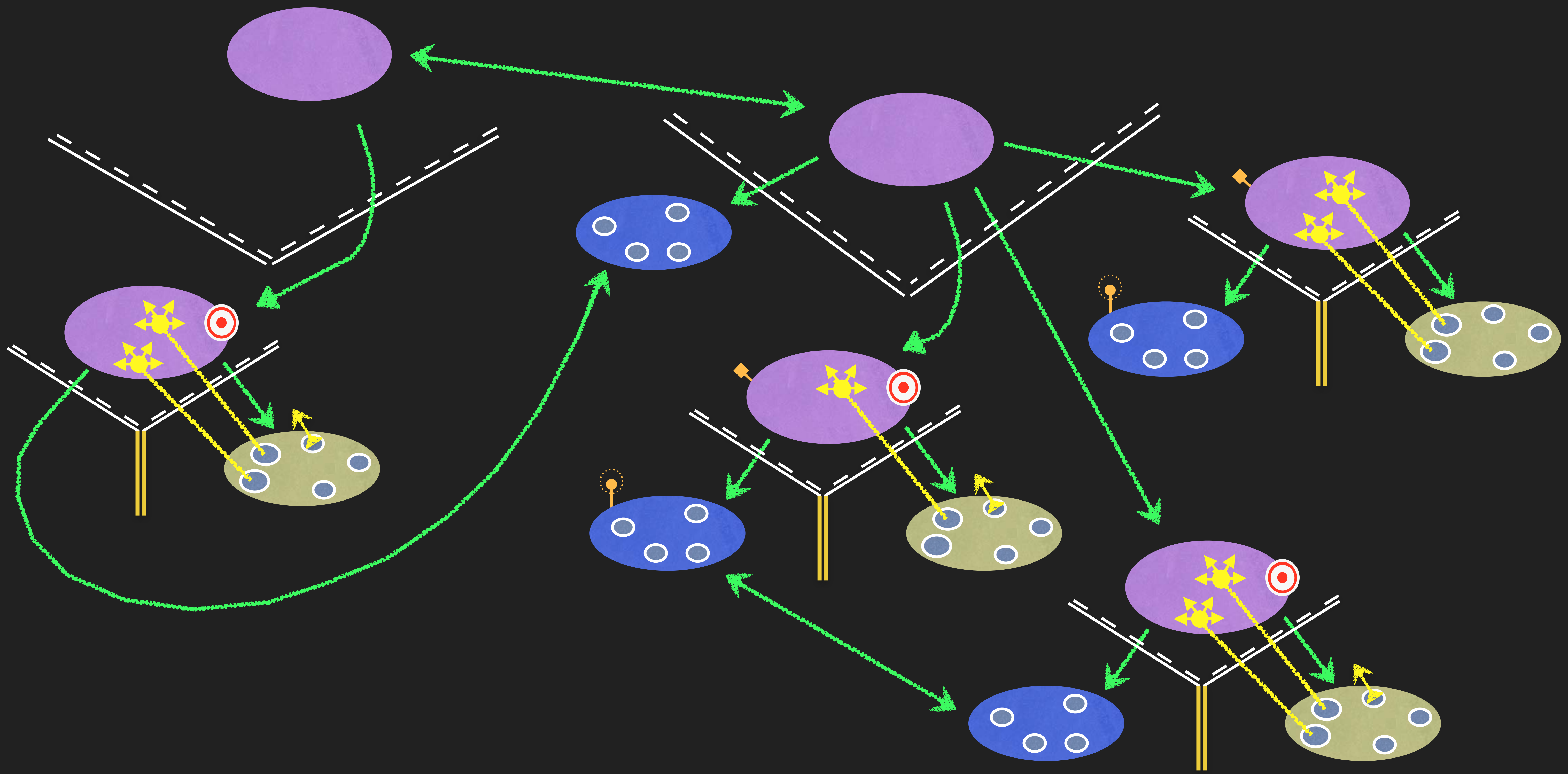
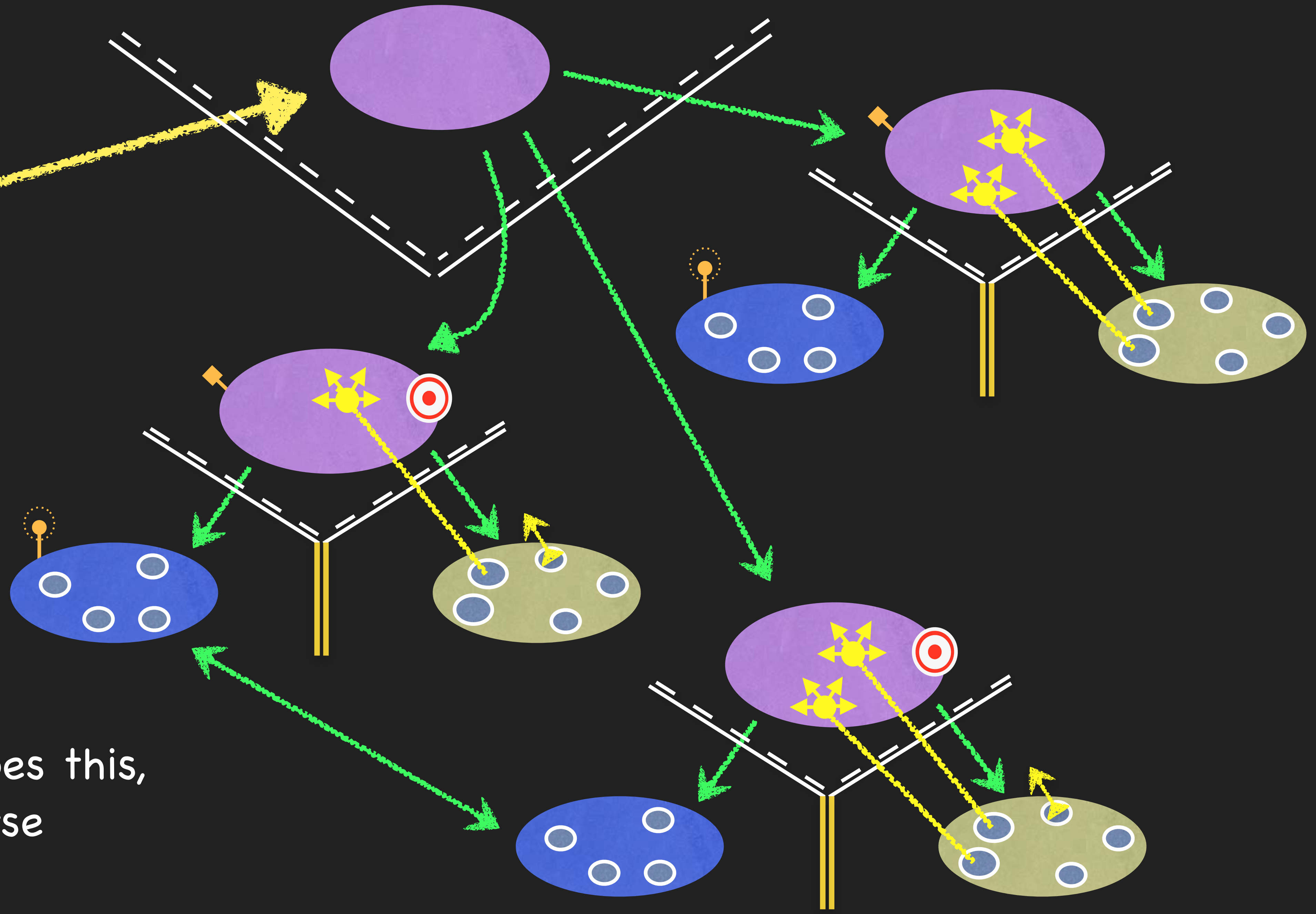
MVCs working together



Multiple MVCs

- To build more powerful applications

  To do this, we must combine MVCs …



iOS provides some Controllers whose View is "other MVCs" *

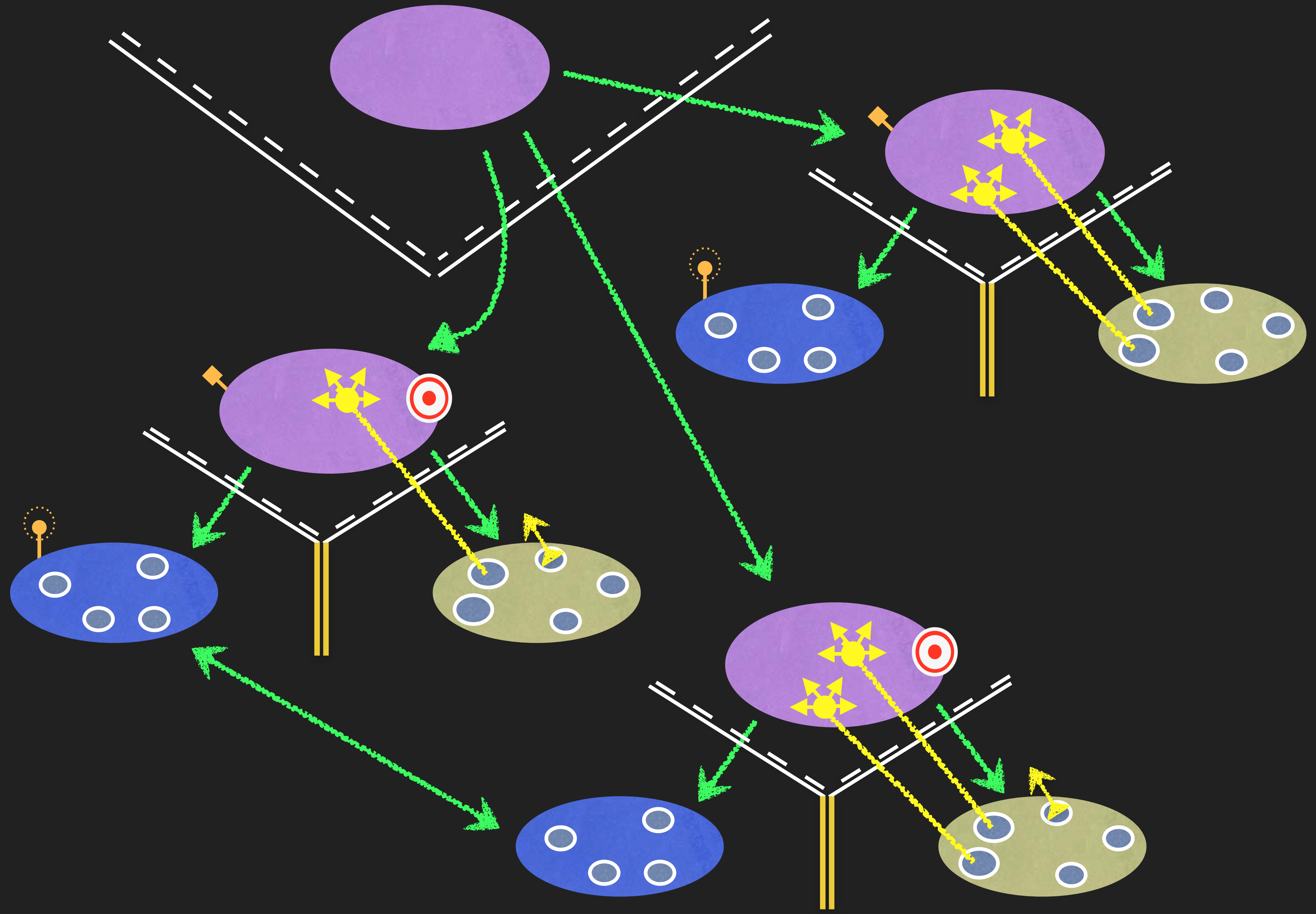
* you could build your own Controller that does this, but we're not going to cover that in this course

Multiple MVCs

- Time to build more powerful applications

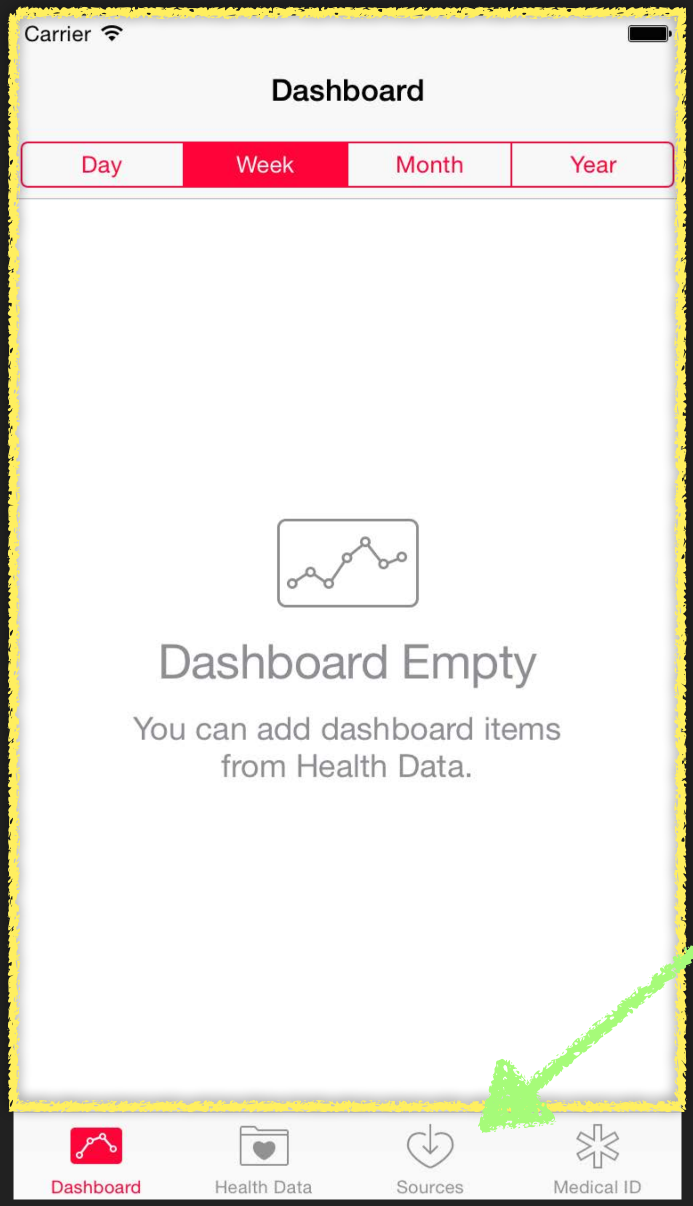
  To do this, we must combine MVCs ...

  iOS provides some Controllers
  whose View is "other MVCs"
  Examples:
  `UITabBarController`
  `UINavigationController`



UITabBarController

- It lets the user choose between different MVCs …

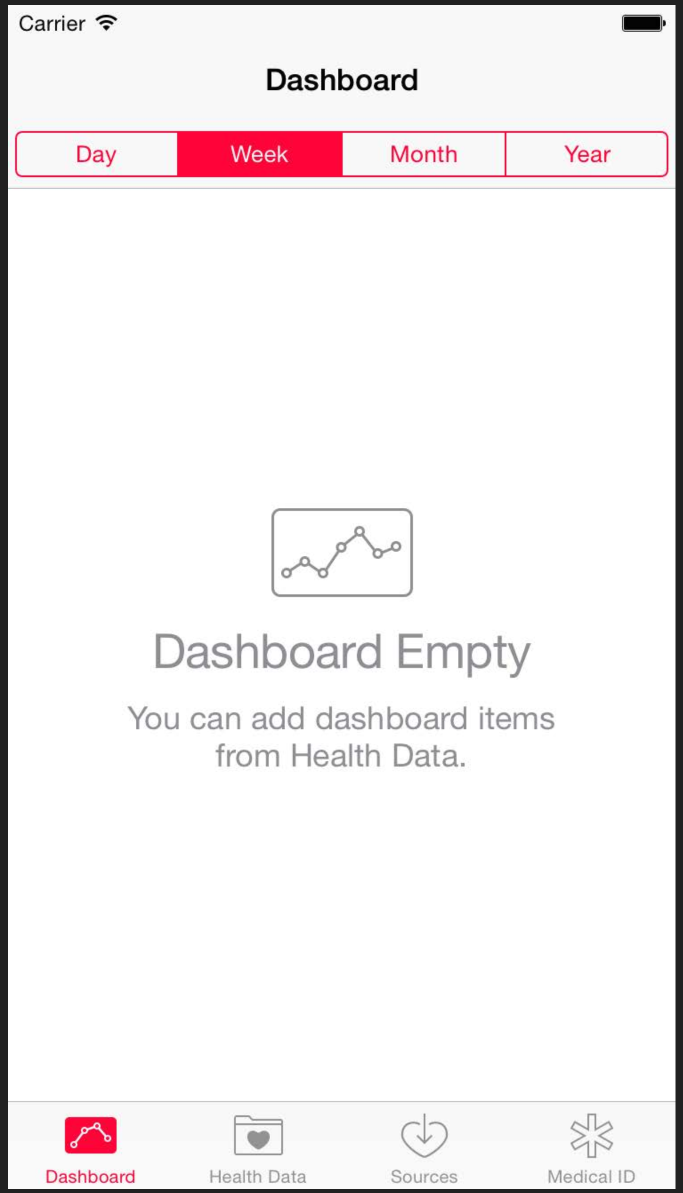


A "Dashboard" MVC

The icon, title and even a "badge value" on these
is determined by the MVCs themselves via their property:
`var tabBarItem: UITabBarItem!`
But usually you just set them in your storyboard.

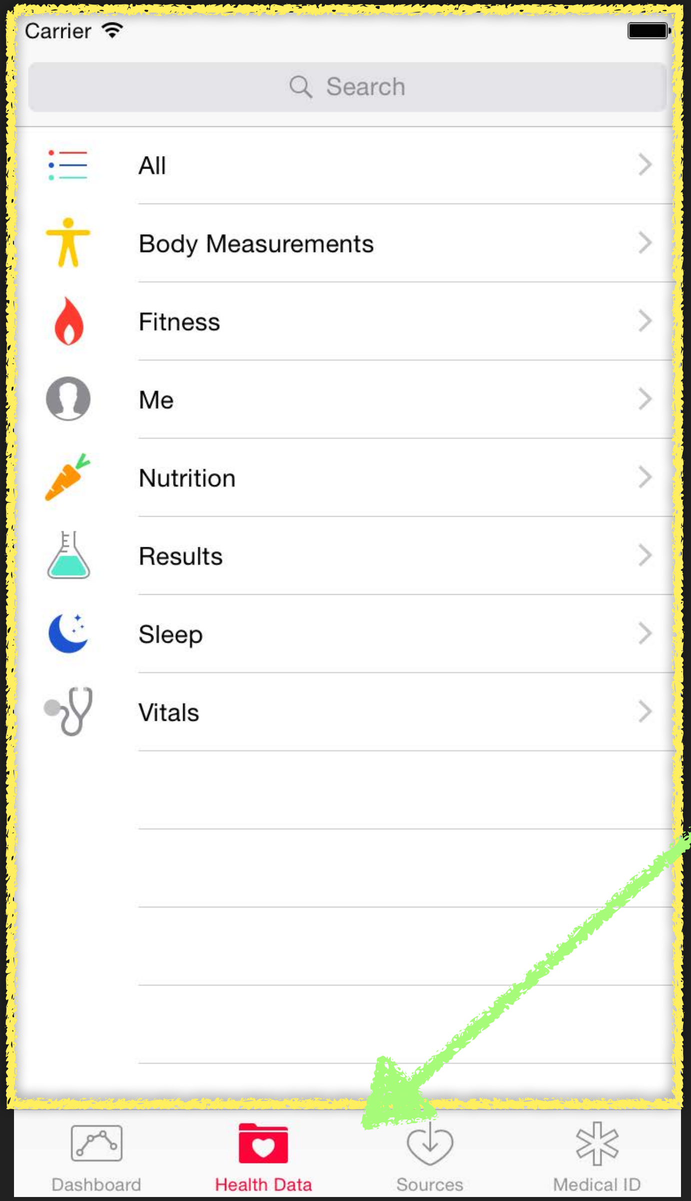
UITabBarController

- It lets the user choose between different MVCs …



Dashboard Empty

You can add dashboard items from Health Data.

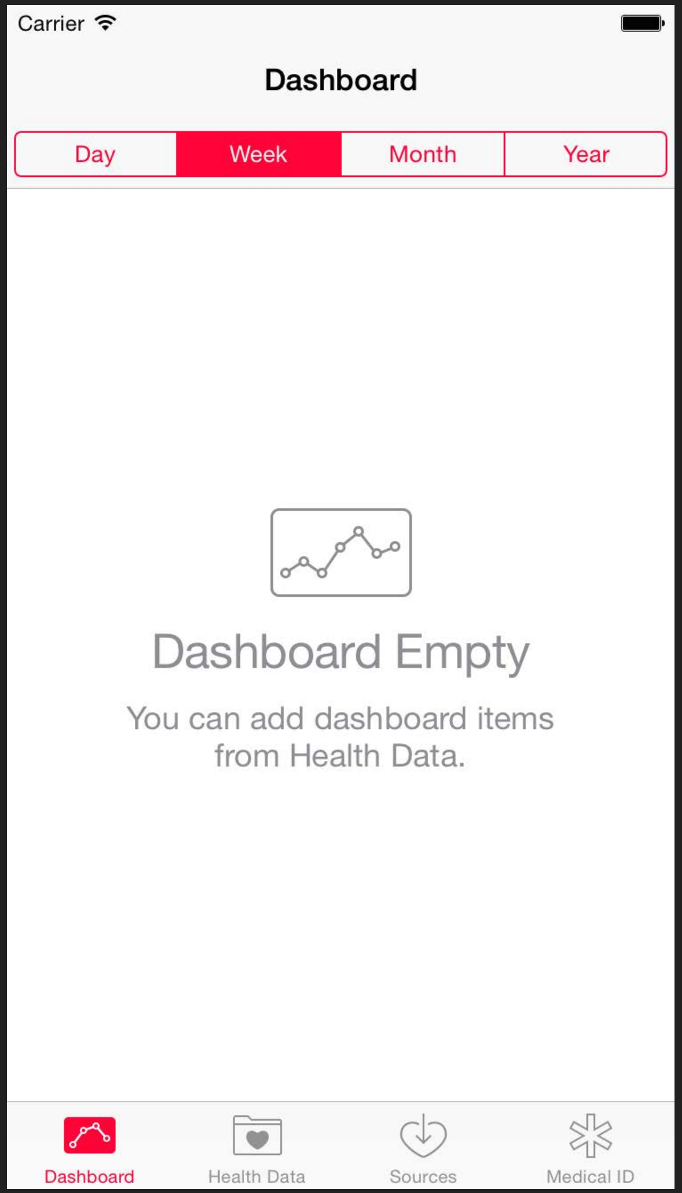


A "Health Data" MVC

If there are too many tabs to fit here,
the UITabBarController will automatically
present a UI for the user to manage the overflow!

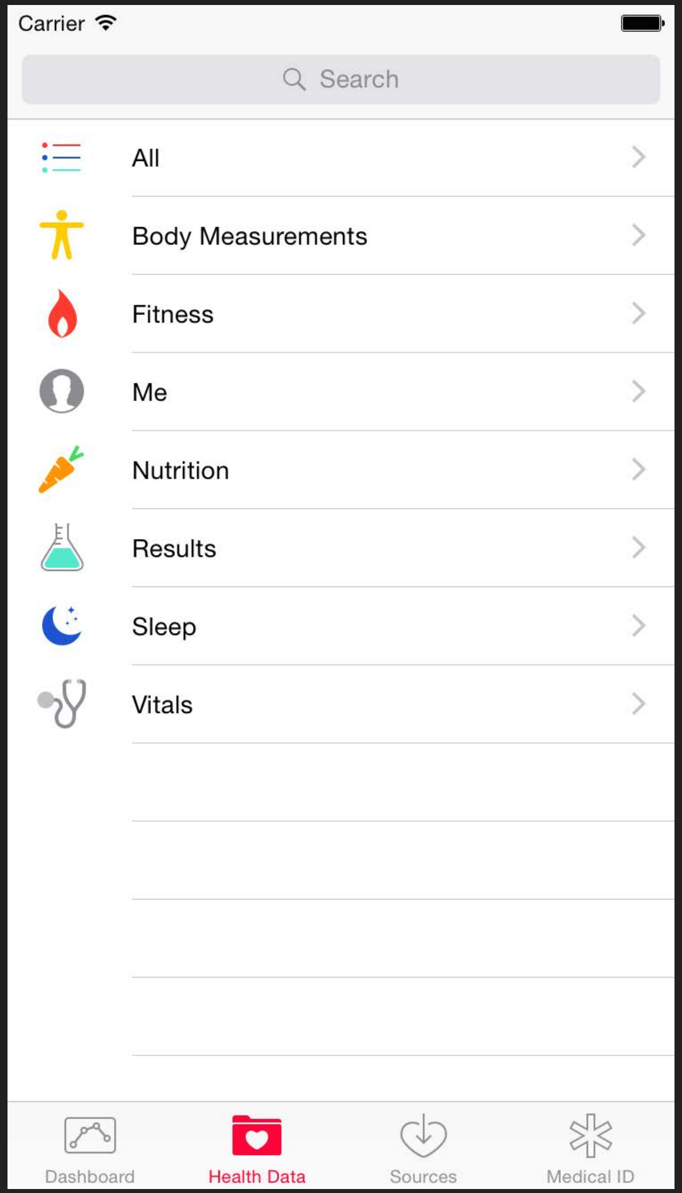
UITabBarController

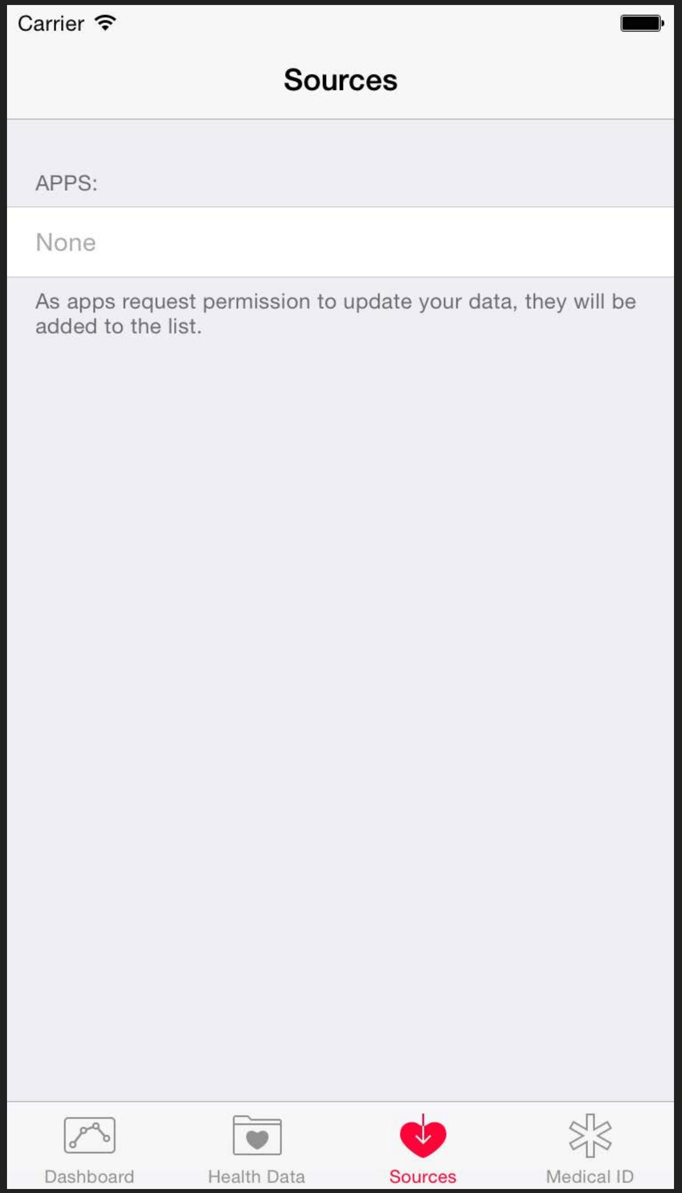
- It lets the user choose between different MVCs …



Dashboard Empty

You can add dashboard items from Health Data.





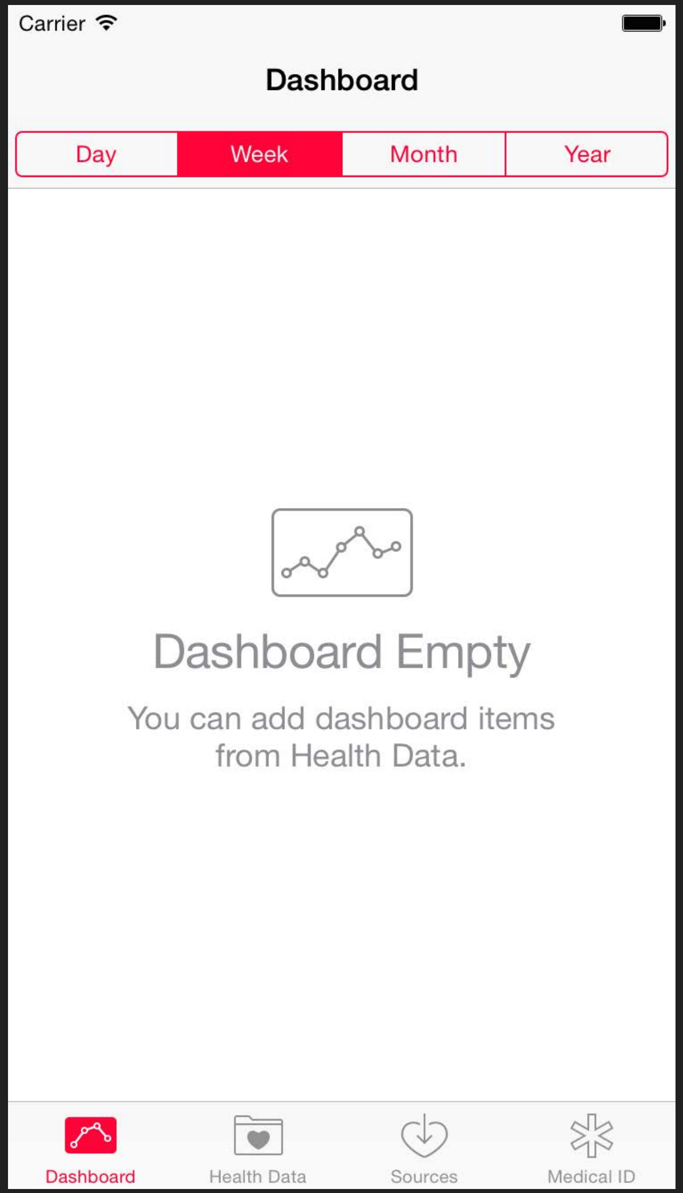
APPS:

None

As apps request permission to update your data, they will be added to the list.

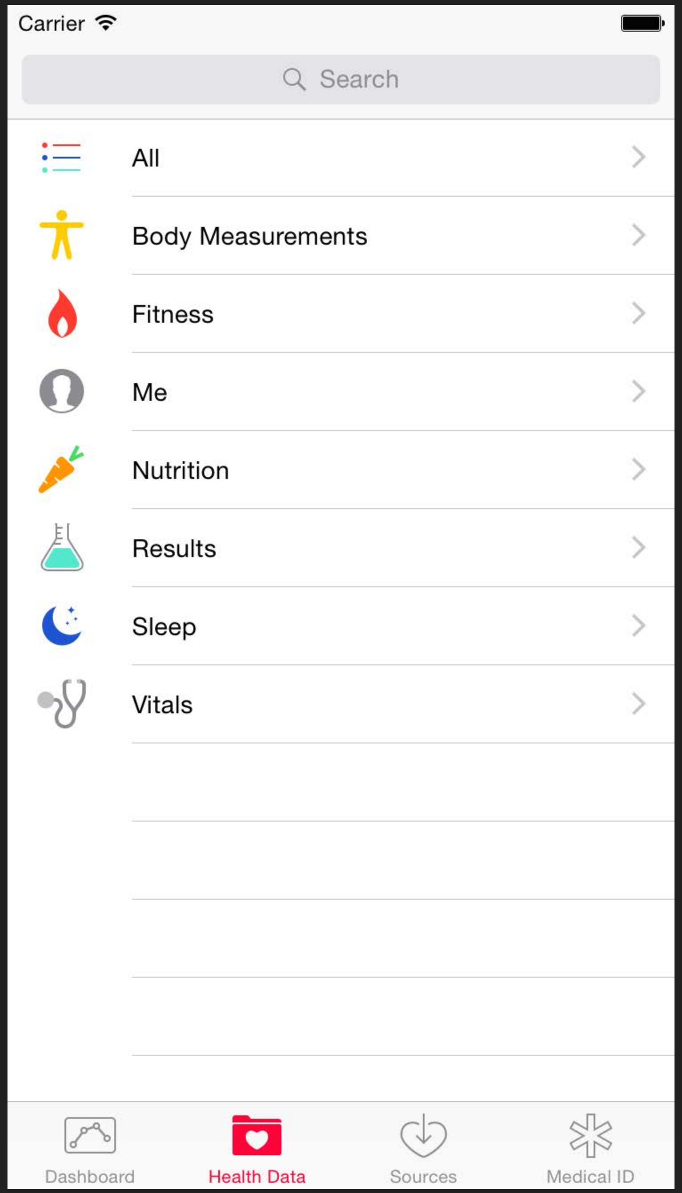
UITabBarController

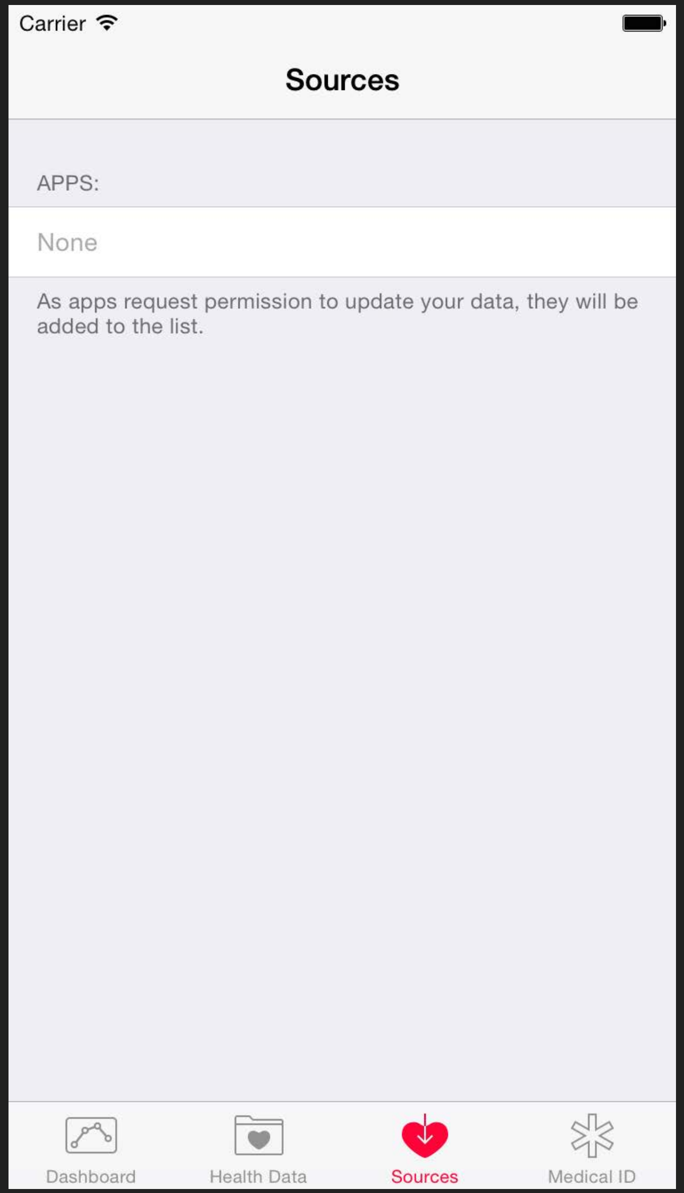
- It lets the user choose between different MVCs …



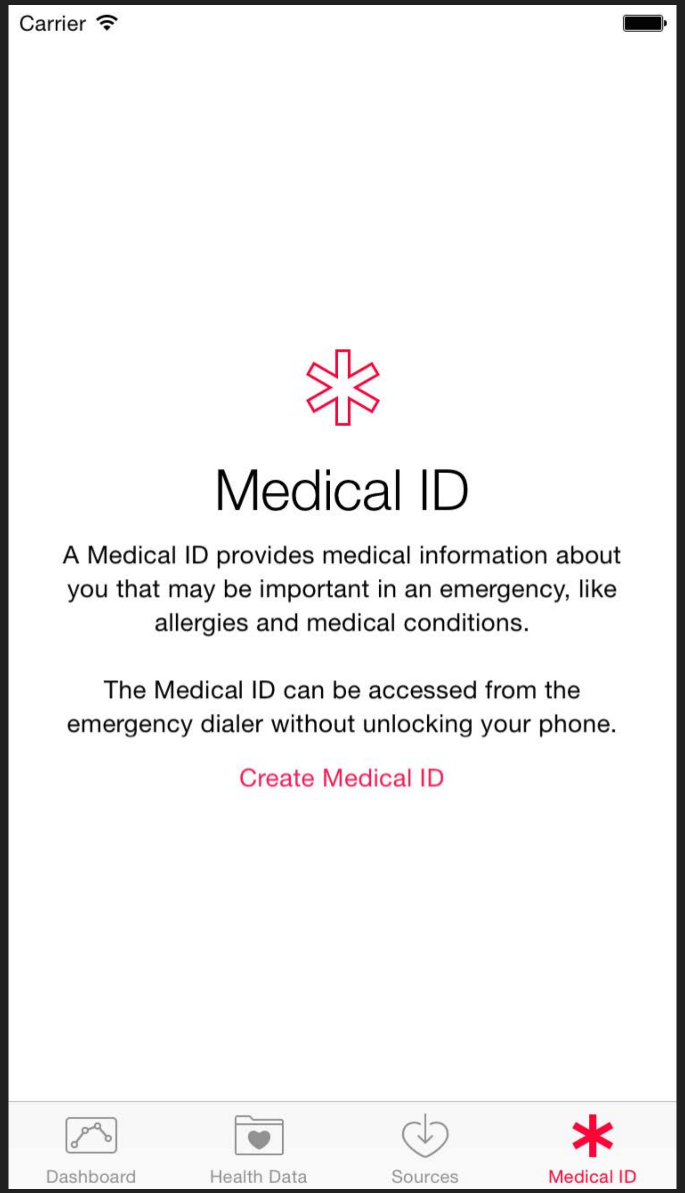
Dashboard Empty

You can add dashboard items from Health Data.





As apps request permission to update your data, they will be added to the list.



Medical ID

A Medical ID provides medical information about you that may be important in an emergency, like allergies and medical conditions.

The Medical ID can be accessed from the emergency dialer without unlocking your phone.

Create Medical ID

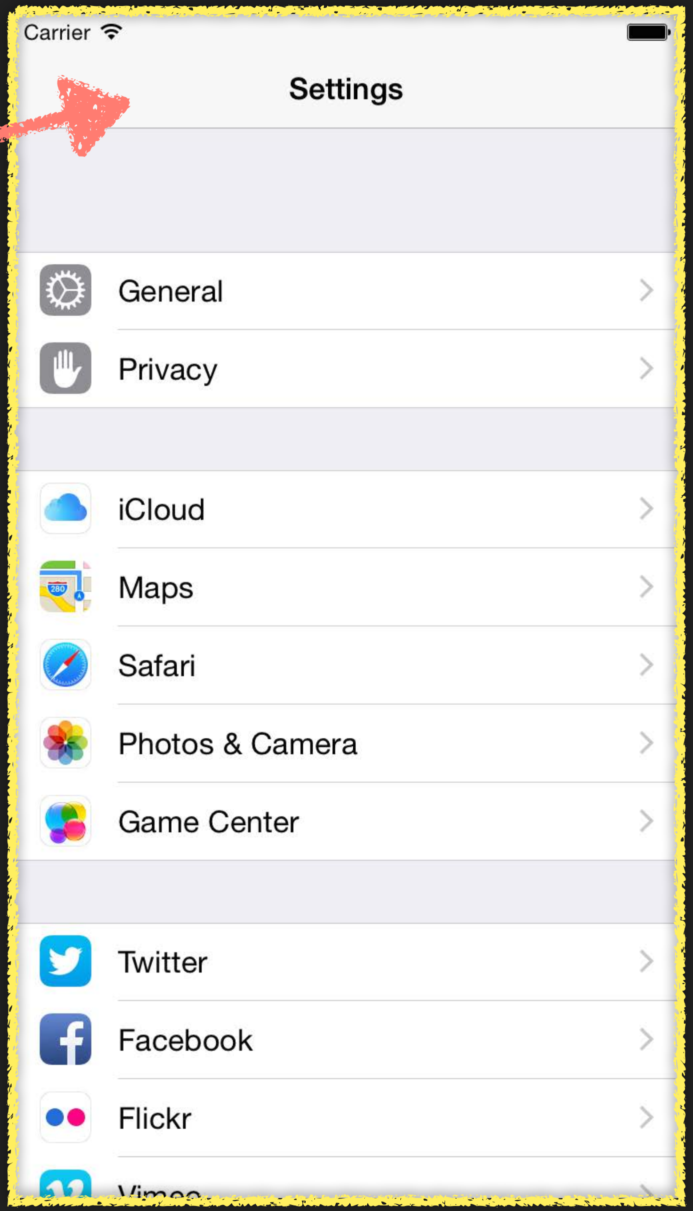
UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …

This top area is drawn by the UINavigationController

But the contents of the top area (like the title or any buttons on the right) are determined by the MVC currently showing (in this case, the "All Settings" MVC)

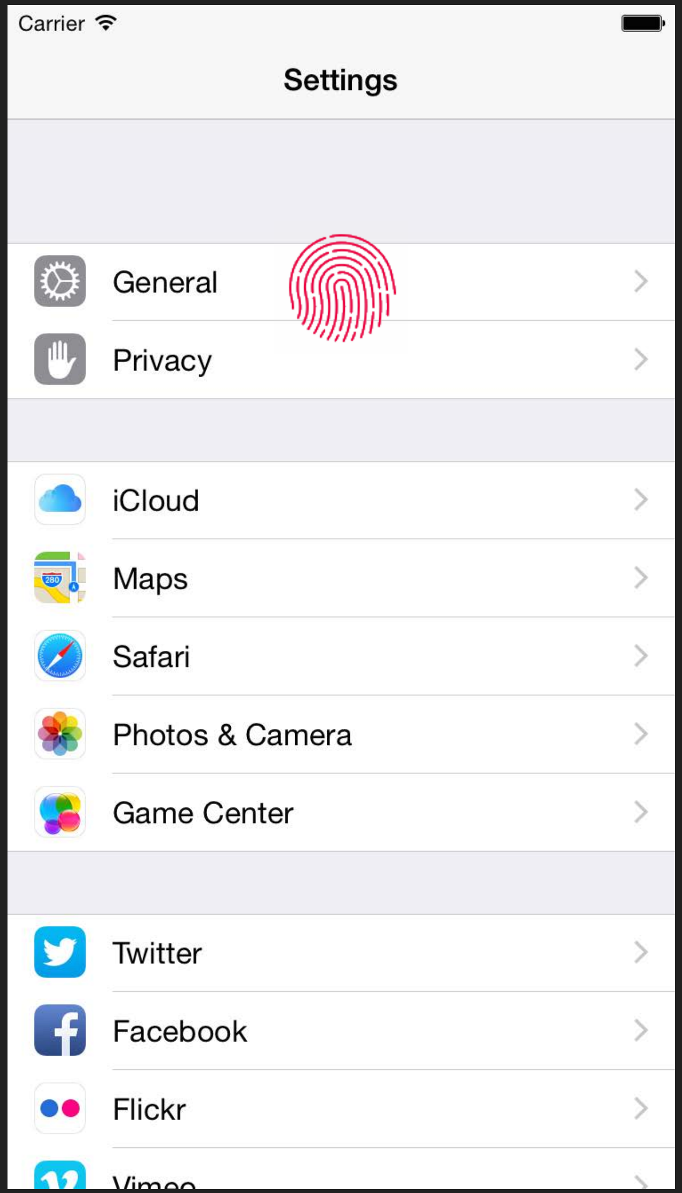
Each MVC communicates these contents via its UIViewController's navigationItem property



An "All Settings" MVC

UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …



Carrier

Settings

General

Privacy

iCloud

Maps

Safari

Photos & Camera

Game Center

Twitter

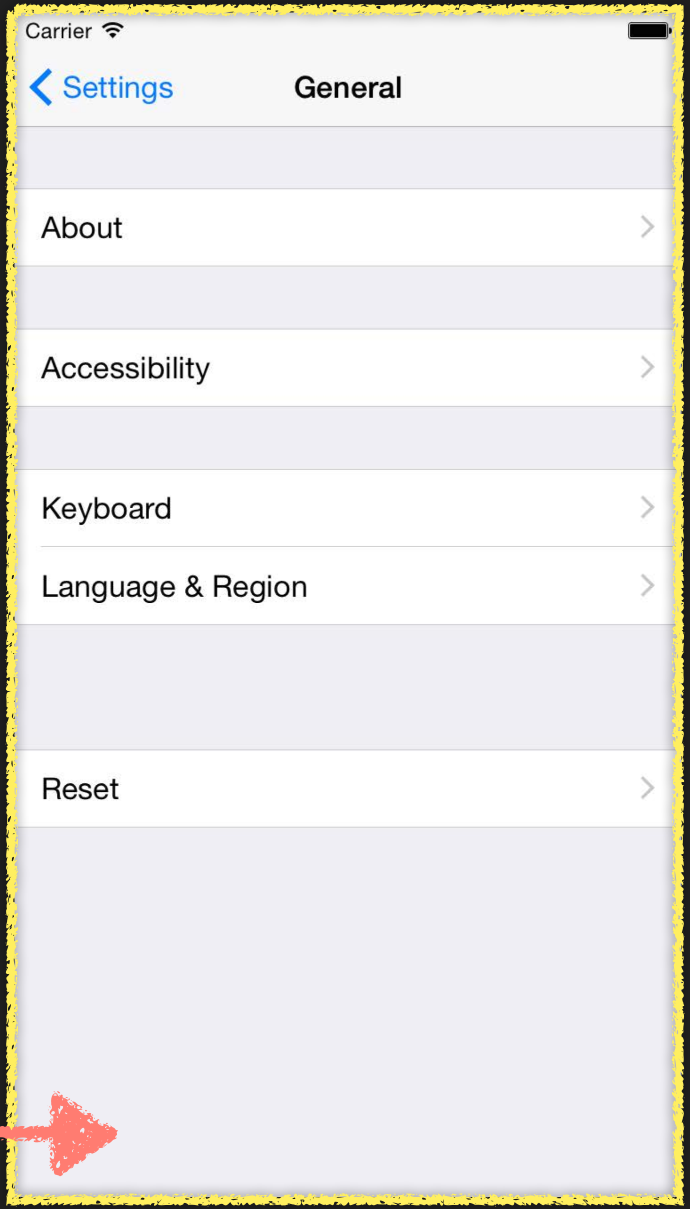
Facebook

Flickr

Vimeo

UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …



Carrier

Settings | General

About

Accessibility

Keyboard

Language & Region

Reset

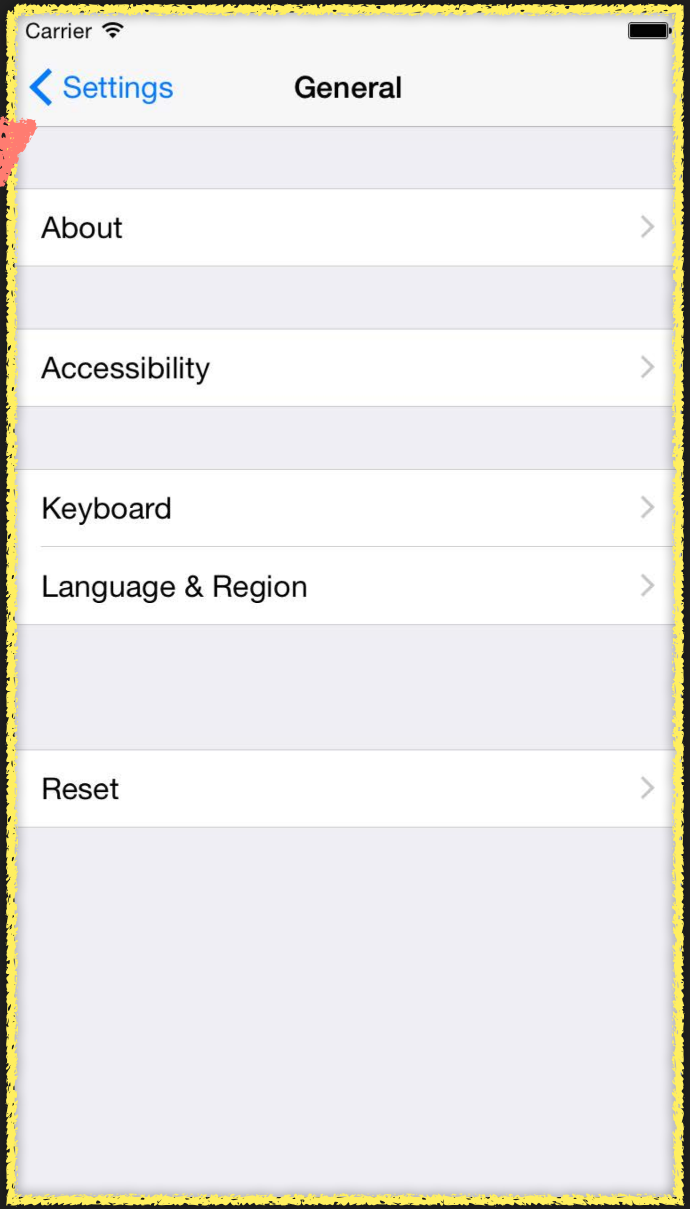
A "General Settings" MVC

It's possible to add MVC-specific buttons here too via the `UIViewController`'s `toolbarItems` property

UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …

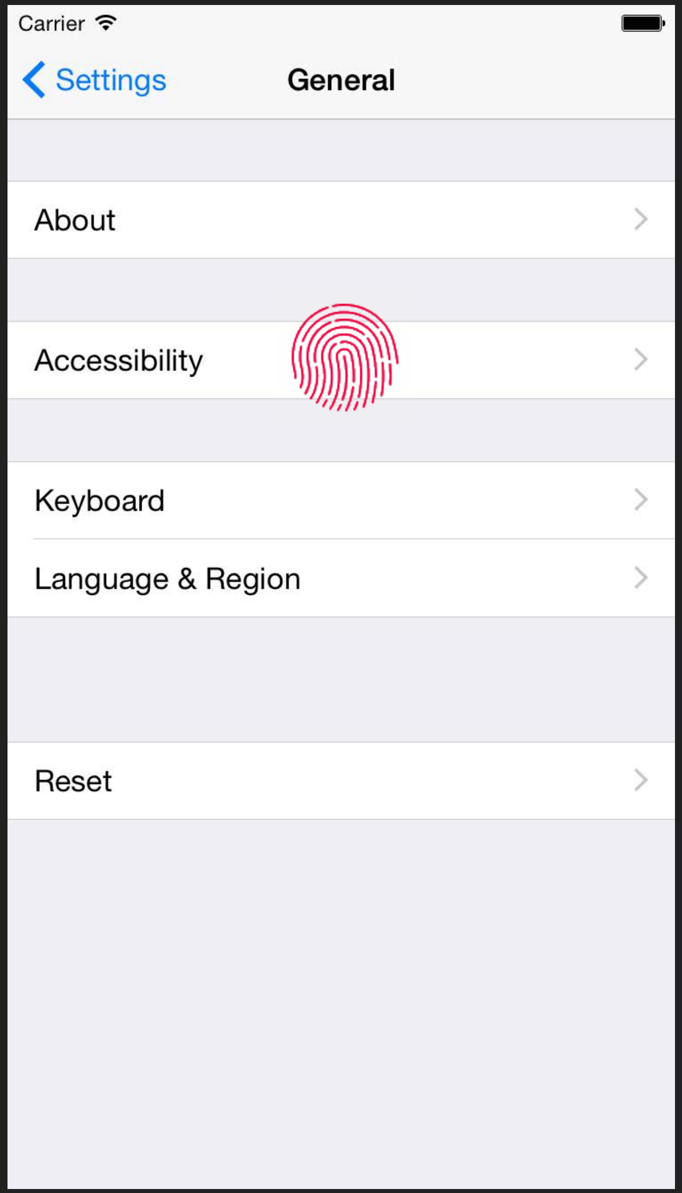
Notice this "back" button has appeared. This is placed here automatically by the UINavigationController.



A "General Settings" MVC

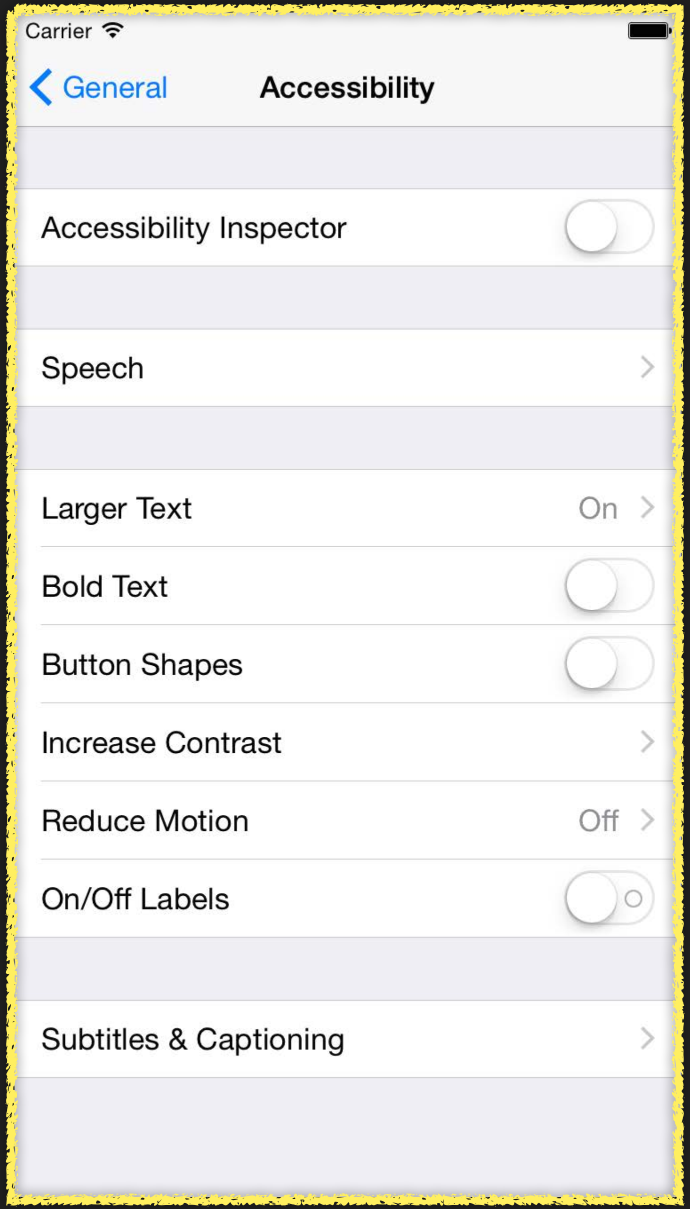
UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …



UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …



Accessibility Inspector ○

Speech >

Larger Text On >

Bold Text ○

Button Shapes ○

Increase Contrast >

Reduce Motion Off >

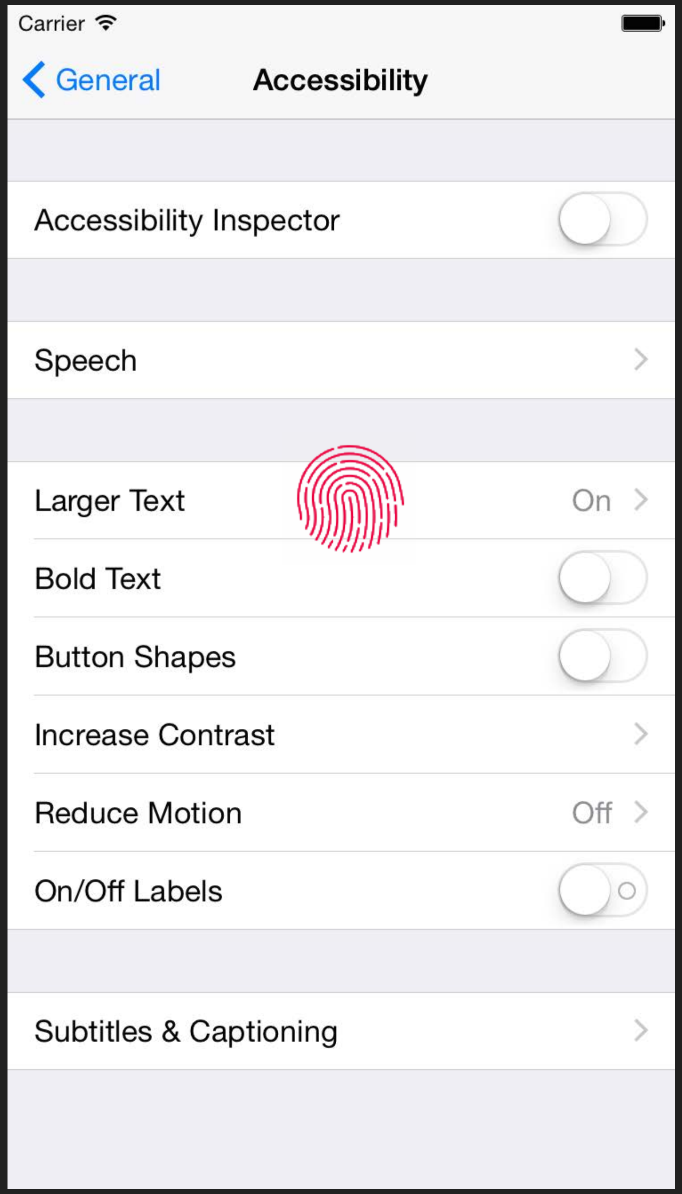
On/Off Labels ○

Subtitles & Captioning >

An "Accessibility" MVC

UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …



Accessibility Inspector ○

Speech >

Larger Text On >

Bold Text ○

Button Shapes ○

Increase Contrast >

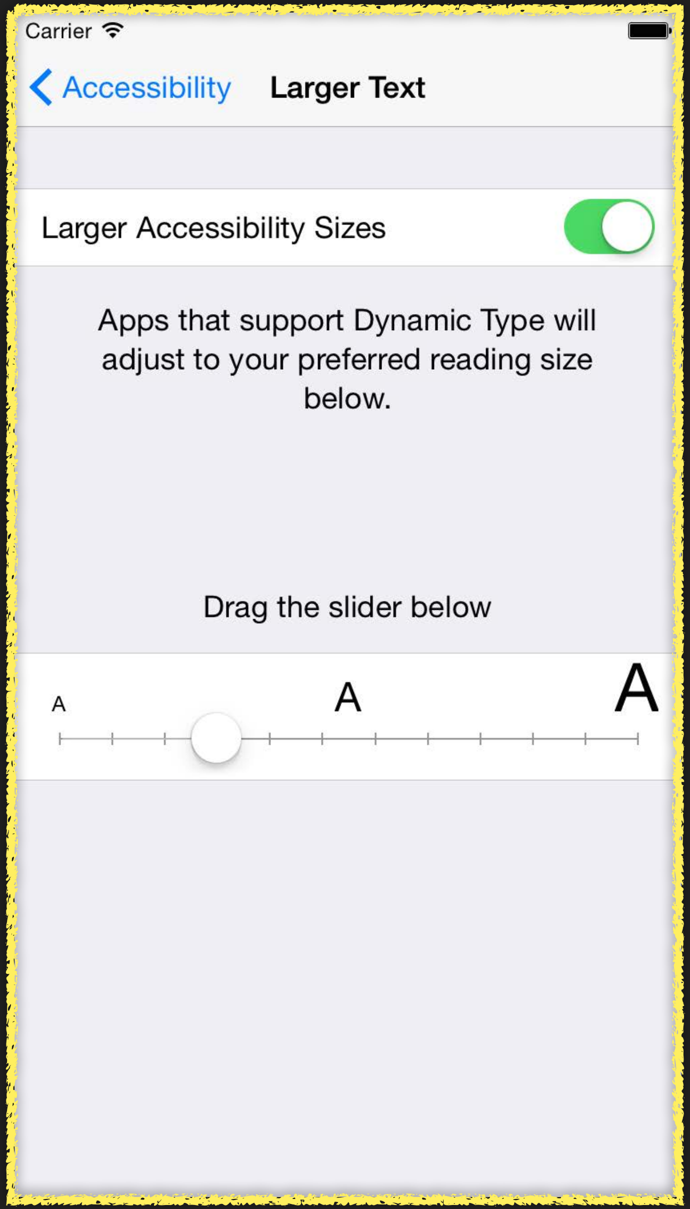
Reduce Motion Off >

On/Off Labels ○

Subtitles & Captioning >

UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …



Carrier

Accessibility Larger Text

Larger Accessibility Sizes

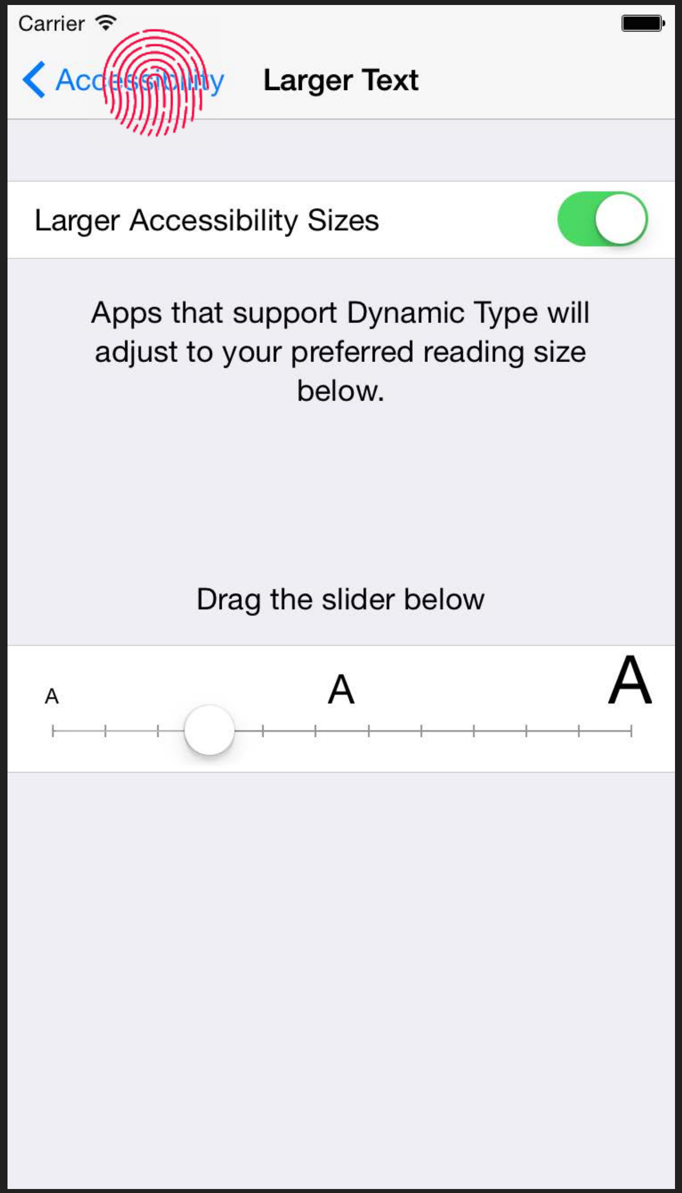
Apps that support Dynamic Type will adjust to your preferred reading size below.

Drag the slider below

A "Larger Text" MVC

UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …



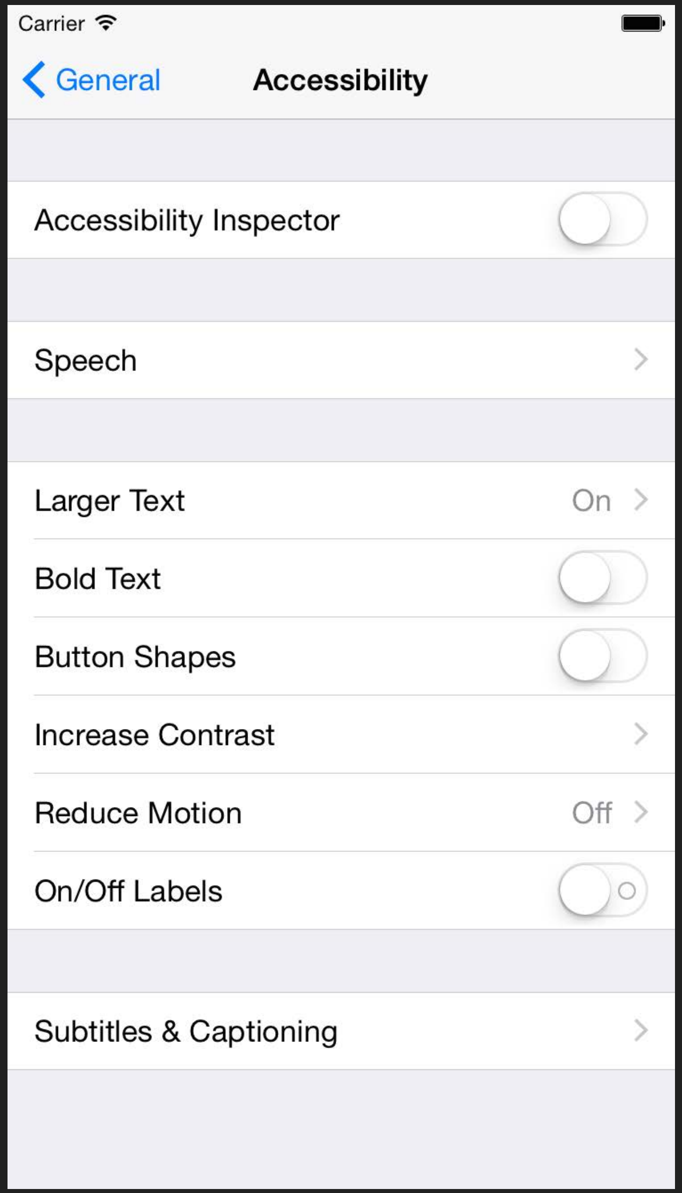
Larger Accessibility Sizes

Apps that support Dynamic Type will adjust to your preferred reading size below.

Drag the slider below

UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …



Accessibility Inspector

Speech

Larger Text On

Bold Text

Button Shapes

Increase Contrast

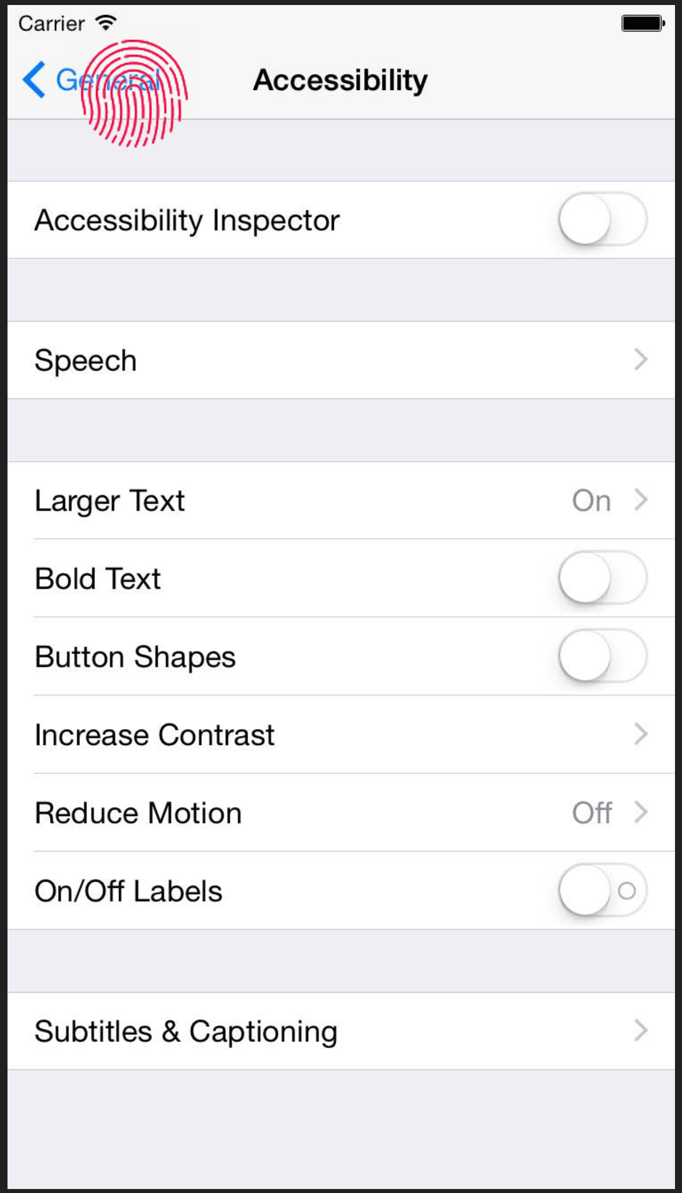
Reduce Motion Off

On/Off Labels

Subtitles & Captioning

UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …



Carrier

< General

Accessibility

Accessibility Inspector

Speech >

Larger Text On >

Bold Text

Button Shapes

Increase Contrast >

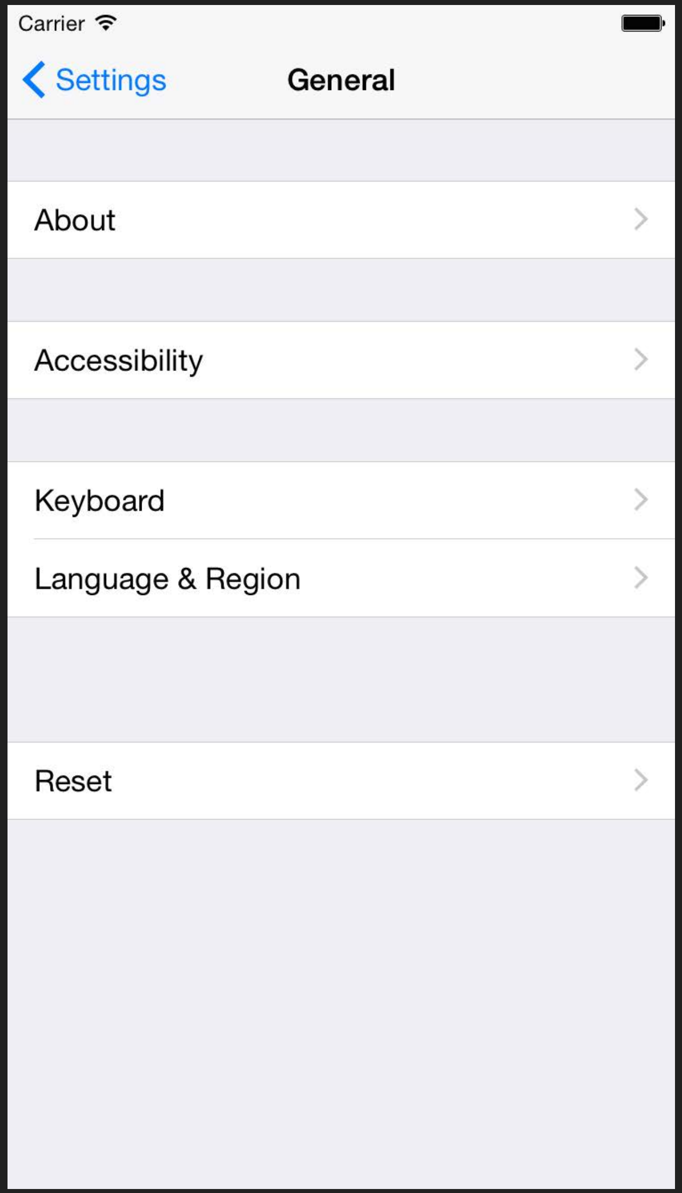
Reduce Motion Off >

On/Off Labels

Subtitles & Captioning >

UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …



Carrier

Settings General

About

Accessibility

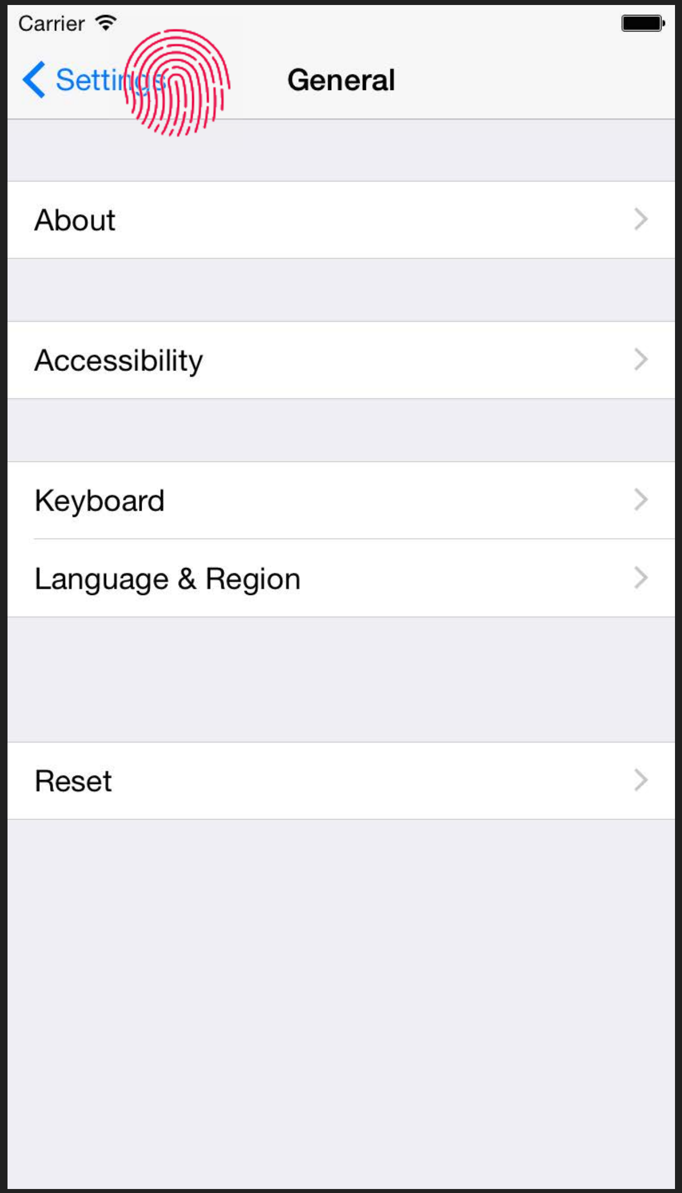
Keyboard

Language & Region

Reset

UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …



Carrier

Settings

General

About

Accessibility

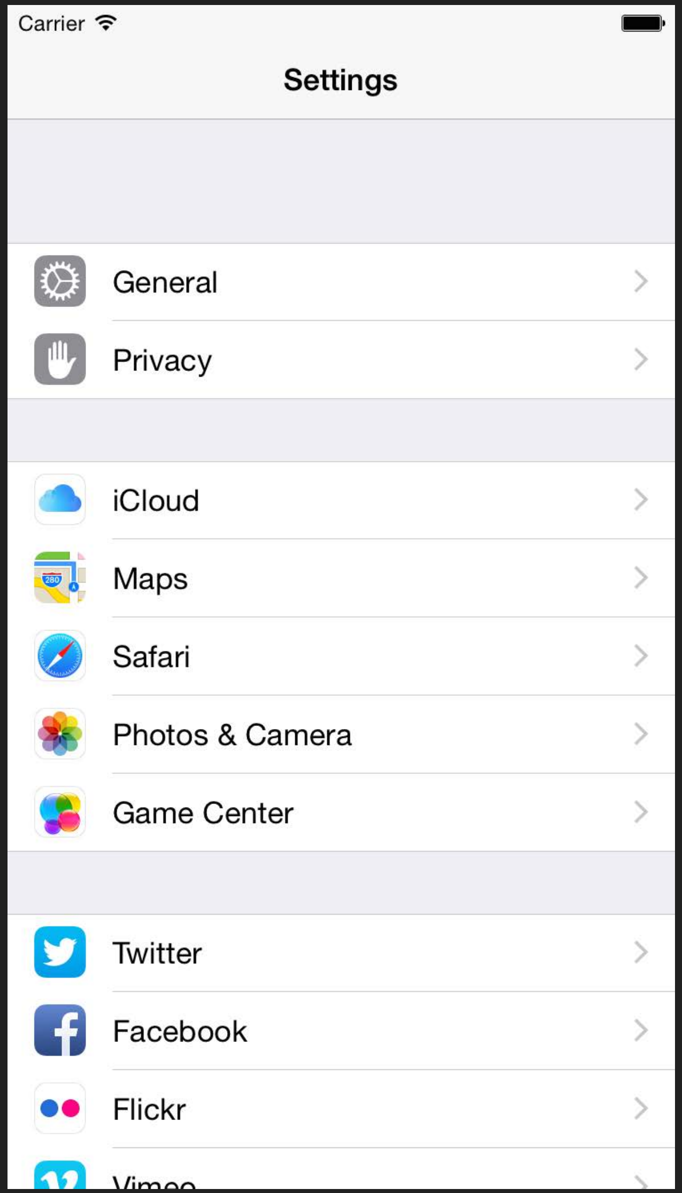
Keyboard

Language & Region

Reset

UINavigationController

- Pushes and pops MVCs off of a stack (like a stack of cards) …



Carrier

Settings

General

Privacy

iCloud

Maps

Safari

Photos & Camera

Game Center

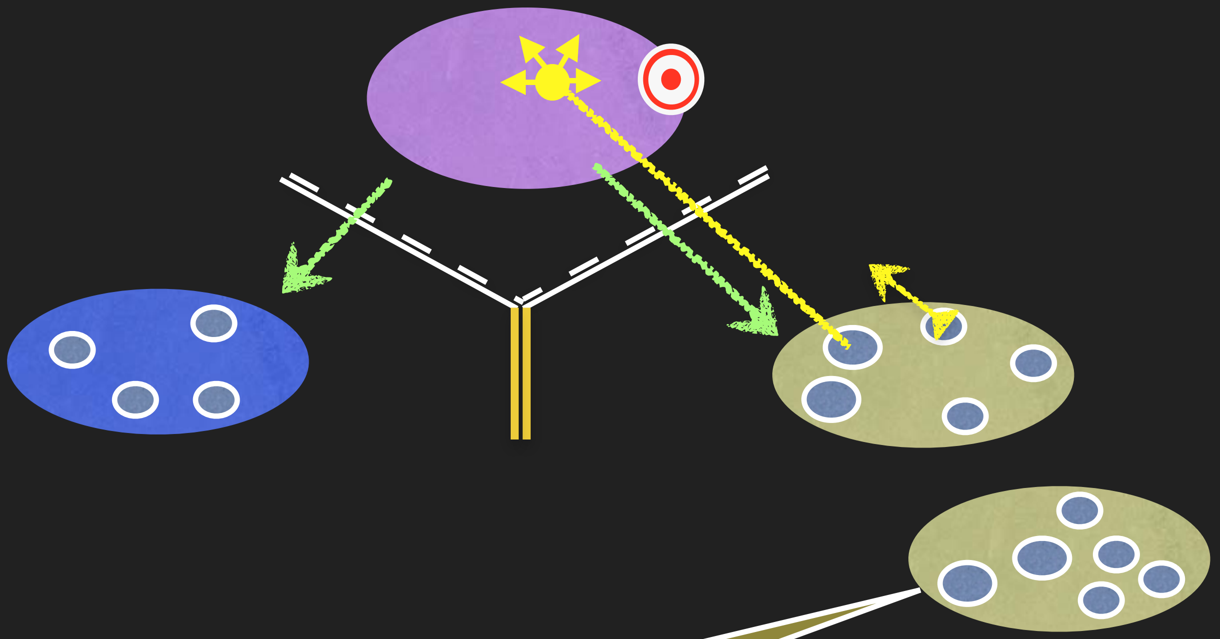
Twitter

Facebook

Flickr

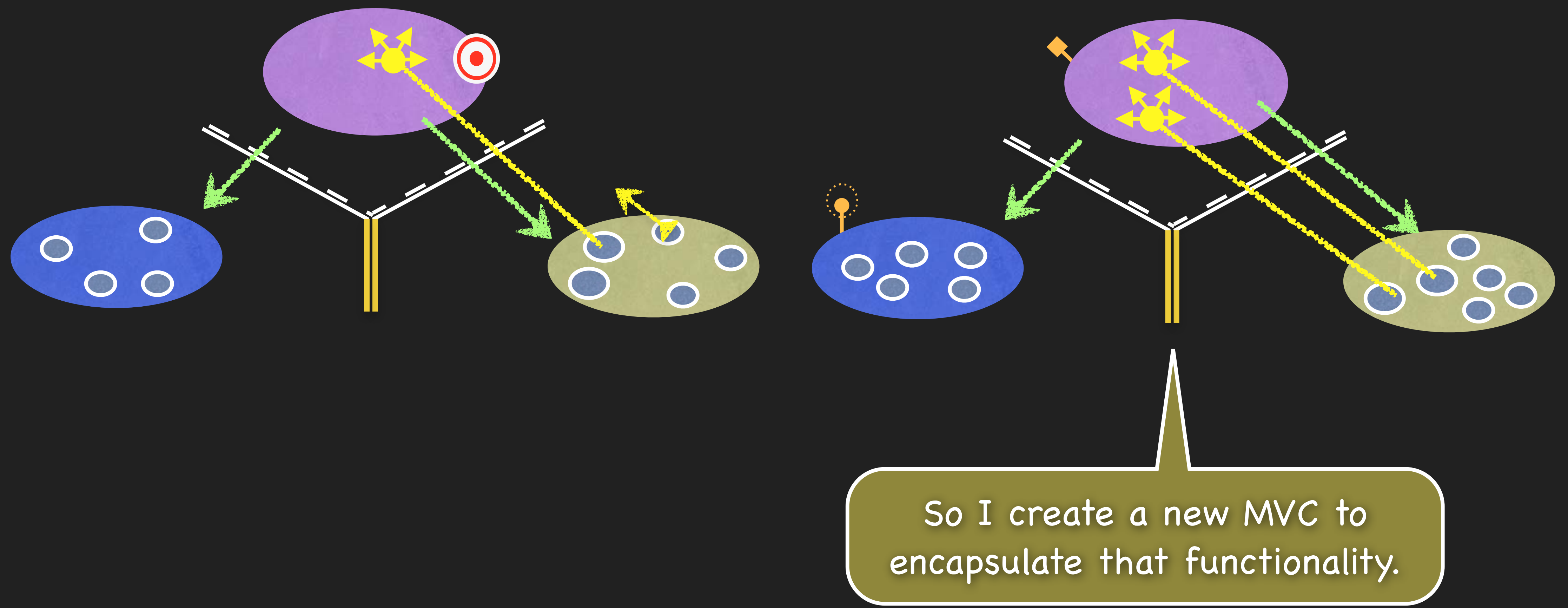
Vimeo

UINavigationController



I want more features, but it doesn't make sense to put them all in one MVC!

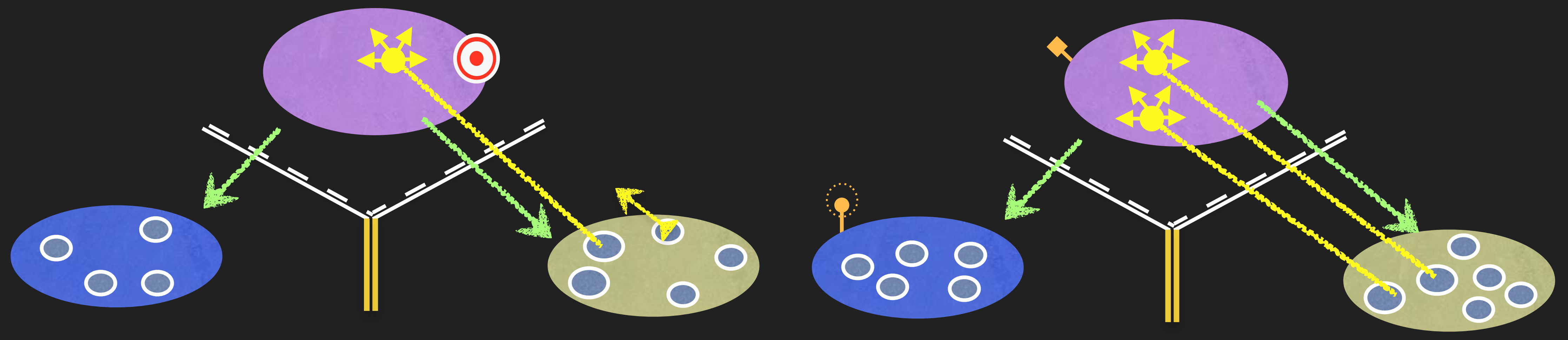
UINavigationController



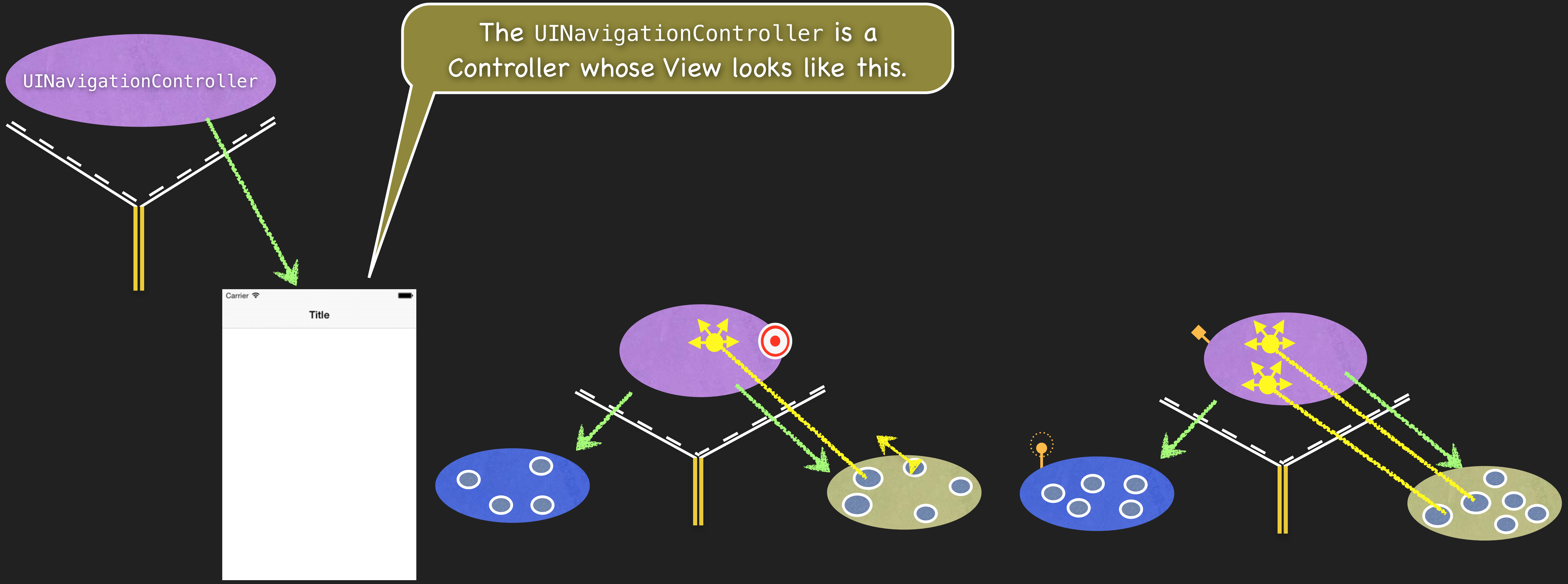
So I create a new MVC to encapsulate that functionality.

UINavigationController

We can use a UINavigationController to let them share the screen.

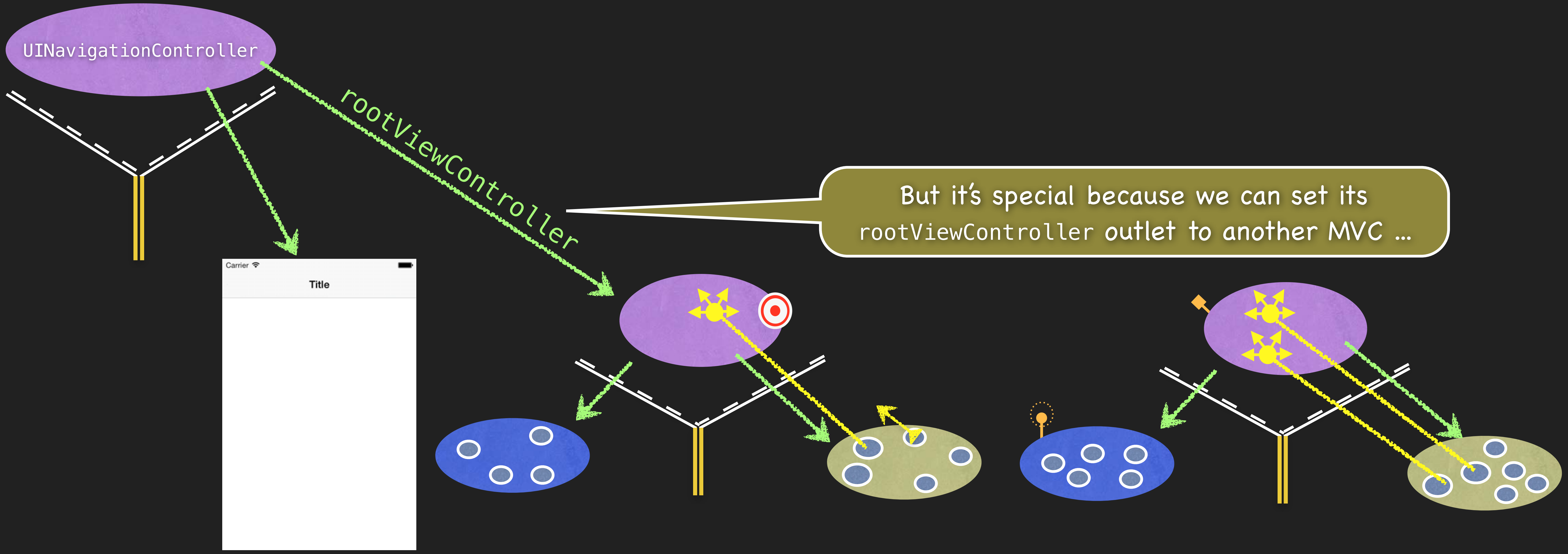


UINavigationController



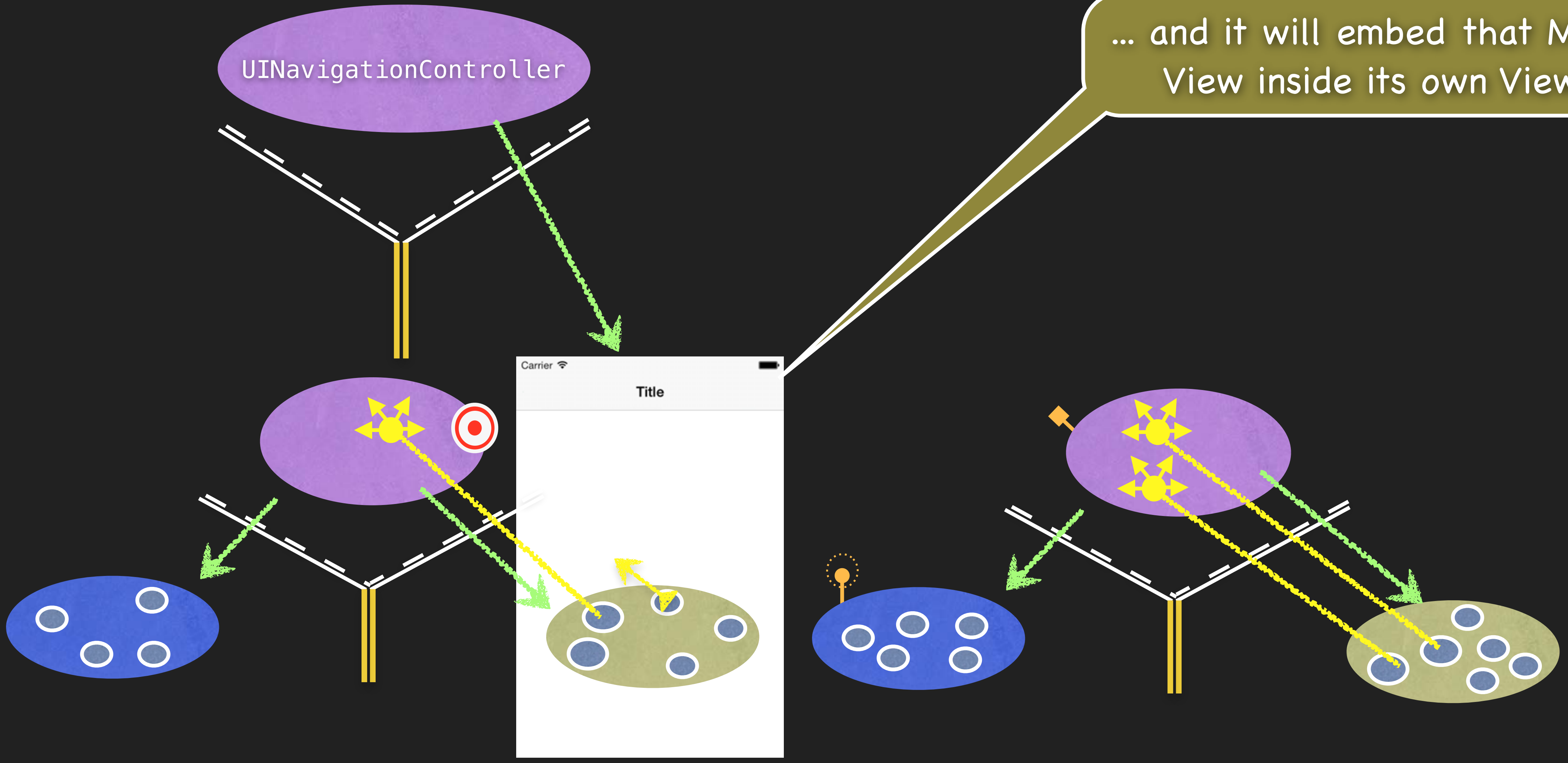
The UINavigationController is a Controller whose View looks like this.

UINavigationController



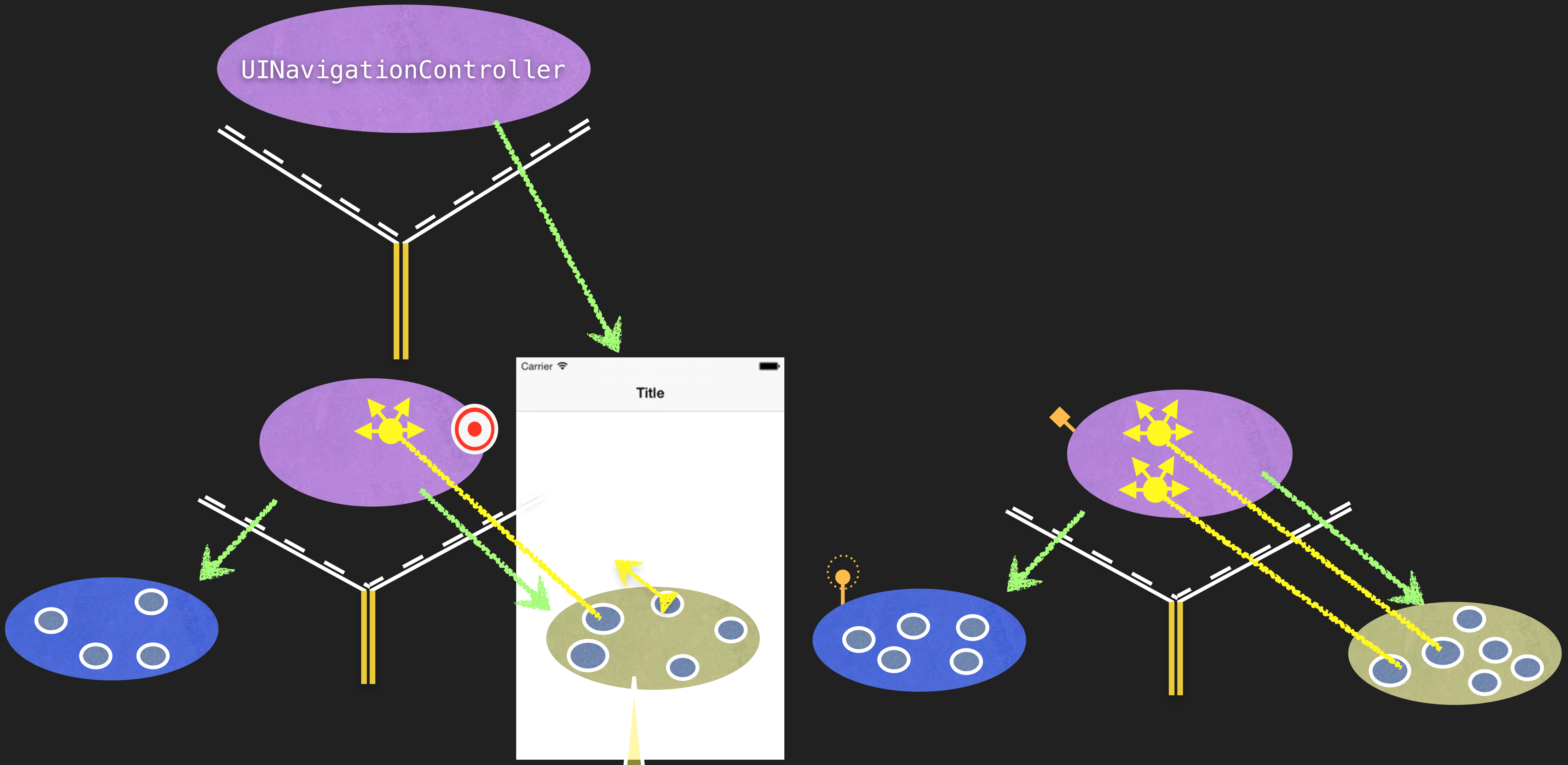
But it’s special because we can set its `rootViewController` outlet to another MVC …

UINavigationController



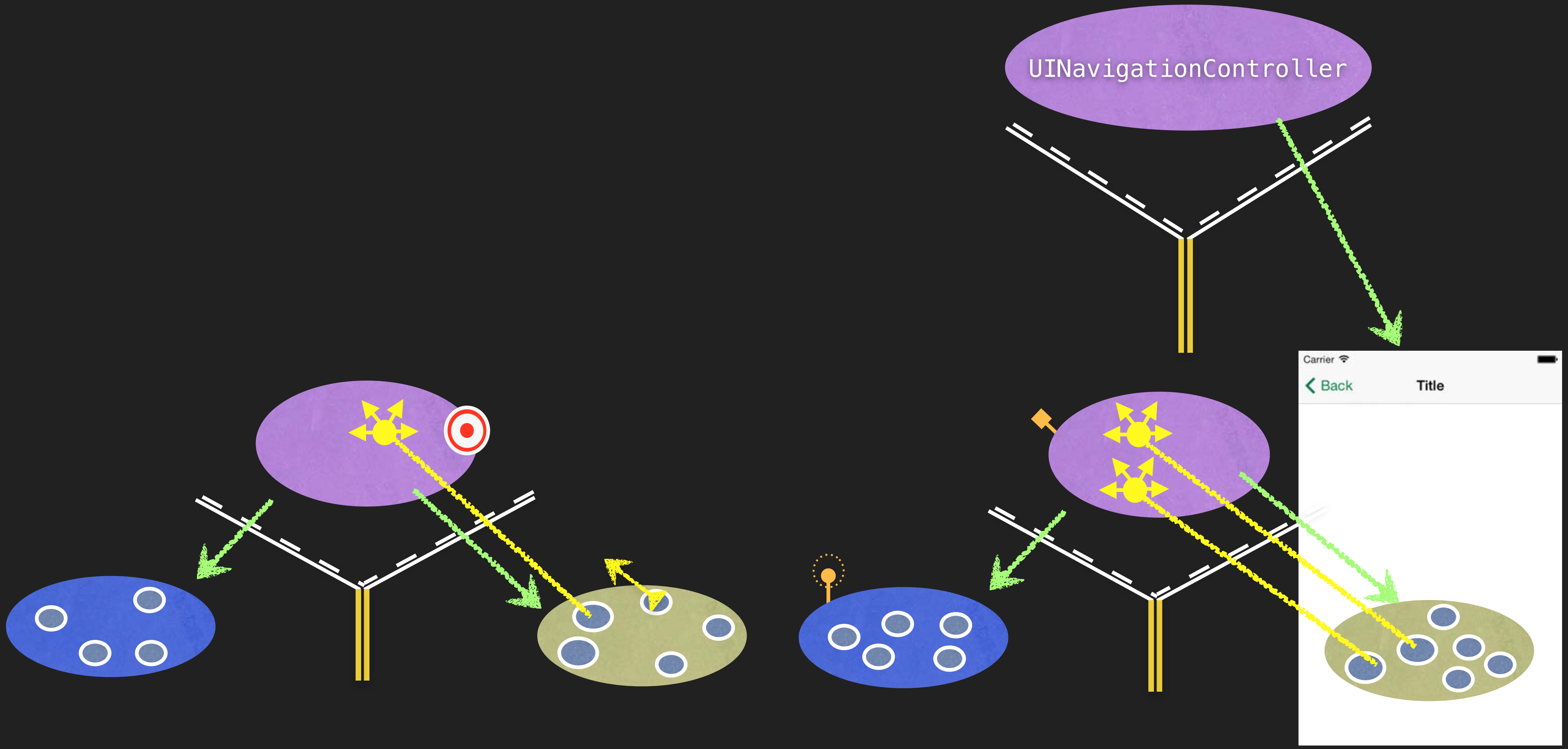
... and it will embed that MVC's View inside its own View.

UINavigationController



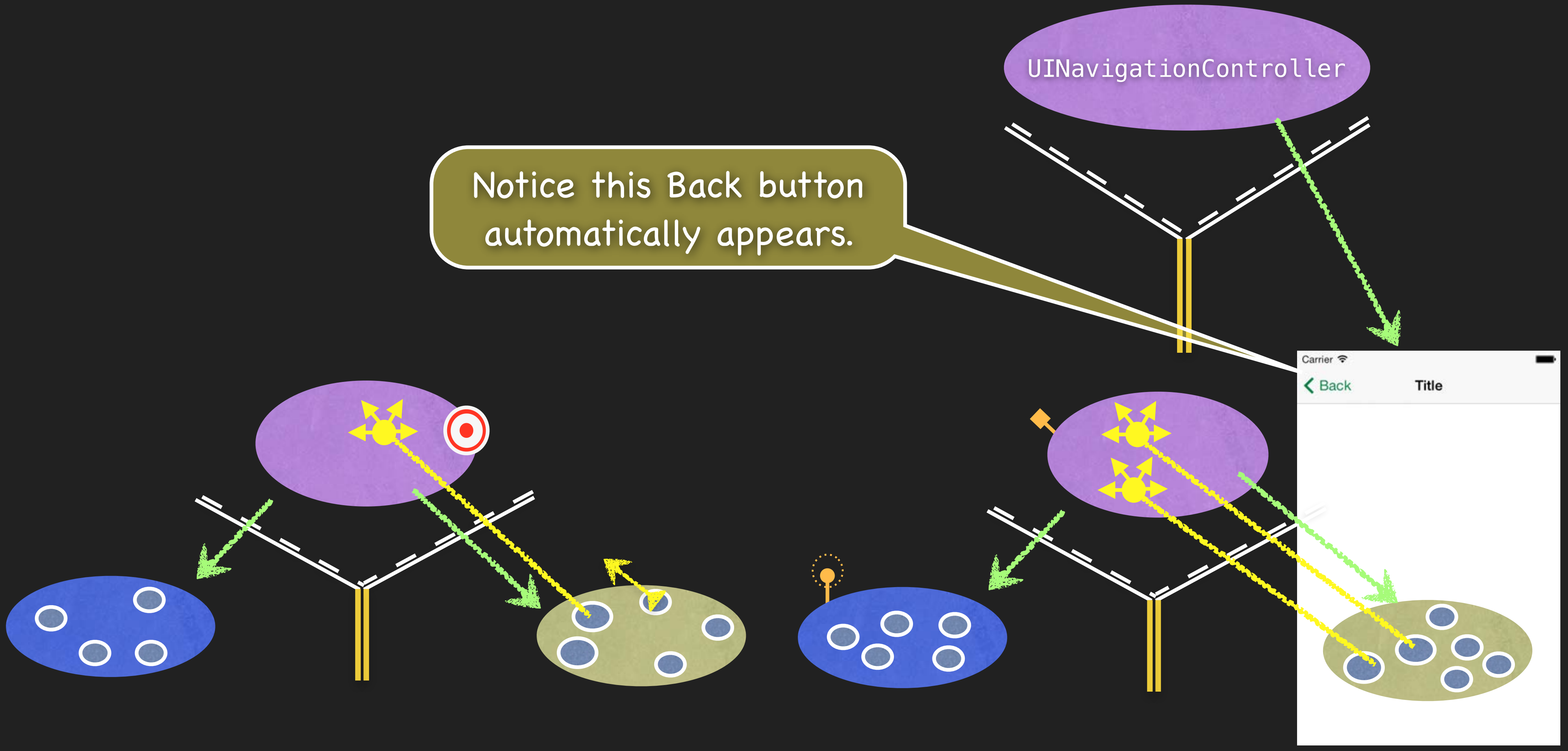
Then a UI element in this View (e.g. a UIButton) can segue to the other MVC and its View will now appear in the UINavigationController instead.

UINavigationController



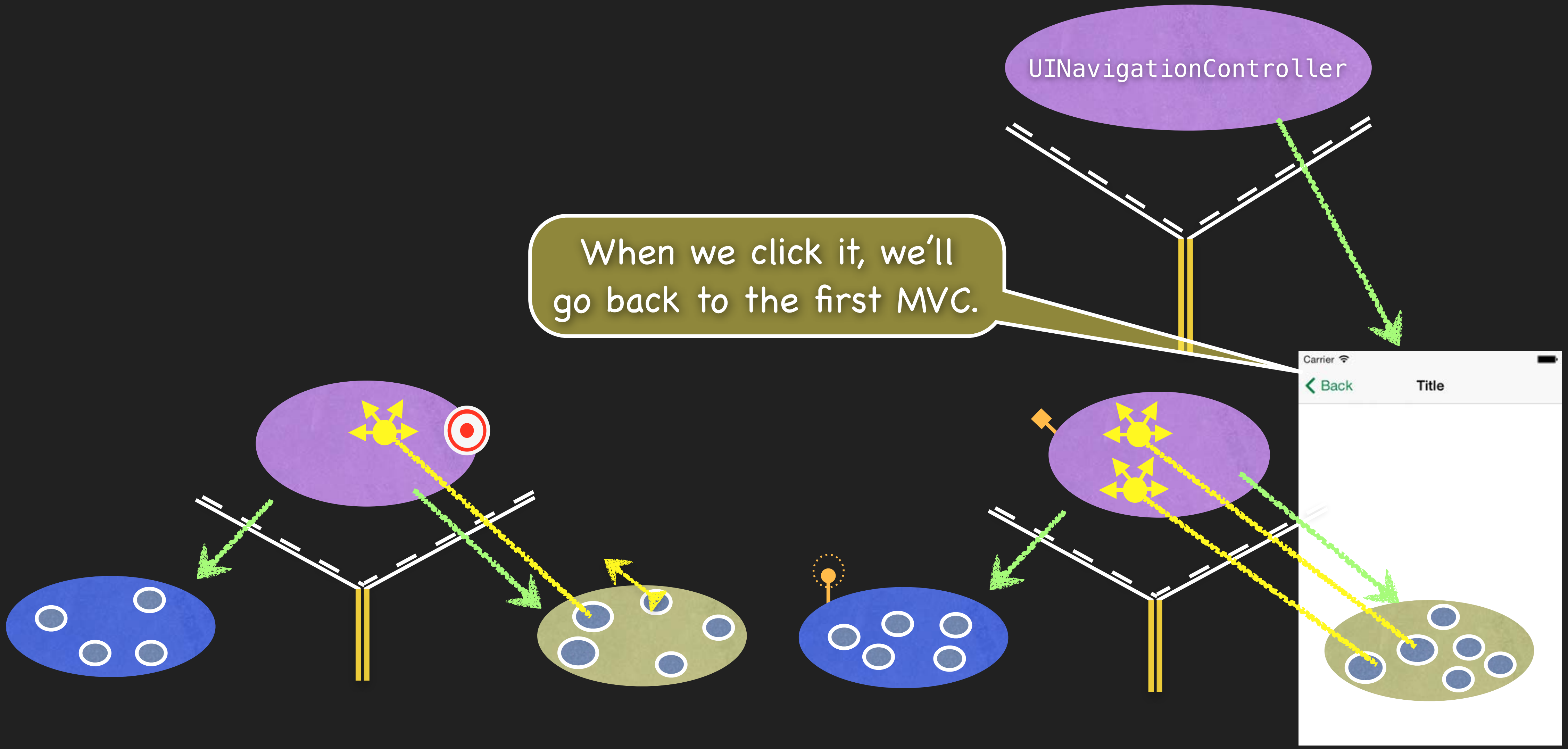
We call this kind of segue a "Show (push) segue".

UINavigationController



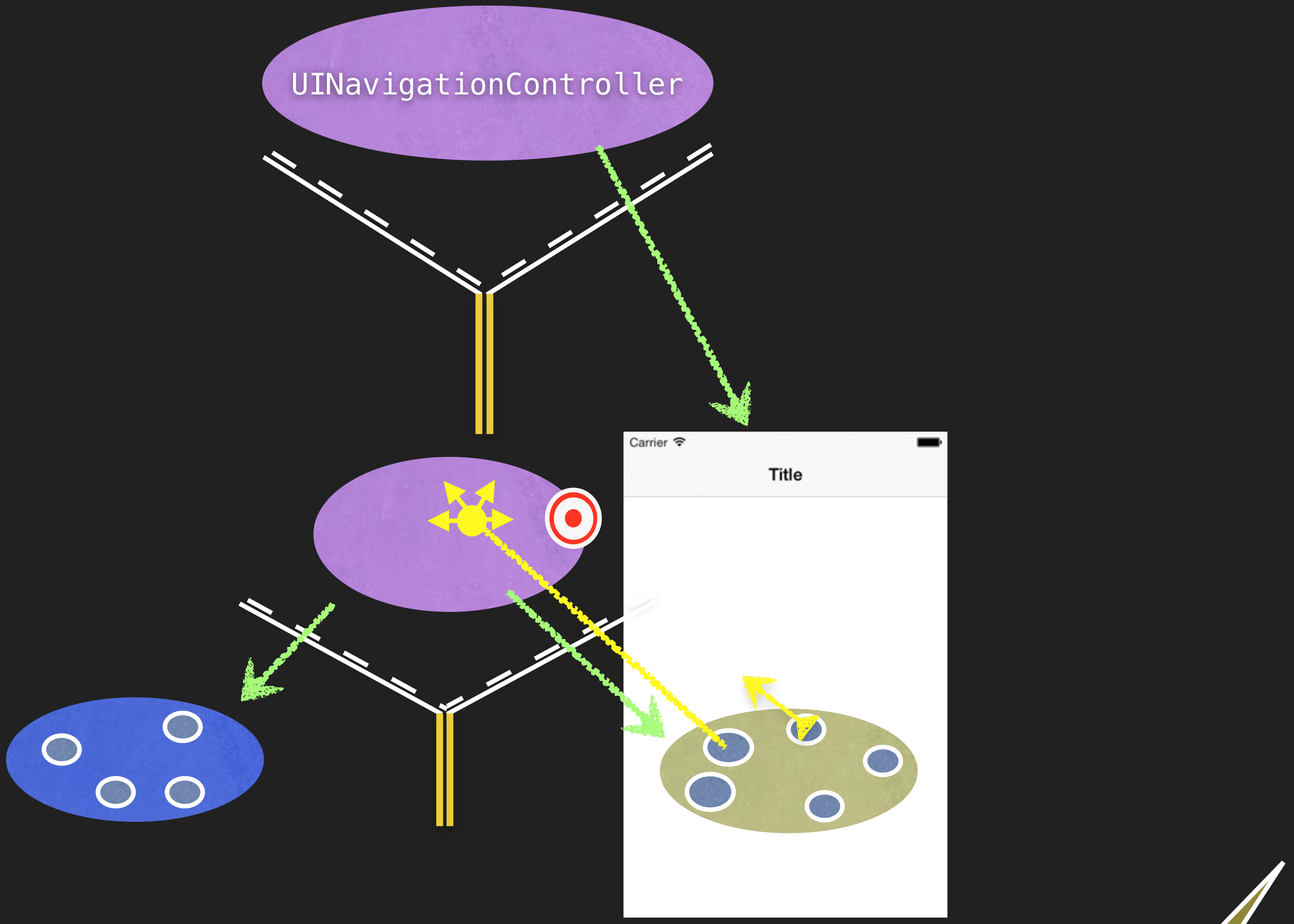
Notice this Back button automatically appears.

UINavigationController



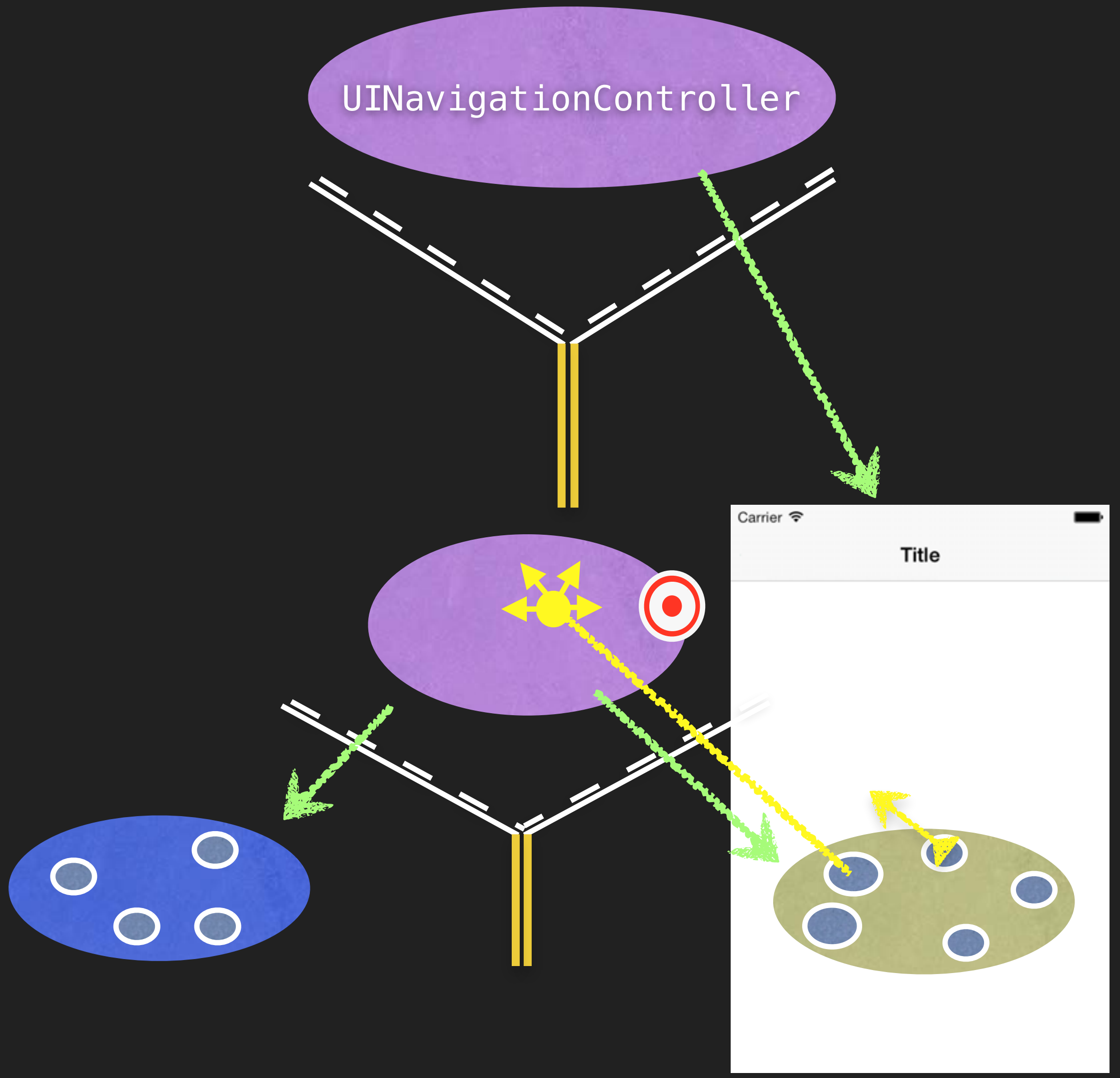
When we click it, we'll go back to the first MVC.

UINavigationController



Notice that after we back out of an MVC,
it disappears (it is deallocated from the heap, in fact).

UINavigationController



Accessing the sub-MVCs

- You can get the sub-MVCs via the `viewControllers` property

```
var viewControllers: [UIViewController]? { get set } // can be optional (e.g. for tab bar)
// for a tab bar, they are in order, left to right, in the array
// for a navigation controller, [0] is the root and the rest are in order on the stack
// even though this is settable, usually setting happens via storyboard, segues, or other
// for example, navigation controller's push and pop methods
```

- But how do you get ahold of the TBC or NC itself?

Every `UIViewController` knows the Split View, Tab Bar or Navigation Controller it is currently in

These are `UIViewController` properties …

```
var tabBarController: UITabBarController? { get }
var navigationController: UINavigationController? { get }
```

Pushing/Popping

- Adding (or removing) MVCs from a `UINavigationController`

```
func pushViewController(_ vc: UIViewController, animated: Bool)
func popViewController(animated: Bool)
```

But we usually don't do this. Instead we use Segues. More on this in a moment.