iOS App Development

2.3 View Controller

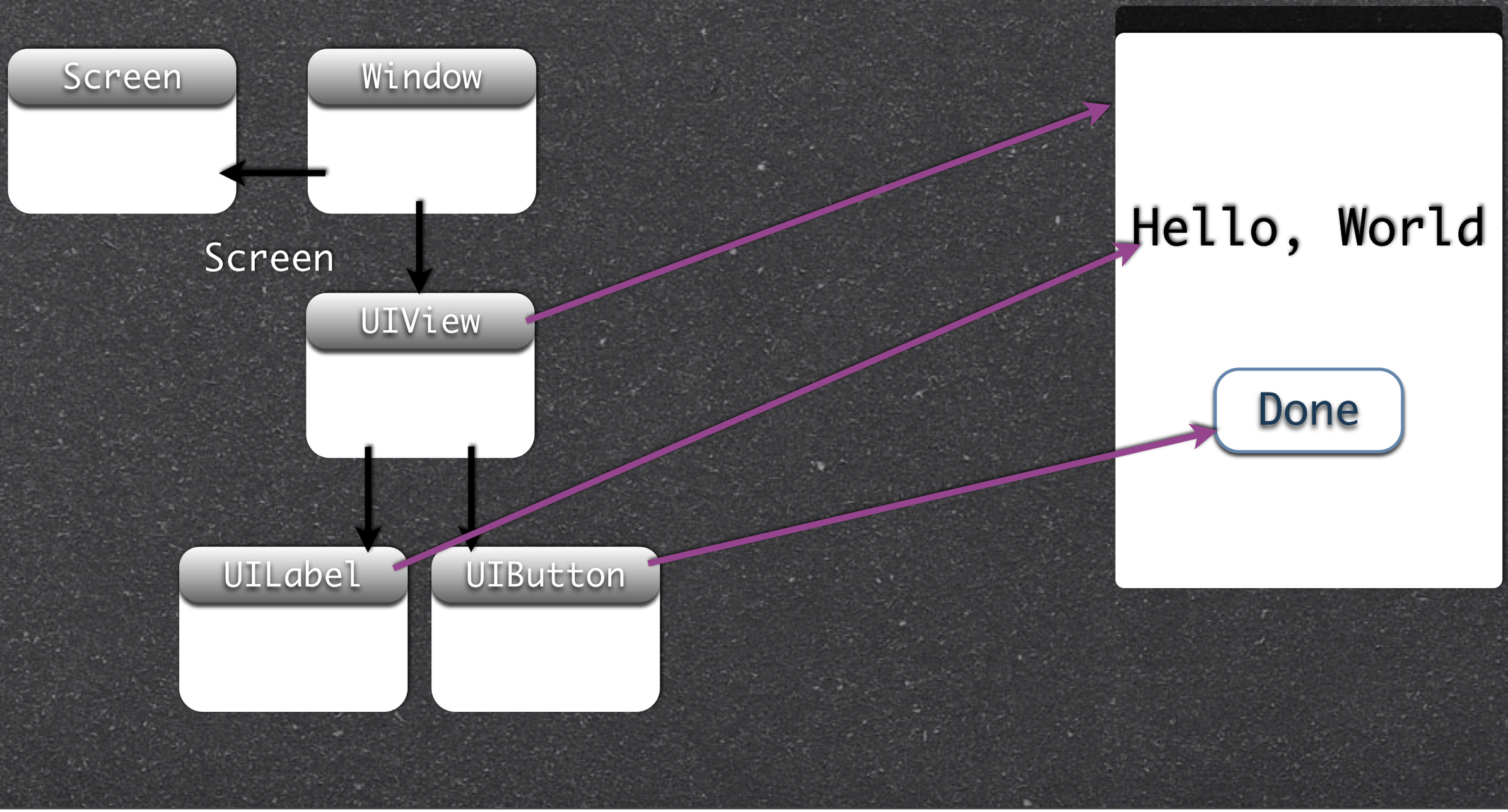
View Controller

- “View Controller Programming Guide for iOS” from Apple
- Vital link between application’s data and visual appearance

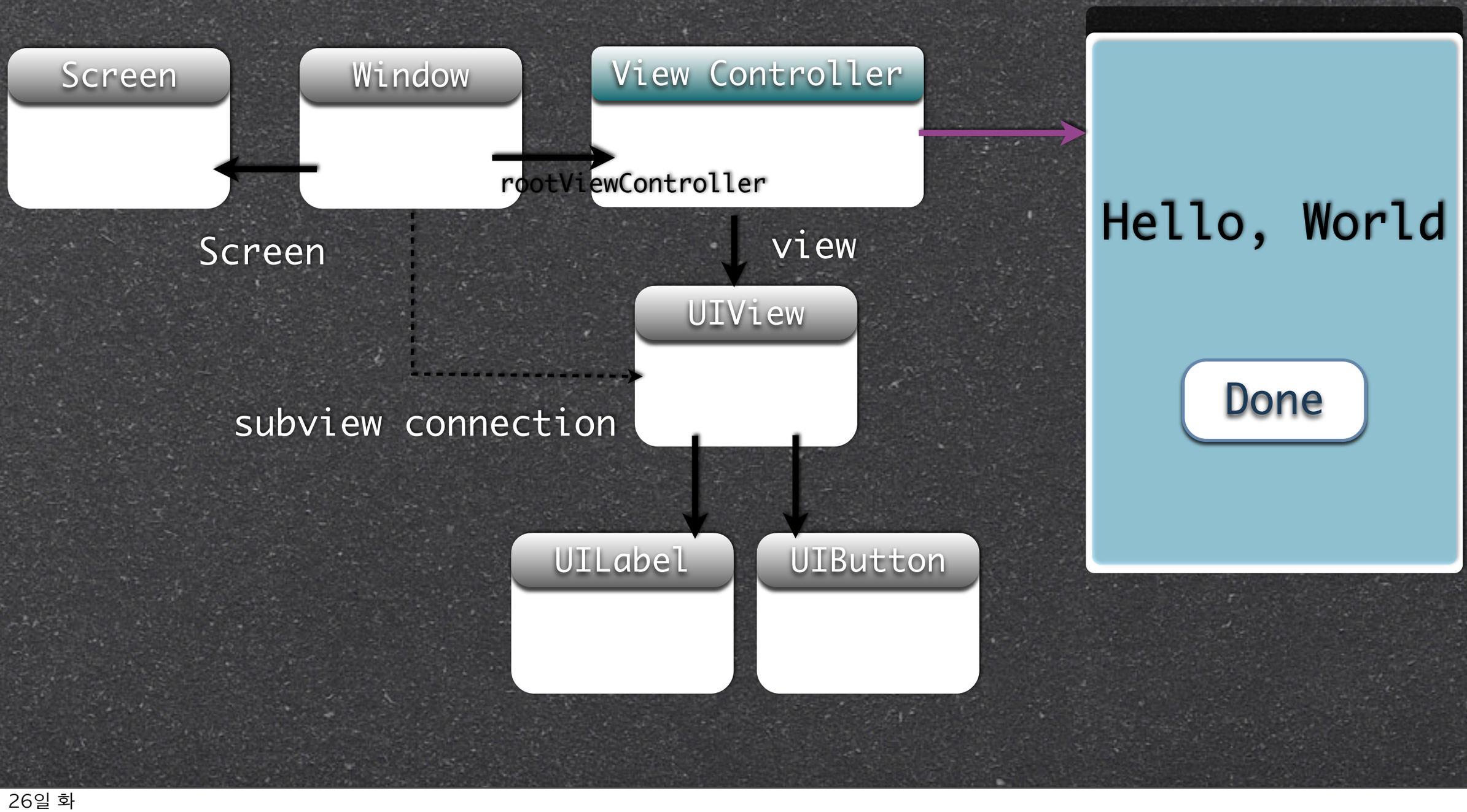
View Controller

- Manages Displayed Content (UIView objects)

View Hierarchy in iOS



View Management by View Controller



View Controller Categories by Roles

- Content View Controller
  - Custom Content View Controller:
    - i.e. Viewing Image
  - Table View Controller
- Container View Controller
  - Navigation Controller: Master->Detail
  - Tab Bar Controller: Clock Application
    - Multiple Tabs
    - A Tab-> A Child View Controller

Custom Content View Controller

Custom Content View Controller

UIViewController

UIView

UIImageView

Tabular Content View Controller

Table View Controller

UITableViewController

UITableView

UITableViewCell

Navigation Controller

UINavigationController

Content View Controller

UINavigationController

- iOS 에서 대표적인 화면전환 방법중 하나
- 내부 Stack: 여러개의 View Controller를 보관.
- Stack의 가장 위에 있는 View Controller가 유일하게 화면에 나타난다

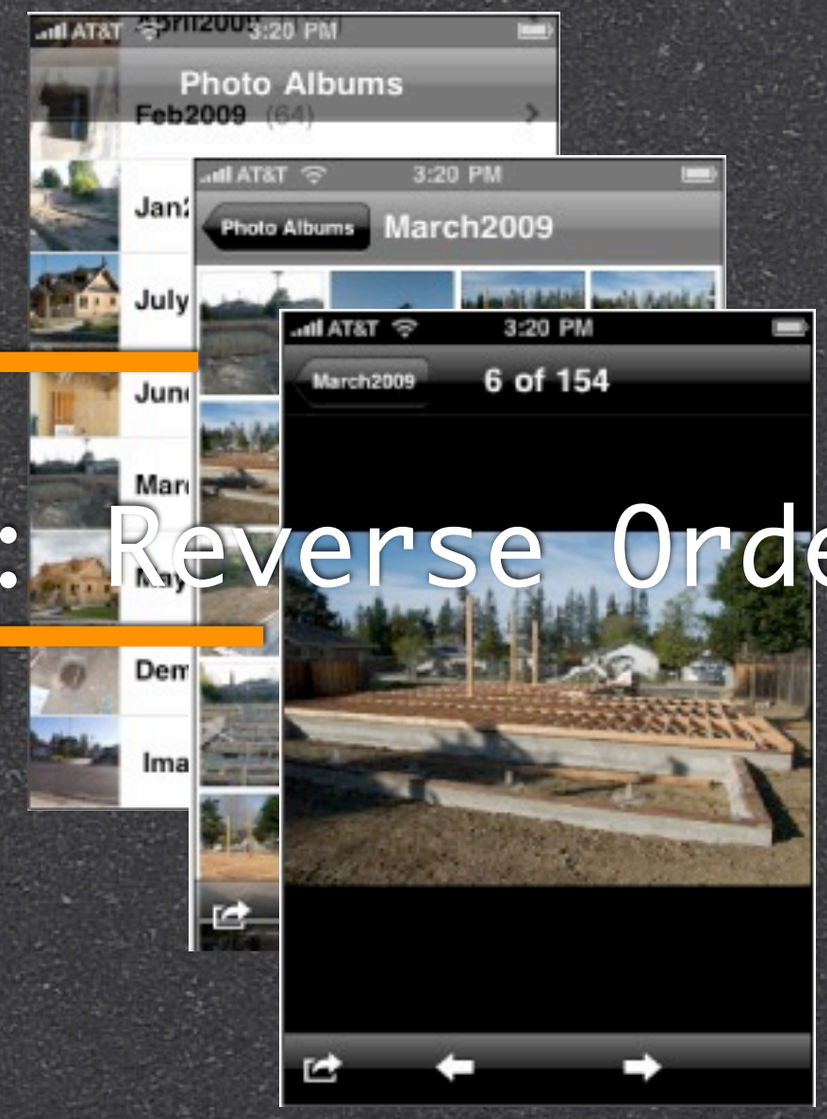
Push

Push

Push

* Pop: Reverse Order

Push



Navigation Controller

View Controllers

Navigation Controller Customization

- By Content View Controller @properties
- UIViewController
  - view
  - title
  - toolbarItems
  - back button
- Popping off

Tab Bar Controller

Content View Controller

UITabbarController

Other Container Controllers

- Split View Controller (iPad)
- Popover Controller (iPad)
- Page View Controller (View Controllers as Pages of a Book)

View Controller Transitions

- Tab Bar Controller
- Navigation Controller
  - Push to Detail
  - Pop to Master
- Page Controller
- Content View Controller -> Modal Presentation
- Popover (iPad)
- Segue: Push, Modal, Popover, and Custom

Instantiating A View Controller

- Storyboard contains View Controllers and their relationships
  - Container and Content
  - Segue
- By Storyboard
  - Initial View Controller
  - Performing Segue
  - From Code: - instantiateViewControllerWithIdentifier:

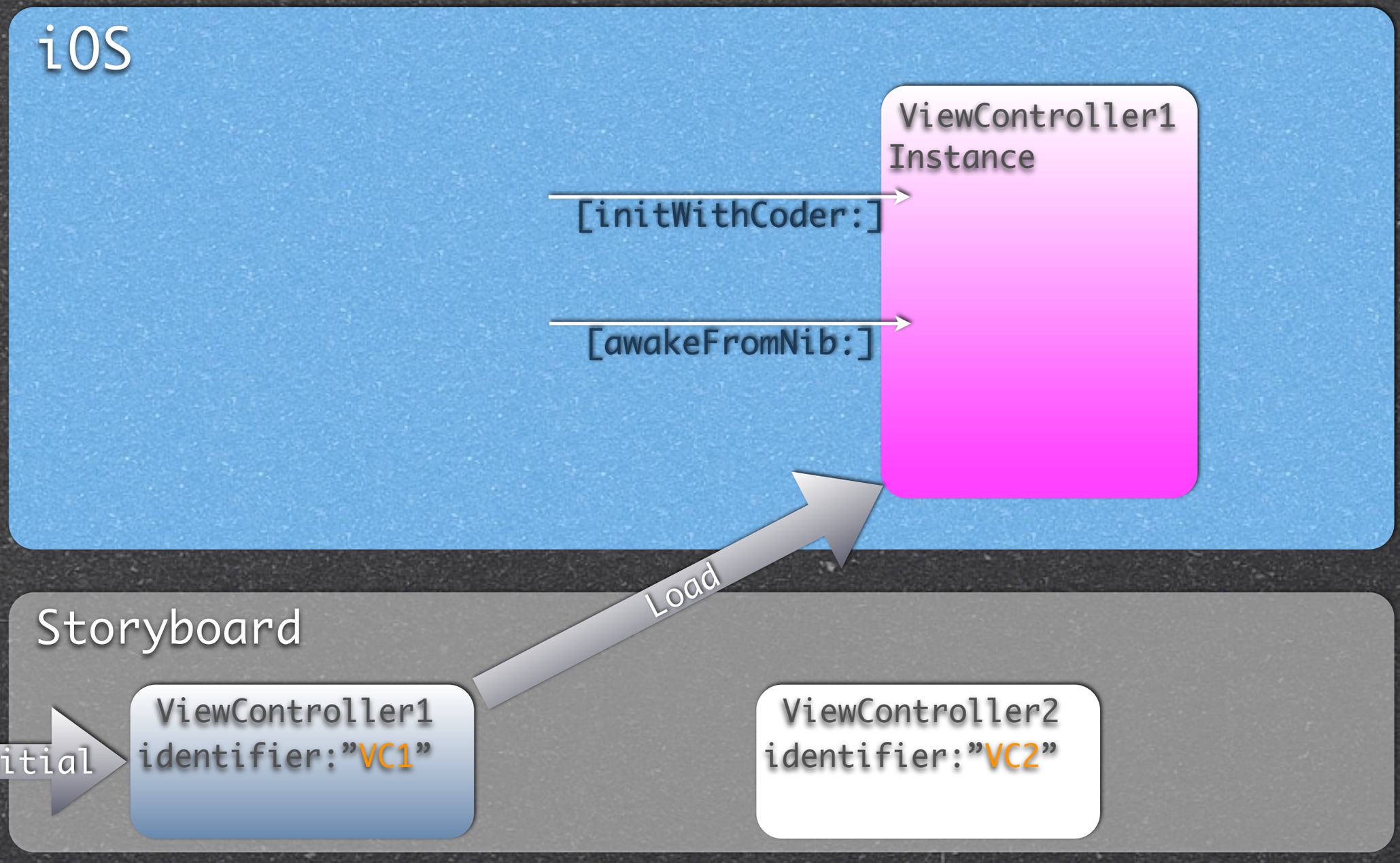
Instantiating A View Controller

- Programmatically
  - [... alloc] init] or [... initWithXXX:]
- To Display...
  - Root View Controller of a Window
  - Push to Navigation Controller
  - Present Modal
  - Embed into Container Controller

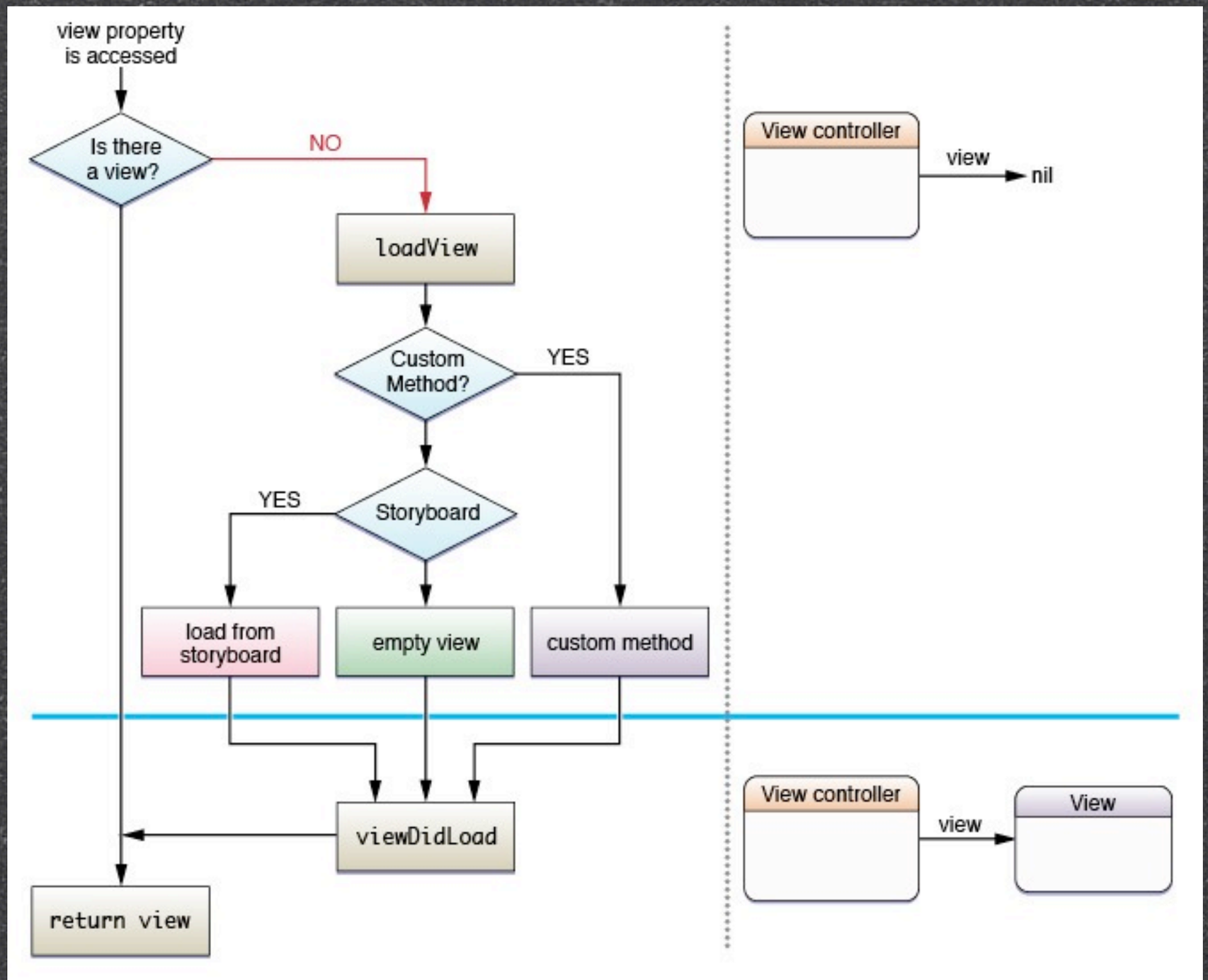
View Controller Resource Management

- Initialization
- Loading/Unloading
- Creating View
- Memory Management

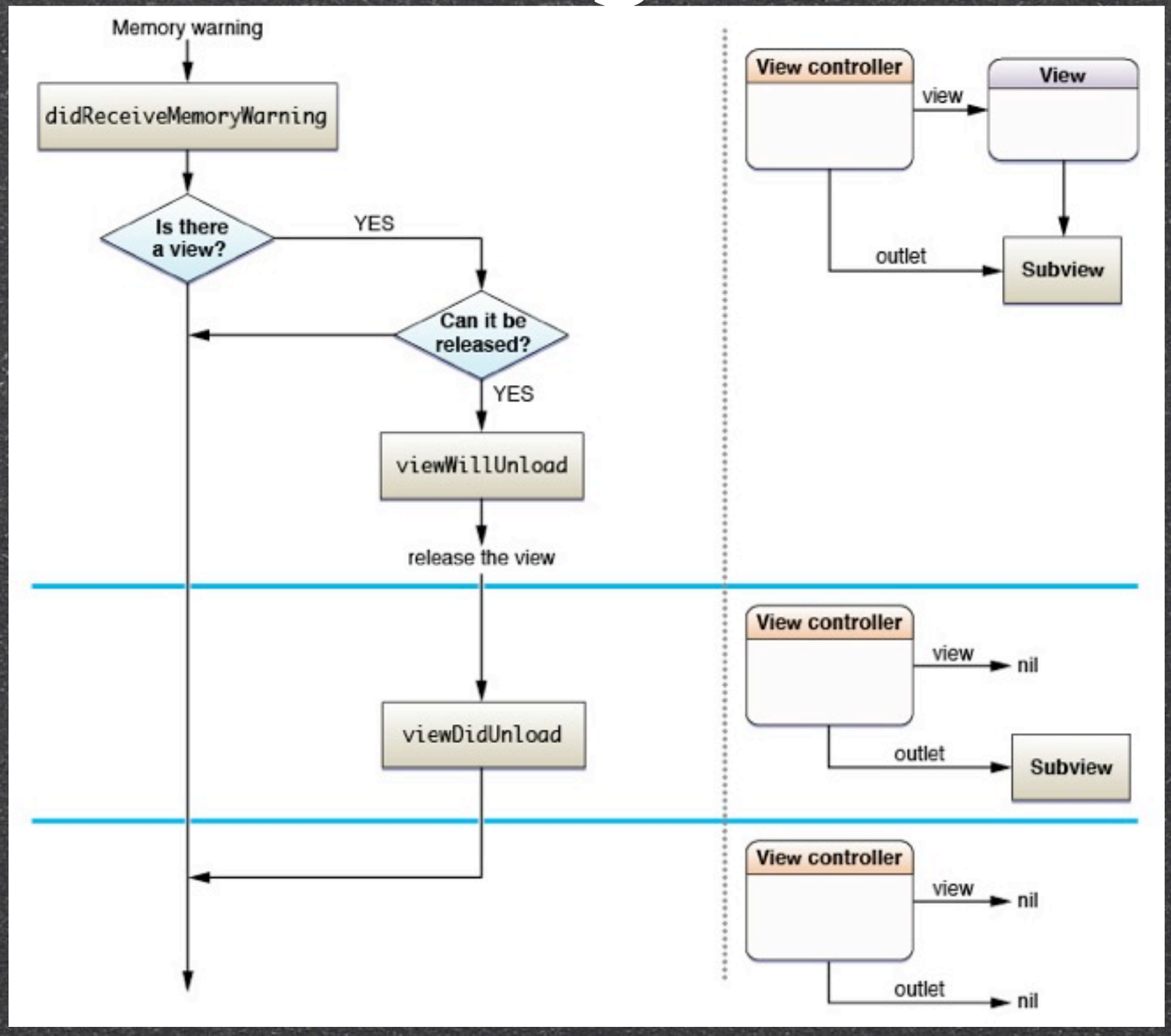
View Controller Initialization



Loading a View



Unloading a View



Creating a View Programmatically

```
- (void)loadView
{
// 1.create my own view instead of default behavior, such as storyboard
    CGRect applicationFrame = [[UIScreen mainScreen] applicationFrame];
    UIView *contentView = [[UIView alloc] initWithFrame:applicationFrame];
    contentView.backgroundColor = [UIColor blackColor];

    self.view = contentView;

// 2.additional customization
    levelView = [[LevelView alloc] initWithFrame:applicationFrame viewController:self];
    [self.view addSubview:levelView];
}
```

Memory Management in View Controller

Methods	Task
init or initWithCoder:	Not recommended: Allocation of only critical data needed anytime
loadView	Only when creating your own view
viewDidLoad	•Typical allocation •View objects are guaranteed to exist
viewDidUnload	Release strong references to view objects
didReceiveMemoryWarning	Low Memory! Release as much as possible viewDidLoad will reallocate
dealloc	Not recommended: Release any critical, yet unreleased data

View Controller View Management

- View Resizing
- Layout

View Controller Responding to Events

- Event Flow
- Device Orientation Change
- Display-Related Notifications

Presenting View Controllers

- From Other View Controllers
- Dismissing a Presented View Controller
  - Delegation
- Standard System View Controllers

Thank You