Cocoa Development Tips

Twenty-nine things you may not know about Cocoa

Session 236

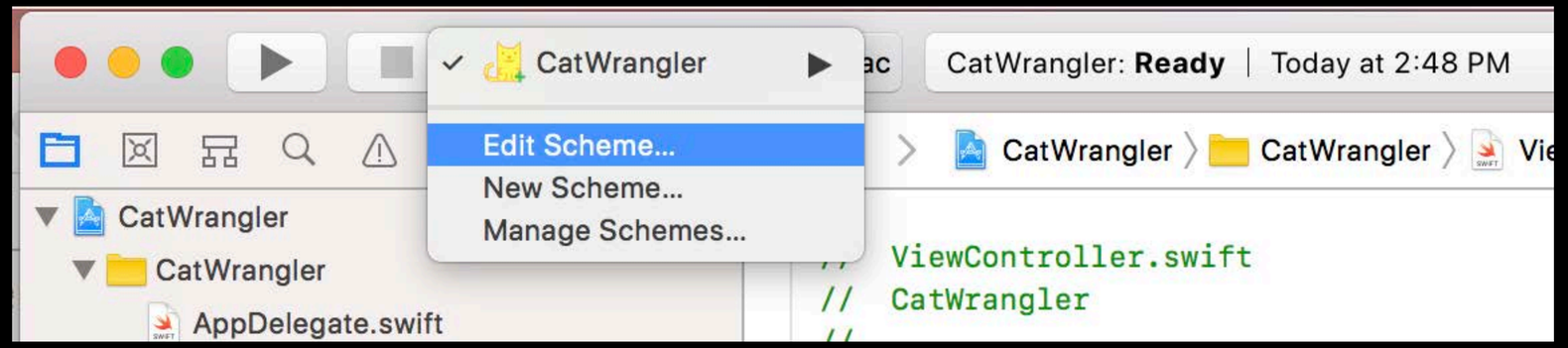
Rachel Goldeen, Cocoa Engineer
Vincent Hittson, Cocoa Engineer

© 2017 Apple Inc. All rights reserved. Redistribution or public display not permitted without written permission from Apple.

Internationalization

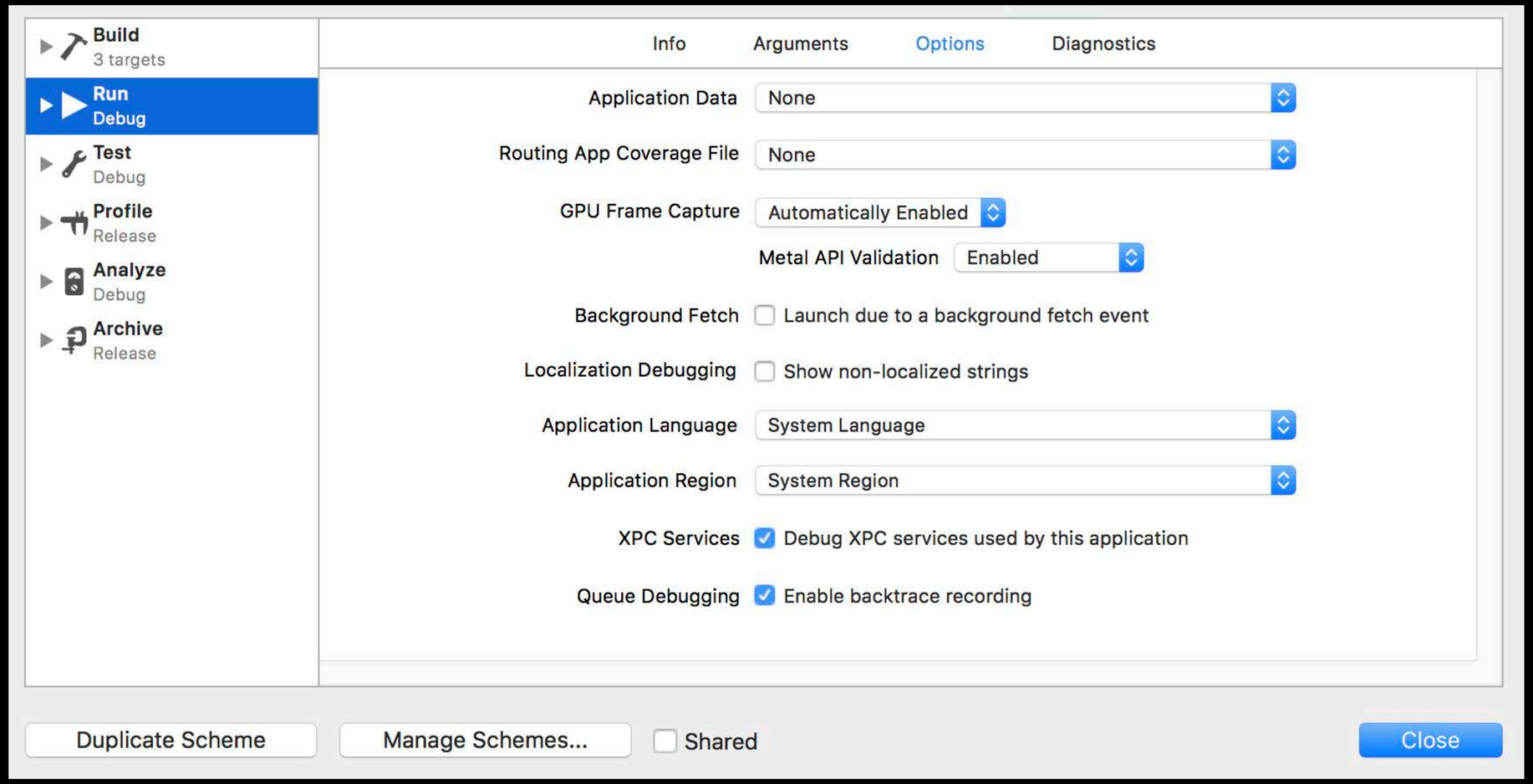
Internationalization

Xcode Scheme Options



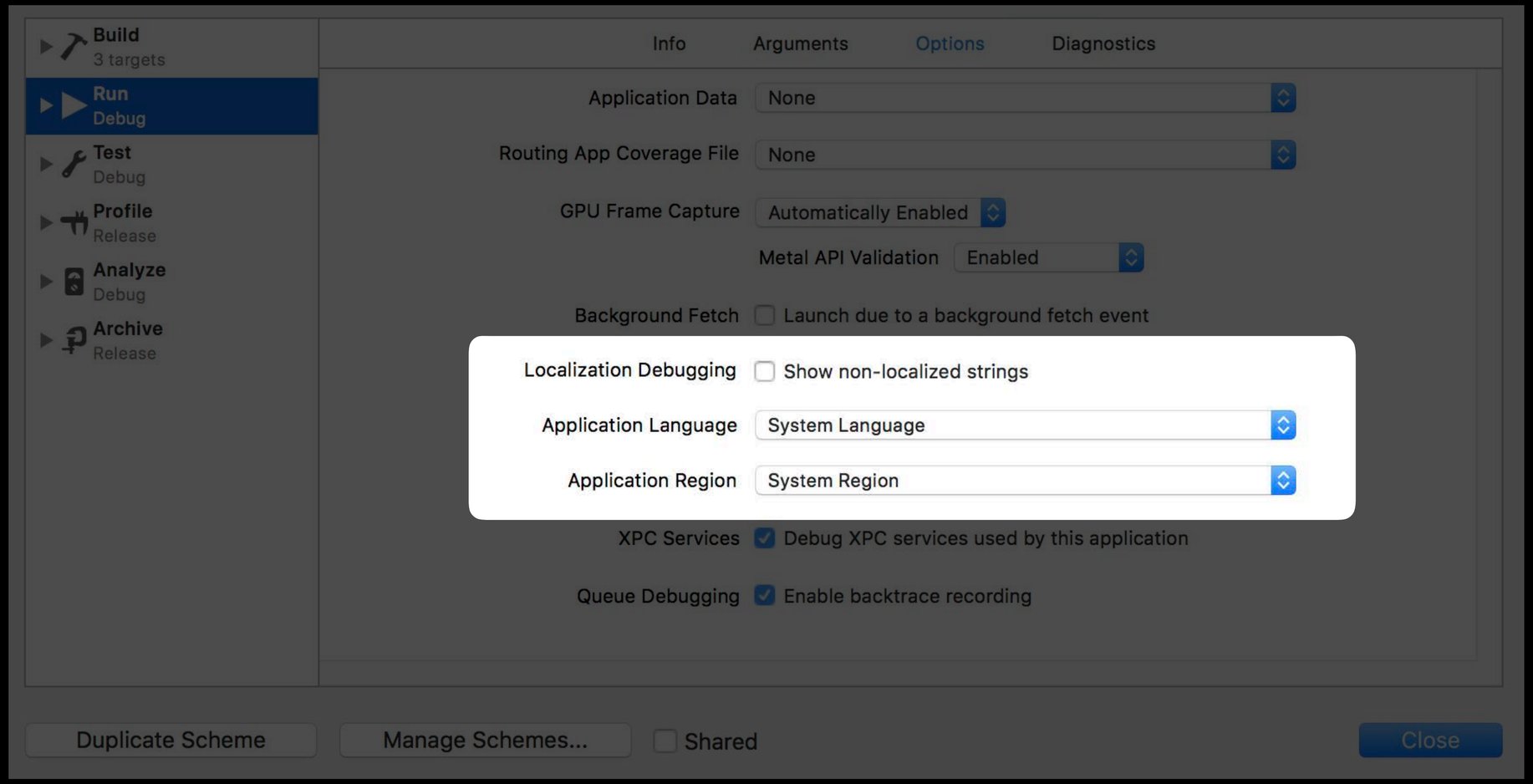
Internationalization

Xcode Scheme Options



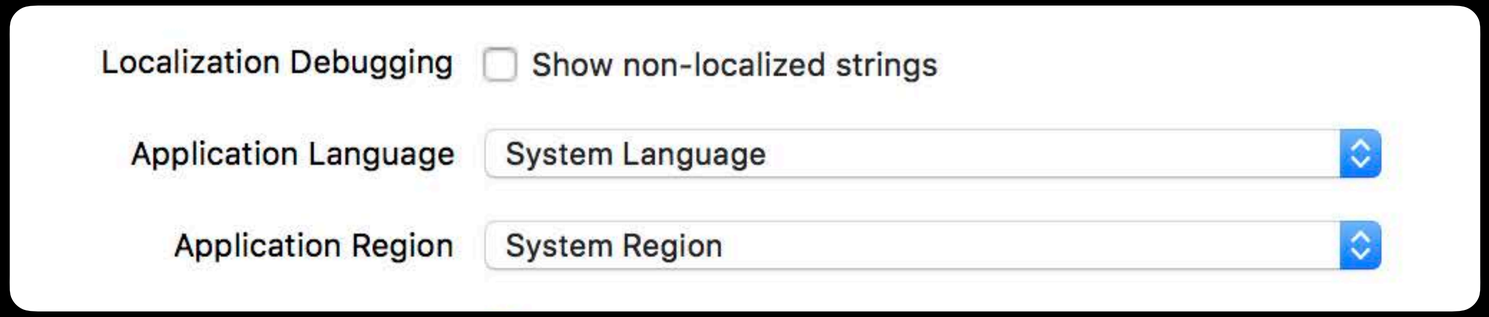
Internationalization

Xcode Scheme Options



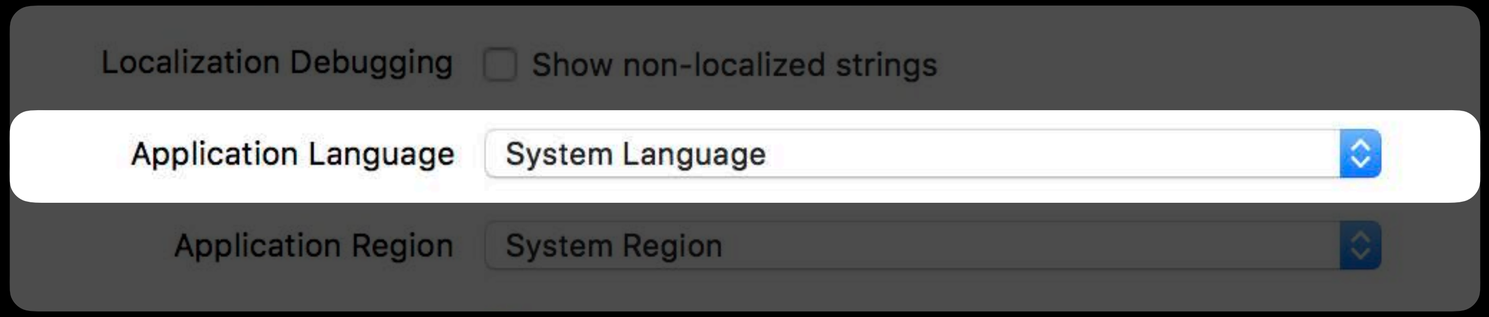
Internationalization

Xcode Scheme Options



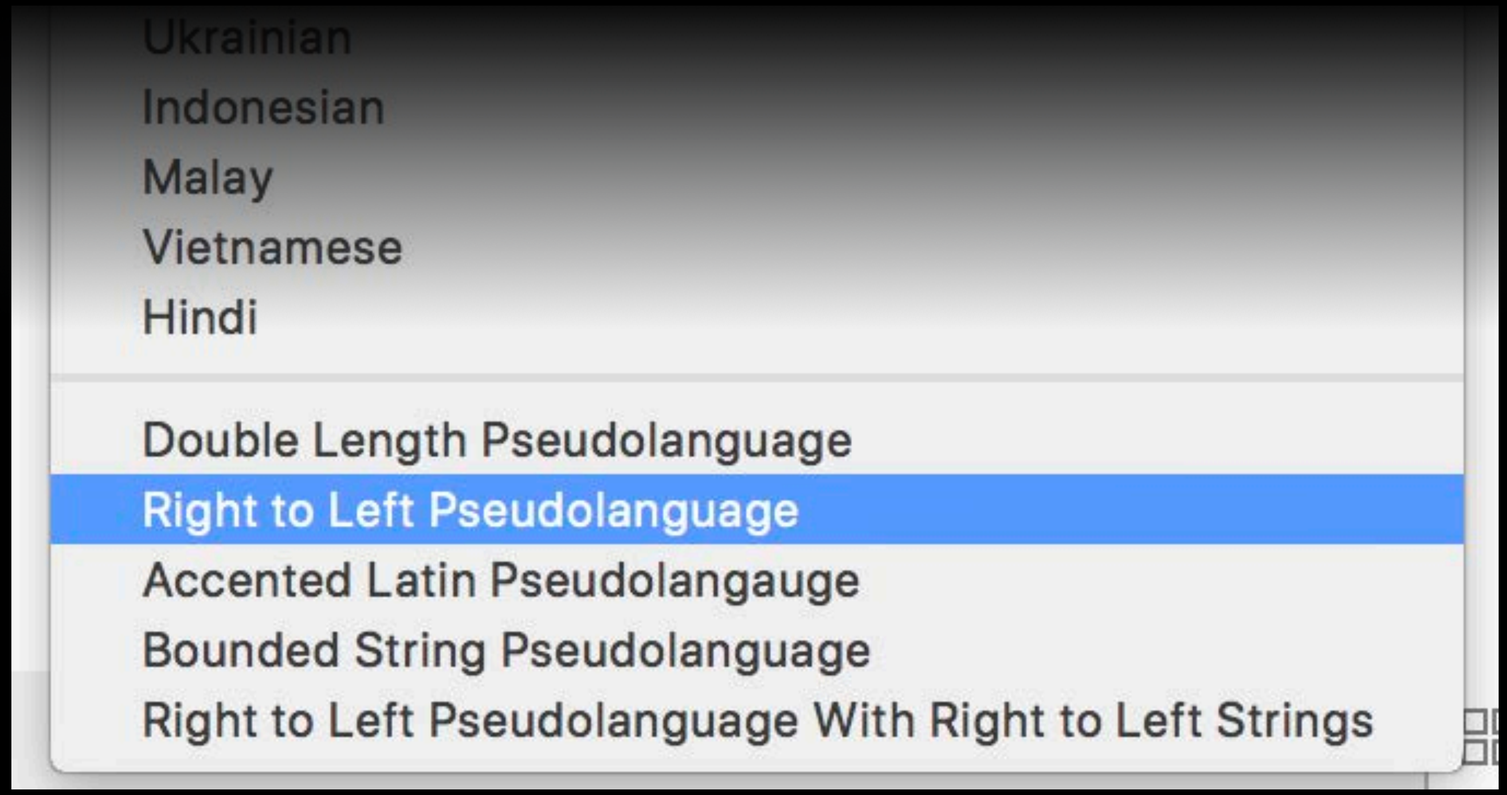
Internationalization

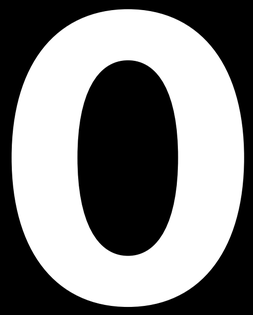
Xcode Scheme Options



Internationalization

Xcode Scheme Options



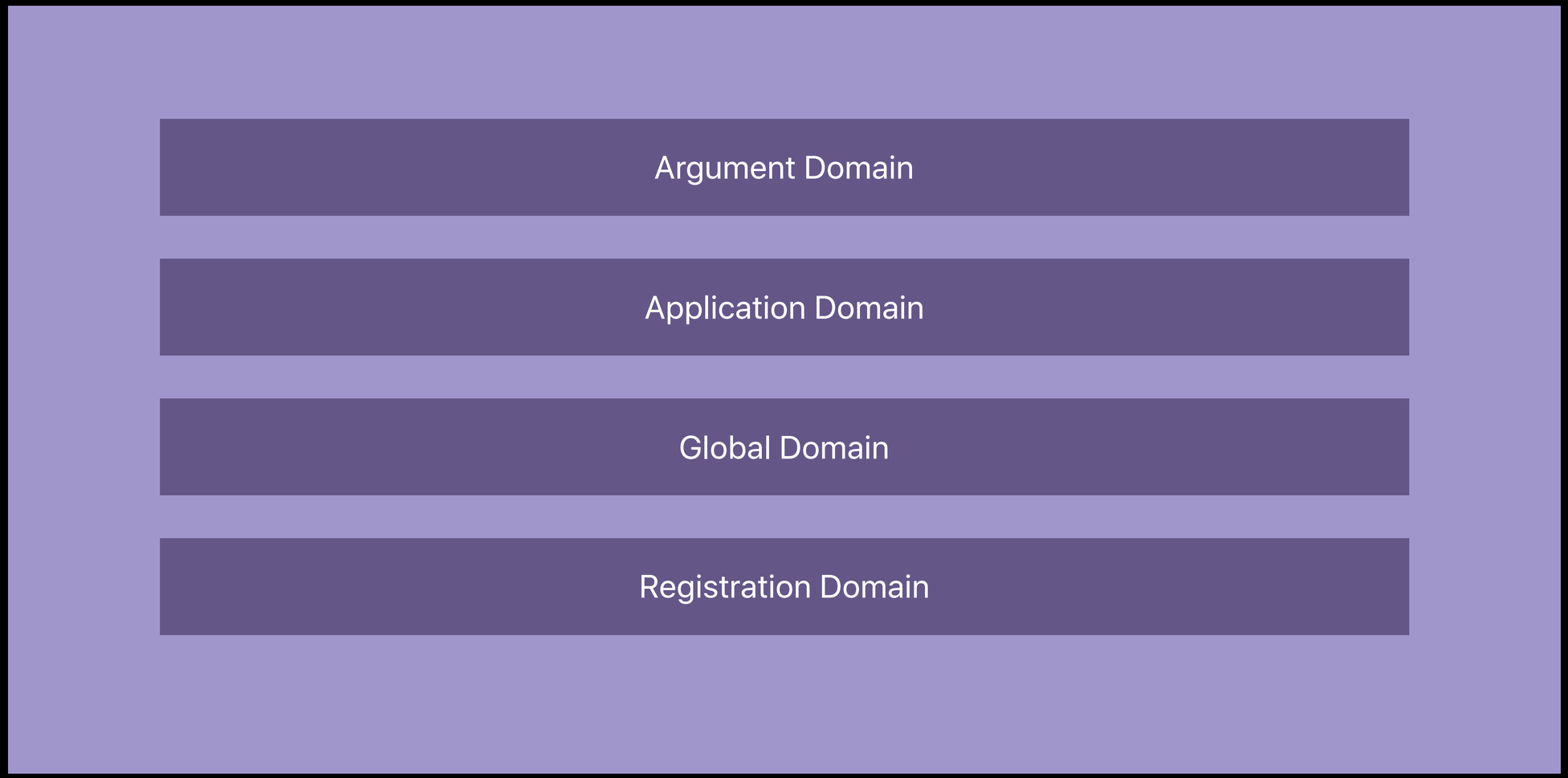


User Defaults

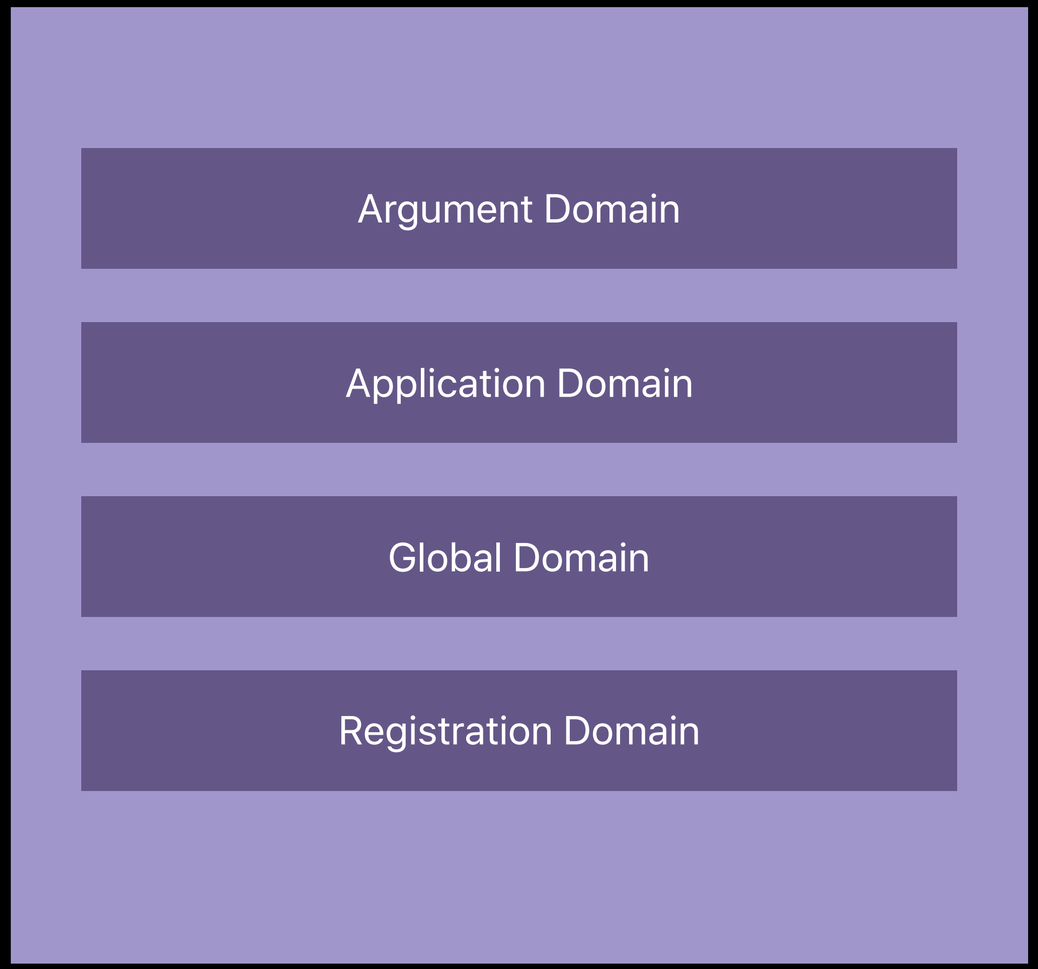
User Defaults

```
UserDefaults.standard
```

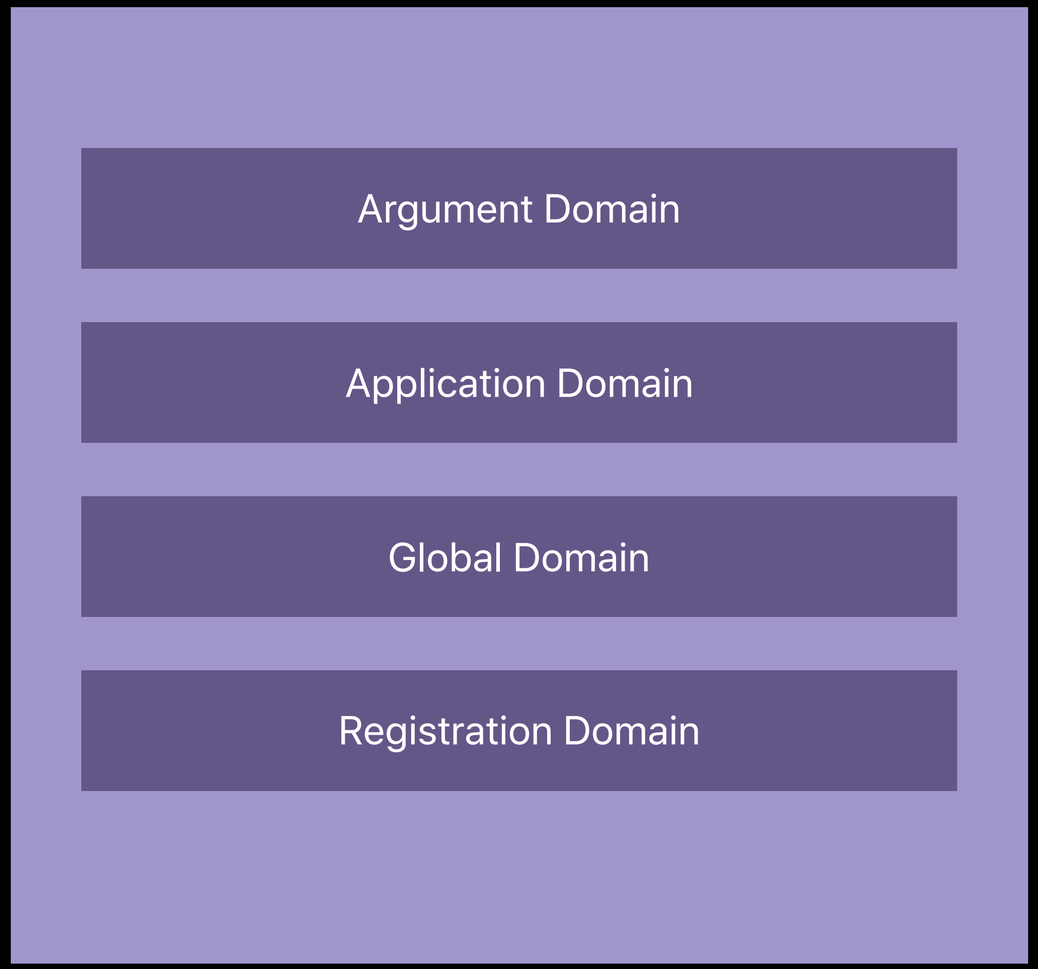
User Defaults



User Defaults

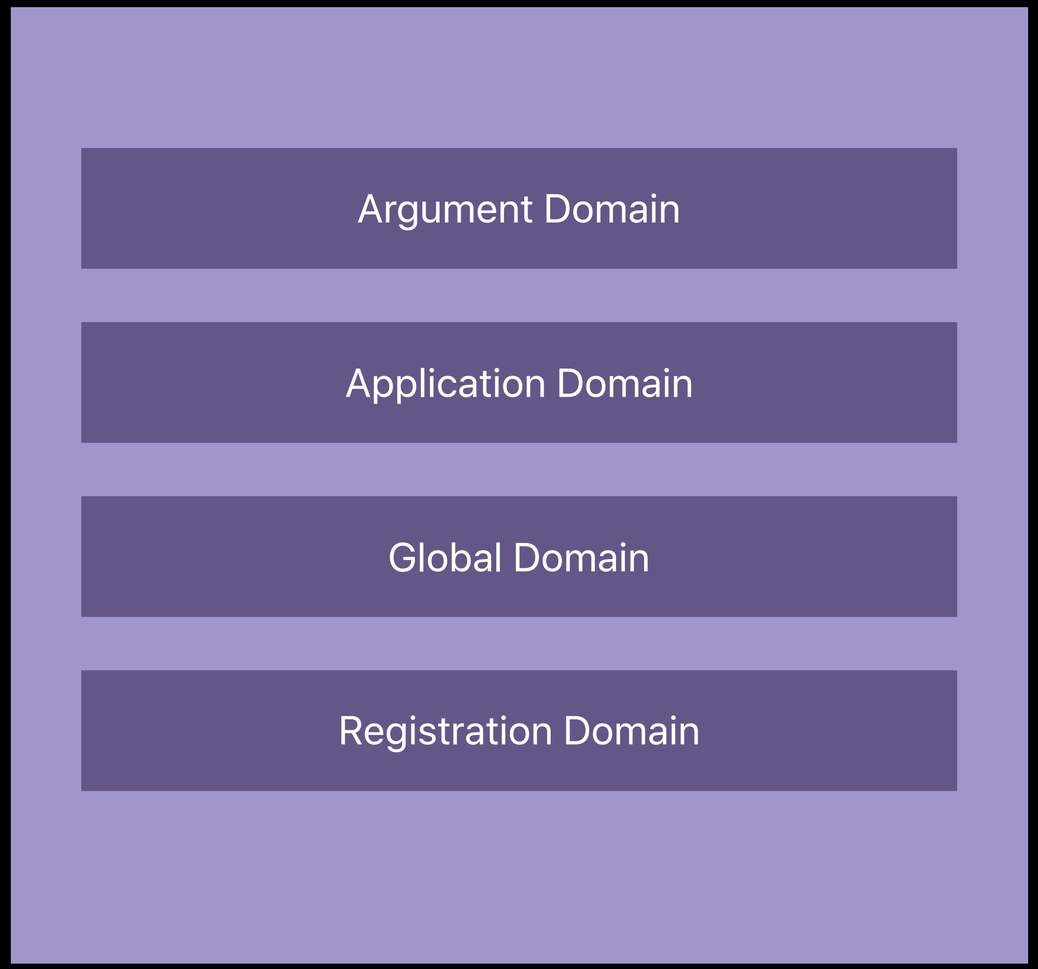


User Defaults



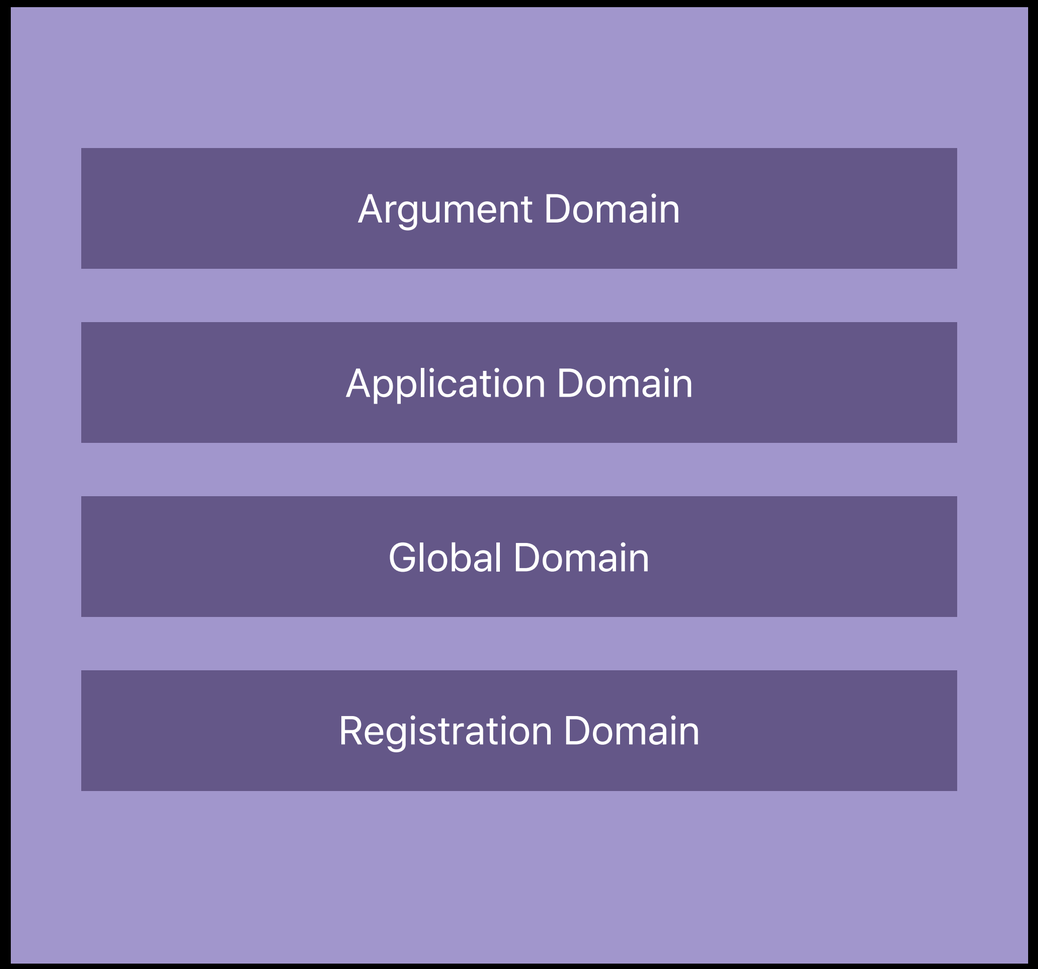

```
UserDefaults.standard.object(forKey:"showAtLaunch")
```

User Defaults

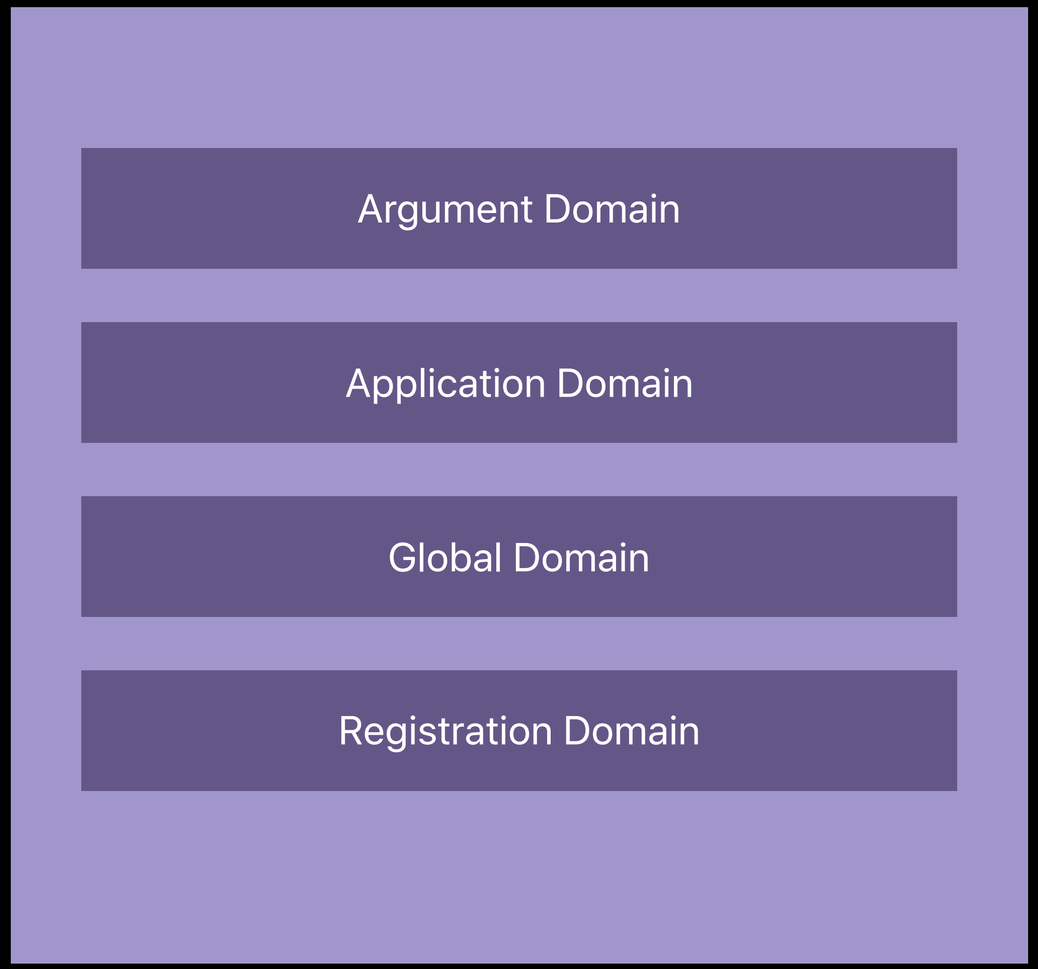



```
UserDefaults.standard.object(forKey:"showAtLaunch")
```

User Defaults

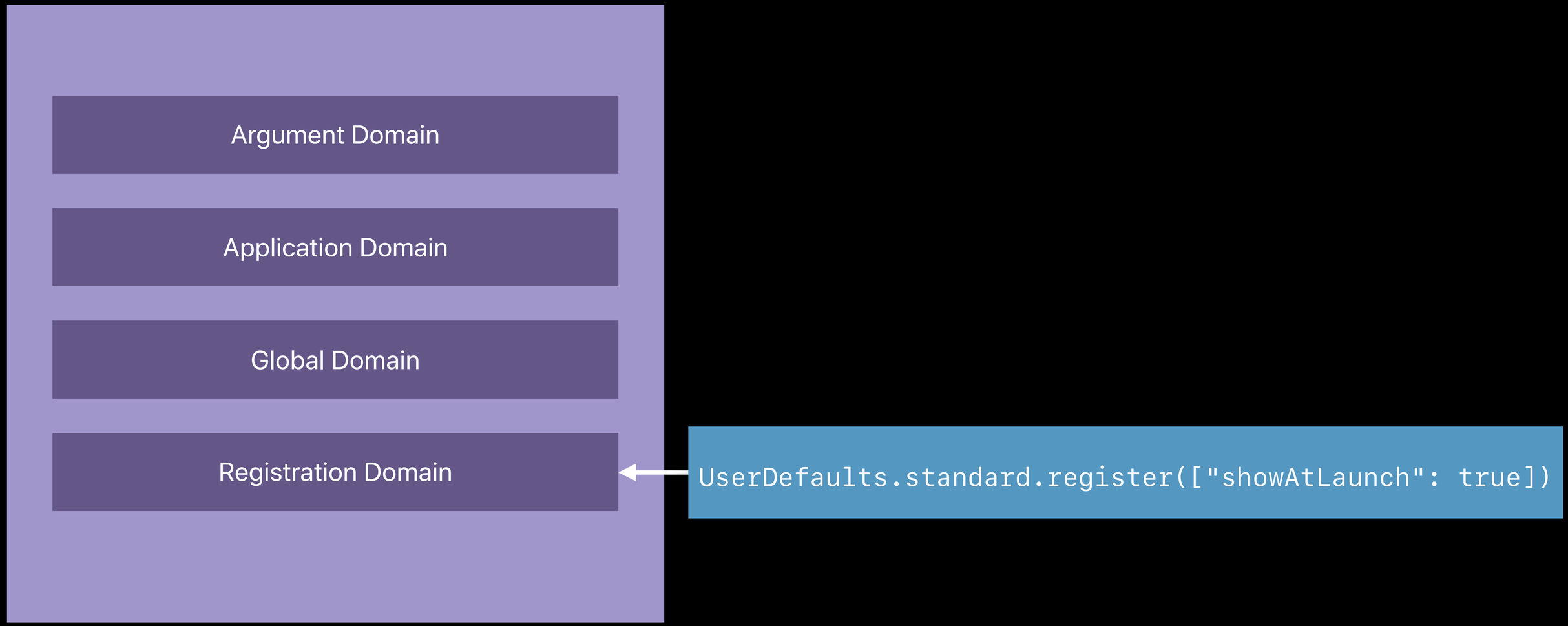


User Defaults

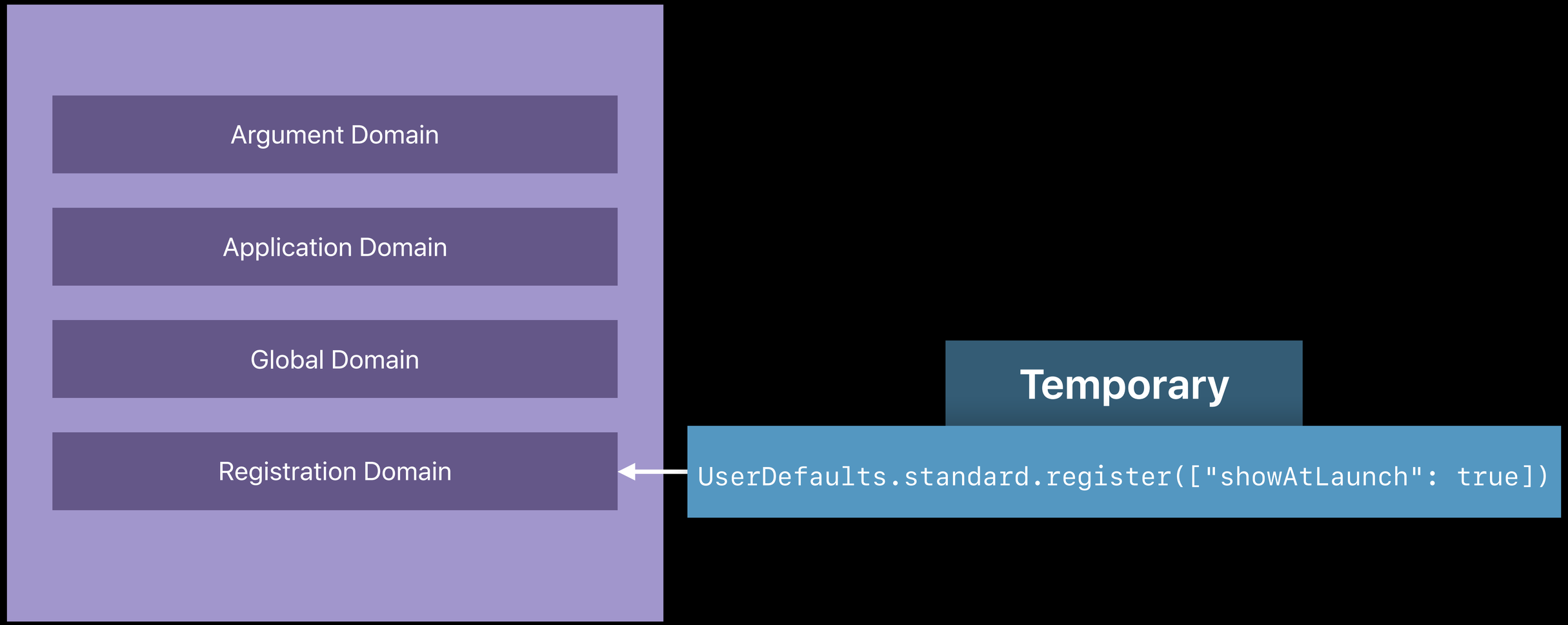



```
UserDefaults.standard.register(["showAtLaunch": true])
```

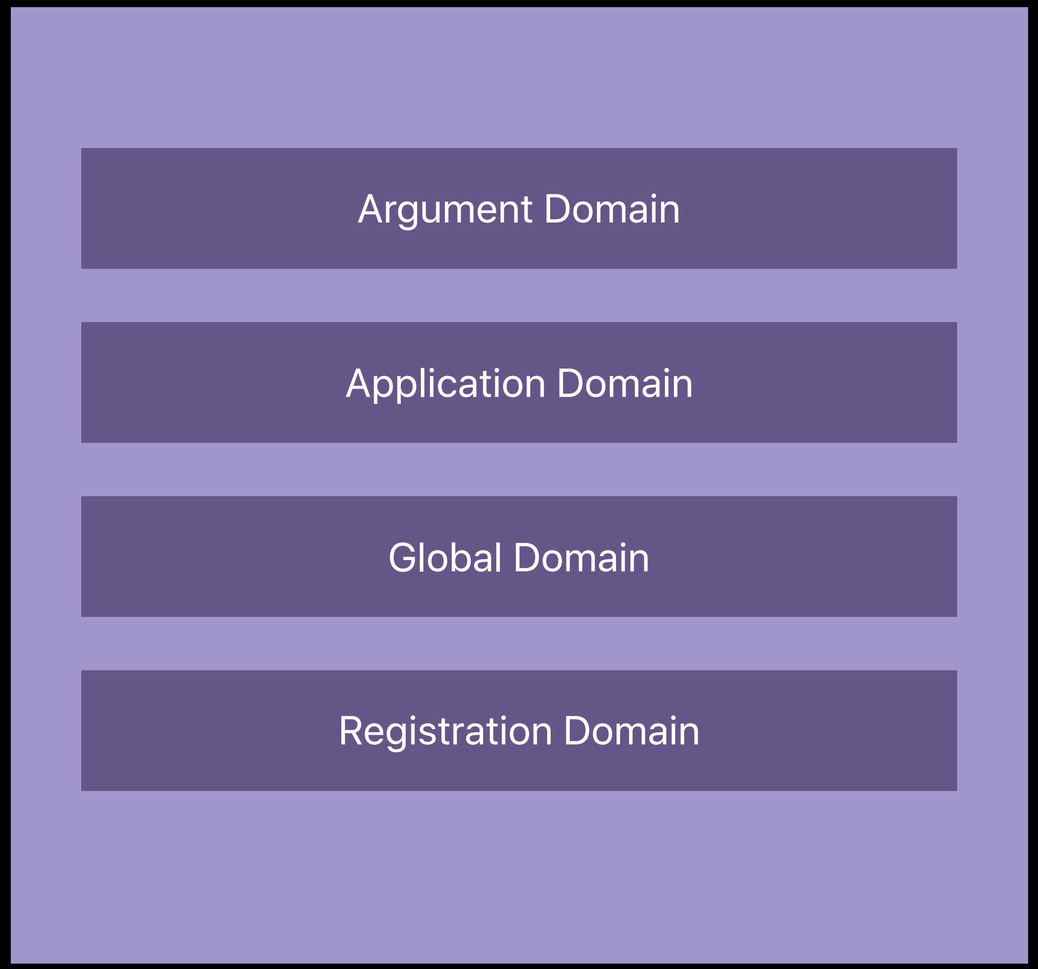
User Defaults



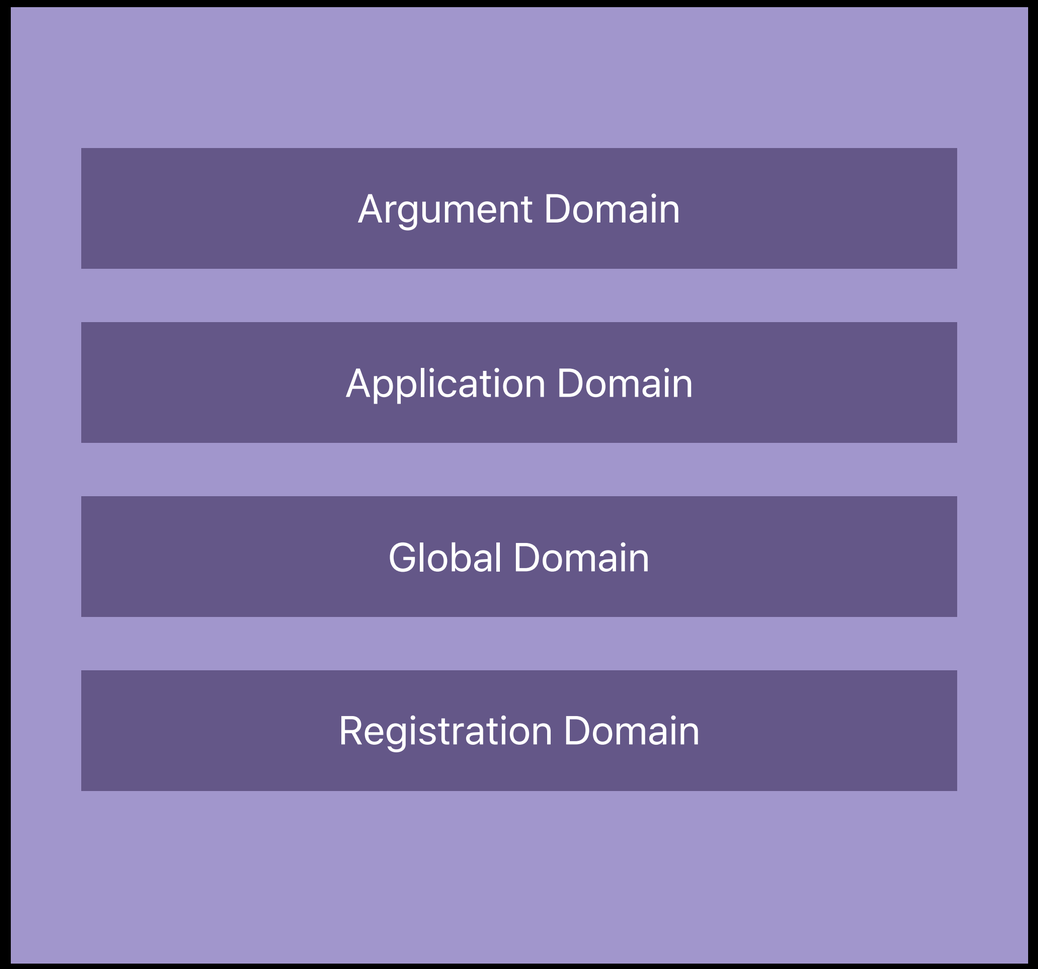
User Defaults



User Defaults

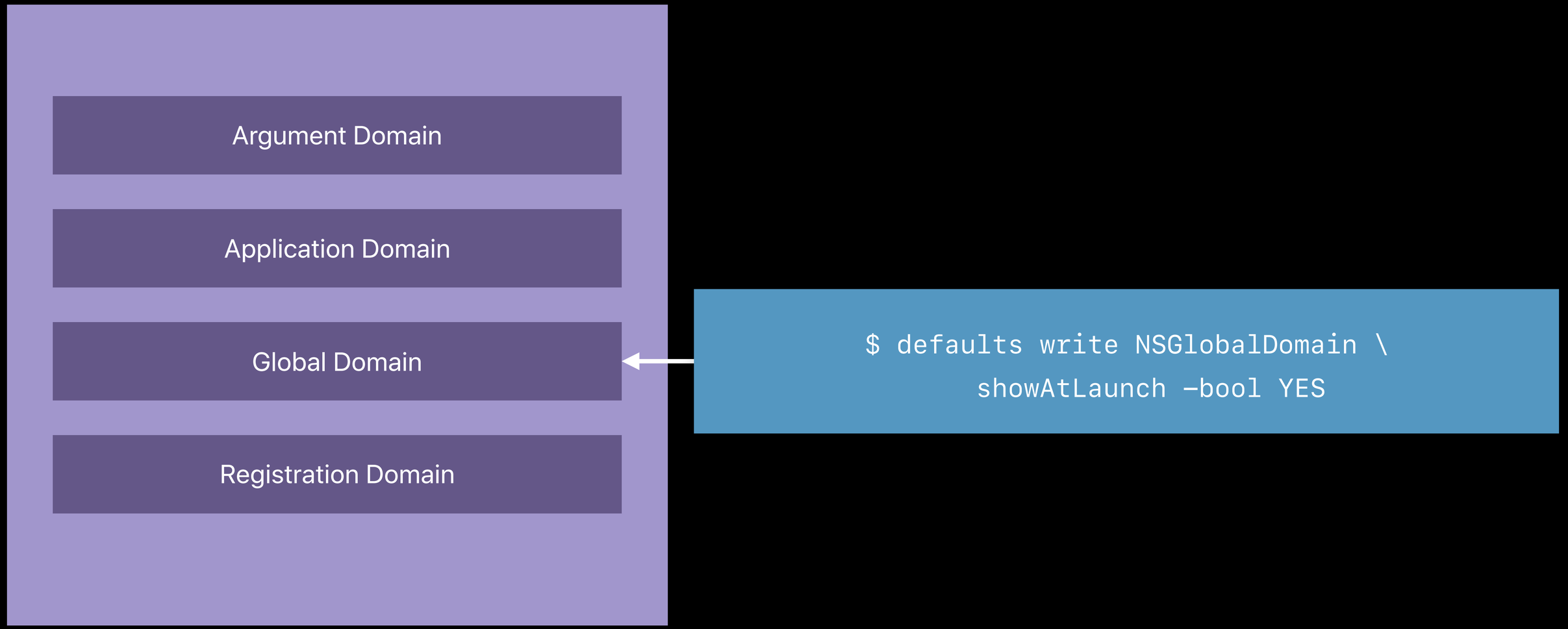


User Defaults

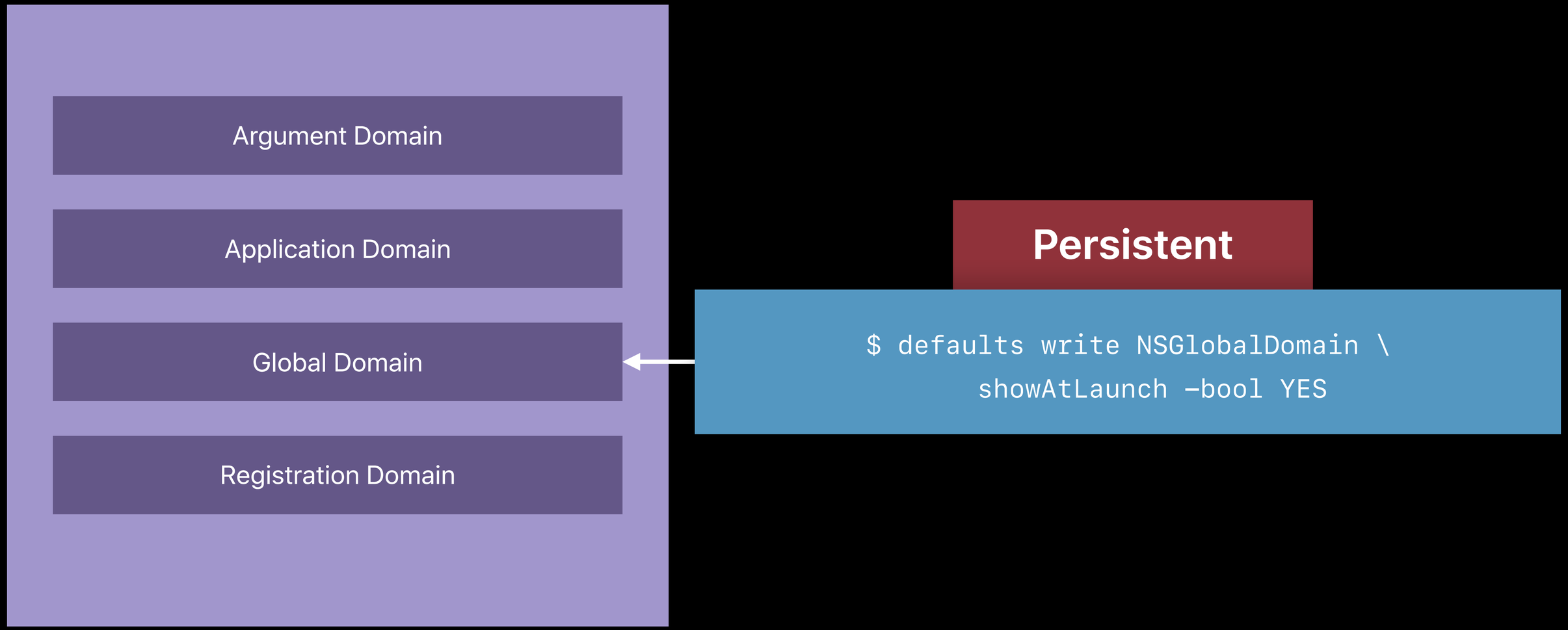



```
$ defaults write NSGlobalDomain \
    showAtLaunch -bool YES
```

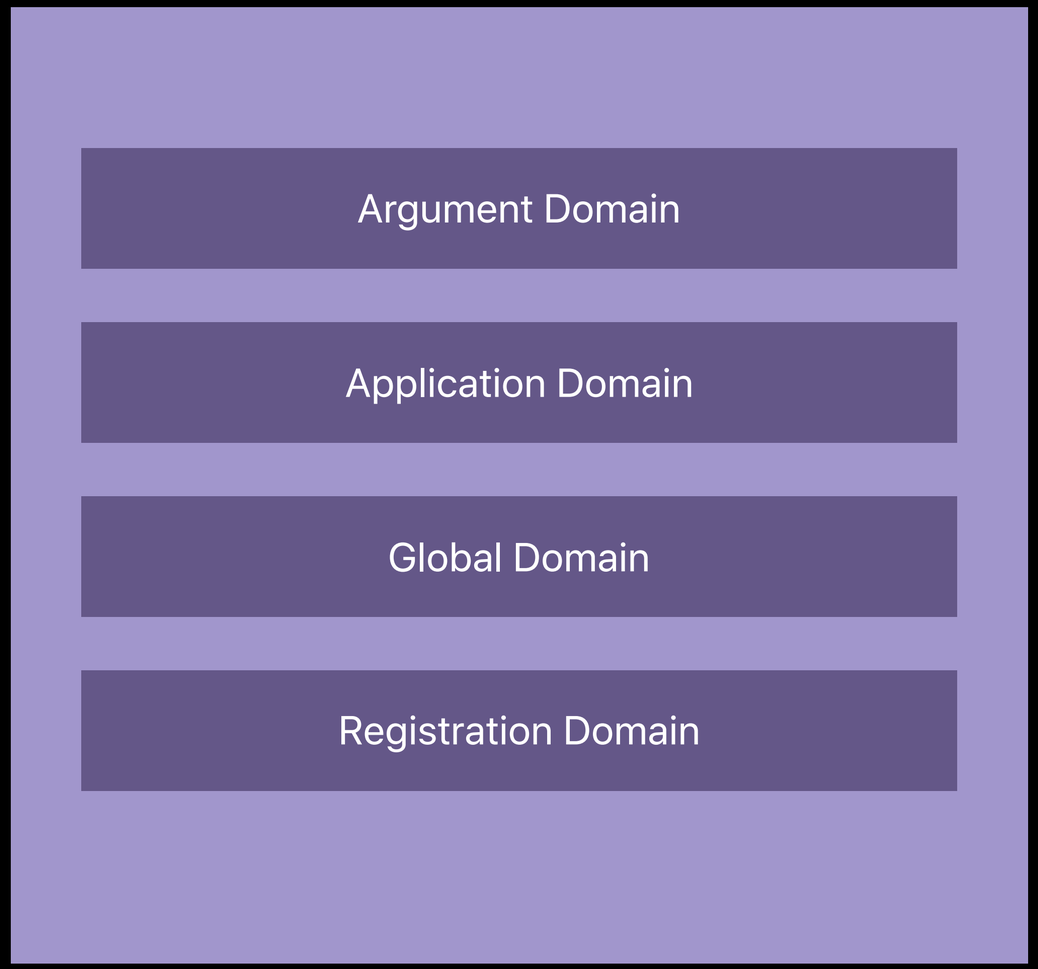
User Defaults



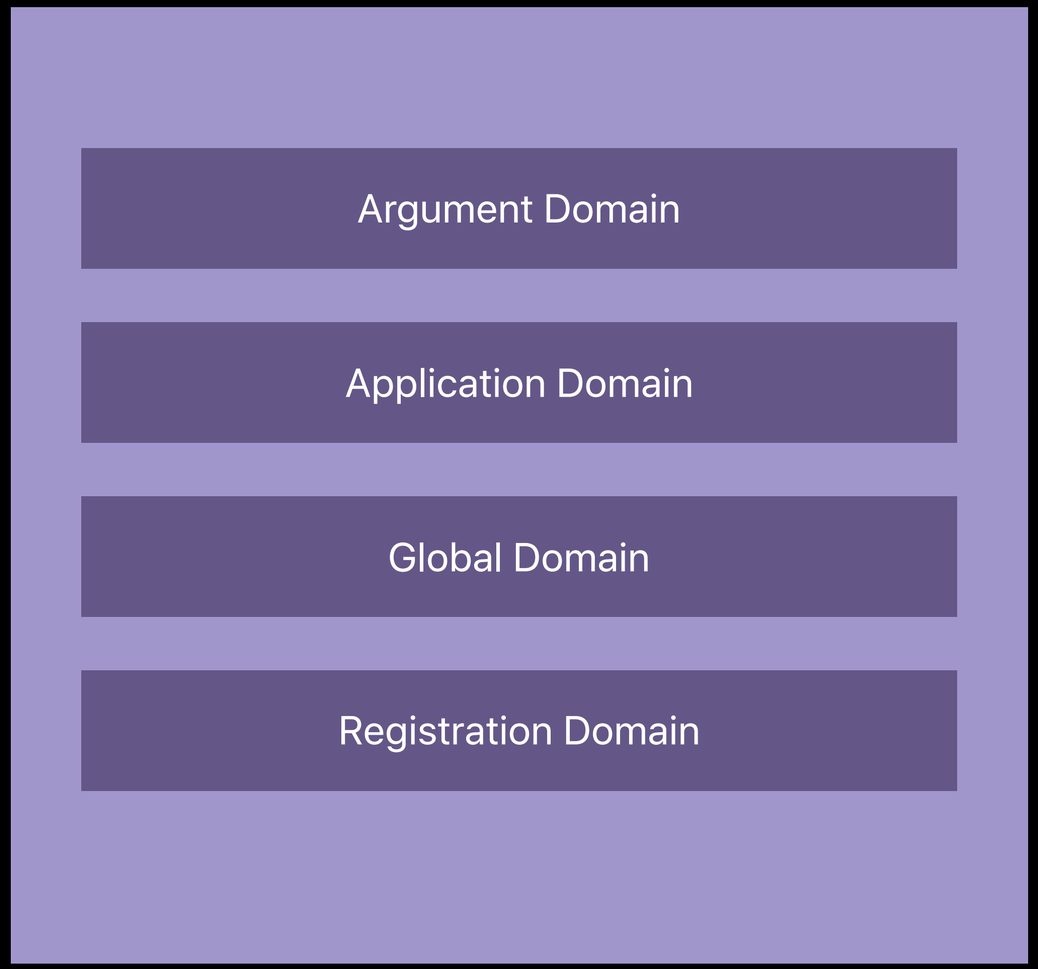
User Defaults



User Defaults

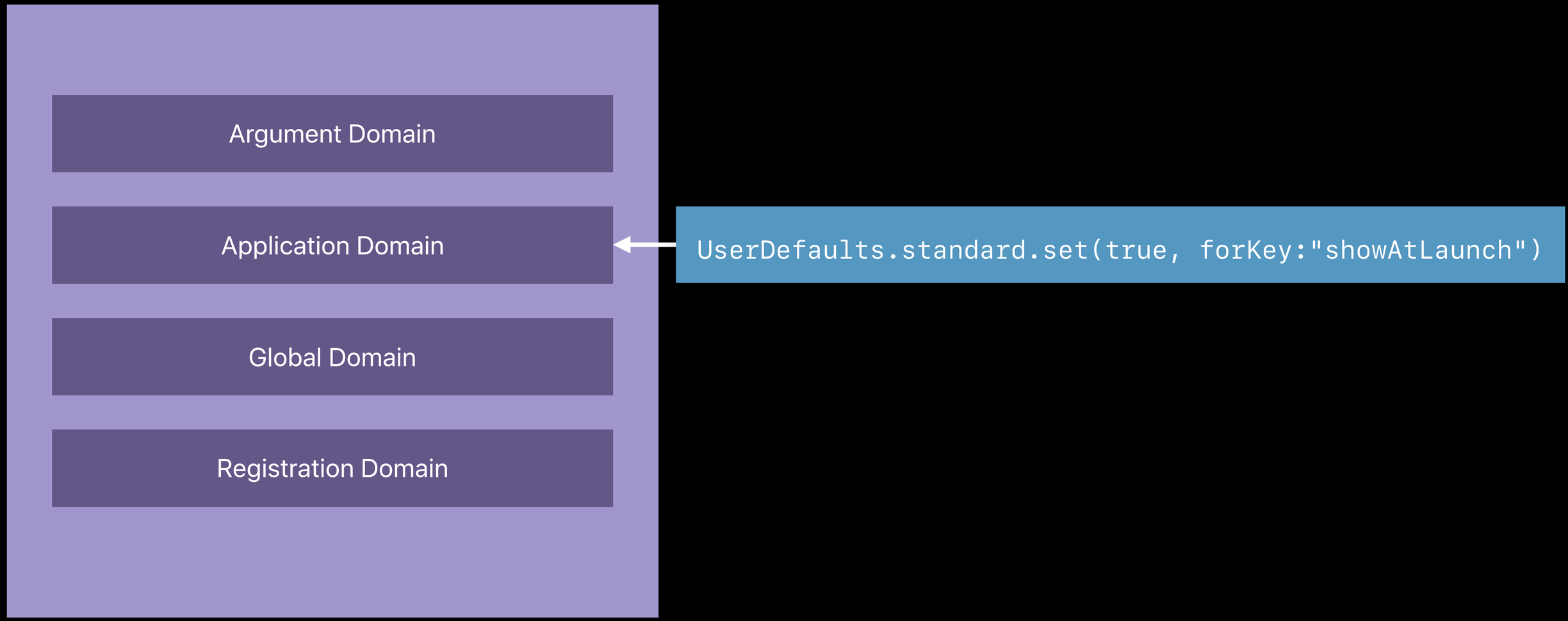


User Defaults

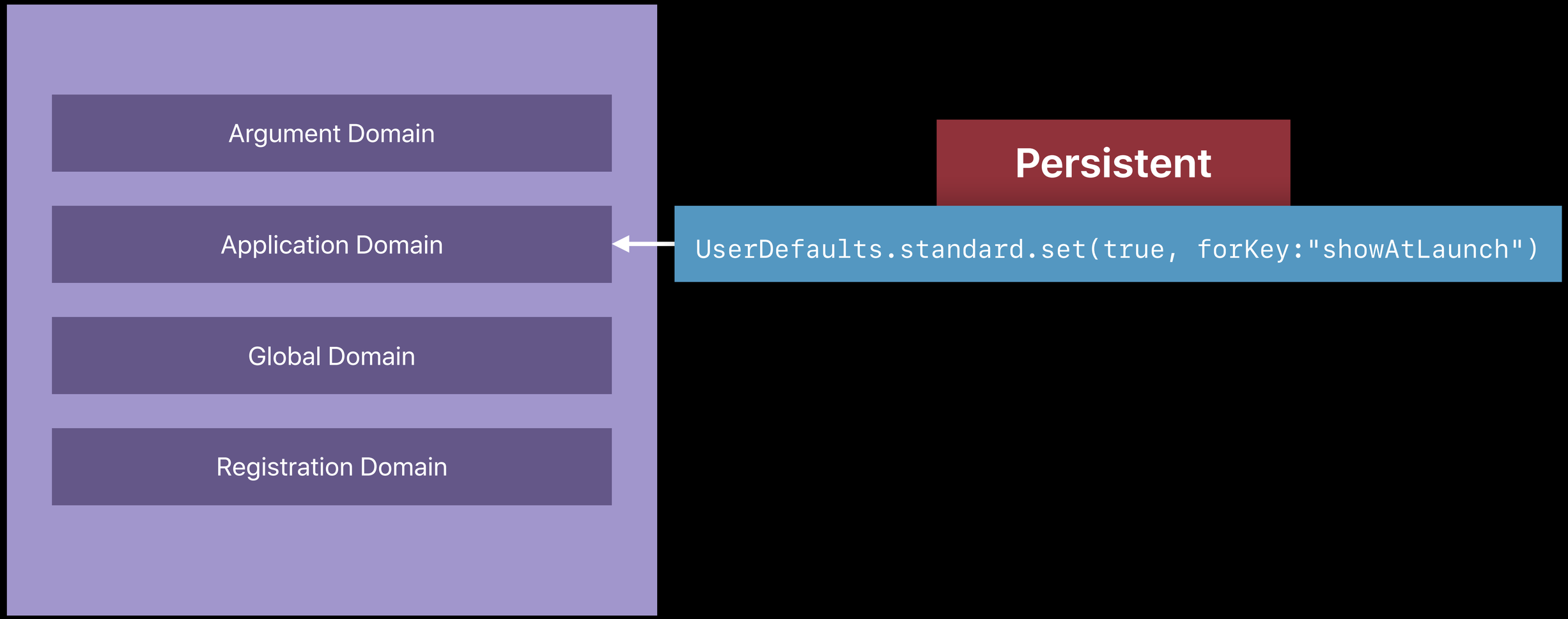



```
UserDefaults.standard.set(true, forKey:"showAtLaunch")
```

User Defaults

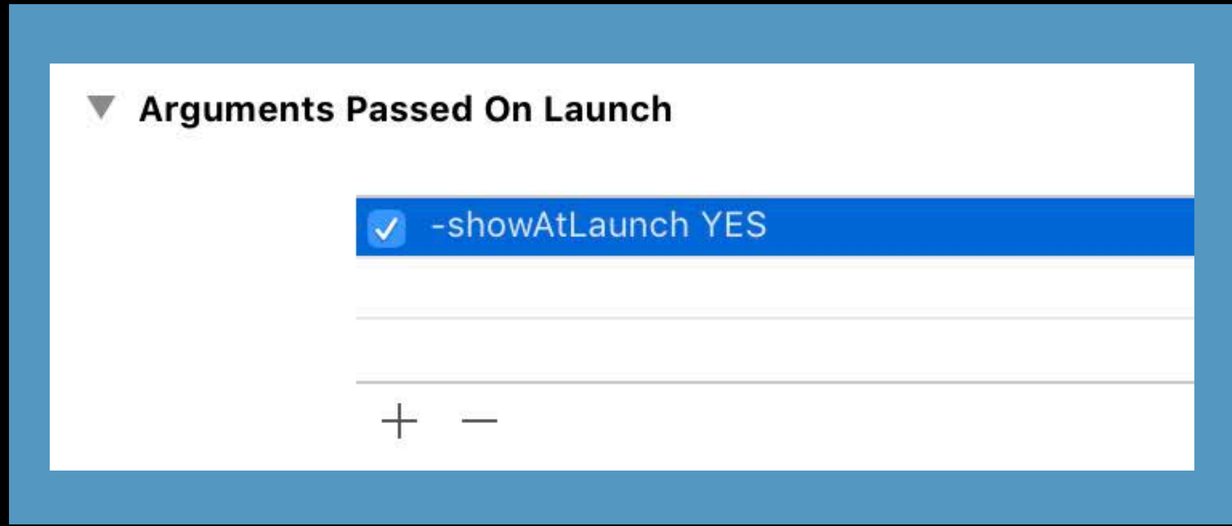


User Defaults



User Defaults

- Argument Domain
- Application Domain
- Global Domain
- Registration Domain



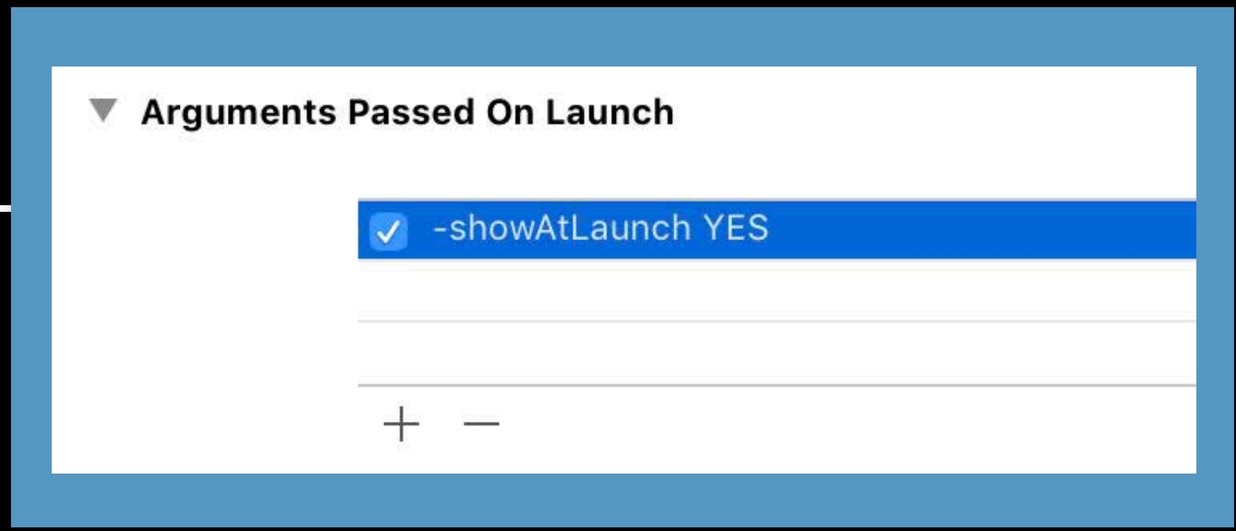
User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain



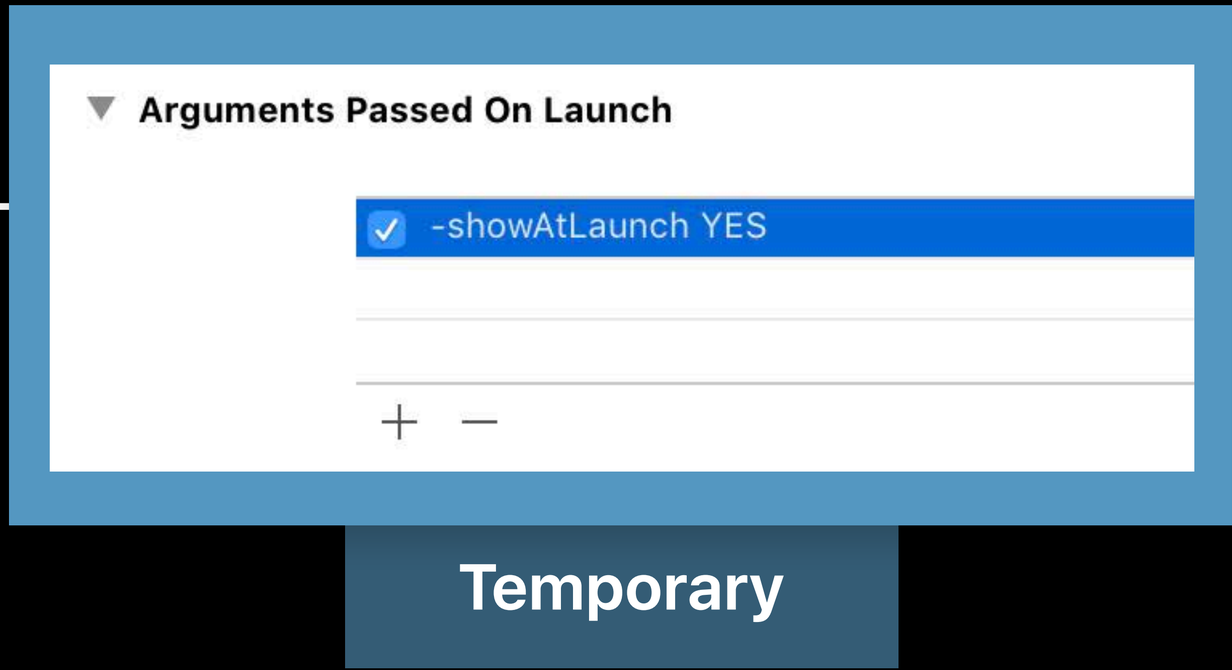
User Defaults

Argument Domain

Application Domain

Global Domain

Registration Domain



Temporary

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: …)
        }
    }
}
```

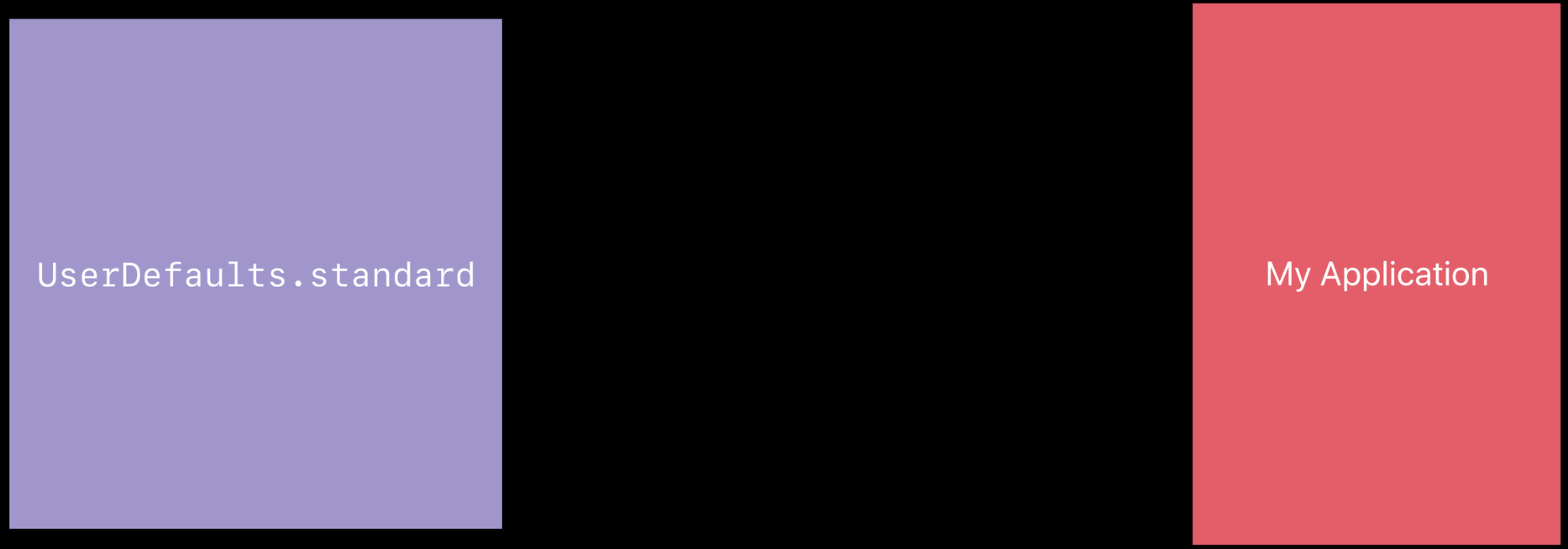
```
class MyUserDefaultsObserver: NSObject {
    private var ctx = 0
    func startObserving() {
       UserDefaults.standard.addObserver(self, forKeyPath: "showAtLaunch", options: [],
                                                    context: &ctx)
    }
    func stopObserving {
        UserDefaults.standard.removeObserver(self, 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: …)
        }
    }
}
```

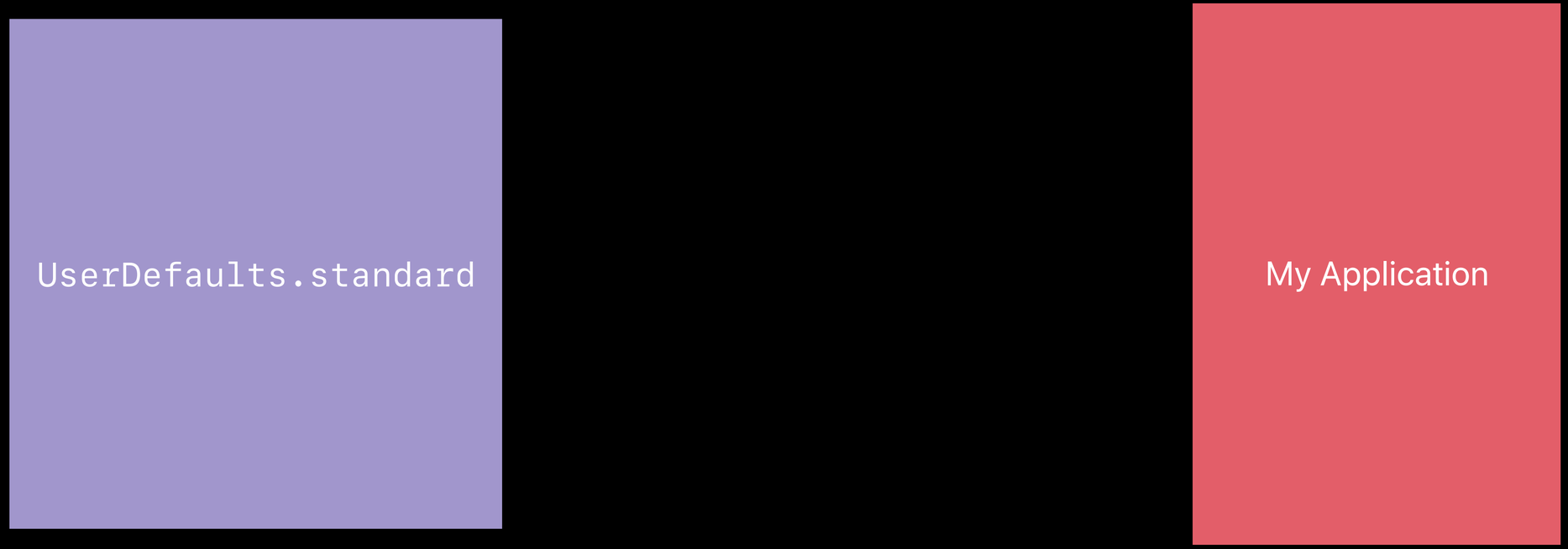
User Defaults

User Defaults



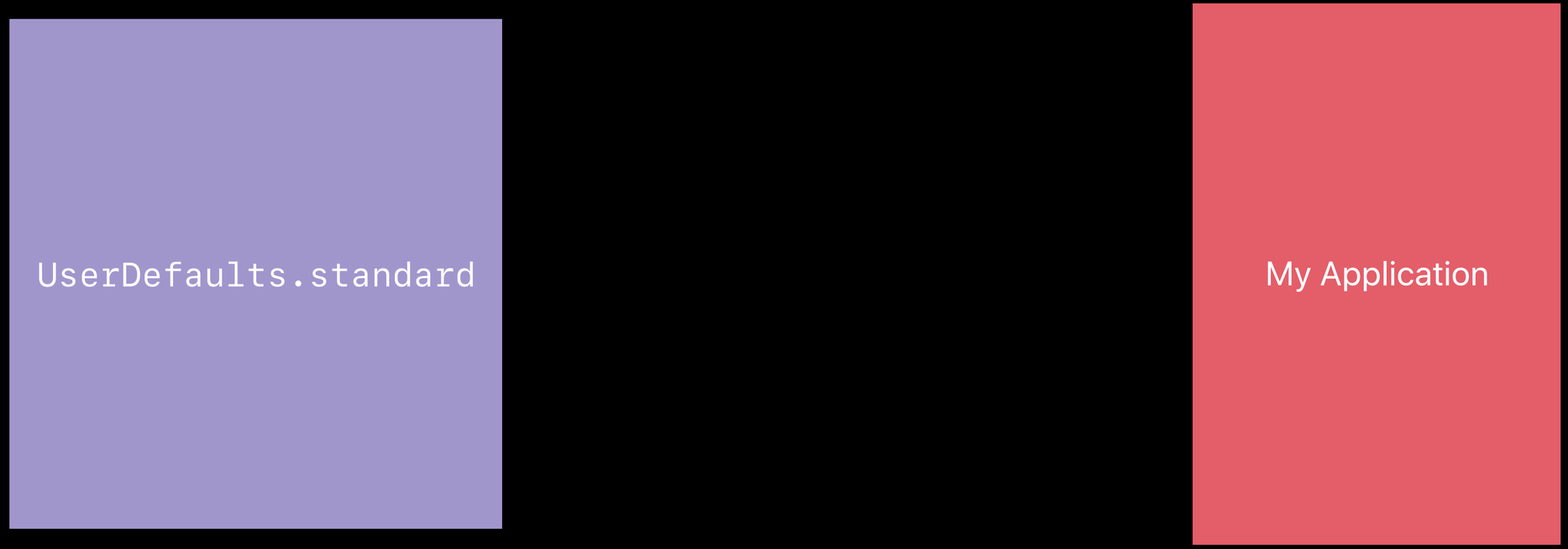
User Defaults

User Defaults

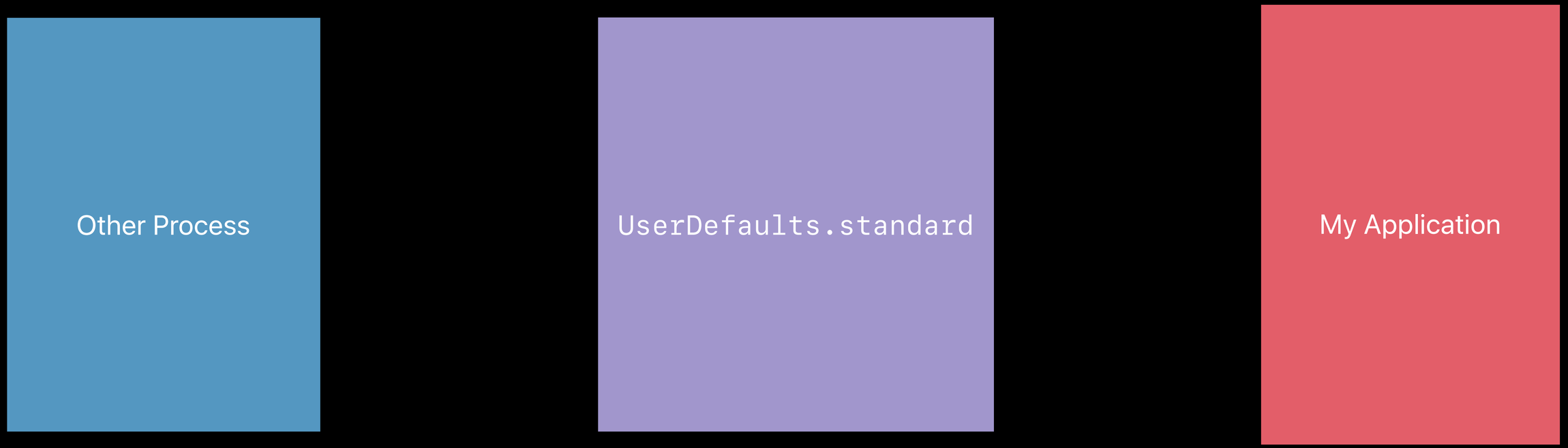


User Defaults

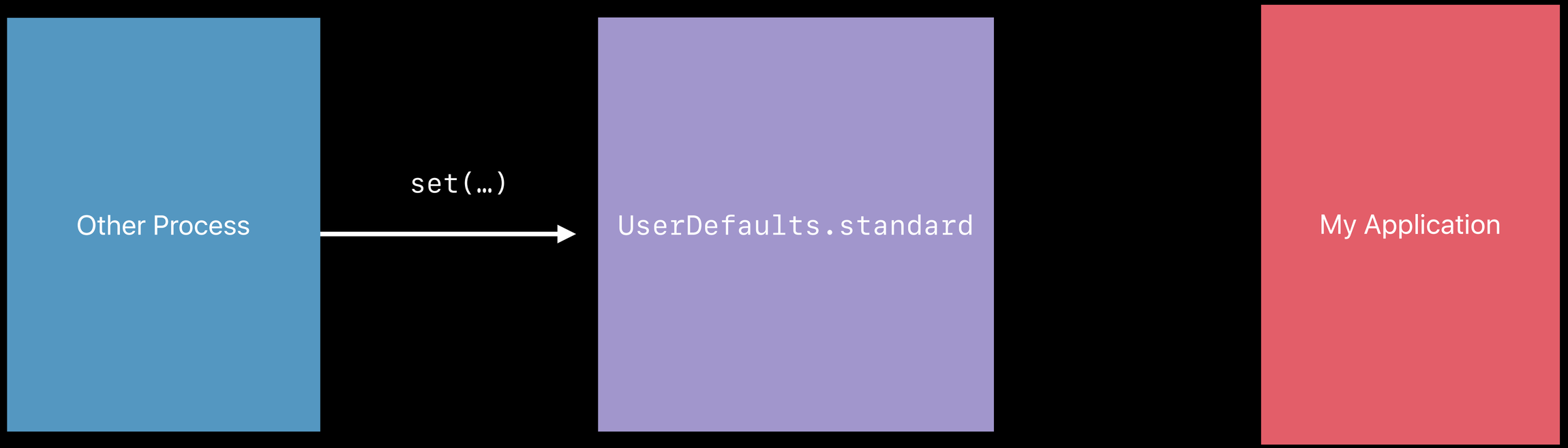
User Defaults



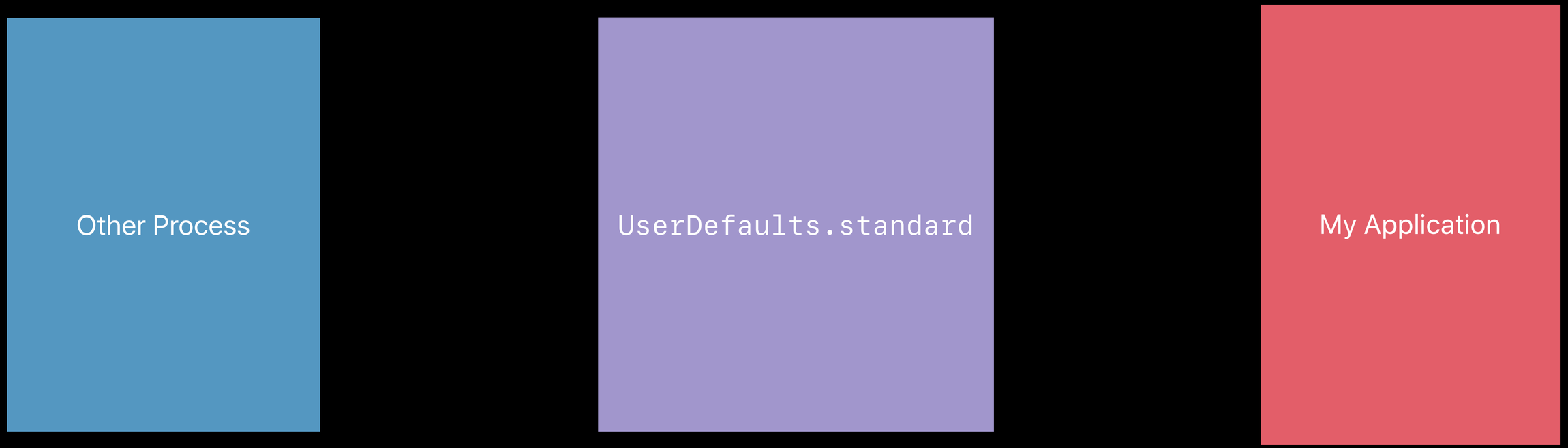
User Defaults



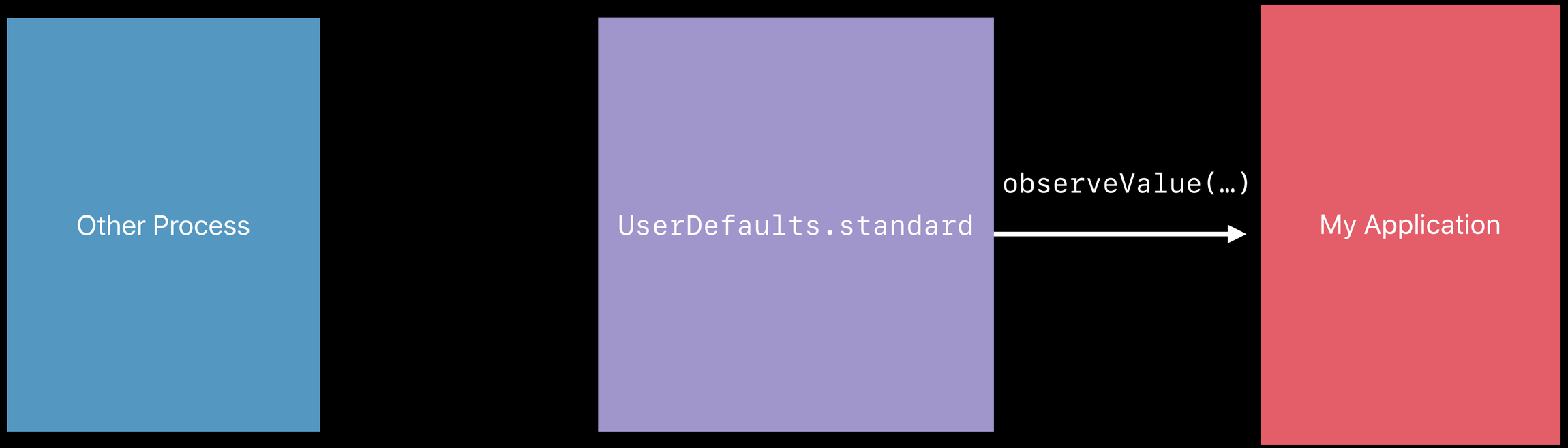
User Defaults



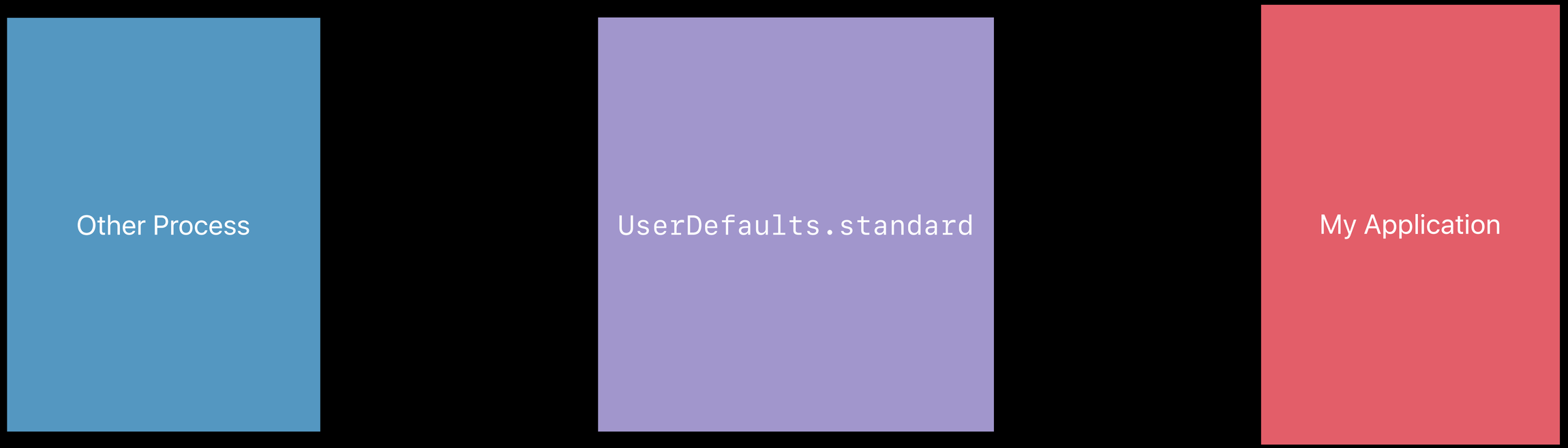
User Defaults



User Defaults



User Defaults



NEW

```
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()
}
```

NEW

```
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()
}
```

NEW

```
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()
}
```

NEW

```
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()
}
```

NEW

```
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/+WxgD6AAAPSbwD4pTJg0QK0CZA2QK0CZCxkIEyAwRLb/j/uqvrPd68f+T8v+j/5
d+3v89/P/p/pf/rn+PA+kH0HY8NoBsAMCKIu4JCcHNa4XAPztvtvoaHatuD3eKQzN1NdNZzsb9+8P5bDV8v/ZuNVFSCg1
1bGixouwcP/5bGApQAABANGAJFAghgdjEIFsoCcNjAId/9vfvv8LfvxxOs64Bse/v7j4b+Kff7tNA8ywhZqZoSVUqr/LG
aheO9OT5w8u5R01m3kv0T8b7HvfCN17LwzsiPfg/tdl64//N2Rc9mAABorJxTzCoPtzL6o8rCMYow0cdQeLDjb5Dj3tog
Ut1rupUyJr6vfG8fjq9mm3B3RhFuDIQp5CrSIY7AkMPnCGnypF9Enh7PCrv1KRogFGFLAALR63FtFhIKITpWiFXLk4mmJ
GyhGPgSVCYRoDJKRwkAmxOQGyqPnf+BnQhFiieDxZKPVIazbEZPEifeOwkKHl0h8DvhGQ9CIJ+X/cRP256DJ+A9oQqbIh
u9aQyEAnLyZPY8TJ846gT6Xygnr5RCBPIrWTvZYnpdQTz2KtaaQl2yFaBmHv8mddTm0RjgUs0AAA4P/5bGAQQAAA9hGAS
EA8EBCBsQEoLhAJ2//tet7/6x8gO0sSGae3rM3VUoc5RwmpFzXBc5yL2UUFRSJKWVldlXfNV4qhPCBEkvBDGoUIiCAaEm
Dvyms7n+/569VIjJvJcfPgAAARRRIBIkvWKsRQrFSxa0P2/K6fre78DwfPWtooAA7/+WxgCsAAAPQRgGhgMYYCCbCoQI4
RCAWw//ty+XIl4yBxb16igW28pmWXEywjHf/riV3BvCOEAAEzTI0d5phGYo2+v+HzxgAAAKvV+L6nwtLw4soBwP/5bGAP
oAAA9BGAZCAJJgQhAZCAQhoIhAjiAJoFMzl8P2c62HfeE/xm+IPeH4MPcZMyfeF/M77l8ZnfeHIZ33JQnCW7J6muscXBt
qrbRpag2fQf4Nn0H+BHLYJ1nVHVOPdW253mWvMfBgAC11njl1W3k6vofG8vm5SgAAHA"

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

```
let data64 = "//lsYAFgAADQQAf/+WxgD6AAAPSbwD4pTJg0QK0CZA2QK0CZCxkIEyAwRLb/j/uqvrPd68f+T8v+j/5
d+3v89/P/p/pf/rn+PA+kH0HY8NoBsAMCKIu4JCcHNa4XAPztvtvoaHatuD3eKQzN1NdNZzsb9+8P5bDV8v/ZuNVFSCg1
1bGixouwcP/5bGApQAABANGAJFAghgdjEIFsoCcNjAId/9vfvv8LfvxxOs64Bse/v7j4b+Kff7tNA8ywhZqZoSVUqr/LG
aheO9OT5w8u5R01m3kv0T8b7HvfCN17LwzsiPfg/tdl64//N2Rc9mAABorJxTzCoPtzL6o8rCMYow0cdQeLDjb5Dj3tog
Ut1rupUyJr6vfG8fjq9mm3B3RhFuDIQp5CrSIY7AkMPnCGnypF9Enh7PCrv1KRogFGFLAALR63FtFhIKITpWiFXLk4mmJ
GyhGPgSVCYRoDJKRwkAmxOQGyqPnf+BnQhFiieDxZKPVIazbEZPEifeOwkKHl0h8DvhGQ9CIJ+X/cRP256DJ+A9oQqbIh
u9aQyEAnLyZPY8TJ846gT6Xygnr5RCBPIrWTvZYnpdQTz2KtaaQl2yFaBmHv8mddTm0RjgUs0AAA4P/5bGAQQAAA9hGAS
EA8EBCBsQEoLhAJ2//tet7/6x8gO0sSGae3rM3VUoc5RwmpFzXBc5yL2UUFRSJKWVldlXfNV4qhPCBEkvBDGoUIiCAaEm
Dvyms7n+/569VIjJvJcfPgAAARRRIBIkvWKsRQrFSxa0P2/K6fre78DwfPWtooAA7/+WxgCsAAAPQRgGhgMYYCCbCoQI4
RCAWw//ty+XIl4yBxb16igW28pmWXEywjHf/riV3BvCOEAAEzTI0d5phGYo2+v+HzxgAAAKvV+L6nwtLw4soBwP/5bGAP
oAAA9BGAZCAJJgQhAZCAQhoIhAjiAJoFMzl8P2c62HfeE/xm+IPeH4MPcZMyfeF/M77l8ZnfeHIZ33JQnCW7J6muscXBt
qrbRpag2fQf4Nn0H+BHLYJ1nVHVOPdW253mWvMfBgAC11njl1W3k6vofG8vm5SgAAHA"
```

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

```
let data64 = "//lsYAFgAADQQAf/+WxgD6AAAPSbwD4pTJg0QK0CZA2QK0CZCxkIEyAwRLb/j/uqvrPd68f+T8v+j/5
d+3v89/P/p/pf/rn+PA+kH0HY8NoBsAMCKIu4JCcHNa4XAPztvtvoaHatuD3eKQzN1NdNZzsb9+8P5bDV8v/ZuNVFSCg1
1bGixouwcP/5bGApQAABANGAJFAghgdjEIFsoCcNjAId/9vfvv8LfvxxOs64Bse/v7j4b+Kff7tNA8ywhZqZoSVUqr/LG
aheO9OT5w8u5R01m3kv0T8b7HvfCN17LwzsiPfg/tdl64//N2Rc9mAABorJxTzCoPtzL6o8rCMYow0cdQeLDjb5Dj3tog
Ut1rupUyJr6vfG8fjq9mm3B3RhFuDIQp5CrSIY7AkMPnCGnypF9Enh7PCrv1KRogFGFLAALR63FtFhIKITpWiFXLk4mmJ
GyhGPgSVCYRoDJKRwkAmxOQGyqPnf+BnQhFiieDxZKPVIazbEZPEifeOwkKHl0h8DvhGQ9CIJ+X/cRP256DJ+A9oQqbIh
u9aQyEAnLyZPY8TJ846gT6Xygnr5RCBPIrWTvZYnpdQTz2KtaaQl2yFaBmHv8mddTm0RjgUs0AAA4P/5bGAQQAAA9hGAS
EA8EBCBsQEoLhAJ2//tet7/6x8gO0sSGae3rM3VUoc5RwmpFzXBc5yL2UUFRSJKWVldlXfNV4qhPCBEkvBDGoUIiCAaEm
Dvyms7n+/569VIjJvJcfPgAAARRRIBIkvWKsRQrFSxa0P2/K6fre78DwfPWtooAA7/+WxgCsAAAPQRgGhgMYYCCbCoQI4
RCAWw//ty+XIl4yBxb16igW28pmWXEywjHf/riV3BvCOEAAEzTI0d5phGYo2+v+HzxgAAAKvV+L6nwtLw4soBwP/5bGAP
oAAA9BGAZCAJJgQhAZCAQhoIhAjiAJoFMzl8P2c62HfeE/xm+IPeH4MPcZMyfeF/M77l8ZnfeHIZ33JQnCW7J6muscXBt
qrbRpag2fQf4Nn0H+BHLYJ1nVHVOPdW253mWvMfBgAC11njl1W3k6vofG8vm5SgAAHA"
```

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

```
let data64 = "//lsYAFgAADQQAf/+WxgD6AAAPSbwD4pTJg0QK0CZA2QK0CZCxkIEyAwRLb/j/uqvrPd68f+T8v+j/5
d+3v89/P/p/pf/rn+PA+kH0HY8NoBsAMCKIu4JCcHNa4XAPztvtvoaHatuD3eKQzN1NdNZzsb9+8P5bDV8v/ZuNVFSCg1
1bGixouwcP/5bGApQAABANGAJFAghgdjEIFsoCcNjAId/9vfvv8LfvxxOs64Bse/v7j4b+Kff7tNA8ywhZqZoSVUqr/LG
aheO9OT5w8u5R01m3kv0T8b7HvfCN17LwzsiPfg/tdl64//N2Rc9mAABorJxTzCoPtzL6o8rCMYow0cdQeLDjb5Dj3tog
Ut1rupUyJr6vfG8fjq9mm3B3RhFuDIQp5CrSIY7AkMPnCGnypF9Enh7PCrv1KRogFGFLAALR63FtFhIKITpWiFXLk4mmJ
GyhGPgSVCYRoDJKRwkAmxOQGyqPnf+BnQhFiieDxZKPVIazbEZPEifeOwkKHl0h8DvhGQ9CIJ+X/cRP256DJ+A9oQqbIh
u9aQyEAnLyZPY8TJ846gT6Xygnr5RCBPIrWTvZYnpdQTz2KtaaQl2yFaBmHv8mddTm0RjgUs0AAA4P/5bGAQQAAA9hGAS
EA8EBCBsQEoLhAJ2//tet7/6x8gO0sSGae3rM3VUoc5RwmpFzXBc5yL2UUFRSJKWVldlXfNV4qhPCBEkvBDGoUIiCAaEm
Dvyms7n+/569VIjJvJcfPgAAARRRIBIkvWKsRQrFSxa0P2/K6fre78DwfPWtooAA7/+WxgCsAAAPQRgGhgMYYCCbCoQI4
RCAWw//ty+XIl4yBxb16igW28pmWXEywjHf/riV3BvCOEAAEzTI0d5phGYo2+v+HzxgAAAKvV+L6nwtLw4soBwP/5bGAP
oAAA9BGAZCAJJgQhAZCAQhoIhAjiAJoFMzl8P2c62HfeE/xm+IPeH4MPcZMyfeF/M77l8ZnfeHIZ33JQnCW7J6muscXBt
qrbRpag2fQf4Nn0H+BHLYJ1nVHVOPdW253mWvMfBgAC11njl1W3k6vofG8vm5SgAAHA"

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

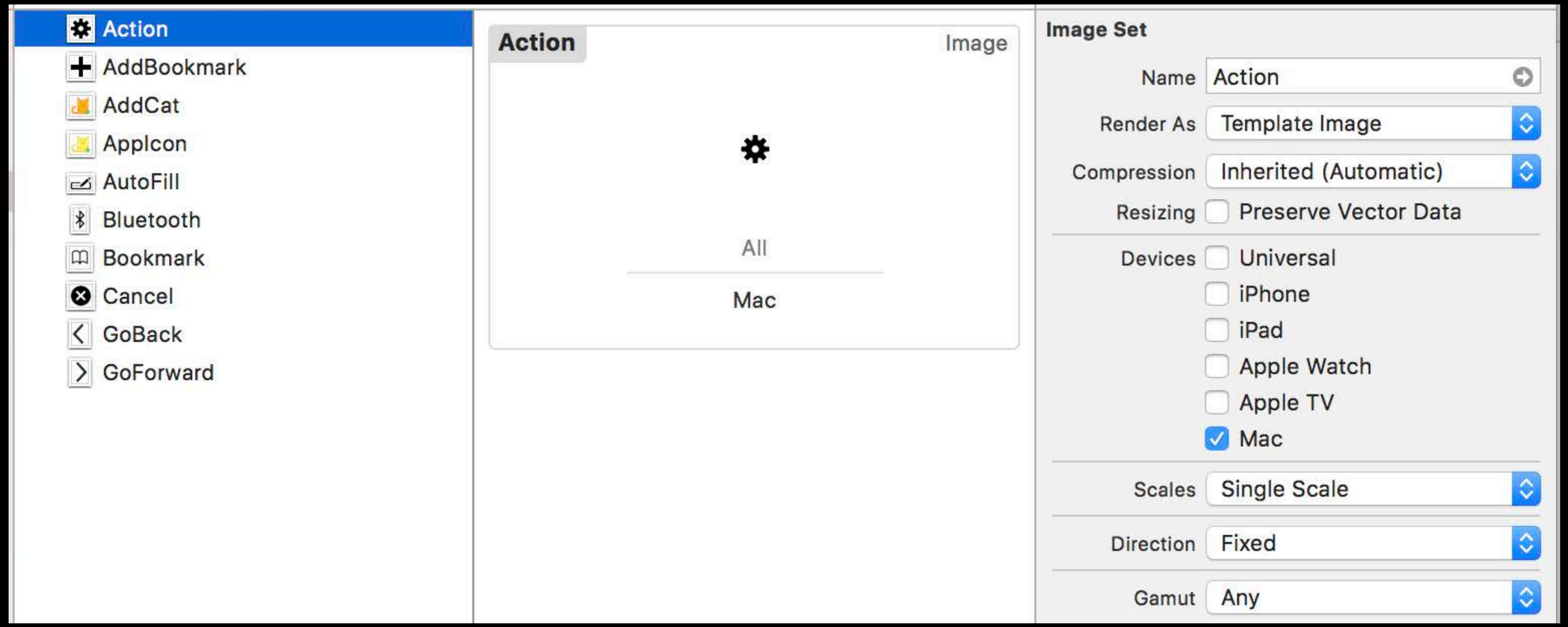
```
soundData.base64EncodedString()
```

```
"//lsYAFgAADQQAf/+WxgD6AAAPSbwD4pTJg0QK0CZA2QK0CZCxkIEyAwRLb/j/uqvrPd68f+T8v+j/5d+3v89/P/p/pf
/rn+PA+kH0HY8NoBsAMCKIu4JCcHNa4XAPztvtvoaHatuD3eKQzN1NdNZzsb9+8P5bDV8v/ZuNVFSCg11bGixouwcP/5b
GApQAABANGAJFAghgdjEIFsoCcNjAId/9vfvv8LfvxxOs64Bse/v7j4b+Kff7tNA8ywhZqZoSVUqr/LGaheO9OT5w8u5R
01m3kv0T8b7HvfCN17LwzsiPfg/tdl64//N2Rc9mAABorJxTzCoPtzL6o8rCMYow0cdQeLDjb5Dj3togUt1rupUyJr6vf
G8fjq9mm3B3RhFuDIQp5CrSIY7AkMPnCGnypF9Enh7PCrv1KRogFGFLAALR63FtFhIKITpWiFXLk4mmJGyhGPgSVCYRoD
JKRwkAmxOQGyqPnf+BnQhFiieDxZKPVIazbEZPEifeOwkKHl0h8DvhGQ9CIJ+X/cRP256DJ+A9oQqbIhu9aQyEAnLyZPY
8TJ846gT6Xygnr5RCBPIrWTvZYnpdQTz2KtaaQl2yFaBmHv8mddTm0RjgUs0AAA4P/5bGAQQAAA9hGASEA8EBCBsQEoLh
AJ2//tet7/6x8gO0sSGae3rM3VUoc5RwmpFzXBc5yL2UUFRSJKWVldlXfNV4qhPCBEkvBDGoUIiCAaEmDvyms7n+/569V
IjJvJcfPgAAARRRIBIkvWKsRQrFSxa0P2/K6fre78DwfPWtooAA7/+WxgCsAAAPQRgGhgMYYCCbCoQI4RCAWw//ty+XIl
4yBxb16igW28pmWXEywjHf/riV3BvCOEAAEzTI0d5phGYo2+v+HzxgAAAKvV+L6nwtLw4soBwP/5bGAPoAAA9BGAZCAJJ
gQhAZCAQhoIhAjiAJoFMzl8P2c62HfeE/xm+IPeH4MPcZMyfeF/M77l8ZnfeHIZ33JQnCW7J6muscXBtqrbRpag2fQf4N
n0H+BHLYJ1nVHVOPdW253mWvMfBgAC11njl1W3k6vofG8vm5SgAAHA"
```

2X

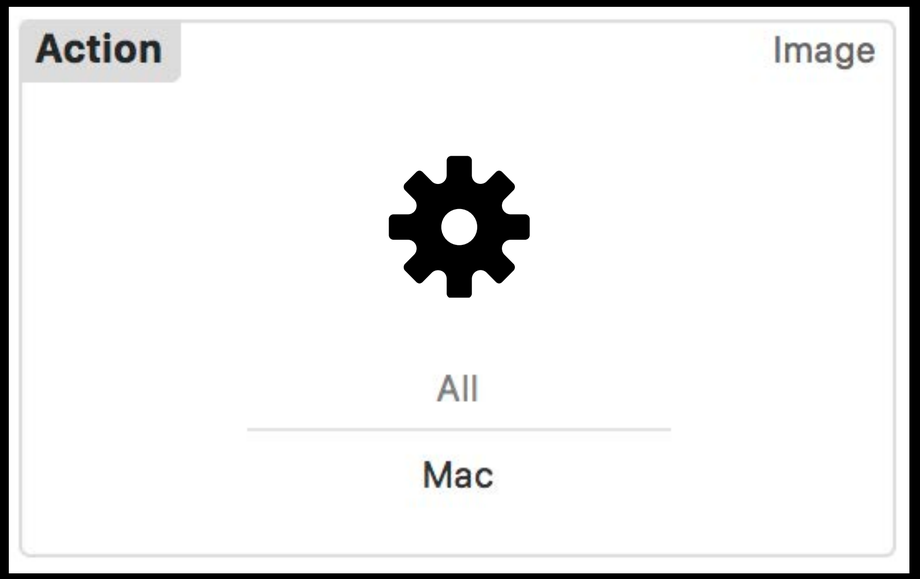
Asset Catalogs

Asset Catalogs



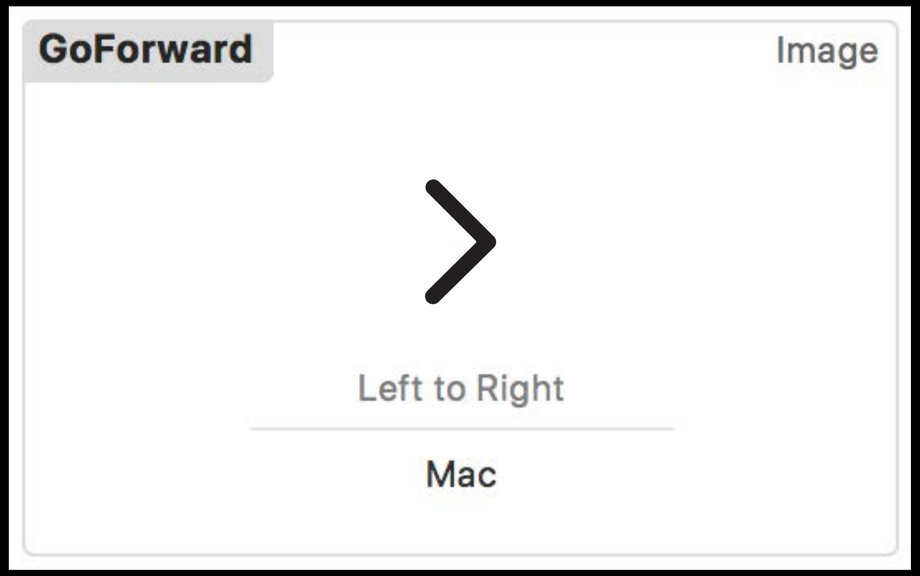
Asset Catalogs

Layout direction



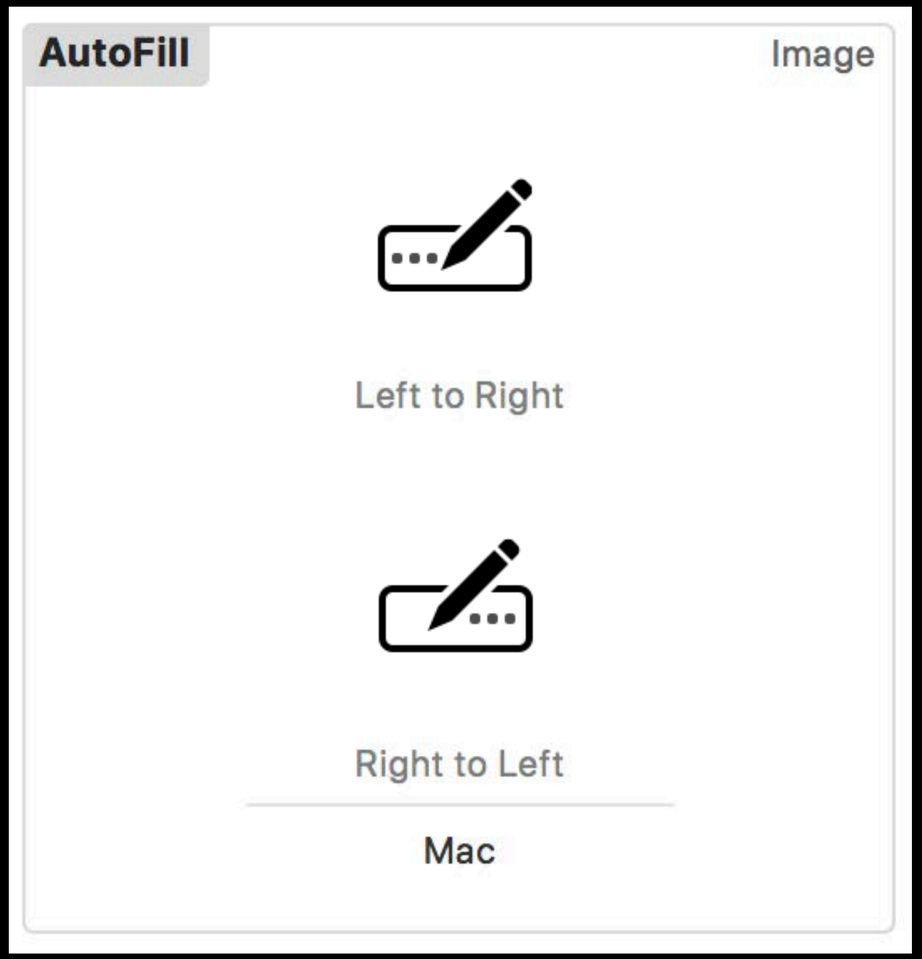
Asset Catalogs

Layout direction



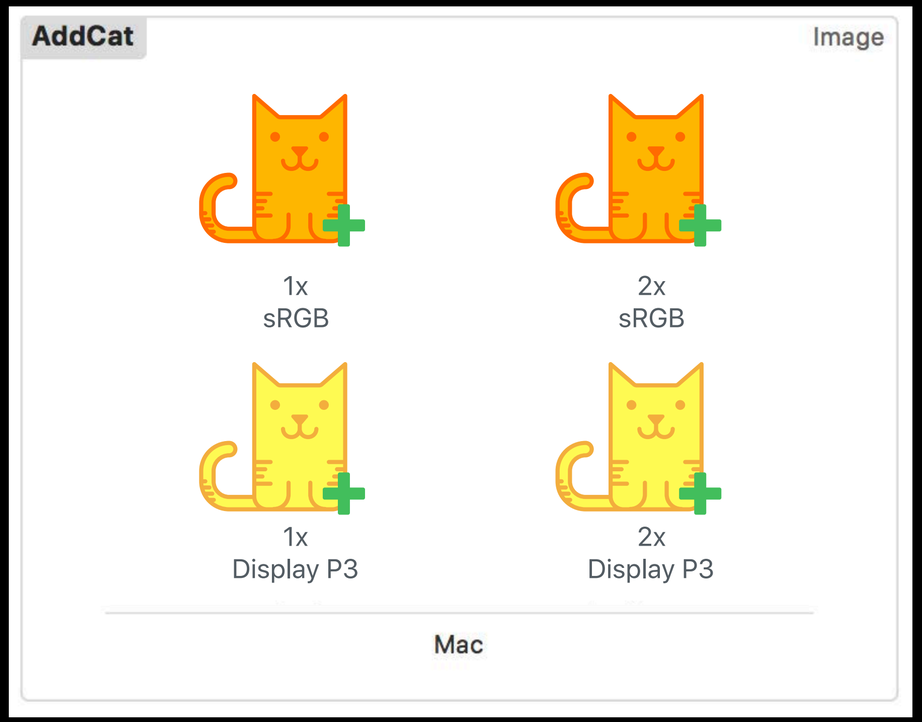
Asset Catalogs

Layout direction



Asset Catalogs

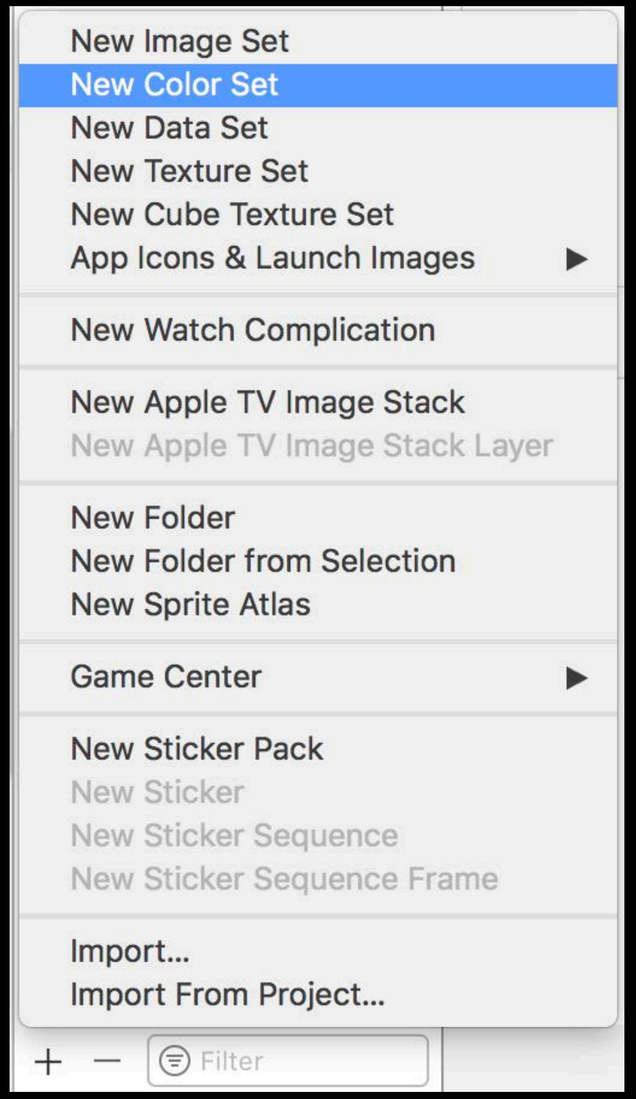
Display gamut



Asset Catalogs

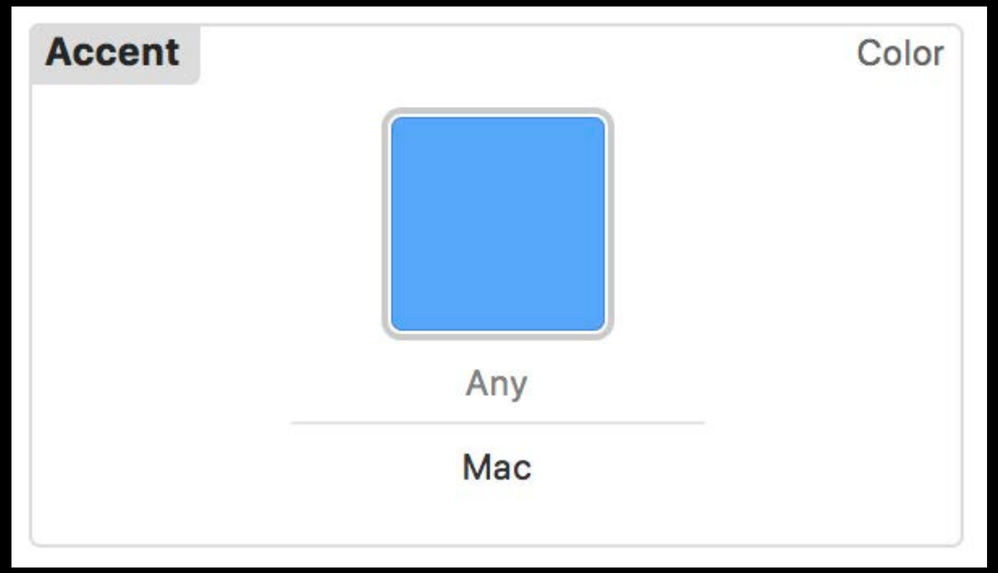
Colors

NEW



Asset Catalogs

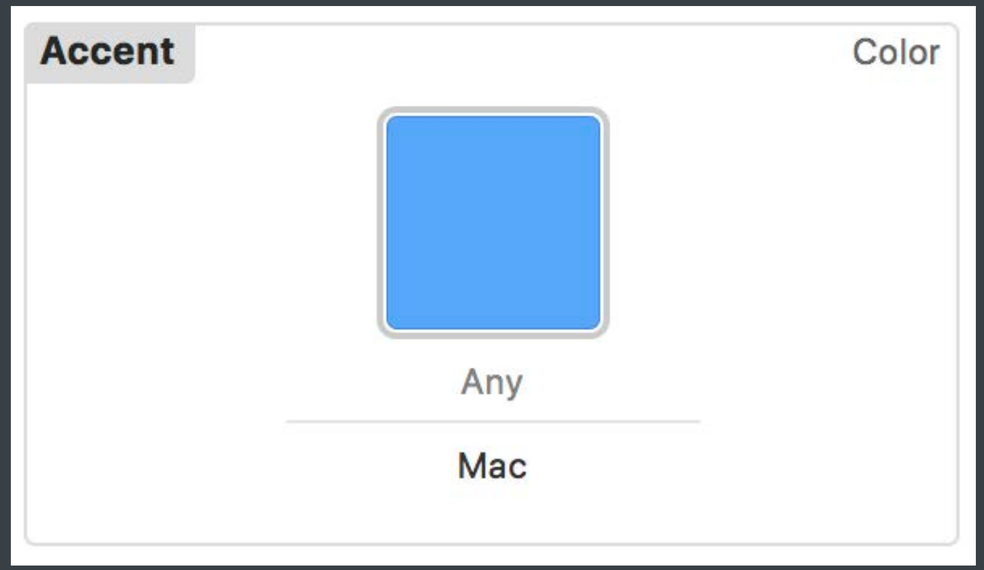
Colors



Asset Catalogs

Colors

```swift
//Using colors from Asset Catalogs

//You can add your color name to the extensible NSColor.Name enum
extension NSColor.Name {
    static let accent = NSColor.Name("Accent")
}
//Then you can create your color using that name

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



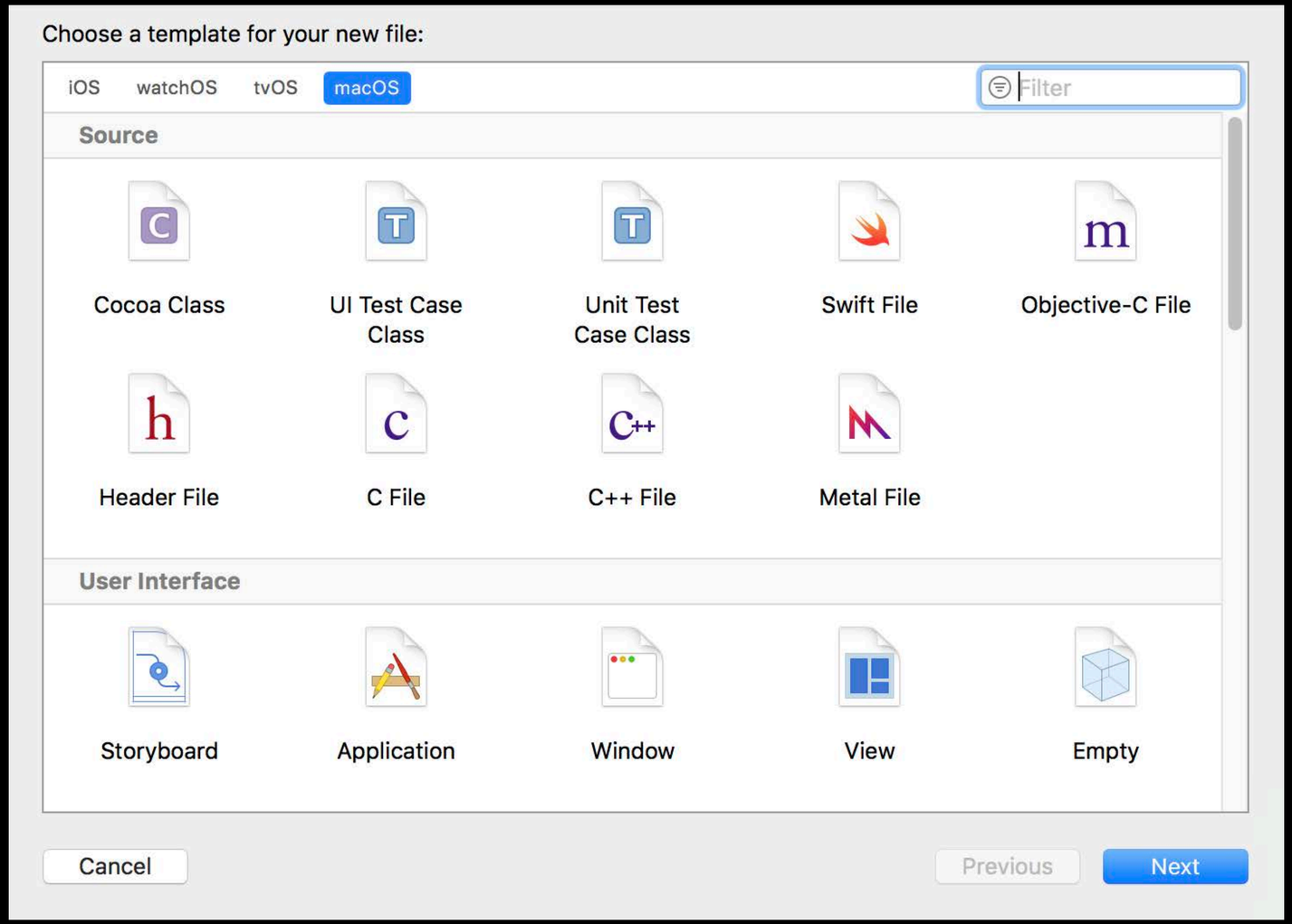
1

Unit Tests

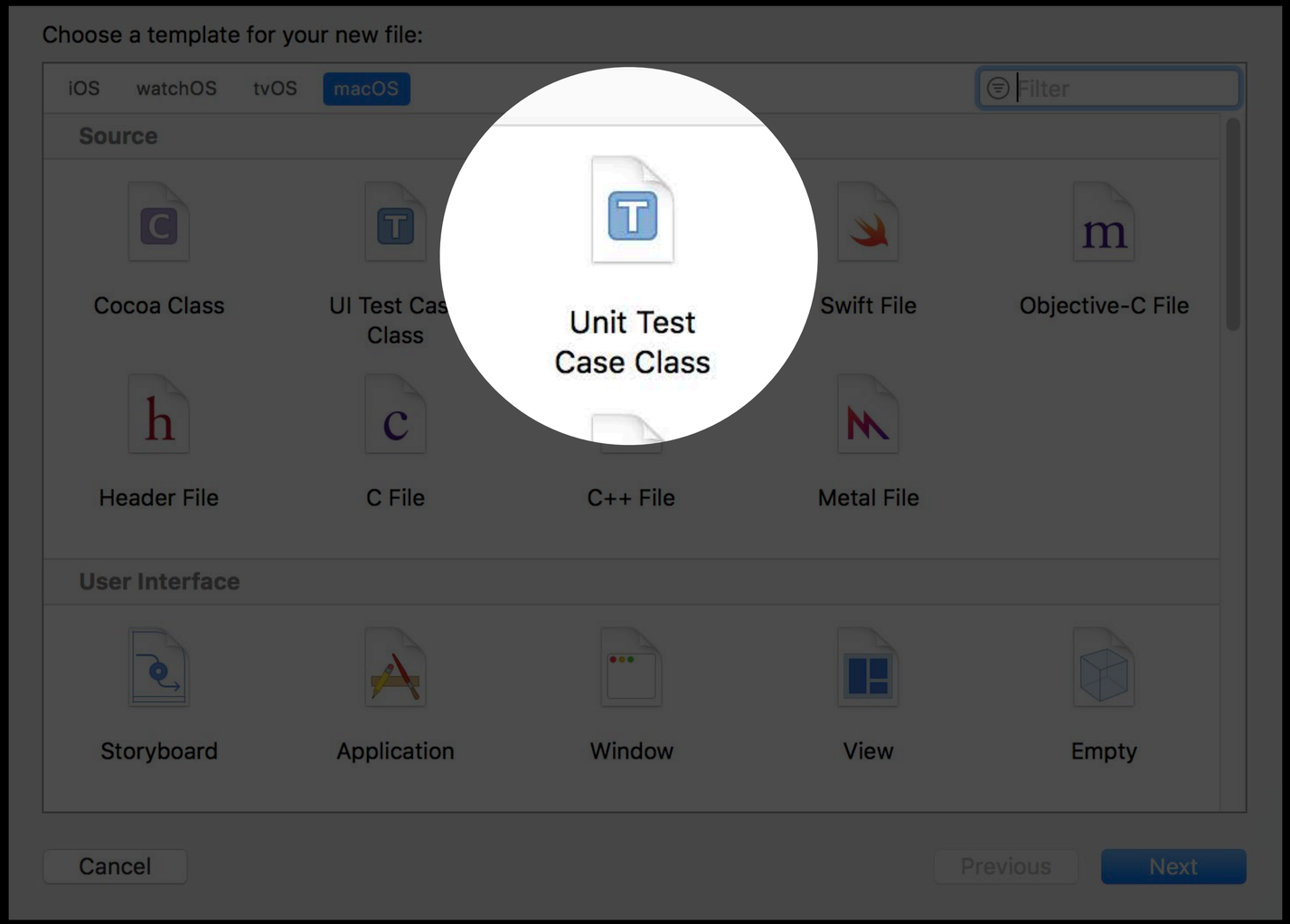
Unit Tests

Even the most basic test is better than no test

Unit Tests


Choose a template for your new file:
iOS
watchOS
tvOS
macOS
Filter
Source
Cocoa Class
UI Test Case Class
Unit Test Case Class
Swift File
Objective-C File
Header File
C File
C++ File
Metal File
User Interface
Storyboard
Application
Window
View
Empty
Cancel
Previous
Next

Unit Tests


Choose a template for your new file:
iOS
watchOS
tvOS
macOS
Filter
Source
Cocoa Class
UI Test Cas
Class
Unit Test
Case Class
Swift File
Objective-C File
Header File
C File
C++ File
Metal File
User Interface
Storyboard
Application
Window
View
Empty
Cancel
Previous
Next

```
func testExample() {
    // This is an example of a functional test case.
    // Use XCTAssert and related functions to verify your tests
    // produce the correct results.
}
```

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

27

NSBox

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

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

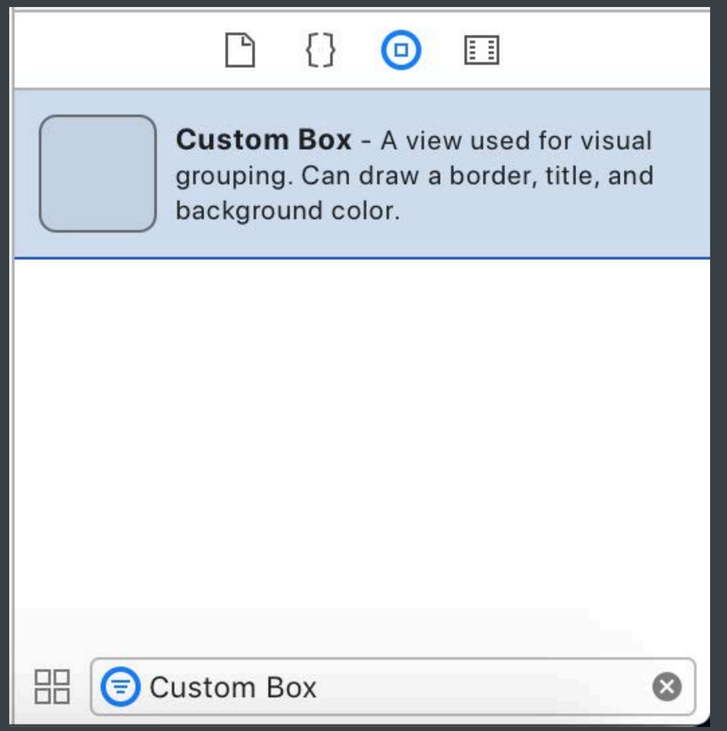
```
class MyView: NSView {
    public var backgroundColor = NSColor.clear
    override func draw(_ dirtyRect: NSRect) {
        backgroundColor.setFill()
        bounds.fill()
    }
}
```

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

    public var backgroundColor = NSColor.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?

}
```

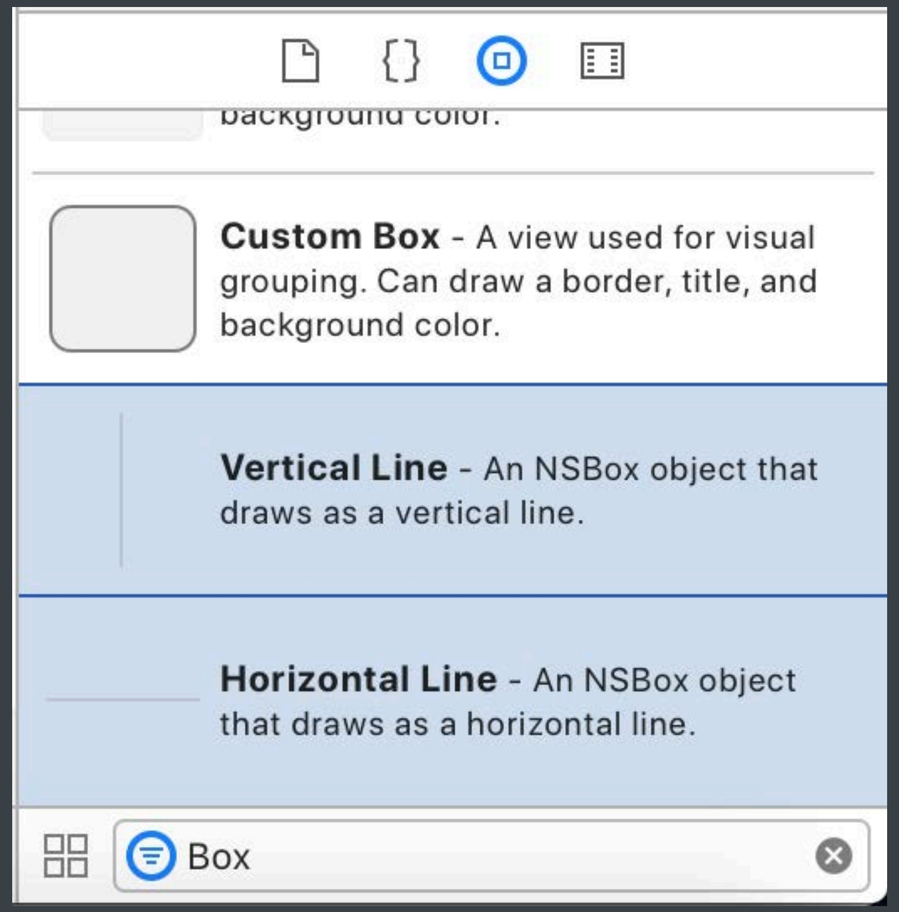
```
open class NSBox : NSView {
    open var borderWidth: CGFloat
    open var cornerRadius: CGFloat
    @NSCopying open var borderColor: NSColor
    @NSCopying open var fillColor: NSColor

…

    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
```

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



8

Restorable State

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

```
extension NSResponder {
    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

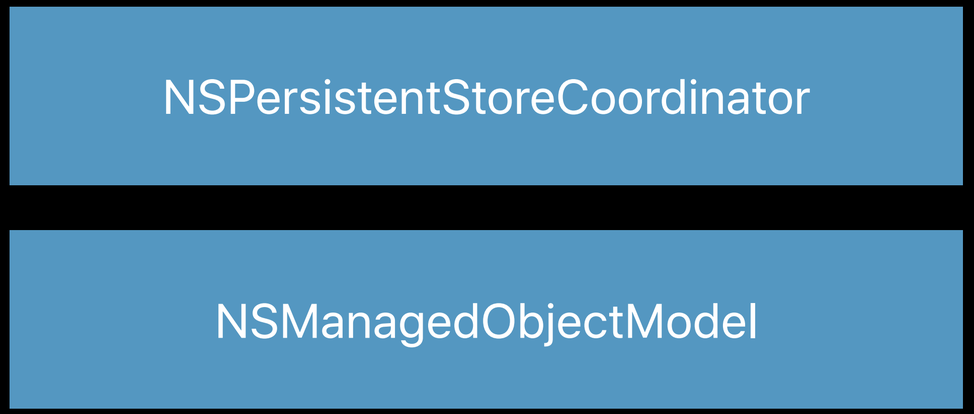
Core Data: NSPersistentContainer

What Is a Core Data Stack?

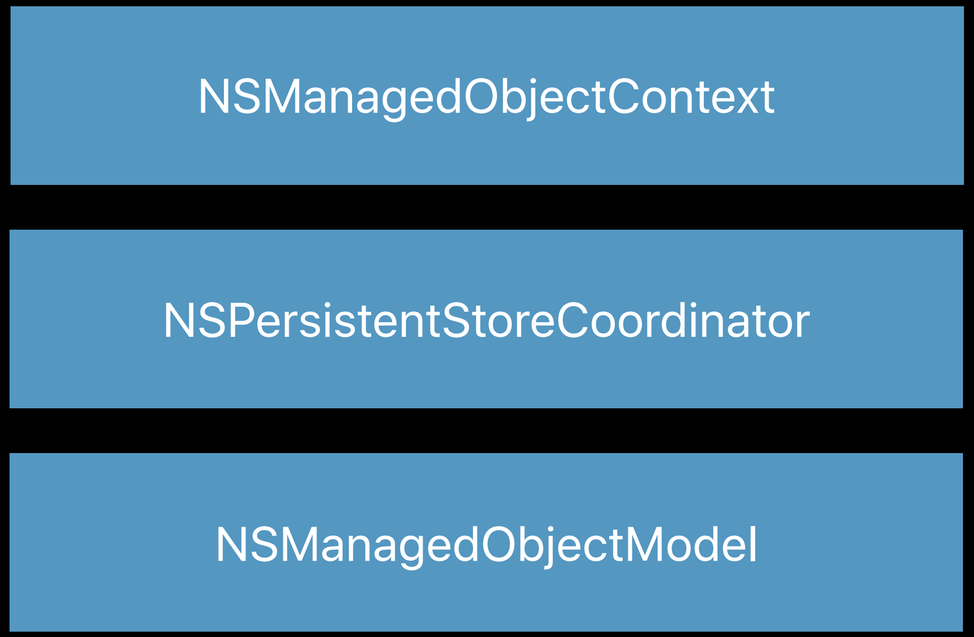
What Is a Core Data Stack?

NSManagedObjectModel

What Is a Core Data Stack?



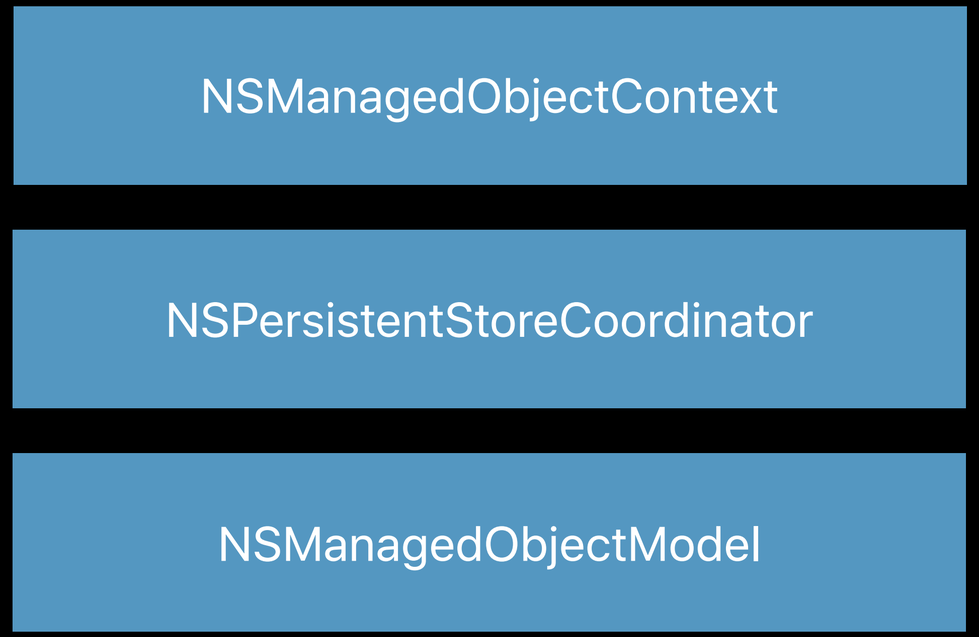
What Is a Core Data Stack?



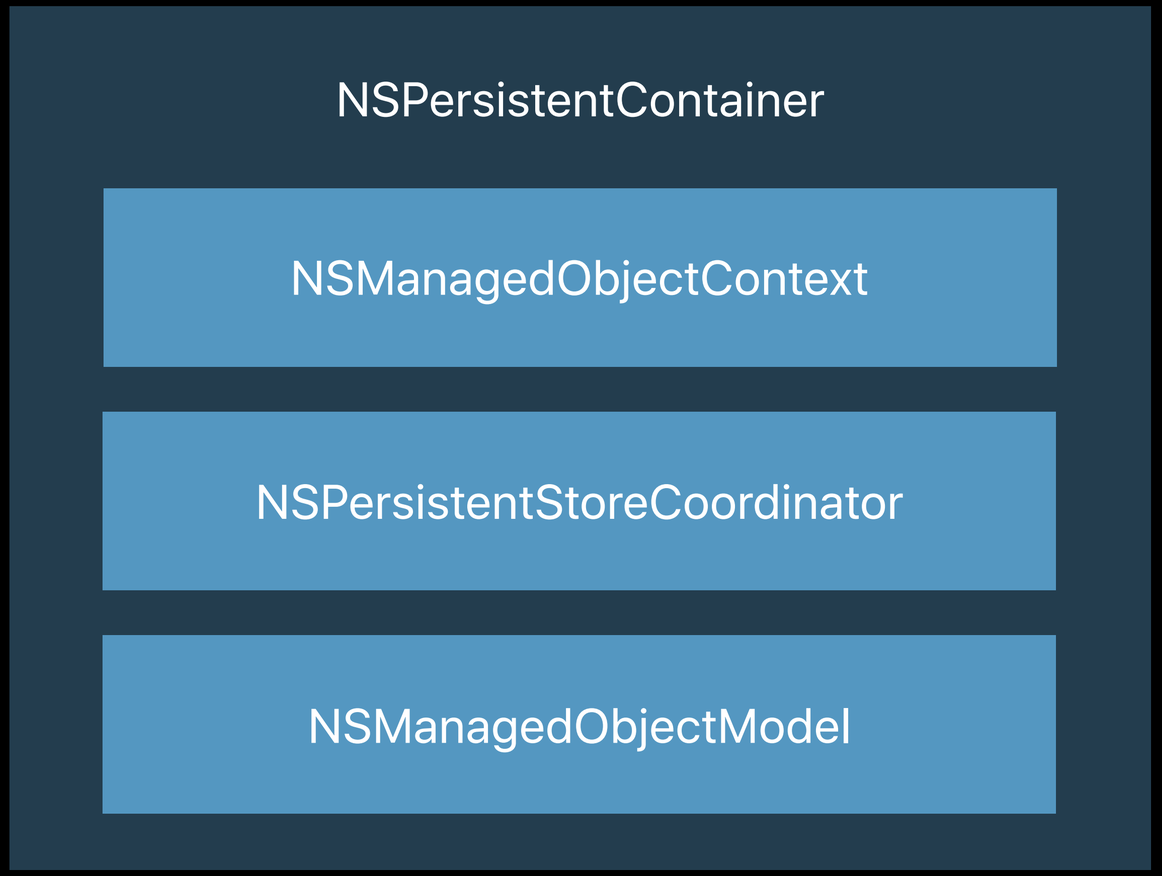
```
class DataControllerOld: NSObject {
    var applicationDocumentsDirectory: URL = {
        let urls = FileManager.default.urls(for: .documentDirectory, in: .userDomainMask)
        return urls[urls.count-1]}()
    var managedObjectModel: NSManagedObjectModel = {
        let modelURL = Bundle.main.url(forResource: "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



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()
        }
    }
}
```

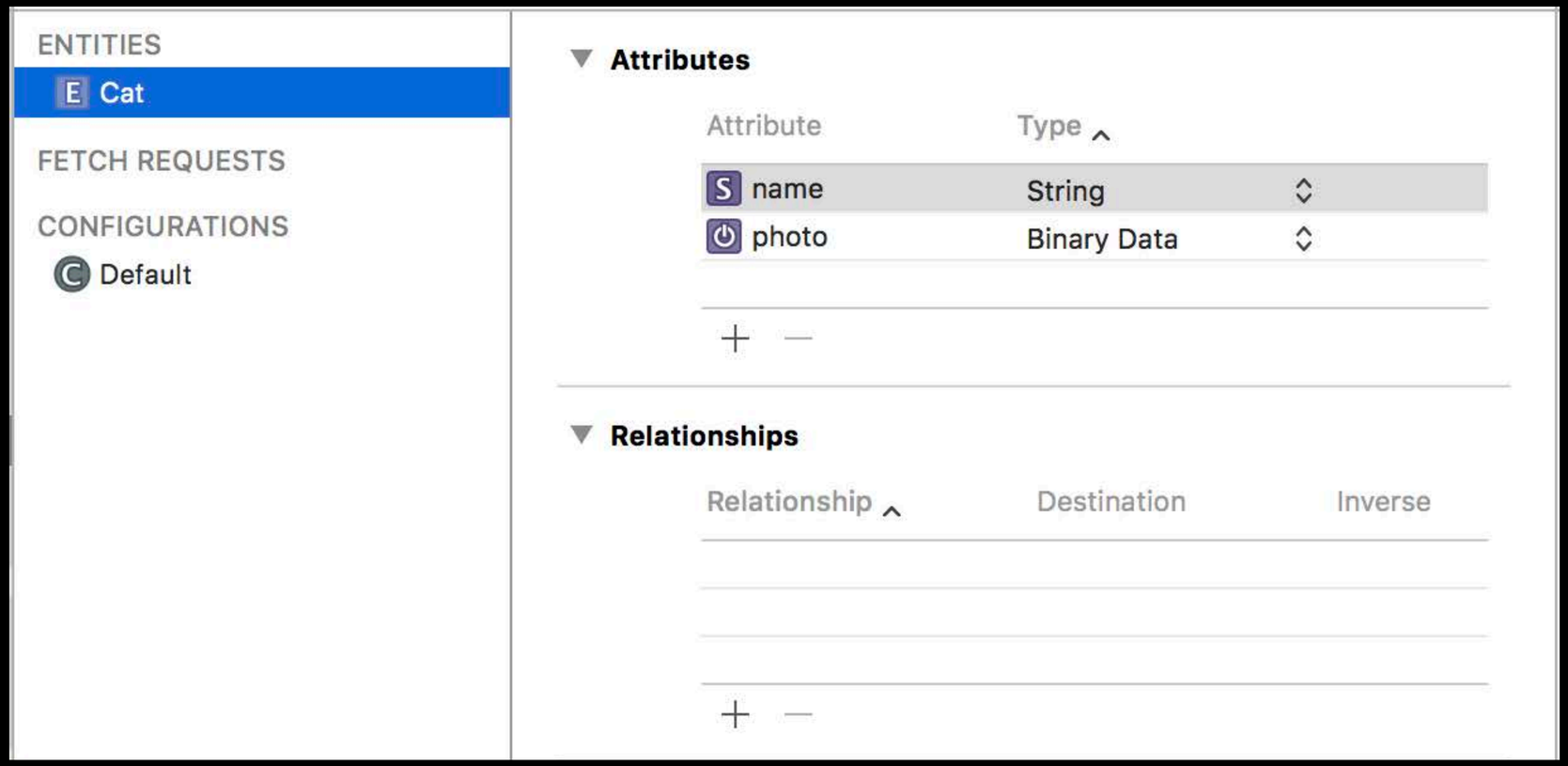
21

Core Data: Arrays


Cat Wrangler

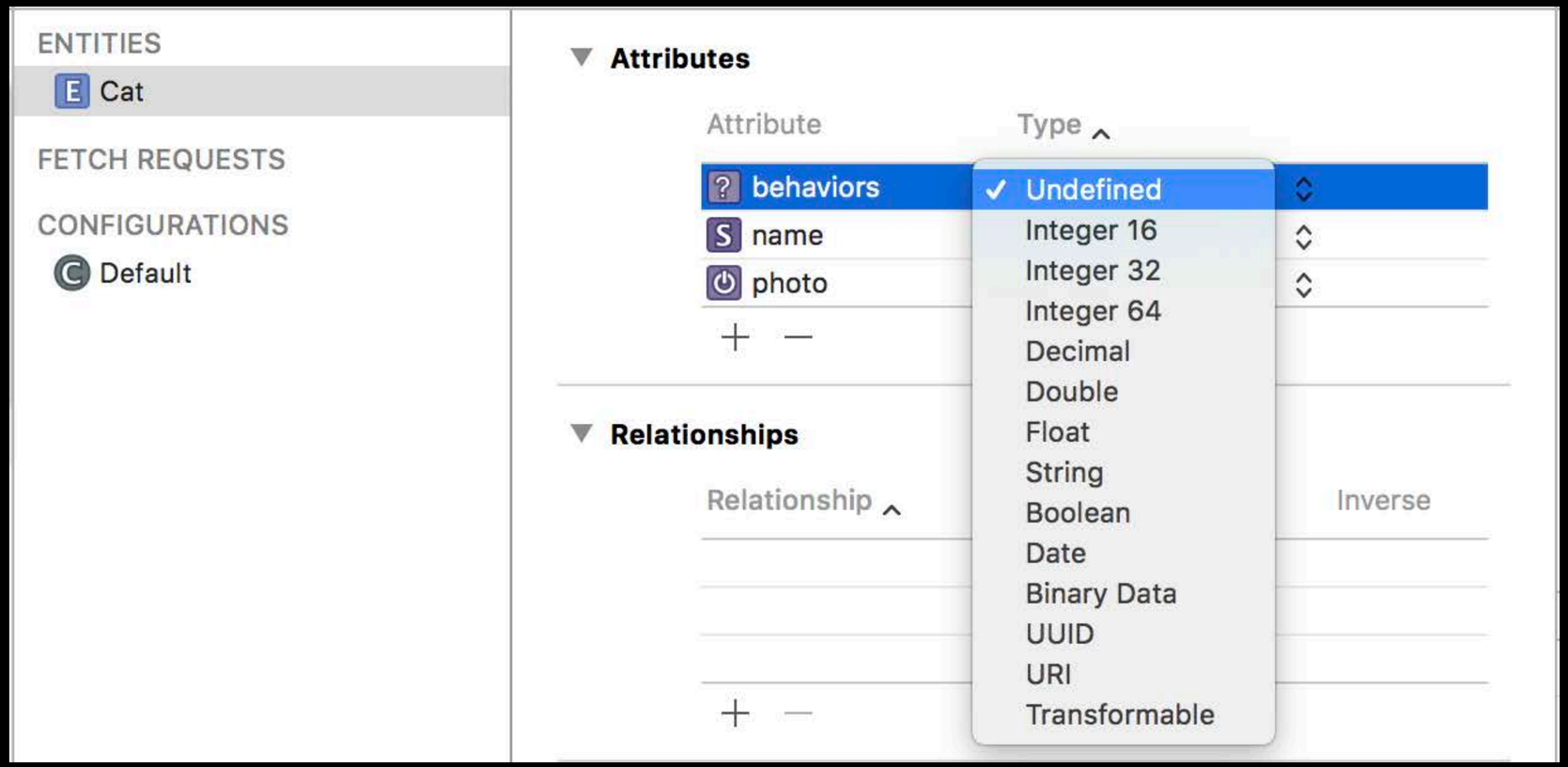
Core Data

Arrays



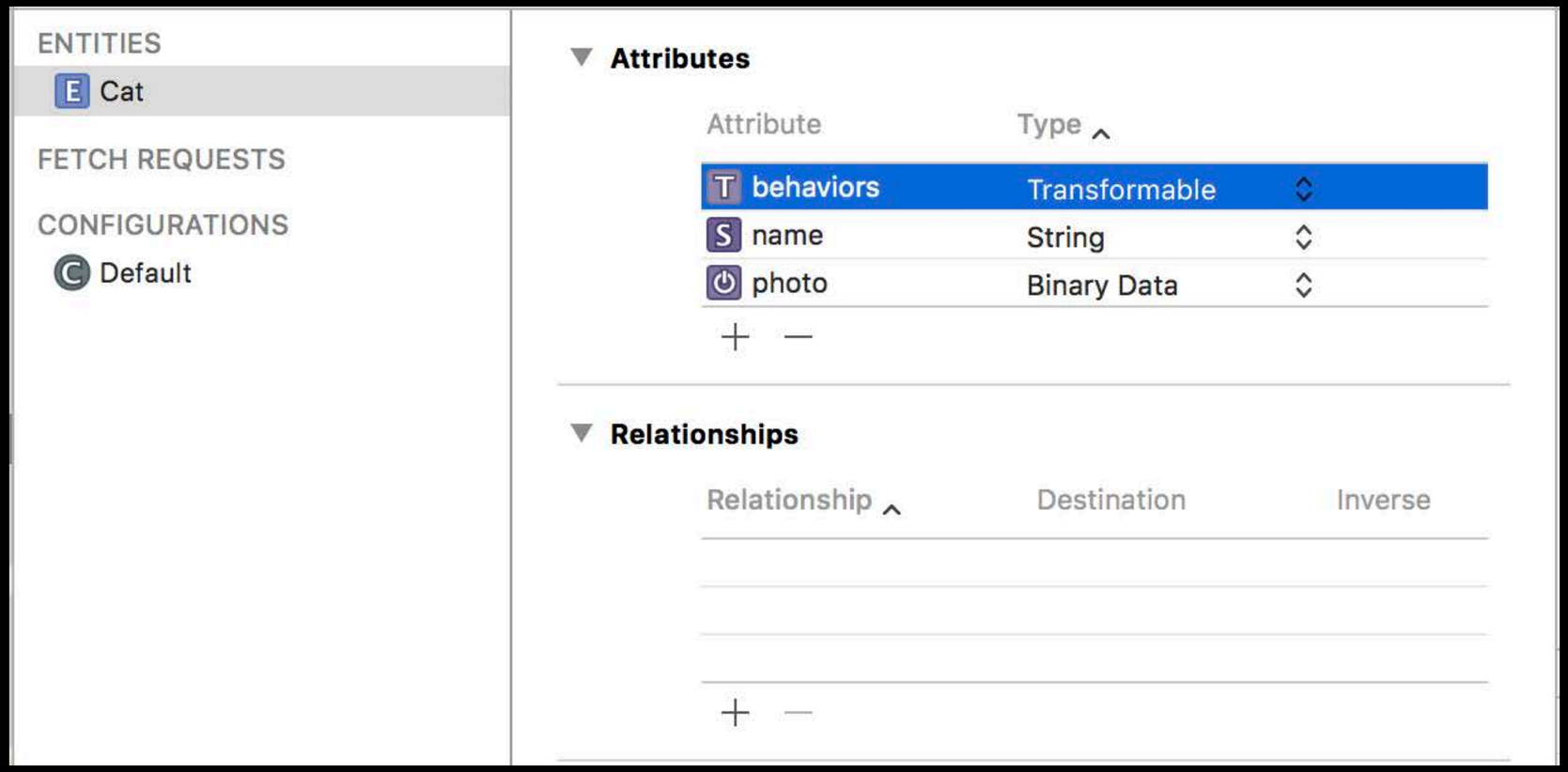
Core Data

Arrays



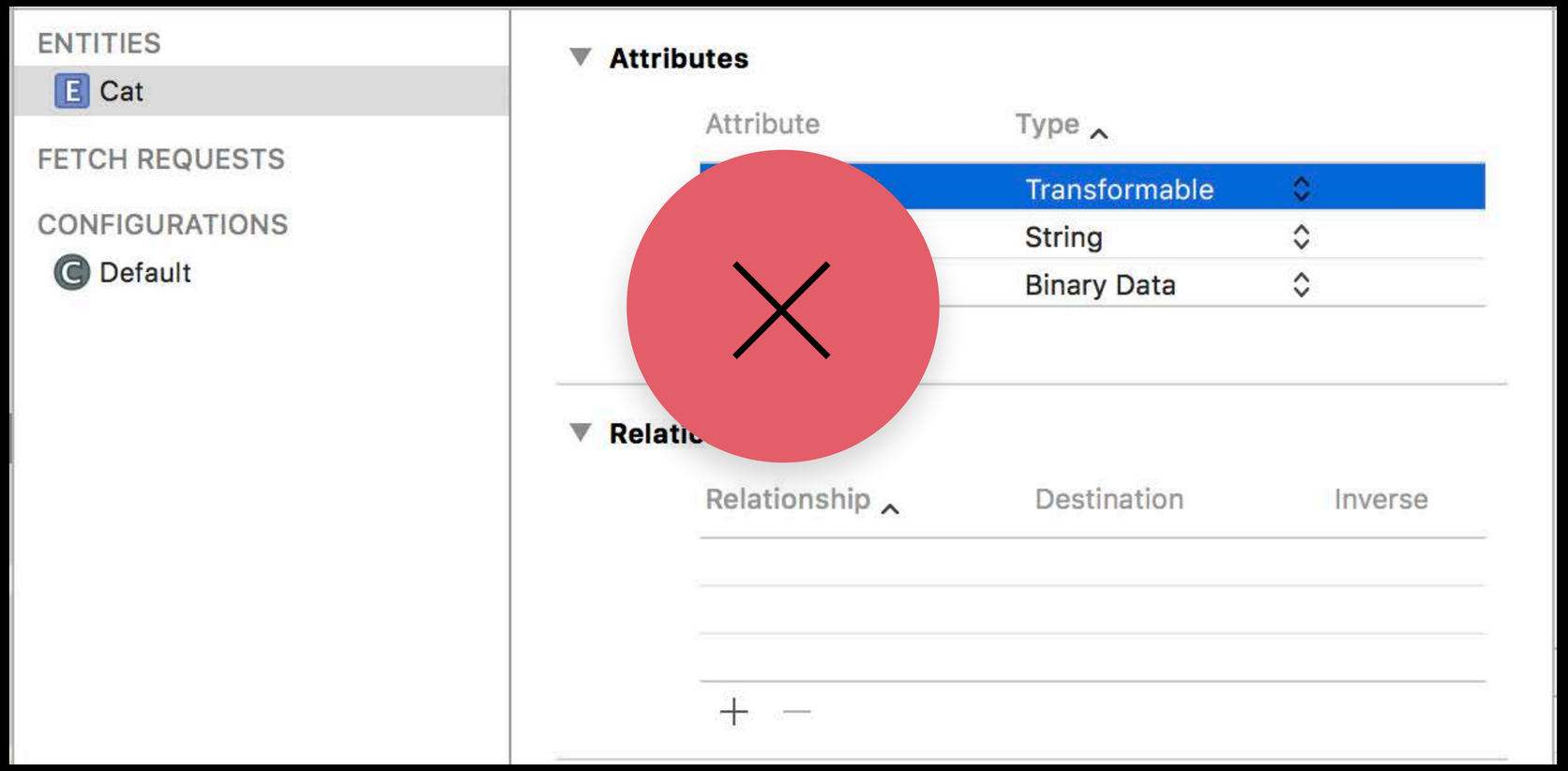
Core Data

Arrays



Core Data

Arrays



Core Data

Arrays

Core Data

Arrays

Added overhead for serializing and deserializing the array and its contents

Core Data

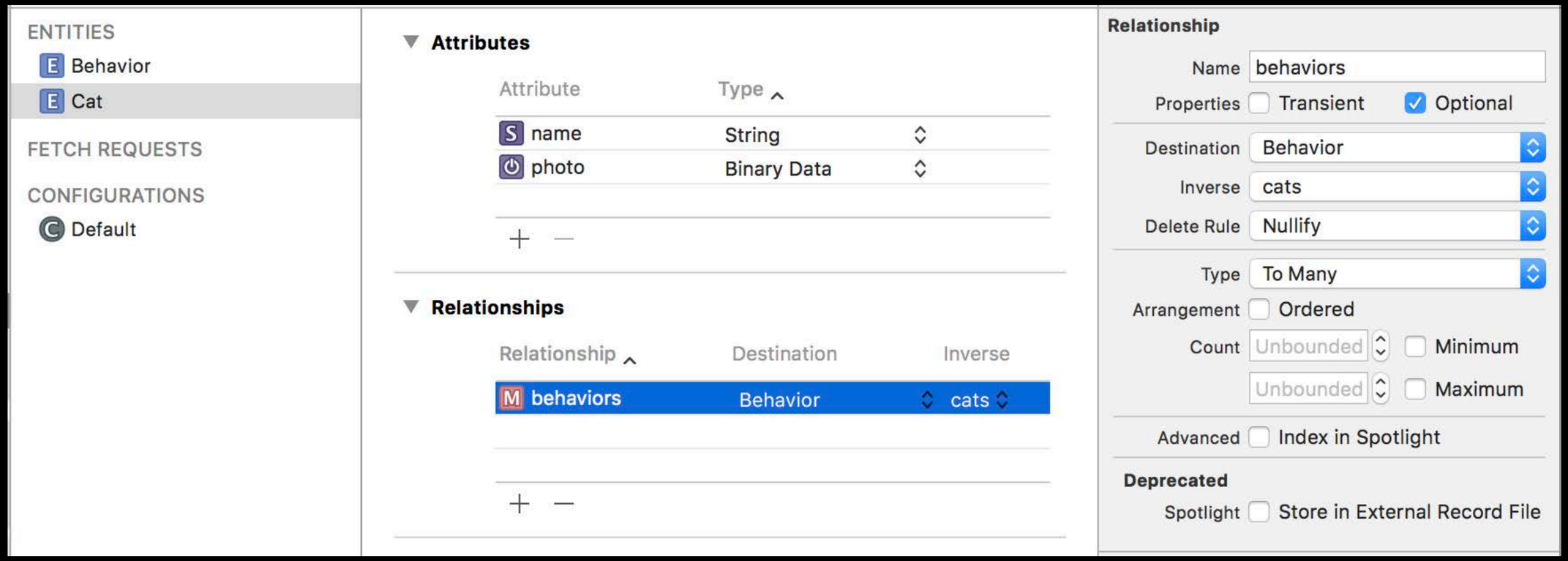
Arrays

Added overhead for serializing and deserializing the array and its contents

Fetch requests are much slower

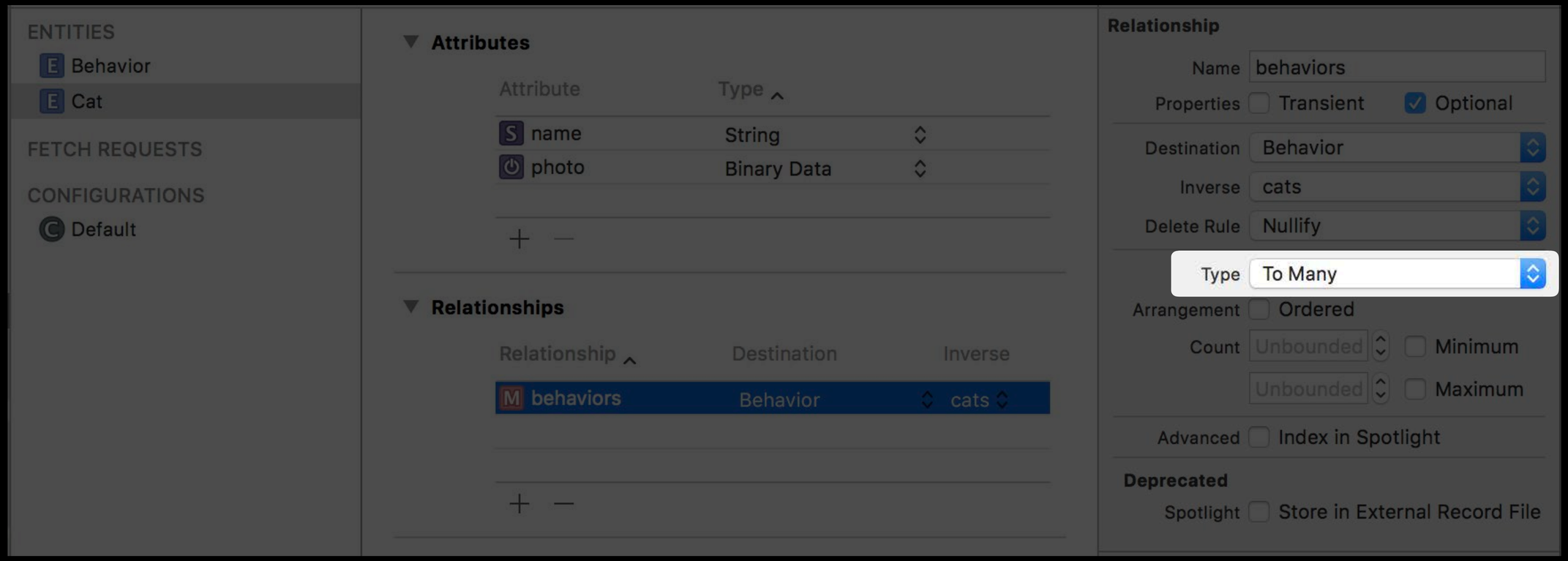
Core Data

Relationships



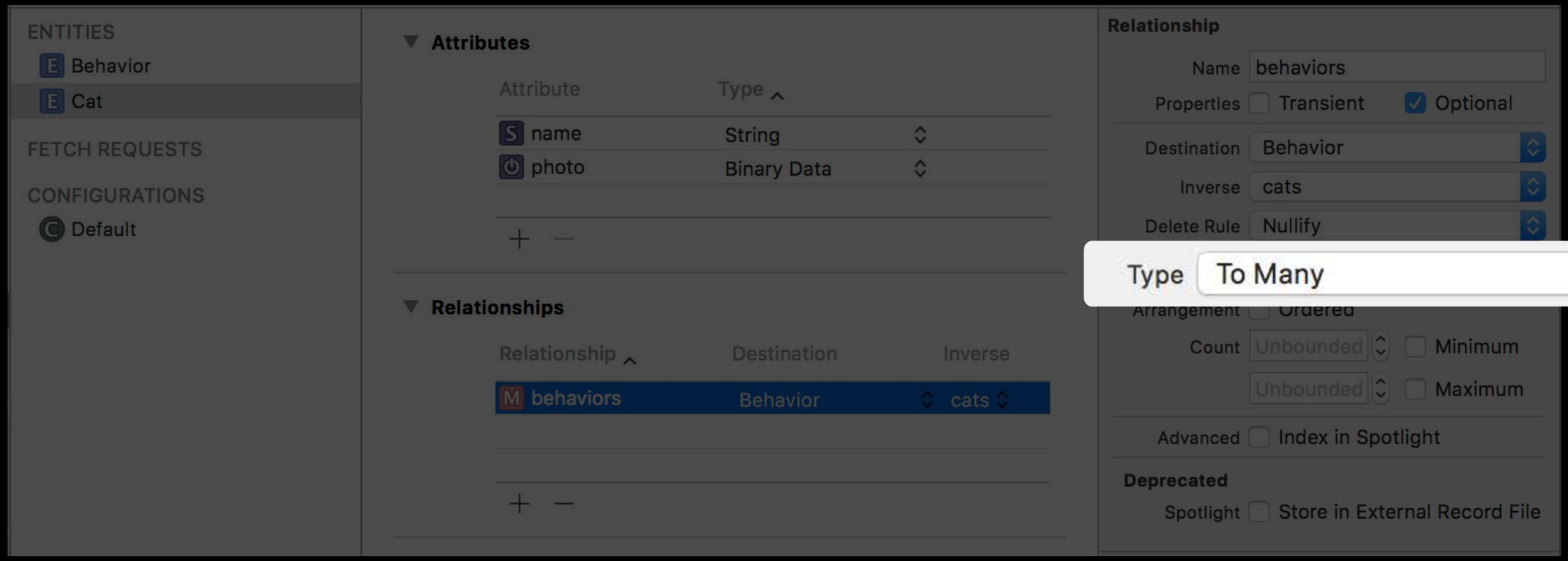
Core Data

To many relationships



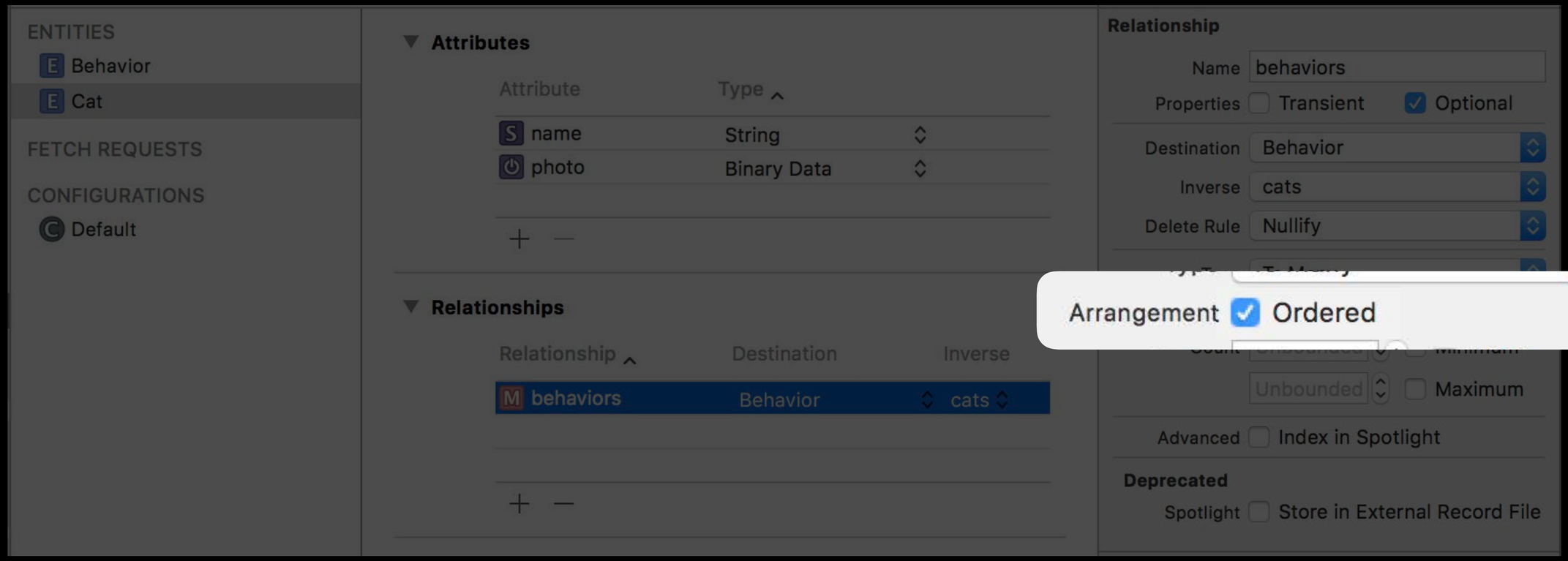
Core Data

To many relationships



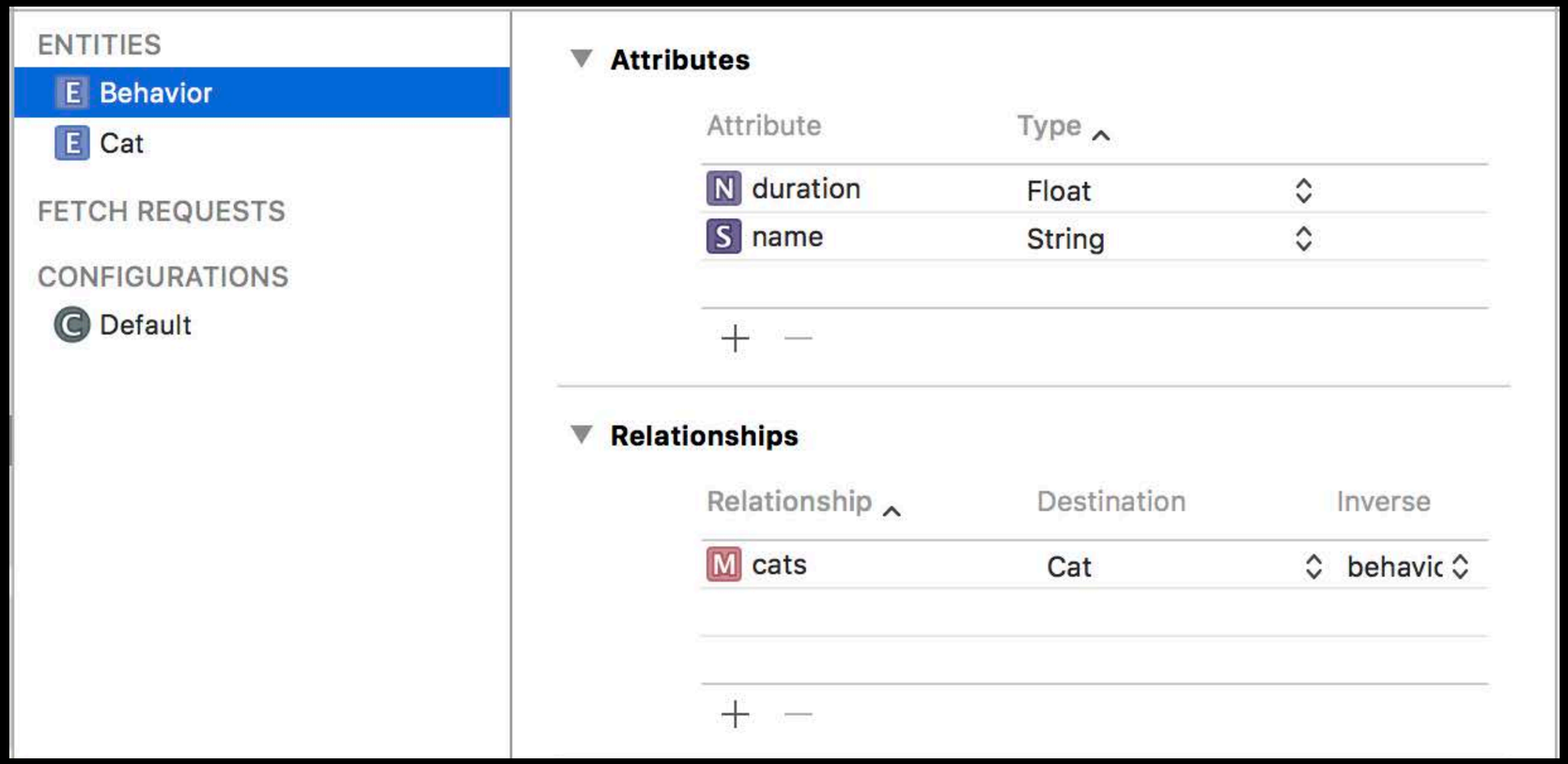
Core Data

To many relationships



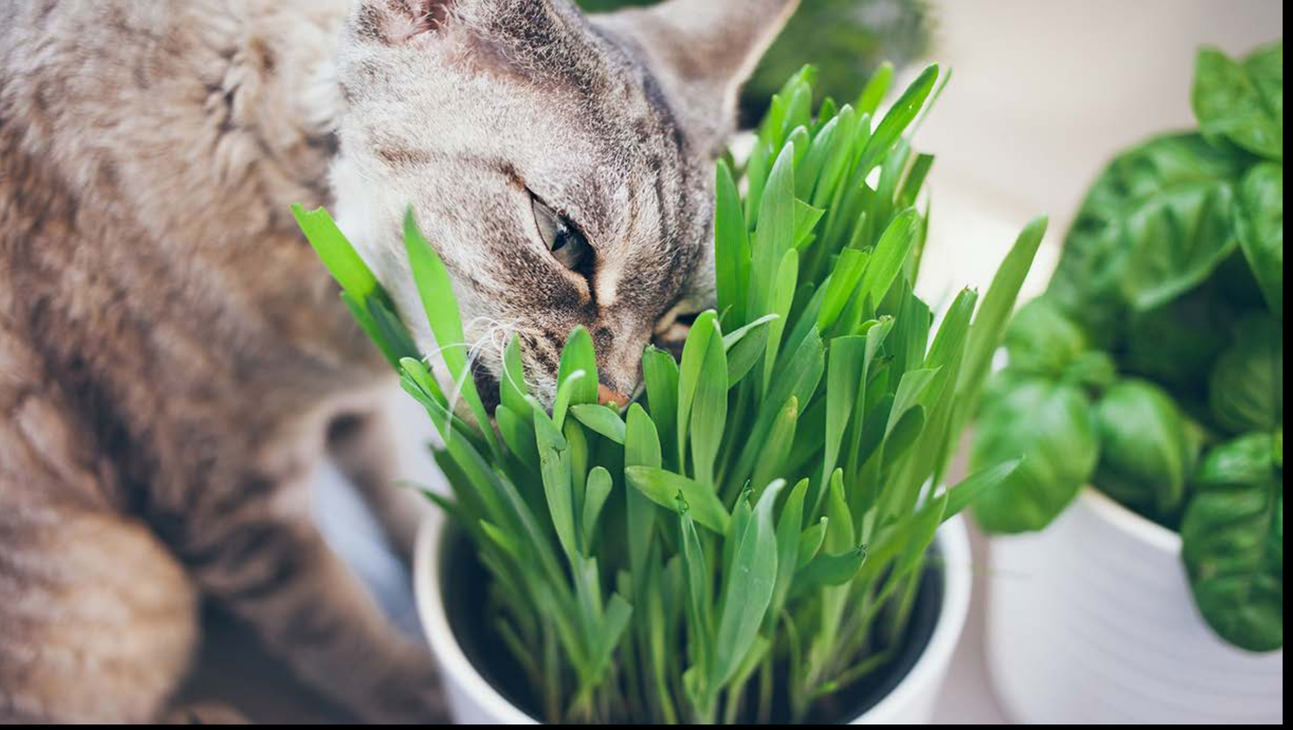
Core Data

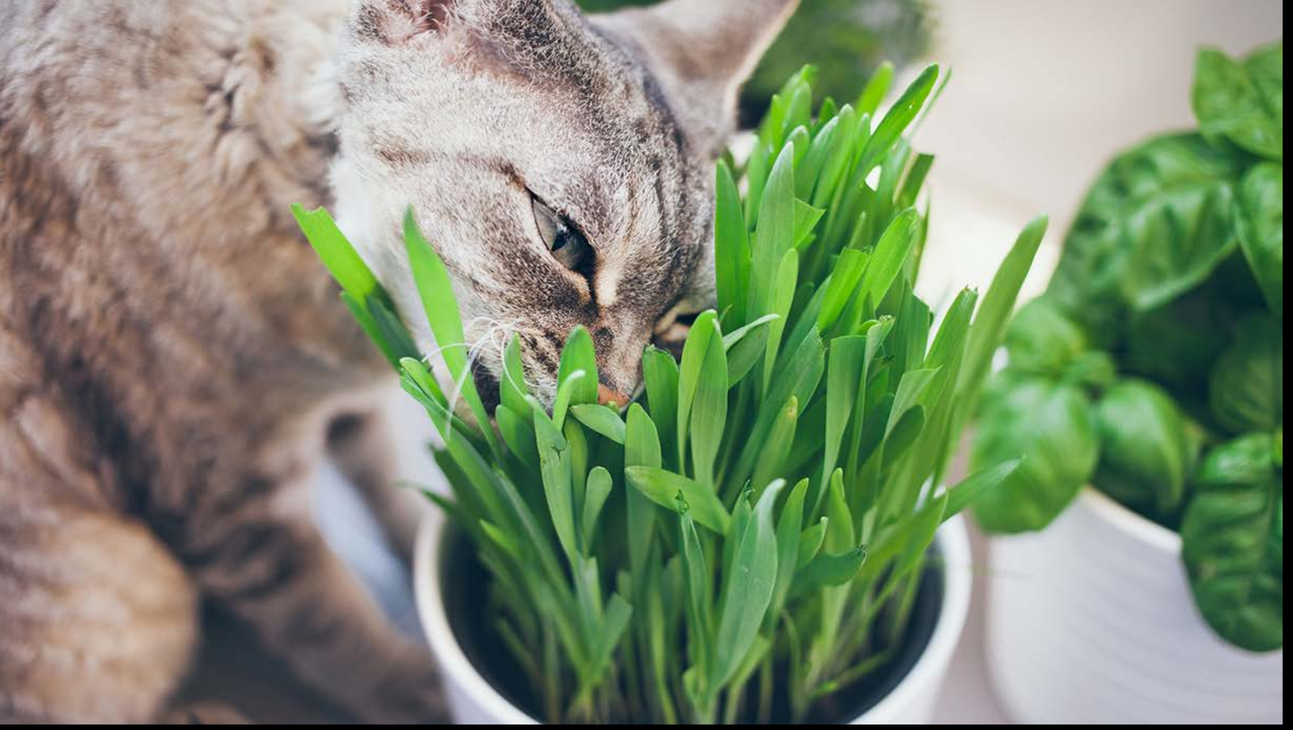
To many relationships

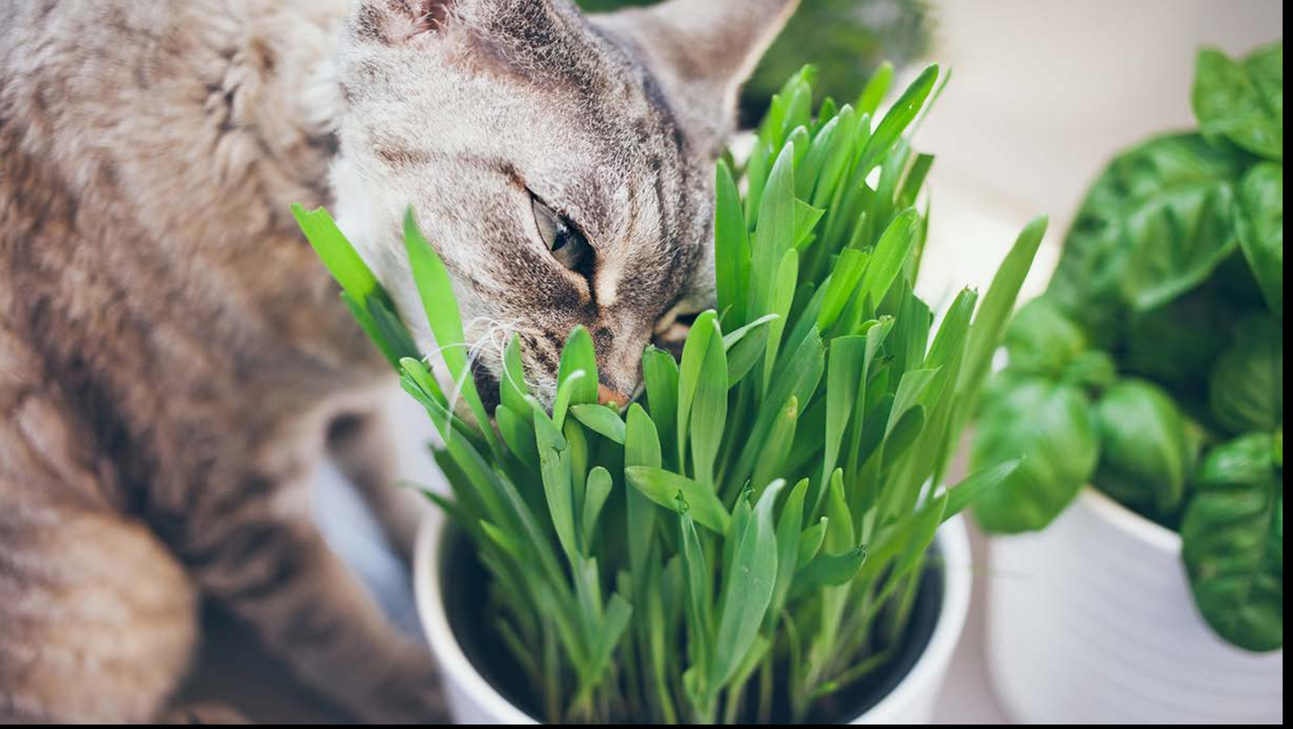


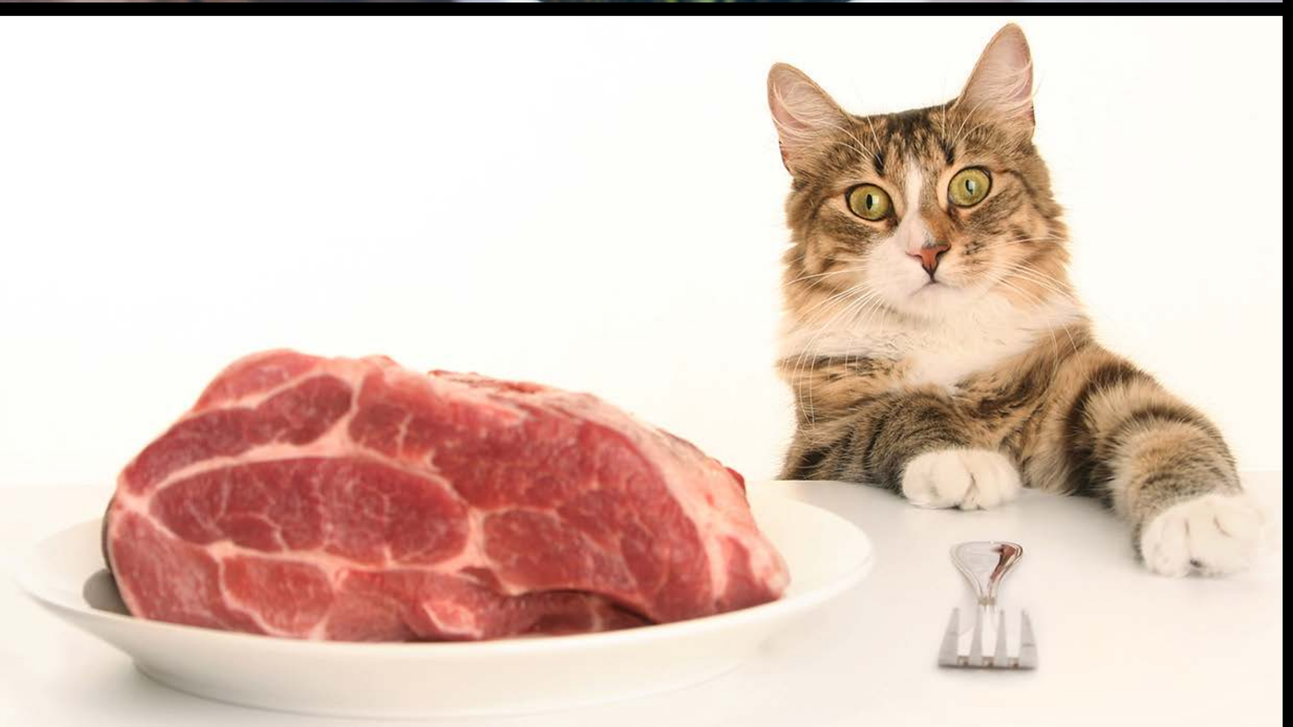
34

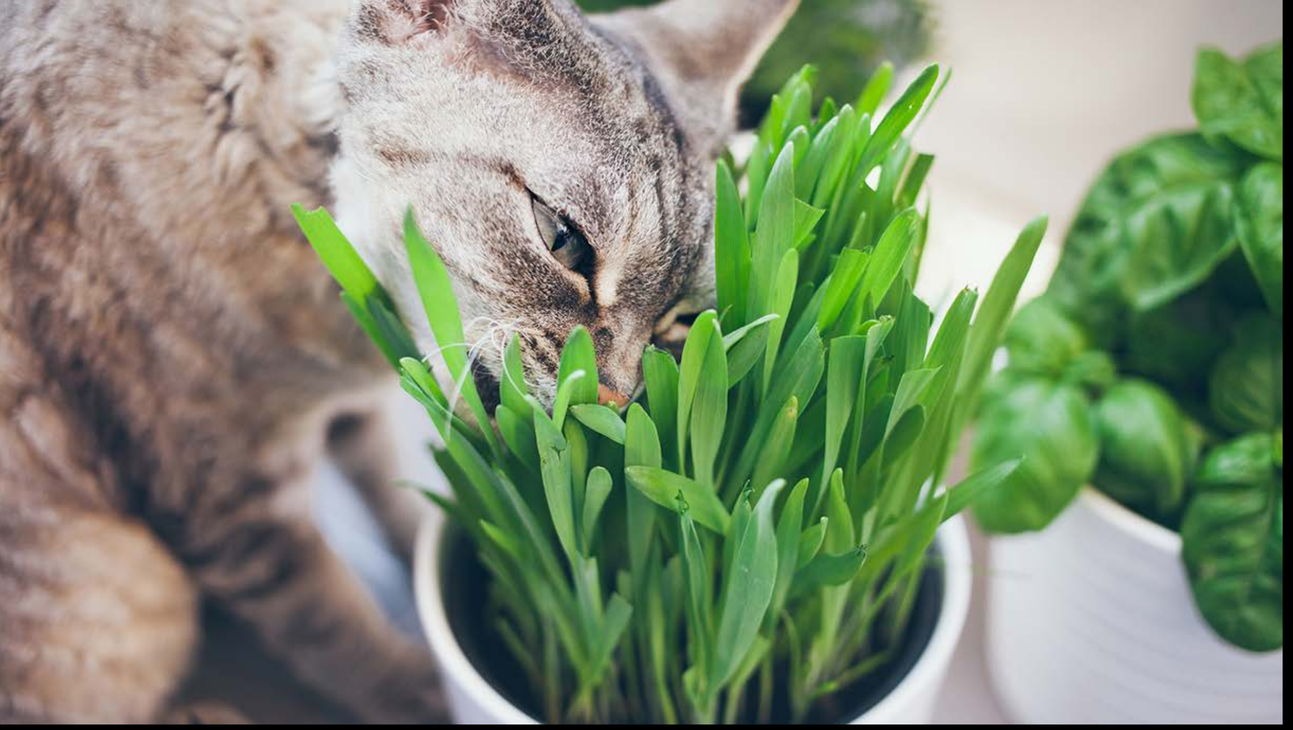
Core Data: Migration

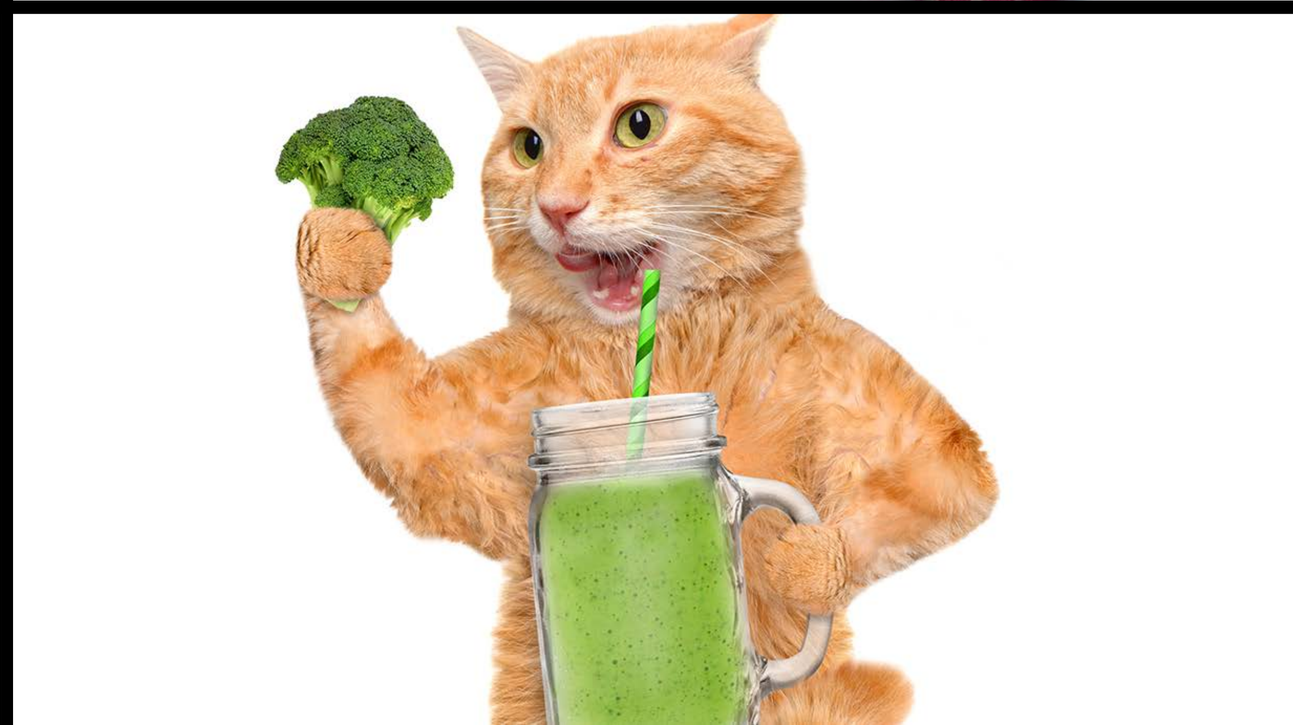






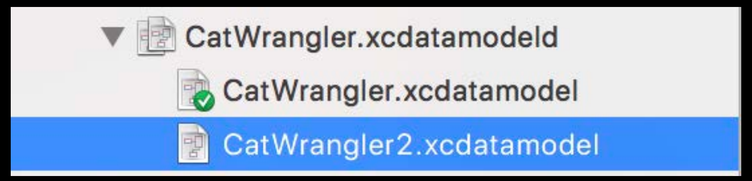






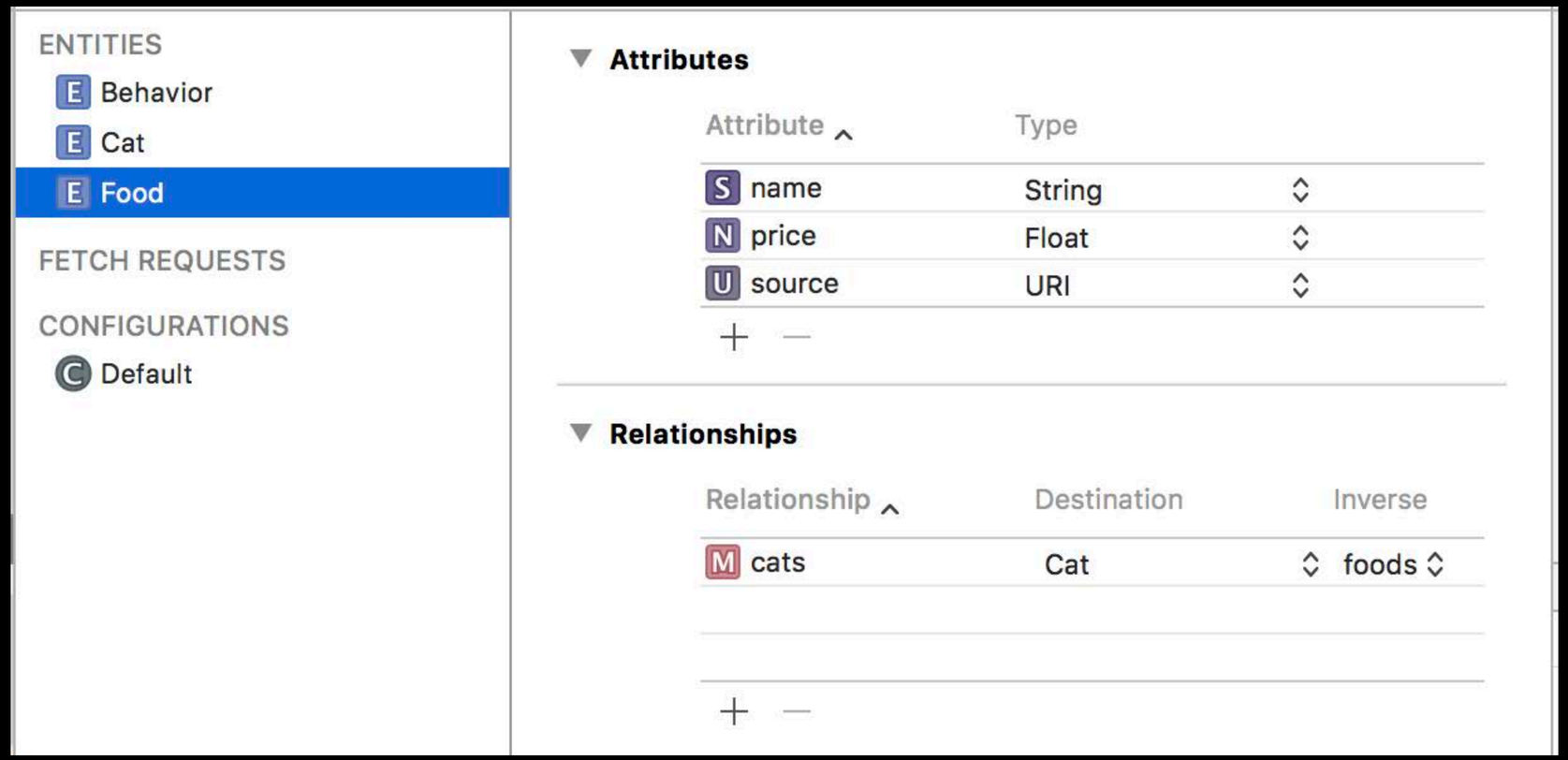
Core Data

Migration



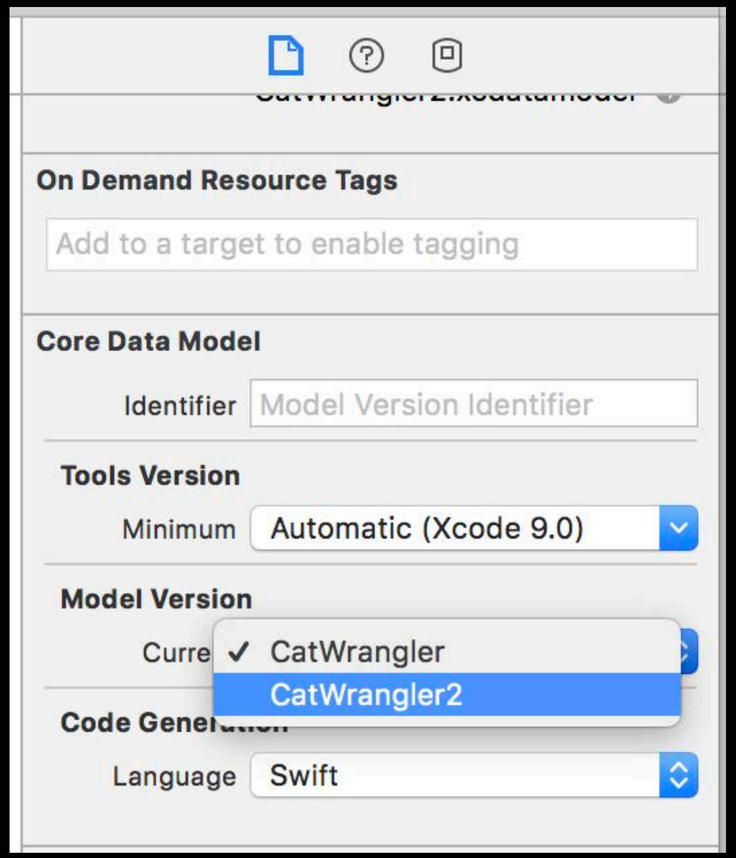
Core Data

Migration



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 NSErrors 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
}
```

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

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

```
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)
```

```
public var NSFileWriteNoPermissionError: Int { get } // Write error (permission problem)
public var NSFileWriteInvalidFileNameError: Int { get } // Write error (invalid file name)
public var NSFileWriteFileExistsError: Int { get } // Write error (file exists)
public var NSFileWriteInapplicableStringEncodingError: Int { get } // Write error (string
                                        encoding not applicable) also NSStringEncodingErrorKey
public var NSFileWriteUnsupportedSchemeError: Int { get } // Write error (unsupported URL
                                                    scheme)
public var NSFileWriteOutOfSpaceError: Int { get } // Write error (out of disk space)
public var NSFileWriteVolumeReadOnlyError: Int { get } // Write error (readonly volume)

// NSFileManager unmount errors
public var NSFileManagerUnmountUnknownError: Int { get } // The volume could not be unmounted
                                                    (reason unknown)
public var NSFileManagerUnmountBusyError: Int { get } // The volume could not be unmounted
                                                  because it is in use
```

```
// 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
])
```

```
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
])
```

NEW

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

NEW

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

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!
```

0xa

Accessibility

Accessibility

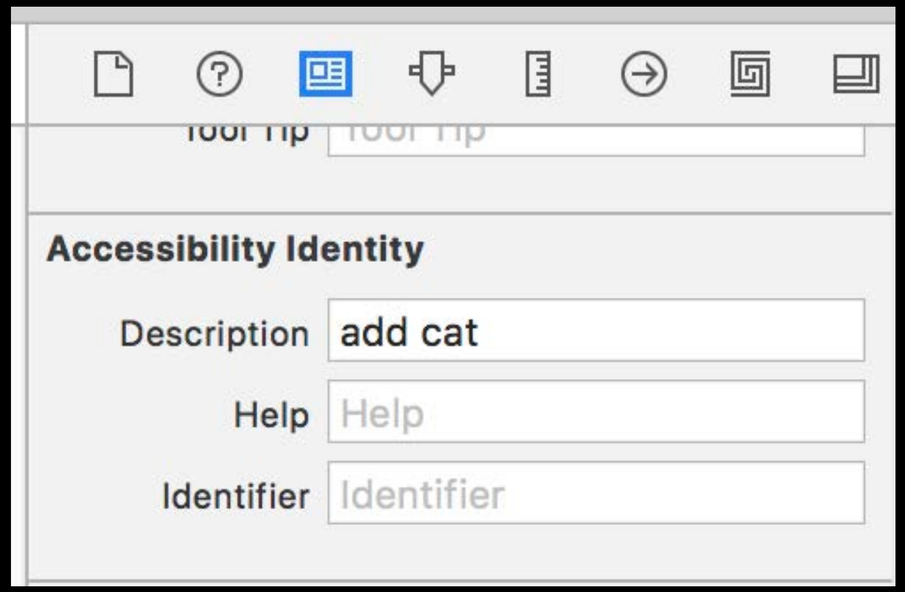
VoiceOver

VoiceOver information is easy to add in Interface Builder

VoiceOver comes built-in on Apple products

Accessibility

VoiceOver

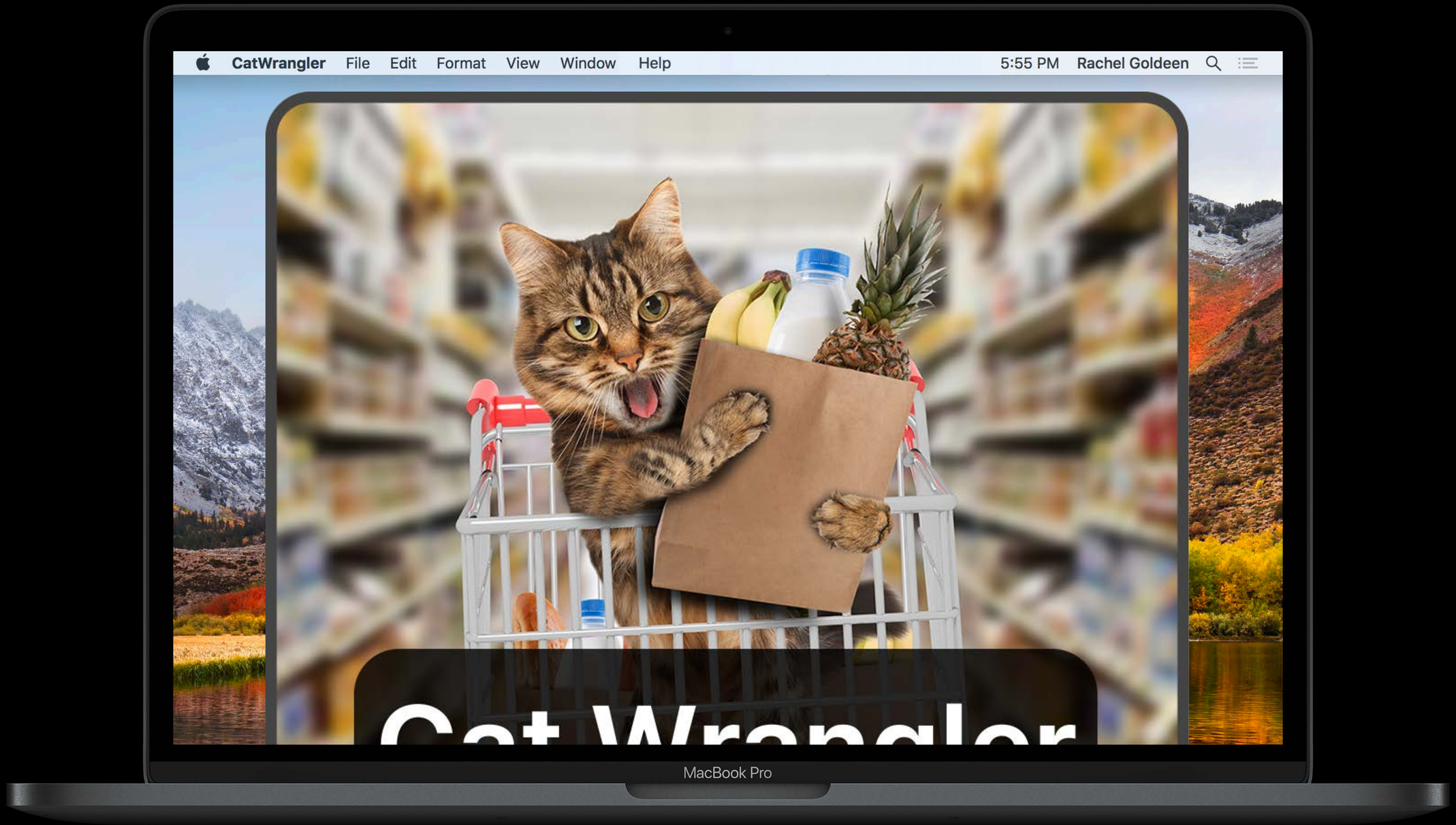


Accessibility

Screen resolution

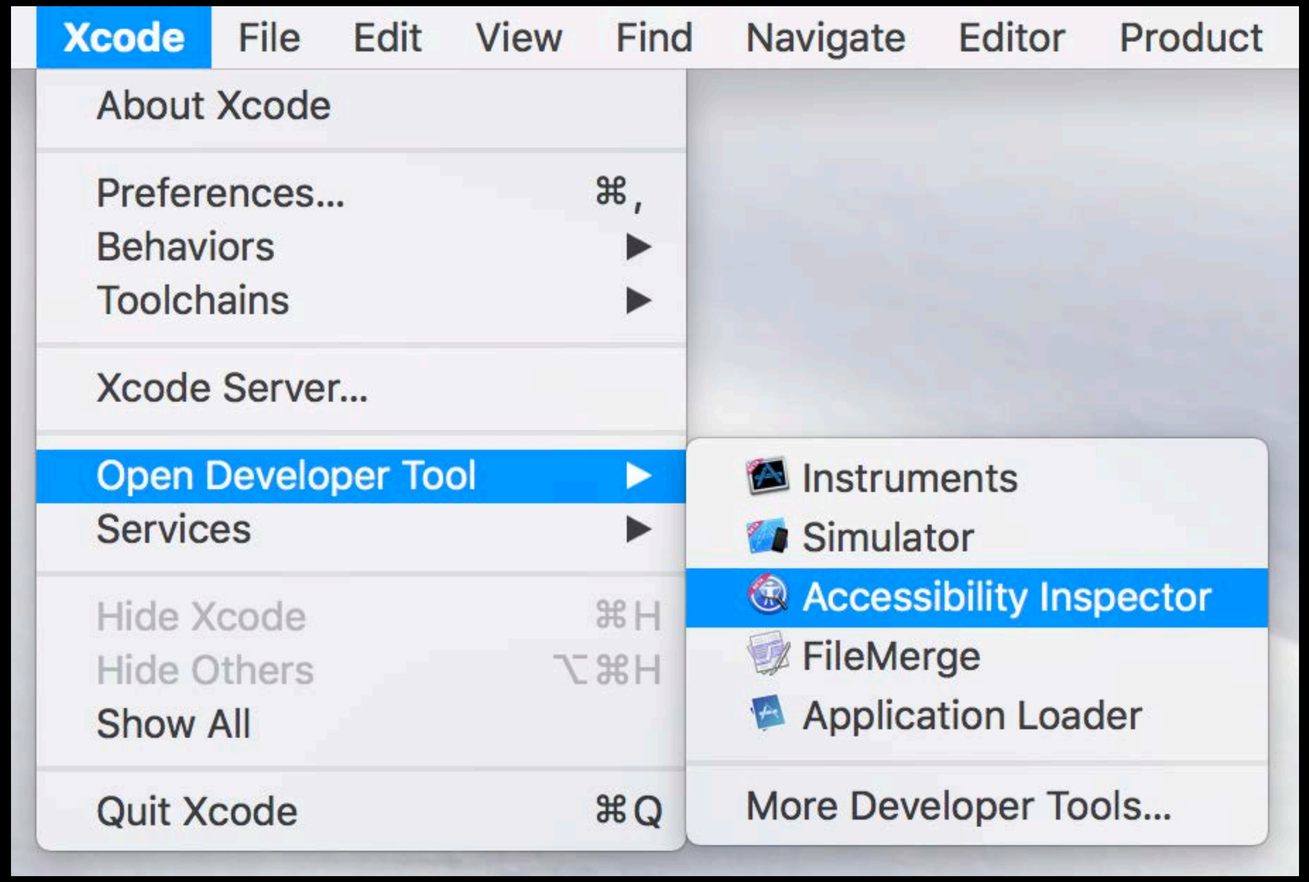
Test at 1024 x 640

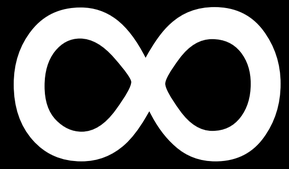

CatWrangler File Edit Format View Window Help
Tue 5:55 PM Rachel Goldeen
Cat Wrangler
MacBook Pro


CatWrangler File Edit Format View Window Help
5:55 PM Rachel Goldeen
MacBook Pro

Accessibility

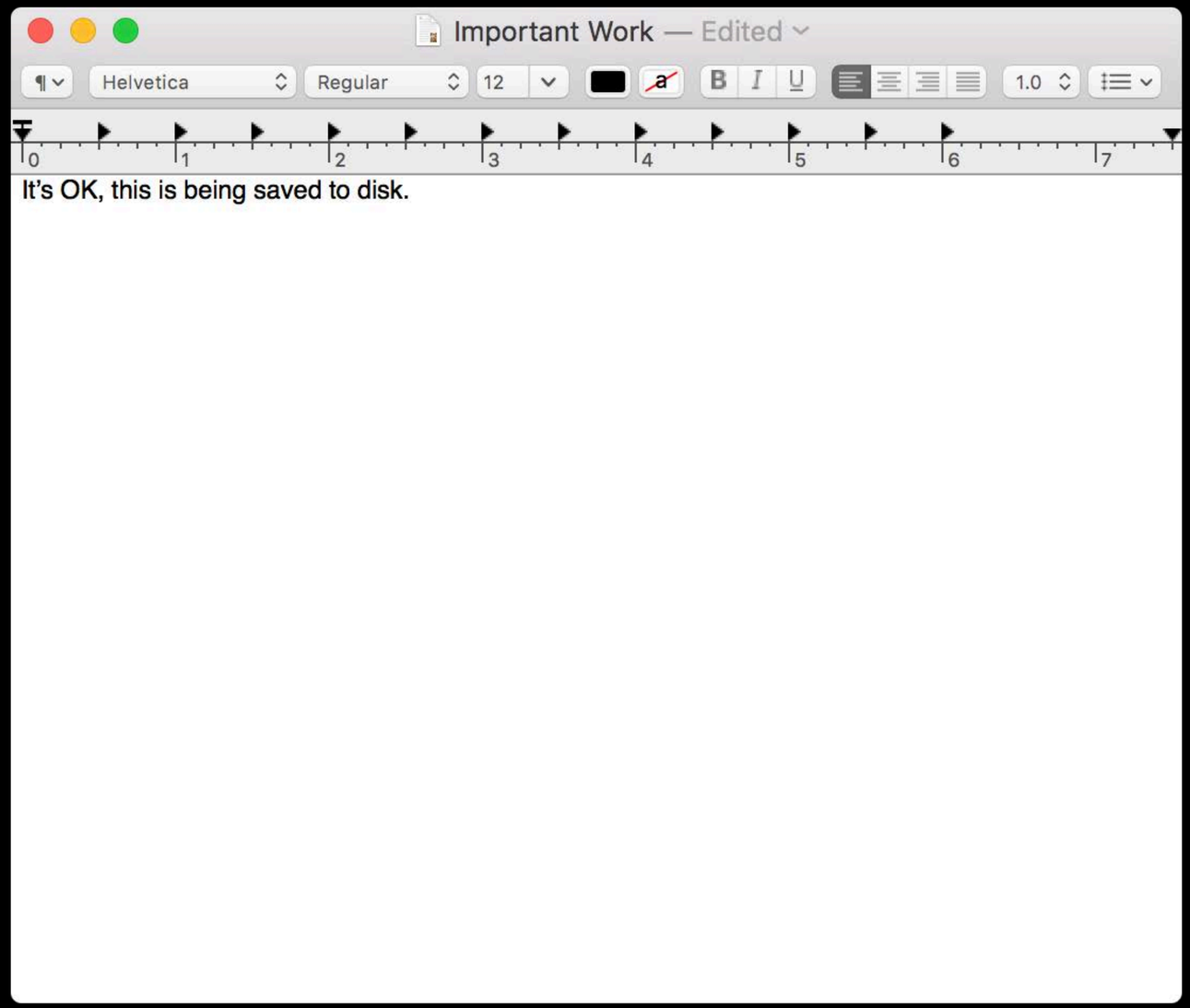
Accessibility Inspector in Xcode



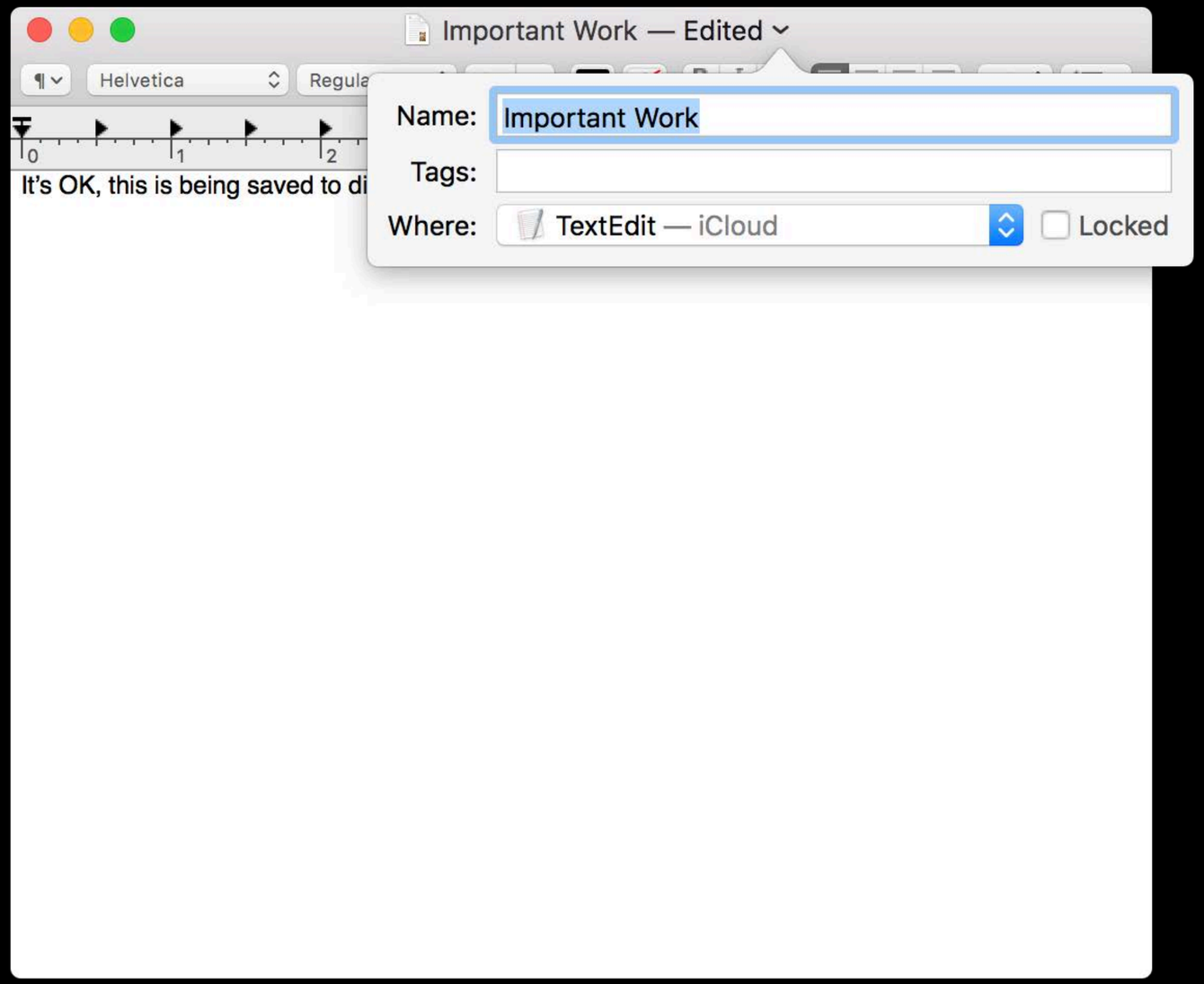


Documents

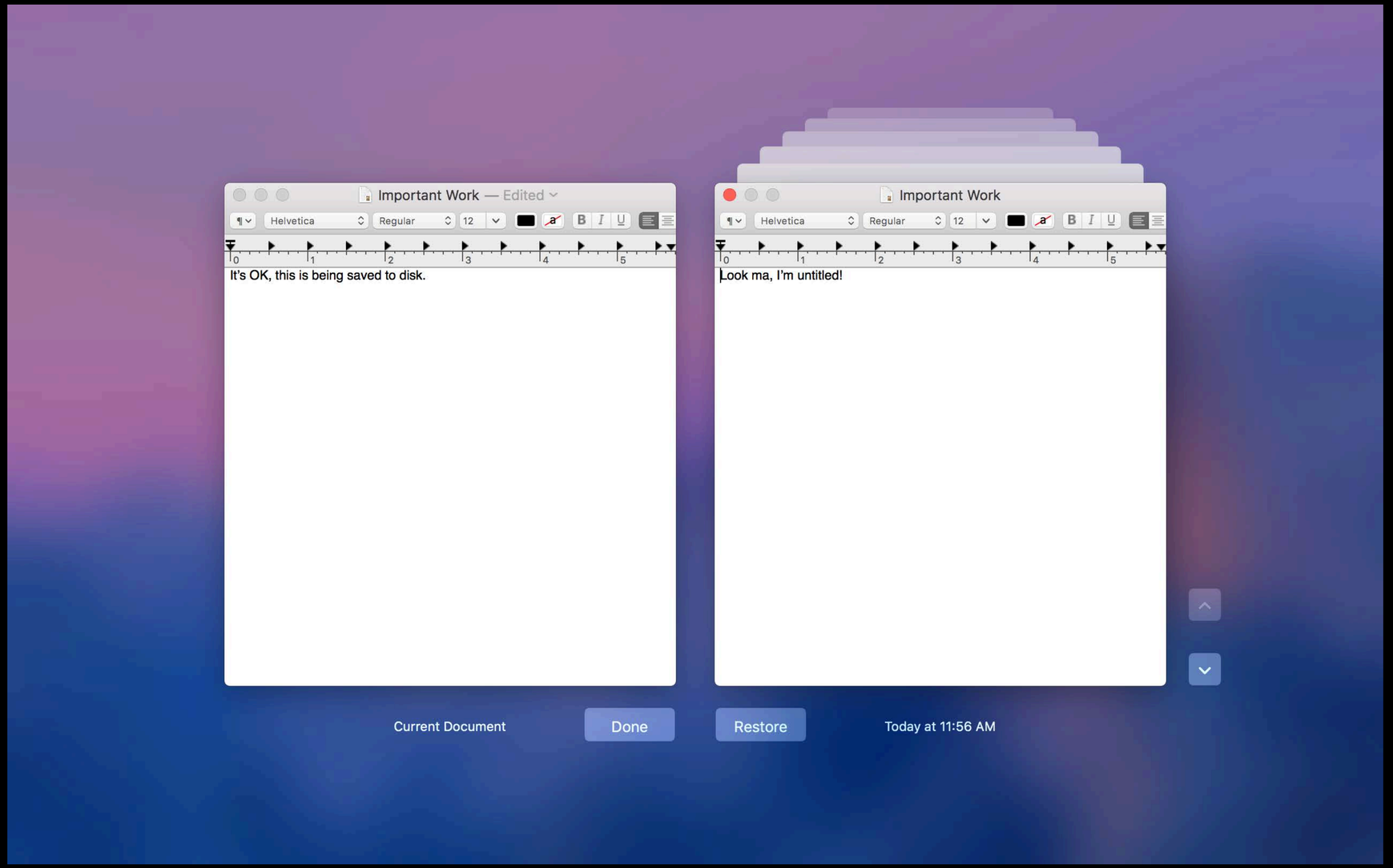
Document-based App



Document-based App

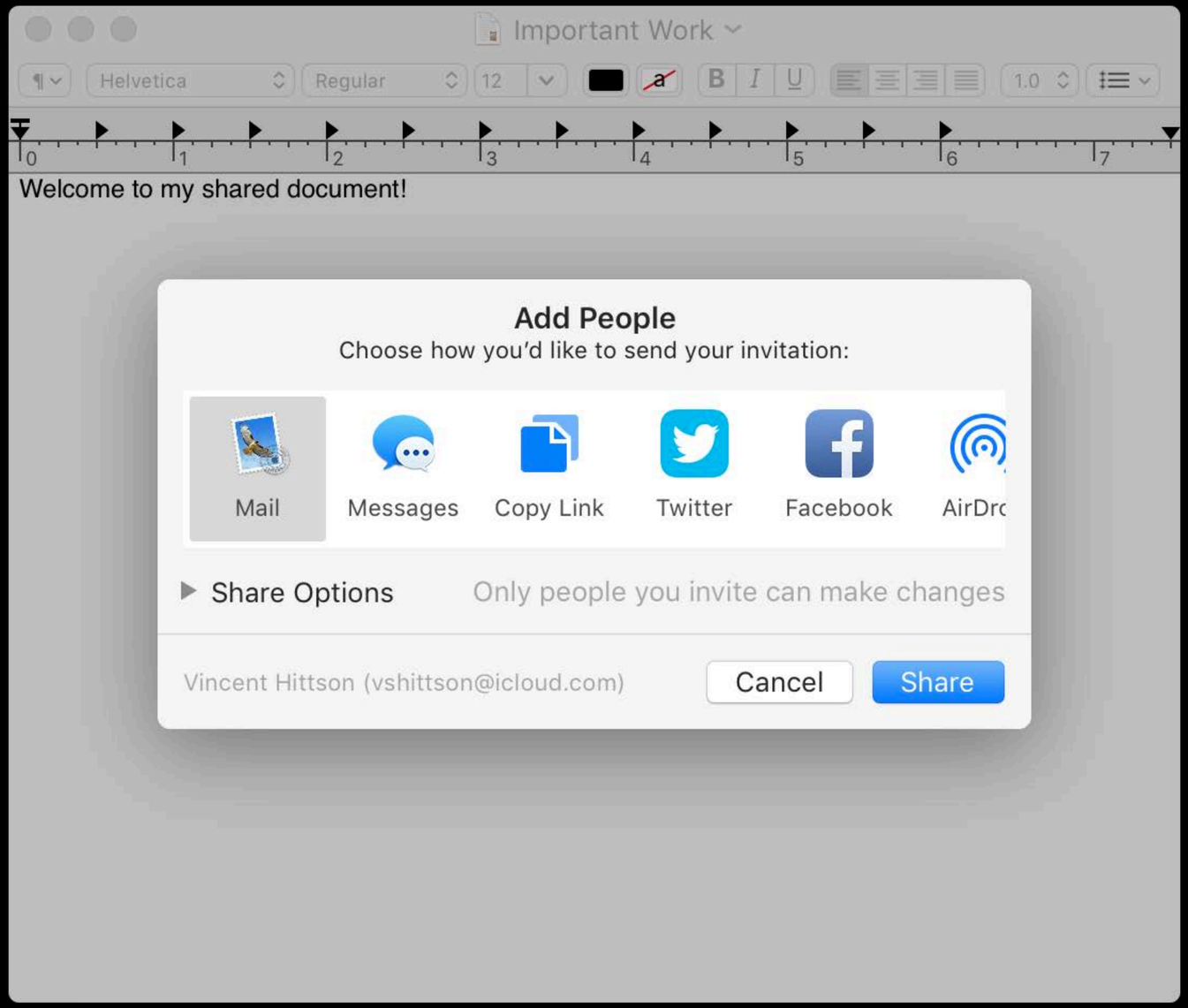


Document-based App



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: NSException)
}
```

```
open class NSApplication : NSResponder, … {
    open func reportException(_ exception: NSException)
}
```

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

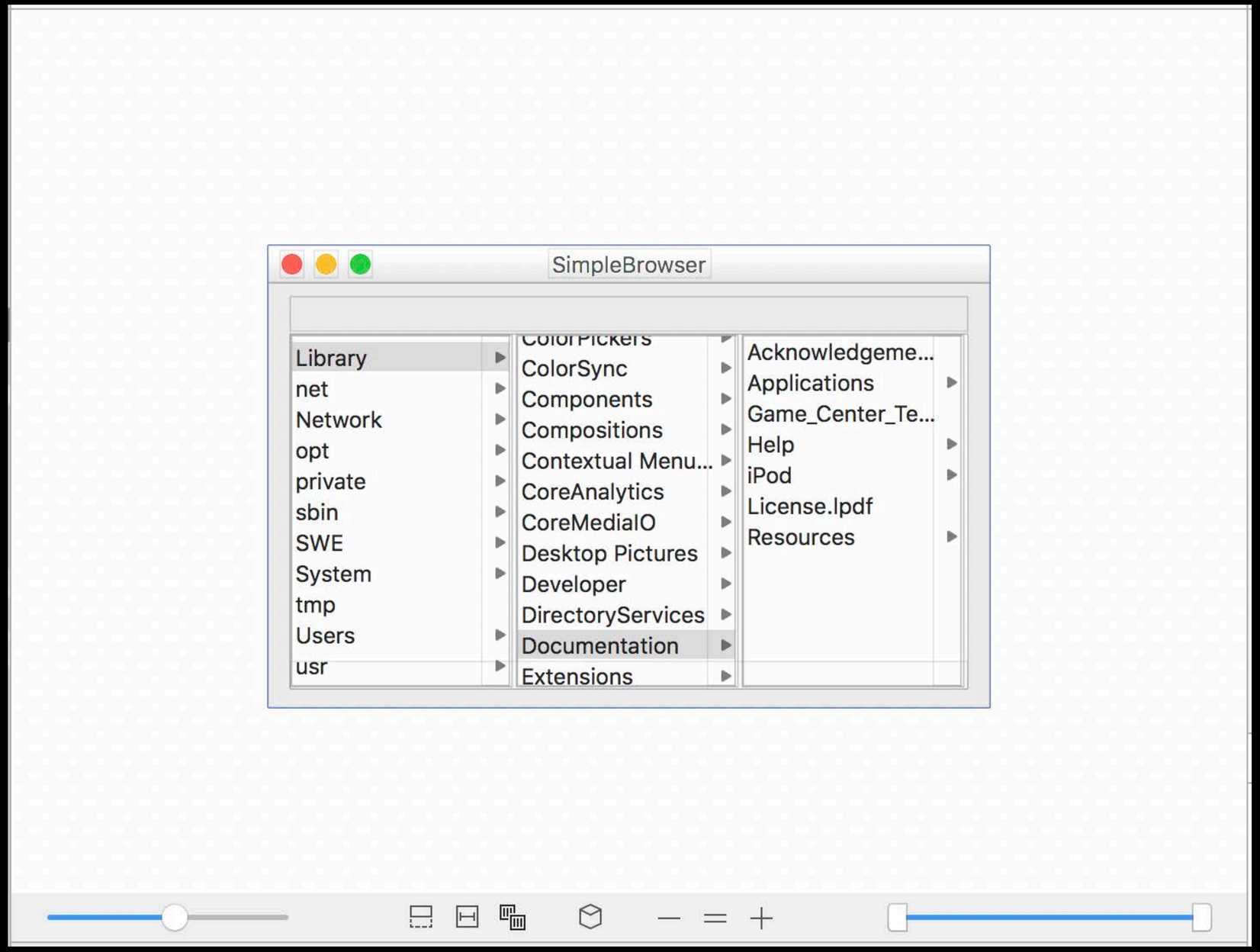
Simulate location

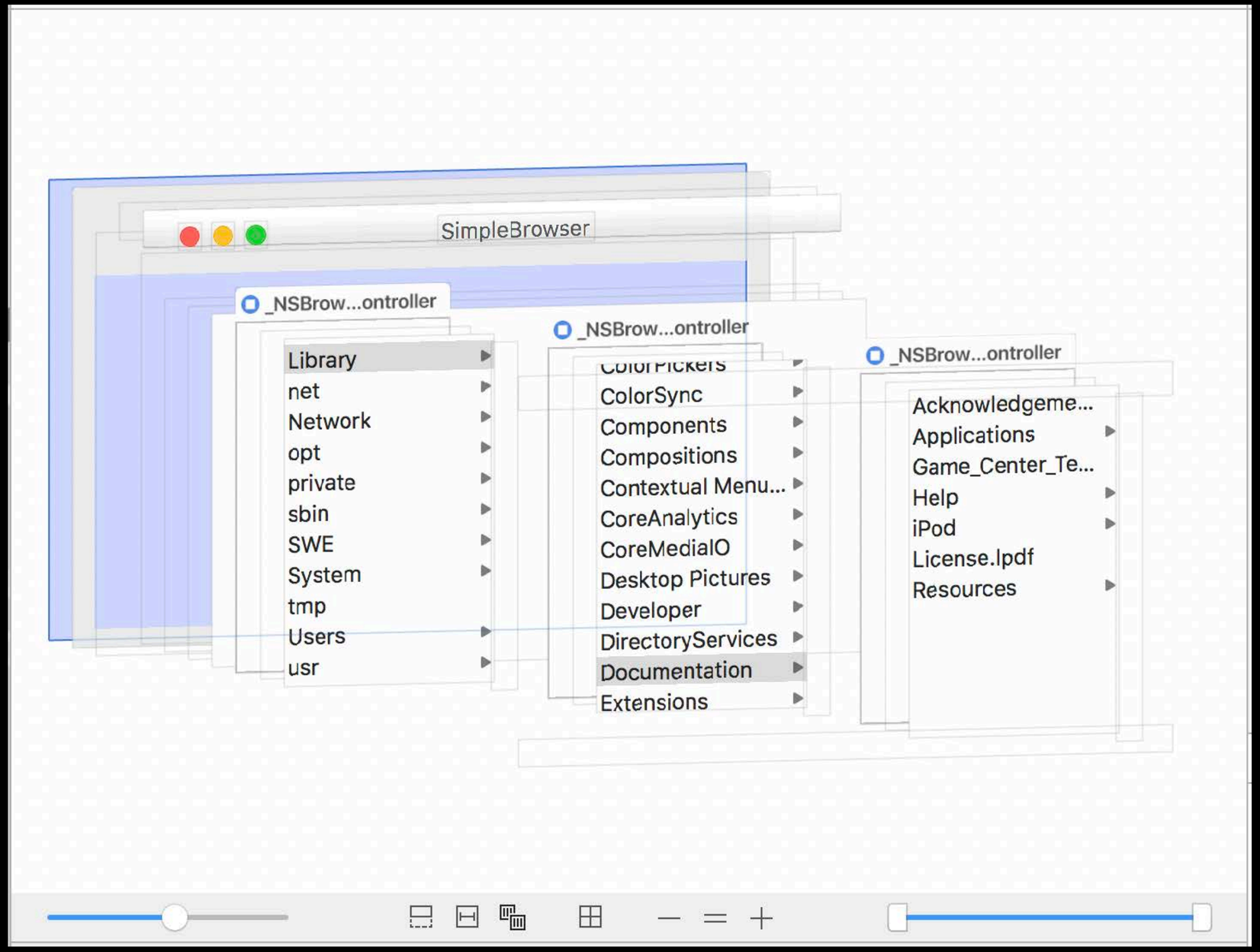
Xcode Debugging Modes

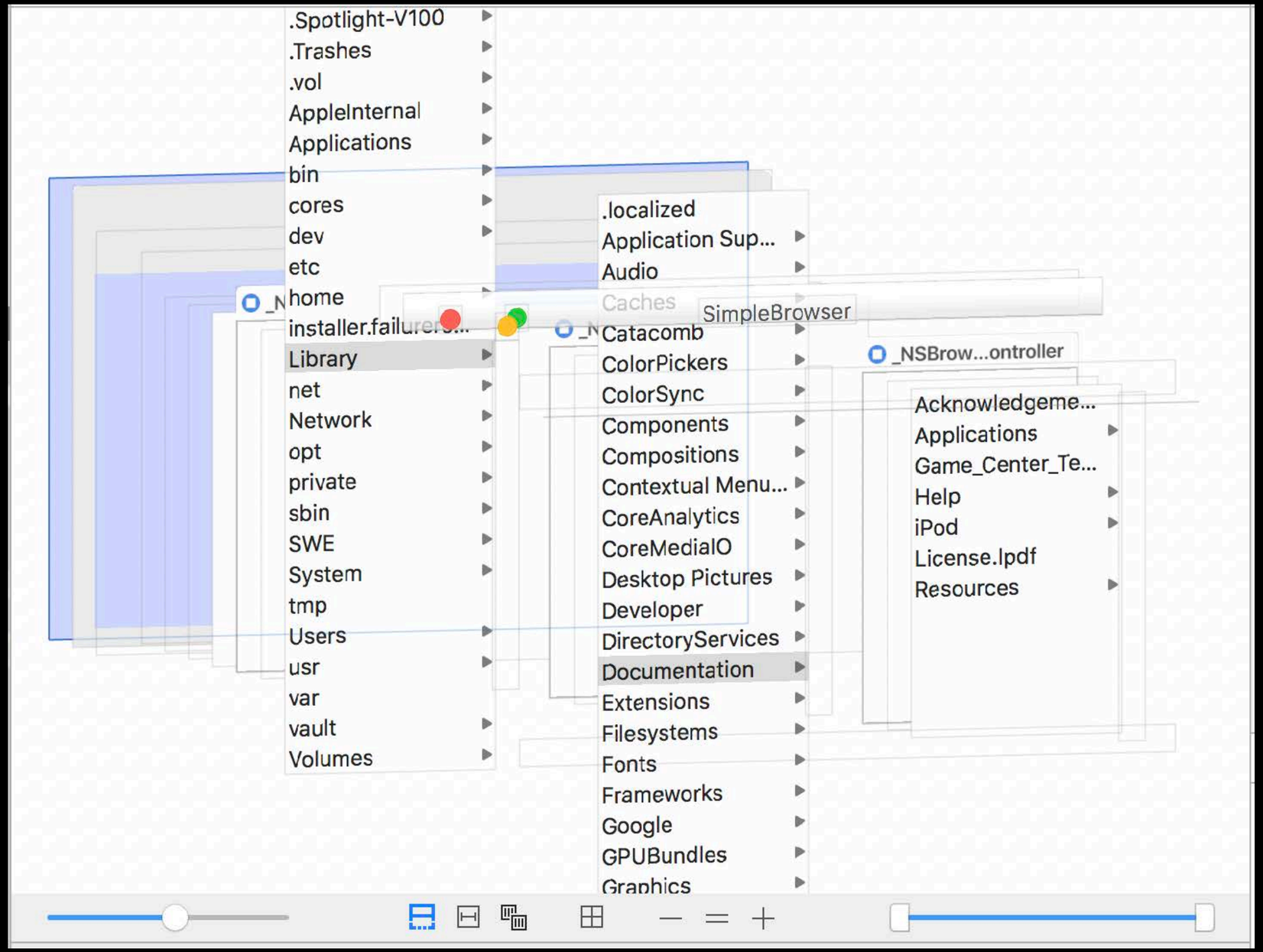
Debug view hierarchy

Debug memory graph

Simulate location


SimpleBrowser
Library
net
Network
opt
private
sbin
SWE
System
tmp
Users
usr
ColorPickers
ColorSync
Components
Compositions
Contextual Menu...
CoreAnalytics
CoreMediaIO
Desktop Pictures
Developer
DirectoryServices
Documentation
Extensions
Acknowledgeme...
Applications
Game_Center_Te...
Help
iPod
License.lpdf
Resources


SimpleBrowser
_NSBrow...ontroller
Library
net
Network
opt
private
sbin
SWE
System
tmp
Users
usr
_NSBrow...ontroller
ColorSync
Components
Compositions
Contextual Menu...
CoreAnalytics
CoreMediaIO
Desktop Pictures
Developer
DirectoryServices
Documentation
Extensions
_NSBrow...ontroller
Acknowledgeme...
Applications
Game_Center_Te...
Help
iPod
License.lpdf
Resources


.Spotlight-V100
.Trashes
.vol
AppleInternal
Applications
bin
cores
dev
etc
home
installer.failure...
Library
net
Network
opt
private
sbin
SWE
System
tmp
Users
usr
var
vault
Volumes
.localized
Application Sup...
Audio
Caches
Catacomb
ColorPickers
ColorSync
Components
Compositions
Contextual Menu...
CoreAnalytics
CoreMediaIO
Desktop Pictures
Developer
DirectoryServices
Documentation
Extensions
Filesystems
Fonts
Frameworks
Google
GPUBundles
Graphics
SimpleBrowser
_NSBrow...ontroller
Acknowledgeme...
Applications
Game_Center_Te...
Help
iPod
License.lpdf
Resources

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

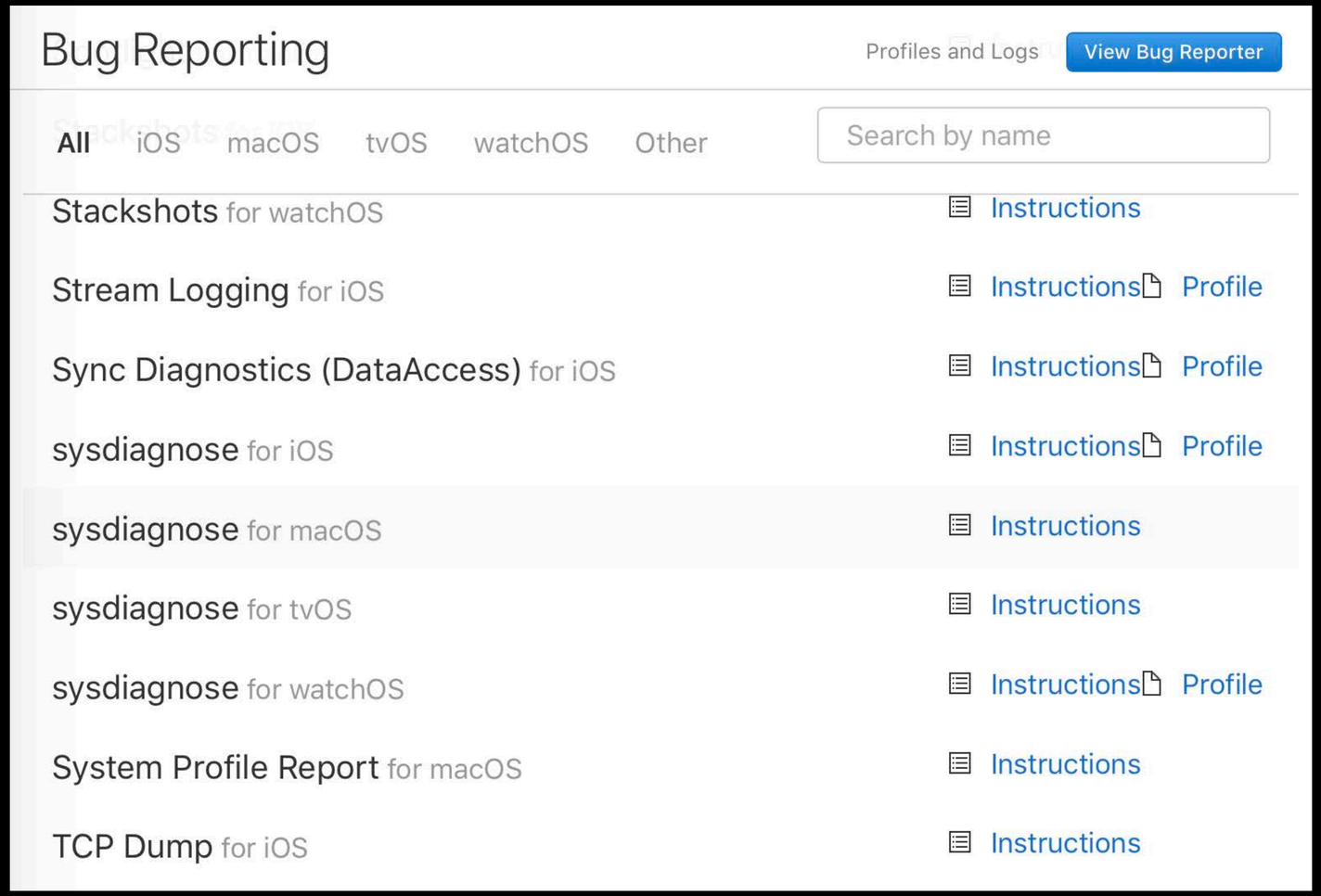
Include any resources (image, database, etc.) that might be needed

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/



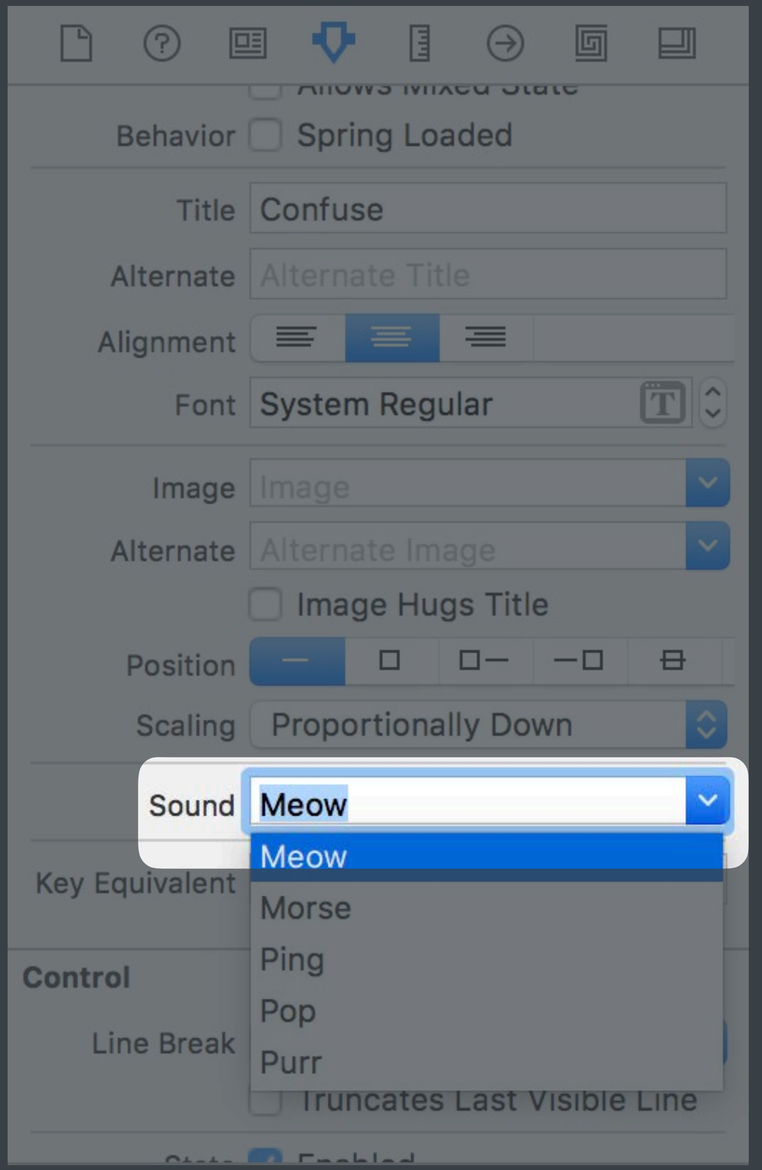
44.1

Bells and Whistles

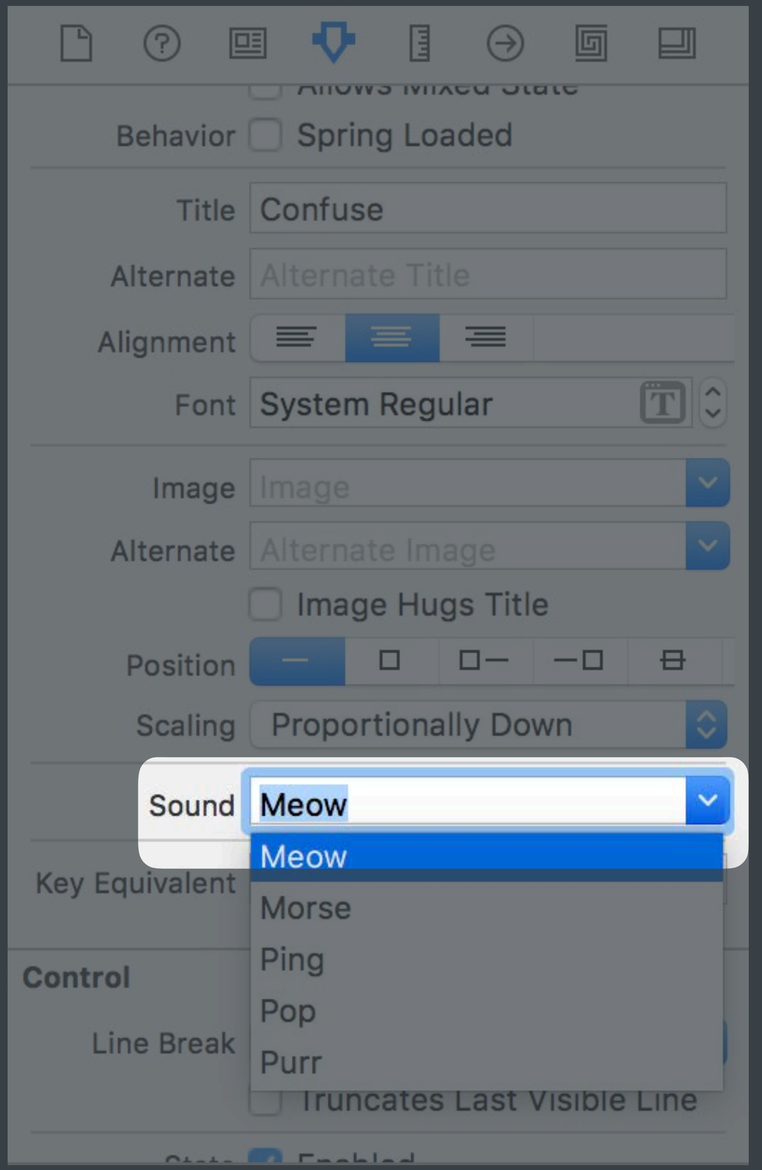
```
extension NSButton {
    open var sound: NSSound?
}
```

```
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:

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.

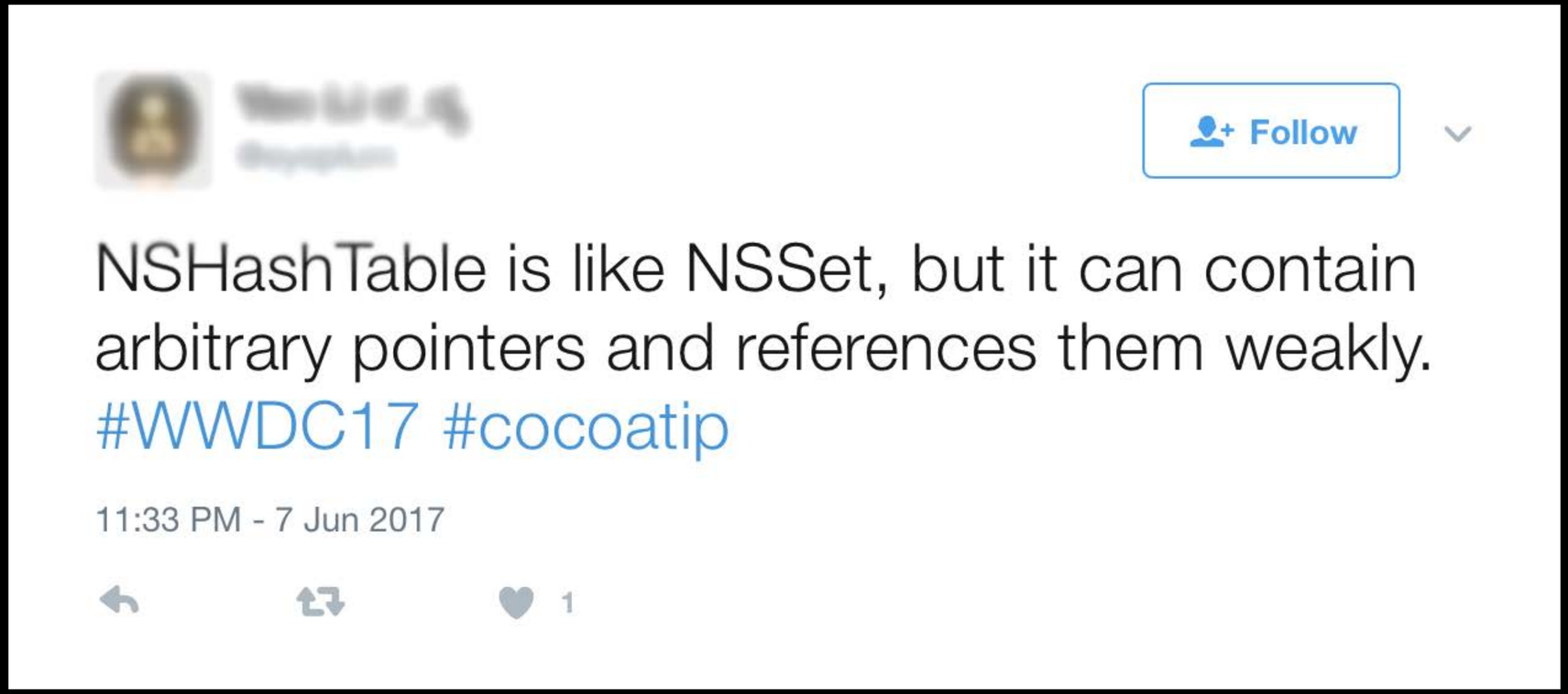
Some of the major topics covered in this document:

- NSCollectionView Responsive Scrolling
- New Enumerations
- Accessibility
- NSLevelIndicator
- Layer-backed Views
- NSTableView Automatic Row Heights

```
/*
        NSImage.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.
+ (NSImage *)imageWithSize:(NSSize)size flipped:
(BOOL)drawingHandlerShouldBeCalledWithFlippedContext drawingHandler:
                          (BOOL (^)(NSRect dstRect))drawingHandler NS_AVAILABLE_MAC(10_8);

...
```

Tweet Your Tips!

Use #WWDC17 and #cocoatip


Follow
NSHashTable is like NSSet, but it can contain arbitrary pointers and references them weakly. #WWDC17 #cocoatip
11:33 PM - 7 Jun 2017
1

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"]
```

[redacted]

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

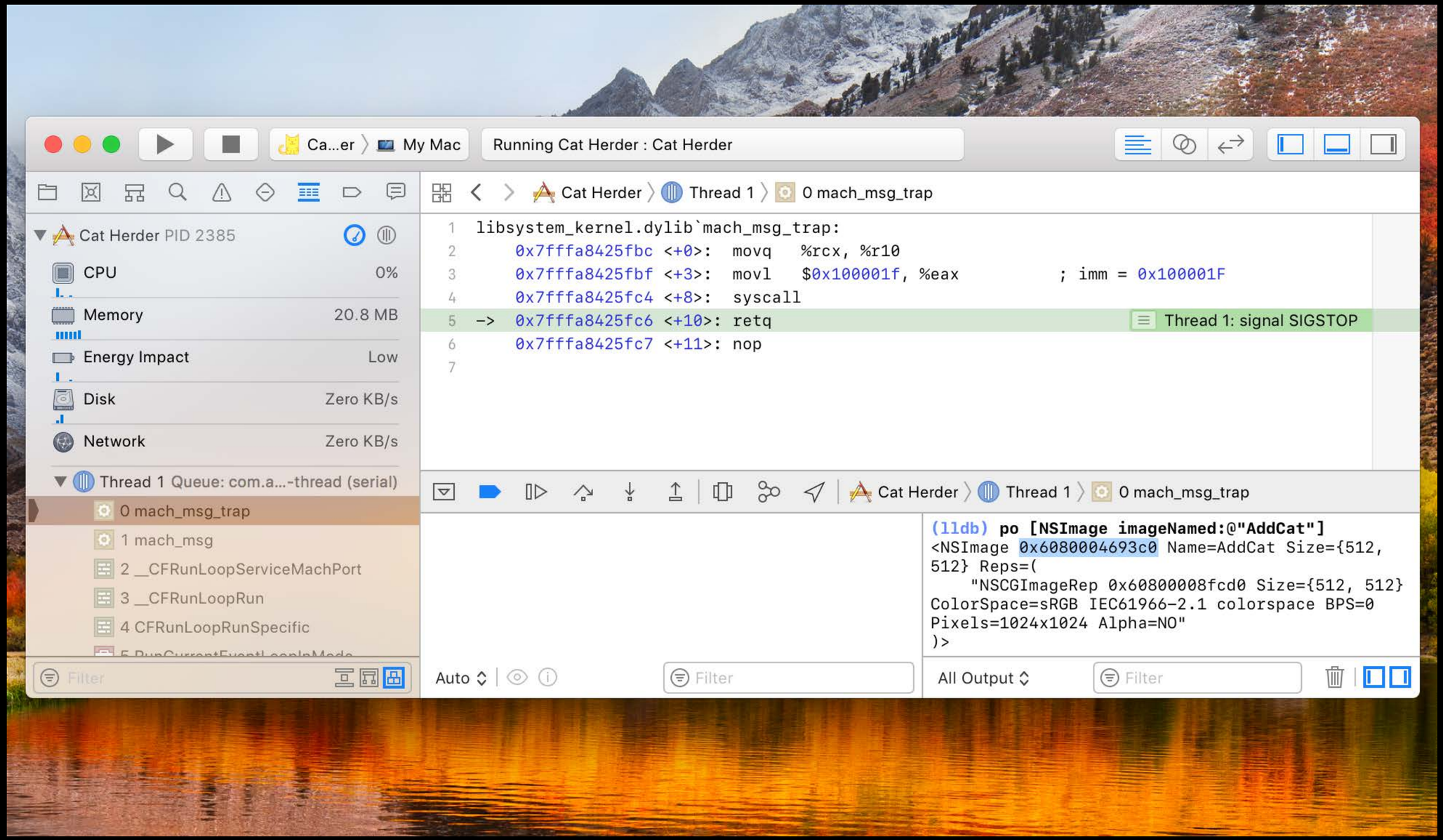
Follow

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

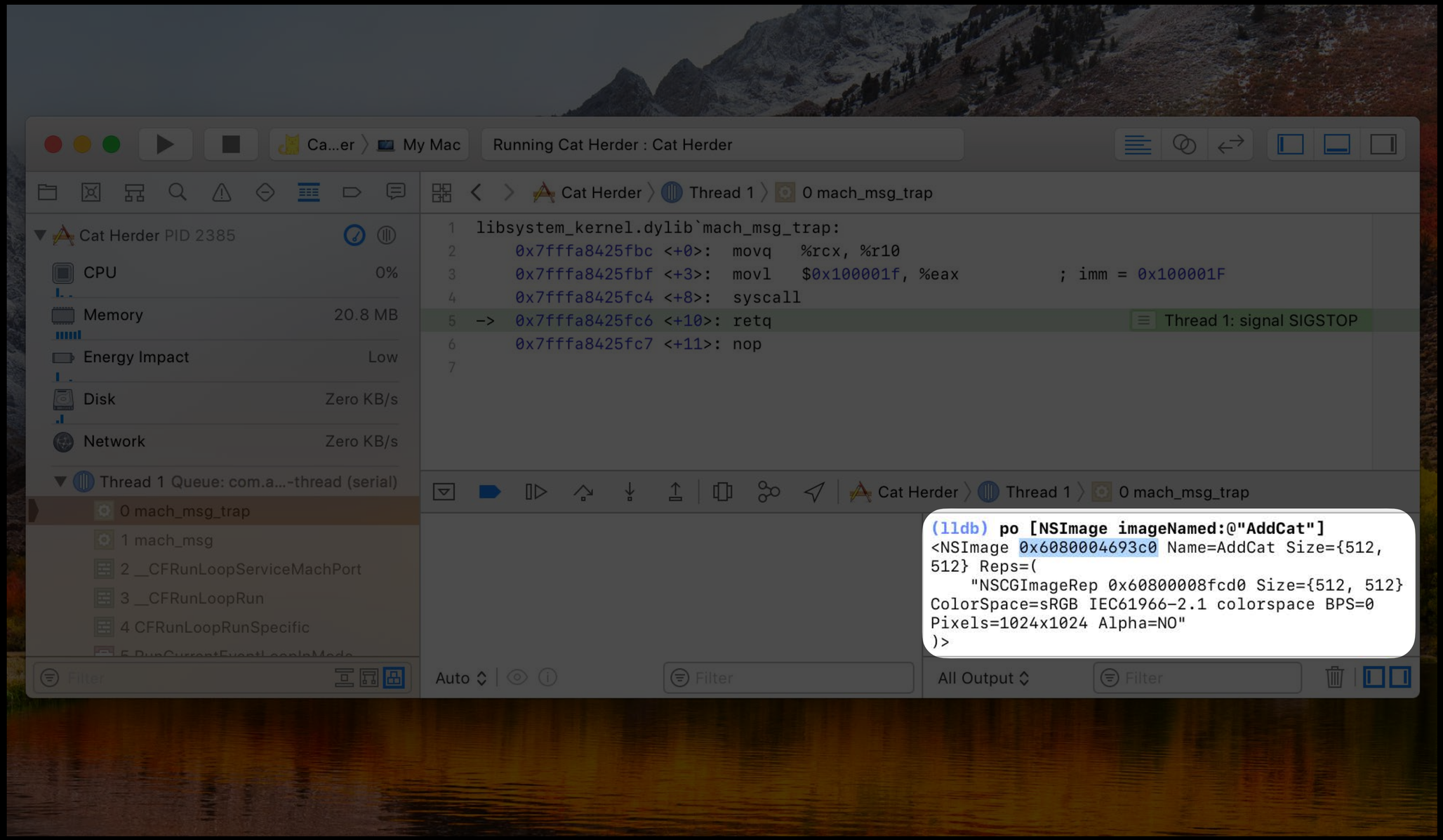
8:07 AM - 8 Jun 2017

3 6

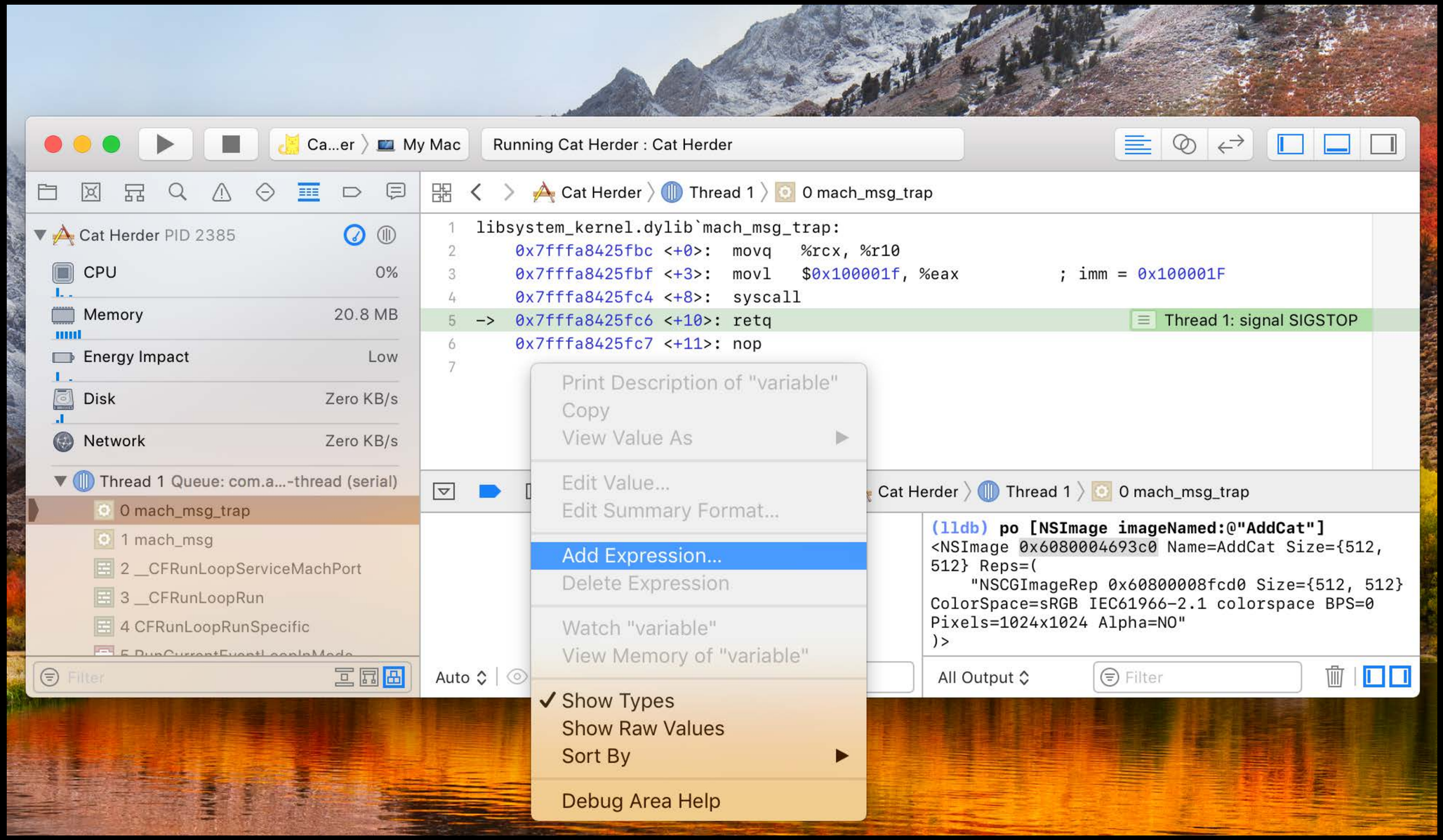
Quick Look Expression



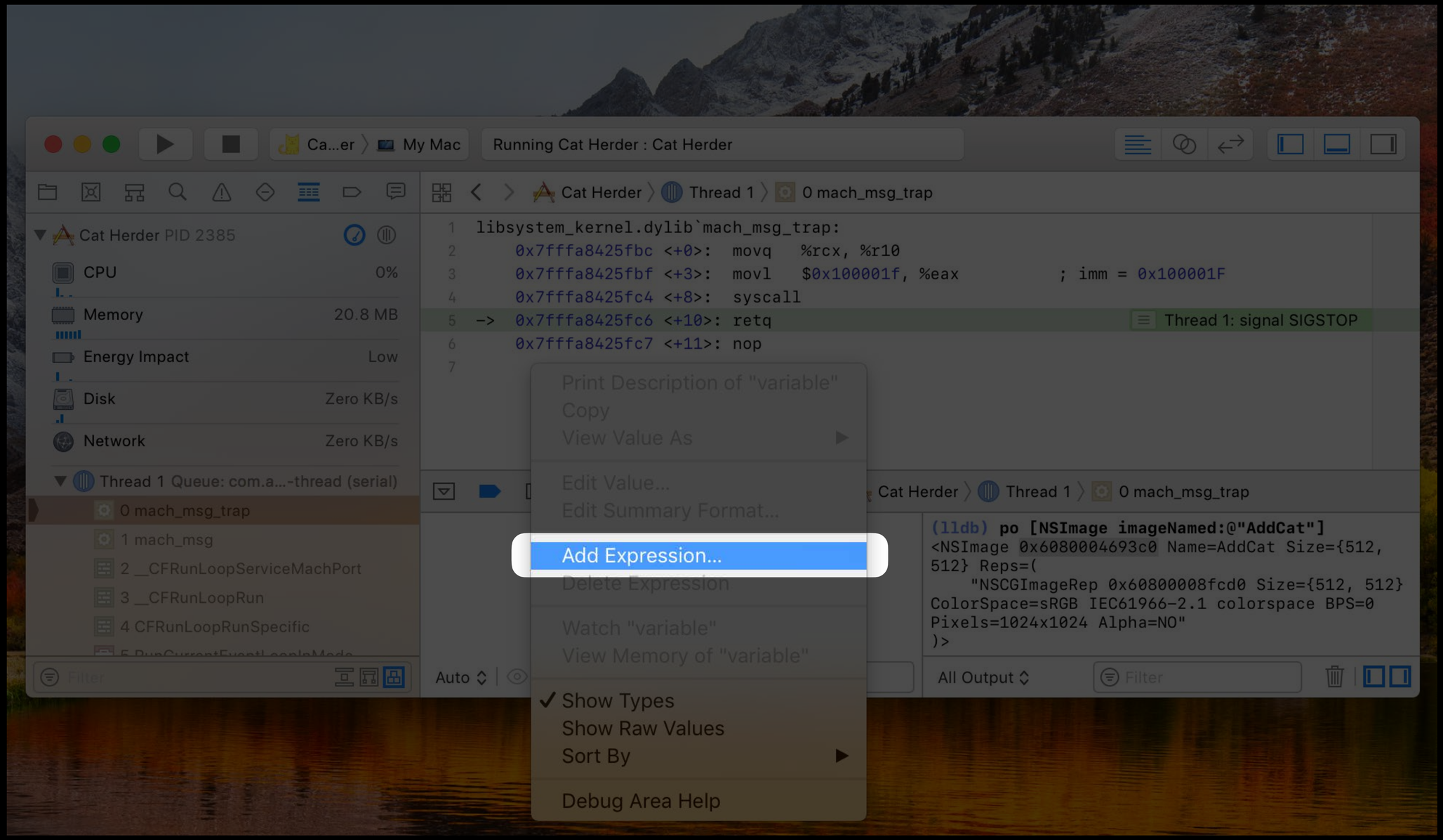
Quick Look Expression



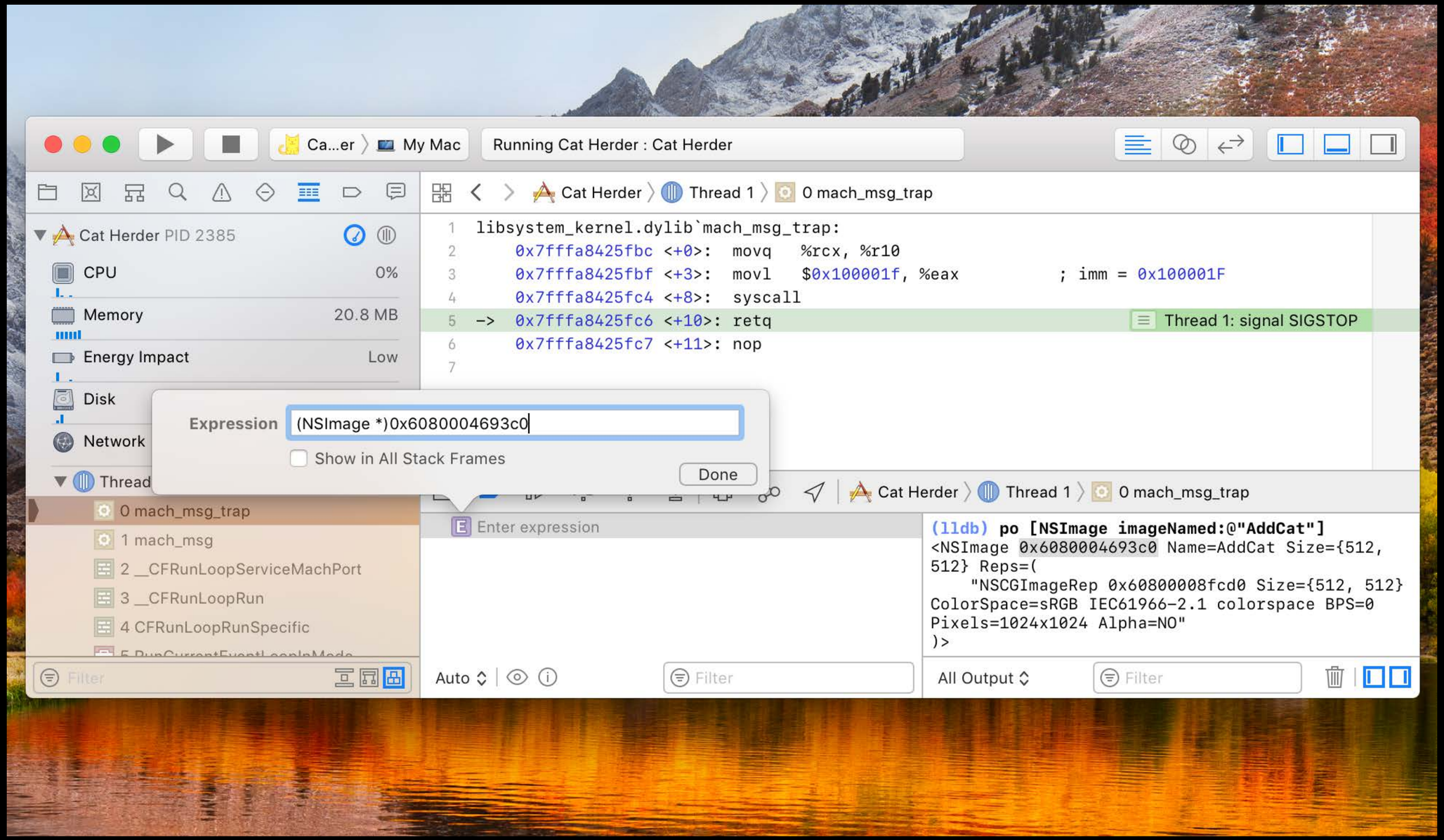
Quick Look Expression



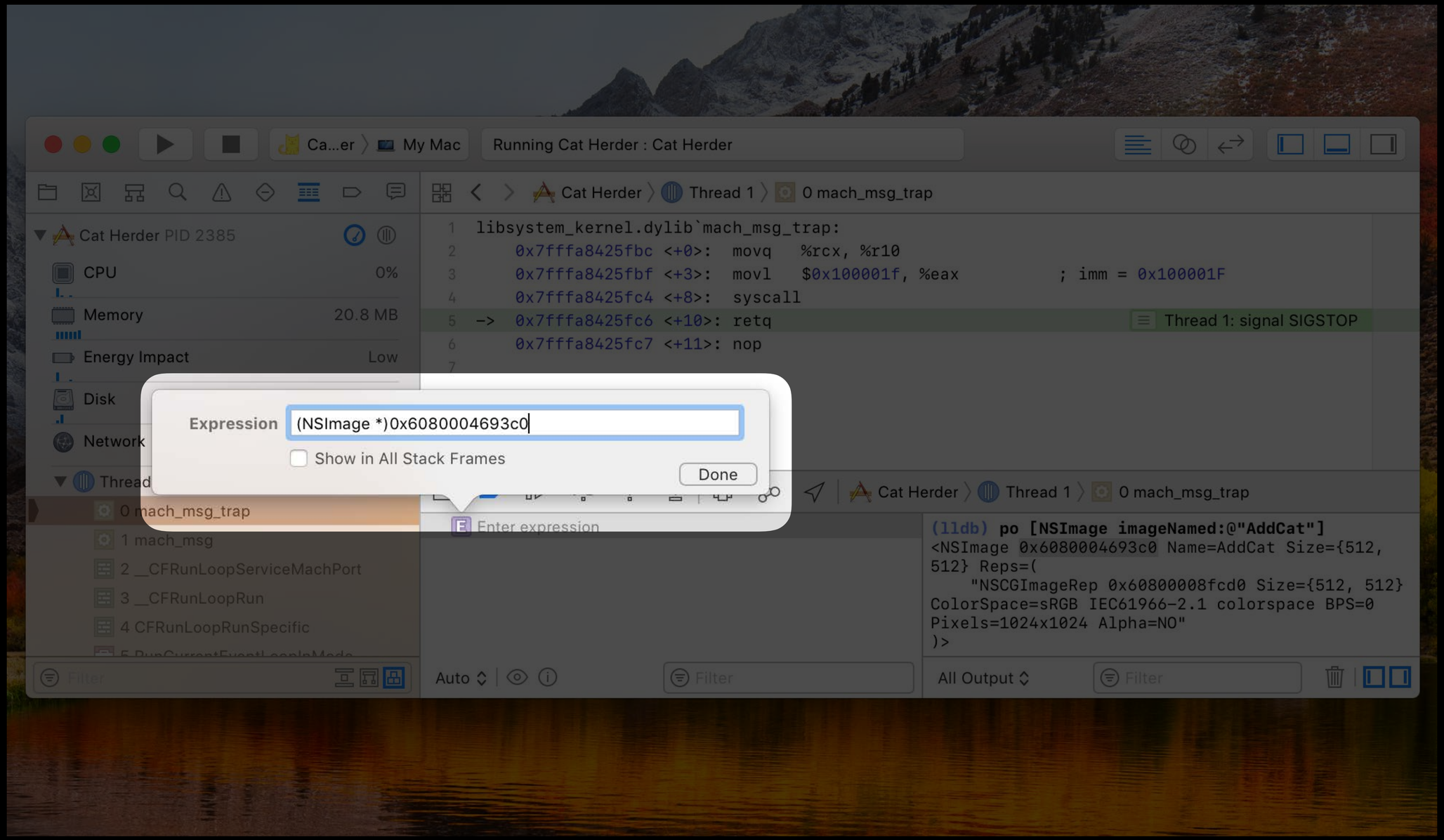
Quick Look Expression



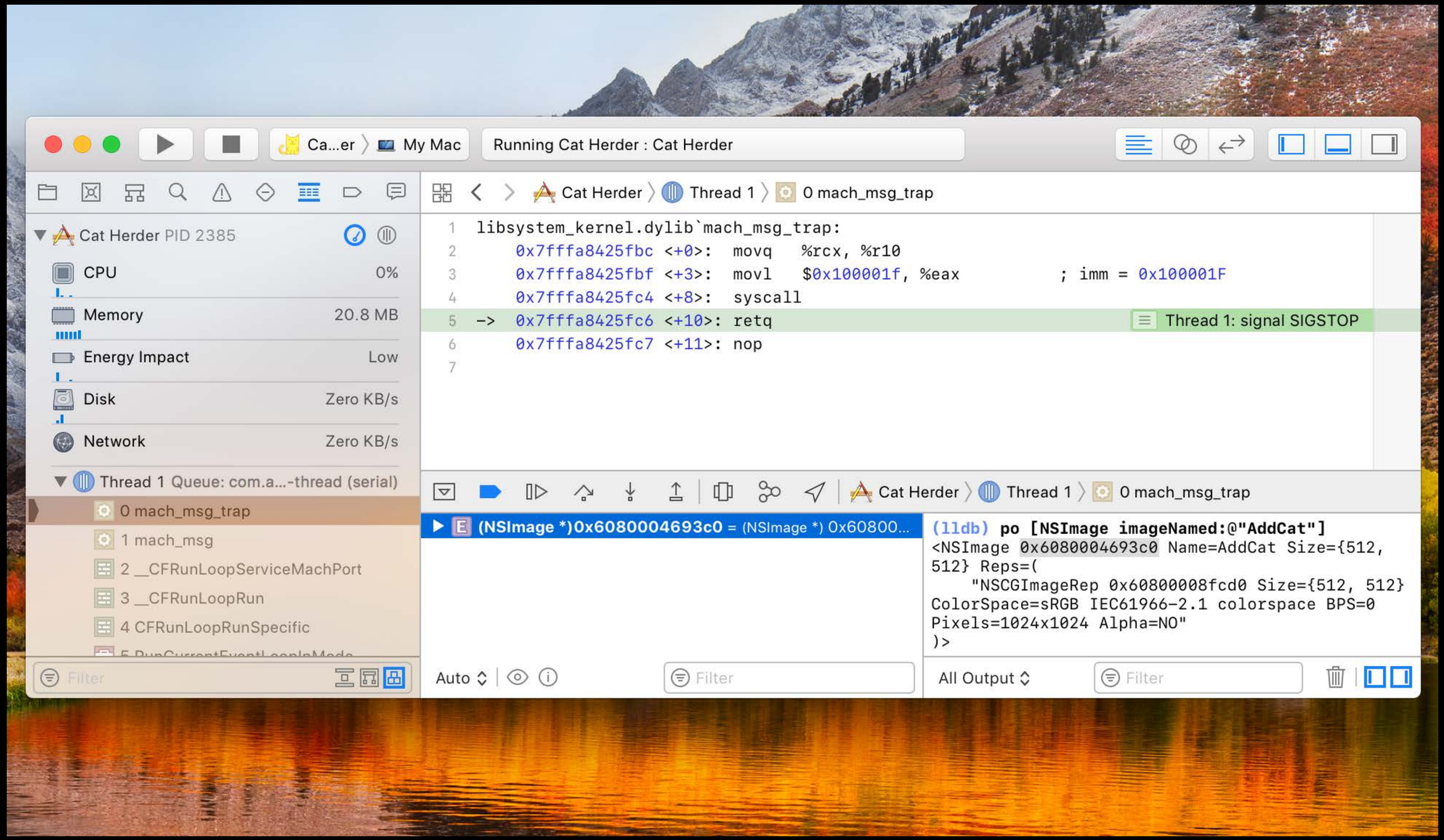
Quick Look Expression



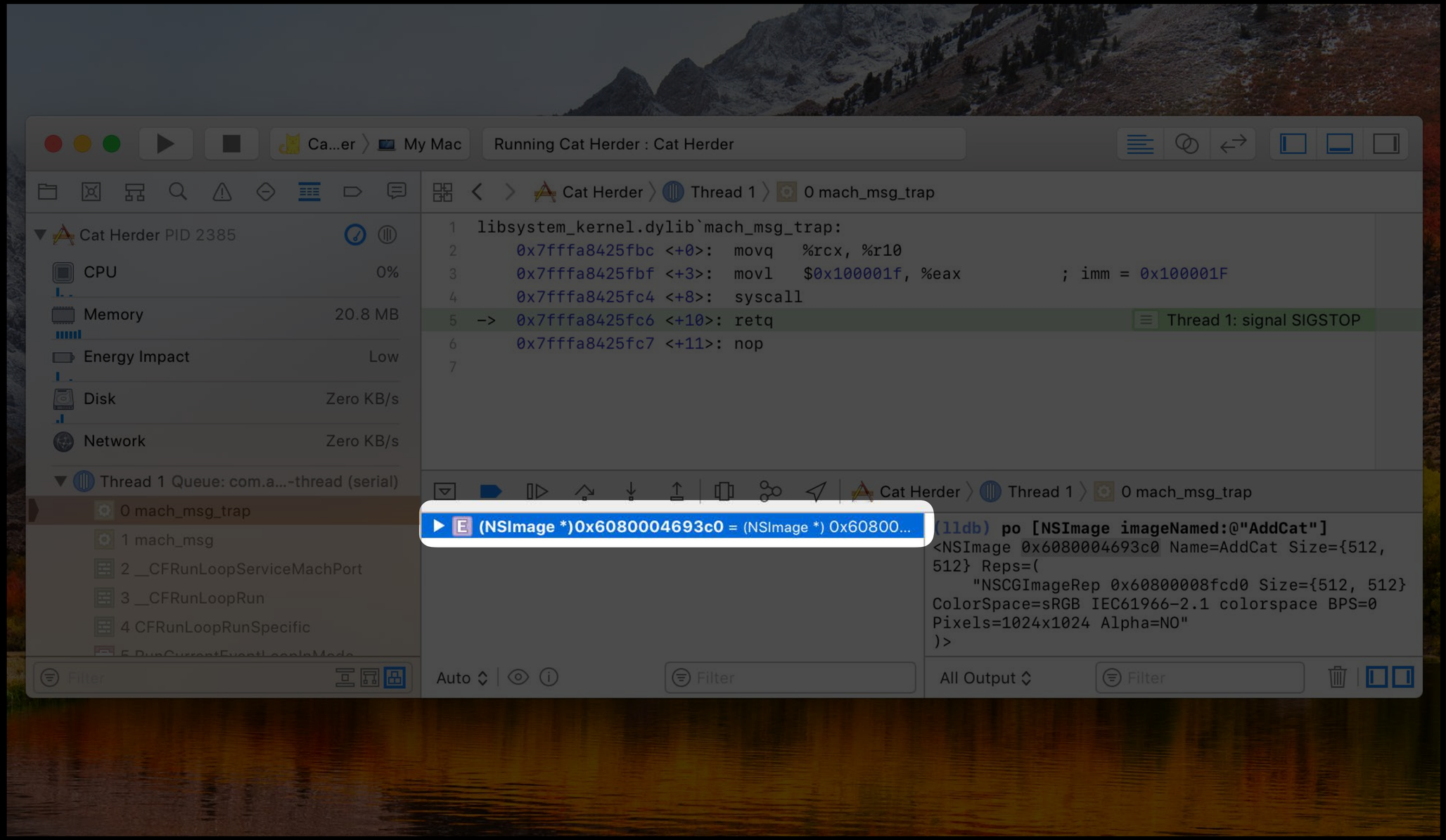
Quick Look Expression



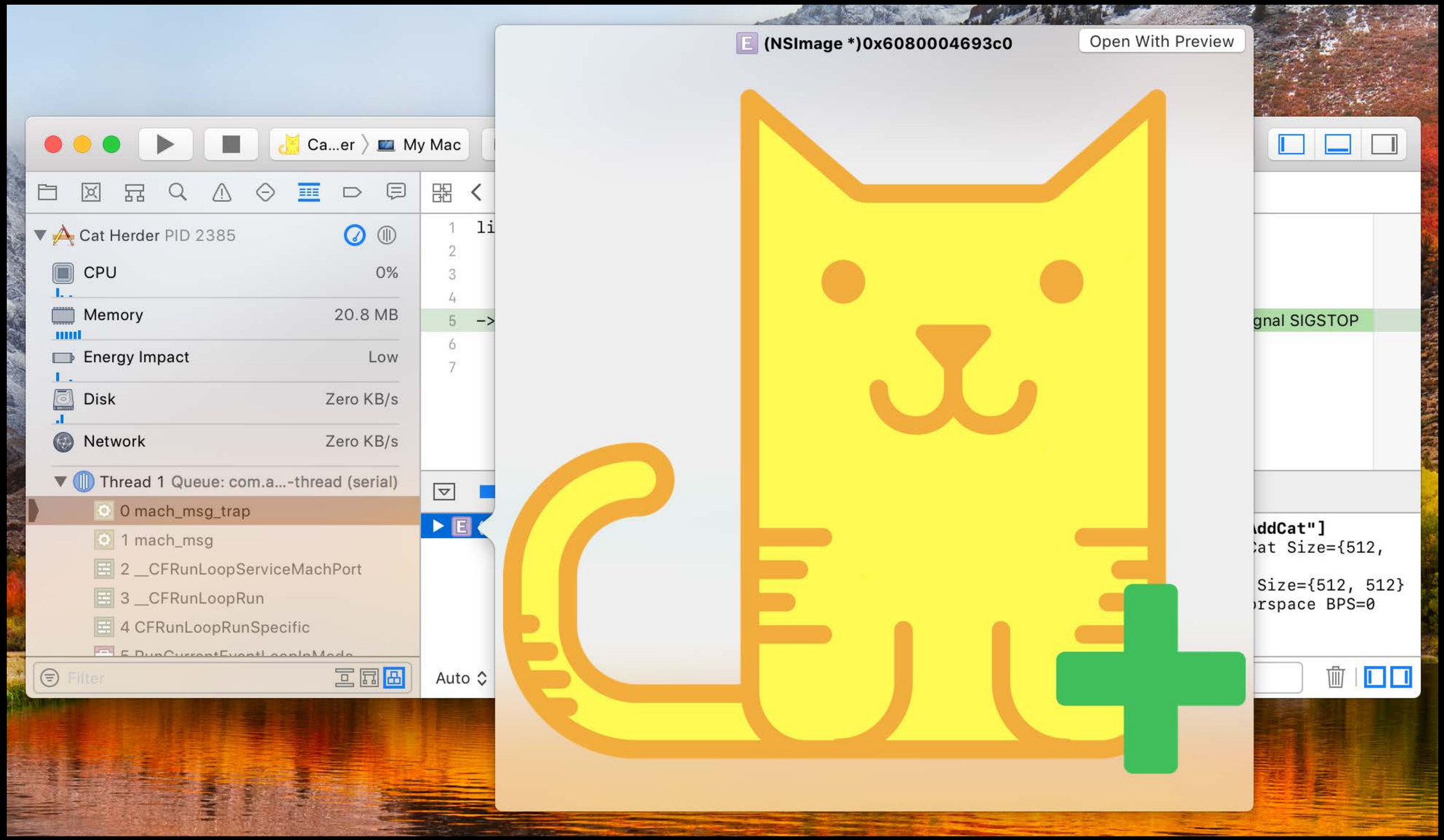
Quick Look Expression



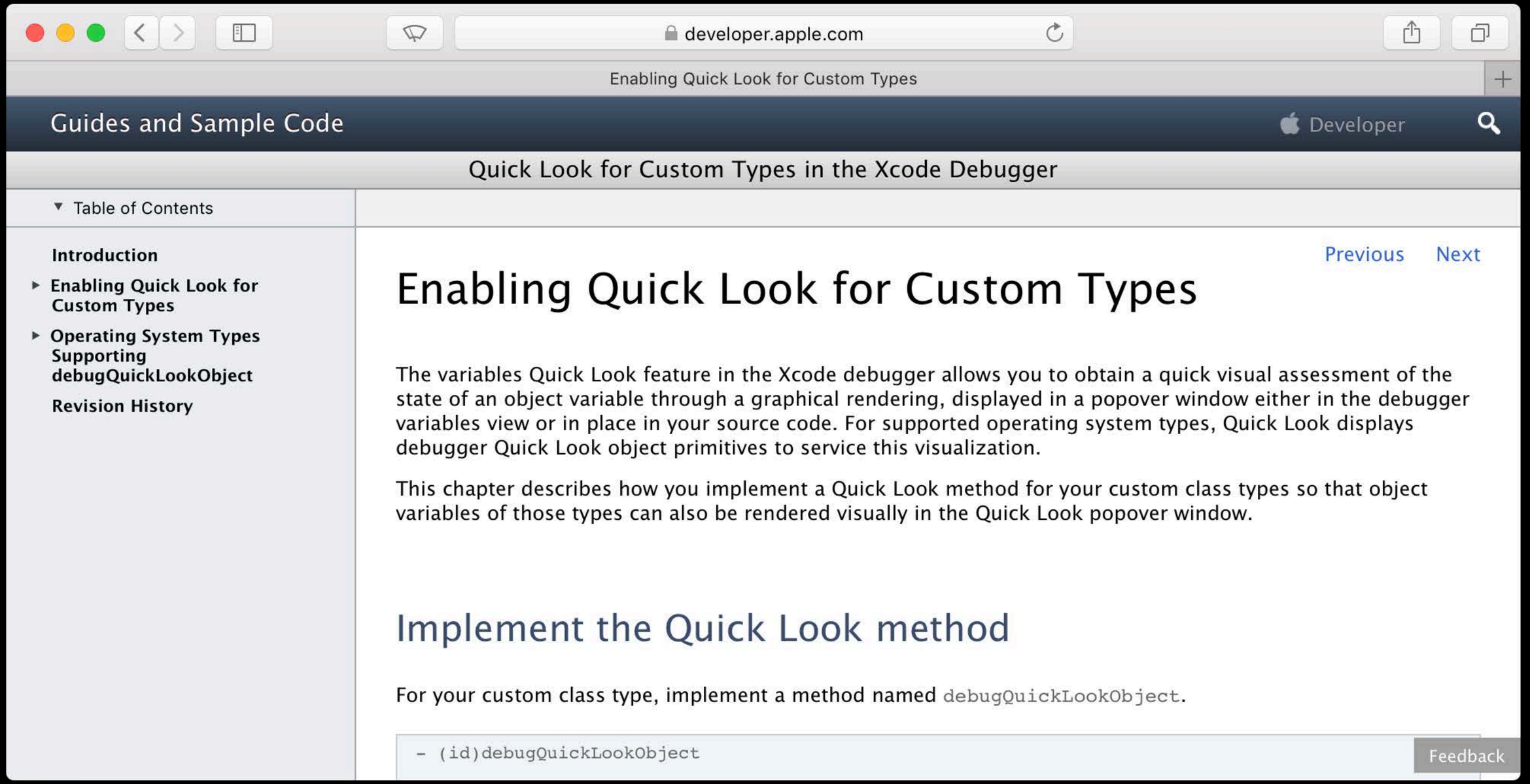
Quick Look Expression



Quick Look Expression



Quick Look Expression



More Information

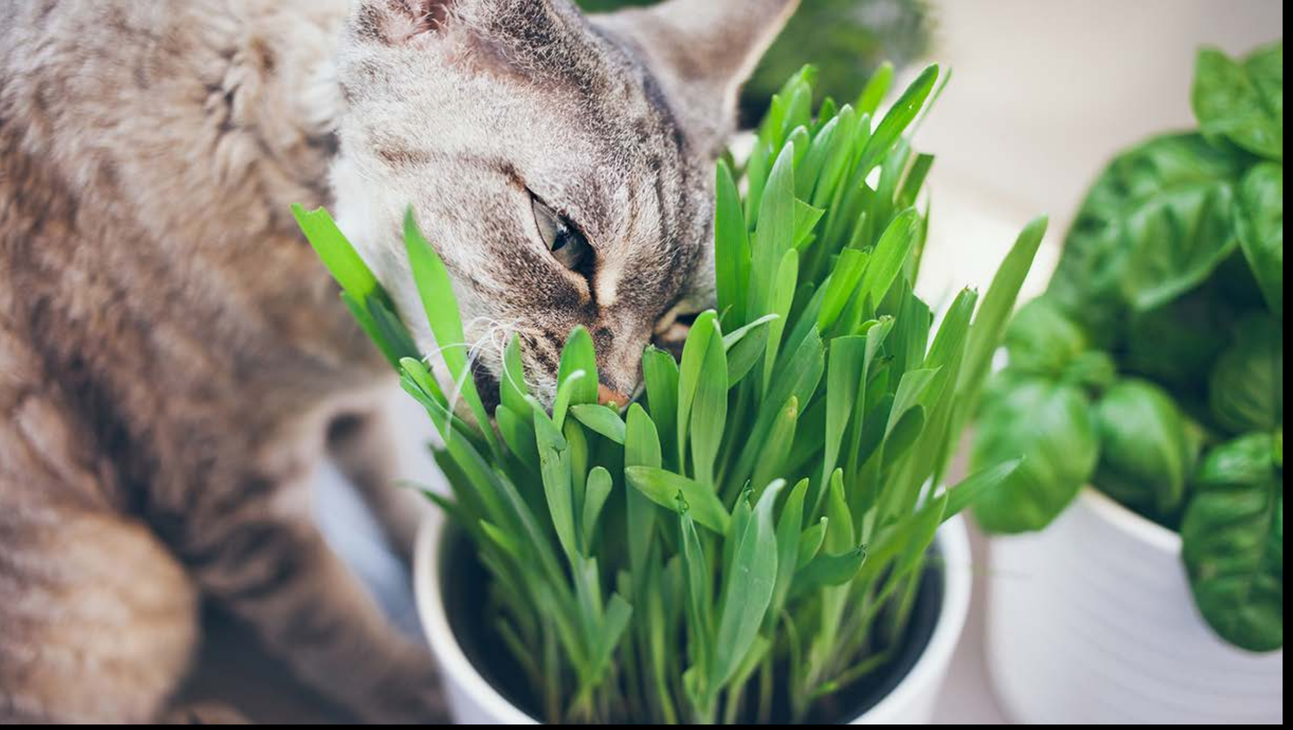
https://developer.apple.com/wwdc17/236

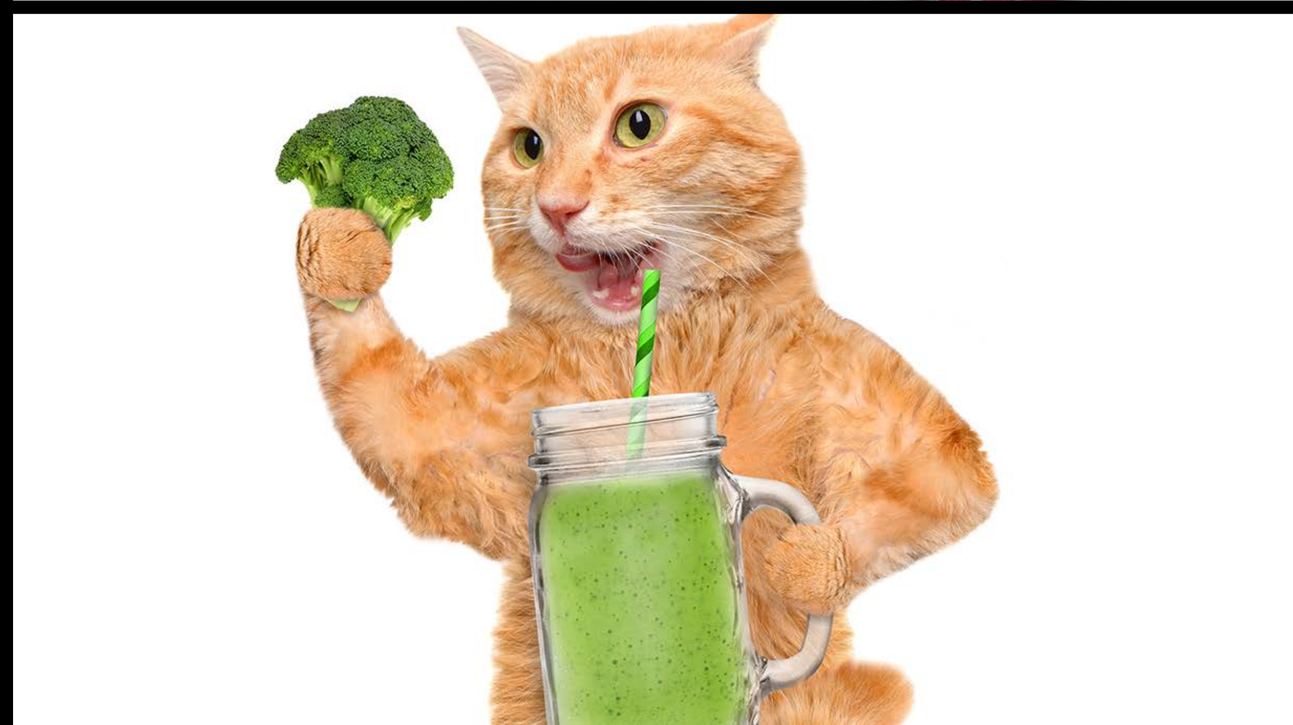
Related Sessions

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






WWDC17