

Cocoa Development Tips

Twenty-nine things you may not know about Cocoa

Session 236

Rachel Goldeen, Cocoa Engineer

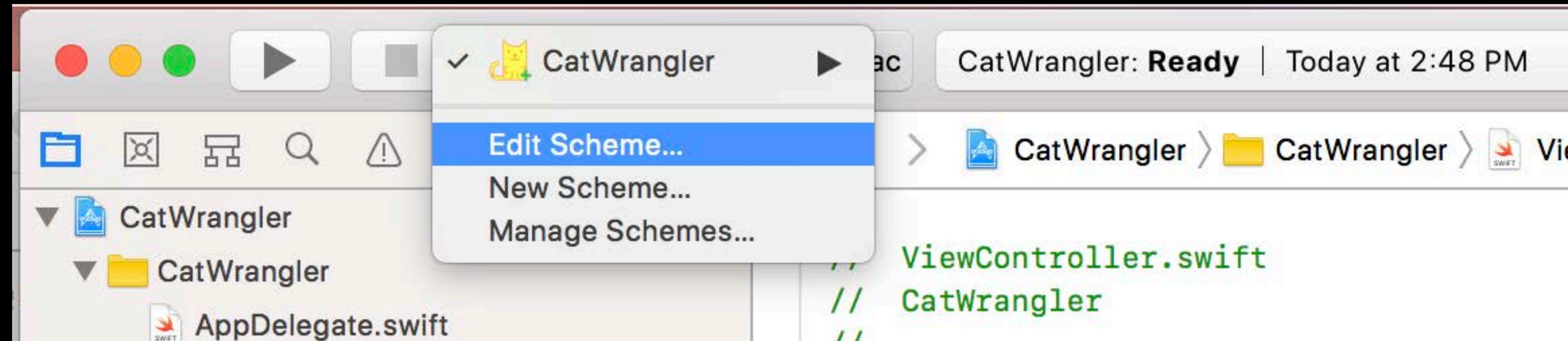
Vincent Hittson, Cocoa Engineer

π

Internationalization

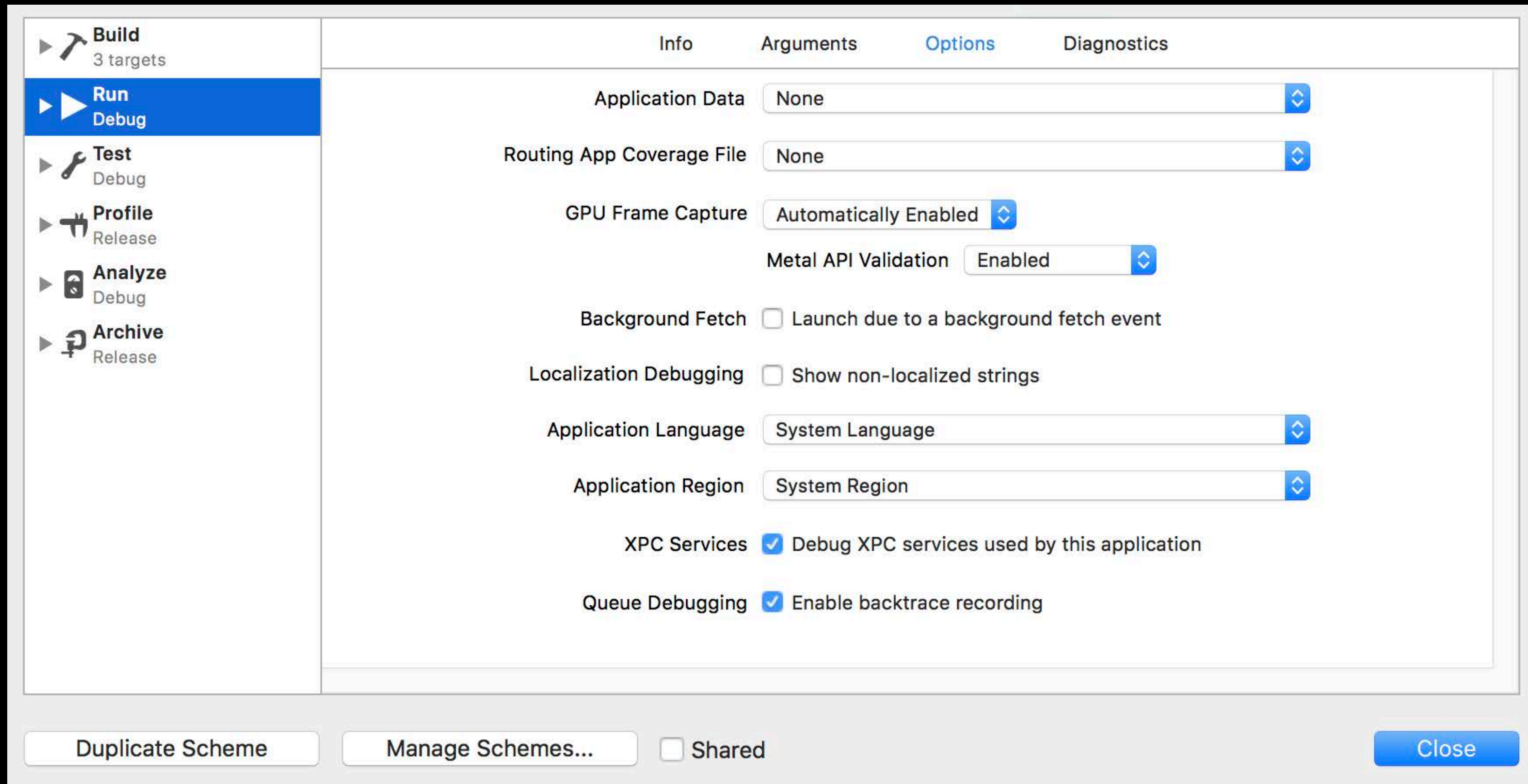
Internationalization

Xcode Scheme Options



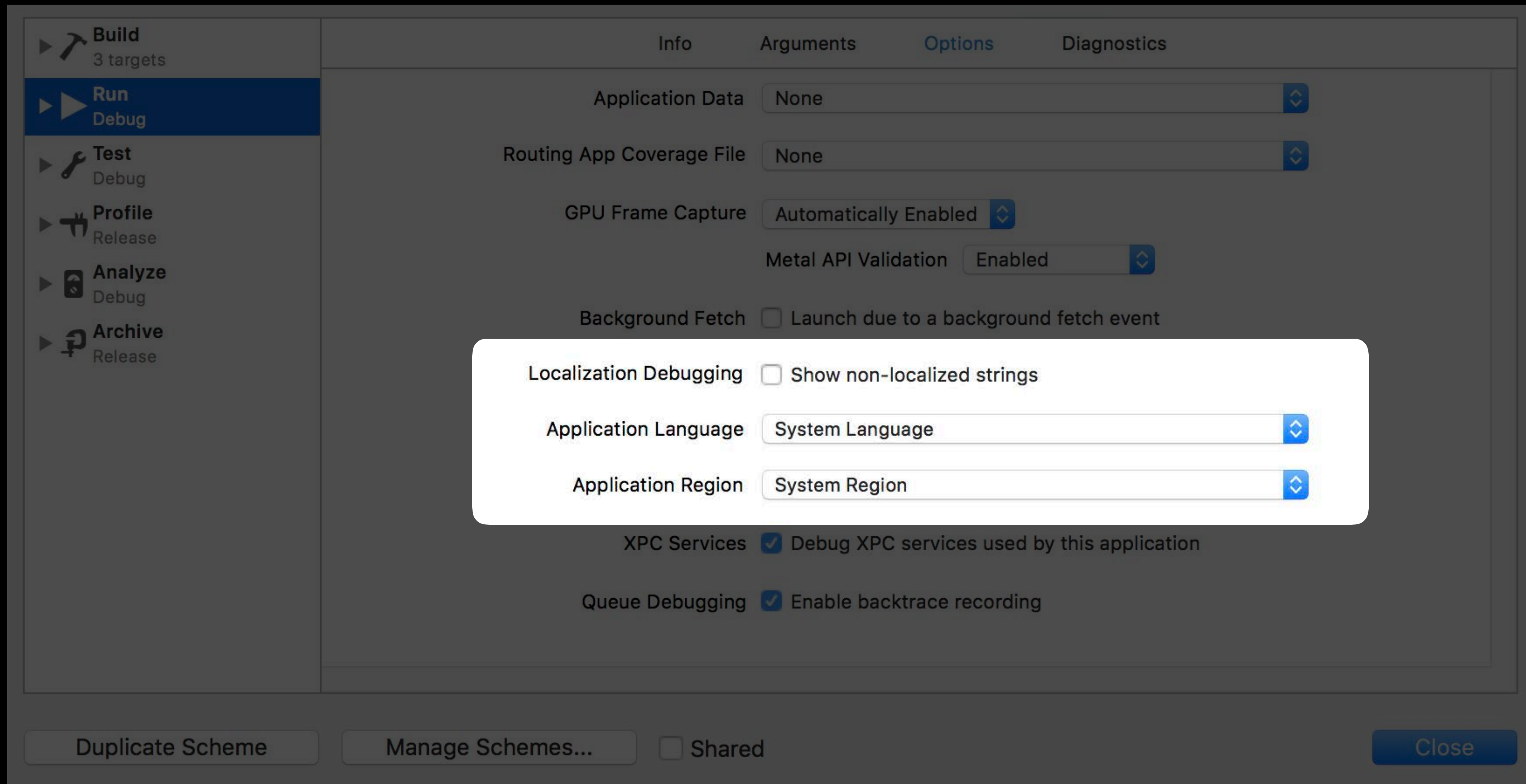
Internationalization

Xcode Scheme Options



Internationalization

Xcode Scheme Options



Internationalization

Xcode Scheme Options

Localization Debugging Show non-localized strings

Application Language

Application Region

Internationalization

Xcode Scheme Options

Localization Debugging Show non-localized strings

Application Language System Language



Application Region System Region



Internationalization

Xcode Scheme Options

Ukrainian

Indonesian

Malay

Vietnamese

Hindi

Double Length Pseudolanguage

Right to Left Pseudolanguage

Accented Latin Pseudolanguage

Bounded String Pseudolanguage

Right to Left Pseudolanguage With Right to Left Strings

0

User Defaults

User Defaults

UserDefaults.standard

User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

```
UserDefaults.standard.object(forKey:"showAtLaunch")
```


User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

```
UserDefaults.standard.object(forKey: "showAtLaunch")
```

User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

```
UserDefaults.standard.register(["showAtLaunch": true])
```

User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

```
UserDefaults.standard.register(["showAtLaunch": true])
```



User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

Temporary

`UserDefaults.standard.register(["showAtLaunch": true])`

User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

```
$ defaults write NSGlobalDomain \  
    showAtLaunch -bool YES
```

User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

```
$ defaults write NSGlobalDomain \  
  showAtLaunch -bool YES
```

User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

Persistent

```
$ defaults write NSGlobalDomain \  
  showAtLaunch -bool YES
```

User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain

```
UserDefaults.standard.set(true, forKey:"showAtLaunch")
```

User Defaults

Argument Domain

Application Domain

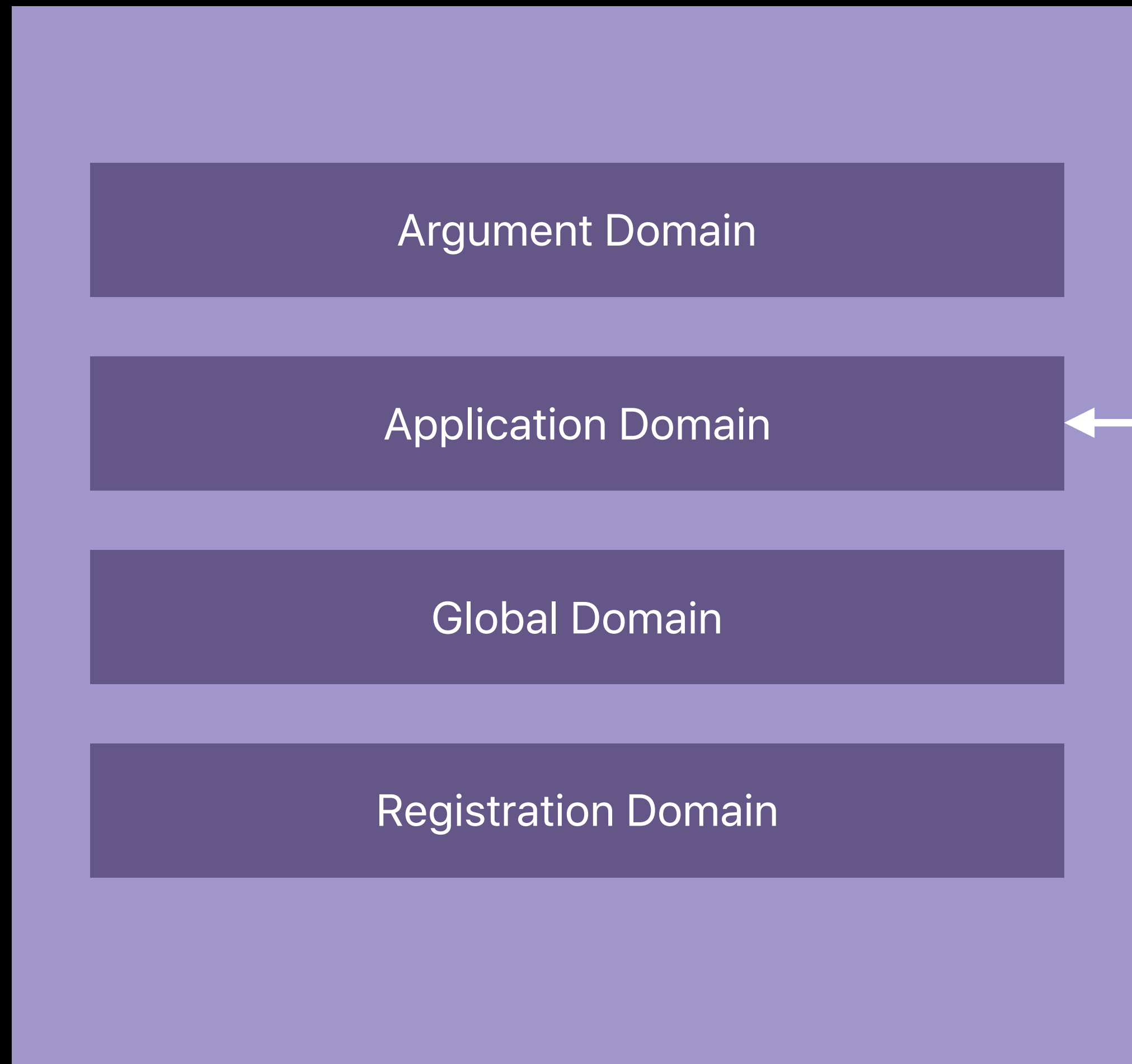
Global Domain

Registration Domain

```
UserDefaults.standard.set(true, forKey:"showAtLaunch")
```

A diagram illustrating the selection of a user defaults domain. On the left, a light purple rectangle contains four stacked, darker purple bars representing different domains: Argument Domain, Application Domain, Global Domain, and Registration Domain. A white arrow points from the Application Domain bar to a blue rectangular box on the right. This box contains the code snippet: UserDefaults.standard.set(true, forKey:"showAtLaunch").

User Defaults



Persistent

`UserDefaults.standard.set(true, forKey:"showAtLaunch")`

User Defaults

Argument Domain

Application Domain

Global Domain

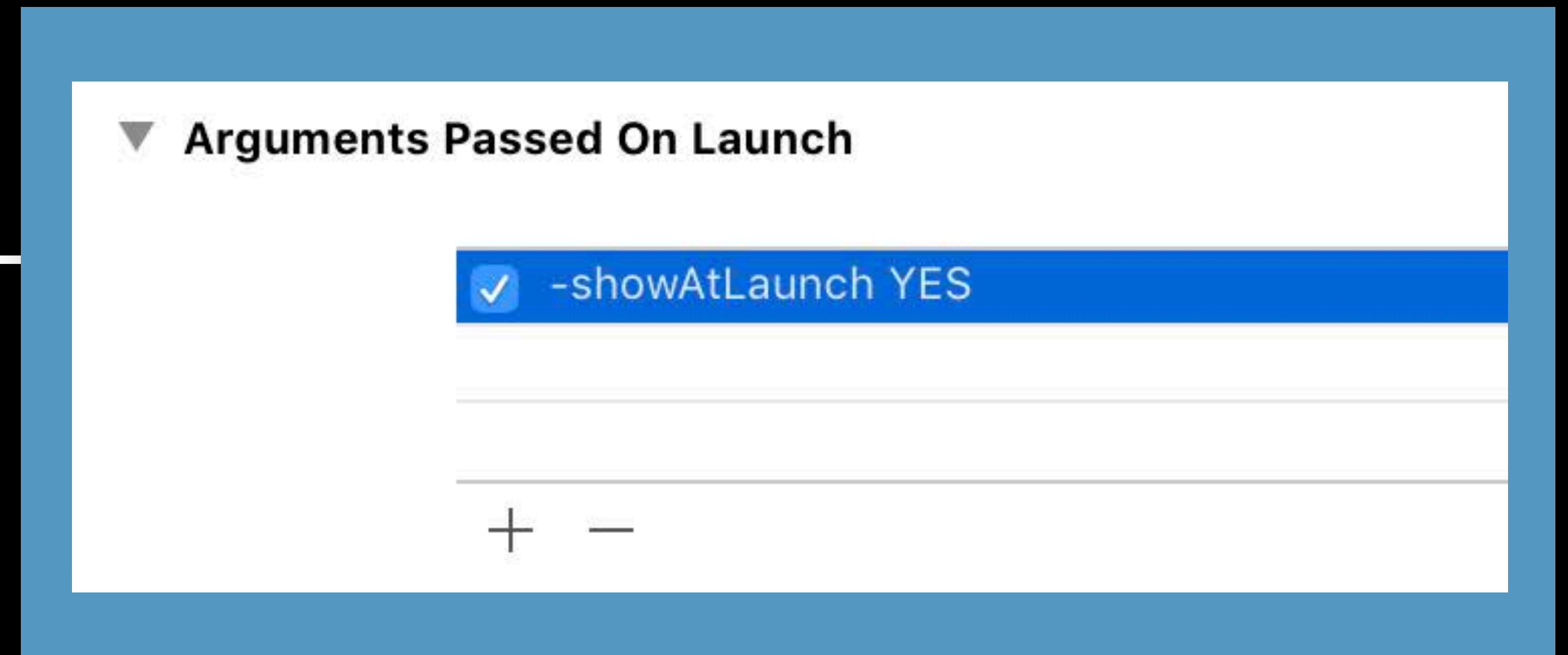
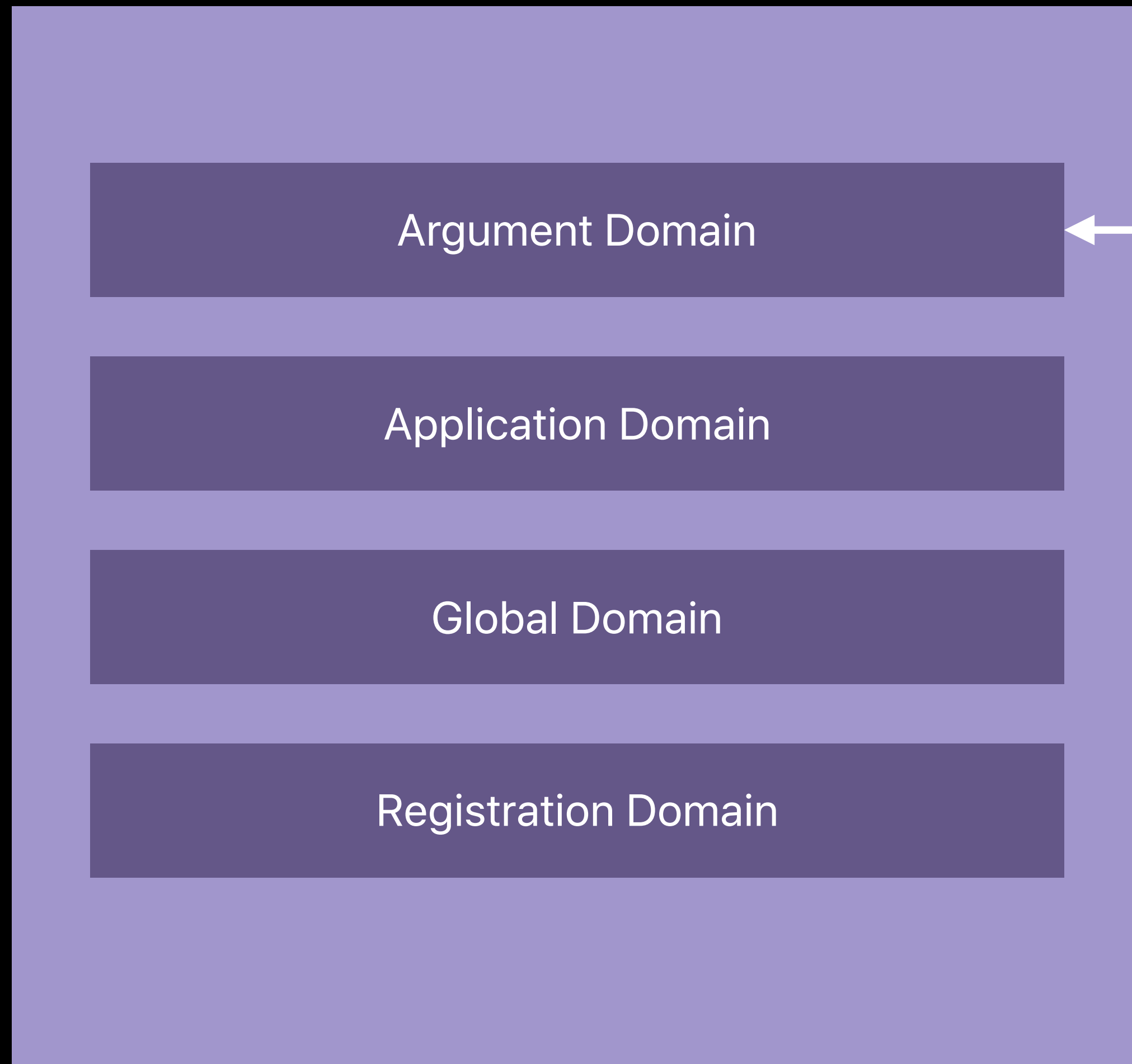
Registration Domain

▼ Arguments Passed On Launch

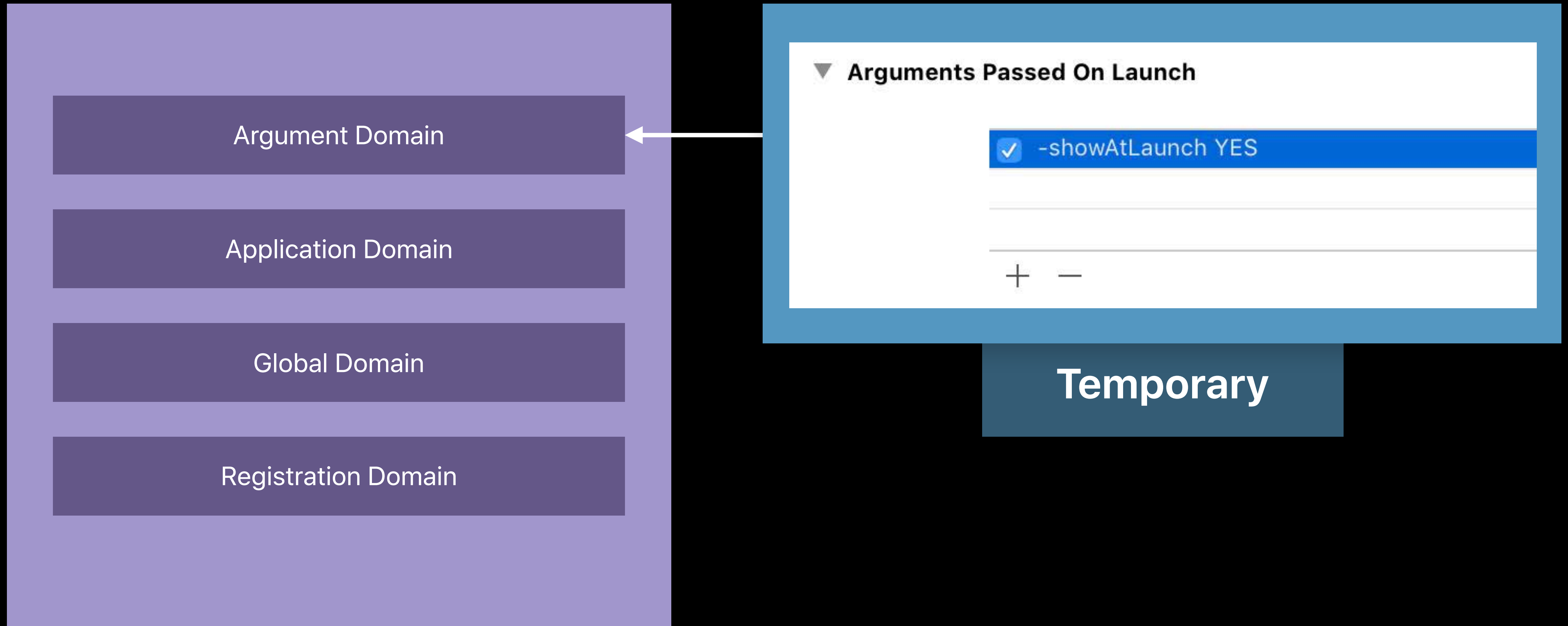
-showAtLaunch YES

+ -

User Defaults



User Defaults



User Defaults

User Defaults

```
-NSViewLayoutFeedbackLoopDebuggingEnabled YES
```


User Defaults

```
-NSViewLayoutFeedbackLoopDebuggingEnabled YES
```

```
-NSApplicationCrashOnExceptions YES
```

User Defaults

Key-value observing is supported for `UserDefaults.standard`

- Add an observer with the user defaults key as the `keyPath`
- Supports cross-process observation

```
class MyUserDefaultsObserver: NSObject {
    private var ctx = 0
    func startObserving() {
        UserDefaults.standard.addObserver(self, forKeyPath: "showAtLaunch", options: [],
                                          context: &ctx)
    }
    func stopObserving {
        UserDefaults.standard.removeObserver(self, forKeyPath: "showAtLaunch", context: &ctx)
    }
    override func observeValue(forKeyPath keyPath: String?, of object: Any?, ..., context: ...) {
        if context == &ctx {
            preferencesChanged()
        }
        else {
            super.observeValue(forKeyPath: keyPath, of: object, change: change, context: ...)
        }
    }
}
```

```
class MyUserDefaultsObserver: NSObject {
    private var ctx = 0
    func startObserving() {
        UserDefaults.standard.addObserver(self, forKeyPath: "showAtLaunch", options: [],
                                         context: &ctx)
    }
    func stopObserving {
        UserDefaults.standard.removeObserver(self, forKeyPath: "showAtLaunch", context: &ctx)
    }
    override func observeValue(forKeyPath keyPath: String?, of object: Any?, ..., context: ...) {
        if context == &ctx {
            preferencesChanged()
        }
        else {
            super.observeValue(forKeyPath: keyPath, of: object, change: change, context: ...)
        }
    }
}
```

```
class MyUserDefaultsObserver: NSObject {
    private var ctx = 0
    func startObserving() {
        UserDefaults.standard.addObserver(self, forKeyPath: "showAtLaunch", options: [],
                                          context: &ctx)
    }
    func stopObserving {
        UserDefaults.standard.removeObserver(self, forKeyPath: "showAtLaunch", context: &ctx)
    }
    override func observeValue(forKeyPath keyPath: String?, of object: Any?, ..., context: ...) {
        if context == &ctx {
            preferencesChanged()
        }
        else {
            super.observeValue(forKeyPath: keyPath, of: object, change: change, context: ...)
        }
    }
}
```

```
class MyUserDefaultsObserver: NSObject {
    private var ctx = 0
    func startObserving() {
        UserDefaults.standard.addObserver(self, forKeyPath: "showAtLaunch", options: [],
                                          context: &ctx)
    }
    func stopObserving {
        UserDefaults.standard.removeObserver(self, forKeyPath: "showAtLaunch", context: &ctx)
    }
    override func observeValue(forKeyPath keyPath: String?, of object: Any?, ..., context: ...) {
        if context == &ctx {
            preferencesChanged()
        }
        else {
            super.observeValue(forKeyPath: keyPath, of: object, change: change, context: ...)
        }
    }
}
```

```
class MyUserDefaultsObserver: NSObject {
    private var ctx = 0
    func startObserving() {
        UserDefaults.standard.addObserver(self, forKeyPath: "showAtLaunch", options: [],
                                         context: &ctx)
    }
    func stopObserving {
        UserDefaults.standard.removeObserver(self, forKeyPath: "showAtLaunch", context: &ctx)
    }
    override func observeValue(forKeyPath keyPath: String?, of object: Any?, ..., context: ...) {
        if context == &ctx {
            preferencesChanged()
        }
        else {
            super.observeValue(forKeyPath: keyPath, of: object, change: change, context: ...)
        }
    }
}
```

User Defaults

User Defaults

UserDefaults.standard

My Application

User Defaults

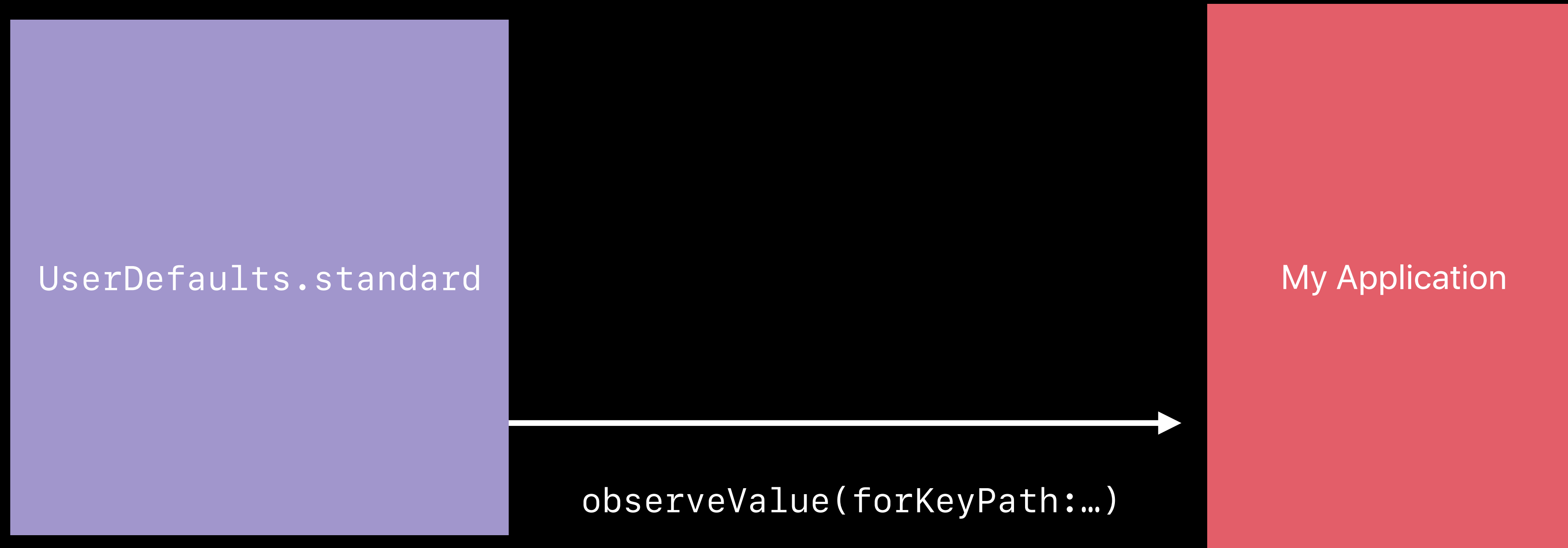


User Defaults

UserDefaults.standard

My Application

User Defaults



User Defaults

UserDefaults.standard

My Application

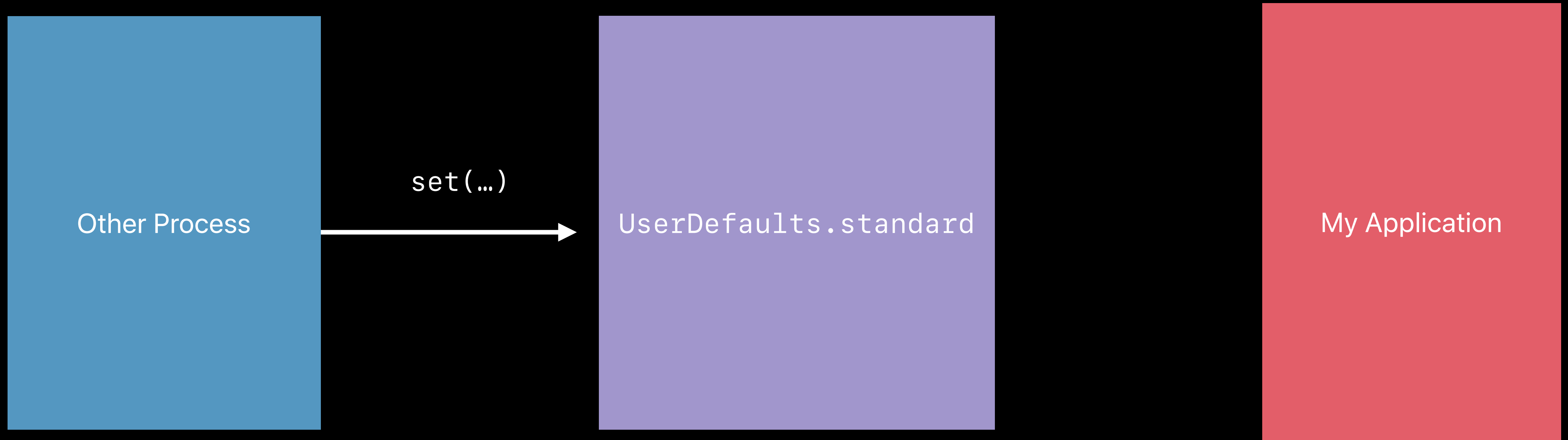
User Defaults

Other Process

UserDefaults.standard

My Application

User Defaults



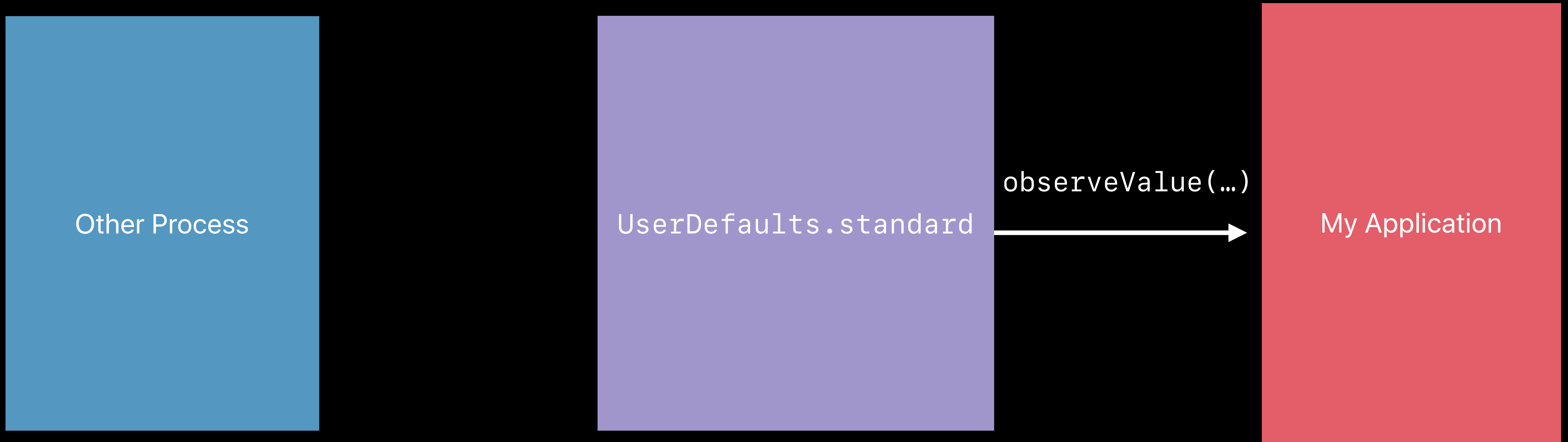
User Defaults

Other Process

UserDefaults.standard

My Application

User Defaults



User Defaults

Other Process

UserDefaults.standard

My Application

```
extension UserDefaults {  
    @objc dynamic var showAtLaunch: Bool {  
        return self.value(forKey:"showAtLaunch") as? Bool ?? false  
    }  
}  
  
let observation = UserDefaults.standard.observe(\.showAtLaunch) { observed, change in  
    preferencesChanged()  
}
```

```
extension UserDefaults {
    @objc dynamic var showAtLaunch: Bool {
        return self.value(forKey:"showAtLaunch") as? Bool ?? false
    }
}

let observation = UserDefaults.standard.observe(\.showAtLaunch) { observed, change in
    preferencesChanged()
}
```

```
extension UserDefaults {  
    @objc dynamic var showAtLaunch: Bool {  
        return self.value(forKey:"showAtLaunch") as? Bool ?? false  
    }  
}  
  
let observation = UserDefaults.standard.observe(\.showAtLaunch) { observed, change in  
    preferencesChanged()  
}
```

```
extension UserDefaults {  
    @objc dynamic var showAtLaunch: Bool {  
        return self.value(forKey:"showAtLaunch") as? Bool ?? false  
    }  
}
```

```
let observation = UserDefaults.standard.observe(\.showAtLaunch) { observed, change in  
    preferencesChanged()  
}
```

```
extension UserDefaults {  
    @objc dynamic var showAtLaunch: Bool {  
        return self.value(forKey:"showAtLaunch") as? Bool ?? false  
    }  
}
```

```
let observation = UserDefaults.standard.observe(\.showAtLaunch) { observed, change in  
    preferencesChanged()  
}
```

64

QmFzZSA2NAo=

64

Base 64

```
let data64 = "//lsYAFgAADQQAf/+WxgD6AAAPsbwD4pTJg0QK0CZA2QK0CZCzkIEyAwRLb/j/uqvrPd68f+T8v+j/5  
d+3v89/P/p/pf/rn+PA+kH0HY8NoBsAMCKIu4JCcHNa4XAPzvtvvoaHatuD3eKQzN1NdNZzsb9+8P5bDV8v/ZuNVFSCg1  
1bGixouwcP/5bGApQAABANGAJFAghgdjEIFsoCcNjAId/9vfvv8Lfvxx0s64Bse/v7j4b+Kff7tNA8ywhZqZoSVUqr/LG  
ahe090T5w8u5R01m3kv0T8b7HvfCN17LwzsiPfg/tdl64//N2Rc9mAABorJxTzCoPtzL6o8rCMYow0cdQeLDjb5Dj3tog  
Ut1rupUyJr6vfG8fjq9mm3B3RhFuDIQp5CrSIY7AkMPnCGnypF9Enh7PCrv1KRogFGFLAALR63FtFhIKITpWiFXLk4mmJ  
GyhGPgSVCYRoDJKRwkAmx0QGyqPnf+BnQhFiieDxZKPVIazbEZPEife0wkKH10h8DvhGQ9CIJ+X/cRP256DJ+A9oQqbIh  
u9aQyEAnLyZPY8TJ846gT6Xygnr5RCBPirWtvZYnpdQTz2KtaaQ12yFaBmHv8mddTm0RjgUs0AAA4P/5bGAQQAAA9hGAS  
EA8EBCBsQEoLhAJ2//tet7/6x8g00sSGae3rM3VUoc5RwmpFzXBc5yL2UUFRSJKWV1d1XfNV4qhPCBEkvBDGoUIiCAaEm  
DvymS7n+/569VIjJvJcFPgAAARRRIBIkVWksRQrFSxa0P2/K6fre78DwfpWtooAA7/+WxgCsAAAPQRgGhgMYCCbCoQI4  
RCAWw//ty+XI14yBxb16igW28pmWXYejHf/riV3BvCOEAAEzTI0d5phGYo2+v+HzxgAAAKvV+L6nwtLw4soBwP/5bGAP  
oAAA9BGAZCAJJgQhAZCAQhoIhAjiAJoFMz18P2c62HfeE/xm+IPeH4MPcZMyfeF/M7718ZnfeHIZ33JQnCW7J6muscXBt  
qrbRpag2fQf4Nn0H+BHLYJ1nVHVOPdW253mWvMfBgAC11nj11W3k6vofG8vm5SgAAHA"
```

```
let soundData = Data(base64Encoded: data64)!  
let sound = NSSound(data: soundData)!  
sound.play()
```

```
let data64 = "//lsYAFgAADQQAf/+WxgD6AAAPsbwD4pTJg0QK0CZA2QK0CZCzkIEyAwRLb/j/uqvrPd68f+T8v+j/5  
d+3v89/P/p/pf/rn+PA+kH0HY8NoBsAMCKIu4JCcHNa4XAPzvtvvoaHatuD3eKQzN1NdNZzsb9+8P5bDV8v/ZuNVFSCg1  
1bGixouwcP/5bGApQAABANGAJFAghgdjEIFsoCcNjAId/9vfvv8Lfvxx0s64Bse/v7j4b+Kff7tNA8ywhZqZoSVUqr/LG  
ahe090T5w8u5R01m3kv0T8b7HvfCN17LwzsiPfg/tdl64//N2Rc9mAABorJxTzCoPtzL6o8rCMYow0cdQeLDjb5Dj3tog  
Ut1rupUyJr6vfG8fjq9mm3B3RhFuDIQp5CrSIY7AkMPnCGnypF9Enh7PCrv1KRogFGFLAALR63FtFhIKITpWiFXLk4mmJ  
GyhGPgSVCYRoDJKRwkAmx0QGyqPnf+BnQhFiieDxZKPVIazbEZPEife0wkKH10h8DvhGQ9CIJ+X/cRP256DJ+A9oQqbIh  
u9aQyEAnLyZPY8TJ846gT6Xygnr5RCBPiRwTvZYnpdQTz2KtaaQ12yFaBmHv8mddTm0RjgUs0AAA4P/5bGAQQAAA9hGAS  
EA8EBCBsQEoLhAJ2//tet7/6x8g00sSGae3rM3VUoc5RwmpFzXBc5yL2UUFRSJKWV1d1XfNV4qhPCBEkvBDGoUIiCAaEm  
DvymS7n+/569VIjJvJcFPgAAARRRIBIkVWksRQrFSxa0P2/K6fre78DwfpWtooAA7/+WxgCsAAAPQRgGhgMYCCbCoQI4  
RCAWw//ty+XI14yBxb16igW28pmWXYejHf/riV3BvCOEAAEzTI0d5phGYo2+v+HzxgAAAKvV+L6nwtLw4soBwP/5bGAP  
oAAA9BGAZCAJJgQhAZCAQhoIhAjiAJoFMz18P2c62HfeE/xm+IPeH4MPcZMyfeF/M7718ZnfeHIZ33JQnCW7J6muscXBt  
qrbRpag2fQf4Nn0H+BHLYJ1nVHVOPdW253mWvMfBgAC11nj11W3k6vofG8vm5SgAAHA"
```

```
let soundData = Data(base64Encoded: data64)!  
let sound = NSSound(data: soundData)!  
sound.play()
```

```
let data64 = "//lsYAFgAADQQAf/+WxgD6AAAPsbwD4pTJg0QK0CZA2QK0CZCzkIEyAwRLb/j/uqvrPd68f+T8v+j/5  
d+3v89/P/p/pf/rn+PA+kH0HY8NoBsAMCKIu4JCcHNa4XAPzvtvvoaHatuD3eKQzN1NdNZzsb9+8P5bDV8v/ZuNVFSCg1  
1bGixouwcP/5bGApQAABANGAJFAghgdjEIFsoCcNjAId/9vfvv8Lfvxx0s64Bse/v7j4b+Kff7tNA8ywhZqZoSVUqr/LG  
ahe090T5w8u5R01m3kv0T8b7HvfCN17LwzsiPfg/tdl64//N2Rc9mAABorJxTzCoPtzL6o8rCMYow0cdQeLDjb5Dj3tog  
Ut1rupUyJr6vfG8fjq9mm3B3RhFuDIQp5CrSIY7AkMPnCGnypF9Enh7PCrv1KRogFGFLAALR63FtFhIKITpWiFXLk4mmJ  
GyhGPgSVCYRoDJKRwkAmx0QGyqPnf+BnQhFiieDxZKPVIazbEZPEife0wkKH10h8DvhGQ9CIJ+X/cRP256DJ+A9oQqbIh  
u9aQyEAnLyZPY8TJ846gT6Xygnr5RCBPiRwTvZYnpdQTz2KtaaQ12yFaBmHv8mddTm0RjgUs0AAA4P/5bGAQQAAA9hGAS  
EA8EBCBsQEoLhAJ2//tet7/6x8g00sSGae3rM3VUoc5RwmpFzXBc5yL2UUFRSJKWV1d1XfNV4qhPCBEkvBDGoUIiCAaEm  
DvymS7n+/569VIjJvJcFPgAAARRRIBIkVWksRQrFSxa0P2/K6fre78DwfpWtooAA7/+WxgCsAAAPQRgGhgMYCCbCoQI4  
RCAWw//ty+XI14yBxb16igW28pmWXYejHf/riV3BvCOEAAEzTI0d5phGYo2+v+HzxgAAAKvV+L6nwtLw4soBwP/5bGAP  
oAAA9BGAZCAJJgQhAZCAQhoIhAjiAJoFMz18P2c62HfeE/xm+IPeH4MPcZMyfeF/M7718ZnfeHIZ33JQnCW7J6muscXBt  
qrbRpag2fQf4Nn0H+BHLYJ1nVHVOPdW253mWvMfBgAC11nj11W3k6vofG8vm5SgAAHA"
```

```
let soundData = Data(base64Encoded: data64)!  
let sound = NSSound(data: soundData)!  
sound.play()
```



```
let data64 = "//lsYAFgAADQQAf/+WxgD6AAAPSBwD4pTJg0QK0CZA2QK0CZCzkIEyAwRLb/j/uqvrPd68f+T8v+j/5  
d+3v89/P/p/pf/rn+PA+kH0HY8NoBsAMCKIu4JCcHNa4XAPzvtvvoaHatuD3eKQzN1NdNZzsb9+8P5bDV8v/ZuNVFSCg1  
1bGixouwcP/5bGApQAABANGAJFAghgdjEIFsoCcNjAId/9vfvv8Lfvxx0s64Bse/v7j4b+Kff7tNA8ywhZqZoSVUqr/LG  
ahe090T5w8u5R01m3kv0T8b7HvfCN17LwzsiPfg/tdl64//N2Rc9mAABorJxTzCoPtzL6o8rCMYow0cdQeLDjb5Dj3tog  
Ut1rupUyJr6vfG8fjq9mm3B3RhFuDIQp5CrSIY7AkMPnCGnypF9Enh7PCrv1KRogFGFLAALR63FtFhIKITpWiFXLk4mmJ  
GyhGPgSVCYRoDJKRwkAmx0QGyqPnf+BnQhFiieDxZKPVIazbEZPEife0wkKH10h8DvhGQ9CIJ+X/cRP256DJ+A9oQqbIh  
u9aQyEAnLyZPY8TJ846gT6Xygnr5RCBPirWtvZYnpdQTz2KtaaQ12yFaBmHv8mddTm0RjgUs0AAA4P/5bGAQQAAA9hGAS  
EA8EBCBsQEoLhAJ2//tet7/6x8g00sSGae3rM3VUoc5RwmpFzXBc5yL2UUFRSJKWV1d1XfNV4qhPCBEkvBDGoUIiCAaEm  
DvymS7n+/569VIjJvJcFPgAAARRRIBIkVWksRQrFSxa0P2/K6fre78DwfpWtooAA7/+WxgCsAAAPQRgGhgMYCCbCoQI4  
RCAWw//ty+XI14yBxb16igW28pmWxEywjHf/riV3BvCOEAAEzTI0d5phGYo2+v+HzxgAAAKvV+L6nwtLw4soBwP/5bGAP  
oAAA9BGAZCAJJgQhAZCAQhoIhAjiAJoFMz18P2c62HfeE/xm+IPeH4MPcZMyfeF/M7718ZnfeHIZ33JQnCW7J6muscXBt  
qrbRpag2fQf4Nn0H+BHLYJ1nVHVOPdW253mWvMfBgAC11nj11W3k6vofG8vm5SgAAHA"
```

```
let soundData = Data(base64Encoded: data64)!  
let sound = NSSound(data: soundData)!  
sound.play()
```

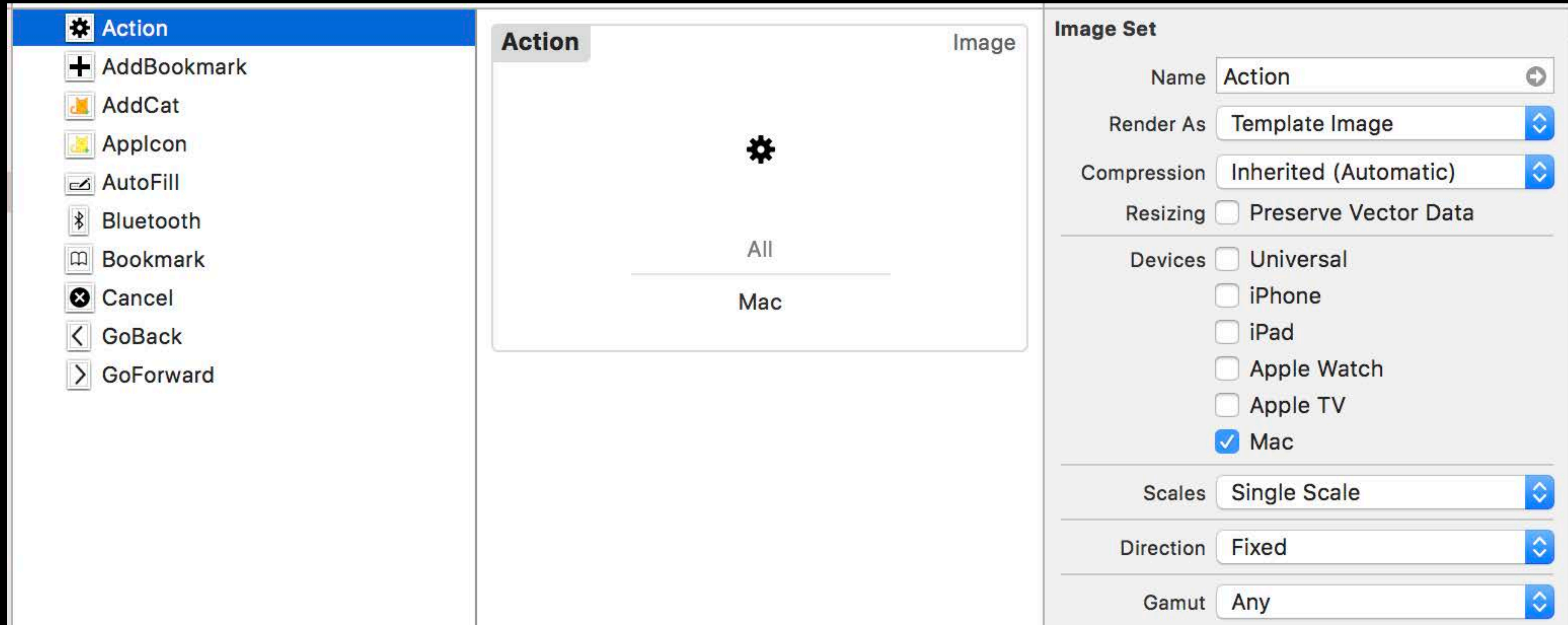
```
soundData.base64EncodedString()
```

```
"//1sYAFgAADQQAf/+WxgD6AAAPSBwD4pTJg0QK0CZA2QK0CZCzkIEyAwRLb/j/uqvrPd68f+T8v+j/5d+3v89/P/p/pf/rn+PA+kH0HY8NoBsAMCKIu4JCcHNa4XAPzvtvoaHatuD3eKQzN1NdNZzsb9+8P5bDV8v/ZuNVFSCg11bGixouwcP/5bGApQAABANGAJFAghgdjEIFsoCcNjAId/9vfvv8Lfvxx0s64Bse/v7j4b+Kff7tNA8ywhZqZoSVUqr/LGahe090T5w8u5R01m3kv0T8b7HvfCN17LwzsiPfg/tdl64//N2Rc9mAABorJxTzCoPtzL6o8rCMYow0cdQeLDjb5Dj3togUt1rupUyJr6vfG8fjq9mm3B3RhFuDIQp5CrSIY7AkMPnCGnypF9Enh7PCrv1KRogFGFLAALR63FtFhIKITpWiFXLk4mmJGyhGPgSVCYRoDJKRwkAmx0QGyqPnf+BnQhFiieDxZKPVIazbEZPEife0wkKH10h8DvhGQ9CIJ+X/cRP256DJ+A9oQqbIhu9aQyEAnLyZPY8TJ846gT6Xygnr5RCBPIrWTvZYnpdQTz2KtaaQ12yFaBmHv8mddTm0RjgUs0AAA4P/5bGAQQAAA9hGASEA8EBCBsQEoLhAJ2//tet7/6x8g00sSGae3rM3VUoc5RwmpFzXBc5yL2UUFRSJKWV1d1XfNV4qhPCBEkvBDGoUIiCAaEmDvymS7n+/569VIjJvJcFPgAAARRRIBIkVWksRQrFSxa0P2/K6fre78DwfpWtooAA7/+WxgCsAAAPQRgGhgMYCCbCoQI4RCaww//ty+XI14yBxb16igW28pmWxEywjHf/riV3BvCOEAAEzTI0d5phGYo2+v+HzxgAAAKvV+L6nwtLw4soBwP/5bGAPoAAA9BGAZCAJJgQhAZCAQhoIhAjiAJoFMz18P2c62HfeE/xm+IPeH4MPcZMyfeF/M7718ZnfeHIZ33JQnCW7J6muscXBtqrbRpag2fQf4Nn0H+BHLYJ1nVHVOPdW253mWvMfBgAC11nj11W3k6vofG8vm5SgAAHA"
```

2X

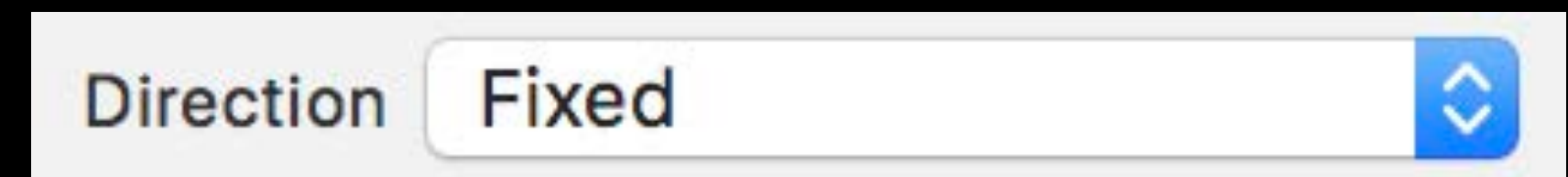
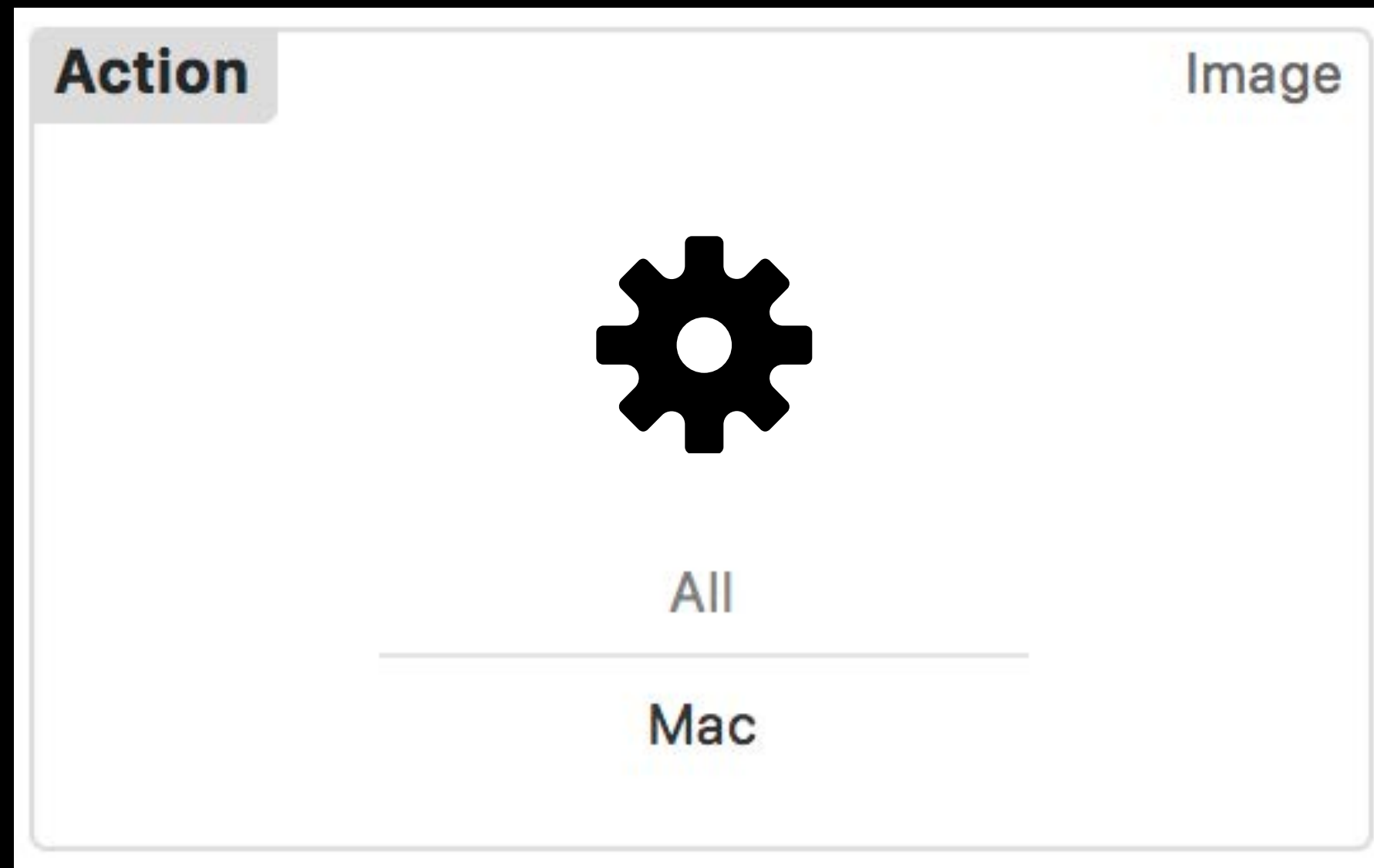
Asset Catalogs

Asset Catalogs



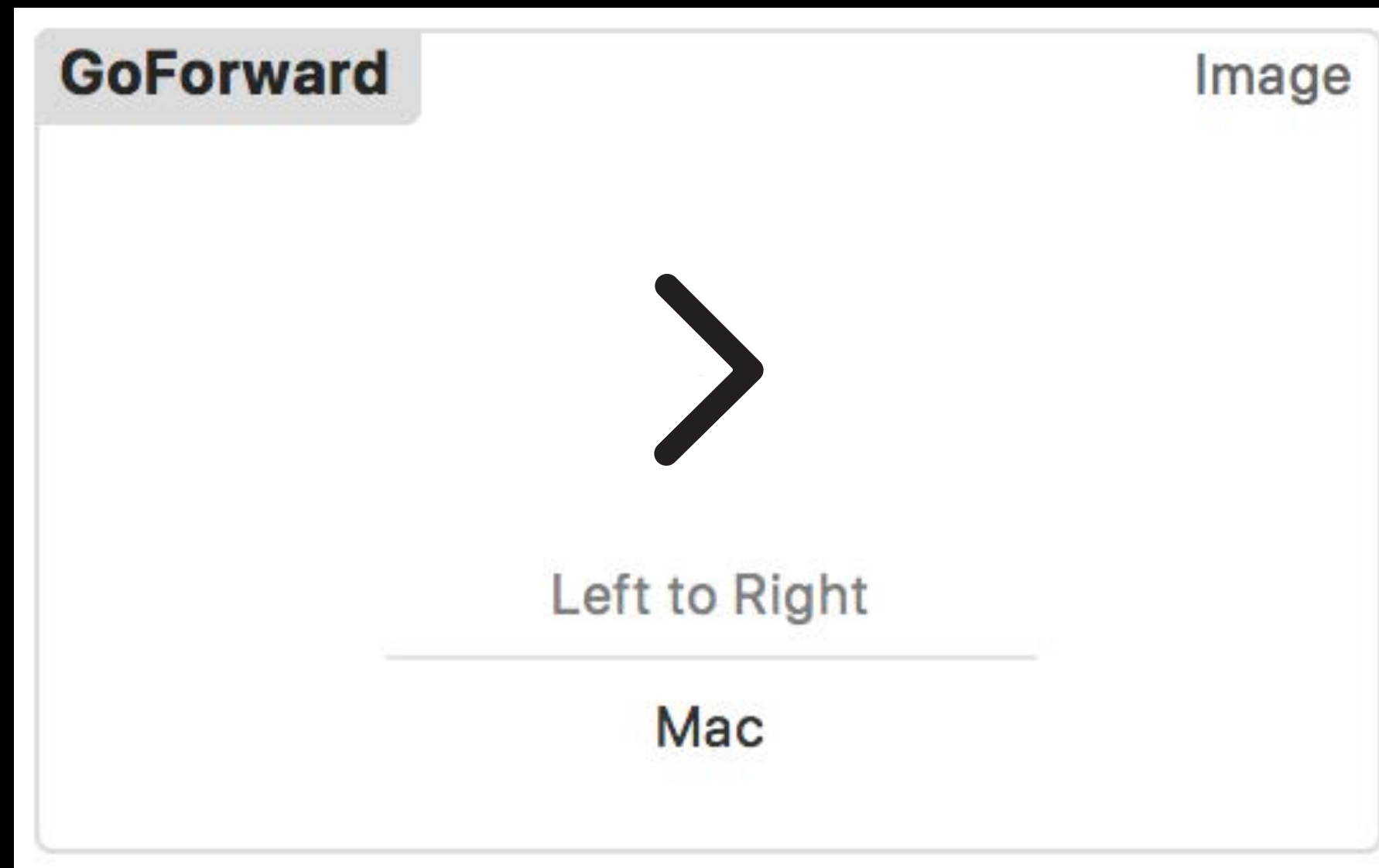
Asset Catalogs

Layout direction



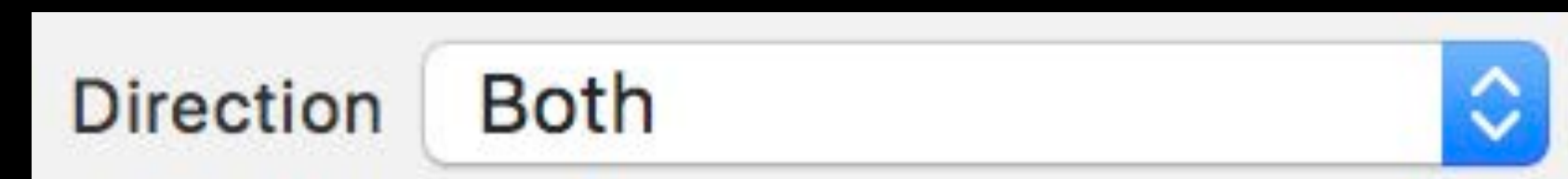
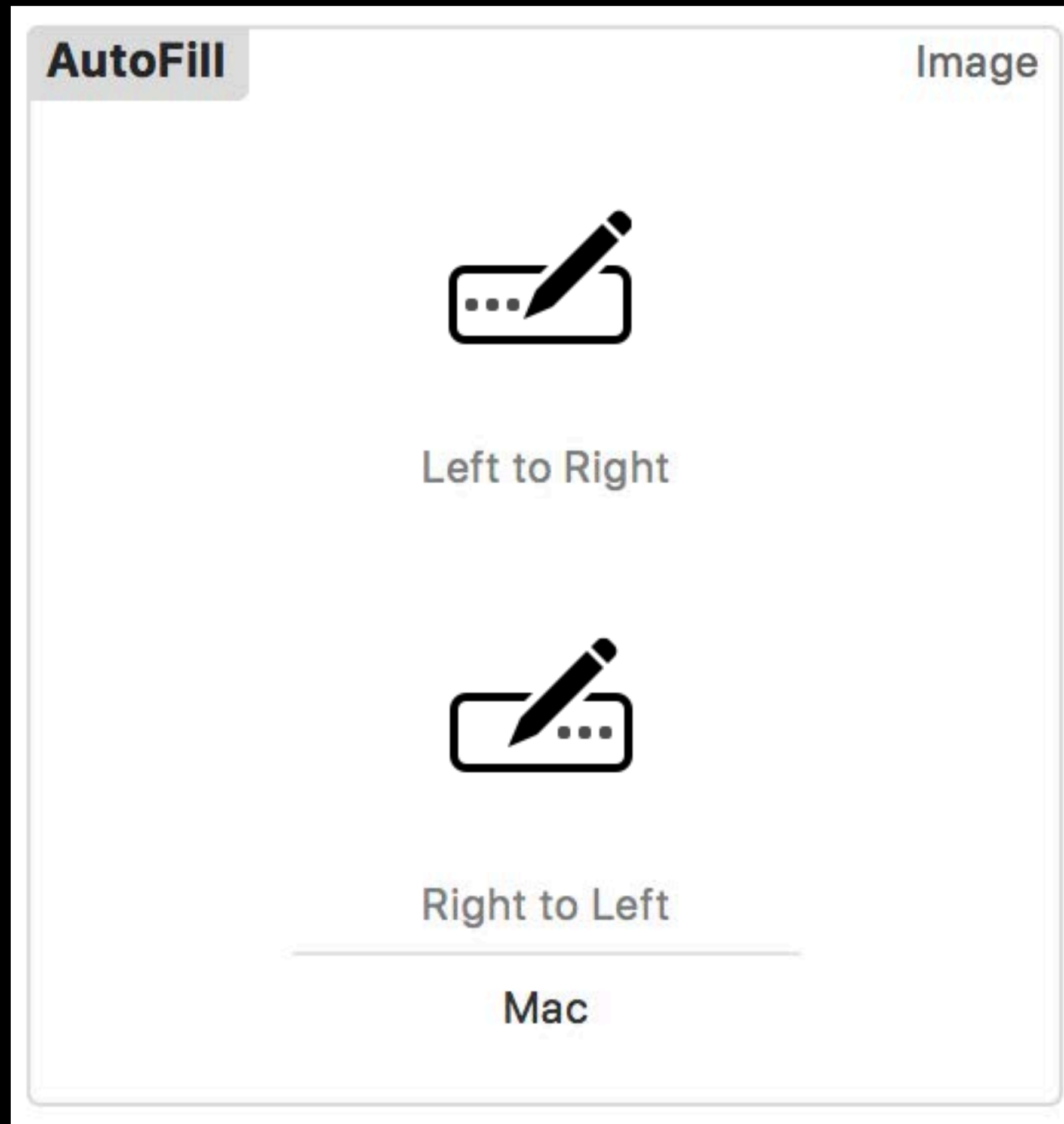
Asset Catalogs

Layout direction



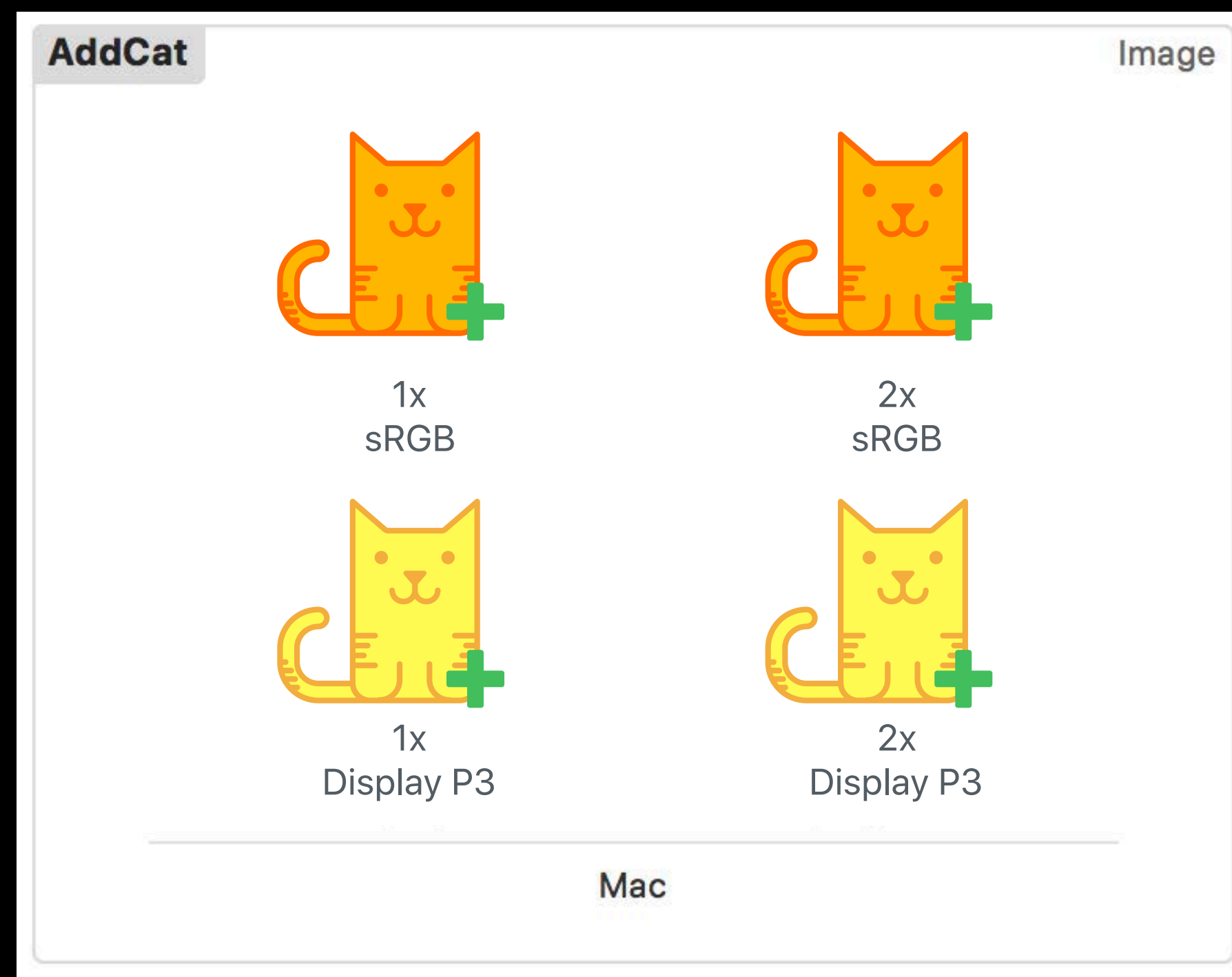
Asset Catalogs

Layout direction



Asset Catalogs

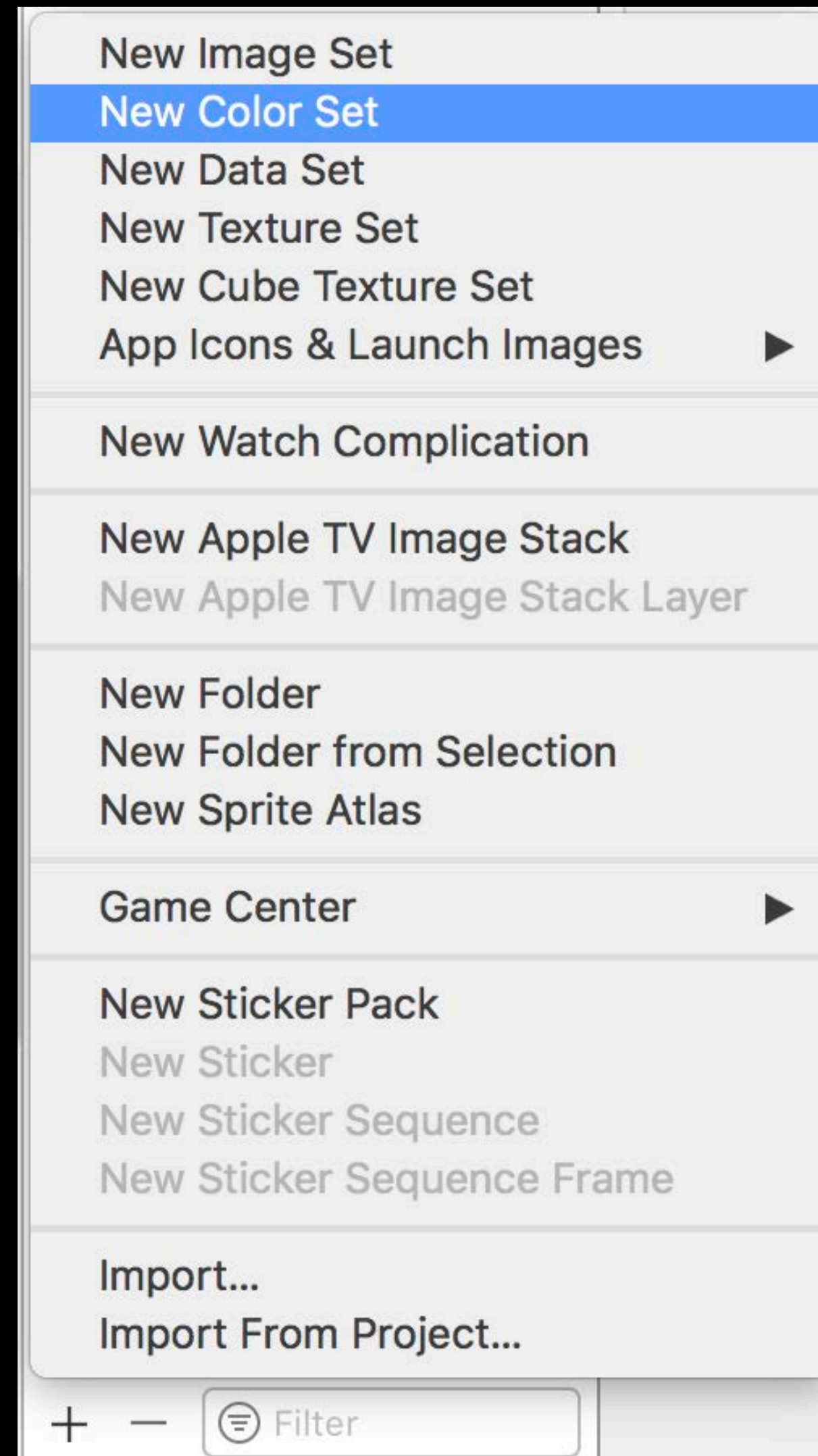
Display gamut



Asset Catalogs

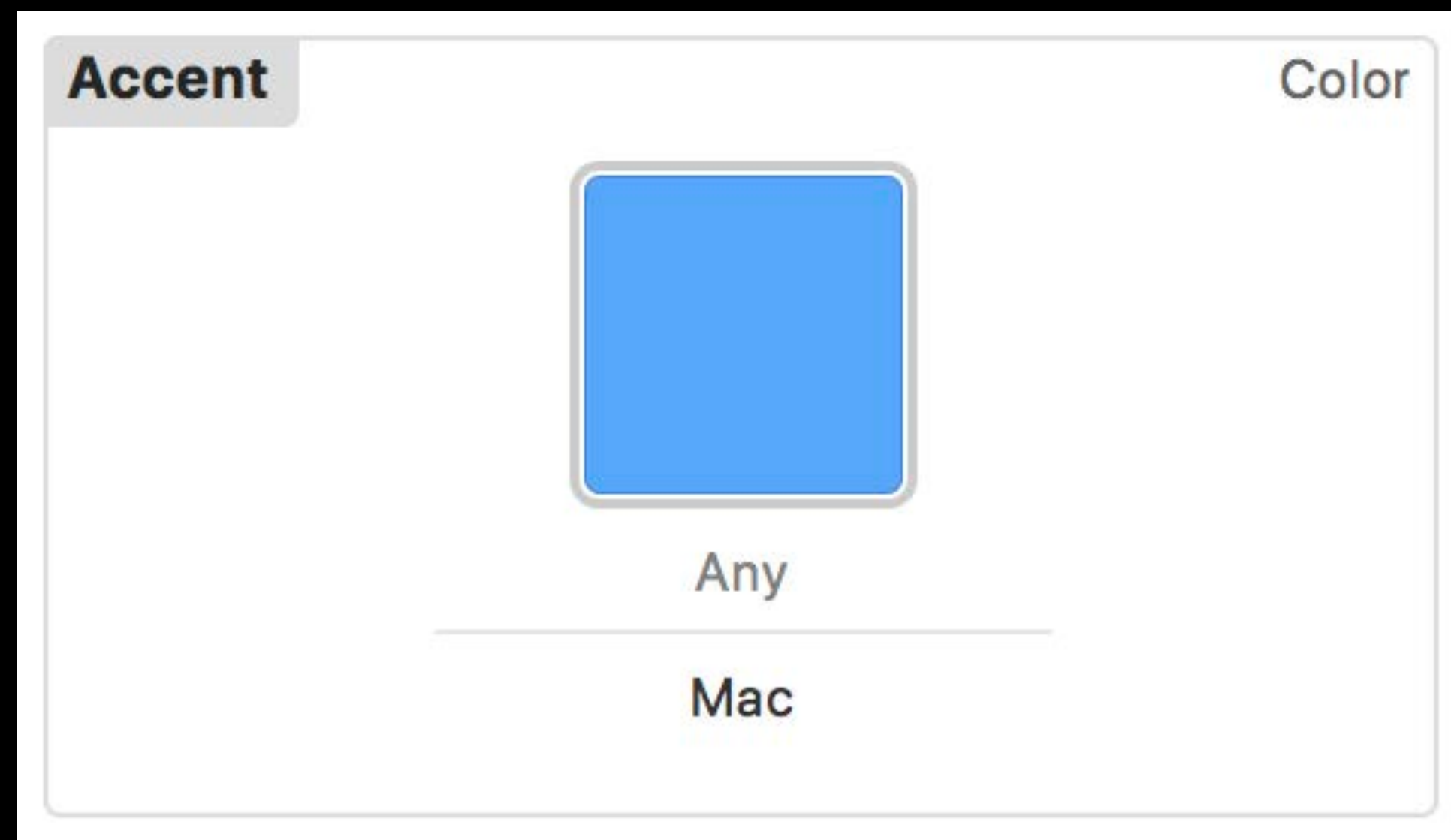
Colors

NEW



Asset Catalogs

Colors



Asset Catalogs

Colors

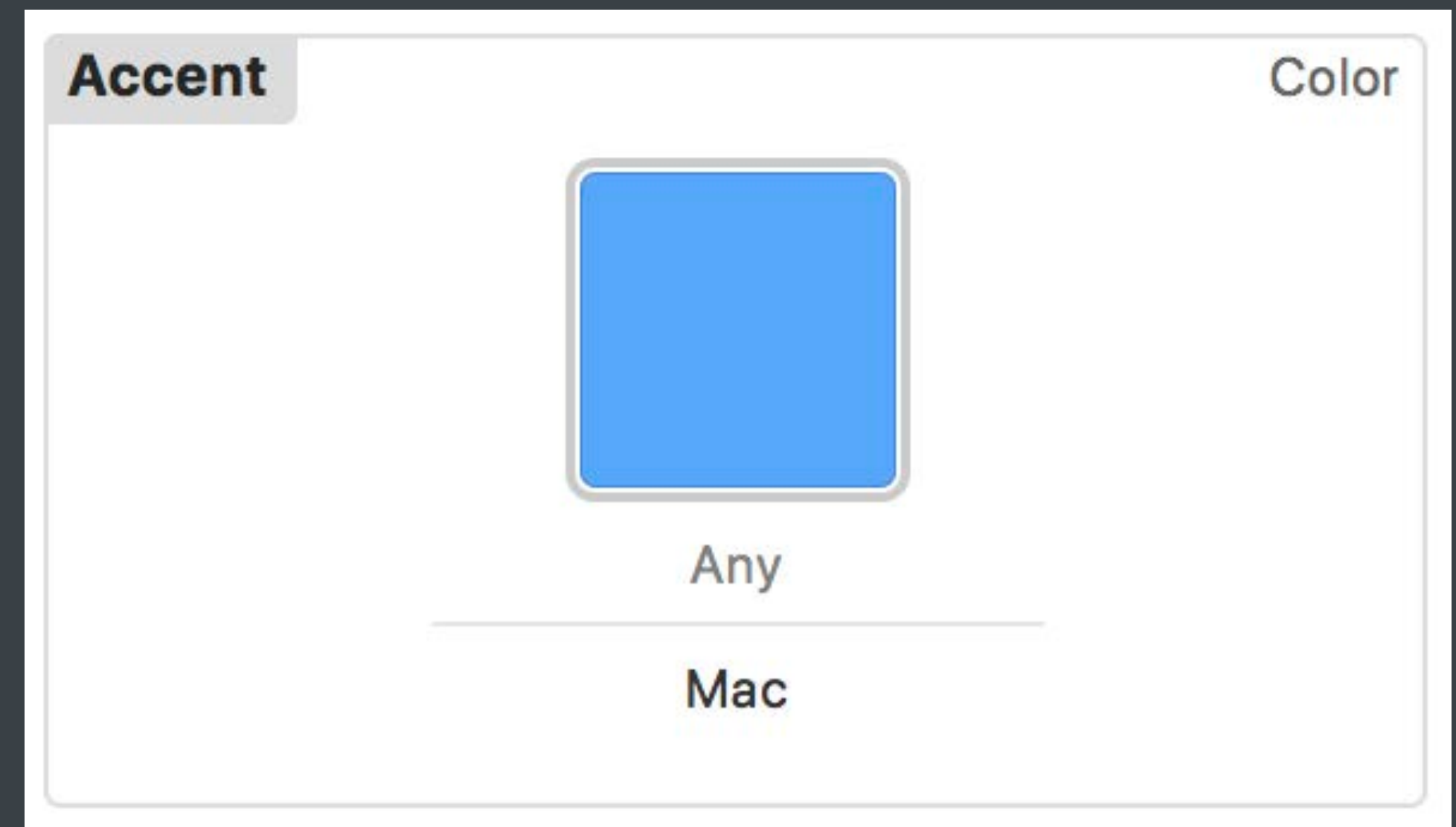


```
//Using colors from Asset Catalogs

//You can add your color name to the extensible UIColor.Name enum
extension UIColor.Name {
    static let accent = UIColor.Name("Accent")
}

//Then you can create your color using that name

let accentColor = UIColor(named: .accent)
```



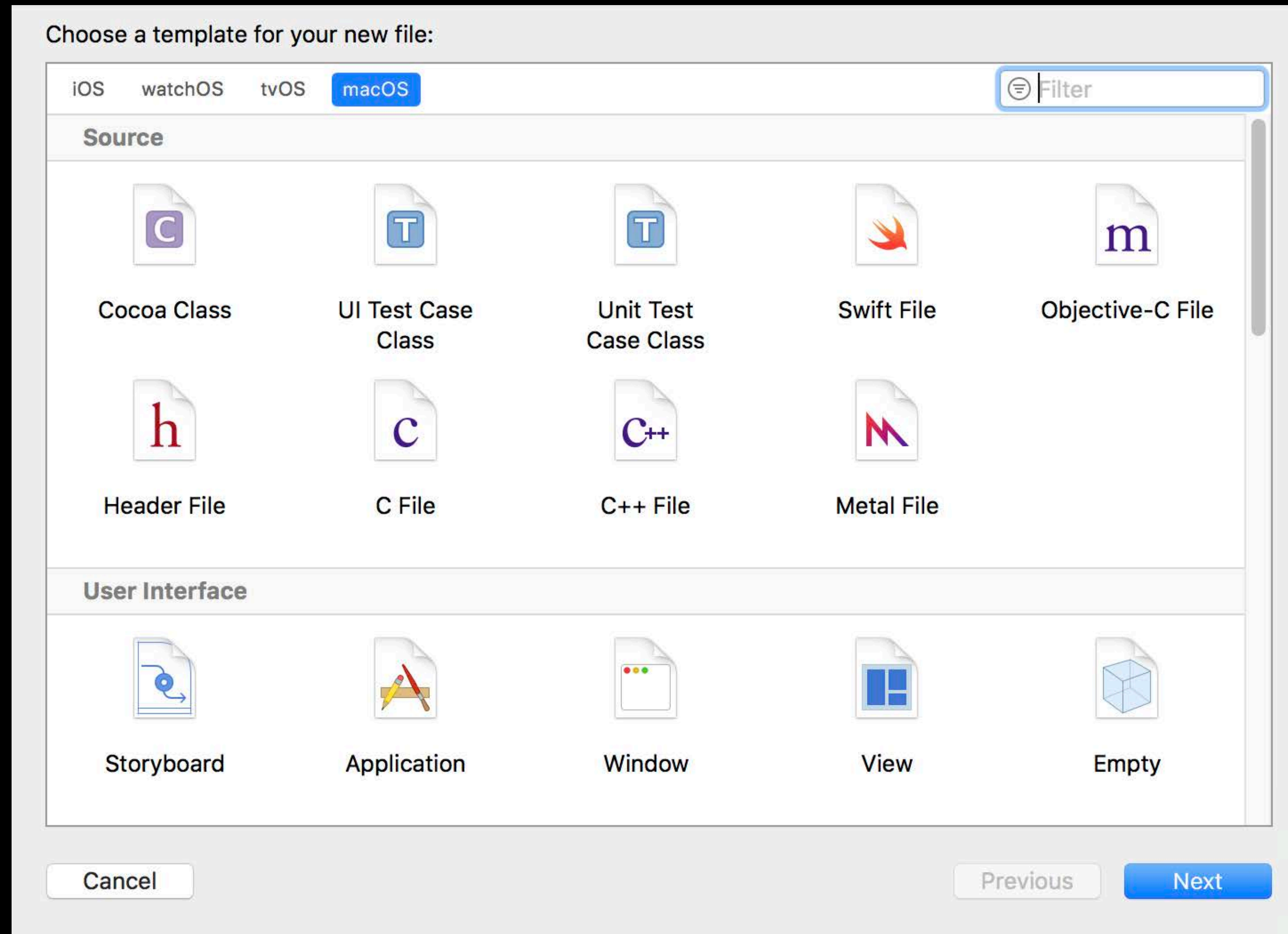
1

Unit Tests

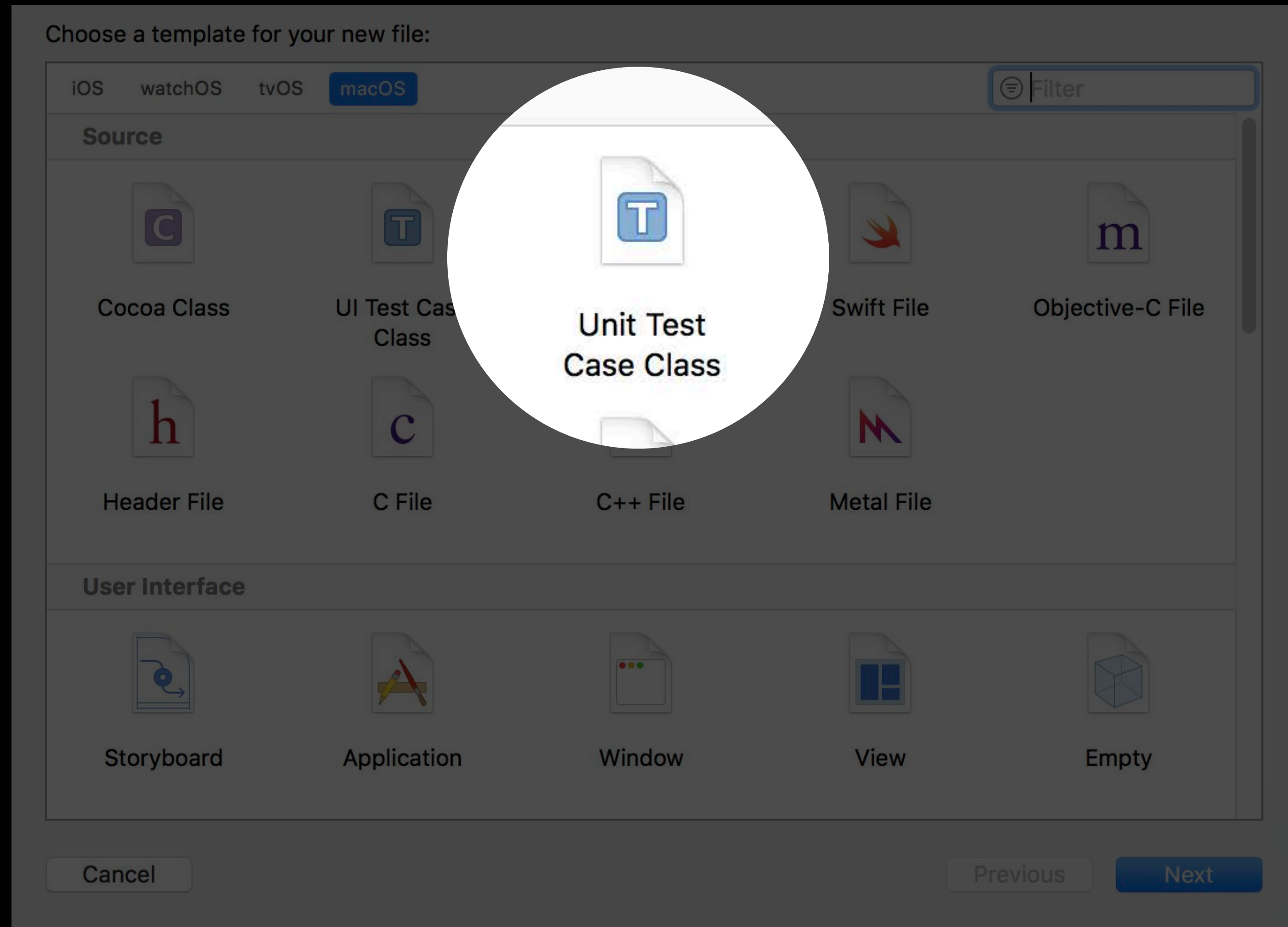
Unit Tests

Even the most basic test is better than no test

Unit Tests



Unit Tests




```
func testCatImage() {  
    let addCatImage = UIImage(named: UIImage.Name("AddCat"))  
    XCTAssertNotNil(addCatImage)  
}
```

27

NSBox




```
let view = MyView()  
view.backgroundColor = .systemPurple
```

```
let view = MyView()  
view.backgroundColor = .systemPurple
```

```
class MyView: NSView {
    public var backgroundColor = NSColor.clear
    override func draw(_ dirtyRect: NSRect) {
        backgroundColor.setFill()
        bounds.fill()
    }
}
```

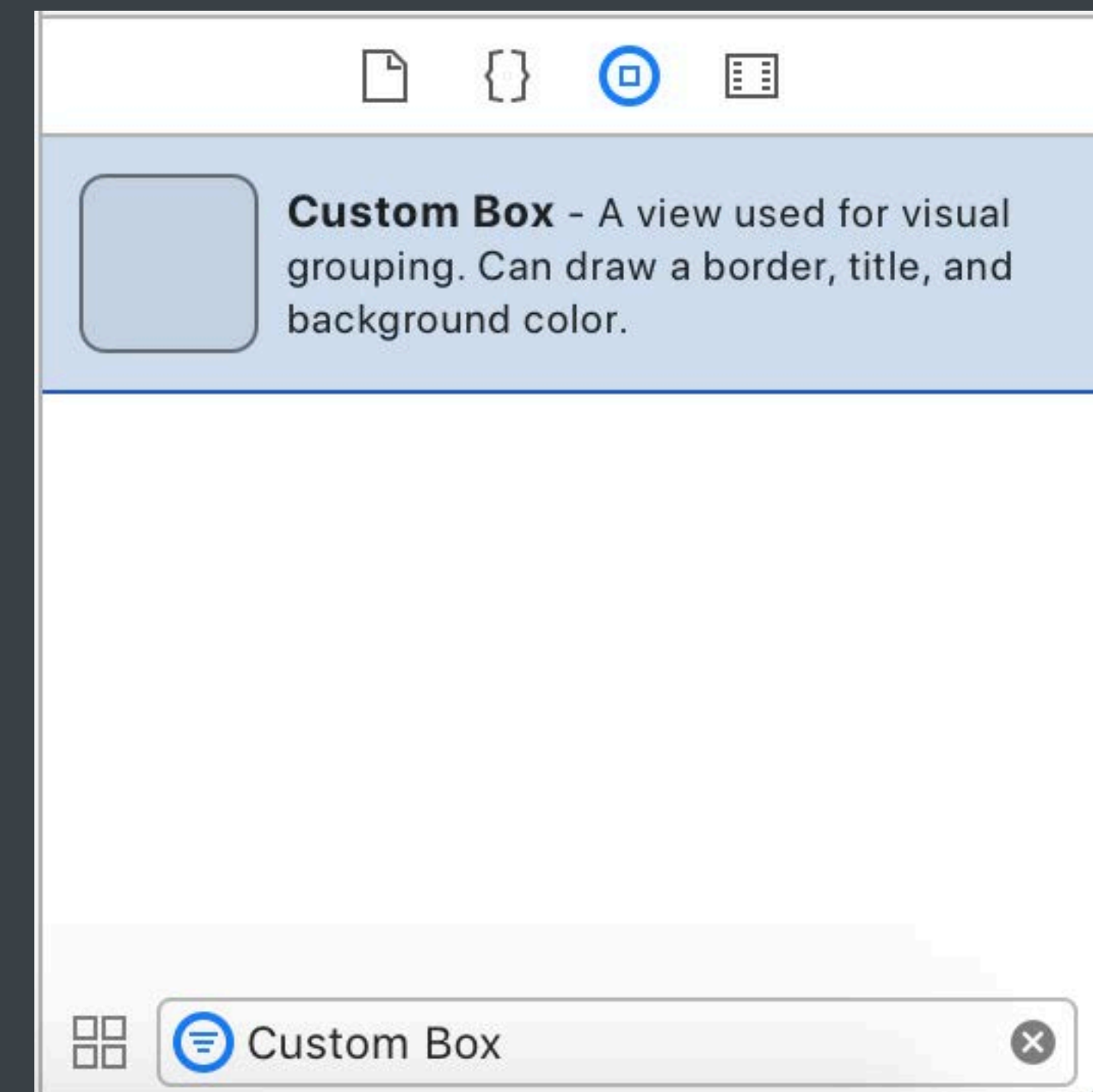
```
class MyView: UIView {
    override init(frame: CGRect) {
        super.init(frame: frame)
        wantsLayer = true
        wantsUpdateLayer = true
    }
    required init?(coder: NSCoder) {
        super.init(coder: coder)
        wantsLayer = true
        wantsUpdateLayer = true
    }

    public var backgroundColor = UIColor.clear
    override func updateLayer() {
        layer?.backgroundColor = backgroundColor.cgColor
    }
}
```

```
let box = NSBox()  
box.boxType = .custom  
box.borderWidth = 0  
box.fillColor = .systemPurple
```

```
let box = NSBox()  
box.boxType = .custom  
box.borderWidth = 0  
box.fillColor = .systemPurple
```

```
let box = NSBox()  
box.boxType = .custom  
box.borderWidth = 0  
box.fillColor = .systemPurple
```



```
open class NSBox : NSView {
    open var borderWidth: CGFloat
    open var cornerRadius: CGFloat
    @NSCopying open var borderColor: NSColor
    @NSCopying open var fillColor: NSColor

    ...

    open var contentView: NSView?
}
}
```



```
open class NSBox : NSView {  
    open var borderWidth: CGFloat  
    open var cornerRadius: CGFloat  
    @NSCopying open var borderColor: NSColor  
    @NSCopying open var fillColor: NSColor
```

...

```
    open var contentView: NSView?
```

```
}
```

```
open class NSBox : NSView {  
    open var borderWidth: CGFloat  
    open var cornerRadius: CGFloat  
    @NSCopying open var borderColor: NSColor  
    @NSCopying open var fillColor: NSColor
```

```
...
```

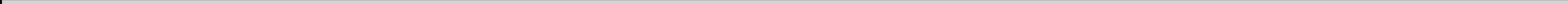
```
open var contentView: NSView?
```

```
}
```

```
open class NSBox : NSView {
    open var borderWidth: CGFloat
    open var cornerRadius: CGFloat
    @NSCopying open var borderColor: NSColor
    @NSCopying open var fillColor: NSColor

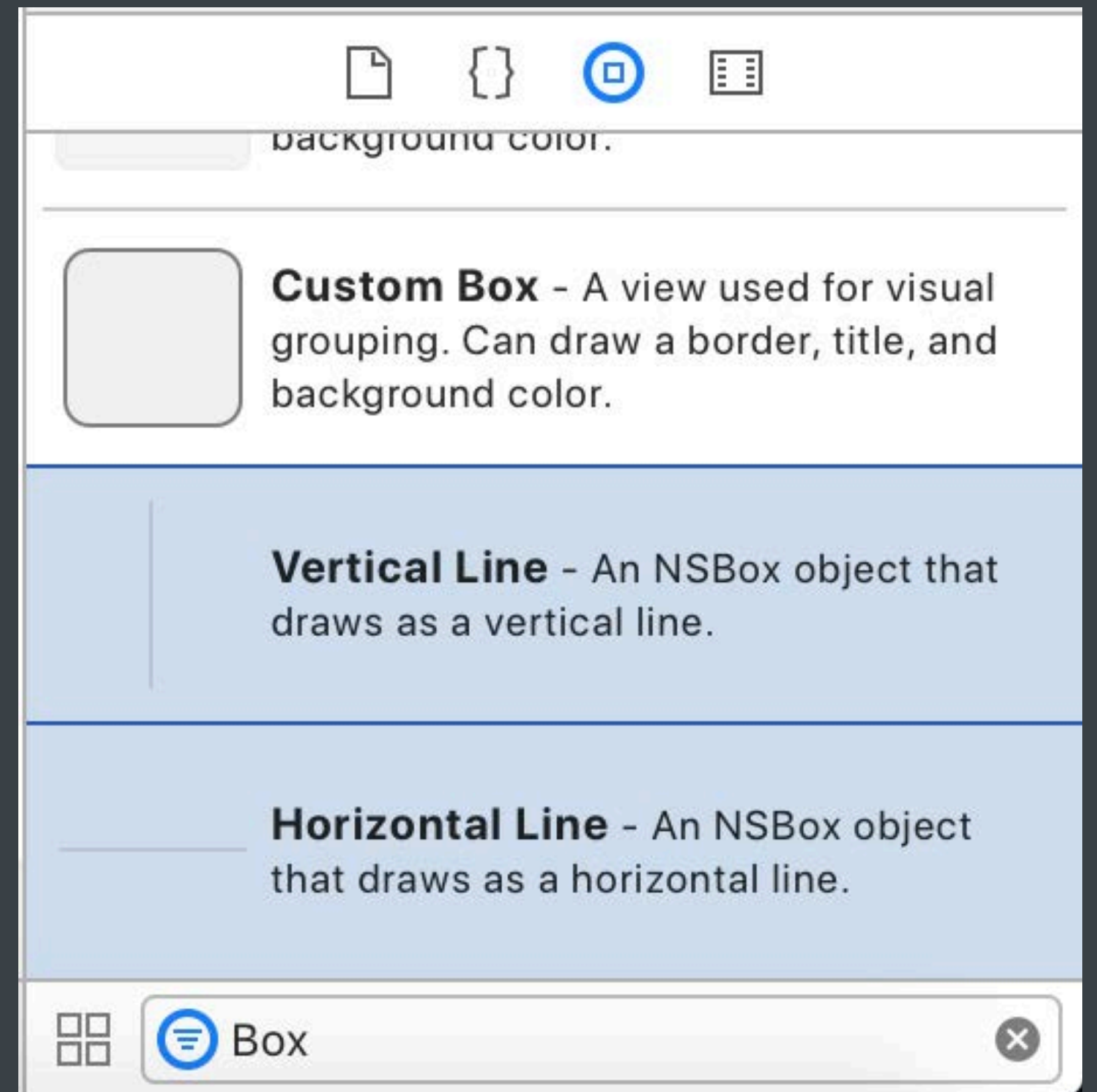
    ...

    open var contentView: NSView?
}
```




```
let box = NSBox()  
box.boxType = .separator
```

```
let box = NSBox()  
box.boxType = .separator
```



8

Restorable State


```
extension UIResponder {  
    open func encodeRestorableState(with coder: NSCoder)  
    open func restoreState(with coder: NSCoder)  
    open func invalidateRestorableState()  
}
```

```
extension UIResponder {  
    open func encodeRestorableState(with coder: NSCoder)  
    open func restoreState(with coder: NSCoder)  
    open func invalidateRestorableState()  
}
```

```
var selectedItem: String { didSet { invalidateRestorableState() } }
var documentIdentifier: String { didSet { invalidateRestorableState() } }
override func encodeRestorableState(with coder: NSCoder) {
    super.encodeRestorableState(with: coder)
    coder.encode(documentIdentifier, forKey: "documentIdentifier")
    coder.encode(selectedItem, forKey: "selectedItem")
}
override func restoreState(with coder: NSCoder) {
    super.restoreState(with: coder)
    if let di = coder.decodeObject(forKey: "documentIdentifier") as? String {
        documentIdentifier = di
    }

    if let si = coder.decodeObject(forKey: "selectedItem") as? String {
        selectedItem = si
    }
}
```

```
var selectedItem: String { didSet { invalidateRestorableState() } }
var documentIdentifier: String { didSet { invalidateRestorableState() } }
override func encodeRestorableState(with coder: NSCoder) {
    super.encodeRestorableState(with: coder)
    coder.encode(documentIdentifier, forKey: "documentIdentifier")
    coder.encode(selectedItem, forKey: "selectedItem")
}
override func restoreState(with coder: NSCoder) {
    super.restoreState(with: coder)
    if let di = coder.decodeObject(forKey: "documentIdentifier") as? String {
        documentIdentifier = di
    }

    if let si = coder.decodeObject(forKey: "selectedItem") as? String {
        selectedItem = si
    }
}
```

```
var selectedItem: String { didSet { invalidateRestorableState() } }
var documentIdentifier: String { didSet { invalidateRestorableState() } }
override func encodeRestorableState(with coder: NSCoder) {
    super.encodeRestorableState(with: coder)
    coder.encode(documentIdentifier, forKey: "documentIdentifier")
    coder.encode(selectedItem, forKey: "selectedItem")
}
override func restoreState(with coder: NSCoder) {
    super.restoreState(with: coder)
    if let di = coder.decodeObject(forKey: "documentIdentifier") as? String {
        documentIdentifier = di
    }

    if let si = coder.decodeObject(forKey: "selectedItem") as? String {
        selectedItem = si
    }
}
```

```
var selectedItem: String { didSet { invalidateRestorableState() } }
var documentIdentifier: String { didSet { invalidateRestorableState() } }
override func encodeRestorableState(with coder: NSCoder) {
    super.encodeRestorableState(with: coder)
    coder.encode(documentIdentifier, forKey: "documentIdentifier")
    coder.encode(selectedItem, forKey: "selectedItem")
}
```

```
override func restoreState(with coder: NSCoder) {
    super.restoreState(with: coder)
    if let di = coder.decodeObject(forKey: "documentIdentifier") as? String {
        documentIdentifier = di
    }

    if let si = coder.decodeObject(forKey: "selectedItem") as? String {
        selectedItem = si
    }
}
```

```
var selectedItem: String { didSet { invalidateRestorableState() } }
var documentIdentifier: String { didSet { invalidateRestorableState() } }
override func encodeRestorableState(with coder: NSCoder) {
    super.encodeRestorableState(with: coder)
    coder.encode(documentIdentifier, forKey: "documentIdentifier")
    coder.encode(selectedItem, forKey: "selectedItem")
}
override func restoreState(with coder: NSCoder) {
    super.restoreState(with: coder)
    if let di = coder.decodeObject(forKey: "documentIdentifier") as? String {
        documentIdentifier = di
    }

    if let si = coder.decodeObject(forKey: "selectedItem") as? String {
        selectedItem = si
    }
}
```

```
extension NSResponder {  
    open class var restorableStateKeyPaths: [String] { get }  
}
```



```
extension NSResponder {  
    open class var restorableStateKeyPaths: [String] { get }  
}
```

```
@objc dynamic var selectedItem: String

@objc dynamic var documentIdentifier: String

override class var restorableStateKeyPaths: [String] {
    return super.restorableStateKeyPaths + [
        #keyPath(ViewController.documentIdentifier),
        #keyPath(ViewController.selectedItem)]
}
```

```
@objc dynamic var selectedItem: String
```

```
@objc dynamic var documentIdentifier: String
```

```
override class var restorableStateKeyPaths: [String] {  
    return super.restorableStateKeyPaths + [  
        #keyPath(ViewController.documentIdentifier),  
        #keyPath(ViewController.selectedItem)]  
    }  
}
```

```
@objc dynamic var selectedItem: String
```

```
@objc dynamic var documentIdentifier: String
```

```
override class var restorableStateKeyPaths: [String] {  
    return super.restorableStateKeyPaths + [  
        #keyPath(ViewController.documentIdentifier),  
        #keyPath(ViewController.selectedItem)]  
    }  
}
```

```
@objc dynamic var selectedItem: String

@objc dynamic var documentIdentifier: String

override class var restorableStateKeyPaths: [String] {
    return super.restorableStateKeyPaths + [
        #keyPath(ViewController.documentIdentifier),
        #keyPath(ViewController.selectedItem)]
}
```

13

Core Data: NSPersistentContainer

What Is a Core Data Stack?

What Is a Core Data Stack?

NSManagedObjectContext

What Is a Core Data Stack?

NSPersistentStoreCoordinator

NSManagedObjectModel

What Is a Core Data Stack?

NSManagedObjectContext

NSPersistentStoreCoordinator

NSManagedObjectModel

```
class DataControllerOld: NSObject {
    var applicationDocumentsDirectory: URL = {
        let urls = FileManager.default.urls(for: .documentDirectory, in: .userDomainMask)
        return urls[urls.count-1]}()
    var managedObjectModel: NSManagedObjectModel = {
        let modelURL = Bundle.main.url(forResource: "AppName", withExtension: "momd")!
        return NSManagedObjectModel(contentsOf: modelURL)!}()
    var persistentStoreCoordinator: NSPersistentStoreCoordinator = {
        let coordinator = NSPersistentStoreCoordinator(managedObjectModel:
self.managedObjectModel)
        let url = self.applicationDocumentsDirectory.appendingPathComponent("AppName.sqlite")
        do {
            try coordinator.addPersistentStore(ofType: NSSQLiteStoreType, configurationName:
nil, at: url, options: nil)
        } catch {
            // Replace this with code to handle the error appropriately.
            fatalError("Unresolved error \(error), \(error.userInfo)")
        }
        return coordinator}()
```

```
let coordinator = NSPersistentStoreCoordinator(managedObjectModel:
self.managedObjectModel)
    let url = self.applicationDocumentsDirectory.appendingPathComponent("AppName.sqlite")
    do {
        try coordinator.addPersistentStore(ofType: NSSQLiteStoreType, configurationName:
nil, at: url, options: nil)
    } catch {
        // Replace this with code to handle the error appropriately.
        fatalError("Unresolved error \(error), \(error.userInfo)")
    }
    return coordinator}()

var managedObjectContext: NSManagedObjectContext = {
    let coordinator = self.persistentStoreCoordinator
    var managedObjectContext =
NSManagedObjectContext(concurrencyType: .mainQueueConcurrencyType)
    managedObjectContext.persistentStoreCoordinator = coordinator
    return managedObjectContext
}()
}
```

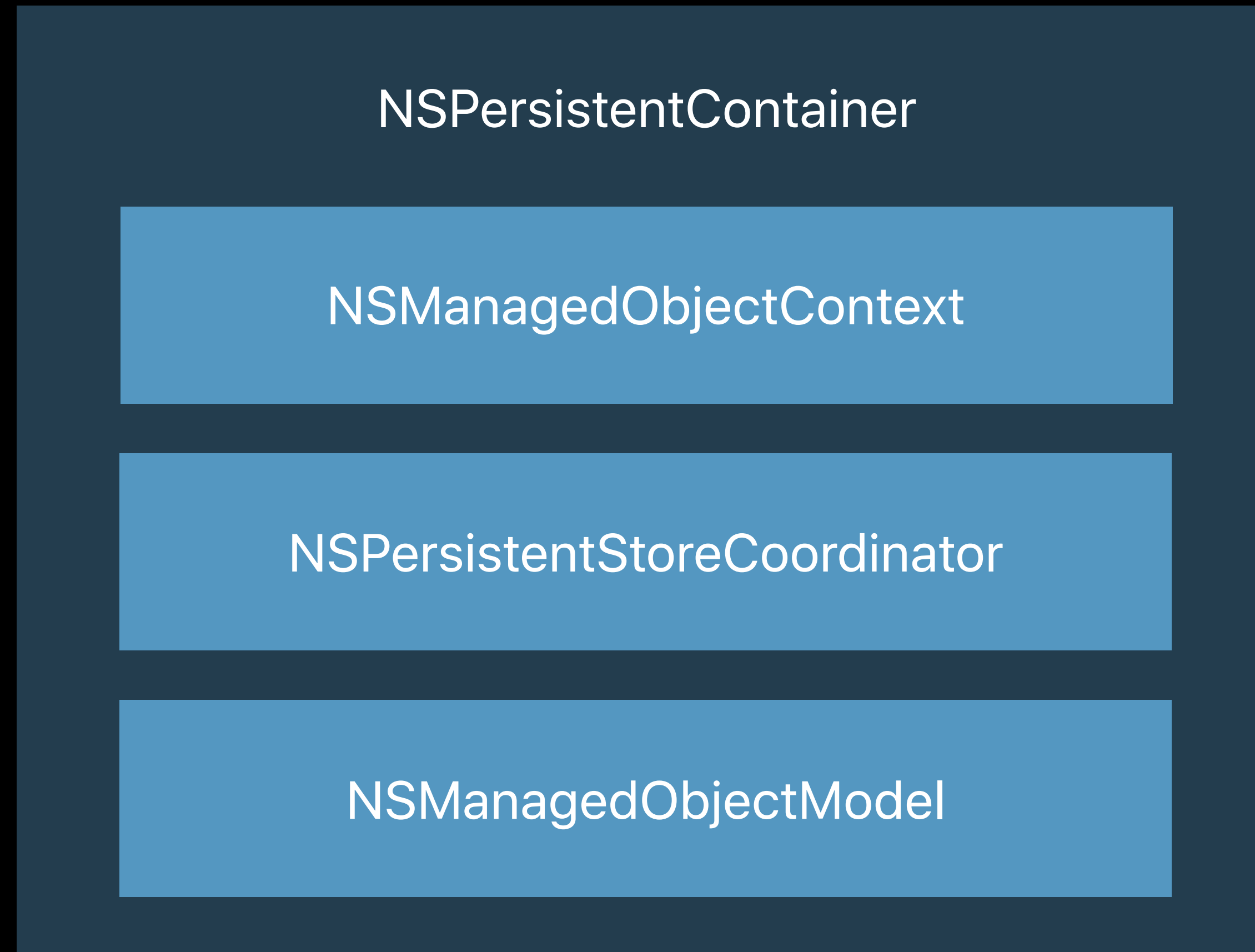
NSPersistentContainer Encapsulates the Stack

NSManagedObjectContext

NSPersistentStoreCoordinator

NSManagedObjectModel

NSPersistentContainer Encapsulates the Stack



```
class DataControllerNew: NSObject {
    var persistentContainer: NSPersistentContainer
    init(completionClosure: @escaping () -> ()) {
        persistentContainer = NSPersistentContainer(name: "DataModel")
        persistentContainer.loadPersistentStores() { (description, error) in
            if let error = error {
                // Handle the error here, depending on what it is
            }
            completionClosure()
        }
    }
}
```

```
class DataControllerNew: NSObject {
    var persistentContainer: NSPersistentContainer
    init(completionClosure: @escaping () -> ()) {
        persistentContainer = NSPersistentContainer(name: "DataModel")
        persistentContainer.loadPersistentStores() { (description, error) in
            if let error = error {
                // Handle the error here, depending on what it is
            }
            completionClosure()
        }
    }
}
```


21

Core Data: Arrays



Cat Wrangler

Core Data

Arrays

ENTITIES

- E** Cat

FETCH REQUESTS

CONFIGURATIONS

- C** Default

▼ **Attributes**

Attribute	Type ^	
S name	String	◇
⏻ photo	Binary Data	◇

+ -

▼ **Relationships**

Relationship ^	Destination	Inverse

+ -







Core Data

Arrays

The screenshot shows the Xcode Core Data editor interface. On the left, the 'ENTITIES' pane lists 'Cat' under the 'E' icon. Below it are sections for 'FETCH REQUESTS' and 'CONFIGURATIONS' with a 'Default' configuration. The main area is titled 'Attributes' and contains a table with columns 'Attribute' and 'Type'. The 'behaviors' attribute is selected, and a dropdown menu is open, showing 'Undefined' as the selected type. Other types include Integer 16, Integer 32, Integer 64, Decimal, Double, Float, String, Boolean, Date, Binary Data, UUID, URI, and Transformable. Below the attributes is a section for 'Relationships' with an empty table and a '+' button.

Attribute	Type ^
? behaviors	✓ Undefined
S name	Integer 16
photo	Integer 32
	Integer 64
	Decimal
	Double
	Float
	String
	Boolean
	Date
	Binary Data
	UUID
	URI
	Transformable

Relationship ^

Inverse

Core Data

Arrays

ENTITIES

- E** Cat

FETCH REQUESTS

CONFIGURATIONS

- C** Default

▼ **Attributes**

Attribute	Type ^	
T behaviors	Transformable	◇
S name	String	◇
⏻ photo	Binary Data	◇

+ -

▼ **Relationships**

Relationship ^	Destination	Inverse

+ -

Core Data

Arrays

The screenshot shows the Core Data model editor interface. On the left, the 'ENTITIES' section is expanded to show 'Cat'. Below it are 'FETCH REQUESTS' and 'CONFIGURATIONS' with 'Default' selected. The main area is titled 'Attributes' and contains a table with columns 'Attribute' and 'Type'. The 'Type' column is expanded to show 'Transformable', 'String', and 'Binary Data'. A red circle with a black 'X' is overlaid on the 'Transformable' row, indicating an error. Below the 'Attributes' section, the 'Relationships' section is partially visible, showing columns for 'Relationship', 'Destination', and 'Inverse'. At the bottom of the 'Relationships' section, there are '+' and '-' icons.

Attribute	Type ^
	Transformable
	String
	Binary Data

Relationship ^	Destination	Inverse

+ -

Core Data

Arrays

Core Data

Arrays

Added overhead for serializing and deserializing the array and its contents

Core Data

Arrays

Added overhead for serializing and deserializing the array and its contents

Fetch requests are much slower

Core Data Relationships

ENTITIES

- E** Behavior
- E** Cat

FETCH REQUESTS

CONFIGURATIONS

- C** Default

Attributes

Attribute	Type ^
S name	String
P photo	Binary Data

+ -

Relationships

Relationship ^	Destination	Inverse
M behaviors	Behavior	cats

+ -

Relationship

Name: behaviors

Properties: Transient Optional

Destination: Behavior

Inverse: cats

Delete Rule: Nullify

Type: To Many

Arrangement: Ordered

Count: Unbounded Minimum

Unbounded Maximum

Advanced: Index in Spotlight

Deprecated

Spotlight: Store in External Record File

Core Data

To many relationships

The screenshot displays the Core Data configuration interface. On the left, a sidebar lists 'ENTITIES' (Behavior, Cat), 'FETCH REQUESTS', and 'CONFIGURATIONS' (Default). The main area is divided into 'Attributes' and 'Relationships' sections. The 'Attributes' section shows 'name' (String) and 'photo' (Binary Data). The 'Relationships' section shows a relationship named 'behaviors' with destination 'Behavior' and inverse 'cats'. A dropdown menu is open over the 'behaviors' relationship, showing 'Type' set to 'To Many'. The right-hand pane shows the 'Relationship' configuration for 'behaviors', including 'Name', 'Properties' (Optional checked), 'Destination' (Behavior), 'Inverse' (cats), 'Delete Rule' (Nullify), 'Type' (To Many), 'Arrangement' (Ordered), 'Count' (Unbounded), and 'Advanced' (Index in Spotlight). A 'Deprecated' section at the bottom includes 'Spotlight' and 'Store in External Record File' options.

Attribute	Type
name	String
photo	Binary Data

Relationship	Destination	Inverse
behaviors	Behavior	cats

Relationship Configuration for 'behaviors':

- Name: behaviors
- Properties: Transient, Optional
- Destination: Behavior
- Inverse: cats
- Delete Rule: Nullify
- Type: To Many
- Arrangement: Ordered
- Count: Unbounded (with Minimum and Maximum options)
- Advanced: Index in Spotlight
- Deprecated: Store in External Record File

Core Data

To many relationships

The screenshot displays the Core Data configuration interface. On the left, a sidebar lists 'ENTITIES' (Behavior, Cat), 'FETCH REQUESTS', and 'CONFIGURATIONS' (Default). The main area is divided into 'Attributes' and 'Relationships' sections. The 'Attributes' section shows 'name' (String) and 'photo' (Binary Data). The 'Relationships' section shows a relationship named 'behaviors' between 'Behavior' and 'cats' entities, with a cardinality of 'To Many'. A tooltip is visible over the 'Type' dropdown, which is set to 'To Many'. The right-hand pane shows the 'Relationship' configuration for 'behaviors', including 'Name', 'Properties' (Optional checked), 'Destination' (Behavior), 'Inverse' (cats), 'Delete Rule' (Nullify), 'Arrangement' (Ordered), 'Count' (Unbounded), and 'Advanced' options (Index in Spotlight).

Attribute	Type
name	String
photo	Binary Data

Relationship	Destination	Inverse
behaviors	Behavior	cats

Relationship Configuration for 'behaviors':

- Name: behaviors
- Properties: Transient, Optional
- Destination: Behavior
- Inverse: cats
- Delete Rule: Nullify
- Arrangement: Ordered
- Count: Unbounded (Minimum , Maximum)
- Advanced: Index in Spotlight
- Deprecated: Store in External Record File

Core Data

To many relationships

The screenshot displays the Core Data configuration interface. On the left, a sidebar lists 'ENTITIES' (Behavior, Cat), 'FETCH REQUESTS', and 'CONFIGURATIONS' (Default). The main area is divided into 'Attributes' and 'Relationships' sections. The 'Attributes' section shows 'name' (String) and 'photo' (Binary Data). The 'Relationships' section shows a relationship named 'behaviors' between 'Behavior' and 'cats' entities, with a cardinality of 'M' to '1'. A white callout box highlights the 'Arrangement' dropdown menu, which is set to 'Ordered'. The 'Relationship' configuration panel on the right shows the name 'behaviors', properties 'Optional' (checked) and 'Transient' (unchecked), destination 'Behavior', inverse 'cats', and delete rule 'Nullify'. Other options include 'Unbounded' (checked) and 'Maximum' (unchecked) for cardinality, and 'Index in Spotlight' (unchecked) under 'Advanced'. A 'Deprecated' section includes 'Store in External Record File' (unchecked).

Attribute	Type
name	String
photo	Binary Data

Relationship	Destination	Inverse
behaviors	Behavior	cats

Arrangement Ordered

Core Data

To many relationships

The screenshot displays a Core Data model editor interface. On the left, a sidebar lists the model's components: ENTITIES (Behavior, Cat), FETCH REQUESTS, and CONFIGURATIONS (Default). The 'Behavior' entity is selected. The main area shows the 'Attributes' and 'Relationships' for the selected entity.

ENTITIES

- E** Behavior
- E** Cat

FETCH REQUESTS

CONFIGURATIONS

- C** Default

Attributes

Attribute	Type ^	
N duration	Float	⬇
S name	String	⬇

+ -

Relationships

Relationship ^	Destination	Inverse
M cats	Cat	⬇ behavic ⬇

+ -

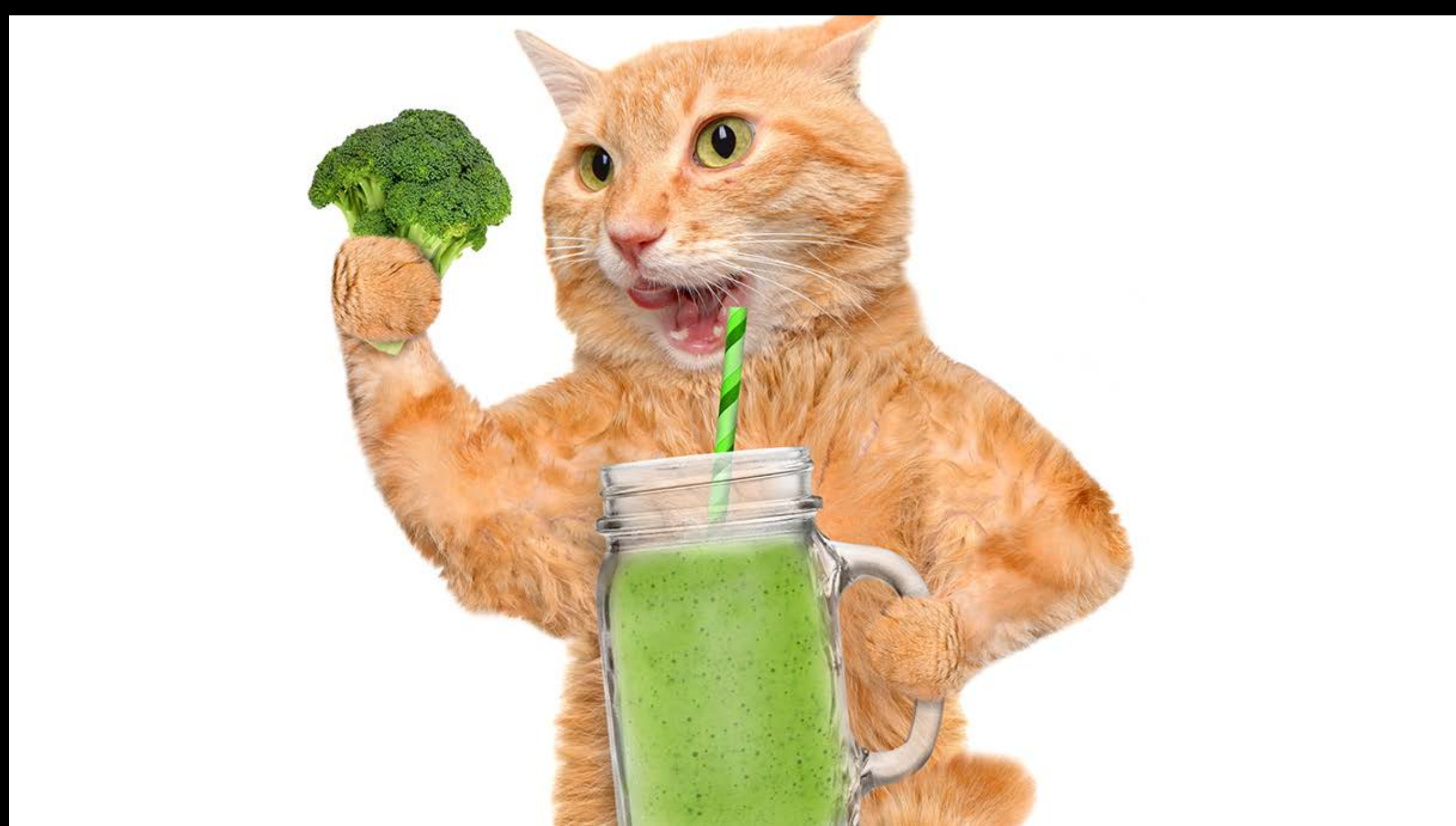
34

Core Data: Migration



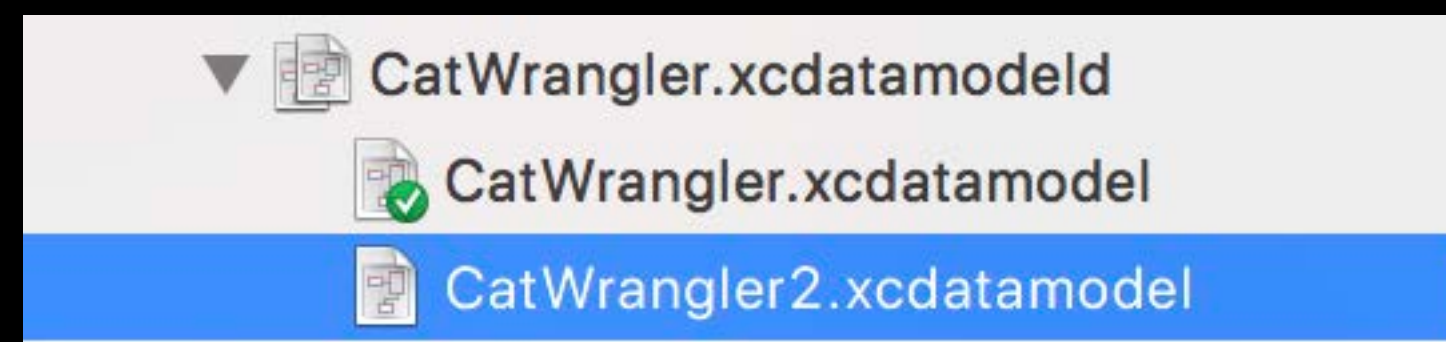






Core Data

Migration



Core Data

Migration

ENTITIES

- E** Behavior
- E** Cat
- E** Food

FETCH REQUESTS

CONFIGURATIONS

- C** Default

▼ Attributes

Attribute ^	Type	
S name	String	◇
N price	Float	◇
U source	URI	◇

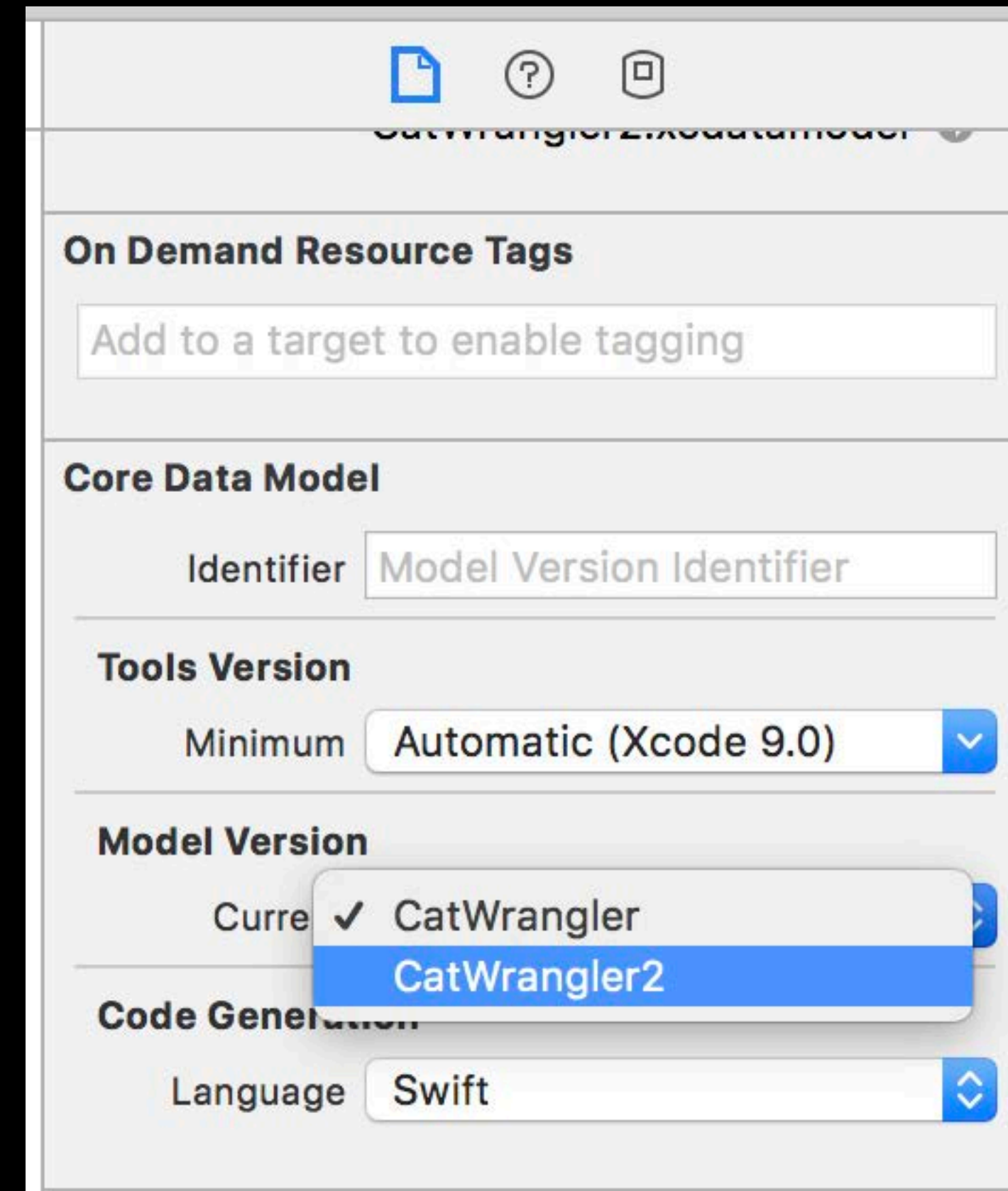
+ -

▼ Relationships

Relationship ^	Destination	Inverse
M cats	Cat	◇ foods ◇

+ -

Core Data Migration



Core Data

Migration

Lightweight migration automatic with `NSPersistentContainer`

Core Data

Migration

Lightweight migration automatic with `NSPersistentContainer`

Otherwise use `shouldInferMappingModelAutomatically` and `NSMigratePersistentStoresAutomaticallyOption`

55

Core Data: Error Handling

Core Data

Error handling

Pay attention to errors

Most important when adding a persistent store

App won't work if persistent store can't be added

Handle these errors even if you ignore all others

```
class DataControllerNew: NSObject {
    var persistentContainer: NSPersistentContainer
    init(completionClosure: @escaping () -> ()) {
        persistentContainer = NSPersistentContainer(name: "DataModel")
        persistentContainer.loadPersistentStores() { (description, error) in
            // Some common error conditions are:
            // - not enough storage space
            // - unable to access store due to permissions issues or
            //     data protection
            // - trying to add a store created with an older model
            //     without a proper migration strategy
            if let error = error {
                // Handle the error here, depending on what it is
            }
            completionClosure()
        }
    }
}
```

404

NSError

NSError

Cocoa APIs return NSError objects that you can present to the user


```
extension NSResponder {  
    open func presentError(_: Error) -> Bool  
}
```

```
extension NSResponder {  
    open func presentError(_: Error) -> Bool  
}
```

```
extension NSResponder {  
    open func presentError(_: Error) -> Bool  
}
```



**The file "Cat List" couldn't be opened
because it isn't in the correct format.**

The file might be corrupted, truncated, or in an
unexpected format.

OK

```
throw NSError(domain: NSCocoaErrorDomain, code: NSFileReadCorruptFileError,  
              userInfo: [NSURLErrorKey: fileURL ])
```



**The file "Cat List" couldn't be opened
because it isn't in the correct format.**

The file might be corrupted, truncated, or in an
unexpected format.

OK

```
throw NSError(domain: NSCocoaErrorDomain, code: NSFileReadCorruptFileError,  
              userInfo: [NSURLErrorKey: fileURL ])
```



**The file "Cat List" couldn't be opened
because it isn't in the correct format.**

The file might be corrupted, truncated, or in an
unexpected format.

OK

```
public var NSFileNoSuchFileError: Int { get } // Attempt to do a file system operation on a
                                                non-existent file

public var NSFileLockingError: Int { get } // Couldn't get a lock on file

public var NSFileReadUnknownError: Int { get } // Read error (reason unknown)

public var NSFileReadNoPermissionError: Int { get } // Read error (permission problem)

public var NSFileReadInvalidFileNameError: Int { get } // Read error (invalid file name)

public var NSFileReadCorruptFileError: Int { get } // Read error (file corrupt, bad format,
                                                etc)

public var NSFileReadNoSuchFileError: Int { get } // Read error (no such file)

public var NSFileReadInapplicableStringEncodingError: Int { get } // Read error (string
                                                encoding not applicable) also NSStringEncodingErrorKey

public var NSFileReadUnsupportedSchemeError: Int { get } // Read error (unsupported URL
                                                scheme)

public var NSFileReadTooLargeError: Int { get } // Read error (file too large)

public var NSFileReadUnknownStringEncodingError: Int { get } // Read error (string encoding of
                                                file contents could not be determined)

public var NSFileWriteUnknownError: Int { get } // Write error (reason unknown)
```



```
// Other errors
public var NSKeyValueValidationError: Int { get } // KVC validation error
public var NSFormattingError: Int { get } // Formatting error
public var NSUserCancelledError: Int { get } // User cancelled operation (this one often
doesn't deserve a panel and might be a good one to special case)
public var NSFeatureUnsupportedError: Int { get } // Feature unsupported error

// Executable loading errors
public var NSExecutableNotLoadableError: Int { get } // Executable is of a type that is not
loadable in the current process
public var NSExecutableArchitectureMismatchError: Int { get } // Executable does not provide
an architecture compatible with the current process
public var NSExecutableRuntimeMismatchError: Int { get } // Executable has Objective C runtime
information incompatible with the current process
public var NSExecutableLoadError: Int { get } // Executable cannot be loaded for some other
reason, such as a problem with a library it depends on
public var NSExecutableLinkError: Int { get } // Executable fails due to linking issues
```

```
throw NSError(domain: NSCocoaErrorDomain, code: NSFileReadCorruptFileError,  
    userInfo: [  
        NSLocalizedRecoverySuggestionErrorKey:  
            NSLocalizedString("Re-download the file and try again.",  
                comment: "Recovery suggestion when importing file via download."),  
        NSURLErrorKey: fileURL  
    ]  
])
```



**The file "Cat List" couldn't be opened
because it isn't in the correct format.**

Re-download the file and try again.

OK

```
throw NSError(domain: NSCocoaErrorDomain, code: NSFileReadCorruptFileError,  
    userInfo: [  
        NSLocalizedRecoverySuggestionErrorKey:  
            NSLocalizedString("Re-download the file and try again.",  
                comment: "Recovery suggestion when importing file via download."),  
        NSURLErrorKey: fileURL  
    ]  
])
```



**The file "Cat List" couldn't be opened
because it isn't in the correct format.**

Re-download the file and try again.

OK

```
throw CocoaError.error(.fileReadCorruptFile,  
    userInfo: [  
        NSLocalizedRecoverySuggestionErrorKey:  
            NSLocalizedString("Re-download the file and try again.",  
                comment: "Recovery suggestion when importing file via download."),  
        NSURLErrorKey: fileURL  
    ]  
)
```



**The file "Cat List" couldn't be opened
because it isn't in the correct format.**

Re-download the file and try again.

OK


```
throw CocoaError.error(.fileReadCorruptFile,  
    userInfo: [  
        NSLocalizedRecoverySuggestionErrorKey:  
            NSLocalizedString("Re-download the file and try again.",  
                comment: "Recovery suggestion when importing file via download."),  
        NSErrorKey: fileURL  
    ]  
])
```



**The file "Cat List" couldn't be opened
because it isn't in the correct format.**

Re-download the file and try again.

OK

NSError

NSError



Cat is sitting in the corner meowing at the wall.

Confuse the cat and try again.

OK


```
struct CatError {
    public static var domain = "CatErrorDomain"
    public enum code : Int {
        case notFound
        case busyEating
        case meowingAtWall
        case attackingDust
    }
}

throw NSError(domain: CatError.domain, code: CatError.code.meowingAtWall.rawValue, userInfo: [
    NSLocalizedFailureErrorKey:
        NSLocalizedString("Cat is sitting in the corner meowing at the wall.",
            comment:"Failure message when cat is meowing at wall"),
    NSLocalizedRecoverySuggestionErrorKey:
        NSLocalizedString("Confuse the cat and try again later",
            comment:"Recovery suggestion when cat is meowing at wall"),
])
```

```
struct CatError {
    public static var domain = "CatErrorDomain"
    public enum code : Int {
        case notFound
        case busyEating
        case meowingAtWall
        case attackingDust
    }
}

throw NSError(domain: CatError.domain, code: CatError.code.meowingAtWall.rawValue, userInfo: [
    NSLocalizedFailureErrorKey:
        NSLocalizedString("Cat is sitting in the corner meowing at the wall.",
            comment:"Failure message when cat is meowing at wall"),
    NSLocalizedRecoverySuggestionErrorKey:
        NSLocalizedString("Confuse the cat and try again later",
            comment:"Recovery suggestion when cat is meowing at wall"),
])
```

```
struct CatError {
    public static var domain = "CatErrorDomain"
    public enum code : Int {
        case notFound
        case busyEating
        case meowingAtWall
        case attackingDust
    }
}

throw NSError(domain: CatError.domain, code: CatError.code.meowingAtWall.rawValue, userInfo: [
    NSLocalizedFailureErrorKey:
        NSLocalizedString("Cat is sitting in the corner meowing at the wall.",
            comment:"Failure message when cat is meowing at wall"),
    NSLocalizedRecoverySuggestionErrorKey:
        NSLocalizedString("Confuse the cat and try again later",
            comment:"Recovery suggestion when cat is meowing at wall"),
])
```

```
struct CatError {
    public static var domain = "CatErrorDomain"
    public enum code : Int {
        case notFound
        case busyEating
        case meowingAtWall
        case attackingDust
    }
}
```

```
throw NSError(domain: CatError.domain, code: CatError.code.meowingAtWall.rawValue, userInfo: [
    NSLocalizedFailureErrorKey:
        NSLocalizedString("Cat is sitting in the corner meowing at the wall.",
            comment:"Failure message when cat is meowing at wall"),
    NSLocalizedRecoverySuggestionErrorKey:
        NSLocalizedString("Confuse the cat and try again later",
            comment:"Recovery suggestion when cat is meowing at wall"),
])
```

```
struct CatError {
    public static var domain = "CatErrorDomain"
    public enum code : Int {
        case notFound
        case busyEating
        case meowingAtWall
        case attackingDust
    }
}

throw NSError(domain: CatError.domain, code: CatError.code.meowingAtWall.rawValue, userInfo: [
    NSLocalizedFailureErrorKey:
        NSLocalizedString("Cat is sitting in the corner meowing at the wall.",
            comment:"Failure message when cat is meowing at wall"),
    NSLocalizedRecoverySuggestionErrorKey:
        NSLocalizedString("Confuse the cat and try again later",
            comment:"Recovery suggestion when cat is meowing at wall"),
])
```

```
NSError.setUserInfoValueProvider(forDomain: CatError.domain) { (error: Error, key) -> Any? in
    let errorCode = CatError.code(rawValue:(error as NSError).code)!
    switch (key) {
    case NSLocalizedFailureErrorKey:
        switch (errorCode) {
        case .notFound:
            return NSLocalizedString("Cat not found.",
                                     comment: "Failure message when the cat is not found.")
        case .meowingAtWall:
            return NSLocalizedString("Cat is sitting in the corner meowing at the wall.",
                                     comment:"Failure message when cat is meowing at wall")
        ...
        ...
    }
})
```

```
throw NSError(domain: CatError.domain, code: CatError.meowingAtWall.rawValue)
```

```
NSError.setUserInfoValueProvider(forDomain: CatError.domain) { (error: Error, key) -> Any? in
    let errorCode = CatError.code(rawValue:(error as NSError).code)!
    switch (key) {
    case NSLocalizedFailureErrorKey:
        switch (errorCode) {
        case .notFound:
            return NSLocalizedString("Cat not found.",
                                     comment: "Failure message when the cat is not found.")
        case .meowingAtWall:
            return NSLocalizedString("Cat is sitting in the corner meowing at the wall.",
                                     comment:"Failure message when cat is meowing at wall")
        ...
        ...
    }
})
```

```
throw NSError(domain: CatError.domain, code: CatError.meowingAtWall.rawValue)
```



```
NSError.setUserInfoValueProvider(forDomain: CatError.domain) { (error: Error, key) -> Any? in
    let errorCode = CatError.code(rawValue:(error as NSError).code)!
    switch (key) {
    case NSLocalizedFailureErrorKey:
        switch (errorCode) {
        case .notFound:
            return NSLocalizedString("Cat not found.",
                                    comment: "Failure message when the cat is not found.")
        case .meowingAtWall:
            return NSLocalizedString("Cat is sitting in the corner meowing at the wall.",
                                    comment:"Failure message when cat is meowing at wall")
        ...
        ...
    }
})
```

```
throw NSError(domain: CatError.domain, code: CatError.meowingAtWall.rawValue)
```

```
NSError.setUserInfoValueProvider(forDomain: CatError.domain) { (error: Error, key) -> Any? in
    let errorCode = CatError.code(rawValue:(error as NSError).code)!
    switch (key) {
    case NSLocalizedFailureErrorKey:
        switch (errorCode) {
        case .notFound:
            return NSLocalizedString("Cat not found.",
                                     comment: "Failure message when the cat is not found.")
        case .meowingAtWall:
            return NSLocalizedString("Cat is sitting in the corner meowing at the wall.",
                                     comment:"Failure message when cat is meowing at wall")
        ...
        ...
    }
})
```

```
throw NSError(domain: CatError.domain, code: CatError.meowingAtWall.rawValue)
```

```
NSError.setUserInfoValueProvider(forDomain: CatError.domain) { (error: Error, key) -> Any? in
    let errorCode = CatError.code(rawValue:(error as NSError).code)!
    switch (key) {
    case NSLocalizedFailureErrorKey:
        switch (errorCode) {
        case .notFound:
            return NSLocalizedString("Cat not found.",
                                     comment: "Failure message when the cat is not found.")
        case .meowingAtWall:
            return NSLocalizedString("Cat is sitting in the corner meowing at the wall.",
                                     comment:"Failure message when cat is meowing at wall")
        ...
        ...
    }
})
```

```
throw NSError(domain: CatError.domain, code: CatError.meowingAtWall.rawValue)
```

```
open class NSError : NSObject, NSCopying, NSSecureCoding {  
  
    /* Return titles of buttons that are appropriate for displaying in an alert. These should  
    match the string provided as a part of localizedRecoverySuggestion. The first string would be  
    the title of the right-most and default button, the second one next to it, and so on. ...  
    */  
    open var localizedRecoveryOptions: [String]? { get }  
  
    /* Return an object that conforms to the NSErrorRecoveryAttempting informal protocol. The  
    recovery attempter must be an object that can correctly interpret an index into the array  
    returned by localizedRecoveryOptions. ...  
    */  
    open var recoveryAttempter: Any? { get }  
  
}
```

```
open class NSError : NSObject, NSCopying, NSSecureCoding {
```

```
    /* Return titles of buttons that are appropriate for displaying in an alert. These should
    match the string provided as a part of localizedRecoverySuggestion. The first string would be
    the title of the right-most and default button, the second one next to it, and so on. ...
    */
```

```
    open var localizedRecoveryOptions: [String]? { get }
```

```
    /* Return an object that conforms to the NSErrorRecoveryAttempting informal protocol. The
    recovery attempter must be an object that can correctly interpret an index into the array
    returned by localizedRecoveryOptions. ...
    */
```

```
    open var recoveryAttempter: Any? { get }
```

```
}
```

```
public let NSLocalizedRecoveryOptionsErrorKey: String // NSArray of NSStrings corresponding to  
button titles.
```

```
public let NSRecoveryAttempterErrorKey: String // Instance of a subclass of NSObject that  
conforms to the NSErrorRecoveryAttempting informal protocol
```

6

Shared Key Sets


```
extension NSDictionary {  
    open class func sharedKeySet(forKeys: [NSCopying]) -> Any  
}
```

```
extension NSMutableDictionary {  
    public init(sharedKeySet: Any)  
}
```

```
extension NSDictionary {  
    open class func sharedKeySet(forKeys: [NSCopying]) -> Any  
}
```

```
extension NSMutableDictionary {  
    public init(sharedKeySet: Any)  
}
```

```
extension NSDictionary {  
    open class func sharedKeySet(forKeys: [NSCopying]) -> Any  
}
```

```
extension NSMutableDictionary {  
    public init(sharedKeySet: Any)  
}
```

```
extension NSDictionary {  
    open class func sharedKeySet(forKeys: [NSCopying]) -> Any  
}
```

```
extension NSMutableDictionary {  
    public init(sharedKeySet: Any)  
}
```

```
let catKeySet = NSDictionary.sharedKeySet(forKeys: [
    NSString(string:"Name"),
    NSString(string:"Photo"),
    NSString(string:"Status")])

let cat = NSMutableDictionary(sharedKeySet: catKeySet)
cat["Name"] = "Pixel"
cat["Status"] = "Meowing at wall"
cat["FavoriteCorner"] = "North-West" // Not as efficient, but works!
```

```
let catKeySet = NSDictionary.sharedKeySet(forKeys: [  
    NSString(string:"Name"),  
    NSString(string:"Photo"),  
    NSString(string:"Status")])
```

```
let cat = NSMutableDictionary(sharedKeySet: catKeySet)  
cat["Name"] = "Pixel"  
cat["Status"] = "Meowing at wall"  
cat["FavoriteCorner"] = "North-West" // Not as efficient, but works!
```

```
let catKeySet = NSDictionary.sharedKeySet(forKeys: [  
    NSString(string:"Name"),  
    NSString(string:"Photo"),  
    NSString(string:"Status")])
```

```
let cat = NSMutableDictionary(sharedKeySet: catKeySet)  
cat["Name"] = "Pixel"  
cat["Status"] = "Meowing at wall"  
cat["FavoriteCorner"] = "North-West" // Not as efficient, but works!
```



```
let catKeySet = NSDictionary.sharedKeySet(forKeys: [  
    NSString(string:"Name"),  
    NSString(string:"Photo"),  
    NSString(string:"Status")])
```

```
let cat = NSMutableDictionary(sharedKeySet: catKeySet)
```

```
cat["Name"] = "Pixel"
```

```
cat["Status"] = "Meowing at wall"
```

```
cat["FavoriteCorner"] = "North-West" // Not as efficient, but works!
```

```
let catKeySet = NSDictionary.sharedKeySet(forKeys: [
    NSString(string:"Name"),
    NSString(string:"Photo"),
    NSString(string:"Status")])

let cat = NSMutableDictionary(sharedKeySet: catKeySet)
cat["Name"] = "Pixel"
cat["Status"] = "Meowing at wall"
cat["FavoriteCorner"] = "North-West" // Not as efficient, but works!
```

```
let catKeySet = NSDictionary.sharedKeySet(forKeys: [
    NSString(string:"Name"),
    NSString(string:"Photo"),
    NSString(string:"Status")])

let cat = NSMutableDictionary(sharedKeySet: catKeySet)
cat["Name"] = "Pixel"
cat["Status"] = "Meowing at wall"
cat["FavoriteCorner"] = "North-West" // Not as efficient, but works!
```

Oxa

Accessibility

Accessibility

VoiceOver

VoiceOver information is easy to add in Interface Builder

VoiceOver comes built-in on Apple products

Accessibility

VoiceOver



root tip root tip

Accessibility Identity

Description

Help

Identifier

Accessibility

Screen resolution

Test at 1024 x 640



Cat Wrangler



CatWrangler

File

Edit

Format

View

Window

Help

5:55 PM

Rachel Goldeen

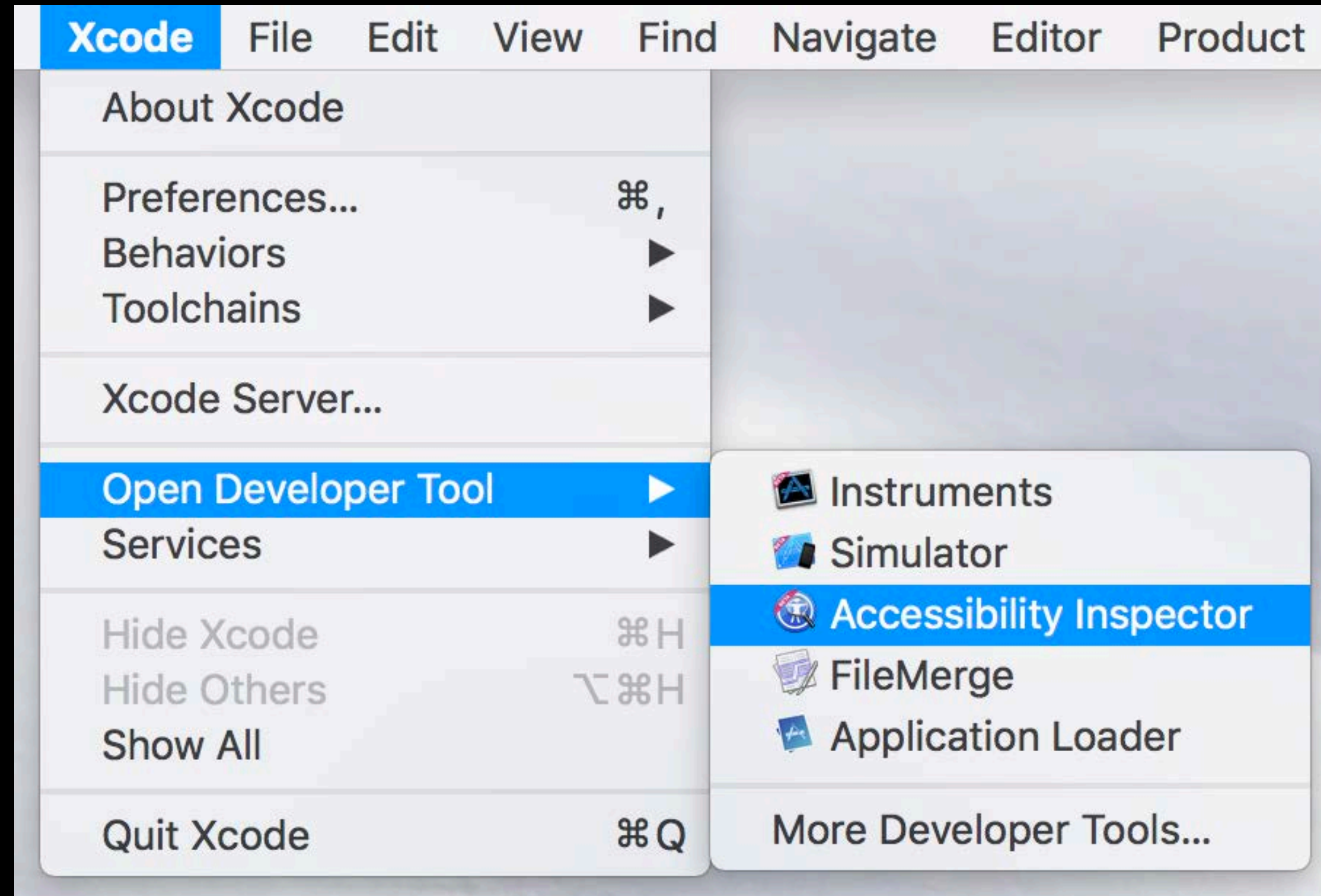


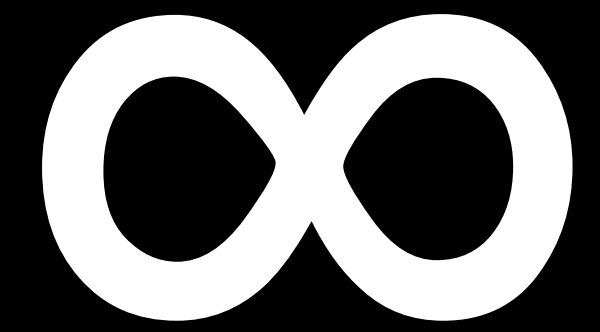
Cat Wrangler

MacBook Pro

Accessibility

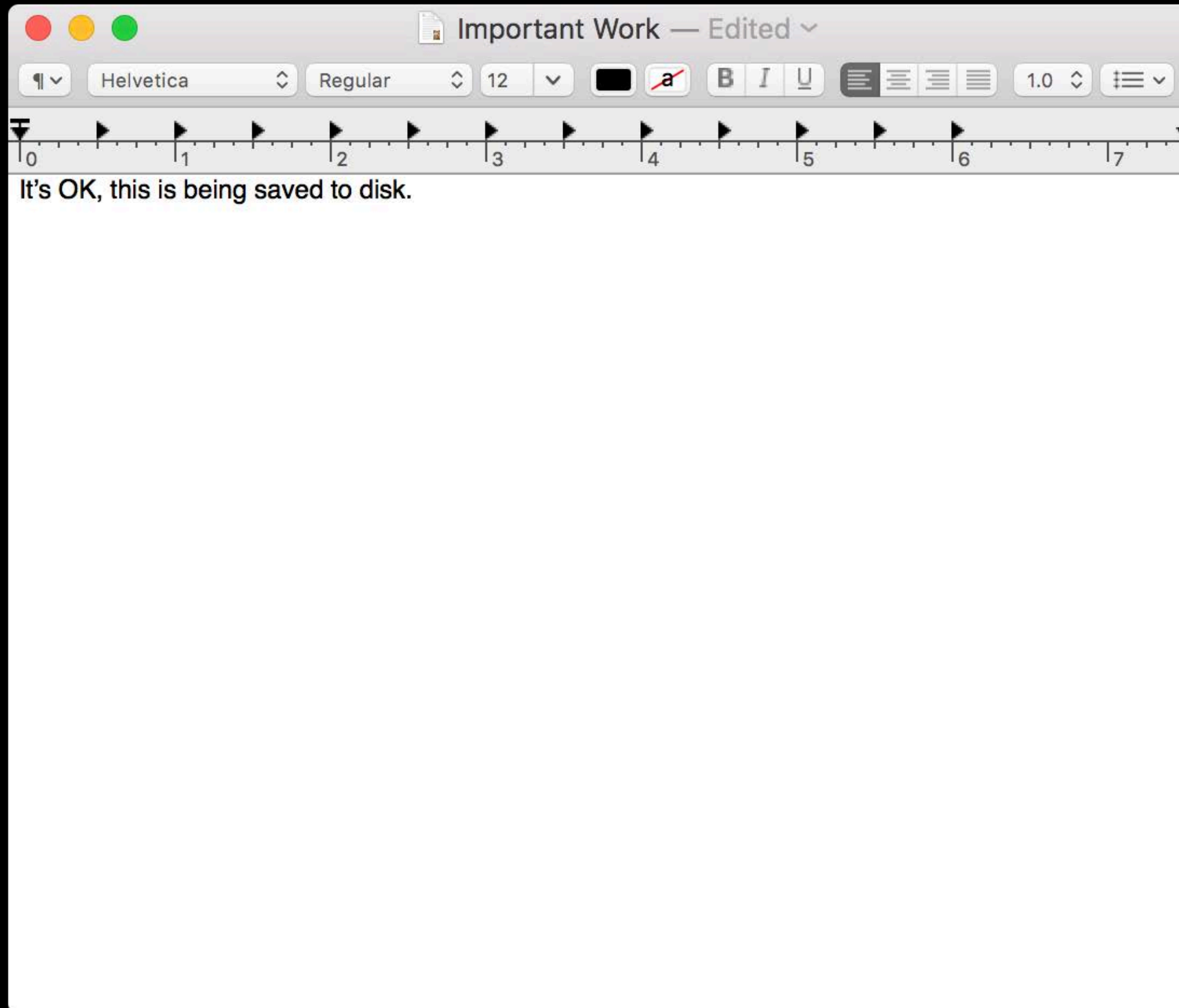
Accessibility Inspector in Xcode



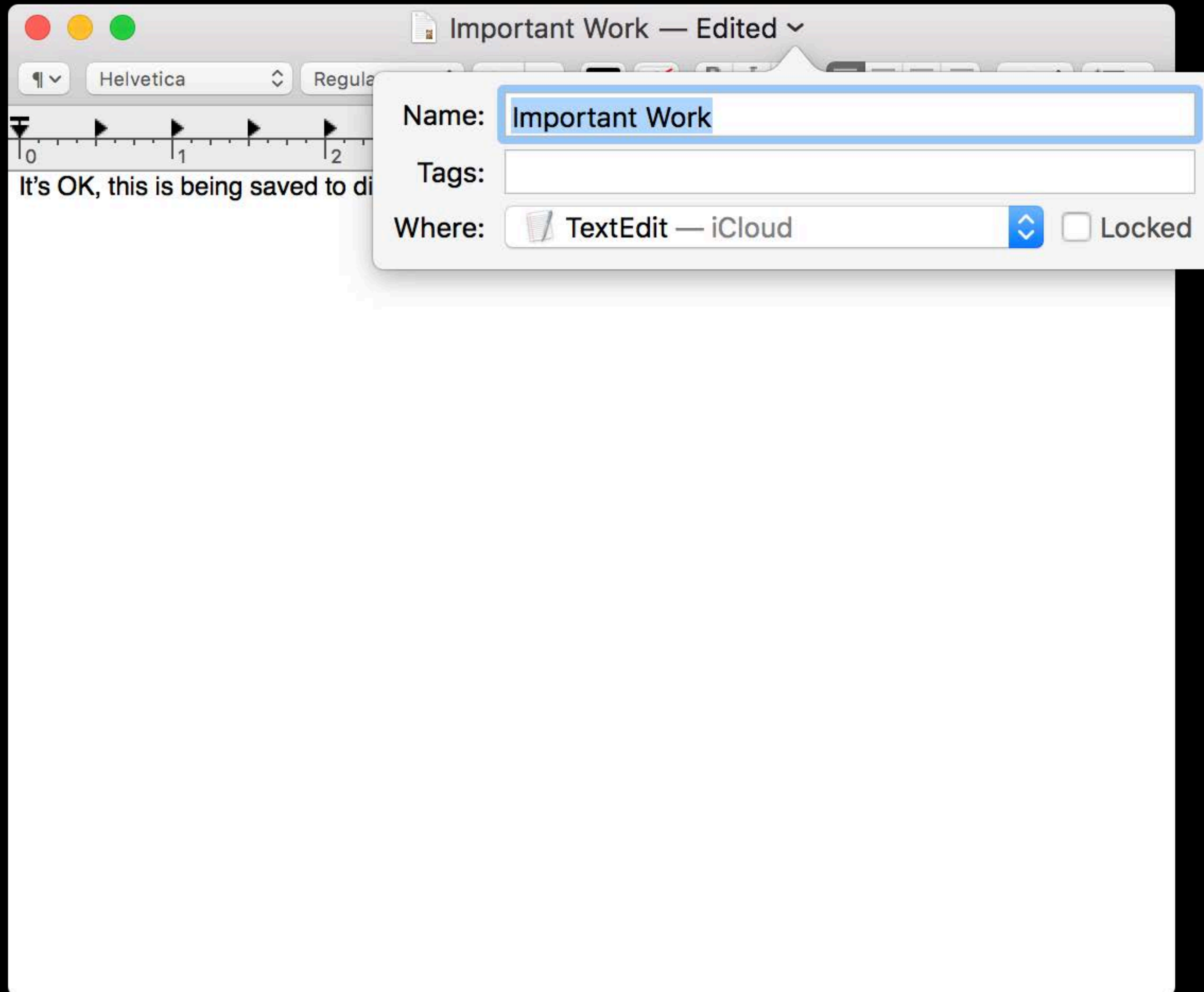


Documents

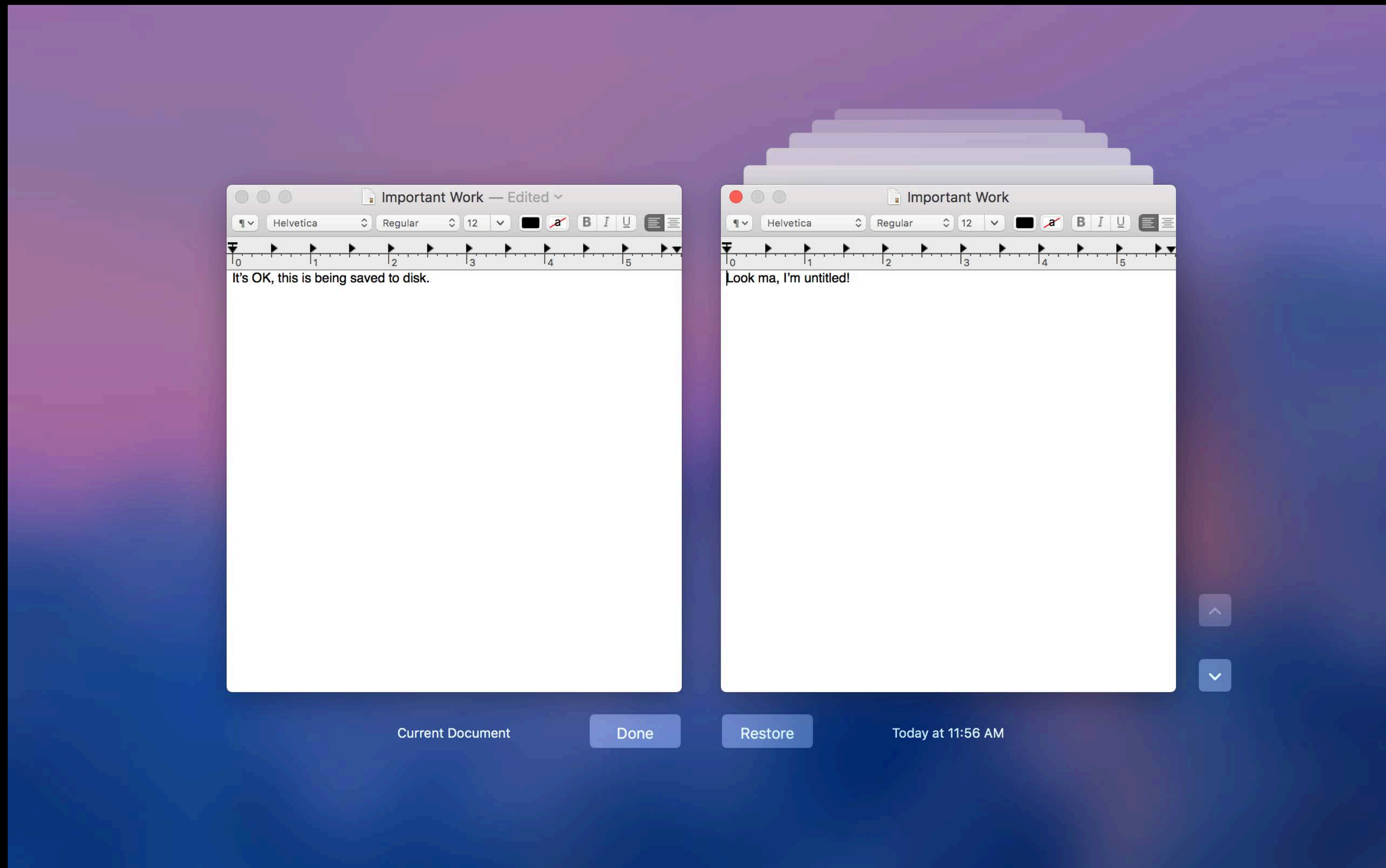
Document-based App



Document-based App

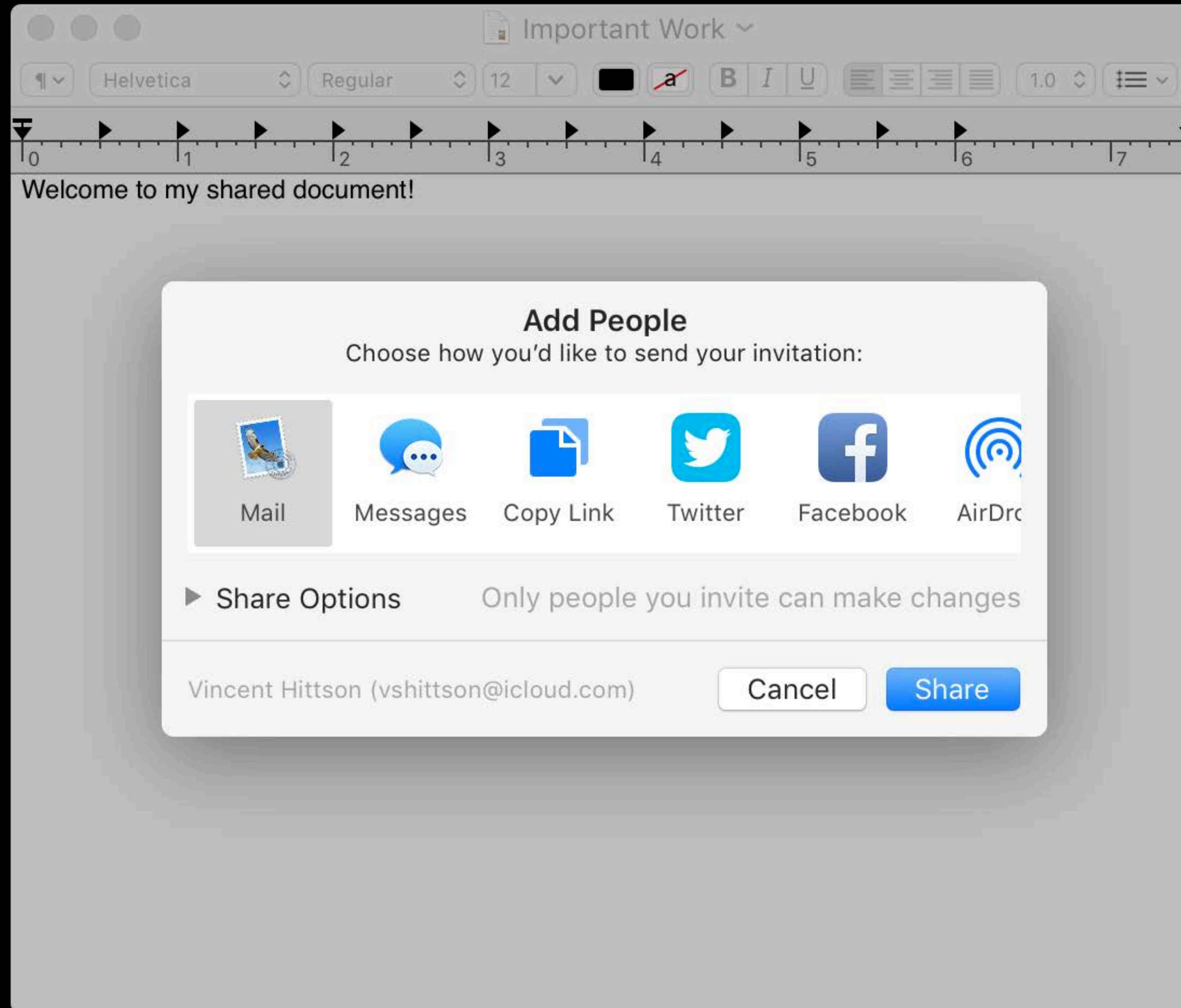


Document-based App



Document-based App

NEW



Use NSDocument!

```
open class NSDocument : ... {  
    open class var autosavesInPlace: Bool { get }  
}
```

```
open class NSDocument : ... {  
    open class var autosavesInPlace: Bool { get }  
}
```

```
class MyDocument : NSDocument {  
    override class var autosavesInPlace {  
        return true  
    }  
}
```

```
class MyDocument : NSDocument {  
    override class var autosavesInPlace {  
        return true  
    }  
}
```


42

Reporting Exceptions

```
open class NSApplication : NSResponder, ... {  
    open func reportException(_ exception: NSError)  
}
```

```
open class NSApplication : NSResponder, ... {  
    open func reportException(_ exception: NSError)  
}
```

NaN

Debugging Modes in Xcode

Xcode Debugging Modes



Xcode Debugging Modes

Debug view hierarchy



Xcode Debugging Modes

Debug view hierarchy

Debug memory graph



Xcode Debugging Modes

Debug view hierarchy

Debug memory graph

Simulate location



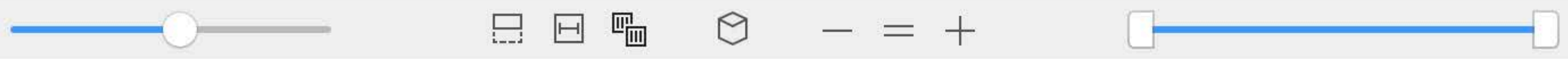
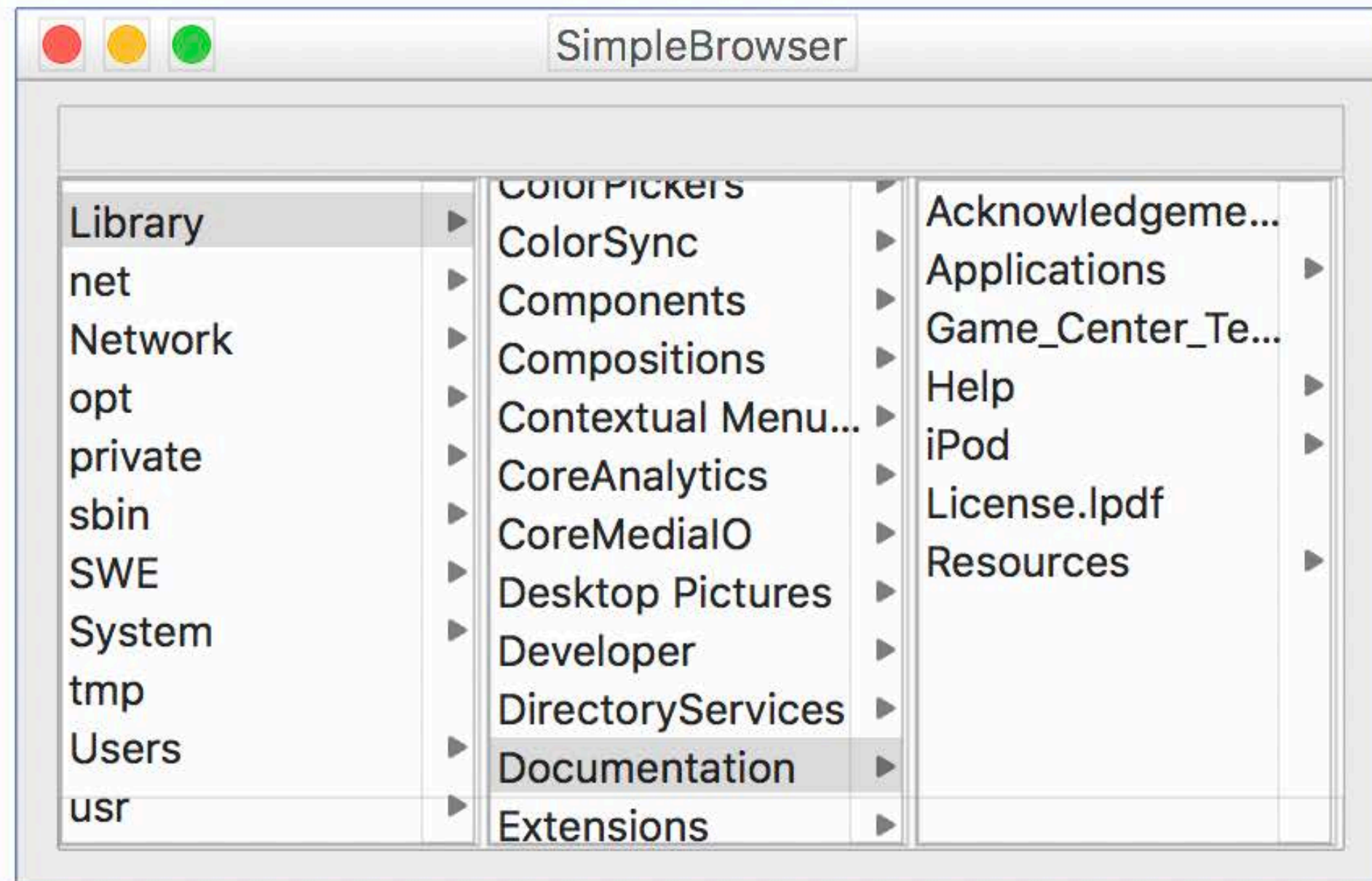
Xcode Debugging Modes

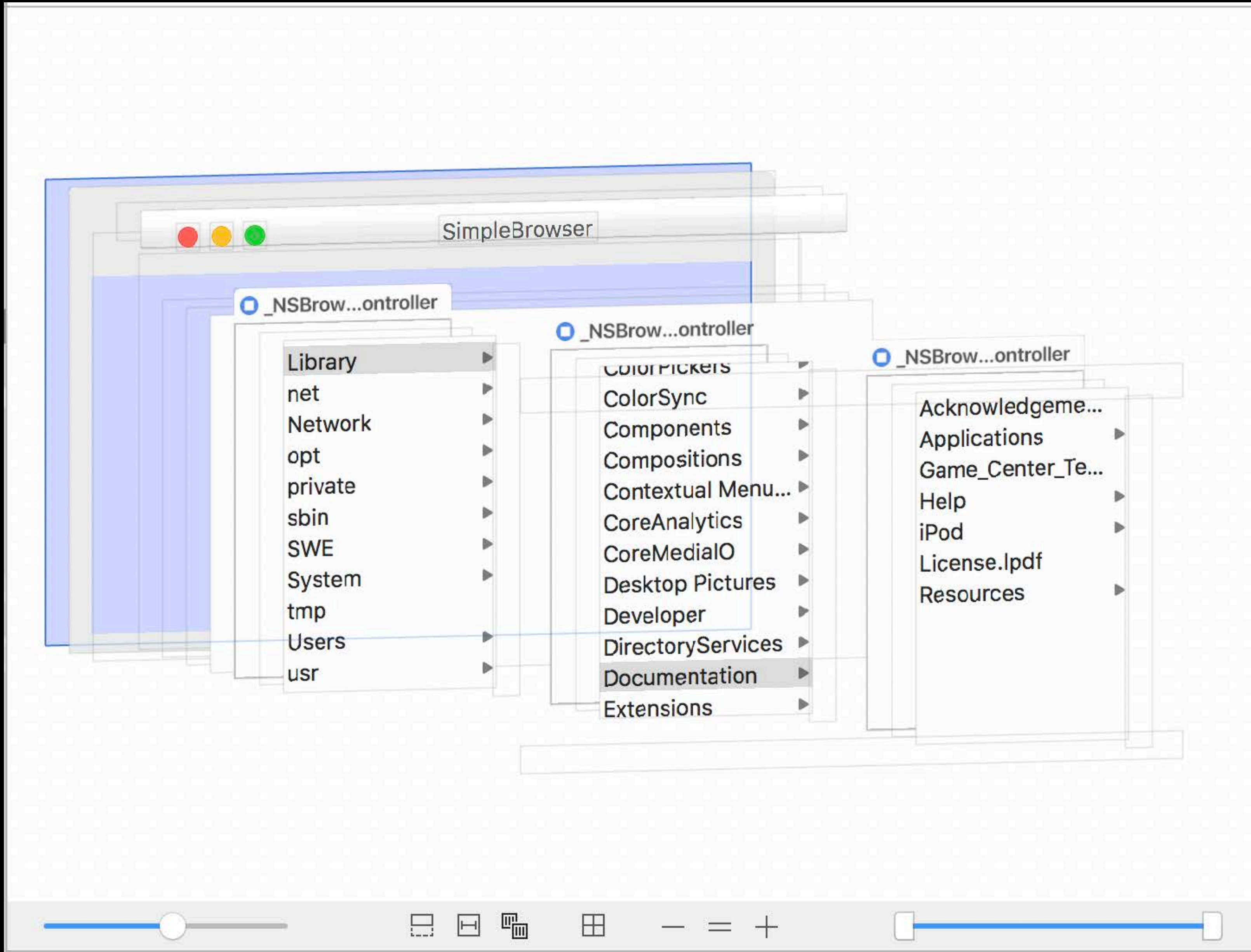
Debug view hierarchy

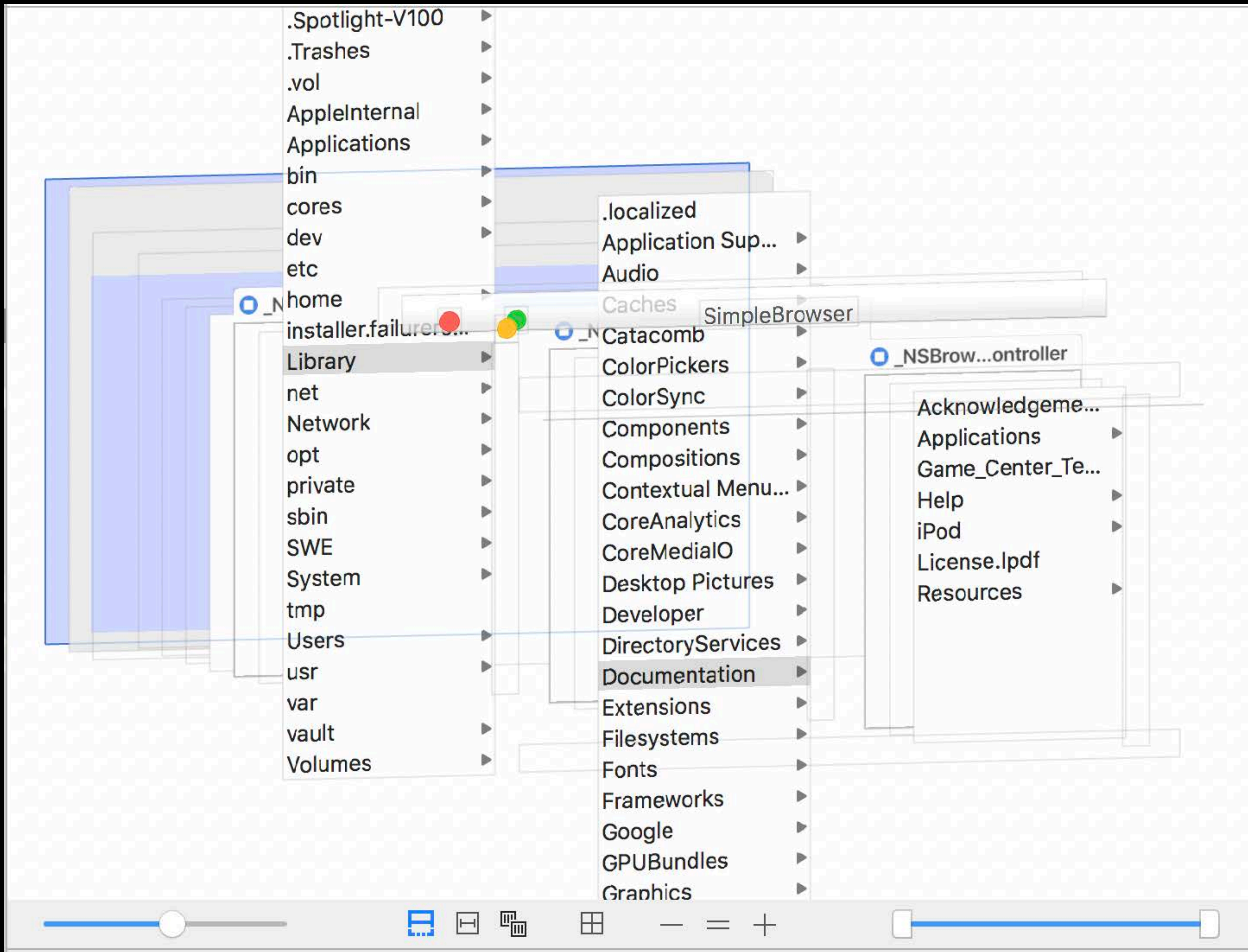
Debug memory graph

Simulate location









30512012

Write Bug Reports

How to Make Your Bug Reports Easier to Fix

<https://bugreport.apple.com/>

How to Make Your Bug Reports Easier to Fix

<https://bugreport.apple.com/>

Include steps to reproduce

How to Make Your Bug Reports Easier to Fix

<https://bugreport.apple.com/>

Include steps to reproduce

Include a sample app that builds and shows the problem

How to Make Your Bug Reports Easier to Fix

<https://bugreport.apple.com/>

Include steps to reproduce

Include a sample app that builds and shows the problem

Include any resources (image, database, etc.) that might be needed

How to Make Your Bug Reports Easier to Fix

<https://bugreport.apple.com/>

Include steps to reproduce

Include a sample app that builds and shows the problem

Include any resources (image, database, etc.) that might be needed

Attach a sysdiagnose or other logs

How to Make Your Bug Reports Easier to Fix

<https://bugreport.apple.com/>

Include steps to reproduce

Include a sample app that builds and shows the problem

Include any resources (image, database, etc.) that might be needed

Attach a sysdiagnose or other logs

- <https://developer.apple.com/bug-reporting/profiles-and-logs/>

How to Make Your Radars Easier to Fix

<https://bugreport.apple.com/>

The screenshot displays the 'Bug Reporting' section of the Apple website. At the top, there are navigation links for 'Profiles and Logs' and a 'View Bug Reporter' button. Below this is a filter menu with tabs for 'All', 'iOS', 'macOS', 'tvOS', 'watchOS', and 'Other', along with a search box labeled 'Search by name'. The main content area lists several diagnostic tools, each with a list icon and links to 'Instructions' and 'Profile' (where applicable).

Tool Name	Instructions	Profile
Stackshots for watchOS	Instructions	
Stream Logging for iOS	Instructions	Profile
Sync Diagnostics (DataAccess) for iOS	Instructions	Profile
sysdiagnose for iOS	Instructions	Profile
sysdiagnose for macOS	Instructions	
sysdiagnose for tvOS	Instructions	
sysdiagnose for watchOS	Instructions	Profile
System Profile Report for macOS	Instructions	
TCP Dump for iOS	Instructions	

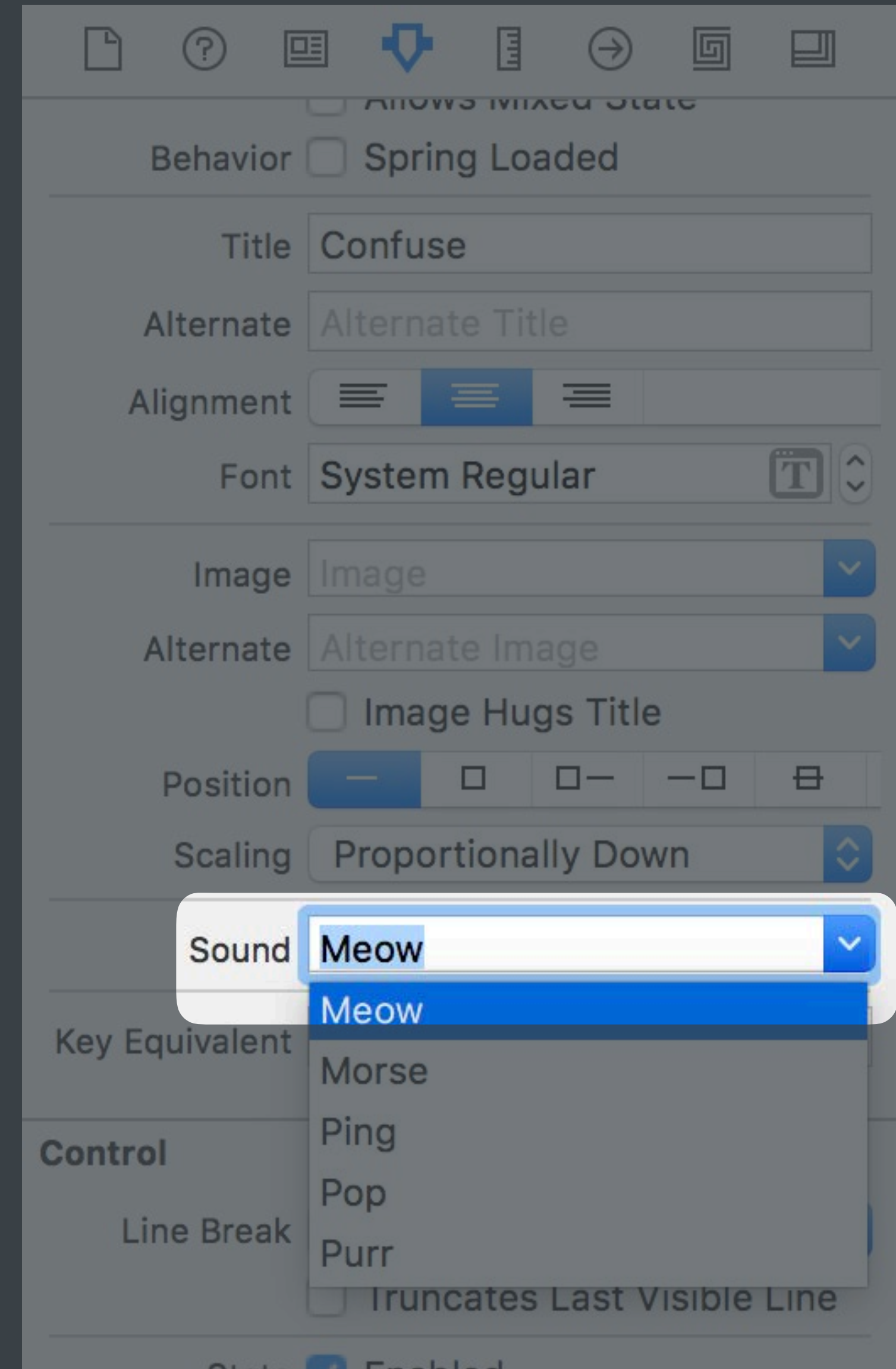
44.1

Bells and Whistles

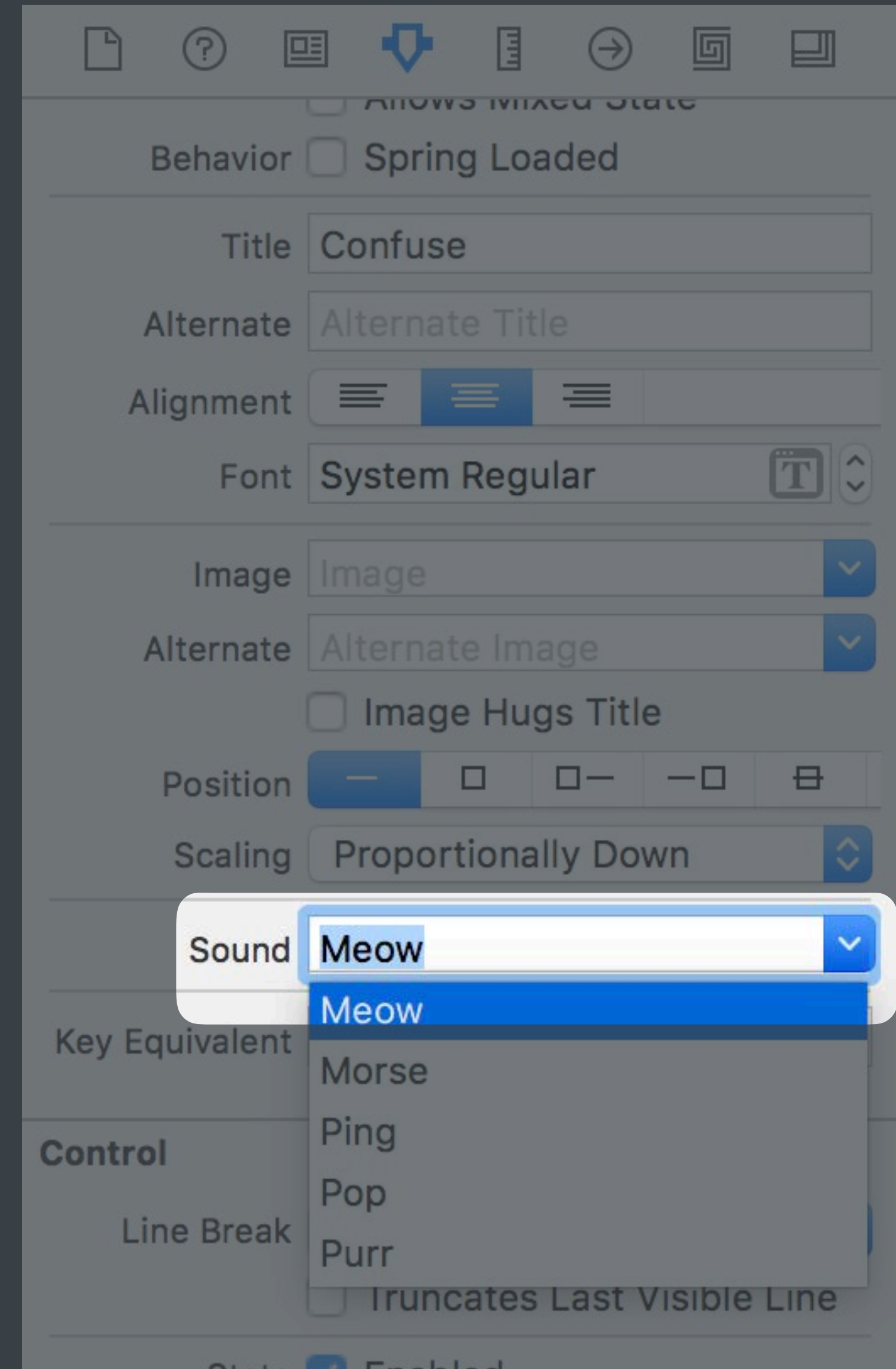

```
extension NSButton {  
    open var sound: NSSound?  
}
```

```
extension NSButton {  
    open var sound: NSSound?  
}
```

```
extension NSButton {  
    open var sound: NSSound?  
}
```



```
extension NSButton {  
    open var sound: NSSound?  
}
```



29

Demo



Cat Herder

N+1

Add Your Own Tips

New Documentation

<https://developer.apple.com/documentation>

New Documentation

<https://developer.apple.com/documentation>

Topics grouped by task

New Documentation

<https://developer.apple.com/documentation>

Topics grouped by task

Hierarchical structure

New Documentation

<https://developer.apple.com/documentation>

Topics grouped by task

Hierarchical structure

Example:




New Documentation

<https://developer.apple.com/documentation>

Topics grouped by task

Hierarchical structure

Example:

 AppKit >  Views and Controls >  NSView

New Documentation

<https://developer.apple.com/documentation>

New Documentation

<https://developer.apple.com/documentation>

Reference

New Documentation

<https://developer.apple.com/documentation>

Reference

Conceptual

New Documentation

<https://developer.apple.com/documentation>

Reference

Conceptual

Sample Code

New Documentation

<https://developer.apple.com/documentation>

Reference

Conceptual

Sample Code

Release Notes

<https://developer.apple.com/library/content/releasenotes/AppKit/RN-AppKit/index.html>

macOS 10.13 Release Notes Cocoa Application Framework

The Cocoa Application Framework (also referred to as the Application Kit, or AppKit) is one of the core Cocoa frameworks. It provides functionality and associated APIs for applications, including objects for graphical user interfaces (GUIs), event-handling mechanisms, application services, and drawing and image composition facilities.

Some of the major topics covered in this document:

- [NSCollectionView Responsive Scrolling](#)
- [New Enumerations](#)
- [Accessibility](#)
- [NSLevelIndicator](#)
- [Layer-backed Views](#)
- [NSTableView Automatic Row Heights](#)

```
/*
    UIImage.h
    Application Kit
    Copyright (c) 1994–2017, Apple Inc.
    All rights reserved.
*/

...

// Note that the block passed to the below method may be invoked whenever and on whatever
// thread the image itself is drawn on. Care should be taken to ensure that all state accessed
// within the drawingHandler block is done so in a thread safe manner.
+ (UIImage *)imageWithSize:(NSSize)size flipped:
    (BOOL)drawingHandlerShouldBeCalledWithFlippedContext drawingHandler:
        (BOOL (^)(NSRect dstRect))drawingHandler NS_AVAILABLE_MAC(10_8);

...
```

Tweet Your Tips!

Use #WWDC17 and #cocoatip



THOMAS H. H.
Developer

 Follow



NSMutableDictionary is like NSMutableSet, but it can contain arbitrary pointers and references them weakly.
[#WWDC17](#) [#cocoatip](#)

11:33 PM - 7 Jun 2017



1



John Wick
@johnwick

 Follow



Simplify complex ObjC generics w/typedefs

```
typedef NSDictionary<NSString*, NSString*>  
StringDict;
```

```
StringDict* a = @[@"b": @"c"];
```

[#cocoatip](#)

10:52 AM - 8 Jun 2017



2

```
typealias StringDict = Dictionary<String, String>
```

```
var a: StringDict = [:]
```

```
a = ["b": "c"]
```



Michael Barkoch
@michaelbarkoch

 Follow



If you're coming from iOS, there's
NSWindowController in AppKit. There are
times you should use it over
NSViewController. [#wwdc2017](#) [#cocoatip](#)

7:22 PM - 7 Jun 2017



1



2



Thomas Zischling
Developer

 Follow



Use Xcode's "Add Expression" to get QuickLook previews for arbitrary addresses (Space to open QL panel) [#WWDC17](#)
[#cocoatip](#)

8:07 AM - 8 Jun 2017



3



6

Quick Look Expression

The screenshot shows the Xcode interface with the Quick Look Expression window open. The window title is "Running Cat Herder : Cat Herder". The breadcrumb navigation shows "Cat Herder > Thread 1 > 0 mach_msg_trap".

The thread dump shows the following assembly instructions:

```
1 libsystem_kernel.dylib`mach_msg_trap:
2 0x7fffa8425fbc <+0>: movq %rcx, %r10
3 0x7fffa8425fbf <+3>: movl $0x100001f, %eax ; imm = 0x100001F
4 0x7fffa8425fc4 <+8>: syscall
5 -> 0x7fffa8425fc6 <+10>: retq Thread 1: signal SIGSTOP
6 0x7fffa8425fc7 <+11>: nop
7
```

The Quick Look expression window shows the following result:

```
(lldb) po [UIImage imageNamed:@"AddCat"]
<UIImage 0x6080004693c0 Name=AddCat Size={512, 512} Reps=(
  "NSCGImageRep 0x60800008fcd0 Size={512, 512}
  ColorSpace=sRGB IEC61966-2.1 colorspace BPS=0
  Pixels=1024x1024 Alpha=NO"
)>
```


Quick Look Expression

The screenshot shows the Xcode LLDB interface. The top toolbar includes standard window controls and navigation icons. The breadcrumb path is "Cat Herder > Thread 1 > 0 mach_msg_trap". The main pane displays assembly code for the `libsystem_kernel.dylib`mach_msg_trap:` function, with the instruction `retq` at address `0x7fffa8425fc6` highlighted. A tooltip for this instruction reads "Thread 1: signal SIGSTOP".

The left sidebar shows the process "Cat Herder PID 2385" with system metrics: CPU (0%), Memory (20.8 MB), Energy Impact (Low), Disk (Zero KB/s), and Network (Zero KB/s). Below this, the thread queue is visible, with "0 mach_msg_trap" selected.

The bottom toolbar contains filter and view options. A white tooltip box in the bottom right corner displays the following Quick Look expression and its output:

```
(lldb) po [UIImage imageNamed:@"AddCat"]  
<UIImage 0x6080004693c0 Name=AddCat Size={512, 512} Reps=(  
  "NSCGImageRep 0x60800008fcd0 Size={512, 512}  
  ColorSpace=sRGB IEC61966-2.1 colorspace BPS=0  
  Pixels=1024x1024 Alpha=NO"  
)>
```


Quick Look Expression

The screenshot shows the Xcode interface with the Quick Look Expression window open. The window title is "Running Cat Herder : Cat Herder". The left sidebar shows the project structure for "Cat Herder PID 2385", including system metrics (CPU, Memory, Energy Impact, Disk, Network) and a thread queue. The main area displays assembly code for the "mach_msg_trap" function in "libsystem_kernel.dylib". The current instruction is highlighted: "0x7fffa8425fc6 <+10>: retq". A context menu is open over this instruction, listing various actions such as "Print Description of 'variable'", "Copy", "View Value As", "Edit Value...", "Edit Summary Format...", "Add Expression..." (highlighted in blue), "Delete Expression", "Watch 'variable'", "View Memory of 'variable'", "Show Types", "Show Raw Values", "Sort By", and "Debug Area Help". The right sidebar shows the Quick Look preview of the selected expression: "(lldb) po [UIImage imageNamed:@"AddCat"]", displaying the image's metadata.

```
1 libsystem_kernel.dylib`mach_msg_trap:
2 0x7fffa8425fbc <+0>: movq %rcx, %r10
3 0x7fffa8425fbf <+3>: movl $0x100001f, %eax ; imm = 0x100001F
4 0x7fffa8425fc4 <+8>: syscall
5 -> 0x7fffa8425fc6 <+10>: retq
6 0x7fffa8425fc7 <+11>: nop
7
```

(lldb) po [UIImage imageNamed:@"AddCat"]
<UIImage 0x6080004693c0 Name=AddCat Size={512, 512} Reps=(
 "NSCGImageRep 0x60800008fcd0 Size={512, 512}
 ColorSpace=sRGB IEC61966-2.1 colorspace BPS=0
 Pixels=1024x1024 Alpha=NO"
)>

Quick Look Expression

The image shows a screenshot of Xcode's Quick Look Expression window. The window title is "Running Cat Herder : Cat Herder". The main area displays assembly code for the function `libsystem_kernel.dylib`mach_msg_trap:`. The code is as follows:

```
1  libsystem_kernel.dylib`mach_msg_trap:
2      0x7fffa8425fbc <+0>:  movq   %rcx, %r10
3      0x7fffa8425fbf <+3>:  movl   $0x100001f, %eax      ; imm = 0x100001F
4      0x7fffa8425fc4 <+8>:  syscall
5  ->  0x7fffa8425fc6 <+10>: retq
6      0x7fffa8425fc7 <+11>: nop
7
```

A context menu is open over the code, with the "Add Expression..." option highlighted in blue. The menu items are:

- Print Description of "variable"
- Copy
- View Value As
- Edit Value...
- Edit Summary Format...
- Add Expression...**
- Delete Expression
- Watch "variable"
- View Memory of "variable"
- ✓ Show Types
- Show Raw Values
- Sort By
- Debug Area Help

The background of the window shows the "Cat Herder" application running on "My Mac". The left sidebar displays system metrics (CPU, Memory, Energy Impact, Disk, Network) and a thread list for "Thread 1 Queue: com.a...-thread (serial)". The thread list includes "0 mach_msg_trap", "1 mach_msg", "2 __CFRunLoopServiceMachPort", "3 __CFRunLoopRun", "4 CFRunLoopRunSpecific", and "5 RunCurrentEventLoopInMode". The right sidebar shows the Quick Look Expression result for the selected line of code:

```
(lldb) po [UIImage imageNamed:@"AddCat"]
<UIImage 0x6080004693c0 Name=AddCat Size={512, 512} Reps=(
  "NSCGImageRep 0x60800008fcd0 Size={512, 512}
  ColorSpace=sRGB IEC61966-2.1 colorspace BPS=0
  Pixels=1024x1024 Alpha=NO"
)>
```


Quick Look Expression

The screenshot shows the Xcode interface with the Quick Look Expression window open. The expression `(UIImage *)0x6080004693c0` is entered into the text field. The window displays the following output:

```
(lldb) po [UIImage imageNamed:@"AddCat"]  
<UIImage 0x6080004693c0 Name=AddCat Size={512, 512} Reps=(  
  "NSCGImageRep 0x60800008fcd0 Size={512, 512}  
  ColorSpace=sRGB IEC61966-2.1 colorspace BPS=0  
  Pixels=1024x1024 Alpha=NO"  
)>
```

The background of the Xcode window shows the Cat Herder application running on Thread 1, with the `mach_msg_trap` function highlighted. The stack trace includes the following instructions:

```
1 libsystem_kernel.dylib`mach_msg_trap:  
2 0x7fffa8425fbc <+0>: movq %rcx, %r10  
3 0x7fffa8425fbf <+3>: movl $0x100001f, %eax ; imm = 0x100001F  
4 0x7fffa8425fc4 <+8>: syscall  
5 -> 0x7fffa8425fc6 <+10>: retq Thread 1: signal SIGSTOP  
6 0x7fffa8425fc7 <+11>: nop  
7
```


Quick Look Expression

The image shows a screenshot of Xcode's Quick Look Expression dialog box. The dialog box is a white rounded rectangle with a blue border, containing the following elements:

- Expression:** A text input field containing the expression `(UIImage *)0x6080004693c0`.
- Show in All Stack Frames:** A checkbox that is currently unchecked.
- Done:** A button located at the bottom right of the dialog box.

The background shows the Xcode interface with the following details:

- Top Bar:** Shows the process name "Cat Herder" and the thread "Thread 1".
- Left Panel:** Displays system metrics for "Cat Herder PID 2385", including CPU (0%), Memory (20.8 MB), Energy Impact (Low), and Disk usage.
- Debugger Window:** Shows assembly code for `libsystem_kernel.dylib`mach_msg_trap:`. The current instruction is `0x7fffa8425fc6 <+10>: retq`, which is highlighted in green. A message "Thread 1: signal SIGSTOP" is visible on the right.
- Bottom Panel:** Shows the Quick Look result for the expression, displaying a detailed description of an `UIImage` object: `(lldb) po [UIImage imageNamed:@"AddCat"]`. The output shows the image's name, size, and other properties.

Quick Look Expression

The screenshot shows the Xcode debugger interface. The top toolbar includes play, stop, and search icons. The breadcrumb navigation shows 'Cat Herder > Thread 1 > 0 mach_msg_trap'. The left sidebar displays system metrics for 'Cat Herder PID 2385' (CPU 0%, Memory 20.8 MB, Energy Impact Low, Disk Zero KB/s, Network Zero KB/s) and a thread queue for 'com.a...-thread (serial)'. The main pane shows assembly code for 'libsystem_kernel.dylib`mach_msg_trap:' with instructions at addresses 0x7fffa8425fbc through 0x7fffa8425fc7. Instruction 5 is highlighted: '-> 0x7fffa8425fc6 <+10>: retq'. A tooltip for this instruction shows 'Thread 1: signal SIGSTOP'. The bottom pane shows a Quick Look expression: '(lldb) po [UIImage imageNamed:@"AddCat"]'. The result is a dictionary: '<UIImage 0x6080004693c0 Name=AddCat Size={512, 512} Reps=("NSCGImageRep 0x60800008fcd0 Size={512, 512} ColorSpace=sRGB IEC61966-2.1 colorspace BPS=0 Pixels=1024x1024 Alpha=NO")>'. The bottom toolbar includes 'Filter', 'Auto', and 'All Output' options.

Quick Look Expression

The screenshot shows the Xcode IDE with the Quick Look Expression window open. The window title is "Cat Herder" and the current thread is "Thread 1" at address "0 mach_msg_trap". The expression being evaluated is `(NSData *)0x6080004693c0 = (NSData *) 0x6080004693c0`. The resulting memory dump shows the following data:

```
(11db) po [NSData imageNamed:@"AddCat"]  
<NSData 0x6080004693c0 Name=AddCat Size={512, 512} Reps=(  
  "NSDataImageRep 0x60800008fcd0 Size={512, 512}  
  ColorSpace=sRGB IEC61966-2.1 colorspace BPS=0  
  Pixels=1024x1024 Alpha=NO"  
)>
```

The background of the IDE shows the "Cat Herder" application running on "My Mac". The left sidebar displays system metrics for the application, including CPU usage (0%), Memory (20.8 MB), Energy Impact (Low), Disk (Zero KB/s), and Network (Zero KB/s). The main editor area shows assembly code for the `libsystem_kernel.dylib`mach_msg_trap:` function, with the current instruction being `retq` at address `0x7fffa8425fc6`. A notification "Thread 1: signal SIGSTOP" is visible on the right side of the assembly view.

Quick Look Expression

The image shows a Mac OS desktop with a Quick Look window open over a yellow cat image. The Quick Look window title is `(NSImage *)0x6080004693c0` and includes an `Open With Preview` button. A green plus sign is overlaid on the bottom right of the image. In the background, a System Monitor window is visible, showing system metrics for a process named "Cat Herder PID 2385".

System Monitor Data:

Component	Value
CPU	0%
Memory	20.8 MB
Energy Impact	Low
Disk	Zero KB/s
Network	Zero KB/s

Thread Queue:

Thread ID	Thread Name
0	mach_msg_trap
1	mach_msg
2	__CFRunLoopServiceMachPort
3	__CFRunLoopRun
4	CFRunLoopRunSpecific
5	RunCurrentEventLoopInMod...

Quick Look Expression Output:

```
addCat"]  
Cat Size={512,  
Size={512, 512}  
orspace BPS=0
```


Quick Look Expression

The screenshot shows a web browser window with the URL `developer.apple.com`. The page title is "Enabling Quick Look for Custom Types". The navigation bar includes "Guides and Sample Code" and the "Developer" logo. The main content area is titled "Quick Look for Custom Types in the Xcode Debugger".

Table of Contents:

- Introduction
- ▶ Enabling Quick Look for Custom Types
- ▶ Operating System Types Supporting `debugQuickLookObject`
- Revision History

Enabling Quick Look for Custom Types [Previous](#) [Next](#)

The variables Quick Look feature in the Xcode debugger allows you to obtain a quick visual assessment of the state of an object variable through a graphical rendering, displayed in a popover window either in the debugger variables view or in place in your source code. For supported operating system types, Quick Look displays debugger Quick Look object primitives to service this visualization.

This chapter describes how you implement a Quick Look method for your custom class types so that object variables of those types can also be rendered visually in the Quick Look popover window.

Implement the Quick Look method

For your custom class type, implement a method named `debugQuickLookObject`.

```
- (id)debugQuickLookObject
```

[Feedback](#)

More Information

<https://developer.apple.com/wwdc17/236>

Related Sessions

Localizing with Xcode 9

WWDC 2017

What's New in Swift

WWDC 2017

What's New in Cocoa

WWDC 2017

What's New in Core Data

WWDC 2017

What's New in Foundation

WWDC 2017

What's New in Accessibility

WWDC 2017

Labs

Cocoa Lab

Technology Lab B

Fri 1:50PM–3:20PM

