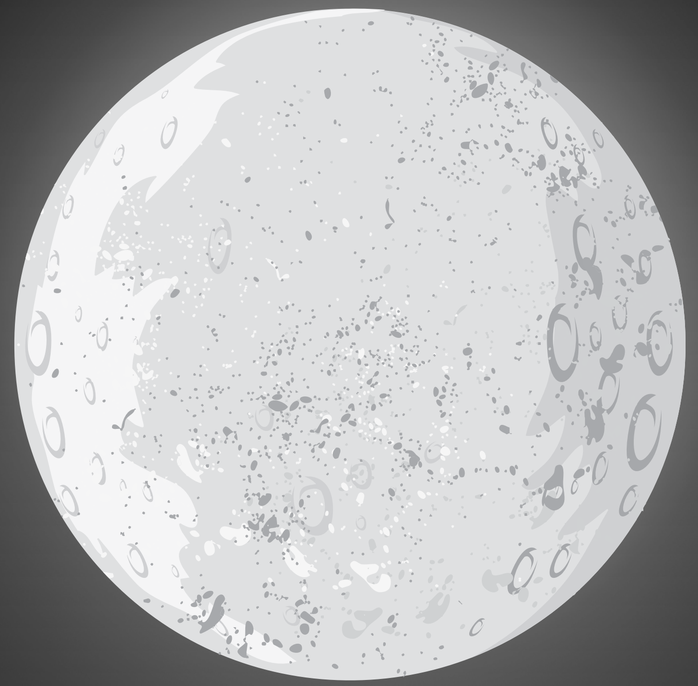


SwiftUI 与灵动岛

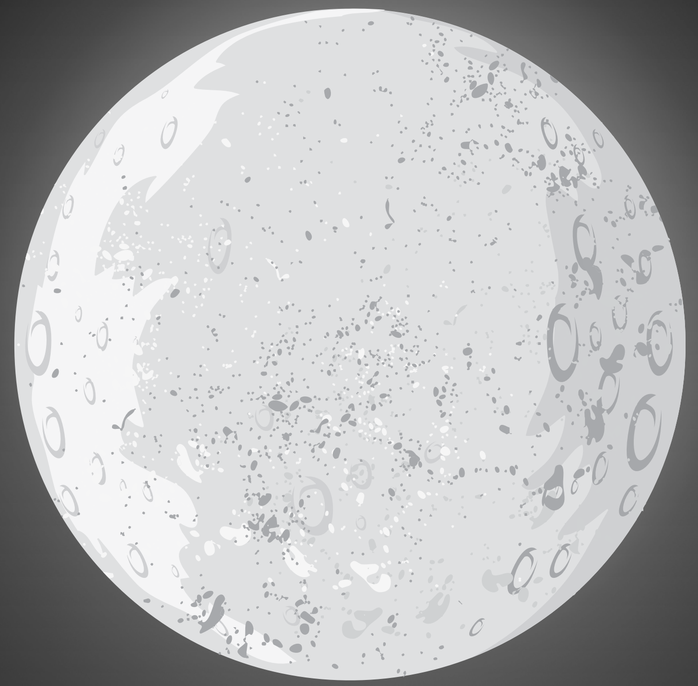

Declarative or Imperative?

声明式UI是一种趋势

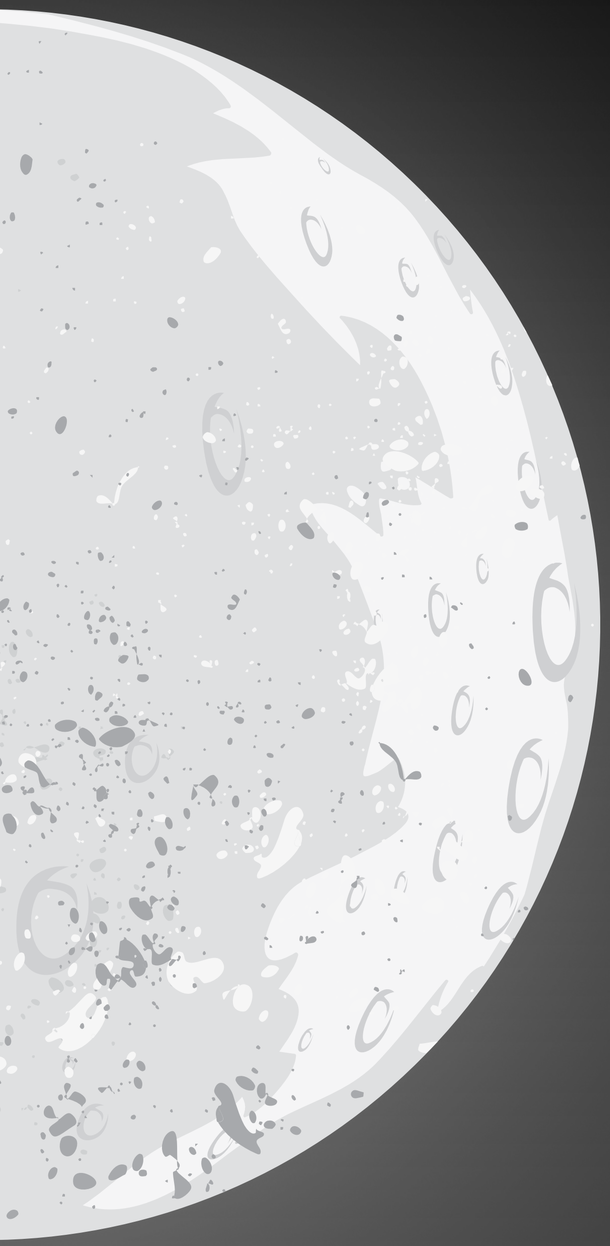
JavaScript - React, Vue

Android - Jetpack Compose

Flutter

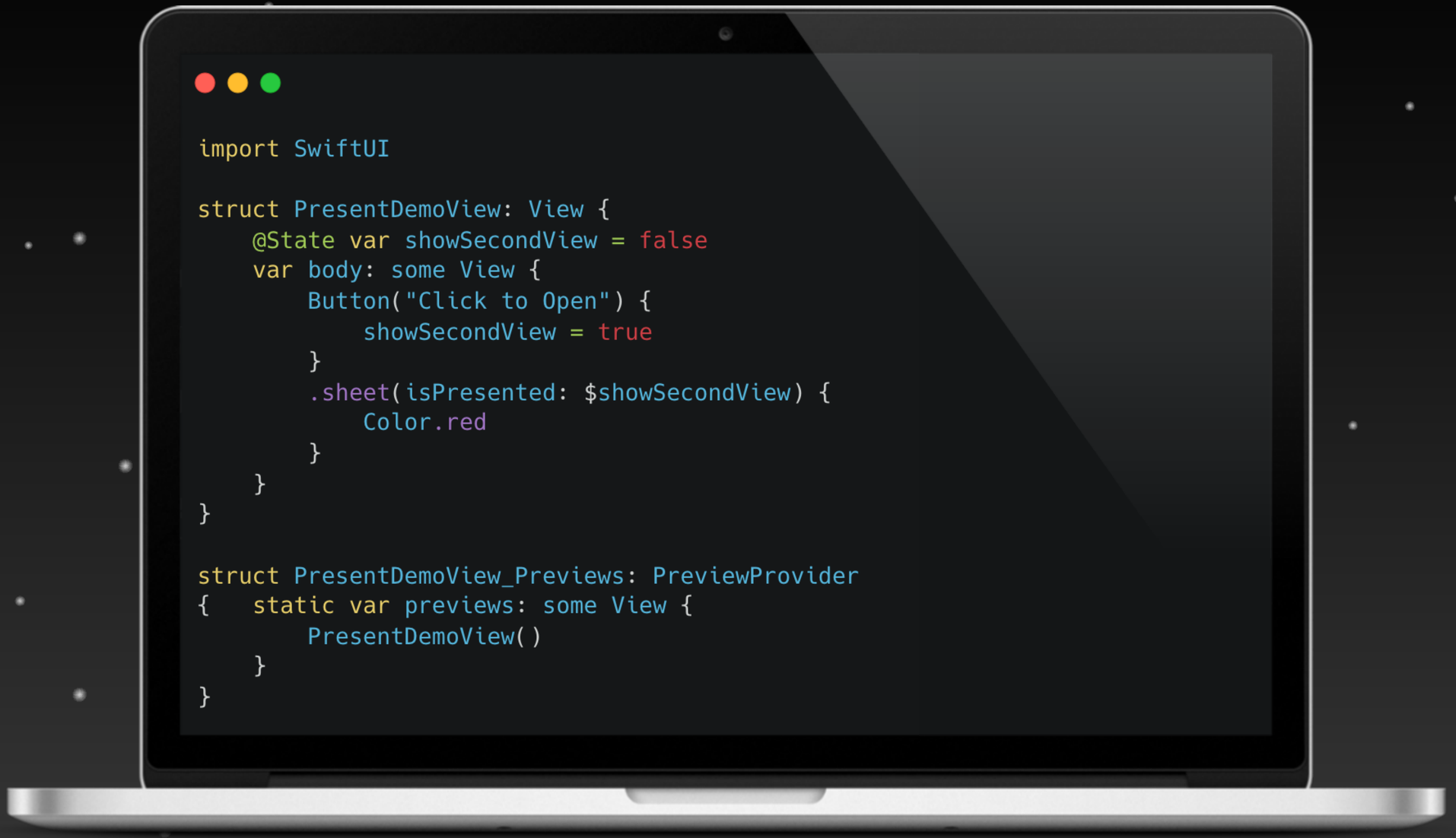


UI = f(state)

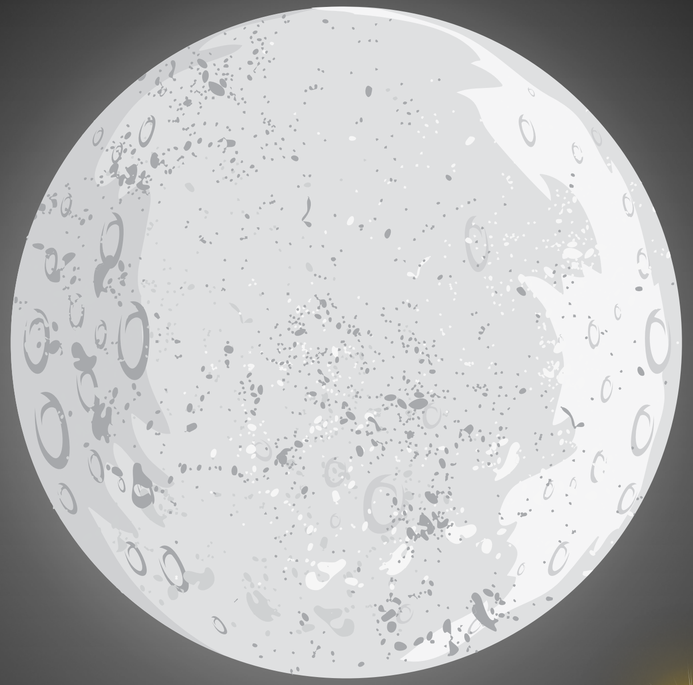


Talk is Cheap, show me the code

```
import UIKit

class PresentDemoViewController: UIViewController {
    private lazy var presentButton: UIButton = {
        let instance = UIButton()
        instance.setTitle("Click to Open", for: .normal)
        instance.addTarget(self, action: #selector(handlePresentView), for:
.touchUpInside)
        return instance
    }()

    override func viewDidLoad() {
        super.viewDidLoad()
        view.addSubview(presentButton)
        // Do any additional setup after loading the view.
    }

    @objc func handlePresentView() {
        let secondVC = UIViewController()
        present(secondVC, animated: true)
    }
}
```



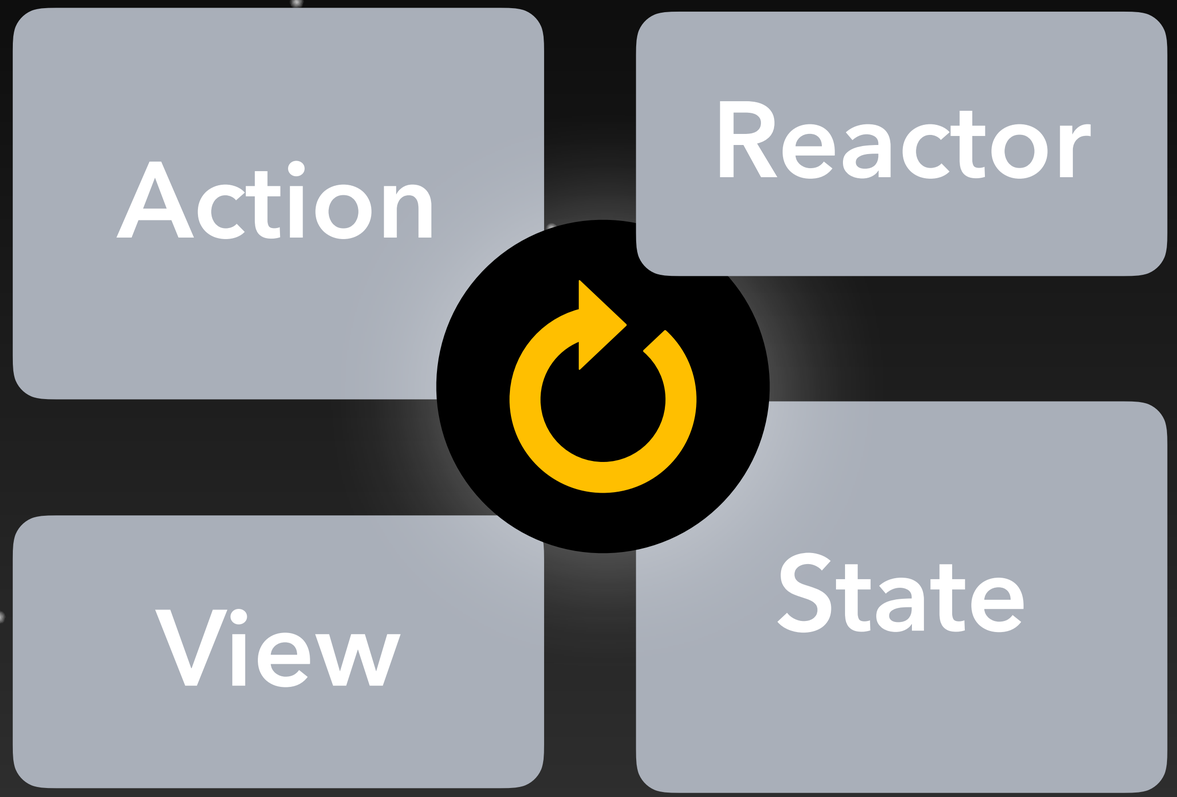
```swift
import SwiftUI

struct PresentDemoView: View {
    @State var showSecondView = false
    var body: some View {
        Button("Click to Open") {
            showSecondView = true
        }
        .sheet(isPresented: $showSecondView) {
            Color.red
        }
    }
}

struct PresentDemoView_Previews: PreviewProvider
{   static var previews: some View {
        PresentDemoView()
    }
}
```



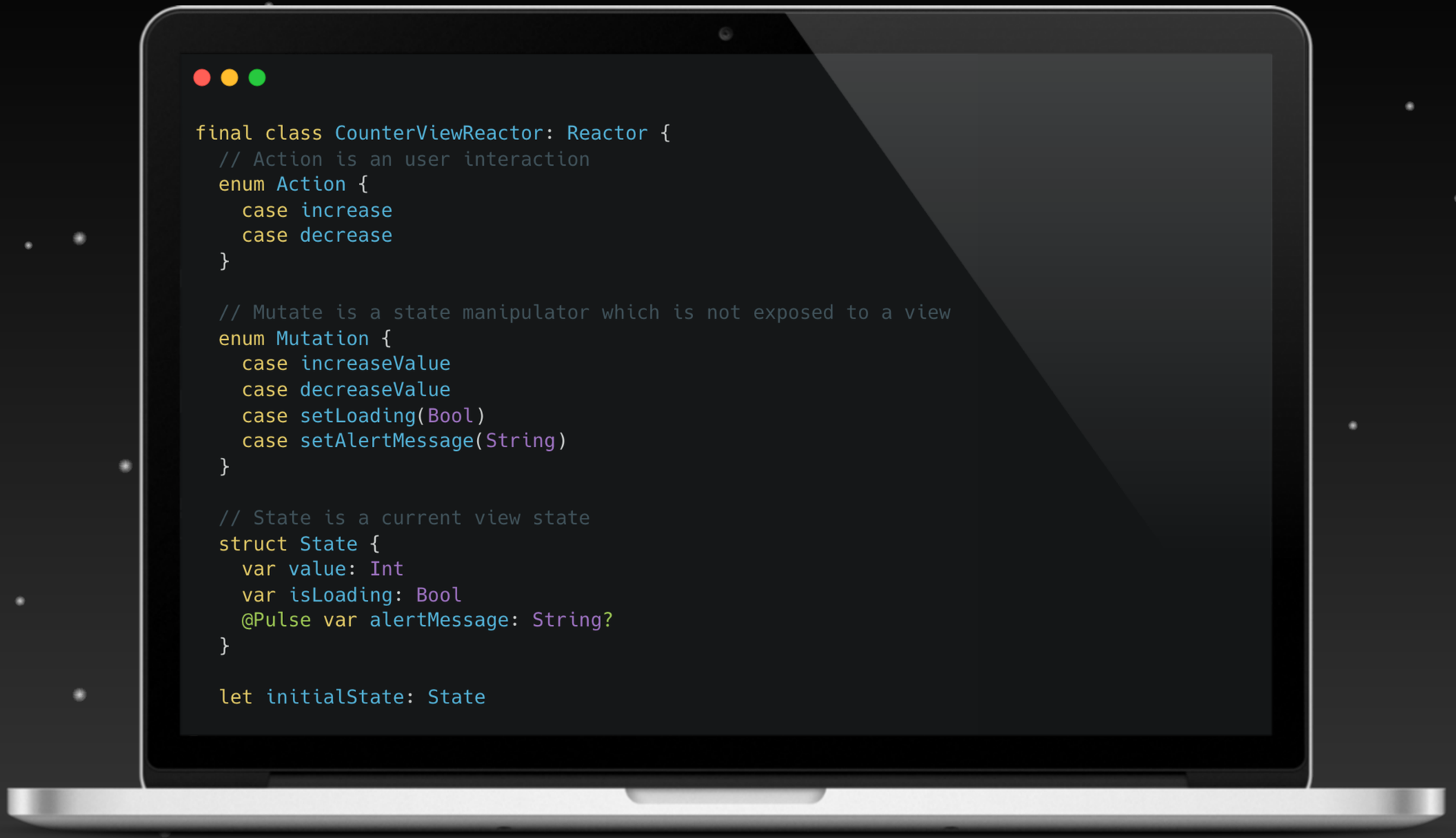
UIKit 就一定不能用 State驱动么？

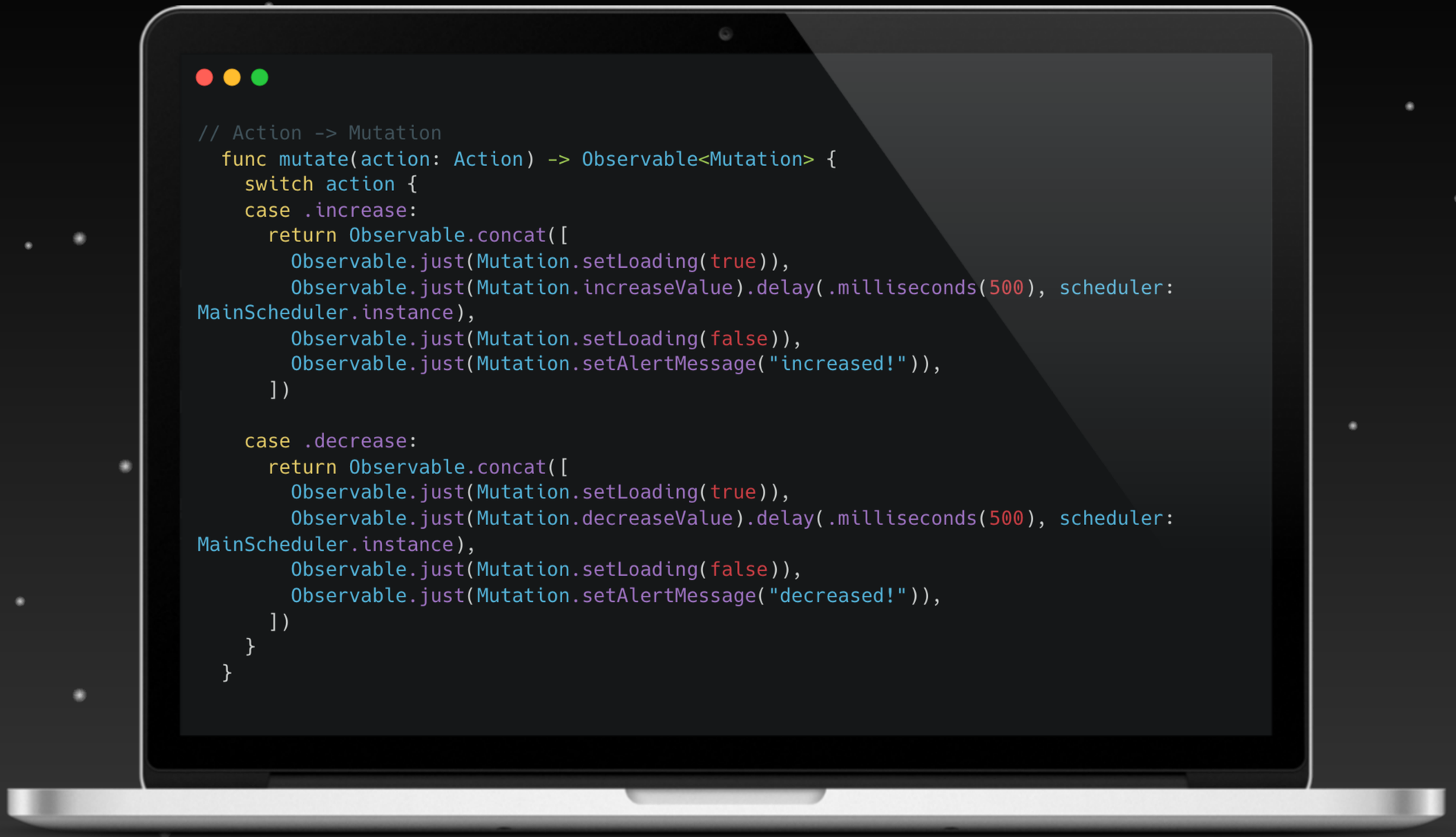
可以，但是成本高昂

ReactorKit

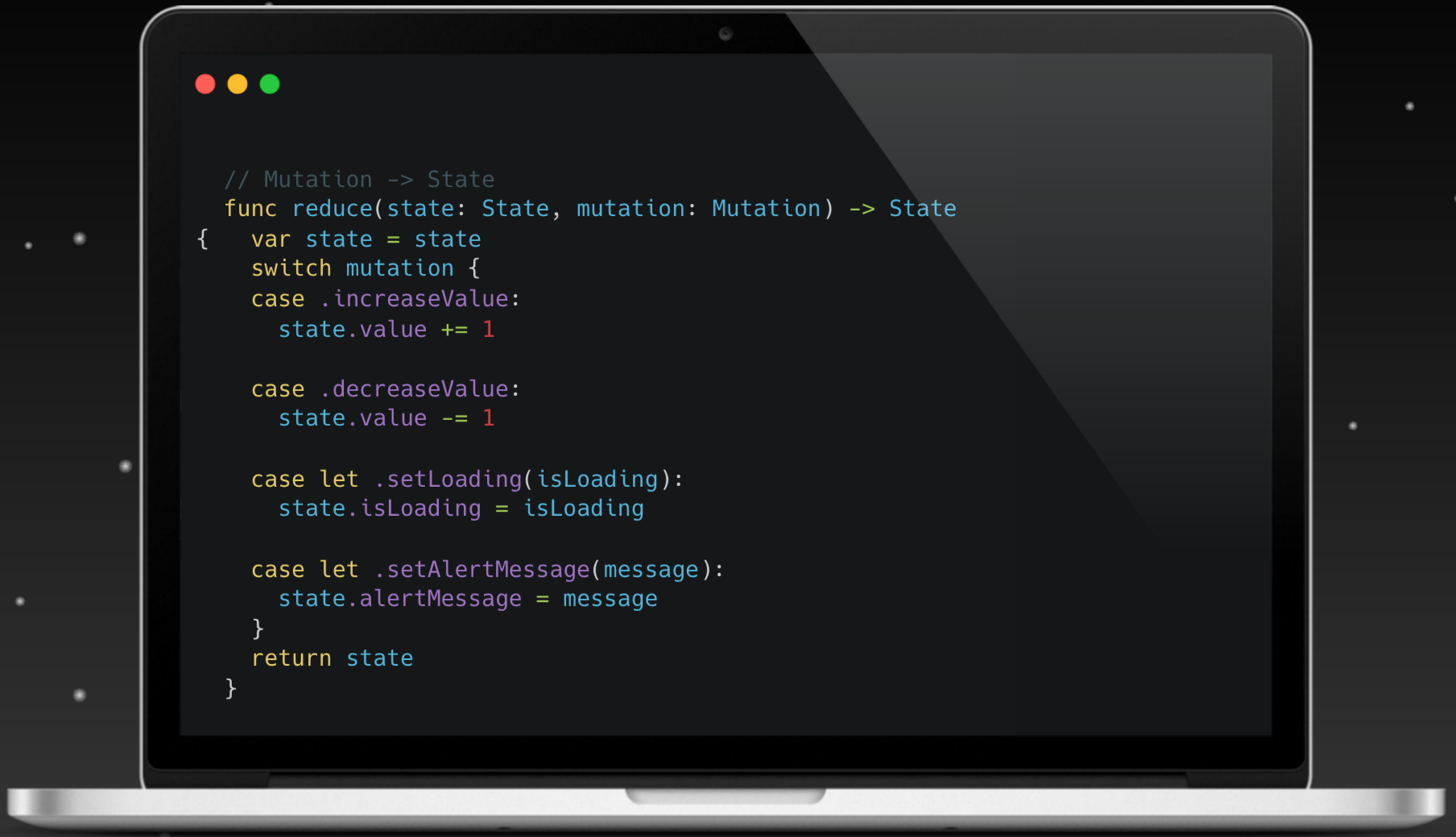

Action
Reactor
View
State

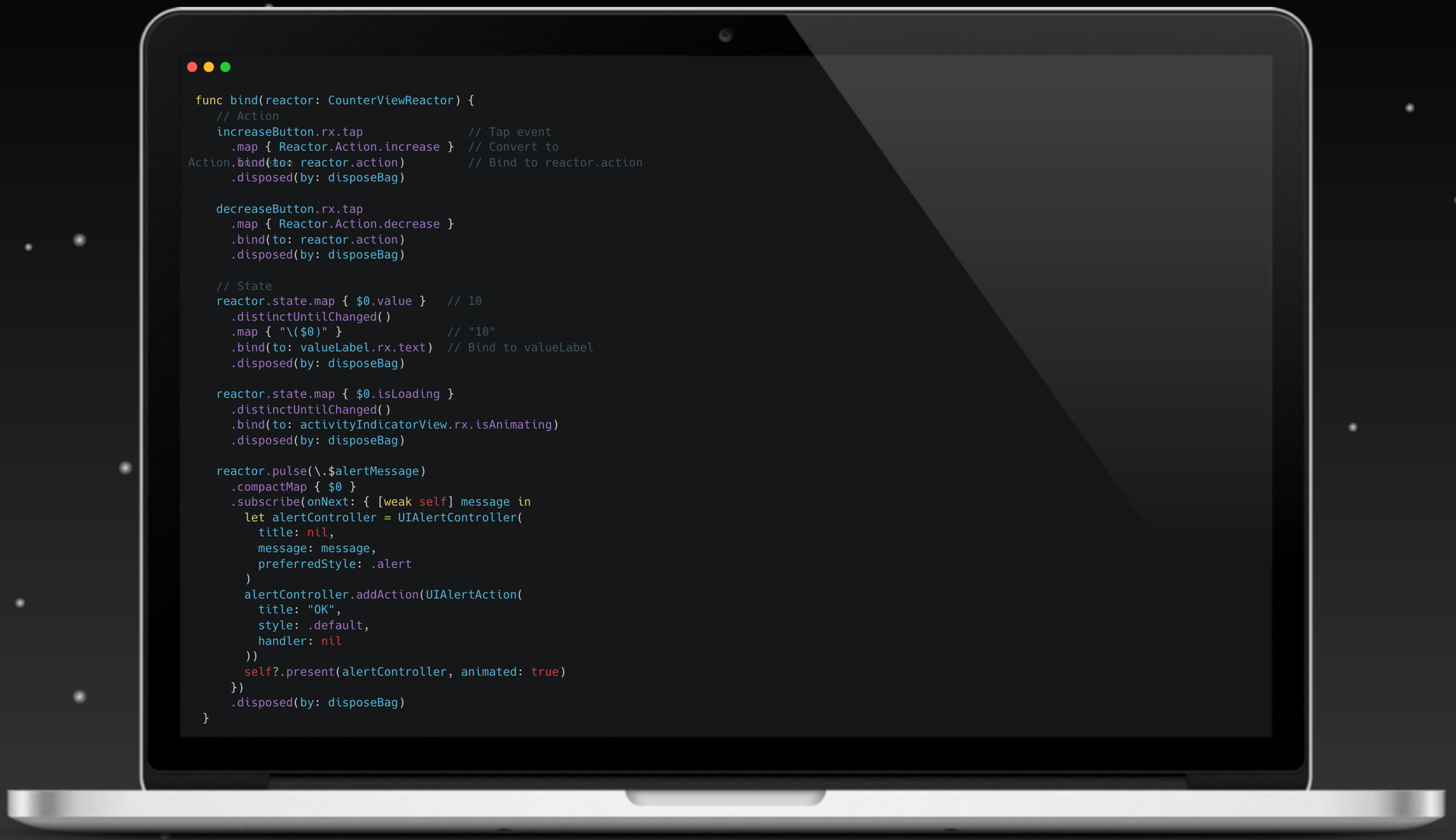

iPhone 14 Pro Max – iOS 16.2
3:35
GitHubSearch
-
1
+


final class CounterViewReactor: Reactor {
// Action is an user interaction
enum Action {
case increase
case decrease
}
// Mutate is a state manipulator which is not exposed to a view
enum Mutation {
case increaseValue
case decreaseValue
case setLoading(Bool)
case setAlertMessage(String)
}
// State is a current view state
struct State {
var value: Int
var isLoading: Bool
@Pulse var alertMessage: String?
}
let initialState: State

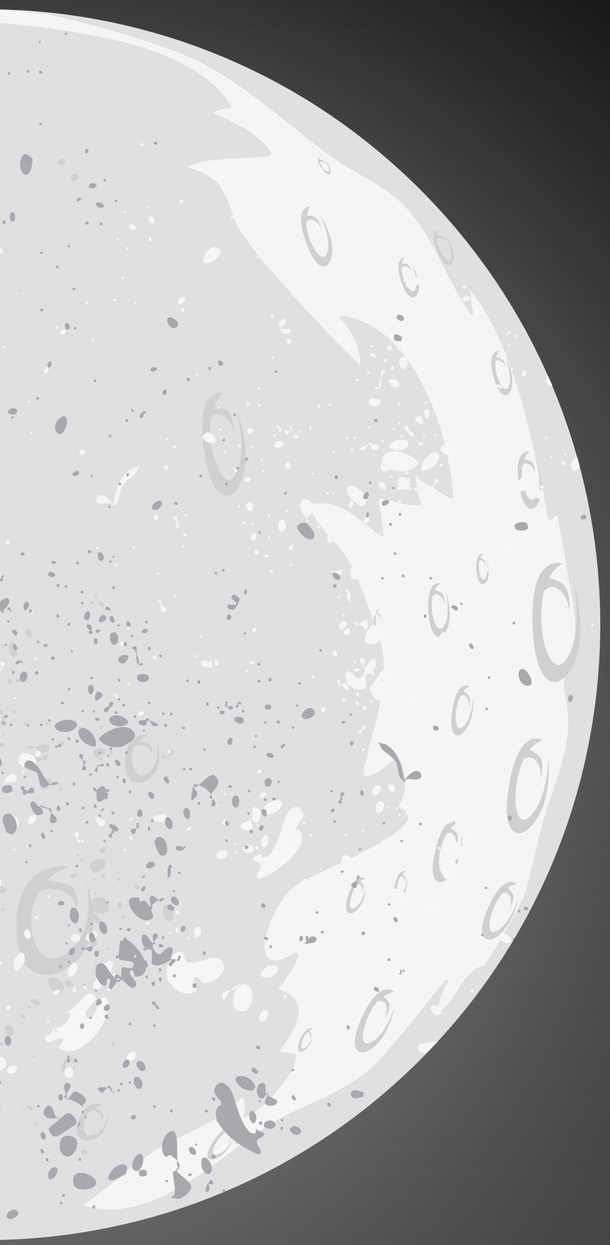


```
// Action -> Mutation
  func mutate(action: Action) -> Observable<Mutation> {
    switch action {
    case .increase:
      return Observable.concat([
        Observable.just(Mutation.setLoading(true)),
        Observable.just(Mutation.increaseValue).delay(.milliseconds(500), scheduler:
MainScheduler.instance),
        Observable.just(Mutation.setLoading(false)),
        Observable.just(Mutation.setAlertMessage("increased!")),
      ])

    case .decrease:
      return Observable.concat([
        Observable.just(Mutation.setLoading(true)),
        Observable.just(Mutation.decreaseValue).delay(.milliseconds(500), scheduler:
MainScheduler.instance),
        Observable.just(Mutation.setLoading(false)),
        Observable.just(Mutation.setAlertMessage("decreased!")),
      ])
    }
  }
```


// Mutation -> State
func reduce(state: State, mutation: Mutation) -> State
{
var state = state
switch mutation {
case .increaseValue:
state.value += 1
case .decreaseValue:
state.value -= 1
case let .setLoading(isLoading):
state.isLoading = isLoading
case let .setAlertMessage(message):
state.alertMessage = message
}
return state
}



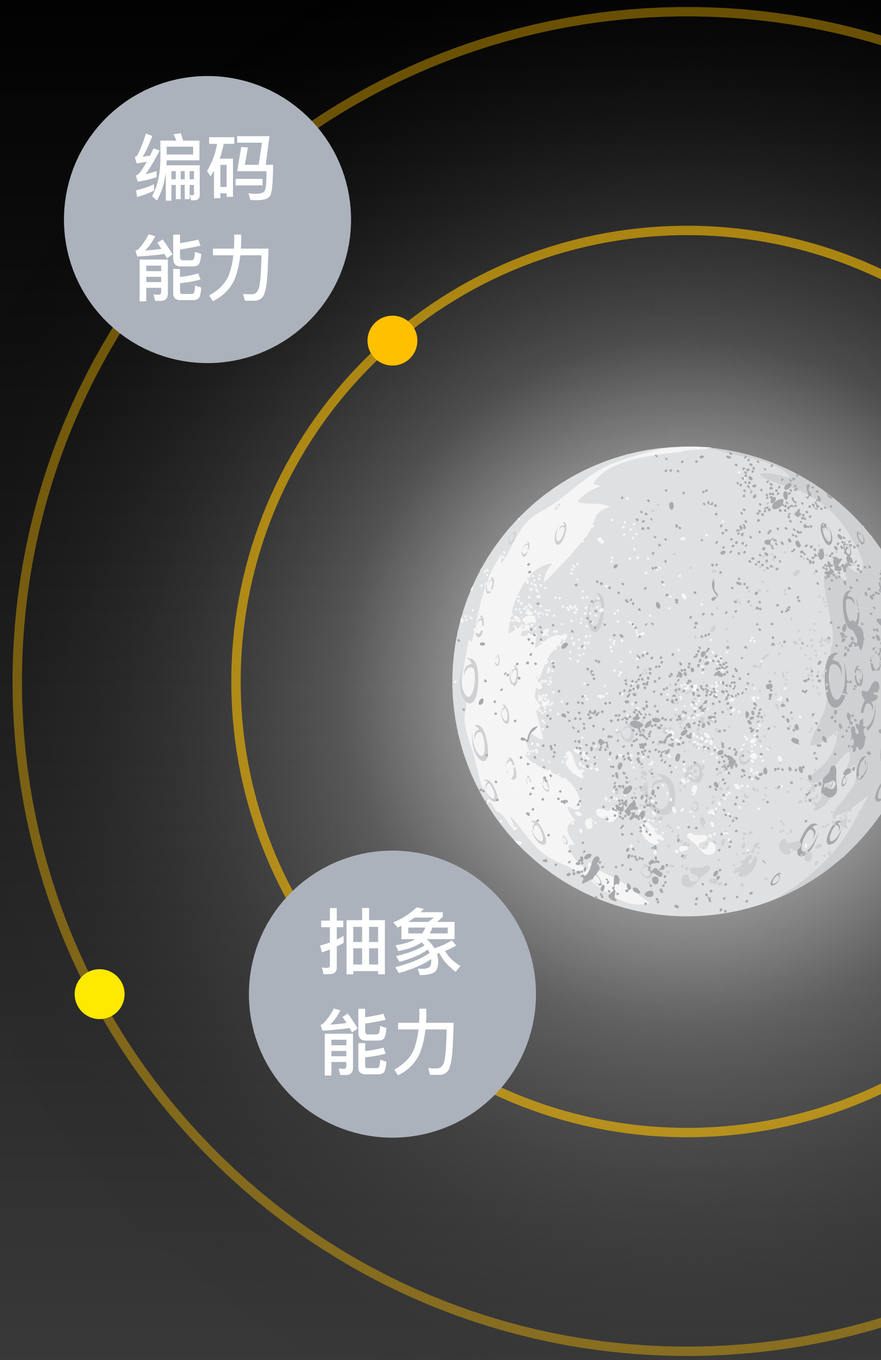
```swift
func bind(reactor: CounterViewReactor) {
    // Action
    increaseButton.rx.tap              // Tap event
      .map { Reactor.Action.increase }  // Convert to
Action.bind(to: reactor.action)         // Bind to reactor.action
      .disposed(by: disposeBag)

    decreaseButton.rx.tap
      .map { Reactor.Action.decrease }
      .bind(to: reactor.action)
      .disposed(by: disposeBag)

    // State
    reactor.state.map { $0.value }   // 10
      .distinctUntilChanged()
      .map { "\($0)" }               // "10"
      .bind(to: valueLabel.rx.text)  // Bind to valueLabel
      .disposed(by: disposeBag)

    reactor.state.map { $0.isLoading }
      .distinctUntilChanged()
      .bind(to: activityIndicatorView.rx.isAnimating)
      .disposed(by: disposeBag)

    reactor.pulse(\.$alertMessage)
      .compactMap { $0 }
      .subscribe(onNext: { [weak self] message in
        let alertController = UIAlertController(
          title: nil,
          message: message,
          preferredStyle: .alert
        )
        alertController.addAction(UIAlertAction(
          title: "OK",
          style: .default,
          handler: nil
        ))
        self?.present(alertController, animated: true)
      })
      .disposed(by: disposeBag)
  }
```



框架的演进，是为了更好的team code

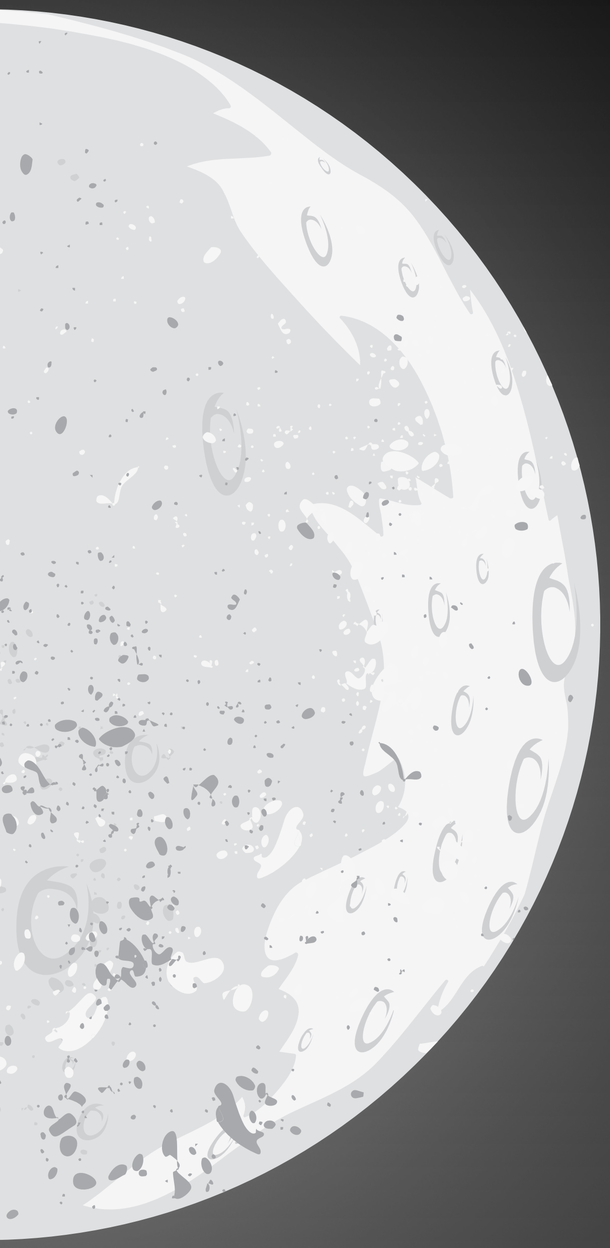
什么是好的团队代码

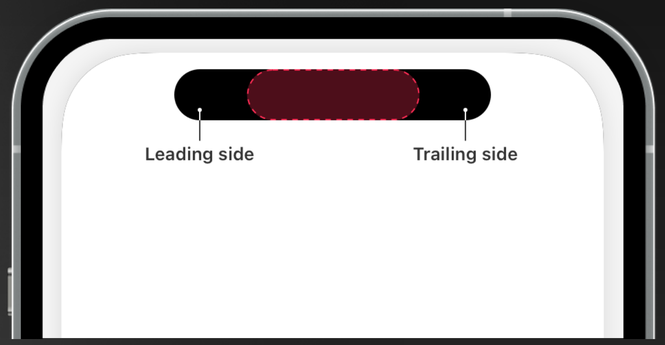
Quality

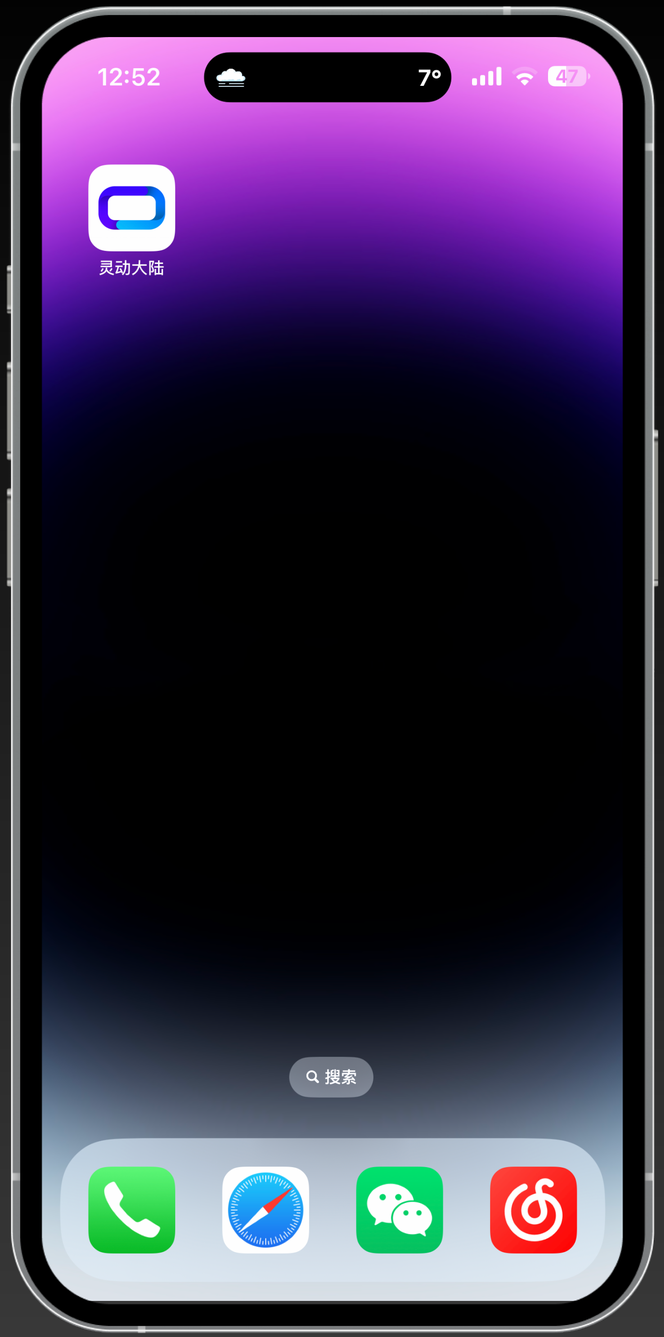


灵动岛

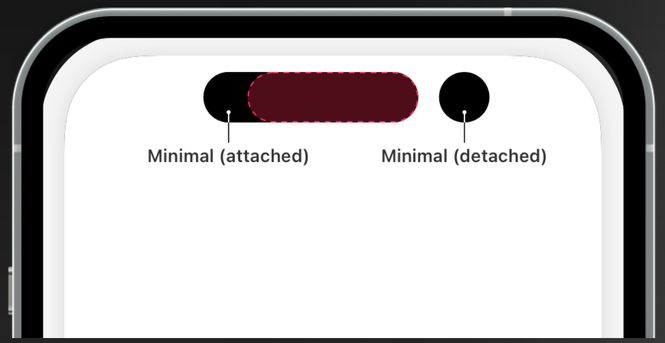

9:41
Wednesday, June 28
9:41
3 items
Juan C.
Arriving in 8 minutes
9:41
8min
FaceTime
Calendar
Mail
Notes
Photos
Camera
News
TV
Reminders
Clock
Maps
Health
Podcasts
App Store
Wallet
Settings
Search
3 items
Juan C.
Arriving in 8 minutes
Contact Juan C.
Photos
Camera
News
TV
Reminders
Clock
Maps
Health
Podcasts
App Store
Wallet
Settings
Search

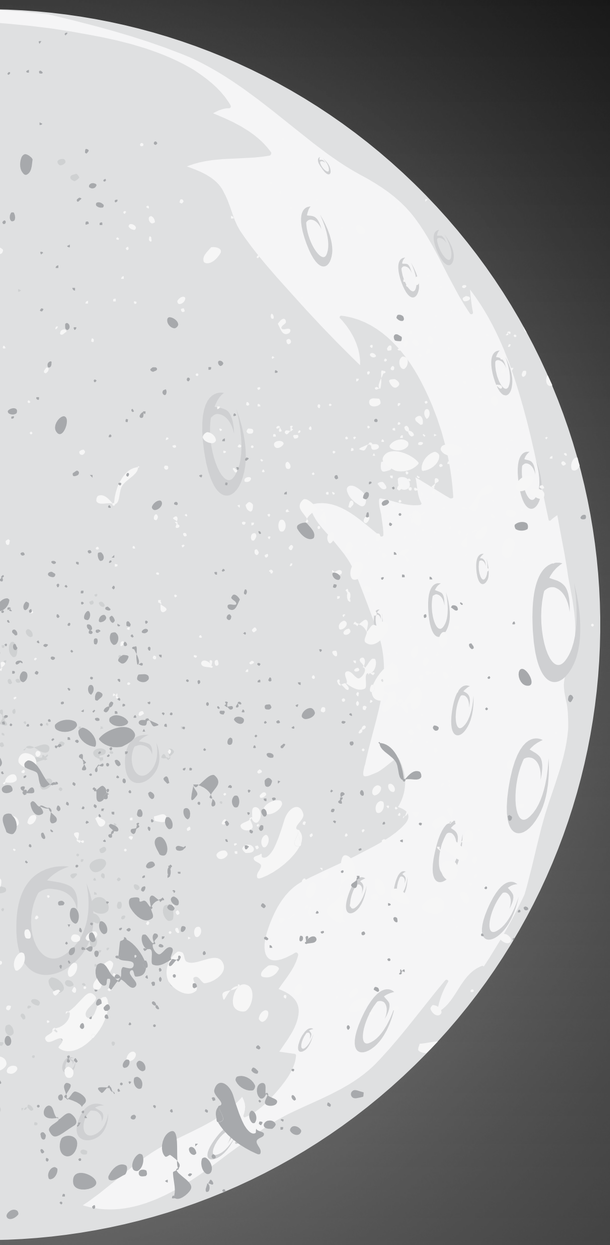


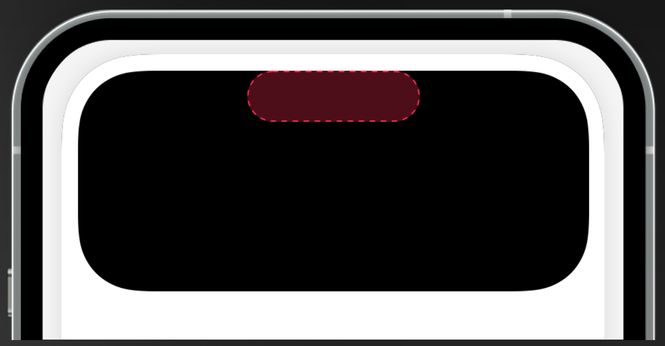

Leading side
Trailing side


12:52
7°
灵动大陆
搜索

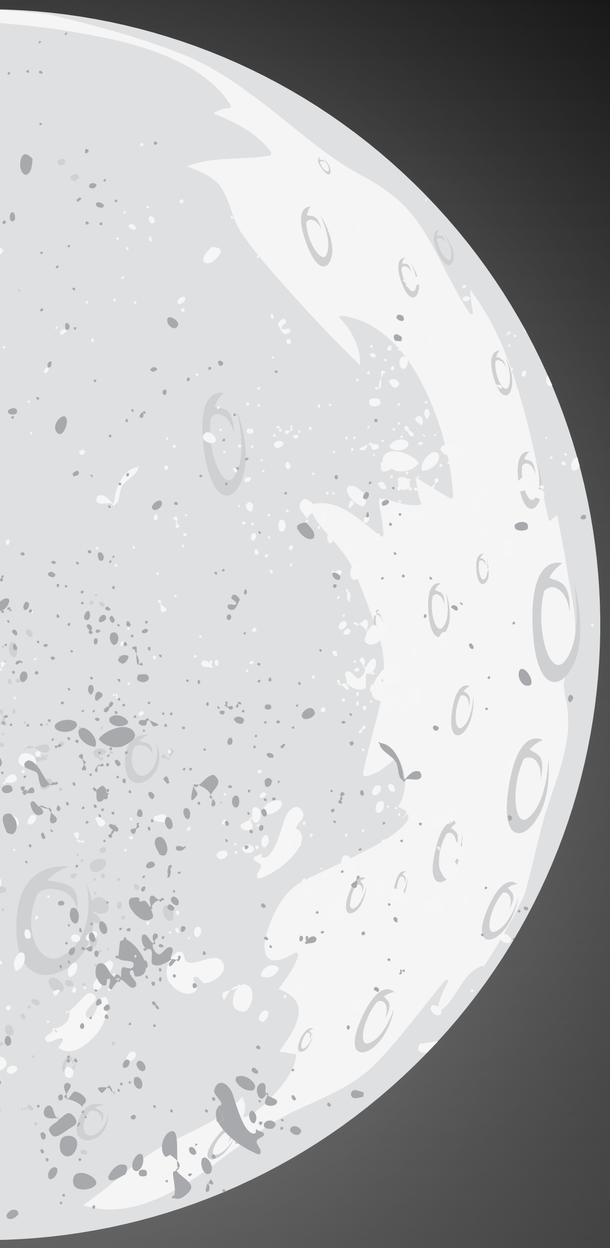

12:52
7°
灵动大陆
搜索

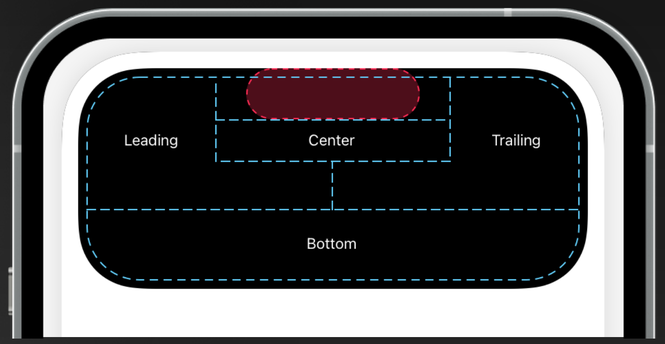

Minimal (attached)
Minimal (detached)

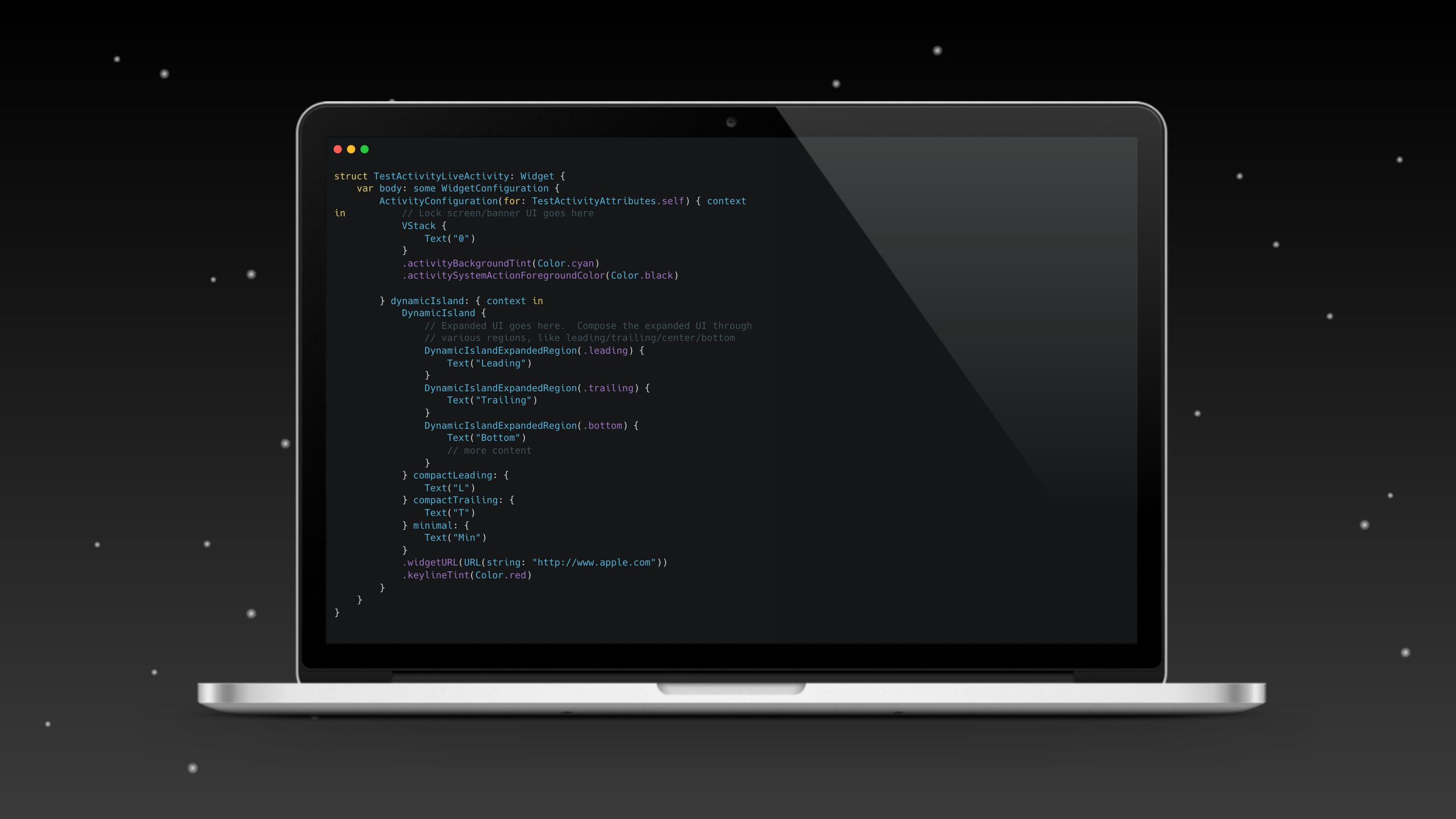





7°
雾
01-06 12:52:42
多云，云渐少，明天早晨 8 点钟后转至晴
空气质量 96
湿度 63%
北东北风 3.4 km/h
搜索




Leading
Center
Trailing
Bottom


struct TestActivityLiveActivity: Widget {
var body: some WidgetConfiguration {
ActivityConfiguration(for: TestActivityAttributes.self) { context
in
// Lock screen/banner UI goes here
VStack {
Text("0")
}
.activityBackgroundTint(Color.cyan)
.activitySystemActionForegroundColor(Color.black)
} dynamicIsland: { context in
DynamicIsland {
// Expanded UI goes here. Compose the expanded UI through
// various regions, like leading/trailing/center/bottom
DynamicIslandExpandedRegion(.leading) {
Text("Leading")
}
DynamicIslandExpandedRegion(.trailing) {
Text("Trailing")
}
DynamicIslandExpandedRegion(.bottom) {
Text("Bottom")
// more content
}
} compactLeading: {
Text("L")
} compactTrailing: {
Text("T")
} minimal: {
Text("Min")
}
.widgetURL(URL(string: "http://www.apple.com"))
.keylineTint(Color.red)
}
}
}


THANK YOU