

Developing Applications for iOS



Lecture 6: View Controllers and Storyboarding

Prof. Dr. Radu Ionescu
raducu.ionescu@gmail.com
Faculty of Mathematics and Computer Science
University of Bucharest

Content

- MVCs Working Together
- Segues
- Navigation Controllers
- View Controllers
- Tab Bar Controller

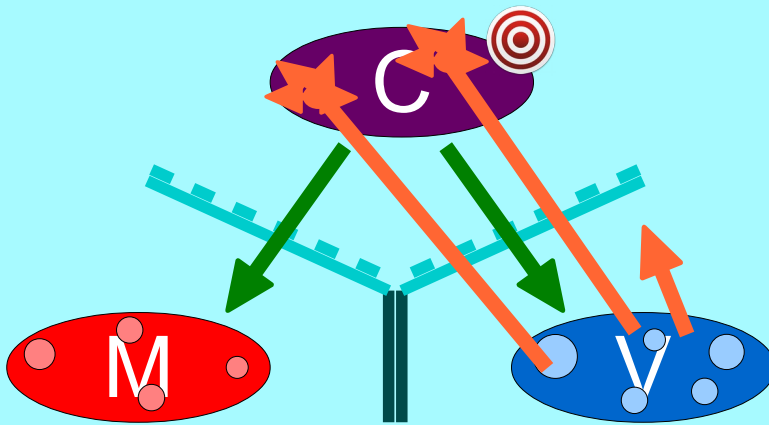
View Controller

- Your Controller in an MVC grouping is always a subclass of `UIViewController`.
- It manages a View (made up of subviews that you usually have some outlets/actions to/from).
- It is the link between that View and the Model (which is UI-independent).

So how do we grow our application to use multiple MVCs?

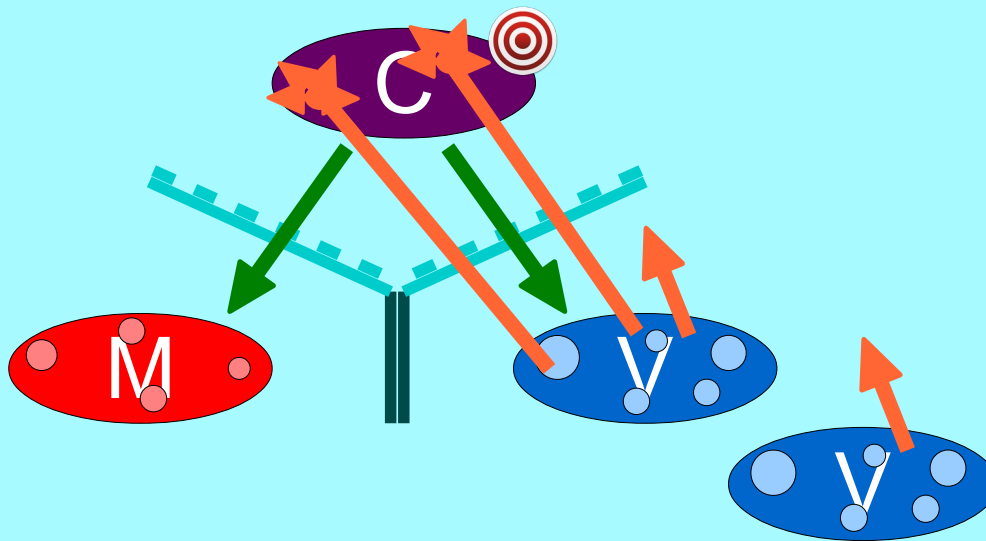
- We need infrastructure to manage them all.
- That's what storyboards and "controllers of controllers" are all about.

MVCs Working Together



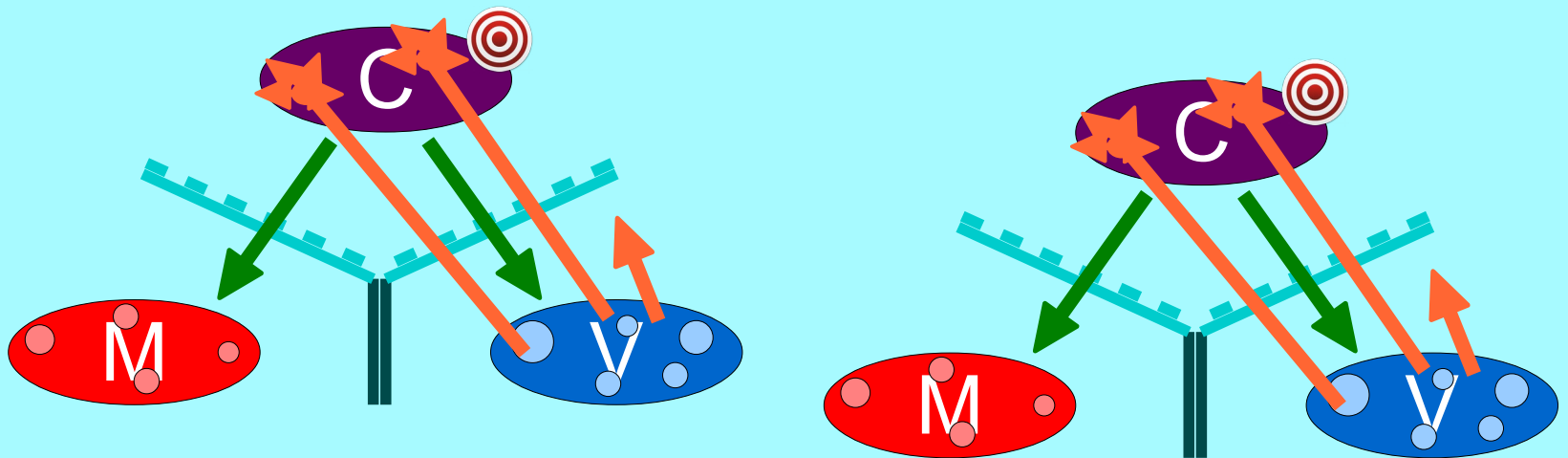
- What happens when your application gets more features?

MVCs Working Together



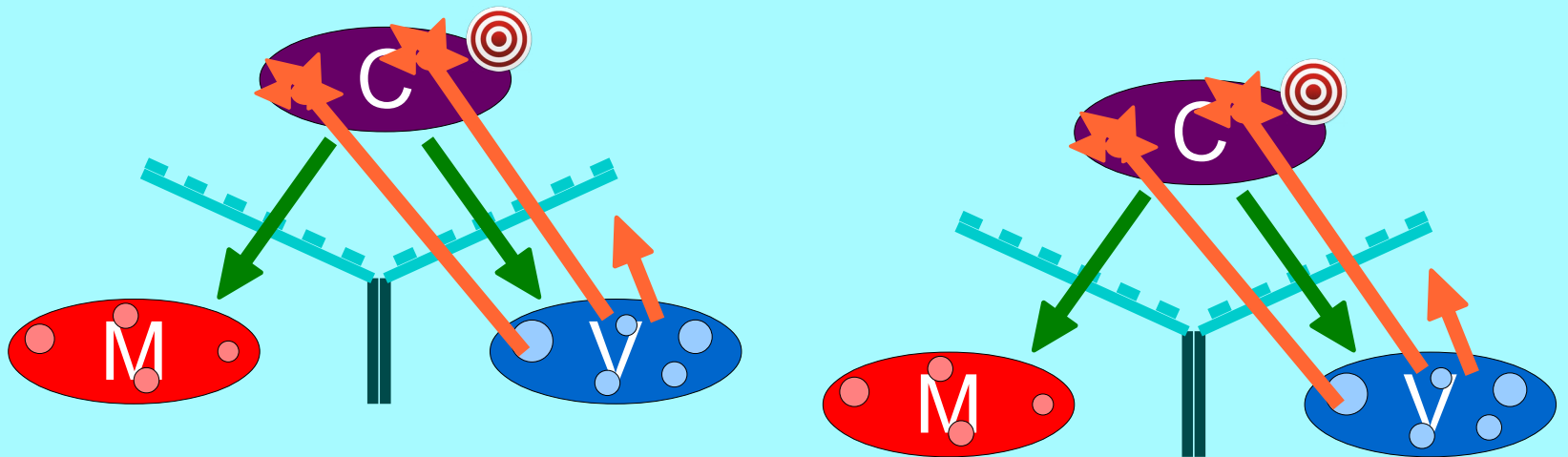
- What happens when your application gets more features?
- We have to add more Views. Now all of your UI can't fit in one MVC's view.

MVCs Working Together



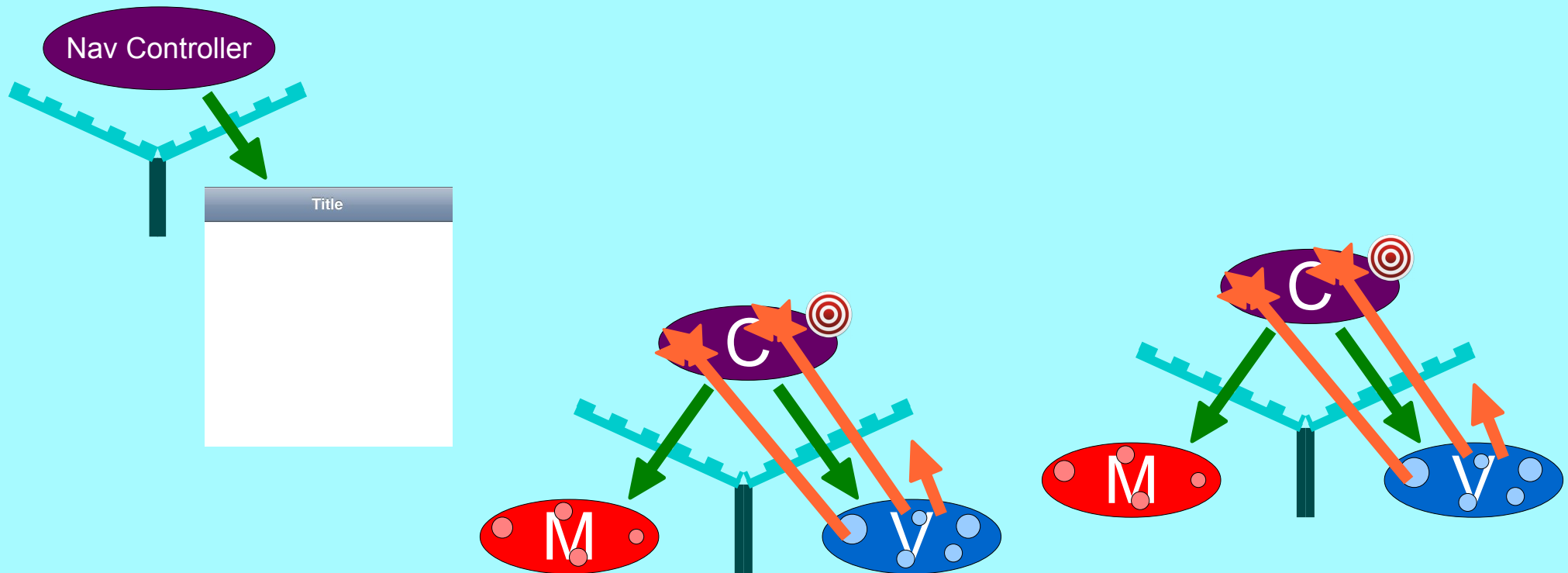
- What happens when your application gets more features?
- We never have an MVC's view span across screens. So we'll have to create a new MVC for these new features.

MVCs Working Together



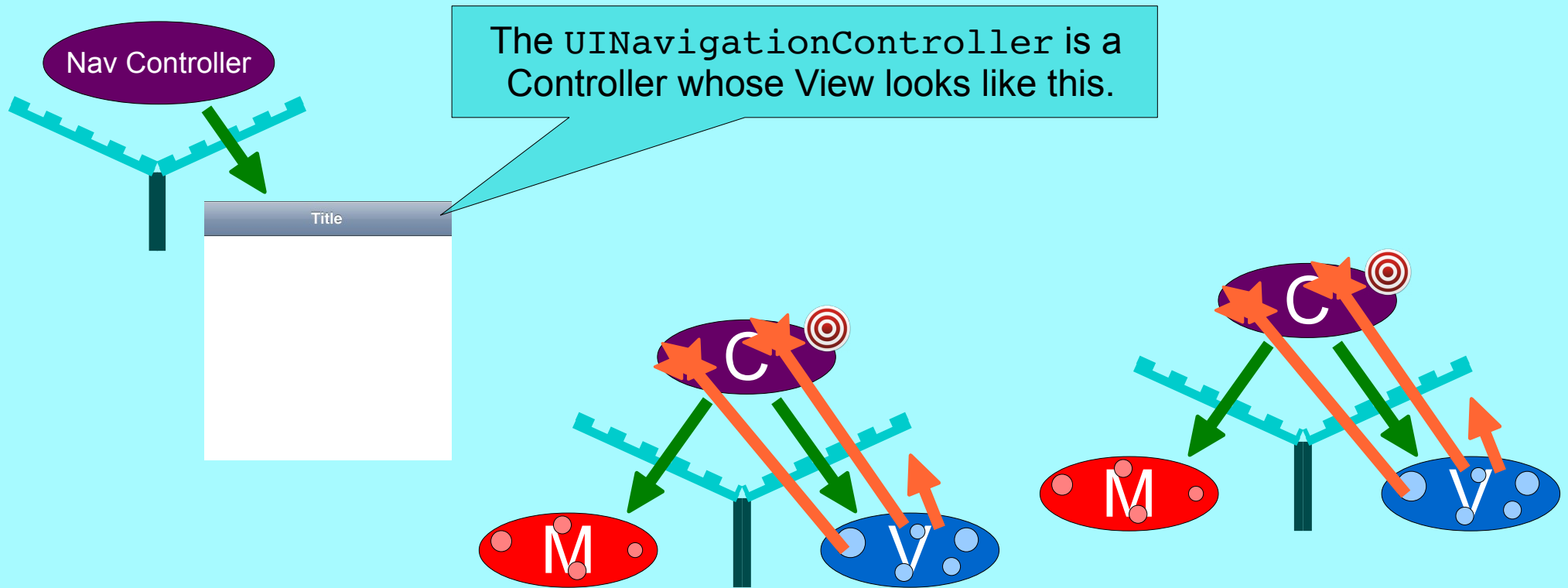
- So how do we switch the screen to show this other MVC?

MVCs Working Together

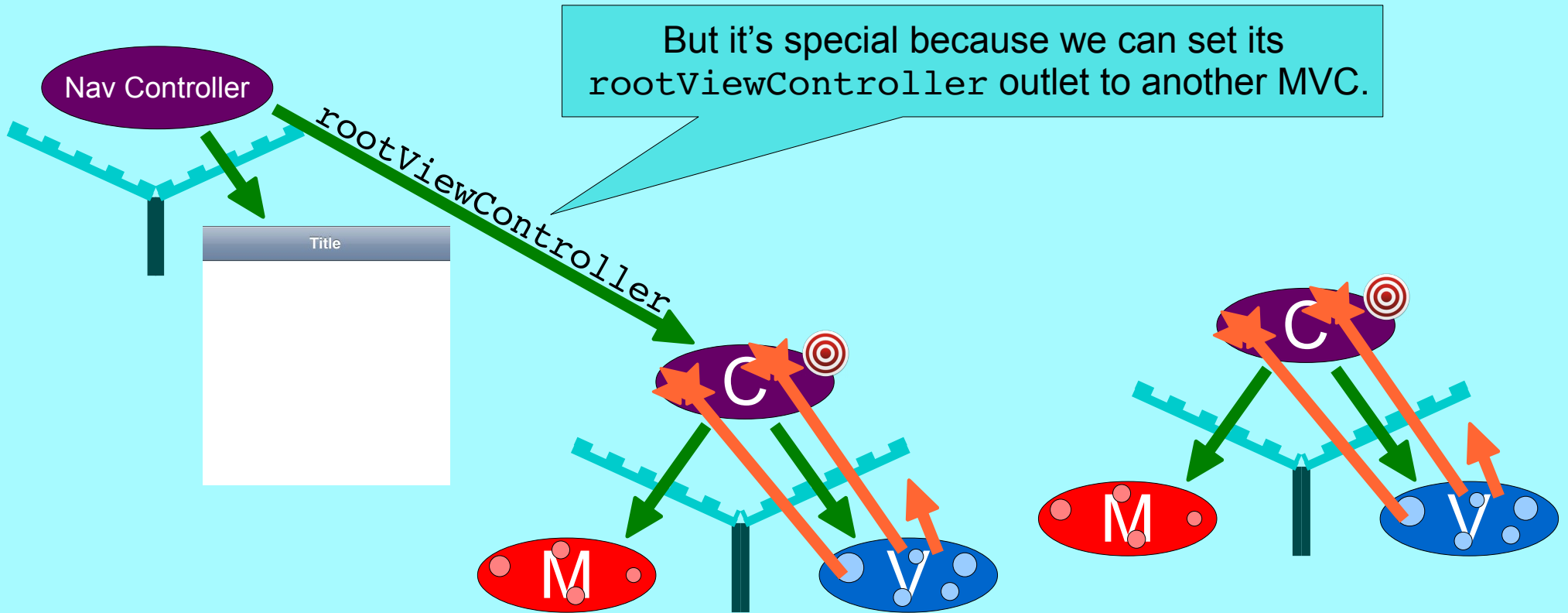


- So how do we switch the screen to show this other MVC?
- We use a “controller of controllers” to do that. For example, a UINavigationController.

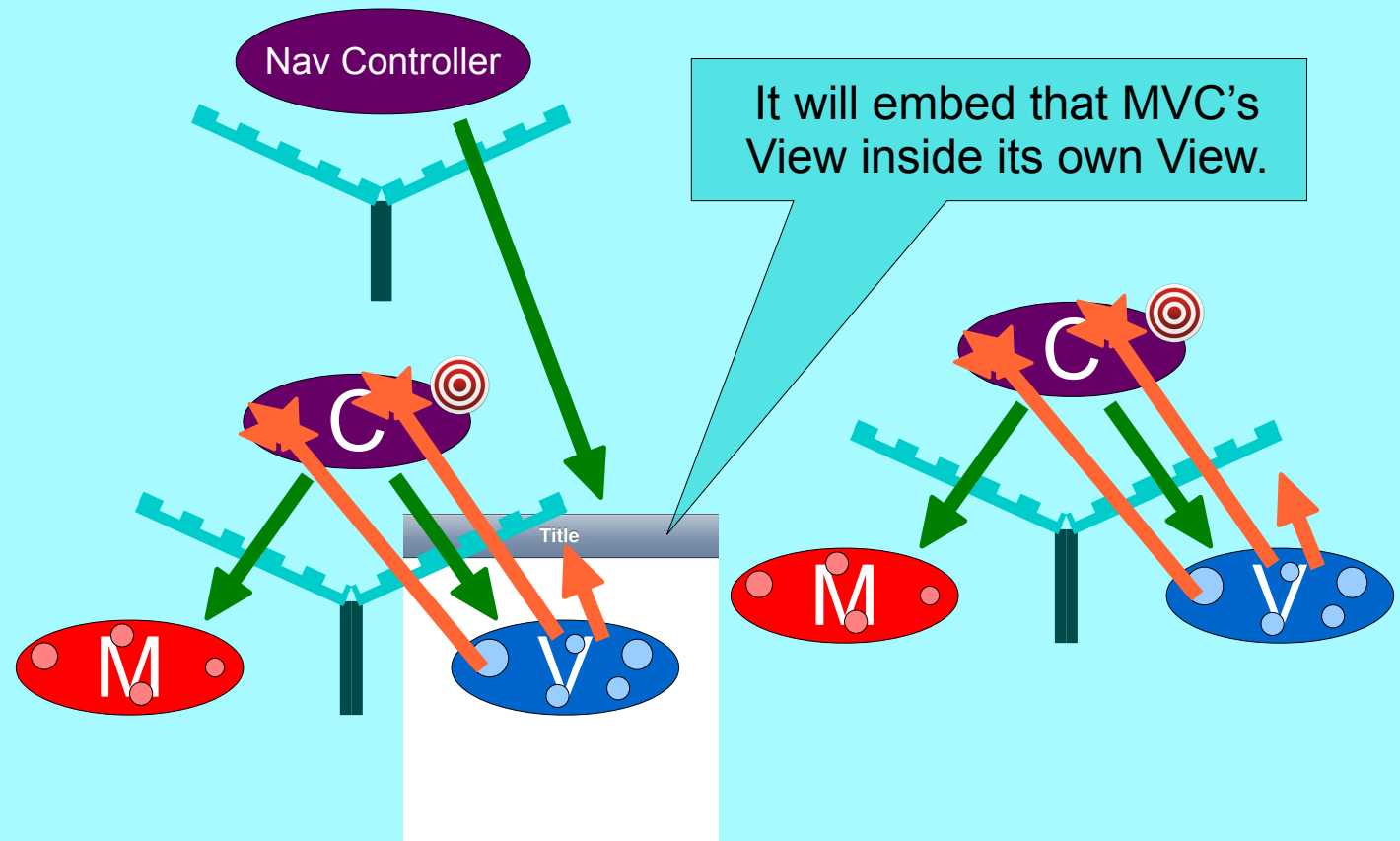
MVCs Working Together



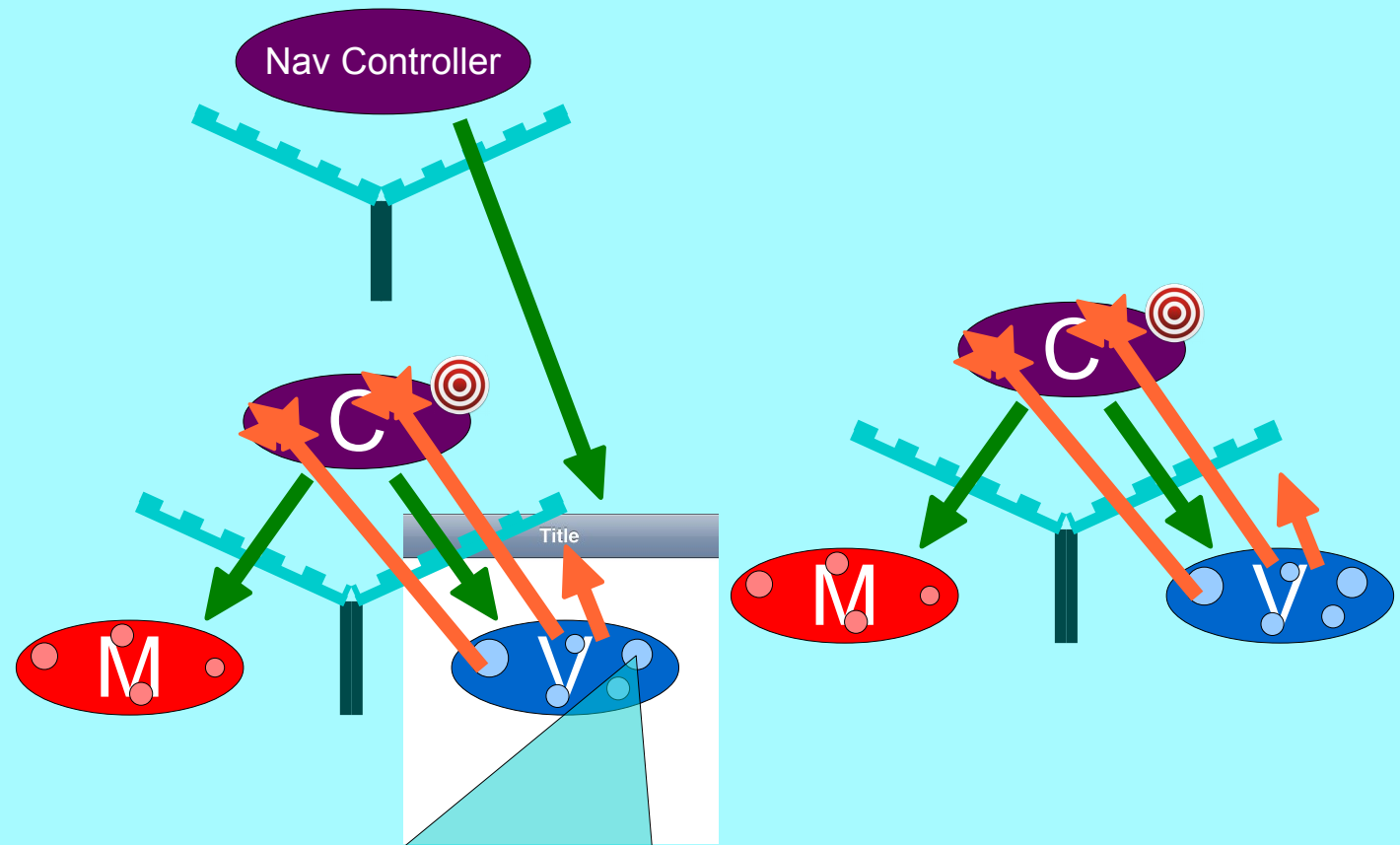
MVCs Working Together



MVCs Working Together

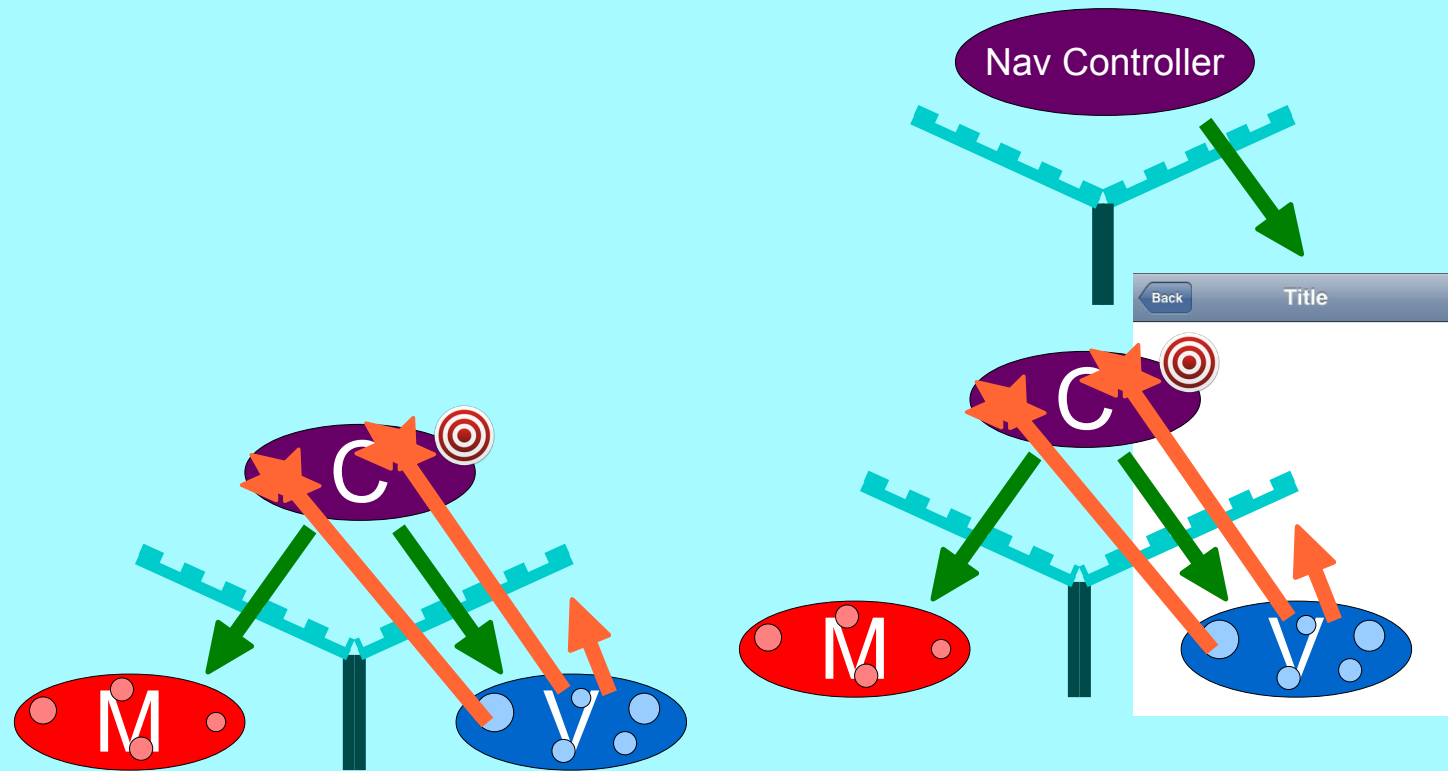


MVCs Working Together

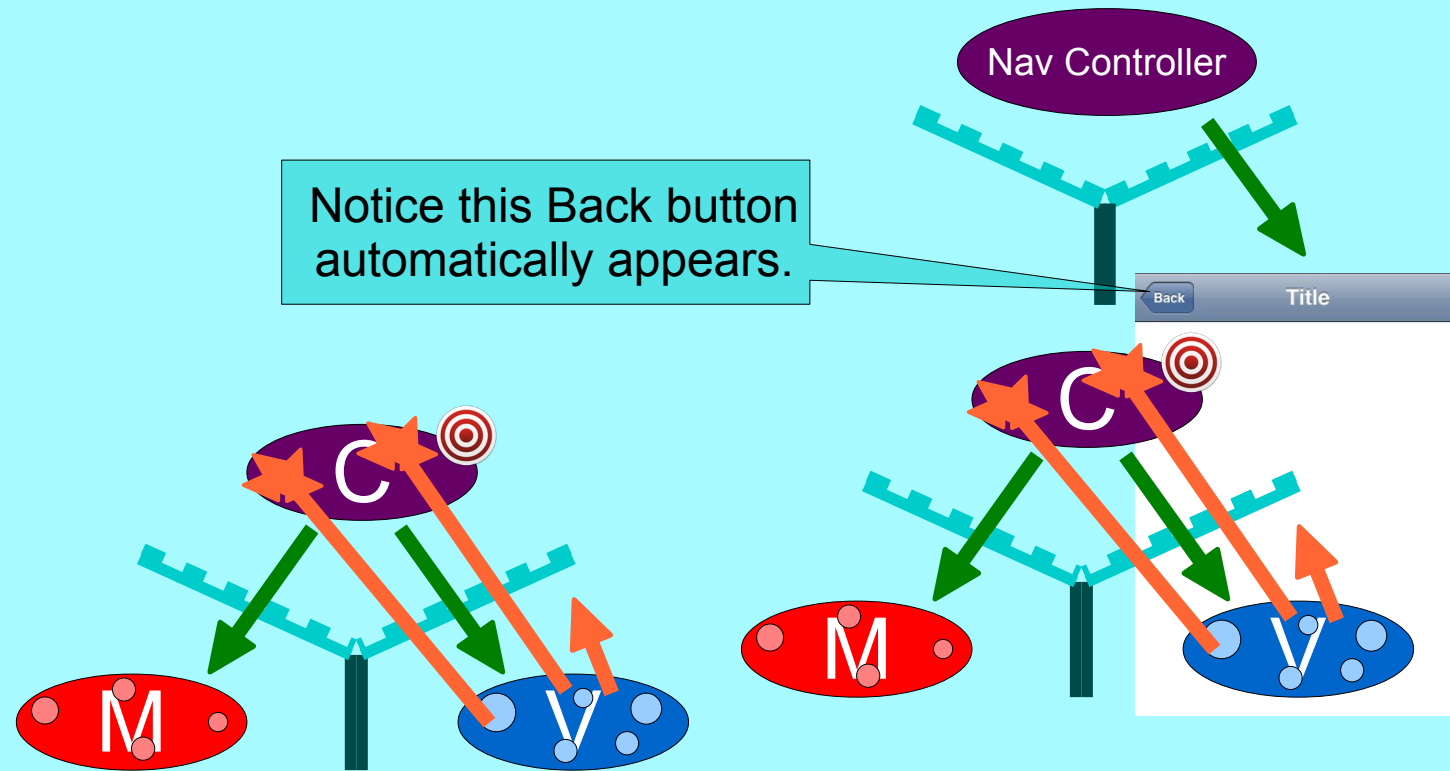


Then a UI element in this View (e.g. a UIButton) can **segue** to the other MVC and its View will now appear in the UINavigationController.

MVCs Working Together



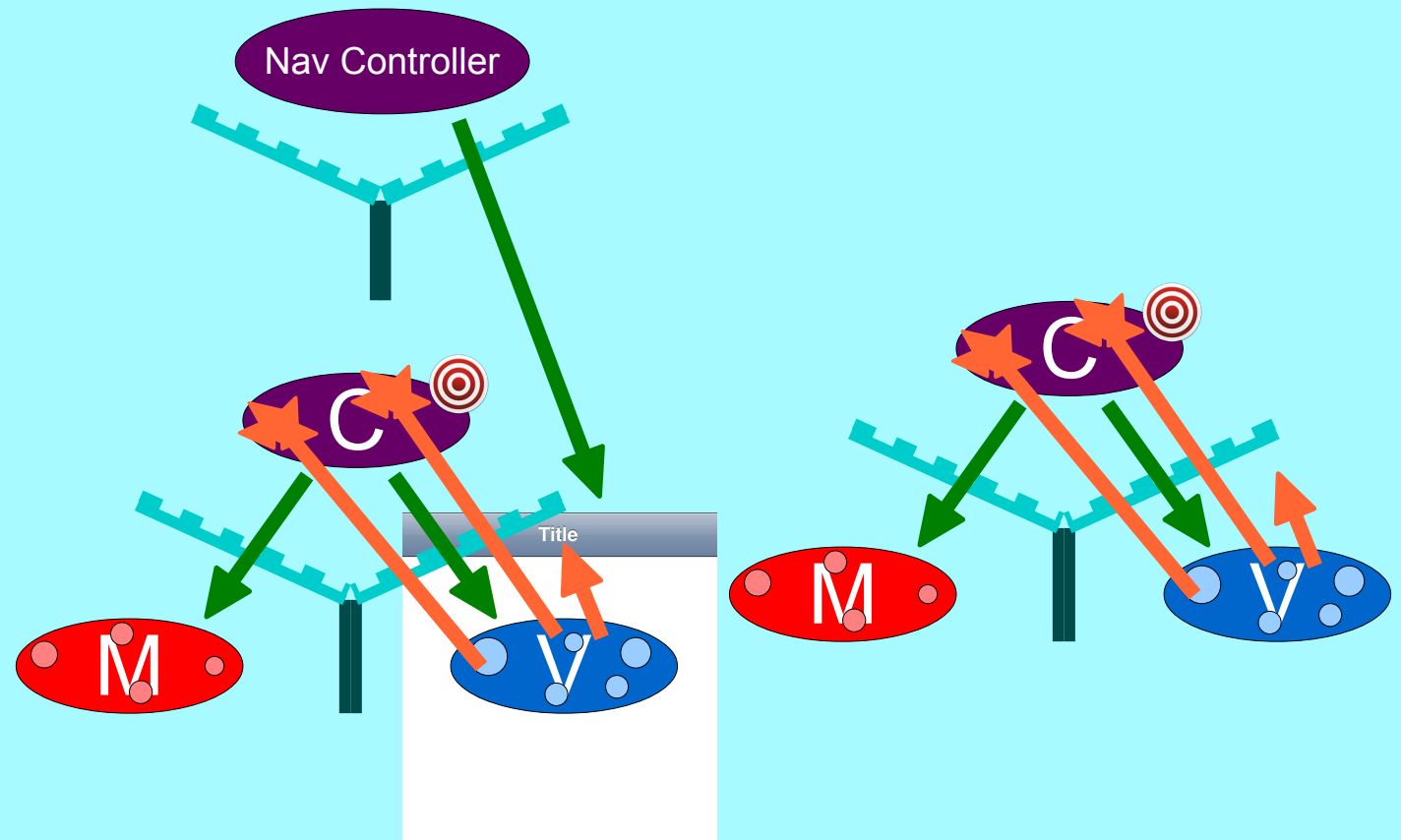
MVCs Working Together



MVCs Working Together



MVCs Working Together



Segues

Demo

- Let's talk about how the segue gets set up first.
- Then we'll look at how we create a `UINavigationController` in our storyboard.

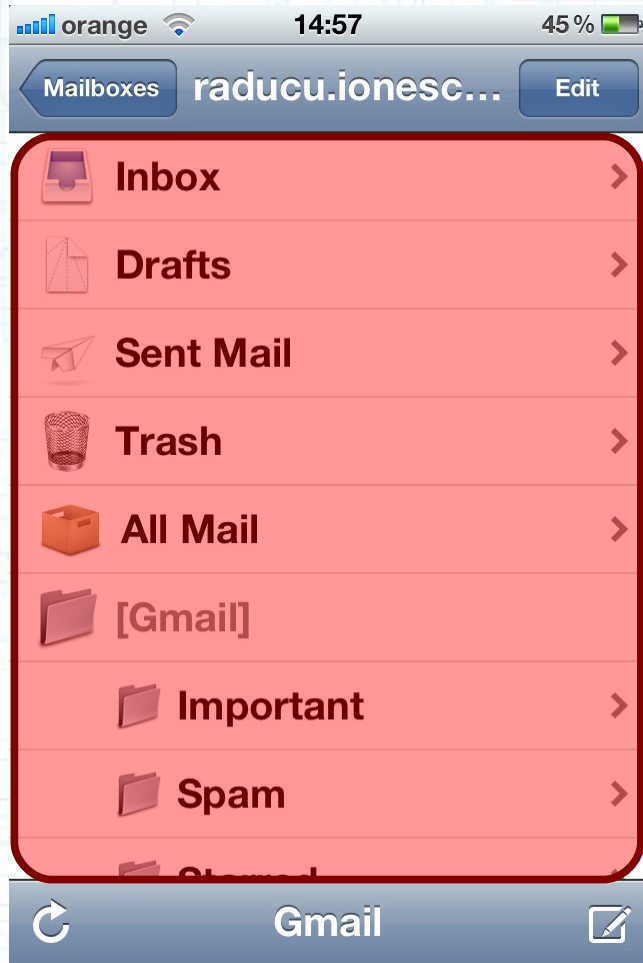
Segues

Demo

- You use the segue identifier in `prepare(for:sender:)` to figure out which segue is happening.
- Or you can use it to programmatically force a segue with `performSegue(withIdentifier:sender:)`.
- You can embed a View Controller in a `UINavigationController` from the Editor menu.

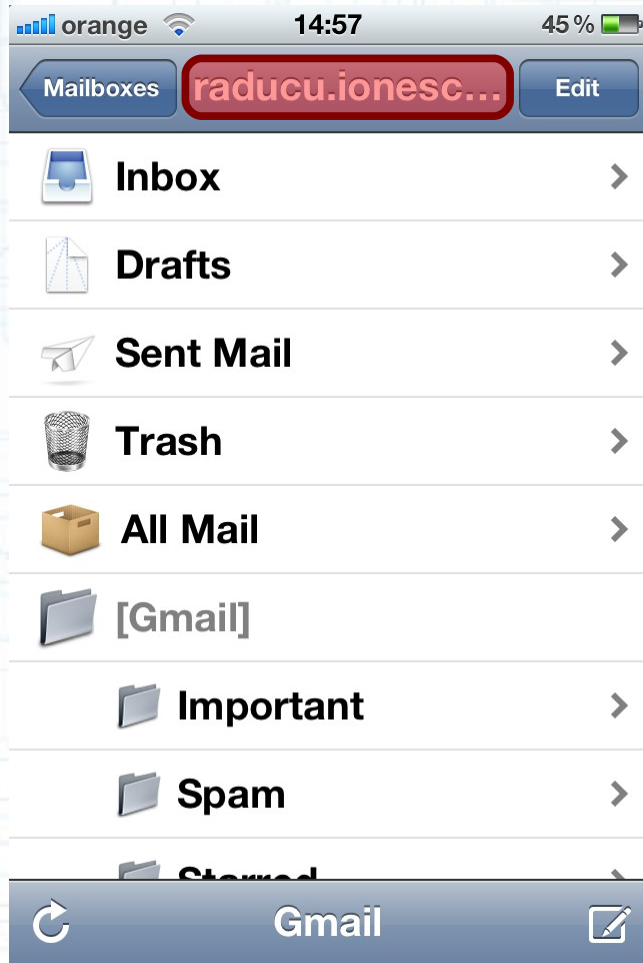
UINavigationController

- `UIView` obtained from the `view` property of the `UINavigationController` most recently pushed (or root).

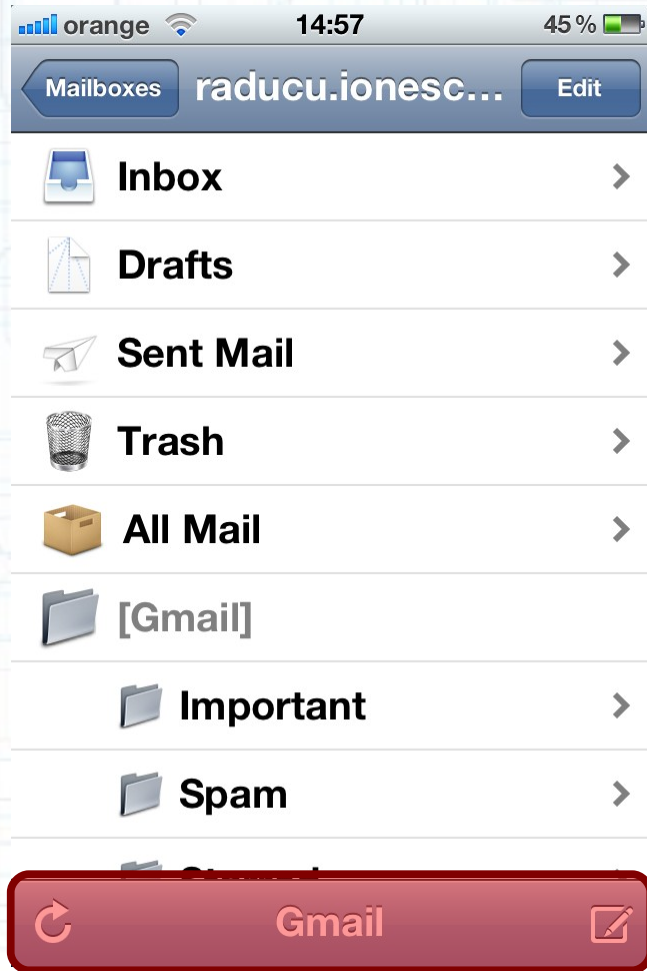


UINavigationController

- `UIView` obtained from the `view` property of the `UINavigationController` most recently pushed (or root).
- `String` obtained from the `title` property of the `UINavigationController` most recently pushed (or root).

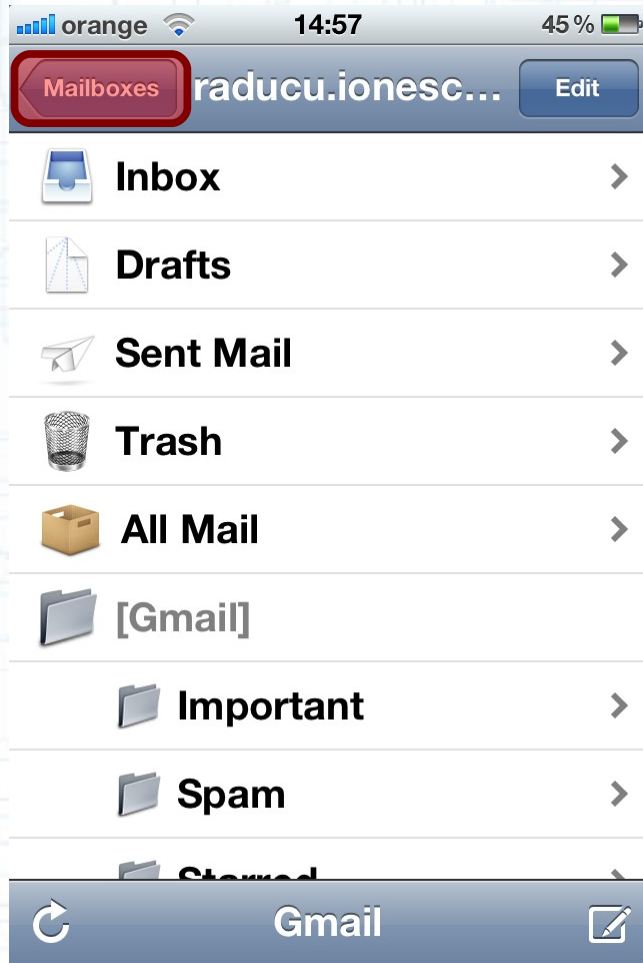


UINavigationController



- `UIView` obtained from the `view` property of the `UINavigationController` most recently pushed (or root).
- `String` obtained from the `title` property of the `UINavigationController` most recently pushed (or root).
- An `Array` of `UIBarButtonItem`s obtained from the `toolbarItems` property of the `UINavigationController` most recently pushed (or root).

UINavigationController



- `UIView` obtained from the `view` property of the `UINavigationController` most recently pushed (or root).
- `String` obtained from the `title` property of the `UINavigationController` most recently pushed (or root).
- An `Array` of `UIBarButtonItem`s obtained from the `toolbarItems` property of the `UINavigationController` most recently pushed (or root).
- A `UIBarButtonItem` item whose title is an `String` obtained from the `title` property of the previous `UINavigationController` that was pushed. It is being displayed on a button provided by the navigation controller which, when touched, will cause the previous `UINavigationController` to reappear. This is a “back” button.

UINavigationController

When does a pushed MVC pop off?

- Usually because the user presses the “back” button (shown on the previous slide).
- But it can happen programmatically as well with this `UINavigationController` instance method:

```
func popViewController(animated: Bool) -> UIViewController?
```

- This does the same thing as clicking the back button.
- Somewhat rare to call this method. Usually we want the user in control of navigating the stack.
- But you might do it if some action the user takes in a view makes it irrelevant to be on screen.

UINavigationController

Example

- Let's say we push an MVC which displays a database record and has a delete button with this action:

```
@IBAction func deleteCurrentRecord(sender: UIButton)
{
    // delete the record we are displaying
    // we just deleted the record we are displaying!
    // so it does not make sense to be on screen anymore
    self.navigationController?.popViewController(animated:true)
}
```

- All `UINavigationController`s know the `UINavigationController` they are in. This is `nil` if they are not in one.

View Controller

Other kinds of segues besides Push

- Modal

Puts the view controller up in a way that blocks the app until it is dismissed.

People often use Modal UIs as a shortcut, so we don't want to go to that too early. We'll talk about Modal in detail later.

- Replace

Replaces the right-hand side of a `UISplitViewController`.

- Popover

Puts the view controller on the screen in a popover.

- Unwind

Lets you dismiss view controllers that have been presented.

- Custom

You can create your own subclasses of `UIStoryboardSegue`.

View Controller

Firing off a segue from code

- Sometimes it makes sense to segue directly when a button is touched, but not always.
- For example, what if you want to conditionally segue?
- You can programmatically invoke segues using this method in `UIViewController`:

```
func performSegue(withIdentifier identifier: String,  
                  sender: Any?)
```

- The `identifier` is set in the attributes inspector in Xcode (we've seen how to do this during the Demo).
- The `sender` is the initiator of the segue (a `UIButton` or yourself (a `UIViewController`) usually).

View Controller

Firing off a segue from code

- Here is an example:

```
@IBAction func bookHotelRoom()
{
    if self.desiredRoom == .doubleRoom
    {
        self.performSegue(withIdentifier: "AskAboutDouble",
                           sender: self)
    }
    else
    {
        self.performSegue(withIdentifier: "AskAboutSingle",
                           sender: self)
    }
}
```

Segues

When a segue happens, what goes on in my code?

- The segue offers the source View Controller the opportunity to “prepare” the new View Controller to come on screen.
- This method is sent to the View Controller that contains the button that initiated the segue:

```
override func prepare(for segue: UIStoryboardSegue,  
                      sender: Any?)  
{  
    if segue.identifier == "DoAParticularThing"  
    {  
        let newController = segue.destination  
        /* Send messages to newController to prepare it to  
         * appear on screen.  
         * The segue will do the work of putting the  
         * new controller on screen. */  
        newController.view.backgroundColor = UIColor.black  
    }  
}
```


Segues

When a segue happens, what goes on in my code?

- You should pass data the new View Controller needs in `prepare(for:sender:)` and “let it run”.
- Think of the new View Controller as part of the View of the Controller that initiates the segue.
- It must play by the same rules as a View in a MVC.
- For example, it should not talk back to you except through delegation.
- So, for complicated MVC relationships, you might well set the new View Controller’s delegate to `self` here.

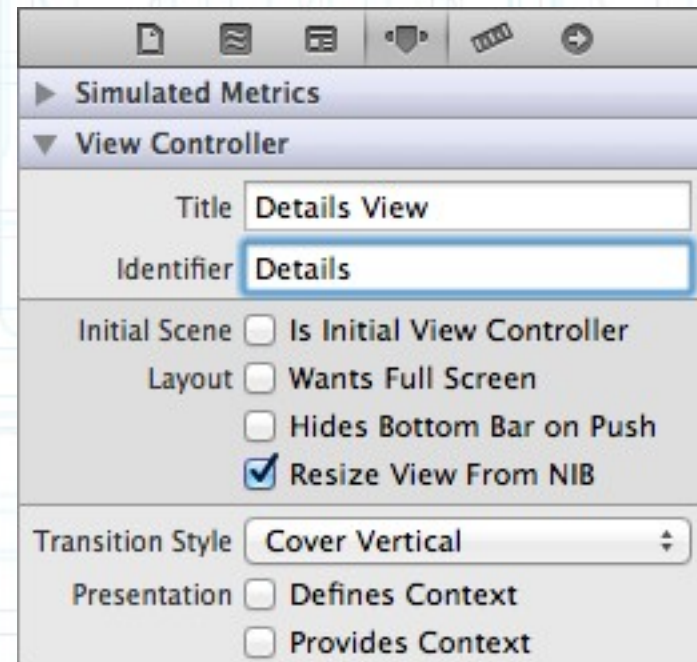
View Controller

Instantiating a `UIViewController` by name from a storyboard

- Sometimes you might want to put a View Controller on screen yourself (without using a segue).

```
let vcID = "Details"  
let controller = self.storyboard?.instantiateViewController(  
    withIdentifier: vcID)
```

- Usually you get the storyboard above from `self.storyboard` in an existing `UIViewController`.
- The identifier `vcID` must match a string you set in Interface Builder to identify a `UIViewController` there.



View Controller

Instantiating a `UIViewController` by name from a storyboard

- Example: Creating a `UIViewController` in a target/action method. Lay out the View for a `DetailsViewController` in the storyboard and name it “Details”.

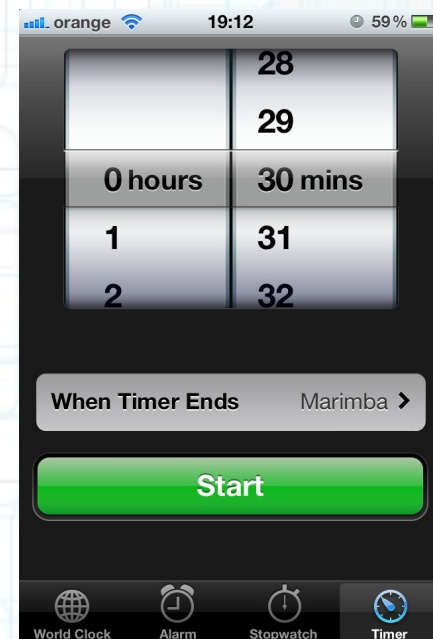
```
@IBAction func showDetails()  
{  
    let details = self.storyboard?.instantiateViewController(  
        withIdentifier: "Details") as! DetailsViewController  
    details.infoDetailsNeeds = self.info  
    self.navigationController?.pushViewController(details,  
                                                animated: true)  
}
```

- Note use of `self.navigationController` again when we push the new View Controller.

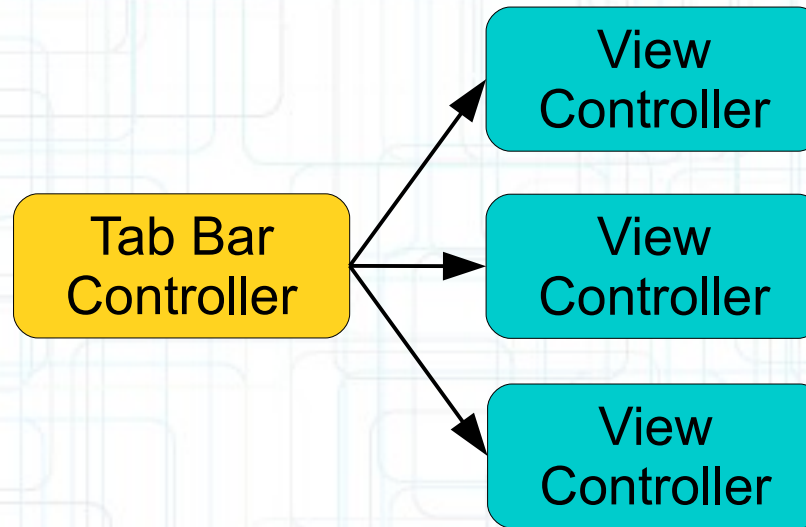
UITabBarController

Another “controller of controllers”

- Mostly set up with CTRL-drag just like navigation controller.



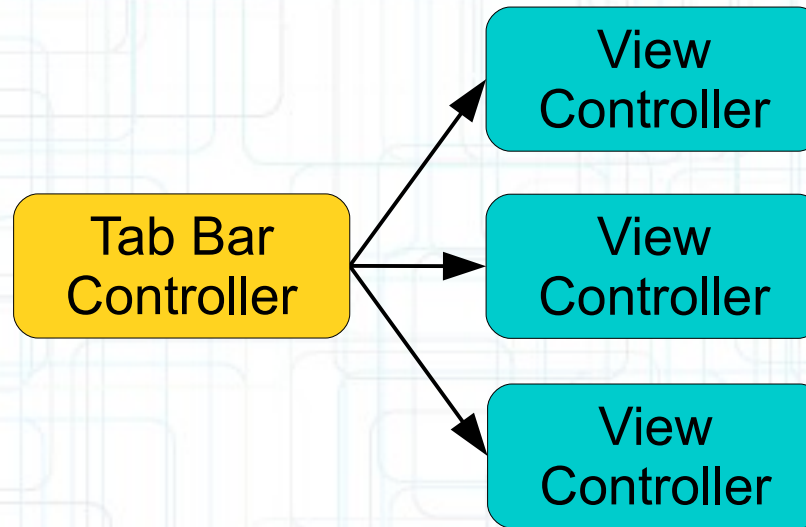
UITabBarController



- You CTRL-drag to create these connections in Interface Builder.
- Doing so is setting the following property inside your `UITabBarController`:

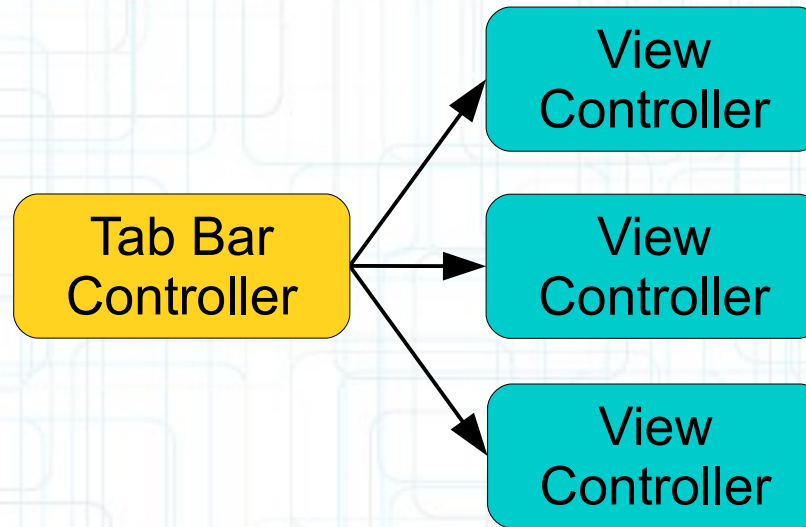
```
var viewControllers: [UIViewController]? { get set }
```

UITabBarController



By default this is the `UIViewController`'s `title` property (and no image). But usually you set both of these in your storyboard in Interface Builder.

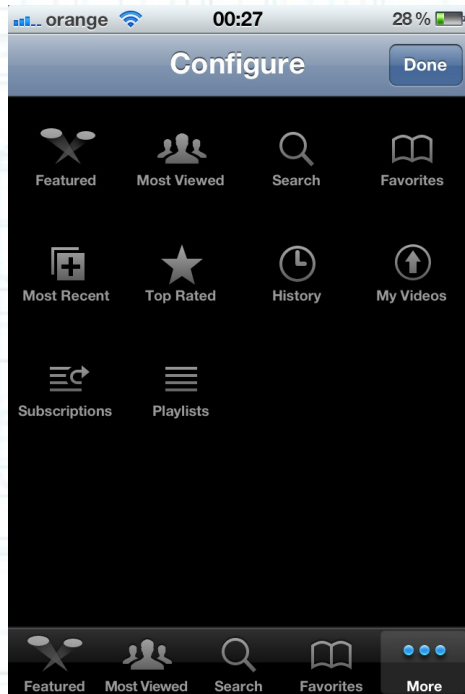
UITabBarController



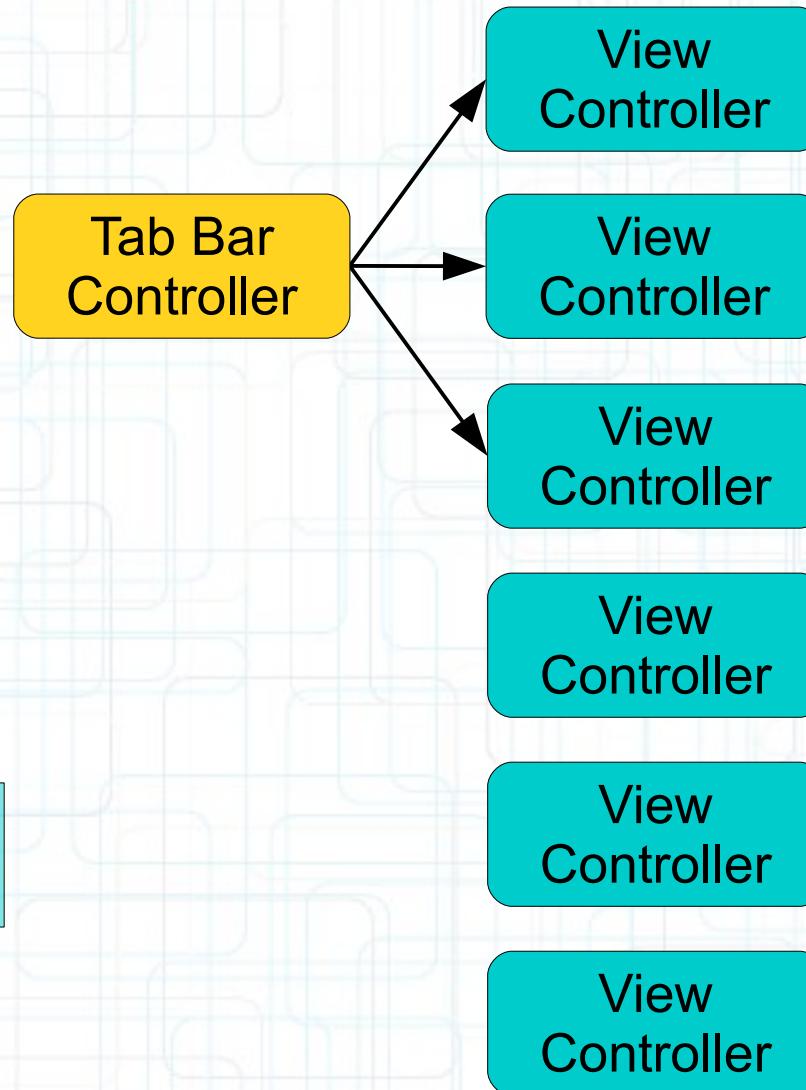
- `UIViewController`'s `tabBarItem` property can be used to set attributes for that View Controller's tab.
- Example:

```
func somethingHappenedToShowABadgeValue()  
{  
    self.tabBarItem.badgeValue = "-"  
}
```

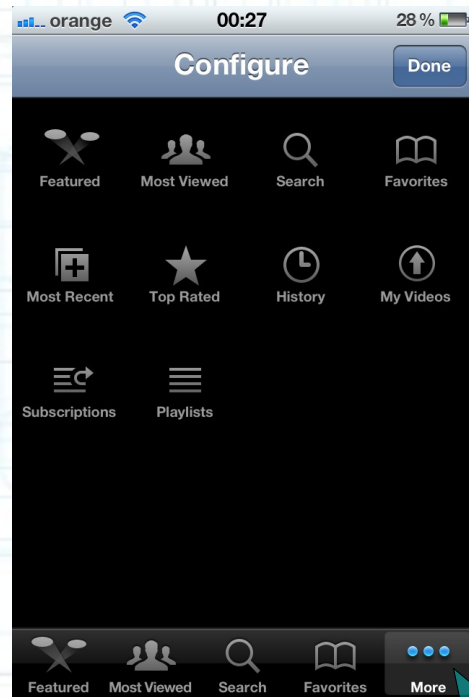

UITabBarController



What if there are more than 5 View Controllers?



UITabBarController



Tab Bar
Controller

View
Controller

View
Controller

View
Controller

View
Controller

View
Controller

View
Controller

- More button brings up a UI to let the user edit which buttons appear on bottom row.
- All happens automatically.

Combine

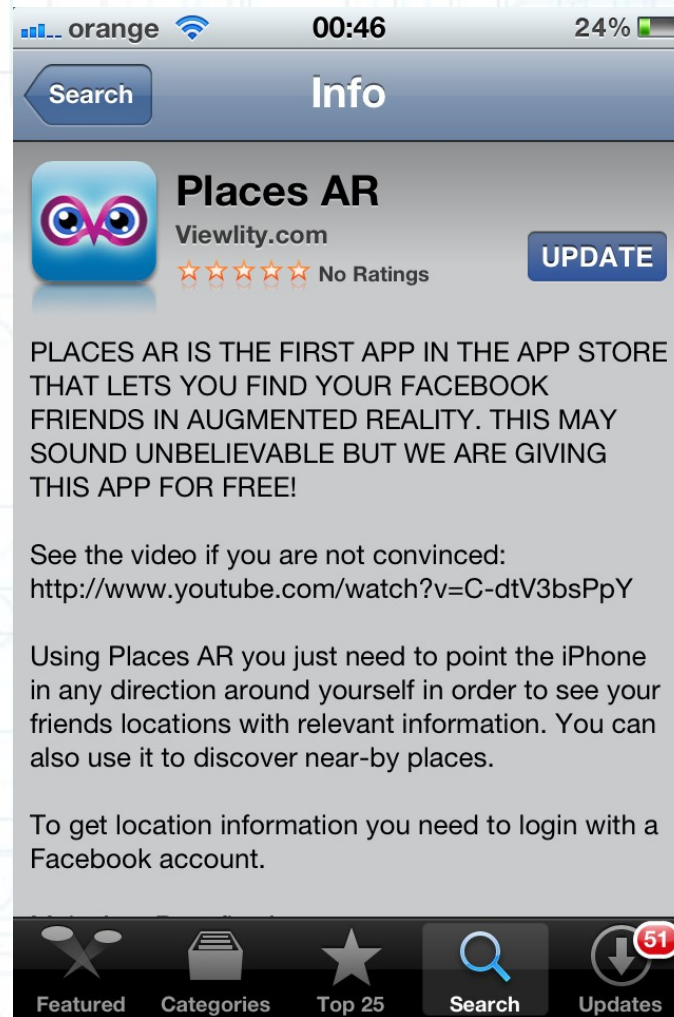
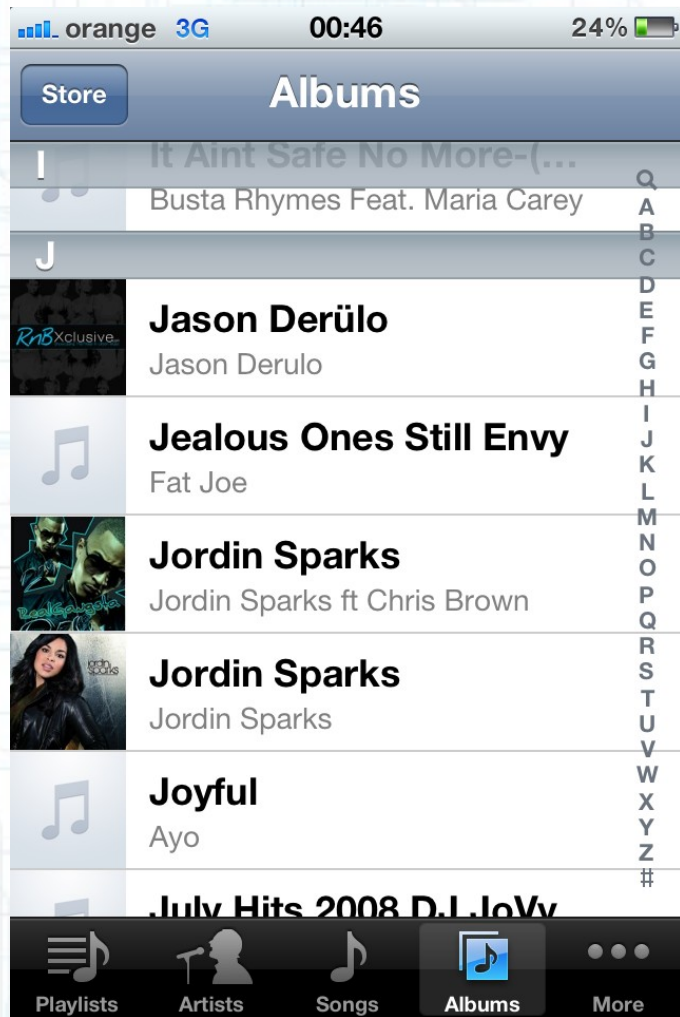
Is it possible to combine UINavigationController and UITabBarController?

- Certainly. Quite common.
- The UINavigationController always goes “inside” the UITabBarController.
- Never the other way around.

Demo (continued)

- How to combine the two “controller of controllers”.

Combine



UINavigationController

Modifying buttons and toolbar items in a navigation controller

- You can set most of this up in Interface Builder by dragging items into your scene.
- But you may want to add buttons or change buttons at run time too. Use `UINavigationController`'s `navigationItem` property:

```
var navigationItem: UINavigationController { get }
```


UINavigationController

Modifying buttons and toolbar items in a navigation controller

- Think of `navigationItem` as a holder for things `UINavigationController` will need when that `UIViewController` appears on screen.

```
var leftBarButtonItems:
    [UIBarButtonItem]? { get set }
var titleView: UIView? { get set }
var rightBarButtonItems:
    [UIBarButtonItem]? { get set }
```



Details View Controller

- When this `UIViewController` is not on the top of the Navigation Controller stack:

```
@property (nonatomic, copy)
    UIBarButtonItem *backButtonItem;
```



These bar button items are not set via the `navigationItem`. They are set via the `toolbarItems` property in `UIViewController`.

Next Time

Table Views:

- UITableView
- Creating Table View MVCs
- UITableViewDataSource
- UITableViewDelegate