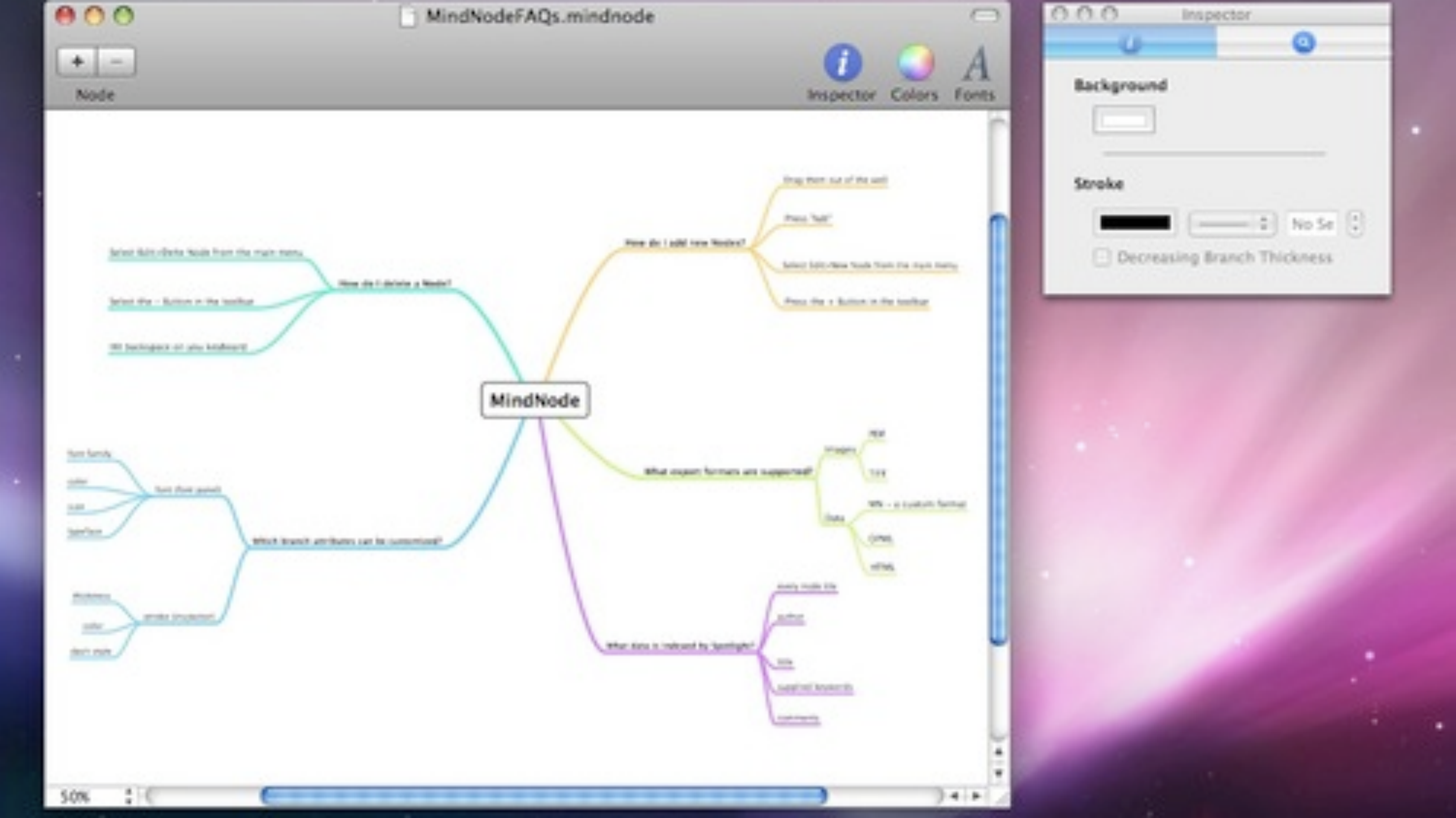
Sharing Code
between *iOS* & *macOS*

Matthias Tretter, @myell0w

March 15th,
2008


MindNodeFAQs.mindnode
Node
Inspector
Colors
Fonts
MindNode
How do I delete a Node?
How do I add new Nodes?
What export formats are supported?
Which branch attributes can be customized?
What does is indexed by Spotlight?
50%
Inspector
Background
Stroke
Decreasing Branch Thickness

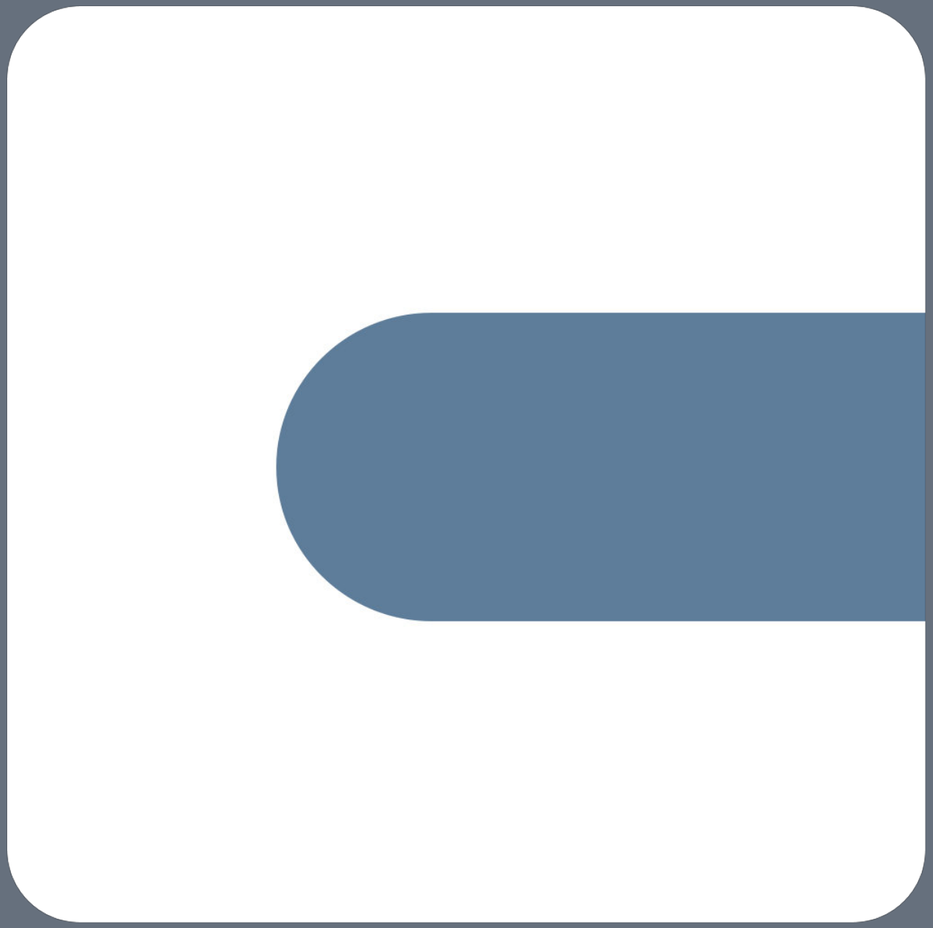
7

Back to the Mac

Error: Too many errors

Seriously, the Mac?

Isn't the Mac... dead?

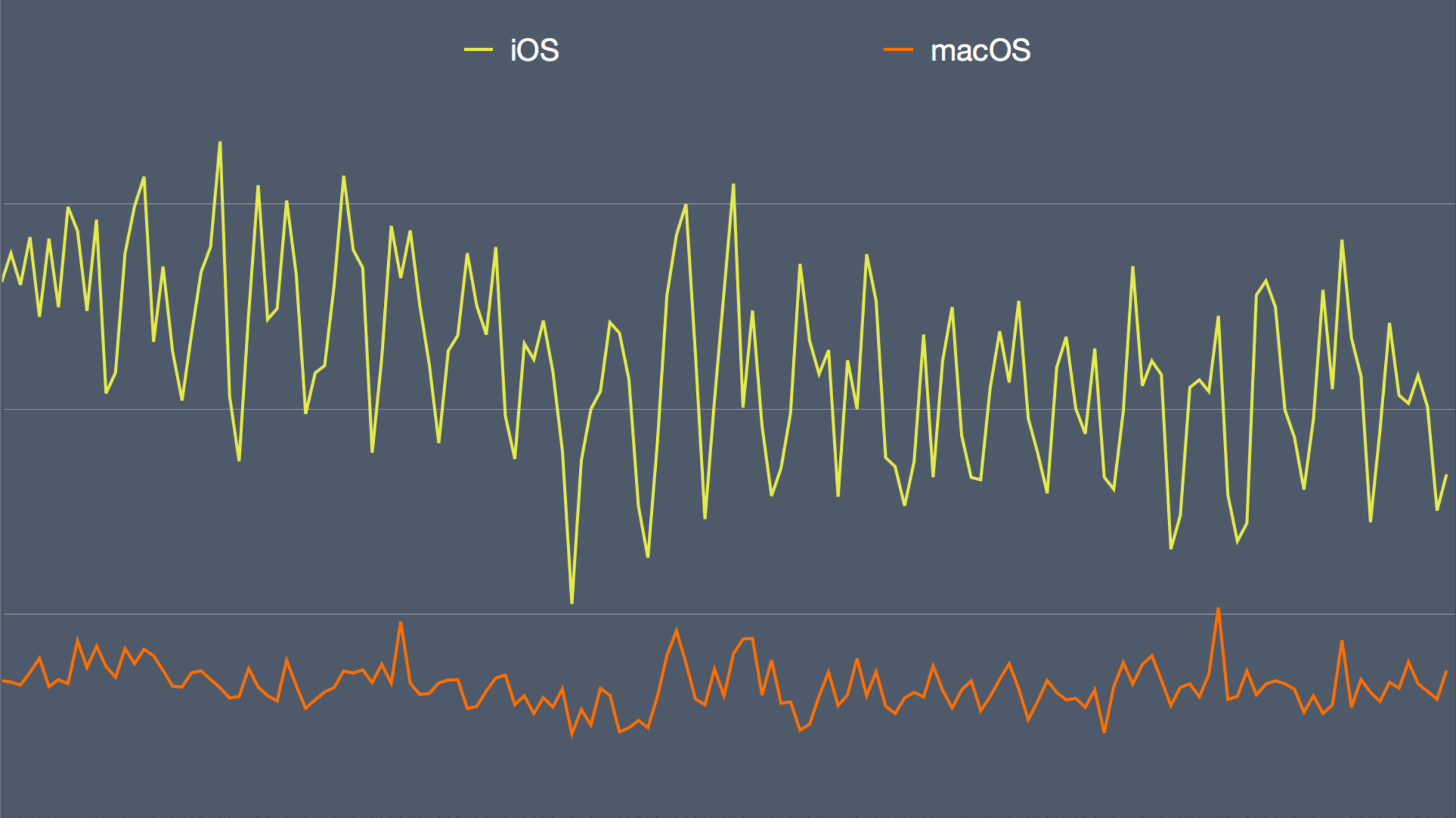


“The #1 reason people didn’t buy Vesper is because it wasn’t available for the Mac“

— John Gruber

“We would immediately design Vesper for Mac. And that’s the product we’d have shipped first“

— John Gruber


iOS
macOS

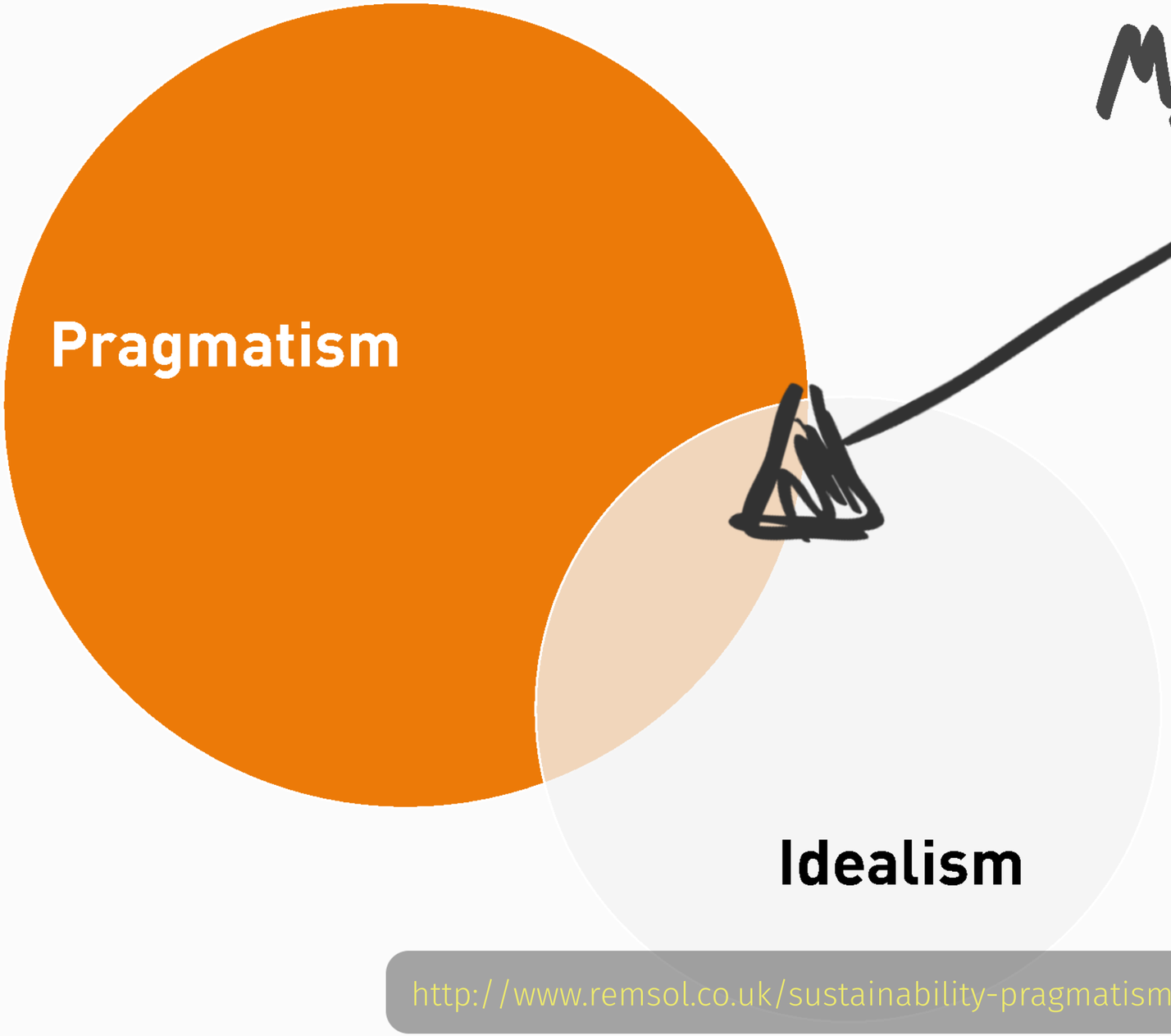
Why?

→ Testability ✅

→ Single Source of Truth 👩‍⚕️

→ Efficiency 🚀

WHERE THE MAGIC HAPPENS



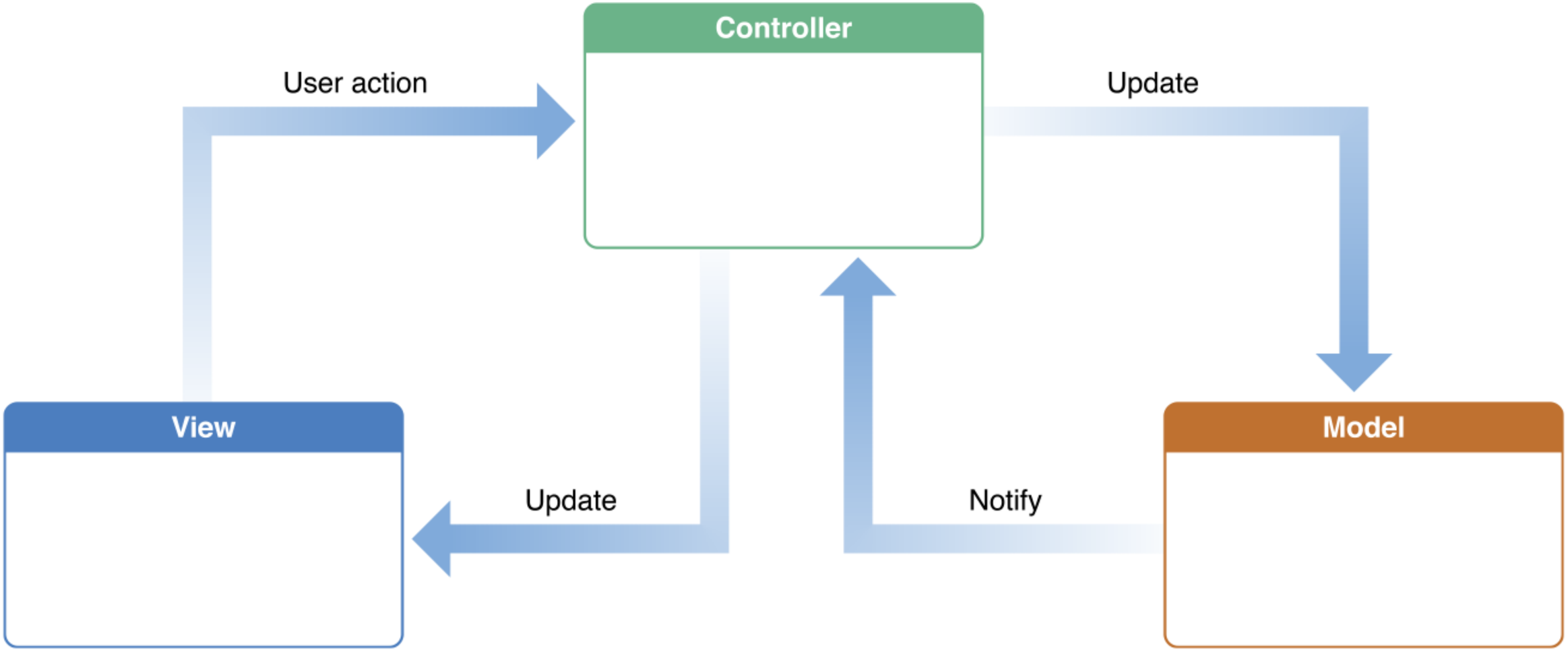
http://www.remsol.co.uk/sustainability-pragmatism-vs-idealism/

✌ Tip #1

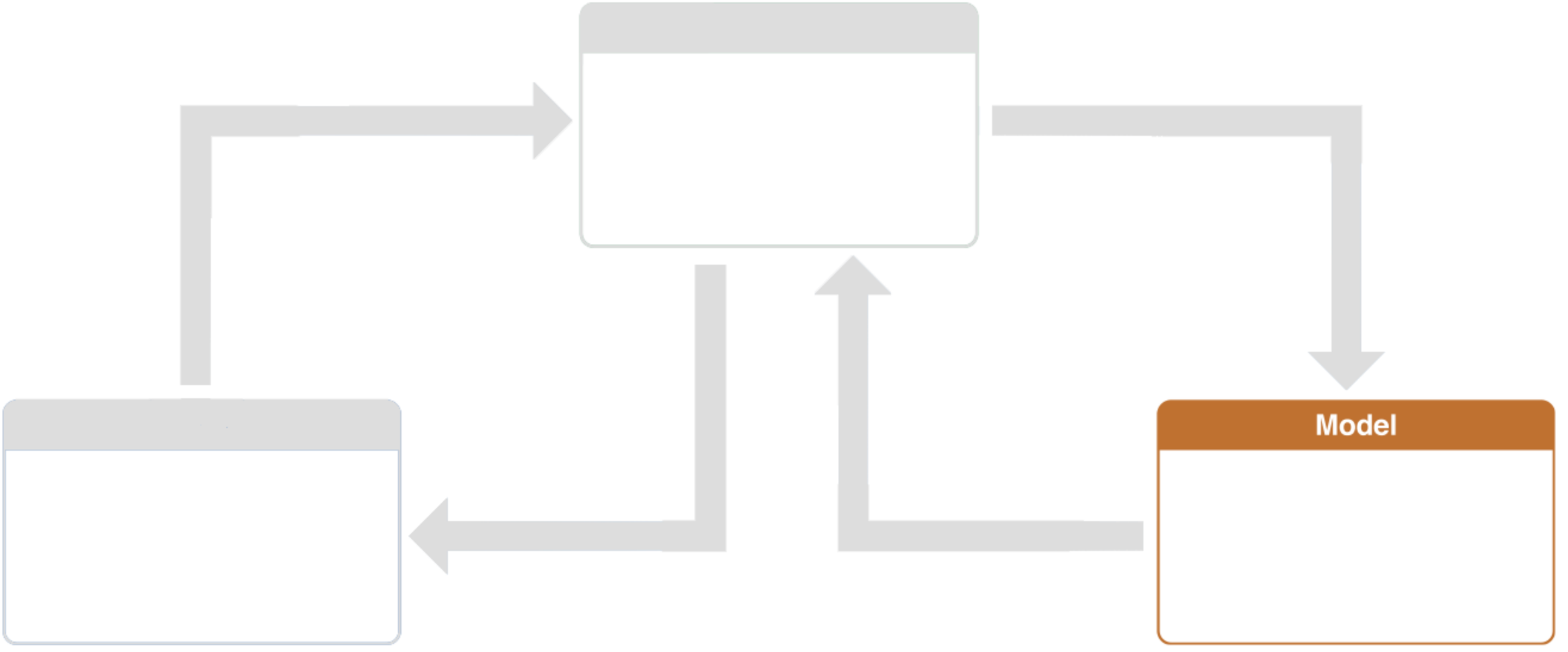
be *pragmatic*

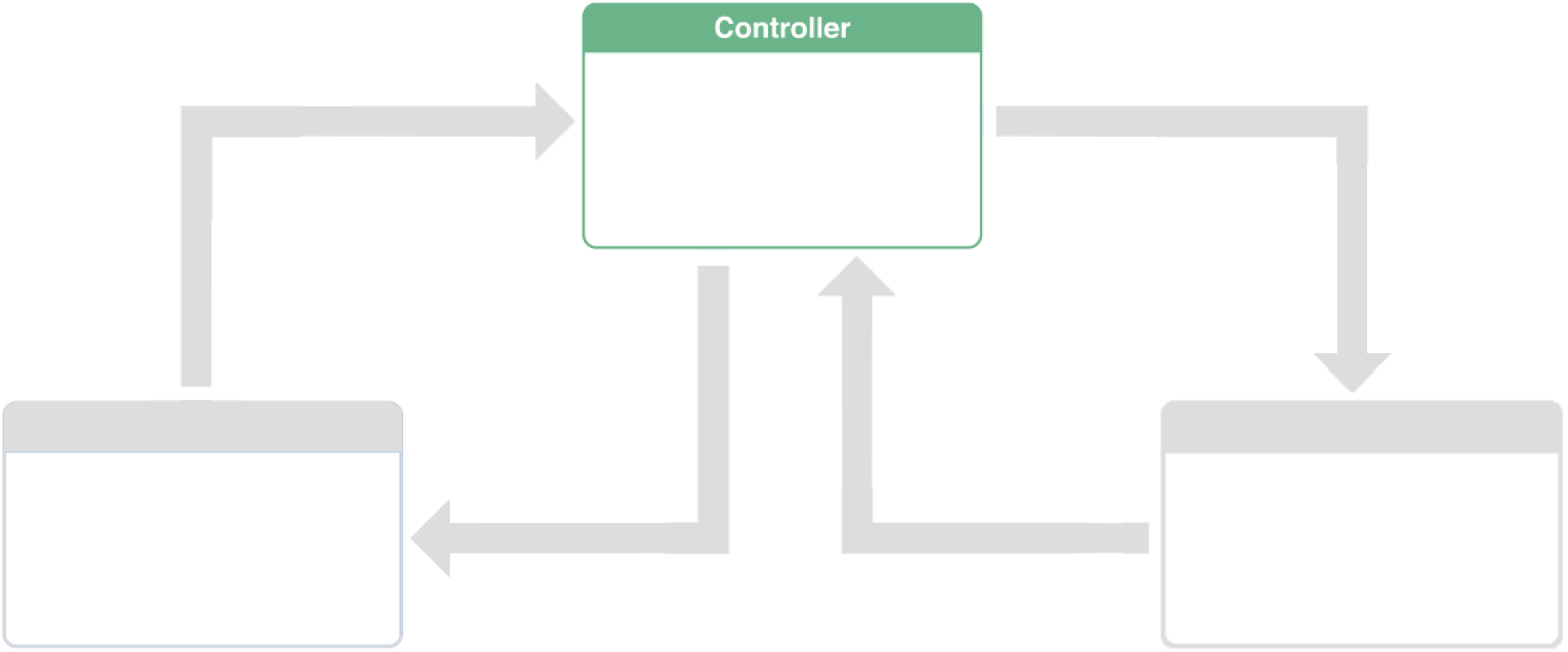
MVC

Makes you
Vomit
Constantly



https://developer.apple.com/library/content/documentation/General/Conceptual/DevPedia-CocoaCore/MVC.html


Model


Controller

- Things to do in Verona
  - The Verona Arena
  - Romeo and Juliet's home
  - The Church of San Zeno Maggiore
  - Piazza delle Erbe
  - Piazza dei Signori
  - Castelvecchio

- Attend #Pragma Conf

```
class CanvasViewController: UIViewController {
    override func viewDidLoad() {
        // ...
        let pan = UIPanGestureRecognizer(target: self,
                                         action: #selector(handlePan))
        self.canvas.addGestureRecognizer(pan)
    }

    @objc func handlePan(_ pan: UIPanGestureRecognizer) {
        let state = pan.state
        let location = pan.location(in: self.canvas)
        let (node, nodeView) = self.node(at: location)

        // update location of view + model
        self.handleDrag(of: (node, nodeView) to: location, state: state)
    }
}
```

```
class CanvasViewController: NSViewController {
    override func viewDidLoad() {
        // ...
        let window = self.view.window
        window.trackEvents(matching: .any, mode: .eventTracking) { event in

            if event.type == .leftMouseDragged {
                let location = self.canvas.convert(event.locationInWindow, from: nil)
                let (node, nodeView) = self.node(at: location)

                // update location of view + model
                self.handleDrag(of: (node, nodeView) to: location, event: event)
            }

        }
    }
}
```

✌ Tip #2

don't take on too many *responsibilities*

Does Singleton violate the single responsibility principle?

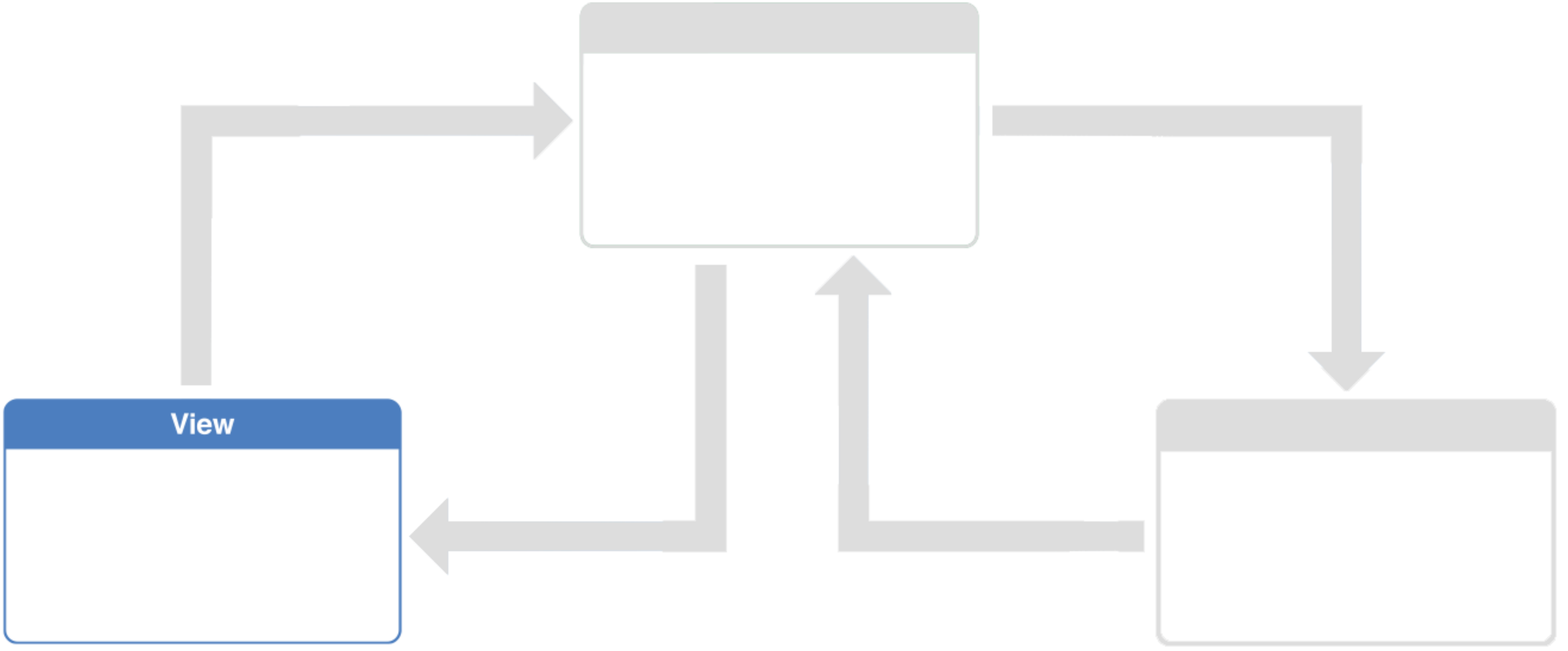
4

[redacted]

```
class DragController {
    enum Phase {
        case start, moving, end
    }

    init(draggedNode: Node) { ... }

    func drag(to location: CGPoint, phase: Phase) {
        if phase == .moving {
            // update view
            self.draggedView.center = location
        } else if phase == .end {
            if let nodeAtLocation = self.node(at: location) {
                // attach `draggedNode` to nodeAtLocation
            }
            // update model object
            self.draggedNode.location = location
        }
    }
}
```

✌ Tip #3

keep well-defined *boundaries*


View

UIKit vs. AppKit

Shimming

(Conditional Compilation)

```
#if os(macOS)
    import AppKit
    typealias View = NSView
#elseif os(iOS)
    import UIKit
    typealias View = UIView
#endif

class NodeView: View {
 #if os(macOS)
     override var isFlipped: Bool { return true }
 #endif
 // ...
 }
```

```swift
#if os(macOS)
    import AppKit
    typealias View = NSView
#elseif os(iOS)
    import UIKit
    typealias View = UIView
#endif

class NodeView: View { ... }
```

Louis D'hauwe
@LouisDhauwe

I should probably write a script to find all classes that have both a UIKit and AppKit variant 😅

```swift
#if os(macOS)

    import AppKit

    public typealias View           = NSView
    public typealias ViewController = NSViewController
    public typealias Window         = NSWindow
    public typealias Control        = NSControl
    public typealias TextView       = NSTextView
    public typealias TextField      = NSTextField
    public typealias Button         = NSButton
    public typealias Font           = NSFont
    public typealias Color          = NSColor
    public typealias StackView      = NSStackView
    public typealias Image          = NSImage
    public typealias BezierPath     = NSBezierPath
    public typealias ScrollView     = NSScrollView
    public typealias Screen         = NSScreen

#else

    import UIKit

    public typealias View           = UIView
    public typealias ViewController = UIViewController
    public typealias Window         = UIWindow
    public typealias Control        = UIControl
    public typealias TextView       = UITextView
    public typealias TextView       = UITextView
```

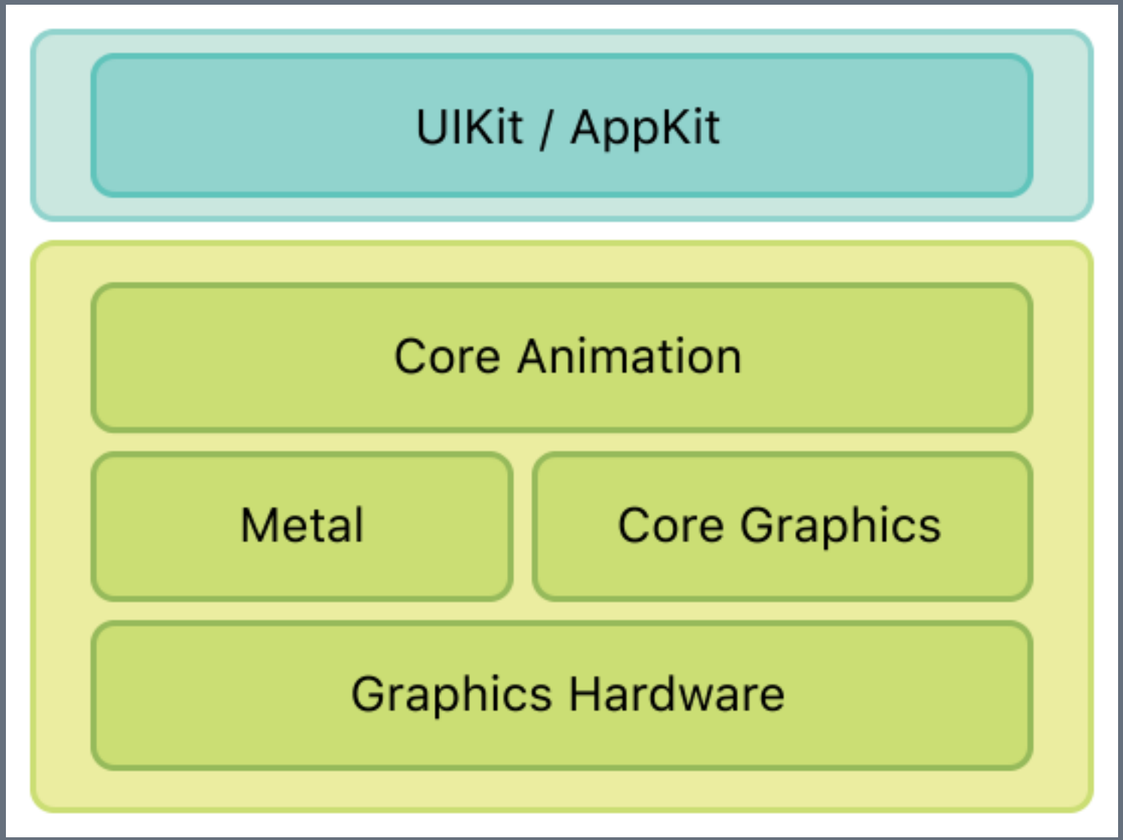
✌ Tip #4

use *shimming* in select places

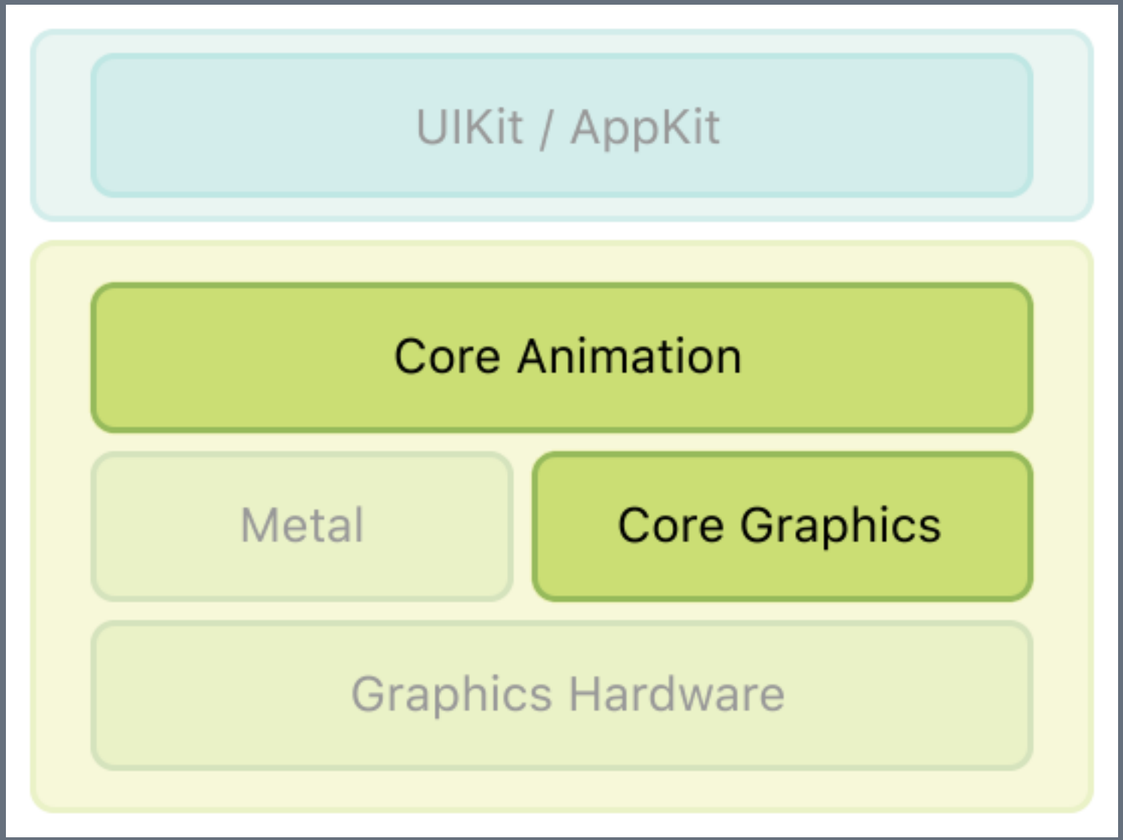
```
extension NSView {
    var center: CGPoint {
        get { ... }
        set { ... }
    }

    var alpha: CGFloat {
        get { return self.alphaValue }
        set { self.alphaValue = newValue }
    }
}
```

```
protocol ViewProtocol {
    var center: CGPoint { get set }
    var alpha: CGFloat { get set }
    ...
}

extension NSView: ViewProtocol { }
extension UIView: ViewProtocol { }

let view: ViewProtocol = UIView()
```

http://blog.creativelive.com/5-drawing-exercises-turn-make-anyone-artist/



https://developer.apple.com/library/content/documentation/Cocoa/Conceptual/CoreAnimation_guide/

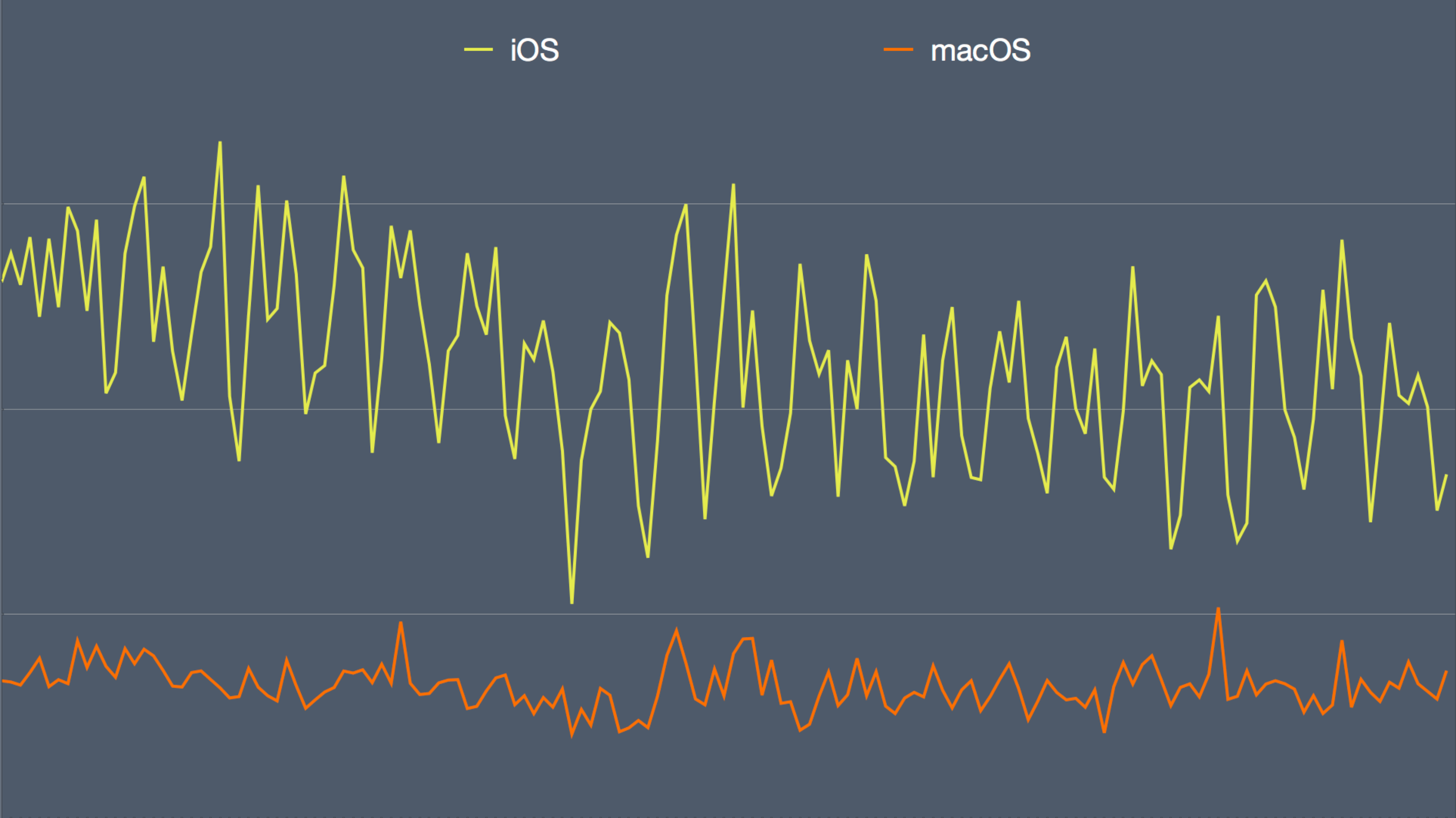

UIKit / AppKit
Core Animation
Metal
Core Graphics
Graphics Hardware

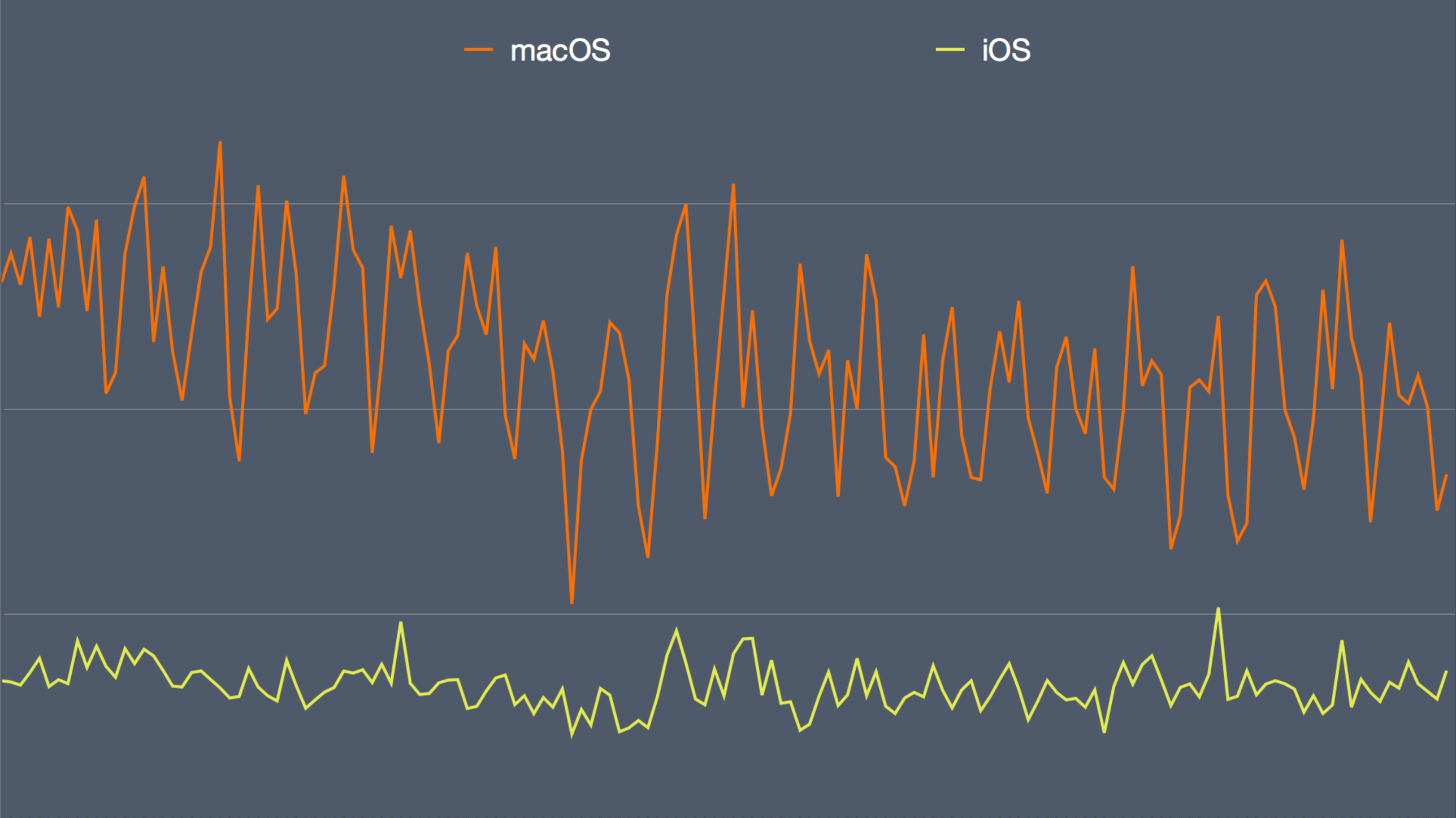
✌ Tip #5

consider *Core Graphics* & *Core Animation*

✌

1. be **pragmatic**
2. don't take on too many **responsibilities**
3. keep well-defined **boundaries**
4. use **shimming** in select places
5. consider **Core Graphics** & **Core Animation**


iOS
macOS


macOS
iOS

Questions?

@myell0w