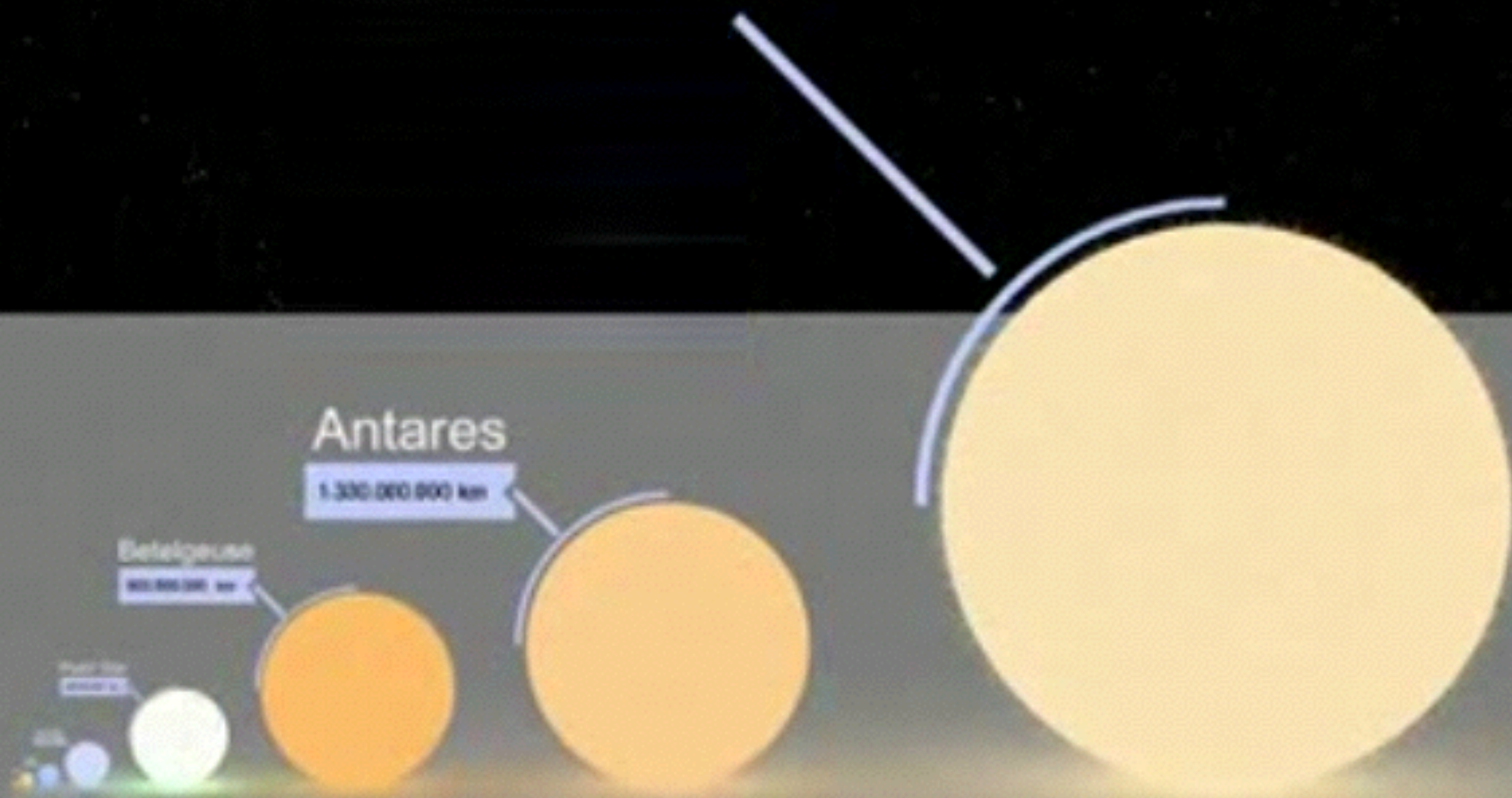




VIPER-архитектура в iOS.

Беседы на тему Massive-View-Controller.

UIViewController



Massive-View-Controller — корень всех бед

- View обычно представлен набором UIKit КОМПОНЕНТОВ
- Model - зачастую Core Data объекты и менеджер
- На все остальное есть Master Card

Massive-View-Controller — корень всех бед

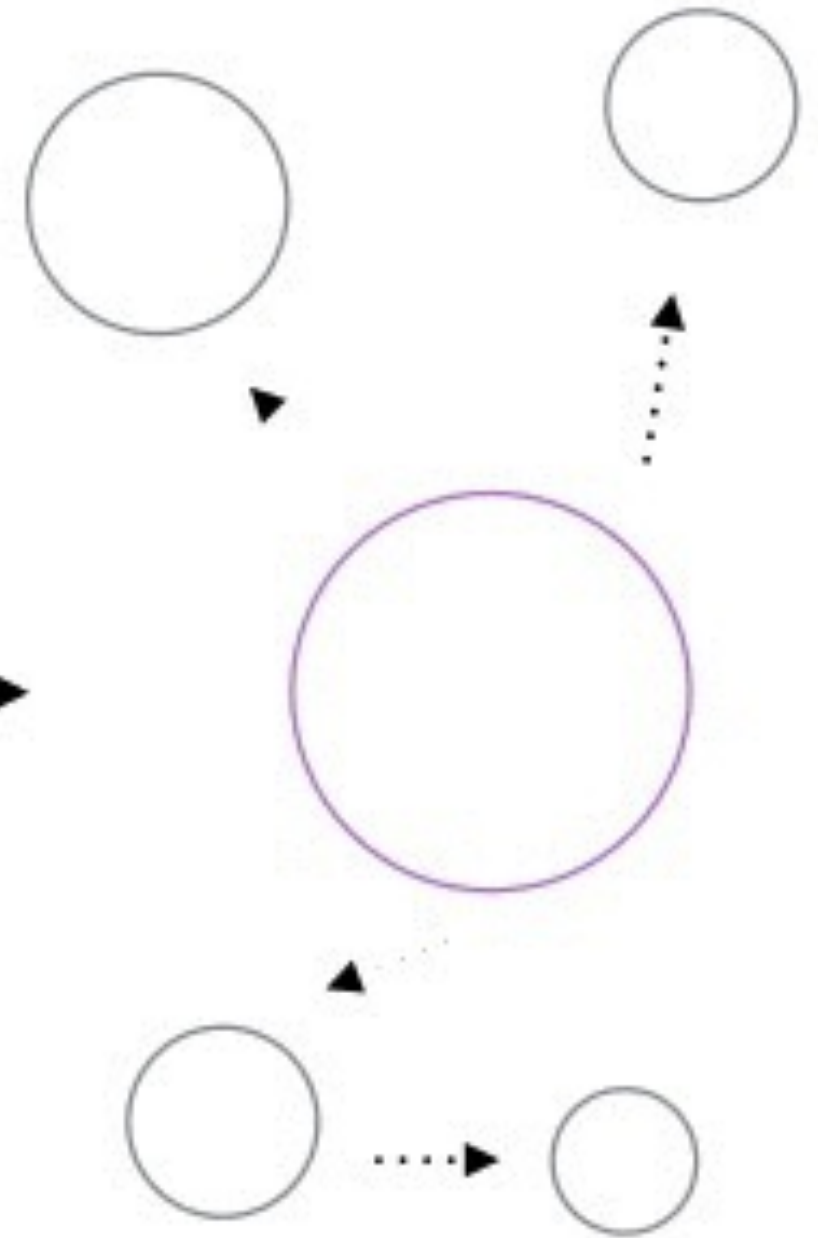
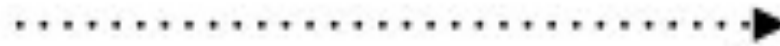
- View обычно представлен набором UIKit КОМПОНЕНТОВ
- Model - зачастую Core Data объекты и менеджер
- Все остальное достается ViewController'у

Проблема достаточно
известна

MVC on diet



Massive View Controller



Light View Controller



Colin Campbell

@Colin_Campbell



Follow

iOS architecture, where MVC stands for
Massive View Controller



Reply



Retweet



Favorite



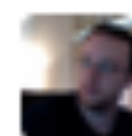
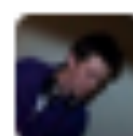
More

205

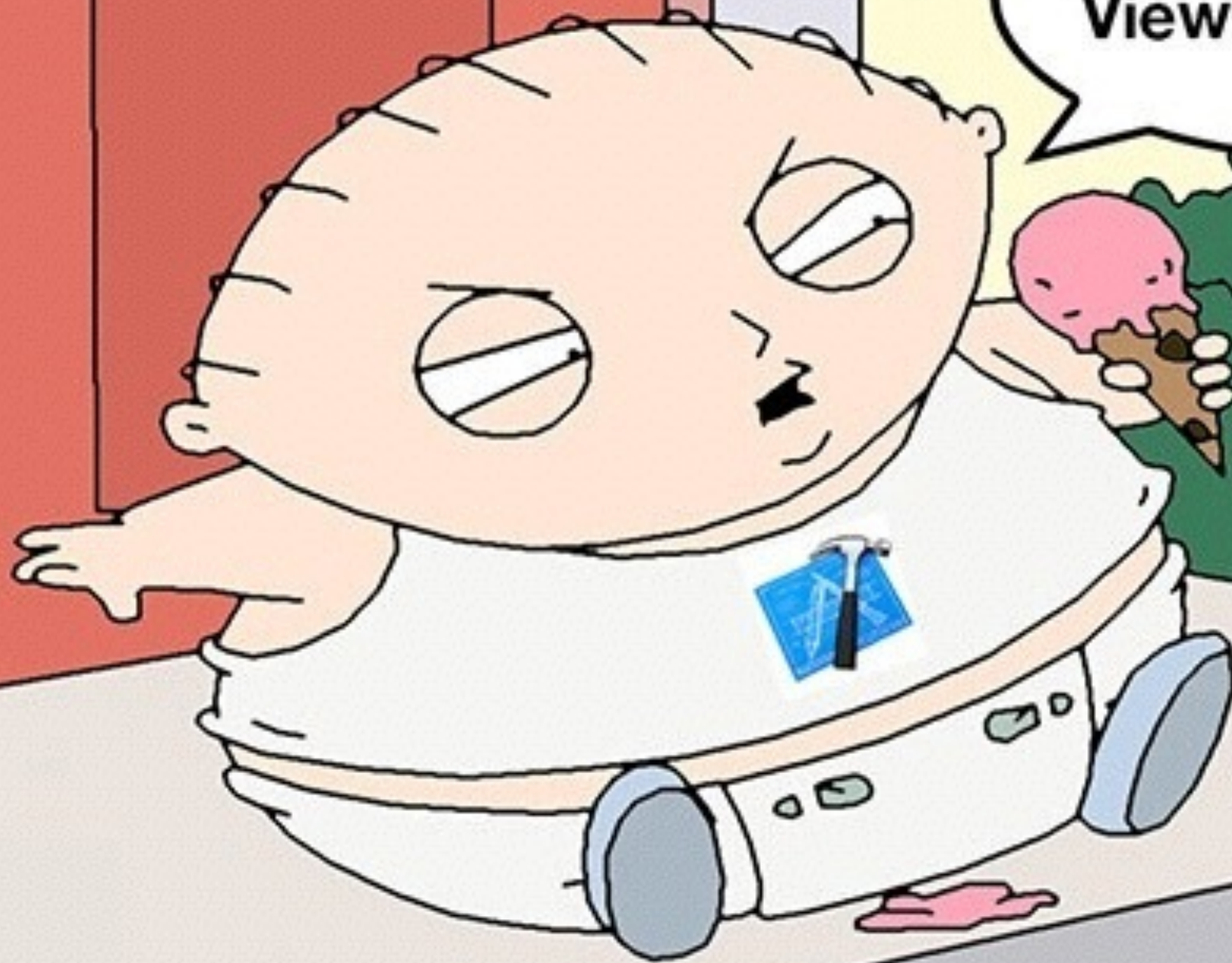
RETWEETS

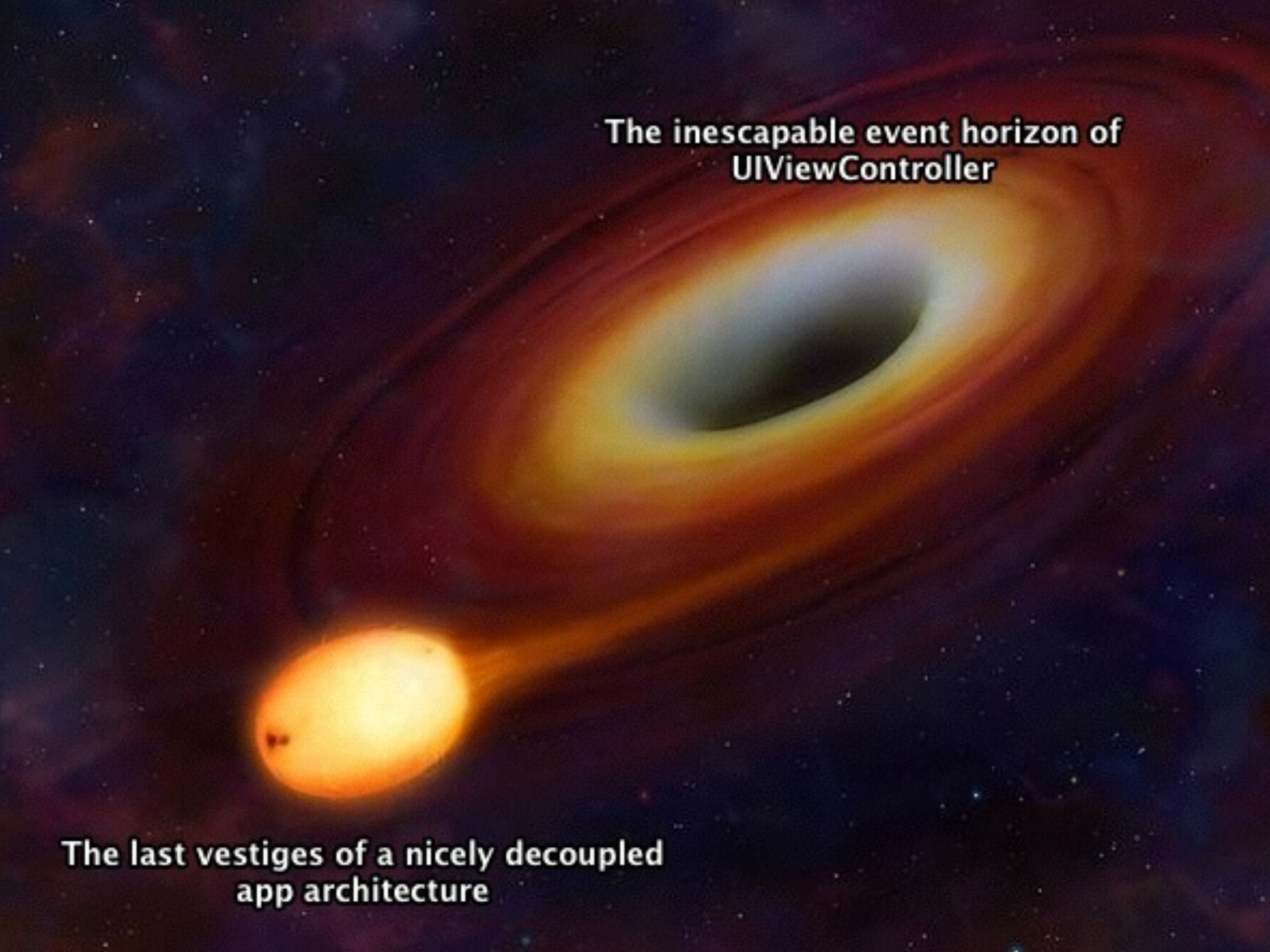
80

FAVORITES



**Damn you
ViewControllers**



A black hole with a bright orange-yellow accretion disk against a starry space background. The black hole is a dark, irregular shape in the center of the disk. The disk is a bright, glowing ring of orange and yellow light. The background is a dark blue space filled with many small white stars.

**The inescapable event horizon of
UIViewController**

**The last vestiges of a nicely decoupled
app architecture**

Massive View Controller, some call it.

Getting rid of Massive View Controller in iOS?

dreaded Massive View Controller antipattern.

Massive View Controller, so common in iOS apps.

8 Patterns to Help You Destroy Massive View Controller

Massive View Controller in the iOS world is a big problem

how to avoid making a "massive view controller."

Patterns to Avoid Massive View Controllers

Что в итоге достается ViewController'y?

- Бизнес логика
- Логика навигации
- Создание иерархии View, layout код
- Код Core Data стека (бывает и такое)
- Он выступает делегатом для ВСЕГО

Что же тогда ему делать,
если все это вынести за
его пределы?

- Загрузка view с nib или без
- Доступ к таким событиям, как -viewDidAppear, -didRotateFromInterfaceOrientation, -didReceiveMemoryWarning
- Восстановление состояния (state restoration)
- Accessibility
- Управление доступными ориентациями интерфейса
- Возможность добавить его в UINavigationController, UITabBarController
- Возможность вложить его в другой ViewController
- Участвует в цепочке событий, имея возможность отвечать на некоторые события ввода (например, касания)
- тысячи других вещей

UIViewController - это
больше View чем
Controller

Как же его спасти от
лишнего веса?

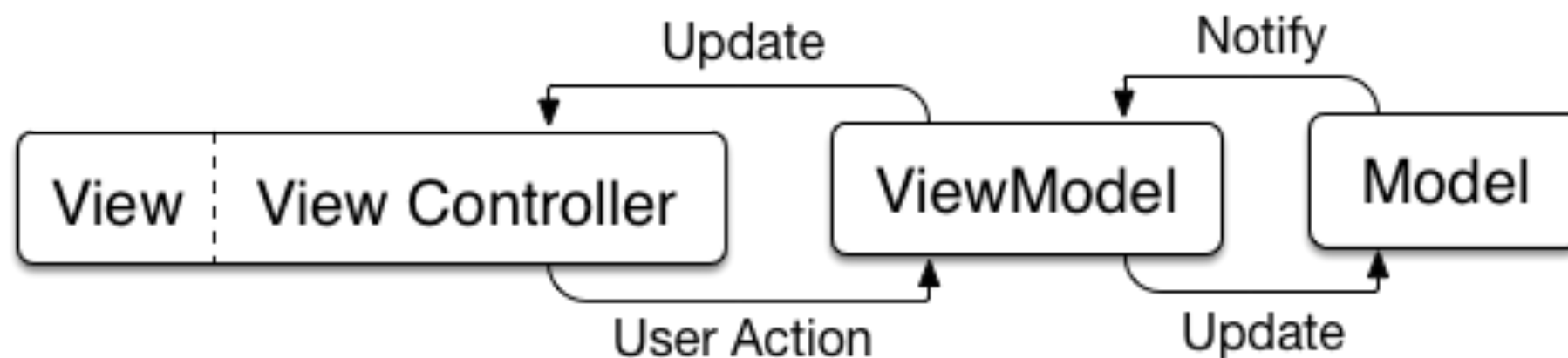
- Вынести код создания и размещения view в UIView
- Вынести код, связанный с базой данных, в DataManager
- Вынести код, связанный с работой с сетевым сервисом в APIManager
- Вынести код DataSource для UITableView в отдельный класс
- Подробнее здесь <http://www.objc.io/issue-1/lighter-view-controllers.html>

Уже в разы легче.

Теперь перейдем к более
радикальным методам.

Альтернативы MVC

Model-View-ViewModel (Model-ViewController-Presenter)



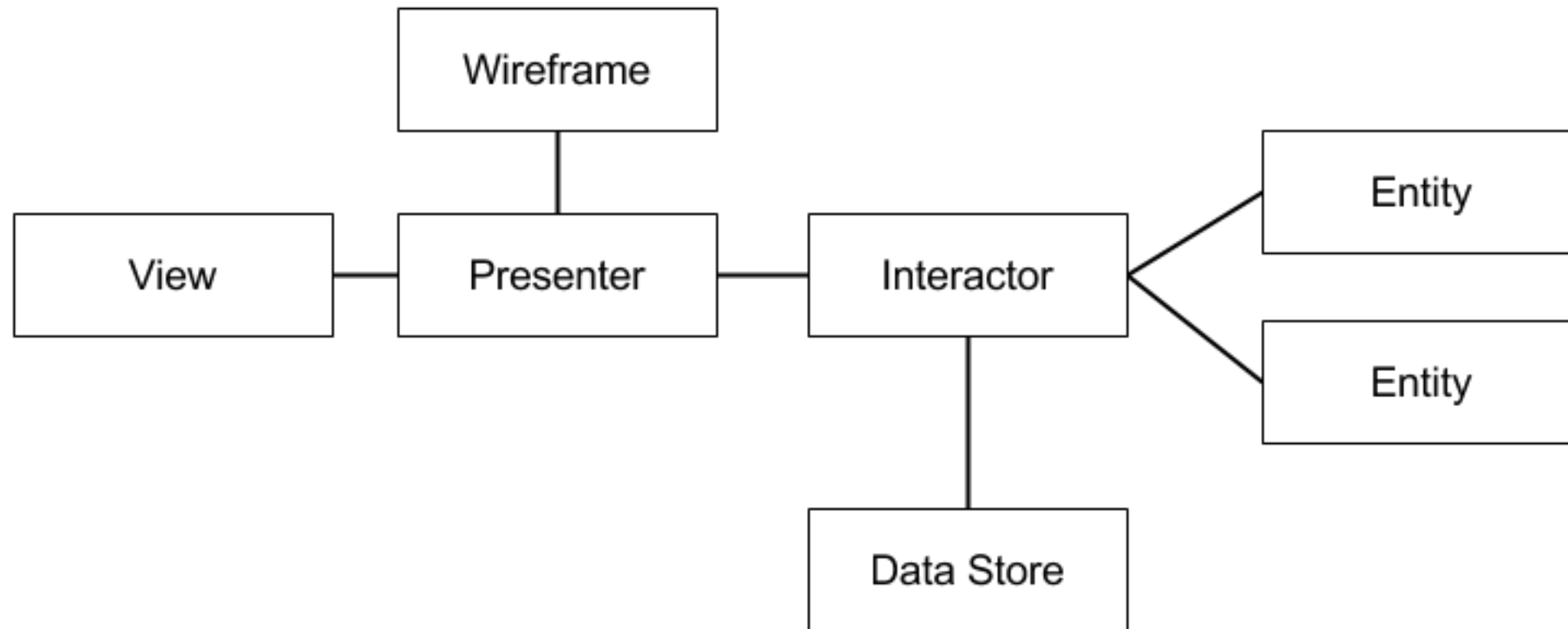
Подробнее по тщательно отобранным ссылкам:

<http://www.teehanlax.com/blog/model-view-viewmodel-for-ios/>

<https://medium.com/@ramshandilya/lets-discuss-mvvm-for-ios-a7960c2f04c7>

<https://medium.com/ios-apprentice/model-view-controller-presenter-8bb4149fa5ef>

VIPER



View

- UIViewController + (XIB или UIView subclass или и то и другое)
- Предоставляет интерфейс () для работы UI:
 - Интерфейс реализован как @protocol
 - Содержит методы в стиле showName:(NSString *)name
 - View пассивный и никогда не запрашивает данные
- Определяет интерфейс для событий, которые происходят по нажатию кнопок, по жестам и т.д.
 - Реализован как @protocol
 - Содержит методы в стиле addButtonTapped:
 - Работает по этому интерфейсу с Presenter'ом

Presenter

- NSObject
- Реализует интерфейс обработки событий, которые приходят с View
- Управляет View через его публичный интерфейс.
- Делает запросы к Interactor'у:
 - С новыми данными, полученными от View
 - Чтобы получить данные для отображения во View
- Данные - простые сущности, наследники NSObject с в основном readonly свойствами.
- Делает запросы к Wireframe для переходов между экранами

Wireframe

- NSObject
- Создает View, Presenter, Interactor и соединяет их через протоколы.
- Владеет UIWindow, UINavigationController, UIViewController
- Отвечает за переходы между экранами

Interactor

- NSObject
- Определяет конкретный use-case и содержит его бизнес логику
- Полностью независим от UI, может/должен быть покрыт юнит тестами.
- Связывает Presenter и Data Store
 - Получает данные от Presenter'а, обрабатывает их, и отправляет в Data Store
 - Отдает Presenter'у обработанные данные полученные из Data Store

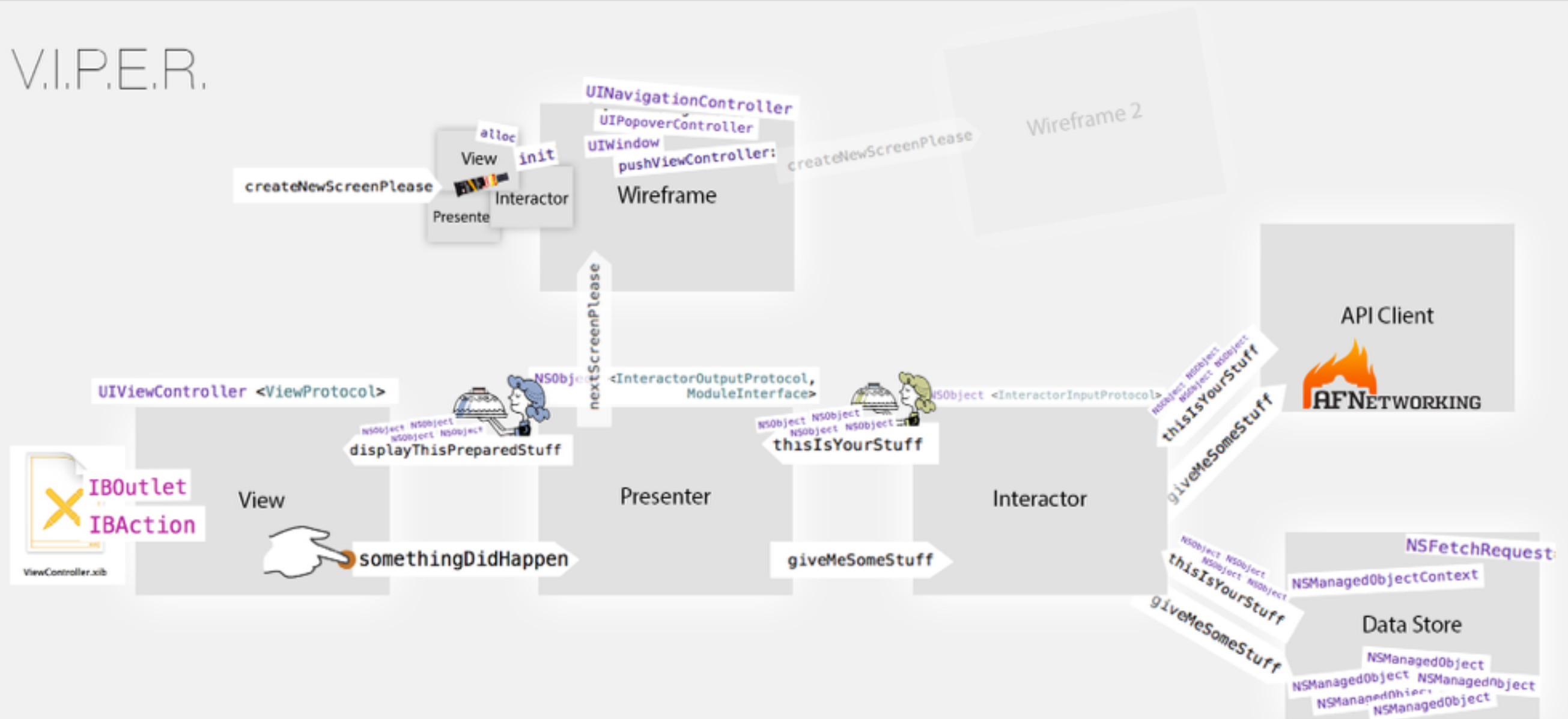
Entities (сущности)

- NSObject
- Создаются в Data Store
- Модифицируются в Interactor'е согласно бизнес-логике
- НЕ являются наследниками NSManagedObject. Ничего не знают о базе данных.
- Просто структуры данных. Вся логика должна быть в Interactor'е.

Data Store

- Отвечает за предоставление сущностей Interactor'y
- Содежит логику хранения данных
- Может быть независимо протестирован

V.I.P.E.R.



Примеры

VIPER TODO, Counter

<https://github.com/mutualmobile/VIPER-TODO>

<https://github.com/mutualmobile/Counter>


<https://github.com/tar500/viper-counter-app>

Should I use Viper architecture for my next iOS application, or it is still very new to use?



Pedro Piñera Buendia

1 upvote by Kuba Henryk Ratajczak.


Yes, I totally recommend it. I applied it in Redbooth for our mobile apps and I even developed a module to generate VIPER modules from a template: [pepibumur/viper-module-generator](https://github.com/pepibumur/viper-module-generator) . Although it's a heavy task at first having to refactor and split every component into multiple ones you'll see that at the end, it'll provide a lot of advantages to your project. More testable and less coupled code, single responsibilities componentes, ...


a module to generate VIPER modules


VIPER module generator (vipergen) + “light” template


<https://github.com/tar500/viper-module-generator>

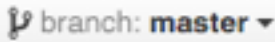
Gem to generate VIPER modules to use them in your Objective-C/Swift projects — Edit


 84 commits


 1 branch

 4 releases

 1 contributor


 **viper-module-generator** / +

 Pull Request








 Compare

This branch is 4 commits ahead of teambox:master

Update README.md

 tar500 authored a minute ago

latest commit [a1b03b0e1d](#)

 VIPERGenDemo	Version 0.2.0	2 months ago
 bin	Almost working at all	3 months ago
 lib	Add light template (Objective-C only)	3 hours ago
 slides/viper_slides_codemotion.key	Fixed misspelling issue	2 months ago
 spec	Fixed tests	2 months ago
 .gitignore	Created initial structure	3 months ago
 .rspec	Added tests	3 months ago

?

:)

Ссылки и материалы

Компания “разработчик” <http://mutualmobile.github.io/blog/2013/12/04/viper-introduction/>

Статья на крутом сайте <http://www.objc.io/issue-13/viper.html> (смотрите еще доп. ссылки в конце статьи!)

Опыт одной из команд <https://medium.com/brigade-engineering/brigades-experience-using-an-mvc-alternative-36ef1601a41f>

Как убить Massive-View-Controller <http://khanlou.com/2014/09/8-patterns-to-help-you-destroy-massive-view-controller/>

Model-View-Whatever <http://khanlou.com/2014/03/model-view-whatever/>

Кто ты, UIViewController? <http://doing-it-wrong.mikeweller.com/2013/06/ios-app-architecture-and-tdd-1.html>

Также смотрите ссылки в этой презентации