```
1077   }
1078
1079   @end
1080
1081
```

MyModule
- XYZMyModuleHelper.h
- XYZMyModuleHelper.m
- XYZMyModuleViewController.h
- XYZMyModuleViewController.m

▼ 📁 MyToolbox
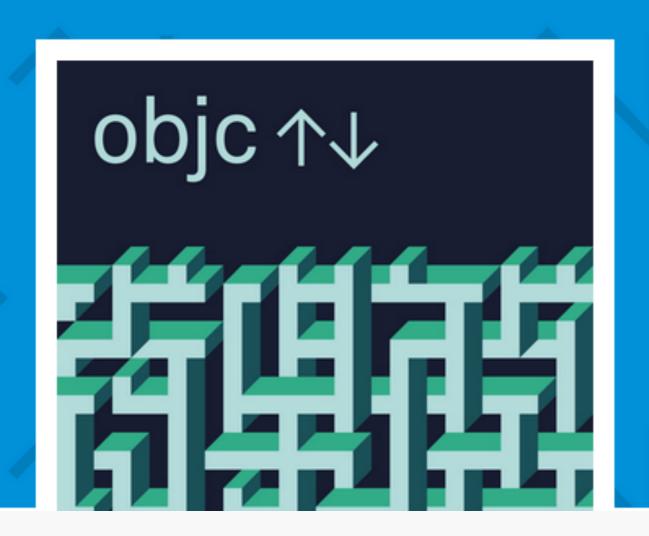　　h UIView+XYZLine.h
　　m UIView+XYZLine.m
　　h UIView+XYZThinLine.h
　　m UIView+XYZThinLine.m
　　h UIView+LineDrawingExtensions.h
　　m UIView+LineDrawingExtensions.m
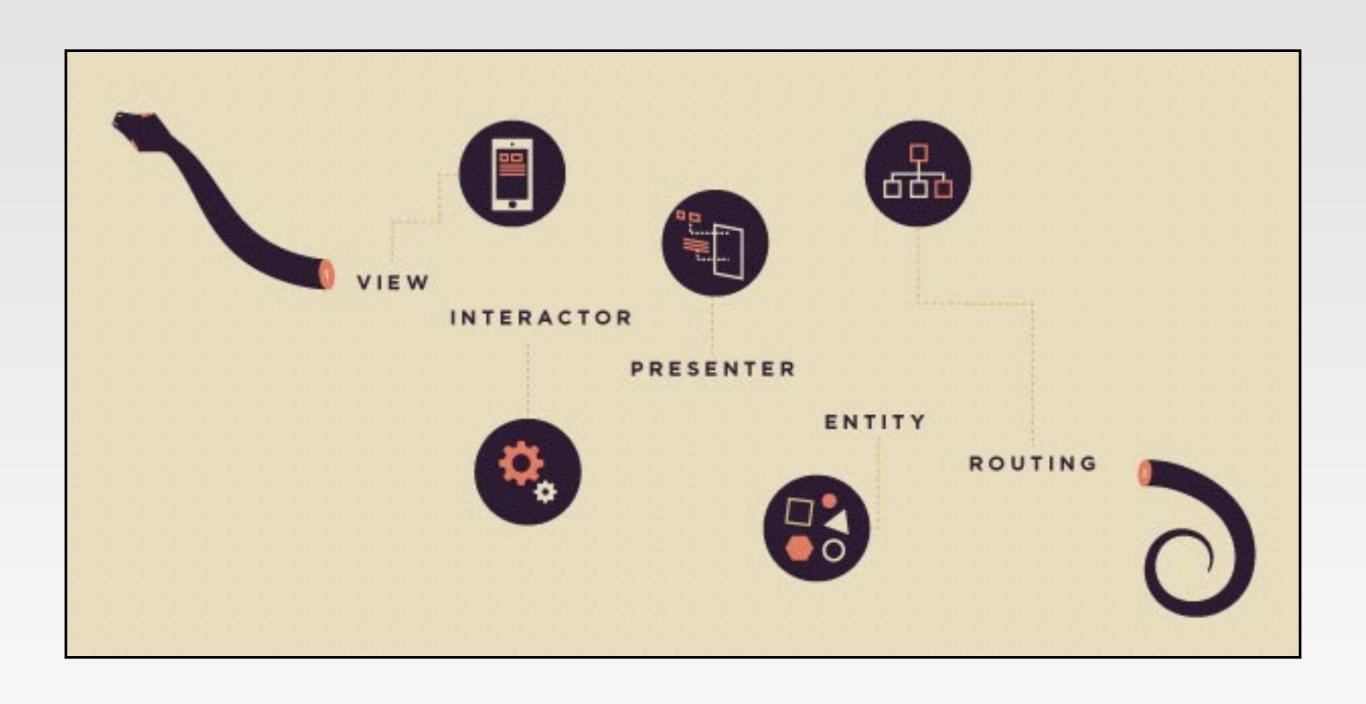
iOS Architecture

Real Life VIPER

iOS Tech Lead

Outware mobile

objc ↑↓

Architecture

Cover art by Sarah Lincoln
@sarahlincoln

VIPER

Clean Architecture

by Robert C Martin
(Uncle Bob)

VIEW

INTERACTOR

PRESENTER

ENTITY

ROUTING

Artist unknown
Sourced from objc.io article

PONSO

Plain Old NSObject

- **Used to communicate between layers**

- **Prevent high coupling**

- **Simple enough to be tested on a whim**

PONSOs can be made immutable

Talk "Enemy of the State" by Justin Spahr-Summers about value types and immutability

DIP

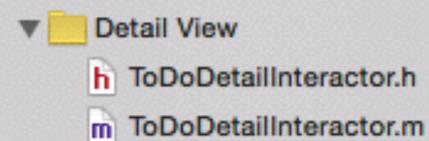## Dependency Inversion Principle

Router

Real Life Use

In a (reasonably) big enterprise project

Benefits of a Clean Architecture

- **Clear cut blocks**

- **Smaller files**

- **Better code coverage**

- **Consistency**

Clear cut blocks

Detail View
  ToDoDetailInteractor.h
  ToDoDetailInteractor.m
  ToDoDetailPresenter.h
  ToDoDetailPresenter.m
  ToDoDetailViewController.h
  ToDoDetailViewController.m
  ToDoDetailViewController.xib
  ToDoDetailViewModel.h
  ToDoDetailViewModel.m

Smaller files

Better code coverage

And better tests

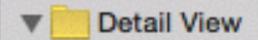# Consistency

Problems

- **Verbosity**

- **Developer On-Boarding**

- **Developer Engagement**

Verbosity

#1 complain among developers

▼ 📁 Detail View
- h ToDoDetailEventHandler.h
- h ToDoDetailInteractor.h
- m ToDoDetailInteractor.m
- h ToDoDetailModuleFactory.h
- m ToDoDetailModuleFactory.m
- h ToDoDetailPresentationModel.h
- m ToDoDetailPresentationModel.m
- h ToDoDetailPresenter.h
- m ToDoDetailPresenter.m
- h ToDoDetailUserInterface.h
- h ToDoDetailViewController.h
- m ToDoDetailViewController.m
- x ToDoDetailViewController.xib
- h ToDoDetailViewModel.h
- m ToDoDetailViewModel.m

Developer On-Boarding

Developer Engagement

Easy Improvements

- **Naming Convention**

- **Dogma**

Naming Convention

Dogma

What we learned with VIPER

V-I-P-E-R

View, Interactor, Presenter, Entity, Router

V-C-VM-M-R

View, Controller, ViewModel, Model, Router

VIPER®

VIPER™

MVC

Massive View Controller

Design Patterns

should be prescriptive

Help make small choices upfront

so you don't waste time when developing

Facts

- **Best test coverage ever**

  - **80+%**

  - **5000+ test cases**

  - **ViewControllers all smaller than 450 lines**

- **Pretty good for a big project**

  - **10+ developers**

  - **8+ months**

Forced us to think about architectural design

More than before

Epilogue

- **Processes**

- **Tooling**

  - **Xcode templates**

  - **Scripting**

  - **Anything that reduces boilerplate**

Acknowledgements

- **Jeff Gilbert and Conrad Stoll (Mutual Mobile)**

  - **"Inventors" of VIPER**

- **Robert C Martin *aka* Uncle Bob**

- **Objc.io**

  - **Sara Lincoln (Issue #13 artwork)**

Links

Founding article: http://www.objc.io/issues/13-architecture/viper/

Source: http://mutualmobile.github.io/blog/2013/12/04/viper-introduction/

MVVM: http://www.objc.io/issues/13-architecture/mvvm/

Clean Architecture by Robert C Martin

http://blog.8thlight.com/uncle-bob/2012/08/13/the-clean-architecture.html

https://vimeo.com/43612849

UIKonf 2015 - Brian Gesiak: iOS API Design: Swift Patterns - "Parameter Objects"

https://www.youtube.com/watch?v=yu6KND7dJBA

Justin Spahr-Summers - Enemy of the State

https://www.youtube.com/watch?v=7AqXBuJOJkY

Theses slides: https://github.com/jeanetienne/viper-slides

Jet

@jeanetienne

Questions?

Thank you!