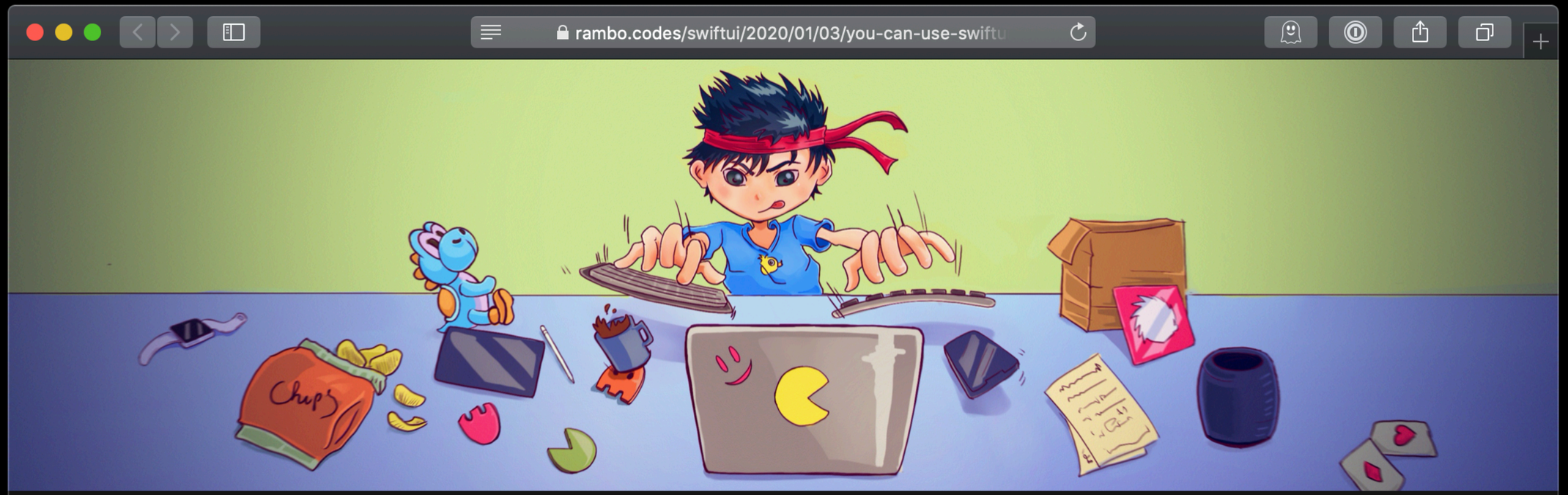
What if your controller isn’t a controller?

MVC



You can use SwiftUI today

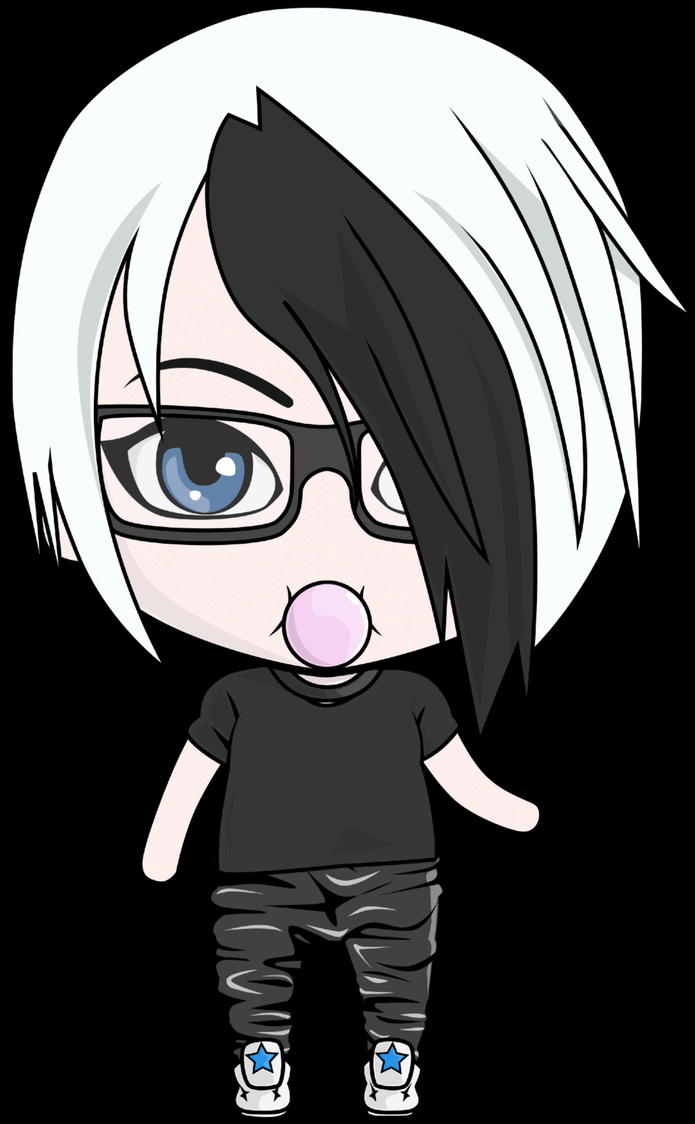
Jan 3, 2020

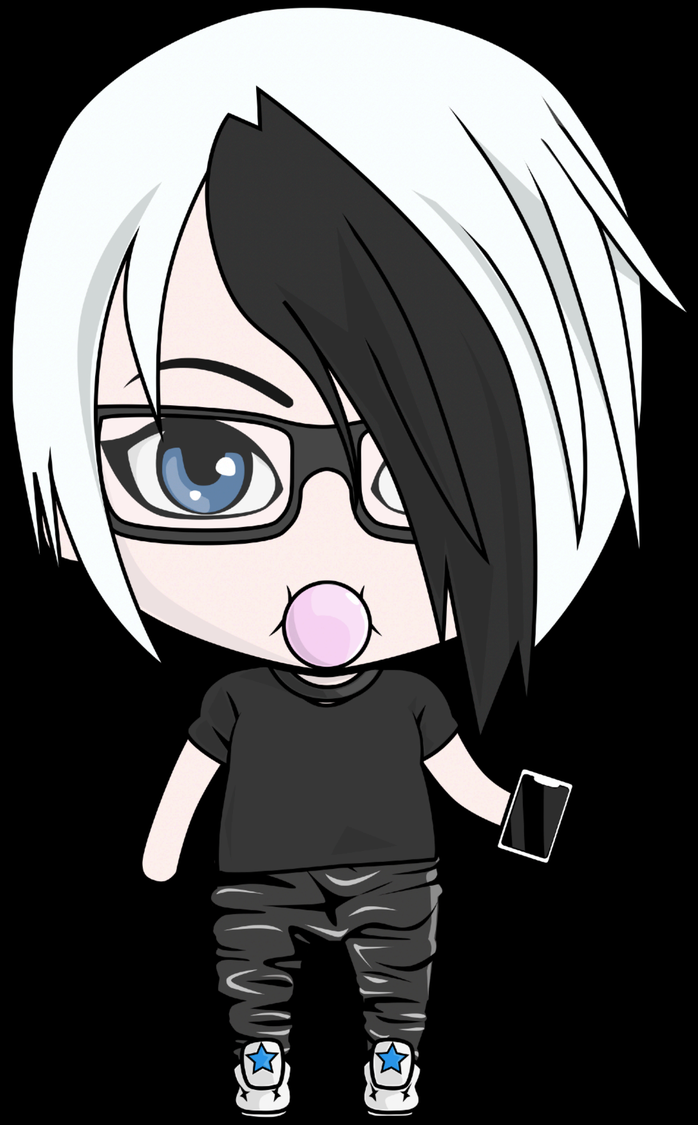
Whenever Apple introduces a new technology for developers, there's always the question: "Can I use this thing right now for my projects?". The answer many times ends up being the good, old, and boring "it depends", but more often than not it leans toward "probably better to wait for a few OS versions".

With SwiftUI, it's the same. While there are some brave developers using it to create production apps - especially on the Apple Watch - most developers seem to agree that it's kind of like using Swift when it first came out.

I've been asked the question many times, and I tend to agree that going all-in on SwiftUI right now for an app you plan on

Story time





MDD

Medium-driven development

Acme Inc. Follow
Apr 16, 2019 · 11 min read ★

Today, we're very proud to introduce the best architecture for iOS apps, which we call PIE: "Presenter, Interactor, Entity". This new architecture is going to revolutionize how you write iOS apps, with functional, composable and testable protocolo-oriented interfaces.


iOS


10:49
FaceTime
Wednesday
22
Calendar
Photos
Camera
1
Mail
Clock
280
Maps
Weather
Reminders
Notes
Stocks
News
Books
App Store
Podcasts
tv
TV
Health
Home
Wallet
Settings


DEC
17

iOS Architecture Generator

PIE

Presenter Interactor Entity

GENERATE ANOTHER ONE

Real artists ship

- Containers
- Generic <>
- View controllers
- Flow controllers

- Containers
- Generic <>
- View controllers
- Flow controllers

Containers

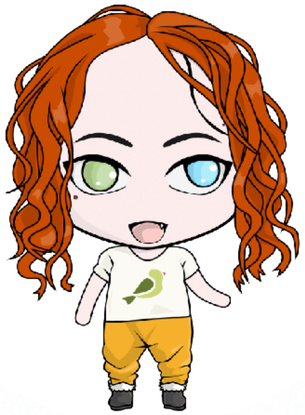

Cancelar
Pronto
All
Animals
^_^'
Animehair
Christmas

ChibiCardsViewController - Seus Chibis

ChibiEditorViewController

Cancelar

Pronto



T...ler

T...ler

InventoryDrawerViewController

ModernInventoryCollectionViewController

InventoryFiltersViewController

All

Animals

Animehair

Christmas

InventoryCategoriesStripViewController

```swift
let child = MyViewController()

addChild(child)

child.view.translatesAutoresizingMaskIntoConstraints = false
view.addSubview(child.view)

NSLayoutConstraint.activate([
    child.view.leadingAnchor.constraint(equalTo: view.leadingAnchor),
    child.view.trailingAnchor.constraint(equalTo: view.trailingAnchor),
    child.view.topAnchor.constraint(equalTo: view.topAnchor),
    child.view.bottomAnchor.constraint(equalTo: view.bottomAnchor)
])

child.didMove(toParent: self)
```

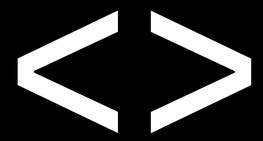
```swift
public extension UIViewController {
    func install(_ child: UIViewController) {
        addChild(child)

        child.view.translatesAutoresizingMaskIntoConstraints = false
        view.addSubview(child.view)

        NSLayoutConstraint.activate([
            child.view.leadingAnchor.constraint(equalTo: view.leadingAnchor),
            child.view.trailingAnchor.constraint(equalTo: view.trailingAnchor),
            child.view.topAnchor.constraint(equalTo: view.topAnchor),
            child.view.bottomAnchor.constraint(equalTo: view.bottomAnchor)
        ])

        child.didMove(toParent: self)
    }
}
```

- Loading
- Loaded
- Empty
- Error

```
public final class StateViewController: UIViewController {

    public enum State {
        case loading(message: String)
        case content(controller: UIViewController)
        case error(message: String)
        case empty(message: String)
    }

  // ...

}
```

```
public var state: State = .loading(message: "Loading") {
    didSet {
        applyState()
    }
}
```

Generic



```swift
public final class ContainerCollectionViewCell<V: UIView>: UICollectionViewCell {

    public lazy var view: V = {
        return V()
    }()

    public override init(frame: CGRect) {
        super.init(frame: frame)

        view.translatesAutoresizingMaskIntoConstraints = false
        contentView.addSubview(view)

        NSLayoutConstraint.activate([
            view.leadingAnchor.constraint(equalTo: contentView.leadingAnchor),
            view.trailingAnchor.constraint(equalTo: contentView.trailingAnchor),
            view.topAnchor.constraint(equalTo: contentView.topAnchor),
            view.bottomAnchor.constraint(equalTo: contentView.bottomAnchor)
        ])
    }

}
```

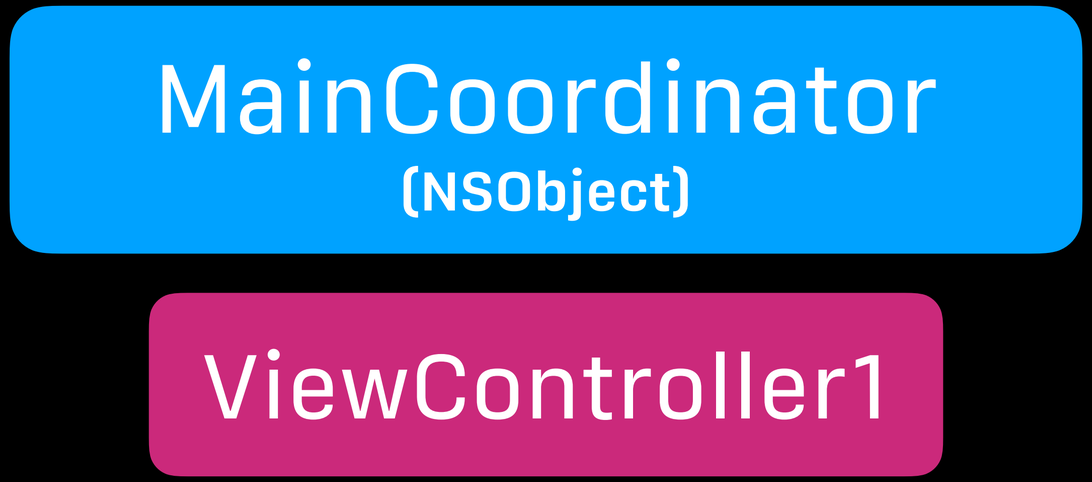
```
class GenericCollectionViewController<V: UIView, C: ContainerCollectionViewCell<V>>: UICollectionViewController {

    init(viewType: V.Type) {
        super.init(collectionViewLayout: makeDefaultLayout())
    }

    var numberOfItems: () -> Int = { 0 } {
        didSet {
            collectionView?.reloadData()
        }
    }

    var configureView: (IndexPath, V) -> () = { _, _ in } {
        didSet {
            collectionView?.reloadData()
        }
    }

    var didSelectView: (IndexPath, V) -> () = { _, _ in }

    // ...
}
```

View controllers

DUMB = GOOD

Flow controllers

Coordinator


MainCoordinator
(NSObject)
ViewController1

MainCoordinator
(NSObject)

ViewController1

present!

DetailCoordinator
(NSObject)

ViewController2

MainCoordinator
(NSObject)

ViewController1

dismiss!

DetailCoordinator
(NSObject)

Still here 🤘

FlowController
: UIViewController

MainFlowController
(UIViewController)

child > ViewController1

MainFlowController
(UIViewController)

child > ViewController1

present!

(viewDidLoad,
viewWillAppear,
viewDidAppear)

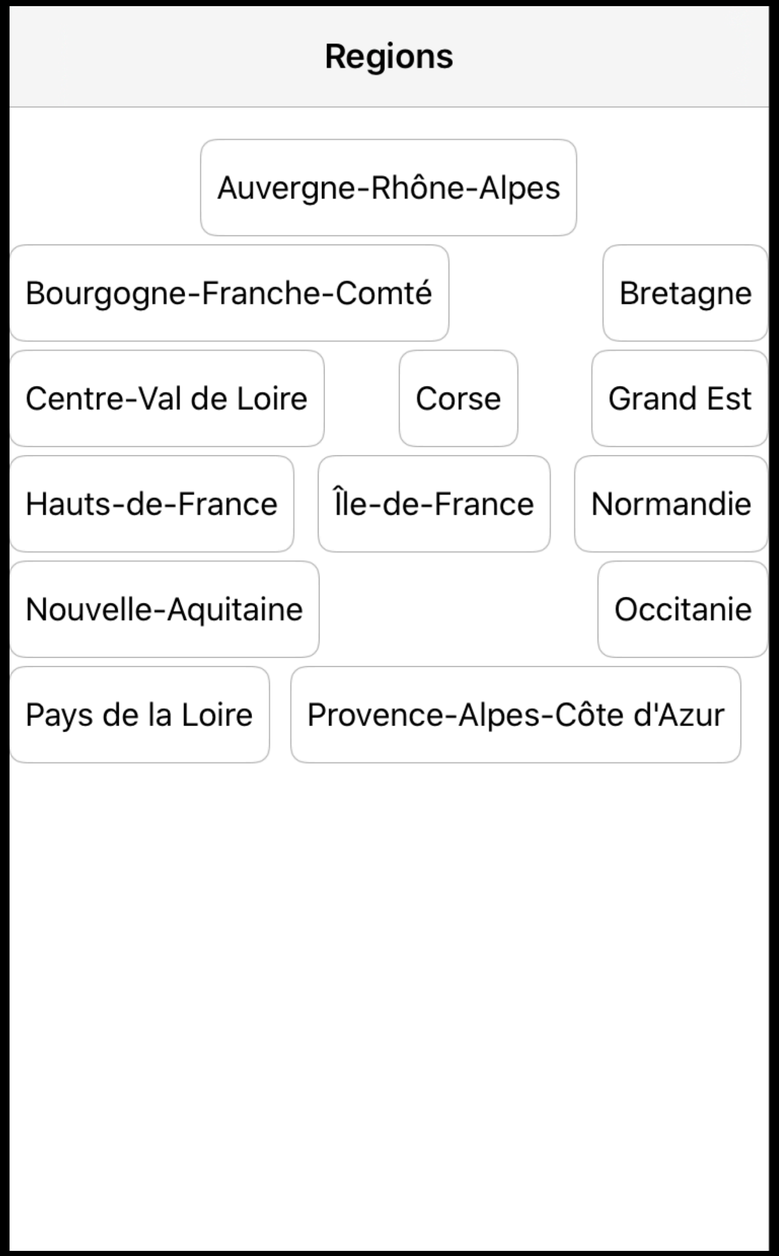
DetailFlowController
(UIViewController)

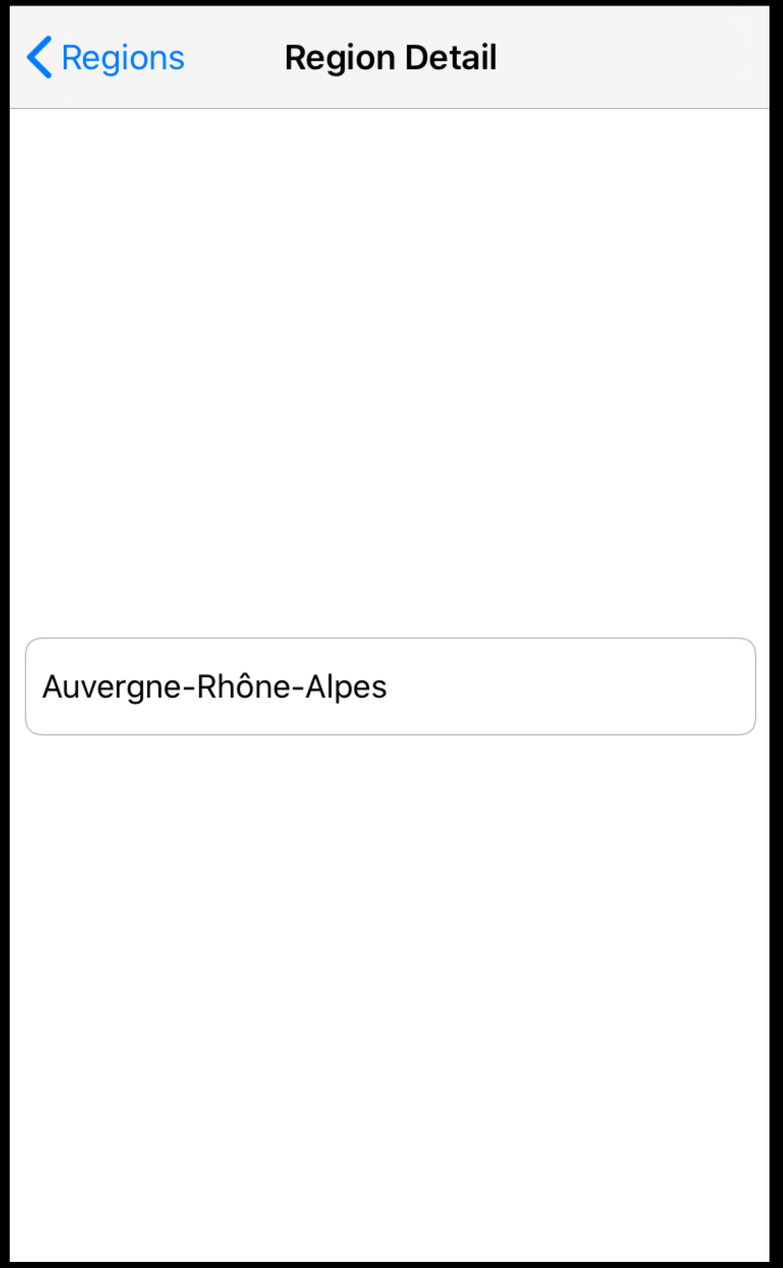
child > ViewController2

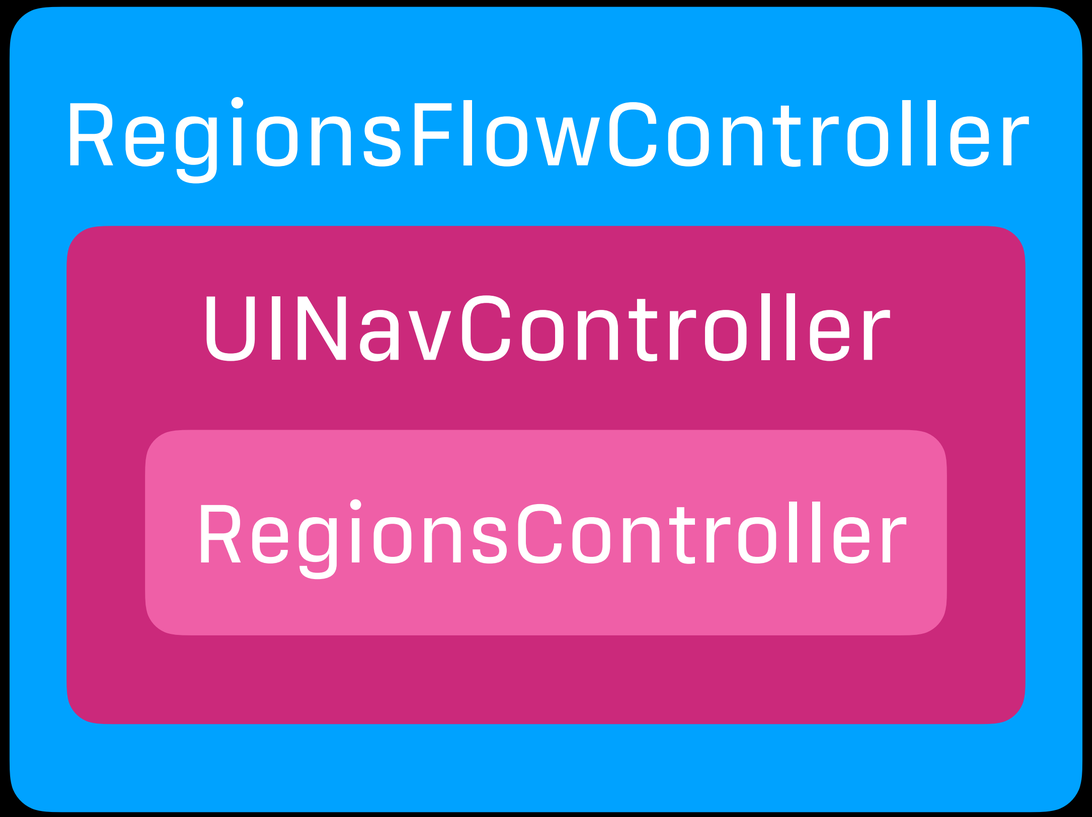
MainFlowController
(UIViewController)

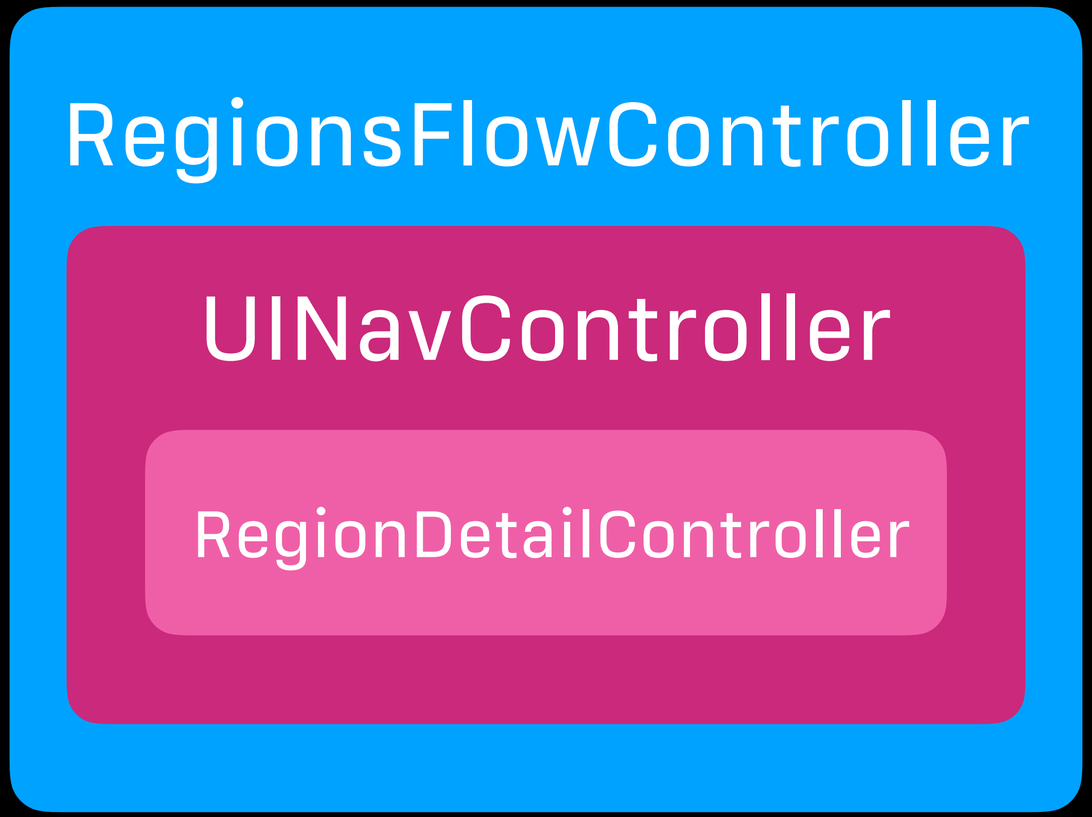
child > ViewController1

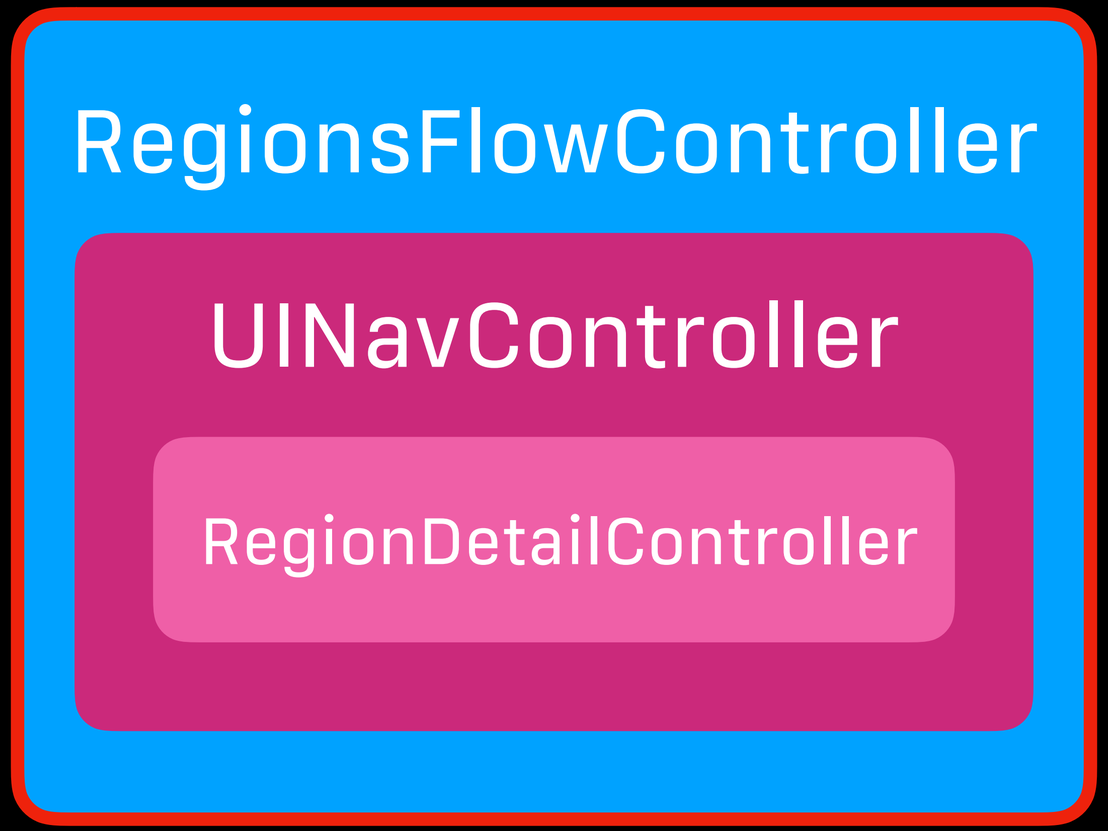
dismiss!


Regions
Auvergne-Rhône-Alpes
Bourgogne-Franche-Comté
Bretagne
Centre-Val de Loire
Corse
Grand Est
Hauts-de-France
Île-de-France
Normandie
Nouvelle-Aquitaine
Occitanie
Pays de la Loire
Provence-Alpes-Côte d'Azur


Regions
Region Detail
Auvergne-Rhône-Alpes


RegionsFlowController
UINavController
RegionsController


RegionsFlowController
UINavController
RegionDetailController


RegionsFlowController
UINavController
RegionDetailController

More

M, V or C?

ViewModel

```
struct Post: Hashable, Codable {
    let title: String
    let author: String
    let publishedAt: Date
}
```

```swift
struct PostViewModel: Hashable {
    let title: String
    let author: PersonNameComponents
    let publishedAt: String
}

extension PostViewModel {
    init(post: Post) {
        // ...
    }
}
```

MVC

Takeaways

You’ll be fine

Products are more important than the tools used to make them

Thank You

rambo.codes

@_inside

github.com/insidegui