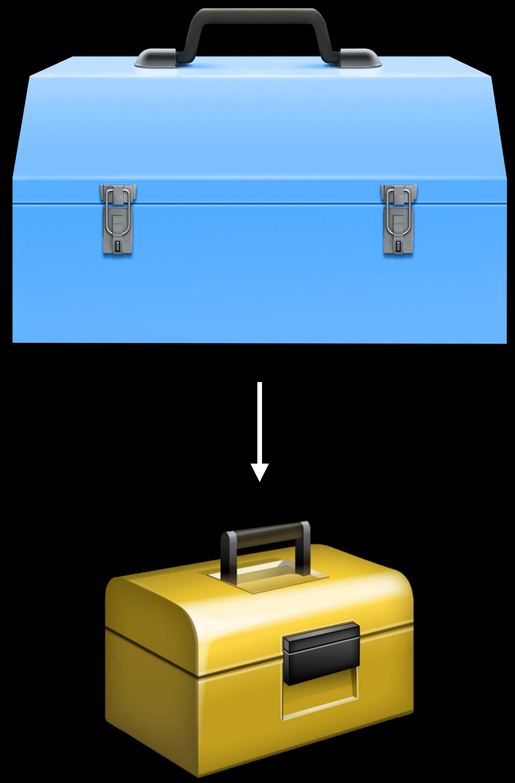
NEW



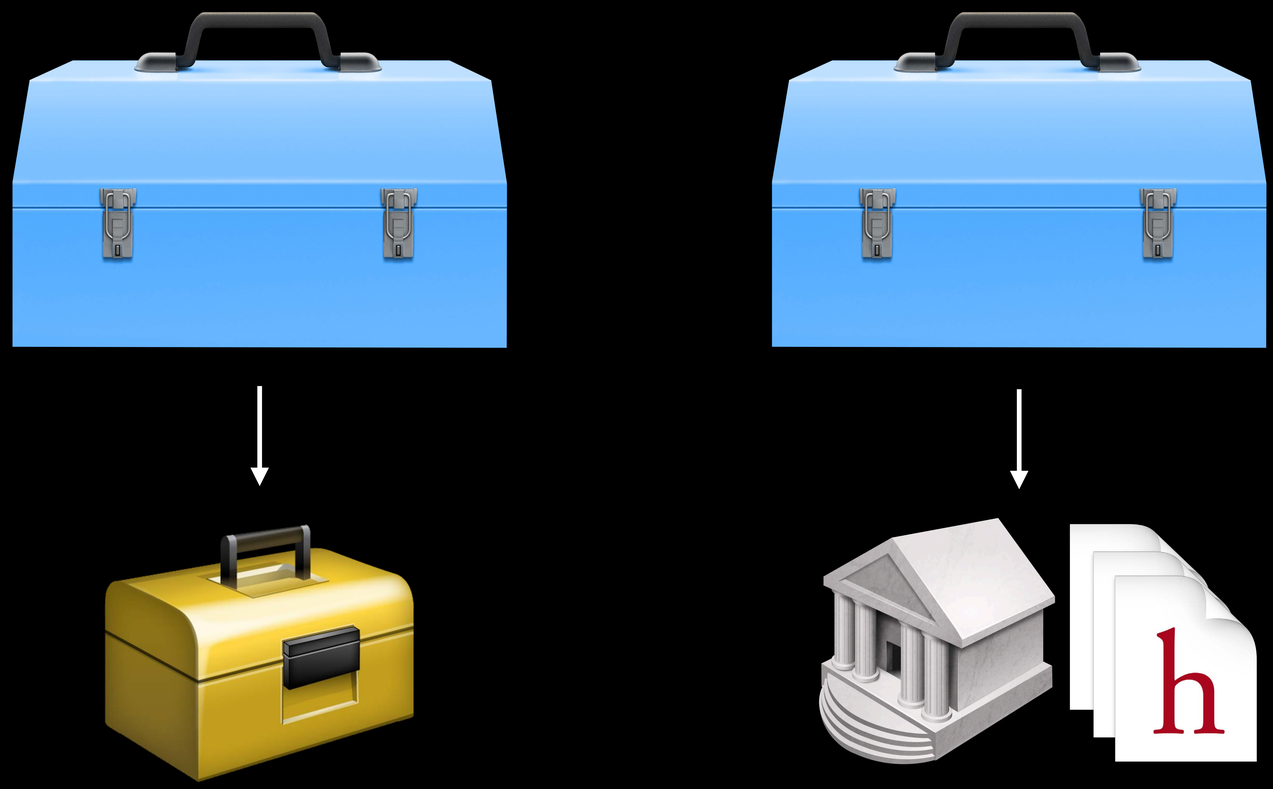
NEW

NEW

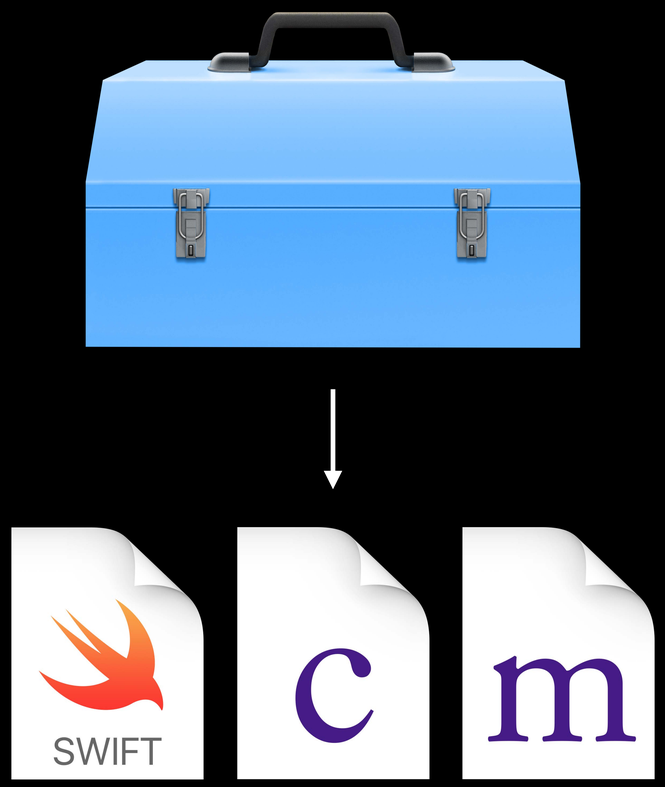
NEW



NEW



NEW

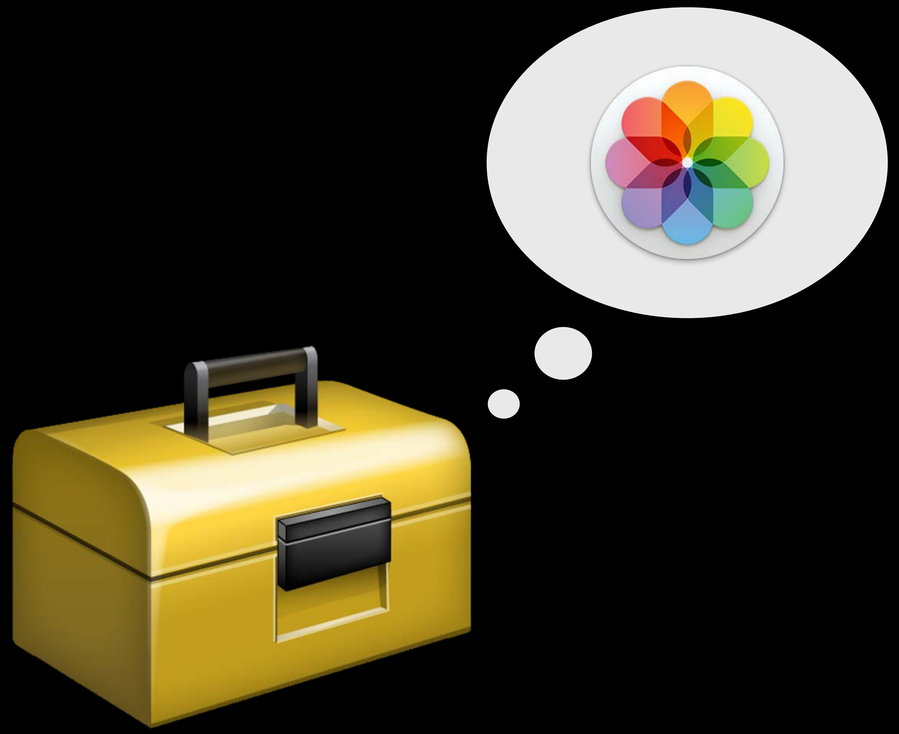


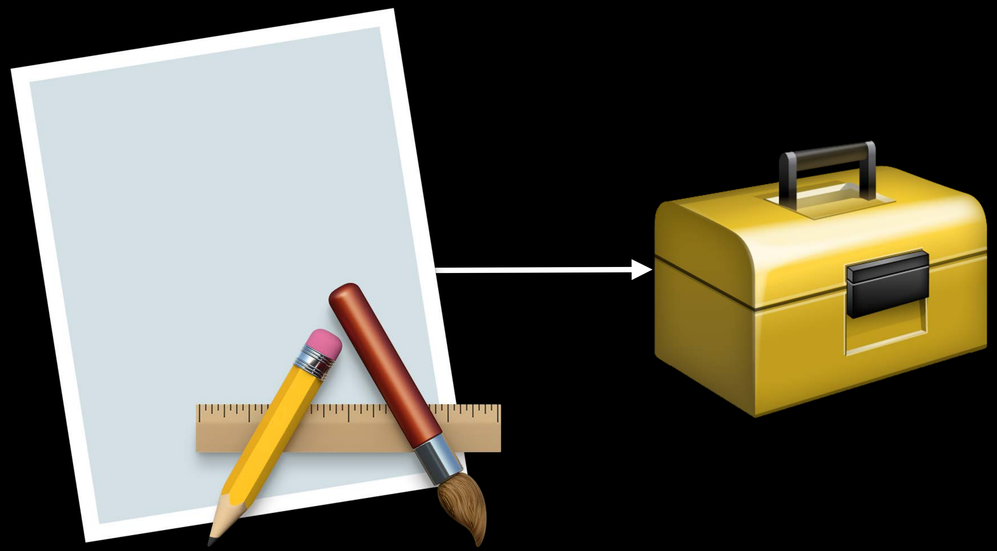
Demo

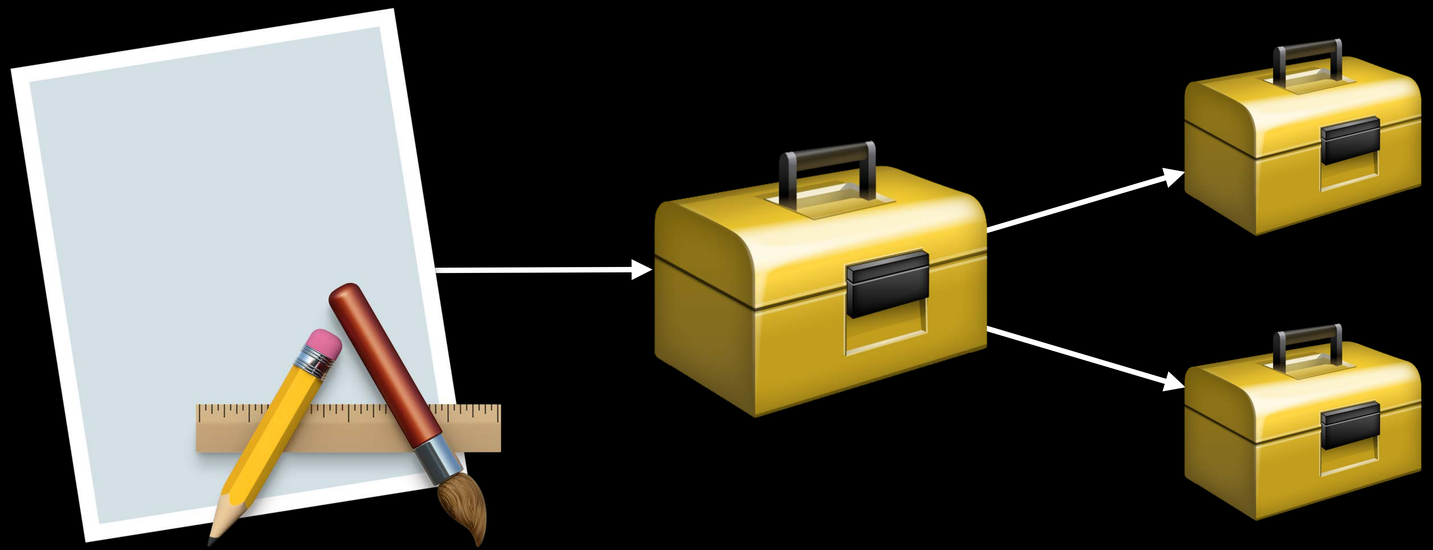
Using an XCFramework

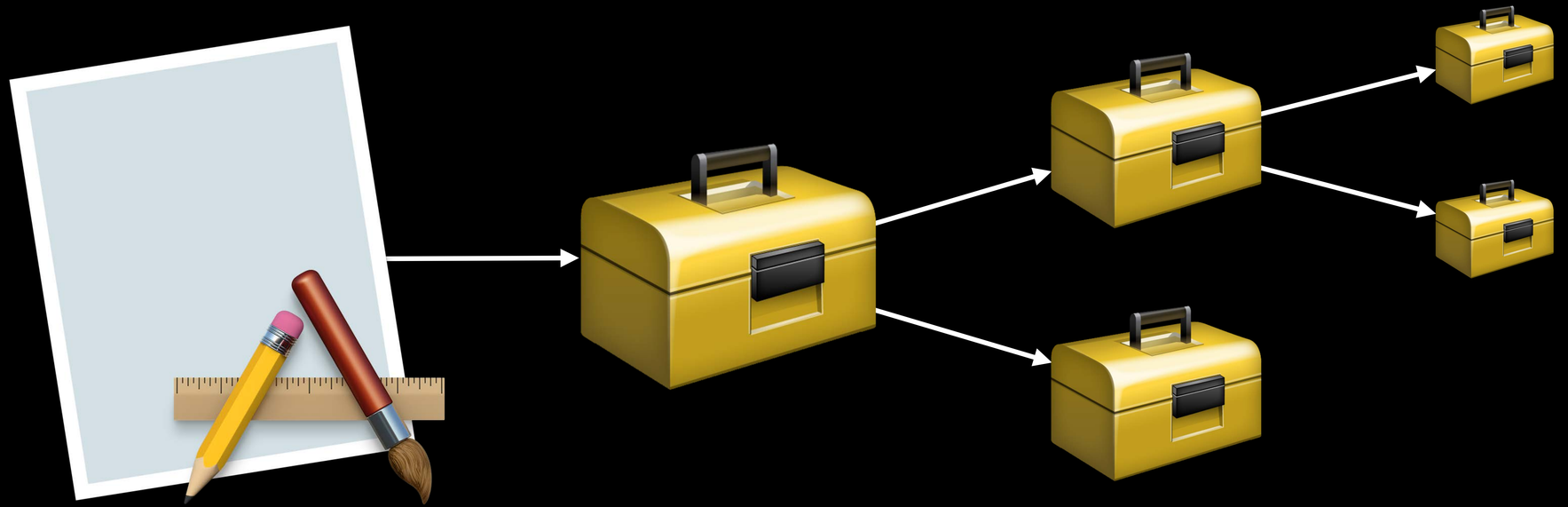
Using Frameworks

Trust









Adopting Swift Packages in Xcode	WWDC 2019
Creating Swift Packages	WWDC 2019


XCODE

Creating an XCFramework

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

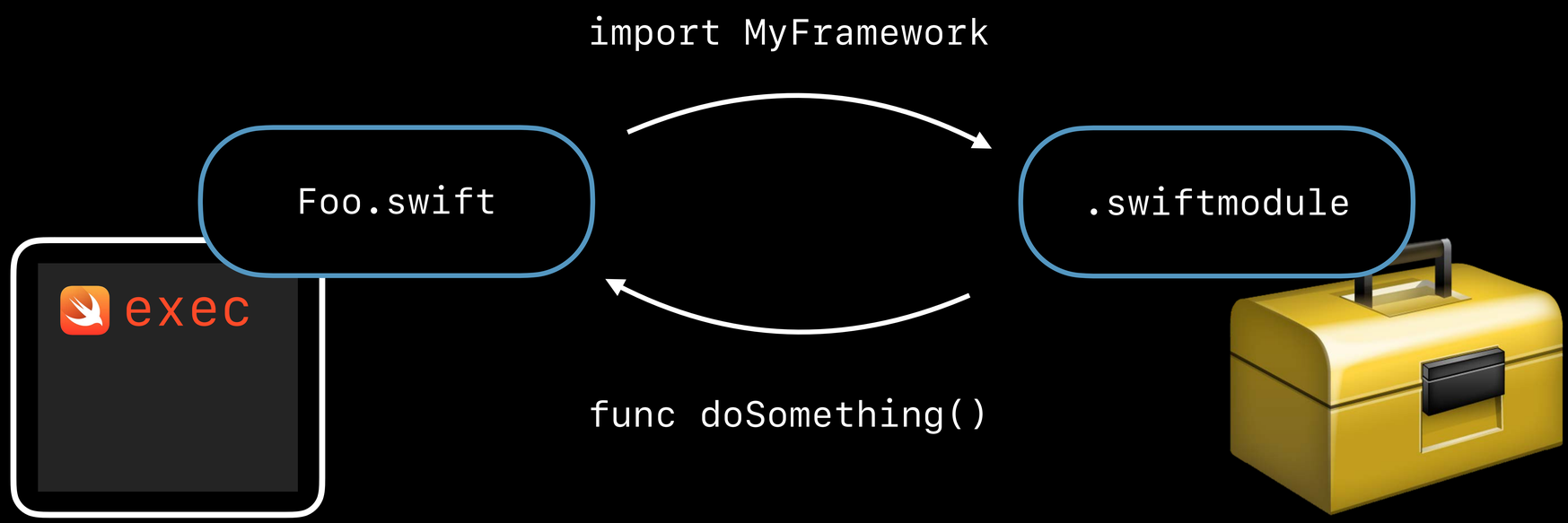
New Build Setting in Xcode 11

NEW

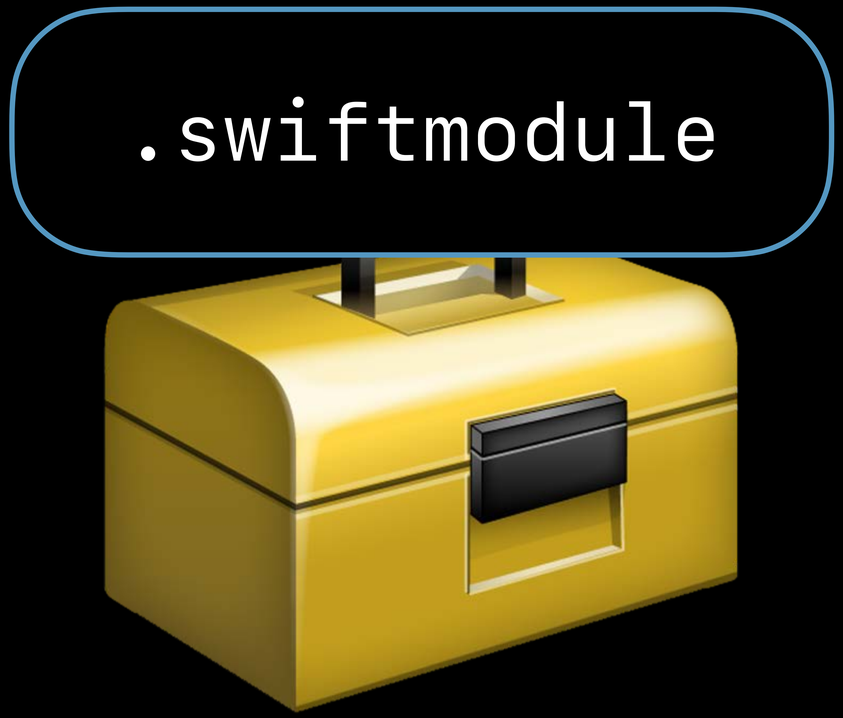
Compiled module was created by a newer version of the compiler

Importing a Module

Importing a Module

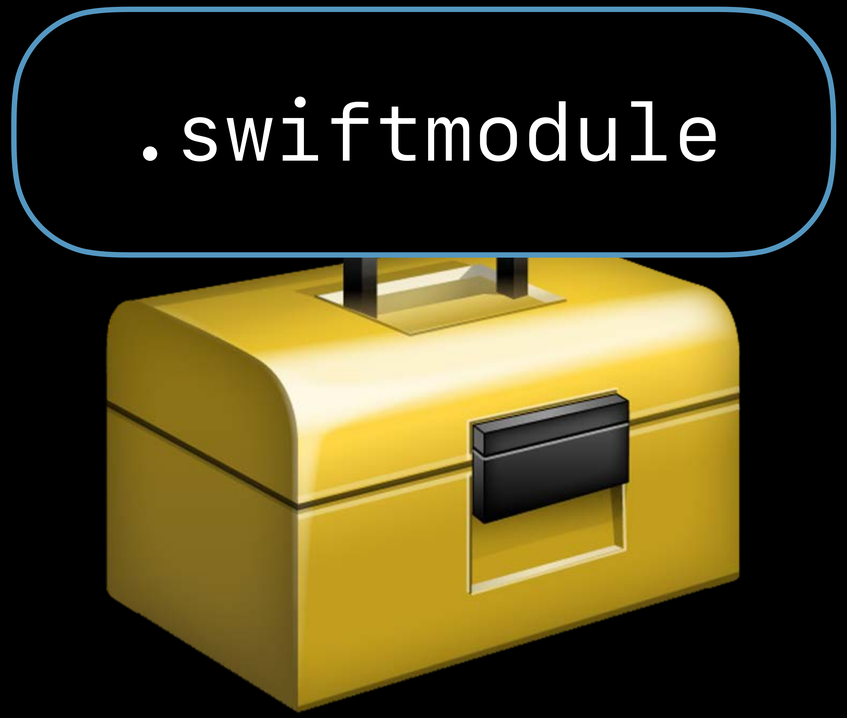


Compiled Module Format



Compiled Module Format

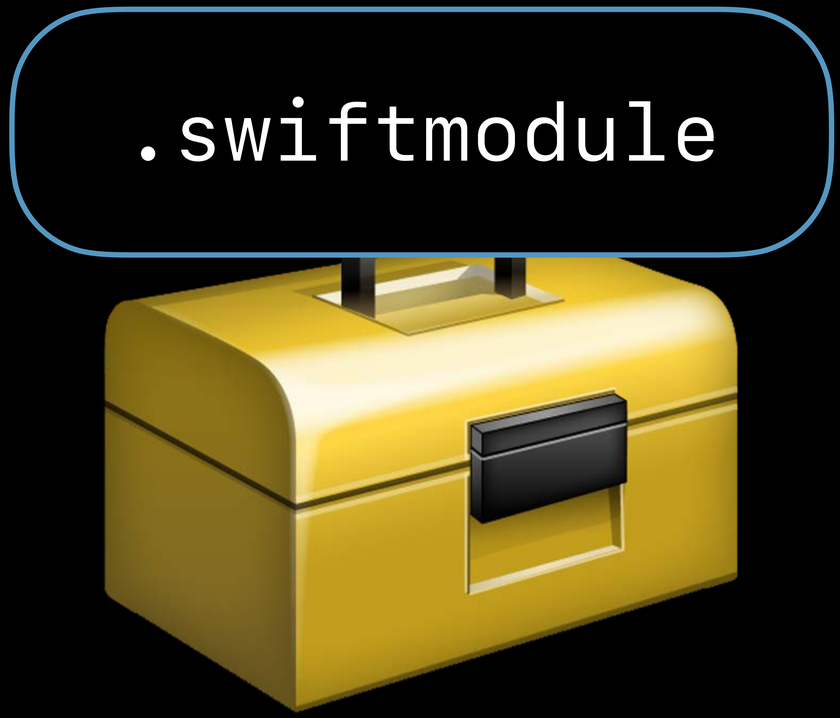
Serialized, binary format



Compiled Module Format

Serialized, binary format

Internal compiler data structures



Compiled Module Format

Serialized, binary format

Internal compiler data structures

Changes between compiler versions

Swift Module Interfaces

NEW

Swift Module Interfaces

NEW

Textual listing of public API

Swift Module Interfaces

NEW

Textual listing of public API

Compatible across compiler versions

Swift Module Interfaces

NEW

Textual listing of public API

Compatible across compiler versions

Enabled with “Build Libraries for Distribution”

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```swift
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```swift
// swift-interface-format-version: 1.0
// swift-compiler-version: Swift version 5.1
// swift-module-flags: -target arm64-apple-ios13.0
                       -enable-library-evolution
                       -swift-version 5 -O
                       -module-name FlightKit

import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
```

```swift
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```swift
// swift-interface-format-version: 1.0
// swift-compiler-version: Swift version 5.1
// swift-module-flags: -target arm64-apple-ios13.0
                       -enable-library-evolution
                       -swift-version 5 -O
                       -module-name FlightKit

import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
```

```swift
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
// swift-interface-format-version: 1.0
// swift-compiler-version: Swift version 5.1
// swift-module-flags: -target arm64-apple-ios13.0
                       -enable-library-evolution
                       -swift-version 5 -O
                       -module-name FlightKit
```

```swift
import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
```

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
-module-name FlightKit

import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}

public struct Location {
  public var coordinates: FlightKit.Coordinates
```

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
-module-name FlightKit

import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}

public struct Location {
  public var coordinates: FlightKit.Coordinates
```

```swift
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```swift
import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}

public struct Location {
  public var coordinates: FlightKit.Coordinates
}
```

```swift
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```swift
import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}

public struct Location {
  public var coordinates: FlightKit.Coordinates
}
```

```swift
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```swift
import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}

public struct Location {
  public var coordinates: FlightKit.Coordinates
}
```

```swift
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```swift
import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}

public struct Location {
  public var coordinates: FlightKit.Coordinates
}
```

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
-module-name FlightKit

import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}

public struct Location {
  public var coordinates: FlightKit.Coordinates
```

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
-module-name FlightKit

import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}

public struct Location {
  public var coordinates: FlightKit.Coordinates
```

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
-module-name FlightKit

import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}

public struct Location {
  public var coordinates: FlightKit.Coordinates
```

```
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
-module-name FlightKit

import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}

public struct Location {
  public var coordinates: FlightKit.Coordinates
```

```
public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
-module-name FlightKit

import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}

public struct Location {
  public var coordinates: FlightKit.Coordinates
```

```
public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}
```

```
public enum Speed {
  case leisurely
  case fast
}
```

```
public struct Location {
  public var coordinates: Coordinates
}
```

```
-module-name FlightKit

import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}
```

```
public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}
```

```
public struct Location {
  public var coordinates: FlightKit.Coordinates
```

```
public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
-module-name FlightKit

import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}

public struct Location {
  public var coordinates: FlightKit.Coordinates
```

```swift
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```swift
import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}

public struct Location {
  public var coordinates: FlightKit.Coordinates
}
```

```swift
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}
```

```swift
public struct Location {
  public var coordinates: Coordinates
}
```

```swift
import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
    b: FlightKit.Speed) -> Swift.Bool
  public func hash(
    into hasher: inout Swift.Hasher)
}
```

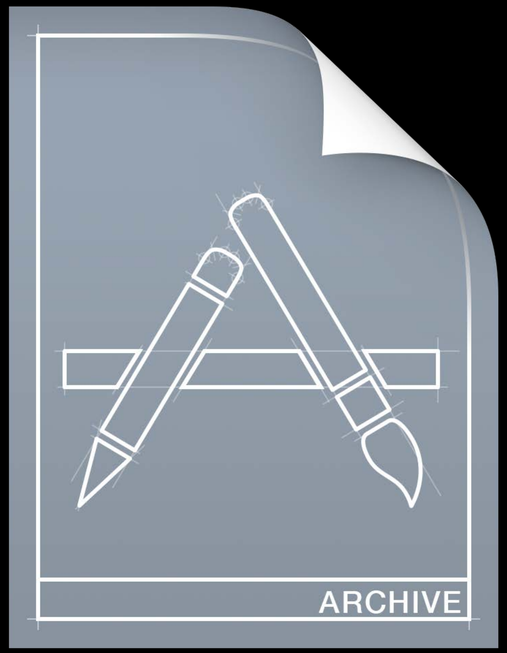
```swift
public struct Location {
  public var coordinates: FlightKit.Coordinates
}
```

```swift
import UIKit

public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```swift
// swift-interface-format-version: 1.0
// swift-compiler-version: Swift version 5.1
// swift-module-flags: -target arm64-apple-ios13.0
                       -enable-library-evolution
                       -swift-version 5 -O
                       -module-name FlightKit

import Swift
import UIKit

public class Spaceship {
  public let name: Swift.String

  public init(name: Swift.String)
  public func fly(
    to destination: FlightKit.Location,
    speed: FlightKit.Speed)

  @objc deinit
}

public enum Speed: Swift.Hashable {
  case leisurely
  case fast

  public static func ==(
    a: FlightKit.Speed,
```

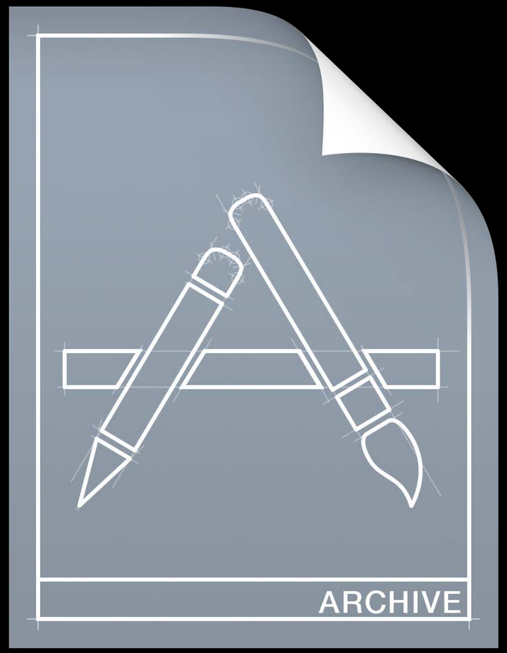
Building an XCFramework

Archiving Your Framework



Archiving Your Framework

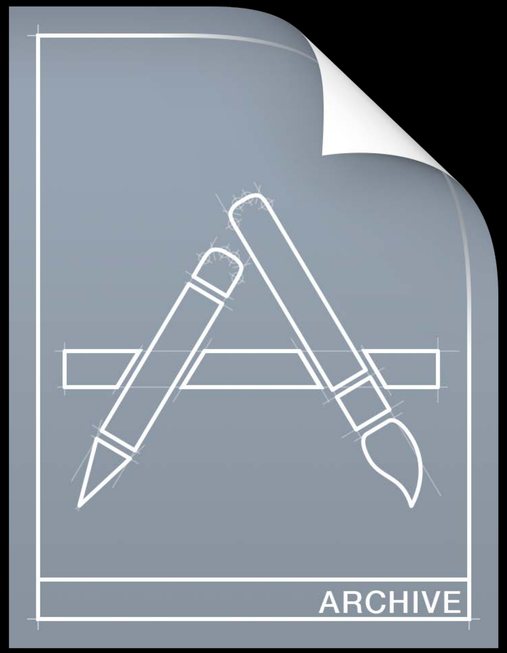
Builds framework in Release



Archiving Your Framework

Builds framework in Release

Available in Xcode Organizer

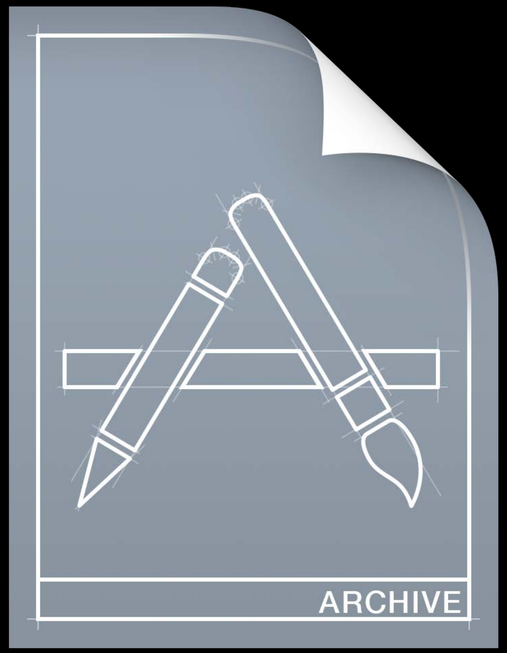


Archiving Your Framework

Builds framework in Release

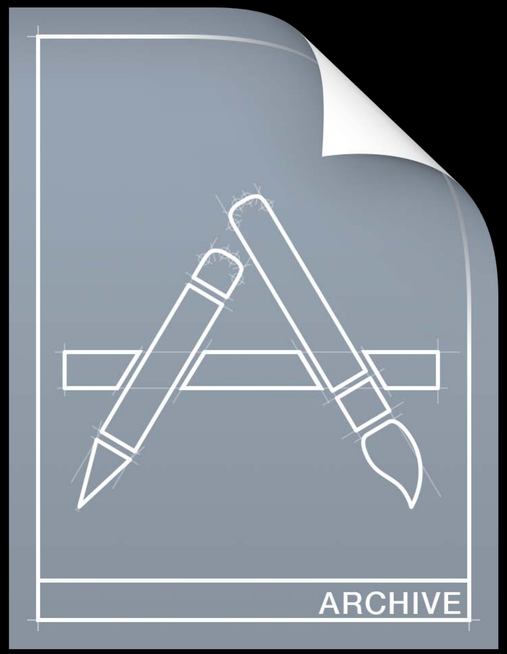
Available in Xcode Organizer

Keeps track of dSYMs



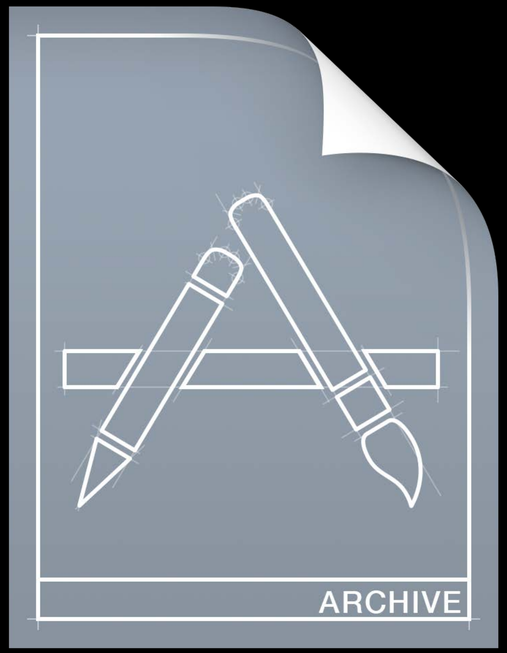
Archiving Your Framework

```
xcodebuild archive
```



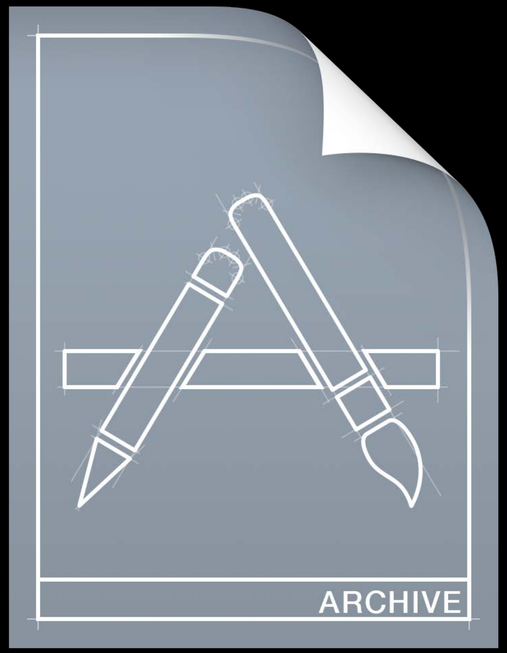
Archiving Your Framework

```
xcodebuild archive
   -scheme FlightKit
```



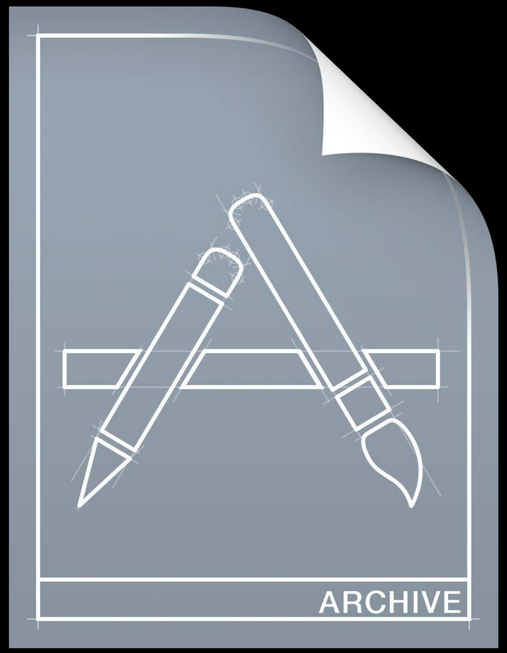
Archiving Your Framework

```
xcodebuild archive
   -scheme FlightKit
   -destination "…"
   -destination "…"
         …
   -destination "…"
```



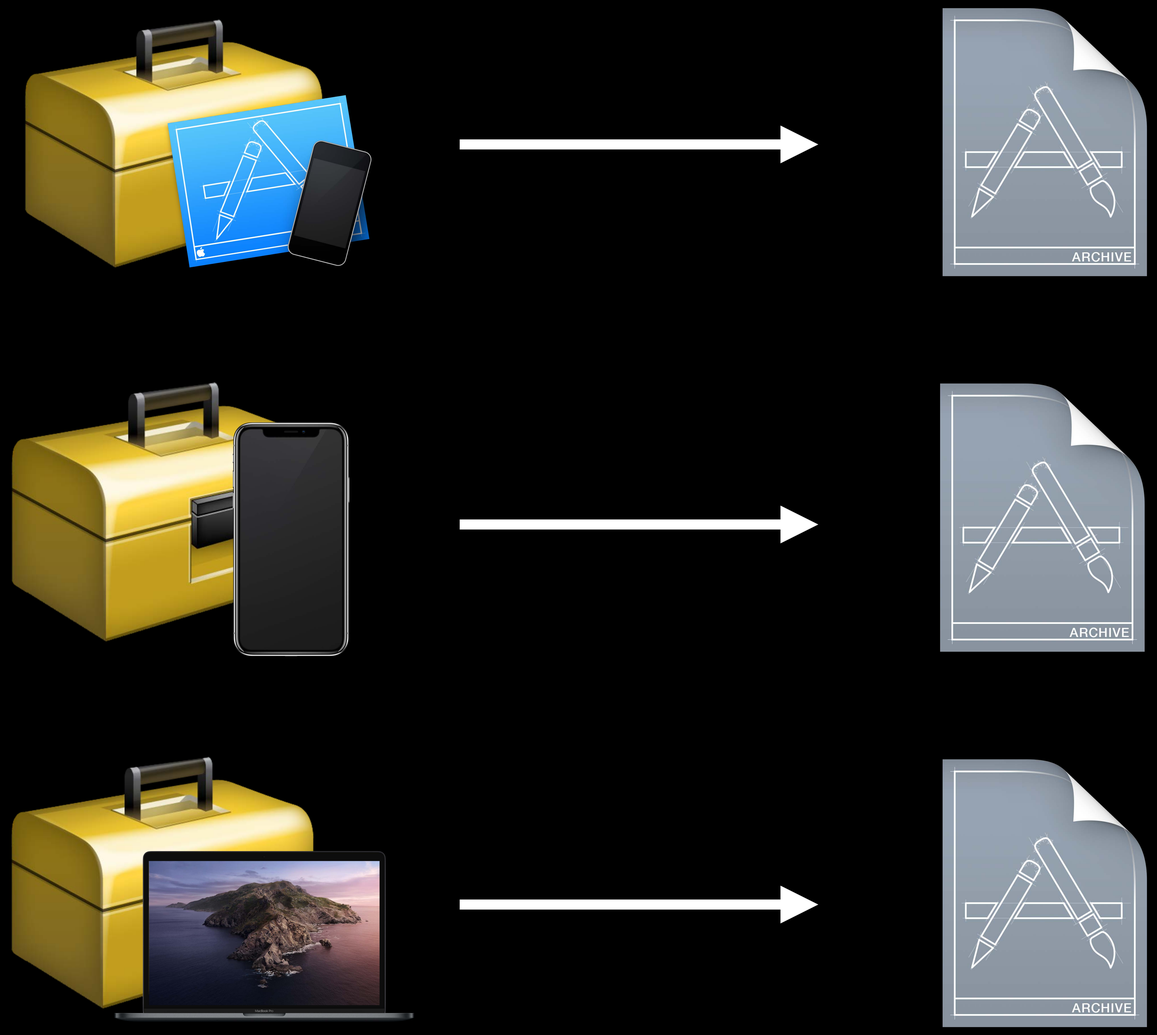
Archiving Your Framework

```
xcodebuild archive
   -scheme FlightKit
   -destination "…"
   -destination "…"
         …
   -destination "…"
   SKIP_INSTALL=NO
```

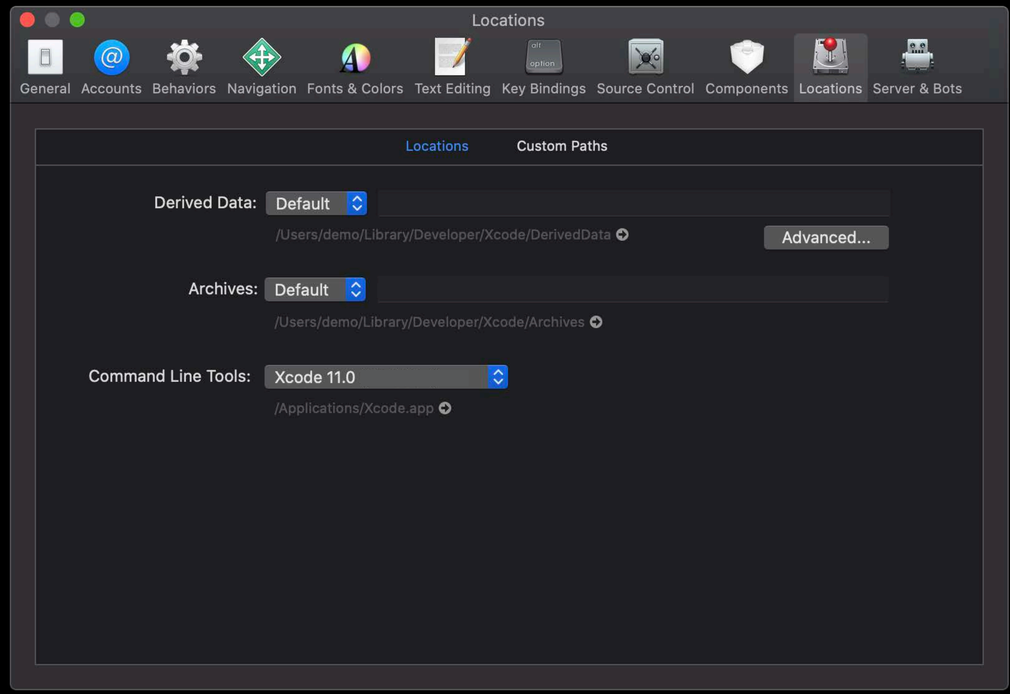


Archiving Your Framework

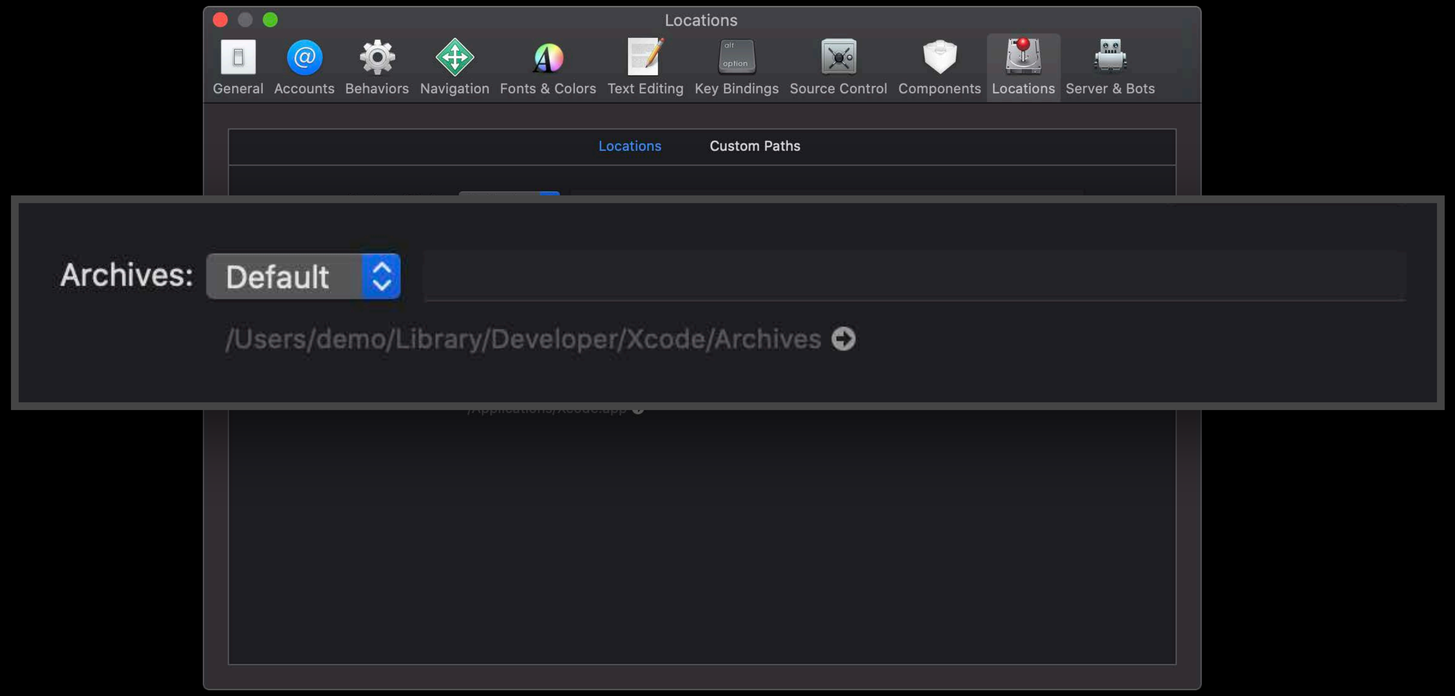
Archiving Your Framework



Archiving Your Framework

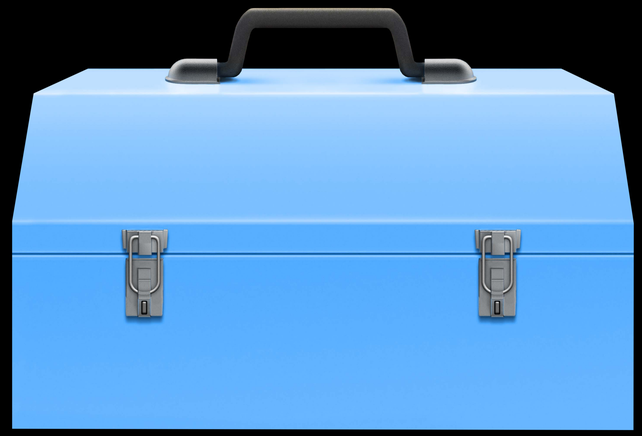


Archiving Your Framework



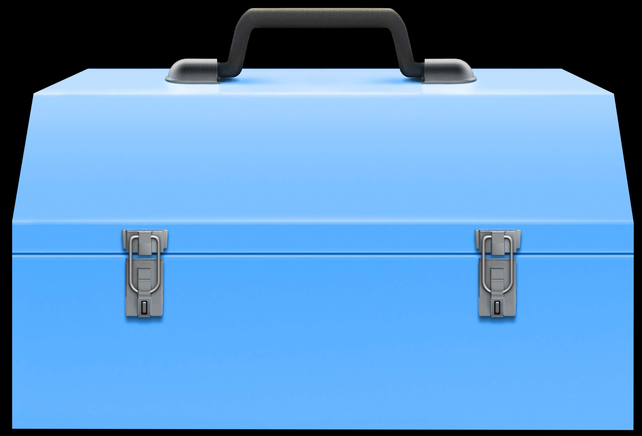
Building an XCFramework

NEW



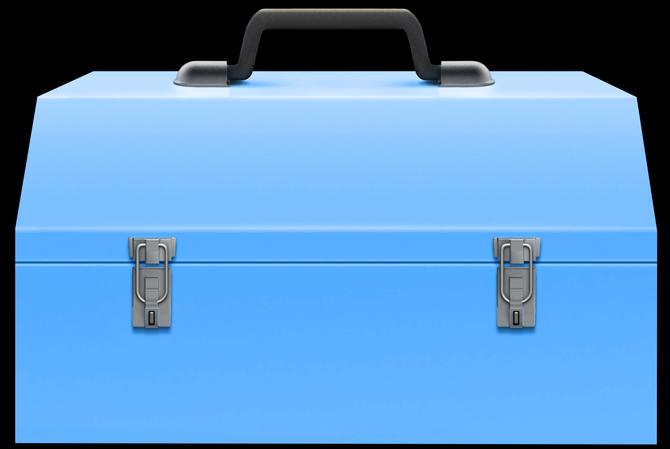
Building an XCFramework

NEW



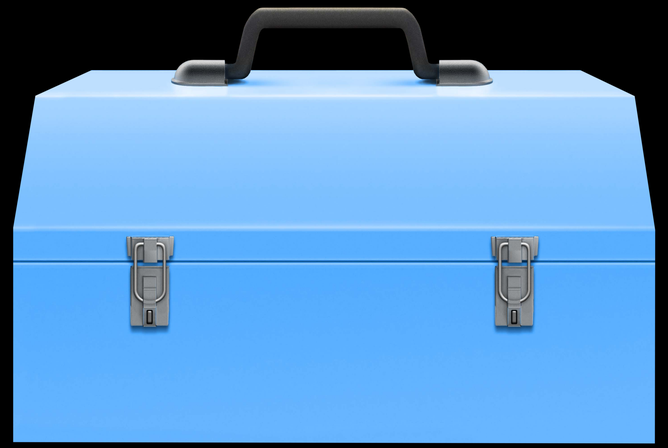
Building an XCFramework

```
xcodebuild -create-xcframework
```

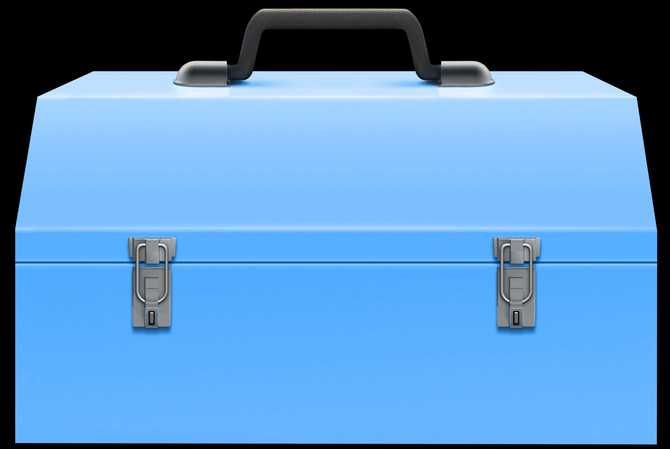


Building an XCFramework

```
xcodebuild -create-xcframework
   -framework [path]
   -framework [path]
         …
   -framework [path]
   -output FlightKit.xcframework
```

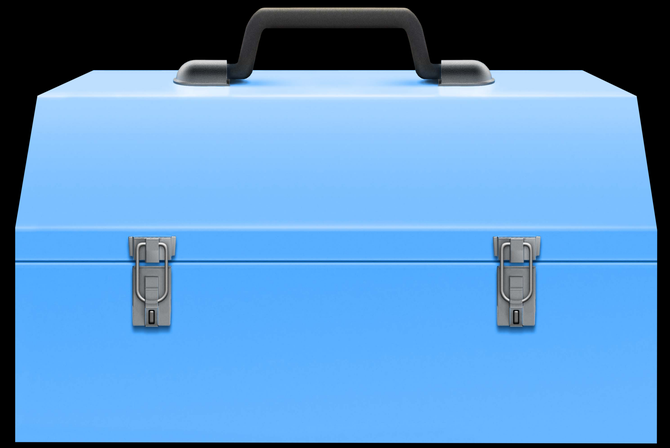


Building an XCFramework



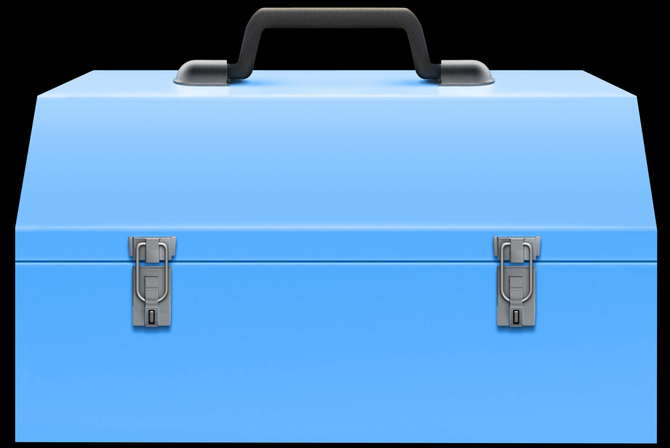
Building an XCFramework

1. Enable “Build Libraries for Distribution”



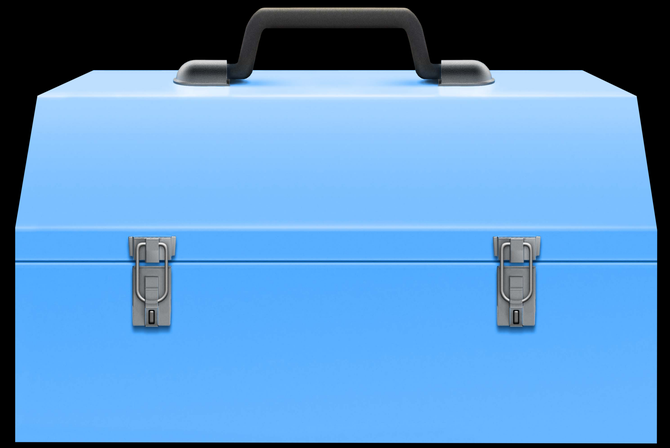
Building an XCFramework

1. Enable “Build Libraries for Distribution”
2. `xcodebuild archive`



Building an XCFramework

1. Enable “Build Libraries for Distribution”
2. `xcodebuild archive`
3. `xcodebuild -create-xcframework`



Framework Author Considerations

Jordan Rose, Swift Team

Evolving your framework

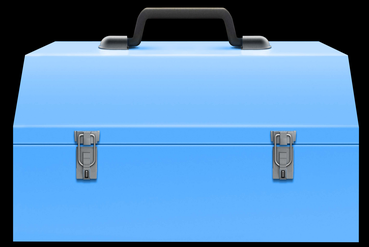
Trading flexibility for optimizability

Helping your clients

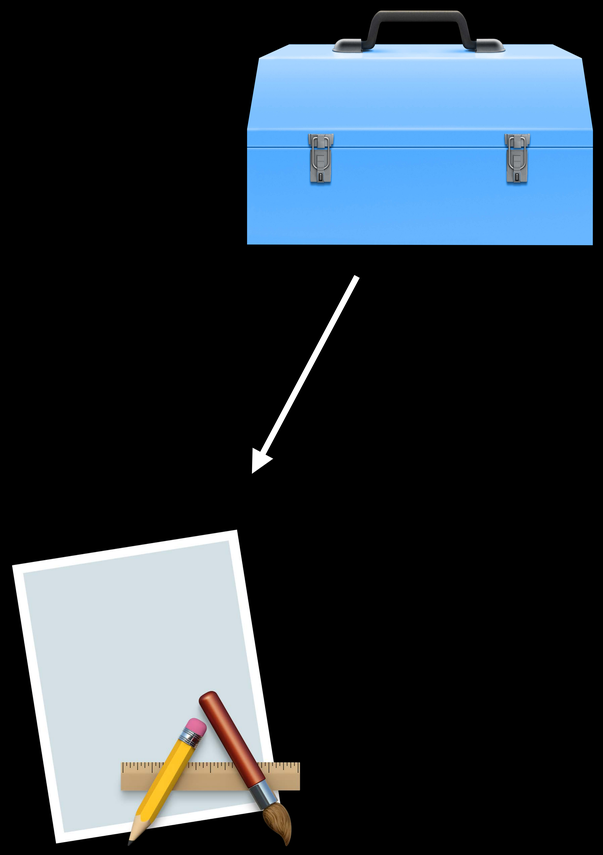
Evolving Your Framework

Evolving Your Framework

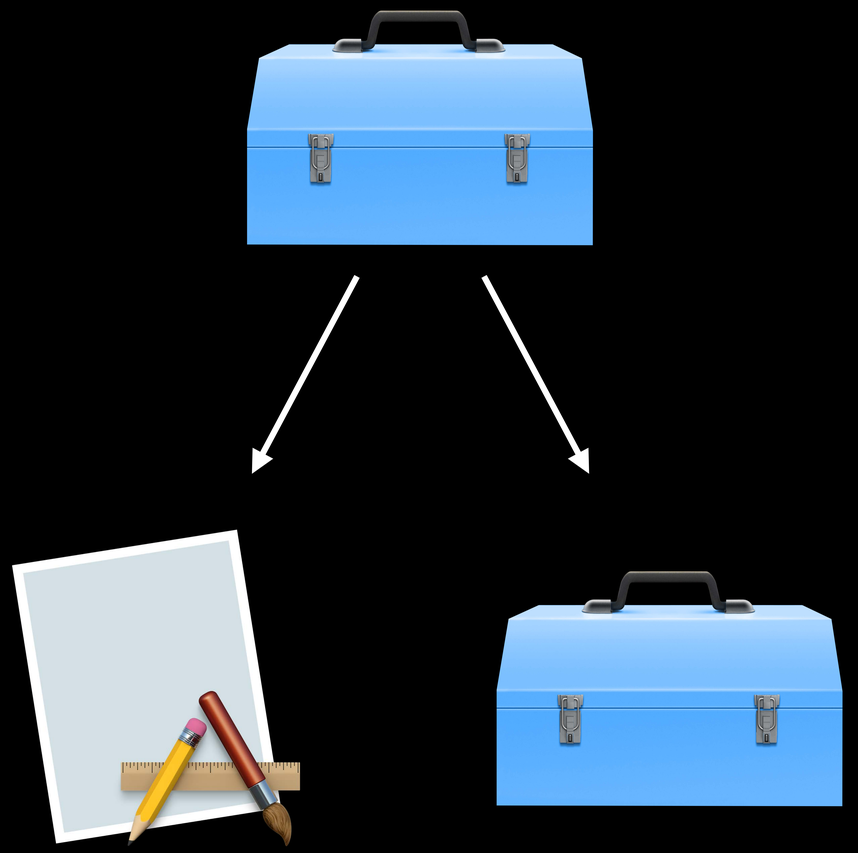
Why Is Binary Compatibility Important?



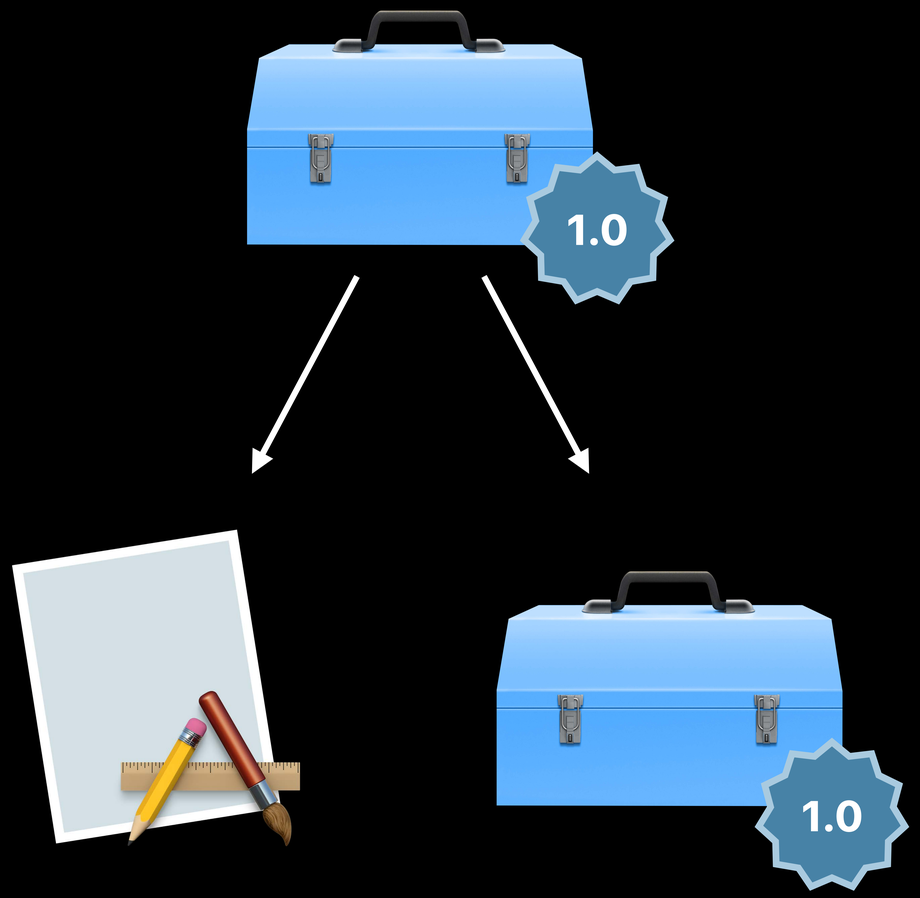
Why Is Binary Compatibility Important?



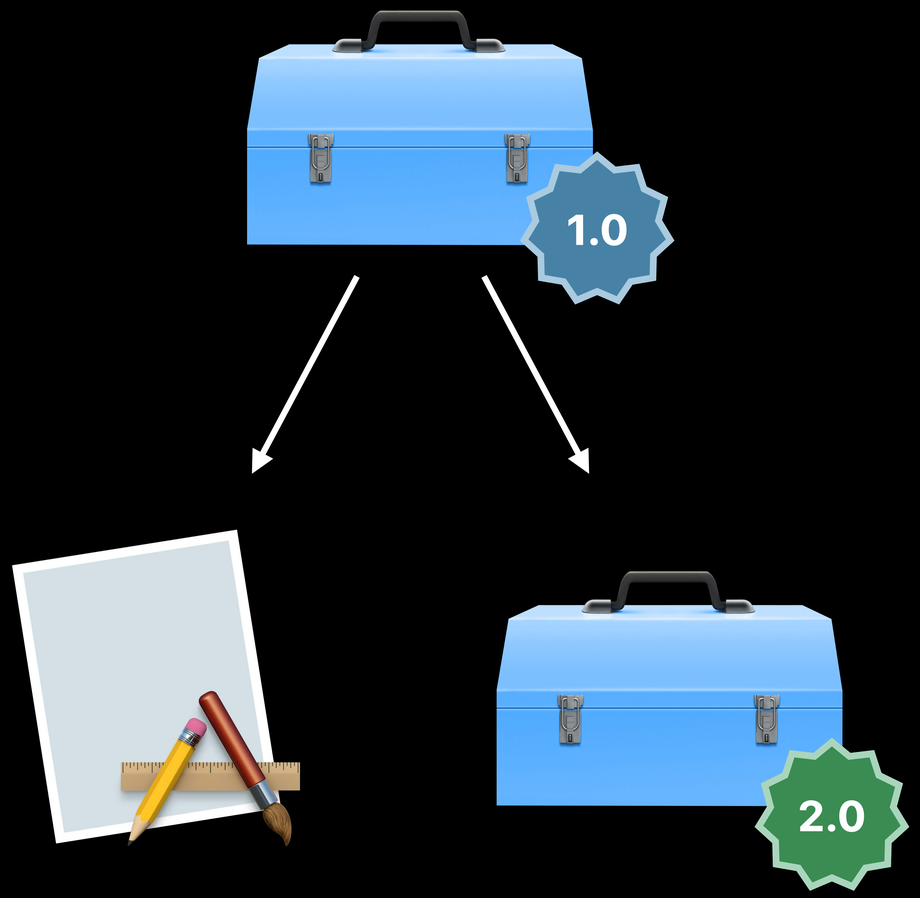
Why Is Binary Compatibility Important?



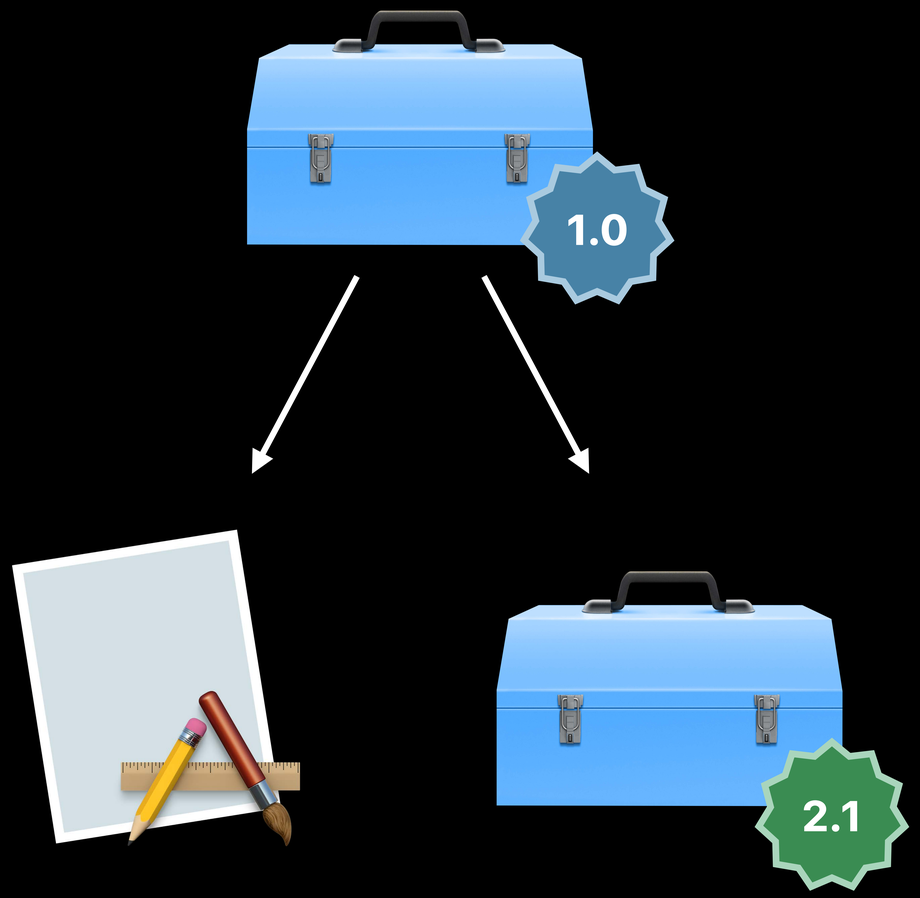
Why Is Binary Compatibility Important?



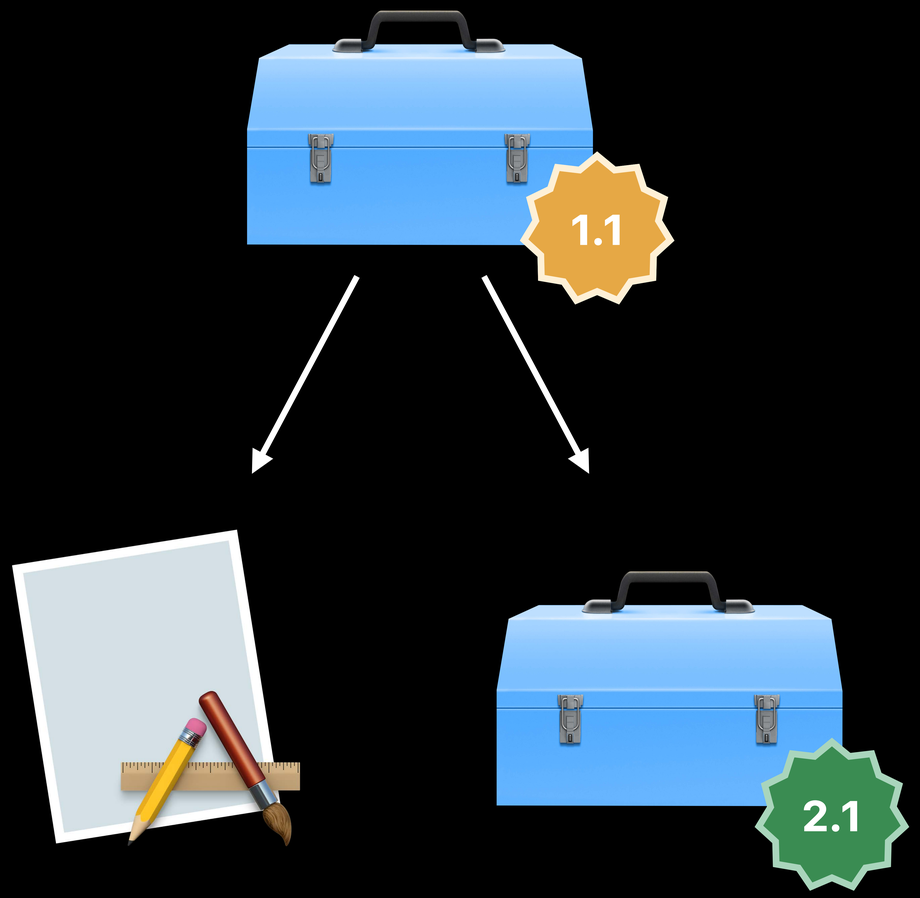
Why Is Binary Compatibility Important?



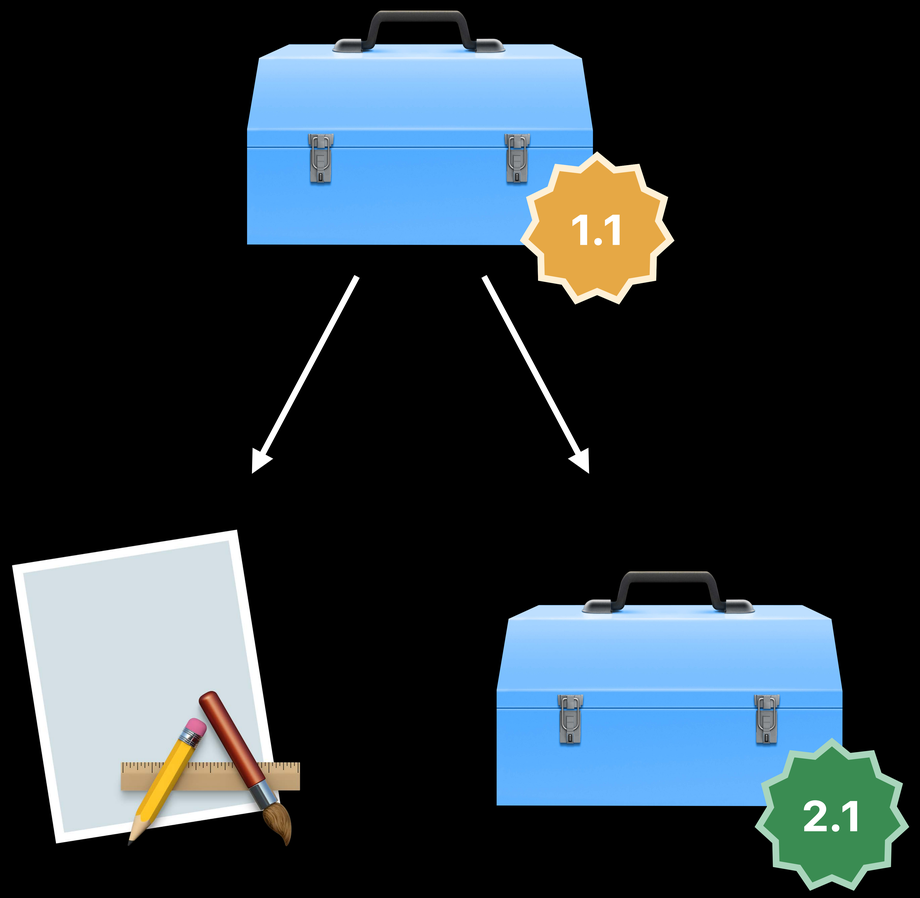
Why Is Binary Compatibility Important?



Why Is Binary Compatibility Important?



Why Is Binary Compatibility Important?



Framework Versions

Framework Versions

Key	Type	Value
▼ Information Property List	Dictionary	(8 items)
Localization native development re...	String	$(DEVELOPMENT_LANGUAGE)
Executable file	String	$(EXECUTABLE_NAME)
Bundle identifier	String	$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	$(PRODUCT_NAME)
Bundle OS Type code	String	$(PRODUCT_BUNDLE_PACKAGE_TYPE)
Bundle versions string, short	String	1.0
Bundle version	String	$(CURRENT_PROJECT_VERSION)

Semantic Versioning

semver.org

1.2.4

Patch Version
Bug fixes

1.2.4

Minor Version
Compatible additions

1.2.4

Major Version
Breaking changes (source, binary, semantic)

```
public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
public class Spaceship {
  public let name: String
  private static var defaultLocation: Location?
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation =
      Spaceship.defaultLocation ?? .launchpad
  }

  public func doABarrelRoll() {
    // …
  }

  public func fly(
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
  case ludicrous
```

```
public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
public class Spaceship {
  public let name: String
  private static var defaultLocation: Location?
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation =
      Spaceship.defaultLocation ?? .launchpad
  }

  public func doABarrelRoll() {
    // …
  }

  public func fly(
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
  case ludicrous
```

```
public class Spaceship {
  public let name: String
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
}

public struct Location {
  public var coordinates: Coordinates
}
```

```
public class Spaceship {
  public let name: String
  private static var defaultLocation: Location?
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation =
      Spaceship.defaultLocation ?? .launchpad
  }

  public func doABarrelRoll() {
    // …
  }

  public func fly(
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
  case ludicrous
```

1.0.0

```
public class Spaceship {
  public let name: String

  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.0.0

```
public class Spaceship {
  public let name: String
  private static var defaultLocation: Location?
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation =
      Spaceship.defaultLocation ?? .launchpad
  }

  public func doABarrelRoll() {
    // …
  }

  public func fly(
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.0.0

```
public class Spaceship {
  public let name: String

  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }



  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}


public enum Speed {
```

1.0.0

```
public class Spaceship {
  public let name: String
  private static var defaultLocation: Location?
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation =
      Spaceship.defaultLocation ?? .launchpad
  }

  public func doABarrelRoll() {
    // …
  }

  public func fly(
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.0.0

```
public class Spaceship {
  public let name: String

  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.0.0

```
public class Spaceship {
  public let name: String
  private static var defaultLocation: Location?
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation =
      Spaceship.defaultLocation ?? .launchpad
  }

  public func doABarrelRoll() {
    // …
  }

  public func fly(
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.0.0

```
public class Spaceship {
  public let name: String

  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.0.0

```
public class Spaceship {
  public let name: String
  private static var defaultLocation: Location?
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation =
      Spaceship.defaultLocation ?? .launchpad
  }

  public func doABarrelRoll() {
    // …
  }

  public func fly(
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.0.0

```
public class Spaceship {
  public let name: String

  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.0.1

```
public class Spaceship {
  public let name: String
  private static var defaultLocation: Location?
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation =
      Spaceship.defaultLocation ?? .launchpad
  }

  public func doABarrelRoll() {
    // …
  }

  public func fly(
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.0.0

```
public class Spaceship {
  public let name: String

  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.0.1

```
public class Spaceship {
  public let name: String
  private static var defaultLocation: Location?
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation =
      Spaceship.defaultLocation ?? .launchpad
  }

  public func doABarrelRoll() {
    // …
  }

  public func fly(
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.0.0

```
public class Spaceship {
  public let name: String

  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.1.0

```
public class Spaceship {
  public let name: String
  private static var defaultLocation: Location?
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation =
      Spaceship.defaultLocation ?? .launchpad
  }

  public func doABarrelRoll() {
    // …
  }

  public func fly(
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.0.0

```
public class Spaceship {
  public let name: String

  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}


public enum Speed {
```

1.1.0

```
public class Spaceship {
  public let name: String
  private static var defaultLocation: Location?
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation =
      Spaceship.defaultLocation ?? .launchpad
  }

  public func doABarrelRoll() {
    // …
  }

  public func fly(
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
```

1.0.0

```
public class Spaceship {
  public let name: String

  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation = .launchpad
  }

  public func fly(
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}


public enum Speed {
```

2.0.0

```
public class Spaceship {
  public let name: String
  private static var defaultLocation: Location?
  private var currentLocation: Location

  public init(name: String) {
    self.name = name
    currentLocation =
      Spaceship.defaultLocation ?? .launchpad
  }

  public func doABarrelRoll() {
    // …
  }

  public func fly(
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}


public enum Speed {
```

1.0.0

```
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}


public enum Speed {
  case leisurely
  case fast

}

public struct Location {
  public var coordinates: Coordinates

}
```

2.0.0

```
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
  case ludicrous
}

public struct Location: Hashable {
  public var coordinates: Coordinates
  public var label: String
}
```

1.0.0

```
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}


public enum Speed {
  case leisurely
  case fast

}

public struct Location {
  public var coordinates: Coordinates

}
```

2.0.0

```
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
  case ludicrous
}

public struct Location: Hashable {
  public var coordinates: Coordinates
  public var label: String
}
```

1.0.0

```
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}


public enum Speed {
  case leisurely
  case fast

}

public struct Location {
  public var coordinates: Coordinates

}
```

2.0.0

```
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
  case ludicrous
}

public struct Location: Hashable {
  public var coordinates: Coordinates
  public var label: String
}
```

1.0.0

```
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}


public enum Speed {
  case leisurely
  case fast

}

public struct Location {
  public var coordinates: Coordinates

}
```

2.0.0

```
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
  case ludicrous
}

public struct Location: Hashable {
  public var coordinates: Coordinates
  public var label: String
}
```

1.0.0

```
    to destination: Location,
    speed: Speed) {
    currentLocation = destination
  }
}


public enum Speed {
  case leisurely
  case fast

}

public struct Location {
  public var coordinates: Coordinates

}
```

2.1.0

```
    to destination: Location,
    speed: Speed,
    stealthily: Bool = false) {
    currentLocation = destination
  }
}

public enum Speed {
  case leisurely
  case fast
  case ludicrous
}

public struct Location: Hashable {
  public var coordinates: Coordinates
  public var label: String
}
```

Implications for API Design

Implications for API Design

Keep your interface small — easy to add, hard to remove!

Implications for API Design

Keep your interface small — easy to add, hard to remove!

Choose names and requirements carefully

Implications for API Design

Keep your interface small — easy to add, hard to remove!

Choose names and requirements carefully

Avoid unnecessary extensibility (open classes, arbitrary callbacks, etc.)

Behind the Scenes

Extra indirection at module boundaries

```
public class Spaceship {
  public init()
  public func fly(
    to destination: Location,
    speed: Speed)
}
```

Behind the Scenes

Extra indirection at module boundaries

```
public class Spaceship {
  public init()
  public func fly(
    to destination: Location,
    speed: Speed)
}
```

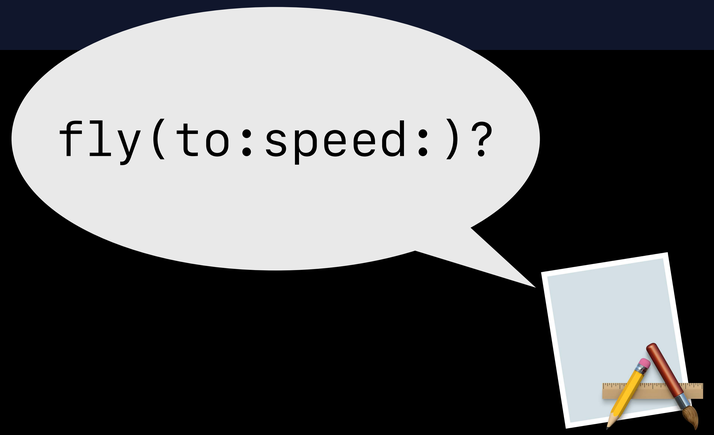
```
spaceship.fly(to: home, speed: .fast)
```

Behind the Scenes

Extra indirection at module boundaries

```
public class Spaceship {
  public init()
  public func fly(
    to destination: Location,
    speed: Speed)
}
```

```
spaceship.fly(to: home, speed: .fast)
```

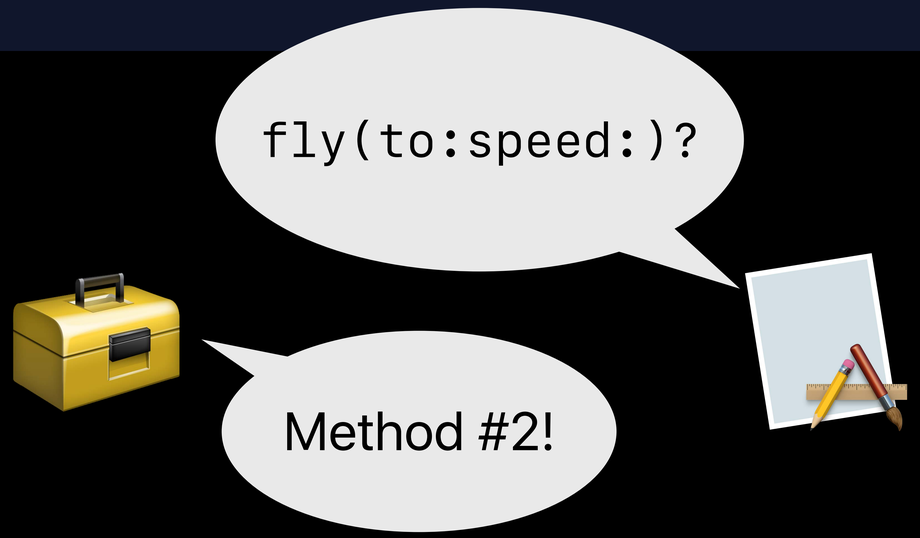


Behind the Scenes

Extra indirection at module boundaries

```
public class Spaceship {
  public init()
  public func fly(
    to destination: Location,
    speed: Speed)
}
```

```
spaceship.fly(to: home, speed: .fast)
```



Behind the Scenes

Extra indirection when clients use the framework’s structs and enums

```
public enum Speed {
  case leisurely
  case fast
}
```

```
spaceship.fly(to: home, speed: .fast)
```

Behind the Scenes

Extra indirection when clients use the framework’s structs and enums

```
public enum Speed {
  case leisurely
  case fast
}
```

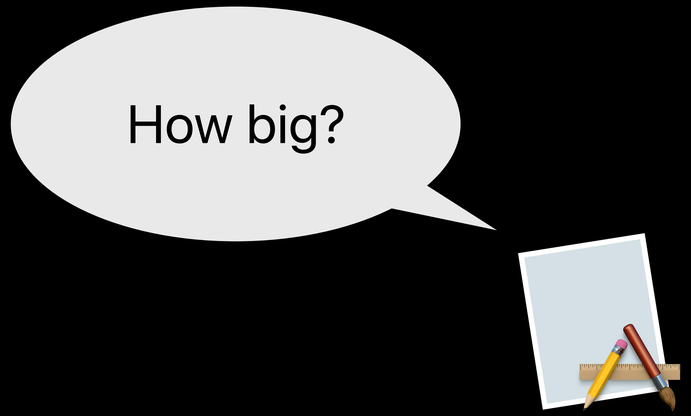
```
spaceship.fly(to: home, speed: .fast)
```

Behind the Scenes

Extra indirection when clients use the framework’s structs and enums

```
public enum Speed {
  case leisurely
  case fast
}
```

```
spaceship.fly(to: home, speed: .fast)
```

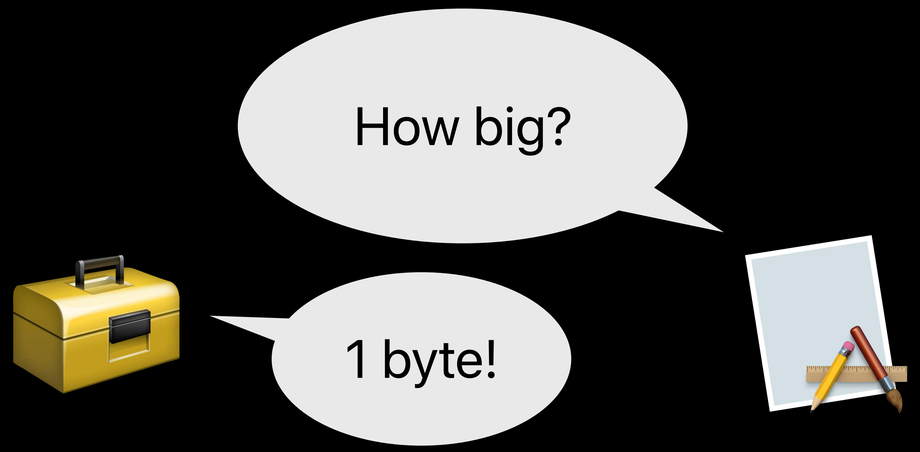


Behind the Scenes

Extra indirection when clients use the framework’s structs and enums

```
public enum Speed {
  case leisurely
  case fast
}
```

```
spaceship.fly(to: home, speed: .fast)
```



Behind the Scenes

Extra indirection when clients use the framework’s structs and enums

```
public enum Speed {
  case leisurely
  case fast
}
```

```
spaceship.fly(to: home, speed: .fast)
```

Behind the Scenes

Extra indirection when clients use the framework’s structs and enums

```
public enum Speed {
  case leisurely
  case fast
}
```

```
spaceship.fly(to: home, speed: .fast)
```

Trading Flexibility for Optimizability


Flexibility
Optimizability

Flexibility

Optimizability


Flexibility
Optimizability


Build Options
Setting
FlightKit
Build Libraries for Distribution
Yes


Flexibility
Optimizability


Build Options
Setting
FlightKit
Build Libraries for Distribution
Yes


Flexibility
Optimizability


Build Options
Setting
FlightKit
Build Libraries for Distribution
Yes

Flexibility

Optimizability

`@inlinable` functions

`@frozen` enums

`@frozen` structs

```
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>

  internal var currentCargo: Cargo?
}
```

```
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>

  internal var currentCargo: Cargo?
}
```

```
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>

  internal var currentCargo: Cargo?
}
```

```
// swift-interface-format-version: 1.0
// swift-compiler-version: Swift version 5.1
// swift-module-flags: -target arm64-apple-ios13.0
                       -enable-library-evolution
                       -swift-version 5 -O
                       -module-name FlightKit
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>
}
```

```
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>

  internal var currentCargo: Cargo?
}
```

```
// swift-interface-format-version: 1.0
// swift-compiler-version: Swift version 5.1
// swift-module-flags: -target arm64-apple-ios13.0
                       -enable-library-evolution
                       -swift-version 5 -O
                       -module-name FlightKit
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>
}
```

```swift
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>

  internal var currentCargo: Cargo?
}
```

```swift
// swift-interface-format-version: 1.0
// swift-compiler-version: Swift version 5.1
// swift-module-flags: -target arm64-apple-ios13.0
                       -enable-library-evolution
                       -swift-version 5 -O
                       -module-name FlightKit
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>
}
```

```swift
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>

  internal var currentCargo: Cargo?
}
```

```swift
// swift-interface-format-version: 1.0
// swift-compiler-version: Swift version 5.1
// swift-module-flags: -target arm64-apple-ios13.0
                       -enable-library-evolution
                       -swift-version 5 -O
                       -module-name FlightKit
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>
}
```

```swift
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>

  internal var currentCargo: Cargo?
}
```

```swift
// swift-interface-format-version: 1.0
// swift-compiler-version: Swift version 5.1
// swift-module-flags: -target arm64-apple-ios13.0
                       -enable-library-evolution
                       -swift-version 5 -O
                       -module-name FlightKit
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>
}
```

```swift
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>

  internal var currentCargo: Cargo?
}
```

```swift
// swift-interface-format-version: 1.0
// swift-compiler-version: Swift version 5.1
// swift-module-flags: -target arm64-apple-ios13.0
                       -enable-library-evolution
                       -swift-version 5 -O
                       -module-name FlightKit
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFro
  internal va                    rement<UnitMass>
}
```

cargo.mass <= self.capacity

“But what happens if I change it?”

```swift
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>

  internal var currentCargo: Cargo?
}
```

```swift
// swift-interface-format-version: 1.0
// swift-compiler-version: Swift version 5.1
// swift-module-flags: -target arm64-apple-ios13.0
                       -enable-library-evolution
                       -swift-version 5 -O
                       -module-name FlightKit
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFro
  internal va                rement<UnitMass>
}
```

cargo.mass <= self.capacity

```swift
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity &&
           !cargo.isRadioactive
  }

  @usableFromInline
  internal var capacity: Measurement<UnitMass>

  internal var currentCargo: Cargo?
}
```

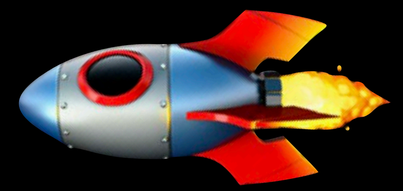
```swift
// swift-interface-format-version: 1.0
// swift-compiler-version: Swift version 5.1
// swift-module-flags: -target arm64-apple-ios13.0
                       -enable-library-evolution
                       -swift-version 5 -O
                       -module-name FlightKit
public class CargoShip: Spaceship {
  @inlinable
  public func canCarry(
    _ cargo: Cargo) -> Bool {
    return cargo.mass <= self.capacity
  }

  @usableFro
  internal va                    rement<UnitMass>
}
```

cargo.mass <= self.capacity

```
cargo.mass <=
self.capacity &&
!cargo.isRadioactive
```

```
cargo.mass <=
self.capacity
```

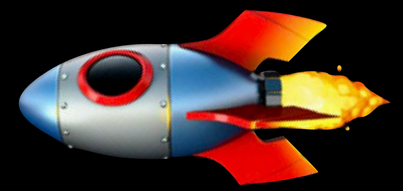
```
cargo.mass <=
self.capacity &&
!cargo.isRadioactive
```

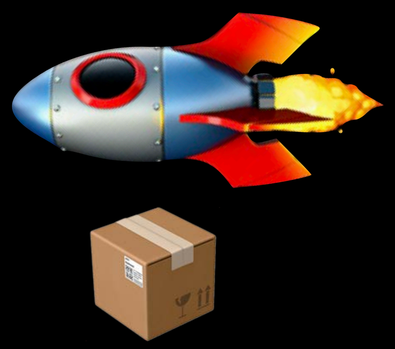


```
cargo.mass <=
self.capacity
```

```
cargo.mass <=
self.capacity &&
!cargo.isRadioactive
```

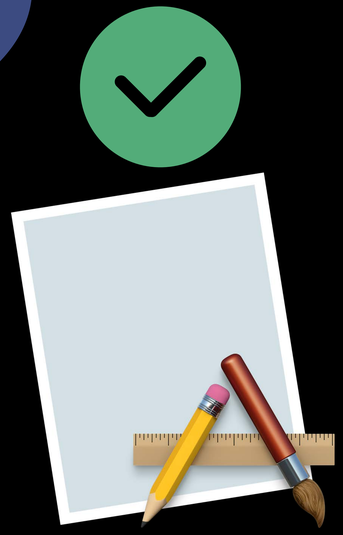
```
cargo.mass <=
self.capacity
```

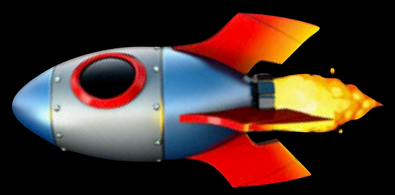




```
cargo.mass <=
self.capacity &&
!cargo.isRadioactive
```

```
cargo.mass <=
self.capacity
```



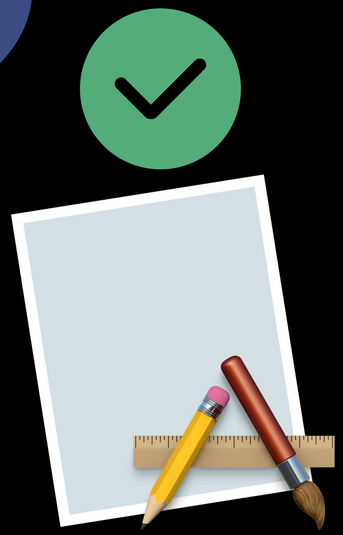


```
cargo.mass <=
self.capacity &&
!cargo.isRadioactive
```

```
cargo.mass <=
self.capacity
```

```
cargo.mass <=
self.capacity &&
!cargo.isRadioactive
```

```
cargo.mass <=
self.capacity
```

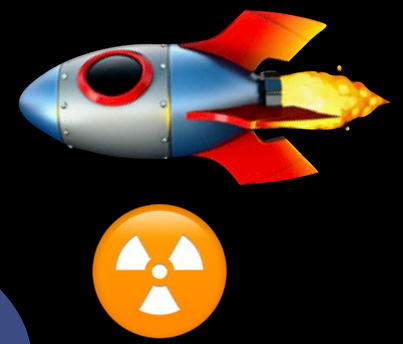


```
cargo.mass <=
self.capacity &&
!cargo.isRadioactive
```

```
cargo.mass <=
self.capacity
```

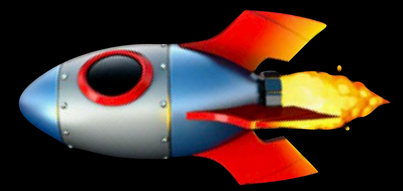
```
cargo.mass <=
self.capacity &&
!cargo.isRadioactive
```

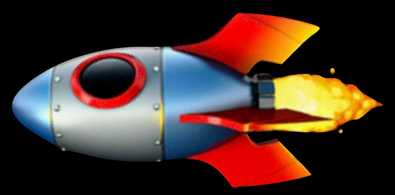
```
cargo.mass <=
self.capacity
```



```
cargo.mass <=
self.capacity &&
!cargo.isRadioactive
```

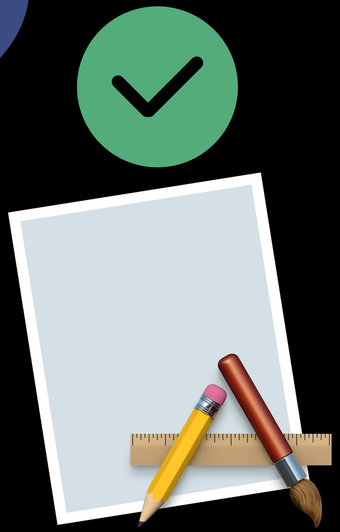
```
cargo.mass <=
self.capacity
```

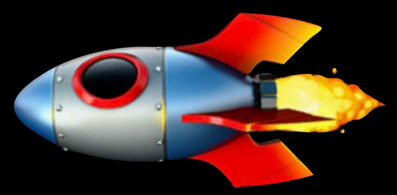




```
cargo.mass <=
self.capacity &&
!cargo.isRadioactive
```

```
cargo.mass <=
self.capacity
```





```
cargo.mass <=
self.capacity &&
!cargo.isRadioactive
```

```
cargo.mass <=
self.capacity
```

Changing an `@inlinable` Function

Rule of thumb — don’t change the output or observable behavior

- Better algorithms are okay

`@frozen` Enums

```
public enum FlightPlanKind {
  case oneWay
  case roundTrip
}
```

`@frozen` Enums

```
public enum FlightPlanKind {
  case oneWay
  case roundTrip
}
```

```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
@unknown default:
  print("Have a good flight!")
}
```

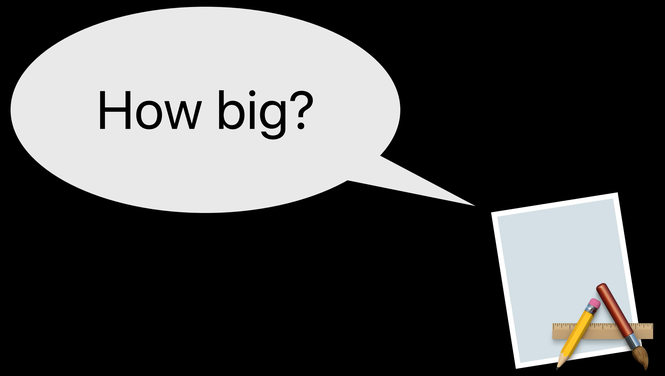
`@frozen` Enums

```
public enum FlightPlanKind {
  case oneWay
  case roundTrip
}
```

```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
@unknown default:
  print("Have a good flight!")
}
```

`@frozen` Enums

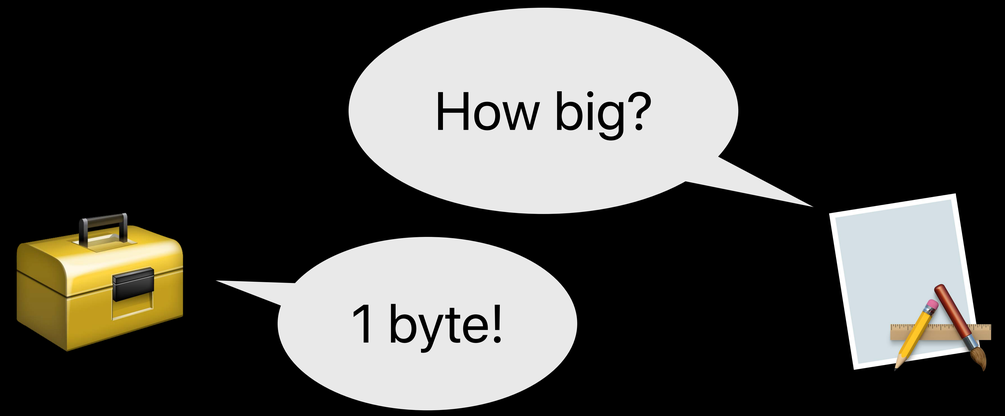
```
public enum FlightPlanKind {
  case oneWay
  case roundTrip
}
```




```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
@unknown default:
  print("Have a good flight!")
}
```

`@frozen` Enums

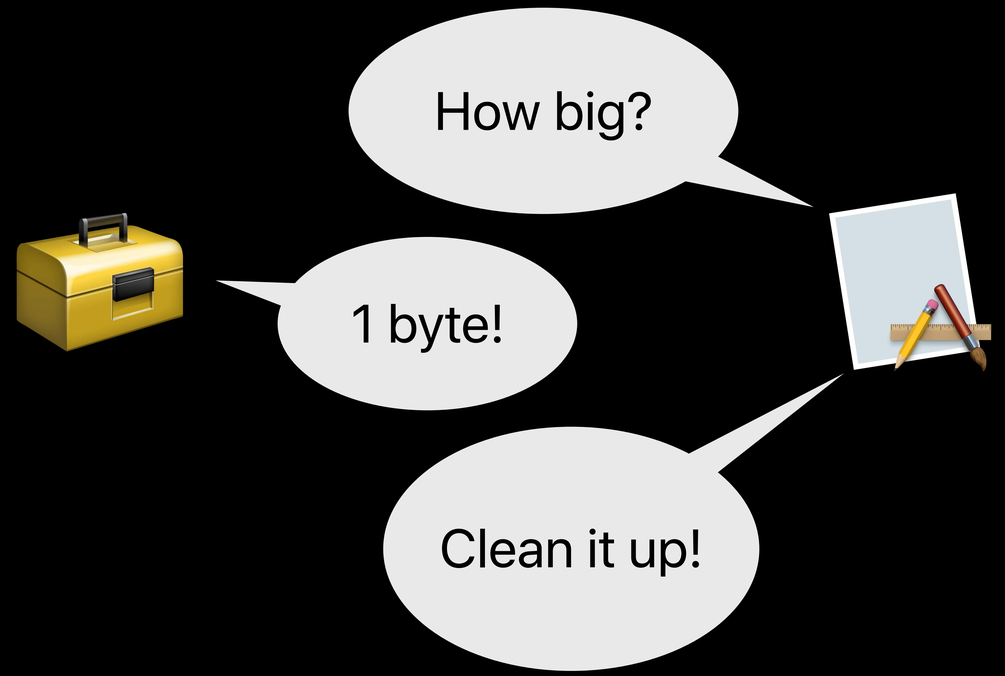
```
public enum FlightPlanKind {
  case oneWay
  case roundTrip
}
```




```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
@unknown default:
  print("Have a good flight!")
}
```

`@frozen` Enums

```
public enum FlightPlanKind {
  case oneWay
  case roundTrip
}
```




```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
@unknown default:
  print("Have a good flight!")
}
```

@frozen Enums

NEW

```
@frozen public enum FlightPlanKind {
  case oneWay
  case roundTrip
}
```

```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
@unknown default:
  print("Have a good flight!")
}
```

@frozen Enums

NEW

```
@frozen public enum FlightPlanKind {
  case oneWay
  case roundTrip
}
```

```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
@unknown default:
  print("Have a good flight!")
}
```

`@frozen` Enums

NEW

```
@frozen public enum FlightPlanKind {
  case oneWay
  case roundTrip
}
```

Promise not to add new cases

```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
@unknown default:
  print("Have a good flight!")
}
```

`@frozen` Enums

NEW

```
@frozen public enum FlightPlanKind {
  case oneWay
  case roundTrip
}
```

Promise not to add new cases

Clients won’t write `default` in switches

```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
@unknown default:
  print("Have a good flight!")
}
```

`@frozen` Enums

NEW

```
@frozen public enum FlightPlanKind {
  case oneWay
  case roundTrip
}
```

Promise not to add new cases

Clients won’t write `default` in switches

```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
}
```

`@frozen` Enums

NEW

```
@frozen public enum FlightPlanKind {
  case oneWay
  case roundTrip
}
```

```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
}
```

Promise not to add new cases

Clients won’t write `default` in switches

Better performance, especially when no cases have associated values

Changing `@frozen` Enums

```
@frozen public enum FlightPlanKind {
  case oneWay
  case roundTrip
  case multiHop([Location])
}
```

```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
}
```

Changing `@frozen` Enums

```
@frozen public enum FlightPlanKind {
  case oneWay
  case roundTrip
  case multiHop([Location])
}
```

```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
}
```

Changing `@frozen` Enums

```
@frozen public enum FlightPlanKind {
  case oneWay
  case roundTrip
  case multiHop([Location])
}
```

2.1.0

```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
}
```

Changing `@frozen` Enums

```
@frozen public enum FlightPlanKind {
  case oneWay
  case roundTrip
  case multiHop([Location])
}
```

3.0.0

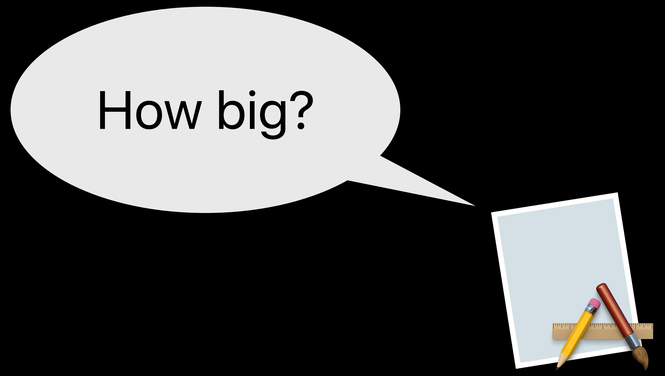
```
switch kind {
case .oneWay:
  print("See you there!")
case .roundTrip:
  print("See you in a few weeks!")
}
```

`@frozen` Structs

```
public struct Coordinates {
  public var radius, azimuth, inclination: Double
  public init(r: Double, alpha: Double, theta: Double) {…}
}
```

`@frozen` Structs

```
public struct Coordinates {
  public var radius, azimuth, inclination: Double
  public init(r: Double, alpha: Double, theta: Double) {…}
}
```

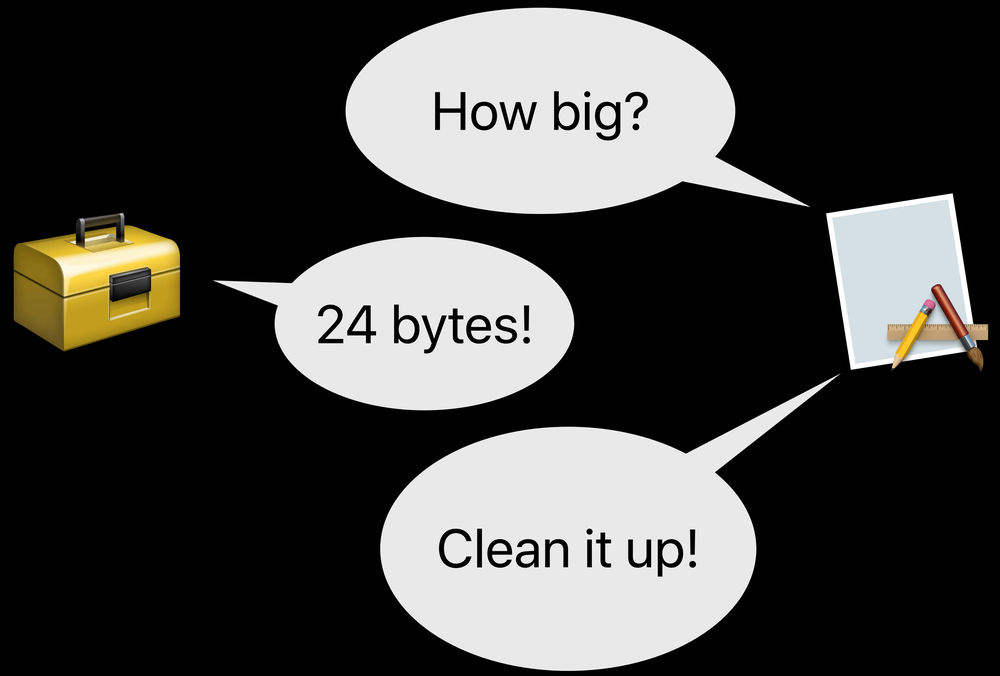


`@frozen` Structs

```
public struct Coordinates {
  public var radius, azimuth, inclination: Double
  public init(r: Double, alpha: Double, theta: Double) {…}
}
```

`@frozen` Structs

```
public struct Coordinates {
  public var radius, azimuth, inclination: Double
  public init(r: Double, alpha: Double, theta: Double) {…}
}
```



`@frozen` Structs

NEW

```
@frozen public struct Coordinates {
  public var radius, azimuth, inclination: Double
  public init(r: Double, alpha: Double, theta: Double) {…}
}
```

@frozen Structs

NEW

```
@frozen public struct Coordinates {
  public var radius, azimuth, inclination: Double
  public init(r: Double, alpha: Double, theta: Double) {…}
}
```

`@frozen` Structs

NEW

```
@frozen public struct Coordinates {
  public var radius, azimuth, inclination: Double
  public init(r: Double, alpha: Double, theta: Double) {…}
}
```

Promise not to add or reorder stored properties

`@frozen` Structs

NEW

```
@frozen public struct Coordinates {
  public var radius, azimuth, inclination: Double
  public init(r: Double, alpha: Double, theta: Double) {…}
}
```

Promise not to add or reorder stored properties

Types of stored properties must be public or `@usableFromInline`

`@frozen` Structs

NEW

```
@frozen public struct Coordinates {
  public var radius, azimuth, inclination: Double
  @inlinable public init(r: Double, alpha: Double, theta: Double) {…}
}
```

Promise not to add or reorder stored properties

Types of stored properties must be public or `@usableFromInline`

Allows inlinable initializers

`@frozen` Structs

NEW

```
@frozen public struct Coordinates {
  public var radius, azimuth, inclination: Double
  @inlinable public init(r: Double, alpha: Double, theta: Double) {…}
}
```

Promise not to add or reorder stored properties

Types of stored properties must be public or `@usableFromInline`

Allows inlinable initializers

Flexibility Is the Default

Flexibility Is the Default

Breaking changes are inconvenient

Flexibility Is the Default

Breaking changes are inconvenient

These attributes only affect client code

Flexibility Is the Default

Breaking changes are inconvenient

These attributes only affect client code

Profile before reaching for `@inlinable` or `@frozen`

Helping Your Clients

Entitlements

Entitlements

Entitlements

Document your requirements

Entitlements

Document your requirements

Minimize entitlement requests

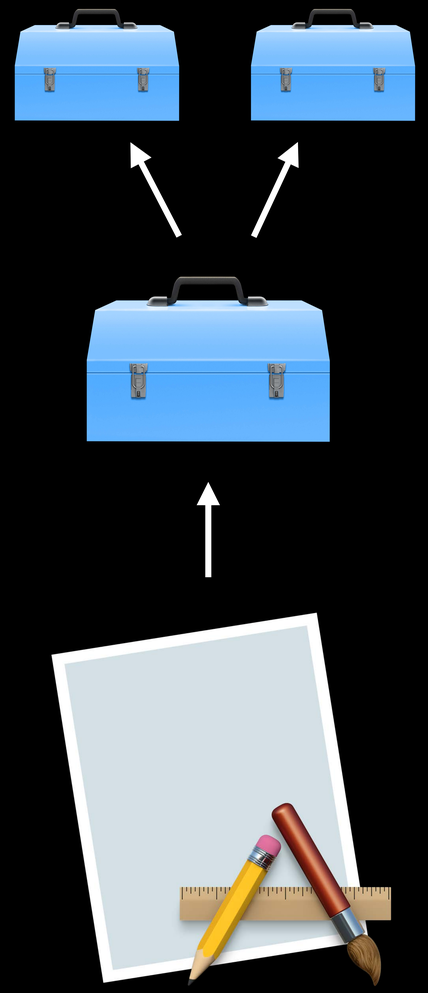
Entitlements

Document your requirements

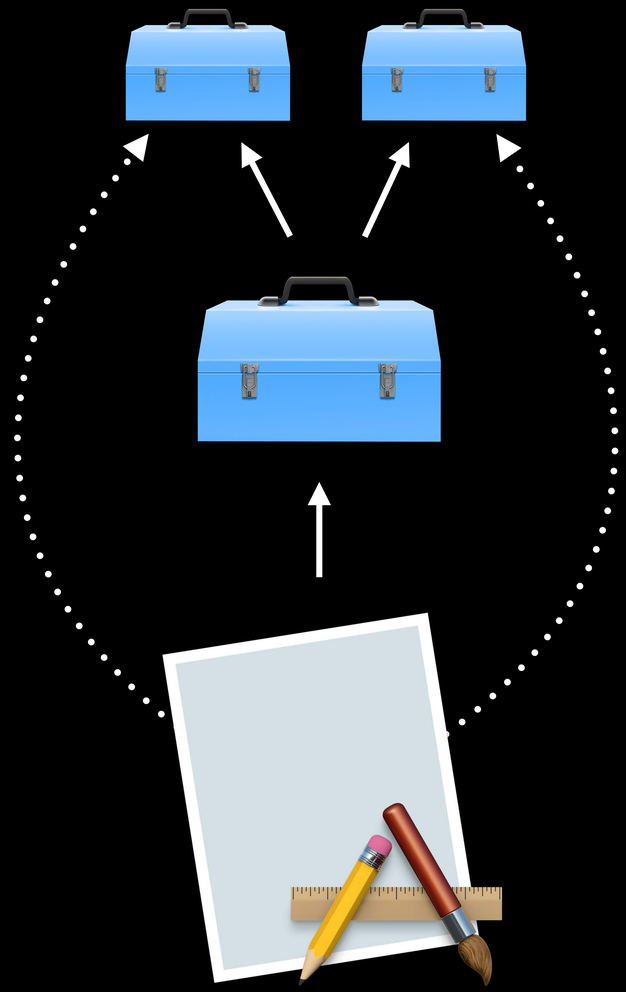
Minimize entitlement requests

Handle denial gracefully

Dependencies

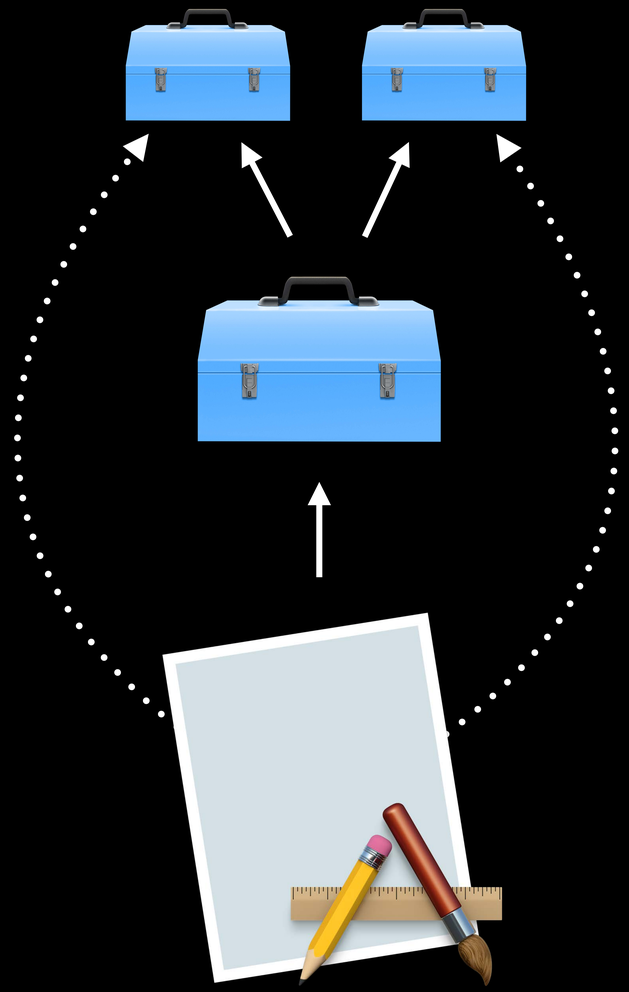


Dependencies



Dependencies

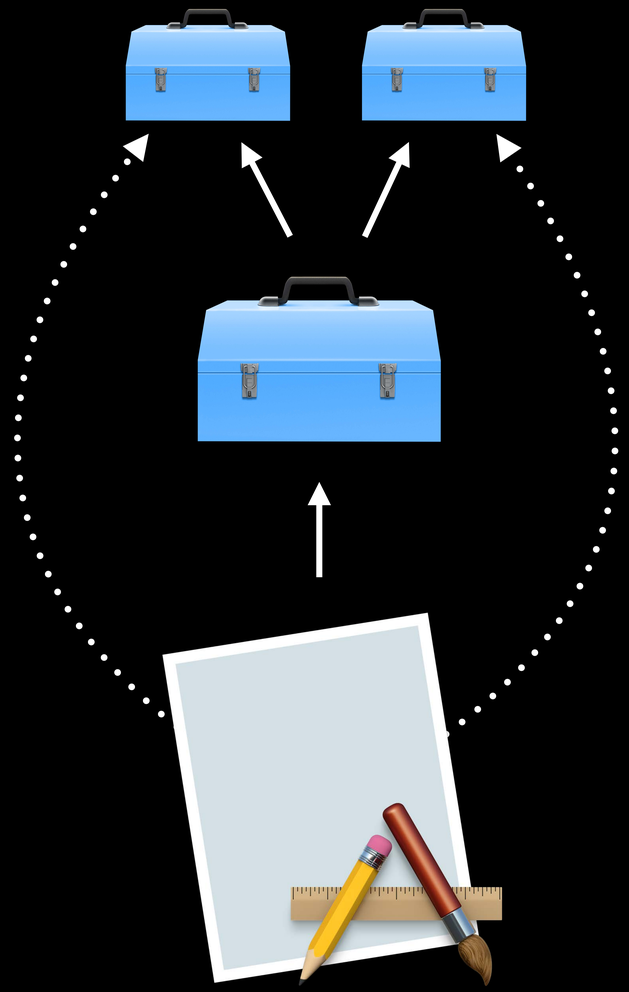
Document your dependencies



Dependencies

Document your dependencies

Minimize your dependencies

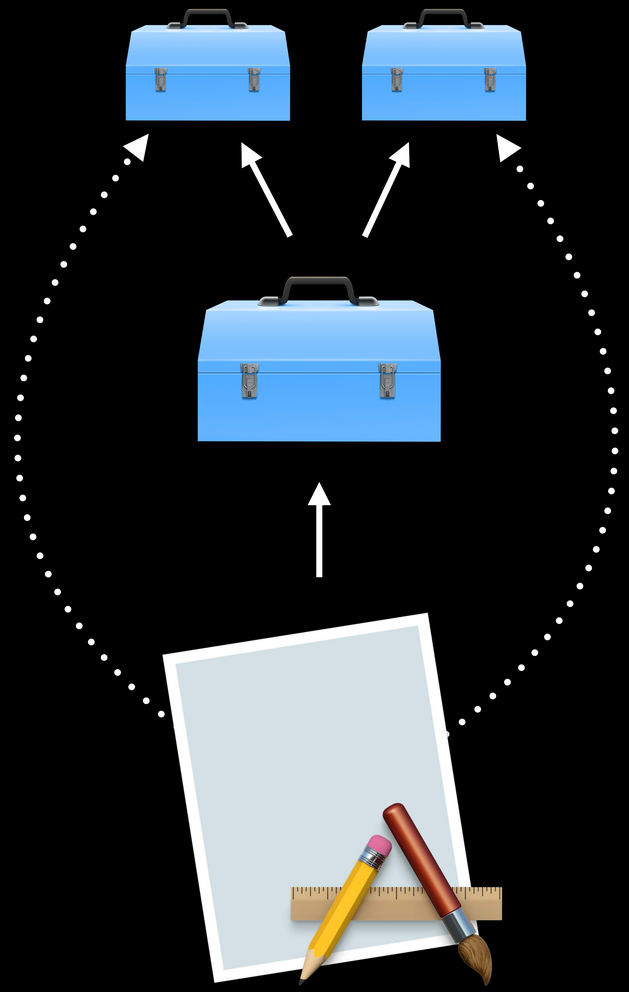


Dependencies

Document your dependencies

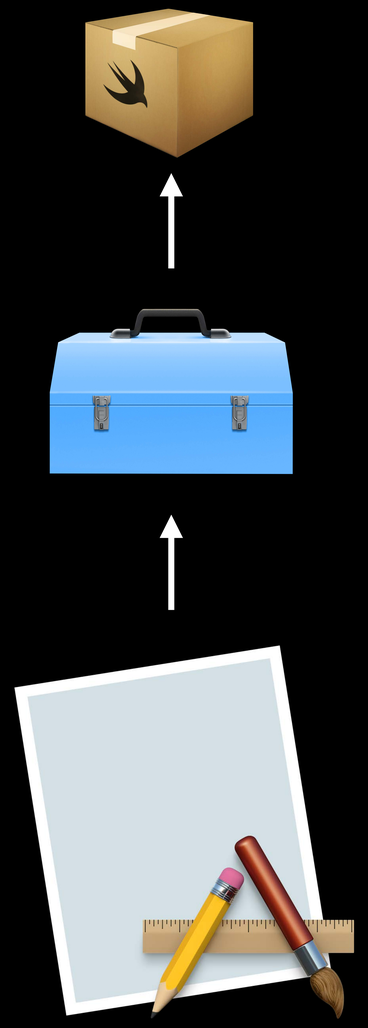
Minimize your dependencies

Dependencies must use
“Build Libraries for Distribution”

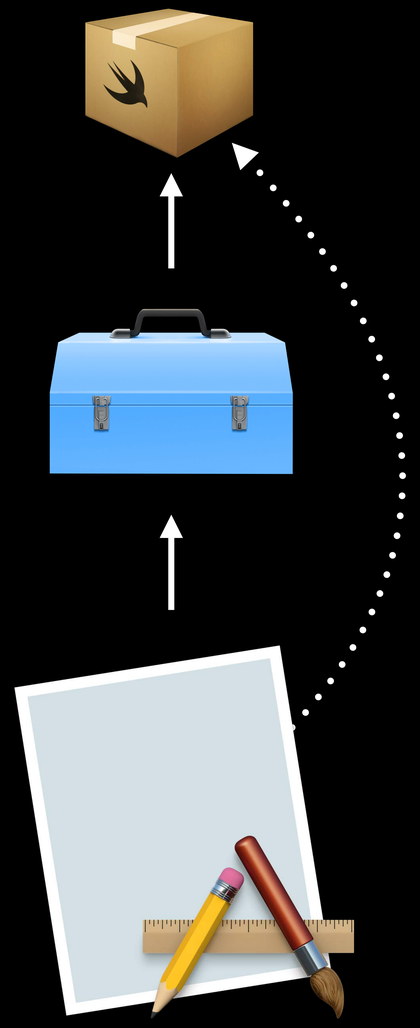


Binary Frameworks Cannot Depend on Packages

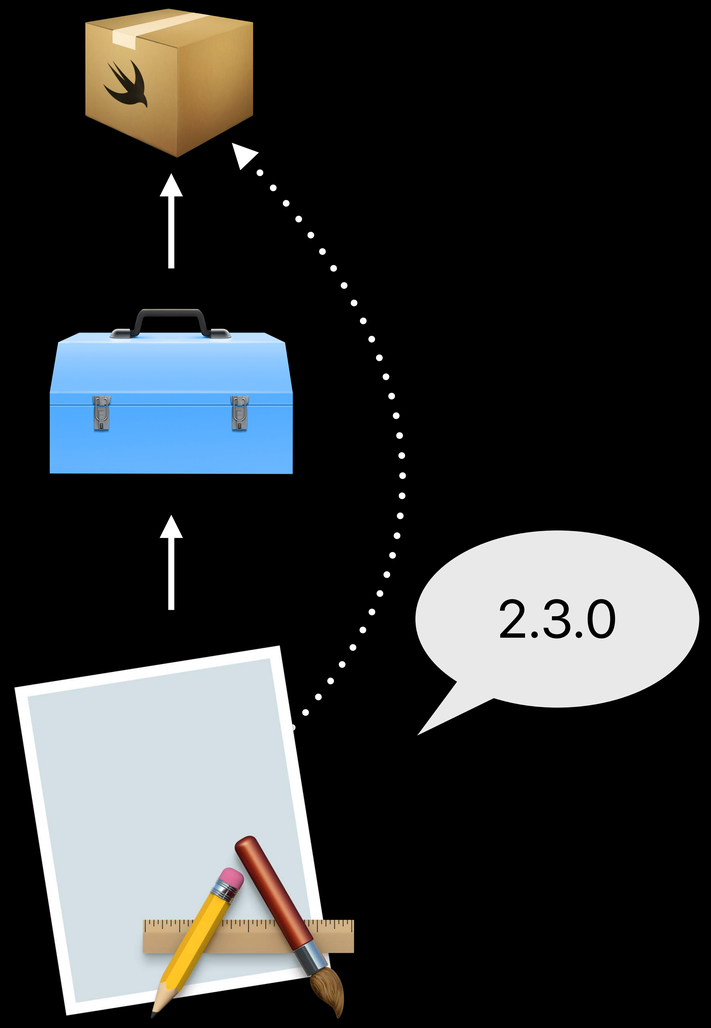
Binary Frameworks Cannot Depend on Packages



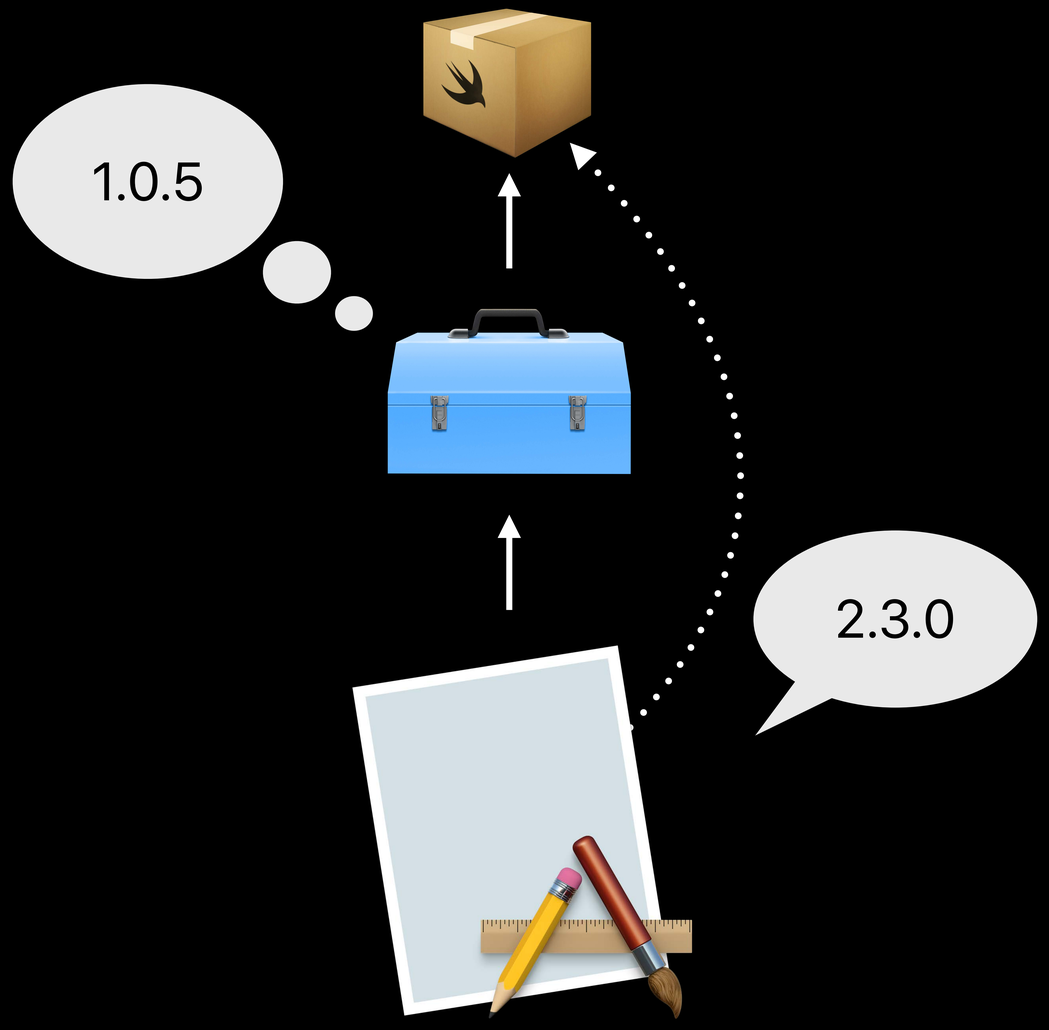
Binary Frameworks Cannot Depend on Packages



Binary Frameworks Cannot Depend on Packages



Binary Frameworks Cannot Depend on Packages



Your Objective-C Interface

```
@import FlightKit;
```

Your Objective-C Interface

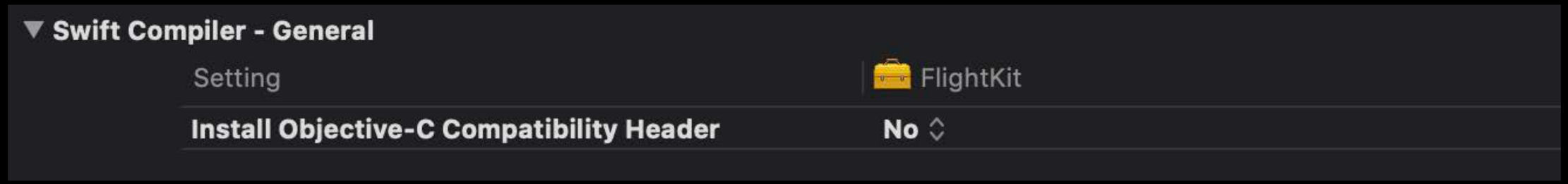
```
@import FlightKit;
```

If Your Swift Code Has No Objective-C API

```
@import FlightKit;
```

If Your Swift Code Has No Objective-C API

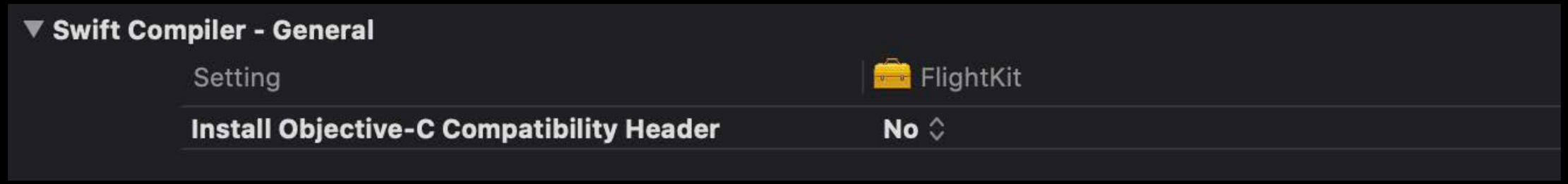
```
@import FlightKit;
```



If Your Swift Code Has No Objective-C API

```
@import FlightKit;
```

FlightKit.h



If Your Framework Has No Objective-C API

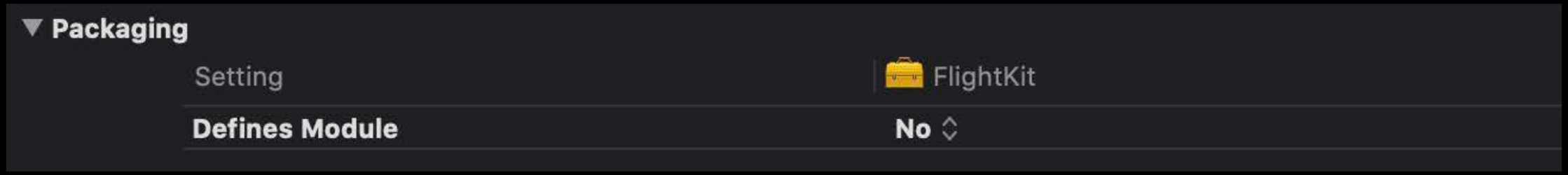
```
@import FlightKit;
```

FlightKit.h

If Your Framework Has No Objective-C API

```
@import FlightKit;
```

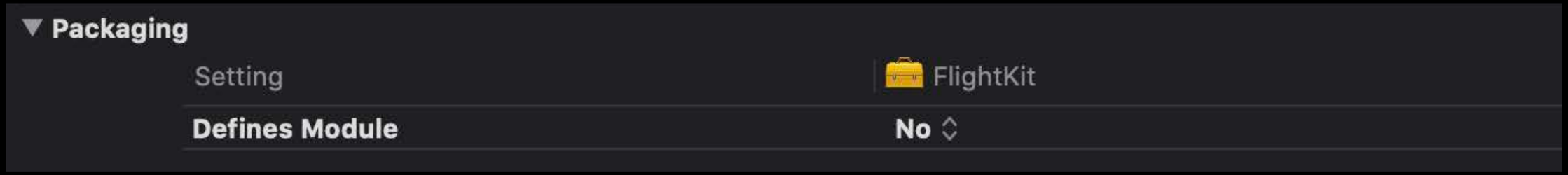
FlightKit.h



If Your Framework Has No Objective-C API

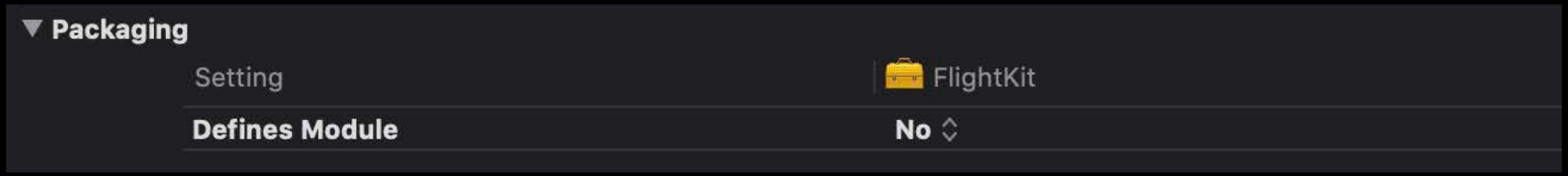
```
@import FlightKit;
```

FlightKit.h



If Your Framework Has No Objective-C API

```
@import FlightKit;
```



Summary

Summary

XCFrameworks — for distributing multiple framework variants

“Build Libraries for Distribution” build setting

Framework owners — know your responsibilities

More Information

developer.apple.com/wwdc19/416

Swift Open Hours	Thursday, 3:00
Building, Signing, and Distributing Lab	Friday, 9:00


WWDC19